RP04

DISKLESS CONTROLLER TEST

MD-11-DERPT-A

PART II (STATIC 1B)

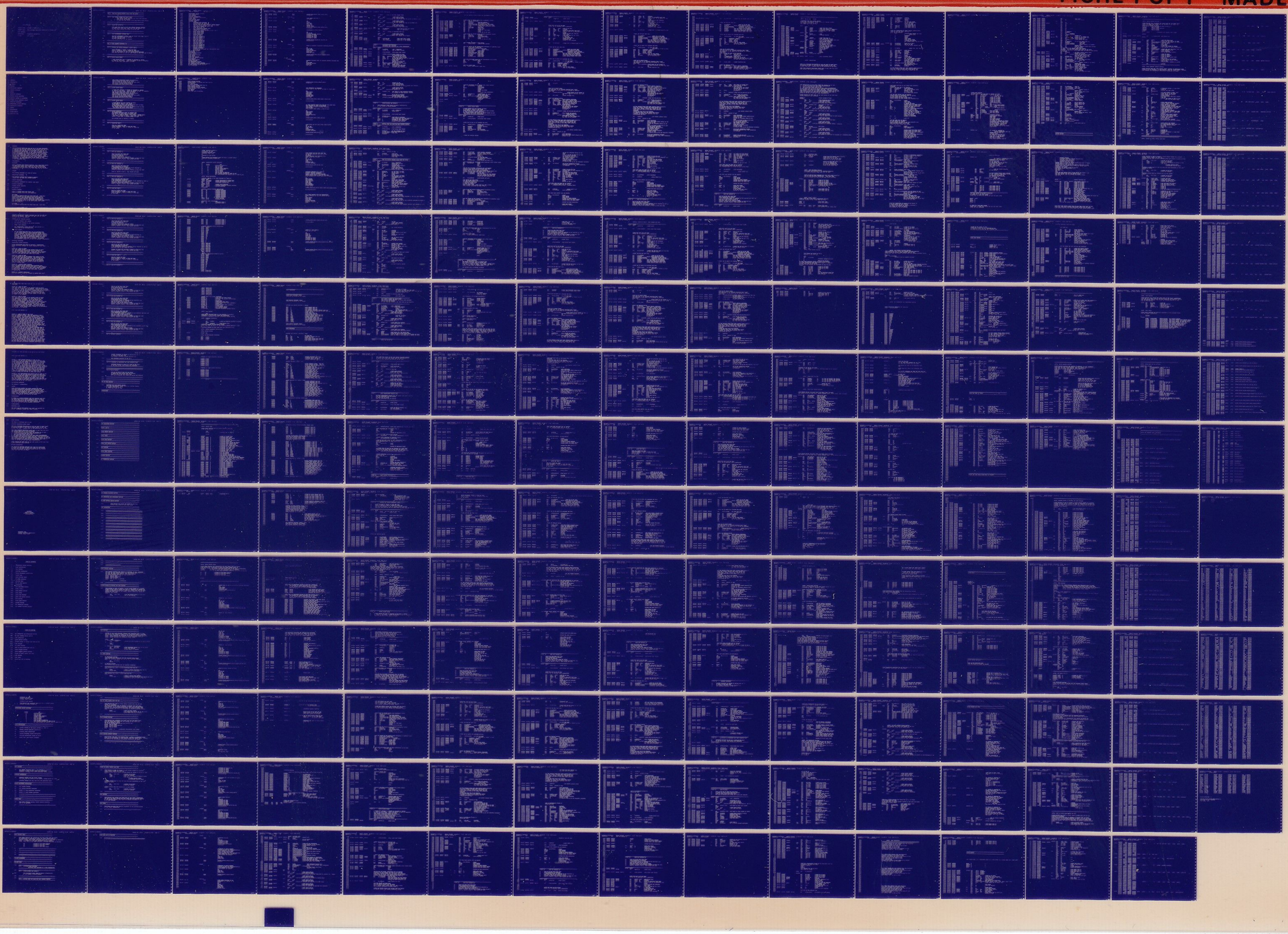
EP-DERPT-A-DL

FICHE 1 OF 1

MAY 1978

digital

MADE IN USA



IDENTIFICATION

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

PRODUCT NAME: RP04 DISKLESS CONTROLLER TEST-PART II (STATIC 1B)

DATE CREATED: MAY 21, 1975

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: SUB MALLICK

CONTENTS

1.0 ABSTRACT

THIS DIAGNOSTIC TESTS THE RH70 AND DCL OF AN RP04 SUBSYSTEM.
IT DOES NOT USE THE DISK SURFACE OR ANY SIGNALS FROM THE MDLI.
IT REQUIRES THAT THE DCL CABLE BE PLUGGED INTO THE MDLI OR
BE APPROPIATELY TERMINATED. IF THE DISK IS POWERED UP, IT
IS REQUIRED TO GET THE DISK TO THE "HEADS UNLOADED" POSITION.
AFTER A SUCCESSFUL RUN (WITH NO ERRORS) OF THIS DIAGNOSTIC
IT CAN BE ASSERTED THAT, "THAT PART OF THE DCL THAT HANDLES
DATA OR DATA ASSOCIATED LOGIC IS WORKING PROPERLY". THIS
IMPLIES THAT, THAT PART OF THE LOGIC WHICH HANDLES MECHANICAL
COMMANDS OR ITS ASSOCIATED LOGIC IS NOT TESTED IN THIS
DIAGNOSTIC. ALL DATA COMMANDS USE THE MAINTENANCE REGISTER
IN THE WRAPAROUND MODE.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 COMPUTER WITH CONSOLE TELETYPE, AND A RP04 DISK SYSTEM.
THE RP04 DISK SYSTEM WILL CONSIST OF AN RH70 CONTROLLER,
A DISK CONTROL LOGIC (DCL), THE CABLE FROM THE DCL CAN BE
CONNECTED TO THE MDLI BUT IF NOT THAT CABLE MUST BE PROPERLY
TERMINATED.

2.2 STORAGE

THIS PROGRAM REQUIRES 16K WORDS OF MEMORY.

2.3 PRELIMINARY PROGRAMS

THIS PROGRAM ASSUMES THAT MAINDEC-11-DERPST-
(LATEST REV) HAS BEEN RUN WITHOUT ERRORS

3.0 LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING .ABS TAPES

4.0 STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1

4.2 STARTING ADDRESS

START AT ADDRESS 200---FOR NORMAL RUN
START AT ADDRESS 210---FOR UNIT SELECTION

200 START
ALL SWITCHES MUST BE DOWN FOR WORST CASE RUN. WITH THIS
STARTING ADDRESS ALL THE RP04S ON THE SYSTEM WILL BE
TESTED ONE AT A TIME BEFORE "END PASS" IS PRINTED OUT.
TESTING WILL START WITH THE LOWEST UNIT NUMBER DRIVE
THAT IS POWERED UP (THAT IS THE LOWEST UNIT NUMBER RHAS REGISTER
THAT RESPONDS)THEN GO ON TO THE NEXT HIGHER UNIT NUMBER
THAT IS POWERED UP.

210 START
ALL SWITCHES MUST BE DOWN FOR WORST CASE RUN. WITH THIS

STARTING ADDRESS THE CONSOLE TELETYPE WILL ASK FOR THE UNIT NUMBER TO BE TESTED. THEN ONLY THAT UNIT WILL BE TESTED FOR EACH PASS OF THE PROGRAM.

4.3 PROGRAM AND/OR OPERATOR ACTION

1. LOAD THE PROGRAM INTO MEMORY.

2. SET STARTING ADDRESS ON THE SWITCH REGISTER

3. PRESS "LOAD ADDRESS".

4. SET "OPERATIONAL SWITCH SETTINGS" (SEE SECTION 5.1) WORST CASE IS ALL SWITCHES DOWN.

5. PRESS "START".

6. FOR THE FIRST PASS EACH TEST WILL BE EXECUTED ONCE ON THE DRIVES PRESENT OR DRIVE SELECTED BEFORE "END PASS" IS PRINTED. THE FIRST PASS WILL REQUIRE OPERATOR INTERVENTION IF THE PROGRAM IS NOT RUN UNDER AN "ACT-11" MONITOR. THE SECOND AND SUBSEQUENT PASSES WILL EXECUTE EACH TEST FOUR TIMES ON EACH DRIVES PRESENT OR DRIVE SELECTED BEFORE "END PASS" IS PRINTED. THE SECOND AND SUBSEQUENT PASSED DO NOT NEED ANY OPERATOR INTERVENTION.

5.0 OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

SWITCH DEFINITIONS ARE GIVEN IN SECTION 9 "OPERATIONAL SWITCH SETTINGS" HOWEVER THE DETAIL DESCRIPTION ARE GIVEN HERE.

SWITCH 15 - HALT ON ERROR
WHEN THIS SWITCH IS SET, IF THE PROGRAM FINDS AN ERROR THEN THE APPROPIATE INFORMATION WILL BE PRINTED OUT AND THEN THE PROGRAM WILL HALT. AFTER THIS HALT, PRESSING "CONTINUE" WILL CONTINUE WITH THE PROGRAM TILL THE NEXT ERROR IS FOUND WHEN THE SAME THING WILL HAPPEN.

SWITCH 14 - LOOP ON TEST
WHEN THIS SWITCH IS SET THE PROGRAM WILL BEGIN TO LOOP ON THE CURRENT TEST BEING EXECUTED. FOR EXAMPLE IF THIS SWITCH IS SET WHEN THE PROGRAM IS IN TEST 10 THEN THE PROGRAM WILL KEEP EXECUTING ALL OF TEST 10 REPEATEDLY. ONE WAY TO BE SURE THAT THE PROGRAM IS IN THE EXPECTED TEST IS TO SET THIS SWITCH DURING AN ERROR PRINTOUT OR DURING A PROGRAM HALT.

SWITCH 13 - INHIBIT ERROR TYPEOUTS
WHEN THIS SWITCH IS SET FURTHER ERROR PRINTOUTS WILL CEASE, HOWEVER OPERATOR INSTRUCTIONS SUCH AS "STOP DRIVE X" WILL CONTINUE. AT THE END OF PASS "TOTAL NUMBER OF ERRORS ON THIS PASS ON DRIVE X" WILL BE TRUE, THAT IS, ALTHOUGH PRINTOUTS WERE INHIBITED IF THAT PASS FOUND 6 ERRORS, IT WILL SAY SO.

SWITCH 11 - INHIBIT ITERATIONS
WHEN THIS SWITCH IS SET THE PROGRAM ON SECOND PASS WILL

NOT REPEAT EACH TEST FOUR TIMES BUT WILL DO EACH TEST
ONCE ONLY.

SWITCH 10 - BELL ON ERROR
WHEN THIS SWITCH IS SET, IF THE PROGRAM FINDS AN ERROR
THE "BELL" OR "ALARM" WILL BE SOUNDED. THIS SWITCH IS USEFUL
WHEN SWITCH 11 IS SET YET INFORMATION IS NEEDED WHEN ANY ERROR
IS DETECTED, TAKE THE EXAMPLE OF A PROGRAM LOOPING ON A TEST WITH
SWITCH 11 SET TO HELP SCOPING. THEN IF THIS SWITCH IS
SET AND THE BELL OR ALARM SOUNDS IT MEANS THAT THE ERROR
IS PRESENT BUT IF THE BELL OR ALARM STOPS IT MEANS THAT
THE ERROR IS NOT PRESENT.

SWITCH 9 - LOOP ON ERROR
WHEN THIS SWITCH IS SET, IF THE PROGRAM FINDS AN ERROR
THEN GENERALLY THE PROGRAM WILL LOOP BACK TO THE LAST
EXECUTED "SCOPE" STATEMENT. IF ON THE SECOND TIME
THROUGH AN ERROR IS FOUND IT WILL AGAIN LOOP BACK TO
THAT "SCOPE" STATEMENT. THIS LOOPING WILL CONTINUE AS LONG
AS THE ERROR IS PRESENT AND THIS SWITCH IS SET. HOWEVER
IF THE ERROR IS NOT PRESENT AT ANY TIME THEN IT WILL
CONTINUE NORMALLY WITH THE PROGRAM. EACH TIME THE ERROR
IS ENCOUNTERED PRINTOUT WILL TAKE PLACE UNLESS SWITCH 11
IS ALSO SET. DURING BEGUG, USING A SCOPE, IT IS RECOMMENDED
THAT SWITCH 11 IS ALSO SET.

NOTE: ALSO SEE SECTION 8.3

SWITCH 0 - LOOP ON TEST IN SWR <7:0>
THIS IS A SPECIAL SWITCH. WHEN SET SWITCHES 0 THRU 7
HAVE ONE MEANING AND WHEN RESET SWITCHES 0 THRU 7 HAVE
ANOTHER MEANING. THIS MEANS THAT ANY SETTING OF SWITCH
0 THRU 7 MUST BE DONE WITH SWITCH 8 IN THE APPROPIATE
POSITION. WHEN THIS SWITCH IS SET THEN SWITCHES 0 THRU
; GIVE THE TEST NUMBER TO BE LOOPED ON. FOR EXAMPLE
WITH SWITCH 8 SET AND SWITCH 3 SET THE PROGRAM WILL LOOP
ON TEST 10. HOWEVER THIS SETTING MUST BE DONE AT THE
BEGINNING OF THE PROGRAM THEN ALL THE TESTS FROM 1 TO 10
WILL BE EXECUTED AND THEN TEST 10 WILL BE REPEATED OVER
AND OVER AGAIN. WHEN THIS SWITCH IS NOT SET THEN SWITCHES
0 THRU 7 HAVE THE MEAING ITS NAME INDICATES.
FOR EXAMPLE SWITCH 7 IS "STOP FURTHER COMPARES; THAT IS
IF SWITCH 8 IS NOT SET AND SWITCH 7 IS SET THEN WHEN A
DATA ERROR IS DETECTED NO FURTHER COMPARES WILL BE DONE.
FOR EXAMPLE IN A 256 WORD BUFFER IF ALL THE WORDS ARE IN
ERROR THEN AFTER SEEING THE PRINTOUT FOR THE FIRST FEW
WORDS SETTING SWITCH 7 ONLY WILL STOP FURTHER PRINTOUTS
OF THIS ERROR AND GO ON WITH THE TEST RATHER THAN PRINT
ALL THE 256 WORDS. HOWEVER IF THIS WAS DONE WITH SWITCH 11
THEN THE NEXT ERROR THAT THE PROGRAM DETECTS IN A SUB-
SEQUENT TEST WILL ALSO BE LOST. BUT WITH SWITCH 7, ONLY
THIS GROUP OF DATA ERRORS ARE NOT PRINTED OUT. ANOTHER
EXAMPLE OF SWITCH 8 BEING LOW IS WITH SWITCH 6, WHICH
IS "ECC TEST-COMPARE END RESULT ONLY". THAT IS IF SWITCH
8 IS NOT SET AND SWITCH 6 IS SET THEN ON ECC TESTS (TEST
120 THRU TEST 134) INSTEAD OF COMPARING CONTENTS OF THE
POSITION REGISTER AND PATTERN REGISTER AFTER EVERY CLOCK,

COMPARES WILL ONLY BE DONE AT THE END OF ALL THE CLOCKS.

NOTE: ALSO SEE SECTION 8.3

SWITCH 7 - STOP FURTHER COMPARES IF SW08 IS LOW.
IF SWITCH 8 IS SET AND THIS SWITCH IS ALSO SET THEN THIS SWITCH GIVES THE TEST NUMBER TO BE LOOPED ON AS INDICATED IN THE DESCRIPTION OF SWITCH 8. IF SWITCH 8 IS NOT SET AND THIS SWITCH IS SET THEN THE PROGRAM WILL DO AS THE NAME INDICATES. FOR EXAMPLE IN A 256 WORD BUFFER IF ALL THE WORDS ARE IN ERROR THEN AFTER SEEING THE ERROR PRINTOUTS FOR THE FIRST FEW WORDS THEN SETTING SWITCH 7 WITH SWITCH 8 NOT SET WILL STOP THE PRINTOUT OF ALL 256 WORDS BUT WILL NOT STOP THE PRINTOUT OF ANOTHER ERROR IN ANY SUBSEQUENT TEST. IT IS EXPECTED THAT SWITCH 7 AFTER BEING SET FOR A WHILE TO STOP PRINTING ALL THE 256 WORDS WILL BE RESET AGAIN TO ENABLE THE PRINTING OF OTHER DATA ERRORS.

SWITCH 6 - ECC TEST-COMPARE END RESULTS ONLY IF SW06 IS LOW
IF SWITCH 8 IS SET AND THIS SWITCH IS ALSO SET THEN THIS SWITCH GIVES THE TEST NUMBER TO BE LOOPED ON AS INDICATED IN THE DESCRIPTION OF SWITCH 8. IF SWITCH 8 IS NOT SET AND THIS SWITCH IS SET THEN ON ECC TESTS (TEST 120 THRU TEST 134) INSTEAD OF COMPARING CONTENTS OF THE POSITION AND PATTERN REGISTERS AFTER EVERY CLOCK, COMPARES WILL BE DONE ONLY AT THE END OF ALL THE CLOCKS.

5.2 SUB-ROUTINE ABSTRACTS

SEE SECTION 9 "SUBROUTINES"

6.0 ERRORS

ERROR PRINTOUTS CONTAIN THE ERROR ADDRESS AND OTHER PERTINENT INFORMATION CONCERNING THE PARTICULAR FAILURE. THIS INFORMATION MAY BE THE CONTENTS OF RELEVANT RP04 REGISTERS OR GOOD/RECEIVED DATA. IF THE ERROR OCCURRED IN A SUBROUTINE, THE ADDRESS OF THE SUBROUTINE CALL IS ALSO GIVEN. REFER TO THE PROGRAM LISTING AT THE STATED ADDRESS TO DETERMINE THE CAUSE OF THE ERROR.

7.0 RESTRICTIONS

IF THERE IS A DRIVE CONNECTED THEN THE OPERATOR MUST HAVE THE DRIVE PORT SWITCH LOCKED EITHER ON PORT A OR PORT B BUT NEVER LEAVE IT IN THE PROGRAMMABLE STATE. IF THERE IS NO DRIVE CONNECTED THEN THE CABLE NORMALLY GOING FROM THE DCL TO THE MDLI MUST BE PROPERLY TERMINATED.

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

THE FIRST PASS OF THE PROGRAM WILL TAKE 1.75 MINUTES PER DRIVE. SUBSEQUENT PASSES WILL TAKE 7 MINUTE.

8.2 STACK POINTER

THE STACK IS INITIALLY SET TO 1000

8.3 OPERATOR SELECTABLE SCOPE LOOPS

HERE IS A DETAILED EXPLAINATION OF HOW THE LOOP ON ERROR WORKS.
ON HITTING AN ERROR IF THE LOOP ON ERROR SWITCH IS SET, THE
PROGRAM GOES BACK - USUALLY BACK TO THE BEGINNING OF THE TEST.

WHEN THIS OPERATOR SELECTABLE SCOPE LOOP IS USED THEN THE POINT
THE PROGRAM GOES BACK TO CAN BE CHANGED.
THE RESTRICTIONS TO THE POINT WHERE THE PROGRAM CAN GO ARE: -
1. IT MUST BE WITHIN THE TEST UNDER CONSIDERATION
2. LOOP ON ERROR SWITCH MUST BE SET
3. THE ERROR MUST OCCUR WITHIN THE TEST UNDER CONSIDERATION
IF THE ERROR DOES NOT OCCUR WITHIN THE TEST UNDER CONSIDERATION
THE PROGRAM WILL REVERT TO NORMAL OPERATION. HOWEVER, IF LOOP ON
TEST SWITCH IS SET AND THIS OPERATOR SELECTABLE SCOPE LOOP IS USED
THEN THE PROGRAM WILL LOOP BACK TO THE SELECTED POINT WHEN IT
COMES TO THE END OF THE TEST UNDER CONSIDERATION.

AFTER LOOPING FOR SOME TIME IF THE LOOP SWITCH IS PUT DOWN THEN
NORMAL OPERATION WILL CONTINUE.

9.0 PROGRAM DESCRIPTION

THE FOLLOWING SECTIONS DESCRIBE EACH TEST AND SUBROUTINES
IN DETAIL AND CAN ALSO BE USED AS AN INDEX TO THE LISTING.
THE LEFT MOST COLUMN IS THE LINE NUMBER WITHIN THE LISTING
WHERE THAT ITEM WILL BE FOUND.

DOCUMENT
****
MAINDEC-11-DERPTA-A
****

TABLE OF CONTENTS
****

TABLE OF CONTENTS
****

PROGRAM BY SUB MALLICK

THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
PACKAGE (MAINDEC-11-DZQAC-A3).

```
                ;*****************************************************************
13              ;OPERATIONAL SWITCH SETTINGS
                ;*****************************************************************

14
                ;SWITCH                 USE
                ;------          --------------------
                ;  15            HALT ON ERROR
                ;  14            LOOP ON TEST
                ;  13            INHIBIT ERROR TYPEOUTS
                ;  11            INHIBIT ITERATIONS
                ;  10            BELL ON ERROR
                ;   9            LOOP ON ERROR
                ;   8            LOOP ON TEST IN SWR<7:0>
                ;   7            STOP FURTHER COMPARES IF SW08 IS LOW
                ;   6            ECC TEST-COMPARE END RESULTS ONLY IF SW08 IS LOW

                ;*****************************************************************
27              ;BASIC DEFINITIONS
                ;*****************************************************************

29              ;       INITIAL ADDRESS OF THE STACK POINTER *** 1000 ***

40              ;       GENERAL PURPOSE REGISTER DEFINITIONS

52              ;       PRIORITY LEVEL DEFINITIONS

62              ;       "SWITCH REGISTER" SWITCH DEFINITIONS

90              ;       DATA BIT DEFINITIONS (BIT00 TO BIT15)

118             ;       BASIC "CPU" TRAP VECTOR ADDRESSES
```

```
                ;*****************************************************************
        133     ; TRAP CATCHER
                ;*****************************************************************

                136     ; 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

                ;*****************************************************************
        140     ; STARTING ADDRESS(ES)
                ;*****************************************************************

                146     ; STARTING ADDRESS 200 FOR NORMAL STARTS
                        ; THIS WILL TEST ALL RP04'S ON THE SYSTEM A SINGLE DRIVE AT A TIME

                        ; STARTING ADDRESS 210 WILL TEST ONLY ONE SPECIFIED DRIVE

                ;*****************************************************************
        151     ; MEMORY MANAGEMENT DEFINITIONS
                ;*****************************************************************

                153     ; KT11 VECTOR ADDRESS

                157     ; KT11 STATUS REGISTER ADDRESSES

                164     ; KERNAL "I" PAGE DESCRIPTOR REGISTERS

                175     ; KERNAL "I" PAGE ADDRESS REGISTERS

                186     ;*****************************************************************

                188     ;*****************************************************************

                ;*****************************************************************
        190     ; COMMON TAGS
                ;*****************************************************************

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

                244     ;*****************************************************************
```

```
                ;*******************************************************************
246             ;ERROR POINTER TABLE
                ;*******************************************************************

248             ;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:

254                     EM              ;;POINTS TO THE ERROR MESSAGE
                        DH              ;;POINTS TO THE DATA HEADER
                        DT              ;;POINTS TO THE DATA
                        DF              ;;POINTS TO THE DATA FORMAT

263             ;*******************************************************************

711             ;*******************************************************************

716             ;*******************************************************************

750             ;*******************************************************************
                ;*******************************************************************

                ;*******************************************************************
908             ;REGISTER ADDRESSES
                ;*******************************************************************

                ;*******************************************************************
1074            ;REGISTER TEST
                ;*******************************************************************

1158            ;*******************************************************************
                ;TEST 1  REFERENCE EACH REGISTER
                ;                       REFERENCE EACH REGISTER BY A MOVE INSTRUCTION
                ;*******************************************************************

1202            ;*******************************************************************
                ;TEST 2  RHCS2-CONTROL AND STATUS 2

1205                    ;THIS PARTIALLY TESTS RHCS2 TO ENABLE DETERMINATION
                        ;OF THE NUMBER OF DRIVES PRESENT

1208            ;*******************************************************************

1223            ;*******************************************************************
                ;TEST 3  PARTIAL TEST FOR RHAS FOR UNIT NUMBERS PRESENT
                ;*******************************************************************
```

M1

```
1243    ********************************************************************
        TEST 4  TEST FOR DRIVES PRESENT USING RHAS AND RHCS2
        ********************************************************************

1353    ********************************************************************
        TEST 5  TEST SERIAL NUMBER AND DRIVE TYPEI
                        READ SERIAL NUMBER REGISTER AND DRIVE TYPE REGISTER
                        TYPE IT OUT AND PROCEED
                        TO LOOP HERE SET SWITCH 8 AND THIS TEST NO AND RESTART

1359    ********************************************************************

1395    ********************************************************************
        TEST 6  CHECK MOL TO BE LOW

1398            MAKE SURE THAT DRIVE IS OFF LINE BEFORE STARTING PROGRAM
                IF DRIVE IS ON LINE THEN AFTER TYPE OUT THE PROGRAM WILL
                HANG FOR EVER WAITING FOR DRIVE TO GO OFF LINE

1402    ********************************************************************

1434    ********************************************************************
        TEST 7  PACK ACKNOWLEDGE COMMAND TEST

1437            THE PACK ACKNOWLEDGE COMMAND WILL BE LOADED INTO RHCS1 WITH GO
                THEN ALL REGISTERS WILL BE CHECKED
                RH CLEAR WILL BE GIVEN
                THEN ALL REGISTERS WILL BE CHECKED

1442    ********************************************************************

1501    ********************************************************************
        TEST 10 MAKE CURRENT CYLINDER = 0
        ********************************************************************

1513    ********************************************************************
        TEST 11 CONTROL AND STATUS REGISTER 1 BITS 8 AND 9

1516            WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
                TRACK 0, SECTOR 0, KEYS 0, NUMBER OF WORDS 256
                DATA IS THE CONTENTS OF THE TTY READER STATUS REGISTER
                THIS WILL USE BITS A16 AND A17 WHEN THERE IS MORE THAN 28K OF MEMORY

1521    ********************************************************************

1608    ********************************************************************
        TEST 12 DRIVE TIMING ERROR

1611            A READ HEADER AND DATA IS STARTED ON CYLINDER 0, SECTOR
                0, TRACK 0, 260 WORDS.  AFTER THE HEADER IS READ IN CORRECTLY
                THEN NO SYNC BYTE (DATA SYNC) IS GIVEN.
```

A2

```
                1614            THEN NORMAL DIAGNOSTIC CLOCKS AND DIAGNOSTIC
                                SECTOR CLOCKS ARE GIVEN FOR 24 BYTES,
                                THEN 536 BYTES OF SECTOR CLOCKS ONLY ARE GIVEN,
                                THIS IS TO TO BRING SECTOR PULSE UP
                                THIS SHOULD SET DRIVE TIMING ERROR
                        ;*******************************************************

                1762    ;*******************************************************
                        TEST 13 DRIVE TIMING ERROR

                1765            A WRITE DATA COMMAND IS STARTED ON CYLINDER 0, SECTOR
                                0, TRACK 0, 256 WORDS.  AFTER THE HEADER IS READ IN CORRECTLY
                                THEN NO SYNC BYTE (DATA SYNC) IS GIVEN,
                                THEN NORMAL DIAGNOSTIC CLOCKS AND DIAGNOSTIC
                                SECTOR CLOCKS ARE GIVEN FOR 24 BYTES,
                                THEN 536 BYTES OF SECTOR CLOCKS ONLY ARE GIVEN,
                                THIS IS TO TO BRING SECTOR PULSE UP
                                THIS SHOULD SET DRIVE TIMING ERROR
                        ;*******************************************************

                1908    ;*******************************************************
                        TEST 14 DRIVE TIMING ERROR

                1911            A WRITE HEADER AND DATA COMMAND IS GIVEN
                                TO CYLINDER 0, TRACK 0, 256. WORDS
                                AFTER SECTOR IS FOUND (THE SECTOR FOUND FLOP IS HIGH)
                                NO MORE DIAGNOSTIC CLOCKS ARE GIVEN,
                                ONLY SECTOR CLOCKS ARE GIVEN TILL SECTOR PULSE IS HIGH
                                 THIS SHOULD SET DRIVE TIMING ERROR
                        ;*******************************************************

                2032    ;*******************************************************
                        TEST 15 SECTOR SELECTION
                                THE SECTOR SELECTION LOGIC IS CHECKED HERE
                                EACH SECTOR ON TRACK ZERO IS WRITTEN INTO
                                DATA IS - 19 WORDS OF ZEROS - SYNC WORDS, 4 HEADER WORDS
                                1 CRC WORD, 5 WORDS OF ZEROS, 1 SYNC WORD, 100 ZEROS
                                (DATA), 1 SYNC WORD, 70 SECTOR NUMBER TO VARY

                                THE WRITTEN DATA IS CHECKED IN MEMORY

                2042    ;*******************************************************

                2205    ;*******************************************************
                        TEST 16 WRITE ECC TEST 1

                2208            THIS IS A WRITE ECC TEST
                                WRITE CYLINDER0, FORMAT 16 BITS PER WORD
                                TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
                                OF ALL ZEROS,
```

```
2213    ;*****************************************************************
2353    ;*****************************************************************
        TEST 17 READ ECC ENABLED 1A

2356            THIS IS AN ECC READ DATA TEST
                ERROR CORRECTION IS ENABLED
                NO ERROR IS INSERTED
                GOOD DATA USED IS 256 WORDS OF 0
                COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
                TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA

2363    ;*****************************************************************
2524    ;*****************************************************************
        TEST 20 READ ECC ENABLED 1B

2527            THIS IS AN ECC READ DATA TEST
                ERROR CORRECTION IS ENABLED
                A CORRECTABLE ERROR IS INSERTED IN BIT POSITION 32
                GOOD DATA USED IS 256 WORDS OF 0
                COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD

2532            TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA

2534    ;*****************************************************************
2701    ;*****************************************************************
        TEST 21 READ ECC ENABLED 1C

2704            THIS IS AN ECC READ DATA TEST
                ERROR CORRECTION IS ENABLED
                A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 21 THRU 32
                GOOD DATA USED IS 256 WORDS OF 0
                COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
                TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA

2711    ;*****************************************************************
2891    ;*****************************************************************
        TEST 22 WRITE ECC TEST 2

2894            THIS IS A WRITE ECC TEST
                WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
                TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
                OF ALL ONES,

2899    ;*****************************************************************
```

```
3040    **********************************************************************
        TEST 23 READ ECC ENABLED 2A

3043            THIS IS AN ECC READ DATA TEST
                ERROR CORRECTION IS ENABLED
                NO ERROR IS INSERTED
                GOOD DATA USED IS 256 WORDS OF 177777
                COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
                TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA

3050    **********************************************************************

3212    **********************************************************************
        TEST 24 READ ECC ENABLED 2B

3215            THIS IS AN ECC READ DATA TEST
                ERROR CORRECTION IS ENABLED
                A CORRECTABLE ERROR IS INSERTED IN BIT POSITION 32
                GOOD DATA USED IS 256 WORDS OF 177777
                COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
                TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA

3222    **********************************************************************

3389    **********************************************************************
        TEST 25 READ ECC ENABLED 2C

3392            THIS IS AN ECC READ DATA TEST
                ERROR CORRECTION IS ENABLED
                A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 32 AND 21
                GOOD DATA USED IS 256 WORDS OF 177777

3396            COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
                TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA

3399    **********************************************************************

3580    **********************************************************************
        TEST 26 WRITE ECC TEST 3

3583            THIS IS A WRITE ECC TEST
                WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
                TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
                OF ALL 52525.

3588    **********************************************************************

3729    **********************************************************************
        TEST 27 READ ECC ENABLED 3A
```

```
                3732            THIS IS AN ECC READ DATA TEST
                                ERROR CORRECTION IS ENABLED
                                NO ERROR IS INSERTED
                                GOOD DATA USED IS 256 WORDS OF 52525
                                COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
                                TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA

                3739    ********************************************************************

                3901    ********************************************************************
                        TEST 30 READ ECC ENABLED 3B

                3904            THIS IS AN ECC READ DATA TEST
                                ERROR CORRECTION IS ENABLED
                                A CORRECTABLE ERROR IS INSERTED IN BIT POSITION 4128
                                THIS IS THE LAST BIT OF THE ECC
                                GOOD DATA USED IS 256 WORDS OF 52525
                                COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
                                TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA

                3912    ********************************************************************

                4085    ********************************************************************
                        TEST 31 READ ECC ENABLED 3C

                4088            THIS IS AN ECC READ DATA TEST
                                ERROR CORRECTION IS ENABLED
                                A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 296 THRU 308
                                THIS IS IN WORD NUMBER 19 AND 20
                                GOOD DATA USED IS 256 WORDS OF 52525
                                COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
                                TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA

                4096    ********************************************************************

                4275    ********************************************************************
                        TEST 32 READ ECC ENABLED 3D

                4278            THIS IS AN ECC READ DATA TEST
                                ERROR CORRECTION IS ENABLED
                                A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 32 AND 4096
                                4096 IS THE LAST DATA BIT
                                GOOD DATA USED IS 256 WORDS OF 52525
                                COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
                                TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA

                4286    ********************************************************************

                4470    ********************************************************************
                        TEST 33 PROGRAM INTERRUPT
```

```
        4473            PROGRAM INTERRUPT IS TESTED BY SETTING RDY AND IE
                        IN RHCS1 AT THE SAME TIME
                        THIS SHOULD INTERRUPT THROUGH LOCATION 254

        4476            THE PROCESSOR PRIORITY IS SET TO 4
                ***************************************************************

        4512    ***************************************************************
                TEST 34 INTERRUPT AT PROCESSOR AND DISK PRIORITY SAME

        4515            PROCESSOR PRIORITY IS SET AT 5 (SAME AS THE DISK)
                        IE AND RDY IS SET, THIS SHOULD NOT INTERRUPT
                ***************************************************************

        4548    ***************************************************************
                ***************************************************************
                TEST 35 END OF DRIVE

        4552            THIS IS THE END OF TEST FOR ONE DRIVE
                        IF THERE ARE MORE DRIVES THEN THE PROGRAM
                        JUMPS TO TEST 5 FOR NEXT DRIVE TEST
                        END PASS IS REACHED ONLY AFTER ALL DRIVES ARE COMPLETE

        4557    ***************************************************************

        4593    ***************************************************************

        *****************************************************************
4595    END OF PASS ROUTINE
        *****************************************************************

        4597    INCREMENT THE PASS NUMBER ($PASS)
                TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                IF THERES A MONITOR GO TO IT
                IF THERE ISN'T JUMP TO TST1

        *****************************************************************
4637    SUBROUTINES
        *****************************************************************
```

F2

```
                ;*****************************************************************
                ;SAVE REGISTERS ROUTINE
                ;*****************************************************************

                ;*****************************************************************
                ;FLOAT 1 AND 0
                ;*****************************************************************

                ;*****************************************************************
                ;CLEAR MEMORY ROUTINE
                ;*****************************************************************

                ;*****************************************************************
                ;LOCAL TRAPS
                ;*****************************************************************

                ;*****************************************************************
                ;CLEAD DISK ROUTINE
                ;*****************************************************************

                ;*****************************************************************
                ;CHECK DISK STATUS ROUTINE
                ;*****************************************************************

                ;*****************************************************************
                ;SAVE ROUTINE
                ;*****************************************************************

                ;*****************************************************************
                ;WRITE CHECK ROUTINE
                ;*****************************************************************

                ;*****************************************************************
                ;COMPARE ROUTINE
                ;*****************************************************************

                ;*****************************************************************
                ;CRC GENERATION ROUTINE
                ;*****************************************************************
```

```
                ;*****************************************************************
5436            ;JAM CURRENT CYLINDER ROUTINE
                ;*****************************************************************

                ;*****************************************************************
5473            ;ECC GENERARION AND COMPARISON ROUTINE
                ;*****************************************************************

                ;*****************************************************************
5809            ;RH BASE ADDRESS CHANGE ROUTINE
                ;*****************************************************************

5810                    ;THIS ROUTINE WILL ALLOW THE CHANGE OF THE BASE
                        ;ADDRESS FROM 176700 TO ANY TYPED VALUE

                ;*****************************************************************
5881            ;DISK SIMULATION
                ;*****************************************************************

5882            ;*****************************************************************
                ;*****************************************************************

6028            ;*****************************************************************

6050            ;*****************************************************************

6057            ;*****************************************************************
                ;*****************************************************************

6184            ;*****************************************************************
                ;*****************************************************************

6251            ;*****************************************************************
                ;*****************************************************************

6319            ;*****************************************************************
                ;*****************************************************************

6392            ;*****************************************************************
                ;*****************************************************************

6424            ;*****************************************************************
                ;*****************************************************************

6541            ;*****************************************************************
                ;*****************************************************************

6685            ;*****************************************************************
                ;*****************************************************************
```

```
                6798    ;*****************************************************************
                6800    ;*****************************************************************
                6847    ;*************************
                        ;*****************************************************************
                        ;*****************************************************************
        6850    ;SCOPE HANDLER ROUTINE
                        ;*****************************************************************
                6852    ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS, IT WILL INCREMENT
                        ;AND LOAD THE TEST NUMBER(STSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                        ;AND LOAD THE ERROR FLAG (SERFLG) 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
                6922    ;*****************************************************************
                        ;*****************************************************************
        6924    ;CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
                        ;*****************************************************************
                6926    ;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
                6998    ;*****************************************************************
```

```
                ;*****************************************************************
        6992    ;TYPE ROUTINE
                ;*****************************************************************

                6994    ;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

                        ;2) USING A JSR INSTRUCTION
                        ;       MOV     PS,-(SP)        ;;PUSH PROCESSOR STATUS WORD ON THE STACK
                        ;       JSR     PC,$TYPE        ;;CALL TYPE ROUTINE
                        ;       MESADDR                 ;;FIRST ADRESS OF MESSAGE

                7037    ;*****************************************************************

                ;*****************************************************************
        7039    ;TTY INPUT ROUTINE
                ;*****************************************************************

                7048    ;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

                7065    ;TK SERVICE ROUTINE
                        ;THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT

                7087    ;*****************************************************************
                        ;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                        ;CALL:
                        ;       RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                        ;       RETURN HERE             ;;CHARACTER IS ON THE STACK

                7107    ;*****************************************************************
                        ;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
```

```
                7143    ;*****************************************************************

                        ;*****************************************************************
        7145    ;READ AN OCTAL NUMBER FROM THE TTY
                        ;*****************************************************************

                7147    ;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

                7197    ;*****************************************************************

                        ;*****************************************************************
        7199    ;ERROR HANDLER ROUTINE
                        ;*****************************************************************

                7201    ;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 $ERRTYP 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

                7243    ;*****************************************************************

                        ;*****************************************************************
        7245    ;ERROR MESSAGE TYPEOUT ROUTINE
                        ;*****************************************************************

                7247    ;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                        ;ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                        ;AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

                7300    ;*****************************************************************
                        ;*****************************************************************
```

```
                ;*****************************************************************
        7303    ;BINARY TO OCTAL (ASCII) AND TYPE
                ;*****************************************************************

                7305    ;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

                7379    ;*****************************************************************

                ;*****************************************************************
        7381    ;TRAP DECODER
                ;*****************************************************************

                7383    ;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.

                ;*****************************************************************
        7396    ;TRAP TABLE
                ;*****************************************************************

                7398    ;THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                        ;BY THE "TRAP" INSTRUCTION.

                7418    ;*****************************************************************
```

L2

```
                ;*****************************************************************
        7420    ;POWER DOWN AND UP ROUTINES
                ;*****************************************************************

                7460    ;*************************************************************************
                        ;*************************************************************************
```

M2

```
4176                                    .TITLE  MAINDEC-11-DERPTA-A
 (1)                                    ;*COPYRIGHT (C) 1975
 (1)                                    ;*DIGITAL EQUIPMENT CORP.
 (1)                                    ;*MAYNARD, MASS. 01754
 (1)                                    ;*
 (1)                                    ;*PROGRAM BY SUB MALLICK
 (1)                                    ;*
 (1)                                    ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
 (1)                                    ;*PACKAGE (MAINDEC-11-DZQAC-AS).
 (1)                                    ;*
4177
4745
 (1)                                    .SBTTL  OPERATIONAL SWITCH SETTINGS
 (1)                                    ;*
 (1)                                    ;*      SWITCH                  USE
 (1)                                    ;*      ------          --------------------
 (1)                                    ;*        15            HALT ON ERROR
 (1)                                    ;*        14            LOOP ON TEST
 (1)                                    ;*        13            INHIBIT ERROR TYPEOUTS
 (1)                                    ;*        11            INHIBIT ITERATIONS
 (1)                                    ;*        10            BELL ON ERROR
 (1)                                    ;*         9            LOOP ON ERROR
 (1)                                    ;*         8            LOOP ON TEST IN SWR<7:0>
4746                                    ;*         7            STOP FURTHER COMPARES IF SW08 IS LOW
4747                                    ;*         6            ECC TEST-COMPARE END RESULTS ONLY IF SW08 IS LOW
4748
 (1)                                    .SBTTL  BASIC DEFINITIONS
 (1)
 (1)                                    ;*INITIAL ADDRESS OF THE STACK POINTER *** 1000 ***
 (1)          001000                    STACK=  1000
 (1)                                    .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
 (1)                                    .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
 (1)          177776                    PS=     177776          ;;PROCESSOR STATUS WORD
 (1)                                    .EQUIV  PS,PSW
 (1)          177774                    STKLMT= 177774          ;;STACK LIMIT REGISTER
 (1)          177772                    PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
 (1)          177570                    SWR=    177570          ;;SWITCH REGISTER
 (1)          177570                    DISPLAY=SWR
 (1)
 (1)                                    ;*GENERAL PURPOSE REGISTER DEFINITIONS
 (1)          000000                    R0=     %0              ;;GENERAL REGISTER
 (1)          000001                    R1=     %1              ;;GENERAL REGISTER
 (1)          000002                    R2=     %2              ;;GENERAL REGISTER
 (1)          000003                    R3=     %3              ;;GENERAL REGISTER
 (1)          000004                    R4=     %4              ;;GENERAL REGISTER
 (1)          000005                    R5=     %5              ;;GENERAL REGISTER
 (1)          000006                    R6=     %6              ;;GENERAL REGISTER
 (1)          000007                    R7=     %7              ;;GENERAL REGISTER
 (1)                                    .EQUIV  R6,SP           ;;STACK POINTER
 (1)                                    .EQUIV  R7,PC           ;;PROGRAM COUNTER
 (1)
 (1)                                    ;*PRIORITY LEVEL DEFINITIONS
 (1)          000000                    PR0=    0               ;;PRIORITY LEVEL 0
 (1)          000040                    PR1=    40              ;;PRIORITY LEVEL 1
```

```
(1)                 000100          PR2=    100             ;;PRIORITY LEVEL 2
(1)                 000140          PR3=    140             ;;PRIORITY LEVEL 3
(1)                 000200          PR4=    200             ;;PRIORITY LEVEL 4
(1)                 000240          PR5=    240             ;;PRIORITY LEVEL 5
(1)                 000300          PR6=    300             ;;PRIORITY LEVEL 6
(1)                 000340          PR7=    340             ;;PRIORITY LEVEL 7
(1)
(1)                                 ;*"SWITCH REGISTER" SWITCH DEFINITIONS
(1)                 100000          SW15=   100000
(1)                 040000          SW14=   40000
(1)                 020000          SW13=   20000
(1)                 010000          SW12=   10000
(1)                 004000          SW11=   4000
(1)                 002000          SW10=   2000
(1)                 001000          SW09=   1000
(1)                 000400          SW08=   400
(1)                 000200          SW07=   200
(1)                 000100          SW06=   100
(1)                 000040          SW05=   40
(1)                 000020          SW04=   20
(1)                 000010          SW03=   10
(1)                 000004          SW02=   4
(1)                 000002          SW01=   2
(1)                 000001          SW00=   1
(1)                                 .EQUIV  SW09,SW9
(1)                                 .EQUIV  SW08,SW8
(1)                                 .EQUIV  SW07,SW7
(1)                                 .EQUIV  SW06,SW6
(1)                                 .EQUIV  SW05,SW5
(1)                                 .EQUIV  SW04,SW4
(1)                                 .EQUIV  SW03,SW3
(1)                                 .EQUIV  SW02,SW2
(1)                                 .EQUIV  SW01,SW1
(1)                                 .EQUIV  SW00,SW0
(1)
(1)                                 ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)             100000              BIT15=  100000
(1)             040000              BIT14=  40000
(1)             020000              BIT13=  20000
(1)             010000              BIT12=  10000
(1)             004000              BIT11=  4000
(1)             002000              BIT10=  2000
(1)             001000              BIT09=  1000
(1)             000400              BIT08=  400
(1)             000200              BIT07=  200
(1)             000100              BIT06=  100
(1)             000040              BIT05=  40
(1)             000020              BIT04=  20
(1)             000010              BIT03=  10
(1)             000004              BIT02=  4
(1)             000002              BIT01=  2
(1)             000001              BIT00=  1
(1)                                 .EQUIV  BIT09,BIT9
(1)                                 .EQUIV  BIT08,BIT8
```

```
(1)                                 .EQUIV  BIT07,BIT7
(1)                                 .EQUIV  BIT06,BIT6
(1)                                 .EQUIV  BIT05,BIT5
(1)                                 .EQUIV  BIT04,BIT4
(1)                                 .EQUIV  BIT03,BIT3
(1)                                 .EQUIV  BIT02,BIT2
(1)                                 .EQUIV  BIT01,BIT1
(1)                                 .EQUIV  BIT00,BIT0
(1)
(1)                                 ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(1)          000004                 ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
(1)          000010                 RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)          000014                 TBITVEC=14              ;;"T" BIT
(1)          000014                 TRTVEC= 14              ;;TRACE TRAP
(1)          000014                 BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
(1)          000020                 IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)          000024                 PWRVEC= 24              ;;POWER FAIL
(1)          000030                 EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
(1)          000034                 TRAPVEC=34              ;;"TRAP" TRAP
(1)          000060                 TKVEC=  60              ;;TTY KEYBOARD VECTOR
(1)          000064                 TPVEC=  64              ;;TTY PRINTER VECTOR
(1)          000240                 PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
4749
4750
(1)                                 .SBTTL  TRAP CATCHER
(1)
(1)          000000                         .=0
(1)                                 ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ",.+2,HALT"
(1)                                 ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
(1)                                 ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
(1)
(1)                                 .SBTTL  STARTING ADDRESS(ES)
(1)          000200                         .=200
(1)
(1)  000200  000137  004220                 JMP     @#BEGIN         ;;JUMP TO STARTING ADDRESS OF PROGRAM
4751         000210                         .=210
4752  000210  000137  004210                 JMP     @#BEGIN2        ;JUMP SELECT TEST
4753                                ;*STARTING ADDRESS 200 FOR NORMAL STARTS
4754                                ;*THIS WILL TEST ALL RP04'S ON THE SYSTEM A SINGLE DRIVE AT A TIME
4755                                ;*
4756                                ;*STARTING ADDRESS 210 WILL TEST ONLY ONE SPECIFIED DRIVE
4757
(1)                                 .SBTTL  MEMORY MANAGEMENT DEFINITIONS
(1)
(1)                                 ;*KT11 VECTOR ADDRESS
(1)
(1)          000250                 MMVEC=  250
(1)
(1)                                 ;*KT11 STATUS REGISTER ADDRESSES
(1)
(1)          177572                 SR0=    177572
(1)          177574                 SR1=    177574
(1)          177576                 SR2=    177576
(1)          172516                 SR3=    172516
```

```
   (1)
   (1)                                  ;*KERNAL "I" PAGE DESCRIPTOR REGISTERS
   (1)
   (1)          172300                  KIPDR0= 172300
   (1)          172302                  KIPDR1= 172302
   (1)          172304                  KIPDR2= 172304
   (1)          172306                  KIPDR3= 172306
   (1)          172310                  KIPDR4= 172310
   (1)          172312                  KIPDR5= 172312
   (1)          172314                  KIPDR6= 172314
   (1)          172316                  KIPDR7= 172316
   (1)
   (1)                                  ;*KERNAL "I" PAGE ADDRESS REGISTERS
   (1)
   (1)          172340                  KIPAR0= 172340
   (1)          172342                  KIPAR1= 172342
   (1)          172344                  KIPAR2= 172344
   (1)          172346                  KIPAR3= 172346
   (1)          172350                  KIPAR4= 172350
   (1)          172352                  KIPAR5= 172352
   (1)          172354                  KIPAR6= 172354
   (1)          172356                  KIPAR7= 172356
   (1)
  4758                                  ;*****************************************************************
  4759          001110                          .=1110
```

```
 4760                                   ;*****************************************************************
  (1)
  (1)                                   .SBTTL  COMMON TAGS
  (1)
  (1)                                   ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
  (1)                                   ;*USED IN THE PROGRAM.
  (1)
  (1)           000046                          .=46
  (1)  000046   023352                          $ENDAD                  ;;LOGICAL END OF PROGRAM
  (1)
  (1)           001100                          .=1100
  (1)
  (1)  001100                           $CMTAG:                         ;;START OF COMMON TAGS
  (1)  001100   000000                  $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
  (1)  001102      000                  $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
  (1)  001103      000                  $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
  (1)  001104   000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
  (1)  001106   000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP
  (1)  001110   000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
  (1)  001112   000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
  (1)  001114      000                  $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
  (1)  001115      001                  $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
  (1)  001116   000000                  $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
  (1)  001120   000000                  $GDADR: .WORD   0               ;;CONTAINS  OF 'GOOD' DATA
  (1)  001122   000000                  $BDADR: .WORD   0               ;;CONTAINS  OF 'BAD' DATA
  (1)  001124   000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
  (1)  001126   000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
  (1)  001130   000000  000000  000000          .WORD   0,0,0           ;;RESERVED--NOT TO BE USED
  (1)  001136   177560                  $TKS:   177560                  ;;TTY KBD STATUS
  (1)  001140   177562                  $TKB:   177562                  ;;TTY KBD BUFFER
  (1)  001142   177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG.
  (1)  001144   177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG.
  (1)  001146      000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
  (1)  001147      002                  $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
  (1)  001150      012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
  (1)  001151      000                  $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
  (1)  001152   000000                  $REGAD: .WORD   0               ;;CONTAINS THE  FROM
  (1)                                                                   ;;WHICH  ($REG0) WAS OBTAINED
  (3)  001154   000000                  $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
  (3)  001156   000000                  $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
  (3)  001160   000000                  $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
  (3)  001162   000000                  $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
  (3)  001164   000000                  $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
  (3)  001166   000000                  $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
  (3)  001170   000000                  $TMP0:  .WORD   0               ;;USER DEFINED
  (3)  001172   000000                  $TMP1:  .WORD   0               ;;USER DEFINED
  (3)  001174   000000                  $TMP2:  .WORD   0               ;;USER DEFINED
  (3)  001176   000000                  $TMP3:  .WORD   0               ;;USER DEFINED
  (3)  001200   000000                  $TMP4:  .WORD   0               ;;USER DEFINED
  (3)  001202   000000                  $TMP5:  .WORD   0               ;;USER DEFINED
  (1)  001204   000000                  $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
  (1)  001206   000000                  $ESCAPE:0                       ;;ESCAPE ON ERROR
  (1)  001210   177607  000377         $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
  (1)  001214      077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
```

```
   (1)  001215     015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
   (1)  001216  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
```

```
    (2)                                 ;*****************************************************************
    (1)
    (1)                                 .SBTTL  ERROR POINTER TABLE
    (1)
    (1)                                 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
    (1)                                 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
    (1)                                 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
    (1)                                 ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
    (1)                                 ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
    (1)
    (1)                                 ;*      EM              ;;POINTS TO THE ERROR MESSAGE
    (1)                                 ;*      DH              ;;POINTS TO THE DATA HEADER
    (1)                                 ;*      DT              ;;POINTS TO THE DATA
    (1)                                 ;*      DF              ;;POINTS TO THE DATA FORMAT
    (1)
    (1)
    (1)  001220                         $ERRTB:
   4761
   4762
   4763                                 ;*****************************************************************
   4764                                 ;
   4765
   4766                                 ;ITEM1
   4767  001220  040130                          EM1                     ;WRONG DATA IN READING OR WRITING HARDWARE REGISTER
   4768  001222  042734                          DH1                     ;PC
   4769                                                                   ;REG, ADDR,
   4770                                                                   ;GOOD DATA
   4771                                                                   ;RECEIVED DATA
   4772  001224  047054                                  DT1             ;$ERRPC,REGADR,$GDDAT,$BDDAT
   4773  001226  047554                                  DF1             ;0,0,0,0,0
   4774
   4775
   4776
   4777
   4778
   4779                                 ;ITEM2
   4780  001230  040213                          EM2                     ;ERROR ON  DATA COMMAND
   4781
   4782  001232  045720                          DH33                    ;PC
   4783                                                                   ;PC OF JSR
   4784                                                                   ;TEST NO
   4785                                                                   ;WORD NO,
   4786                                                                   ;GOOD DATA
   4787                                                                   ;CONTENTS OF RHCS1
   4788                                                                   ;CONTENTS OF RHDS1
   4789                                                                   ;CONTENTS OF RHER1
   4790  001234  047426                                  DT33            ;$ERRPC,PCJSR,$TSTNM,ERWORD,$GDDAT,CS1,DS1,ER1
   4791  001236  047717                                  DF33            ;0,0,0,1,0,0,0,0
   4792
   4793
   4794                                 ;ITEM3
   4795  001240  040213                          EM2                     ;ERROR ON  DATA COMMAND
   4796
   4797  001242  045475                          DH32                    ;PC
```

```
                                                                ;PC OF JSR
                                                                ;TEST NO
                                                                ;WOR. NO,
                                                                ;GOOD DATA
                                                                ;BAD DATA
                                                                ;CONTENTS OF RHCS1
                                                                ;CONTENTS OF RHDS1
                                                                ;CONTENTS OF RHER1

001244  047402                          DT32                  ;SERRPC,PCJSR,STSTNM,ERWORD,SGDDAT,SBDDAT,CS1,DS1,ER1
001246  047706                          DF32                  ;0,0,0,1,0,0,0,0,0,


                                ;ITEM4
001250  040213                          EM2                   ;ERROR ON  DATA COMMAND

001252  045273                          DH31                  ;PC
                                                              ;TEST NO
                                                              ;WORD NO,
                                                              ;GOOD DATA
                                                              ;BAD DATA
                                                              ;CONTENTS OF RHCS1
                                                              ;CONTENTS OF RHDS1
                                                              ;CONTENTS OF RHER1

001254  047360                          DT31                  ;SERRPC,STSTNM,ERWORD,SGDDAT,SBDDAT,CS1,DS1,ER1
001256  047676                          DF31                  ;0,0,1,0,0,0,0,0,


                                ;ITEM5
001260  000000                          0                     ;
001262  000000                          0                     ;
001264  047360                          DT31                  ;SERRPC,STSTNM,ERWORD,SGDDAT,SBDDAT,CS1,DS1,ER1
001266  047676                          DF31                  ;0,0,1,0,0,0,0,0,


                                ;ITEM6
001270  040242                          EM6                   ;ERROR ON WRITE HEADER AND DATA

001272  045475                          DH32                  ;PC
                                                              ;PC OF JSR
                                                              ;TEST NO
                                                              ;WORD NO,
                                                              ;GOOD DATA
                                                              ;BAD DATA
                                                              ;CONTENTS OF RHCS1
                                                              ;CONTENTS OF RHDS1
                                                              ;CONTENTS OF RHER1

001274  047402                          DT32                  ;SERRPC,PCJSR,STSTNM,ERWORD,SGDDAT,SBDDAT,CS1,DS1,ER1
001276  047706                          DF32                  ;0,0,0,1,0,0,0,0,0,
```

```
                                ;ITEM7
001300  040242                          EM6             ;ERROR ON WRITE HEADER AND DATA
001302  043057                          DH2             ;PC
                                                        ;TEST NO
                                                        ;WORD NO,
                                                        ;GOOD DATA
                                                        ;BAD DATA
001304  047102                          DT3             ;$ERRPC,$TSTNM,ERWORD,$GDDAT,$BDDAT
001306  047565                          DF3             ;0,0,1,0,0,

                                ;ITEM10
001310  000000                          0               ;
001312  000000                          0               ;
001314  047102                          DT3             ;$ERRPC,$TSTNM,ERWORD,$GDDAT,$BDDAT
001316  047565                          DF3             ;0,0,1,0,0,

                                ;ITEM11
001320  040301                          EM11            ;CONTROLLER OR DRIVE STATUS
001322  043202                          DH11            ;PC
                                                        ;TEST NO
                                                        ;FAILING REG, ADDR
                                                        ;CONTENTS OF RHCS1
                                                        ;CONTENTS OF RHCS2
                                                        ;CONTENTS OF RHDS1
                                                        ;CONTENTS OF RHER1
001324  047116                          DT11            ;$ERRPC,$TSTNM,$BDADR,CS1,CS2,DS1,ER1
001326  047572                          DF11            ;0,0,0,0,0,0

                                ;ITEM12
001330  040301                          EM11            ;WRONG DATA FROM SILO

001332  042734                          DH1             ;PC
                                                        ;REG,ADDR
                                                        ;GOOD DATA
                                                        ;RECEIVED DATA
001334  047054                          DT1             ;$ERRPC,REGADR,$GDDAT,$BDDAT
001336  047554                          DF1             ;0,0,0,0

                                ;ITEM13
001340  000000                          0
001342  000000                          0
001344  047054                          DT1             ;$ERRPC,TSTNM,REGADR,$GDDAT,$BDDAT
001346  047554                          DF1             ;0,0,0,0,0

                                ;ITEM14
001350  040334                          EM14            ;REGISTER FAILED
001352  043361                          DH14            ;PC
                                                        ;FAILING REG, ADDR
```

K3

```
                                                                ;CONTENTS OF FAILING REG.
                                                                ;CONTENTS OF RHCS1
                                                                ;CONTENTS OF RHCS2
                                                                ;CONTENTS OF RHDS1
                                                                ;CONTENTS OF RHER1
001354  047136                          DT14                  ;$ERRPC,$BDADR,$BDDAT,CS1,CS2,DS1,ER1
001356  047601                          DF14                  ;0,0,0,0,0,0,0

                                ;ITEM15
001360  040354                          EM15                  ;SPECIFIED REG, NON EXISTANT SO ABORT
                                                               ;PROGRAM
001362  043563                          DH15                  ;PC
                                                               ;ADDR. OF REG
001364  047160                          DT15                  ;$ERRPC,TEMP1
001366  047611                          DF15                  ;0,0

                                ;ITEM16
001370  040425                          EM16                  ;WAIT LOOP FAILED
001372  043605                          DH16                  ;PC
                                                               ;WAT PC
                                                               ;BIT WANTED
                                                               ;REG. ADR.
                                                               ;REG. CONT.
001374  047166                          DT16                  ;$ERRPC,$TMP3,$TMP1,$TMP0,$BDDAT
001376  047613                          DF16                  ;0,0,0,0

                                ;ITEM17
001400  040446                          EM17                  ;WRITE CHECK FAILING
001402  043724                          DH17                  ;PC
                                                               ;TEST NO
                                                               ;CONTENTS OF RHBA
                                                               ;CONTENTS OF RHDB
                                                               ;CONTENTS OF RHWC
                                                               ;CONTENTS OF RHCS1
                                                               ;CONTENTS OF RHCS2
001404  047202                          DT17                  ;$ERRPC,$TSTNM,$BA,DB,WC,CS1,CS2
001406  047617                          DF17                  ;0,0,0,0,0,0,0

                                ;ITEM20
001410  040472                          EM20                  ;REGISTER FAILING
001412  044107                          DH20                  ;PC
                                                               ;TST NO
                                                               ;CONTENTS OF RHER1
                                                               ;CONTENTS OF RHER2
                                                               ;CONTENTS OF RHER3
                                                               ;CONTENTS OF RHAS
                                                               ;CONTENTS OF RHDS1
001414  047222                          DT20                  ;$ERRPC,TSTNM ER1,ER2,ER3,AS,DS1
001416  047626                          DF20                  ;0,0,0,0,0,0,0
```

```
                                        ;ITEM21
                                EM21            ;INTERRUPT FAILING
                                DH21            ;PC
                                                ;TEST NO
                                                ;CONTENTS OF RHCS1
                                                ;CONTENTS OF RHAS
                                                ;CONTENTS OF RHDS1
                                DT21            ;$ERRPC,TSTNM,CS1,AS,DS1
                                DF21            ;0,0,0,0,0


                                        ;ITEM22
                                EM22            ;MISSMATCH IN DRIVE PRESENT
                                                ;LOOKING AT RHAS AND RHCS2-NED(BIT@12)
                                                ;DRIVE PRESENT DO NOT AGREE
                                                ;NOTE: ON DUAL PORT SYSTEM
                                                ;DRIVE ON OTHER PORT WILL NOT GIVE NED
                                                ;HENCE THERE WILL BE A MISSMATCH
                                                ;177777-MEANS NOT PRESENT
                                DH22            ;PC
                                                ;TEST NO
                                                ;RHAS UNIT
                                                ;RHCS2 UNIT
                                                ;
                                DT22            ;$ERRPC,TSTNM$,$GDDAT,$BDDAT
                                DF22            ;0,0,0,0


                                        ;ITEM23
                                0               ;MISSMATCH IN DRIVE PRESENT
                                                ;LOOKING AT RHAS AND RHCS2-NED(BIT@12)
                                                ;DRIVE PRESENT DO NOT AGREE
                                                ;177777-MEANS NOT PRESENT
                                0               ;PC
                                                ;TEST NO
                                                ;RHAS UNIT
                                                ;RHCS2 UNIT
                                                ;
                                DT22            ;$ERRPC,TSTNM$,$GDDAT,$BDDAT
                                DF22            ;0,0,0,0


                                        ;ITEM 24
                                EM24            ;LOOK AHEAD REGISTER AT THE
                                                ;BEGINNING OF A SECTOR IS IN
                                                ;ERROR
                                DH24            ;PC
                                                ;RHDST
                                                ;BAD RHLA
                                                ;GOOD RHLA
                                                ;SECTOR NO
                                                ;SECTOR CLOCK
```

```
5014  001454  047270                    DT24                ;$ERRPC,DST,$BDDAT,$TMP1,$TMP2,$TMP3
5015  001456  047646                    DF24                ;0,0,0,0,0
5016
5017                                ;ITEM 25
5018  001460  041223                    EM25                ;LOOK AHEAD REGISTER IS
5019                                                          ;IN ERROR
5020
5021  001462  044503                    DH24                ;PC
5022                                                          ;RHDST
5023                                                          ;BAD RHLA
5024                                                          ;GOOD RHLA
5025                                                          ;SECTOR NO
5026                                                          ;SECTOR CLOCK
5027  001464  047270                    DT24                ;$ERRPC,DST,$BDDAT,$TMP1,$TMP2,$TMP3
5028  001466  047646                    DF24                ;0,0,0,0,0
5029                                ;ITEM26
5030  001470  040301                    EM11                ;CONTROLLER OR DRIVE STATUS
5031
5032  001472  044641                    DH26                ;PC
5033                                                          ;PC OF JSR
5034                                                          ;FAILING REGISTER ADDRESS
5035                                                          ;CONTENTS OF RHCS1
5036                                                          ;CONTENTS OF RHCS2
5037                                                          ;CONTENTS OF RHDS1
5038                                                          ;CONTENTS OF RHER1
5039
5040  001474  047306                    DT26                ;$ERRPC,PCJSR,$BDADR,CS1,CS2,DS1,ER1
5041  001476  047654                    DF26                ;0,0,0,0,0,0,
5042
5043
5044
5045                                ;ITEM27
5046  001500  040130                    EM1                 ;ERROR IN READING OR WRITING HARDWARE REGISTER
5047
5048  001502  045023                    DH27                ;PC
5049                                                          ;PC OF JSR
5050                                                          ;TEST NUMBER
5051                                                          ;FAILING REGISTER
5052                                                          ;GOOD DATA
5053                                                          ;RECEIVED DATA
5054
5055  001504  047326                    DT27                ;$ERRPC,PCJSR,TSTNM,REGADR,$GDDAT,$BDDAT
5056  001506  047663                    DF27                ;0,0,0,0,0,0
5057
5058
5059
5060                                ;ITEM30
5061  001510  041263                    EM30                ;CURRENT CYLINDER DOES NOT REFLECT DESIRED CYLINDER REG.
5062  001512  045162                    DH30                ;PC
5063                                                          ;PC OF JSR
5064                                                          ;REGISTER ADDRESS
5065                                                          ;GOOD DATA
5066                                                          ;BAD DATA
5067
```

```
5068  001514  047344                    DT30                ;$ERRPC,PCJSR,REGADR,$GDDAT,$BDDAT
5069  001516  047671                    DF30                ;0,0,0,0,0
5070
5071
5072
5073                            ;ITEM31
5074  001520  041405                    EM31                ;ECC GENERATED IS INCORRECT
5075                                                        ;EVERY WORD IN THIS SECTOR IS GIVEN IN "DATA USED"
5076
5077  001522  046131                    DH34                ;PC
5078                                                        ;TEST NUMBER
5079                                                        ;GOOD ECC1
5080                                                        ;GOOD EC2C
5081                                                        ;WRITTEN ECC1
5082                                                        ;WRITTEN ECC2
5083                                                        ;DATA USED
5084
5085  001524  047450                    DT34                ;$ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK
5086
5087  001526  047727                    DF34                ;0,0,0,0,0,0,0
5088
5089
5090                            ;ITEM32
5091  001530  041530                    EM32                ;ON READ COMMAND AFTER DATA AND ECC HAVE BEEN READ
5092                                                        ;ECC REGISTER OR RHER1 IS IN ERROR
5093                                                        ;ONLY LOWER 11 BITS OF PATTERN REGISTER
5094                                                        ;CAN BE READ
5095                                                        ;THIS SHUOLD MATCH LOWER 11 BITS OF ECC1
5096
5097  001532  046304                    DH35                ;PC
5098                                                        ;TEST NUMBER
5099                                                        ;GOOD ECC1
5100                                                        ;GOOD ECC2
5101                                                        ;PATTERN REGISTER
5102                                                        ;RHER1
5103
5104  001534  047470                    DT35                ;$ERRPC,TSTNM,GECC1,GECC2,EC2,ER1
5105
5106  001536  047736                    DF35                ;0,0,0,0,0,0
5107
5108
5109
5110                            ;ITEM33
5111  001540  042014                    EM33                ;HIGH COUNT BIT NOT HIGH AFTER 38859 CLOCKS
5112  001542  046500                    DH36                ;PC
5113                                                        ;PC OF JSR
5114                                                        ;TEST NUMBER
5115                                                        ;RHMR
5116                                                        ;POSITION REG,
5117                                                        ;PATTERN REGISTER
5118
5119  001544  047512                    DT36                ;$ERRPC,PCJSR,TSTNM,MR,EC1,EC2
5120
5121  001546  047746                    DF36                ;0,0,0,0,0,0
```

```
5122
5123                                    ;ITEM34
5124  001550  042066                            EM34            ;ZERO DETECT BIT NOT HIGH WHEN THE
5125                                                            ;32 BIT ECC REGISTER HAS ITS 21 BITS
5126                                                            ;OF ZEROS
5127                                                            ;ERROR PRINTOUT WILL CONTINUE TILL
5128                                                            ;ZERO DETECT BIT IS HIGH
5129  001552  046500                            DH36            ;PC
5130                                                            ;PC OF JSR
5131                                                            ;TEST NUMBER
5132                                                            ;RHNR
5133                                                            ;POSITION REG.
5134                                                            ;PATTERN REGISTER
5135
5136  001554  047512                            DT36            ;SERRPC,PCJSR,TSTNM,NR,EC1,EC2
5137
5138  001556  047746                            DF36            ;0,0,0,0,0,0
5139
5140
5141
5142                                    ;ITEM35
5143  001560  042161                            EM35            ;POSITION REGISTER OR 11 BITS OF
5144                                                            ;PATTERN REGISTER INCORRECT
5145                                                            ;LOWER 11 BITS OF PATTERN REGISTER
5146                                                            ;SHOULD MATCH LOWER 11 BITS OF GOOD ECC;
5147                                                            ;DATA ENVELOPE AND M-CODE ZEROS ARE IN DECIMAL
5148
5149  001562  046636                            DH37            ;PC
5150                                                            ;TEST NUMBER
5151                                                            ;ECC POSITION
5152                                                            ;GOOD POSITION
5153                                                            ;GOOD ECC1
5154                                                            ;GOOD ECC2
5155                                                            ;ECC PATTERN
5156                                                            ;DATA ENVELOPE
5157                                                            ;M-CODE ZEROS
5158
5159  001564  047530                            DT37            ;SERRPC,TSTNM,EC1,POSITI,GECC1,GECC2,EC2,DATENV,ECODE
5160
5161  001566  047754                            DF37            ;0,0,0,0,0,0,0,0,0
5162
5163
5164
5165                                    ;ITEM36
5166  001570  042460                            EM36            ;ON A READ COMMAND WITH NON CORRECTABLE
5167                                                            ;ERROR INSERTED DCK AND ECH SHOULD BE SET
5168  001572  046304                            DH35            ;PC
5169                                                            ;TEST NUMBER
5170                                                            ;GOOD ECC1
5171                                                            ;GOOD ECC2
5172                                                            ;PATTERN REGISTER
5173                                                            ;POSITION REGISTER
5174                                                            ;RNER1
5175
```

```
5176  001574  047470                        DT35                    ;@ERRPC,TSTNM,GECC1,GECC2,EC2,EC1,ER1
5177
5178  001576  047736                        DF35                    ;0,0,0,0,0,0,0
5179
5180
5181
5182
5183
5184
5185                                ;ITEM37
5186  001600  042646                        EM37                    ;ERROR ON  DATA COMMAND
5187                                                                  ;WITH A16 A17 USED
5188
5189  001602  045273                        DH31                    ;PC
5190                                                                  ;TEST NO
5191                                                                  ;WORD NO,
5192                                                                  ;GOOD DATA
5193                                                                  ;BAD DATA
5194                                                                  ;CONTENTS OF RHCS1
5195                                                                  ;CONTENTS OF RHDS1
5196                                                                  ;CONTENTS OF RHER1
5197
5198  001604  047360                        DT31                    ;@ERRPC,@TSTNM,ERWORD,@GDDAT,@BDDAT,CS1,DS1,ER1
5199  001606  047676                        DF31                    ;0,0,1,0,0,0,0,0,
5200
5201
5202
5203                                ;ITEM40
5204  001610  000000                        0                       ;
5205  001612  000000                        0                       ;
5206  001614  047360                        DT31                    ;@ERRPC,@TSTNM,ERWORD,@GDDAT,@BDDAT,CS1,DS1,ER1
5207  001616  047676                        DF31                    ;0,0,1,0,0,0,0,0,
5208
5209
```

```
                                ;*****************************************************************
                                ;RH11 REGISTERS



                                ;*****************************************************************
                                ;WORD COUNT REGISTER (RHWC)
                                ;EACH BIT IS CALLED BY BIT NUMBER


                                ;BUS ADDRESS REGISTER (RHBA)
                                ;EACH BIT IS CALLED BY BIT NUMBER


                                ;CONTROL AND STATUS REGISTER 2 (RHCS2)

000001                          US1=    1               ;UNIT SELECT (BIT 00)
000002                          US2=    2               ;UNIT SELECT (BIT 01)
000004                          US4=    4               ;UNIT SELECT (BIT 02)
000010                          BAI=    10              ;BUS ADDRESS INCREMENT INHIBIT (BIT 03)
000020                          PAT=    20              ;INVERT PARITY ON MASS BUS TO EVEN (BIT 04)
000040                          CLR=    40              ;CLEAR (BIT 05)
000100                          IR=     100             ;INPUT READY (BIT 06)
000200                          OR=     200             ;OUTPUT READY (BIT 07)
000400                          MPE=    400             ;MASS BUS PARITY ERROR (BIT 08)
001000                          NXF=    1000            ;MISSED TRANSFER ERROR (BIT 09)
002000                          PGE=    2000            ;PROGRAM ERROR (BIT 010)
004000                          NEM=    4000            ;NON EXISTANT MEMORY (BIT 011)
010000                          NED=    10000           ;NON EXISTANT DRIVE (BIT 012)
020000                          UPE=    20000           ;UNIBUS PARITY ERROR (BIT 013)
040000                          WCE=    40000           ;WRITE CHECK ERROR (BIT 014)
100000                          DLT=    100000          ;DATA LATE (BIT 015)

                                ;DATA BUFFER REGISTER (RHDB)
                                ;EACH BIT IS CALLED BY BIT NUMBER


                                ;*****************************************************************
                                ;RP04 REGISTERS
                                ;*****************************************************************


                                ;CONTROL AND STATUS 1 REGISTER, (000)

000001                          GO=     1               ;GO (BIT 00)
000100                          IE=     100             ;INTERRUPT ENABLE (BIT 06)
000200                          RDY=    200             ;READY (BIT 07)
000400                          A16=    400             ;HIGH ORDER UNIBUS BITS (BIT 08)
001000                          A17=    1000            ;HIGH ORDER UNIBUS BITS (BIT 09)
002000                          PSEL=   2000            ;PORT SELECT (BIT 010)
```

```
5265            004000                  DVA=    4000                    ;DEVICE AVAILABLE (BIT #11)
5266            020000                  MCPE=   20000                   ;MASSBUSS PARITY ERROR (BIT #13)
5267            040000                  TRE=    40000                   ;TRANSFER ERROR (BIT #14)
5268            100000                  SC=     100000                  ;SPECIAL CONDITION (BIT #15)
5269
5270                                    ;STATUS REGISTER (RHDS1) (#01)
5271
5272            000001                  DFS=    1                       ;DRIVE FORWARD 5"/SEC.  (BIT #0)
5273            000002                  DFF20=  2                       ;DRIVE FORWARD 20"/SEC.  (BIT #1)
5274            000004                  DIGB=   4                       ;DRIVE TO INNER GAURD BAND (BIT #2)
5275            000010                  GRV=    10                      ;GO REVERSE (BIT #3)
5276            000020                  DL64=   20                      ;DIFFERENCE LESS THAN 64 (BIT #4)
5277            000040                  DE1=    40                      ;DIFFERENCE EQUALS 1 (BIT #5)
5278            000100                  VV=     100                     ;VOLUME VALID (BIT #6)
5279            000200                  DRY=    200                     ;DRIVE READY (BIT #7)
5280            000400                  DPR=    400                     ;DRIVE PRESENT (BIT #8)
5281            001000                  PROG=   1000                    ;PROGRAMABLE (BIT #9)
5282            002000                  LST=    2000                    ;LAST SECTOR TRANSFERRED (BIT #10)
5283            004000                  WRL=    4000                    ;WRITE LOCK (BIT #11)
5284            010000                  MOL=    10000                   ;MEDIUM ON-LINE (BIT #12)
5285            020000                  PIP=    20000                   ;POSITIONING OPERATION IN PROGRESS (BIT #13)
5286            040000                  ERR=    40000                   ;COMPOSIT ERROR. (BIT #14)
5287            100000                  ATA=    100000                  ;ATTENTION ACTIVE (BIT #15)
5288
5289                                    ;ERROR REGISTER #1 (RHER1) (#02)
5290            000001                  ILF=    1                       ;ILLEGAL FUNCTION (BIT #0)
5291            000002                  ILR=    2                       ;ILLEGAL REGISTER (BIT #1)
5292            000004                  RMR=    4                       ;REGISTER MODIFICATION REFUSED (BIT #2)
5293            000010                  PAR=    10                      ;PARITY ERROR (BIT #3)
5294            000020                  FER=    20                      ;FORMAT ERROR (BIT #4)
5295            000040                  WCF=    40                      ;WRITE CLOCK FAIL (BIT #5)
5296            000100                  ECH=    100                     ;ECC HARD ERROR (BIT #6)
5297            000200                  HCE=    200                     ;HEADER COMPARE ERROR (BIT #7)
5298            000400                  HCRC=   400                     ;HEADER CRC ERROR (BIT #8)
5299            001000                  AOE=    1000                    ;ADDRESS OVERFLOW ERROR (BIT #9)
5300            002000                  IAE=    2000                    ;INVALID ADDRESS ERROR (BIT #10)
5301            004000                  WLE=    4000                    ;WRITE LOCK ERROR (BIT #11)
5302            010000                  DTE=    10000                   ;DRIVE TIMING ERROR (BIT #12)
5303            020000                  OPI=    20000                   ;OPERATION INCOMPLETE (BIT #13)
5304            040000                  UNS=    40000                   ;DRIVE UNSAFE (BIT #14)
5305            100000                  DCK=    100000                  ;DATA CHECK ERROR (BIT 15)
5306
5307                                    ;MAINTAINABILITY REGISTER (RHMR)(#03)
5308
5309            000001                  DMD=    1                       ;DIAGINOSTIC MODE (BIT #0)
5310            000002                  MCLK=   2                       ;MAINTAINABILITY CLOCK (BIT #1)
5311            000004                  MINX=   4                       ;MAINTAINABILITY INDEX (BIT #2)
5312            000010                  MSTCK=  10                      ;MAINTAINABILITY SECTOR CLOCK (BIT #3)
5313            000020                  MRD=    20                      ;MAINTAINABILITY READ (BIT #4)
5314            000040                  MWR=    40                      ;MAINTAINABILITY WRITE (BIT #5)
5315            000200                  DENVL=  200                     ;DATA ENVELOPE (BIT #7)
5316            000400                  ZER=    400                     ;ZERO DETECT (BIT #8)
5317            001000                  DTSY=   1000                    ;MAINTAINABILITY SYNC DETECTED (BIT #9)
5318
```

```
                                ;ATTENTION SUMMARY PSEUDO-REGISTER (RHAS) (#04)

000001                          AT0=    1               ;DEVICE 0 (BIT #0)
000002                          AT1=    2               ;DEVICE 1 (BIT #1)
000004                          AT2=    4               ;DEVICE 2 (BIT #2)
000010                          AT3=    10              ;DEVICE 3 (BIT #3)
000020                          AT4=    20              ;DEVICE 4 (BIT #4)
000040                          AT5=    40              ;DEVICE 5 (BIT #5)
000100                          AT6=    100             ;DEVICE 6 (BIT #6)
000200                          AT7=    200             ;DEVICE 7 (BIT #7)

                                ;DESIRED SECTOR/TRACK ADDRESS REGISTER (RHDST) (#1)
                                ;EACH BIT IS CALLED BY BIT NUMBER
                                ;DRIVE TYPE REGISTER (RHDT) (#06)
                                ;EACH BIT IS CALLED BY BIT NUMBER
                                ;LOOK-AHEAD REGISTER (RHLA) (#07)

000001                          EXT1=   1               ;EXTENSION 1 (BIT #0)
000002                          EXT2=   2               ;EXTENSION 2 (BIT #1)
000004                          EXT4=   4               ;EXTENSION 3 (BIT #2)
000010                          EXT10=  10              ;EXTENSION 4 (BIT #3)
000020                          EXT20=  20              ;EXTENSION 5 (BIT #4)
000040                          EXT40=  40              ;EXTENSION 6 (BIT #5)
000100                          SC1=    100             ;SECTOR COUNT FIELD 0 (BIT #6)
000200                          SC2=    200             ;SECTOR COUNT FIELD 1 (BIT #7)
000400                          SC4=    400             ;SECTOR COUNT FIELD 2 (BIT #8)
001000                          SC10=   1000            ;SECTOR COUNT FIELD 3 (BIT #9)
002000                          SC20=   2000            ;SECTOR COUNT FIELD 4 (BIT #10)
004000                          TRK1=   4000            ;TRACK FIELD 1 (BIT #11)
010000                          TRK2=   10000           ;TRACK FIELD 2 (BIT #12)
020000                          TRK4=   20000           ;TRACK FIELD 3 (BIT #13)
040000                          TRK10=  40000           ;TRACK FIELD 4 (BIT #14)
100000                          TRK20=  100000          ;TRACK FIELD 5 (BIT #15)

                                ;ERROR REGISTER #2 (RHER2) (#10)

000001                          WCU=    1               ;WRITE CURRENT UNSAFE (BIT #0)
000002                          CSF=    2               ;CURRENT SINK FAILURE (BIT #1)
000004                          WSU=    4               ;WRITE SELECT UNSAFE (BIT #2)
000010                          CSU=    10              ;CURRENT SWITCH UNSAFE (BIT #3)
000020                          MSE=    20              ;MOTOR SEQUENCE ERROR (BIT #4)
000040                          TDF=    40              ;TRANSITIONS DETECTOR FAILURE (BIT #5)
000100                          TUF=    100             ;TRANSITIONS UNSAFE (BIT #6)
000200                          FEN=    200             ;FAILSAFE ENABLED (BIT #7)
000400                          WRU=    400             ;WRITE READY UNSAFE (BIT #8)
001000                          MHS=    1000            ;MULTIPLE HEAD SELECT (BIT #9)
002000                          NHS=    2000            ;NO HEAD SELECTION (BIT #10)
004000                          IXE=    4000            ;INDEX ERROR (BIT #11)
010000                          VU30=   10000           ;30VOLT UNSAFE (BIT #12)
020000                          PLU=    20000           ;PLO UNSAFE (BIT #13)
100000                          ACU=    100000          ;ACUNSAFE (BIT #15)

                                ;OFFSET REGISTER (RHOF) (#11)
```

```
5373            000001                  OF25=   1                       ;OFFSET 25 MICRO INCHES (BIT 00)
5374            000002                  OF50=   2                       ;OFFSET 50 MICRO INCHES (BIT 01)
5375            000004                  OF100=  4                       ;OFFSET 100 MICRO INCHES (BIT 02)
5376            000010                  OF200=  10                      ;OFFSET 200 MICRO INCHES (BIT 03)
5377            000020                  OF400=  20                      ;OFFSET 400 MICRO INCHES (BIT 04)
5378            000040                  OF800=  40                      ;OFFSET 800 MICRO INCHES (BIT 05)
5379
5380            000200                  OFREV=  200                     ;OFFSET NEGATIVE (REVERSE) (BIT 07)
5381            002000                  HCI=    2000                    ;HEADER COMPARE INHIBIT (BIT 010)
5382            004000                  ECI=    4000                    ;ERROR CORRECTION CODE INHIBIT (BIT 011)
5383            010000                  FMT22=  10000                   ;FORMAT BIT (BIT 012
5384
5385                                    ;DESIRED CYLINDER ADDRESS (RHCA) (012)
5386                                    ;EACH BIT IS CALLED BY BIT NUMBER.
5387                                    ;CURRENT CYLINDER ADDRESS (RHCC) (013)
5388                                    ;EACH BIT IS CALLED BY BIT NUMBER
5389                                    ;SERIAL NUMBER REGISTER (RHSN) (014)
5390                                    ;EACH IS CALLED BY BIT NUMBER
5391                                    ;ERROR REGISTER #3 (RHER3) (015)
5392
5393            000001                  PSU=    1                       ;PACK SPEED UNSAFE (BIT 00)
5394            000002                  VUF=    2                       ;VELOCITY UNSAFE (BIT 01)
5395            000010                  UNR=    10                      ;ANY UNSAFE EXCEPT READ/WRITE (BIT 03)
5396            000020                  PRE=    20                      ;DISK PACK ROTATION ERROR (BIT 04)
5397            000040                  ACL=    40                      ;AC LOW (BIT 05)
5398            000100                  DCL=    100                     ;DC LOW (BIT 06)
5399            040000                  SKI=    40000                   ;SEEK INCOMPLETE (BIT 014)
5400            100000                  OCYL=   100000                  ;OFF CYLINDER (BIT 015)
5401
5402                                    ;ECC POSITION REGISTER (RHEC1) (016)
5403                                    ;EACH BIT IS CALLED BY BIT NUMBER
5404                                    ;ECC PATTERN REGISTER (RHEC2) (017)
5405                                    ;EACH BIT IS CALLED BY BIT NUMBER
```

```
                                        .SBTTL  REGISTER ADDRESSES
                                        ;RP04 VECTOR ADDRESS
001620  000254                          RPVEC:  254                             ;RP04 VECTOR ADDRESS
                                        ;RP04 DISK I/O REGISTERS LOCATED IN THE RH11 CONTROLLER
                                        ;NOTE:  THE CONTENTS OF THESE LOCATIONS WILL BE DIFFRENT
                                        ;       IF THE "CHANGE BASE ADDRESS" ROUTINE IS USED,
                                        ;       THIS ROUTINE STARTS AT LOCATION TAGED "BASECH"
001622  176722                          RHDB:   176722                          ;DATA BUFFER SEE NOTE ABOVE
001624  176702                          RHWC:   176702                          ;WORD COUNT SEE NOTE ABOVE
001626  176704                          RHBA:   176704                          ;BUS ADDRESS SEE NOTE ABOVE
001630  176710                          RHCS2:  176710                          ;CONTROL AND STATUS 2 SEE NOTE ABOVE
                                        ;RP04 DISK I/O REGISTERS LOCATED IN THE RP04 DEVICE LOGIC
                                        ;NOTE:  THE CONTENTS OF THESE LOCATIONS WILL BE DIFFRENT
                                        ;       IF THE "CHANGE BASE ADDRESS ROUTINE IS USED,
                                        ;       THIS ROUTINE STARTS AT LOCATION TAGED "BASECH"
001632  176700                          RHCS1:  176700                  ;CONTROL AND STATUS 1 SEE NOTE ABOVE
001634  176714                          RHER1:  176714                  ;ERROR #1 SEE NOTE ABOVE
001636  176706                          RHDST:  176706                  ;DESIRED SECTOR/TRACK ADDRESS SEE NOTE ABOVE
001640  176740                          RHER2:  176740                  ;ERROR #2 SEE NOTE ABOVE
001642  176732                          RHOF:   176732                  ;OFFSET SEE NOTE ABOVE
001644  176734                          RHCA:   176734                  ;DESIRED CYLINDER ADDRESS SEE NOTE ABOVE
001646  176742                          RHER3:  176742                  ;ERROR #3 SEE NOTE ABOVE
001650  176716                          RHAS:   176716                  ;ATTENTION SUMMARY SEE NOTE ABOVE
001652  176724                          RHMR:   176724                  ;MAINTAINABILITY SEE NOTE ABOVE
001654  176712                          RHDS1:  176712                  ;DRIVE STATUS SEE NOTE ABOVE
001656  176726                          RHDT:   176726                  ;DRIVE TYPE SEE NOTE ABOVE
001660  176730                          RHSN:   176730                  ;SERIAL NUMBER SEE NOTE ABOVE
001662  176744                          RHEC1:  176744                  ;ECC POSITION SEE NOTE ABOVE
001664  176746                          RHEC2:  176746                  ;ECC PATTERN SEE NOTE ABOVE
001666  176720                          RHLA:   176720                  ;LOOK-AHEAD SEE NOTE ABOVE
001670  176736                          RHCC:   176736                  ;CURRENT CYLINDER ADDRESS SEE NOTE ABOVE
```

```
                                ;THE FOLLOWING LOCATIONS ARE RESERVED FOR REGISTERS
                                ;ANY TIME THERE IS AN ERROR ALL THESE WILL BE FILLED
                                ;ONLY SOME MAY BE PRINTED BUT ALL WILL BE FILLED TRUE
                                ;FOR THE TIME JUST AFTER THE "ERROR" ERROR COMMAND

001672  000000  DB:     0                       ;DATA BUFFER
001674  000000  WC:     0                       ;WORD COUNT
001676  000000  BA:     0                       ;BUS ADDRESS
001700  000000  CS2:    0                       ;CONTROL AND STATUS 2

001702  000000  CS1:    0                       ;CONTROL AND STATUS 1
001704  000000  ER1:    0                       ;ERROR #1
001706  000000  DST:    0                       ;DESIRED SECTOR/TRACK ADDRESS
001710  000000  ER2:    0                       ;ERROR #2
001712  000000  OF:     0                       ;OFFSET
001714  000000  CA:     0                       ;DESIRED CYLINDER ADDRESS
001716  000000  ER3:    0                       ;ERROR #3
001720  000000  AS:     0                       ;ATTENTION SUMMARY
001722  000000  MR:     0                       ;MAINTAINABILITY
001724  000000  DS1:    0                       ;DRIVE STATUS
001726  000000  DT:     0                       ;DRIVE TYPE
001730  000000  SN:     0                       ;SERIAL NUMBER
001732  000000  EC1:    0                       ;ECC POSITION
001734  000000  EC2:    0                       ;ECC PATTERN
001736  000000  LA:     0                       ;LOOK-AHEAD
001740  000000  CC:     0                       ;CURRENT CYLINDER ADDRESS

001742  000010  UNITS:  .BLKW   8.      ;THIS IS FILLED WITH -1
001762  000000  UNIT:   .WORD   0       ;UNIT UNDER TEST
001764  000000  NOUNIT: .WORD   0       ;NUMBER OF UNITS PRESENT
                                        ;USED TO KEEP TRACK OF UNIT UNDER TEST
001766  000000  NUNIT:  .WORD   0       ;USED TO DETERMIN IF THERE ARE MORE
                                        ;THAN ONE UNIT
001770  000000  SELECT: .WORD   0       ;ALL ONES INDICATE UNIT TO BE SELECTED
001772  000000  UNITSL: .WORD   0       ;UNIT NO. SELECTED

001774  000000  ERFLGS: 0                       ;ERROR FLAG

001776  000000  SAVDT:  0               ;SAVE DRIVE TYPE REGISTER
                                        ;FOR COMPARISON IN DRIVE CLEAR TEST
                                        ;AND RH INIT TEST
002000  000000  SAVSN:  0               ;SAVE SERIAL NUMBER REGISTER
                                        ;FOR COMPARISON IN DRIVE CLEAR TEST
                                        ;AND RH INIT TEST
```

```
5515
5516  002002  000000  PCJSR:  0          ;SAVE PC OF JSR WHICH GAVE THE ERROR
5517
5518
5519
5520  002004  000000  ATTENT: 0          ;ATTENTION BIT FOR PRESENT UNIT
5521  002006  000000  TOTALAT:     0                ;TATAL ATTENTION BITS
5522
5523
5524  002010  000000  TMPILL: 0                ;TEMPORARY ILLEGAL FUNCTION
5525
5526
5527  002012  000000  TSECC:  0                ;FLAG TO SAY IF ECC TEST OR NOT
5528                                           ;WHEN =177777 IT IS AN ECC TEST
5529                                           ;WHEN =0IT IS NOT AN ECC TEST
5530
5531  002014  000000  TESDTE: 0          ;FLAG TO SAY IF DRIVE TIMING ERROR OR NOT
5532                                           ;WHEN = 177777 IT IS A DTE TEST
5533                                           ;WHEN = 0      IT IS NOT A DTE TEST
5534
5535
5536  002016  000000  TAGDTE: 0                ;TEMPORARY TAG USED IN DRIVE TIMING
5537                                           ;ERROR TEST
```

```
                                ;FUNCTION EQUATES

                                        ;TABLE OF FUNCTIONS FOR RHCS1 THEN "GO" BIT HAS TO BE SET
002020                          FUTABL:
002020  000000                  NOPERA: 0                       ;NO OPERATION
002022  000002                  UNLOAD: 2                       ;UNLOAD (STAND BY)
002024  000006                  RECALI: 6                       ;RECALIBRATE
002026  000010                  DCLEAR: 10                      ;DRIVE CLEAR
002030  000012                  RELEAS: 12                      ;RELEASE (DUAL-PORT OPERATION)
002032  000030                  SERCH:  30                      ;SEARCH COMMAND
002034  000050                  WRCHEK: 50                      ;WRITE CHECK DATA
002036  000052                  WRCHDT: 52                      ;WRITE CHECK HEADER AND DATA
002040  000060                  WRIDAT: 60                      ;WRITE DATA
002042  000062                  WRIFOR: 62                      ;WRITE HEADER AND DATA (FORMAT)
002044  000070                  READAT: 70                      ;READ DATA
002046  000072                  REFOR:  72                      ;READ HEADER AND DATA
002050  000004                  SEECOM: 4                       ;SEEK COMMAND
002052  000014                  OFSETC: 14                      ;OFFSET COMMAND
002054  000016                  RETCL:  16                      ;RETURNTO CENTERLINE
002056  000022                  PKACK:  22                      ;PACK ACKNOWLEDGE
002060  000020                  READIN: 20                      ;READ IN
002062  000000                  ILLEGL: .WORD                   ;COMPUTED ILLEGAL FUNCTION

                                        ;DATA BUFFER FOR READ WRITE
002064  000422                  WRFROM: .BLKW   274.            ;WRITE FROM THIS BUFFER
003130  000422                  REINTO: .BLKW   274.            ;READ INTO THIS BUFFER
004174  000000                  TSTNM:  0                       ;TEST NUMBER
004176  000000                  FIRST:  0                       ;IF ZERO WILL TYPE HEADER
                                                                ;IF ONES WILL NOT TYPE HEADER


                                ;TABLE FOR ATTENTION BITS
                                ;ATTENTION TABLE
004200  001  002  004           ATABLE: .BYTE   1,2,4,10,20,40,100,200
004203  010  020  040
004206  100  200
```

```
5577                                                    .SBTTL  REGISTER TEST
5578  004210  012737  177777  001770  BEGIN2: MOV     #-1,@#SELECT    ;SELECT UNIT
5579  004216  000402                  BR      START
5580  004220  005037  001770          BEGIN:  CLR     @#SELECT        ;DO NOT SELECT UNIT
5581                                                                  ;NORMAL RUN
5582
5583  004224                          START:
5584  004224  012737  000340  177776          MOV     #340,@#PS       ;;LOCK OUT ALL INTERRUPTS
 (1)  004232  012706  001100                  MOV     #^CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
 (1)  004236  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
 (1)  004240  022706  001136                  CMP     #STKS,R6        ;;DONE?
 (1)  004244  001374                          BNE     .-6             ;;LOOP BACK IF NO
 (1)  004246  012706  001000                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
 (1)  004252  012737  035460  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
 (1)  004260  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
 (1)  004266  012737  037152  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
 (1)  004274  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
 (1)  004302  012737  037716  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
 (1)  004310  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
 (1)  004316  012737  037764  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
 (1)  004324  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
 (1)  004332  005067  174646                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
 (1)  004336  005067  174644                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 (1)  004342  112767  000001  174545          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
 (1)  004350  012767  004350  174530          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
 (1)  004356  012767  004356  174524          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
5588
5589
5590  004364  012767  000000  173404          MOV     #0,PS                   ;SET PROCESSOR STATUS TO 0
5591  004372  012777  035416  178220          MOV     #RPVECT,@RPVEC          ;THIS IS FOR UNTIMELY RP04 INTERRUPTS
5592  004400  004737  036424                  JSR     PC,@#TKINT      ;INITILIZE TK
5593  004404  005737  004176                  TST     @#FIRST                 ;IS THIS FIRST TIME ROUND
5594  004410  001001                          BNE     1$                      ;BRANCH IF NOT
5595  004412  000402                          BR      2$
5596  004414  000137  005232          1$:     JMP     @#END1
5597  004420                          2$:
 (1)  004420  104400  004426                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)  004424  000437                          BR      64$             ;;GET OVER THE ASCII
 (1)                                          ;;.ASCII        <15><12>/RP04 DISKLESS CONTROLLER TEST-PART II (STATIC 1B) - DERPT
 (1)  004524                          64$:
5598  004524  104400  004532                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)  004530  000425                          BR      65$             ;;GET OVER THE ASCII
 (1)                                          ;;.ASCII        <15><12>/MAKE SURE DCL IS LOCKED ON CORRECT PORT/
 (1)  004604                          65$:
5599  004604  104400  004612                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)  004610  000425                          BR      66$             ;;GET OVER THE ASCII
 (1)                                          ;;.ASCII        <15><12>/IF CHANGES ARE REQUIRED ON SWITCH THEN/
 (1)  004664                          66$:
5600  004664  104400  004672                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)  004670  000430                          BR      67$             ;;GET OVER THE ASCII
 (1)                                          ;;.ASCII        <15><12>/A CYCLE UP SEQUENCE IS REQUIRED FOR STROBING/
 (1)  004752                          67$:
5601  004752  104400  004760                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)  004756  000414                          BR      68$             ;;GET OVER THE ASCII
```

```
 (1)                                        ;;,ASCIZ    <15><12>/THE PORT SELECT FLOP/
 (1)  005010                          6881:
5602  005010  104400  005016                TYPE    ,.+4            ;;TYPE ASCIZ STRING
 (1)  005014  000432                      BR      698             ;;GET OVER THE ASCIZ
 (1)                                      ;;,ASCIZ    <15><12>/ALL DCL UNDER TEST MUST BE LOCKED ON CORRECT PORT/
 (1)  005102                          6981:
5603  005102  104400  005110                TYPE    ,.+4            ;;TYPE ASCIZ STRING
 (1)  005106  000427                      BR      708             ;;GET OVER THE ASCIZ
 (1)                                      ;;,ASCIZ    <15><12>/ALL DCL NOT UNDER TEST MUST BE SWITCHED OFF/
 (1)  005166                          7081:
5604  005166  104400  005174                TYPE    ,.+4            ;;TYPE ASCIZ STRING
 (1)  005172  000417                      BR      718             ;;GET OVER THE ASCIZ
 (1)                                      ;;,ASCIZ    <15><12>/OR LOCKED ON THE OTHER PORT/
 (1)  005232                          7181:
5605  005232  012737  177777  004176  SND1:  MOV     #-1,@#FIRST             ;NEXT TIME DO NOT GIVE HEADER
5606  005240  005737  001770                TST     @#SELECT                ;WAS IT A 200 START
5607
 (1)  005244  001435                      BEQ     TST1    ;BRANCH IF STARTING FROM 200
 (1)
5608  005246  104400  005254                TYPE    ,.+4            ;;TYPE ASCIZ STRING
 (1)  005252  000423                      BR      648             ;;GET OVER THE ASCIZ
 (1)                                      ;;,ASCIZ    <15><12>/SELECT UNIT NUMBER TO BE TESTED ?  /
 (1)  005322                          6481:
5609  005322  104416                      RDOCT
5610  005324  042716  177770                BIC     #177770,(SP)            ;ONLY KEEP LAST 3 BITS
5611  005330  011637  001762                MOV     (SP),@#UNIT             ;SAVE UNIT TO BE TESTED
5612  005334  012637  001772                MOV     (SP)+,@#UNITSL          ;SAVE UNIT TO BE TESTED
5613
5614
5615
5619                                   ;*****************************************************************
 (3)                                   ;*TEST 1        REFERENCE EACH REGISTER
 (4)                                   ;*              REFERENCE EACH REGISTER BY A MOVE INSTRUCTION
 (3)                                   ;*****************************************************************
 (2)  005340  000004                  TST1:   SCOPE
 (1)  005342  012767  000001  173634          MOV     #1,#TIMES       ;;DO 1 ITERATION
5620  005350  012706  001000                MOV     #STACK, SP      ;SET UP STACK POINTER
5621
 (1)  005354  012737  000001  004174          MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
 (1)
5622  005362  012737  037160  000030          MOV     #REGSA1,@#EMTVEC;ERROR VECTOR SO THAT
5623                                                                  ;NO REGISTERS ARE SAVED
5624  005370  012737  005416  000004          MOV     #2$,    @#ERRVEC ;SET UP FOR BUS TIMEOUT
5625  005376  012700  000024                MOV     #24,    R0      ;THERE ARE 24 REG TO TEST
5626  005402  012701  001622                MOV     #RHDB,  R1      ;R1 NOW HAS ADDR OF ADDR OF FIRST REG.
5627  005406  013102                  1$:     MOV     @(R1)+, R2      ;READ HARDWARE REG.
5628  005410  005300                          DEC     R0              ;COUNT DOWN
5629  005412  001375                          BNE     1$              ;BRANCH IF 24 NOT DONE
5630  005414  000471                          BR      3$              ;BRANCH IF 24 DONE
5631  005416  012737  000006  000004  2$:     MOV     #ERRVEC+2,@#ERRVEC ;RESTORE TRAP CATCHER
5632  005424  022626                          CMP     (SP)+,  (SP)+   ;CLEAN STACK
5633  005426  016167  177776  173536          MOV     -2(R1), #TMP1   ;STORE FAILING REG ADDR
5634  005434  104015                          ERROR   15              ;REGISTER NON EXISTANT
5635  005436  032737  020000  177570          BIT     #BW13,@#SWR     ;INHIBIT ERROR PRINTOUT ?
```

```
 5636  005444  001053                          BNE     48              ;BRANCH IF YES
 5637  005446  104400  005454                  TYPE    ,.+4            ;;TYPE ASCII STRING
  (1)  005452  000431                          BR      648             ;;GET OVER THE ASCII
  (1)                                          ;;.ASCII        <15><12>/IF BASE ADDRESS IS TO BE CHANGED HALT PROGRAM /
  (1)  005536                          6481:
 5638  005536  104400  005544                  TYPE    ,.+4            ;;TYPE ASCII STRING
  (1)  005542  000411                          BR      658             ;;GET OVER THE ASCII
  (1)                                          ;;.ASCII        <15><12>/AND RESTART AT /
  (1)  005566                          6581:
 5639  005566  012746  030056                  MOV     @BASECH,-(SP)   ;GET READY TO TYPE STARTING ADDRESS
 5640                                          ;OF "CHANGE OF BASE ADDRESS" ROUTINE
 5641  005572  104402                          TYPOC
 5642  005574  000137  023246          48:     JMP     @#EOP           ;GO TO END OF PROGRAM
 5643  005600  012737  037152  000030  38:     MOV     @#ERROR,@#EMTVEC;RESTORE ERROR VECTOR
 5644                                                                  ;SO THAT REGISTERS ARE SAVED
 5645  005606  012737  000006  000004          MOV     @#ERRVEC+2,@#ERRVEC ;RESTORE TRAP CATCHER
 5646
 5647
 5648
 5655
 5656                                  ;*********************************************************************
  (3)                                  ;*TEST 2         RHC82-CONTROL AND STATUS 2
  (4)
  (4)                                  ;*      THIS PARTIALLY TESTS RHC82 TO ENABLE DETERMINATION
  (4)                                  ;*      OF THE NUMBER OF DRIVES PRESENT
  (4)
  (3)                                  ;*********************************************************************
  (2)  005614  000004                  TST2:   SCOPE
  (1)  005616  012767  000001  173360          MOV     #1,@#TIMES      ;;DO 1 ITERATION
 5657  005624  012706  001000                  MOV     @#STACK,SP      ;RESET STACK
 5658
 5659
  (1)  005630  012737  000002  004174          MOV     @#TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
  (1)
 5660  005636  013737  001630  005652          MOV     @#RHCS2,@#UN+2
 5661  005644  004537  024236                  JSR     R5,@#BITST      ;TEST BITS IN REGISTER
 5662  005650  000017                  UN:     17                      ;ONLY THESE BITS ARE TEST READ/WRITE
 5663  005652  000000                          .WORD   0               ;ADDRESS OF REG. BEING TESTED
 5664  005654  104001                          ERROR   1               ;IN CORRECT DATA RECEIVED
 5665  005656  000207                          RTS     PC              ;RETURN TO BLT3 ROUTINE
 5666
 5667                                  ;*********************************************************************
  (3)                                  ;*TEST 3         PARTIAL TEST FOR RHAS FOR UNIT NUMBERS PRESENT
  (3)                                  ;*********************************************************************
  (2)  005660  000004                  TST3:   SCOPE
  (1)  005662  012767  000001  173314          MOV     #1,@#TIMES      ;;DO 1 ITERATION
 5668
  (1)  005670  012737  000003  004174          MOV     @#TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
  (1)
 5669  005676  013701  001650                  MOV     @#RHAS,R1       ;R1 HAS ADDRESS OF RHAS
 5670  005702  012711  177777                  MOV     #-1,@R1 ;THIS CLEARS RHAS (SURPRISED!)
 5671  005706  011137  001126                  MOV     @R1,@#BDDAT     ;TEST DATA
 5672  005712  105737  001126                  TSTB    @#BDDAT
 5673
```

```
  (1)  005716  001405                          BEQ     TST4    ;BRANCH IF GOOD
  (1)
5674   005720  005037  001124                 CLR     @#GDDAT         ;GOOD DATA
5675   005724  010137  024234                 MOV     R1,@#REGADR     ;FAILING REG, RHAS
5676   005730  104001                         ERROR   1               ;RHAS DOES NOT CLEAR
5677                                                                   ;WITH ONES
5678
5679                                ;*********************************************************
 (3)                               ;*TEST 4        TEST FOR DRIVES PRESENT USING RHAS AND RHCS2
 (3)                               ;*********************************************************
 (2)  005732  000004                TST4:   SCOPE
 (1)  005734  012767  000001  173242        MOV     #1,#TIMES       ;;DO 1 ITERATION
5680   005742  000005                        RESET                   ;START WITH AN INIT
5681   005744  004737  036424                JSR     PC,@#TKINT      ;INITILIZE TK
5682   005750  032737  020000  177570        BIT     #SW13,@#SWR     ;INHIBIT ERROR TYPEOUT?
5683   005756  001030                        BNE     4$              ;BRANCH IF YES
5684   005760  104400  005766                TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)  005764  000425                        BR      64$             ;;GET OVER THE ASCII
 (1)                                        ;;,ASCII      <15><12>/LOOKING AT RHAS =  RP04 DRIVES PRESENT /
 (1)  006040                        64$:
5685   006040  013701  001650        4$:     MOV     @#RHAS,R1       ;R1 HAS ADDR. OF RHAS
5686   006044  013702  001630                MOV     @#RHCS2,R2      ;R2 HAS ADDR. OF RHCS2
5687   006050  005012                        CLR     @R2             ;CLEAR RHCS2
5688   006052  012700  000010                MOV     #8.,R0          ;COUNT
5689   006056  013704  001634                MOV     @#RHER1,R4      ;R4 HAS ADDR. OF RHER1
5690   006062  012714  177777        1$:     MOV     #-1,@R4         ;MOVE ERRORS INTO RHER1
5691   006066  005212                        INC     @R2             ;INCREMENT UNIT NO.
5692   006070  005300                        DEC     R0              ;COUNT
5693   006072  001373                        BNE     1$              ;BRANCH IF 8 NOT DONE
5694   006074  111137  002006                MOVB    @R1,@#TOTALAT   ;SAVE TOTAL ATTENTION
5695                                                                  ;USED IN DRIVE CLEAR TEST
5696   006100  105037  002007                CLRB    @#TOTALAT+1     ;CLEAR UPPER BYTE
5697   006104  105711                        TSTB    @R1             ;TEST FOR ANY DRIVES PRESENT
5698   006106  001402                        BEQ     2$              ;IF SOME  NOT THERE BRANCH
5699   006110  000167  000436                JMP     XE2             ;NONE THERE
5700   006114  032737  020000  177570  2$:   BIT     #SW13,@#SWR     ;INHIBIT ERROR TYPE OUT?
5701   006122  001402                        BEQ     3$              ;BRANCH IF NO
5702   006124  000167  000706                JMP     TST5    ;OUT
5703   006130                             3$:
 (1)  006130  104400  006136                TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)  006134  000412                        BR      65$             ;;GET OVER THE ASCII
 (1)                                        ;;,ASCII      <15><12>/NO DRIVES-RHAS=0/
 (1)  006162                        65$:
5704   006162  104400  006170                TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)  006166  000436                        BR      66$             ;;GET OVER THE ASCII
 (1)                                        ;;,ASCII      <15><12>/WRITING ONES INTO ERROR REGISTER #1 FOR ALL UNIT NUMBERS/
 (1)  006264                        66$:
5705   006264  104400  006272                TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)  006270  000441                        BR      67$             ;;GET OVER THE ASCII
 (1)                                        ;;,ASCII      <15><12>/DOES NOT SET ANY BIT IN THE ATTENTION REGISTER SO ABORT P
 (1)  006374                        67$:
5706   006374  104400  006402                TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)  006400  000440                        BR      68$             ;;GET OVER THE ASCII
 (1)                                        ;;,ASCII      <15><12>/TO LOOP ON THIS TEST WITHOUT PRINTOUT SET SWITCHS 13  8 A
```

```
 (1)  006502                           68$:
      006502  000137  023246                        JMP     @#$EOP            ;GO OUT
      006506  104400  006514                        TYPE    ,.+4              ;;TYPE ASCIZ STRING
 (1)  006512  000410                                BR      69$               ;;GET OVER THE ASCIZ
 (1)                                                ;;.ASCIZ       <15><12>/TEST DRIVE 0/
 (1)  006534                           69$:
      006534  005037  001742                        CLR     @#UNITS
      006540  012767  000001  173216                MOV     #1,NOUNIT         ;NO. UNITS PRESENT=1
      006546  005037  001762                        CLR     @#UNIT
      006552                           XE2:
      006552  012700  000010           2$:          MOV     #8.,R0            ;COUNTER
      006556  012703  001742                        MOV     #UNITS,R3         ;POINTER
      006562  012723  177777           3$:          MOV     #-1,(R3)+         ;PRESET BLOCK TO ALL ONES
      006566  005300                                DEC     R0                ;COUNT
      006570  001374                                BNE     3$                ;BRANCH IF 8 NOT DONE
      006572  012703  001742                        MOV     #UNITS,R3         ;POINTER
      006576  005005                                CLR     R5
      006600  005037  001764                        CLR     @#NOUNIT          ;NO. OF UNITS PRESENT
      006604  012700  000010                        MOV     #8.,R0            ;COUNTER
      006610  011137  001170                        MOV     @R1,@#$TMP0       ;TEMPORARY STORAGE
      006614  006037  001170           4$:          ROR     @#$TMP0 ;SET CARRY IF ONE IN  0 BIT

      006620  103065                                BCC     5$
      006622  010577  173002                        MOV     R5,@RHCS2         ;INSERT UNIT NUMBER
      006626  022777  024020  173022                CMP     #24020,@RHDT      ;IS THIS A DUAL PORT RP04
      006634  001450                                BEQ     6$                ;BRANCH IF YES
      006636  022777  020020  173012                CMP     #20020,@RHDT      ;IS THIS A SINGLE PORT RP04
      006644  001444                                BEQ     6$                ;BRANCH IF YES
      006646  104400  006654                        TYPE    ,.+4              ;;TYPE ASCIZ STRING
 (1)  006652  000410                                BR      64$               ;;GET OVER THE ASCIZ
 (1)                                                ;;.ASCIZ       <15><12>/UNIT NUMBER /
 (1)  006674                           64$:
      006674  010546                                MOV     R5,-(SP)          ;GET READY TO TYPE UNIT NUMBER
      006676  104410                                TYPDS
      006700  104400  006706                        TYPE    ,.+4              ;;TYPE ASCIZ STRING
 (1)  006704  000405                                BR      65$               ;;GET OVER THE ASCIZ
 (1)                                                ;;.ASCIZ       /,  RHDT= /
 (1)  006720                           65$:
      006720  017746  172732                        MOV     @RHDT,-(SP)       ;GET READY TO TYPE RHDT
      006724  104402                                TYPOC
      006726  104400  006734                        TYPE    ,.+4              ;;TYPE ASCIZ STRING
 (1)  006732  000410                                BR      66$               ;;GET OVER THE ASCIZ
 (1)                                                ;;.ASCIZ       / ---NOT AN RP04/
 (1)  006754                           66$:
      006754  000407                                BR      5$                ;NO RP04 FOUND SO BRANCH
      006756  010523                   6$:          MOV     R5,(R3)+
      006760  104400  001215                        TYPE    ,$CRLF
      006764  010546                                MOV     R5,-(SP)
      006766  104410                                TYPDS                             ;TYPE DRIVE NO.
      006770  005237  001764                        INC     @#NOUNIT
      006774  005205                   5$:          INC     R5
      006776  005300                                DEC     R0
      007000  001305                                BNE     4$
      007002  013737  001742  001762                MOV     @#UNITS,@#UNIT
```

D5

```
                                                MOV     @#NOUNIT,@#NUNIT        ;SAVE NO. OF UNITS
                                                DEC     @#NUNIT                 ;IF NUNIT = 0 THEN ONLY ONE UNIT
                                                                                ;IF NUNIT MORE THAN 0 THEN MORE THAN ONE UNIT
                                                TST     @#SELECT                ;STARTING ADDRESS 200 ?
                                                BEQ     TST5    ;BRANCH IF STARTING FROM 200
                                                MOV     @#UNITSL,@#UNIT         ;SET UNIT NUMBER

                                        ;*******************************************************************
                                        ;*TEST 5        TEST SERIAL NUMBER AND DRIVE TYPE!
                                        ;*              READ SERIAL NUMBER REGISTER AND DRIVE TYPE REGISTER
                                        ;*              TYPE IT OUT AND PROCEED
                                        ;*              TO LOOP HERE SET SWITCH 0 AND THIS TEST NO AND RESTART
                                        ;*******************************************************************
007036  000004                        TST5:   SCOPE
007040  012767  000001  172136                MOV     #1,#TIMES       ;;DO 1 ITERATION
007046  012767  007250  172032                MOV     #16,#LPADR      ;;SET SCOPE LOOP ADDRESS
007054  004737  024470                        JSR     PC,@#CLDISK             ;FILL UNIT NO.
007060  005037  002004                        CLR     @#ATTENT                ;CLEAR
007064  013700  001762                        MOV     @#UNIT,R0               ;R0 CONTAINS UNIT NO
007070  116037  004200  002004                MOVB    ATABLE(R0),@#ATTENT     ;SET APPROPIATE ATTENTION BIT
007076  104408  007104                        TYPE    ,.+4                    ;;TYPE ASCII STRING
007102  000415                                BR      648                     ;;GET OVER THE ASCII
                                                ;;.ASCII        <15><12>/TESTING DRIVE NUMBER  /
007136                                  648:
007136  013746  001762                        MOV     @#UNIT,-(SP)            ;UNIT NO. TO STACK
007142  104402                                TYPOC                           ;TYPE  DRIVE NO.
007144  104408  001215                        TYPE    ,#CRLF
007150  104408  007156                        TYPE    ,.+4                    ;;TYPE ASCII STRING
007154  000410                                BR      658                     ;;GET OVER THE ASCII
                                                ;;.ASCII        <15><12>/SERIAL NO. = /
007176                                  658:
007176  017746  172456                        MOV     @RHSN,-(SP)             ;;SAVE @RHSN FOR TYPEOUT
007202  104402                                TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007204  104408  001215                        TYPE    ,#CRLF
007210  104408  007216                        TYPE    ,.+4                    ;;TYPE ASCII STRING
007214  000410                                BR      668                     ;;GET OVER THE ASCII
                                                ;;.ASCII        <15><12>/DRIVE TYPE = /
007236                                  668:
007236  017746  172414                        MOV     @RHDT,-(SP)             ;;SAVE @RHDT FOR TYPEOUT
007242  104402                                TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007244  104408  001215                        TYPE    ,#CRLF
007250  005777  172404                1$:     TST     @RHSN                   ;READ SERIAL NO. AND DRIVE TYPE
007254  005777  172376                        TST     @RHDT                   ;THESE TWO ARE TO HELP #SCOPE LOOPS
007260  017737  172374  002000                MOV     @RHSN,@#SAVSN   ;SAVE TO CHECK IF CLR RHC82 BIT 5 CLEARS ANY BITS
007266  017737  172364  001776                MOV     @RHDT,@#SAVDT   ;SAVE TO CHECK IF CLR RHC82 BIT 5 CLEARS ANY BITS


                                        ;*******************************************************************
                                        ;*TEST 6        CHECK MOL TO BE LOW
```

```
 (4)                                    ;*      MAKE SURE THAT DRIVE IS OFF LINE BEFORE STARTING PROGRAM
 (4)                                    ;*      IF DRIVE IS ON LINE THEN AFTER TYPE OUT THE PROGRAM WILL
 (4)                                    ;*      HANG FOR EVER WAITING FOR DRIVE TO GO OFF LINE
 (4)
 (3)                                    ;*********************************************************************
 (2)  007274  000004                    TST6:   SCOPE
5791
 (1)  007276  012737  000006  004174            MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
 (1)
5792  007304  004737  024470                    JSR     PC,@#CLDISK     ;GIVE INITILIZE
5793  007310  032713  010000                    BIT     #MOL,@R3        ;CHECK MOL IN RHDS1
5794
 (1)  007314  001550                            BEQ     TST7    ;BRANCH IF MOL LOW
 (1)
5795  007316  104400  007324                    TYPE    ,.+4            ;;TYPE ASCIZ STRING
 (1)  007322  000421                            BR      648             ;;GET OVER THE ASCIZ
 (1)                                            ;;.ASCIZ        <15><12>/DRIVE IS ON LINE - MOL IS HIGH/
 (1)  007366                            648:
5796  007366  104400  007374                    TYPE    ,.+4            ;;TYPE ASCIZ STRING
 (1)  007372  000424                            BR      658             ;;GET OVER THE ASCIZ
 (1)                                            ;;.ASCIZ        <15><12>/HIT STOP ON DRIVE TO GET IT OFF LINE/
 (1)  007444                            658:
5797  007444  104400  007452                    TYPE    ,.+4            ;;TYPE ASCIZ STRING
 (1)  007450  000438                            BR      668             ;;GET OVER THE ASCIZ
 (1)                                            ;;.ASCIZ        <15><12>/PROGRAM WILL HANG TESTING MOL TILL MOL IS LOW/
 (1)  007532                            668:
5798  007532  032713  010000            18:     BIT     #MOL,@R3        ;CHECK MOL IN RHDS1
5799  007536  001375                            BNE     18              ;BRANCH IF MOL IS HIGH
5800  007540  104400  007546                    TYPE    ,.+4            ;;TYPE ASCIZ STRING
 (1)  007544  000434                            BR      678             ;;GET OVER THE ASCIZ
 (1)                                            ;;.ASCIZ        <15><12>/GOOD - MOL IS NOW LOW , PROGRAM WILL NOW BE EXECUTED/
 (1)  007636                            678:
5801
5802
5803
5812
5813                                    ;*********************************************************************
 (3)                                    ;*TEST 7         PACK ACKNOWLEDGE COMMAND TEST
 (4)
 (4)                                    ;*      THE PACK ACKNOWLEDGE COMMAND WILL BE LOADED INTO RHCS1 WITH GO
 (4)                                    ;*      THEN ALL REGISTERS WILL BE CHECKED
 (4)                                    ;*      RH CLEAR WILL BE GIVEN
 (4)                                    ;*      THEN ALL REGISTERS WILL BE CHECKED
 (4)
 (3)                                    ;*********************************************************************
 (2)  007636  000004                    TST7:   SCOPE
5814  007640  012706  001000                    MOV     #STACK,SP       ;RESET STACK
5815
 (1)  007644  012737  000007  004174            MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
 (1)
5816
5817  007652  004737  024470                    JSR     PC,@#CLDISK     ;INIT AND SET UP GENERAL REG,
5818                                                                    ;AND UNIT NUMBER
5819  007656  012777  000001  171766            MOV     #DMD,@RHMR      ;SET DIAGNOSTIC MODE
```

```
                                ;SAVE REGISTERS FOR COMPARISON AFTER GO
007672  004037  025146          JSR     R0,@#SAVER      ;SAVE
007676  001624                  RHWC                    ;FROM
007700  003130                  REINTO                  ;TO
007702  000023                  19.                     ;NUMBER OF REGISTERS SAVED

                                ;GIVE GO TO PACK ACKNOWLEDGE COMMAND
007704  052777  000001  171720  BIS     #GO,@RHCS1      ;GO TO PACK ACKNOWLEDGE COMMAND

                                ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
007712  052737  000100  003160  BIS     #VV,@#REINTO+30 ;SAVED RHDS1

                                ;AFTER GO HAS BEEN GIVEN TO PACK ACKNOWLEDGE COMMAND
                                ;SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
                                ;BE DONE
007720  004037  025146          JSR     R0,@#SAVER      ;SAVE
007724  001624                  RHWC                    ;FROM
007726  002064                  WRFROM
007730  000023                  19.                     ;NUMBER OF REGISTERS SAVED

                                ;AS UPPER BYTE OF RHAS CAN BE CHANGING IN A DUAL PORT
                                ;OPERATION THE UPPER BYTE OF RHAS WILL BE SAVED AS IS
                                ;SO THAT THE COMPARES ARE ONLY VALID FOR THE LOWER BYTE
007732  113737  003155  002111  MOVB    @#REINTO+25,@#WRFROM+25;SAVE UPPER RHAS

                                ;COMPARE REGISTERS BEFORE PACK ACKNOWLEDGE COMMAND
                                ;WITH AFTER GO
007740  004037  025342          JSR     R0,@#COMPAR     ;COMPARE
007744  003130                  REINTO                  ;GOOD BUFFER
007746  002064                  WRFROM                  ;TEST BUFFER
007750  000023                  19.                     ;NUMBER
007752  007760                  1$                      ;RETURN FOR ERROR
007754  007760                  1$                      ;SAME
007756  010000                  2$                      ;RETURN FOR GOOD COMPARISON
007760  013705  031136  1$:     MOV     @#ERWORD,R5     ;GETTING READY TO INDEX
007764  060505                  ADD     R5,R5           ;DOUBLE ERROR WORD
007766  016537  001622  024234  MOV     RHWC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS

007774  104001                  ERROR   1               ;IMPROPER REGISTER CHANGE
                                                        ;AFTER PACK ACKNOWLEDGE COMMAND
                                                        ;WITH GO IS GIVEN
007776  000207                  RTS     PC              ;RETURN TO COMPARISION

010000                  2$:

                                ;*****************************************************************
 (4)                            ;*TEST 10       MAKE CURRENT CYLINDER = 0
 (4)                            ;*****************************************************************
 (3)  010000  000004  TST10:    SCOPE
 (1)  010002  012706  001000    MOV     #STACK,SP       ;RESET STACK
```

```
   (1)  010006  004737  024470                    JSR PC,@#CLDISK          ;INIT DRIVE
   (1)  010012  012777  000001  171632            MOV     #DMD,@RHMR               ;SET DIAGNOSTIC MODE
   (1)  010020  004037  027004                    JSR     R0,@#MAKECYL             ;SUBROUTINE TO GIVE A SEEK
   (1)                                                                             ;COMMAND FOLOWED BY A INIT
   (1)                                                                             ;THIS SHUOLD CHANGE RHCC
   (1)  010024  000000                            0                                ;CHANGE RHCC TO 0
   (1)
  5878                                    ;*****************************************************************
   (3)                                    ;*TEST 11       CONTROL AND STATUS REGISTER 1 BITS 8 AND 9
   (4)
   (4)                                    ;*      WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
   (4)                                    ;*      TRACK 0, SECTOR 0, KEYS 0, NUMBER OF WORDS 256
   (4)                                    ;*      DATA IS THE CONTENTS OF THE TTY READER STATUS REGISTER
   (4)                                    ;*      THIS WILL USE BITS A16 AND A17 WHEN THERE IS MORE THAN 28K OF MEMORY
   (4)
   (3)                                    ;*****************************************************************
   (2)  010026  000004                    TST11:  SCOPE
  5879  010030  000167  000310                    JMP     TST12   ;ON RH70 JUMP TO NEXT TEST
  5880  010034  012706  001000                    MOV     #STACK,SP       ;RESET STACK
  5881
  5882
   (1)  010040  012737  000011  004174            MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
   (1)
  5883  010046  004037  024406                    JSR     R0,@#CLAREA     ;CLEAR SIMULATED DISK
  5884  010052  032734                            .WORD   DISK            ;FROM
  5885  010054  033760                            .WORD   TOLGAP+16       ;TO
  5886  010056  000000                            .WORD   0               ;DATA
  5887                                    ;THESE ARE SETUP FOR DISKLESS USE ONLY
  5888  010060  012737  010000  031016            MOV     #FMT22,@#CYL;CYLINDER 0
  5889                                                                            ;16 BITS PER WORD
  5890  010066  005037  031020                    CLR     @#SECTR         ;SECTOR 0 TRACK 0
  5891  010072  005037  031022                    CLR     @#KEY1          ;KEY1 0
  5892  010076  005037  031024                    CLR     @#KEY2          ;KEY2 0
  5893  010102  012737  000400  031064            MOV     #256.,@#NOWORD  ;NO OF DATA WORDS
  5894  010110  012737  000001  031026            MOV     #1,@#X          ;WRITE DATA
  5895  010116  004537  025646                    JSR     R5,@#CRC        ;GO TO CALCULATE CRC
  5896  010122  031016                            CYL
  5897  010124  032716                            WCRC
  5898
  5899
  5900                                    ;THESE ARE REGULAR SETUPS
  5901
  5902
  5903  010126  004737  024470                    JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
  5904  010132  012777  177400  171464            MOV     #-256.,@RHWC    ;256 DATA WORDS
  5905  010140  013777  001136  171460            MOV     @#STKS,@RHBA    ;STARTING ADDRESS OF WRITE BUFFER
  5906  010146  017737  170764  002010            MOV     @#STKS,@#TMPILL ;TEMPORARY STORAGE OF DATA
  5907  010154  005077  171456                    CLR     @RHDST  ;SECTOR 0 TRACK 0
  5908  010160  012777  010000  171454            MOV     #FMT22,@RHOF    ;16 BITS PER WORD FORMAT
  5909  010166  005077  171452                    CLR     @RHCA   ;CYLINDER 0
  5910  010172  004737  024524                    JSR     PC,@#CHECKT     ;CHECK FOR DVA,RDY,DPR,DRY
  5911  010176  013746  002040                    MOV     @#WRIDAT,-(SP)  ;WRITE DATA=60
  5912  010202  052716  001400                    BIS     #A16!A17,(SP)   ;SET HIGH ORDER UNIBUS BITS
  5913  010206  012611                            MOV     (SP)+,@R1       ;FILL RHCS1
```

```
5914  010210  052777  000010  171412          BIS     #BAI,@RHCS2     ;SET BUS ADDRESS INHIBIT
5915  010216  005037  001774                  CLR     @#ERFLG#        ;CLEAR ERROR FLAG
5916  010222  004737  030706                  JSR     PC,@#COMHD      ;WRITE DATA
5917                                  ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5918                                  ;FROM THE "COMHD" ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
5919                                  ;HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY
5920                                  ;AND SYNCS WERE CORRECTLY DETECTED
5921                                  ;DATA IS TO BE CHECKED
5922  010226  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
5923  010232  005737  001774                  TST     @#ERFLG#                ;HAS ANY ERRORS OCCURED?
5924
 (1)  010236  001042                          BNE     TST12   ;;BRANCH IF YES
 (1)
5925  010240  013700  002010                  MOV     @#TMPILL,R0             ;GOOD DATA
5926  010244  012701  032734                  MOV     #DISK,R1        ;DATA WRITTEN INTO "DISK"
5927  010250  012702  000400                  MOV     #256.,R2                ;COUNTER
5928  010254  012737  000401  031136  18$:    MOV     #257.,@#ERWORD  ;FOR ERROR WORD
5929  010262  020021                          CMP     R0,(R1)+        ;COMPARE GOOD DATA WITH DATA ON DISK
5930  010264  001425                          BEQ     3$              ;BRANCH IF GOOD
5931  010266  013737  002010  001124          MOV     @#TMPILL,@#GDDAT        ;GOOD DATA
5932  010274  014137  001126                  MOV     -(R1),@#BDDAT   ;BAD DATA
5933  010300  160237  031136                  SUB     R2,@#ERWORD     ;ERROR WORD NO
5934  010304  005737  001774                  TST     @#ERFLG#                ;ANY ERRORS ALREADY THERE?
5935  010310  001002                          BNE     2$              ;BRANCH IF YES
5936  010312  104037                          ERROR   37              ;ERROR ON WRITE DATA COMMAND
5937                                                                  ;SEE NEXT ERROR COMMENTS
5938  010314  000401                          BR      64$             ;BRANCH TO AVOID PRINTING NEXT ERROR
5939  010316  104040                  2$:     ERROR   40              ;WORD NO GIVES WORD IN ERROR
5940                                                                  ;ERROR OCCURED WHILE WRITING
5941                                                                  ;WITH A16 A17 OF RHCS1 SET
5942  010320  005721                  64$:    TST     (R1)+           ;UNDO -(R1) FOR BAD DATA
5943  010322  013746  177570                  MOV     @#SWR,-(SP)     ;GET SWITCH SETTING
5944  010326  042716  177177                  BIC     #177177,(SP)    ;KEEP ONLY SWITCH 7 AND 0
5945  010332  022726  000200                  CMP     #SW07,(SP)+     ;IS 7 SET AND 0 RESET
5946
 (1)  010336  001402                          BEQ     TST12   ;BRANCH OUT IF YES
 (1)
5947  010340  005302                  3$:     DEC     R2              ;IF NOT COUNT 256 WORDS
5948  010342  001344                          BNE     1$              ;BRANCH IF 256 NOT DONE
5949
5950
5951
5952
5953
5954
5955
5956
5968
5969                                  ;*****************************************************************
 (3)                                  ;*TEST 12       DRIVE TIMING ERROR
 (4)
 (4)                                  ;*      A READ HEADER AND DATA IS STARTED ON CYLINDER 0, SECTOR
 (4)                                  ;*      0, TRACK 0, 260 WORDS.  AFTER THE HEADER IS READ IN CORRECTLY
 (4)                                  ;*      THEN NO SYNC BYTE (DATA SYNC) IS GIVEN.
```

```
 (4)                                    ;*      THEN NORMAL DIAGNOSTIC CLOCKS AND DIAGNOSTIC
 (4)                                    ;*      SECTOR CLOCKS ARE GIVEN FOR 24 BYTES,
 (4)                                    ;*      THEN 536 BYTES OF SECTOR CLOCKS ONLY ARE GIVEN,
 (4)                                    ;*      THIS IS TO TO BRING SECTOR PULSE UP
 (4)                                    ;*      THIS SHOULD SET DRIVE TIMING ERROR
 (3)                                    ;*****************************************************************
 (2) 010344  000004                     TST12:  SCOPE
5970 010346  012706  001000                     MOV     #STACK,SP       ;RESET STACK
5971
 (1) 010352  012737  000012  004174             MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
 (1)
5972
5973
5974                                            ;THESE ARE TO SETUP FOR DISKLESS USE
5975 010360  012737  010000  031016             MOV     #FMT22,@#CYL    ;16 BITS PER WORD
5976                                                                    ;CYLINDER 0
5977 010366  005037  031020                     CLR     @#SECTR         ;SECTOR 0
5978                                                                    ;TRACK 0
5979 010372  005037  031022                     CLR     @#KEY1          ;KEY1 = 0
5980 010376  005037  031024                     CLR     @#KEY2          ;KEY2 = 0
5981 010402  005037  031026                     CLR     @#X             ;THIS IS A READ COMMAND
5982 010406  004537  025646                     JSR     R5,@#CRC        ;GO TO CALCULATE CRC
5983 010412  031016                             CYL
5984 010414  032716                             WCRC
5985
5986
5987                                            ; THESE ARE REGULAR SETUPS
5988 010416  004737  024470                     JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
5989 010422  012777  177374  171174             MOV     #-260.,@RHWC    ;256 DATA WORDS +HEADER
5990 010430  012777  003130  171170             MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF BUFFER
5991 010436  005077  171174                     CLR     @RHDST          ;TRACK = 0
5992                                                                    ;SECTOR = 0
5993 010442  012777  014000  171172             MOV     #FMT22!ECI,@RHOF ;16 BITS PER WORD
5994                                                                    ;ECC CORRECTION INHIBITED
5995 010450  005077  171170                     CLR     @RHCA           ;CYLINDER = 0
5996 010454  013711  002046                     MOV     @#REFOR,@R1     ;READ HEADER AND DATA = 72
5997
5998                                            ;SAVE REGISTERS FOR COMPARISON AFTER SIMULATED DRIVE TIMING ERROR
5999 010460  004037  025146                     JSR     R0,@#SAVER      ;SAVE
6000 010464  001624                             RHWC                    ;FROM
6001 010466  001674                             WC                      ;TO
6002 010470  000023                             19.                     ;NUMBER
6003
6004                                            ;NOW GO WILL BE GIVEN, EVERYTHING WILL BE TREATED
6005                                            ;NORMALLY FOR THE HEADER, BUT WHEN IT IS TIME FOR
6006                                            ;DATA ONLY, SECTOR CLOCKS WILL BE GIVEN, NO DIAGNOSTIC
6007                                            ;CLOCKS WILL BE GIVEN, THIS SHOULD BRING SECTOR PULSE HIGH
6008                                            ;WITHOUT PUTTING READ DOWN HENCE DTE WILL COME UP.
6009 010472  012737  177777  002014             MOV     #-1,@#TESDTE    ;SET DTE TEST
6010 010500  012737  177777  001774             MOV     #-1,@#ERFLG     ;THIS WILL BRING THE READ HEADER
6011                                                                    ;AND DATA PROCESS OUT AFTER THE
6012                                                                    ;HEADER HAS BEEN CORRECTLY READ
6013
6014 010506  004737  030706                     JSR     PC,@#COMHD      ;READ HEADER AND DATA
```

```
                                        ;NOW THE HEADER HAS BEEN READ
                                        ;NOW 560 SECTOR CLOCKS WILL BE GIVEN
                                        ;GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
                                        ;GAP 2 BYTES, TOLERANCE 20 BYTES, EXTRA 2

                                        ;THESE 560 SECTOR CLOCKS ARE DIVIDED INTO TWO GROUPS
                                        ;24 SECTOR CLOCKS WITH NORMAL DIAGNOSTIC CLOCKS
                                        ;AND 536 SECTOR CLOCKS WITHOUT ANY DIAGNOSTIC CLOCKS



                                        ;THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC CLOCKS
010512  012701  000030                  MOV     #24.,R1 ;COUNTER.
010516  013700  001652                  MOV     @#RHMR,R0       ;GET RHMR ADDRESS
010522  012710  000001                  MOV     #DMD,@R0        ;SET DIAGNOSTIC MODE
010526  052710  000012          18$:    BIS     #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND CLOCK
010532  042710  000012                  BIC     #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND CLOCK
010536  012702  000007                  MOV     #7,R2           ;COUNTER FOR DIAGNOSTIC CLOCKS
010542  052710  000002          48$:    BIS     #MCLK,@R0       ;SET CLOCK
010546  042710  000002                  BIC     #MCLK,@R0       ;CLEAR CLOCK
010552  005302                          DEC     R2              ;COUNT TO 7
010554  001372                          BNE     4$              ;BRANCH IF 7 NOT DONE
010556  005301                          DEC     R1              ;COUNT
010560  001362                          BNE     1$              ;BRANCH IF 24 NOT DONE

                                        ;THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC CLOCKS
010562  012701  001030                  MOV     #536.,R1        ;COUNTER.
010566  052710  000010          58$:    BIS     #MSTCK,@R0      ;SET SECTOR CLOCK
010572  042710  000010                  BIC     #MSTCK,@R0      ;CLEAR SECTOR CLOCK
010576  005301                          DEC     R1              ;COUNT
010600  001372                          BNE     5$              ;BRANCH IF 536 NOT DONE



                                        ;NOW DTE SHOULD BE SET CHANGE SAVED REGISTERS TO EXPECTED VALUE
010602  012737  177400  001674          MOV     #-256.,@#WC ;SAVED RHWC
010610  012737  003140  001676          MOV     #REINTO+<4.*2>,@#BA ;SAVED RHBA
010616  052737  140000  001702          BIS     #SC!TRE,@#CS1           ;SAVED RHCS1
010624  052737  010000  001704          BIS     #DTE,@#ER1      ;SAVED RHER1
010632  012737  000401  001722          MOV     #401,@#MR       ;SAVED RHMR
010640  052737  140000  001724          BIS     #ATA!ERR,@#DS1 ;SAVED RHDS1
010646  012737  000100  001736          MOV     #100,@#LA       ;SAVED RHLA
010654  012737  000001  001706          MOV     #1,@#DST        ;SAVED RHDST
010662  013737  002004  001720          MOV     @#ATTENT,@#AS   ;SAVED RHAS


                                        ;NOW SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN BE DONE
010670  004037  025146                  JSR     R0,     @#SAVER ;SAVE
010674  001624                          RHWC                    ;FROM
010676  002064                          WRFROM                  ;TO
010700  000023                          19.                     ;NUMBER
```

K5

```
                                        ;FOR RHAS UPPER BYTE
010702  113737  001721  002111        MOVB    @#AS+1,@#WRFROM+25 ;UPPER RHAS

                                        ;COMPARE HEADER READ
010710  004037  025342                JSR     R0,@#COMPAR     ;COMPARE
010714  031016                        CYL                     ;GOOD BUFFER
010716  003130                        REINTO                  ;TEST BUFFER
010720  000004                        4.                      ;NUMBER
010722  010730                        6$                      ;RETURN FOR ERROR
010724  010730                        6$                      ;SAME
010726  010734                        7$                      ;RETURN FOR GOOD COMPARISON
010730  104010                6$:     ERROR   10              ;HEADER READ IN DURING THIS TEST IS
                                                              ;IN ERROR

010732  000207                        RTS     PC              ;RETURN

010734                        7$:                             ;GOOD


                                        ;COMPARE REGISTERS BEFORE COMMAND WITH AFTER COMMAND
010734  004037  025342                JSR     R0,@#COMPAR     ;COMPARE
010740  001674                        WC                      ;GOOD BUFFER
010742  002064                        WRFROM                  ;TEST BUFFER
010744  000022                        18.                     ;NUMBER
010746  010754                        2$                      ;RETURN FOR ERROR
010750  010754                        2$                      ;SAME
010752  010774                        3$                      ;RETURN FOR GOOD COMPARISON

010754  013705  031136        2$:     MOV     @#ERWORD,R5     ;GETTING READY TO INDEX
010760  060505                        ADD     R5,R5           ;DOUBLE ERROR WORD
010762  016537  001622  024234        MOV     RHWC-2(R5),@#REGADR ;FAILING REGISTER
010770  104001                        ERROR   1               ;IMPROPER REGISTER
                                                              ;CHANGE WHILE EXPECTING
                                                              ;DTE ERROR
010772  000207                        RTS     PC              ;RETURN

010774                        3$:                             ;GOOD


                                ;*****************************************************************
                                ;*TEST 13       DRIVE TIMING ERROR

                                ;*      A WRITE DATA COMMAND IS STARTED ON CYLINDER 0, SECTOR
                                ;*      0, TRACK 0, 256 WORDS.   AFTER THE HEADER IS READ IN CORRECTLY
                                ;*      THEN NO SYNC BYTE (DATA SYNC) IS GIVEN,
                                ;*      THEN NORMAL DIAGNOSTIC CLOCKS AND DIAGNOSTIC
                                ;*      SECTOR CLOCKS ARE GIVEN FOR 24 BYTES,
                                ;*      THEN 536 BYTES OF SECTOR CLOCKS ONLY ARE GIVEN,
                                ;*      THIS IS TO TO BRING SECTOR PULSE UP
                                ;*      THIS SHOULD SET DRIVE TIMING ERROR
                                ;*****************************************************************
010774  000004                T$T13:  SCOPE
010776  012706  001000                MOV     @#STACK,SP      ;RESET STACK
```

```
 6122
  (1)  011002  012737  000013  004174          MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
  (1)
 6123
 6124
 6125                                           ;THESE ARE TO SETUP FOR DISKLESS USE
 6126  011010  012737  010000  031016          MOV     #FMT22,@#CYL    ;16 BITS PER WORD
 6127                                                                  ;CYLINDER 0
 6128  011016  005037  031020                  CLR     @#SECTR         ;SECTOR 0
 6129                                                                  ;TRACK 0
 6130  011022  005037  031022                  CLR     @#KEY1          ;KEY1 = 0
 6131  011026  005037  031024                  CLR     @#KEY2          ;KEY2 = 0
 6132  011032  012737  177777  031026          MOV     #-1,@#X         ;THIS IS A WRITE DATA COMMAND
 6133  011040  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
 6134  011044  031016                          CYL
 6135  011046  032716                          WCRC
 6136
 6137
 6138                                           ; THESE ARE REGULAR SETUPS
 6139  011050  004737  024470                  JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
 6140  011054  012777  177400  170542          MOV     #-256.,@RHWC    ;256 DATA WORDS
 6141  011062  012777  002064  170536          MOV     #WRFROM,@RHBA   ;STARTING ADDRESS OF BUFFER
 6142  011070  005077  170542                  CLR     @RHDST          ;TRACK = 0
 6143                                                                  ;SECTOR = 0
 6144  011074  012777  010000  170548          MOV     #FMT22,@RHOF ;16 BITS PER WORD
 6145                                                                  ;ECC CORRECTION INHIBITED
 6146  011102  005077  170536                  CLR     @RHCA           ;CYLINDER = 0
 6147  011106  013711  002040                  MOV     @#WRIDAT,@R1    ;WRITE DATA = 60
 6148
 6149                                           ;SAVE REGISTERS FOR COMPARISON AFTER SIMULATED DRIVE TIMING ERROR
 6150  011112  004037  025146                  JSR     R0,@#SAVER      ;SAVE
 6151  011116  001624                          RHWC                    ;FROM
 6152  011120  001674                          WC                      ;TO
 6153  011122  000023                          19.                     ;NUMBER
 6154
 6155                                           ;NOW GO WILL BE GIVEN, EVERYTHING WILL BE TREATED
 6156                                           ;NORMALLY FOR THE HEADER, BUT WHEN IT IS TIME FOR
 6157                                           ;DATA ONLY, SECTOR CLOCKS WILL BE GIVEN, NO DIAGNOSTIC
 6158                                           ;CLOCKS WILL BE GIVEN, THIS SHOULD BRING SECTOR PULSE HIGH
 6159                                           ;WITHOUT PUTTING READ DOWN HENCE DTE WILL COME UP.
 6160  011124  012737  177777  002014          MOV     #-1,@#TESDTE    ;SET DTE TEST
 6161  011132  012737  177777  001774          MOV     #-1,@#ERFLGS    ;THIS WILL BRING THE READ HEADER
 6162                                                                  ;AND DATA PROCESS OUT AFTER THE
 6163                                                                  ;HEADER HAS BEEN CORRECTLY READ
 6164
 6165  011140  004737  030706                  JSR     PC,@#COMHD      ;READ HEADER AND DATA
 6166
 6167                                           ;NOW THE HEADER HAS BEEN READ
 6168                                           ;NOW 560 SECTOR CLOCKS WILL BE GIVEN
 6169                                           ;GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
 6170                                           ;GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 2
 6171
 6172                                           ;THESE 560 SECTOR CLOCKS ARE DIVIDED INTO TWO GROUPS
 6173                                           ;24 SECTOR CLOCKS WITH NORMAL DIAGNOSTIC CLOCKS
```

```
                                                ;AND 536 SECTOR CLOCKS WITHOUT ANY DIAGNOSTIC CLOCKS



                                                ;THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC CLOCKS
011144  012701  000030                          MOV     #24.,R1 ;COUNTER.
011150  013700  001652                          MOV     @#RHMR,R0       ;GER RHMR ADDRESS
011154  012710  000001                          MOV     #DMD,@R0        ;SET DIAGNOSTIC MODE
011160  052710  000012          18:             BIS     #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND CLOCK
011164  042710  000012                          BIC     #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND CLOCK
011170  012702  000007                          MOV     #7,R2           ;COUNTER FOR DIAGNOSTIC CLOCKS
011174  052710  000002          48:             BIS     #MCLK,@R0       ;SET CLOCK
011200  042710  000002                          BIC     #MCLK,@R0       ;CLEAR CLOCK
011204  005302                                  DEC     R2              ;CUOUNT TO 7
011206  001372                                  BNE     48              ;BRANCH IF 7 NOT DONE
011210  005301                                  DEC     R1              ;COUNT
011212  001362                                  BNE     18              ;BRANCH IF 24 NOT DONE

                                                ;THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC CLOCKS
011214  012701  001030                          MOV     #536.,R1        ;COUNTER.
011220  052710  000010          58:             BIS     #MSTCK,@R0      ;SET SECTOR CLOCK
011224  042710  000010                          BIC     #MSTCK,@R0      ;CLEAR SECTOR CLOCK
011230  005301                                  DEC     R1              ;COUNT
011232  001372                                  BNE     58              ;BRANCH IF 536 NOT DONE



                                                ;ECC PATTERN REGISTER IS NOT CHECKED
011234  017737  170424  001734                  MOV     @RHEC2,@#EC2    ;RHEC2 IS NOT CHECKED


                                                ;NOW DTE SHOULD BE SET CHANGE SAVED REGISTERS TO EXPECTED VALUE
011242  012737  177416  001674                  MOV     #-242.,@#WC ;SAVED RHWC
011250  012737  002120  001676                  MOV     #WRFROM+<14.*2>,@#BA ;SAVED RHBA
011256  052737  140000  001702                  BIS     #SC!TRE,@#CS1           ;SAVED RHCS1
011264  052737  000300  001700                  BIS     #OR!IR,@#CS2    ;SAVED RHCS2
011272  052737  010000  001704                  BIS     #DTE,@#ER1      ;SAVED RHER1
011300  012737  000201  001722                  MOV     #DENVL!DMD,@#MR ;SAVED RHMR
011306  052737  140000  001724                  BIS     #ATA!ERR,@#DS1 ;SAVED RHDS1
011314  012737  000100  001736                  MOV     #100,@#LA       ;SAVED RHLA
011322  012737  000001  001706                  MOV     #1,@#DST        ;SAVED RHDST
011330  013737  002004  001720                  MOV     @#ATTENT,@#AS   ;SAVED RHAS


                                                ;NOW SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN BE DONE
011336  004037  025146                          JSR     R0,     @#SAVER ;SAVE
011342  001624                                  RHWC                    ;FROM
011344  003130                                  REINTO                  ;TO
011346  000023                                  19.                     ;NUMBER

                                                ;FOR RHAS UPPER BYTE
011350  113737  001721  003155                  MOVB    @#AS+1,@#REINTO+25 ;UPPER RHAS
```

```
 6228
 6229
 6230                                        ;COMPARE REGISTERS BEFORE COMMAND WITH AFTER COMMAND
 6231  011356  004037  025342                JSR     R0,@#COMPAR     ;COMPARE
 6232  011362  001674                        WC                      ;GOOD BUFFER
 6233  011364  003130                        REINTO                  ;TEST BUFFER
 6234  011366  000022                        18.                     ;NUMBER
 6235  011370  011376                        2$                      ;RETURN FOR ERROR
 6236  011372  011376                        2$                      ;SAME
 6237  011374  011416                        3$                      ;RETURN FOR GOOD COMPARISON
 6238
 6239  011376  013705  031136          2$:   MOV     @#ERWORD,R5     ;GETTING READY TO INDEX
 6240  011402  060505                        ADD     R5,R5           ;DOUBLE ERROR WORD
 6241  011404  016537  001622  024234        MOV     RHWC-2(R5),@#REGADR ;FAILING REGISTER
 6242  011412  104001                        ERROR   1               ;IMPROPER REGISTER
 6243                                                                ;CHANGE WHILE EXPECTING
 6244                                                                ;DTE ERROR
 6245  011414  000207                        RTS     PC              ;RETURN
 6246
 6247  011416                          3$:                           ;GOOD
 6248
 6249
 6259
 6260                                        ;*****************************************************************
  (3)                                       ;*TEST 14       DRIVE TIMING ERROR
  (4)
  (4)                                       ;*      A WRITE HEADER AND DATA COMMAND IS GIVEN
  (4)                                       ;*      TO CYLINDER 0, TRACK 0, 256. WORDS
  (4)                                       ;*      AFTER SECTOR IS FOUND (THE SECTOR FOUND FLOP IS HIGH)
  (4)                                       ;*      NO MORE DIAGNOSTIC CLOCKS ARE GIVEN,
  (4)                                       ;*      ONLY SECTOR CLOCKS ARE GIVEN TILL SECTOR PULSE IS HIGH
  (4)                                       ;*       THIS SHOULD SET DRIVE TIMING ERROR
  (3)                                       ;*****************************************************************
  (2)  011416  000004                   TST14:  SCOPE
 6261  011420  012706  001000                  MOV     @#STACK,SP      ;RESET STACK
 6262
  (1)  011424  012737  000014  004174          MOV     @#TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
  (1)
 6263
 6264
 6265                                        ;THESE ARE TO SET UP FOR DISKLESS USE ONLY
 6266
 6267  011432  012737  010000  034122          MOV     @#PAT22,@#WCYL  ;FORMAT 22=16 BITWORDS AND
 6268                                                                  ;CYLINDER 0
 6269  011440  005037  034124                  CLR     @#WSECTR        ;TRACK=0, SECTOR=0
 6270  011444  005037  034126                  CLR     @#WKEY1         ;KEY1=0
 6271  011450  005037  034130                  CLR     @#WKEY2         ;KEY2=0
 6272  011454  012737  000400  034162          MOV     #256.,@#WNWORD  ;256 DATAWORDS
 6273  011462  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
 6274  011466  034122                          WCYL
 6275  011470  034132                          GCRC
 6276
 6277
 6278                                        ; THESE ARE REGULAR SETUPS
```

```
 6279  011472  004737  024470                  JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
 6280  011476  012777  177374  170120          MOV     #-260.,,@RHWC   ;256 DATA WORDS 4HEADER
 6281  011504  012777  002064  170114          MOV     #WRFROM,@RHBA   ;STARTING ADDRESS OF BUFFER
 6282  011512  005077  170120                  CLR     @RHDST          ;TRACK = 0
 6283                                                                  ;SECTOR = 0
 6284  011516  012777  014000  170116          MOV     #FMT22!ECI,@RHOF ;16 BITS PER WORD
 6285                                                                  ;ECC CORRECTION INHIBITED
 6286  011524  005077  170114                  CLR     @RHCA           ;CYLINDER = 0
 6287  011530  013711  002042                  MOV     @#WRIFOR,@R1    ;WRITE HEADER AND DATA = 62
 6288
 6289                                          ;SAVE REGISTERS FOR COMPARISON AFTER SIMULATED DRIVE TIMING ERROR
 6290  011534  004037  025146                  JSR     R0,@#SAVER      ;SAVE
 6291  011540  001624                          RHWC                    ;FROM
 6292  011542  001674                          WC                      ;TO
 6293  011544  000023                          19.                     ;NUMBER
 6294
 6295                                          ;NOW GO WILL BE GIVEN, EVERYTHING WILL BE TREATED
 6296                                          ;NORMALLY TILL HEADER IS TO BE GIVEN, THEN ONLY
 6297                                          ;SECTOR CLOCKS WILL BE GIVEN, NO DIAGNOSTIC
 6298                                          ;CLOCKS WILL BE GIVEN, THIS SHOULD BRING SECTOR PULSE HIGH
 6299                                          ;WITHOUT PUTTING READ DOWN HENCE DTE WILL COME UP.
 6300  011546  012737  177777  002014          MOV     #-1,@#TESDTE    ;SET DTE TEST
 6301  011554  012737  177777  001774          MOV     #-1,@#ERFLGS    ;THIS WILL BRING THE READ HEADER
 6302                                                                  ;AND DATA PROCESS OUT AFTER THE
 6303                                                                  ;HEADER HAS BEEN CORRECTLY READ
 6304
 6305  011562  004737  033776                  JSR     PC,@#COMWHD     ;WRITE HEADER AND DATA
 6306
 6307                                          ;NOW SECTOR HAS BEEN FOUND
 6308                                          ;NOW 609 SECTOR CLOCKS WILL BE GIVEN
 6309                                          ;39 BYTES FOR SECTOR GAP
 6310                                          ;1 BYTE FOR HEADER SYNC
 6311                                          ;8 BYTES FOR HEADER
 6312                                          ;GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
 6313                                          ;GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 3
 6314
 6315
 6316
 6317                                          ;THIS GIVES 609 SECTOR CLOCKS WITHOUT DIAGNOSTIC CLOCKS
 6318  011566  012701  001141                  MOV     #609.,,R1       ;COUNTER,
 6319  011572  052710  000010          58:     BIS     #WSTCK,@R0      ;SET SECTOR CLOCK
 6320  011576  042710  000010                  BIC     #WSTCK,@R0      ;CLEAR SECTOR CLOCK
 6321  011602  005301                          DEC     R1              ;COUNT
 6322  011604  001372                          BNE     58              ;BRANCH IF 536 NOT DONE
 6323
 6324
 6325
 6326                                          ;NOW DTE SHOULD BE SET CHANGE SAVED REGISTERS TO EXPECTED VALUE
 6327  011606  012737  177404  001674          MOV     #-252.,,@#WC ;SAVED RHWC
 6328  011614  012737  002104  001676          MOV     #WRFROM+<8.*2>,@#BA ;SAVED RHBA
 6329  011622  052737  140000  001702          BIS     #SCITRE,@#CS1           ;SAVED RHCS1
 6330  011630  042737  000100  001700          BIC     #IR,@#CS2       ;SAVED RHCS2
 6331  011636  052737  000200  001700          BIS     #OR,@#CS2       ;SAVED RHCS2
 6332  011644  052737  010000  001704          BIS     #DTE,@#ER1      ;SAVED RHER1
```

```
 6333  011652  012737  000401  001722          MOV     #401,@#MR       ;SAVED RHMR
 6334  011660  052737  140000  001724          BIS     #ATA!ERR,@#DS1  ;SAVED RHDS1
 6335  011666  012737  000100  001736          MOV     #100,@#LA       ;SAVED RHLA
 6336  011674  012737  000001  001706          MOV     #1,@#DST        ;SAVED RHDST
 6337  011702  013737  002004  001720          MOV     @#ATTENT,@#AS   ;SAVED RHAS
 6338
 6339
 6340                                          ;NOW SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN BE DONE
 6341  011710  004037  025146                  JSR     R0,     @#SAVER ;SAVE
 6342  011714  001624                          RHWC                    ;FROM
 6343  011716  003130                          REINTO                  ;TO
 6344  011720  000023                          19.                     ;NUMBER
 6345
 6346                                          ;FOR RHAS UPPER BYTE
 6347  011722  113737  001721  003155          MOVB    @#AS+1,@#REINTO+25 ;UPPER RHAS
 6348
 6349
 6350
 6351                                          ;COMPARE REGISTERS BEFORE COMMAND WITH AFTER COMMAND
 6352  011730  004037  025342                  JSR     R0,@#COMPAR     ;COMPARE
 6353  011734  001674                          WC                      ;GOOD BUFFER
 6354  011736  003130                          REINTO                  ;TEST BUFFER
 6355  011740  000022                          18.                     ;NUMBER
 6356  011742  011750                          28                      ;RETURN FOR ERROR
 6357  011744  011750                          28                      ;SAME
 6358  011746  011770                          38                      ;RETURN FOR GOOD COMPARISON
 6359
 6360  011750  013705  031136          28:     MOV     @#ERWORD,R5     ;GETTING READY TO INDEX
 6361  011754  060505                          ADD     R5,R5           ;DOUBLE ERROR WORD
 6362  011756  016537  001622  024234          MOV     RHWC-2(R5),@#REGADR ;FAILING REGISTER
 6363  011764  104001                          ERROR   1               ;IMPROPER REGISTER
 6364                                                                  ;CHANGE WHILE EXPECTING
 6365                                                                  ;DTE ERROR
 6366  011766  000207                          RTS     PC              ;RETURN
 6367
 6368  011770                          38:                             ;GOOD
 6369
 6370
 6371
 6382                                          ;*****************************************************************
  (3)                                          ;*TEST 15       SECTOR SELECTION
  (4)                                          ;*      THE SECTOR SELECTION LOGIC IS CHECKED HERE
  (4)                                          ;*      EACH SECTOR ON TRACK ZERO IS WRITTEN INTO
  (4)                                          ;*      DATA IS - 19 WORDS OF ZEROS - SYNC WORDS, 4 HEADER WORDS
  (4)                                          ;*      1 CRC WORD, 5 WORDS OF ZEROS, 1 SYNC WORD, 100 ZEROS
  (4)                                          ;*      (DATA), 1 SYNC WORD, 70 SECTOR NUMBER TO VARY
  (4)                                          ;*
  (4)                                          ;*      THE WRITTEN DATA IS CHECKED IN MEMORY
  (4)
  (3)                                          ;*****************************************************************
  (2)  011770  000004                  TST15:  SCOPE
 6383  011772  012706  001000                  MOV     @#STACK,SP      ;RESET STACK
 6384
 6385
```

```
   (1)  011776  012737  000015  004174          MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
   (1)
  6386
  6387  012004  012737  000026  002016          MOV     #22.,@#TAGDTE   ;22 SECTORS
  6388                                                                  ;THIS TEST REPEATS
  6389                                                                  ;ITSELF 22 TIMES
  6390
  6391                                          ;THE FOLLOWING INITIALIZES FOR SECTOR 0
  6392
  6393  012012  005037  012122                  CLR     @#883+2         ;HEADER (SECTOR)
  6394  012016  012737  000025  012126          MOV     #21.,@#884+2    ;HEADER (KEY1)
  6395  012024  012737  000025  012132          MOV     #21.,@#885+2    ;HEADER (KEY2)
  6396  012032  005037  012160                  CLR     @#887+2         ;DATA (SECTOR)
  6397  012036  005037  012242                  CLR     @#8810+2                ;DATA
  6398  012042  005037  012272                  CLR     @#8812+2                ;SECTOR (SIMULATED DISK)
  6399  012046  012737  000025  012300          MOV     #21.,@#8813+2   ;KEY1 (SIMULATED DISK)
  6400  012054  012737  000025  012306          MOV     #21.,@#8814+2   ;KEY2 (SIMULATED DISK)
  6401  012062  005037  012346                  CLR     @#8815+2                ;SECTOR (RHDST)
  6402
  6403                                          ;CLEAR SIMULATED DISK AREA
  6404  012066                          8811:
  6405  012066  012700  032636          10$:    MOV     #SECGAP,R0      ;POINTER
  6406  012072  012701  000460                  MOV     #304.,R1        ;COUNTER
  6407  012076  005020                  20$:    CLR     (R0)+           ;CLEAR SIMULATED DISK AREA
  6408  012100  005301                          DEC     R1              ;COUNT
  6409  012102  001375                          BNE     20$
  6410
  6411                                          ;SETUP GENERAL REGISTERS
  6412  012104  004737  024470                  JSR     PC,@#CLDISK
  6413
  6414                                          ;SETUP WRITE FROM BUFFER
  6415  012110  012700  002064                  MOV     #WRFROM,R0
  6416
  6417                                          ;HEADER
  6418  012114  012720  010000                  MOV     #FMT22,(R0)+    ;FORMAT 16 BITS PER WORD
  6419                                                                  ;CYLINDER 0
  6420  012120  012720  000000          8831:   MOV     #0,(R0)+        ;SECTOR TO VARY
  6421  012124  012720  000025          8841:   MOV     #21.,(R0)+      ;KEY1 TO VARY
  6422  012130  012720  000025          8851:   MOV     #21.,(R0)+      ;KEY2 TO VARY
  6423
  6424                                          ;DATA IN WRITE FROM BUFFER ALTHOUGH THIS IS DATA AND NOT
  6425                                          ;HEADER THE SECTOR WITH SYNC BYTES WILL BE GIVEN AS DATA
  6426                                          ;DATA IS - 19 WORDS OF ZEROS - SYNC WORDS, 4 HEADER WORDS
  6427                                          ;1 CRC WORD, 5 WORDS OF ZEROS, 1 SYNC WORD, 100 ZEROS
  6428                                          ;(DATA), 1 SYNC WORD, 70 SECTOR NUMBER TO VARY
  6429
  6430  012134  012705  000023                  MOV     #19.,R5         ;COUNTER
  6431  012140  005020                  60$:    CLR     (R0)+           ;19 ZEROS
  6432  012142  005305                          DEC     R5              ;COUNT
  6433  012144  001375                          BNE     60$             ;19 DONE?
  6434  012146  012720  031120                  MOV     #RSYNC,(R0)+    ;SYNC = 14400
  6435  012152  012720  010000                  MOV     #FMT22,(R0)+    ;CYLINDER 0
  6436  012156  012720  000000          8871:   MOV     #0,(R0)+        ;SECTOR TO VARY
  6437  012162  005020                          CLR     (R0)+
```

```
6438  012164  005020                          CLR     (R0)+
6439  012166  004537  025646                  JSR     R5,@#CRC        ;CALCULATE CRC FOR ABOVE 4 WORDS
6440  012172  002134                          WRFROM+50               ;4 WORDS START FROM HERE
6441  012174  002144                          WRFROM+60               ;PUT CRC HERE
6442
6443  012176  005720                          TST     (R0)+           ;INCREMENT R0
6444
6445  012200  012705  000005                  MOV     #5.,R5
6446  012204  005020                  88:     CLR     (R0)+           ;5 WORDS OF ZEROS
6447  012206  005305                          DEC     R5              ;COUNT
6448  012210  001375                          BNE     88              ;BRANCH IF 5 NOT DONE
6449
6450  012212  013720  031120                  MOV     @#RSYNC,(R0)+   ;SYNC = 14400
6451
6452  012216  012705  000144                  MOV     #100.,R5
6453  012222  005020                  98:     CLR     (R0)+           ;100 WORDS OF ZEROS
6454  012224  005305                          DEC     R5
6455  012226  001375                          BNE     98
6456
6457  012230  013720  031120                  MOV     @#RSYNC,(R0)+   ;SYNC = 14400
6458  012234  012705  000106                  MOV     #70.,R5
6459  012240  012720  000000          8810:   MOV     #0,(R0)+        ;SECTOR TO VARY
6460  012244  005305                          DEC     R5
6461  012246  001374                          BNE     8810
6462
6463                                          ;CLEAR REST OF 256 WORDS THAT IS 54 WORDS OF ZEROS
6464
6465  012250  012705  000066                  MOV     #54.,R5
6466  012254  005020                  118:    CLR     (R0)+
6467  012256  005305                          DEC     R5
6468  012260  001375                          BNE     118
6469
6470                                          ;THESE ARE TO BE SET UP FOR DISKLESS USE ONLY
6471  012262  012737  010000  034122          MOV     #FMT22,@#WCYL   ;FORMAT = 16 BIT WORDS
6472                                                                  ;CYLINDER = 0
6473  012270  012737  000000  034124  8812:   MOV     #0,@#WSECTR     ;SECTOR TO VARY
6474  012276  012737  000025  034126  8813:   MOV     #21.,@#WKEY1    ;KEY1 TO VARY
6475  012304  012737  000025  034130  8814:   MOV     #21.,@#WKEY2    ;KEY2 TO VARY
6476  012312  012737  000312  034162          MOV     #202.,@#FNWORD  ;202 DATA WORDS
6477  012320  004537  025646                  JSR     R5,@#CRC        ;CALCULATE CRC
6478  012324  034122                          WCYL                    ;FIRST WORD
6479  012326  034132                          GCRC                    ;PUT HERE
6480
6481                                          ;THESE ARE REGULAR SETUPS
6482  012330  012777  177400  167266          MOV     #-256.,@RHWC    ;202 DATA, 4 HEADER
6483  012336  012777  002064  167262          MOV     #WRFROM,@RHBA   ;FILL BUS ADDRESS
6484  012344  012777  000000  167264  8815:   MOV     #0,@RHDST       ;SECTOR TO VARY
6485  012352  013777  002042  167252          MOV     @#WRIFOR,@RHCS1 ;GET READY TO DO
6486                                                                  ;WRITE HEADER AND DATA
6487                                                                  ;WITH 62 IN RHCS1
6488  012360  012777  010000  167254          MOV     #FMT22,@RHOF    ;16 BITS PER WORD FORMAT
6489  012366  005077  167252                  CLR     @RHCA           ;CYLINDER = 0
6490
6491  012372  005037  001774                  CLR     @#ERFLG         ;CLEAR ERROR FLAG
```

```
 6492
 6493
  (1)
  (1)  012376  004767  012122                JSR     PC,CHECKT               ;CHECK DVA, RDY, DPR, DRY
  (1)
 6494
 6495  012402  004737  033776                JSR     PC,@#COMWMD     ;WRITE HEADER AND DATA COMMAND
 6496  012406  005737  001774                TST     @#ERFLG#        ;HAS ANY ERRORS OCCURRED
 6497
  (1)  012412  001046                        BNE     T#T16   ;BRANCH IF YES
  (1)
 6498
 6499  012414  004737  024706                JSR     PC,@#CHECKE     ;CHECK DVA,RDY,DRY,DPR
 6500
 6501                                        ;NOW COMPARE "DISK" BUFFER WITH "REINTO" BUFFER
 6502  012420  004037  025342                JSR     R0,@#COMPAR     ;CHECK
 6503  012424  002074                        WRFROM+0.               ;GOOD BUFFER
 6504  012426  032734                        DISK                    ;TEST BUFFER
 6505  012430  000400                        256.                    ;NUMBER OF WORDS
 6506  012432  012440                        16$                     ;RETURN POINT FOR ERROR  HEADER
 6507  012434  012444                        17$                     ;RETURN POINT FOR ERROR DATA
 6508  012436  012450                        18$                     ;RETURN FOR GOOD COMPARISON
 6509  012440  104007                16$:    ERROR   7
 6510  012442  000207                        RTS     PC
 6511  012444  104010                17$:    ERROR   10
 6512  012446  000207                        RTS     PC
 6513
 6514                                        ;THE FOLLOWING INCREMENTS ARE TO CHANGE THE ABOVE SET UP
 6515                                        ;TO WRITE ON THE NEXT SECTOR
 6516
 6517  012450  005237  012122        18$:    INC     @#883+2         ;HEADER (SECTOR)
 6518  012454  005337  012126                DEC     @#884+2         ;HEADER (KEY1)
 6519  012460  005337  012132                DEC     @#885+2         ;HEADER (KEY2)
 6520  012464  005237  012160                INC     @#887+2         ;DATA (SECTOR)
 6521  012470  005237  012242                INC     @#8818+2                ;DATA
 6522  012474  005237  012272                INC     @#8812+2                ;SECTOR (SIMULATED DISK)
 6523  012500  005337  012300                DEC     @#8813+2                ;KEY1 (SIMULATED DISK)
 6524  012504  005337  012306                DEC     @#8814+2                ;KEY2 (SIMULATED DISK)
 6525  012510  005237  012346                INC     @#8815+2                ;SECTOR (RMD8T)
 6526
 6527  012514  005337  002016        8$:     DEC     @#TAGDTE                ;COUNT DOWN FOR 22 SECTORS
 6528  012520  001001                        BNE     1$              ;BRANCH IF 22 SECTORS NOT DONE
 6529
  (1)  012522  000402                        BR      T#T16   ;GO OUT
  (1)
 6530  012524  000137  012066        1$:     JMP     @#881   ;GO BACK
 6531
 6532
 6533
 6542
 6543                                        ;*****************************************************************
  (3)                                        ;*TEST 16       WRITE ECC TEST 1
  (4)
  (4)                                        ;*      THIS IS A WRITE ECC TEST
```

```
 (4)                                    ;*      WRITE CYLINDER0, FORMAT 16 BITS PER WORD
 (4)                                    ;*      TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
 (4)                                    ;*      OF ALL ZEROS.
 (4)
 (3)                                    ;*************************************************************
 (2)  012530  000004                    TST16:  SCOPE
6544  012532  012706  001000                    MOV     #STACK,SP       ;RESET STACK
6545
6546
 (1)  012536  012737  000016  004174            MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
 (1)
6547  012544  012700  032636                    MOV     #SECGAP,R0      ;POINTER
6548  012550  012701  000402                    MOV     #258.,R1        ;COUNTER
6549  012554  012720  177777            18:     MOV     #-1,(R0)+       ;FILL SIMULATER DISK WITH ONES
6550  012560  005301                            DEC     R1
6551  012562  001374                            BNE     18
6552  012564  004767  011700                    JSR     PC,CLDISK                       ;THIS IS USED TO SET GENERAL REGISTERS
6553
6554                                    ;THESE ARE FOR ECC TEST ONLY
6555
6556  012570  012737  177777  002012            MOV     #-1,@#TSECC             ;THIS IS AN ECC TEST
6557  012576  005037  027140                    CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
6558  012602  013737  027134  027136            MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
6559  012610  013737  027142  027150            MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
6560  012616  005037  027126                    CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
6561  012622  005037  027130                    CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
6562  012626  005037  027144                    CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
6563  012632  005037  027146                    CLR     @#ECODE ;CLEAR LEADING ZEROS CLOCK COUNT
6564
6565
6566
6567
6568                                    ;THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
6569
6570  012636  012737  010000  034122            MOV     #FMT22,@#WCYL   ;FORMAT22=16BIT WORDS AND
6571                                                                    ;CYLINDER 0
6572  012644  012737  000001  034124            MOV     #1,@#WSECTR     ;TRACK=0, SECTOR=1
6573  012652  005037  034126                    CLR     @#WKEY1         ;KEY1=0
6574  012656  005037  034130                    CLR     @#WKEY2         ;KEY2=0
6575  012662  012737  000400  034162            MOV     #256.,@#FNWORD  ;256 DATA WORDS
6576  012670  004537  025646                    JSR     R5,@#CRC        ;GO TO CALCULATE CRC
6577  012674  034122                            WCYL
6578  012676  034132                            GCRC
6579
6580                                    ;THESE ARE REGULAR SETUPS
6581
6582  012700  012777  177374  166716            MOV     #-260.,@RHWC    ;256 DATA WORDS 4 HEADER WORDS
6583  012706  012700  002064                    MOV     #WRFROM,R0      ;THESE TWO INSTRUCTIONS GETS
6584  012712  010077  166710                    MOV     R0,@RHBA        ;ADDR. OF WRFROM INTO R0 AND
6585                                                                    ;BUS ADDRESS REGISTER
6586  012716  012720  010000                    MOV     #FMT22,(R0)+    ;FORMAT=16 BIT WORDS
6587                                                                    ;CYLINDER=0
6588  012722  012720  000001            28:     MOV     #1,(R0)+        ;TRACK=0, SECTOR=1, KEYS=0
6589  012726  005020                            CLR     (R0)+           ;KEY1=0
```

```
 6590  012730  005020                          CLR     (R0)+           ;KEY2=0
 6591  012732  012705  000400                  MOV     #256.,R5        ;COUNTER
 6592  012736  012720  000000          38:     MOV     #0,(R0)+        ;MOVE ALL ZEROS FOR DATA
 6593  012742  005305                          DEC     R5
 6594  012744  001374                          BNE     38              ;BRANCH IF DATA NOT COMPLETE
 6595  012746  012777  000001  166662          MOV     #1,@RHDST       ;TRACK=0 SECTOR=1
 6596
 6597
  (1)
  (1)  012754  004767  011544                  JSR     PC,CHECKT               ;CHECK DVA, RDY, DPR, DRY
  (1)
 6598
 6599  012760  013711  002042                  MOV     @#WRIFOR,@R1    ;GET READY FOR WRITE HEADER AND
 6600                                                                  ;DATA WITH 62 IN RHCS1
 6601  012764  005037  001774                  CLR     @#ERFLG#        ;CLEAR ERROR FLAG
 6602  012770  012777  010000  166644          MOV     #FMT22,@RHOF    ;FORMAT BIT=1 (16 BIT WORDS)
 6603  012776  005077  166642                  CLR     @RHCA           ;CYLINDER =0
 6604  013002  004737  033776                  JSR     PC,@#COMWHD     ;WRITE HEADER AND DATA
 6605
 6606                                          ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 6607                                          ;FROM THE "COMWHD" ROUTINE THAT MEANS ALL HEADER ON DISK
 6608                                          ;IS GOOD IE, ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
 6609                                          ;ALL ZEROS AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
 6610                                          ;THEY ARE ALL ZEROS
 6611
 6612  013006  005737  001774                  TST     @#ERFLG#                ;HAS ANY ERRORS OCCURED?
 6613                                                                          ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
 6614
  (1)  013012  001053                          BNE     TST17   ;;BRANCH IF YES
  (1)
 6615
 6616
 6617                                  ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
 6618  013014  023737  027126  033734          CMP     @#GECC1,@#WECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
 6619  013022  001402                          BEQ     68              ;BRANCH IF GOOD
 6620  013024  104031                          ERROR   31              ;LOW ORDER ECC IN ERROR
 6621  013026  000405                          BR      78              ;BRANCH TO CONTINUE
 6622  013030  023737  027130  033736  68:     CMP     @#GECC2,@#WECC2;COMPARE SOFTWARE ECC WITH HARDWARE ECC
 6623  013036  001401                          BEQ     78              ;BRANCH IF GOOD
 6624  013040  104031                          ERROR   31              ;HIGH ORDER ECC IN ERROR
 6625
 6626
 6627  013042  004737  024706          78:     JSR     PC,@#CHECKE     ;CHECK DVA,RDY,DRY,DPR
 6628
 6629
 6630
 6631                                  ;FILL "REINTO" BUFFER WITH EXPECTED DATA
 6632  013046  004037  024406                  JSR     R0,@#CLAREA     ;FILL REINTO BUFFER
 6633  013052  003130                          REINTO                  ;FROM
 6634  013054  004126                          REINTO+<256.*2>         ;TO
 6635  013056  000000                          .WORD   0               ;DATA
 6636
 6637  013060  013737  027126  004130          MOV     @#GECC1,@#REINTO+<256.*2>;FILL ECC1
 6638  013066  013737  027130  004132          MOV     @#GECC2,@#REINTO+<257.*2>;FILL ECC2
```

```
6639  013074  004037  024406                  JSR     R0,@#CLAREA     ;FILL REST
6640  013100  004134                          REINTO+<256.*2>         ;FROM
6641  013102  004170                          REINTO+<272.*2>         ;TO
6642  013104  000000                          0                       ;DATA
6643
6644
6645  013106  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
6646
6647
6648                                  ;NOW COMPARE "DISK" BUFFER WITH "REINTO"
6649  013112  004037  025342                  JSR     R0,@#COMPAR     ;CHECK
6650  013116  003130                          REINTO                  ;GOOD BUFFER
6651  013120  032734                          DISK                    ;TEST BUFFER
6652  013122  000402                          256.                    ;NUMBER OF WORDS CHECKED
6653  013124  013132                          48                      ;RETURN POINT FOR ERROR HEADER
6654  013126  013136                          58                      ;RETURN POINT FOR ERROR DATA
6655
 (1)  013130  013142                          T8T17                   ;RETURN FOR GOOD COMPARISON
 (1)
6656  013132  104007                  48:     ERROR   7               ;READ ERROR 10 NEXT
6657  013134  000207                          RT#     PC              ;RETURN TO COMPARE
6658  013136  104010                  58:     ERROR   10              ;WORD NOS 1 TO 256 ARE
6659                                                                  ;DATA WORDS
6660                                                                  ;WORD NOS 257 AND 258
6661                                                                  ;ARE ECC WHICH ARE CHECKED
6662                                                                  ;WORD NOS 259
6663                                                                  ;IS DATA GAP
6664                                                                  ;WORD NOS 260 TO 273
6665                                                                  ;ARE TOLERANCE GAP
6666  013140  000207                          RT#     PC              ;RETURN TO COMPARE
6667
6668
6669
6670
6671
6682
6683                                  ;*****************************************************************
 (3)                                  ;*TEST 17        READ ECC ENABLED 1A
 (4)
 (4)                                  ;*       THIS IS AN ECC READ DATA TEST
 (4)                                  ;*       ERROR CORRECTION IS ENABLED
 (4)                                  ;*       NO ERROR IS INSERTED
 (4)                                  ;*       GOOD DATA USED IS 256 WORDS OF 0
 (4)                                  ;*       COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
 (4)                                  ;*       TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
 (4)
 (3)                                  ;*****************************************************************
 (2)  013142  000004                  T8T17:  SCOPE
6684  013144  012706  001000                  MOV     #STACK,SP       ;RESET STACK
6685
6686
 (1)  013150  012737  000017  004174          MOV     #TTNO,@#T#TNM   ;THIS SAVES TEST NUMBER
 (1)
6687
```

```
                                        ;       SETUP FOR WHAT IS TO BE READ
                                        ;       HEADER CRC IS RESTORED FROM A SUBROUTINE
013156  012746  000000                  MOV     #0,     -(SP)   ;DATA TO BE READ
013162  012705  000400                  MOV     #256.,  R5      ;COUNTER
013166  012700  032734                  MOV     #DISK,  R0      ;START OF SIMULATED DISK DATA
013172  011620                  18:     MOV     (SP),   (R0)+   ;MOVE IN DATA ON TO SIMULATED DISK
013174  005305                          DEC     R5              ;COUNT
013176  001375                          BNE     1B              ;BRANCH IF 256 NOT COMPLETE
013200  005726                          TST     (SP)+           ;UNDO -(SP)
013202  022020                          CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
013204  012705  000017                  MOV     #15.,   R5      ;1 DATA GAP
                                                                ;14 TOLERANCE GAP
013210  005020                  2$:     CLR     (R0)+           ;CLEAR DATA GAP, AND
013212  005305                          DEC     R5              ;TOLERANCE GAP
013214  001375                          BNE     2$              ;BRANCH IF NOT COMPLETE


013216  004737  027722                  JSR     PC,@#FILLEC     ;INSERT THE TWO ECC WORDS ON THE DISK
                                                                ;IN THE CORRECT PLACE


                                        ;THESE ARE FOR ECC TEST ONLY

013222  012737  177777  002012          MOV     #-1,@#TSECC             ;THIS IS AN ECC TEST
013230  005037  027140                  CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
013234  013737  027134  027136          MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
013242  013737  027142  027150          MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
013250  005037  027126                  CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
013254  005037  027130                  CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
013260  005037  027144                  CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
013264  005037  027146                  CLR     @#ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT


                                        ;THESE ARE TO SETUP FOR DISKLESS USE ONLY

013270  012737  010000  031016          MOV     #FMT22,@#CYL    ;16 BITS PER WORD
                                                                ;CYLINDER 0, FORMAT 16 BITS
013276  112737  000000  031021          MOVB    #0,     @#SECOTR+1 ;TRACK 0
013304  112737  000000  031020          MOVB    #0,     @#SECOTR ;SECTOR 0
013312  012737  000000  031022          MOV     #0,     @#KEY1  ;KEY1=0
013320  012737  000000  031024          MOV     #0,     @#KEY2  ;KEY2=0
013326  012737  000400  031076          MOV     #256.,  @#DAWORD ;NO. OF DATA WORDS
013334  005037  031026                  CLR     @#X             ;THIS IS A READ COMMAND
013340  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
013344  031016                          CYL
013346  032716                          WCRC


                                        ;THESE ARE REGULAR SETUPS
013350  004737  024470                  JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
013354  012777  177374  166242          MOV     #-256.*4.,@RHWC ;256. DATA 4 HEADER WORDS
```

K6

```
6742  013362  012777  003130  166236          MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
6743  013370  112746  000000                  MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
6744  013374  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
6745  013402  012677  166230                  MOV     (SP)+,  @RHDST  ;TRACK/SECTOR IN RHDST
6746  013406  012777  010000  166226          MOV     #FMT22,@RHOF ;16 BITS PER WORD
6747                                                                  ;ECC CORRECTION NOT INHIBIT
6748                                                                  ;BECAUSE ECC IS NOT GOING
6749                                                                  ;TO BE CHECKED
6750  013414  005077  166224                  CLR     @RHCA           ;CYLINDER 0
6751
6752  013420  004737  024524                  JSR     PC,     @#CHECKT ;CHECK FOR DVA,RDY,NOL,DPR,DRY
6753
6754  013424  013711  002046                  MOV     @#REFOR,@R1     ;READ HEADER AND DATA#72
6755  013430  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
6756  013434  004737  030706                  JSR     PC,     @#COMMD ;READ HEADER AND DATA
6757                                                                  ;IF THERE ARE READ ERRORS THEN
6758                                                                  ;ECC WILL NOT BE CHECKED
6759
6760
6761                                          ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6762                                          ;FROM THE "COMMD" ROUTINE THAT MEANS SECTOR GAP,
6763                                          ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
6764                                          ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
6765                                          ;DETECTED
6766                                          ;HEADER AND DATA ARE TO BE CHECKED,
6767                                          ;IN CHECKING READ DATA THE WRITE FROM BUFFER
6768                                          ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
6769                                          ;COMPARISONS ARE MADE
6770
6771  013440  005737  001774                  TST     @#ERFLG#                ;ANY ERRORS ALREADY THERE
6772
 (1)  013444  001077                          BNE     TST20   ;BRANCH IF YES
 (1)
6773  013446  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
6774  013452  005737  001704                  TST     @#ER1           ;NO ERRORS SHOULD BE SET
6775  013456  001401                          BEQ     6$              ;BRANCH IF NO ERRORS SET
6776  013460  104032                          ERROR   32              ;32 BIT ECC REGISTER SHOULD BE ZERO
6777                                                                  ;ONLY 11 OF THE 32 BITS CAN BE SEEN
6778                                                                  ;IN THE PATERN REGISTER
6779                                                                  ;DCK SHOULD BE SET IN RMER1
6780  013462  013746  027126          6$:     MOV     @#GECC1,-(SP)   ;GET PATTERN REGISTER
6781  013466  042716  174000                  BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
6782  013472  022637  001734                  CMP     (SP)+,@#EC2     ;COMPARE PATTERN REGISTER
6783  013476  001401                          BEQ     7$              ;BRANCH IF GOOD
6784  013500  104032                          ERROR   32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6785
6786
6787
6788
6789                                          ;ADD 16 MAINTENANCE CLOCKS TO
6790                                          ;BRING EBL DOWN
6791
6792  013502  012700  000020          7$:     MOV     #16.,,R0        ;COUNTER
6793  013506  052777  000002  166136  8$:     BIS     #MCLK,@RHMR     ;SET CLOCK
```

```
6794  013514  042777  000002  166130          BIC     #MCLK,@RHMR     ;CLEAR CLOCK
6795  013522  005300                          DEC     R0              ;COUNT
6796  013524  001370                          BNE     3$              ;BRANCH IF 16 CLOCKS NOT DONE
6797  013526  004737  024706                  JSR     PC,@#CHECKE     ;CHECK DVA,DRY,RDY,DPR
6798  013532  012700  002064                  MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
6799  013536  012720  010000                  MOV     #0!FMT22,(R0)+  ;CYLINDER 0
6800  013542  112746  000000                  MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
6801  013546  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
6802  013554  012620                          MOV     (SP)+,  (R0)+   ;GET TRACK/SECTOR IN BUFFER
6803  013556  012720  000000                  MOV     #0,     (R0)+   ;KEY1 IN BUFFER
6804  013562  012720  000000                  MOV     #0,     (R0)+   ;KEY2 IN BUFFER
6805  013566  012701  000400                  MOV     #256.,  R1      ;DATA WORD COUNTER
6806  013572  012702  000000                  MOV     #0,     R2      ;DATA
6807  013576  010220                  38$:    MOV     R2,     (R0)+   ;DATA INTO BUFFER
6808  013600  005301                          DEC     R1              ;COUNT
6809  013602  001375                          BNE     3$              ;BRANCH IF 256 NOT DONE
6810
6811
6812  013604  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
6813  013610  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS#
6814
6815
6816
6817                                          ;NOW READ DATA BUFFER WILL BE CHECKED
6818
6819  013614  004037  025342                  JSR     R0,@#COMPAR     ;CHECK
6820  013620  002064                          WRFROM                  ;GOOD BUFFER
6821  013622  003130                          REINTO                  ;TEST BUFFER
6822  013624  000404                          4*256.                  ;NUMBER OF WORDS CHECKED
6823  013626  013634                          4$                      ;RETURN POINT FOR ERROR HEADER
6824  013630  013640                          5$                      ;RETURN POINT FOR ERROR DATA
6825
 (1)  013632  013644                          TST20                   ;RETURN FOR GOOD COMPARISON
 (1)
6826  013634  104004                  4$:     ERROR   4               ;READ NEXT ERROR
6827  013636  000207                          RTS     PC              ;RETURN TO "COMPAR"
6828  013640  104005                  5$:     ERROR   5               ;WORD NOS 1 TO 4 ARE
6829                                                                  ;HEADER WORDS
6830                                                                  ;5 TO 260 ARE DATA WORDS
6831  013642  000207                          RTS     PC              ;RETURN TO "COMPAR"
6832
6833
6834
6835
6846
6847                                  ;*****************************************************************
 (3)                                  ;*TEST 20        READ ECC ENABLED 1B
 (4)
 (4)                                  ;*      THIS IS AN ECC READ DATA TEST
 (4)                                  ;*      ERROR CORRECTION IS ENABLED
 (4)                                  ;*      A CORRECTABLE ERROR IS INSERTED IN BIT POSITION 32
 (4)                                  ;*      GOOD DATA USED IS 256 WORDS OF 0
 (4)                                  ;*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
 (4)                                  ;*      TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
```

```
 (4)
 (3)                                    ;*****************************************************************
 (2)  013644  000004                    TST20:  SCOPE
6848  013646  012706  001000                     MOV     #STACK,SP       ;RESET STACK
6849
6850
 (1)  013652  012737  000020  004174           MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
 (1)
6851
6852
6853                                    ;       SETUP FOR WHAT IS TO BE READ
6854                                    ;       HEADER CRC IS RESTORED FROM A SUBROUTINE
6855
6856  013660  012746  000000                     MOV     #0,     -(SP)   ;DATA TO BE READ
6857  013664  012705  000400                     MOV     #256.,  R5      ;COUNTER
6858  013670  012700  032734                     MOV     #DISK,  R0      ;START OF SIMULATED DISK DATA
6859  013674  011620                    1$:     MOV     (SP),   (R0)+   ;MOVE IN DATA ON TO SIMULATED DISK
6860  013676  005305                             DEC     R5              ;COUNT
6861  013700  001375                             BNE     1$              ;BRANCH IF 256 NOT COMPLETE
6862  013702  005726                             TST     (SP)+           ;UNDO -(SP)
6863  013704  022020                             CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
6864  013706  012705  000017                     MOV     #15.,   R5      ;1 DATA GAP
6865                                                                     ;14 TOLERANCE GAP
6866  013712  005020                    2$:     CLR     (R0)+           ;CLEAR DATA GAP, AND
6867  013714  005305                             DEC     R5              ;TOLERANCE GAP
6868  013716  001375                             BNE     2$              ;BRANCH IF NOT COMPLETE
6869
6870
6871  013720  004737  027722                     JSR     PC,@#FILLEC     ;INSERT ECC IN PROPER PLACE ON DISK
6872
6873
6874
6875                                    ;THESE ARE FOR ECC TEST ONLY
6876
6877  013724  012737  177777  002012     MOV     #-1,@#TSECC             ;THIS IS AN ECC TEST
6878  013732  005037  027140             CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
6879  013736  013737  027134  027136     MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
6880  013744  013737  027142  027150     MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
6881  013752  005037  027126             CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
6882  013756  005037  027130             CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
6883  013762  005037  027144             CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
6884  013766  005037  027146             CLR     @#ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
6885
6886
6887                                    ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
6888
6889  013772  012737  010000  031016     MOV     #FMT22,@#CYL    ;16 BITS PER WORD
6890                                                             ;CYLINDER 0, FORMAT 16 BITS
6891  014000  112737  000000  031021     MOVB    #0,     @#SECOTR+1 ;TRACK 0
6892  014006  112737  000000  031020     MOVB    #0,     @#SECOTR ;SECTOR 0
6893  014014  012737  000000  031022     MOV     #0,     @#KEY1  ;KEY1=0
6894  014022  012737  000000  031024     MOV     #0,     @#KEY2  ;KEY2=0
6895  014030  012737  000400  031076     MOV     #256.,  @#DAWORD ;NO. OF DATA WORDS
6896  014036  005037  031026             CLR     @#X             ;THIS IS A READ COMMAND
```

```
6897  014042  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
6898  014046  031016                          CYL
6899  014050  032716                          WCRC
6900
6901
6902                                          ;THIS IS TO INSERT ERROR
6903                                          ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
6904                                          ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
6905                                          ;THIS MOVE
6906  014052  012737  100000  032736          MOV     #100000,@#DISK+2        ;FORCE ERROR ON BIT NUMBER 32
6907                                                                          ;SO ERROR POSITION REGISTER WILL SHOW
6908                                                                          ;22
6909  014060  012737  000026  014226          MOV     #22,,@#8008             ;INSERT POSITION REG.
6910
6911
6912                                          ;THESE ARE REGULAR SETUPS
6913  014066  004737  024470                  JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
6914  014072  012777  177374  165524          MOV     #-256.-4.,@#RHWC ;256. DATA & HEADER WORDS
6915  014100  012777  003130  165520          MOV     #REINTO,@#RHBA  ;STARTING ADDRESS OF READ BUFFER
6916  014106  112746  000000                  MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
6917  014112  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
6918  014120  012677  165512                  MOV     (SP)+,  @#RHDST ;TRACK/SECTOR IN RHDST
6919  014124  012777  010000  165510          MOV     #FMT22,@#RHOF ;16 BITS PER WORD
6920                                                                  ;ECC CORRECTION NOT INHIBIT
6921                                                                  ;BECAUSE ECC IS NOT GOING
6922                                                                  ;TO BE CHECKED
6923  014132  005077  165506                  CLR     @#RHCA          ;CYLINDER 0
6924
6925  014136  004737  024524                  JSR     PC,     @#CHECKT ;CHECK FOR DVA,RDY,NOL,DPR,DRY
6926
6927  014142  013711  002046                  MOV     @#REFOR,@R1     ;READ HEADER AND DATA=72
6928  014146  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
6929  014152  004737  030706                  JSR     PC,     @#COMMD ;READ HEADER AND DATA
6930                                                                  ;IF THERE ARE READ ERRORS THEN
6931                                                                  ;ECC WILL NOT BE CHECKED
6932
6933
6934                                          ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6935                                          ;FROM THE "COMMD" ROUTINE THAT MEANS SECTOR GAP,
6936                                          ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
6937                                          ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
6938                                          ;DETECTED
6939                                          ;HEADER AND DATA ARE TO BE CHECKED,
6940                                          ;IN CHECKING READ DATA THE WRITE FROM BUFFER
6941                                          ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
6942                                          ;COMPARISONS ARE MADE
6943
6944  014156  005737  001774                  TST     @#ERFLG#                ;ANY ERRORS ALREADY THERE
6945
 (1)  014162  001074                          BNE     TST21   ;BRANCH IF YES#
 (1)
6946  014164  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
6947  014170  022737  100000  001704          CMP     #DCK,@#ER1      ;ONLY DATA CHECK ERROR SHOULD BE SET
6948  014176  001401                          BEQ     6$              ;BRANCH IF YES
```

```
 6949  014200  104032                          ERROR   32              ;32 BIT ECC REGISTER SHOULD BE NON
 6950                                                                  ;ZERO
 6951                                                                  ;ONLY 11 OF THE 32 BITS CAN BE SEEN
 6952                                                                  ;IN THE PATERN REGISTER
 6953                                                                  ;DCK SHOULD BE SET IN RMER1
 6954  014202  013746  027126          68:     MOV     @#GECC1,-(SP)   ;GET PATTERN REGISTER
 6955  014206  042716  174000                  BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
 6956  014212  022637  001734                  CMP     (SP)+,@#EC2     ;COMPARE PATTERN REGISTER
 6957  014216  001401                          BEQ     7$              ;BRANCH IF GOOD
 6958  014220  104032                          ERROR   32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
 6959
 6960  014222  004037  027550          71:     JSR     R0,@#ECORR      ;GO TO ECC CORRECTION PROCESS
 6961  014226  000026                  88:     22.                     ;EXPECTED POSITION REG. WHEN CORRECTION
 6962                                                                  ;IS COMPLETE
 6963
 6964
 6965
 6966  014230  004737  024706                  JSR     PC,@#CHECKE     ;CHECK DVA,DRY,RDY,DPR
 6967  014234  012700  002064                  MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
 6968  014240  012720  010000                  MOV     #1@1FMT22,(R0)+ ;CYLINDER 0
 6969  014244  112746  000000                  MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
 6970  014250  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
 6971  014256  012620                          MOV     (SP)+,  (R0)+   ;GET TRACK/SECTOR IN BUFFER
 6972  014260  012720  000000                  MOV     #0,     (R0)+   ;KEY1 IN BUFFER
 6973  014264  012720  000000                  MOV     #0,     (R0)+   ;KEY2 IN BUFFER
 6974  014270  012701  000400                  MOV     #256.,  R1      ;DATA WORD COUNTER
 6975  014274  012702  000000                  MOV     #0,     R2      ;DATA
 6976  014300  010220                  38:     MOV     R2,     (R0)+   ;DATA INTO BUFFER
 6977  014302  005301                          DEC     R1              ;COUNT
 6978  014304  001375                          BNE     3$              ;BRANCH IF 256 NOT DONE
 6979
 6980                                                                  ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
 6981                                                                  ;NOW THE INSERTED ERROR WILL BE PUT IN
 6982  014306  012737  100000  002076          MOV     #100000,@#WRFROM+<5*2>  ;INSERTED ERROR
 6983
 6984
 6985
 6986  014314  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
 6987  014320  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
 6988
 6989
 6990                                                                  ;NOW READ DATA BUFFER WILL BE CHECKED
 6991
 6992  014324  004037  025342                  JSR     R0,@#COMPAR     ;CHECK
 6993  014330  002064                          WRFROM                  ;GOOD BUFFER
 6994  014332  003130                          REINTO                  ;TEST BUFFER
 6995  014334  000404                          4*256.                  ;NUMBER OF WORDS CHECKED
 6996  014336  014344                          4$                      ;RETURN POINT FOR ERROR HEADER
 6997  014340  014350                          5$                      ;RETURN POINT FOR ERROR DATA
 6998
  (1)  014342  014354                          TST21                   ;RETURN FOR GOOD COMPARISON
  (1)
 6999  014344  104004                  48:     ERROR   4               ;READ NEXT ERROR
 7000  014346  000207                          RTS     PC              ;RETURN TO "COMPAR"
```

```
 7001  014350  104005                  581:    ERROR   5               ;WORD NOS 1 TO 4 ARE
 7002                                                                  ;HEADER WORDS
 7003                                                                  ;5 TO 260 ARE DATA WORDS
 7004  014352  000207                          RTS     PC              ;RETURN TO "COMPAR"
 7005
 7016
 7017                                  ;*****************************************************************
  (3)                                  ;*TEST 21       READ ECC ENABLED 1C
  (4)
  (4)                                  ;*      THIS IS AN ECC READ DATA TEST
  (4)                                  ;*      ERROR CORRECTION IS ENABLED
  (4)                                  ;*      A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 21 THRU 32
  (4)                                  ;*      GOOD DATA USED IS 256 WORDS OF 0
  (4)                                  ;*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
  (4)                                  ;*      TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
  (4)
  (3)                                  ;*****************************************************************
  (2)  014354  000004                  TST21:  SCOPE
 7018  014356  012706  001000                  MOV     #STACK,SP       ;RESET STACK
 7019
 7020
  (1)  014362  012737  000021  004174          MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
  (1)
 7021
 7022
 7023                                  ;       SETUP FOR WHAT IS TO BE READ
 7024                                  ;       HEADER CRC IS RESTORED FROM A SUBROUTINE
 7025
 7026  014370  012746  000000                  MOV     #0,     -(SP)   ;DATA TO BE READ
 7027  014374  012705  000400                  MOV     #256.,  R5      ;COUNTER
 7028  014400  012700  032734                  MOV     #DISK,  R0      ;START OF SIMULATED DISK DATA
 7029  014404  011620                  18:     MOV     (SP),   (R0)+   ;MOVE IN DATA ON TO SIMULATED DISK
 7030  014406  005305                          DEC     R5              ;COUNT
 7031  014410  001375                          BNE     1$              ;BRANCH IF 256 NOT COMPLETE
 7032  014412  005726                          TST     (SP)+           ;UNDO -(SP)
 7033  014414  022020                          CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
 7034  014416  012705  000017                  MOV     #15.,   R5      ;1 DATA GAP
 7035                                                                  ;14 TOLERANCE GAP
 7036  014422  005020                  28:     CLR     (R0)+           ;CLEAR DATA GAP, AND
 7037  014424  005305                          DEC     R5              ;TOLERANCE GAP
 7038  014426  001375                          BNE     2$              ;BRANCH IF NOT COMPLETE
 7039
 7040
 7041  014430  004737  027722                  JSR     PC,@#FILLEC     ;INSERT THE TWO ECC WORDS ON THE DISK
 7042                                                                  ;IN THE CORRECT PLACE
 7043
 7044                                  ;THESE ARE FOR ECC TEST ONLY
 7045
 7046  014434  012737  177777  002012          MOV     #-1,@#TSECC             ;THIS IS AN ECC TEST
 7047  014442  005037  027140                  CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
 7048  014446  013737  027134  027136          MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
 7049  014454  013737  027142  027150          MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
 7050  014462  005037  027126                  CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
 7051  014466  005037  027130                  CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
```

```
 7052  014472  005037  027144                  CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
 7053  014476  005037  027146                  CLR     @#ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
 7054
 7055
 7056                                          ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
 7057
 7058  014502  012737  010000  031016          MOV     #FMT22,@#CYL    ;16 BITS PER WORD
 7059                                                                  ;CYLINDER 0, FORMAT 16 BITS
 7060  014510  112737  000000  031021          MOVB    #0,     @#SECOTR+1 ;TRACK 0
 7061  014516  112737  000000  031020          MOVB    #0,     @#SECOTR ;SECTOR 0
 7062  014524  012737  000000  031022          MOV     #0,     @#KEY1  ;KEY1=0
 7063  014532  012737  000000  031024          MOV     #0,     @#KEY2  ;KEY2=0
 7064  014540  012737  000400  031076          MOV     #256.,  @#DAWORD ;NO. OF DATA WORDS
 7065  014546  005037  031026                  CLR     @#X             ;THIS IS A READ COMMAND
 7066  014552  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
 7067  014556  031016                          CYL
 7068  014560  032716                          WCRC
 7069
 7070
 7071                                          ;THIS IS TO INSERT ERROR
 7072                                          ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
 7073                                          ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
 7074                                          ;THIS MOVE
 7075  014562  012737  177760  032736          MOV     #177760,@#DISK+2        ;FORCE ERROR ON BIT NUMBER 21 THRU 32
 7076                                                  ;SO ERROR POSITION REGISTER WILL SHOW
 7077                                                  ;22
 7078  014570  012737  010040  014736          MOV     #4128.,@#RHDB   ;INSERT POSITION REG.
 7079
 7080
 7081                                          ;THESE ARE REGULAR SETUPS
 7082  014576  004737  024470                  JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
 7083  014602  012777  177374  165014          MOV     #-256.,-4.,,@RHWC ;256. DATA 4 HEADER WORDS
 7084  014610  012777  003130  165010          MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
 7085  014616  112746  000000                  MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
 7086  014622  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
 7087  014630  012677  165002                  MOV     (SP)+,  @RHDST  ;TRACK/SECTOR IN RHDST
 7088  014634  012777  010000  165000          MOV     #FMT22,@RHOF ;16 BITS PER WORD
 7089                                                  ;ECC CORRECTION NOT INHIBIT
 7090                                                  ;BECAUSE ECC IS NOT GOING
 7091                                                  ;TO BE CHECKED
 7092  014642  005077  164776                  CLR     @RHCA           ;CYLINDER 0
 7093
 7094  014646  004737  024524                  JSR     PC,     @#CHECKT ;CHECK FOR DVA,RDY,MOL,DPR,DRY
 7095
 7096  014652  013711  002046                  MOV     @#REFOR,@R1     ;READ HEADER AND DATA=72
 7097  014656  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
 7098  014662  004737  030706                  JSR     PC,     @#COMMD ;READ HEADER AND DATA
                                                                      ;IF THERE ARE READ ERRORS THEN
                                                                      ;ECC WILL NOT BE CHECKED


                                          ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
                                          ;FROM THE "COMMD" ROUTINE THAT MEANS SECTOR GAP,
                                          ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
```

```
                                                ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
                                                ;DETECTED
                                                ;HEADER AND DATA ARE TO BE CHECKED,
                                                ;IN CHECKING READ DATA THE WRITE FROM BUFFER
                                                ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
                                                ;COMPARISONS ARE MADE

014666  005737  001774                          TST     @#ERFLG#                ;ANY ERRORS ALREADY THERE

014672  001103                                  BNE     TST22   ;BRANCH IF YES
 (1)
014674  004737  024170                          JSR     PC,@#PUTREG     ;SAVE REGISTERS
014700  022737  100000  001704                  CMP     #DCK,@#ER1      ;ONLY DATA CHECK ERROR SHOULD BE SET
014706  001401                                  BEQ     6$              ;BRANCH IF YES
014710  104032                                  ERROR   32              ;32 BIT ECC REGISTER SHOULD BE NON
                                                                        ;ZERO
                                                                        ;ONLY 11 OF THE 32 BITS CAN BE SEEN
                                                                        ;IN THE PATERN REGISTER
                                                                        ;DCK SHOULD BE SET IN RMER1
014712  013746  027126          6$:             MOV     @#GECC1,-(SP)   ;GET PATTERN REGISTER
014716  042716  174000                          BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
014722  022637  001734                          CMP     (SP)+,@#EC2     ;COMPARE PATTERN REGISTER
014726  001401                                  BEQ     7$              ;BRANCH IF GOOD
014730  104032                                  ERROR   32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT

014732  004037  027550          7$:             JSR     R0,@#ECORR      ;GO TO ECC CORRECTION PROCESS
014736  000000                  8$:             .WORD                   ;EXPECTED POSITION REG. WHEN CORRECTION
                                                                        ;IS COMPLETE



                                                ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
                                                ;DETECTED
014740  004737  024170                          JSR     PC,@#PUTREG     ;SAVE REGISTERS
014744  022737  100100  001704                  CMP     #DCK!ECH,@#ER1  ;WITH ERRORS INSERTED IN BIT POSITION 21
                                                                        ;THRU 32 HARD ERROR BIT SHOULD SET
014752  001401                                  BEQ     9$              ;BRANCH IF GOOD
014754  104036                                  ERROR   36              ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
                                                                        ;32 ECH SHOULD SET



014756  004737  024706          9$:             JSR     PC,@#CHECKE     ;CHECK DVA,DRY,RDY,DPR
014762  012700  002064                          MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
014766  012720  010000                          MOV     #@!FMT22,(R0)+  ;CYLINDER 0
014772  112746  000000                          MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
014776  112766  000000  000001                  MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
015004  012620                                  MOV     (SP)+,  (R0)+   ;GET TRACK/SECTOR IN BUFFER
015006  012720  000000                          MOV     #0,     (R0)+   ;KEY1 IN BUFFER
015012  012720  000000                          MOV     #0,     (R0)+   ;KEY2 IN BUFFER
015016  012701  000400                          MOV     #256.,  R1      ;DATA WORD COUNTER
015022  012702  000000                          MOV     #0,     R2      ;DATA
015026  010220                  10$:            MOV     R2,     (R0)+   ;DATA INTO BUFFER
015030  005301                                  DEC     R1              ;COUNT
015032  001375                                  BNE     10$             ;BRANCH IF 256 NOT DONE
```

```
                                        ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
                                        ;NOW THE INSERTED ERROR WILL BE PUT IN
015034  012737  177760  002076          MOV     #177760,@#WRFROM+<5*2>  ;INSERTED ERROR


015042  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
015046  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS


                                        ;NOW READ DATA BUFFER WILL BE CHECKED

015052  004037  025342                  JSR     R0,@#COMPAR     ;CHECK
015056  002064                          WRFROM                  ;GOOD BUFFER
015060  003130                          REINTO                  ;TEST BUFFER
015062  000404                          4*256.                  ;NUMBER OF WORDS CHECKED
015064  015072                          4$                      ;RETURN POINT FOR ERROR HEADER
015066  015076                          5$                      ;RETURN POINT FOR ERROR DATA

 (1)    015070  015102                  TST22                   ;RETURN FOR GOOD COMPARISON
 (1)
015072  104004                  4$:     ERROR   4               ;READ NEXT ERROR
015074  000207                          RTS     PC              ;RETURN TO "COMPAR"
015076  104005                  5$:     ERROR   5               ;WORD NOS 1 TO 4 ARE
                                                                ;HEADER WORDS
                                                                ;5 TO 260 ARE DATA WORDS
015100  000207                          RTS     PC              ;RETURN TO "COMPAR"


                                ;*******************************************************************
                                ;*TEST 22       WRITE ECC TEST 2

                                ;*      THIS IS A WRITE ECC TEST
                                ;*      WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
                                ;*      TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
                                ;*      OF ALL ONES.

                                ;*******************************************************************
 (2)    015102  000004          TST22:  SCOPE
015104  012706  001000                  MOV     @#STACK,SP      ;RESET STACK


 (1)    015110  012737  000022  004174  MOV     @#TTNO,@#TSTNM  ;THIS SAVES TEST NUMBER
 (1)
015116  012700  032636                  MOV     @#SECGAP,R0     ;POINTER
015122  012701  000400                  MOV     @#384.,R1       ;COUNTER
015126  005020                  1$:     CLR     (R0)+           ;CLEAR SIMULATED DISK AREA
015130  005301                          DEC     R1
015132  001375                          BNE     1$
```

```
 7207  015134  004767  007330                  JSR     PC,CLDISK                       ;THIS IS USED TO SET GENERAL REGISTERS
 7208
 7209                                          ;THESE ARE FOR ECC TEST ONLY
 7210
 7211  015140  012737  177777  002012          MOV     #-1,@#TSECC             ;THIS IS AN ECC TEST
 7212  015146  005037  027140                  CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
 7213  015152  013737  027134  027136          MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
 7214  015160  013737  027142  027150          MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
 7215  015166  005037  027126                  CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
 7216  015172  005037  027130                  CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
 7217  015176  005037  027144                  CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
 7218  015202  005037  027146                  CLR     @#ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
 7219
 7220
 7221
 7222
 7223                                          ;THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
 7224
 7225  015206  012737  010000  034122          MOV     #FMT22,@#WCYL   ;FORMAT22=16BIT WORDS AND
 7226                                                                  ;CYLINDER 0
 7227  015214  012737  000001  034124          MOV     #1,@#WSECTR     ;TRACK=0, SECTOR=1
 7228  015222  005037  034126                  CLR     @#WKEY1         ;KEY1=0
 7229  015226  005037  034130                  CLR     @#WKEY2         ;KEY2=0
 7230  015232  012737  000400  034162          MOV     #256.,@#FNWORD  ;256 DATA WORDS
 7231  015240  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
 7232  015244  034122                          WCYL
 7233  015246  034132                          GCRC
 7234
 7235                                          ;THESE ARE REGULAR SETUPS
 7236
 7237  015250  012777  177374  164346          MOV     #-260.,@RHWC    ;256 DATA WORDS 4 HEADER WORDS
 7238  015256  012700  002064                  MOV     #WRFROM,R0      ;THESE TWO INSTRUCTIONS GETS
 7239  015262  010077  164340                  MOV     R0,@RHBA        ;ADDR. OF WRFROM INTO R0 AND
 7240                                                                  ;BUS ADDRESS REGISTER
 7241  015266  012720  010000                  MOV     #FMT22,(R0)+    ;FORMAT=16 BIT WORDS
 7242                                                                  ;CYLINDER=0
 7243  015272  012720  000001          28$:    MOV     #1,(R0)+        ;TRACK=0, SECTOR=1, KEYS=0
 7244  015276  005020                          CLR     (R0)+           ;KEY1=0
 7245  015300  005020                          CLR     (R0)+           ;KEY2=0
 7246  015302  012705  000400                  MOV     #256.,R5        ;COUNTER
 7247  015306  012720  177777          38$:    MOV     #-1,(R0)+       ;MOVE ALL ONES FOR DATA
 7248  015312  005305                          DEC     R5
 7249  015314  001374                          BNE     3$              ;BRANCH IF DATA NOT COMPLETE
 7250  015316  012777  000001  164312          MOV     #1,@RHDST       ;TRACK=0 SECTOR=1
 7251
 7252
  (1)
  (1)  015324  004767  007174                  JSR     PC,CHECKT               ;CHECK DVA, RDY, DPR, DRY
  (1)
 7253
 7254  015330  013711  002042                  MOV     @#WRIFOR,@R1    ;GET READY FOR WRITE HEADER AND
 7255                                                                  ;DATA WITH 62 IN RHCS1
 7256  015334  005037  001774                  CLR     @#ERFLG#        ;CLEAR ERROR FLAG
 7257  015340  012777  010000  164274          MOV     #FMT22,@RHOF    ;FORMAT BIT=1 (16 BIT WORDS)
```

```
 7258  015346  005077  164272                  CLR     @#RHCA          ;CYLINDER =0
 7259  015352  004737  033776                  JSR     PC,@#COMWHD     ;WRITE HEADER AND DATA
 7260
 7261                                          ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 7262                                          ;FROM THE "COMWHD" ROUTINE THAT MEANS ALL HEADER ON DISK
 7263                                          ;IS GOOD IE, ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
 7264                                          ;ALL ONES AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
 7265                                          ;THEY ARE ALL ZEROS
 7266
 7267  015356  005737  001774                  TST     @#ERFLG#                ;HAS ANY ERRORS OCCURED?
 7268                                                                          ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
 7269
  (1)  015362  001053                          BNE     TST2$   ;;BRANCH IF YES
  (1)
 7270
 7271
 7272                                  ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
 7273  015364  023737  027126  033734          CMP     @#GECC1,@#WECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
 7274  015372  001402                          BEQ     6$              ;BRANCH IF GOOD
 7275  015374  104031                          ERROR   31              ;LOW ORDER ECC IN ERROR
 7276  015376  000405                          BR      7$              ;BRANCH TO CONTINUE
 7277  015400  023737  027130  033736  6$:     CMP     @#GECC2,@#WECC2;COMPARE SOFTWARE ECC WITH HARDWARE ECC
 7278  015406  001401                          BEQ     7$              ;BRANCH IF GOOD
 7279  015410  104031                          ERROR   31              ;HIGH ORDER ECC IN ERROR
 7280
 7281
 7282  015412  004737  024706          7$:     JSR     PC,@#CHECKE     ;CHECK DVA,RDY,DRY,DPR
 7283
 7284
 7285
 7286
 7287                                  ;FILL "REINTO" BUFFER WITH EXPECTED DATA
 7288  015416  004037  024406                  JSR     R0,@#CLAREA     ;FILL REINTO BUFFER
 7289  015422  003130                          REINTO                  ;FROM
 7290  015424  004126                          REINTO+<255.*2>                 ;TO
 7291  015426  177777                          .WORD   -1              ;DATA
 7292
 7293  015430  013737  027126  004130          MOV     @#GECC1,@#REINTO+<256.*2>;FILL ECC1
 7294  015436  013737  027130  004132          MOV     @#GECC2,@#REINTO+<257.*2>;FILL ECC2
 7295  015444  004037  024406                  JSR     R0,@#CLAREA     ;FILL REST
 7296  015450  004134                          REINTO+<258.*2>         ;FROM
 7297  015452  004170                          REINTO+<272.*2>         ;TO
 7298  015454  000000                          0                       ;DATA
 7299
 7300
 7301  015456  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
 7302
 7303
 7304                                  ;NOW COMPARE "DISK" BUFFER WITH "REINTO"
 7305  015462  004037  025342                  JSR     R0,@#COMPAR     ;CHECK
 7306  015466  003130                          REINTO                  ;GOOD BUFFER
 7307  015470  032734                          DISK                    ;TEST BUFFER
 7308  015472  000402                          256.                    ;NUMBER OF WORDS CHECKED
 7309  015474  015502                          4$                      ;RETURN POINT FOR ERROR HEADER
```

```
                                        50                      ;RETURN POINT FOR ERROR DATA
                                        TST23                   ;RETURN FOR GOOD COMPARISON
                                48$:    ERROR   7               ;READ ERROR 10 NEXT
                                        RTS     PC              ;RETURN TO COMPARE
                                58$:    ERROR   10              ;WORD NOS 1 TO 256 ARE
                                                                ;DATA WORDS
                                                                ;WORD NOS 257 AND 258
                                                                ;ARE ECC WHICH ARE CHECKED
                                                                ;WORD NOS 259
                                                                ;IS DATA GAP
                                                                ;WORD NOS 260 TO 273
                                                                ;ARE TOLERANCE GAP
                                        RTS     PC              ;RETURN TO COMPARE


                        ;*****************************************************************
                        ;*TEST 23       READ ECC ENABLED 2A
                        ;*      THIS IS AN ECC READ DATA TEST
                        ;*      ERROR CORRECTION IS ENABLED
                        ;*      NO ERROR IS INSERTED
                        ;*      GOOD DATA USED IS 256 WORDS OF 177777
                        ;*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
                        ;*      TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
                        ;*****************************************************************
                TST23:  SCOPE
                        MOV     #STACK,SP       ;RESET STACK

                        MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER


                ;       SETUP FOR WHAT IS TO BE READ
                ;       HEADER CRC IS RESTORED FROM A SUBROUTINE

                        MOV     #-1,    -(SP)   ;DATA TO BE READ
                        MOV     #256.,  R5      ;COUNTER
                        MOV     #DISK,  R0      ;START OF SIMULATED DISK DATA
                18$:    MOV     (SP),   (R0)+   ;MOVE IN DATA ON TO SIMULATED DISK
                        DEC     R5              ;COUNT
                        BNE     18$             ;BRANCH IF 256 NOT COMPLETE
                        TST     (SP)+           ;UNDO -(SP)
                        CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
                        MOV     #15.,   R5      ;15 DATA GAP
                                                ;14 TOLERANCE GAP
                28$:    CLR     (R0)+           ;CLEAR DATA GAP, AND
```

```
 7359  015562  005305                          DEC     R5              ;TOLERANCE GAP
 7360  015564  001375                          BNE     2$              ;BRANCH IF NOT COMPLETE
 7361
 7362
 7363  015566  004737  027722                  JSR     PC,$$FILLEC     ;INSERT THE TWO ECC WORDS ON THE DISK
 7364                                                                  ;IN THE CORRECT PLACE
 7365
 7366                                          ;THESE ARE FOR ECC TEST ONLY
 7367
 7360  015572  012737  177777  002012          MOV     #-1,$$TSECC             ;THIS IS AN ECC TEST
 7369  015600  005037  027140                  CLR     $$POSITI        ;CLEAR ERROR POSITION COUNTER
 7370  015604  013737  027134  027136          MOV     $$NCODE,$$NCOUNT        ;TEMPORARY N-CODE COUNTER
 7371  015612  013737  027142  027150          MOV     $$HARDER,$$HADTMP       ;TEMPORARY HARD ERROR COUNTER
 7372  015620  005037  027126                  CLR     $$GECC1         ;ECC LOW ORDER TO BE GENERATED
 7373  015624  005037  027130                  CLR     $$GECC2         ;ECC HIGH ORDER TO BE GENERATED
 7374  015630  005037  027144                  CLR     $$DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
 7375  015634  005037  027146                  CLR     $$ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
 7376
 7377
 7378                                          ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
 7379
 7380  015640  012737  010000  031016          MOV     #FMT22,$$CYL    ;16 BITS PER WORD
 7381                                                                  ;CYLINDER 0, FORMAT 16 BITS
 7382  015646  112737  000000  031021          MOVB    #0,     $$SECOTR+1 ;TRACK 0
 7383  015654  112737  000000  031020          MOVB    #0,     $$SECOTR ;SECTOR 0
 7384  015662  012737  000000  031022          MOV     #0,     $$KEY1  ;KEY1=0
 7385  015670  012737  000000  031024          MOV     #0,     $$KEY2  ;KEY2=0
 7386  015676  012737  000400  031076          MOV     #256.,  $$DAWORD ;NO. OF DATA WORDS
 7387  015704  005037  031026                  CLR     $$X             ;THIS IS A READ COMMAND
 7388  015710  004537  025646                  JSR     R5,$$CRC        ;GO TO CALCULATE CRC
 7389  015714  031016                          CYL
 7390  015716  032716                          MCRC
 7391
 7392
 7393
 7394
 7395                                          ;THESE ARE REGULAR SETUPS
 7396  015720  004737  024470                  JSR     PC,$$CLDISK     ;SETUP GENERAL REGISTERS
 7397  015724  012777  177374  163672          MOV     #-256.,-4.,@RWWC ;256. DATA 4 HEADER WORDS
 7398  015732  012777  003130  163666          MOV     #READ0,@RWBA    ;STARTING ADDRESS OF READ BUFFER
 7399  015740  112746  000000                  MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
 7400  015744  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
 7401  015752  012677  163660                  MOV     (SP)+,  @RWDST  ;TRACK/SECTOR IN RWDST
 7402  015756  012777  010000  163656          MOV     #FMT22,@RWCOF ;16 BITS PER WORD
 7403                                                                  ;ECC CORRECTION NOT INHIBIT
 7404                                                                  ;BECAUSE ECC IS NOT GOING
 7405                                                                  ;TO BE CHECKED
 7406  015764  005077  163654                  CLR     @RWCA           ;CYLINDER 0
 7407
 7408  015770  004737  024524                  JSR     PC,     $$CHECKT ;CHECK FOR DVA,RDY,NOL,DPR,DRY
 7409
 7410  015774  013711  002046                  MOV     $$REFOR,@R1     ;READ HEADER AND DATA=72
 7411  016000  005037  001774                  CLR     $$ERFLG$                ;CLEAR ERROR FLAG
 7412  016004  004737  030706                  JSR     PC,     $$COMMD ;READ HEADER AND DATA
```

```
                                                        ;IF THERE ARE READ ERRORS THEN
                                                        ;ECC WILL NOT BE CHECKED



                                        ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
                                        ;FROM THE "COMMD" ROUTINE THAT MEANS SECTOR GAP,
                                        ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
                                        ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
                                        ;DETECTED
                                        ;HEADER AND DATA ARE TO BE CHECKED.
                                        ;IN CHECKING READ DATA THE WRITE FROM BUFFER
                                        ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
                                        ;COMPARISONS ARE MADE

016010  005737  001774                  TST     @#ERFLG#                ;ANY ERRORS ALREADY THERE

 (1)    016014  001077                  BNE     TST24   ;BRANCH IF YES
 (1)
016016  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
016022  005737  001704                  TST     @#ER1           ;NO ERRORS SHOULD BE SET
016026  001401                          BEQ     .+4             ;BRANCH IF NO ERRORS SET
016030  104032                          ERROR   32              ;32 BIT ECC REGISTER SHOULD BE ZERO
                                                                ;ONLY 11 OF THE 32 BITS CAN BE SEEN
                                                                ;IN THE PATTERN REGISTER
                                                                ;DCK SHOULD BE SET IN RMER1
016032  013746  027126          68$:    MOV     @#GECC1,-(SP)   ;GET PATTERN REGISTER
016036  042716  174000                  BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
016042  022637  001734                  CMP     (SP)+,@#EC2     ;COMPARE PATTERN REGISTER
016046  001401                          BEQ     .+4             ;BRANCH IF GOOD
016050  104032                          ERROR   32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT



                                        ;ADD 16 MAINTENANCE CLOCKS TO
                                        ;BRING EBL DOWN

016052  012700  000020          78$:    MOV     #16.,R0         ;COUNTER
016056  052777  000002  163566  88$:    BIS     @#NCLK,@RHMR    ;SET CLOCK
016064  042777  000002  163560          BIC     @#NCLK,@RHMR    ;CLEAR CLOCK
016072  005300                          DEC     R0              ;COUNT
016074  001370                          BNE     .-6             ;BRANCH IF 16 CLOCKS NOT DONE
016076  004737  024706                  JSR     PC,@#CHECKE     ;CHECK DVA,DRY,RDY,DPR
016102  012700  002064                  MOV     @#WRFROM,R0     ;GETTING READY TO FILL EXPECTED DATA
016106  012720  010000                  MOV     @#BIFMT22,(R0)+ ;CYLINDER 0
016112  112746  000000                  MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
016116  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
016124  012620                          MOV     (SP)+,  (R0)+   ;GET TRACK/SECTOR IN BUFFER
016126  012720  000000                  MOV     #0,     (R0)+   ;KEY1 IN BUFFER
016132  012720  000000                  MOV     #0,     (R0)+   ;KEY2 IN BUFFER
016136  012701  000400                  MOV     #256.,  R1      ;DATA WORD COUNTER
016142  012702  177777                  MOV     #-1,    R2      ;DATA
016146  010220                  38$:    MOV     R2,     (R0)+   ;DATA INTO BUFFER
016150  005301                          DEC     R1              ;COUNT
```

```
 7465  016152  001375                        BNE     38              ;BRANCH IF 256 NOT DONE
 7466
 7467
 7468
 7469
 7470  016154  005037  001774                CLR     @#ERFLG#                ;CLEAR ERROR FLAG
 7471
 7472  016160  004737  024170                JSR     PC,@#PUTREG     ;SAVE REGISTERS
 7473
 7474                                        ;NOW READ DATA BUFFER WILL BE CHECKED
 7475
 7476  016164  004037  025342                JSR     R0,@#COMPAR     ;CHECK
 7477  016170  002064                        WRFROM                  ;GOOD BUFFER
 7478  016172  003130                        REINTO                  ;TEST BUFFER
 7479  016174  000404                        4*256.                  ;NUMBER OF WORDS CHECKED
 7480  016176  016204                        48                      ;RETURN POINT FOR ERROR HEADER
 7481  016200  016210                        58                      ;RETURN POINT FOR ERROR DATA
 7482
  (1)  016202  016214                        T8T24                   ;RETURN FOR GOOD COMPARISON
  (1)
 7483  016204  104004                48$:    ERROR   4               ;READ NEXT ERROR
 7484  016206  000207                        RTS     PC              ;RETURN TO "COMPAR"
 7485  016210  104005                58$:    ERROR   5               ;WORD NOS 1 TO 4 ARE
 7486                                                                ;HEADER WORDS
 7487                                                                ;5 TO 260 ARE DATA WORDS
 7488  016212  000207                        RTS     PC              ;RETURN TO "COMPAR"
 7489
 7490
 7491
 7492
 7503
 7504                                        ;*****************************************************************
  (3)                                       ;*TEST 24        READ ECC ENABLED 2B
  (4)
  (4)                                       ;*      THIS IS AN ECC READ DATA TEST
  (4)                                       ;*      ERROR CORRECTION IS ENABLED
  (4)                                       ;*      A CORRECTABLE ERROR IS INSERTED IN BIT POSITION 32
  (4)                                       ;*      GOOD DATA USED IS 256 WORDS OF 177777
  (4)                                       ;*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
  (4)                                       ;*      TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
  (4)
  (3)                                       ;*****************************************************************
  (2)  016214  000004                T8T24:  SCOPE
 7505  016216  012706  001000                MOV     @#STACK,SP      ;RESET STACK
 7506
 7507
  (1)  016222  012737  000024  004174        MOV     @#TINO,@#T8TNM  ;THIS SAVES TEST NUMBER
  (1)
 7508
 7509
 7510                                        ;       SETUP FOR WHAT IS TO BE READ
 7511                                        ;       HEADER CRC IS RESTORED FROM A SUBROUTINE
 7512
 7513  016230  012746  177777                MOV     #-1,    -(SP)   ;DATA TO BE READ
```

M7

```
7514  016234  012705  000400                  MOV     #256.,  R5      ;COUNTER
7515  016240  012700  032734                  MOV     #DISK,  R0      ;START OF SIMULATED DISK DATA
7516  016244  011620                  18:     MOV     (SP),   (R0)+   ;MOVE IN DATA ON TO SIMULATED DISK
7517  016246  005305                          DEC     R5              ;COUNT
7518  016250  001375                          BNE     1B              ;BRANCH IF 256 NOT COMPLETE
7519  016252  005726                          TST     (SP)+           ;UNDO -(SP)
7520  016254  022020                          CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
7521  016256  012705  000017                  MOV     #15.,   R5      ;1 DATA GAP
7522                                                                   ;14 TOLERANCE GAP
7523  016262  005020                  28:     CLR     (R0)+           ;CLEAR DATA GAP, AND
7524  016264  005305                          DEC     R5              ;TOLERANCE GAP
7525  016266  001375                          BNE     2B              ;BRANCH IF NOT COMPLETE
7526
7527
7528  016270  004737  027722                  JSR     PC,@#FILLEC     ;INSERT ECC IN PROPER PLACE ON DISK
7529
7530
7531
7532                                          ;THESE ARE FOR ECC TEST ONLY
7533
7534  016274  012737  177777  002012          MOV     #-1,@#TSECC             ;THIS IS AN ECC TEST
7535  016302  005037  027140                  CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
7536  016306  013737  027134  027136          MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
7537  016314  013737  027142  027150          MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
7538  016322  005037  027126                  CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
7539  016326  005037  027130                  CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
7540  016332  005037  027144                  CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
7541  016336  005037  027146                  CLR     @#ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
7542
7543
7544                                          ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
7545
7546  016342  012737  010000  031016          MOV     #FMT22,@#CYL    ;16 BITS PER WORD
7547                                                                   ;CYLINDER 0, FORMAT 16 BITS
7548  016350  112737  000000  031021          MOVB    #0,     @#SECOTR+1 ;TRACK 0
7549  016356  112737  000000  031020          MOVB    #0,     @#SECOTR ;SECTOR 0
7550  016364  012737  000000  031022          MOV     #0,     @#KEY1  ;KEY1=0
7551  016372  012737  000000  031024          MOV     #0,     @#KEY2  ;KEY2=0
7552  016400  012737  000400  031076          MOV     #256.,  @#DAWORD ;NO. OF DATA WORDS
7553  016406  005037  031026                  CLR     @#X             ;THIS IS A READ COMMAND
7554  016412  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
7555  016416  031016                          CYL
7556  016420  032716                          WCRC
7557
7558
7559                                          ;THIS IS TO INSERT ERROR
7560                                          ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
7561                                          ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
7562                                          ;THIS MOVE
7563  016422  012737  077777  032736          MOV     #77777,@#DISK+2 ;FORCE ERROR ON BIT NUMBER 32
7564                                                                   ;SO ERROR POSITION REGISTER WILL SHOW
7565                                                                   ;22
7566  016430  012737  000026  016576          MOV     #22.,@#88       ;INSERT POSITION REG,
7567
```

```
7568
7569                                                    ;THESE ARE REGULAR SETUPS
7570  016436  004737  024470                            JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
7571  016442  012777  177374  163154                    MOV     #-256.,-4.,,@RHWC ;256. DATA 4 HEADER WORDS
7572  016450  012777  003130  163150                    MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
7573  016456  112746  000000                            MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
7574  016462  112766  000000  000001                    MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
7575  016470  012677  163142                            MOV     (SP)+,  @RHDST  ;TRACK/SECTOR IN RHDST
7576  016474  012777  010000  163140                    MOV     #FMT22,@RHOF ;16 BITS PER WORD
7577                                                                    ;ECC CORRECTION NOT INHIBIT
7578                                                                    ;BECAUSE ECC IS NOT GOING
7579                                                                    ;TO BE CHECKED
7580  016502  005077  163136                            CLR     @RHCA           ;CYLINDER 0
7581
7582  016506  004737  024524                            JSR     PC,     @#CHECKT ;CHECK FOR DVA,RDY,NOL,DPR,DRY
7583
7584  016512  013711  002046                            MOV     @#REFOR,@R1     ;READ HEADER AND DATA=72
7585  016516  005037  001774                            CLR     @#ERFLG#                ;CLEAR ERROR FLAG
7586  016522  004737  030706                            JSR     PC,     @#COMMD ;READ HEADER AND DATA
7587                                                                    ;IF THERE ARE READ ERRORS THEN
7588                                                                    ;ECC WILL NOT BE CHECKED
7589
7590
7591                                                    ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
7592                                                    ;FROM THE "COMMD" ROUTINE THAT MEANS SECTOR GAP,
7593                                                    ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
7594                                                    ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
7595                                                    ;DETECTED
7596                                                    ;HEADER AND DATA ARE TO BE CHECKED,
7597                                                    ;IN CHECKING READ DATA THE WRITE FROM BUFFER
7598                                                    ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
7599                                                    ;COMPARISONS ARE MADE
7600
7601  016526  005737  001774                            TST     @#ERFLG#                ;ANY ERRORS ALREADY THERE
7602
 (1)  016532  001074                                    BNE     TST2B   ;BRANCH IF YES
 (1)
7603  016534  004737  024170                            JSR     PC,@#PUTREG     ;SAVE REGISTERS
7604  016540  022737  100000  001764                    CMP     #DCK,@#ER1      ;ONLY DATA CHECK ERROR SHOULD BE SET
7605  016546  001401                                    BEQ     6$              ;BRANCH IF YES
7606  016550  104032                                    ERROR   32              ;32 BIT ECC REGISTER SHOULD BE NON
7607                                                                    ;ZERO
7608                                                                    ;ONLY 11 OF THE 32 BITS CAN BE SEEN
7609                                                                    ;IN THE PATERN REGISTER
7610                                                                    ;DCK SHOULD BE SET IN ER1
7611  016552  013746  027126            6$:             MOV     @#GECC1,-(SP)   ;GET PATTERN REGISTER
7612  016556  042716  174000                            BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
7613  016562  022637  001734                            CMP     (SP)+,@#EC2     ;COMPARE PATTERN REGISTER
7614  016566  001401                                    BEQ     7$              ;BRANCH IF GOOD
7615  016570  104032                                    ERROR   32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
7616
7617  016572  004037  027550            7$:             JSR     R0,@#ECORR      ;GO TO ECC CORRECTION PROCESS
7618  016576  000026                    8$:             22.                     ;EXPECTED POSITION REG, WHEN CORRECTION
7619                                                                    ;IS COMPLETE
```

```
                                    ;CHECK DVA,DRY,RDY,DPR
016600  004737  024706              JSR     PC,@#CHECKE
016604  012700  002064              MOV     #WRFROM,R0          ;GETTING READY TO FILL EXPECTED DATA
016610  012720  010000              MOV     #01FMT22,(R0)+      ;CYLINDER 0
016614  112746  000000              MOVB    #0,     -(SP)       ;IN LOWER BYTE GET SECTOR
016620  1127e6  000000  000001      MOVB    #0,     1(SP)       ;GET TRACK IN HIGHER BYTE
016626  012620                      MOV     (SP)+,  (R0)+       ;GET TRACK/SECTOR IN BUFFER
016630  012720  000000              MOV     #0,     (R0)+       ;KEY1 IN BUFFER
016634  012720  000000              MOV     #0,     (R0)+       ;KEY2 IN BUFFER
016640  012701  000400              MOV     #256.,  R1          ;DATA WORD COUNTER
016644  012702  177777              MOV     #-1,    R2          ;DATA
016650  010220              38:     MOV     R2,     (R0)+       ;DATA INTO BUFFER
016652  005301                      DEC     R1                  ;COUNT
016654  001375                      BNE     38                  ;BRANCH IF 256 NOT DONE

                                    ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
                                    ;NOW THE INSERTED ERROR WILL BE PUT IN
016656  012737  077777  002076      MOV     #77777,@#WRFROM+<5*2>   ;INSERTED ERROR

016664  004737  024170              JSR     PC,@#PUTREG         ;SAVE REGISTERS

016670  005037  001774              CLR     @#ERFLG             ;CLEAR ERROR FLAG


                                    ;NOW READ DATA BUFFER WILL BE CHECKED

016674  004037  025342              JSR     R0,@#COMPAR         ;CHECK
016700  002064                      WRFROM                      ;GOOD BUFFER
016702  003130                      REINTO                      ;TEST BUFFER
016704  000404                      4*256.                      ;NUMBER OF WORDS CHECKED
016706  016714                      48                          ;RETURN POINT FOR ERROR HEADER
016710  016720                      58                          ;RETURN POINT FOR ERROR DATA

016712  016724                      TST2S                       ;RETURN FOR GOOD COMPARISON

016714  104004              48:     ERROR   4                   ;READ NEXT ERROR
016716  000207                      RTS     PC                  ;RETURN TO "COMPAR"
016720  104005              58:     ERROR   5                   ;WORD NOS 1 TO 4 ARE
                                                                ;HEADER WORDS
                                                                ;5 TO 260 ARE DATA WORDS
016722  000207                      RTS     PC                  ;RETURN TO "COMPAR"


                                    ;*****************************************************************
                                    ;*TEST 2S        READ ECC ENABLED 2C

                                    ;*      THIS IS AN ECC READ DATA TEST
                                    ;*      ERROR CORRECTION IS ENABLED
                                    ;*      A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 32 AND 21
                                    ;*      GOOD DATA USED IS 256 WORDS OF 177777
                                    ;*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
```

```
 (4)                                    ;*      TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
 (4)
 (3)                                    ;*****************************************************************
 (2)  016724  000004                    TST25:  SCOPE
7675  016726  012706  001000                     MOV     #STACK,SP       ;RESET STACK
7676
7677
 (1)  016732  012737  000025  004174            MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
 (1)
7678
7679
7680                                    ;       SETUP FOR WHAT IS TO BE READ
7681                                    ;       HEADER CRC IS RESTORED FROM A SUBROUTINE
7682
7683  016740  012746  177777                     MOV     #-1,    -(SP)   ;DATA TO BE READ
7684  016744  012705  000400                     MOV     #256.,  R5      ;COUNTER
7685  016750  012700  032734                     MOV     #DISK,  R0      ;START OF SIMULATED DISK DATA
7686  016754  011620                    10$:    MOV     (SP),   (R0)+   ;MOVE IN DATA ON TO SIMULATED DISK
7687  016756  005305                            DEC     R5              ;COUNT
7688  016760  001375                            BNE     10$             ;BRANCH IF 256 NOT COMPLETE
7689  016762  005726                            TST     (SP)+           ;UNDO -(SP)
7690  016764  022020                            CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
7691  016766  012705  000017                     MOV     #15.,   R5      ;1 DATA GAP
7692                                                                    ;14 TOLERANCE GAP
7693  016772  005020                    20$:    CLR     (R0)+           ;CLEAR DATA GAP, AND
7694  016774  005305                            DEC     R5              ;TOLERANCE GAP
7695  016776  001375                            BNE     20$             ;BRANCH IF NOT COMPLETE
7696
7697
7698  017000  004737  027722                     JSR     PC,@#FILLEC     ;INSERT THE TWO ECC WORDS ON THE DISK
7699                                                                    ;IN THE CORRECT PLACE
7700
7701                                    ;THESE ARE FOR ECC TEST ONLY
7702
7703  017004  012737  177777  002012            MOV     #-1,@#TSECC             ;THIS IS AN ECC TEST
7704  017012  005037  027140                     CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
7705  017016  013737  027134  027136            MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
7706  017024  013737  027142  027150            MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
7707  017032  005037  027126                     CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
7708  017036  005037  027130                     CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
7709  017042  005037  027144                     CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
7710  017046  005037  027146                     CLR     @#ECODE         ;CLEAR LEADING ZEROS CLOCK COUNT
7711
7712
7713                                    ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
7714
7715  017052  012737  010000  031016            MOV     #PNT22,@#CYL    ;16 BITS PER WORD
7716                                                                    ;CYLINDER 0, FORMAT 16 BITS
7717  017060  112737  000000  031021            MOVB    #0,     @#SECOTR+1 ;TRACK 0
7718  017066  112737  000000  031020            MOVB    #0,     @#SECOTR ;SECTOR 0
7719  017074  012737  000000  031022            MOV     #0,     @#KEY1  ;KEY1=0
7720  017102  012737  000000  031024            MOV     #0,     @#KEY2  ;KEY2=0
7721  017110  012737  000400  031076            MOV     #256.,  @#DAWORD ;NO. OF DATA WORDS
7722  017116  005037  031026                     CLR     @#X             ;THIS IS A READ COMMAND
```

```
 7723  017122  004537  025646                JSR     R5,@#CRC        ;GO TO CALCULATE CRC
 7724  017126  031016                        CYL
 7725  017130  032716                        WCRC
 7726
 7727
 7728                                        ;THIS IS TO INSERT ERROR
 7729                                        ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
 7730                                        ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
 7731                                        ;THIS MOVE
 7732  017132  012737  077757  032736        MOV     #77757,@#DISK+2         ;FORCE ERROR ON BIT NUMBER 32 AND 21
 7733                                                                        ;SO ERROR POSITION REGISTER WILL SHOW
 7734                                                                        ;22
 7735  017140  012737  010040  017306        MOV     #4128.,@#RHOF           ;INSERT POSITION REG.
 7736
 7737
 7738                                        ;THESE ARE REGULAR SETUPS
 7739  017146  004737  024470                JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
 7740  017152  012777  177374  162444        MOV     #-256.,-4.,@RHWC ;256. DATA 4 HEADER WORDS
 7741  017160  012777  003130  162440        MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
 7742  017166  112746  000000                MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
 7743  017172  112766  000000  000001        MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
 7744  017200  012677  162432                MOV     (SP)+,  @RHDST  ;TRACK/SECTOR IN RHDST
 7745  017204  012777  010000  162430        MOV     #FMT22,@RHOF ;16 BITS PER WORD
 7746                                                                ;ECC CORRECTION NOT INHIBIT
 7747                                                                ;BECAUSE ECC IS NOT GOING
 7748                                                                ;TO BE CHECKED
 7749  017212  005077  162426                CLR     @RHCA           ;CYLINDER 0
 7750
 7751  017216  004737  024524                JSR     PC,     @#CHECKT ;CHECK FOR DVA,RDY,MOL,DPR,DRY
 7752
 7753  017222  013711  002046                MOV     @#REFOR,@R1     ;READ HEADER AND DATA=72
 7754  017226  005037  001774                CLR     @#ERFLG#                ;CLEAR ERROR FLAG
 7755  017232  004737  030706                JSR     PC,     @#COMND ;READ HEADER AND DATA
                                                                        ;IF THERE ARE READ ERRORS THEN
 7757                                                                   ;ECC WILL NOT BE CHECKED
 7758
 7759
 7760                                        ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 7761                                        ;FROM THE "COMND" ROUTINE THAT MEANS SECTOR GAP,
 7762                                        ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
 7763                                        ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
 7764                                        ;DETECTED
 7765                                        ;HEADER AND DATA ARE TO BE CHECKED,
 7766                                        ;IN CHECKING READ DATA THE WRITE FROM BUFFER
 7767                                        ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
 7768                                        ;COMPARISONS ARE MADE
 7769
 7770  017236  005737  001774                TST     @#ERFLG#                ;ANY ERRORS ALREADY THERE
 7771
  (1)  017242  001103                        BNE     TST26   ;BRANCH IF YES
  (1)
 7772  017244  004737  024170                JSR     PC,@#PUTREG     ;SAVE REGISTERS
 7773  017250  022737  100000  001704        CMP     @DCK,@#ER1      ;ONLY DATA CHECK ERROR SHOULD BE SET
 7774  017256  001401                        BEQ     6$              ;BRANCH IF YES
```

```
 7775  017260  104032                          ERROR   32              ;32 BIT ECC REGISTER SHOULD BE NON
 7776                                                                  ;ZERO
 7777                                                                  ;ONLY 11 OF THE 32 BITS CAN BE SEEN
 7778                                                                  ;IN THE PATERN REGISTER
 7779                                                                  ;DCK SHOULD BE SET IN RHER1
 7780  017262  013746  027126          68$:   MOV     @#GECC1,-(SP)   ;GET PATTERN REGISTER
 7781  017266  042716  174000                 BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
 7782  017272  022637  001734                 CMP     (SP)+,@#EC2     ;COMPARE PATTERN REGISTER
 7783  017276  001401                          BEQ     7$              ;BRANCH IF GOOD
 7784  017300  104032                          ERROR   32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
 7785
 7786  017302  004037  027550          70$:   JSR     R0,@#ECORR      ;GO TO ECC CORRECTION PROCESS
 7787  017306  000000                   88$:   .WORD                   ;EXPECTED POSITION REG, WHEN CORRECTION
 7788                                                                  ;IS COMPLETE
 7789
 7790  017310  004737  024170                 JSR     PC,@#PUTREG     ;SAVE REGISTERS
 7791
 7792
 7793  017314  022737  100100  001704         CMP     #DCK!ECH,@#ER1  ;WITH ERRORS INSERTED IN BIT POSITION 21
 7794                                                                  ;AND 32 HARD ERROR BIT SHOULD SET
 7795  017322  001401                          BEQ     9$              ;BRANCH IF GOOD
 7796  017324  104036                          ERROR   36              ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
 7797                                                                  ;32 NCE SHOULD SET
 7798
 7799
 7800
 7801
 7802  017326  004737  024706          90$:   JSR     PC,@#CHECKE     ;CHECK DVA,DRY,RDY,DPR
 7803  017332  012700  002064                 MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
 7804  017336  012720  010000                 MOV     #0!FMT22,(R0)+  ;CYLINDER 0
 7805  017342  112746  000000                 MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
 7806  017346  112766  000000  000001         MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
 7807  017354  012620                          MOV     (SP)+,  (R0)+   ;GET TRACK/SECTOR IN BUFFER
 7808  017356  012720  000000                 MOV     #0,     (R0)+   ;KEY1 IN BUFFER
 7809  017362  012720  000000                 MOV     #0,     (R0)+   ;KEY2 IN BUFFER
 7810  017366  012701  000400                 MOV     #256.,  R1      ;DATA WORD COUNTER
 7811  017372  012702  177777                 MOV     #-1,    R2      ;DATA
 7812  017376  010220                   38$:   MOV     R2,     (R0)+   ;DATA INTO BUFFER
 7813  017400  005301                          DEC     R1              ;COUNT
 7814  017402  001375                          BNE     3$              ;BRANCH IF 256 NOT DONE
 7815
 7816                                         ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
 7817                                         ;NOW THE INSERTED ERROR WILL BE PUT IN
 7818  017404  012737  077757  002076         MOV     #77757,@#WRFROM+<5*2>   ;INSERTED ERROR
 7819
 7820
 7821  017412  004737  024170                 JSR     PC,@#PUTREG     ;SAVE REGISTERS
 7822
 7823  017416  005037  001774                 CLR     @#ERFLG         ;CLEAR ERROR FLAG
 7824
 7825
 7826                                         ;NOW READ DATA BUFFER WILL BE CHECKED
 7827
 7828  017422  004037  025342                 JSR     R0,@#COMPAR     ;CHECK
```

```
  7829  017426  002064                          WRFROM                  ;GOOD BUFFER
  7830  017430  003130                          REINTO                  ;TEST BUFFER
  7831  017432  000404                          4*256.                  ;NUMBER OF WORDS CHECKED
  7832  017434  017442                          48                      ;RETURN POINT FOR ERROR HEADER
  7833  017436  017446                          58                      ;RETURN POINT FOR ERROR DATA
  7834
   (1)  017440  017452                          TST26                   ;RETURN FOR GOOD COMPARISON
   (1)
  7835  017442  104004                  48:     ERROR   4               ;READ NEXT ERROR
  7836  017444  000207                          RTS     PC              ;RETURN TO "COMPAR"
  7837  017446  104005                  58:     ERROR   5               ;WORD NOS 1 TO 4 ARE
  7838                                                                   ;HEADER WORDS
  7839                                                                   ;5 TO 260 ARE DATA WORDS
  7840  017450  000207                          RTS     PC              ;RETURN TO "COMPAR"
  7841
  7842
  7843
  7844
  7845
  7846
  7855
  7856                          ;*****************************************************************
   (3)                          ;*TEST 26       WRITE ECC TEST 3
   (4)
   (4)                          ;*      THIS IS A WRITE ECC TEST
   (4)                          ;*      WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
   (4)                          ;*      TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
   (4)                          ;*      OF ALL 52525.
   (4)
   (3)                          ;*****************************************************************
   (2)  017452  000004          TST26:  SCOPE
  7857  017454  012706  001000          MOV     #STACK,SP       ;RESET STACK
  7858
  7859
   (1)  017460  012737  000026  004174  MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
   (1)
  7860  017466  012700  032636          MOV     #SECGAP,R0      ;POINTER
  7861  017472  012701  000460          MOV     #304.,R1        ;COUNTER
  7862  017476  005020                  1$:     CLR     (R0)+           ;CLEAR SIMULATED DISK AREA
  7863  017500  005301                          DEC     R1
  7864  017502  001375                          BNE     1$
  7865  017504  004767  004760                  JSR     PC,CLDISK                       ;THIS IS USED TO SET GENERAL REGISTERS
  7866
  7867                                  ;THESE ARE FOR ECC TEST ONLY
  7868
  7869  017510  012737  177777  002012          MOV     #-1,@#TSECC             ;THIS IS AN ECC TEST
  7870  017516  005037  027140                  CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
  7871  017522  013737  027134  027136          MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
  7872  017530  013737  027142  027150          MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
  7873  017536  005037  027126                  CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
  7874  017542  005037  027130                  CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
  7875  017546  005037  027144                  CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
  7876  017552  005037  027146                  CLR     @#ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
  7877
```

```
                                        ;THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
                                        ;
017556  012737  010000  034122          MOV     #FMT22,@#WCYL   ;FORMAT22=16BIT WORDS AND
                                                                ;CYLINDER 0
017564  012737  000001  034124          MOV     #1,@#WSECTR     ;TRACK=0, SECTOR=1
017572  005037  034126                  CLR     @#WKEY1         ;KEY1=0
017576  005037  034130                  CLR     @#WKEY2         ;KEY2=0
017602  012737  000400  034162          MOV     #256.,,@#FNWORD ;256 DATA WORDS
017610  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
017614  034122                          WCYL
017616  034132                          GCRC
                                        ;
                                        ;THESE ARE REGULAR SETUPS
                                        ;
017620  012777  177374  161776          MOV     #-260.,,@RHWC   ;256 DATA WORDS 4 HEADER WORDS
017626  012700  002064                  MOV     #WRFROM,R0      ;THESE TWO INSTRUCTIONS GETS
017632  010077  161770                  MOV     R0,@RHBA        ;ADDR. OF WRFROM INTO R0 AND
                                                                ;BUS ADDRESS REGISTER
017636  012720  010000                  MOV     #FMT22,(R0)+    ;FORMAT=16 BIT WORDS
                                                                ;CYLINDER=0
017642  012720  000001          281     MOV     #1,(R0)+        ;TRACK=0, SECTOR=1, KEYS=0
017646  005020                          CLR     (R0)+           ;KEY1=0
017650  005020                          CLR     (R0)+           ;KEY2=0
017652  012705  000400                  MOV     #256.,,R5       ;COUNTER
017656  012720  052525          381     MOV     #052525,(R0)+   ;MOVE ALL 52525 FOR DATA
017662  005305                          DEC     R5
017664  001374                          BNE     3$              ;BRANCH IF DATA NOT COMPLETE
017666  012777  000001  161742          MOV     #1,@RHDST       ;TRACK=0 SECTOR=1
                                        ;
                                        ;
 (1)
 (1)    017674  004767  004624          JSR     PC,CHECKT               ;CHECK DVA, RDY, DPR, DRY
 (1)
                                        ;
017700  013711  002042                  MOV     @#WRIFOR,@R1    ;GET READY FOR WRITE HEADER AND
                                                                ;DATA WITH 62 IN RHCS1
017704  005037  001774                  CLR     @#ERFLG         ;CLEAR ERROR FLAG
017710  012777  010000  161724          MOV     #FMT22,@RHOF    ;FORMAT BIT=1 (16 BIT WORDS)
017716  005077  161722                  CLR     @RHCA           ;CYLINDER =0
017722  004737  033776                  JSR     PC,@#COMWHD     ;WRITE HEADER AND DATA
                                        ;
                                        ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
                                        ;FROM THE "COMWHD" ROUTINE THAT MEANS ALL HEADER ON DISK
                                        ;IS GOOD IE, ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
                                        ;ALL 52525 AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
                                        ;THEY ARE ALL ZEROS
                                        ;
017726  005737  001774                  TST     @#ERFLG         ;HAS ANY ERRORS OCCURED?
                                                                ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
                                        ;
 (1)    017732  001053                  BNE     TST27   ;;BRANCH IF YES
```

```
  (1)
 7928
 7929
 7930                                   ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
 7931  017734  023737  027126  033734          CMP     @#GECC1,@#WECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
 7932  017742  001402                          BEQ     6$              ;BRANCH IF GOOD
 7933  017744  104031                          ERROR   31              ;LOW ORDER ECC IN ERROR
 7934  017746  000405                          BR      7$              ;BRANCH TO CONTINUE
 7935  017750  023737  027130  033736  6$:     CMP     @#GECC2,@#WECC2;COMPARE SOFTWARE ECC WITH HARDWARE ECC
 7936  017756  001401                          BEQ     7$              ;BRANCH IF GOOD
 7937  017760  104031                          ERROR   31              ;HIGH ORDER ECC IN ERROR
 7938
 7939
 7940  017762  004737  024706          7$:     JSR     PC,@#CHECKE     ;CHECK DVA,RDY,DRY,DPR
 7941
 7942
 7943
 7944
 7945                                   ;FILL "REINTO" BUFFER WITH EXPECTED DATA
 7946  017766  004037  024406                  JSR     R0,@#CLAREA     ;FILL REINTO BUFFER
 7947  017772  003130                          REINTO                  ;FROM
 7948  017774  004126                          REINTO+<256.*2>         ;TO
 7949  017776  052525                          .WORD   52525           ;DATA
 7950
 7951  020000  013737  027126  004130          MOV     @#GECC1,@#REINTO+<256.*2>;FILL ECC1
 7952  020006  013737  027130  004132          MOV     @#GECC2,@#REINTO+<257.*2>;FILL ECC2
 7953  020014  004037  024406                  JSR     R0,@#CLAREA     ;FILL REST
 7954  020020  004134                          REINTO+<258.*2>         ;FROM
 7955  020022  004170                          REINTO+<272.*2>         ;TO
 7956  020024  000000                          0               ;DATA
 7957
 7958
 7959  020026  005037  001774                  CLR     @#ERFLG         ;CLEAR ERROR FLAG
 7960
 7961
 7962                                   ;NOW COMPARE "DISK" BUFFER WITH "REINTO"
 7963  020032  004037  025342                  JSR     R0,@#COMPAR     ;CHECK
 7964  020036  003130                          REINTO                  ;GOOD BUFFER
 7965  020040  032734                          DISK                    ;TEST BUFFER
 7966  020042  000402                          256.                    ;NUMBER OF WORDS CHECKED
 7967  020044  020052                          4$                      ;RETURN POINT FOR ERROR HEADER
 7968  020046  020056                          5$                      ;RETURN POINT FOR ERROR DATA
 7969
  (1)  020050  020062                          TST27                   ;RETURN FOR GOOD COMPARISON
  (1)
 7970  020052  104007                  4$:     ERROR   7               ;READ ERROR 10 NEXT
 7971  020054  000207                          RTS     PC              ;RETURN TO COMPARE
 7972  020056  104010                  5$:     ERROR   10              ;WORD NOS 1 TO 256 ARE
 7973                                          ;DATA WORDS
 7974                                          ;WORD NOS 257 AND 258
 7975                                          ;ARE ECC WHICH ARE CHECKED
 7976                                          ;WORD NOS 259
 7977                                          ;IS DATA GAP
 7978                                          ;WORD NOS 260 TO 273
```

```
                                                                        ;ARE TOLERANCE GAP
020060  000207                                  RTS     PC              ;RETURN TO COMPARE






                                        ;********************************************************************
                                        ;*TEST 27       READ ECC ENABLED 3A

                                        ;*      THIS IS AN ECC READ DATA TEST
                                        ;*      ERROR CORRECTION IS ENABLED
                                        ;*      NO ERROR IS INSERTED
                                        ;*      GOOD DATA USED IS 256 WORDS OF 52525
                                        ;*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
                                        ;*      TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA

                                        ;********************************************************************
020062  000004                          TST27:  SCOPE
020064  012706  001000                          MOV     #STACK,SP       ;RESET STACK


020070  012737  000027  004174                  MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER



                                        ;       SETUP FOR WHAT IS TO BE READ
                                        ;       HEADER CRC IS RESTORED FROM A SUBROUTINE

020076  012746  052525                          MOV     #52525, -(SP)   ;DATA TO BE READ
020102  012705  000400                          MOV     #256.,  R5      ;COUNTER
020106  012700  032734                          MOV     #DISK,  R0      ;START OF SIMULATED DISK DATA
020112  011620                          10$:    MOV     (SP),   (R0)+   ;MOVE IN DATA ON TO SIMULATED DISK
020114  005305                                  DEC     R5              ;COUNT
020116  001375                                  BNE     1$              ;BRANCH IF 256 NOT COMPLETE
020120  005726                                  TST     (SP)+           ;UNDO -(SP)
020122  022020                                  CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
020124  012705  000017                          MOV     #15.,   R5      ;1 DATA GAP
                                                                        ;14 TOLERANCE GAP
020130  005020                          20$:    CLR     (R0)+           ;CLEAR DATA GAP, AND
020132  005305                                  DEC     R5              ;TOLERANCE GAP
020134  001375                                  BNE     20              ;BRANCH IF NOT COMPLETE


020136  004737  027722                          JSR     PC,@#FILLEC     ;INSERT THE TWO ECC WORDS ON THE DISK
                                                                        ;IN THE CORRECT PLACE

                                                ;THESE ARE FOR ECC TEST ONLY

020142  012737  177777  002012                  MOV     #-1,@#TSECC             ;THIS IS AN ECC TEST
020150  005037  027140                          CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
020154  013737  027134  027136                  MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
020162  013737  027142  027150                  MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
```

```
 8030  020170  005037  027126                 CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
 8031  020174  005037  027130                 CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
 8032  020200  005037  027144                 CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
 8033  020204  005037  027146                 CLR     @#ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
 8034
 8035
 8036                                         ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
 8037
 8038  020210  012737  010000  031016          MOV     #FMT22,@#CYL    ;16 BITS PER WORD
 8039                                                                 ;CYLINDER 0, FORMAT 16 BITS
 8040  020216  112737  000000  031021          MOVB    #0,     @#SECOTR+1 ;TRACK 0
 8041  020224  112737  000000  031020          MOVB    #0,     @#SECOTR ;SECTOR 0
 8042  020232  012737  000000  031022          MOV     #0,     @#KEY1  ;KEY1=0
 8043  020240  012737  000000  031024          MOV     #0,     @#KEY2  ;KEY2=0
 8044  020246  012737  000400  031076          MOV     #256.,  @#DAWORD ;NO. OF DATA WORDS
 8045  020254  005037  031026                 CLR     @#X             ;THIS IS A READ COMMAND
 8046  020260  004537  025646                 JSR     R5,@#CRC        ;GO TO CALCULATE CRC
 8047  020264  031016                         CYL
 8048  020266  032716                         WCRC
 8049
 8050
 8051
 8052
 8053                                         ;THESE ARE REGULAR SETUPS
 8054  020270  004737  024470                 JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
 8055  020274  012777  177374  161323          MOV     #-256.*4.,@#RHWC ;256. DATA & HEADER WORDS
 8056  020302  012777  003130  161316          MOV     #READ0,@#RHBA   ;STARTING ADDRESS OF READ BUFFER
 8057  020310  112746  000000                 MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
 8058  020314  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
 8059  020322  012677  161310                 MOV     (SP)+,  @#RHDST ;TRACK/SECTOR IN RHDST
 8060  020326  012777  010000  161306          MOV     #FMT22,@#RHOF ;16 BITS PER WORD
 8061                                                                 ;ECC CORRECTION NOT INHIBIT
 8062                                                                 ;BECAUSE ECC IS NOT GOING
 8063                                                                 ;TO BE CHECKED
 8064  020334  005077  161304                 CLR     @#RHCA          ;CYLINDER 0
 8065
 8066  020340  004737  024524                 JSR     PC,     @#CHECKT ;CHECK FOR DVA,RDY,NOL,DPR,DRY
 8067
 8068  020344  013711  002046                 MOV     @#REFOR,@#R1    ;READ HEADER AND DATA=72
 8069  020350  005037  001774                 CLR     @#ERFLG#                ;CLEAR ERROR FLAG
 8070  020354  004737  030706                 JSR     PC,     @#COMMD ;READ HEADER AND DATA
 8071                                                                 ;IF THERE ARE READ ERRORS THEN
 8072                                                                 ;ECC WILL NOT BE CHECKED
 8073
 8074
 8075                                         ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 8076                                         ;FROM THE "COMMD" ROUTINE THAT MEANS SECTOR GAP,
 8077                                         ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
 8078                                         ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
 8079                                         ;DETECTED
 8080                                         ;HEADER AND DATA ARE TO BE CHECKED,
 8081                                         ;IN CHECKING READ DATA THE WRITE FROM BUFFER
 8082                                         ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
 8083                                         ;COMPARISONS ARE MADE
```

```
 8084
 8085  020360  005737  001774                TST     @#ERFLG#                ;ANY ERRORS ALREADY THERE
 8086
  (1)  020364  001077                        BNE     TST30   ;BRANCH IF YES
  (1)
 8087  020366  004737  024170                JSR     PC,@#PUTREG     ;SAVE REGISTERS
 8088  020372  005737  001704                TST     @#ER1           ;NO ERRORS SHOULD BE SET
 8089  020376  001401                        BEQ     6$              ;BRANCH IF NO ERRORS SET
 8090  020400  104032                        ERROR   32              ;32 BIT ECC REGISTER SHOULD BE ZERO
 8091                                                                ;ONLY 11 OF THE 32 BITS CAN BE SEEN
 8092                                                                ;IN THE PATERN REGISTER
 8093                                                                ;DCK SHOULD BE SET IN RMER1
 8094  020402  013746  027126          6$:   MOV     @#GECC1,-(SP)   ;GET PATTERN REGISTER
 8095  020406  042716  174000                BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
 8096  020412  022637  001734                CMP     (SP)+,@#EC2     ;COMPARE PATTERN REGISTER
 8097  020416  001401                        BEQ     7$              ;BRANCH IF GOOD
 8098  020420  104032                        ERROR   32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
 8099
 8100
 8101
 8102
 8103                                        ;ADD 16 MAINTENANCE CLOCKS TO
 8104                                        ;BRING EBL DOWN
 8105
 8106  020422  012700  000020          7$:   MOV     #16.,R0         ;COUNTER
 8107  020426  052777  000002  161216  8$:   BIS     #MCLK,@RHMR     ;SET CLOCK
 8108  020434  042777  000002  161210        BIC     #MCLK,@RHMR     ;CLEAR CLOCK
 8109  020442  005300                        DEC     R0              ;COUNT
 8110  020444  001370                        BNE     8$              ;BRANCH IF 16 CLOCKS NOT DONE
 8111  020446  004737  024706                JSR     PC,@#CHECKE     ;CHECK DVA,DRY,RDY,DPR
 8112  020452  012700  002064                MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
 8113  020456  012720  010000                MOV     #@#IFNT22,(R0)+ ;CYLINDER 0
 8114  020462  112746  000000                MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
 8115  020466  112766  000000  000001        MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
 8116  020474  012620                        MOV     (SP)+,  (R0)+   ;GET TRACK/SECTOR IN BUFFER
 8117  020476  012720  000000                MOV     #0,     (R0)+   ;KEY1 IN BUFFER
 8118  020502  012720  000000                MOV     #0,     (R0)+   ;KEY2 IN BUFFER
 8119  020506  012701  000400                MOV     #256.,  R1      ;DATA WORD COUNTER
 8120  020512  012702  052525                MOV     #052525, R2     ;DATA
 8121  020516  010220                  3$:   MOV     R2,     (R0)+   ;DATA INTO BUFFER
 8122  020520  005301                        DEC     R1              ;COUNT
 8123  020522  001375                        BNE     3$              ;BRANCH IF 256 NOT DONE
 8124
 8125
 8126
 8127
 8128  020524  005037  001774                CLR     @#ERFLG#                ;CLEAR ERROR FLAG
 8129
 8130  020530  004737  024170                JSR     PC,@#PUTREG     ;SAVE REGISTERS
 8131
 8132                                        ;NOW READ DATA BUFFER WILL BE CHECKED
 8133
 8134  020534  004037  025342                JSR     R0,@#COMPAR     ;CHECK
 8135  020540  002064                        WRFROM                  ;GOOD BUFFER
```

```
 8136  020542  003130                        REINTO                  ;TEST BUFFER
 8137  020544  000404                        4*256.                  ;NUMBER OF WORDS CHECKED
 8138  020546  020554                        48                      ;RETURN POINT FOR ERROR HEADER
 8139  020550  020560                        58                      ;RETURN POINT FOR ERROR DATA
 8140
  (1)  020552  020564                        TST39                   ;RETURN FOR GOOD COMPARISON
  (1)
 8141  020554  104004              48:       ERROR   4               ;READ NEXT ERROR
 8142  020556  000207                        RTS     PC              ;RETURN TO "COMPAR"
 8143  020560  104005              58:       ERROR   5               ;WORD NOS 1 TO 4 ARE
 8144                                                                ;HEADER WORDS
 8145                                                                ;5 TO 260 ARE DATA WORDS
 8146  020562  000207                        RTS     PC              ;RETURN TO "COMPAR"
 8147
 8148
 8149
 8150
 8162
 8163                            ;*****************************************************************
  (3)                            ;*TEST 38       READ ECC ENABLED 3B
  (4)
  (4)                            ;*      THIS IS AN ECC READ DATA TEST
  (4)                            ;*      ERROR CORRECTION IS ENABLED
  (4)                            ;*      A CORRECTABLE ERROR IS INSERTED IN BIT POSITION 4128
  (4)                            ;*      THIS IS THE LAST BIT OF THE ECC
  (4)                            ;*      GOOD DATA USED IS 256 WORDS OF 52525
  (4)                            ;*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
  (4)                            ;*      TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
  (4)
  (3)                            ;*****************************************************************
  (2)  020564  000004            TST38:  SCOPE
 8164  020566  012706  001000              MOV     #STACK,SP       ;RESET STACK
 8165
 8166
  (1)  020572  012737  000038  004174      MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
  (1)
 8167
 8168
 8169                            ;       SETUP FOR WHAT IS TO BE READ
 8170                            ;       HEADER CRC IS RESTORED FROM A SUBROUTINE
 8171
 8172  020600  012746  052525              MOV     #52525, -(SP)   ;DATA TO BE READ
 8173  020604  012705  000400              MOV     #256.,  R5      ;COUNTER
 8174  020610  012700  032734              MOV     #DISK,  R0      ;START OF SIMULATED DISK DATA
 8175  020614  011620              18:       MOV     (SP),   (R0)+   ;MOVE IN DATA ON TO SIMULATED DISK
 8176  020616  005305                        DEC     R5              ;COUNT
 8177  020620  001375                        BNE     18              ;BRANCH IF 256 NOT COMPLETE
 8178  020622  005726                        TST     (SP)+           ;UNDO -(SP)
 8179  020624  022020                        CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
 8180  020626  012705  000017              MOV     #15.,   R5      ;1 DATA GAP
 8181                                                                ;14 TOLERANCE GAP
 8182  020632  005020              28:       CLR     (R0)+           ;CLEAR DATA GAP, AND
 8183  020634  005305                        DEC     R5              ;TOLERANCE GAP
 8184  020636  001375                        BNE     28              ;BRANCH IF NOT COMPLETE
```

```
 8185
 8186
 8187  020640  004737  027722                JSR     PC,@#FILLEC     ;INSERT ECC IN PROPER PLACE ON DISK
 8188
 8189
 8190
 8191                                        ;THESE ARE FOR ECC TEST ONLY
 8192
 8193  020644  012737  177777  002012        MOV     #-1,@#TSECC             ;THIS IS AN ECC TEST
 8194  020652  005037  027140                CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
 8195  020656  013737  027134  027136        MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
 8196  020664  013737  027142  027150        MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
 8197  020672  005037  027126                CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
 8198  020676  005037  027130                CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
 8199  020702  005037  027144                CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
 8200  020706  005037  027146                CLR     @#ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
 8201
 8202
 8203                                        ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
 8204
 8205  020712  012737  010000  031016        MOV     #FMT22,@#CYL    ;16 BITS PER WORD
 8206                                                                ;CYLINDER 0, FORMAT 16 BITS
 8207  020720  112737  000000  031021        MOVB    #0,     @#SECTR+1 ;TRACK 0
 8208  020726  112737  000000  031020        MOVB    #0,     @#SECTR ;SECTOR 0
 8209  020734  012737  000000  031022        MOV     #0,     @#KEY1  ;KEY1=0
 8210  020742  012737  000000  031024        MOV     #0,     @#KEY2  ;KEY2=0
 8211  020750  012737  000400  031076        MOV     #256.,  @#DAWORD ;NO. OF DATA WORDS
 8212  020756  005037  031026                CLR     @#X             ;THIS IS A READ COMMAND
 8213  020762  004537  025646                JSR     R5,@#CRC        ;GO TO CALCULATE CRC
 8214  020766  031016                        CYL
 8215  020770  032716                        WCRC
 8216
 8217
 8218                                        ;THIS IS TO INSERT ERROR
 8219                                        ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
 8220                                        ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
 8221                                        ;THIS MOVE
 8222                                        ;THIS CHANGES THE LAST BIT OF THE ECC
 8223  020772  013746  033736                MOV     @#WECC2,-(SP)   ;GET LAST ECC
 8224  020776  005116                        COM     (SP)            ;INVERT ALL BITS OF WECC2
 8225  021000  042716  077777                BIC     #^C100000,(SP)  ;KEEP BIT 16
 8226  021004  042737  100000  033736        BIC     #100000,@#WECC2 ;CLEAR BIT 16 IN ECC
 8227  021012  052637  033736                BIS     (SP)+,@#WECC2   ;THIS WILL SET BIT 16 IF IT WAS 0
 8228                                                                ;OR WILL SET NOTHING IF IT WAS A 1
 8229
 8230
 8231  021016  012737  010026  021164        MOV     #4118.,@#8      ;INSERT POSITION REG.
 8232
 8233
 8234                                        ;THESE ARE REGULAR SETUPS
 8235  021024  004737  024470                JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
 8236  021030  012777  177374  160566        MOV     #-256.,-4.,@RHWC ;256. DATA 4 HEADER WORDS
 8237  021036  012777  003130  160562        MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
 8238  021044  112746  000000                MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
```

```
 8239  021050  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
 8240  021056  012677  160554                  MOV     (SP)+,  @RHDST  ;TRACK/SECTOR IN RHDST
 8241  021062  012777  010000  160552          MOV     #FMT22,@RHOF ;16 BITS PER WORD
 8242                                                                  ;ECC CORRECTION NOT INHIBIT
 8243                                                                  ;BECAUSE ECC IS NOT GOING
 8244                                                                  ;TO BE CHECKED
 8245  021070  005077  160550                  CLR     @RHCA           ;CYLINDER 0
 8246
 8247  021074  004737  024524                  JSR     PC,     @#CHECKT ;CHECK FOR DVA,RDY,MOL,DPR,DRY
 8248
 8249  021100  013711  002046                  MOV     @#REFOR,@R1     ;READ HEADER AND DATA=72
 8250  021104  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
 8251  021110  004737  030706                  JSR     PC,     @#COMMD ;READ HEADER AND DATA
 8252                                                                  ;IF THERE ARE READ ERRORS THEN
 8253                                                                  ;ECC WILL NOT BE CHECKED
 8254
 8255
 8256                                          ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 8257                                          ;FROM THE "COMMD" ROUTINE THAT MEANS SECTOR GAP,
 8258                                          ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
 8259                                          ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
 8260                                          ;DETECTED
 8261                                          ;HEADER AND DATA ARE TO BE CHECKED,
 8262                                          ;IN CHECKING READ DATA THE WRITE FROM BUFFER
 8263                                          ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
 8264                                          ;COMPARISONS ARE MADE
 8265
 8266  021114  005737  001774                  TST     @#ERFLG#                ;ANY ERRORS ALREADY THERE
 8267
  (1)  021120  001071                          BNE     TST31   ;BRANCH IF YES
  (1)
 8268  021122  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
 8269  021126  022737  100000  001784          CMP     #DCK,@#ER1      ;ONLY DATA CHECK ERROR SHOULD BE SET
 8270  021134  001401                          BEQ     6$              ;BRANCH IF YES
 8271  021136  104032                          ERROR   32              ;32 BIT ECC REGISTER SHOULD BE NON
 8272                                                                  ;ZERO
 8273                                                                  ;ONLY 11 OF THE 32 BITS CAN BE SEEN
 8274                                                                  ;IN THE PATERN REGISTER
 8275                                                                  ;DCK SHOULD BE SET IN RHER1
 8276  021140  013746  027126          6$:     MOV     @#GECC1,-(SP)   ;GET PATTERN REGISTER
 8277  021144  042716  174000                  BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
 8278  021150  022637  001734                  CMP     (SP)+,@#EC2     ;COMPARE PATTERN REGISTER
 8279  021154  001401                          BEQ     7$              ;BRANCH IF GOOD
 8280  021156  104032                          ERROR   32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
 8281
 8282  021160  004037  027550          7$:     JSR     R0,@#ECORR      ;GO TO ECC CORRECTION PROCESS
 8283  021164  010026                  8$:     4110.                   ;EXPECTED POSITION REG, WHEN CORRECTION
 8284                                                                  ;IS COMPLETE
 8285
 8286
 8287
 8288  021166  004737  024706                  JSR     PC,@#CHECKE     ;CHECK DVA,DRY,RDY,DPR
 8289  021172  012700  002064                  MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
 8290  021176  012720  010000                  MOV     #@#FMT22,(R0)+  ;CYLINDER 0
```

```
 8291  021202  112746  000000                  MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
 8292  021206  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
 8293  021214  012620                          MOV     (SP)+,  (R0)+   ;GET TRACK/SECTOR IN BUFFER
 8294  021216  012720  000000                  MOV     #0,     (R0)+   ;KEY1 IN BUFFER
 8295  021222  012720  000000                  MOV     #0,     (R0)+   ;KEY2 IN BUFFER
 8296  021226  012701  000400                  MOV     #256.,  R1      ;DATA WORD COUNTER
 8297  021232  012702  052525                  MOV     #52525, R2      ;DATA
 8298  021236  010220                  38$:    MOV     R2,     (R0)+   ;DATA INTO BUFFER
 8299  021240  005301                          DEC     R1              ;COUNT
 8300  021242  001375                          BNE     3$              ;BRANCH IF 256 NOT DONE
 8301
 8302                                          ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
 8303                                          ;NOW THE INSERTED ERROR WILL BE PUT IN
 8304                                          ;BUT INSERTED ERROR IS IN ECC SO DATA IS NOT WRONG
 8305
 8306
 8307  021244  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
 8308
 8309  021250  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
 8310
 8311
 8312                                          ;NOW READ DATA BUFFER WILL BE CHECKED
 8313
 8314  021254  004037  025342                  JSR     R0,@#COMPAR     ;CHECK
 8315  021260  002064                          WRFROM                  ;GOOD BUFFER
 8316  021262  003130                          REINTO                  ;TEST BUFFER
 8317  021264  000404                          4+256.                  ;NUMBER OF WORDS CHECKED
 8318  021266  021274                          4$                      ;RETURN POINT FOR ERROR HEADER
 8319  021270  021300                          5$                      ;RETURN POINT FOR ERROR DATA
 8320
  (1)  021272  021304                          TST31                   ;RETURN FOR GOOD COMPARISON
  (1)
 8321  021274  104004                  4$:     ERROR   4               ;READ NEXT ERROR
 8322  021276  000207                          RTS     PC              ;RETURN TO "COMPAR"
 8323  021300  104005                  5$:     ERROR   5               ;WORD NOS 1 TO 4 ARE
 8324                                                                  ;HEADER WORDS
 8325                                                                  ;5 TO 260 ARE DATA WORDS
 8326  021302  000207                          RTS     PC              ;RETURN TO "COMPAR"
 8327
 8339
 8340                                          ;*****************************************************************
  (3)                                          ;*TEST 31       READ ECC ENABLED 3C
  (4)
  (4)                                          ;*      THIS IS AN ECC READ DATA TEST
  (4)                                          ;*      ERROR CORRECTION IS ENABLED
  (4)                                          ;*      A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 296 THRU 308
  (4)                                          ;*      THIS IS IN WORD NUMBER 19 AND 20
  (4)                                          ;*      GOOD DATA USED IS 256 WORDS OF 52525
  (4)                                          ;*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
  (4)                                          ;*      TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
  (4)
  (3)                                          ;*****************************************************************
  (2)  021304  000004                  TST31:  SCOPE
 8341  021306  012706  001000                  MOV     #STACK,SP       ;RESET STACK
```

c9

```
 8342
 8343
  (1)  021312  012737  000031  004174          MOV     #TTNO,@#T8TNM   ;THIS SAVES TEST NUMBER
  (1)
 8344
 8345
 8346                                  ;       SETUP FOR WHAT IS TO BE READ
 8347                                  ;       HEADER CRC IS RESTORED FROM A SUBROUTINE
 8348
 8349  021320  012746  052525                  MOV     #52525, -(SP)   ;DATA TO BE READ
 8350  021324  012705  000400                  MOV     #256.,  R5      ;COUNTER
 8351  021330  012700  032734                  MOV     #DISK,  R0      ;START OF SIMULATED DISK DATA
 8352  021334  011620                  18:     MOV     (SP),   (R0)+   ;MOVE IN DATA ON TO SIMULATED DISK
 8353  021336  005305                          DEC     R5              ;COUNT
 8354  021340  001375                          BNE     1B              ;BRANCH IF 256 NOT COMPLETE
 8355  021342  005726                          TST     (SP)+           ;UNDO -(SP)
 8356  021344  022020                          CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
 8357  021346  012705  000017                  MOV     #15.,   R5      ;1 DATA GAP
 8358                                                                  ;14 TOLERANCE GAP
 8359  021352  005020                  28:     CLR     (R0)+           ;CLEAR DATA GAP, AND
 8360  021354  005305                          DEC     R5              ;TOLERANCE GAP
 8361  021356  001375                          BNE     2B              ;BRANCH IF NOT COMPLETE
 8362
 8363
 8364  021360  004737  027722                  JSR     PC,@#FILLEC     ;INSERT THE TWO ECC WORDS ON THE DISK
 8365                                                                  ;IN THE CORRECT PLACE
 8366
 8367                                          ;THESE ARE FOR ECC TEST ONLY
 8368
 8369  021364  012737  177777  002012          MOV     #-1,@#T8ECC             ;THIS IS AN ECC TEST
 8370  021372  005037  027140                  CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
 8371  021376  013737  027134  027136          MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
 8372  021404  013737  027142  027150          MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
 8373  021412  005037  027126                  CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
 8374  021416  005037  027130                  CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
 8375  021422  005037  027144                  CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
 8376  021426  005037  027146                  CLR     @#ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
 8377
 8378
 8379                                          ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
 8380
 8381  021432  012737  010000  031016          MOV     #FMT22,@#CYL    ;16 BITS PER WORD
 8382                                                                  ;CYLINDER 0, FORMAT 16 BITS
 8383  021440  112737  000000  031021          MOVB    #0,     @#SECOTR+1 ;TRACK 0
 8384  021446  112737  000000  031020          MOVB    #0,     @#SECOTR ;SECTOR 0
 8385  021454  012737  000000  031022          MOV     #0,     @#KEY1  ;KEY1=0
 8386  021462  012737  000000  031024          MOV     #0,     @#KEY2  ;KEY2=0
 8387  021470  012737  000400  031076          MOV     #256.,  @#DAWORD ;NO. OF DATA WORDS
 8388  021476  005037  031026                  CLR     @#X             ;THIS IS A READ COMMAND
 8389  021502  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
 8390  021506  031016                          CYL
 8391  021510  032716                          WCRC
 8392
 8393
```

```
 8394                                           ;THIS IS TO INSERT ERROR
 8395                                           ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
 8396                                           ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
 8397                                           ;THIS MOVE
 8398  021512  012737  152652  033000           MOV     #152652,@#DISK+44;INSERT ERROR IN POSITION 296 THRU 304
 8399                                                                    ;IN WORD NUMBER 19
 8400  021520  012737  052532  033002           MOV     #52532,@#DISK+46;INSERT ERROR IN POSITION 305 THRU 308
 8401                                                                    ;IN WORD NUMBER 20
 8402  021526  012737  010040  021674           MOV     #4128.,@#RHOF   ;INSERT POSITION REG.
 8403
 8404
 8405                                           ;THESE ARE REGULAR SETUPS
 8406  021534  004737  024470                  JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
 8407  021540  012777  177374  160056           MOV     #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
 8408  021546  012777  003130  160052           MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
 8409  021554  112746  000000                   MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
 8410  021560  112766  000000  000001           MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
 8411  021566  012677  160044                   MOV     (SP)+,  @RHDST  ;TRACK/SECTOR IN RHDST
 8412  021572  012777  010000  160042           MOV     #FMT22,@RHOF ;16 BITS PER WORD
 8413                                                                    ;ECC CORRECTION NOT INHIBIT
 8414                                                                    ;BECAUSE ECC IS NOT GOING
 8415                                                                    ;TO BE CHECKED
 8416  021600  005077  160040                   CLR     @RHCA           ;CYLINDER 0
 8417
 8418  021604  004737  024524                  JSR     PC,     @#CHECKT ;CHECK FOR DVA,RDY,MOL,DPR,DRY
 8419
 8420  021610  013711  002046                  MOV     @#REFOR,@R1     ;READ HEADER AND DATA=72
 8421  021614  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
 8422  021620  004737  030706                  JSR     PC,     @#COMMD ;READ HEADER AND DATA
 8423                                                                    ;IF THERE ARE READ ERRORS THEN
 8424                                                                    ;ECC WILL NOT BE CHECKED
 8425
 8426
 8427                                           ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 8428                                           ;FROM THE "COMMD" ROUTINE THAT MEANS SECTOR GAP,
 8429                                           ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
 8430                                           ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
 8431                                           ;DETECTED
 8432                                           ;HEADER AND DATA ARE TO BE CHECKED,
 8433                                           ;IN CHECKING READ DATA THE WRITE FROM BUFFER
 8434                                           ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
 8435                                           ;COMPARISONS ARE MADE
 8436
 8437  021624  005737  001774                  TST     @#ERFLG#                ;ANY ERRORS ALREADY THERE
 8438
  (1)  021630  001106                           BNE     TST32   ;BRANCH IF YES
  (1)
 8439  021632  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
 8440  021636  022737  100000  001704           CMP     @DCK,@#ER1      ;ONLY DATA CHECK ERROR SHOULD BE SET
 8441  021644  001401                           BEQ     6$              ;BRANCH IF YES
 8442  021646  104032                           ERROR   32              ;32 BIT ECC REGISTER SHOULD BE NON
 8443                                                                    ;ZERO
 8444                                                                    ;ONLY 11 OF THE 32 BITS CAN BE SEEN
 8445                                                                    ;IN THE PATERN REGISTER
```

```
8446
8447  021650  013746  027126          68$:    MOV     @#GECC1,-(SP)   ;DCK SHOULD BE SET IN RHER1
8448  021654  042716  174000                  BIC     #174000,(SP)    ;GET PATTERN REGISTER
8449  021660  022637  001734                  CMP     (SP)+,@#EC2     ;KEEP ONLY 11 BITS
8450  021664  001401                          BEQ     7$              ;COMPARE PATTERN REGISTER
8451  021666  104032                          ERROR   32              ;BRANCH IF GOOD
8452                                                                  ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
8453  021670  004037  027550          78$:    JSR     R0,@#ECORR      ;GO TO ECC CORRECTION PROCESS
8454  021674  000000                  88$:    .WORD                   ;EXPECTED POSITION REG, WHEN CORRECTION
8455                                                                  ;IS COMPLETE
8456
8457
8458
8459  021676  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
8460  021702  022737  100100  001704          CMP     #DCK!ECH,@#ER1  ;WITH ERRORS INSERTED IN BIT POSITION 21
8461                                                                  ;THRU 32 HARD ERROR BIT SHOULD SET
8462  021710  001401                          BEQ     9$              ;BRANCH IF GOOD
8463  021712  104036                          ERROR   36              ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
8464                                                                  ;32 HCE SHOULD SET
8465
8466
8467
8468  021714  004737  024706          98$:    JSR     PC,@#CHECKE     ;CHECK DVA,DRY,RDY,DPR
8469  021720  012700  002064                  MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
8470  021724  012720  010000                  MOV     #@!FMT22,(R0)+  ;CYLINDER 0
8471  021730  112746  000000                  MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
8472  021734  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
8473  021742  012620                          MOV     (SP)+,  (R0)+   ;GET TRACK/SECTOR IN BUFFER
8474  021744  012720  000000                  MOV     #0,     (R0)+   ;KEY1 IN BUFFER
8475  021750  012720  000000                  MOV     #0,     (R0)+   ;KEY2 IN BUFFER
8476  021754  012701  000400                  MOV     #256.,  R1      ;DATA WORD COUNTER
8477  021760  012702  052525                  MOV     #052525, R2     ;DATA
8478  021764  010220                  38$:    MOV     R2,     (R0)+   ;DATA INTO BUFFER
8479  021766  005301                          DEC     R1              ;COUNT
8480  021770  001375                          BNE     3$              ;BRANCH IF 256 NOT DONE
8481
8482                                          ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
8483                                          ;NOW THE INSERTED ERROR WILL BE PUT IN
8484  021772  012737  152652  002140          MOV     #152652,@#WRFROM+54;INSERT ERROR IN POSITION 296 THRU 304
8485                                                                  ;IN WORD NUMBER 19 IN DATA
8486  022000  012737  052532  002142          MOV     #052532,@#WRFROM+56;INSERT ERROR IN POSITION 305 THRU 308
8487                                                                  ;IN WORD NUMBER 20 IN DATA
8488
8489  022006  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
8490
8491
8492  022012  005037  001774                  CLR     @#ERFLG         ;CLEAR ERROR FLAG
8493
8494
8495                                          ;NOW READ DATA BUFFER WILL BE CHECKED
8496
8497  022016  004037  025342                  JSR     R0,@#COMPAR     ;CHECK
8498  022022  002064                          WRFROM                  ;GOOD BUFFER
8499  022024  003130                          REINTO                  ;TEST BUFFER
```

```
 8500  022026  000404                          4*256.                  ;NUMBER OF WORDS CHECKED
 8501  022030  022036                          48                      ;RETURN POINT FOR ERROR HEADER
 8502  022032  022042                          58                      ;RETURN POINT FOR ERROR DATA
 8503
  (1)  022034  022046                          TST32                   ;RETURN FOR GOOD COMPARISON
  (1)
 8504  022036  104004                  48:     ERROR   4               ;READ NEXT ERROR
 8505  022040  000207                          RTS     PC              ;RETURN TO "COMPAR"
 8506  022042  104005                  58:     ERROR   5               ;WORD NOS 1 TO 4 ARE
 8507                                                                   ;HEADER WORDS
 8508                                                                   ;5 TO 260 ARE DATA WORDS
 8509  022044  000207                          RTS     PC              ;RETURN TO "COMPAR"
 8510
 8522
 8523                                  ;*****************************************************************
  (3)                                  ;*TEST 32       READ ECC ENABLED 3D
  (4)
  (4)                                  ;*      THIS IS AN ECC READ DATA TEST
  (4)                                  ;*      ERROR CORRECTION IS ENABLED
  (4)                                  ;*      A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 32 AND 4096
  (4)                                  ;*      4096 IS THE LAST DATA BIT
  (4)                                  ;*      GOOD DATA USED IS 256 WORDS OF 52525
  (4)                                  ;*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
  (4)                                  ;*      TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
  (4)
  (3)                                  ;*****************************************************************
  (2)  022046  000004                  TST32:  SCOPE
 8524  022050  012706  001000                  MOV     #STACK,SP       ;RESET STACK
 8525
 8526
  (1)  022054  012737  000032  004174          MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
  (1)
 8527
 8528
 8529                                  ;       SETUP FOR WHAT IS TO BE READ
 8530                                  ;       HEADER CRC IS RESTORED FROM A SUBROUTINE
 8531
 8532  022062  012746  052525                  MOV     #52525, -(SP)   ;DATA TO BE READ
 8533  022066  012705  000400                  MOV     #256.,  R5      ;COUNTER
 8534  022072  012700  032734                  MOV     #DISK,  R0      ;START OF SIMULATED DISK DATA
 8535  022076  011620                  18:     MOV     (SP),   (R0)+   ;MOVE IN DATA ON TO SIMULATED DISK
 8536  022100  005305                          DEC     R5              ;COUNT
 8537  022102  001375                          BNE     18              ;BRANCH IF 256 NOT COMPLETE
 8538  022104  005726                          TST     (SP)+           ;UNDO -(SP)
 8539  022106  022020                          CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
 8540  022110  012705  000017                  MOV     #15.,   R5      ;1 DATA GAP
 8541                                                                   ;14 TOLERANCE GAP
 8542  022114  005020                  28:     CLR     (R0)+           ;CLEAR DATA GAP, AND
 8543  022116  005305                          DEC     R5              ;TOLERANCE GAP
 8544  022120  001375                          BNE     28              ;BRANCH IF NOT COMPLETE
 8545
 8546
 8547  022122  004737  027722                  JSR     PC,@#FILLEC     ;INSERT THE TWO ECC WORDS ON THE DISK
 8548                                                                   ;IN THE CORRECT PLACE
```

```
 8549
 8550                                           ;THESE ARE FOR ECC TEST ONLY
 8551
 8552  022126  012737  177777  002012          MOV     #-1,@#TSECC             ;THIS IS AN ECC TEST
 8553  022134  005037  027140                  CLR     @#POSITI        ;CLEAR ERROR POSITION COUNTER
 8554  022140  013737  027134  027136          MOV     @#NCODE,@#NCOUNT        ;TEMPORARY N-CODE COUNTER
 8555  022146  013737  027142  027150          MOV     @#HARDER,@#HADTMP       ;TEMPORARY HARD ERROR COUNTER
 8556  022154  005037  027126                  CLR     @#GECC1         ;ECC LOW ORDER TO BE GENERATED
 8557  022160  005037  027130                  CLR     @#GECC2         ;ECC HIGH ORDER TO BE GENERATED
 8558  022164  005037  027144                  CLR     @#DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
 8559  022170  005037  027146                  CLR     @#ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
 8560
 8561
 8562                                           ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
 8563
 8564  022174  012737  010000  031016          MOV     #FMT22,@#CYL    ;16 BITS PER WORD
 8565                                                                 ;CYLINDER 0, FORMAT 16 BITS
 8566  022202  112737  000000  031021          MOVB    #0,     @#SECOTR+1 ;TRACK 0
 8567  022210  112737  000000  031020          MOVB    #0,     @#SECOTR ;SECTOR 0
 8568  022216  012737  000000  031022          MOV     #0,     @#KEY1  ;KEY1=0
 8569  022224  012737  000000  031024          MOV     #0,     @#KEY2  ;KEY2=0
 8570  022232  012737  000400  031076          MOV     #256.,  @#DAWORD ;NO. OF DATA WORDS
 8571  022240  005037  031026                  CLR     @#X             ;THIS IS A READ COMMAND
 8572  022244  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
 8573  022250  031016                          CYL
 8574  022252  032716                          WCRC
 8575
 8576
 8577                                           ;THIS IS TO INSERT ERROR
 8578                                           ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
 8579                                           ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
 8580                                           ;THIS MOVE
 8581  022254  012737  152525  032736          MOV     #152525,@#DISK+2                ;FORCE ERROR ON BIT NUMBER 32
 8582  022262  012737  152525  033732          MOV     #152525,@#DISK+<255.*2>;FORCE ERROR IN BIT 4096
 8583  022270  012737  010040  022436          MOV     #4128.,@#88     ;INSERT POSITION REG.
 8584
 8585
 8586                                           ;THESE ARE REGULAR SETUPS
 8587  022276  004737  024470                  JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
 8588  022302  012777  177374  157314          MOV     #-256.*-4.,@RHWC ;256. DATA 4 HEADER WORDS
 8589  022310  012777  003130  157310          MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
 8590  022316  112746  000000                  MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
 8591  022322  112766  000000  000001          MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
 8592  022330  012677  157302                  MOV     (SP)+,  @RHDST  ;TRACK/SECTOR IN RHDST
 8593  022334  012777  010000  157300          MOV     #FMT22,@RHOF ;16 BITS PER WORD
 8594                                                                 ;ECC CORRECTION NOT INHIBIT
 8595                                                                 ;BECAUSE ECC IS NOT GOING
 8596                                                                 ;TO BE CHECKED
 8597  022342  005077  157276                  CLR     @RHCA           ;CYLINDER 0
 8598
 8599  022346  004737  024524                  JSR     PC,     @#CHECKT ;CHECK FOR DVA,RDY,MOL,DPR,DRY
 8600
 8601  022352  013711  002046                  MOV     @#REFOR,@R1     ;READ HEADER AND DATA=72
 8602  022356  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
```

```
 8603  022362  004737  030706                JSR     PC,     @#COMMD ;READ HEADER AND DATA
 8604                                                                ;IF THERE ARE READ ERRORS THEN
 8605                                                                ;ECC WILL NOT BE CHECKED
 8606
 8607
 8608                                        ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 8609                                        ;FROM THE "COMMD" ROUTINE THAT MEANS SECTOR GAP,
 8610                                        ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
 8611                                        ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
 8612                                        ;DETECTED
 8613                                        ;HEADER AND DATA ARE TO BE CHECKED,
 8614                                        ;IN CHECKING READ DATA THE WRITE FROM BUFFER
 8615                                        ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
 8616                                        ;COMPARISONS ARE MADE
 8617
 8618  022366  005737  001774                TST     @#ERFLG#                ;ANY ERRORS ALREADY THERE
 8619
  (1)  022372  001106                        BNE     TST33   ;BRANCH IF YES
  (1)
 8620  022374  004737  024170                JSR     PC,@#PUTREG     ;SAVE REGISTERS
 8621  022400  022737  100000  001764        CMP     #DCK,@#ER1      ;ONLY DATA CHECK ERROR SHOULD BE SET
 8622  022406  001401                        BEQ     6$              ;BRANCH IF YES
 8623  022410  104032                        ERROR   32              ;32 BIT ECC REGISTER SHOULD BE NON
 8624                                                                ;ZERO
 8625                                                                ;ONLY 11 OF THE 32 BITS CAN BE SEEN
 8626                                                                ;IN THE PATERN REGISTER
 8627                                                                ;DCK SHOULD BE SET IN RHER1
 8628  022412  013746  027126        6$:     MOV     @#GECC1,-(SP)   ;GET PATTERN REGISTER
 8629  022416  042716  174000                BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
 8630  022422  022637  001734                CMP     (SP)+,@#EC2     ;COMPARE PATTERN REGISTER
 8631  022426  001401                        BEQ     7$              ;BRANCH IF GOOD
 8632  022430  104032                        ERROR   32              ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
 8633
 8634  022432  004037  027550        7$:     JSR     R0,@#ECORR      ;GO TO ECC CORRECTION PROCESS
 8635  022436  000000                8$:     .WORD                   ;EXPECTED POSITION REG, WHEN CORRECTION
 8636                                                                ;IS COMPLETE
 8637
 8638
 8639
 8640  022440  004737  024170                JSR     PC,@#PUTREG     ;SAVE REGISTERS
 8641  022444  022737  100100  001704        CMP     #DCK!ECH,@#ER1  ;WITH ERRORS INSERTED IN BIT POSITION 32
 8642                                                                ;AND 4096 HARD ERROR BIT SHOULD SET
 8643  022452  001401                        BEQ     9$              ;BRANCH IF GOOD
 8644  022454  104036                        ERROR   36              ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
 8645                                                                ;32 HCE SHOULD SET
 8646
 8647
 8648
 8649
 8650  022456  004737  024706        9$:     JSR     PC,@#CHECKE     ;CHECK DVA,DRY,RDY,DPR
 8651  022462  012700  002064                MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
 8652  022466  012720  010000                MOV     #@!FMT22,(R0)+  ;CYLINDER 0
 8653  022472  112746  000000                MOVB    #0,     -(SP)   ;IN LOWER BYTE GET SECTOR
 8654  022476  112766  000000  000001        MOVB    #0,     1(SP)   ;GET TRACK IN HIGHER BYTE
```

```
 8655  022504  012620                          MOV     (SP)+,  (R0)+   ;GET TRACK/SECTOR IN BUFFER
 8656  022506  012720  000000                  MOV     #0,     (R0)+   ;KEY1 IN BUFFER
 8657  022512  012720  000000                  MOV     #0,     (R0)+   ;KEY2 IN BUFFER
 8658  022516  012701  000400                  MOV     #256.,  R1      ;DATA WORD COUNTER
 8659  022522  012702  052525                  MOV     #052525,R2      ;DATA
 8660  022526  010220              3$:         MOV     R2,     (R0)+   ;DATA INTO BUFFER
 8661  022530  005301                          DEC     R1              ;COUNT
 8662  022532  001375                          BNE     3$              ;BRANCH IF 256 NOT DONE
 8663
 8664                                          ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
 8665                                          ;NOW THE INSERTED ERROR WILL BE PUT IN
 8666  022534  012737  152525  002076          MOV     #152525,@#WRFROM+<5*2>  ;INSERTED ERROR IN BIT 32
 8667  022542  012737  152525  003072          MOV     #152525,@#WRFROM+<259.*2>       ;INSERT ERROR IN BIT 4096
 8668
 8669
 8670
 8671  022550  005037  001774                  CLR     @#ERFLG#                ;CLEAR ERROR FLAG
 8672  022554  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
 8673
 8674
 8675                                          ;NOW READ DATA BUFFER WILL BE CHECKED
 8676
 8677  022560  004037  025342                  JSR     R0,@#COMPAR     ;CHECK
 8678  022564  002064                          WRFROM                  ;GOOD BUFFER
 8679  022566  003130                          REINTO                  ;TEST BUFFER
 8680  022570  000404                          4*256.                  ;NUMBER OF WORDS CHECKED
 8681  022572  022600                          4$                      ;RETURN POINT FOR ERROR HEADER
 8682  022574  022604                          5$                      ;RETURN POINT FOR ERROR DATA
 8683
  (1)  022576  022610                          TST33                   ;RETURN FOR GOOD COMPARISON
  (1)
 8684  022600  104004              4$:         ERROR   4               ;READ NEXT ERROR
 8685  022602  000207                          RTS     PC              ;RETURN TO "COMPAR"
 8686  022604  104005              5$:         ERROR   5               ;WORD NOS 1 TO 4 ARE
 8687                                                                  ;HEADER WORDS
 8688                                                                  ;5 TO 260 ARE DATA WORDS
 8689  022606  000207                          RTS     PC              ;RETURN TO "COMPAR"
 8690
 8691
 8692
 8693
 8694
 8695
 8696
 8697
 8698
 8706
 8707                                          ;*****************************************************************
  (3)                                         ;*TEST 33        PROGRAM INTERRUPT
  (4)
  (4)                                         ;*      PROGRAM INTERRUPT IS TESTED BY SETTING RDY AND IE
  (4)                                         ;*      IN RHCS1 AT THE SAME TIME
  (4)                                         ;*      THIS SHOULD INTERRUPT THROUGH LOCATION 254
  (4)                                         ;*      THE PROCESSOR PRIORITY IS SET TO 4
```

```
 (3)                                ;*****************************************************************
 (2)  022610  000004                TST33:  SCOPE
0708
0709
 (1)  022612  012737  000033  004174          MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
 (1)
0710  022620  012706  001000                  MOV     #STACK,SP       ;RESET STACK
0711  022624  004737  024470                  JSR     PC,@#CLDISK     ;CLEAR DISK
0712  022630  013700  001620                  MOV     @#RPVEC,R0      ;GET VECTOR ADDRESS
0713  022634  012720  022702                  MOV     #RPTRP1,(R0)+   ;SET INTERRUPT VECTOR
0714  022640  012710  000340                  MOV     #340,(R0)       ;SET SERVICE ROUTINE PRIORITY
0715  022644  012767  000200  155124          MOV     #200,P8         ;SET PROCESSOR PRIORITY
0716  022652  012711  000300                  MOV     #RDY!IE,@R1     ;RDY, IE IN RH8C1 SHOULD CAUSE INTERRUPT
0717  022656  013737  025024  001172          MOV     @#TIMCNT,@#TMP1 ;COUNTER
0718  022664  005337  001172          18:     DEC     @#TMP1          ;WAIT FOR INTERRUPT
0719  022670  001375                          BNE     18              ;BRANCH IF NOT ZERO
0720                                                                  ;BEFORE THIS IS ZERO INTERRUPT SHOULD
0721                                                                  ;OCCUR
0722  022672  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
0723  022676  104021                          ERROR   21              ;INTERRUPT DID NOT OCCUR
0724
0725
 (1)  022700  000410                          BR      TST34   ;BRANCH TO NEXT TEST
 (1)
0726
0727  022702  022626          RPTRP1: CMP     (SP)+,(SP)+     ;RESTORE STACK
0728  022704  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
0729  022710  022737  004200  001702          CMP     @#DVA!RDY,@#C81 ;IE SHOULD BE LOW
0730
 (1)  022716  001401                          BEQ     TST34   ;BRANCH IF GOOD
 (1)
0731  022720  104021                          ERROR   21              ;INTERRUPT OCCURED BUT
0732                                                                  ;IE FAILED TO RESET
0733
0734
0735                                ;*****************************************************************
 (3)                                ;*TEST 34       INTERRUPT AT PROCESSOR AND DISK PRIORITY SAME
 (4)
 (4)                                ;*      PROCESSOR PRIORITY IS SET AT 5 (SAME AS THE DISK)
 (4)                                ;*      IE AND RDY IS SET, THIS SHOULD NOT INTERRUPT
 (3)                                ;*****************************************************************
 (2)  022722  000004                TST34:  SCOPE
0741
0742
 (1)  022724  012737  000034  004174          MOV     #TTNO,@#TSTNM   ;THIS SAVES TEST NUMBER
 (1)
0743  022732  012706  001000                  MOV     #STACK,SP       ;RESET STACK
0744  022736  004737  024470                  JSR     PC,@#CLDISK     ;CLEAR DISK
0745  022742  013700  001620                  MOV     @#RPVEC,R0      ;GET VECTOR ADDRESS
0746  022746  012720  023006                  MOV     #RPTRP2,(R0)+   ;SET INTERRUPT VECTOR
0747  022752  012710  000340                  MOV     #340,(R0)       ;SET SERVICE ROUTINE PRIORITY
0748  022756  012767  000240  155012          MOV     #240,P8         ;SET PROCESSOR PRIORITY
0749  022764  012711  000300                  MOV     #RDY!IE,@R1     ;RDY, IE IN RH8C1 SHOULD CAUSE INTERRUPT
0750  022770  013737  025024  001172          MOV     @#TIMCNT,@#TMP1 ;COUNTER
```

```
  8751  022776  005337  001172          18$:    DEC     @#STMP1         ;WAIT FOR INTERRUPT
  8752  023002  001375                          BNE     18              ;BRANCH IF NOT ZERO
  8753                                                                  ;BEFORE THIS IS ZERO INTERRUPT SHOULD
  8754                                                                  ;OCCUR
  8755
   (1)  023004  000404                          BR      TST35   ;NO INTERRUPT SO BRANCH
   (1)
  8756
  8757  023006  022626                  RPTRP2: CMP     (SP)+,(SP)+     ;RESTORE STACK
  8758  023010  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
  8759  023014  104021                          ERROR   21              ;INTERRUPT OCCURRED WITH
  8760                                                                  ;PROCESSOR STATUS SAME
  8761                                                                  ;AS DISK
  8762
  8763
  8764
  8765
  8766                          ;*****************************************************************
  8775                          ;*****************************************************************
   (3)                          ;*TEST 35       END OF DRIVE
   (4)
   (4)                          ;*      THIS IS THE END OF TEST FOR ONE DRIVE
   (4)                          ;*      IF THERE ARE MORE DRIVES THEN THE PROGRAM
   (4)                          ;*      JUMPS TO TEST 5 FOR NEXT DRIVE TEST
   (4)                          ;*      END PASS IS REACHED ONLY AFTER ALL DRIVES ARE COMPLETE
   (4)
   (3)                          ;*****************************************************************
   (2)  023016  000004          TST35:  SCOPE
   (1)  023020  012767  000001  156156          MOV     #1,@#TIMES      ;;DO 1 ITERATION
  8776  023026  004737  024470                  JSR     PC,@#CLDISK
  8777  023032  012767  000000  154736          MOV     #0,PS           ;REINSTATE PS TO 0
  8778  023040  104400  023046                  TYPE    ,.+4            ;;TYPE ASCII STRING
   (1)  023044  000425                          BR      648             ;;GET OVER THE ASCII
   (1)                                          ;;.ASCII        <15><12>/TOTAL ERRORS ON THIS PASS ON  UNIT NO. /
   (1)  023120                          648:
  8779  023120  013746  001762                  MOV     @#UNIT,-(SP)    ;GET READY TO TYPE UNIT NUMBER
  8780  023124  104410                          TYPDS
  8781  023126  104400  023134                  TYPE    ,.+4            ;;TYPE ASCII STRING
   (1)  023132  000402                          BR      658             ;;GET OVER THE ASCII
   (1)                                          ;;.ASCII        /= /
   (1)  023140                          658:
  8782  023140  013746  001112                  MOV     @#ERTTL,-(SP)   ;GET READY TO TYPE NUMBER OF ERRORS
  8783  023144  104410                          TYPDS
  8784  023146  005037  001112                  CLR     @#ERTTL         ;CLEAR TOTAL NUMBER OF ERRORS
  8785  023152  005737  001770                  TST     @#SELECT        ;STARTING FROM 200 ?
  8786  023156  001415                          BEQ     38              ;BRANCH IF YES
  8787  023160  005067  155716                  CLR     STSTNM          ;CLEAR TEST NUMBER
  8788  023164  005237  001100                  INC     @#PASS          ;INCREASE PASS COUNT
  8789  023170  104400  023366                  TYPE    ,ENDMG          ;TYPE END PASS #
  8790  023174  013746  001100                  MOV     @#PASS,-(SP)
  8791  023200  104410                          TYPDS
  8792  023202  104400  023403                  TYPE    ,ENULL
  8793  023206  000137  007036                  JMP     @#TST5          ;JUMP TEST 5
  8794  023212  005337  001764          38:     DEC     @#NOUNITS       ;NO. OF UNITS PRESENT DECREMENT
```

```
 8795  023216  001413                        BEQ     $EOP            ;BRANCH IF ALL DRIVES COMPLETE
 8796  023220  013700  001762                MOV     @#UNIT,R0       ;UNIT UNDER TEST
 8797  023224  012701  001742                MOV     #UNITS,R1       ;TABLE
 8798  023230  022100                18:     CMP     (R1)+,R0        ;IS THIS UNIT JUST TESTED
 8799  023232  001401                        BEQ     28              ;BRANCH IF YES
 8800  023234  000775                        BR      18              ;BRANCH IF NO
 8801  023236  011137  001762          28:   MOV     (R1),@#UNIT     ;THIS IS NEXT UNIT
 8802  023242  000137  007036                JMP     @#TSTS          ;GO FOR NEXT TESTS.
```

```
8808                                    ;*****************************************************************
 (1)
 (1)                                    .SBTTL  END OF PASS ROUTINE
 (1)
 (1)                                    ;*INCREMENT THE PASS NUMBER ($PASS)
 (1)                                    ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
 (1)                                    ;*IF THERES A MONITOR GO TO IT
 (1)                                    ;*IF THERE ISN'T JUMP TO TST1
 (1)
 (1)  023246                            $EOP:
 (1)  023246  000004                            SCOPE
 (1)  023250  005067  155626                    CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 (1)  023254  005067  155724                    CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 (1)  023260  005267  155614                    INC     $PASS           ;;INCREMENT THE PASS NUMBER
 (1)  023264  042767  100000  155606           BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 (1)  023272  005327                            DEC     (PC)+           ;;LOOP?
 (1)  023274  000001                    $EOPCT: .WORD   1
 (1)  023276  003031                            BGT     $DOAGN          ;;YES
 (1)  023300  012737                            MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 (1)  023302  000001                    $ENDCT: .WORD   1
 (1)  023304  023274                            $EOPCT
 (1)  023306  104400  023366                    TYPE    ,$ENDMG         ;;TYPE "END PASS #"
 (2)  023312  016746  155562                    MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 (1)  023316  104410                            TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 (1)  023320  104400  023403                    TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
 (1)  023324  013700  000042            $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 (1)  023330  001414                            BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 (1)  023332  022700  023352                    CMP     #$ENDAD,R0      ;;IS MONITOR ACT11?
 (1)  023336  001004                            BNE     $RESET          ;;NO--BRANCH (IT'S XXDP)
 (1)  023340  022760  177777  000002           CMP     @-1,2(R0)       ;;YES--IS THIS THE LAST PASS?
 (1)  023346  001001                            BNE     $ENDAD          ;;NO--MAKE ANOTHER PASS
 (1)  023350  000005                    $RESET: RESET                   ;;CLEAR THE WORLD
 (1)  023352  004710                    $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 (1)  023354  000240                            NOP                     ;;SAVE ROOM
 (1)  023356  000240                            NOP                     ;;FOR
 (1)  023360  000240                            NOP                     ;;ACT11
 (1)  023362  000137  005340            $DOAGN: JMP     @#TST1          ;;RETURN
 (1)  023366  005015  047105  020104    $ENDMG: .ASCIZ  <15><12>/END PASS #/
 (1)  023374  040523  051523  021440
 (1)  023402     000
 (1)  023403     377     377     000    $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
8809
8810
8811
8812                                            .SBTTL  SUBROUTINES
8813
8814
8815
8816                                    ;HERE IS A DETAILED EXPLAINATION OF HOW THE LOOP ON ERROR WORKS.
8817                                    ;ON HITTING AN ERROR IF THE LOOP ON ERROR SWITCH IS SET, THE
8818                                    ;PROGRAM GOES BACK - USUALLY BACK TO THE BEGINNING OF THE TEST.
8819
8820                                    ;WHEN THIS OPERATOR SELECTABLE SCOPE LOOP IS USED THEN THE POINT
8821                                    ;THE PROGRAM GOES BACK TO CAN BE CHANGED.
```

```
                                        ;THE RESTRICTIONS TO THE POINT WHERE THE PROGRAM CAN GO ARE: -
                                        ;1.  IT MUST BE WITHIN THE TEST UNDER CONSIDERATION
                                        ;2.  LOOP ON ERROR SWITCH MUST BE SET
                                        ;3.  THE ERROR MUST OCCUR WITHIN THE TEST UNDER CONSIDERATION
                                        ;IF THE ERROR DOES NOT OCCUR WITHIN THE TEST UNDER CONSIDERATION
                                        ;THE PROGRAM WILL REVERT TO NORMAL OPERATION,  HOWEVER, IF LOOP ON
                                        ;TEST SWITCH IS SET AND THIS OPERATOR SELECTABLE SCOPE LOOP IS USED
                                        ;THEN THE PROGRAM WILL LOOP BACK TO THE SELECTED POINT WHEN IT
                                        ;COMES TO THE END OF THE TEST UNDER CONSIDERATION.
                                        ;
                                        ;AFTER LOOPING FOR SOME TIME IF THE LOOP SWITCH IS PUT DOWN THEN
                                        ;NORMAL OPERATION WILL CONTINUE.

023406  000000                  TESTAD: 0                       ;FIRST ADDRESS OF TEST
023410                          OPERSEL:
023410  005067  154362                  CLR     PS              ;MAKE PROCESSOR STATUS ZERO
023414  104400  023422                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)    023420  000421                  BR      648             ;;GET OVER THE ASCII
 (1)                                    ;;,ASCII        <15><12>/THE PROGRAM WAS IN TEST NUMBER /
 (1)    023464                  648:
023464  013746  004174                  MOV     @#TSTNM,-(SP)   ;GET READY TO TYPE TEST
023470  104402                          TYPOC                   ;NUMBER
023472  104400  023500                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)    023476  000414                  BR      658             ;;GET OVER THE ASCII
 (1)                                    ;;,ASCII        <15><12>/THE LOOP BACK PC WAS /
 (1)    023530                  658:
023530  013746  001110                  MOV     @#LPERR,-(SP)   ;GET READY TO TYPE LOOP BACK PC
023534  104402                          TYPOC
023536  104400  001215                  TYPE    ,$CRLF
023542  104400  023550                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)    023546  000426                  BR      668             ;;GET OVER THE ASCII
 (1)                                    ;;,ASCII        <15><12>/SET LOOP ON ERROR OR LOOP ON TEST SWITCH/
 (1)    023624                  668:
023624  104400  023632                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)    023630  000420                  BR      678             ;;GET OVER THE ASCII
 (1)                                    ;;,ASCII        <15><12>/TYPE THE FIRST PC OF THE TEST/
 (1)    023672                  678:
023672  104400  023700                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)    023676  000423                  BR      688             ;;GET OVER THE ASCII
 (1)                                    ;;,ASCII        <15><12>/ FOLLOWED BY A CARRIAGE RETURN  /<15><12>
 (1)    023746                  688:
023746  104416                          RDOCT
023750  062716  000002                  ADD     #2,(SP)         ;GET LPADR
023754  012637  001106                  MOV     (SP)+,@#LPADR
023760  104400  023766                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)    023764  000417                  BR      698             ;;GET OVER THE ASCII
 (1)                                    ;;,ASCII        <15><12>/TYPE THE PC WHERE YOU WANT/
 (1)    024024                  698:
024024  104400  024032                  TYPE    ,.+4            ;;TYPE ASCII STRING
 (1)    024030  000441                  BR      708             ;;GET OVER THE ASCII
 (1)                                    ;;,ASCII        <15><12>/ THE PROGRAM TO LOOP BACK TO FOLLOWED BY A CARRIAGE RETUR
 (1)    024134                  708:
024134  104416                          RDOCT
024136  012637  001110                  MOV     (SP)+,@#LPERR   ;GET LPERR
```

```
 8855  024142  013746  001106                 MOV     @#LPADR,-(SP)
 8856                                          ;THIS CLEARS UP GARBAGE
 8857  024146  005037  031132                 CLR     @#NOSYNC                ;CLEAR FLAG FOR HEADER ERROR COMMANDS
 8858  024152  005037  002012                 CLR     @#TSECC                 ;CLEAR FLAG FOR ECC TEST
 8859                                                                          ;WHEN =177777 IT IS AN ECC TEST
 8860                                                                          ;WHEN =0 IT IS NOT AN ECC TEST
 8861
 8862  024156  005037  027132                 CLR     @#TSECCG                ;EVEN IN AN ECC TEST EVERY CLOCK
 8863                                                                          ;IS NOT TO GENERATE ECC
 8864                                                                          ;IF =177777 GENERATE ECC
 8865                                                                          ;IF =0 DO NOT GENERATE ECC
 8866  024162  005037  002014                 CLR     @#TESDTE                ;DRIVE TIMING ERROR TEST
 8867  024166  000002                         RTI
 8868
 8869
 8870
 8871                                          .SBTTL  SAVE REGISTERS ROUTINE
 8872                                          ;THIS SAVES THE CONTENTS OF ALL HARDWARE REGISTERS
 8873                                          ;IN MEMORY LOCATIONS TAGED FROM "WC" TO "EC2"
 8874
 8875
 8876                                          ;THIS IS DONE SO THAT COMPARES ARE DONE WITH SAVED LOCATIONS
 8877                                          ;AND NOT THE REGISTERS THEMSELVES.   THIS WILL MAKE
 8878                                          ;ERROR PRINTOUTS FOR GOOD AND BAD DATA ALWAYS DIFFRENT
 8879
 8880  024170                          PUTREG: MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (2)  024170  010046                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2)  024172  010146                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (2)  024174  010246
 8881  024176  012700  001624                 MOV     #RHWC,R0        ;STARTING ADDRESS OF REG
 8882  024202  012701  001674                 MOV     #WC,R1          ;STARTING ADDRESS OF WERE SAVED
 8883  024206  012702  000023                 MOV     #RHCC-RHWC+2/2,R2 ;NUMBER OF REG, INTO R2
 8884  024212  013021                  100:    MOV     @(R0)+,(R1)+    ;SAVE HARDWARE REG,
 8885  024214  005302                         DEC     R2
 8886  024216  001375                         BNE     100
 8887  024220  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
  (2)  024222  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
  (2)  024224  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
 8888  024226  000207                         RTS     PC
 8889
 8890
 8891
 8892
 8893
 8894                                                  .SBTTL  FLOAT 1 AND 0
 8895                                          ;FLOAT A ONE AND A ZERO THRU A DESIGNATED REGISTER
 8896                                          ;ABSOLUTE ADDRESS OF REG, UNDER TEST IS IN R4
 8897
 8898  024230  000000                  MASK:   0                       ;BITS UNDER TEST
 8899  024232  000000                  LERR:   0                       ;ERROR HLT ADDRESS
 8900  024234  000000                  REGADR: 0
 8901
 8902  024236  012567  177766          BITST:  MOV     (R5)+,  MASK    ;FETCH DATA MASK
 8903  024242  012504                         MOV     (R5)+,  R4      ;GET ADDRESS OF REG, UNDER TEST
```

```
 8904  024244  010467  177764                MOV     R4,     REGADR
 8905  024250  010567  177756                MOV     R5,     LERR    ;GET ERROR RETURN ADDR.
 8906  024254  062705  000004                ADD     #4,     R5      ;MODIFY RETURN ADDR. TO JUMP OVER RTS
 8907  024260  012703  000001                MOV     #1,     R3      ;INITIALIZE DATA PATTERN
 8908  024264  004767  000016        BLT1:   JSR     PC,     BLT2    ;OUTPUT FLOATING ZERO
 8909  024270  004767  000012                JSR     PC,     BLT2    ;OUTPUT FLOATING ONE
 8910  024274  000241                        CLC
 8911  024276  006103                        ROL     R3              ;SHIFT PATTERN
 8912  024300  005703                        TST     R3
 8913  024302  001370                        BNE     BLT1            ;BRANCH IF NOT COMPLETE
 8914  024304  000205                        RTS     R5              ;RETURN TO TEST
 8915  024306  005103                BLT2:   COM     R3              ;COMPLEMENT PATTERN
 8916  024310  012737  024316  024450        MOV     #BLT3,  @#LAD   ;SET SCOPE LOOP
 8917  024316  010337  001124        BLT3:   MOV     R3,@#GDDAT      ;STORE GOOD DATA
 8918  024322  005137  024230                COM     @#MASK          ;AND MASK WITH PATTERN
 8919  024326  043737  024230  001124        BIC     @#MASK, @#GDDAT         ;CLEAR THE REST
 8920  024334  005137  024230                COM     @#MASK          ;RESTORE MASK
 8921  024340  013714  001124                MOV     @#GDDAT,(R4)    ;OUTPUT TO REGISTER
 8922  024344  011437  001126                MOV     (R4),@#BDDAT    ;INPUT FROM REGISTER
 8923  024350  005137  024230                COM     @#MASK
 8924  024354  043737  024230  001126        BIC     @#MASK,@#BDDAT ;AND MASK OUT RECEIVED DATA
 8925  024362  005137  024230                COM     @#MASK          ;RESTORE MASK
 8926  024366  023737  001124  001126        CMP     @#GDDAT,@#BDDAT         ;IS DATA CORRECT
 8927  024374  001403                        BEQ     1$              ;BRANCH IF GOOD
 8928  024376  004777  177630                JSR     PC,     @#LERR  ;GO TO REPORT ERROR
 8929  024402  104420                        SCOP1                   ;LOCAL SCOPE LOOP
 8930  024404  000207                1$:     RTS     PC
 8931                                        .SBTTL  CLEAR MEMORY ROUTINE
 8932
 8933
 8934
 8935                                ;       THIS CLEARS ANY BLOCK OF MEMORY
 8936                                ;       FILLING IT WITH ANY DATA
 8937                                ;
 8938                                ;       CALL
 8939                                ;       JSR     R0,CLAREA
 8940                                ;       X                       ;STARTING ADDRESS OF BLOCK
 8941                                ;       Y
 8942                                ;       Z                       ;DATA TO BE FILLED
 8943                                ;R1 WILL HAVE STARTING ADDRESS OF BLOCK TO BE FILLED
 8944                                ;R2 AFTER SUBTRACTION WILL HAVE TWICE NUMBER OF LOCATIONS
 8945                                ;R3 WILL HAVE DATA TO BE FILLED
 8946                                ;TO AVOID DIVIDE ROUTINE TWO DECREMENT R2 WILL BE USED
 8947  024406                        CLAREA:
  (2)  024406  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2)  024410  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (2)  024412  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 8948  024414  012001                        MOV     (R0)+,R1        ;FROM
 8949  024416  012002                        MOV     (R0)+,R2        ;TO
 8950  024420  012003                        MOV     (R0)+,R3        ;DATA
 8951  024422  160102                        SUB     R1,R2           ;NO. OF LOCATIONS MINUS TWO
 8952  024424  062702  000002                ADD     #2,R2           ;GET TWICE NO OF LOCATIONS
 8953  024430  010321                1$:     MOV     R3,(R1)+        ;MOVE IN DATA
 8954  024432  005302                        DEC     R2
```

```
 8955  024434  005302                    DEC     R2
 8956  024436  001374                    BNE     10              ;BRANCH IF NOT COMPLETE
 8957  024440  012603                    MOV     (SP)+,R3        ;;POP STACK INTO R3
  (2)  024442  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
  (2)  024444  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
 8958  024446  000200                    RTS     R0              ;RETURN
```

```
                                                .SBTTL  LOCAL TRAPS
024450  000000                          LAD:    0

024452  032737  001000  177570  T.SCOP: BIT     #SW09,  @#SWR
024460  001402                          BEQ     1$
024462  013716  024450                  MOV     @#LAD,  (SP)
024466  000002                          1$:     RTI

                                        ;EXAMPLE OF THE USE OF THE ABOVE
                                        ;THIS WILL LOOP BETWEEN X: AND SCOP1 PROVIDED THERE IS NO "NEWTST"
                                        ;MOV    #X,     @#LAD
                                        ;X:     ---     ---
                                        ;       ---     ---
                                        ;       ---     ---
                                        ;       SCOP1


                                                .SBTTL  CLEAD DISK ROUTINE

024470  013701  001632          CLDISK: MOV     @#RHCS1,        R1      ;R1 WILL BE CONTROL AND STATUS1
024474  013702  001630                  MOV     @#RHCS2,        R2      ;R2 WILL BE CONTROL AND STATUS2
024500  013703  001654                  MOV     @#RHDS1,        R3      ;R3 WILL BE DISK STATUS REGISTER1
024504  013704  001634                  MOV     @#RHER1,        R4      ;R4 WILL BE ERROR REGISTER #1

024510  012712  000040                  MOV     #CLR,@R2                ;CLEAR ALL REG,
024514  013712  001762                  MOV     @#UNIT,@R2      ;REINSTATE UNIT NO,
024520  005011                          CLR     @R1                     ;CLEAR FUNCTION BITS
024522  000207                          RTS     PC


                                                .SBTTL  CHECK DISK STATUS ROUTINE


                                        ;THIS CHECKS DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1
                                        ;AND CHECKS MEDIUM ON LINE (MOL), DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1

024524  011637  002002          CHECKT: MOV     (SP),@#PCJSR    ;SAVE PC OF JSR+4
024530  162737  000004  002002          SUB     #4,@#PCJSR      ;GET PC OF JSR
024536  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
024542  022737  004200  001702          CMP     #DVA!RDY,@#CS1  ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
                                                                ;AND BE READY
024550  001423                          BEQ     3$              ;BRANCH IF GOOD
024552  032737  004000  001702          BIT     #DVA,   @#CS1   ;BAD SO TEST DEVICE AVAILABLE
024560  001004                          BNE     1$              ;BRANCH IF DVA THERE
024562  010137  001122                  MOV     R1,@#BDADR      ;ADDRESS OF BAD REGISTER (RHCS1)
024566  104026                          ERROR   26              ;RHCS1 DID NOT HAVE DEVICE
                                                                ;AVAILABLE RIGHT AT THE START
024570  000413                          BR      3$              ;BRANCH TO NEXT COMPARE
024572  032737  000200  001702  1$:     BIT     #RDY,   @#CS1   ;TEST READY
024600  001003                          BNE     2$              ;IF RDY THERE BRANCH
024602  010137  001122                  MOV     R1,@#BDADR      ;ADDRESS OF BAD REGISTER (RHCS1)
024606  104026                          ERROR   26              ;RHCS1 DID NOT HAVE READY
                                                                ;RIGHT AT THE START
024610  000403                  2$:     BR      3$              ;BRANCH TO NEXT COMPARE
```

```
 9014  024612  010137  001122                    MOV     R1,@#BDADR      ;ADDRESS OF BAD REGISTER (RMCS1)
 9015  024616  104026                            ERROR   26              ;RMCS1 HAD SOME BITS OTHER
 9016                                                                    ;THAN DVA AND RDY SET
 9017                                                                    ;ALL OTHER BITS SHOULD BE 0
 9018  024620  013746  001724            38:     MOV     @#DS1,-(SP)     ;GET RHDS1
 9019  024624  042716  001100                    BIC     #VV!PROG,(SP)   ;CLEAR VV AND PROGRAMABLE BIT
 9020  024630  022726  000600                    CMP     #DPR!DRY,(SP)+  ;RHDS1 SHOULD HAVE THESE SET
 9021  024634  001423                            BEQ     7$              ;BRANCH IF GOOD
 9022  024636  032737  000400  001724  48:      BIT     #DPR,   @#DS1   ;TEST DRIVE PRESENT
 9023  024644  001004                            BNE     5$              ;IF NOL WAS THERE SO BRANCH
 9024  024646  010337  001122                    MOV     R3,@#BDADR      ;ADDRESS OF BAD REGISTER (RHDS1)
 9025  024652  104026                            ERROR   26              ;RHDS1 DOES NOT HAVE DPR
 9026  024654  000413                            BR      7$              ;BRANCH OUT
 9027  024656  032737  000200  001724  58:      BIT     #DRY,   @#DS1   ;TEST DRIVE READY
 9028  024664  001004                            BNE     6$              ;IF DPR WAS THERE SO BRANCH
 9029  024666  010337  001122                    MOV     R3,@#BDADR      ;ADDRESS OF BAD REGISTER (RHDS1)
 9030  024672  104026                            ERROR   26              ;RHDS1 DOES NOT HAVE DRY
 9031  024674  000403                            BR      7$              ;BRANCH OUT
 9032  024676  010337  001122            68:     MOV     R3,@#BDADR      ;ADDRESS OF BAD REGISTER (RHDS1)
 9033  024702  104026                            ERROR   26              ;RHDS1 HAS SOME BITS OTHER
 9034                                                                    ;THAN NOL, DRY, DPR, SET
 9035                                                                    ;ALL OTHER BITS SHOULD BE 0
 9036  024704  000207                    78:     RTS     PC                      ;RETURN TO TEST NO.
 9037
 9038
 9039
 9040                                    ;THIS CHECKS DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1
 9041                                    ;AND CHECKS MEDIUM ON LINE (NOL), DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1
 9042
 9043  024706  011637  002002            CHECKE: MOV     (SP),@#PCJSR    ;SAVE PC OF JSR+4
 9044  024712  162737  000004  002002            SUB     #4,@#PCJSR      ;GET PC OF JSR
 9045  024720  004737  024170                    JSR     PC,@#PUTREG     ;SAVE REGISTERS
 9046  024724  032737  000200  001702            BIT     #RDY,@#CS1      ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
 9047                                                                    ;AND BE READY
 9048  024732  001004                            BNE     1$              ;BRANCH IF GOOD
 9049  024734  010137  001122                    MOV     R1,@#BDADR      ;FAILING REGISTER
 9050  024740  104026                            ERROR   26              ;RHCS1 IS IN ERROR
 9051                                                                    ;DOES NOT HAVE DVA, RDY
 9052  024742  000427                            BR      4$              ;BRANCH
 9053  024744  032737  004000  001702  1$:      BIT     #DVA,@#CS1      ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
 9054                                                                    ;AND BE READY
 9055  024752  001004                            BNE     2$              ;BRANCH IF GOOD
 9056  024754  010137  001122                    MOV     R1,@#BDADR      ;FAILING REGISTER
 9057  024760  104026                            ERROR   26              ;RHCS1 IS IN ERROR
 9058                                                                    ;DOES NOT HAVE DVA, RDY
 9059  024762  000417                            BR      4$              ;BRANCH OUT
 9060  024764  032737  000200  001724  2$:      BIT     #DRY,@#DS1      ;RHDS1 SHOULD HAVE DPR,DRY
 9061  024772  001004                            BNE     3$              ;BRANCH IF THERE
 9062  024774  010337  001122                    MOV     R3,@#BDADR      ;FAILING REGISTER RHDS1
 9063  025000  104026                            ERROR   26              ;RHDS1 DOES NOT HAVE DPR,DRY
 9064  025002  000407                            BR      4$              ;BRANCH OUT
 9065  025004  032737  000400  001724  3$:      BIT     #DPR,@#DS1      ;RHDS1 SHOULD HAVE DPR,DRY
 9066  025012  001003                            BNE     4$              ;BRANCH IF THERE
 9067  025014  010337  001122                    MOV     R3,@#BDADR      ;FAILING REGISTER RHDS1
```

```
9068  025020  104026                        ERROR   26              ;RHD81 DOES NOT HAVE DPR,DRY
9069  025022  000207              48:       RTS     PC
9070
9071
9072
9073
9074
9075
9076                                ;       WAIT LOOP
9077                                ;       ONE  LOOP OR ONE COUNT = 5.15 MICROSEC WITH BIPOLAR MEMORY (MIN)
9078                                ;       ONE LOOP OR ONE COUNT = 11.86 MICROSEC WITH CORE (MIN)
9079                                ;       WITH CORE ERROR IS INDICATED AFTER ABOUT 650 MILLISEC (MIN)
9080  025024  177777              TIMCNT: 177777                  ;WAITING COUNT
9081  025026  010046              WAIT,T: MOV     R0,-(SP)                ;SAVE R0
9082  025030  016600  000002             MOV     2(SP),R0                ;GET ADRESS OF REG, ADDRESS
9083  025034  010037  001176             MOV     R0,@#TMP3               ;WAT PC+2 IN @TMP3
9084  025040  162737  000002  001176     SUB     #2,@#TMP3               ;WAT PC FOR TYPEOUT
9085  025046  012037  001170             MOV     (R0)+,@#TMP0            ;WAIT REGISTER ADDRESS
9086  025052  012037  001172             MOV     (R0)+,@#TMP1            ;WAIT ON BIT
9087  025056  010066  000002             MOV     R0,2(SP)                ;RESTORE RETURN ON STACK
9088  025062  012600                      MOV     (SP)+,R0                ;RESTORE R0
9089  025064  013737  025024  001174     MOV     @#TIMCNT,@#TMP2         ;TEMPORARY COUNT
9090  025072  033777  001172  154070 18:  BIT     @#TMP1,@#TMP0   ;IS REQUIRED BIT THERE?
9091  025100  001021                      BNE     2$              ;BRANCH IF YES
9092  025102  005337  001174             DEC     @#TMP2          ;COUNT
9093  025106  001371                      BNE     1$              ;BRANCH IF NOT TIME UP
9094  025110  013737  025024  001174     MOV     @#TIMCNT,@#TMP2         ;TEMPORARY COUNT
9095  025116  033777  001172  154044 38:  BIT     @#TMP1,@#TMP0   ;IS REQUIRED BIT THERE?
9096  025124  001007                      BNE     2$              ;BRANCH IF YES
9097  025126  005337  001174             DEC     @#TMP2          ;COUNT
9098  025132  001371                      BNE     3$              ;BRANCH IF NOT TIME UP
9099  025134  017737  154030  001126     MOV     @#TMP0,@#BDDAT  ;REGISTER CONTENTS
9100  025142  104016                      ERROR   16              ;WAITED ON BIT FAILED TO SET
9101  025144  000002              2$:     RTI
9102                                ;       CALL FOR THE ABOVE WAITLOOP IS
9103                                ;
9104                                ;       MOV     @A,@#X$         ;A CONTAINS REGISTER ADDRESS
9105                                ;       -       -       -       ;HENCE X$ WILL HAVE ABSOLUTE REG. ADR.
9106                                ;       -       -       -
9107                                ;       -       -       -
9108                                ;       WAT
9109                                ;X$:    0                       ;ABSOLUTE REG. ADDRESS UNDER WAIT
9110                                ;       .WORD   0               ;BIT WAITED FOR
9111                                ;                               ;CONTINUE
9112
9113
9114
9115                                ;THIS IS A SUBROUTINE TO SAVE REGISTERS
9116                                ;IN THE REGISTER TABLE TO ANY LOCATION
9117                                .SBTTL  SAVE ROUTINE
9118                                ;THE CALL IS
9119                                ;JSR    R0,@#SAVER
9120                                ;FROM
9121                                ;TO
```

```
                                        ;NUMBER OF WORDS SAVED

025146                          SAVER:
(2) 025146  010146                      MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(2) 025150  010246                      MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(2) 025152  010346                      MOV     R3,-(SP)        ;;PUSH R3 ON STACK
025154  012001                          MOV     (R0)+,R1        ;FROM
025156  012002                          MOV     (R0)+,R2        ;TO
025160  012003                          MOV     (R0)+,R3        ;NUMBER
025162  013122                  18:     MOV     @(R1)+,(R2)+    ;SAVE REGISTER CONTENTS
025164  005303                          DEC     R3              ;COUNT
025166  001375                          BNE     18              ;BRANCH IF NOT DONE
025170  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
(2) 025172  012602                      MOV     (SP)+,R2        ;;POP STACK INTO R2
(2) 025174  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
025176  000200                          RTS     R0


                                        .SBTTL  WRITE CHECK ROUTINE
                                        ;THIS IS A SUBROUTINE TO DO WRITE CHECK HEADER AND DATA
                                        ;CYLINDER 0, TRACK 1, SECTOR 1, KEYS 0

                                        ;THESE ARE TO SET UP FOR DISKLESS USE ONLY
025200  012737  010000  031016  WRCHHD: MOV     #FMT22,@#CYL    ;CYLINDER 0 FORMAT 16 BIT WORDS
025206  112737  000001  031021          MOVB    #1,@#SECOTR+1   ;TRACK=1
025214  112737  000001  031020          MOVB    #1,@#SECOTR     ;SECTOR=1
025222  005037  031022                  CLR     @#KEY1          ;KEY1=0
025226  005037  031024                  CLR     @#KEY2          ;KEY2=0
025232  012767  000044  003636          MOV     #36.,DAWORD     ;NO OF DATA WORDS
025240  005037  031026                  CLR     @#X             ;THIS IS A READ OPERATION
025244  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
025250  031016                          CYL
025252  032716                          WCRC

                                        ;THESE ARE REGULAR SETUPS

025254  004737  024470                  JSR     PC,@#CLDISK     ;SET UP GENERAL REGISTERS
                                                                ;AND CLEAR DISK REGISTERS
025260  012777  177730  154336          MOV     #-40.,@RHWC     ;36 DATA WORDS 4 HEADER WORDS
025266  012777  003130  154332          MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
025274  112746  000001                  MOVB    #1,-(SP)        ;SECTOR=1
025300  112766  000001  000001          MOVB    #1,1(SP)        ;TRACK=1 IN UPPER BYTE
025306  012677  154324                  MOV     (SP)+,@RHDST    ;TRACK=1, SECTOR=1 IN RHDST
025312  012777  014000  154322          MOV     #FMT22!ECI,@RHOF ;16 BIT WORDS
                                                                ;ECC CORRECTION INHIBIT BECAUSE
                                                                ;ECC LOGIC IS NOT CHECKED YET
025320  005077  154320                  CLR     @RHCA           ;CYLINDER=0
025324  004737  024524                  JSR     PC,@#CHECKT     ;CHECK FOR DVA,RDY,NOL,DPR,DRY
025330  013711  002036                  MOV     @#WRCHDT,@R1    ;WRITE CHECK HEADER AND DATA=B2
                                                                ;INTO RHCS1
025334  004737  030706                  JSR     PC,@#COMHD      ;WRITE CHECK HEADER AND DATA
                                                                ;SAME AS READ HEADER AND DATA
```

```
                                        RTS     PC              ;RETURN TO WRITE CHECK TEST
                                        .SBTTL  COMPARE ROUTINE
                                ;THIS IS A SUBROUTINE TO COMPARE TWO BLOCKS IN MEMORY
                                ;R1 HAS GOOD DATA BUFFER ADDRESS
                                ;R2 HAS TEST DATA BUFFER ADDRESS
                                ;$TMP0 HAS ADDRESS OF RETURN ON ERROR TO PRINT HEADER
                                ;$TMP1 HAS ADDRESS OF RETURN ON ERROR TO PRINT DATA
                                ;R3 HAS NUMBER OF WORDS TO BE COMPARED
                                ;R4 HAS ONE MORE THAN NUMBER OF WORDS TO BE COMPARED

                                COMPAR:
                                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
                                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
                                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
                                        MOV     (R0)+,R1        ;ADDRESS OF GOOD DATA BUFFER
                                        MOV     (R0)+,R2        ;ADDRESS OF TEST DATA BUFFER
                                        MOV     (R0)+,R3        ;NO OF WORDS TO BE COMPARED
                                        MOV     (R0)+,$TMP0     ;RETURN ON ERROR TO PRINT HEADER
                                        MOV     (R0)+,$TMP1     ;RETURN ON ERROR TO PRINT DATA
                                        MOV     (R0),R0         ;RETURN ON NO ERROR
                                        MOV     R3,R4           ;NO OF WORDS TO BE COMPARED
                                        INC     R4
                                1$:     MOV     R4,@#ERWORD     ;FOR ERROR WORD NO
                                        CMP     (R1)+,(R2)+     ;COMPARE GOOD WITH TEST DATA
                                        BEQ     3$              ;BRANCH IF GOOD

                                        MOV     -(R1),@#GDDAT   ;GOOD DATA
                                        MOV     -(R2),@#BDDAT   ;BAD DATA
                                        SUB     R3,@#ERWORD     ;ERROR WORD NO.
                                        TST     @#ERFLG#        ;ANY ERRORS ALREAY THERE
                                        BNE     2$              ;BRANCH IF YES
                                        JSR     PC,@#TMP0       ;RETURN TO PRINT HEADER
                                        BR      5$              ;BRANCH TO AVOID PRINTING NEXT ERROR
                                2$:     JSR     PC,@#TMP1       ;RETURN TO PRINT DATA
                                5$:     CMP     (R1)+,(R2)+     ;UNDO -(R1) AND -(R2) FOR ERRORS
                                        MOV     @#SWR,-(SP)     ;GET SWITCH SETTING
                                        BIC     #^C600,(SP)     ;KEEP ONLY SWITCH 7 AND 8
                                        CMP     #SW07,(SP)+     ;IS 7 SET AND 8 RESET
                                        BEQ     4$              ;BRANCH OUT IF YES
                                3$:     DEC     R3              ;COUNT
                                        BNE     1$              ;BRANCH IF ALL NOT DEVICE
                                4$:
                                        MOV     (SP)+,R5        ;;POP STACK INTO R5
                                        MOV     (SP)+,R4        ;;POP STACK INTO R4
                                        MOV     (SP)+,R3        ;;POP STACK INTO R3
                                        MOV     (SP)+,R2        ;;POP STACK INTO R2
                                        MOV     (SP)+,R1        ;;POP STACK INTO R1
                                        RTS     R0              ;RETURN TO MAIN PROGRAM
```

```
                                        ;THIS IS A SUBROUTINE TO DO WRITE CHECK DATA
                                        ;CYLINDER 0, TRACK 1, SECTOR 1, KEYS 0

                                        ;THESE ARE TO SET UP FOR DISKLESS USE ONLY
025504  012737  010000  031016  WRCHDA: MOV     #FMT22,@#CYL    ;CYLINDER 0 FORMAT 16 BIT WORDS
025512  112737  000001  031021          MOVB    #1,@#SECOTR+1   ;TRACK=1
025520  112737  000001  031020          MOVB    #1,@#SECOTR     ;SECTOR=1
025526  005037  031022                  CLR     @#KEY1          ;KEY1=0
025532  005037  031024                  CLR     @#KEY2          ;KEY2=0
025536  012737  000040  031076          MOV     #32.,@#DAWORD   ;NO OF DATA WORDS
025544  005037  031026                  CLR     @#X             ;THIS IS A READ OPERATION

025550  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
025554  031016                          CYL
025556  032716                          WCRC

                                        ;THESE ARE REGULAR SETUPS

025560  004737  024470                  JSR     PC,@#CLDISK     ;SET UP GENERAL REGISTERS
                                                                ;AND CLEAR DISK REGISTERS

025564  012777  177740  154032          MOV     #-32.,@RHWC     ;36 DATA WORDS 4 HEADER WORDS
025572  012777  003130  154026          MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
025600  112746  000001                  MOVB    #1,-(SP)        ;SECTOR=1
025604  112766  000001  000001          MOVB    #1,1(SP)        ;TRACK=1 IN UPPER BYTE
025612  012677  154020                  MOV     (SP)+,@RHDST    ;TRACK=1, SECTOR=1 IN RHDST
025616  012777  014000  154016          MOV     #FMT22!ECI,@RHOF        ;16 BIT WORDS
                                                                ;ECC CORRECTION INHIBIT BECAUSE
                                                                ;ECC LOGIC IS NOT CHECKED YET
025624  005077  154014          CLR     @RHCA           ;CYLINDER=0
025630  004737  024524          JSR     PC,@#CHECKT     ;CHECK FOR DVA,RDY,NOL,DPR,DRY
025634  013711  002034          MOV     @#WRCHEK,@R1    ;WRITE CHECK DATA=0 INTO RHCS1
025640  004737  030706          JSR     PC,@#COMHD      ;WRITE CHECK HEADER AND DATA
                                                        ;SAME AS READ HEADER AND DATA

025644  000207                  RTS     PC              ;RETURN TO WRITE CHECK TEST

                                        .SBTTL  CRC GENERATION ROUTINE
                                        ;THIS IS A SUBROUTINE TO CALCULATE CRC FOR THE FOUR
                                        ;HEADER WORDS AND STORE THEM IN "WCRC" AND "GCRC"
                                        ;R1 - REGISTER FOR CRC, INCREMENTED CRC VALUE IS HERE
                                        ;R2 - THIS HAS BIT POSITION 2 VALUE C
                                        ;R3 - THIS HAS BIT POSITION 16 I.E, OUTPUT BIT VALUE B
                                        ;R4 - THIS HAS BIT POSITION 15 VALUE E
                                        ;@TMP0 - NUMBER OF WORDS
                                        ;@TMP2 - NUMBER OF BITS PER WORD = 16
                                        ;@TMP3 - TEMPORARY REG,
                                        ;@TMP4 - TEMPORARY REG TO TRANSFER CARRY
                                        ;@TMP5 - THIS HAS DATA BIT VALUE D

                                        ;FETCH DATA BIT D
                                        ;B = D XOR 16
                                        ;C = B XOR 2
```

```
                                                ;E = B XOR 15
                                                ;ROTATE RIGHT ONE POSITION
                                                ;B GOES TO POSITION 1
                                                ;C GOES TO POSITION 3
                                                ;E GOES TO POSITION 16
                                                ;REPET 64 TIMES
                                                ;CALL   JSR     R5,@#CRC
                                                ;X      ;FIRST LOCATION AT
                                                ;Y      ;PUT CRC IN WCRC FOR READ GCRC FOR WRITE
                                
025646                          CRC:
 (2)  025646  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
9280  025650  012500                            MOV     (R5)+,R0        ;GET POINTER TO CYL NO.
9281  025652  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 (2)  025654  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 (2)  025656  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 (2)  025660  010446                            MOV     R4,-(SP)        ;;PUSH R4 ON STACK
9282  025662  005001                            CLR     R1              ;CLEAR WORKING LOCATION
9283  025664  005037  001202                    CLR     @#TMP5
9284  025670  012737  000004  001170            MOV     #4,@#TMP0       ;WORD COUNT
9285  025676  012037  001176          168:      MOV     (R0)+,@#TMP3    ;TEMPORARY WORD STORAGE
9286  025702  012767  000020  153264            MOV     #16.,@TMP2      ;BIT COUNT
9287  025710  013737  001176  001200            MOV     @#TMP3,@#TMP4   ;TEMPORARY WORD STORAGE
9288  025716  006037  001176          158:      ROR     @#TMP3          ;GET LSB INTO "C"
9289  025722  006037  001202                    ROR     @#TMP5          ;GET ABOVE "C" INTO @TMP5
9290  025726  032701  000001                    BIT     #BIT0,R1        ;IS POSITION 15 HIGH
9291  025732  001403                            BEQ     18              ;BRANCH IF POSITION 16 LOW
9292  025734  012703  100000                    MOV     #BIT15,R3       ;GET POSITION 16
9293  025740  000401                            BR      28
9294  025742  005003                  18:       CLR     R3              ;GET POSITION 16
9295  025744  063703  001202          28:       ADD     @#TMP5,R3       ;XOR POSITION 16 WITH D
9296                                            ;TO GIVE B
9297  025750  032701  040000                    BIT     #BIT14,R1       ;IS POSITION 2 HIGH
9298  025754  001403                            BEQ     38              ;BRANCH IF POSITION 2 LOW
9299  025756  012702  100000                    MOV     #BIT15,R2       ;GET POSITION 2
9300  025762  000401                            BR      48
9301  025764  005002                  38:       CLR     R2              ;GET POSITION 2
9302  025766  060302                  48:       ADD     R3,R2           ;XOR B WITH POSITION 2
9303                                            ;TO GIVE C
9304  025770  032701  000002                    BIT     #BIT1,R1        ;IS POSITION 15 HIGH
9305  025774  001403                            BEQ     58              ;BRANCH IF PCSITION 15 LOW
9306  025776  012704  100000                    MOV     #BIT15,R4       ;GET POSITION 15
9307  026002  000401                            BR      68
9308  026004  005004                  58:       CLR     R4              ;GET POSITION 15
9309  026006  060304                  68:       ADD     R3,R4           ;XOR POSITION 15 WITH B
9310                                            ;TO GIVE E
9311  026010  006037  001200                    ROR     @#TMP4          ;GET LSB INTO "C"
9312  026014  006001                            ROR     R1              ;GET ABOVE C INTO R1
9313  026016  005703                            TST     R3              ;TEST B
9314  026020  100403                            BMI     78              ;BRANCH IF B=1
9315  026022  042701  100000                    BIC     #BIT15,R1       ;SET B IN POSITION 1
9316  026026  000402                            BR      108
9317  026030  052701  100000          78:       BIS     #BIT15,R1       ;SET B IN POSITION 1
9318  026034  005702                  108:      TST     R2              ;TEST C
```

```
 9319  026036  100403                        BMI     11$             ;BRANCH IF C=1
 9320  026040  042701  020000                BIC     #BIT13,R1       ;GET C IN POSITION 3
 9321  026044  000402                        BR      12$
 9322  026046  052701  020000          11$:  BIS     #BIT13,R1       ;GET C IN POSITION 3
 9323  026052  005704                  12$:  TST     R4              ;TEST E
 9324  026054  100403                        BMI     13$             ;BRANCH IF E=1
 9325  026056  042701  000001                BIC     #BIT0,R1        ;GET E IN POSITION 16
 9326  026062  000402                        BR      14$
 9327  026064  052701  000001          13$:  BIS     #BIT0,R1        ;GET E IN POSITION 16
 9328  026070  005337  001174          14$:  DEC     @#TMP2          ;BIT COUNTER
 9329  026074  001310                        BNE     15$             ;BRANCH IF 16 NOT DONE
 9330  026076  005337  001170                DEC     @#TMP0          ;WORD COUNTER
 9331  026102  001275                        BNE     16$             ;BRANCH IF 4 NOT DONE
 9332  026104  010135                        MOV     R1,0(R5)+       ;PUT CRC WHERE DESIRED
 9333  026106  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
  (2)  026110  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
  (2)  026112  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
  (2)  026114  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
  (2)  026116  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 9334  026120  000205                        RTS     R5
 9335
 9336
 9337
 9338
 9339                                        ;THIS IS A SUBROUTINE TO SET UP THE SIMULATOR DISK FOR
 9340                                        ;CYLINDER 0 (16 BITS PER WORD)
 9341                                        ;TRACK 1, SECTOR 1
 9342                                        ;KEY1 1
 9343                                        ;KEY2 1
 9344                                        ;CRC THROUGH THE JSR R5,@#CRC
 9345                                        ;256 WORDS OF 177400
 9346
 9347                              ;CALL     JSR     PC,@#SETDSK
 9348
 9349  026122                    SETDSK:
  (2)  026122  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (2)  026124  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2)  026126  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 9350  026130  012700  177400                MOV     #177400,R0      ;DATA IN THE DISK
 9351  026134  012701  000400                MOV     #256.,R1        ;COUNTER
 9352  026140  012702  032734                MOV     #DISK,R2        ;START OF SIMULATOR DISK
 9353  026144  010022                  18:   MOV     R0,(R2)+        ;MOVE IN DATA
 9354  026146  005301                        DEC     R1              ;COUNT FOR 256
 9355  026150  001375                        BNE     18              ;BRANCH IF 256 NOT COMPLETE
 9356  026152  012701  000021                MOV     #17.,R1         ;2 ECC WORDS, 1 DATA GAP
 9357                                                                  ;14 TOLERANCE GAP
 9358  026156  005022                  28:   CLR     (R2)+           ;CLEAR ECC,DATA GAP AND
 9359                                                                  ;TOLERANCE GAP
 9360  026160  005301                        DEC     R1              ;COUNT
 9361  026162  001375                        BNE     28              ;BRANCH IF NOT COMPLETE
 9362
 9363                                        ;NOW SET UP FOR DISKLESS USE
 9364
 9365  026164  012737  010000  031016        MOV     #FMT22,@#CYL    ;CYLINDER 0 (16 BIT WORDS)
```

```
 9366  026172  112737  000001  031021          MOVB    #1,@#SECOTR+1   ;TRACK=1
 9367  026200  112737  000001  031020          MOVB    #1,@#SECOTR     ;SECTOR=1
 9368  026206  012737  000001  031022          MOV     #1,@#KEY1       ;KEY1=1
 9369  026214  012737  000001  031024          MOV     #1,@#KEY2       ;KEY2=1
 9370  026222  016737  152152  031076          MOV     256.,,@#DAWORD  ;NO. OF DATA WORDS
 9371  026230  004537  025646                  JSR     R5,@#CRC        ;GO TO CALCULATE CRC
 9372  026234  031016                          CYL                     ;FIRST CRC WORD
 9373  026236  032716                          WCRC                    ;PUT CALCULATED CRC
 9374  026240  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
  (2)  026242  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
  (2)  026244  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 9375  026246  000207                          RTS     PC
 9376
 9377                                         ;THIS IS A SUBROUTINE TO CHECK HEADER COMPARE ERROR
 9378                                         ;(BIT #7) AND CRC ERROR (BIT #0)
 9379                                         ;CALL JSR R0,@#HCCRCE
 9380                                         ;       COM             ;COMMAND=READ HEADER AND DATA
 9381                                         ;                       ;       =WRITE DATA
 9382                                         ;       C               ;CYLINDER
 9383                                         ;       S               ;SECTOR
 9384                                         ;       T               ;TRACK
 9385                                         ;       -N.             ;WORD COUNT
 9386                                         ;       B               ;RHBA BUFFER START
 9387                                         ;       X               ;1=WRITE DATA 0=READ
 9388                                         ;       H               ;H=1 HEADER CHECK, H=0 CRC CHECK
 9389
 9390  026250  010037  002002  HCCRCE: MOV     R0,@#PCJSR      ;SAVE PC OF JSR+4
 9391  026254  162737  000004  002002          SUB     #4,@#PCJSR      ;GET PC OF JSR
 9392  026262  004737  024470                  JSR     PC,@#CLDISK     ;INIT AND SETUP GENERAL REG.
 9393
  (1)
  (1)  026266  004767  176232                  JSR     PC,CHECKT               ;CHECK DVA, RDY, DPR, DRY
  (1)
 9394  026272  011037  001202                  MOV     (R0),@#TMPS     ;SAVE COMMAND
 9395  026276  012011                          MOV     (R0)+,@R1       ;COMMAND
 9396  026300  012077  153340                  MOV     (R0)+,@RHCA     ;CYLINDER
 9397  026304  112046                          MOVB    (R0)+,-(SP)     ;SECTOR
 9398  026306  105720                          TSTB    (R0)+           ;UP DATE R0
 9399  026310  112066  000001                  MOVB    (R0)+,1(SP)     ;TRACK
 9400  026314  105720                          TSTB    (R0)+           ;UPDATE R0
 94C1  026316  012677  153314                  MOV     (SP)+,@RHDST    ;TRACK SECTOR
 9402  026322  012077  153276                  MOV     (R0)+,@RHWC     ;NO. OF DATA WORDS +4 HEADER
 9403                                                                  ;IF A READ HEADER AND DATA
 9404  026326  012077  153274                  MOV     (R0)+,@RHBA     ;STARTING ADDRESS OF BUFFER
 9405  026332  012037  031026                  MOV     (R0)+,@#X       ;X=0 READ HEADER AND DATA
 9406                                                                  ;X=1 WRITE DATA
 9407  026336  012777  014000  153276          MOV     #FMT22!ECI,@RHOF ;16 BITS PER WORD
 9408                                                                  ;ECC CORRECTION INHIBIT
 9409  026344  005037  001774                  CLR     @#ERFLG         ;CLEAR ERROR FLAG
 9410  026350  004737  030706                  JSR     PC,@#COMHD      ;COMMAND
 9411
 9412                                         ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
 9413                                         ;FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
 9414                                         ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
```

```
                                        ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
                                        ;DETECTED
                                        ;HEADER AND DATA ARE TO BE CHECKED,
026354  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
026360  005737  001774                  TST     @#ERFLG#        ;ANY ERRORS ALREADY THERE
026364  001034                          BNE     10#             ;BRANCH IF YES
026366  005737  031026                  TST     @#X             ;IS THIS A READ
026372  001015                          BNE     3#              ;IF A WRITE DATA BRANCH

                                        ;NOW THE READ BUFFER WILL BE CHECKED
                                        ;HEADER SHOULD BE COMPLETELY READ AS WRITTEN
                                        ;NO DATA WORDS SHOULD BE READ
                                        ;REINTO BUFFER HAS BEEN FILLED WITH 0
                                        ;WRFROM BUFFER HAS BEEN FILLED WITH EXPECTED DATA

026374  004037  025342                  JSR     R0,@#COMPAR     ;CHECK
026400  002064                          WRFROM                  ;GOOD DATA
026402  003130                          REINTO                  ;TEST BUFFER
026404  000400                          256.                    ;4 HEADER 252 DATA
026406  026414                          1#                      ;RETURN POINT FOR ERROR HEADER
026410  026420                          2#                      ;RETURN POINT FOR ERROR DATA
026412  026456                          10#                     ;RETURN FOR GOOD COMPARISON
026414  104004            1#:   ERROR   4                       ;READ NEXT ERROR 5
026416  000207                  RTS     PC                      ;RETURN TO COMPARISON SUBROUTINE
026420  104005            2#:   ERROR   5                       ;WORD NO 1 THRU 4 ARE
                                        ;HEADER WORDS AND HENCE
                                        ;SHOULD BE READ AS WRITTEN ON
                                        ;DISK, WORD NOS. 5 ONWARDS
                                        ;SHOULD NOT BE READ AND HENCE
                                        ;READ INTO BUFFER
                                        ;SHOULD BE UNCHANGED
026422  000207                  RTS     PC                      ;RETURN TO COMPARISON

026424  000414                  BR      10#             ;JUMP OUT

                                        ;NOW THE DISK WILL BE CHECKED
                                        ;NO DATA SHOULD BE WRITTEN
                                        ;REINTO BUFFER HAS BEEN FILLED WITH EXPECTED DATA
                                        ;DISK HAS BEEN FILLED WITH 177400
                                        ;WRFROM HAS BEEN FILLED WITH 125252

026426  004037  025342            3#:   JSR     R0,@#COMPAR     ;CHECK
026432  003130                          REINTO                  ;GOOD DATA BUFFER
026434  032734                          DISK                    ;TEST BUFFER
026436  000400                          256.
026440  026446                          4#                      ;RETURN POINT FOR ERROR HEADER
026442  026452                          5#                      ;RETURN POINT FOR ERROR DATA
026444  026456                          10#                     ;RETURN POINT FOR GOOD COMPARISON
026446  104004            4#:   ERROR   4                       ;READ NEXT ERROR 5
026450  000207                  RTS     PC                      ;RETURN TO COMPARISON SUBROUTINE
026452  104005            5#:   ERROR   5                       ;WORD NO ARE ALL DATA
                                        ;WORDS THE SHOULD NOT
                                        ;HAVE BEEN CHANGED BY THE
                                        ;WRITE COMMAND
```

```
9469  026454  000207                          RTS     PC              ;RETURN TO COMPARISON SUBROUTINE
9470  026456  005720                  106:    TST     (R0)+           ;IS THIS A MCRC ON MCE CHECK?
9471  026460  001442                          BEQ     6$              ;BRANCH IF MCRC
9472  026462  022737  000072  001202          CMP     #72,@#TMPS      ;IS THIS A READ COMMAND
9473  026470  001417                          BEQ     11$             ;BRANCH IF YES
9474  026472  017737  153136  001126          MOV     @RHER1,@#BDDAT  ;TEST DATA
9475  026500  022737  000200  001126          CMP     #MCE,@#BDDAT    ;ONLY HEADER COMPARE BIT?
9476                                                                  ;SHOULD BE SET
9477  026506  001470                          BEQ     7$              ;BRANCH IF GOOD
9478  026510  013737  001634  024234          MOV     @#RHER1,@#REGADR ;REGISTER ADDRESS RHER1
9479  026516  012737  000200  001124          MOV     #MCE,@#GDDAT    ;GOOD DATA
9480  026524  104027                          ERROR   27              ;AFTER AN ERROR ON THE
9481                                                                  ;HEADER ONLY MCE SHOULD
9482  026526  000460                          BR      7$              ;BE SET
9483  026530                          116:
9484  026530  017737  153100  001126          MOV     @RHER1,@#BDDAT  ;TEST DATA
9485  026536  022737  100200  001126          CMP     #DCK!MCE,@#BDDAT        ;ONLY HEADER COMPARE BIT?
9486                                                                  ;SHOULD BE SET
9487                                                                  ;DCK IS SET BECAUSE ECC IS NOT READ
9488  026544  001451                          BEQ     7$              ;BRANCH IF GOOD
9489  026546  013737  001634  024234          MOV     @#RHER1,@#REGADR ;REGISTER ADDRESS RHER1
9490  026554  012737  100200  001124          MOV     #DCK!MCE,@#GDDAT        ;GOOD DATA
9491  026562  104027                          ERROR   27              ;AFTER AN ERROR ON THE
9492                                                                  ;HEADER ONLY MCE SHOULD
9493  026564  000441                          BR      7$              ;BE SET
9494  026566  022737  000072  001202  66:     CMP     #72,@#TMPS      ;IS THIS A READ COMMAND?
9495  026574  001417                          BEQ     12$             ;BRANCH IF A READ
9496  026576  017737  153032  001126          MOV     @RHER1,@#BDDAT  ;TEST DATA
9497  026604  022737  000400  001126          CMP     #MCRC,@#BDDAT   ;ONLY CRC ERROR SHOULD BE THERE
9498  026612  001426                          BEQ     7$
9499  026614  013737  001634  024234          MOV     @#RHER1,@#REGADR ;REG. ADDR = RHER1
9500  026622  012737  000400  001124          MOV     #MCRC,@#GDDAT   ;GOOD DATA
9501  026630  104027                          ERROR   27              ;AFTER A CRC ERROR ONLY CRC
9502                                                                  ;SHOULD BE SET
9503  026632  000416                          BR      7$              ;BRANCH OUT
9504  026634  017737  152774  001126  126:    MOV     @RHER1,@#BDDAT  ;TEST DATA
9505
9506  026642  022737  100400  001126          CMP     #DCK!MCRC,@#BDDAT;MCRC AND DCK SHOULD BE SET
9507                                                                  ;DCK IS SET BECAUSE ECC IS NOT READ
9508  026650  001407                          BEQ     7$              ;BRANCH IF GOOD
9509  026652  012737  100400  001124          MOV     #DCK!MCRC,@#GDDAT;GOOD DATA
9510  026660  013737  001634  024234          MOV     @#RHER1,@#REGADR;FAILING REGISTER RHER1
9511  026666  104027                          ERROR   27              ;AFTER A CRC ERROR ON A READ
9512                                                                  ;DCK AND MCRC SHOULD BE SET
9513                                                                  ;DCK IS SET BECAUSE ECC IS NOT READ
9514  026670  000200                  7$:     RTS     R0              ;RETURN TO MAIN TEST
9515
9516
9517
9518                                          ;THIS IS A SUBROUTINE TO LEAVE AT THE MIDDLE OF
9519                                          ;A WRITE HEADER AND DATA COMMAND
9520                                          ;IT TRYS TO GET SECTOR 10, TRACK 0, CYLINDER 0
9521                                          ;BUT COMES OUT AFTER ONE SECTOR
9522                                          ;THE COMMAND OS JSR PC,@#MIDDLE
```

```
                                                        ;BAI IS SET

        026672                                  MIDDLE:
        026672  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
        026674  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
        026676  013777  002042  152726                  MOV     @#WRIFOR,@RHCS1 ;WRITE HEADER AND DATA=62
                                                                                ;IN RHCS1
        026704  012777  177766  152712                  MOV     #-10.,@RHWC     ;10 WORDS
        026712  012777  002064  152706                  MOV     #WRFROM,@RHBA   ;BUS ADDRESS=WRFROM
        026720  012777  000010  152710                  MOV     #10,@RHDST      ;DESIRED TRACK=0 SECTOR=10
        026726  052777  000010  152674                  BIS     #BAI,@RHCS2     ;BUS ADDRESS INCREMENT INHIBIT
        026734  012777  010000  152700                  MOV     #FMT22,@RHOF    ;FORMAT 16 BIT WORDS
        026742  005077  152676                          CLR     @RHCA           ;CYLINDER=0
        026746  012737  000001  026774                  MOV     #1,@#MID        ;SECTOR IS SET TO 1 SO THAT
                                                                                ;WE CAN GET OUT AT THE
                                                                                ;MIDDLE OF AN OPERATION
                                                                                ;LOOKING FOR SECTOR 10
        026754  012777  000001  152670                  MOV     #DMD,@RHMR      ;SET DIAGNOSTIC MODE
        026762  052777  000001  152642                  BIS     #GO,@RHCS1              ;GO TO RHCS1 WITH 62
        026770  004137  035032                          JSR     R1,@#SEARCH
        026774  000000                          MID:    .WORD   0               ;SECTOR
        026776  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
        027000  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
        027002  000207                                  RTS     PC



                                                        .SBTTL  JAM CURRENT CYLINDER ROUTINE
                                                ;THIS SUBROUTINE WILL CHANGE THE CURRENT CYLINDER REGISTER
                                                ;THIS IS DONE BY GIVING A SEEK COMMAND THEN AN INIT
                                                ;WHICH WILL LOAD THE CURRENT CYLINDER WITH THE DESIRED CYLINDER VALUE
                                                ;
                                                ;CALL IS
                                                ;JSR    R0,@#MAKECYL
                                                ;XC             ;DESIRED VALUE OF CURRENT CYLINDER


        027004                                  MAKECYL:
        027004  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
        027006  010037  002002                          MOV     R0,@#PCJSR      ;PC OF JSR+4
        027012  162737  000004  002002                  SUB     #4,@#PCJSR      ;SAVE PC OF JSR
        027020  012005                                  MOV     (R0)+,R5        ;GETTING READY TO FILL DESIRED CYLINDER
        027022  010577  152616                          MOV     R5,@RHCA        ;FILL DESIRED CYLINDER REGISTER
        027026  005077  152604                          CLR     @RHDST          ;MAKE SURE DESIRED SECTOR TRACK IS NOT ILLEGAL
        027032  013777  002050  152572                  MOV     @#SEECOM,@RHCS1 ;FILL SEEK COMMAND
        027040  012777  000001  152604                  MOV     #DMD,@RHMR      ;SET DIAGNOSTIC MODE
        027046  052777  000001  152556                  BIS     #GO,@RHCS1      ;GO TO SEEK
        027054  000240                                  NOP                     ;ALLOW TIME FOR SEEK TO HANG UP
        027056  000240                                  NOP                     ;ALLOW TIME FOR SEEK TO HANG UP
        027060  000240                                  NOP                     ;ALLOW TIME FOR SEEK TO HANG UP
        027062  000240                                  NOP                     ;ALLOW TIME FOR SEEK TO HANG UP
        027064  004737  024470                          JSR     PC,@#CLDISK     ;GIVE INIT
        027070  017737  152574  001126                  MOV     @RHCC,@#$BDDAT  ;TEST DATA
        027076  020537  001126                          CMP     R5,@#$BDDAT     ;COMPARE CURRENT CYLINDER
```

```
 9573  027102  001406                          BEQ     18              ;BRANCH IF GOOD
 9574  027104  010537  001124                  MOV     R5,@#GDDAT      ;GOOD VALUE OF RHCC
 9575  027110  013737  001670  024234          MOV     @#RHCC,@#REGADR ;FAILING REGISTER ADDRESS
 9576  027116  104030                          ERROR   30              ;CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER
 9577                                                                  ;REGISTER AFTER A SEEK AND AN INIT
 9578  027120                          18:
  (2)  027120  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
 9579  027122  000200                          RTS     R0
 9580
 9581
 9582                                          .SBTTL  ECC GENERARION AND COMPARISON ROUTINE
 9583
 9584
 9585
 9586
 9587                                  ;THIS SUBROUTINE GENERATES AND TESTS ECC
 9588                                  ;CALL    JSR PC,ECTEST
 9589
 9590
 9591
 9592
 9593
 9594          100000                  PIE1    =100000
 9595          040000                  PIE2    =40000
 9596          020000                  PIE3    =20000
 9597          010000                  PIE4    =10000
 9598          004000                  PIE5    =4000
 9599          002000                  PIE6    =2000
 9600          001000                  PIE7    =1000
 9601          000400                  PIE8    =400
 9602          000200                  PIE9    =200
 9603          000100                  PIE10   =100
 9604          000040                  PIE11   =40
 9605          000020                  PIE12   =20
 9606          000010                  PIE13   =10
 9607          000004                  PIE14   =4
 9608          000002                  PIE15   =2
 9609          000001                  PIE16   =1
 9610          100000                  PIE17   =100000
 9611          040000                  PIE18   =40000
 9612          020000                  PIE19   =20000
 9613          010000                  PIE20   =10000
 9614          004000                  PIE21   =4000
 9615          002000                  PIE22   =2000
 9616          001000                  PIE23   =1000
 9617          000400                  PIE24   =400
 9618          000200                  PIE25   =200
 9619          000100                  PIE26   =100
 9620          000040                  PIE27   =40
 9621          000020                  PIE28   =20
 9622          000010                  PIE29   =10
 9623          000004                  PIE30   =4
 9624          000002                  PIE31   =2
 9625          000001                  PIE32   =1
```

```
 9626
 9627  027124  000000                ECDATA: 0               ;DATA BIT FOR ECC
 9628                                                        ;IF ALL ONES THEN CURRENT BIT IS A ONE
 9629                                                        ;IF ZERO THEN CURRENT BIT IS A ZERO
 9630
 9631  027126  000000                GECC1:  0               ;LOW ORDER ECC WORD TO BE GENERATED HERE
 9632                                                        ;=R1
 9633
 9634  027130  000000                GECC2:  0               ;HIGH ORDER ECC WORD TO BE GENERATED HERE
 9635                                                        ;=R2
 9636
 9637  027132  000000                TSECCG: 0               ;IF =177777 GENERATE AND TEST ECC FOR THIS BIT
 9638                                                        ;IF =0 DO NOT GENERATE AND TEST ECC FOR THIS BIT
 9639
 9640  027134  113713                NCODE:  38859.          ;N-CODE WORD
 9641  027136  000000                NCOUNT: 0       ;TEMPORARY N CODE
 9642  027140  000000                POSITI: 0               ;POSITION REGISTER
 9643  027142  010041                HARDER: 4129.           ;HARD ERROR COUNT
 9644                                                        ;TRUE COUNT IS 4128 BUT AS COMPARES ARE
 9645                                                        ;DONE ONE STAGE LATER SO 4129
 9646  027144  000000                DATENV: 0               ;DATA ENVELOPE FOR TYPE OUT
 9647                                                        ;MAX FOR WRITE IS 4096
 9648                                                        ;MAX FOR READ IS 4128
 9649  027146  000000                ZCODE:  0               ;LEADING ZEROS ENVELOPE FOR TYPE OUT
 9650                                                        ;THIS IS SHUT OFF WHEN POSITION COUNTER
 9651                                                        ;IN ENABLED
 9652                                                        ;MAX COUNT IS 38859
 9653
 9654
 9655
 9656  027150  000000                HADTMP: 0               ;TEMPORARY HARD ERROR COUNT
 9657  027152  000000                P3:     0
 9658  027154  000000                P12:    0
 9659  027156  000000                P22:    0
 9660  027160  000000                P24:    0
 9661
 9662
 9663
 9664
 9665
 9666  027162                        ECTEST:
  (2)  027162  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (2)  027164  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2)  027166  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (2)  027170  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (2)  027172  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (2)  027174  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 9667  027176  013701  027126                MOV     @#GECC1,R1              ;ECC1 WORD
 9668  027202  013702  027130                MOV     @#GECC2,R2              ;ECC2 WORD
 9669  027206  005737  027124                TST     @#ECDATA                ;IS CURRENT BIT A ONE
 9670  027212  001406                        BEQ     2$                      ;BRANCH IF CURRENT DATA D=0
 9671                                        ;IF CARRY IS NOT ZERO THEN D=1
 9672                                        ;INVERT X32 TO GIVE R0
 9673  027214  010103                1$:     MOV     R1,R3
```

F11

```
 9674  027216  052703  177776                BIS     #^CPIE32,R3
 9675  027222  005103                        COM     R3
 9676  027224  010300                        MOV     R3,R0
 9677  027226  000404                        BR      3$
 9678
 9679                                        ;IF CARRY IS ZERO THEN D=0
 9680                                        ;X32 BECOMES R0
 9681  027230  010103                2$:     MOV     R1,R3
 9682  027232  042703  177776                BIC     #^CPIE32,R3
 9683  027236  010300                        MOV     R3,R0
 9684
 9685  027240  000241                3$:     CLC
 9686  027242  006000                        ROR     R0
 9687  027244  006000                        ROR     R0
 9688  027246  005700                        TST     R0
 9689  027250  001462                        BEQ     10$             ;BRANCH IF R0=0
 9690                                        ;INVERT X2
 9691
 9692  027252  010203                        MOV     R2,R3
 9693  027254  052703  137777                BIS     #^CPIE2,R3
 9694  027260  005103                        COM     R3
 9695  027262  010337  027152                MOV     R3,@#P3
 9696  027266  006237  027152                ASR     @#P3
 9697
 9698                                        ;INVERT X11
 9699
 9700
 9701  027272  010203                        MOV     R2,R3
 9702  027274  052703  177737                BIS     #^CPIE11,R3
 9703  027300  005103                        COM     R3
 9704  027302  010337  027154                MOV     R3,@#P12
 9705  027306  006237  027154                ASR     @#P12
 9706
 9707                                        ;INVERT X21
 9708
 9709  027312  010103                        MOV     R1,R3
 9710  027314  052703  173777                BIS     #^CPIE21,R3
 9711  027320  005103                        COM     R3
 9712  027322  010337  027156                MOV     R3,@#P22
 9713  027326  006237  027156                ASR     @#P22
 9714
 9715                                        ;INVERT X23
 9716
 9717  027332  010103                        MOV     R1,R3
 9718  027334  052703  176777                BIS     #^CPIE23,R3
 9719  027340  005103                        COM     R3
 9720  027342  010337  027160                MOV     R3,@#P24
 9721  027346  006237  027160                ASR     @#P24
 9722
 9723                                        ;NOW THAT R0 FOR POSITION 1
 9724                                        ;         P3 FOR POSITION 3
 9725                                        ;        P12 FOR POSITION 12
 9726                                        ;        P22 FOR POSITION 22
 9727                                        ;        P24 FOR POSITION 24
```

```
                                        ;ARE KNOWN THE ROTATE WILL BE DONE AND
                                        ;THESE BITS JAMED IN
                                        ;
027352  006002                          ROR     R2
027354  006001                          ROR     R1
027356  053700  027152                  BIS     @#P3,R0
027362  053700  027154                  BIS     @#P12,R0
027366  042702  120020                  BIC     #PIE1!PIE3!PIE12,R2
027372  050002                          BIS     R0,R2
                                        ;
027374  005000                          CLR     R0
027376  053700  027156                  BIS     @#P22,R0
027402  053700  027160                  BIS     @#P24,R0
027406  042701  002400                  BIC     #PIE22!PIE24,R1
027412  050001                          BIS     R0,R1
027414  000404                          BR      128
                                        ;
                                        ;THE PROGRAM COMES HERE IF R0=0
                                        ;SO AFTER ROTATE R0 GETS PUT INTO POSITION 1
027416  006002                  108$:   ROR     R2
027420  006001                          ROR     R1
027422  042702  100000                  BIC     #PIE1,R2
027426  010137  027126          128$:   MOV     R1,@#GECC1      ;SAVE ECC1
027432  010237  027130                  MOV     R2,@#GECC2      ;SAVE ECC2
027436  005737  027132                  TST     @#TSECCG        ;IS HARDWARE TO BE CHECKED
                                        ;IF =1777777 TEST HARDWARE
                                        ;IF = 0 DO NOT TEST HARDWARE
027442  001432                          BEQ     148             ;BRANCH IF HARDWARE NOT TO BE CHECKED
                                        ;
                                        ;CHECK HARDWARE
027444  032737  000400  177570          BIT     #SW8,@#SWR      ;IS SWITCH 8 SET
027452  001005                          BNE     158             ;BRANCH IF SW8 IS SET
027454  032737  000100  177570          BIT     #SW6,@#SWR      ;IS SWITCH 6 SET
027462  001401                          BEQ     158             ;BRANCH IF SW6 IS NOT SET
027464  000421                          BR      148             ;IF SWITCH 8 IS NOT SET AND
                                        ;SWITCH 6 IS SET THEN
                                        ;DO NOT DO COMPARES
027466  010146                  158$:   MOV     R1,-(SP)        ;GOOD PATTERN REGISTER
027470  042716  174000                  BIC     #174000,(SP)    ;GET ONLY PATTERN BITS
027474  022677  152164                  CMP     (SP)+,@RHEC2    ;COMPARE PATTERN REGISTER
027500  001404                          BEQ     138             ;BRANCH IF GOOD
                                        ;TO SAVE TIME
027502  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
027506  104035                          ERROR   35              ;PATTERN REGISTER IN 11 BITS IN ERROR
027510  000407                          BR      148             ;BRANCH OUT
027512  023777  027140  152142  138$:   CMP     @#POSITI,@RHEC1 ;COMPARE POISTION REGISTER
027520  001403                          BEQ     148             ;BRANCH IF GOOD
                                        ;TO SAVE TIME
027522  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
027526  104035                          ERROR   35              ;POSITION REGISTER IN ERROR
                                        ;"DATA ENVLOP" GIVES NUMBER OF CLOCK
                                        ;PULSES FROM BEGINING OF COMMAND
                                        ;THAT IS THE CLOCKS IN THE R/W DATA FIELD ENVELOPE
```

```
                                                                        ;
                                                                        ;IN A WRITE THERE ARE 10000 OCTAL CLOCKS
                                                                        ;IN A READ THERE ARE 10040 OCTAL CLOCKS
                                                                        ;
                                                                        ;
                                                                        ;"N-CODE ZEROS" GIVE THE NUMBER OF CLOCKS
                                                                        ; GIVEN FOR THE LEADING ZEROS FIELD
                                                                        ;MAX COUNT IS 113713 OCTAL
                                                                        ;
                                                                        ;"GOOD POSITION" GIVES NUMBER OF CLOCKS
                                                                        ;GIVEN AFTER LEADING ZEROS WHICH IS FOR THE DATA
                                                                        ;FIELD
                                                                        ;MAX COUNT IS 10040 OR 10041 OCTAL


027530                                  1481:
 (2) 027530  012605                             MOV     (SP)+,R5        ;;POP STACK INTO R5
 (2) 027532  012604                             MOV     (SP)+,R4        ;;POP STACK INTO R4
 (2) 027534  012603                             MOV     (SP)+,R3        ;;POP STACK INTO R3
 (2) 027536  012602                             MOV     (SP)+,R2        ;;POP STACK INTO R2
 (2) 027540  012601                             MOV     (SP)+,R1        ;;POP STACK INTO R1
 (2) 027542  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
027544  000207                                  RTS     PC


                                        ;THIS SUBROUTINE WILL CONTROL THE ECC GENERATION ROUTINE
                                        ;FOR ERROR CORRECTION PROCESS
                                        ;CALL   JSR,    PC,@#ECORR
                                        ;       XP      ;EXPECTED POSITION REGISTER WHEN CORRECTION IS COMPLETE


027546  000000                          ERPOS:  0               ;POSITION REG, WHEN CORRECTION IS COMPLETE


027550  010037  002002                  ECORR:  MOV     R0,@#PCJSR      ;SAVE PC OF JSR + 4
027554  162737  000004  002002                  SUB     #4,@#PCJSR      ;SAVE PC OF JSR
027562  012037  027546                          MOV     (R0)+,@#ERPOS   ;GET POSITION REG, WHEN CORRECTION IS COMPLETE
027566  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
027570  013701  001652                          MOV     @#RHMR,R1       ;MAINTENANCE REGISTER
027574  012711  000001                          MOV     #DMD,@R1        ;SET DIAGNOSTIC MODE BIT
027600  005037  027124                          CLR     @#ECDATA        ;ECC DATA IS ZERO


027604  005737  027140                  18:     TST     @#POSITI        ;IS  SOFTWARE POSITION NON ZERO
027610  001007                                  BNE     28              ;BRANCH IF N-CODE 8 COMPLETE
027612  005337  027136                          DEC     @#NCOUNT        ;DECREMENT N-CODE
027616  001001                                  BNE     68              ;BRANCH IF N-CODE IS NOT COMPLETE
027620  000403                                  BR      28              ;BRANCH  AS N-CODE IS COMPLETE
027622  005237  027146                  68:     INC     @#ZCODE         ;INCREMENT CLOCKS GIVEN FOR LEADING ZEROS
027626  000420                                  BR      38              ;BRANCH AS N-CODE IS NOT COMPLETE
                                                                        ;GO TO GIVE CLOCK AND TEST ECC
```

```
 9830  027630  005237  027140          281:    INC     @#POSITI        ;INCREMENT SOFTWARE POSITION
 9831  027634  023737  027546  027140          CMP     @#ERPOS,@#POSITI;HAVE ENOUGH CLOCKS BEEN GIVEN TO DETECT ERROR
 9832  027642  103012                          BHIS    38              ;BRANCH IF MORE CLOCKS TO BE GIVEN
 9833  027644  023737  027150  027140          CMP     @#HADTMP,@#POSITI;HAVE ENOUGH CLOCKS BEEN GIVEN FOR HARD ERROR
 9834                                                                  ;THAT IS HAVE 4120 MORE CLOCKS BEEN GIVEN
 9835  027652  001415                          BEQ     58              ;BRANCH IF YES
 9836  027654  032711  000400                  BIT     #ZER,@R1        ;CHECK ZERO DETECT BIT IN RHMR
 9837  027660  001016                          BNE     48              ;BRANCH IS ZER SET
 9838                                          ;TO SAVE TIME
 9839  027662  004737  024170                  JSR     PC,@#PUTREG     ;SAVE REGISTERS
 9840  027666  104034                          ERROR   34              ;ZERO DETECT BIT NOT HIGH
 9841                                                                  ;WHEN 21 BITS IN ECC 32 BIT REGISTER IS  0
 9842
 9843
 9844  027670  052711  000002          381:    BIS     #MCLK,@R1       ;SET CLOCK
 9845  027674  042711  000002                  BIC     #MCLK,@R1       ;CLEAR CLOCK
 9846  027700  004737  027162                  JSR     PC,@#ECTEST     ;GO TO GENERATE AND TEST ECC
 9847  027704  000737                          BR      18              ;CONTINUE
 9848
 9849                                          ;THIS EXTRA CLOCK IS TO BRING ECH HIGH
 9850                                          ;AFTER THIS CLOCK POSITION REGISTER MAY BE 10040 OR 10041 OCTAL
 9851  027706  052711  000002          581:    BIS     #MCLK,@R1       ;SET CLOCK
 9852  027712  042711  000002                  BIC     #MCLK,@R1       ;CLEAR CLOCK
 9853
 9854  027716                          481:
  (2)  027716  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 9855  027720  000200                          RTS     R0
 9856
 9857
 9858
 9859
 9860
 9861
 9862                                  ;THIS SUBROUTINE GENERATES THE ECC FOR WHAT IS ON DISK AND INSERTS THEM
 9863                                  ;ON LOCATIONS "DISK+1000" AND "DISK+1002"
 9864
 9865
 9866
 9867  027722                          FILLEC1:
  (2)  027722  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (2)  027724  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2)  027726  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (2)  027730  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (2)  027732  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (2)  027734  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 9868  027736  005037  027140                  CLR     @#POSITI        ;CLEAR POSITION
 9869  027742  005037  027126                  CLR     @#GECC1         ;CLEAR GECC1
 9870  027746  005037  027130                  CLR     @#GECC2         ;CLEAR
 9871  027752  012701  032734                  MOV     #DISK,R1        ;POINTER TO DATA FOR ECC GENERATION
 9872  027756  012702  000400                  MOV     #256.,R2        ;COUNTER FOR NUMBER OF DATA WORDS
 9873  027762  012703  000020          981:    MOV     #16.,R3         ;COUNTER FOR NUMBER OF BITS PER WORD
 9874  027766  012104                          MOV     (R1)+,R4        ;DATA IN R4
 9875  027770  006004                  1081:   ROR     R4              ;GET ONE DATA BIT IN CARRY
 9876  027772  103004                          BCC     118             ;BRANCH IF DATA BIT IS ZERO
```

```
  9877  027774  012737  177777  027124          MOV     #-1,@#ECDATA    ;ECC DATA BIT IS A ONE
  9878  030002  000402                          BR      12$             ;BRANCH TO GENERATE ECC
  9879  030004  005037  027124          11$:    CLR     @#ECDATA        ;ECC DATA BIT IS A ZERO
  9880  030010  004737  027162          12$:    JSR     PC,@#ECTEST     ;GO TO GENERATE ECC
  9881  030014  005303                          DEC     R3              ;DECREMENT BIT COUNT
  9882  030016  001364                          BNE     10$             ;BRANCH IF 16 BITS NOT DONE
  9883  030020  005302                          DEC     R2              ;DECREMENT WORD COUNT
  9884  030022  001357                          BNE     9$              ;BRANCH IF 256 WORDS NOT DONE
  9885  030024  013737  027126  033734          MOV     @#GECC1,@#DISK+<256.*2>;INSERT ECC1 ON DISK
  9886  030032  013737  027130  033736          MOV     @#GECC2,@#DISK+<257.*2>;INSERT ECC2 ON DISK
  9887  030040  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
   (2)  030042  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
   (2)  030044  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
   (2)  030046  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
   (2)  030050  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
   (2)  030052  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
  9888  030054  000207                          RTS     PC
  9889
  9890
  9891
  9892
  9893
  9894                                          .SBTTL  RH BASE ADDRESS CHANGE ROUTINE
  9895                                  ;*      THIS ROUTINE WILL ALLOW THE CHANGE OF THE BASE
  9896                                  ;*      ADDRESS FROM 176700 TO ANY TYPED VALUE
  9897
  9898  030056                          BASECH:
   (1)  030056  104400  030064                  TYPE    ,.+4            ;;TYPE ASCIZ STRING
   (1)  030062  000424                          BR      64$             ;;GET OVER THE ASCIZ
   (1)                                          ;;,ASCIZ        <15><12>/PRESENT BASE ADDRESS OF REGISTERS IS/
   (1)  030134                          64$:
  9899  030134  013746  001632                  MOV     @#RHCS1,-(SP)   ;GET READY TO TYPE OLD BASE
  9900  030140  104402                          TYPOC
  9901  030142  104400  030150                  TYPE    ,.+4            ;;TYPE ASCIZ STRING
   (1)  030146  000425                          BR      65$             ;;GET OVER THE ASCIZ
   (1)                                          ;;,ASCIZ        <15><12>/TYPE NEW BASE ADDRESS FOLLOWED BY 'CR'/
   (1)  030222                          65$:
  9902  030222  104416                          RDOCT
  9903  030224  012700  001622                  MOV     #RHDB,R0        ;GET STARTING ADDRESS OF RGISTERS
  9904  030230  012701  000024                  MOV     #20.,R1         ;NUMBER OF REGISTERS
  9905  030234  042710  177700          1$:     BIC     #^C77,(R0)      ;CLEAR OLD BASE
  9906  030240  051620                          BIS     (SP),(R0)+      ;SET NEW BASE
  9907  030242  005301                          DEC     R1              ;COUNT
  9908  030244  001373                          BNE     1$              ;BRANCH IF 20 NOT DONE
  9909  030246  104400  030254                  TYPE    ,.+4            ;;TYPE ASCII STRING
   (1)  030252  000417                          BR      66$             ;;GET OVER THE ASCIZ
   (1)                                          ;;,ASCIZ        <15><12>/PRESENT VECTOR ADDRESS IS /
   (1)  030312                          66$:
  9910  030312  013746  001620                  MOV     @#RPVEC,-(SP)   ;GET READY TO TYPE OLD VECTOR ADDRESS
  9911  030316  104402                          TYPOC
  9912  030320  104400  030326                  TYPE    ,.+4            ;;TYPE ASCII STRING
   (1)  030324  000437                          BR      67$             ;;GET OVER THE ASCIZ
   (1)                                          ;;,ASCIZ        <15><12>/TYPE NEW VECTOR ADDRESS OR RETYPE OLD ONE FOLLOWED BY "CR
   (1)  030424                          67$:
```

```
  9913  030424  104416                          RDOCT
  9914  030426  012637  001620                  MOV     (SP)+,@#RPVEC   ;SETUP VECTOR ADDRESS
  9915  030432  104400  030440                  TYPE    ,.+4            ;;TYPE ASCIZ STRING
   (1)  030436  000421                          BR      688             ;;GET OVER THE ASCIZ
   (1)                                          ;;,ASCIZ        <15><12>/RESTART PROGRAM FROM 200 OR 210/
   (1)  030502                          688:
  9916  030502  104400  030510                  TYPE    ,.+4            ;;TYPE ASCIZ STRING
   (1)  030506  000414                          BR      698             ;;GET OVER THE ASCIZ
   (1)                                          ;;,ASCIZ        <15><12>/NEW BASE WILL REMAIN/
   (1)  030540                          698:
  9917  030540  013746  001632                  MOV     @#RHCS1,-(SP)
  9918  030544  104402                          TYPOC
  9919  030546  104400  030554                  TYPE    ,.+4            ;;TYPE ASCIZ STRING
   (1)  030552  000415                          BR      708             ;;GET OVER THE ASCIZ
   (1)                                          ;;,ASCIZ        <15><12>/NEW VECTOR WILL REMAIN /
   (1)  030606                          708:
  9920  030606  013746  001620                  MOV     @#RPVEC,-(SP)
  9921  030612  104402                          TYPOC
  9922  030614  104400  030622                  TYPE    ,.+4            ;;TYPE ASCIZ STRING
   (1)  030620  000416                          BR      718             ;;GET OVER THE ASCIZ
   (1)                                          ;;,ASCIZ        <15><12>/UNTIL PROGRAM IS RELOADED/
   (1)  030656                          718:
  9923  030656  000000                          HALT
  9924
  9925
  9926
  9927
  9928
  9929                                ;THIS IS A LITTLE ROUTINE THAT TESTS NED BIT 11 IN RHCS2
  9930                                ;THIS LOOPS HERE FOR EVER
  9931                                ;TO BE USED ONLY IF DRIVES PRESENT LOOKING AT NED DOES NOT AGREE
  9932                                ;WITH WHAT IS REALY THERE
  9933  030660  000000                ERUNIT: 0                       ;UNIT UNDER MANUAL TEST
  9934  030662  004737  024470       ERSTART:JSR     PC,@#CLDISK     ;SET GENERAL REG,
  9935  030666  013712  030660               MOV     @#ERUNIT,@R2    ;SELECT UNIT
  9936  030672  005714                18:     TST     @R4             ;TEST RHER;
  9937  030674  032712  010000               BIT     #NED,@R2        ;TEST NED
  9938  030700  001401                       BEQ     28              ;BRANCH IF GOOD
  9939  030702  000773                       BR      18              ;NED NOT SET
  9940  030704  000772                28:     BR      18              ;NED SET
```

```
                        .SBTTL  DISK SIMULATION
                ;*****************************************************************
                ;*****************************************************************
                ;IN A WRITE HEADER AND DATA COMMAND FILL THE FOLLOWING
                ;WCLY=WITH CYLINDER TO BE ON DISK
                ;WSECTR=WITH SECTOR AND TRACK TO BE ON DISK
                ;WKEY1= WITH KEY1 TO BE ON DISK
                ;WKEY2= WITH KEY2 TO BE ON DISK
                ;FNWORD= NO OF DATA WORDS TO BE WRITTEN ON DISK
                ;THE COMMAND THEN IS   JSR PC,COMWHD
                ;
                ;
                ;
                ;
                ;IN A WRITE DATA COMMAND FILL THE FOLLOWING
                ;CYL=WITH CYLINDER TO BE FOUND ON DISK
                ;SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
                ;KEY1= WITH KEY1 TO BE FOUND ON DISK
                ;KEY2= WITH KEY2 TO BE FOUND ON DISK
                ;X= 1  MUST BE ONE
                ;NOWORD= WITH NUMBER OF DATA WORDS TO BE WRITTEN
                ;THE COMMAND THEN IS JSR PC,COMMD
                ;
                ;
                ;
                ;
                ;
                ;
                ;IN A READ HEADER AND DATA COMMAND FILL THE FOLLOWING
                ;CYL= WITH CYLINDER TO BE FOUND ON DISK
                ;SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
                ;KEY1= WITH KEY1 TO BE FOUND ON DISK
                ;KEY2=WITH KEY2 TO BE FOUND ON DISK
                ;DAWORD= WITH NUMBER OF WORDS TO BE FOUND ON DISK
                ;;X=0 MUST BE ZERO
                ;THE COMMAND THEN IS JSR PC,COMMD
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;
                ;IN A READ DATA COMMAND FILL THE FOLLOWING
                ;CYL= WITH CYLINDER TO BE FOUND ON DISK
                ;SECOTR= WITH SECTOR AND TRACK TO BE FOUND ON DISK
                ;KEY1= WITH KEY1 TO BE FOUND ON DISK
                ;KEY2=WITH KEY2 TO BE FOUND ON DISK
                ;DAWORD= WITH NUMBER OF WORDS TO BE FOUND ON DISK
                ;;X=0 MUST BE ZERO
                ;THE COMMAND THEN IS JSR PC,COMMD
```

```
  9996                                  ;
```

```
 9998
 9999
10000
10001
10002
10003
10004
10005                                   ;WRITE DATA COMMAND
10006                                   ;OR READ COMMAND I.E DATA ONLY OR HEADER AND DATA
10007  030706  011637  002002          COMND:  MOV     (SP),@#PCJSR    ;SAVE PC OF JSR + 4
10008  030712  162737  000004  002002          SUB     #4,@#PCJSR              ;SAVE PC OF JSR
10009  030720  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (2)  030722  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2)  030724  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (2)  030726  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (2)  030730  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (2)  030732  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
10010  030734  012777  000001  150710          MOV     #DMD,@RHMR      ;SET DIAGNOSTIS MODE
10011  030742  052777  000001  150662          BIS     #GO,@RHCS1      ;GO
10012  030750  016746  000044                  MOV     SECOTR, -(SP)   ;GET DESIRED SECTOR/TRACK
10013  030754  042716  177740                  BIC     #177740,        (SP)    ;MAKE ONLY SECTOR
10014  030760  012637  030770                  MOV     (SP)+,  @#TRK   ;SAVE SECTOR
10015  030764  004137  035032                  JSR     R1,     @#SEARCH        ;DO SEARCH SECTOR
10016  030770  000000                  TRK:    .WORD   0
10017  030772  012701  000240                  MOV     #+NOP,R1                ;GOING TO MOVE NOPS
10018  030776  010137  031030                  MOV     R1,@#SSYN       ;NOP INTO SSYN
10019  031002  010137  031032                  MOV     R1,@#HEDGAP     ;NOP INTO HEDGAP
10020  031006  010137  031034                  MOV     R1,@#HEDSYN     ;NOP INTO HEDSYN
10021  031012  004137  031140                  JSR     R1,     @#RDHEAD
10022  031016  000000                  CYL:    .WORD   0               ;CYLINDER ADDRESS
10023  031020  000000                  SECOTR: .WORD   0               ;SECTOR/TRACK ADDRESS
10024  031022  000000                  KEY1:   .WORD   0               ;KEY1 WORD
10025  031024  000000                  KEY2:   .WORD   0               ;KEY2 WORD
10026  031026  000000                  X:      .WORD   0               ;X=1 WRITE COMMAND
10027                                                                   ;X=0 READ COMMAND
10028  031030  000240                  SSYN:   NOP                     ;IF "ERROR 2" INSERTED BY RDHEAD
10029                                                                   ;SUBROUTINE THEN THE FIRST SYNC,
10030                                                                   ;IS NOT DETECTED,  NO BAD DATA
10031                                                                   ;IS GIVEN BECAUSE SYNC=144000
10032                                                                   ;CANNOT BE READ,  WORD NO
10033                                                                   ;IS "1" BECAUSE THIS IS THE FIRST
10034                                                                   ;WORD TESTED
10035
10036
10037
10038  031032  000240                  HEDGAP: NOP                     ;IF "ERROR 3" INSERTED BY
10039                                                                           ;RDHEAD SUBROUTINE THEN THE
10040                                                                   ;HEADER GAP 0'S WERE NOT
10041                                                                   ;WRITTEN RIGHT,
10042                                                                   ;IF "WORD NO" CONTAINS SAY
10043                                                                   ;3(8) THEN IT IS THE THIRD
10044                                                                   ;WORD OF A 5 WORD HEADER
10045                                                                   ;GAP THAT IS WRONG
10046                                                                   ;"BAD DATA" CONTAINS WHAT IS
```

```
                                                        ;GOING ON THE DISK
                        HEDSYN: NOP
                                                                ;IF "ERROR 3" INSERTED BY RDHEAD
                                                        ;SUBROUTINE THEN THE HEADER SYNC,
                                                        ;GENERATED BY DCL IS WRONG
                                                        ;OR THE LAST BYTE
                                                        ;OF THE HEADER GAP 0'S IS WRONG
                                                        ;IN EITHER CASE WORD NO=6
                                                        ;RIGHT BYTE IS HEADER #
                                                        ;LEFT BYTE IS SYNC
                                                        ;"BAD DATA" HAS WHAT IS GOING
                                                        ;ON DISK
031036  005737  001774          TST     @#ERFLG#                ;ARE ANY ERRORS DETECTED
031042  001017                  BNE     OUT             ;IF YES BRANCH
031044  005737  031026          TST     @#X
031050  001410                  BEQ     DAREAD
031052  005737  031132          TST     @#NOSYNC        ;IS THIS FORCED HEADER ERROR COMMAND
                                                        ;IF YES  NOSYNC=-1 THEN WRITE OR READ
                                                        ;IS SHUT OFF SO BRANCH OUT
                                                        ;IF  NOSYNC=0 THEN CONTINUE
031056  001011                  BNE     OUT             ;BRANCH IF SET
031060  004137  032404          JSR     R1,     @#WRDATA        ;WRITE DATA
031064  000000          NOWORD: .WORD   0               ;NO OF WORDS TO BE WRITTEN
031066  000000          Y:      .WORD   0               ;
031070  000404                  BR      OUT             ;
031072  004137  035306  DAREAD: JSR     R1,     @#REDATA        ;READ DATA
031076  000000          DAWORD: .WORD   0               ;NO OF WORDS TO BE READ
031100  000000                  .WORD   0
031102                  OUT:
  (2)   031102  012605          MOV     (SP)+,R5        ;;POP STACK INTO R5
  (2)   031104  012604          MOV     (SP)+,R4        ;;POP STACK INTO R4
  (2)   031106  012603          MOV     (SP)+,R3        ;;POP STACK INTO R3
  (2)   031110  012602          MOV     (SP)+,R2        ;;POP STACK INTO R2
  (2)   031112  012601          MOV     (SP)+,R1        ;;POP STACK INTO R1
  (2)   031114  012600          MOV     (SP)+,R0        ;;POP STACK INTO R0
031116  000207                  RTS     PC
                        ;*****************************************************************
                        ;THE DISK SECTOR IS DEVIDED AS FOLLOWS
                        ;19 WORDS OF 0, ONE WORD 144000
                        ;THESE MAKE 39 BYTES FOR SECTOR GAP AND ONE SYNC, BYTE

031120  014400          RSYNC:  14400
031122  000000          RCYL:   0
031124  000000          RSETR:  0
031126  000000          RKEY1:  0
031130  000000          RKEY2:  0

                        ;5 WORDS OF 0 ONE WORD 144000
                        ;THESE MAKE 11 BYTES FOR HEADER GAP AND ONE SYNC, BYTE
                        ;THESE ARE DCL GENERATED
```

```
10095
10096
10097
10098                                   ;THERE ARE 256 WORDS OF DATA
10099                                   ;THERE ARE 2 WORDS FOR ECC GENERATED BY DCL
10100                                   ;15 WORDS OF 0 FOR DATA GAP AND TOLERANCE GAP
10101                                   ;*****************************************************************
10102
10103
10104
10105
10106
10107
10108                                   ;*****************************************************************
10109                                   ;READ DISK HEADER
10110                                   ;*****************************************************************
10111
10112
10113
10114
10115
10116  031132  000000                   NOSYNC: 0                       ;FORCED HEADER ERROR = -1
10117                                                                   ;NORMAL = 0
10118  031134  000000                   TY:     0                       ;ERROR TYPE NO.
10119  031136  000000                   ERWORD: 0                       ;ERROR WORD NO.
10120
10121
10122
10123
10124  031140  012137  031122           RDHEAD: MOV     (R1)+,  @#RCYL  ;STORE CYLINDER ADDRESS
10125  031144  012137  031124                   MOV     (R1)+,  @#RSETR ;STORE SECTOR AND TRACK ADDRESS
10126  031150  012137  031126                   MOV     (R1)+,  @#RKEY1 ;STORE KEY1
10127  031154  012137  031130                   MOV     (R1)+,  @#RKEY2 ;STORE KEY2
10128  031160  012137  031730                   MOV     (R1)+,  @#COMPA ;STORE COMPARE OR NOT
10129  031164  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10130  031166  013700  001652                   MOV     @#RHMR, R0      ;R0 CONTAINS MAINTANENCE REG.
10131  031172  012705  000002                   MOV     #2,     R5      ;R5 IS A COUNTER FOR WORDS
10132  031176  012710  000001                   MOV     #DMD,   @R0     ;DIAG. MODE
10133  031202  052710  000010                   BIS     #MSTCK,@R0      ;SET SECTOR FOR FIRST WORD
10134  031206  052710  000002                   BIS     #MCLK,@R0       ;SET CLOCK FOR FIRST WORD
10135  031212  042710  000012                   BIC     #MSTCK!MCLK,@R0 ;RESET SECTOR AND CLOCK
10136  031216  000404                           BR      2$              ;BRANCH OVER GIVING SECTOR FOR FIRST TIME
10137  031220  012710  000013           1$:     MOV     #MSTCK!MCLK!DMD,@R0;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
10138  031224  042710  000012                   BIC     #MSTCK!MCLK,@R0 ;RESET SECTOR, CLOCK
10139  031230  012702  000007           2$:     MOV     #7,     R2      ;R2 IS A COUNTER FOR BYTES
10140  031234  052710  000002           3$:     BIS     #MCLK,  @R0     ;SET CLOCK
10141  031240  042710  000002                   BIC     #MCLK,  @R0     ;RESET CLOCK
10142  031244  005302                           DEC     R2              ;BYTE COUNTER
10143  031246  001372                           BNE     3$              ;BRANCH IF BYTE NOT COMPLETE
10144  031250  005305                           DEC     R5              ;WORD COUNTER
10145  031252  001362                           BNE     1$              ;BRANCH IF WORD NOT COMPLETE
10146  031254  012702  000022                   MOV     #18.,   R2      ;NO OF WORDS OF ZEROS
10147  031260  005037  031726           4$:     CLR     @#WORD          ;READ 0
10148  031264  004737  031732                   JSR     PC,     @#READ  ;GO TO READ
```

D12

```
10149  031270  005302                          DEC     R2              ;COUNT
10150  031272  001372                          BNE     48
10151  031274  013737  031120  031726          MOV     @#RSYNC,@#WORD  ;SYNC, WORD
10152  031302  004737  031732                  JSR     PC,     @#READ
10153  031306  032710  001000                  BIT     #DTSY,  @R0     ;SYNC, BYTE DETECTED?
10154  031312  001012                          BNE     58              ;BRANCH IF SYNC DETECTED
10155  031314  012737  000001  031136          MOV     #1,     @#ERWORD        ;ERROR WORD NO
10156  031322  013737  031120  001124          MOV     @#RSYNC,@#GDDAT         ;SYNC WORD
10157  031330  012737  104002  031030          MOV     #104002,@#SYN   ;INSERT "ERROR 2" IN SYN
10158  031336  000571                          BR      138             ;BRANCH OUT
10159  031340  013737  031122  031726  58:     MOV     @#RCYL, @#WORD  ;SETUP CYLINDER
10160  031346  004737  031732                  JSR     PC,     @#READ  ;READ
10161  031352  013737  031124  031726          MOV     @#RSETR,@#WORD  ;SETUP SECTOR/TRACK
10162  031360  004737  031732                  JSR     PC,@#READ       ;READ
10163  031364  013737  031126  031726          MOV     @#RKEY1,@#WORD  ;SETUP KEY1
10164  031372  004737  031732                  JSR     PC,@#READ       ;READ
10165  031376  013737  031130  031726          MOV     @#RKEY2,@#WORD  ;SETUP KEY2
10166  031404  004737  031732                  JSR     PC,@#READ       ;READ
10167  031410  013737  032716  031726          MOV     @#WCRC,@#WORD   ;SETUP CRC
10168  031416  004737  031732                  JSR     PC,@#READ       ;READ
10169  031422  005737  002014                  TST     @#TESDTE        ;IS THIS A DRIVE TIMING ERROR
10170  031426  001135                          BNE     138             ;BRANCH OUT IF YES
10171  031430  005737  031730                  TST     @#COMPA         ;IS THIS A READ OR WRITE COMMAND
10172  031434  001472                          BEQ     118
10173  031436  012705  032720                  MOV     #HEGAP, R5      ;POINTER FOR HEADER GAP
10174  031442  012702  000005                  MOV     #5,     R2      ;NO OF WORDS OF ZEROS
10175  031446  012737  000006  031136  68:     MOV     #6,@#ERWORD     ;ERROR WORD NO SET
10176  031454  004737  032164                  JSR     PC,@#WRITE      ;FOR HEADER GAP
10177  031460  005737  032162                  TST     @#WWORD         ;TEST WRITTEN WORD
10178  031464  001413                          BEQ     78              ;BRANCH IF GOOD THAT IS 0
10179  031466  160237  031136                  SUB     R2,@#ERWORD     ;WORD NO IN ERROR
10180  031472  005037  001124                  CLR     @#@#GDDAT               ;GOOD WORD SHOULD BE 0
10181  031476  013767  032162  147422          MOV     @#WWORD,        @#BDDAT ;BAD DATA
10182  031504  012737  104003  031032          MOV     #104003,@#HEDGAP        ;"ERROR 2" GOES IN HEDGAP
10183  031512  000503                          BR      138             ;BRANCH OUT
10184  031514  012725  032162          78:     MOV     @#WWORD,(R5)+   ;SAVE HEADER GAP
10185  031520  005302                          DEC     R2
10186  031522  001351                          BNE     68
10187  031524  004737  032164                  JSR     PC,     @#WRITE ;WRITE HEADER (DATA) GAP SYNC
10188  031530  023737  031120  032162          CMP     @#RSYNC,@#WWORD
10189  031536  001426                          BEQ     108
10190  031540  005737  031132                  TST     @#NOSYNC        ;IS THIS FORCED HEADER ERROR COMMAND
10191                                                                  ;IF YES  NOSYNC=-1 THEN WRITE OR READ
10192                                                                  ;IS SHUT OFF SO BRANCH OUT
10193                                                                  ;IF NO NOSYNC=0 THEN CONTINUE
10194  031544  001406                          BEQ     148             ;BRANCH IF TRUE ERROR
10195  031546  005737  032162                  TST     @#WWORD         ;
10196  031552  001420                          BEQ     108             ;BRANCH IF GOOD
10197  031554  005037  001124                  CLR     @#@#GDDAT       ;IT SHOULD BE ZERO
10198  031560  000403                          BR      158             ;BRANCH TO TYPE ERROR
10199  031562  013737  031120  001124  148:     MOV     @#RSYNC,@#@#GDDAT       ;GOOD DATA
10200  031570  013737  032162  001126  158:     MOV     @#WWORD,@#@#BDDAT       ;BAD DATA
10201  031576  012737  000006  031136          MOV     #6,     @#ERWORD
10202  031604  012737  104003  031034          MOV     #104003,@#HEDSYN
```

```
 10203  031612  000443                          BR      138             ;BRANCH OUT
 10204  031614  013725  032162          108:    MOV     @#^WORD,(R5)+   ;SAVE DATA SYNC,
 10205  031620  000440                          BR      138
 10206                                  ;READ COMMAND START FROM HERE
 10207  031622  012702  000005          118:    MOV     #5,     R2
 10208  031626  005067  000074          128:    CLR     WORD
 10209  031632  004767  000074                  JSR     PC,     READ    ;READ HEADER GAP
 10210  031636  005302                          DEC     R2              ;IS 5 HEADER GAP ZEROS COMPLETE
 10211  031640  001372                          BNE     128             ;IF NOT BRANCH
 10212  031642  013737  031120  031726          MOV     @#RSYNC,@#WORD  ;SYNC WORD
 10213  031650  004767  000056                  JSR     PC,     READ    ;READ HEADER (DATA) SYNC)
 10214  031654  005737  031132                  TST     @#NOSYNC
 10215  031660  001404                          BEQ     168             ;IF NOT ERROR COMMAND BRANCH
 10216  031662  032710  001000                  BIT     #DTSY,@R0       ;SYNC, DETECTED
 10217  031666  001415                          BEQ     138             ;IF ZERO BRANCH OUT
 10218  031670  000403                          BR      178             ;IF NOT ZERO BRANCH TO ERROR
 10219  031672  032710  001000          168:    BIT     #DTSY,  @R0     ;SYNC, DETECTED?
 10220  031676  001011                          BNE     138             ;BRANCH IF YES
 10221  031700  012737  000006  031136  178:    MOV     #6,@#ERWORD     ;ERROR WORD NO,
 10222  031706  013737  031120  001124          MOV     @#RSYNC,@#GDDAT;SYNC WORD
 10223  031714  012737  104002  031034          MOV     #104002,@#MEDSYN
 10224  031722                          138:
   (2)  031722  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 10225  031724  000201                          RTS     R1
 10226
 10227
 10228
 10229
 10230
 10231
 10232
 10233
 10234                                  ;*****************************************************************
 10235                                  ;READ ONE WORD IN "WORD"
 10236                                  ;*****************************************************************
 10237
 10238
 10239
 10240
 10241  031726  000000                  WORD:   0
 10242  031730  000000                  COMPA:  0
 10243
 10244
 10245
 10246
 10247  031732                          READ:
   (2)  031732  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 10248  031734  012705  000002                  MOV     #2,     R5      ;WORD COUNTER
 10249  031740  012710  000001                  MOV     #DMD,   @R0     ;SET DIAG. MODE
 10250  031744  006037  031726                  ROR     @#WORD          ;CHECKING IF THERE IS A ONE
 10251  031750  103002                          BCC     18              ;IF NO ONE BRANCH
 10252  031752  052710  000020                  BIS     #MRD,   @R0     ;SET BIT 4 IF DATA HAS ONE
 10253  031756  012702  000007          18:     MOV     #7,     R2      ;BYTE COUNTER
 10254  031762  052710  000012                  BIS     #MSTCK!MCLK,    @R0     ;SET CLOCK,DATA IF ANY, SECTOR
```

```
10255  031766  005737  027132          TST     @#TSECCG        ;IS THIS BIT TO GENERATE AND TEST ECC
10256  031772  001411                  BEQ     6$              ;BRANCH IF NO
10257  031774  032710  000020          BIT     #MRD,@R0        ;IS DATA BIT A ONE
10258  032000  001404                  BEQ     5$              ;BRANCH IF DATA BIT IS 0
10259  032002  012737  177777  027124  MOV     #-1,@#ECDATA    ;ECC DATA BIT IS A ONE
10260  032010  000402                  BR      6$              ;BRANCH
10261  032012  005037  027124  5$:     CLR     @#ECDATA        ;ECC DATA BIT IS A 0
10262  032016  012746  000001  6$:     MOV     #DMD,   -(SP)   ;KEEP ONLY DIAG. MODE
10263  032022  006037  031726          ROR     @#WORD          ;CHECKING IF THERE IS A ONE
10264  032026  103002                  BCC     2$              ;IF NO ONE BRANCH
10265  032030  012716  000021          MOV     #MRD!DMD,       (SP)    ;KEEP DATA AND DIAG. MODE
10266  032034  012610          2$:     MOV     (SP)+,  @R0     ;PUT IN DATA,RESET CLOCK, SECTOR
10267  032036  005737  027132          TST     @#TSECCG        ;IS ECC TO BE GENERATED FOR THIS BIT
10268  032042  001404                  BEQ     3$              ;BRANCH IF NO
10269  032044  005237  027144          INC     @#DATENV        ;NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
10270  032050  004737  027162          JSR     PC,@#ECTEST     ;GO TO GENERATE AND TEST ECC
10271  032054  052710  000002  3$:     BIS     #MCLK,  @R0     ;SET CLOCK
10272  032060  005737  027132          TST     @#TSECCG        ;IS THIS BIT TO GENERATE ECC
10273  032064  001411                  BEQ     8$              ;BRANCH IF NO
10274  032066  032710  000020          BIT     #MRD,@R0        ;IS DATA BIT A ONE
10275  032072  001404                  BEQ     7$              ;BRANCH IF DATA BIT IS = 0
10276  032074  012737  177777  027124  MOV     #-1,@#ECDATA    ;ECC DATA BIT IS A ONE
10277  032102  000402                  BR      8$              ;BRANCH
10278  032104  005037  027124  7$:     CLR     @#ECDATA        ;ECC DATA BIT IS = 0
10279  032110  012746  000001  8$:     MOV     #DMD,   -(SP)   ;KEEP DIAG. MODE
10280  032114  006037  031726          ROR     @#WORD          ;CHECKING IF THERE IS A ONE
10281  032120  103002                  BCC     4$              ;BRANCH IF NO ONE
10282  032122  012716  000021          MOV     #MRD!DMD,(SP)   ;KEEP DIAG. MODE AND DATA
10283  032126  012610          4$:     MOV     (SP)+,  @R0     ;SET DATA, DIAG. MODE, CLEAR CLOCK
10284  032130  005737  027132          TST     @#TSECCG        ;IS THIS BIT TO GENERATE ECC
10285  032134  001404                  BEQ     9$              ;BRANCH IF NO
10286  032136  005237  027144          INC     @#DATENV        ;NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
10287  032142  004737  027162          JSR     PC,@#ECTEST     ;GO TO GENERATE AND TEST ECC
10288  032146  005302                  DEC     R2              ;BYTE COUNTER
10289  032150  001341                  BNE     3$              ;BRANCH IF ONE BYTE NOT COMPLETE
10290  032152  005305                  DEC     R5              ;WORD COUNTER
10291  032154  001300                  BNE     1$              ;BRANCH IF ONE WORD NOT COMPLETE
10292  032156  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
10293  032160  000207                  RTS     PC
10294
10295
10296
10297
10298
10299
10300                          ;*****************************************************************
10301                          ;WRITE ONE WORD WHICH COMES BACK IN "WWORD"
10302                          ;*****************************************************************
10303
10304
10305
10306
10307
10308
```

```
 10309  032162  000000                  WWORD:  0
 10310
 10311
 10312
 10313
 10314  032164                          WRITE:
   (2)  032164  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
   (2)  032166  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
   (2)  032170  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
   (2)  032172  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 10315  032174  012705  000002                  MOV     #2,     R5      ;WORD COUNTER
 10316  032200  012710  000001                  MOV     #1,     @R0     ;SET DIAG. MODE
 10317  032204  012702  000007          18:     MOV     #7,     R2      ;BYTE COUNTER
 10318  032210  012710  000013                  MOV     #MSTCK!MCLK!DMD,@R0;SET SECTOR AND CLOCK
 10319  032214  032710  000040                  BIT     #MWR,   @R0     ;CHECK WRITEBIT IN MAINT. REG.
 10320  032220  001406                          BEQ     2$              ;BRANCH IF ZERO
 10321  032222  012737  177777  027124          MOV     #-1,@#ECDATA    ;ECC DATA BIT IS A ONE
 10322  032230  000261                          SEC                     ;SET CARRY
 10323  032232  006003                          ROR     R3              ;MOVE 1 FORWARD
 10324  032234  000404                          BR      3$
 10325  032236  005037  027124          2$:     CLR     @#ECDATA        ;ECC DATA BIT IS = 0
 10326  032242  000241                          CLC                     ;CLEAR CARRY
 10327  032244  006003                          ROR     R3              ;MOVE 0 FOR WWORD
 10328  032246  012710  000001          3$:     MOV     #DMD,   @R0     ;CLEAR SECTOR AND CLOCK
 10329  032252  005737  027132                  TST     @#TSECCG        ;IS THIS BIT TO GENERATE ECC
 10330  032256  001404                          BEQ     4$              ;BRANCH IF NO
 10331  032260  005237  027144                  INC     @#DATENV        ;NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
 10332  032264  004737  027162                  JSR     PC,@#ECTEST     ;GO TO GENERATE AND TEST ECC
 10333  032270  052710  000002          4$:     BIS     #MCLK,  @R0     ;SET CLOCK
 10334  032274  032710  000040                  BIT     #MWR,   @R0     ;CHECK WRITE BIT IN MAINT. REG.
 10335  032300  001406                          BEQ     5$              ;BRANCH IF ZERO
 10336  032302  012737  177777  027124          MOV     #-1,@#ECDATA    ;ECC DATA BIT IS A ONE
 10337  032310  000261                          SEC                     ;SET CARRY
 10338  032312  006003                          ROR     R3              ;MOVE 1 FOR WWORD
 10339  032314  000404                          BR      6$
 10340  032316  005037  027124          5$:     CLR     @#ECDATA        ;ECC DATA BIT IS ZERO
 10341  032322  000241                          CLC                     ;CLEAR CARRY
 10342  032324  006003                          ROR     R3              ;MOVE 0 FOR WWORD
 10343  032326  012710  000001          6$:     MOV     #DMD,   @R0     ;CLEAR CLOCK
 10344  032332  005737  027132                  TST     @#TSECCG        ;IS THIS BIT TO GENERATE ECC
 10345  032336  001404                          BEQ     7$              ;BRANCH IF NO
 10346  032340  005237  027144                  INC     @#DATENV        ;NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
 10347  032344  004737  027162                  JSR     PC,@#ECTEST     ;GO TO GENERATE AND TEST ECC
 10348  032350  005302                  7$:     DEC     R2              ;COUNT FOR BYTE END
 10349  032352  001346                          BNE     4$              ;IF NOT BYTE END BRANCH
 10350  032354  005305                          DEC     R5              ;COUNT FOR WORD END
 10351  032356  001312                          BNE     1$              ;IF NOT WORD END BRANCH
 10352  032360  010337  032162                  MOV     R3,     @#WWORD ;STORE WORD
 10353  032364  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
   (2)  032366  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
   (2)  032370  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
   (2)  032372  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 10354  032374  000207                          RTS     PC
 10355
```

```
10356
10357
10358
10359
10360
10361                                   ;*****************************************************************
10362                                   ;WRITE DATA
10363                                   ;*****************************************************************
10364
10365
10366
10367
10368
10369
10370
10371  032376  000000                   COUNTD: 0
10372  032400  000400                   FORMAT: 256.
10373  032402  000000                   ZWORDS: 0
10374  032404                           WRDATA:
10375  032404  011137  032376                   MOV     (R1),   @#COUNTD        ;STORE NO. OF WORDS TO BE WRITTEN
10376  032410  012102                           MOV     (R1)+,  R2              ;SAME IN R2
10377  032412  012137  031730                   MOV     (R1)+,  @#COMPA ;COMPARE OR NOT
10378  032416  010046                           MOV     R0,-(SP)                ;;PUSH R0 ON STACK
  (2)  032420  010146                           MOV     R1,-(SP)                ;;PUSH R1 ON STACK
  (2)  032422  010246                           MOV     R2,-(SP)                ;;PUSH R2 ON STACK
  (2)  032424  010346                           MOV     R3,-(SP)                ;;PUSH R3 ON STACK
  (2)  032426  010446                           MOV     R4,-(SP)                ;;PUSH R4 ON STACK
10379  032430  012701  000016                   MOV     #14.,   R1              ;NO. OF TOLERANCE GAP WORDS
10380  032434  012703  033742                   MOV     #TOLGAP,R3              ;START OF TOLERANCE GAP TABLE
10381  032440  012723  177777           18:     MOV     #-1,    (R3)+   ;MAKE IT 177777
10382  032444  005301                           DEC     R1                      ;IS 14 COMPLETED
10383  032446  001374                           BNE     1B                      ;IF NO BRANCH
10384  032450  013700  001652                   MOV     @#RHMR, R0              ;R0 CONTAINS MAINTANENCE REG,
10385  032454  013746  032400                   MOV     @#FORMAT,       -(SP)
10386  032460  163716  032376                   SUB     @#COUNTD,       (SP)
10387  032464  011637  032402                   MOV     (SP),@#ZWORDS   ;NO. OF ZERO WORDS TO BE WRITTEN
10388  032470  012604                           MOV     (SP)+,  R4
10389  032472  005737  002012                   TST     @#TSECC                 ;IS THIS AN ECC TEST
10390  032476  001403                           BEQ     7$                      ;BRANCH IF NO
10391  032500  012737  177777  027132           MOV     #-1,@#TSECCG    ;THESE BITS ARE TO GENERATE ECC
10392  032506  012703  032734           7$:     MOV     #DISK,  R3              ;SIMULATED DISK AREA
10393  032512  004737  032164           2$:     JSR     PC,     @#WRITE ;WRITE ON SIMULATED DISK
10394  032516  013723  032162                   MOV     @#WWORD,(R3)+   ;STORE ON SIMULATED DISK
10395  032522  005302                           DEC     R2
10396  032524  001372                           BNE     2$
10397  032526  005704                           TST     R4      ;ANY ZEROS TO BE WRITTEN
10398  032530  001406                           BEQ     4$      ;BRANCH IF NONE TO BE WRITTEN
10399  032532  004737  032164           3$:     JSR     PC,     @#WRITE ;WRITE ZEROS ON SIMULATED DISK
10400  032536  013723  032162                   MOV     @#WWORD,        (R3)+   ;STORE
10401  032542  005304                           DEC     R4
10402  032544  001372                           BNE     3$
10403  032546  005037  027132           4$:     CLR     @#TSECCG        ;NO MORE ECC TO BE GENERATED
10404  032552  012701  000002                   MOV     #2,     R1
10405  032556  004767  177402           5$:     JSR     PC,     WRITE   ;WRITE ECC1 AND ECC2 ON SIMULATED DISK
```

```
10406  032562  013723  032162                  MOV     @#WWORD,(R3)+   ;STORE ON WEEC1 AND WEEC2
10407  032566  005301                          DEC     R1
10408  032570  001372                          BNE     58
10409  032572  004767  177366                  JSR     PC,     WRITE   ;WRITE DATA GAP
10410  032576  013723  032162                  MOV     @#WWORD,(R3)+   ;STORE
10411  032602  012701  000016                  MOV     #14.,   R1
10412  032606  004737  032164          68:     JSR     PC,     @#WRITE ;WRITE TOLERANCE GAP ZEROS
10413  032612  013723  032162                  MOV     @#WWORD,(R3)+   ;STORE
10414  032616  005301                          DEC     R1
10415  032620  001372                          BNE     68
10416  032622  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
  (2)  032624  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
  (2)  032626  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
  (2)  032630  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
  (2)  032632  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10417  032634  000201                          RTS     R1
10418
10419
10420
10421
10422
10423
10424
10425
10426                          ;*****************************************************************
10427                          ;WRITE HEADER AND DATA
10428                          ;
10429                          ;
10430                          ;THIS IS THE SIMULATED DISK
10431                          ;ONLY ONE SECTOR OF SPACE IS ALLOWED
10432                          ;*****************************************************************
10433
10434
10435
10436
10437
10438
10439
10440  032636  000023  SECGAP:  .BLKW   19.              ;SECTOR GAP 38 BYTES OF 0
10441  032704  000001  WSSYNC:  .BLKW   1               ;SECTOR GAP 1 BYTE OF 0 ONE SYNC BYTE
10442  032706  000004  HEADER:  .BLKW   4               ;HEADER = CYL, SECTOR/TRACK, KEY1, KEY2
10443  032716  000001  WCRC:    .BLKW   1               ;CRC
10444  032720  000005  HEGAP:   .BLKW   5               ;HEADER GAP 10 BYTES OF 0
10445  032732  000001  HDWSYN:  .BLKW   1               ;HEADER GAP 1 BYTE OF 0 ONE SYNC, BYTE
10446  032734          SILOTB:                 ;USED IN SILO TEST AS SILO TABLE
10447  032734  000400  DISK:    .BLKW   256.            ;DATA SPACE
10448  033734  000001  WECC1:   .BLKW   1               ;ECC1
10449  033736  000001  WECC2:   .BLKW   1               ;ECC2
10450  033740  000001  DTAGAP:  .BLKW   1               ;DATA GAP 2 BYTES OF 0
10451  033742  000016  TOLGAP:  .BLKW   14.              ;TOLERANCE GAP 28 BYTES OF 0
10452
10453
10454
10455
```

```
10456
10457
10458                                   ;*****************************************************************
10459                                   ;WRITE HEADER AND DATA
10460                                   ;*****************************************************************
10461
10462
10463
10464
10465  034776  011637  002002          COMWHD: MOV     (SP),@#PCJSR    ;SAVE PC OF JSR + 4
10466  034002  162737  000004  002002          SUB     #4,@#PCJSR              ;SAVE PC OF JSR
10467  034010  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (2)  034012  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2)  034014  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (2)  034016  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (2)  034020  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (2)  034022  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
10468  034024  012777  000001  145620          MOV     #DMD,@RHMR              ;SET DIAGNOSTIC MODE
10469  034032  052777  000001  145572          BIS     #GO,@RHCS1              ;GO
10470  034040  013746  034124                  MOV     @#WSECTR,       -(SP)   ;GET DESIRED SECTOR/TRACK
10471  034044  042716  177740                  BIC     #177740,        (SP)    ;MAKE ONLY SECTOR
10472  034050  012637  034060                  MOV     (SP)+,  @#WTRK  ;SAVE SECTOR
10473  034054  004137  035032                  JSR     R1,@#SEARCH     ;DO SEARCH SECTOR
10474  034060  000000          WTRK:   .WORD   0                       ;SECTOR NO.
10475  034062  012701  000240                  MOV     #+NOP,R1                ;GOING TO MOVE NOPS
10476  034066  010137  034134                  MOV     R1,@#SEGPER     ;NOP INTO SEGAP
10477  034072  010137  034136                  MOV     R1,@#FSYNER     ;NOP INTP FSYNER
10478  034076  010137  034140                  MOV     R1,@#ERHEAD     ;NOP INTO ERHEAD
10479  034102  010137  034142                  MOV     R1,@#ERCRC      ;NOPINTO ERCRC
10480  034106  010137  034144                  MOV     R1,@#ERHDGP     ;NOP INTO ERHDGAP
10481  034112  010137  034146                  MOV     R1,@#HDESYN     ;NOP INTO HDESYN
10482  034116  004137  034216                  JSR     R1,@#WRHEAD
10483  034122  000000          WCYL:   0                               ;CYLINDER
10484  034124  000000          WSECTR: 0                               ;SECTOR AND TRACK
10485  034126  000000          WKEY1:  0                               ;KEY1
10486  034130  000000          WKEY2:  0                               ;KEY2
10487  034132  000000          GCRC:   0                               ;GOOD CRC
10488
10489  034134  000240          SEGPER: NOP                             ;IF "ERROR 6" INSERTED BY
10490                                                                  ;WRHEAD SUBROUTINE THEN
10491                                                                  ;SECTOR GAP GOING ON DISK
10492                                                                  ;IS NOT RIGHT
10493                                                                  ;WORD NO, CONTAINS WHICH
10494                                                                  ;WORD IS WRONG THAT IS
10495                                                                  ;FIRST OF TENTH OR WHAT EVER NO,
10496                                                                  ;BAD WORD IS WHAT IS GOING ON DISK
10497  034136  000240          FSYNER: NOP                             ;IF "ERROR 6" INSERTED BY
10498                                                                  ;WRHEAD SUBROUTINE THEN
10499                                                                  ;THE LAST 0 BYTE OF SECTOR
10500                                                                  ;GAP OF FIRST SYNC, BYTE
10501                                                                  ;AFTER SECTOR GAP IS IN
10502                                                                  ;ERROR
10503                                                                  ;WORD NO, CONTAINS 20
10504                                                                  ;RIGHT BYTE IS SECTOR GAP
```

```
10505                                                           ;LEFT BYTE IS SYNC, BYTE
10506                                                           ;BAD WORD IS WHAT IS GOING ON
10507                                                           ;DISK
10508
10509  034140  000240                   ERHEAD: NOP
10510                                                           ;IF "ERROR 6" INSERTED BY
10511                                                           ;WRHEAD SUBROUTINE THEN
10512                                                           ;HEADER GOING ON DISK
10513                                                           ;IS WRONG
10514                                                           ;WORD NO 1 = CYLINDER NO
10515                                                           ;WORD NO 2 = SECTOR/TRACK
10516                                                           ;WORD NO 3 = KEY1
10517                                                           ;WORD NO 4 = KEY2
10518                                                           ;BAD WORD IS WHAT IS GOING ON
10519                                                           ;DISK
10520
10521  034142  000240                   ERCRC:  NOP
10522                                                           ;IF "ERROR 6" INSERTED BY
10523                                                           ;WRHEAD SUBROUTINE THEN CRC WRITTEN
10524                                                           ;ON DISK IS IN ERROR
10525                                                           ;GOOD DATA IS WHAT SHOULD BE ON DISK
10526                                                           ;BAD DATA IS WHAT IS GOING ON DISK
10527
10528  034144  000240                   ERHDGP: NOP
10529                                                           ;IF "ERROR 6" INSERTED BY
10530                                                           ;WRHEAD SUBROUTINE THEN HEADER
10531                                                           ;GAP GOING ON DISK IS WRONG
10532                                                           ;WORD NO GIVES WHICH OF
10533                                                           ;THE HEADER GAP WORDS
10534                                                           ;ARE WRONG, FOR EXAMPLE
10535                                                           ;WORD NO 1 = FIRST HEADER
10536                                                           ;            GAP WORD
10537                                                           ;BAD WORD IS WHAT OS GOING ON DISK
10538
10539  034146  000240                   HDESYN: NOP
10540                                                           ;IF "ERROR 6" INSERTED BY
10541                                                           ;WRHEAD SUBROUTINE THEN LAST
10542                                                           ;HEADER GAP BYTE OR HEADER
10543                                                           ;SYNC BYTE GOING ON DISK IS WRONG
10544                                                           ;WORD NO = 9
10545                                                           ;BAD DATA IS WHAT IS GOING
10546                                                           ;ON DISK RIGHT BYTE IS HEADER
10547                                                           ;GAP 8 BYTE, LEFT BYTE IS HEADER
10548                                                           ;GAP SYNC.
10549
10550
10551  034150  005737  001774                  TST     @#ERFLG#               ;ARE ANY ERRORS DETECTED
10552  034154  001004                           BNE     FOUT            ;IF YES BRANCH
10553  034156  004137  032404                  JSR     R1,@#WRDATA
10554  034162  000000                   FNWORD: ,WORD   0               ;FORMAT COMMAND NO, OF DATA
10555  034164  000000                           ,WORD   0
10556  034166                           FOUT:
  (2)  034166  012605                           MOV     (SP)+,R5        ;;POP STACK INTO R5
  (2)  034170  012604                           MOV     (SP)+,R4        ;;POP STACK INTO R4
```

```
   (2)  034172  012603                    MOV     (SP)+,R3        ;;POP STACK INTO R3
   (2)  034174  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
   (2)  034176  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
   (2)  034200  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
10557   034202  000207                    RTS     PC
10558
10559
10560
10561
10562
10563
10564                           ;*****************************************************************
10565                           ;WRITE HEADER
10566                           ;*****************************************************************
10567
10568
10569
10570
10571
10572                           ;R0 = MAINT.REG.; R1 = SIMULATED DISK; R2 = BYTE COUNT; R3 = WRITE WORD; R5 = WORD COUNT
10573
10574
10575
10576   034204  000000          SCYL:   0                       ;
10577   034206  000000          SSECTR: 0                       ;
10578   034210  000000          SKEY1:  0                       ;
10579   034212  000000          SKEY2:  0                       ;
10580   034214  000000          SCRC:   0                       ;
10581
10582
10583   034216  012137  034204  WRHEAD: MOV     (R1)+,  @#SCYL
10584   034222  012137  034206          MOV     (R1)+,  @#SSECTR
10585   034226  012137  034210          MOV     (R1)+,  @#SKEY1
10586   034232  012137  034212          MOV     (R1)+,  @#SKEY2
10587   034236  012137  034214          MOV     (R1)+,  @#SCRC
10588   034242  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10589   034244  012701  032636          MOV     @#SECGAP,R1     ;SIMULATED DISK INDICATOR
10590   034250  013700  001652          MOV     @#RHMR, R0      ;R0 NOW HAS MAINT. REG. ADDR.
10591   034254  012710  000001          MOV     @#DMD,  @R0     ;SET DIAG. MODE
10592   034260  012705  000002          MOV     #2,     R5      ;WORD COUNTER
10593   034264  052710  000010          BIS     @#MSTCK,@R0     ;SET SECTOR FOR FIRST BYTE
10594   034270  012710  000013  18:     MOV     @#MSTCK!MCLK!DMD,@R0;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
10595   034274  032710  000040          BIT     @#MWR,  @R0     ;CHECK WRITE BIT IN MAINT. REG.
10596   034300  001403                  BEQ     28
10597   034302  000261                  SEC                     ;SET CARRY
10598   034304  006003                  ROR     R3              ;MOVE ONE FORWARD
10599   034306  000402                  BR      38              ;
10600   034310  000241          28:     CLC                     ;CLEAR CARRY
10601   034312  006003                  ROR     R3              ;MOVE ZERO FORWARD
10602   034314  012710  000001  38:     MOV     @#DMD,  @R0     ;CLEAR CLOCK, SECTOR
10603   034320  012702  000007          MOV     #7,     R2      ;BYTE COUNTER
10604   034324  052710  000002  48:     BIS     @#MCLK, @R0     ;SET CLOCK
10605   034330  032710  000040          BIT     @#MWR,  @R0     ;CHECK WRITE BIT IN MAINT.REG.
10606   034334  001403                  BEQ     58              ;BRANCH IF ZERO
```

```
10607  034336  000261                          SEC                     ;SET CARRY
10608  034340  006003                          ROR     R3              ;MOVE ONE FORWARD
10609  034342  000402                          BR      6$
10610  034344  000241                  5$:     CLC
10611  034346  006003                          ROR     R3
10612  034350  012710  000001          6$:     MOV     #DMD,   @R0
10613  034354  005302                          DEC     R2
10614  034356  001362                          BNE     4$
10615  034360  005305                          DEC     R5
10616  034362  001342                          BNE     1$
10617  034364  010321                          MOV     R3,     (R1)+
10618  034366  005703                          TST     R3
10619  034370  001414                          BEQ     7$
10620  034372  012737  000001  031136          MOV     #1,@#ERWORD
10621  034400  005037  001124                  CLR     @#GDDAT
10622  034404  010337  001126                  MOV     R3,     @#BDDAT
10623  034410  012737  104006  034134          MOV     #104006,@#SEGPER
10624  034416  000137  035024                  JMP     @#17$           ;BRANCH OUT
10625  034422  012702  000022          7$:     MOV     #18.,   R2      ;COUNT NO. OF SECTOR GAP
10626  034426  012737  000024  031136  10$:    MOV     #20.,@#ERWORD   ;COUNT TO GIVE ERROR WORD
10627  034434  004737  032164                  JSR     PC,     @#WRITE ;WRITE SECTOR GAP
10628  034440  013721  032162                  MOV     @#WWORD,(R1)+   ;STORE SECTOR GAP WORD
10629  034444  001413                          BEQ     11$
10630  034446  160237  031136                  SUB     R2,@#ERWORD     ;IF NOT GET ERROR WORD NO.
10631  034452  005037  001124                  CLR     @#GDDAT                 ;GOOD WORD
10632  034456  013737  032162  001126          MOV     @#WWORD,@#BDDAT         ;BAD WORD
10633  034464  012737  104006  034134          MOV     #104006,@#SEGPER        ;STORE "ERROR 6" IN SEGPER
10634  034472  000554                          BR      17$             ;BRANCH OUT
10635  034474  005302                  11$:    DEC     R2              ;HAVE 18 WORDS OF ZEROS BEEN WRITTEN
10636  034476  001353                          BNE     10$             ;IF NOT BRANCH
10637                                          ;AT THIS POINT THE SECTOR FOUND FLOP SHOULD
10638                                          ;BE HIGH,   SO THAT THE HEADER SYNC BYTE CAN BE GIVEN
10639                                          ;HOWEVER IN THE DRIVE TIMING ERROR TEST THE REST OF THE ROUTINE
10640                                          ;IS ABORTED
10641  034500  005737  002014                  TST     @#TESDTE        ;IS THIS A DRIVE TIMING ERROR
10642  034504  001147                          BNE     17$     .       ;BRANCH OUT IF YES
10643  034506  004737  032164                  JSR     PC,     @#WRITE ;WRITE ONE SECTOR GAP 0 BYTE
10644                                                                  ;AND ONE SYNC, BYTE = 230
10645  034512  013711  032162                  MOV     @#WWORD,(R1)    ;SAVE 0 BYTE AND SYNC BYTE
10646  034516  023721  031120                  CMP     @#RSYNC,(R1)+   ;IF SYNC, BYTE RIGHT
10647  034522  001414                          BEQ     12$             ;IF YES BRANCH
10648  034524  012737  000024  031136          MOV     #20.,@#ERWORD   ;IF NOT GET READY FOR ERROR
10649  034532  013737  031120  001124          MOV     @#RSYNC,@#GDDAT         ;GOOD WORD
10650  034540  014137  001126                  MOV     -(R1),@#BDDAT   ;BAD WORD
10651  034544  012737  104006  034136          MOV     #104006,@#FSYNER        ;INSERT "ERROR 6" IN FSYNER
10652  034552  000524                          BR      17$             ;BRANCH OUT
10653  034554  012702  000004          12$:    MOV     #4,     R2      ;FOUR HEADER WORDS
10654  034560  012703  034204                  MOV     #SCYL,  R3      ;POINTER FOR HEADER TABLE
10655  034564  012737  000005  031136  13$:    MOV     #5,@#ERWORD     ;ERROR WORD NO SET
10656  034572  004737  032164                  JSR     PC,     @#WRITE ;WRITE 4 HEADER WORDS
10657  034576  013711  032162                  MOV     @#WWORD,(R1)    ;STORE WRITTEN WORD
10658  034602  022321                          CMP     (R3)+,  (R1)+   ;IS IT RIGHT?
10659  034604  001412                          BEQ     14$             ;IF GOOD BRANCH
10660                                                                  ;IF NOT GET READY FOR PRINT
```

```
 10661  034606  160237  031136                  SUB     R2,@#ERWORD     ;WORD NO
 10662  034612  014337  001124                  MOV     -(R3),@#GDDAT   ;GOOD DATA
 10663  034616  014137  001126                  MOV     -(R1),@#BDDAT   ;BAD DATA
 10664  034622  012737  104006  034140          MOV     #104006,@#ERHEAD        ;INSERT "ERROR 6"
 10665  034630  000475                          BR      178             ;BRANCH OUT
 10666  034632  005302                  1481:   DEC     R2              ;ARE 4 HEADER WORDS DONE?
 10667  034634  001353                          BNE     138             ;IF NOT BRANCH
 10668  034636  004737  032164                  JSR     PC,     @#WRITE ;WRITE CRC
 10669  034642  013711  032162                  MOV     @#WWORD,(R1)    ;STORE CRC
 10670  034646  022137  034132                  CMP     (R1)+,@#GCRC    ;COMPARE GOOD CRC
 10671  034652  001414                          BEQ     208             ;BRANCH IF GOOD
 10672  034654  014137  001126                  MOV     -(R1),@#BDDATA  ;BAD CRC WRITTEN
 10673  034660  013737  034132  001124          MOV     @#GCRC,@#GDDAT  ;GOOD CRC
 10674  034666  012737  000005  031136          MOV     #5,@#ERWORD     ;ERROR WORD NO
 10675  034674  012737  104006  034142          MOV     #104006,@#ERCRC ;INSERT ERROR 6
 10676  034702  000450                          BR      178
 10677  034704  012702  000005          2081:   MOV     #5,     R2      ;NO OF HEADER GAP
 10678  034710  012737  000006  031136  1581:   MOV     #6,@#ERWORD     ;ERROR WORD NO SET
 10679  034716  004737  032164                  JSR     PC,@#WRITE      ;WRITE HEADER GAP
 10680  034722  013721  032162                  MOV     @#WWORD,(R1)+   ;STORE
 10681  034726  001412                          BEQ     168             ;IF GOOD BRANCH
 10682  034730  160237  031136                  SUB     R2,@#ERWORD     ;ERROR WORD NO
 10683  034734  005037  001124                  CLR     @#GDDAT                 ;GOOD DATA
 10684  034740  014137  001126                  MOV     -(R1),@#BDDAT   ;BAD DATA
 10685  034744  012737  104006  034144          MOV     #104006,@#ERHDGP        ;STORE "ERROR 6"
 10686  034752  000424                          BR      178             ;BRANCH OUT
 10687  034754  005302                  1681:   DEC     R2              ;ARE 5 HEADER GAP ZEROS  DONE
 10688  034756  001354                          BNE     158             ;IF NOT BRANCH
 10689  034760  004737  032164                  JSR     PC,@#WRITE
 10690  034764  013711  032162                  MOV     @#WWORD,(R1)
 10691  034770  023721  031120                  CMP     @#RSYNC,(R1)+
 10692  034774  001413                          BEQ     178
 10693  034776  012737  000005  031136          MOV     #5,@#ERWORD
 10694  035004  014137  001126                  MOV     -(R1),  @#BDDAT
 10695  035010  013737  031120  001124          MOV     @#RSYNC,        @#GDDAT
 10696  035016  012737  104006  034146          MOV     #104006,@#HDESYN
 10697  035024                          1781:
   (2)  035024  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 10698  035026  000201                          RTS     R1
 10699
 10700
 10701
 10702
 10703
 10704
 10705
 10706
 10707                                  ;*****************************************************************
 10708                                  ;SEARCH SECTOR
 10709                                  ;*****************************************************************
 10710
 10711
 10712
 10713
```

```
10714
10715
10716                                   ;       R0=RHMR ADDRESS
10717                                   ;       R1=PASSED ARGUMENT (SECTOR SEARCHED FOR)
10718                                   ;       R2=CLOCK COUNT (PER BYTE)
10719                                   ;       R3=SECTOR COUNTER FROM R1
10720                                   ;       R5=BYTES PER WORD COUNT
10721                                   ;BEBORE INDEX IS GIVEN TWO SECTOR CLOCKS ARE GIVEN TO RESET
10722                                   ;SECTOR PULSE IN CASE IT IS SET
10723                                   ;AT BEGINNING OF EACH SECTOR ONE SECTOR CLOCK HAS TO RISE
10724                                   ;BEFORE CLOCK THEN EVERY EIGHT CLOCKS ONE SECTOR CLOCK IS
10725                                   ;IDENTICAL WITH CLOCK
10726                                   ;NUMBERING THE SECTOR CLOCKS AS FOLLOWS
10727                                   ;THE SECTOR CLOCK UNDER INDEX = 0
10728                                   ;THE NEXT = 1
10729                                   ;THE NEXT  = 2
10730                                   ;ETC.
10731                                   ;THEN THE LAST SECTOR CLOCK IN ONE SECTOR HAS NUMBER = 600
10732                                   ;THE NEXT SECTOR THEN HAS 600 SECTOR CLOCKS
10733                                   ;THE NEXT SECTOR THEN HAS ANOTHER 600 SECTOR CLOCKS
10734                                   ;AND SO ON
10735
10736
10737
10738  035030  000000                  SECTR:  0                       ;SECTOR SEARCHED FOR
10739
10740  035032  012137  035030          SEARCH: MOV     (R1)+,  @#SECTR ;SAVE SECTOR SEARCHED FOR
10741  035036  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (2)  035040  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2)  035042  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (2)  035044  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (2)  035046  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (2)  035050  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
10742  035052  013700  001652                  MOV     @#RHMR, R0      ;NOW R0 HAS MAINTENANCE REG. ADR.
10743  035056  013703  035030                  MOV     @#SECTR,R3      ;SECTOR COUNTER
10744  035062  012710  000001                  MOV     #DMD,   @R0     ;SET DIAGNOSTIC MODE
10745  035066  052710  000010                  BIS     #MSTCK, @R0     ;SET SECTOR CLOCK
10746  035072  042710  000010                  BIC     #MSTCK, @R0     ;CLEAR SECTOR CLOCK
10747  035076  052710  000010                  BIS     #MSTCK, @R0     ;SET SECTOR CLOCK
10748  035102  042710  000010                  BIC     #MSTCK, @R0     ;CLEAR SECTOR CLOCK
10749                                                                  ;THE ABOVE TWO SECTOR CLOCKS ARE GIVEN FOR
10750                                                                  ;RESETING SECTOR PULSE
10751                                                                  ;IN CASE IT STARTS SET
10752  035106  052710  000014                  BIS     #MINX!MSTCK,@R0 ;SET INDEX AND SECTOR CLOCK
10753  035112  012710  000001                  MOV     #DMD,   @R0     ;RESET INDEX AND SECTOR CLOCK
10754  035116  005703                          TST     R3              ;IF SECTOR REQUIRED JUMP OUT
10755  035120  001461                          BEQ     7$              ;BRANCH OF SECTOR ZERO REQUIRED
10756                                          ;NOW THE 384 WORDS WILL START
10757
10758
10759
10760                                   ;FOR FIRST BYTE SECTOR CLOCK WILL GO HIGH THEN CLOCK WILL GO HIGH
10761                                   ;BOTH WILL COME DOWN TOGETHER THEN SEVEN CLOCKS WILL BE GIVEN
10762                                   ;;FOR SECOND BYTE AND ALL OTHER BYTES TILL NEXT SECTOR SECTOR CLOCK
```

```
10763                                   ;WILL BE IDENTICAL WITH ONE CLOCK
10764
10765
10766                                           ;ONE WORD ONLY
10767
10768  035122  012702  000010          18:      MOV     #8.,    R2      ;BYTE COUNTER
10769  035126  012705  000002                   MOV     #2,     R5      ;BYTES PER WORD
10770  035132  052710  000010                   BIS     #MSTCK,@R0      ;SET SECTOR CLOCK
10771  035136  052710  000002                   BIS     #MCLK,@R0       ;SET CLOCK
10772  035142  000402                           BR      38              ;BRANCH TO CLEAR SECTOR AND CLOCK
10773  035144  052710  000012          28:      BIS     #MSTCK!MCLK,@R0 ;SET SECTOR AND CLOCK
10774  035150  042710  000012          38:      BIC     #MSTCK!MCLK,@R0 ;CLEAR SECTOR AND CLOCK
10775  035154  052710  000002          88:      BIS     #MCLK,  @R0     ;SET CLOCK
10776  035160  042710  000002                   BIC     #MCLK,  @R0     ;CLEAR CLOCK
10777  035164  005302                           DEC     R2              ;BYTE COUNTER
10778  035166  001372                           BNE     88              ;BRANCH IF BYTE NOT COMPLETE
10779  035170  012702  000007                   MOV     #7,     R2      ;SETUP FOR SECOND BYTE
10780  035174  005305                           DEC     R5              ;IS WORD COMPLETE?
10781  035176  001362                           BNE     28              ;BRANCH IF NOT COMPLETE
10782                                           ;TO GIVE SECTOR CLOCK AND CLOCK
10783
10784
10785                                           ;NOW 383 WORDS ARE LEFT AND ALL ARE IDENTICAL
10786
10787  035200  012701  000457                   MOV     #383.,  R1      ;WORDS PER SECTOR COUNTER
10788  035204  012705  000002          48:      MOV     #2,     R5      ;BYTES PER WORD COUNTER
10789  035210  012702  000007          58:      MOV     #7,     R2      ;BYTE COUNTER (CLOCK COUNTER)
10790  035214  052710  000012                   BIS     #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND CLOCK
  791  035220  042710  000012                   BIC     #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND CLOCK
  792  035224  052710  000002          68:      BIS     #MCLK,  @R0     ;SET CLOCK
10793  035230  042710  000002                   BIC     #MCLK,  @R0     ;RESET CLOCK
10794  035234  005302                           DEC     R2              ;IS BYTE COMPLETE?
10795  035236  001372                           BNE     68              ;BRANCH IF NOT COMPLETE
10796  035240  005305                           DEC     R5              ;IS WORD COMPLETE?
10797  035242  001362                           BNE     58              ;BRANCH IF NOT
10798  035244  005301                           DEC     R1              ;IS SECTOR COMPLETE
10799  035246  001356                           BNE     48              ;BRANCH IF NOT
10800  035250  052710  000010                   BIS     #MSTCK,@R0      ;SET SECTOR
10801  035254  042710  000010                   BIC     #MSTCK,@R0      ;CLEAR SECTOR
10802  035260  005303                           DEC     R3              ;IS REQUIRED NO OF SECTORS COMPLETE
10803  035262  001317                           BNE     18              ;BRANCH IF NOT
10804
10805  035264                          78:
  (2)  035264  012605                           MOV     (SP)+,R5        ;;POP STACK INTO R5
  (2)  035266  012604                           MOV     (SP)+,R4        ;;POP STACK INTO R4
  (2)  035270  012603                           MOV     (SP)+,R3        ;;POP STACK INTO R3
  (2)  035272  012602                           MOV     (SP)+,R2        ;;POP STACK INTO R2
  (2)  035274  012601                           MOV     (SP)+,R1        ;;POP STACK INTO R1
  (2)  035276  012600                           MOV     (SP)+,R0        ;;POP STACK INTO R0
10806  035300  000201                           RTS     R1
10807
10808
10809                                   ;*****************************************************************
10810                                   ;READ ONE SECTOR OF DATA
```

```
10811                                   ;*****************************************************************
10812
10813  035302  000000                   RNO:    0                       ;NO. OF WORDS READ
10814  035304  000000                   RCOM:   0                       ;EXTRA STORAGE
10815
10816
10817
10818  035306  012137  035302           REDATA: MOV     (R1)+,@#RNO     ;SAVE NO. OF WORDS ONLY FOR INFORMATION
10819  035312  012137  035304                   MOV     (R1)+,@#RCOM    ;EXTRA WORD ONLY FOR INFORMATION
10820  035316  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10821  035320  005737  002012                   TST     @#TSECC         ;IS THIS AN ECC TEST
10822  035324  001403                           BEQ     1$              ;BRANCH IF NO
10823  035326  012737  177777  027132           MOV     #-1,@#TSECCG    ;THESE BITS ARE TO GENERATE ECC
10824  035334  012702  000402           1$:     MOV     #258.,R2        ;256 WORDS PER SECTOR
10825                                           ;PLUS 2 ECC WORDS
10826  035340  012703  032734                   MOV     #DISK,R3        ;POINTE TO DISK SIMULATION
10827  035344  012337  031726           2$:     MOV     (R3)+,@#WORD    ;READY TO READ CONTENTS
10828  035350  004737  031732                   JSR     PC,@#READ               ;READ
10829  035354  005302                           DEC     R2              ;IS 256 WORDS DONE?
10830  035356  001372                           BNE     2$              ;IF NOT BRANCH
10831  035360  005737  002012                   TST     @#TSECC         ;IS THIS AN ECC TEST
10832  035364  001012                           BNE     4$              ;BRANCH OUT IF YES
10833  035366  005037  027132                   CLR     @#TSECCG        ;NO MORE ECC BITS ARE TO BE GENERATED
10834  035372  012702  000017                   MOV     #15.,R2         ;ONE DATA GAP, 14 TOLERANCE GAP
10835  035376  012337  031726           3$:     MOV     (R3)+,@#WORD    ;READY TO READ CONTENTS OF WORD
10836  035402  004737  031732                   JSR     PC,@#READ       ;READ
10837  035406  005302                           DEC     R2              ;COUNT
10838  035410  001372                           BNE     3$              ;BRANCH IF 14 NOT DONE
10839  035412                           4$:
  (2)  035412  012601                           MOV     (SP)+,R1        ;;POP STACK INTO R1
10840  035414  000201                           RTS     R1              ;RETURN
10841
10842
10843
10844
10845  035416                   RPVECT:
  (1)  035416  104400  035424               TYPE    ,.+4            ;;TYPE ASCIZ STRING
  (1)  035422  000411                       BR      64$             ;;GET OVER THE ASCIZ
  (1)                                       ;;.ASCIZ      /TRAPED FROM PC = /
  (1)  035446                       64$:
10846  035446  104402                       TYPOC                           ;TYPE FROM PC
10847  035450  012777  035416  144142       MOV     #RPVECT,@#RPVEC         ;RESTORE TRAP RP04 VECTOR
10848  035456  000000                       HALT                            ;CHANGE TO CONTINUE
10849
10850
10851
10852
```

```
                                                          ;*************************
                                        ;*******************************************************
   (1)
   (1)                                  .SBTTL  SCOPE HANDLER ROUTINE
   (1)
   (1)                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
   (1)                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
   (1)                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
   (1)                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
   (1)                                  ;*SW14=1        LOOP ON TEST
   (1)                                  ;*SW11=1        INHIBIT ITERATIONS
   (1)                                  ;*SW09=1        LOOP ON ERROR
   (1)                                  ;*SW08=1        LOOP ON TEST IN SWR<7:0>
   (1)                                  ;*CALL
   (1)                                  ;*      SCOPE           ;;SCOPE=IOT
   (1)
   (1)  035460                          $SCOPE:
   (3)  035460  005037  031132                  CLR     @#NOSYNC                ;CLEAR FLAG FOR HEADER ERROR COMMANDS
   (3)  035464  005037  002012                  CLR     @#TSECC                 ;CLEAR FLAG FOR ECC TEST
   (3)                                                                          ;WHEN =177777 IT IS AN ECC TEST
   (3)                                                                          ;WHEN =0 IT IS NOT AN ECC TEST
   (3)
   (3)  035470  005037  027132                  CLR     @#TSECCG                ;EVEN IN AN ECC TEST EVERY CLOCK
   (3)                                                                          ;IS NOT TO GENERATE ECC
   (3)                                                                          ;IF =177777 GENERATE ECC
   (3)                                                                          ;IF =0 DO NOT GENERATE ECC
   (3)  035474  005037  002014                  CLR     @#TESDTE                ;DRIVE TIMING ERROR TEST
   (3)  035500                          10:
   (1)  035500  006137  177570                  ROL     @#SWR           ;;LOOP ON PRESENT TEST?
   (1)  035504  100511                          BMI     $OVER           ;;YES IF SW14=1
   (1)                                  ;#####START OF CODE FOR THE XOR TESTER#####
   (1)  035506  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
   (1)                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
   (1)  035510  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
   (1)  035514  012737  035534  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
   (1)  035522  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
   (1)  035526  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
   (1)  035532  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
   (1)  035534  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
   (1)  035536  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
   (1)  035542  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
   (1)  035544                          6$:;#####END OF CODE FOR THE XOR TESTER#####
   (1)  035544  032737  000400  177570          BIT     #BIT08,@#SWR    ;;LOOP ON SPEC. TEST?
   (1)  035552  001404                          BEQ     2$              ;;BR IF NO
   (1)  035554  123767  177570  143320          CMPB    @#SWR,$TSTNM    ;;ON THE RIGHT TEST?SWR<7:0>
   (1)  035562  001462                          BEQ     $OVER           ;;BR IF YES
   (1)  035564  105767  143313          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
   (1)  035570  001421                          BEQ     3$              ;;BR IF NO
   (1)  035572  126767  143317  143303          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
   (1)  035600  101015                          BHI     3$              ;;BR IF NO
   (1)  035602  032737  001000  177570          BIT     #BIT09,@#SWR    ;;LOOP ON ERROR?
   (1)  035610  001404                          BEQ     4$              ;;BR IF NO
   (1)  035612  016767  143272  143266  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
   (1)  035620  000443                          BR      $OVER
```

F13

```
(1)  035622  105067  143255          48:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  035626  005067  143352                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  035632  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
(1)  035634  032737  004000  177570  3$:     BIT     #BIT11,@#SWR    ;;INHIBIT ITERATIONS?
(1)  035642  001011                          BNE     1$              ;;BR IF YES
(1)  035644  005767  143230                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(1)  035650  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
(1)  035652  005267  143226                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
(1)  035656  026767  143322  143220          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(1)  035664  002021                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
(1)  035666  012767  000001  143210  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(1)  035674  016767  000044  143302          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(1)  035702  105267  143174          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(1)  035706  011667  143174                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  035712  011667  143172                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
(1)  035716  005067  143264                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)  035722  112767  000001  143165          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1)  035730  016737  143146  177570  $OVER:  MOV     $TSTNM,@#DISPLAY ;;DISPLAY TEST NUMBER
(1)  035736  016716  143144                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  035742  000002                          RTI                     ;;FIXES PS
(1)  035744  000004                  $MXCNT: 4                       ;;MAX. NUMBER OF ITERATIONS
```

```
 10857                                  ;*****************************************************************
   (1)
   (1)                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
   (1)
   (1)                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
   (1)                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT, DEPENDING ON WHETHER THE
   (1)                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
   (1)                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER, LEADING ZEROS WILL ALWAYS BE
   (1)                                  ;*REPLACED WITH SPACES.
   (1)                                  ;*CALL:
   (1)                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
   (1)                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
   (1)
   (2)  035746                          $TYPDS:
   (3)  035746  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
   (3)  035750  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
   (3)  035752  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
   (3)  035754  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
   (3)  035756  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
   (1)  035760  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
   (1)  035764  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
   (1)  035770  100004                          BPL     1$              ;;BR IF INPUT IS POS.
   (1)  035772  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
   (1)  035774  112766  000655  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
   (1)  036002  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
   (1)  036004  012703  036162                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
   (1)  036010  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
   (1)  036014  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
   (1)  036016  016001  036152                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
   (1)  036022  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
   (1)  036024  002402                          BLT     4$              ;;BR IF DONE
   (1)  036026  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
   (1)  036030  000774                          BR      3$
   (1)  036032  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
   (1)  036034  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
   (1)  036036  001002                          BNE     5$              ;;FALL THROUGH IF 0
   (1)  036040  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
   (1)  036042  100407                          BMI     7$              ;;BR IF YES
   (1)  036044  106316                  5$:     ASLB    (SP)            ;;MSD?
   (1)  036046  103003                          BCC     6$              ;;BR IF NO
   (1)  036050  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
   (1)  036056  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
   (1)  036062  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
   (1)  036066  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
   (1)  036070  005720                          TST     (R0)+           ;;JUST INCREMENTING
   (1)  036072  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
   (1)  036076  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
   (1)  036100  003002                          BGT     8$              ;;GO TO EXIT
   (1)  036102  010502                          MOV     R5,R2           ;;GET THE LSD
   (1)  036104  000764                          BR      6$              ;;GO CHANGE TO ASCII
   (1)  036106  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
   (1)  036110  100003                          BPL     9$              ;;BR IF NO
   (1)  036112  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
   (1)  036120  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
```

```
   (3)  036122  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
   (3)  036124  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
   (3)  036126  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
   (3)  036130  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
   (3)  036132  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
   (1)  036134  104400  036162                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
   (1)  036140  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
   (1)  036146  012616                        MOV     (SP)+,(SP)
   (1)  036150  000002                        RTI                     ;;RETURN TO USER
   (1)  036152  023420                $DTBL:  10000.
   (1)  036154  001750                        1000.
   (1)  036156  000144                        100.
   (1)  036160  000012                        10.
   (1)  036162  000004                $DBLK:  .BLKW   4
 10858                                ;*****************************************************************
   (1)
   (1)                                .SBTTL  TYPE ROUTINE
   (1)
   (1)                                ;*ROUTINE TO TYPE ASCII MESSAGE, MESSAGE MUST TERMINATE WITH A 0 BYTE.
   (1)                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
   (1)                                ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
   (1)                                ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
   (1)                                ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
   (1)                                ;*
   (1)                                ;*CALL:
   (1)                                ;*1) USING A TRAP INSTRUCTION
   (1)                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCII STRING
   (1)                                ;*OR
   (1)                                ;*      TYPE
   (1)                                ;*      MESADR
   (1)                                ;*
   (1)                                ;*2) USING A JSR INSTRUCTION
   (1)                                ;*      MOV     PS,-(SP)        ;;PUSH PROCESSOR STATUS WORD ON THE STACK
   (1)                                ;*      JSR     PC,$TYPE        ;;CALL TYPE ROUTINE
   (1)                                ;*      MESADDR                 ;;FIRST ADRESS OF MESSAGE
   (1)
   (1)  036172  105767  142753        $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
   (1)  036176  100002                        BPL     1$              ;;BR IF YES
   (1)  036200  000000                        HALT                    ;;HALT HERE IF NO TERMINAL
   (1)  036202  000407                        BR      3$              ;;LEAVE
   (1)  036204  010046                1$:     MOV     R0,-(SP)        ;;SAVE R0
   (1)  036206  017600  000002                MOV     2(SP),R0        ;;GET ADDRESS OF ASCII STRING
   (1)  036212  112046                2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
   (1)  036214  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
   (1)  036216  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
   (1)  036220  012600                        MOV     (SP)+,R0        ;;RESTORE R0
   (1)  036222  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
   (1)  036226  000002                        RTI                     ;;RETURN
   (1)  036230  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
   (1)  036234  001424                        BEQ     8$
   (1)  036236  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT
   (1)  036242  001004                        BNE     5$
   (1)  036244  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
   (1)  036246  104400  001215                TYPE,   $CRLF
```

```
    (1)  036252  000757                        BR      2$                ;;GET NEXT CHARACTER
    (1)  036254  004767  000052          5$:   JSR     PC,$TYPEC         ;;GO TYPE THIS CHARACTER
    (1)  036260  126726  142664          6$:   CMPB    $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
    (1)  036264  001352                        BNE     2$                ;;IF NO GO GET NEXT CHAR.
    (1)  036266  016746  142654                MOV     $NULL,-(SP)       ;;GET # OF FILLER CHARS. NEEDED
    (1)                                                                   ;;AND THE NULL CHAR.
    (1)  036272  105366  000001          7$:   DECB    1(SP)             ;;DOES A NULL NEED TO BE TYPED?
    (1)  036276  002770                        BLT     6$                ;;BR IF NO--GO POP THE NULL OFF OF STACK
    (1)  036300  004767  000026                JSR     PC,$TYPEC         ;;GO TYPE A NULL
    (1)  036304  000772                        BR      7$                ;;LOOP
    (1)
    (1)                                ;HORIZONTAL TAB PROCESSOR
    (1)
    (1)  036306  112716  000040          8$:   MOVB    #40,(SP)          ;;REPLACE TAB WITH SPACE
    (1)  036312  004767  000014          9$:   JSR     PC,$TYPEC         ;;TYPE A SPACE
    (1)  036316  132767  000007  000052        BITB    #7,$CHARCNT       ;;BRANCH IF NOT AT
    (1)  036324  001372                        BNE     9$                ;;TAB STOP
    (1)  036326  005726                        TST     (SP)+             ;;POP SPACE OFF STACK
    (1)  036330  000730                        BR      2$                ;;GET NEXT CHARACTER
    (1)  036332  105777  142604          $TYPEC: TSTB    @#TPS             ;;WAIT UNTIL PRINTER IS READY
    (1)  036336  100375                        BPL     $TYPEC
    (1)  036340  116677  000002  142576        MOVB    2(SP),@#TPB       ;;LOAD CHAR TO BE TYPED INTO DATA REG.
    (1)  036346  122766  000015  000002        CMPB    #15,2(SP)         ;;BRANCH IF
    (1)  036354  001003                        BNE     1$                ;;NOT <CR>
    (1)  036356  105067  000014                CLRB    $CHARCNT          ;;
    (1)  036362  000406                        BR      $TYPEX            ;;EXIT
    (1)  036364  122766  000012  000002  1$:   CMPB    #12,2(SP)         ;;BRANCH IF
    (1)  036372  002002                        BGE     $TYPEX            ;;<LF>
    (1)  036374  105227                        INCB    (PC)+             ;;INC SPACE
    (1)  036376  000000                $CHARCNT:.WORD  0                 ;;COUNT
    (1)  036400  000207                $TYPEX: RTS     PC
    (1)  036402  000207                        RTS     PC
    (1)                                ;;      EQUATES
    (1)          000200                        CRLF=   200               ;;<CR><LF> EQUIV
    (1)          000011                        HT=     11                ;;<HT>
    (1)
10859                                  ;*******************************************************************

    (1)
    (1)                                .SBTTL  TTY INPUT ROUTINE
    (1)
    (1)  036404  000000                $TKCNT: .WORD   0                 ;;NUMBER OF ITEMS IN QUEUE
    (1)  036406  000000                $TKQIN: .WORD   0                 ;;INPUT POINTER
    (1)  036410  000000                $TKQOUT: .WORD  0                 ;;OUTPUT POINTER
    (1)  036412  000011                $TKQSRT: .BLKB  9.                ;;TTY KEYBOARD QUEUE
    (1)          036423                $TKQEND=.
    (1)          036424                .EVEN
    (1)
    (1)                                ;*TK INITIALIZE ROUTINE
    (1)                                ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
    (1)                                ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
    (1)                                ;
    (1)                                ;*CALL:
    (1)                                ;*      JSR     PC,$TKINT
    (1)                                ;*      RETURN
```

```
(1)
(1)  036424  005067  177754          ;
(1)  036430  012767  036412  177750  $TKINT: CLR     $TKCNT           ;;CLEAR COUNT OF ITEMS IN QUEUE
(1)  036436  016767  177744  177744          MOV     #$TKQSRT,$TKQIN  ;;MOVE THE STARTING ADDRESS OF THE
(1)  036444  012737  036474  000060          MOV     $TKQIN,$TKQOUT   ;;QUEUE INTO THE INPUT & OUTPUT POINTERS,
(1)  036452  012737  000200  000062          MOV     #$TKSRV,@#TKVEC  ;;INITIALIZE THE KEYBOARD VECTOR
(1)  036460  005777  142454                  MOV     #200,@#TKVEC+2   ;;"BR" LEVEL 4
(1)  036464  012777  000100  142444          TST     @#TKB            ;;CLEAR DONE FLAG
(1)  036472  000207                          MOV     #100,@#TKS       ;;ENABLE TTY KEYBOARD INTERRUPT
(1)                                          RTS     PC               ;;RETURN TO CALLER
(1)
(1)                                  ;*TK SERVICE ROUTINE
(1)                                  ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
(1)                                  ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
(1)                                  ;*IT IN THE QUEUE.
(1)                                  ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
(1)                                  ;*UPON RETURN EXIT IS MAKE TO THE "CONTROL-C" RESTART ADDRESS (OPERSEL)
(1)                                  ;
(1)  036474  117746  142440          $TKSRV: MOVB    @#TKB,-(SP)      ;;PICKUP THE CHARACTER
(1)  036500  042716  177600                  BIC     #^C177,(SP)      ;;STRIP THE JUNK
(1)  036504  021627  000003                  CMP     (SP),#3          ;;IS IT A CONTROL C?
(1)  036510  001006                          BNE     1$               ;;BRANCH IF NO
(1)  036512  104400  037005                  TYPE    ,$CNTLC          ;;TYPE A CONTROL-C (^C)
(1)  036516  004767  177702                  JSR     PC,$TKINT        ;;INIT THE KEYBOARD
(1)  036522  000167  164662                  JMP     OPERSEL          ;;CONTROL C RESTART
(1)  036526  022767  000011  177650  1$:     CMP     #9.,$TKCNT       ;;IS THE QUEUE FULL?
(1)  036534  001004                          BNE     2$               ;;BRANCH IF NO
(1)  036536  104400  001210                  TYPE    ,$BELL           ;;RING THE TTY BELL
(1)  036542  005726                          TST     (SP)+            ;;CLEAN CHARACTER OFF OF STACK
(1)  036544  000415                          BR      3$               ;;EXIT
(1)  036546  005267  177632          2$:     INC     $TKCNT           ;;COUNT THIS CHARACTER
(1)  036552  112677  177630                  MOVB    (SP)+,@$TKQIN    ;;AND PUT IT IN QUEUE
(1)  036556  005267  177624                  INC     $TKQIN           ;;UPDATE THE POINTER
(1)  036562  026727  177620  036423          CMP     $TKQIN,#$TKQEND  ;;GO OFF THE END?
(1)  036570  001003                          BNE     3$               ;;BRANCH IF NO
(1)  036572  012767  036412  177606          MOV     #$TKQSRT,$TKQIN  ;;RESET THE POINTER
(1)  036600  000002                  3$:     RTI                      ;;RETURN
(2)                                  ;*****************************************************************
(1)                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                  ;*CALL:
(1)                                  ;*      RDCHR                    ;;GET A CHARACTER FROM THE QUEUE
(1)                                  ;*      RETURN HERE              ;;CHARACTER IS ON THE STACK
(1)                                  ;*                               ;;WITH PARITY BIT STRIPPED OFF
(1)                                  ;
(1)
(1)  036602  011646                  $RDCHR: MOV     (SP),-(SP)       ;;PUSH DOWN THE PC AND
(1)  036604  016666  000004  000002          MOV     4(SP),2(SP)      ;;THE PS
(1)  036612  005066  000004                  CLR     4(SP)            ;;GET READY FOR A CHARACTER
(1)  036616  005037  177776                  CLR     @#PS             ;;ALLOW INTERRUPTS
(1)  036622  005767  177556          1$:     TST     $TKCNT           ;;WAIT ON A CHARACTER
(1)  036626  001775                          BEQ     1$
(1)  036630  005367  177550                  DEC     $TKCNT           ;;DECREMENT THE COUNTER
(1)  036634  117766  177550  000004          MOVB    @$TKQOUT,4(SP)   ;;GET ONE CHARACTER
(1)  036642  005267  177542                  INC     $TKQOUT          ;;UPDATE THE POINTER
(1)  036646  026727  177536  036423          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
```

```
(1)  036654  001003                        BNE     2$                ;;BRANCH IF NO
(1)  036656  012767  036412  177524        MOV     $TKQSRT,$TKQOUT   ;;RESET THE POINTER
(1)  036664  000002                 2$:    RTI                       ;;RETURN
(2)                                 ;*************************************************************
(1)                                 ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1)                                 ;*CALL:
(1)                                 ;*      RDLIN                     ;;INPUT A STRING FROM THE TTY
(1)                                 ;*      RETURN HERE               ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1)                                 ;*                                ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
(1)
(1)  036666  010346                 $RDLIN: MOV     R3,-(SP)          ;;SAVE R3
(1)  036670  012703  036774         1$:     MOV     $TTYIN,R3         ;;GET ADDRESS
(1)  036674  022703  037005         2$:     CMP     $TTYIN+9.,R3      ;;BUFFER FULL?
(1)  036700  101405                         BLOS    4$                ;;BR IF YES
(1)  036702  104412                         RDCHR                     ;;GO READ ONE CHARACTER FROM THE TTY
(1)  036704  112613                         MOVB    (SP)+,(R3)        ;;GET CHARACTER
(1)  036706  122713  000177         10$:    CMPB    #177,(R3)         ;;IS IT A RUBOUT
(1)  036712  001003                         BNE     3$                ;;SKIP IF NOT
(1)  036714  104400  001214         4$:     TYPE    ,$QUES            ;;TYPE A '?'
(1)  036720  000763                         BR      1$                ;;CLEAR THE BUFFER AND LOOP
(1)  036722  111367  000044         3$:     MOVB    (R3),9$           ;;ECHO THE CHARACTER
(1)  036726  104400  036772                 TYPE    ,9$
(1)  036732  122723  000015                 CMPB    #15,(R3)+         ;;CHECK FOR RETURN
(1)  036736  001356                         BNE     2$                ;;LOOP IF NOT RETURN
(1)  036740  105063  177777                 CLRB    -1(R3)            ;;CLEAR RETURN (THE 15)
(1)  036744  104400  001216                 TYPE    ,$LF              ;;TYPE A LINE FEED
(1)  036750  012603                         MOV     (SP)+,R3          ;;RESTORE R3
(1)  036752  011646                         MOV     (SP),-(SP)        ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  036754  016666  000004  000002         MOV     4(SP),2(SP)       ;;      FIRST ASCII CHARACTER ON IT
(1)  036762  012766  036774  000004         MOV     $TTYIN,4(SP)
(1)  036770  000002                         RTI                       ;;RETURN
(1)  036772     000                 9$:     .BYTE   0                 ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  036773     000                         .BYTE   0                 ;;TERMINATOR
(1)  036774  000011                 $TTYIN: .BLKB   9.                ;;RESERVE 9. BYTES FOR TTY INPUT
(1)  037005     136  000503  000012 $CNTLC: .ASCIZ  /^C/<15><12>       ;;CONTROL "C"
10860                                                                          ;FROM THE TTY
10861
10862
10863
10864                               ;*************************************************************
(1)
(1)                                 .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
(1)
(1)                                 ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
(1)                                 ;*CHANGE IT TO BINARY.
(1)                                 ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
(1)                                 ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
(1)                                 ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
(1)                                 ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
(1)                                 ;*CALL:
(1)                                 ;*      RDOCT                     ;;READ AN OCTAL NUMBER
(1)                                 ;*      RETURN HERE               ;;LOW ORDER BITS ARE ON TOP OF THE STACK
(1)                                 ;*                                ;;HIGH ORDER BITS ARE IN $HIOCT
(1)
```

```
(1)  037012  011646                    $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
(1)  037014  016666  000004  000002             MOV     4(SP),2(SP)     ;;INPUT NUMBER
(3)  037022  010046                             MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  037024  010146                             MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  037026  010246                             MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(1)  037030  104414                    1$:      RDLIN                   ;;READ AN ASCIZ LINE
(1)  037032  012600                             MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
(1)  037034  010067  000100                     MOV     R0,5$           ;;AND SAVE IT
(1)  037040  005001                             CLR     R1              ;;CLEAR DATA WORD
(1)  037042  005002                             CLR     R2
(1)  037044  112046                    2$:      MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
(1)  037046  001420                             BEQ     3$              ;;IF ZERO GET OUT
(1)  037050  122716  000060                     CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
(1)  037054  003026                             BGT     4$              ;;IS AN OCTAL DIGIT
(1)  037056  122716  000067                     CMPB    #'7,(SP)
(1)  037062  002423                             BLT     4$
(1)  037064  006301                             ASL     R1              ;;*2
(1)  037066  006102                             ROL     R2
(1)  037070  006301                             ASL     R1              ;;*4
(1)  037072  006102                             ROL     R2
(1)  037074  006301                             ASL     R1              ;;*8
(1)  037076  006102                             ROL     R2
(1)  037100  042716  177770                     BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
(1)  037104  062601                             ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
(1)  037106  000756                             BR      2$              ;;LOOP
(1)  037110  005726                    3$:      TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
(1)  037112  010166  000012                     MOV     R1,12(SP)       ;;SAVE THE RESULT
(1)  037116  010267  000026                     MOV     R2,$HIOCT
(3)  037122  012602                             MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  037124  012601                             MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  037126  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  037130  000002                             RTI                     ;;RETURN
(1)  037132  005726                    4$:      TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
(1)  037134  105010                             CLRB    (R0)            ;;SET A TERMINATOR
(1)  037136  104400                             TYPE                    ;;TYPE UP THRU THE BAD CHAR.
(1)  037140  000000                    5$:      .WORD   0
(1)  037142  104400  001214                     TYPE    ,$QUES          ;;"?" "CR" & "LF"
(1)  037146  000730                             BR      1$              ;;TRY AGAIN
(1)  037150  000000                    $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
```

```
10866                                   ;*****************************************************************
  (1)
  (1)                                   .SBTTL  ERROR HANDLER ROUTINE
  (1)
  (1)                                   ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
  (1)                                   ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
  (1)                                   ;*AND GO TO $ERRTYP ON ERROR
  (1)                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
  (1)                                   ;*SW15=1        HALT ON ERROR
  (1)                                   ;*SW13=1        INHIBIT ERROR TYPEOUTS
  (1)                                   ;*SW10=1        BELL ON ERROR
  (1)                                   ;*SW09=1        LOOP ON ERROR
  (1)                                   ;*CALL
  (1)                                   ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
  (1)
  (1)  037152                           $ERROR:
  (3)
  (3)  037152  012737  177777  001774            MOV     #-1,@#$ERFLG    ;SET ERROR FLAG
  (3)  037160                           REGSA1:
  (3)
  (3)
  (1)  037160  105267  141717           7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
  (1)  037164  001775                           BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
  (1)  037166  016737  141710  177570           MOV     $TSTNM,@#DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
  (1)  037174  032737  002000  177570           BIT     #BIT10,@#SWR    ;;BELL ON ERROR?
  (1)  037202  001402                           BEQ     1$              ;;NO - SKIP
  (1)  037204  104400  001210                   TYPE    ,$BELL          ;;RING BELL
  (1)  037210  005267  141676           1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
  (1)  037214  011667  141676                   MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
  (1)  037220  162767  000002  141670           SUB     #2,$ERRPC
  (1)  037226  117767  141664  141660           MOVB    @#$ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
  (1)  037234  032737  020000  177570           BIT     #BIT13,@#SWR    ;;SKIP TYPEOUT IF SET
  (1)  037242  001004                           BNE     2$              ;;SKIP TYPEOUTS
  (1)  037244  004737  037314                   JSR     PC,@#$ERRTYP    ;;GO TO USER ERROR ROUTINE
  (1)  037250  104400  001215                   TYPE    ,$CRLF
  (1)  037254  005737  177570           2$:     TST     @#SWR           ;;HALT ON ERROR
  (1)  037260  100001                           BPL     3$              ;;SKIP IF CONTINUE
  (1)  037262  000000                           HALT                    ;;HALT ON ERROR!
  (1)  037264  032737  001000  177570   3$:     BIT     #BIT09,@#SWR    ;;LOOP ON ERROR SWITCH SET?
  (1)  037272  001402                           BEQ     4$              ;;BR IF NO
  (1)  037274  016716  141610                   MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
  (1)  037300  005767  141702           4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
  (1)  037304  001402                           BEQ     5$              ;;BR IF NONE
  (1)  037306  016716  141674                   MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
  (1)  037312                           5$:
  (1)  037312  000002                           RTI                     ;;RETURN
10867                                   ;*****************************************************************
  (1)
  (1)                                   .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
  (1)
  (1)                                   ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
  (1)                                   ;*ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
  (1)                                   ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR,
  (1)
```

```
    (1)  037314                          $ERRTYP:
    (1)  037314  104400  001215                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
    (1)  037320  010046                          MOV     R0,-(SP)        ;;SAVE R0
    (1)  037322  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
    (1)  037324  153700  001114                  BISB    @#$ITEMB,R0
    (1)  037330  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
    (1)                                                                  ;;TYPE THE PC OF THE ERROR
    (2)  037332  016746  141560                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
    (2)                                                                  ;;ERROR ADDRESS
    (2)  037336  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    (1)  037340  000445                          BR      10$             ;;GET OUT
    (1)  037342  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
    (1)  037344  006300                          ASL     R0              ;;       WORK FOR THE ERROR TABLE
    (1)  037346  006300                          ASL     R0
    (1)  037350  006300                          ASL     R0
    (1)  037352  062700  001220                  ADD     @#$ERRTB,R0     ;;FORM TABLE POINTER
    (1)  037356  012067  000004                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
    (1)  037362  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
    (1)  037364  104400                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
    (1)  037366  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
    (1)  037370  104400  001215                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
    (1)  037374  012067  000004          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
    (1)  037400  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
    (1)  037402  104400                          TYPE                    ;;TYPE THE "DATA HEADER"
    (1)  037404  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
    (1)  037406  104400  001215                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
    (1)  037412  010146                  5$:     MOV     R1,-(SP)        ;;SAVE R1
    (1)  037414  012001                          MOV     (R0)+,R1        ;;PICKUP "DATA TABLE" POINTER
    (1)  037416  001415                          BEQ     9$              ;;BR IF NO DATA TO BE TYPED
    (1)  037420  012000                          MOV     (R0)+,R0        ;;PICKUP "DATA FORMAT" POINTER
    (1)  037422  105720                  6$:     TSTB    (R0)+           ;;"OCTAL" OR "DECIMAL"
    (1)  037424  001003                          BNE     7$              ;;BR IF DECIMAL
    (2)  037426  013146                          MOV     0(R1)+,-(SP)    ;;SAVE 0(R1)+ FOR TYPEOUT
    (2)  037430  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    (1)  037432  000402                          BR      8$
    (2)  037434                          7$:
    (2)  037434  013146                          MOV     0(R1)+,-(SP)    ;;SAVE 0(R1)+ FOR TYPEOUT
    (2)  037436  104410                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
    (1)  037440  005711                  8$:     TST     (R1)            ;;IS THERE ANOTHER NUMBER?
    (1)  037442  001403                          BEQ     9$              ;;BR IF NO
    (1)  037444  104400  037464                  TYPE    ,11$            ;;TYPE TWO(2) SPACES
    (1)  037450  000764                          BR      6$              ;;LOOP
    (1)
    (1)  037452  012601                  9$:     MOV     (SP)+,R1        ;;RESTORE R1
    (1)  037454  012600                  10$:    MOV     (SP)+,R0        ;;RESTORE R0
    (1)  037456  104400  001215                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
    (1)  037462  000207                          RTS     PC              ;;RETURN
    (1)  037464  020040      000        11$:     .ASCIZ  /  /            ;;TWO(2) SPACES
    (1)          037470                          .EVEN
 10068                                ;*****************************************************************
 10069                                ;*****************************************************************
    (1)
    (1)                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
    (1)
```

```
(1)                                     ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1)                                     ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(1)                                     ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPOS                   ;;CALL FOR TYPEOUT
(1)                                     ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1)                                     ;*      .BYTE   M               ;;M=1 OR 0
(1)                                     ;*                                      ;;1=TYPE LEADING ZEROS
(1)                                     ;*                                      ;;0=SUPPRESS LEADING ZEROS
(1)                                     ;*
(1)                                     ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1)                                     ;*$TYPOS OR $TYPOC
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPON                   ;;CALL FOR TYPEOUT
(1)                                     ;*
(1)                                     ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPOC                   ;;CALL FOR TYPEOUT
(1)
(1)  037470  017646  000000          $TYPOS: MOV     0(SP),-(SP)     ;;PICKUP THE MODE
(1)  037474  116667  000001  000211          MOVB    1(SP),$0FILL    ;;LOAD ZERO FILL SWITCH
(1)  037502  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1)  037506  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(1)  037512  000406                          BR      $TYPON
(1)  037514  112767  000001  000171  $TYPOC: MOVB    #1,$0FILL       ;;SET THE ZERO FILL SWITCH
(1)  037522  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(1)  037530  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
(1)  037536  010346                          MOV     R3,-(SP)        ;;SAVE R3
(1)  037540  010446                          MOV     R4,-(SP)        ;;SAVE R4
(1)  037542  010546                          MOV     R5,-(SP)        ;;SAVE R5
(1)  037544  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  037550  005404                          NEG     R4
(1)  037552  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  037556  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(1)  037562  116704  000125                  MOVB    $0FILL,R4       ;;GET THE ZERO FILL SWITCH
(1)  037566  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(1)  037572  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
(1)  037574  006105                  18:     ROL     R5              ;;ROTATE MSB INTO "C"
(1)  037576  000404                          BR      38              ;;GO DO MSB
(1)  037600  006105                  28:     ROL     R5              ;;FORM THIS DIGIT
(1)  037602  006105                          ROL     R5
(1)  037604  006105                          ROL     R5
(1)  037606  010503                          MOV     R5,R3
(1)  037610  006103                  38:     ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  037612  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  037616  100016                          BPL     78              ;;BR IF NO
(1)  037620  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
(1)  037624  001002                          BNE     48              ;;TEST FOR 0
(1)  037626  005704                          TST     R4              ;;SUPPRESS THIS 0?
(1)  037630  001403                          BEQ     58              ;;BR IF YES
(1)  037632  005204                  48:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
```

```
(1)  037634  052703  000060            BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  037640  052703  000040      5$:   BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  037644  110367  000040            MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  037650  104400  037710            TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  037654  105367  000032      7$:   DECB    $OCNT           ;;COUNT BY 1
(1)  037660  003347                    BGT     2$              ;;BR IF MORE TO DO
(1)  037662  002402                    BLT     6$              ;;BR IF DONE
(1)  037664  005204                    INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  037666  000744                    BR      2$              ;;GO DO THE LAST DIGIT
(1)  037670  012605              6$:   MOV     (SP)+,R5        ;;RESTORE R5
(1)  037672  012604                    MOV     (SP)+,R4        ;;RESTORE R4
(1)  037674  012603                    MOV     (SP)+,R3        ;;RESTORE R3
(1)  037676  016666  000002  000004    MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  037704  012616                    MOV     (SP)+,(SP)
(1)  037706  000002                    RTI                     ;;RETURN
(1)  037710     000             8$:   .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  037711     000                   .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  037712     000             $OCNT: .BYTE  0               ;;OCTAL DIGIT COUNTER
(1)  037713     000             $0FILL: .BYTE 0               ;;ZERO FILL SWITCH
(1)  037714  000000             $OMODE: .WORD 0               ;;NUMBER OF DIGITS TO TYPE
```

```
 10871                                  ;*****************************************************************
   (1)
   (1)                                  .SBTTL  TRAP DECODER
   (1)
   (1)                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
   (1)                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
   (1)                                  ;*OF THE DESIRED ROUTINE, THEN USING THE ADDRESS OBTAINED IT WILL
   (1)                                  ;*GO TO THAT ROUTINE.
   (1)
   (1)  037716  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
   (1)  037720  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
   (1)  037724  005740                          TST     -(R0)           ;;BACKUP BY 2
   (1)  037726  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
   (1)  037730  016000  037736                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
   (1)  037734  000200                          RTS     R0              ;;GO TO ROUTINE
   (1)
   (3)
   (3)                                  .SBTTL  TRAP TABLE
   (3)
   (3)                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
   (3)                                  ;*BY THE "TRAP" INSTRUCTION.
   (3)
   (3)                                  ;       ROUTINE
   (3)                                  ;       -------
   (3)  037736                          $TRPAD:
   (3)  037736  036172                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
   (3)  037740  037514                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
   (3)  037742  037470                          $TYPOS  ;;CALL=TYPOS    TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
   (3)  037744  037530                          $TYPON  ;;CALL=TYPON    TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST CALL)
   (3)  037746  035746                          $TYPDS  ;;CALL=TYPDS    TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
   (3)  037750  036602                          $RDCHR  ;;CALL=RDCHR    TRAP+12(104412) TTY TYPEIN CHARACTER ROUTINE
   (3)  037752  036666                          $RDLIN  ;;CALL=RDLIN    TRAP+14(104414) TTY TYPEIN STRING ROUTINE
   (3)  037754  037012                          $RDOCT  ;;CALL=RDOCT    TRAP+16(104416) READ AN OCTAL NUMBER FROM TTY
 10872  037756  024452                          T.$COP  ;;CALL=$COP1     TRAP+20(104420) MY LOCAL $COPE$
 10873  037760  024524                          CHECKT  ;;CALL=CHECKD    TRAP+22(104422) CHECK DVA,RDY,DPR,DRY
 10874  037762  025026                          WAIT.T  ;;CALL=WAT       TRAP+24(104424) WAIT LOOP
 10875
 10876
 10877
```

```
10879                                   ;*****************************************************************
  (1)
  (1)                                   .SBTTL  POWER DOWN AND UP ROUTINES
  (1)
  (1)                                   ;POWER DOWN ROUTINE
  (1)  037764  012737  040112  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
  (1)  037772  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
  (3)  040000  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)  040002  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (3)  040004  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (3)  040006  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (3)  040010  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (3)  040012  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
  (1)  040014  010667  000076                 MOV     SP,$SAVR6       ;;SAVE SP
  (1)  040020  012737  040032  000024         MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
  (1)  040026  000000                         HALT
  (1)  040030  000776                         BR      .-2             ;;HANG UP
  (1)
  (1)                                   ;POWER UP ROUTINE
  (1)  040032  016706  000060  $PWRUP: MOV     $SAVR6,SP       ;;GET SP
  (1)  040036  005067  000054                 CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
  (1)  040042  005267  000050  1$:    INC     $SAVR6          ;;WAIT FOR THE INC
  (1)  040046  001375                         BNE     1$              ;;OF  WORD
  (3)  040050  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
  (3)  040052  012604                         MOV     (SP)+,R4        ;;POP STACK INTO R4
  (3)  040054  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
  (3)  040056  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
  (3)  040060  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
  (3)  040062  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
  (1)  040064  012737  037764  000024         MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
  (1)  040072  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
  (1)  040100  104400                         TYPE                    ;REPORT THE POWER FAILURE
  (1)  040102  040120                $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
  (1)  040104  012716                         MOV     (PC)+,(SP)      ;;RESTART AT BEGIN
  (1)  040106  004220                $PWRAD: .WORD   BEGIN           ;;RESTART ADDRESS
  (1)  040110  000002                         RTI
  (1)  040112  000000                $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
  (1)  040114  000776                         BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
  (1)  040116  000000                $SAVR6: 0                       ;;PUT THE SP HERE
  (1)  040120  005015  047520  042527 $POWER: .ASCIZ  <15><12>"POWER"
  (1)  040126  000122
  (1)                                       .EVEN
```

```
10881                                   ;*****************************************************************
10882                                   ;
10883                                   ;ERROR AND MESSAGE TABLE CONDIMENTS
10884                                   ;
10885                                   ;*****************************************************************
10886
10887
10888
10889
10890  040130  051127  047117  020107  EM1:    .ASCIZ  /WRONG DATA IN READING OR WRITING HARDWARE REGISTER/
       040136  040504  040524  044440
       040144  020116  042522  042101
       040152  047111  020107  051117
       040160  053440  044522  044524
       040166  043516  044040  051101
       040174  053504  051101  020105
       040202  042522  044507  052123
       040210  051105     000
10891  040213     105  051122  051117  EM2:    .ASCIZ  /ERROR ON  DATA COMMAND/
       040220  047440  020116  042040
       040226  052101  020101  047503
       040234  046515  047101  000104
10892  040242  051105  047522  020122  EM6:    .ASCIZ  /ERROR ON WRITE HEADER AND DATA/
       040250  047117  053440  044522
       040256  042524  044040  040505
       040264  042504  020122  047101
       040272  020104  040504  040524
       040300     000
10893  040301     103  047117  051124  EM11:   .ASCIZ  /CONTROLLER OR DRIVE STATUS/
       040306  046117  042514  020122
       040314  051117  042040  044522
       040322  042526  051440  040524
       040330  052524  000123
10894  040334  042522  044507  052123  EM14:   .ASCIZ  /REGISTER FAILED/
       040342  051105  043040  044501
       040350  042514  000104
10895  040354  050123  041505  043111  EM15:   .ASCIZ  /SPECIFIED REGISTER NON EXISTANT SO ABORT/
       040362  042511  020104  042522
       040370  044507  052123  051105
       040376  047040  047117  042440
       040404  044530  052123  047101
       040412  020124  047523  040440
       040420  047502  052122     000
10896  040425     127  044501  020124  EM16:   .ASCIZ  /WAIT LOOP FAILED/
       040432  047514  050117  043040
       040440  044501  042514  000104
10897  040446  051127  052111  020105  EM17:   .ASCIZ  /WRITE CHECK FAILING/
       040454  044103  041505  020113
       040462  040506  046111  047111
       040470  000107
10898  040472  042522  044507  052123  EM20:   .ASCIZ  /REGISTER FAILING/
       040500  051105  043040  044501
       040506  044514  043516     000
10899  040513     111  052116  051105  EM21:   .ASCIZ  /INTERRUPT FAILING/
```

```
                040520  052522  052120  043040
                040526  044501  044514  043516
                040534     000
 10900          040535     105  051122  051117  EM22:   .ASCII  /ERROR ON DRIVE PRESENT/<15><12>
                040542  047440  020116  051104
                040550  053111  020105  051120
                040556  051505  047105  006524
                040564     012
 10901          040565     124  042510  052440          .ASCII  /THE UNIT NO, FOUND BY SETTING RHAS/<15><12>
                040572  044516  020124  047516
                040600  020056  047506  047125
                040606  020104  054502  051440
                040614  052105  044524  043516
                040622  051040  040510  006523
                040630     012
 10902          040631     104  020117  047516          .ASCII  /DO NOT AGREE WITH THE UNIT NO, FOUND FROM/<15><12>
                040636  020124  043501  042522
                040644  020105  044527  044124
                040652  052040  042510  052440
                040660  044516  020124  047516
                040666  020056  047506  047125
                040674  020104  051106  046517
                040702  005015
 10903          040704  044122  051503  026462          .ASCII  /RHC82-NED BIT 12/<15><12>
                040712  042516  020104  044502
                040720  020124  030443  006462
                040726     012
 10904          040727     061  033467  033467          .ASCII  /177777-MEANS NO UNIT FOUND/<15><12>
                040734  026467  042515  047101
                040742  020123  047516  052440
                040750  044516  020124  047506
                040756  047125  006504     012
 10905          040763     116  052117  035105          .ASCII  /NOTE: ON DUAL PORT SYSTEM, DRIVE ON OTHER PORT WILL NOT GIVE/<15><12>
                040770  047440  020116  052504
                040776  046101  050040  051117
                041004  020124  054523  052123
                041012  046505  020054  051104
                041020  053111  020105  047117
                041026  047440  044124  051105
                041034  050040  051117  020124
                041042  044527  046114  047040
                041050  052117  043440  053111
                041056  006505     012
 10906          041061     116  042105  044040          .ASCIZ  /NED HENCE THERE WILL BE AN EXTRA DRIVE/
                041066  047105  042503  052040
                041074  042510  042522  053440
                041102  046111  020114  042502
                041110  040440  020116  054105
                041116  051124  020101  051104
                041124  053111  000105
 10907          041130  047514  045517  040440  EM24:   .ASCIZ  /LOOK AHEAD REGISTER AT THE BEGINNING OF SECTOR IS IN ERROR/
                041136  042510  042101  051040
                041144  043505  051511  042524
                041152  020122  052101  052040
```

```
                041160  042510  041040  043505
                041166  047111  044516  043516
                041174  047440  020106  042523
                041202  052103  051117  044440
                041210  020123  047111  042440
                041216  051122  051117     000
 10908          041223     114  047517  020113  EM25:   .ASCIZ  /LOOK AHEAD REGISTER IS IN ERHOR/
                041230  044101  040505  020104
                041236  042522  044507  052123
                041244  051105  044440  020123
                041252  047111  042440  051122
                041260  051117     000
 10909          041263     103  051125  042522  EM30:   .ASCII  /CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER REGIHSTER/<15><12>
                041270  052116  041440  046131
                041276  047111  042504  020122
                041304  047504  051505  047040
                041312  052117  046440  052101
                041320  044103  042040  051505
                041326  051111  042105  041440
                041334  046131  047111  042504
                041342  020122  042522  044507
                041350  051510  042524  006522
                041356     012
 10910          041357     101  052106  051105          .ASCIZ  /AFTER A SEEK AND INIT/
                041364  040440  051440  042505
                041372  020113  047101  020104
                041400  047111  052111     000
 10911          041405     105  041503  043440  EM31:   .ASCII  /ECC GENERATED IS INCORRECT/<15><12>
                041412  047105  051105  052101
                041420  042105  044440  020123
                041426  047111  047503  051122
                041434  041505  006524     012
 10912          041441     105  042526  054522          .ASCIZ  /EVERY WORD ON THIS SECTOR IS THAT GIVEN IN "DATA USED"/
                041446  053440  051117  020104
                041454  047117  052040  044510
                041462  020123  042523  052103
                041470  051117  044440  020123
                041476  044124  052101  043440
                041504  053111  047105  044440
                041512  020116  042042  052101
                041520  020101  051525  042105
                041526  000042
 10913          041530  047117  051040  040505  EM32:   .ASCII  /ON READ COMMAND AFTER DATA AND ECC HAVE BEEN READ/<15><12>
                041536  020104  047503  046515
                041544  047101  020104  043101
                041552  042524  020122  040504
                041560  040524  040440  042116
                041566  042440  041503  044040
                041574  053101  020105  042502
                041602  047105  051040  040505
                041610  006504     012
 10914          041613     105  041503  051040          .ASCII  /ECC REGISTERS OR RHER1 IS IN ERROR/<15><12>
                041620  043505  051511  042524
                041626  051522  047440  020122
```

```
                041634  044122  051105  020061
                041642  051511  044440  020116
                041650  051105  047522  006522
                041656     012
 10915          041657     117  046116  020131          .ASCII  /ONLY LOWER 11 BITS OF PATTERN REG, CAN BE READ/<15><12>
                041664  047514  042527  020122
                041672  030461  041040  052111
                041700  020123  043117  050040
                041706  052101  042524  047122
                041714  051040  043505  020056
                041722  040503  020116  042502
                041730  051040  040505  006504
                041736     012
 10916          041737     124  044510  020123          .ASCIZ  /THIS SHOULD MATCH LOWER 11 BITS OF GOOD ECC1/
                041744  044123  052517  042114
                041752  046440  052101  044103
                041760  046040  053517  051105
                041766  030440  020061  044502
                041774  051524  047440  020106
                042002  047507  042117  042440
                042010  041503  000061
 10917          042014  044510  044107  041440  EM33:   .ASCIZ  /HIGH COUNT BIT NOT SET AFTER 30859 CLOCKS/
                042022  052517  052116  041040
                042030  052111  047040  052117
                042036  051440  052105  040440
                042044  052106  051105  031440
                042052  034070  034466  041440
                042060  047514  040503  000123
 10918          042066  042532  047822  042040  EM34:   .ASCIZ  /ZERO DETECT BIT NOT HIGH WHEN 32 BIT ECC REG, HAS 21 ZEROS/
                042074  052105  041505  020124
                042102  044502  020124  047516
                042110  020124  044510  044107
                042116  053440  042510  020116
                042124  031063  041040  052111
                042132  042440  041503  051040
                042140  043505  020056  040510
                042146  020123  030462  055040
                042154  051105  051517     000
 10919          042161     120  051517  052111  EM35:   .ASCII  /POSITION REGISTER OR 11 BITS OF PATTERN REGISTER INCORRECT/<15><12>
                042166  047511  020116  042522
                042174  044507  052123  051105
                042202  047440  020122  030461
                042210  041040  052111  020123
                042216  043117  050040  052101
                042224  042524  047122  051040
                042232  043505  051511  042524
                042240  020122  047111  047503
                042246  051122  041505  006524
                042254     012
 10920          042255     114  053517  051105          .ASCII  /LOWER 11 BITS OF PATTERN REGISTER SHOULD MATCH LOWER/<15><12>
                042262  030440  020061  044502
                042270  051524  047440  020106
                042276  040520  052124  051105
                042304  020116  042522  044507
```

```
                042312  052123  051105  051440
                042320  047510  046125  020104
                042326  040515  041524  020110
                042334  047514  042527  006522
                042342     012
 10921          042343     061  020061  044502          ,ASCII  /11 BITS OF GOOD ECC1/<15><12>
                042350  051524  047440  020106
                042356  047507  042117  042440
                042364  041503  006461     012
 10922          042371     104  052101  042440          ,ASCIZ  /DAT ENVLOP GOOD POSITION AND N-CODE ZEROS ARE IN OCTAL/
                042376  053116  047514  020120
                042404  047507  042117  050040
                042412  051517  052111  047511
                042420  020116  047101  020104
                042426  026516  047503  042504
                042434  055040  051105  051517
                042442  040440  042522  044440
                042450  020116  041517  040524
                042456  000114
 10923          042460  047117  051040  040505  EM36:   ,ASCII  /ON READ COMMAND WITH NON-CORRECTABLE ERROR DCK AND ECH SHOULD BE SET/<1
                042466  020104  047503  046515
                042474  047101  020104  044527
                042502  044124  047040  047117
                042510  041455  051117  042522
                042516  052103  041101  042514
                042524  042440  051122  051117
                042532  042040  045503  040440
                042540  042116  042440  044103
                042546  051440  047510  046125
                042554  020104  042502  051440
                042562  052105  000015
 10924          042566  043111  050040  051517          ,ASCIZ  /IF POSITION REGISTER #10040 OR 10041 IT IS GOOD/
                042574  052111  047511  020116
                042602  042522  044507  052123
                042610  051105  036440  030061
                042616  032060  020060  051117
                042624  030440  030060  030464
                042632  044440  020124  051511
                042640  043440  047517  000104
 10925          042646  051127  052111  047111  EM37:   ,ASCIZ  /WRITING WITH BUS ADDRESS HIGHER THAN 28K CAUSED ERROR/
                042654  020107  044527  044124
                042662  041040  051525  040440
                042670  042104  042522  051523
                042676  044040  043511  042510
                042704  020122  044124  047101
                042712  031040  045470  041440
                042720  052501  042523  020104
                042726  051105  047522  000122
 10926          042734  020040  041520  020040  DH1:    ,ASCII  /  PC      TEST    REG.    GOOD  RECEIVED/<15><12>
                042742  020040  020040  042524
                042750  052123  020040  020040
                042756  042522  027107  020040
                042764  020040  047507  042117
                042772  020040  042522  042503
```

```
                043000  053111  042105  005015
10927           043006  020040  020040  020040          ,ASCIZ  /           NO    ADDR,    DATA    DATA  /
                043014  020040  020040  047040
                043022  020117  020040  040440
                043030  042104  027122  020040
                043036  020040  040504  040524
                043044  020040  020040  040504
                043052  040524  020040      000
10928           043057      040  050040  020103  DH21:   .ASCII  /  PC    TEST      WORD    GOOD     BAD  /<15><12>
                043064  020040  052040  051505
                043072  020124  020040  020040
                043100  053440  051117  020104
                043106  020040  043440  047517
                043114  020104  020040  020040
                043122  040502  020104  006440
                043130      012
10929           043131      040  020040  020040          ,ASCIZ  /           NO        NO     DATA    DATA  /
                043136  020040  020040  047516
                043144  020040  020040  020040
                043152  020040  047516  020040
                043160  020040  042040  052101
                043166  020101  020040  042040
                043174  052101  020101  000040
10930           043202  020040  041520  020040  DH11:   .ASCII  /  PC      TEST  FAILING  CONT,   CONT,   CONT,   CONT, /<15><12>
                043210  020040  020040  042524
                043216  052123  020040  040506
                043224  046111  047111  020107
                043232  041440  047117  027124
                043240  020040  041440  047117
                043246  027124  020040  041440
                043254  047117  027124  020040
                043262  041440  047117  027124
                043270  006440      012
10931           043273      040  020040  020040          ,ASCII  /           NO   REG ADR RHCS1   RHCS2   RHDS1   RHER1/
                043300  020040  020040  020040
                043306  047516  020040  051040
                043314  043505  040440  051104
                043322  051040  041510  030523
                043330  020040  051040  041510
                043336  031123  020040  051040
                043344  042110  030523  020040
                043352  051040  042510  030522
                043360      000
10932           043361      040  020040  041520  DH14:   .ASCII  /  PC     TEST  FAILING   CONT,   CONT,   CONT,   CONT,   CONT, /<15><1
                043366  020040  020040  052040
                043374  051505  020124  043040
                043402  044501  044514  043516
                043410  020040  041440  047117
                043416  027124  020040  041440
                043424  047117  027124  020040
                043432  041440  047117  027124
                043440  020040  041440  047117
                043446  027124  020040  041440
                043454  047117  027124  006440
```

```
                043462  012
10933           043463  040  020040  020040          ,ASCIZ  /           NO   REG ADR  BAD REG  RHCS1   RHCS2   RHDS1   RHER1/
                043470  020040  020040  020040
                043476  047516  020040  051040
                043504  043505  040440  051104
                043512  020040  040502  020104
                043520  042522  020107  051040
                043526  041510  030523  020040
                043534  051040  041510  031123
                043542  020040  051040  042110
                043550  030523  020040  051040
                043556  042510  030522  000
10934           043563  040  020040  041520  DH15:  ,ASCIZ  /   PC   REG, ADR,/
                043570  020040  051040  043505
                043576  020056  042101  027122
                043604  000
10935           043605  040  050040  020103  DH16:  ,ASCII  /  PC      WAT    BIT     REG     REG/<15><12>
                043612  020040  020040  053440
                043620  052101  020040  020040
                043626  044502  020124  020040
                043634  020040  042522  020107
                043642  020040  020040  042522
                043650  006507  012
10936           043653  040  020040  020040          ,ASCIZ  /           PC    WANTED  ADDRESS CONTENTS/
                043660  020040  020040  050040
                043666  020103  020040  053440
                043674  047101  042524  020104
                043702  040440  042104  042522
                043710  051523  041440  047117
                043716  042524  052116  000123
10937           043724  020040  050040  020103  DH17:  ,ASCII  /  PC     TEST   CONT.  CONT.  CONT.  CONT.  CONT. /<15><12>
                043732  020040  020040  042524
                043740  052123  020040  020040
                043746  047503  052116  020056
                043754  020040  047503  052116
                043762  020056  020040  047503
                043770  052116  020056  020040
                043776  047503  052116  020056
                044004  020040  047503  052116
                044012  020056  005015
10938           044016  020040  020040  020040          ,ASCIZ  /           NO     RHBA    RHDB    RHMC    RMCS1   RMCS2 /
                044024  020040  020040  047040
                044032  020117  020040  020040
                044040  044122  040502  020040
                044046  020040  044122  041104
                044054  020040  020040  044122
                044062  041527  020040  020040
                044070  044122  051503  020061
                044076  020040  044122  051503
                044104  020062  000
10939           044107  040  020040  041520  DH20:  ,ASCIZ  /  PC   TEST   CONT   CONT   CONT   CONT   CONT/<15><12>
                044114  020040  052040  051505
                044122  020124  020040  041440
                044130  047117  020124  020040
```

```
                044136  041440  047117  020124
                044144  020040  041440  047117
                044152  020124  020040  041440
                044160  047117  020124  020040
                044166  041440  047117  006524
                044174     012
 10942          044175     040  020040  020040          .ASCIZ  /         NO      RHER1   RHER2   RHER3   RHAS    RHDS1/
                044202  020040  020040  047516
                044210  020040  020040  051040
                044216  042510  030522  020040
                044224  051040  042510  031122
                044232  020040  051040  042510
                044240  031522  020040  051040
                044246  040510  020123  020040
                044254  051040  042110  030523
                044262     000
 10941          044263     040  020040  041520  DH21:   .ASCII  /  PC    TEST    CONT    CONT    CONT  /<15><12>
                044270  020040  020040  042524
                044276  052123  020040  020040
                044304  047503  052116  020040
                044312  020040  047503  052116
                044320  020040  020040  047503
                044326  052116  020040  005015
 10942          044334  020040  020040  020040          .ASCIZ  /           NO      RHCS1   RHAS    RHDS1 /
                044342  020040  020040  047040
                044350  020117  020040  020040
                044356  044122  051503  020061
                044364  020040  044122  051501
                044372  020040  051040  042110
                044400  030523  000040
 10943          044404  020040  041520  020040  DH22:   .ASCII  /  PC    TEST    RHAS    RHCS2   /<15><12>
                044412  020040  042524  052123
                044420  020040  020040  044122
                044426  051501  020040  020040
                044434  044122  051503  020062
                044442  020040  005015
 10944          044446  020040  020040  020040          .ASCIZ  /         NO      UNIT    UNIT/
                044454  020040  047516  020040
                044462  020040  020040  047125
                044470  052111  020040  020040
                044476  047125  052111     000
 10945          044503     120  020103  020040  DH24:   .ASCII  /PC       RHDST   BAD     GOOD    SECTOR  SECTOR/<15><12>
                044510  020040  051040  042110
                044516  052123  020040  041040
                044524  042101  020040  020040
                044532  043440  047517  020104
                044540  020040  051440  041505
                044546  047524  020122  051440
                044554  041505  047524  006522
                044562     012
 10946          044563     040  020040  020040          .ASCIZ  /         CONT.   RHLA    RHLA    NO      CLOCK/
                044570  020040  041440  047117
                044576  027124  020040  051040
                044604  046110  020101  020040
```

```
                044612  051040  046110  020101
                044620  020040  047040  020117
                044626  020040  020040  041440
                044634  047514  045503     000
 10947          044641     120  020103  020040  DH26:   .ASCII  /PC      PC OF   FAILING CONT.    CONT.    CONT.    CONT./<15><12>
                044646  020040  050040  020103
                044654  043117  020040  043040
                044662  044501  044514  043516
                044670  041440  047117  027124
                044676  020040  020040  047503
                044704  052116  020056  020040
                044712  041440  047117  027124
                044720  020040  020040  047503
                044726  052116  006456     012
 10948          044733     040  020040  020040          .ASCIZ/       JSR     REG. ADDR RHCS1   RHCS2   RHDS1   RHER1/
                044740  020040  045040  051123
                044746  020040  020040  051040
                044754  043505  020056  042101
                044762  051104  051040  041510
                044770  030523  020040  051040
                044776  041510  031123  020040
                045004  051040  042110  030523
                045012  020040  051040  042510
                045020  030522     000
 10949          045023     120  020103  020040  DH27:   .ASCII  /PC      PC OF   TEST    FAILING GOOD    RECEIVED/<15><12>
                045030  020040  050040  020103
                045036  043117  020040  052040
                045044  051505  020124  020040
                045052  043040  044501  044514
                045060  043516  043440  047517
                045066  020104  020040  051040
                045074  041505  044505  042526
                045102  006504     012
 10950          045105     040  020040  020040          .ASCIZ /      JSR     NO     REGISTERDATA    DATA/
                045112  020040  045040  051123
                045120  020040  020040  020040
                045126  047516  020040  020040
                045134  051040  043505  051511
                045142  042524  042122  052101
                045150  020101  020040  042040
                045156  052101  000101
 10951          045162  041520  020040  020040  DH30:   .ASCII  /PC      PC OF   REGISTER GOOD   BAD/<15><12>
                045170  020040  041520  047440
                045176  020106  020040  042522
                045204  044507  052123  051105
                045212  043440  047517  020104
                045220  020040  040502  006504
                045226     012
 10952          045227     040  020040  020040          .ASCIZ /      JSR     ADDRESS DATA    DATA/
                045234  020040  045040  051123
                045242  020040  020040  040440
                045250  042104  042522  051523
                045256  042040  052101  020101
                045264  020040  040504  040524
```


B15

```
            045272    000
 10953      045273    040   050040  020103  DH31:   .ASCII  /  PC    TEST      WORD    GOOD     BAD  CONT,   CONT,   CONT   /<15><12
            045300  020040  052040  051505
            045306  020124  020040  020040
            045314  053440  051117  020104
            045322  020040  043440  047517
            045330  020104  020040  020040
            045336  040502  020104  041440
            045344  047117  027124  020040
            045352  041440  047117  027124
            045360  020040  041440  047117
            045366  020124  020040  005015
 10954      045374  020040  020040  020040          .ASCIZ  /         NO        NO     DATA    DATA  RHCS1   RHDS1   RHER1   /
            045402  020040  047040  020117
            045410  020040  020040  020040
            045416  047040  020117  020040
            045424  020040  040504  040524
            045432  020040  020040  040504
            045440  040524  020040  044122
            045446  051503  020061  020040
            045454  044122  051504  020061
            045462  020040  044122  051105
            045470  020061  020040    000
 10955      045475    040   050040  020103  DH32:   .ASCII  /  PC    PC OF   TEST      WORD    GOOD     BAD  CONT,   CONT,   CONT,
            045502  020040  050040  020103
            045510  043117  020040  052040
            045516  051505  020124  020040
            045524  020040  053440  051117
            045532  020104  020040  043440
            045540  047517  020104  020040
            045546  020040  040502  020104
            045554  041440  047117  027124
            045562  020040  041440  047117
            045570  027124  020040  041440
            045576  047117  027124  020040
            045604  006440    012
 10956      045607    040   020040  020040          .ASCIZ  /         JSR       NO        NO     DATA    DATA  RHCS1   RHDS1   RHER1
            045614  020040  020040  051512
            045622  020122  020040  020040
            045630  047516  020040  020040
            045636  020040  020040  047516
            045644  020040  020040  042040
            045652  052101  020101  020040
            045660  042040  052101  020101
            045666  051040  041510  030523
            045674  020040  051040  042110
            045702  030523  020040  051040
            045710  042510  030522  020040
            045716  000040
 10957      045720  020040  041520  020040  DH33:   .ASCII  /  PC    PC OF   TEST      WORD    GOOD     CONT,   CONT,   CONT,   /<15
            045726  020040  041520  047440
            045734  020106  020040  042524
            045742  052123  020040  020040
            045750  020040  047527  042122
```

```
        045756  020040  020040  047507
        045764  042117  020040  020040
        045772  041440  047117  027124
        046000  020040  041440  047117
        046006  027124  020040  041440
        046014  047117  027124  020040
        046022  006440    012
10958   046025    040   020040  020040          .ASCIZ  /       JSR     NO      NO      DATA    RHC81   RHD81   RHER1   /
        046032  020040  020040  051512
        046040  020122  020040  020040
        046046  047516  020040  020040
        046054  020040  020040  047516
        046062  020040  020040  042040
        046070  052101  020101  020040
        046076  020040  044122  051503
        046104  020061  020040  044122
        046112  051504  020061  020040
        046120  044122  051105  020061
        046126  020040    000
10959   046131    120   020103  020040  DH341   .ASCII  /PC      TEST    GOOD    GOOD    WRITTEN WRITTEN DATA/<15><12>
        046136  020040  052040  051505
        046144  020124  020040  043440
        046152  047517  020104  020040
        046160  043440  047517  020104
        046166  020040  053440  044522
        046174  052124  047105  053440
        046202  044522  052124  047105
        046210  042040  052101  006501
        046216    012
10960   046217    040   020040  020040          .ASCIZ  /       NO      ECC1    ECC2    ECC1    ECC2    USED/
        046224  020040  020040  047516
        046232  020040  020040  042440
        046240  041503  020061  020040
        046246  042440  041503  020062
        046254  020040  042440  041503
        046262  020061  020040  042440
        046270  041503  020062  020040
        046276  052440  042523  000104
10961   046304  041520  020040  020040  DH351   .ASCII  /PC      TEST    GOOD    GOOD    PATTERN POSITON GOOD    RHER1/<15><12>
        046312  020040  042524  052123
        046320  020040  020040  047507
        046326  042117  020040  020040
        046334  047507  042117  020040
        046342  020040  040520  052124
        046350  051105  020116  047520
        046356  044523  047524  020116
        046364  047507  042117  020040
        046372  020040  044122  051105
        046400  006461    012
10962   046403    040   020040  020040          .ASCIZ  /       NO      ECC1    ECC2    REG.    REG.    POSITON REG./
        046410  020040  020040  047516
        046416  020040  020040  042440
        046424  041503  020061  020040
        046432  042440  041503  020062
```

```
                046440  020040  051040  043505
                046446  020056  020040  051040
                046454  043505  020056  020040
                046462  050040  051517  052111
                046470  047117  051040  043505
                046476  000056
 10963          046500  041520  020040  020040  DH36:   .ASCII  /PC      PC OF   TEST    RHMR    POSITON PATTERN/<15><12>
                046506  020040  041520  047440
                046514  020106  020040  042524
                046522  052123  020040  020040
                046530  044122  051115  020040
                046536  020040  047520  044523
                046544  047524  020116  040520
                046552  052124  051105  000516
                046560     012
 10964          046561     040  020040  020040          .ASCIZ  /        JSR     NO      CONT.   REG.    REG./
                046566  020040  020040  051512
                046574  020122  020040  020040
                046602  047516  020040  020040
                046610  041440  047117  027124
                046616  020040  051040  043505
                046624  020056  020040  051040
                046632  043505  000056
 10965          046636  041520  020040  020040  DH37:   .ASCII  /PC      TEST    POSITON POSITON GOOD    GOOD    PATTERN DATA    N-CODE/
                046644  020040  042524  052123
                046652  020040  020040  047520
                046660  044523  047524  020116
                046666  047520  044523  047524
                046674  020116  047507  042117
                046702  020040  020040  047507
                046710  042117  020040  020040
                046716  040520  052124  051105
                046724  020116  040504  040524
                046732  020040  020040  026516
                046740  047503  042504  005015
 10966          046746  020040  020040  020040          .ASCIZ  /        NO      ECC     GOOD    ECC1    ECC2    ECC     ENVLOPE ERROR/
                046754  020040  047040  020117
                046762  020040  020040  042440
                046770  041503  020040  020040
                046776  043440  047517  020104
                047004  020040  042440  041503
                047012  020061  020040  042440
                047020  041503  020062  020040
                047026  020040  041505  020103
                047034  020040  047105  046126
                047042  050117  020105  042532
                047050  047522  000123
 10967                                          .EVEN
 10968          047054  001116  004174  024234  DT1:    .WORD   $ERRPC,TSTNM,REGADR,$GDDAT,$BDDAT,0
                047062  001124  001126  000000
 10969          047070  001116  004174  031136  DT2:    .WORD   $ERRPC,TSTNM,ERWORD,$GDDAT,0
                047076  001124  000000
 10970          047102  001116  004174  031136  DT3:    .WORD   $ERRPC,TSTNM,ERWORD,$GDDAT,$BDDAT,0
                047110  001124  001126  000000
```

```
 10971  047116  001116  004174  001122  DT11:   .WORD   $ERRPC,TSTNM,$BDADR,CS1,CS2,DS1,ER1,0
        047124  001702  001700  001724
        047132  001704  000000
 10972  047136  001116  004174  001122  DT14:   .WORD   $ERRPC,TSTNM,$BDADR,$BDDAT,CS1,CS2,DS1,ER1,0
        047144  001126  001702  001700
        047152  001724  001704  000000
 10973  047160  001116  001172  000000  DT15:   .WORD   $ERRPC,$TMP1,0
 10974  047166  001116  001176  001172  DT16:   .WORD   $ERRPC,$TMP3,$TMP1,$TMP0,$BDDAT,0
        047174  001170  001126  000000
 10975  047202  001116  004174  001676  DT17:   .WORD   $ERRPC,TSTNM,BA,DB,WC,CS1,CS2,0
        047210  001672  001674  001702
        047216  001700  000000
 10976  047222  001116  004174  001704  DT20:   .WORD   $ERRPC,TSTNM,ER1,ER2,ER3,AS,DS1,0
        047230  001710  001716  001720
        047236  001724  000000
 10977  047242  001116  004174  001702  DT21:   .WORD   $ERRPC,TSTNM,CS1,AS,DS1,0
        047250  001720  001724  000000
 10978  047256  001116  004174  001124  DT22:   .WORD   $ERRPC,TSTNM,$GDDAT,$BDDAT,0
        047264  001126  000000
 10979  047270  001116  001706  001126  DT24:   .WORD   $ERRPC,DST,$BDDAT,$TMP1,$TMP2,$TMP3,0
        047276  001172  001174  001176
        047304  000000
 10980  047306  001116  002002  001122  DT26:   .WORD   $ERRPC,PCJSR,$BDADR,CS1,CS2,DS1,ER1,0
        047314  001702  001700  001724
        047322  001704  000000
 10981  047326  001116  002002  004174  DT27:   .WORD   $ERRPC,PCJSR,TSTNM,REGADR,$GDDAT,$BDDAT,0
        047334  024234  001124  001126
        047342  000000
 10982  047344  001116  002002  024234  DT30:   .WORD   $ERRPC,PCJSR,REGADR,$GDDAT,$BDDAT,0
        047352  001124  001126  000000
 10983
 10984  047360  001116  004174  031136  DT31:   .WORD   $ERRPC,TSTNM,ERWORD,$GDDAT,$BDDAT,CS1,DS1,ER1,0
        047366  001124  001126  001702
        047374  001724  001704  000000
 10985  047402  001116  002002  004174  DT32:   .WORD   $ERRPC,PCJSR,TSTNM,ERWORD,$GDDAT,$BDDAT,CS1,DS1,ER1,0
        047410  031136  001124  001126
        047416  001702  001724  001704
        047424  000000
 10986  047426  001116  002002  004174  DT33:   .WORD   $ERRPC,PCJSR,TSTNM,ERWORD,$GDDAT,CS1,DS1,ER1,0
        047434  031136  001124  001702
        047442  001724  001704  000000
 10987  047450  001116  004174  027126  DT34:   .WORD   $ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK,0
        047456  027130  033734  033736
        047464  032734  000000
 10988  047470  001116  004174  027126  DT35:   .WORD   $ERRPC,TSTNM,GECC1,GECC2,EC2,EC1,POSITI,ER1,0
        047476  027130  001734  001732
        047504  027140  001704  000000
 10989  047512  001116  002002  004174  DT36:   .WORD   $ERRPC,PCJSR,TSTNM,MR,EC1,EC2,0
        047520  001722  001732  001734
        047526  000000
 10990  047530  001116  004174  001732  DT37:   .WORD   $ERRPC,TSTNM,EC1,POSITI,GECC1,GECC2,EC2,DATENV,ZCODE,0
        047536  027140  027126  027130
        047544  001734  027144  027146
        047552  000000
```

```
10991  047554  000  000  000  DF1:   .BYTE  0,0,0,0,0
       047557  000  000
10992  047561  000  000  001  DF2:   .BYTE  0,0,1,0
       047564  000
10993  047565  000  000  001  DF3:   .BYTE  0,0,1,0,0
       047570  000  000
10994  047572  000  000  000  DF11:  .BYTE  0,0,0,0,0,0,0
       047575  000  000  000
       047600  000
10995  047601  000  000  000  DF14:  .BYTE  0,0,0,0,0,0,0,0
       047604  000  000  000
       047607  000  000
10996  047611  000  000       DF15:  .BYTE  0,0
10997  047613  000  000  000  DF16:  .BYTE  0,0,0,0
       047616  000
10998  047617  000  000  000  DF17:  .BYTE  0,0,0,0,0,0,0
       047622  000  000  000
       047625  000
10999  047626  000  000  000  DF20:  .BYTE  0,0,0,0,0,0,0
       047631  000  000  000
       047634  000
11000  047635  000  000  000  DF21:  .BYTE  0,0,0,0,0
       047640  000  000
11001  047642  000  000  000  DF22:  .BYTE  0,0,0,0
       047645  000
11002  047646  000  000  000  DF24:  .BYTE  0,0,0,0,0,0
       047651  000  000  000
11003  047654  000  000  000  DF26:  .BYTE  0,0,0,0,0,0,0
       047657  000  000  000
       047662  000
11004  047663  000  000  000  DF27:  .BYTE  0,0,0,0,0,0
       047666  000  000  000
11005  047671  000  000  000  DF30:  .BYTE  0,0,0,0,0
       047674  000  000
11006  047676  000  000  001  DF31:  .BYTE  0,0,1,0,0,0,0,0
       047701  000  000  000
       047704  000  000
11007  047706  000  000  000  DF32:  .BYTE  0,0,0,1,0,0,0,0,0
       047711  001  000  000
       047714  000  000  000
11008  047717  000  000  000  DF33:  .BYTE  0,0,0,1,0,0,0,0
       047722  001  000  000
       047725  000  000
11009  047727  000  000  000  DF34:  .BYTE  0,0,0,0,0,0,0
       047732  000  000  000
       047735  000
11010  047736  000  000  000  DF35:  .BYTE  0,0,0,0,0,0,0,0
       047741  000  000  000
       047744  000  000
11011  047746  000  000  000  DF36:  .BYTE  0,0,0,0,0,0
       047751  000  000  000
11012  047754  000  000  000  DF37:  .BYTE  0,0,0,0,0,0,0,0,0
       047757  000  000  000
       047762  000  000  000
```

```
 11013           047766                          .EVEN
 11014
 11015           000001                  .END
```

DERPTA.P11 SYMBOL TABLE

```
ACL    = 000040        ACU    = 100000        AOE    = 001000        AS      001720
ATA    = 100000        ATABLE  004200        ATTENT  002004        AT0    = 000001
AT1    = 000002        AT2    = 000004        AT3    = 000010        AT4    = 000020
AT5    = 000040        AT6    = 000100        AT7    = 000200        A16    = 000400
A17    = 001000        BA      001676        BAI    = 000010        BASECH  030056
BEGIN   004220         BEGIN2  004210        BITBT   024236        BIT0   = 000001
BIT00  = 000001        BIT01  = 000002        BIT02  = 000004        BIT03  = 000010
BIT04  = 000020        BIT05  = 000040        BIT06  = 000100        BIT07  = 000200
BIT08  = 000400        BIT09  = 001000        BIT1   = 000002        BIT10  = 002000
BIT11  = 004000        BIT12  = 010000        BIT13  = 020000        BIT14  = 040000
BIT15  = 100000        BIT2   = 000004        BIT3   = 000010        BIT4   = 000020
BIT5   = 000040        BIT6   = 000100        BIT7   = 000200        BIT8   = 000400
BIT9   = 001000        BLT1    024264         BLT2    024306         BLT3    024316
BPTVEC= 000014         CA      001714         CC      001740         CHECKD= 104422
CHECKE  024706         CHECKT  024524         CLAREA  024406         CLDISK  024470
CLR    = 000040        COMHD   030706         COMPA   031730         COMPAR  025342
COMWHD  033776         COUNTD  032376         CRC     025646         CRLF   = 000200
CSF    = 000002        CSU    = 000010        CS1     001702         CS2     001700
CYL     031016         DAREAD  031072         DATENV  027144         DAWORD  031076
DB      001672         DCK    = 100000        DCL    = 000100        DCLEAR  002026
DENVL  = 000200        DE1    = 000040        DFF20  = 000002        DF1     047554
DF11    047572         DF14    047601         DF15    047611         DF16    047613
DF17    047617         DF2     047561         DF20    047626         DF21    047635
DF22    047642         DF24    047646         DF26    047654         DF27    047663
DF3     047565         DF30    047671         DF31    047676         DF32    047706
DF33    047717         DF34    047727         DF35    047736         DF36    047746
DF37    047754         DF5    = 000001        DH1     042734         DH11    043202
DH14    043361         DH15    043563         DH16    043605         DH17    043724
DH2     043057         DH20    044107         DH21    044263         DH22    044404
DH24    044503         DH26    044641         DH27    045023         DH30    045162
DH31    045273         DH32    045475         DH33    045720         DH34    046131
DH35    046304         DH36    046500         DH37    046636         DIGB   = 000004
DISK    032734         DISPLA= 177570         DLT    = 100000        DL64   = 000020
DMD    = 000001        DPR    = 000400        DRY    = 000200        DST     001706
DS1     001724         DT      001726         DTAGAP  033740        DTE    = 010000
DTSY   = 001000        DT1     047054         DT11    047116         DT14    047136
DT15    047160         DT16    047166         DT17    047202         DT2     047070
DT20    047222         DT21    047242         DT22    047256         DT24    047270
DT26    047306         DT27    047326         DT3     047102         DT30    047344
DT31    047360         DT32    047402         DT33    047426         DT34    047450
DT35    047470         DT36    047512         DT37    047530         DVA    = 004000
ECDATA  027124         ECH    = 000100        ECI    = 004000        ECORR   027550
ECTEST  027162         EC1     001732         EC2     001734         EMTVEC= 000030
EM1     040130         EM11    040301         EM14    040334         EM15    040354
EM16    040425         EM17    040446         EM2     040213         EM20    040472
EM21    040513         EM22    040535         EM24    041130         EM25    041223
EM30    041263         EM31    041405         EM32    041530         EM33    042014
EM34    042066         EM35    042161         EM36    042460         EM37    042646
EM6     040242         ERCRC   034142         ERFLGS  001774         ERHDGP  034144
ERHEAD  034140         ERPOS   027546         ERR    = 040000        ERRVEC= 000004
ERSTAR  030662         ERUNIT  030660         ERWORD  031136         ER1     001704
ER2     001710         ER3     001716         EXT1   = 000001        EXT10  = 000010
EXT2   = 000002        EXT20  = 000020        EXT4   = 000004        EXT40  = 000040
FEN    = 000200        FER    = 000020        FILLEC  027722         FIRST   004176
```

```
FMT22 = 010000          FNWORD  034162          FORMAT  032400          FOUT    034166
FSYNER  034136          FUTABL  002020          GCRC    034132          GECC1   027126
GECC2   027130          GO    = 000001          GRV   = 000010          HADTMP  027150
HARDER  027142          HCCRCE  026250          HCE   = 000200          HCI   = 002000
HCRC  = 000400          HDESYN  034146          HDWSYN  032732          HEADER  032706
HEDGAP  031032          HEDSYN  031034          HEGAP   032720          HT    = 000011
IAE   = 002000          IE    = 000100          ILF   = 000001          ILLEGL  002062
ILR   = 000002          IOTVEC= 000020          IR    = 000100          IXE   = 004000
KEY1    031022          KEY2    031024          KIPAR0= 172340          KIPAR1= 172342
KIPAR2= 172344          KIPAR3= 172346          KIPAR4= 172350          KIPAR5= 172352
KIPAR6= 172354          KIPAR7= 172356          KIPDR0= 172300          KIPDR1= 172302
KIPDR2= 172304          KIPDR3= 172306          KIPDR4= 172310          KIPDR5= 172312
KIPDR6= 172314          KIPDR7= 172316          LA      001736          LAD     024450
LERR    024232          LBT   = 002000          MAKECY  027004          MASK    024230
MCLK  = 000002          MCPE  = 020000          MHS   = 001000          MID     026774
MIDDLE  026672          MINX  = 000004          MMVEC = 000250          MOL   = 010000
MPE   = 000400          MR      001722          MRD   = 000020          MSE   = 000020
MSTCK = 000010          MWR   = 000040          MXF   = 001000          NCODE   027134
NCOUNT  027136          NED   = 010000          NEM   = 004000          NHS   = 002000
NOPERA  002020          NOSYNC  031132          NOUNIT  001764          NOWORD  031064
NUNIT   001766          OCYL  = 100000          OF      001712          OFREV = 000200
OFSETC  002052          OF100 = 000004          OF200 = 000010          OF25  = 000001
OF400 = 000020          OF50  = 000002          OF000 = 000040          OPERSE  023410
OPI   = 020000          OR    = 000200          OUT     031102          PAR   = 000010
PAT   = 000020          PC    =%000007          PCJSR   002002          PGE   = 002000
PIE1  = 100000          PIE10 = 000100          PIE11 = 000040          PIE12 = 000020
PIE13 = 000010          PIE14 = 000004          PIE15 = 000002          PIE16 = 000001
PIE17 = 100000          PIE18 = 040000          PIE19 = 020000          PIE2  = 040000
PIE20 = 010000          PIE21 = 004000          PIE22 = 002000          PIE23 = 001000
PIE24 = 000400          PIE25 = 000200          PIE26 = 000100          PIE27 = 000040
PIE28 = 000020          PIE29 = 000010          PIE3  = 020000          PIE30 = 000004
PIE31 = 000002          PIE32 = 000001          PIE4  = 010000          PIE5  = 004000
PIE6  = 002000          PIE7  = 001000          PIE8  = 000400          PIE9  = 000200
PIP   = 020000          PIRQ  = 177772          PIRQVE= 000240          PKACK   002056
PLU   = 020000          POSITI  027140          PRE   = 000020          PROG  = 001000
PR0   = 000000          PR1   = 000040          PR2   = 000100          PR3   = 000140
PR4   = 000200          PR5   = 000240          PR6   = 000300          PR7   = 000340
PS    = 177776          PSEL  = 002000          PSU   = 000001          PSW   = 177776
PUTREG  024170          PWRVEC= 000024          P12     027154          P22     027156
P24     027160          P3      027152          RCOM    035304          RCYL    031122
RDCHR = 104412          RDHEAD  031140          RDLIN = 104414          RDOCT = 104416
RDY   = 000200          READ    031732          READAT  002044          READIN  002060
RECALI  002024          REDATA  035306          REFOR   002046          REGADR  024234
REGSA1  037160          REINTO  003130          RELEAS  002030          RESVEC= 000010
RETCL   002054          RHAS    001650          RHBA    001626          RHCA    001644
RHCC    001670          RHCS1   001632          RHCS2   001630          RHDB    001622
RHDST   001636          RHDS1   001654          RHDT    001656          RHEC1   001662
RHEC2   001664          RHER1   001634          RHER2   001640          RHER3   001646
RHLA    001666          RHMR    001652          RHOF    001642          RHSN    001660
RH=C    001624          RKEY1   031126          RKEY2   031130          RMR   = 000004
RNO     035302          RPTRP1  022702          RPTRP2  023006          RPVEC   001620
RPVECT  035416          RSETR   031124          RSYNC   031120          R0    =%000000
R1    =%000001          R2    =%000002          R3    =%000003          R4    =%000004
R5    =%000005          R6    =%000006          R7    =%000007          SAVDT   001776
```

```
$AVER   025146          $AV$N   002000          $C      = 100000          $COP1 = 104420
$CRC    034214          $CYL    034204          $C1     = 000100          $C10  = 001000
$C2     = 000200        $C20    = 002000        $C4     = 000400          $EARCH  035032
$ECGAP  032636          $ECOTR  031020          $ECTR   035030          $EECOM  002050
$EGPER  034134          $ELECT  001770          $ERCH   002032          $ETD$K  026122
$ILOT$  032734          $KEY1   034210          $KEY2   034212          $KI   = 040000
$N      001730          $ND1    005232          $P      =%000006          $R0   = 177572
$R1     = 177574        $R2     = 177576        $R3     = 172516          $$ECTR  034206
$$YN    031030          $$1     012066          $$10    012240          $$12    012270
$$13    012276          $$14    012304          $$15    012344          $$2     012514
$$3     012120          $$4     012124          $$5     012130          $$7     012156
$TACK   = 001000        $TART   004224          $TKLMT= 177774          $WR   = 177570
$W0     = 000001        $W00    = 000001        $W01    = 000002          $W02    = 000004
$W03    = 000010        $W04    = 000020        $W05    = 000040          $W06    = 000100
$W07    = 000200        $W08    = 000400        $W09    = 001000          $W1     = 000002
$W10    = 002000        $W11    = 004000        $W12    = 010000          $W13    = 020000
$W14    = 040000        $W15    = 100000        $W2     = 000004          $W3     = 000010
$W4     = 000020        $W5     = 000040        $W6     = 000100          $W7     = 000200
$W8     = 000400        $W9     = 001000        TAGDTE  002016          TBITVE= 000014
TDF     = 000040        TESDTE  002014          TESTAD  023406          TIMCNT  025024
TKVEC   = 000060        TMPILL  002010          TOLGAP  033742          TOTALA  002006
TPVEC   = 000064        TRAPVE= 000034          TRE     = 040000          TRK     030770
TRK1    = 004000        TRK10   = 040000        TRK2    = 010000          TRK20   = 100000
TRK4    = 020000        TRTVEC= 000014          TSECC   002012          TSECCG  027132
TSTNM   004174          TST1    005340          TST10   010000          TST11   010026
TST12   010344          TST13   010774          TST14   011416          TST15   011770
TST16   012530          TST17   013142          TST2    005614          TST20   013644
TST21   014354          TST22   015102          TST23   015512          TST24   016214
TST25   016724          TST26   017452          TST27   020062          TST3    005660
TST30   020564          TST31   021304          TST32   022046          TST33   022610
TST34   022722          TST35   023016          TST4    005732          TST5    007036
TST6    007274          TST7    007636          TTNO    = 000034          TUF     = 000100
TY      031134          TYPD$   = 104410        TYPE    = 104400          TYPOC   = 104402
TYPON   = 104406        TYPO$   = 104404        T.$COP  024452          UN      005650
UNIT    001762          UNIT$   001742          UNIT$L  001772          UNLOAD  002022
UNS     = 040000        UPE     = 020000        US1     = 000001          US2     = 000002
US4     = 000004        UWR     = 000010        VUF     = 000002          VU30    = 010000
VV      = 000100        WAIT.T  025026          WAT     = 104424          WC      001674
WCE     = 040000        WCF     = 000040        WCRC    032716          WCU     = 000001
WCYL    034122          WECC1   033734          WECC2   033736          WKEY1   034126
WKEY2   034130          WLE     = 004000        WORD    031726          WRCHDA  025504
WRCHDT  002036          WRCHEK  002034          WRCHHD  025200          WRDATA  032404
WRFROM  002064          WRHEAD  034216          WRIDAT  002040          WRIFOR  002042
WRITE   032164          WRL     = 004000        WRU     = 000400          W$ECTR  034124
W$SYNC  032704          W$U     = 000004        WTRK    034060          WWORD   032162
X       031026          XE2     006552          Y       031066          ZCODE   027146
ZER     = 000400        ZWORD$  032402          $BDADR  001122          $BDDAT  001126
$BELL   001210          $CHARC  036376          $CMTAG  001100          $CM1    = 000006
$CM2    = 000014        $CM3    = 000006        $CM4    = 000006          $CNTLC  037005
$CRLF   001215          $DBLK   036162          $DOAGN  023362          $DTBL   036152
$ENDAD  023352          $ENDCT  023302          $ENDMG  023366          $ENULL  023403
$EOP    023246          $EOPCT  023274          $ERFLG  001103          $ERMAX  001115
$ERROR  037152          $ERRPC  001116          $ERRTB  001220          $ERRTY  037314
$ERTTL  001112          $E$CAP  001206          $FILLC  001150          $FILL$  001147
```

K15

DERPTA.P11 SYMBOL TABLE

```
$GDADR  001120          $GDDAT  001124          $GET42  023324          $HD    = 000000
$HIOCT  037150          $ICNT   001104          $ILLUP  040112          $ITEMB  001114
$LF     001216          $LPADR  001106          $LPERR  001110          $MXCNT  035744
$NULL   001146          $NWTBT= 000001          $OCNT   037712          $OMODE  037714
$OVER   035730          $PASS   001100          $POWER  040120          $PWRAD  040106
$PWRDN  037764          $PWRMG  040102          $PWRUP  040032          $QUES   001214
$RDCHR  036602          $RDLIN  036666          $RDOCT  037012          $RDSZ  = 000011
$REGAD  001152          $REG0   001154          $REG1   001156          $REG2   001160
$REG3   001162          $REG4   001164          $REG5   001166          $RESET  023350
$SAVR6  040116          $SCOPE  035460          $SETUP= 000017          $SS1   = 000000
$STUP  = 177777         $SVLAD  035702          $SWR   = 167700         $SWRMK= 000000
$TIMES  001204          $TKB    001140          $TKCNT  036404          $TKINT  036424
$TKQEN= 036420          $TKQIN  036406          $TKQOU  036410          $TKQSR  036412
$TKS    001130          $TKSRV  036474          $TMP0   001170          $TMP1   001172
$TMP2   001174          $TMP3   001176          $TMP4   001200          $TMP5   001202
$TN    = 000036         $TPB    001144          $TPFLG  001151          $TPS    001142
$TRAP   037716          $TRP   = 000026          $TRPAD  037736          $TSTNM  001102
$TTYIN  036770          $TYPDS  035746          $TYPE   036172          $TYPEC  036332
$TYPEX  036400          $TYPOC  037514          $TYPON  037530          $TYPOS  037470
$XTSTR  035506          $$TRP  = 000002          $$FILL  037713          .      = 047766
```

ERRORS DETECTED: 0

```
*DERPTA,DERPTA=DERPTA,SML,DERPTA.P11
RUN-TIME:  72 81 1 SECONDS
CORE USED:  29K
```