RP04
RP05, RP06

RP04/5/6 DSKLS 2
CZRJHE0

AH-9215E-MC
FICHE 1 OF 1

```
                        .REM    @
```

IDENTIFICATION

PRODUCT CODE: AC-9213E-MC

PRODUCT NAME: CZRJHE0 RP04/5/6 DISKLESS TEST, PT 2

PRODUCT DATE: NOVEMBER 1981

MAINTAINER: CX DIAGNOSTIC GROUP

AUTHOR: PETE BLACKSTONE

REVISED BY: MIKE LEAVITT

CONTENTS

1.0 ABSTRACT

THIS DIAGNOSTIC TESTS THE RH11 AND DCL OF AN RP04/5/6 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.

THE DIAGNOSTIC DOES NOT DO ANY TESTING OF THE RH70 CONTROLLER
WHEN IT IS USED TO TEST RP04/5/6 DISK DRIVES CONNECTED TO
THAT TYPE OF CONTROLLER. IT IS ASSUMED THAT THE RH70
SPECIFIC CONTROLLER DIAGNOSTICS HAVE BEEN SUCCESSFULLY RUN TO
COMPLETION BEFORE THIS PROGRAM IS RUN.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 COMPUTER WITH CONSOLE TELETYPE, AND AN RP04/5/6 DISK SYSTEM.
THE RP04/5/6 DISK SYSTEM WILL CONSIST OF AN RH11/RH70 CONTROLLER,
AND 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-DZRJG-
(LATEST REV) HAS BEEN RUN WITHOUT ERRORS

3.0 LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING .ABS TAPES

4.0 STARTING PROCEDURE

SWITCH 12 MUST BE SET WHEN THIS PROGRAM IS TO BE RUN USING AN
RH70 CONTROLLER. IT CAN BE SET AT THE FRONT PANEL, OR IN THE
SOFWARE SWITCH REGISTER IF THE OPERATOR SO DESIRES. SEE
PARAGRAPH 5.1 FOR A DESCRIPTION OF SOFTWARE SWITCH REGISTER
OPERATION.

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1

4.2 STARTING ADDRESS

```
                        START AT ADDRESS 200---FOR NORMAL RUN
                        START AT ADDRESS 204---TO SELECT NON-DEFAULT PARAMETERS
                        START AT ADDRESS 210---FOR UNIT SELECTION

                        200 START
                        ALL SWITCHES MUST BE DOWN FOR WORST CASE RUN.  WITH THIS
                        STARTING ADDRESS ALL THE RP04/5/6'S 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.

                        204 RESTART
                        SAME AS 200 START WITH FOLLOWING EXCEPTIONS:  PROGRAM WILL
                        QUERY OPERATOR FOR THE CORRECT CSR AND VECTOR ADDRESS OF THE RHXX
                        CONTROLLER.  WHEN THIS ACTION HAS BEEN COMPLETED, THE PROGRAM
                        WILL AUTOMATICALLY RESTART FROM ADDRESS 200, WITH THE SAME
                        CONVENTIONS AS DESCRIBED FOR A 200 START.

                        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

                IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E.
                AN 11/34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH
                REGISTER IS NOT PRESENT AND WILL USE AN 'SOFTWARE' SWITCH
                REGISTER.  THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE
```

```
                        CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY
                        TYPING A 'CONTROL G'.   THE PROGRAM WILL RECOGNIZE THE
                        'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A
                        HIGHER PRIORITY PROCESSING AN RP04/5/6 INTERRUPT.  THE
                        'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN
                        RESPONSE TIO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

                                'SWR = NNNNNN   NEW ='

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

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



                        SWITCH DEFINITIONS ARE GIVEN IN SECTION 9 "OPERATIONAL
                        SWITCH SETTINGS" HOWEVER THE DETAIL DESCRIPTIONS 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 12 - RH70 CONTROLLER SELECT
                        THIS SWITCH MUST BE SET AT THE START OF THE PROGRAM WHEN THE
                        DISK DRIVES TO BE TESTED ARE CONNECTED TO AN RH70
                        CONTROLLER.  IT MUST NOT BE SET WHEN DISK DRIVES TO BE TESTED
                        ARE CONNECTED TO AN RH11 CONTROLLER.

                        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 8 - 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
                                7 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 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 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.

6.1 'FATAL' ERRORS

IN THE EVENT THAT THE DISK DRIVE BECOMES UNAVAILABLE TO THE
CONTROLLER, POWERS DOWN, OR CERTAIN CRITICAL STATUS BITS
CANNOT BE CLEARED PRIOR TO THE START OF A T4ST SEQUENCE -
THIS INFORMATION WILL BE COMMUNICATED TO THE OPERATOR. IN
ADDITION, THE TTY BELL WILL RING AND THE PROGRAM WILL HALT.
IT IS SUGGESTED THAT IF THIS HAPPENS THE OPERATOR LOAD

ADDRESS 200 (210) AND RESTART THE PROGRAM AS A FIRST ATTEMPT
TO SOLVE THE PROBLEM. IF THE FAILURE CONTINUES TO OCCUR,
THERE ARE TWO OPTIONS FOR THE OPERATOR:

1. LOOK IN THE TEST LISTING FOR THE 'HALT'
INSTRUCTION AND REPLACE IT PLUS THE TWO WORDS
(''TYPE ,CPHALT'') ABOVE WITH 'NOP'S. WITH TTY ERROR
PRINTOUTS INHIBITED, A SCOPE LOOP CAN BE INITIATED
FOR THE TEST IN QUESTION.

2. GO BACK AND RERUN DZRPS AS IT IS QUITE POSSIBLE
THAT A HARD FAILURE HAS OCCURRED IN ONE OF THE
HARDWARE REGISTERS.

IT IS ALSO POSSIBLE TO CONTINUE FROM THE HALT POINT, BUT THIS
IS NOT RECOMMENDED AS ALL FOLLOWING TESTS WILL EXHIBIT THE
SAME SYMPTOMS AND GIVE MISLEADING ERROR PRINTOUTS.

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.

SWITCH 12 MUST BE SET WHEN RUNNING ON AN RH70 CONTROLLER AND
IT MUST NOT BE SET WHEN RUNNING ON AN RH11 CONTROLLER.
BECAUSE OF THIS FACT, THE PROGRAM CANNOT BE RUN IN CHAIN MODE
WHEN USING THE SOFTWARE SWITCH REGISTER AS THE ROUTINE WHICH
ASKS FOR THE SWTICH REGISTER SETTINGS IS NOT OPERABLE WHEN IN
CHAIN MODE.

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.
FOR INSTRUCTIONS REGARDING USAGE OF THESE LOOPS, HIT CONTROL
C ANY TIME WHILE THE PROGRAM IS RUNNING. 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.

                        8.4     PROGRAM REVISION HISTORY


                        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.
                        @
```

```
                                ;*LAST REVISION 02-NOV-81
                                .TITLE  CZRJHE0 RP04/5/6 DSKLS PT2
                                ;*COPYRIGHT (C) 1976,1978,1981
                                ;*DIGITAL EQUIPMENT CORPORATION
                                ;*COLORADO SPGS., CO. 80919
                                ;*
                                ;*PROGRAM BY PETE BLACKSTONE
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C5), 18-MAR-81
                                ;*
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        12            RH70 CONTROLLER SELECT
                                ;*        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
                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK   = 1100
104000                          ERROR   = EMT           ;;BASIC DEFINITION OF ERROR CALL
000004                          SCOPE   = IOT           ;;BASIC DEFINITION OF SCOPE CALL

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

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

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

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

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15   = 100000
040000                  BIT14   = 40000
020000                  BIT13   = 20000
010000                  BIT12   = 10000
004000                  BIT11   = 4000
002000                  BIT10   = 2000
001000                  BIT09   = 1000
000400                  BIT08   = 400
000200                  BIT07   = 200
000100                  BIT06   = 100
000040                  BIT05   = 40
000020                  BIT04   = 20
000010                  BIT03   = 10
000004                  BIT02   = 4
000002                  BIT01   = 2
000001                  BIT00   = 1
001000                  BIT9=BIT09
```

```
                000400          BIT8=BIT08
                000200          BIT7=BIT07
                000100          BIT6=BIT06
                000040          BIT5=BIT05
                000020          BIT4=BIT04
                000010          BIT3=BIT03
                000004          BIT2=BIT02
                000002          BIT1=BIT01
                000001          BIT0=BIT00

                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                000004          ERRVEC  = 4             ;;TIME OUT AND OTHER ERRORS
                000010          RESVEC  = 10            ;;RESERVED AND ILLEGAL INSTRUCTIONS
                000014          TBITVEC = 14            ;;"T" BIT
                000014          TRTVEC  = 14            ;;TRACE TRAP
                000014          BPTVEC  = 14            ;;BREAKPOINT TRAP (BPT)
                000020          IOTVEC  = 20            ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                000024          PWRVEC  = 24            ;;POWER FAIL
                000030          EMTVEC  = 30            ;;EMULATOR TRAP (EMT) **ERROR**
                000034          TRAPVEC = 34            ;;"TRAP" TRAP
                000060          TKVEC   = 60            ;;TTY KEYBOARD VECTOR
                000064          TPVEC   = 64            ;;TTY PRINTER VECTOR
                000240          PIRQVEC = 240           ;;PROGRAM INTERRUPT REQUEST VECTOR
    547
    548                         .SBTTL  RH11/RH70 REGISTERS
    549
    550                         ;*WORD COUNT REGISTER (RHWC)
    551                         ;*EACH BIT IS CALLED BY BIT NUMBER
    552
    553
    554                         ;*BUS ADDRESS REGISTER (RHBA)
    555                         ;*EACH BIT IS CALLED BY BIT NUMBER
    556
    557
    558                         ;*CONTROL AND STATUS REGISTER 2 (RHCS2)
    559
    560         000001          US1=    1                       ;UNIT SELECT (BIT #0)
    561         000002          US2=    2                       ;UNIT SELECT (BIT #1)
    562         000004          US4=    4                       ;UNIT SELECT (BIT #2)
    563         000010          BAI=    10                      ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
    564         000020          PAT=    20                      ;INVERT PARITY ON MASS BUS TO EVEN (BIT #4)
    565         000040          CLR=    40                      ;CLEAR (BIT #5)
    566         000100          IR=     100                     ;INPUT READY (BIT #6)
    567         000200          OR=     200                     ;OUTPUT READY (BIT #7)
    568         000400          MPE=    400                     ;MASS BUS PARITY ERROR (BIT #8)
    569         001000          MXF=    1000                    ;MISSED TRANSFER ERROR (BIT #9)
    570         002000          PGE=    2000                    ;PROGRAM ERROR (BIT #10)
    571         004000          NEM=    4000                    ;NON EXISTANT MEMORY (BIT #11)
    572         010000          NED=    10000                   ;NON EXISTANT DRIVE (BIT #12)
    573         020000          UPE=    20000                   ;UNIBUS PARITY ERROR (BIT #13)
    574         040000          WCE=    40000                   ;WRITE CHECK ERROR (BIT #14)
    575         100000          DLT=    100000                  ;DATA LATE (BIT #15)
    576
    577
    578                         ;*DATA BUFFER REGISTER (RHDB)
    579                         ;*EACH BIT IS CALLED BY BIT NUMBER
    580
```

```
                                ;*RP04 REGISTERS
                                ;*CONTROL AND STATUS 1 REGISTER. (#00)
000001                          GO=     1               ;GO (BIT #0)
000100                          IE=     100             ;INTERRUPT ENABLE (BIT #6)
000200                          RDY=    200             ;READY (BIT #7)
000400                          A16=    400             ;HIGH ORDER UNIBUS BITS (BIT #8)
001000                          A17=    1000            ;HIGH ORDER UNIBUS BITS (BIT #9)
002000                          PSEL=   2000            ;PORT SELECT (BIT #10)
004000                          DVA=    4000            ;DEVICE AVAILABLE (BIT #11)
020000                          MCPE=   20000           ;MASSBUSS PARITY ERROR (BIT #13)
040000                          TRE=    40000           ;TRANSFER ERROR (BIT #14)
100000                          SC=     100000          ;SPECIAL CONDITION (BIT #15)
                                ;*STATUS REGISTER (RHDS1) (#01)
000001                          DF5=    1               ;DRIVE FORWARD 5''/SEC.  (BIT #0)
000002                          DFF20=  2               ;DRIVE FORWARD 20''/SEC.  (BIT #1)
000004                          DIGB=   4               ;DRIVE TO INNER GAVRD BAND (BIT #2)
000010                          GRV=    10              ;GO REVERSE (BIT #3)
000020                          DL64=   20              ;DIFFERENCE LESS THAN 64 (BIT #4)
000040                          DE1=    40              ;DIFFERENCE EQUALS 1 (BIT #5)
000100                          VV=     100             ;VOLUME VALID (BIT #6)
000200                          DRY=    200             ;DRIVE READY (BIT #7)
000400                          DPR=    400             ;DRIVE PRESENT (BIT #8)
001000                          PROG=   1000            ;PROGRAMABLE (BIT #9)
002000                          LST=    2000            ;LAST SECTOR TRANSFERRED (BIT #10)
004000                          WRL=    4000            ;WRITE LOCK (BIT #11)
010000                          MOL=    10000           ;MEDIUM ON-LINE (BIT #12)
020000                          PIP=    20000           ;POSITIONING OPERATION IN PROGRESS (BIT #13)
040000                          ERR=    40000           ;COMPOSIT ERROR. (BIT #14)
100000                          ATA=    100000          ;ATTENTION ACTIVE (BIT #15)
                                ;*ERROR REGISTER #01 (RHER1) (#02)
000001                          ILF=    1               ;ILLEGAL FUNCTION (BIT #0)
000002                          ILR=    2               ;ILLEGAL REGISTER (BIT #1)
000004                          RMR=    4               ;REGISTER MODIFICATION REFUSED (BIT #2)
000010                          PAR=    10              ;PARITY ERROR (BIT #3)
000020                          FER=    20              ;FORMAT ERROR (BIT #4)
000040                          WCF=    40              ;WRITE CLOCK FAIL (BIT #5)
000100                          ECH=    100             ;ECC HARD ERROR (BIT #6)
000200                          HCE=    200             ;HEADER COMPARE ERROR (BIT #7)
000400                          HCRC=   400             ;HEADER CRC ERROR (BIT #8)
001000                          AOE=    1000            ;ADDRESS OVERFLOW ERROR (BIT #9)
002000                          IAE=    2000            ;INVALID ADDRESS ERROR (BIT #10)
004000                          WLE=    4000            ;WRITE LOCK ERROR (BIT #11)
010000                          DTE=    10000           ;DRIVE TIMING ERROR (BIT #12)
020000                          OPI=    20000           ;OPERATION INCOMPLETE (BIT #13)
040000                          UNS=    40000           ;DRIVE UNSAFE (BIT #14)
100000                          DCK=    100000          ;DATA CHECK ERROR (BIT 15)
                                ;*MAINTAINABILITY REGISTER (RHMR)(#03)
000001                          DMD=    1               ;DIAGINOSTIC MODE (BIT #0)
```

RH11/RH70 REGISTERS

```
                                MCLK=   2                       ;MAINTAINABILITY CLOCK (BIT #1)
                                MINX=   4                       ;MAINTAINABILITY INDEX (BIT #2)
                                MSTCK=  10                      ;MAINTAINABILITY SECTOR CLOCK (BIT #3)
                                MRD=    20                      ;MAINTAINABILITY READ (BIT #4)
                                MWR=    40                      ;MAINTAINABILITY WRITE (BIT #5)
                                DENVL=  200                     ;DATA ENVELOPE (BIT #7)
                                ZER=    400                     ;ZERO DETECT (BIT #8)
                                DTSY=   1000                    ;MAINTAINABILITY SYNC DETECTED (BIT #9)

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

                                AT0=    1                       ;DEVICE 0 (BIT #0)
                                AT1=    2                       ;DEVICE 1 (BIT #1)
                                AT2=    4                       ;DEVICE 2 (BIT #2)
                                AT3=    10                      ;DEVICE 3 (BIT #3)
                                AT4=    20                      ;DEVICE 4 (BIT #4)
                                AT5=    40                      ;DEVICE 5 (BIT #5)
                                AT6=    100                     ;DEVICE 6 (BIT #6)
                                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)

                                EXT1=   1                       ;EXTENSION 1 (BIT #0)
                                EXT2=   2                       ;EXTENSION 2 (BIT #1)
                                EXT4=   4                       ;EXTENSION 3 (BIT #2)
                                EXT10=  10                      ;EXTENSION 4 (BIT #3)
                                EXT20=  20                      ;EXTENSION 5 (BIT #4)
                                EXT40=  40                      ;EXTENSION 6 (BIT #5)
                                SC1=    100                     ;SECTOR COUNT FIELD 0 (BIT #6)
                                SC2=    200                     ;SECTOR COUNT FIELD 1 (BIT #7)
                                SC4=    400                     ;SECTOR COUNT FIELD 2 (BIT #8)
                                SC10=   1000                    ;SECTOR COUNT FIELD 3 (BIT #9)
                                SC20=   2000                    ;SECTOR COUNT FIELD 4 (BIT #10)
                                TRK1=   4000                    ;TRACK FIELD 1 (BIT #11)
                                TRK2=   10000                   ;TRACK FIELD 2 (BIT #12)
                                TRK4=   20000                   ;TRACK FIELD 3 (BIT #13)
                                TRK10=  40000                   ;TRACK FIELD 4 (BIT #14)
                                TRK20=  100000                  ;TRACK FIELD 5 (BIT #15)

                                ;*RP04 ERROR REGISTER #2 (RHER2) (#10)

                                WCU=    1                       ;WRITE CURRENT UNSAFE (BIT #0)
                                CSF=    2                       ;CURRENT SINK FAILURE (BIT #1)
                                WSU=    4                       ;WRITE SELECT UNSAFE (BIT #2)
                                CSU=    10                      ;CURRENT SWITCH UNSAFE (BIT #3)
                                MSE=    20                      ;MOTOR SEQUENCE ERROR (BIT #4)
                                TDF=    40                      ;TRANSITIONS DETECTOR FAILURE (BIT #5)
                                TUF=    100                     ;TRANSITIONS UNSAFE (BIT #6)
```

RH11/RH70 REGISTERS

```
                                FEN=    200             ;FAILSAFE ENABLED (BIT #7)
                                WRU=    400             ;WRITE READY UNSAFE (BIT #8)
                                MHS=    1000            ;MULTIPLE HEAD SELECT (BIT #9)
                                NHS=    2000            ;NO HEAD SELECTION (BIT #10)
                                IXE=    4000            ;INDEX ERROR (BIT #11)
                                VU30=   10000           ;30VOLT UNSAFE (BIT #12)
                                PLU=    20000           ;PLO UNSAFE (BIT #13)
                                ACU=    100000          ;ACUNSAFE (BIT #15)

                        ;*RP05/6 ERROR REGISTER #2 (RHER2) (#10)

                                WCU=    1               ;WRITE CURENT UNSAFE
                                CSF=    2               ;CURRENT SINK FAILURE
                                WSU=    4               ;CURENT SELECT UNSAFE
                                CSU=    10              ;CURRENT SWITCH UNSAFE
                                RAW=    20              ;READ AND WRITE
                                TDF=    40              ;TRANSITIONS DETECTOR FAILURE
                                TUF=    100             ;TRANSITIONS UNSAFE
                                ABS=    200             ;ABNORMAL STOP
                                WRU=    400             ;WRITE READY UNSAFE
                                MHS=    1000            ;MULTIPLE HEAD SELECT
                                NHS=    2000            ;NO HEAD SELECTION
                                IXE=    4000            ;INDEX ERROR
                                PLU=    20000           ;PLO UNSAFE

                        ;*OFFSET REGISTER (RHOF) (#11)

                                OF25=   1               ;OFFSET 25 MICRO INCHES (BIT #0)
                                OF50=   2               ;OFFSET 50 MICRO INCHES (BIT #1)
                                OF100=  4               ;OFFSET 100 MICRO INCHES (BIT #2)
                                OF200=  10              ;OFFSET 200 MICRO INCHES (BIT #3)
                                OF400=  20              ;OFFSET 400 MICRO INCHES (BIT #4)
                                OF800=  40              ;OFFSET 800 MICRO INCHES (BIT #5)

                                OFREV=  200             ;OFFSET NEGATIVE (REVERSE) (BIT #7)
                                HCI=    2000            ;HEADER COMPARE INHIBIT (BIT #10)
                                ECI=    4000            ;ERROR CORRECTION CODE INHIBIT (BIT #11)
                                FMT22=  10000           ;FORMAT BIT (BIT #12


                        ;*DESIRED CYLINDER ADDRESS (RHCA) (#12)
                        ;*EACH BIT IS CALLED BY BIT NUMBER.


                        ;*CURRENT CYLINDER ADDRESS (RHCC) (#13)
                        ;*EACH BIT IS CALLED BY BIT NUMBER


                        ;*SERIAL NUMBER REGISTER (RHSN) (#14)
                        ;*EACH IS CALLED BY BIT NUMBER


                        ;*ERROR REGISTER #03 (RHER3) (#15)

                                PSU=    1               ;PACK SPEED UNSAFE (BIT #0)
                                VUF=    2               ;VELOCITY UNSAFE (BIT #1)
                                UWR=    10              ;ANY UNSAFE EXCEPT READ/WRITE (BIT #3)
```

RH11/RH70 REGISTERS

```
000020                  PRE=    20                      ;DISK PACK ROTATION ERROR (BIT #4)
000040                  ACL=    40                      ;AC LOW (BIT #5)
000100                  OPDCL=  100                     ;DC LOW (BIT #6)
040000                  SKI=    40000                   ;SEEK INCOMPLETE (BIT #14)
100000                  OCYL=   100000                  ;OFF CYLINDER (BIT #15)


                        ;*ECC POSITION REGISTER (RHEC1) (#16)
                        ;*EACH BIT IS CALLED BY BIT NUMBER


                        ;*ECC PATTERN REGISTER (RHEC2) (#17)
                        ;*EACH BIT IS CALLED BY BIT NUMBER

                        .SBTTL  MEMORY MANAGEMENT DEFINITIONS

                        ;*KT11 VECTOR ADDRESS

000250                  MMVEC   = 250

                        ;*KT11 STATUS REGISTER ADDRESSES

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

                        ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

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

                        ;*KERNEL "I" PAGE ADDRESS REGISTERS

172340                  KIPAR0  = 172340
172342                  KIPAR1  = 172342
172344                  KIPAR2  = 172344
172346                  KIPAR3  = 172346
172350                  KIPAR4  = 172350
172352                  KIPAR5  = 172352
172354                  KIPAR6  = 172354
172356                  KIPAR7  = 172356

767
```

```
      1                                 .SBTTL  TRAP CATCHER

                000000                          .=0
                                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                000174                  .=174
      000174    000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
      000176    000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER

                                        .SBTTL  STARTING ADDRESS(ES)

      000200    000137  004620                JMP     @#BEGIN           ;;JUMP TO STARTING ADDRESS OF PROGRAM
      000204    000137  035506                JMP     @#BASECH          ;MODIFY ADDRESSES
      000210    000137  004626                JMP     @#BEGIN2          ;SELECT DRIVE START
      4
      5                                 .SBTTL  ACT11 HOOKS

                                        ;;*****************************************************************
                                        ;HOOKS REQUIRED BY ACT11
                000214                          $SVPC=.                 ;SAVE PC
                000046                          .=46
      000046    030746                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                000052                          .=52
      000052    020000                          .WORD   20000           ;;2)SET LOC.52 TO 20000
                000214                          .=$SVPC                 ;; RESTORE PC
      6
```

```
      0                         .SBTTL  COMMON TAGS

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

            001100                      .=1100
   001100                       $CMTAG:                     ;;START OF COMMON TAGS
   001100   000000              $PASS:  .WORD   0           ;;CONTAINS PASS COUNT
   001102      000              $TSTNM: .BYTE   0           ;;CONTAINS THE TEST NUMBER
   001103      000              $ERFLG: .BYTE   0           ;;CONTAINS ERROR FLAG
   001104   000000              $ICNT:  .WORD   0           ;;CONTAINS SUBTEST ITERATION COUNT
   001106   000000              $LPADR: .WORD   0           ;;CONTAINS SCOPE LOOP ADDRESS
   001110   000000              $LPERR: .WORD   0           ;;CONTAINS SCOPE RETURN FOR ERRORS
   001112   000000              $ERTTL: .WORD   0           ;;CONTAINS TOTAL ERRORS DETECTED
   001114      000              $ITEMB: .BYTE   0           ;;CONTAINS ITEM CONTROL BYTE
   001115      001              $ERMAX: .BYTE   1           ;;CONTAINS MAX. ERRORS PER TEST
   001116   000000              $ERRPC: .WORD   0           ;;CONTAINS PC OF LAST ERROR INSTRUCTION
   001120   000000              $GDADR: .WORD   0           ;;CONTAINS ADDRESS OF 'GOOD' DATA
   001122   000000              $BDADR: .WORD   0           ;;CONTAINS ADDRESS OF 'BAD' DATA
   001124   000000              $GDDAT: .WORD   0           ;;CONTAINS 'GOOD' DATA
   001126   000000              $BDDAT: .WORD   0           ;;CONTAINS 'BAD' DATA
   001130   000000                      .WORD   0           ;;RESERVED--NOT TO BE USED
   001132   000000                      .WORD   0
   001134      000              $AUTOB: .BYTE   0           ;;AUTOMATIC MODE INDICATOR
   001135      000              $INTAG: .BYTE   0           ;;INTERRUPT MODE INDICATOR
   001136   000000                      .WORD   0
   001140   177570              SWR:    .WORD   .SWR        ;;ADDRESS OF SWITCH REGISTER
   001142   177570              DISPLAY: .WORD  DDISP       ;;ADDRESS OF DISPLAY REGISTER
   001144   177560              $TKS:   177560              ;;TTY KBD STATUS
   001146   177562              $TKB:   177562              ;;TTY KBD BUFFER
   001150   177564              $TPS:   177564              ;;TTY PRINTER STATUS REG. ADDRESS
   001152   177566              $TPB:   177566              ;;TTY PRINTER BUFFER REG. ADDRESS
   001154      000              $NULL:  .BYTE   0           ;;CONTAINS NULL CHARACTER FOR FILLS
   001155      002              $FILLS: .BYTE   2           ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
   001156      012              $FILLC: .BYTE   12          ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
   001157      000              $TPFLG: .BYTE   0           ;;'TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
   001160   000000              $REGAD: .WORD   0           ;;CONTAINS THE ADDRESS FROM
                                                           ;;WHICH  ($REG0) WAS OBTAINED
   001162   000000              $REG0:  .WORD   0           ;;CONTAINS (($REGAD)+0)
   001164   000000              $REG1:  .WORD   0           ;;CONTAINS (($REGAD)+2)
   001166   000000              $REG2:  .WORD   0           ;;CONTAINS (($REGAD)+4)
   001170   000000              $REG3:  .WORD   0           ;;CONTAINS (($REGAD)+6)
   001172   000000              $REG4:  .WORD   0           ;;CONTAINS (($REGAD)+10)
   001174   000000              $REG5:  .WORD   0           ;;CONTAINS (($REGAD)+12)
   001176   000000              $TMP0:  .WORD   0           ;;USER DEFINED
   001200   000000              $TMP1:  .WORD   0           ;;USER DEFINED
   001202   000000              $TMP2:  .WORD   0           ;;USER DEFINED
   001204   000000              $TMP3:  .WORD   0           ;;USER DEFINED
   001206   000000              $TMP4:  .WORD   0           ;;USER DEFINED
   001210   000000              $TMP5:  .WORD   0           ;;USER DEFINED
   001212   000000              $TIMES: 0                   ;;MAX. NUMBER OF ITERATIONS
   001214   000000              $ESCAPE:0                   ;;ESCAPE ON ERROR ADDRESS
   001216      207      377  377 $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
   001222      077              $QUES:  .ASCII  /?/         ;;QUESTION MARK
   001223      015              $CRLF:  .ASCII  <15>        ;;CARRIAGE RETURN
   001224      012      000     $LF:    .ASCIZ  <12>        ;;LINE FEED
```

COMMON TAGS

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

```
                                .SBTTL  USER DEFINED TAGS

001226  000254                  RPVEC:  254                     ;RP04/5/6 VECTOR ADDRESS

001230  176722                  RHDB:   176722                  ;DATA BUFFER
001232  176702                  RHWC:   176702                  ;WORD COUNT
001234  176704                  RHBA:   176704                  ;BUS ADDRESS
001236  176710                  RHCS2:  176710                  ;CONTROL AND STATUS
001240  176700                  RHCS1:  176700                  ;CONTROL AND STATUS 1 SEE NOTE ABOVE
001242  176714                  RHER1:  176714                  ;ERROR #1 SEE NOTE ABOVE
001244  176706                  RHDST:  176706                  ;DESIRED SECTOR / TRACK ADDRESS
001246  176740                  RHER2:  176740                  ;ERROR #2
001250  176732                  RHOF:   176732                  ;OFFSET
001252  176734                  RHCA:   176734                  ;DESIRED CYLINDER ADDRESS
001254  176742                  RHER3:  176742                  ;ERROR #3
001256  176716                  RHAS:   176716                  ;ATTENTION SUMMARY SEE NOTE ABOVE
001260  176724                  RHMR:   176724                  ;MAINTAINABILITY
001262  176712                  RHDS1:  176712                  ;DRIVE STATUS
001264  176726                  RHDT:   176726                  ;DRIVE TYPE
001266  176730                  RHSN:   176730                  ;SERIAL NUMBER SEE NOTE ABOVE
001270  176744                  RHEC1:  176744                  ;ECC POSITION
001272  176746                  RHEC2:  176746                  ;ECC PATTERN
001274  176720                  RHLA:   176720                  ;LOOK AHEAD
001276  176736                  RHCC:   176736                  ;CURRENT CYLINDER ADDRESS

                                ;ADDITIONAL REGISTERS LOCATED IN THE RH70 CONTROLLER

001300  176752                  RHCS3:  176752                  ;CONTROL AND STATUS REG #3
001302  176750                  RHBAE:  176750                  ;BUS ADDRESS EXTENSION REGISTER

                                ;THE FOLLOWING LOCATIONS ARE RESERVED FOR REGISTER SAVES
                                ;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

001304  000000                  DB:     0                       ;DATA BUFFER
001306  000000                  WC:     0                       ;WORD COUNT
001310  000000                  BA:     0                       ;BUS ADDRESS
001312  000000                  CS2:    0                       ;CONTROL AND STATUS 2

001314  000000                  CS1:    0                       ;CONTROL AND STATUS 1
001316  000000                  ER1:    0                       ;ERROR #1
001320  000000                  DST:    0                       ;DESIRED SECTOR/TRACK ADDRESS
001322  000000                  ER2:    0                       ;ERROR #2
001324  000000                  OF:     0                       ;OFFSET
001326  000000                  CA:     0                       ;DESIRED CYLINDER ADDRESS
001330  000000                  ER3:    0                       ;ERROR #3
001332  000000                  AS:     0                       ;ATTENTION SUMMARY
001334  000000                  MR:     0                       ;MAINTAINABILITY
001336  000000                  DS1:    0                       ;DRIVE STATUS
001340  000000                  DT:     0                       ;DRIVE TYPE
001342  000000                  SN:     0                       ;SERIAL NUMBER
001344  000000                  EC1:    0                       ;ECC POSITION
001346  000000                  EC2:    0                       ;ECC PATTERN
001350  000000                  LA:     0                       ;LOOK-AHEAD
001352  000000                  CC:     0                       ;CURRENT CYLINDER ADDRESS
```

USER DEFINED TAGS

```
                                ;FLAGS AND INTERNAL PROGRAM CONTROL WORDS

001354                          UNITS:  .BLKW   8.              ;THIS IS FILLED WITH -1
001374  000000                  UNIT:   .WORD   0               ;UNIT UNDER TEST
001376  000000                  NOUNIT: .WORD   0               ;NUMBER OF UNITS PRESENT
                                                                ;USED TO KEEP TRACK OF UNIT UNDER TEST
001400  000000                  NUNIT:  .WORD   0               ;USED TO DETERMIN IF THERE ARE MORE
                                                                ;THAN ONE UNIT
001402  000000                  SELECT: .WORD   0               ;ALL ONES INDICATE UNIT TO BE SELECTED
001404  000000                  UNITSL: .WORD   0               ;UNIT NO. SELECTED

001406  000000                  ERFLG$: 0                       ;ERROR FLAG
001410  000000                  SAVDT:  0                       ;SAVE DRIVE TYPE REGISTER
                                                                ;FOR COMPARISON IN DRIVE CLEAR TEST
                                                                ;AND RH INIT TEST
001412  000000                  SAVSN:  0                       ;SAVE SERIAL NUMBER REGISTER
                                                                ;FOR COMPARISON IN DRIVE CLEAR TEST
                                                                ;AND RH INIT TEST
001414  000000                  PCJSR:  0                       ;SAVE PC OF JSR WHICH GAVE THE ERROR
001416  000000                  ATTENT: 0                       ;ATTENTION BIT FOR PRESENT UNIT
001420  000000                  TOTALAT:        0               ;TOTAL ATTENTION BITS
001422  000000                  TMPILL: 0                       ;TEMPORARY ILLEGAL FUNCTION
001424  000000                  TSECC:  0                       ;FLAG TO SAY IF ECC TEST OR NOT
                                                                ;WHEN =177777 IT IS AN ECC TEST
                                                                ;WHEN =0IT IS NOT AN ECC TEST
001426  000000                  TESDTE: 0                       ;FLAG TO SAY IF DRIVE TIMING ERROR OR NOT
                                                                ;WHEN = 177777 IT IS A DTE TEST
                                                                ;WHEN = 0      IT IS NOT A DTE TEST
001430  000000                  TAGDTE: 0                       ;TEMPORARY TAG USED IN DRIVE TIMING
                                                                ;ERROR TEST

                                ;FUNCTION EQUATES

                                ;TABLE OF COMMAND FUNCTIONS FOR RHCS1
                                ;THEN "GO" BIT HAS TO BE SET

001432                          FUTABL:
001432  000000                  NOPERA: 0                       ;NO OPERATION
001434  000002                  UNLOAD: 2                       ;UNLOAD (STAND BY)
001436  000006                  RECALI: 6                       ;RECALIBRATE
001440  000010                  DCLEAR: 10                      ;DRIVE CLEAR
001442  000012                  RELEAS: 12                      ;RELEASE (DUAL-PORT OPERATION)
001444  000030                  SERCH:  30                      ;SEARCH COMMAND
001446  000050                  WRCHEK: 50                      ;WRITE CHECK DATA
001450  000052                  WRCHDT: 52                      ;WRITE CHECK HEADER AND DATA
001452  000060                  WRIDAT: 60                      ;WRITE DATA
001454  000062                  WRIFOR: 62                      ;WRITE HEADER AND DATA (FORMAT)
001456  000070                  READAT: 70                      ;READ DATA
001460  000072                  REFOR:  72                      ;READ HEADER AND DATA
001462  000004                  SEECOM: 4                       ;SEEK COMMAND
001464  000014                  OFSETC: 14                      ;OFFSET COMMAND
001466  000016                  RETCL:  16                      ;RETURNTO CENTERLIN
001470  000022                  PKACK:  22                      ;PACK ACKNOWLEDGE
001472  000020                  READIN: 20                      ;READ IN
001474  000000                  ILLEGL: .WORD                   ;COMPUTED ILLEGAL FUNCTION

                                ;DATA BUFFER FOR READ WRITE
```

USER DEFINED TAGS

```
001500                          WRFROM: .BLKW   274.            ;WRITE FROM THIS BUFFER
002544                          REINTO: .BLKW   274.            ;READ INTO THIS BUFFER

003610  000000                  TSTNM:  0                       ;TEST NUMBER
003612  000000                  FIRST:  0                       ;IF ZERO WILL TYPE HEADER
                                                                ;IF ONES WILL NOT TYPE HEADER
003614  000000                  RH70:   0                       ;FLAG = 1 FOR RH70 CONTROLLER
                                                                ;FLAG = 0 FOR RH11
003616  000000                  SILOSZ: .WORD   0               ;RH SILO SIZE

                                ;TABLE FOR ATTENTION BITS ATTENTION TABLE

003620      001     002     004 ATABLE: .BYTE   1,2,4,10,20,40,100,200
```

ERROR POINTER TABLE

```
                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

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


   003630                       $ERRTB:
 1
 2                              ;ERROR 1
 3
 4 003630  047474                       EM1                     ;WRONG DATA IN READING OR WRITING HARDWARE REGISTER
 5 003632  053655                       DH1                     ;PC
 6                                                              ;REG. ADDR.
 7                                                              ;GOOD DATA
 8                                                              ;RECEIVED DATA
 9 003634  060144                       DT1                     ;$ERRPC,$TSTNM,REGADR,$GDDAT,$BDDAT
10 003636  060676                       DF1                     ;0,0,0,0,0
11
12                              ;ERROR 2
13
14 003640  047557                       EM2                     ;ERROR ON  DATA COMMAND
15
16 003642  056650                       DH33                    ;PC
17                                                              ;PC OF JSR
18                                                              ;TEST NO
19                                                              ;WORD NO.
20                                                              ;GOOD DATA
21                                                              ;CONTENTS OF RHCS1
22                                                              ;CONTENTS OF RHDS1
23                                                              ;CONTENTS OF RHER1
24 003644  060530                       DT33                    ;$ERRPC,PCJSR,$TSTNM,ERWORD,$GDDAT,CS1,DS1,ER1
25 003646  061046                       DF33                    ;0,0,0,1,0,0,0,0
26
27                              ;ERROR 3
28
29 003650  047557                       EM2                     ;ERROR ON  DATA COMMAND
30
31 003652  056434                       DH32                    ;PC
32                                                              ;PC OF JSR
33                                                              ;TEST NO
34                                                              ;WORD NO.
35                                                              ;GOOD DATA
36                                                              ;BAD DATA
37                                                              ;CONTENTS OF RHCS1
38                                                              ;CONTENTS OF RHDS1
39                                                              ;CONTENTS OF RHER1
40
41 003654  060504                       DT32                    ;$ERRPC,PCJSR,$TSTNM,ERWORD,$GDDAT,$BDDAT,CS1,DS1,ER1
42 003656  061035                       DF32                    ;0,0,0,1,0,0,0,0,0,
```

ERROR POINTER TABLE

```
                                ;ERROR 4
003660  047557                          EM2                     ;ERROR ON  DATA COMMAND
003662  056237                          DH31                    ;PC
                                                                ;TEST NO
                                                                ;WORD NO.
                                                                ;GOOD DATA
                                                                ;BAD DATA
                                                                ;CONTENTS OF RHCS1
                                                                ;CONTENTS OF RHDS1
                                                                ;CONTENTS OF RHER1
003664  060462                          DT31                    ;$ERRPC,$TSTNM,ERWORD,$GDDAT,$BDDAT,CS1,DS1,FR1
003666  061025                          DF31                    ;0,0,1,0,0,0,0,0,
                                ;ERROR 5
003670  000000                          0                       ;
003672  000000                          0                       ;
003674  060462                          DT31                    ;$ERRPC,$TSTNM,ERWORD,$GDDAT,$BDDAT,CS1,DS1,ER1
003676  061025                          DF31                    ;0,0,1,0,0,0,0,0,
                                ;ERROR 6
003700  047606                          EM6                     ;ERROR ON WRITE HEADER AND DATA
003702  056434                          DH32                    ;PC
                                                                ;PC OF JSR
                                                                ;TEST NO
                                                                ;WORD NO.
                                                                ;GOOD DATA
                                                                ;BAD DATA
                                                                ;CONTENTS OF RHCS1
                                                                ;CONTENTS OF RHDS1
                                                                ;CONTENTS OF RHER1
003704  060504                          DT32                    ;$ERRPC,PCJSR,$TSTNM,ERWORD,$GDDAT,$BDDAT,CS1,DS1,ER1
003706  061035                          DF32                    ;0,0,0,1,0,0,0,0,0,
                                ;ERROR 7
003710  047606                          EM6                     ;ERROR ON WRITE HEADER AND DATA
003712  053772                          DH2                     ;PC
                                                                ;TEST NO
                                                                ;WORD NO.
                                                                ;GOOD DATA
                                                                ;BAD DATA
003714  060172                          DT3                     ;$ERRPC,$TSTNM,ERWORD,$GDDAT,$BDDAT
003716  060707                          DF3                     ;0,0,1,0,0,
                                ;ERROR 10
003720  000000                          0                       ;
003722  000000                          0                       ;
003724  060172                          DT3                     ;$ERRPC,$TSTNM,ERWORD,$GDDAT,$BDDAT
```

ERROR POINTER TABLE

```
003726  060707                          DF3                     ;0,0,1,0,0,

                                ;ERROR 11

003730  047645                          EM11                    ;CONTROLLER OR DRIVE STATUS
003732  054103                          DH11                    ;PC
                                                                ;TEST NO
                                                                ;FAILING REG. ADDR
                                                                ;CONTENTS OF RHCS1
                                                                ;CONTENTS OF RHCS2
                                                                ;CONTENTS OF RHDS1
                                                                ;CONTENTS OF RHER1
003734  060206                          DT11                    ;$ERRPC,$TSTNM,$BDADR,CS1,CS2,DS1,FR1
003736  060714                          DF11                    ;0,0,0,0,0,0

                                ;ERROR 12

003740  047645                          EM11                    ;WRONG DATA FROM SILO

003742  053655                          DH1                     ;PC
                                                                ;REG.ADDR
                                                                ;GOOD DATA
                                                                ;RECEIVED DATA
003744  060144                          DT1                     ;$ERRPC,REGADR,$GDDAT,$BDDAT
003746  060676                          DF1                     ;0,0,0,0

                                ;ERROR 13

003750  000000                          0
003752  000000                          0
003754  060144                          DT1                     ;$ERRPC,TSTNM,REGADR,$GDDAT,$BDDAT
003756  060676                          DF1                     ;0,0,0,0,0

                                ;ERROR 14

003760  047700                          EM14                    ;REGISTER FAILED
003762  054257                          DH14                    ;PC
                                                                ;FAILING REG. ADDR
                                                                ;CONTENTS OF FAILING REG.
                                                                ;CONTENTS OF RHCS1
                                                                ;CONTENTS OF RHCS2
                                                                ;CONTENTS OF RHDS1
                                                                ;CONTENTS OF RHER1
003764  060226                          DT14                    ;$ERRPC,$BDADR,$BDDAT,CS1,CS2,DS1,ER1
003766  060723                          DF14                    ;0,0,0,0,0,0,0

                                ;ERROR 15

003770  047720                          EM15                    ;SPECIFIED REG. NON EXISTANT SO ABORT
                                                                ;PROGRAM
003772  054455                          DH15                    ;PC
                                                                ;ADDR. OF REG
003774  060250                          DT15                    ;$ERRPC,TMP1
003776  060733                          DF15                    ;0,0

                                ;ERROR 16
```

ERROR POINTER TABLE

```
157 004000  047770                    EM16            ;WAIT LOOP FAILED
158 004002  054505                    DH16            ;PC
159                                                   ;WAT PC
160                                                   ;BIT WANTED
161                                                   ;REG. ADR.
162                                                   ;REG. CONT.
163 004004  060260                    DT16            ;$ERRPC,$TMP3,$TMP1,$TMP0,$BDDAT
164 004006  060736                    DF16            ;0,0,0,0
165
166                             ;ERROR 17
167
168 004010  050011                    EM17            ;WRITE CHECK FAILING
169 004012  054642                    DH17            ;PC
170                                                   ;TEST NO
171                                                   ;CONTENTS OF RHBA
172                                                   ;CONTENTS OF RHDB
173                                                   ;CONTENTS OF RHWC
174                                                   ;CONTENTS OF RHCS1
175                                                   ;CONTENTS OF RHCS2
176 004014  060276                    DT17            ;$ERRPC,$TSTNM,$BA,DB,WC,CS1,CS2
177 004016  060743                    DF17            ;0,0,0,0,0,0,0
178
179                             ;ERROR 20
180
181 004020  050035                    EM20            ;REGISTER FAILING
182 004022  055016                    DH20            ;PC
183                                                   ;TST NO
184                                                   ;CONTENTS OF RHER1
185                                                   ;CONTENTS OF RHER2
186                                                   ;CONTENTS OF RHER3
187                                                   ;CONTENTS OF RHAS
188                                                   ;CONTENTS OF RHDS1
189 004024  060316                    DT20            ;$ERRPC,TSTNM ER1,ER2,ER3,AS,DS1
190 004026  060752                    DF20            ;0,0,0,0,0,0,0
191
192                             ;ERROR 21
193
194 004030  050056                    EM21            ;INTERRUPT FAILING
195 004032  055171                    DH21            ;PC
196                                                   ;TEST NO
197                                                   ;CONTENTS OF RHCS1
198                                                   ;CONTENTS OF RHAS
199                                                   ;CONTENTS OF RHDS1
200 004034  060336                    DT21            ;$ERRPC,TSTNM,CS1,AS,DS1
201 004036  060761                    DF21            ;0,0,0,0,0
202
203                             ;ERROR 22
204
205 004040  050100                    EM22            ;M' 'MATCH IN DRIVE PRESENT
206                                                   ;LO <ING AT RHAS AND RHCS2-NED(BIT#12)
207                                                   ;DRIVE PRESENT DO NOT AGREE
208                                                   ;NOTE: ON DUAL PORT SYSTEM
209                                                   ;DRIVE ON OTHER PORT WILL NOT GIVE NED
210                                                   ;HENCE THERE WILL BE A MISSMATCH
211                                                   ;177777-MEANS NOT PRESENT
212 004042  055304                    DH22            ;PC
213                                                   ;TEST NO
```

ERROR POINTER TABLE

```
                                                                    ;RHAS UNIT
                                                                    ;RHCS2 UNIT
                                                                    ;
004044  060352                          C.22                        ;$ERRPC,TSTNM$,$GDDAT,$BDDAT
004046  060766                          DF22                        ;0,0,0,0

                                ;ERROR 23
004050  000000                          0                           ;MISSMATCH IN DRIVE PRESENT
                                                                    ;LOOKING AT RHAS AND RHCS2-NED(BIT#12)
                                                                    ;DRIVE PRESENT DO NOT AGREE
                                                                    ;177777-MEANS NOT PRESENT
004052  000000                          0                           ;PC
                                                                    ;TEST NO
                                                                    ;RHAS UNIT
                                                                    ;RHCS2 UNIT
                                                                    ;
004054  060352                          DT22                        ;$ERRPC,TSTNM$,$GDDAT,$BDDAT
004056  060766                          DF22                        ;0,0,0,0

                                ;ERROR 24
004060  050473                          EM24                        ;LOOK AHEAD REGISTER AT THE
                                                                    ;BEGINNING OF A SECTOR IS IN
                                                                    ;ERROR
004062  055377                          DH24                        ;PC
                                                                    ;RHDST
                                                                    ;BAD RHLA
                                                                    ;GOOD RHLA
                                                                    ;SECTOR NO
                                                                    ;SECTOR CLOCK
004064  060364                          DT24                        ;$ERRPC,DST,$BDDAT,$TMP1,$TMP2,$TMP3
004066  060772                          DF24                        ;0,0,0,0,0

                                ;ERROR 25
004070  050566                          EM25                        ;LOOK AHEAD REGISTER IS
                                                                    ;IN ERROR
004072  055377                          DH24                        ;PC
                                                                    ;RHDST
                                                                    ;BAD RHLA
                                                                    ;GOOD RHLA
                                                                    ;SECTOR NO
                                                                    ;SECTOR CLOCK
004074  060364                          DT24                        ;$ERRPC,DST,$BDDAT,$TMP1,$TMP2,$TMP3
004076  060772                          DF24                        ;0,0,0,0,0

                                ;ERROR 26
004100  047645                          EM11                        ;CONTROLLER OR DRIVE STATUS
004102  055554                          DH26                        ;PC
                                                                    ;PC OF JSR
                                                                    ;FAILING REGISTER ADDRESS
                                                                    ;CONTENTS OF RHCS1
                                                                    ;CONTENTS OF RHCS2
                                                                    ;CONTENTS OF RHDS1
```

```
                                                                ;CONTENTS OF RHER1

004104  060404                    DT26                        ;$ERRPC,PCJSR,$BDADR,CS1,CS2,DS1,ER1
004106  061001                    DF26                        ;0,0,0,0,0,0,

                        ;ERROR 27

004110  047474                    EM1                         ;ERROR IN READING OR WRITING HARDWARE REGISTER

004112  055751                    DH27                        ;PC
                                                            ;PC OF JSR
                                                            ;TEST NUMBER
                                                            ;FAILING REGISTER
                                                            ;GOOD DATA
                                                            ;RECEIVED DATA

004114  060426                    DT27                        ;$ERRPC,PCJSR,TSTNM,REGADR,$GDDAT,$BDDAT
004116  061011                    DF27                        ;0,0,0,0,0,0

                        ;ERROR 30

004120  050626                    EM30                        ;CURRENT CYLINDER DOES NOT REFLECT DESIRED CYLINDER REG.
004122  056106                    DH30                        ;PC
                                                            ;PC OF JSR
                                                            ;REGISTER ADDRESS
                                                            ;GOOD DATA
                                                            ;BAD DATA

004124  060444                    DT30                        ;$ERRPC,PCJSR,REGADR,$GDDAT,$BDDAT
004126  061017                    DF30                        ;0,0,0,0,0

                        ;ERROR 31

004130  050747                    EM31                        ;ECC GENERATED IS INCORRECT
                                                            ;EVERY WORD IN THIS SECTOR IS GIVEN IN 'DATA USED''

004132  057046                    DH34                        ;PC
                                                            ;TEST NUMBER
                                                            ;GOOD ECC1
                                                            ;GOOD EC2C
                                                            ;WRITTEN ECC1
                                                            ;WRITTEN ECC2
                                                            ;DATA USED

004134  060552                    DT34                        ;$ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK

004136  061056                    DF34                        ;0,0,0,0,0,0,0

                        ;ERROR 32

004140  051071                    EM32                        ;ON READ COMMAND AFTER DATA AND ECC HAVE BEEN READ
                                                            ;ECC REGISTER OR RHER1 IS IN ERROR
                                                            ;ONLY LOWER 11 BITS OF PATTERN REGISTER
                                                            ;CAN BE READ
                                                            ;THIS SHOULD MATCH LOWER 11 BITS OF ECC1

004142  057220                    DH35                        ;PC
```

ERROR POINTER TABLE

```
                                                        ;TEST NUMBER
                                                        ;GOOD ECC1
                                                        ;GOOD ECC2
                                                        ;PATTERN REGISTER
                                                        ;RHER1

004144  060572                  DT35                  ;$ERRPC,TSTNM,GECC1,GECC2,EC2,ER1

004146  061065                  DF35                  ;0,0,0,0,0,0

                        ;ERROR 33

004150  051355                  EM33                  ;HIGH COUNT BIT NOT HIGH AFTER 38859 CLOCKS
004152  057414                  DH36                  ;PC
                                                      ;PC OF JSR
                                                      ;TEST NUMBER
                                                      ;RHMR
                                                      ;POSITION REG.
                                                      ;PATTERN REGISTER

004154  060614                  DT36                  ;$ERRPC,PCJSR,TSTNM,MR,EC1,EC2

004156  061075                  DF36                  ;0,0,0,0,0,0

                        ;ERROR 34

004160  051427                  EM34                  ;ZERO DETECT BIT NOT HIGH WHEN THE
                                                      ;32 BIT ECC REGISTER HAS ITS 21 BITS
                                                      ;OF ZEROS
                                                      ;ERROR PRINTOUT WILL CONTINUE TILL
                                                      ;ZERO DETECT BIT IS HIGH
004162  057414                  DH36                  ;PC
                                                      ;PC OF JSR
                                                      ;TEST NUMBER
                                                      ;RHMR
                                                      ;POSITION REG.
                                                      ;PATTERN REGISTER

004164  060614                  DT36                  ;$ERRPC,PCJSR,TSTNM,MR,EC1,EC2

004166  061075                  DF36                  ;0,0,0,0,0,0

                        ;ERROR 35

004170  051522                  EM35                  ;POSITION REGISTER OR 11 BITS OF
                                                      ;PATTERN REGISTER INCORRECT
                                                      ;LOWER 11 BITS OF PATTERN REGISTER
                                                      ;SHOULD MATCH LOWER 11 BITS OF GOOD ECC1
                                                      ;DATA ENVELOPE AND N-CODE ZEROS ARE IN DECIMAL

004172  057551                  DH37                  ;PC
                                                      ;TEST NUMBER
                                                      ;ECC POSITION
                                                      ;GOOD POSITION
                                                      ;GOOD ECC1
                                                      ;GOOD ECC2
                                                      ;ECC PATTERN
```

```
                                                        ;DATA ENVELOPE
                                                        ;N-CODE ZEROS

004174  060632                  DT37                  ;$ERRPC,TSTNM,EC1,POSITI,GECC1,GECC2,EC2,DATENV,ZCODE

004176  061103                  DF37                  ;0,0,0,0,0,0,0,0,0

                        ;ERROR 36

004200  052016                  EM36                  ;ON A READ COMMAND WITH NON CORRECTABLE
                                                      ;ERROR INSERTED DCK AND ECH SHOULD BE SET
004202  057220                  DH35                  ;PC
                                                      ;TEST NUMBER
                                                      ;GOOD ECC1
                                                      ;GOOD ECC2
                                                      ;PATTERN REGISTER
                                                      ;POSITION REGISTER
                                                      ;RHER1

004204  060572                  DT35                  ;$ERRPC,TSTNM,GECC1,GECC2,EC2,EC1,ER1

004206  061065                  DF35                  ;0,0,0,0,0,0,0

                        ;ERROR 37

004210  052203                  EM37                  ;ERROR ON  DATA COMMAND
                                                      ;WITH A16 A17 USED

004212  056237                  DH31                  ;PC
                                                      ;TEST NO
                                                      ;WORD NO.
                                                      ;GOOD DATA
                                                      ;BAD DATA
                                                      ;CONTENTS OF RHCS1
                                                      ;CONTENTS OF RHDS1
                                                      ;CONTENTS OF RHER1

004214  060462                  DT31                  ;$ERRPC,$TSTNM,ERWORD,$GDDAT,$BDDAT,CS1,DS1,ER1
004216  061025                  DF31                  ;0,0,1,0,0,0,0,0,

                        ;ERROR 40

004220  000000                  0                     ;
004222  000000                  0                     ;
004224  060462                  DT31                  ;$ERRPC,$TSTNM,ERWORD,$GDDAT,$BDDAT,CS1,DS1,ER1
004226  061025                  DF31                  ;0,0,1,0,0,0,0,0,

                        ;ERROR 41

004230  052271                  EM40                  ;THERE WAS A READ/WRITE HEADER & DATA
                                                      ;ERROR DURING 'DTE' TEST SETUP - THE
                                                      ;TEST IS ABORTED AT THAT POINT
004232  057767                  DH40                  ;PC
                                                      ;TEST NO
                                                      ;FAILING REGISTER ADDRESS
                                                      ;CONTENTS OF RHCS1
                                                      ;CONTENTS OF RHCS2
```

ERROR POINTER TABLE

```
                                                                ;CONTENTS OF RHDS1
                                                                ;CONTENTS OF RHER1
004234  060656                  DT40                          ;$ERRPC,$TSTNM,$BDADR,CS1,CS2,DS1,ER1
004236  061114                  DF40                          ;0,0,0,0,0,0

                        ;ERROR 42

004240  061124                  EM42
004242  062570                  DH42
004244  063046                  DT42    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS.
004246  063104                  DF42

                        ;ERROR 43

004250  061207                  EM43
004252  062570                  DH42
004254  063046                  DT42    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS.
004256  063104                  DF42

                        ;ERROR 44

004260  061253                  EM44
004262  062650                  DH44
004264  063056                  DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
004266  063107                  DF44

                        ;ERROR 45

004270  061310                  EM45
004272  062650                  DH44
004274  063056                  DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
004276  063107                  DF44

                        ;ERROR 46

004300  061337                  EM46
004302  062650                  DH44
004304  063056                  DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
004306  063107                  DF44

                        ;ERROR 47

004310  061366                  EM47
004312  062570                  DH42
004314  063046                  DT42    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS.
004316  063104                  DF42

                        ;ERROR 50

004320  061423                  EM50
004322  062650                  DH44
004324  063056                  DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
004326  063107                  DF44

                        ;ERROR 51

004330  061461                  EM51
```

ERROR POINTER TABLE

```
    499 004332  062650                      DH44
    500 004334  063056                      DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    501 004336  063107                      DF44
    502
    503                             ;ERROR 52
    504
    505 004340  061502                      EM52
    506 004342  062650                      DH44
    507 004344  063056                      DT44
    508 004346  063107                      DF44
    509
    510                             ;ERROR 53
    511
    512 004350  061542                      EM53
    513 004352  062650                      DH44
    514 004354  063056                      DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    515 004356  063107                      DF44
    516
    517                             ;ERROR 54
    518
    519 004360  061602                      EM54
    520 004362  062761                      DH54
    521 004364  063072                      DT54    ;ERROR PC, TEST NUMBER.
    522 004366  063114                      DF54
    523
    524                             ;ERROR 55
    525
    526 004370  061641                      EM55
    527 004372  062650                      DH44
    528 004374  063056                      DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    529 004376  063107                      DF44
    530
    531                             ;ERROR 56
    532
    533 004400  061654                      EM56
    534 004402  062650                      DH44
    535 004404  063056                      DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    536 004406  063107                      DF44
    537
    538                             ;ERROR 57
    539
    540 004410  061671                      EM57
    541 004412  062650                      DH44
    542 004414  063056                      DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    543 004416  063107                      DF44
    544
    545                             ;ERROR 60
    546
    547 004420  061720                      EM60
    548 004422  062650                      DH44
    549 004424  063056                      DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    550 004426  063107                      DF44
    551
    552                             ;ERROR 61
    553
    554 004430  062007                      EM61
    555 004432  062650                      DH44
```

ERROR POINTER TABLE

```
    556 004434  063056                    DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    557 004436  063107                    DF44
    558
    559                             ;ERROR 62
    560
    561 004440  062057                    EM62
    562 004442  062650                    DH44
    563 004444  063056                    DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    564 004446  063107                    DF44
    565
    566                             ;ERROR 63
    567
    568 004450  062134                    EM63
    569 004452  062650                    DH44
    570 004454  063056                    DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    571 004456  063107                    DF44
    572
    573                             ;ERROR 64
    574
    575 004460  062212                    EM64
    576 004462  062650                    DH44
    577 004464  063056                    DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    578 004466  063107                    DF44
    579
    580                             ;ERROR 65
    581
    582 004470  062246                    EM65
    583 004472  062650                    DH44
    584 004474  063056                    DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    585 004476  063107                    DF44
    586
    587                             ;ERROR 66
    588
    589 004500  062330                    EM66
    590 004502  062650                    DH44
    591 004504  063056                    DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
    592 004506  063107                    DF44
    593
    594                             ;ERROR 67
    595
    596 004510  062402                    EM67
    597 004512  062761                    DH54
    598 004514  063072                    DT54    ;ERROR PC, TEST NUMBER.
    599 004516  063114                    DF54
    600
    601                             ;ERROR 70
    602
    603 004520  062467                    EM70
    604 004522  062761                    DH54
    605 004524  063072                    DT54    ;ERROR PC, TEST NUMBER.
    606 004526  063114                    DF54
    607
    608                             ;ERROR 71
    609
    610 004530  062541                    EM71
    611 004532  062650                    DH44
    612 004534  063056                    DT44    ;ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
```

ERROR POINTER TABLE

613 004536 063107 DF44

ERROR POINTER TABLE

```
                                        ;THIS ROUTINE HANDLES UNEXPECTED TIMEOUTS

004540  011600                BADTMO: MOV     (SP),R0         ;SAVE PC WHERE THE TIME OUT OCCURED
004542  005740                        TST     -(R0)           ;ADJUST PC -2
004544  022626                        CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
004546  104401  004554                TYPE    ,65$            ;;TYPE ASCIZ STRING
004552  000417                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>/UNEXPECTED BUS TIMEOUT, PC=/
004612                          64$:
004612  010046                        MOV     R0,-(SP)        ;SETUP FOR TYPING OUT PC
004614  104402                        TYPOC
004616  000240                        NOP                     ;PUT 'HALT(0)' INSTRUCTION HERE IF YOU WISH
                                                                ;TO STOP ON UNEXPECTED TIMEOUT.

                                .SBTTL  START OF PROGRAM

004620  005037  001402        BEGIN:  CLR     SELECT          ;DO NOT SELECT UNIT, NORMAL RUN
004624  000403                        BR      START

004626  012737  177777  001402BEGIN2: MOV     #-1,SELECT      ;SELECT UNIT

004634  005227  000000        START:  INC     #0              ;TTY LOOP, WAIT FOR INCREMENT
004640  001375                        BNE     .-4             ;OF WORD
004642  000005                        RESET                   ;RESET THE WORLD

                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
004644  012706  001100                MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
004650  005026                        CLR     (R6)+           ;;CLEAR MEMORY LOCATION
004652  022706  001140                CMP     #SWR,R6         ;;DONE?
004656  001374                        BNE     .-6             ;;LOOP BACK IF NO
004660  012706  001100                MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
004664  012737  043422  000020        MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
004672  012737  000340  000022        MOV     #340,@#IOTVEC+2 ;;LEVEL 7
004700  012737  046072  000030        MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
004706  012737  000340  000032        MOV     #340,@#EMTVEC+2 ;;LEVEL 7
004714  012737  047222  000034        MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
004722  012737  000340  000036        MOV     #340,@#TRAPVEC+2;LEVEL 7
004730  012737  047312  000024        MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
004736  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;LEVEL 7
004744  005037  001212                CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
004750  005037  001214                CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
004754  112737  000001  001115        MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
004762  012737  004762  001106        MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
004770  012737  004770  001110        MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
004776  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
005002  012737  005036  000004        MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
005010  012737  177570  001140        MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
005016  012737  177570  001142        MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
005024  022777  177777  174106        CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
005032  001012                        BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
005034  000403                        BR      65$             ;;BRANCH IF NO TIMEOUT
005036  012716  005044        64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
```

INITIALIZE THE COMMON TAGS

```
        005042  000002                          RTI
        005044  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
        005052  012737  000174  001142          MOV     #DISPREG,DISPLAY
        005060  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

    24                                  ;SETUP "TIMEOUT" TRAP VECTOR FOR UNEXPECTED BUS TIMEOUTS
    25  005064  012737  004540  000004          MOV     #BADTMO,ERRVEC  ;SETUP FOR UNEXPECTED TIMEOUT
    26  005072  012737  000300  000006          MOV     #PR6,ERRVEC+2   ;LEVEL 6
    27
    31                                  .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
        005100  005227  177777                  INC     #-1             ;;FIRST TIME?
        005104  001030                          BNE     67$             ;;BRANCH IF NO
        005106  104401  005114                  TYPE    ,68$            ;;TYPE ASCIZ STRING
        005112  000425                          BR      67$             ;;GET OVER THE ASCIZ
                                        ;;68$:  .ASCIZ  <CRLF>@CZRJHE0 - RP04/5/6 DISKLESS TEST, PT 2@<CRLF>
        005166                          67$:
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
        005166  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
        005172  001006                          BNE     69$             ;;BRANCH IF YES
        005174  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
        005202  001005                          BNE     70$             ;;BRANCH IF NO
        005204  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
        005206  000403                          BR      70$
        005210  112737  000001  001134  69$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
        005216                          70$:
    32  005216  005227  177777                  INC     #-1             ;FIRST TIME THRU ?
    33  005222  001002                          BNE     1$              ;BR IF NO
    34  005224  104401  052441                  TYPE    ,PREAMB         ;TYPE PRE-AMBLE MESSAGE
    35  005230  005037  177776          1$:     CLR     PS              ;SET PROCESSOR STATUS TO 0
    36  005234  012777  043342  173764          MOV     #RPVECT,@RPVEC  ;THIS IS FOR UNTIMELY DRIVE INTERRUPTS
    37  005242  004737  044620                  JSR     PC,$TKINT       ;INITIALIZE THE TTY KEYBOARD
    38
    39  005246  032777  010000  173664  RH70CK: BIT     #SW12,@SWR      ;LOOK TO SEE IF USING RH70
    40  005254  001403                          BEQ     1$              ;IF SW12 = 0, SKIP NEXT
    41  005256  012737  000001  003614          MOV     #1,RH70         ;IF SW12 = 1, CU IS AN RH70
    42
    43  005264  005737  001402          1$:     TST     SELECT          ;200 START?
    44  005270  001434                          BEQ     TST1            ;GO TO FIRST TEST IF STARTING FROM 200
    45  005272  104401  005300                  TYPE    ,65$            ;;TYPE ASCIZ STRING
        005276  000422                          BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>/SELECT UNIT NUMBER TO BE TESTED ?/
        005344                          64$:
    46  005344  104412                          RDOCT
    47  005346  042716  177770                  BIC     #177770,(SP)    ;ONLY KEEP LAST 3 BITS
    48  005352  011637  001374                  MOV     (SP),UNIT       ;SAVE UNIT TO BE TESTED
    49  005356  012637  001404                  MOV     (SP)+,UNITSL    ;SAVE UNIT TO BE TESTED
```

```
                                ;;*****************************************************************
                                ;*TEST 1        REFERENCE EACH REGISTER
                                ;*REFERENCE EACH REGISTER BY A MOVE INSTRUCTION
                                ;;*****************************************************************
                                ;
005362  000004                  TST1:   SCOPE
005364  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
005372  012706  001100          MOV     #STACK,SP       ;SET UP STACK POINTER
005376  012737  000001  003610          MOV     #1,TSTNM        ;MOVE #1 TO TEST NUMBER
                                ;
005404  012737  046106  000030          MOV     #REGSA1,EMTVEC  ;ERROR VECTOR SO THAT NO REGISTERS ARE SAVED
005412  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
005416  012737  005450  000004          MOV     #2$,ERRVEC      ;SETUP FOR BUS TIMEOUT
005424  012700  000024          MOV     #24,R0          ;THERE ARE 24 REG TO TEST
005430  012701  001230          MOV     #RHDB,R1        ;R1 NOW HAS ADDR OF ADDR OF FIRST REG.
005434  013102                  1$:     MOV     @(R1)+,R2       ;READ HARDWARE REG.
005436  005300          DEC     R0              ;COUNT DOWN
005440  001375          BNE     1$              ;BRANCH IF 24 NOT DONE
005442  012637  000004          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
005446  000455          BR      5$              ;BRANCH IF 24 DONE
                                ;
005450  012716  005456          2$:     MOV     #3$,(SP)        ;SETUP RETURN ADDRESS
005454  000002          RTI
005456                          3$:
005456  012637  000004          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
005462  016137  177776  001200          MOV     -2(R1),$TMP1    ;STORE FAILING REG ADDR
005470  104015          EMT     15
005472  032777  020000  173440          BIT     #SW13,@SWR      ;INHIBIT ERROR PRINTOUT ?
005500  001036          BNE     4$              ;BRANCH IF YES
005502  104401  005510          TYPE    ,65$            ;;TYPE ASCIZ STRING
005506  000427          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>/TO CHANGE BASE ADDRESS, RESTART AT ADDRESS /
005566                          64$:
005566  012746  000204          MOV     #204,-(SP)      ;GET READY TO TYPE STARTING ADDRESS
                                                        ;OF ''CHANGE OF BASE ADDRESS'' ROUTINE
005572  104402          TYPOC
005574  000000          HALT                            ;FORCE THE RESTART!
005576  000137  030660          4$:     JMP     $EOP            ;GO TO END OF PROGRAM
                                ;
005602                          5$:
005602  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
005606  012737  005664  000004          MOV     #6$,ERRVEC      ;INITIALIZE VECTOR
005614  005037  003614          CLR     RH70            ;INIT RH INDICATOR ++ C.W
005620  005777  173456          TST     @RHBAE          ;ADDRESS RPBAE (RH11/RH70?)
005624  005237  003614          INC     RH70            ;FOUND AN RH70-SET MASK
005630  104401  005636          TYPE    ,67$            ;;TYPE ASCIZ STRING
005634  000412          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <CRLF><LF>/RH70 CONTROLLER /
005662                          66$:
005662  000420          BR      8$
                                ;
005664  012716  005672          6$:     MOV     #7$,(SP)        ;SETUP RETURN ADDRESS
005670  000002          RTI
005672                          7$:
005672  104401  005700          TYPE    ,69$            ;;TYPE ASCIZ STRING
005676  000412          BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <CRLF><LF>/RH11 CONTROLLER /
005724                          68$:
```

```
     44 005724  012737  046072  000  0  8$:     MOV     #$ERROR,EMTVEC  ;RESTORE ERROR VECTOR SO REGISTERS ARE SAVED
     45 005732  012637  000004                 MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
     46
     47                                 ;FIND THE SILO SIZE
     48                                 ;IF ITS A RH70C MODIFY TESTS 50 AND 51
     49
     50 005736  004737  032054                 JSR     PC,CLDISK               ;CONTROLLER CLEAR
     51 005742  005037  003616                 CLR     SILOSZ                  ;CLEAR SILO COUNTER
     52 005746  013777  003616  173254  9$:     MOV     SILOSZ,@RHDB            ;LOAD SILO
     53 005754  005237  003616                 INC     SILOSZ                  ;KEEP COUNT
     54 005760  032777  000100  173250         BIT     #IR,@RHCS2              ;IS THE SILO FULL?
     55 005766  001367                         BNE     9$                      ;BRANCH IF NO
     56 005770  022737  000406  003616         CMP     #262.,SILOSZ            ;RH70C?
     57 005776  001031                         BNE     10$                     ;BRANCH IF NO
     58 006000  005037  016270                 CLR     VAR1+2                  ;VAR1 IN TEST 50
     59 006004  012737  001500  016276         MOV     #WRFROM,VAR2+2          ;VAR2 IN TEST 50
     60 006012  062737  000400  016276         ADD     #256.,VAR2+2
     61 006020  062737  000400  016276         ADD     #256.,VAR2+2
     62 006026  005037  016736                 CLR     VAR3+2                  ;VAR3 IN TEST 51
     63 006032  012737  001500  016744         MOV     #WRFROM,VAR4+2          ;VAR4 IN TEST 51
     64 006040  062737  000404  016744         ADD     #260.,VAR4+2
     65 006046  062737  000404  016744         ADD     #260.,VAR4+2
     66 006054  012737  052737  016774         MOV     #52737,VAR5             ;VAR5 IN TEST 51
     67 006062  004737  032054           10$:  JSR     PC,CLDISK               ;CONTROLLER CLEAR
     68
     73                                 ;;*****************************************************************
                                        ;*TEST 2        RHCS2-CONTROL AND STATUS 2
                                        ;*THIS PARTIALLY TESTS RHCS2 TO ENABLE DETERMINATION
                                        ;*OF THE NUMBER OF DRIVES PRESENT
                                        ;;*****************************************************************
        006066  000004                  TST2:   SCOPE
        006070  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
     74 006076  012706  001100                  MOV     #STACK,SP       ;RESET STACK
     75 006102  012737  000002  003610          MOV     #2,TSTNM        ;MOVE #2 TO TEST NUMBER
     76
                                        ;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

        006110  005737  003614                  TST     RH70            ;TEST FLAG FOR RH70 CONTROLLER
        006114  001402                          BEQ     64$             ;IF FLAG = 1, THIS TEST IS SKIPPED
        006116  000137  006144                  JMP     TST3            ;JUMP TO NEXT TEST
        006122                          64$:                            ;IF FLAG = 0, DO THIS TEST
     77
     78 006122  013737  001236  006136          MOV     RHCS2,UN+2
     79 006130  004537  031622                  JSR     R5,BITST        ;TEST BITS IN REGISTER
     80 006134  020017                  UN:     .WORD   20017           ;ONLY THESE BITS ARE TEST READ/WRITE
     81 006136  000000                          .WORD   0               ;ADDRESS OF REG. BEING TESTED
     82 006140  104001                          EMT     1
     83 006142  000207                          RTS     PC              ;RETURN TO BLT3 ROUTINE
     84
     85                                 ;;*****************************************************************
                                        ;*TEST 3        PARTIAL TEST OF RHAS FOR UNIT NUMBERS PRESENT
                                        ;;*****************************************************************
        006144  000004                  TST3:   SCOPE
        006146  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
     86 006154  012737  000003  003610          MOV     #3,TSTNM        ;MOVE #3 TO TEST NUMBER
     87
     88 006162  013701  001256                  MOV     RHAS,R1 ;R1 HAS ADDRESS OF RHAS
```

```
 89 006166  012711  177777            MOV     #-1,@R1         ;THIS CLEARS RHAS (SURPRISED!)
 90 006172  011137  001126            MOV     @R1,$BDDAT      ;TEST DATA
 91 006176  105737  001126            TSTB    $BDDAT
 92 006202  001405                    BEQ     TST4            ;BRANCH IF GOOD
 93 006204  005037  001124            CLR     $GDDAT          ;GOOD DATA
 94 006210  010137  031620            MOV     R1,REGADR       ;FAILING REG. RHAS
 95 006214  104001                    EMT     1
 96                                                           ;WITH ONES MOVED INTO IT
 97
 98                             ;;*****************************************************************
                                ;*TEST 4         TEST FOR DRIVES PRESENT USING RHAS AND RHCS2
                                ;;*****************************************************************
    006216  000004              TST4:   SCOPE
    006220  012737  000001  001212      MOV     #1,$TIMES       ;;DO 1 ITERATION
 99 006226  000005                      RESET                   ;START WITH AN INIT
100 006230  004737  044620              JSR     PC,$TKINT       ;INITILIZE TTY KEYBOARD
101
102 006234  032777  020000  172676      BIT     #SW13,@SWR      ;INHIBIT ERROR TYPEOUT ?
103 006242  001024                      BNE     4$              ;SKIP NEXT IF SO
104 006244  104401  006252              TYPE    ,65$            ;;TYPE ASCIZ STRING
    006250  000421                      BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>/LOOKING AT RHAS - DRIVES PRESENT/
    006314                      64$:
105 006314  013701  001256      4$:     MOV     RHAS,R1         ;R1 HAS ADDR. OF RHAS
106 006320  013702  001236              MOV     RHCS2,R2        ;R2 HAS ADDR. OF RHCS2
107 006324  005012                      CLR     @R2             ;CLEAR RHCS2
108 006326  012700  000010              MOV     #8.,R0          ;COUNT
109 006332  013704  001242              MOV     RHER1,R4        ;R4 HAS ADDR. OF RHER1
110
111 006336  012714  177777      1$:     MOV     #-1,@R4         ;MOVE ERRORS INTO RHER1
112 006342  005212                      INC     @R2             ;INCREMENT UNIT NO.
113 006344  005300                      DEC     R0              ;COUNT
114 006346  001373                      BNE     1$              ;BRANCH IF 8 NOT DONE
115 006350  111137  001420              MOVB    @R1,TOTALAT     ;SAVE TOTAL ATTENTION
116                                                          ;USED IN DRIVE CLEAR TEST
117 006354  105037  001421              CLRB    TOTALAT+1       ;CLEAR UPPER BYTE
118 006360  105711                      TSTB    @R1             ;TEST FOR ANY DRIVES PRESENT
119 006362  001402                      BEQ     2$              ;NONE RESPONDING - TYPE THE MESSAGE
120 006364  000137  006414              JMP     XE2             ;SOME THERE - GO FILL 'UNITS" TABLE
121
122 006370  032777  020000  172542  2$: BIT     #SW13,@SWR      ;INHIBIT ERROR TYPE OUT?
123 006376  001402                      BEQ     3$              ;'NO DRIVES" MESSAGE IF NO
124 006400  000137  006742              JMP     SELTST          ;CHECK FOR SELECTED UNIT START AND LOAD
125                                                          ;'UNITS" TABLE WITH DESIRED DRIVE IF SO
126
127 006404  104401  053107      3$:     TYPE    ,NDRVAS         ;TYPE 'NO DRIVES PRESENT IN RHAS'
128 006410  000137  030660              JMP     $EOP            ;GO OUT
129
130                             ;SET UP UNITS TABLE
131
132 006414                      XE2:
133 006414  012700  000010      2$:     MOV     #8.,R0          ;COUNTER
134 006420  012703  001354              MOV     #UNITS,R3       ;POINTER
135 006424  012723  177777      3$:     MOV     #-1,(R3)+       ;PRESET BLOCK TO ALL ONES
136 006430  005300                      DEC     R0              ;COUNT
137 006432  001374                      BNE     3$              ;BRANCH IF 8 NOT DONE
138 006434  012703  001354              MOV     #UNITS,R3       ;POINTER
```

T4 TEST FOR DRIVES PRESENT USING RHAS AND RHCS2

```
                                        CLR     R5
                                        CLR     NOUNIT          ;NO. OF UNITS PRESENT
                                        MOV     #8.,R0          ;COUNTER
                                        MOV     @R1,$TMP0       ;TEMPORARY STORAGE
                                4$:     ROR     $TMP0           ;SET CARRY IF ONE IN  0 BIT

                                        BCC     5$
                                        MOV     R5,@RHCS2       ;INSERT UNIT NUMBER
                                        CMP     #24020,@RHDT    ;IS THIS A DUAL PORT RP04 ?
                                        BEQ     6$              ;TYPE DRIVE NO. IF SO
                                        CMP     #20020,@RHDT    ;IS THIS A SINGLE PORT RP04 ?
                                        BEQ     6$              ;TYPE DRIVE NO.IF YES

                                        CMP     #24021,@RHDT    ;IS THIS A DUAL PORT RP05 ?
                                        BEQ     6$              ;TYPE UNIT NO. OUT
                                        CMP     #20021,@RHDT    ;IS THIS A SINGLE PORT RP05 ?
                                        BEQ      6$             ;TYPE UNIT NO. IF SO

                                        CMP     #24022,@RHDT    ;IS THIS A DUAL PORT RP06 ?
                                        BEQ     6$              ;TYPE THE NO IF SO
                                        CMP     #20022,@RHDT    ;IS THIS A SINGLE PORT RP06 ?
                                        BEQ     6$              ;TYPE THE NO IF SO

                                ;NO...IT'S NOT AN RP04/RP05/RP06 DEVICE
                                ;SO TYPE OUT THE DEVICE TYPE

                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>/UNIT NUMBER /
                                64$:
                                        MOV     R5,-(SP)        ;GET READY TO TYPE UNIT NUMBER
                                        TYPDS
                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                        BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /, RHDT = /
                                66$:
                                        MOV     @RHDT,-(SP)     ;GET READY TO TYPE RHDT
                                        TYPOC
                                        TYPE    ,69$            ;;TYPE ASCIZ STRING
                                        BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  ? - NOT AN RP04/RP05/RP06 DEVICE?
                                68$:
                                        BR      5$              ;NO RP04/5/6 FOUND SO BRANCH
                                6$:     MOV     R5,(R3)+
                                        TYPE    ,$CRLF
                                        MOV     R5,-(SP)        ;PUT DRIVE NO. ON STACK
                                        TYPDS                   ;TYPE DRIVE NO.
                                        INC     NOUNIT          ;INCR TOTAL NO. OF UNITS

                                5$:     INC     R5              ;'RHCS2' UNIT ADDRESS
                                        DEC     R0              ;DRIVE COUNTER DOWN ONE
                                        BNE     4$              ;TEST AND DO NEXT UNIT IF 8 NOT DONE

                                        MOV     UNITS,UNIT      ;SET UNIT NO. TO FIRST ONE FOUND/OR 0
                                        MOV     NOUNIT,NUNIT    ;SAVE NO. OF UNITS
                                        DEC     NUNIT           ;IF NUNIT = 0 THEN ONLY ONE UNIT
                                                                ;IF NUNIT > 0 THEN MORE THAN ONE UNIT
```

```
    187
    188 006742  005737  001402          SELTST: TST     SELECT          ;STARTING ADDRESS 200 ?
    189 006746  001403                          BEQ     TST5            ;BRANCH IF STARTING FROM 200
    190 006750  013737  001404  001374          MOV     UNITSL,UNIT     ;CHANGE UNIT NUMBER TO SELECTED ONE
    191
    198                                 ;;*****************************************************************
                                        ;*TEST 5         TYPE SERIAL NUMBER AND DRIVE TYPE
                                        ;*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
                                        ;;*****************************************************************
        006756  000004                  TST5:   SCOPE
        006760  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
        006766  012737  007426  001106          MOV     #1$,$LPADR      ;;SET SCOPE LOOP ADDRESS
    199 006774  004737  032054                  JSR     PC,CLDISK       ;FILL UNIT NO.
    200 007000  005037  001416                  CLR     ATTENT  ;CLEAR
    201
    202                                 ;TEST FOR UNIT #0
    203
    204 007004  005737  001374                  TST     UNIT            ;IS UNIT #0 NEXT IN THE UNITS TABLE ?
    205 007010  001022                          BNE     10$             ;IF NOT, TEST THIS UNIT
    206 007012  012700  000041                  MOV     #41,R0          ;IF SO, CHECK THE LOAD MEDIA LOCATION
    207 007016  122710  000011                  CMPB    #11,(R0)        ;WAS IS AN RP04/5/6 ?
    208 007022  001015                          BNE     10$             ;NO... GO AHEAD AND TEST UNIT #0
    209 007024  005737  001402                  TST     SELECT  ;WAS UNIT #0 SELECTED ?
    210                                                                 ;(IE. WAS IT A 210 START ?)
    211 007030  001012                          BNE     10$             ;IF SO...TEST UNIT #0
    212
    213                                 ;INCREMENT THE UNITS TABLE TO NEXT DRIVE (IF ANY)
    214                                 ; & DECREMENT THE 'NOUNITS" PRESENT (TO BE TESTED)
    215
    216 007032  012700  001354                  MOV     #UNITS,R0       ;LOAD UNITS TABLE POINTER
    217 007036  005720                          TST     (R0)+           ;SELECT THE NEXT UNIT IN THE TABLE
    218                                                                 ;(DOUBLE INCREMENT THE POINTER)
    219 007040  022710  177777                  CMP     #-1,(R0)        ;IS THERE ANOTHER TABLE ENTRY PRESENT ?
    220 007044  001404                          BEQ     10$             ;IF NOT (LOC = -1) ...MUST USE UNIT #0
    221 007046  011037  001374                  MOV     (R0),UNIT       ;SET UP TO BE THE UNIT UNDER TEST
    222 007052  005337  001376                  DEC     NOUNITS ;DECREMENT BECAUSE UNIT #0 WON'T BE TESTED
    223 007056  013700  001374          10$:    MOV     UNIT,R0 ;R0 CONTAINS THE UNIT UNDER TEST
    224
    236
    237 007062  116037  003620  001416          MOVB    ATABLE(R0),ATTENT  ;SET APPROPRIATE ATTENTION BIT
    238 007070  104401  007076                  TYPE    ,65$            ;;TYPE ASCIZ STRING
        007074  000413                          BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>/TESTING DRIVE NUMBER/
        007124                          64$:
    239 007124  013746  001374                  MOV     UNIT,-(SP)      ;UNIT NO. TO STACK
    240 007130  104405                          TYPDS                   ;TYPE  DRIVE NO.
    241 007132  104401  007140                  TYPE    ,67$            ;;TYPE ASCIZ STRING
        007136  000410                          BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  <CRLF>/SERIAL NO. = /
        007160                          66$:
    242 007160  017746  172102                  MOV     @RHSN,-(SP)     ;;SAVE @RHSN FOR TYPEOUT
        007164  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    243 007166  104401  007174                  TYPE    ,69$            ;;TYPE ASCIZ STRING
        007172  000410                          BR      68$             ;;GET OVER THE ASCIZ
```

```
                                        ;;69$:  .ASCIZ  <CRLF>/DRIVE TYPE = /
     007214                             68$:
 244 007214  017746  172044                     MOV     @RHDT,-(SP)     ;;SAVE @RHDT FOR TYPEOUT
     007220  104402                             TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 245
 246 007222  022777  024020  172034             CMP     #24020,@RHDT    ;DUAL PORT RP04 ?
 247 007230  001425                             BEQ     4$              ;TYPE ASCII MSG OUT
 248 007232  022777  020020  172024             CMP     #20020,@RHDT    ;SINGLE PORT RP04 ?
 249 007240  001421                             BEQ     4$              ;TYPE THE MESSAGE
 250
 251 007242  022777  024021  172014             CMP     #24021,@RHDT    ;DUAL PORT RP05 ?
 252 007250  001433                             BEQ     5$              ;TYPE THE MESSAGE
 253 007252  022777  020021  172004             CMP     #20021,@RHDT    ;SINGLE PORT RP05 ?
 254 007260  001427                             BEQ     5$              ;TYPE THE MESSAGE
 255
 256 007262  022777  024022  171774             CMP     #24022,@RHDT    ;DUAL PORT RP06 ?
 257 007270  001441                             BEQ     6$              ;TYPE THE MESSAGE
 258 007272  022777  020022  171764             CMP     #20022,@RHDT    ;SINGLE PORT RP06 ?
 259 007300  001435                             BEQ     6$              ;TYPE IT OUT
 260 007302  000451                             BR      1$              ;DRIVE IS NOT RP04/5/6 - SO
 261                                                                    ;DO NOT TYPE ANY MESSAGE OUT
 262
 263                                                                    ;-SHOULD NEVER HAPPEN AT THIS POINT
 264                                                                    ;UNLESS DRIVE GOT SICK WHILE TESTING
 265                                                                    ;WAS IN PROGRESS
 266
 267 007304                             4$:
     007304  104401  007312                       TYPE    ,71$            ;;TYPE ASCIZ STR'NG
     007310  000412                             BR      70$             ;;GET OVER THE ASCIZ
                                        ;;71$:  .ASCIZ  <CRLF>/DRIVE IS AN RP04/<CRLF>
     007336                             70$:
 268 007336  000433                             BR      1$              ;SKIP NEXT ONES
 269 007340                             5$:
     007340  104401  007346                       TYPE    ,73$            ;;TYPE ASCIZ STRING
     007344  000412                             BR      72$             ;;GET OVER THE ASCIZ
                                        ;;73$:  .ASCIZ  <CRLF>/DRIVE IS AN RP05/<CRLF>
     007372                             72$:
 270 007372  000415                             BR      1$              ;SKIP NEXT
 271 007374                             6$:
     007374  104401  007402                       TYPE    ,75$            ;;TYPE ASCIZ STRING
     007400  000412                             BR      74$             ;;GET OVER THE ASCIZ
                                        ;;75$:  .ASCIZ  <CRLF>/DRIVE IS AN RP06/<CRLF>
     007426                             74$:
 272
 273 007426  005777  171634             1$:     TST     @RHSN           ;READ SERIAL NO. AND DRIVE TYPE
 274 007432  005777  171626                     TST     @RHDT           ;THESE TWO ARE TO HELP SCOPE LOOPS
 275 007436  017737  171624  001412             MOV     @RHSN,SAVSN     ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
 276 007444  017737  171614  001410             MOV     @RHDT,SAVDT     ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
 277
 283                                    ;;*****************************************************************
                                        ;*TEST 6         CHECK MOL TO BE LOW
                                        ;*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
                                        ;;*****************************************************************
     007452  000004                     TST6:   SCOPE
 284 007454  012737  000006  003610             MOV     #6,TSTNM        ;MOVE #6 TO TEST NUMBER
```

T6 CHECK MOL TO BE LOW

```
                                        JSR     PC,CLDISK       ;GIVE INITILIZE
                                        BIT     #MOL,@R3        ;CHECK MOL IN RHDS1
                                        BEQ     TST7            ;BRANCH IF MOL LOW
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>/DRIVE IS ON LINE - MOL IS HIGH/
                                64$:
                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                        BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <CRLF>/HIT STOP ON DRIVE TO GET IT OFF LINE/
                                66$:
                                        TYPE    ,69$            ;;TYPE ASCIZ STRING
                                        BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <CRLF>/PROGRAM WILL HANG TESTING MOL TILL MOL IS LOW/
                                68$:
                                1$:     BIT     #MOL,@R3        ;CHECK MOL IN RHDS1
                                        BNE     1$              ;BRANCH IF MOL IS HIGH
                                        TYPE    ,71$            ;;TYPE ASCIZ STRING
                                        BR      70$             ;;GET OVER THE ASCIZ
                                ;;71$:  .ASCIZ  <CRLF><LF>/MOL IS NOW LOW. PROGRAM WILL NOW BE EXECUTED/<CRLF>
                                70$:

                                ;;*****************************************************************
                                ;*TEST 7         PACK ACKNOWLEDGE COMMAND TEST
                                ;*THE PACK ACKNOWLEDGE COMMAND WILL BE LOADED INTO RHCS1 WITH CO
                                ;*THEN ALL REGISTERS WILL BE CHECKED
                                ;*RH CLEAR WILL BE GIVEN
                                ;*THEN ALL REGISTERS WILL BE CHECKED
                                ;;*****************************************************************
                                TST7:   SCOPE
                                        MOV     #STACK,SP       ;RESET STACK
                                        MOV     #7,TSTNM        ;MOVE #7 TO TEST NUMBER

                                        JSR     PC,CLDISK       ;INIT AND SET UP GENERAL REG.
                                                                ;AND UNIT NUMBER
                                        MOV     #DMD,@RHMR      ;SET DIAGNOSTIC MODE

                                        MOV     PKACK,@RHCS1    ;LOAD PACK ACKNOWLEDGE COMMAND INTO RHCS1

                                ;SAVE REGISTERS FOR COMPARISON AFTER GO
                                        JSR     R0,SAVER        ;SAVE
                                        RHWC                    ;FROM
                                        REINTO                  ;TO
                                        19.                     ;NUMBER OF REGISTERS SAVED

                                ;GIVE GO TO PACK ACKNOWLEDGE COMMAND
                                        BIS     #GO,@RHCS1      ;GO TO PACK ACKNOWLEDGE COMMAND

                                ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
                                        BIS     #VV,REINTO+30   ;SAVED RHDS1

                                ;AFTER GO HAS BEEN GIVEN TO PACK ACKNOWLEDGE COMMAND
                                ;SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
                                ;BE DONE
                                        JSR     R0,SAVER        ;SAVE
                                        RHWC                    ;FROM
                                        WRFROM
```

```
                                        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
010074  113737  002571  001525        MOVB    REINTO+25,WRFROM+25;SAVE UPPER RHAS


                                ;COMPARE REGISTERS BEFORE PACK ACKNOWLEDGE COMMAND
                                ;WITH AFTER GO
010102  004037  032756                JSR     R0,COMPAR       ;COMPARE
010106  002544                        REINTO                  ;GOOD BUFFER
010110  001500                        WRFROM                  ;TEST BUFFER
010112  000023                        19.                     ;NUMBER
010114  010122                        1$                      ;RETURN FOR ERROR
010116  010122                        1$                      ;SAME
010120  010142                        2$                      ;RETURN FOR GOOD COMPARISON
010122  013705  037030        1$:     MOV     ERWORD,R5       ;GETTING READY TO INDEX
010126  060505                        ADD     R5,R5           ;DOUBLE ERROR WORD
010130  016537  001230  031620        MOV     RHWC-2(R5),REGADR ;FAILING REGISTER ADDRESS

010136  104001                        EMT     1
                                                              ;AFTER PACK ACKNOWLEDGE COMMAND
                                                              ;WITH GO IS GIVEN
010140  000207                        RTS     PC              ;RETURN TO COMPARISION

010142                          2$:


                                ;;**************************************************************
                                ;*TEST 10        MAKE CURRENT CYLINDER = 0
                                ;;**************************************************************
010142  000004                  TST10:  SCOPE
010144  012706  001100                  MOV     #STACK,SP       ;RESET STACK
010150  012737  000010  003610        MOV     #10,TSTNM       ;MOVE #10 TO TEST NUMBER
010156  004737  032054                JSR     PC,CLDISK       ;INIT DRIVE
010162  012777  000001  171070        MOV     #DMD,@RHMR      ;SET DIAGNOSTIC MODE
010170  004037  034434                JSR     R0,MAKECYL      ;DO SEEK COMMAND FOLLOWED BY AN INIT TO
010174  000000                        .WORD   0               ;CHANGE RHCC TO CYLINDER 0
                                ;;**************************************************************
                                ;*TEST 11        BCTA LEGAL REGISTER RESPONSE TEST
                                ;*THE FOLLOWING TESTS WILL ATTEMPT TO CATCH THOSE BUGS WHICH FAULT
                                ;*INSERTION HAS SHOWN US WE MISSED IN THE FIRST PART OF THIS TEST.
                                ;*
                                ;*THIS TEST WILL ASCERTAIN THAT THE ALL RH11 REGISTERS WIIL
                                ;*RESPOND TO I.E. NOT CAUSE A NON-EXISTANT MEMORY ERROR WHEN ACCESSED
                                ;*BY A "TST" INSRUCTION.
                                ;;**************************************************************
010176  000004                  TST11:  SCOPE
010200  012737  000001  001212        MOV     #1,$TIMES       ;;DO 1 ITERATION
010206  012737  000011  003610        MOV     #11,TSTNM       ;MOVE #11 TO TEST NUMBER
010214  012777  000040  171014        MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER

010222  012705  001240                MOV     #RHCS1,R5       ;R5=LIST POINTER.

010226  013737  000004  001176        MOV     TIMOT,$TMP0     ;SAVE TIMEOUT VECTOR.
010234  012737  010272  000004        MOV     #E00,TIMOT      ;SET VECTOR TO E00
```

```
010242  012706  001100          L00:    MOV     #STACK,SP       ;SET STACK POINTER.
010246  011502                  I00:    MOV     (R5),R2         ;R2=RH11 ADDRESS.
010250  010237  031620                  MOV     R2,REGADR
010254  005712                          TST     (R2)            ;DOES THE RH11 REGISTER RESPOND?
010256  062705  000002                  ADD     #2,R5           ;UPDATE ADDRESS
010262  020527  001272                  CMP     R5,#RHEC2       ;AT THE END OF LEGAL RH11 REGISTERS?
010266  101767                          BLOS    I00             ;NOPE
010270  000401                          BR      000             ;YES! GOTO 000.
010272                          E00:
010272  104042                          EMT     42
010274  013737  001176  000004  000:    MOV     $TMP0,TIMOT     ;RESTORE TIMEOUT VECTOR.

                                ;;*****************************************************************
                                ;*TEST 12        BCTA MOVB LO BYTE TO WC
                                ;*THE FOLLOWING 6 TESTS WILL TEST THE BYTE LOGIC FOR THE RH11 REGISTERS
                                ;*NO ATTEMPT IS MADE TO DETERMINE IF ALL POSSIBLE DATA PATTERNS CAN BE
                                ;*LOADED, ONLY THAT THE RH11 HARDWARE WILL ALLOW THE PROGRAM TO SELECTIVLY
                                ;*MANIPULATE BYTES USING THE FOLLOWING COMMANDS:
                                ;*
                                ;*      1). MOVE BYTE BOTH TO AND FROM THE WORD COUNT REGISTER
                                ;*
                                ;*      2). BIT SET INTO THE WORD COUNT REGISTER.
                                ;*
                                ;*      3). BIT CLEAR INTO THE WORD COUNT REGISTER.
                                ;;*****************************************************************
010302  000004                  TST12:  SCOPE
010304  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
010312  012737  000012  003610          MOV     #12,TSTNM       ;MOVE #12 TO TEST NUMBER
010320  012777  000040  170710          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
010326  012706  001100                  MOV     #STACK,SP       ;SET STACK POINTER.
010332  013702  001232                  MOV     RHWC,R2         ;R2=RH11 WC ADDRESS.
010336  010237  031620                  MOV     R2,REGADR       ;REGISTER ADDRESS TO REGADR FOR TYPING.
010342  012737  000377  001124          MOV     #377,$GDDAT     ;$GDDAT=S/B.
010350  005012                  L02:    CLR     (R2)            ;CLEAR RH11 WC.
010352  112712  000377          I02:    MOVB    #377,(R2)       ;SET WC TO LO BYTE.
010356  011237  001126                  MOV     (R2),$BDDAT     ;$BDDAT=ACTUAL WAS
010362  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS
010370  001401                          BEQ     TST13           ;NEXT TEST
010372  104044                          EMT     44

                                ;;*****************************************************************
                                ;*TEST 13        BCTA MOVB HI BYTE TO WC
                                ;;*****************************************************************
010374  000004                  TST13:  SCOPE
010376  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
010404  012737  000013  003610          MOV     #13,TSTNM       ;MOVE #13 TO TEST NUMBER
010412  012777  000040  170616          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
```

T13 BCTA MOVB HI BYTE TO WC

```
    428 010420  012706  001100                  MOV     #STACK,SP       ;SET STACK POINTER.
    429
    430 010424  013702  001232                  MOV     RHWC,R2         ;R2=RH11 WC HI BYTE ADDRESS.
    431 010430  010237  031620                  MOV     R2,REGADR
    432 010434  012737  177400  001124          MOV     #177400,$GDDAT  ;$GDDAT=S/B.
    433
    434 010442  005012                  L03:    CLR     (R2)            ;CLEAR RH11 WC.
    435
    436 010444  005202                          INC     R2              ;R2=HI BYTE ADDRESS.
    437
    438 010446  112712  000377          I03:    MOVB    #377,(R2)       ;SET WC HI BYTE.
    439
    440 010452  005302                          DEC     R2              ;R2=FULL WORD ADDRESS.
    441
    442 010454  011237  001126                  MOV     (R2),$BDDAT     ;$BDDAT=ACTUAL.
    443
    444 010460  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
    445 010466  001401                          BEQ     TST14           ;NEXT TEST
    446 010470  104044                          EMT     44
    447
    448                             ;;**************************************************************
                                    ;*TEST 14        BCTA BISB LO BYTE TO WC
                                    ;;**************************************************************
        010472  000004              TST14:  SCOPE
        010474  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
    449 010502  012737  000014  003610          MOV     #14,TSTNM       ;MOVE #14 TO TEST NUMBER
    450 010510  012777  000040  170520          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
    451
    452 010516  012706  001100                  MOV     #STACK,SP       ;SET STACK POINTER.
    453
    454 010522  013702  001232                  MOV     RHWC,R2         ;R2=RH11 WC ADDRESS
    455 010526  010237  031620                  MOV     R2,REGADR
    456 010532  012737  000377  001124          MOV     #377,$GDDAT     ;$GDDAT=S/B.
    457
    458 010540  005012                  L04:    CLR     (R2)            ;CLEAR RH11 WC.
    459
    460 010542  012712  000252                  MOV     #252,(R2)       ;SET UP WC
    461 010546  152712  000125          I04:    BISB    #125,(R2)       ;DO A BISB
    462
    463 010552  011237  001126                  MOV     (R2),$BDDAT     ;$BDDAT=WAS.
    464
    46? 010556  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS
    46? 010564  001401                          BEQ     TST15           ;NEXT TEST
    467 010566  104045                          EMT     45
    468
    469                             ;;**************************************************************
                                    ;*TEST 15        BCTA BISB HI-BYTE TO WC
                                    ;;**************************************************************
        010570  000004              TST15:  SCOPE
        010572  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
    470 010600  012737  000015  003610          MOV     #15,TSTNM       ;MOVE #15 TO TEST NUMBER
    471 010606  012777  000040  170422          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
    472
    473 010614  012706  001100                  MOV     #STACK,SP       ;SET STACK POINTER.
    474
    475 010620  013702  001232                  MOV     RHWC,R2         ;R2=RH11 WC ADDRESS.
    476 010624  010237  031620                  MOV     R2,REGADR
```

```
                                                                                
  477 010630  012737  177400  001124          MOV     #177400,$GDDAT  ;$GDDAT=S/B.
  478                                                                           
  479 010636  005012                   L05:    CLR     (R2)            ;CLEAR RH11 WC.
  480                                                                           
  481 010640  005202                           INC     R2              ;R2 =HI BYTE.
  482                                                                           
  483 010642  112712  000125                   MOVB    #125,(R2)       ;SET UP RH11 WC.
  484 010646  152712  000252           I05:    BISB    #252,(R2)       ;DO A BISB.
  485                                                                           
  486 010652  005302                           DEC     R2              ;R2=FULL WORD ADDRESS.
  487                                                                           
  488 010654  011237  001126                   MOV     (R2),$BDDAT     ;$BDDAT=WAS.
  489                                                                           
  490 010660  023737  001126  001124           CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
  491 010666  001401                           BEQ     TST16           ;NEXT TEST
  492 010670  104045                           EMT     45
  493                                                                           
  494                                ;;*************************************************************************
                                     ;*TEST 16        BCTA BICB LO-BYTE TO WC
                                     ;;*************************************************************************
      010672  000004                 TST16:  SCOPE
      010674  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
  495 010702  012737  000016  003610          MOV     #16,TSTNM       ;MOVE #16 TO TEST NUMBER
  496 010710  012777  000040  170320          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
  497                                                                           
  498 010716  012706  001100                   MOV     #STACK,SP       ;SET STACK POINTER.
  499                                                                           
  500 010722  013702  001232                   MOV     RHWC,R2         ;R2=RH11 WC ADDRESS.
  501 010726  010237  031620                   MOV     R2,REGADR
  502 010732  012737  000252  001124           MOV     #252,$GDDAT     ;$GDDAT=S/B.
  503                                                                           
  504 010740  005012                   L06:    CLR     (R2)            ;CLEAR RH11 WC.
  505                                                                           
  506 010742  012712  000377                   MOV     #377,(R2)       ;SET WC=0000377
  507                                                                           
  508 010746  142712  000125           I06:    BICB    #125,(R2)       ;DO A BICB
  509                                                                           
  510 010752  011237  001126                   MOV     (R2),$BDDAT     ;$BDDAT=WAS.
  511                                                                           
  512 010756  023737  001126  001124           CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
  513 010764  001401                           BEQ     TST17           ;NEXT TEST
  514 010766  104046                           EMT     46
  515                                                                           
  516                                ;;*************************************************************************
                                     ;*TEST 17        BCTA BICB HI- BYTE TO WC
                                     ;;*************************************************************************
      010770  000004                 TST17:  SCOPE
      010772  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
  517 011000  012737  000017  003610          MOV     #17,TSTNM       ;MOVE #17 TO TEST NUMBER
  518 011006  012777  000040  170222          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
  519                                                                           
  520 011014  012706  001100                   MOV     #STACK,SP       ;SET STACK POINTER.
  521                                                                           
  522 011020  013702  001232                   MOV     RHWC,R2         ;R2=RH11 WC ADDRESS.
  523 011024  010237  031620                   MOV     R2,REGADR
  524 011030  012737  125000  001124           MOV     #125000,$GDDAT  ;$GDDAT=S/B.
  525                                                                           
```

```
    526 011036  005012                  L07:    CLR     (R2)            ;CLEAR RH11 WC.
    527
    528 011040  005202                          INC     R2              ;R2=HI BYTE.
    529
    530 011042  112712  000377                  MOVB    #377,(R2)       ;SET WC=177400
    531 011046  142712  000125          I07:    BICB    #125,(R2)       ;DO A BICB
    532
    533 011052  005302                          DEC     R2              ;R2=FULL WORD.
    534
    535 011054  011237  001126                  MOV     (R2),$BDDAT     ;$BDDAT=WAS.
    536
    537 011060  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
    538 011066  001401                          BEQ     TST20           ;NEXT TEST
    539 011070  104046                          EMT     46
    540
    552                                 ;;***************   ************************************************
                                        ;*TEST 20            ILLEGAL REGISTER TEST
                                        ;*THIS TEST ATTEMPTS TO ACCESS AN ILLEGAL REGISTER WITHIN THE RANGE
                                        ;*OF ADDRESSES SELECTED BY THE XOR'S. THE RH11 ADDRESS DECODE LOGIC WILL ALLOW UP TO 32
                                        ;*CONTIGUOUS REGISTERS TO EXIST, IN OUR CONFIGURATION ONLY 16 WILL REALLY
                                        ;*EXIST, THIS TEST ATTEMPTS TO ACCESS THE 20(8) REGISTER - IF IT RESPONDS
                                        ;*THE TEST WILL TYPE OUT AN ERROR.
                                        ;*
                                        ;*FOR THE CASE OF THE RH70, THE CONFIGURATION ALLOWS 18 OR 32 REGISTERS, SO
                                        ;*WE WILL ATTEMPT TO ADDRESS REGISTER 23(8), 33(8) OR AS BEFORE THE LAST
                                        ;*REGISTER + 2.
                                        ;;*****************************************************************
    011072  000004                      TST20:  SCOPE
    011074  012737  000001  001212              MOV     #1,$TIMES       ;;DO 1 ITERATION
    553 011102  012737  000020  003610          MOV     #20,TSTNM       ;MOVE #20 TO TEST NUMBER
    554 011110  012777  000040  170120          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
    555
    556 011116  005737  003614                  TST     RH70            ;CHECK TO SEE IF RUNNING WITH RH70
    557 011122  001403                          BEQ     1$              ;IF NOT...SKIP NEXT & DO FOLLOWING
    558 011124  013702  001300                  MOV     RHCS3,R2        ;R2 = LAST LEGAL RH70 REG ADDRESS
    559 011130  000402                          BR      2$              ;SKIP NEXT
    560
    561 011132  013702  001272          1$:     MOV     RHEC2,R2        ;R2 = LAST LEGAL RH11 REG ADDRESS.
    562 011136  062702  000002          2$:     ADD     #2,R2           ;R2 = FIRST ILLEGAL ADDRESS.
    563 011142  010237  031620                  MOV     R2,REGADR
    564 011146  005037  001126                  CLR     $BDDAT          ;$BDDAT=WAS.
    565 011152  005037  001124                  CLR     $GDDAT          ;$GDDAT=S/B.
    566
    567 011156  013737  000004  001176          MOV     TIMOT,$TMP0     ;SAVE TIMEOUT VECTOR.
    568 011164  012737  011210  000004          MOV     #010,TIMOT      ;SET TIMEOUT VECTOR TO 010.
    569
    570 011172  012706  001100          L10:    MOV     #STACK,SP       ;SET THE STACK POINTER.
    571
    572 011176  005712                          TST     (R2)            ;TEST ADDRESS.
    573
    574 011200  062702  000024                  ADD     #24,R2          ;THIS MIGHT BE THE CASE OF 32 REGISTERS
    575 011204  005712                          TST     (R2)            ;TEST IT AGAIN
    576 011206  104047                          EMT     47
    577
    578 011210  013737  001176  000004  010:    MOV     $TMP0,TIMOT     ;RESTORE TIMEOUT VECTOR.
    579
    589                                 ;;*****************************************************************
```

```
                                        ;*TEST 21       BCTB (NED) NON-EXISTANT DRIVE TEST. (SET)
                                        ;*THE FOLLOWING TWO TESTS DETERMINE IF ALL NON DRIVES SET THE
                                        ;*NED BIT AND THAT ALL EXISTING DRIVES CLEAR THE NED BIT.
                                        ;*THE TESTS LOOK AT TOTALAT TO DETERMINE WHICH DRIVES EXIST AND TEST THAT
                                        ;*THESE DRIVES WILL CLEAR THE NED BIT. AND ALSO THAT NON EXISTANT DRIVES
                                        ;*WILL SET THE NED BIT.
                                        ;*ON FIRST PASS ONLY, IF ALL DRIVES ARE ON LINE A MESSAGE WILL SO ADVISE THE
                                        ;*OPERATOR. THE TEST WILL CONTINUE REGARLESS OF THE OPERATORS ACTION.
                                        ;;*****************************************************************
     011216  000004                     TST21:  SCOPE
     011220  012737  000001  001212             MOV     #1,$TIMES       ;;DO 1 ITERATION
 590 011226  012737  000021  003610             MOV     #21,TSTNM       ;MOVE #21 TO TEST NUMBER
 591 011234  012777  000040  167774             MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
 592
 593 011242  123727  001420  000377             CMPB    TOTALAT,#377            ;ARE ALL DRIVES ON LINE.
 594 011250  001147                             BNE     U11             ;NO GO RUN THE TEST.
 595
 596 011252  023727  001100  000000             CMP     $PASS,#0        ;IF PASS #0 THEN TYPE MESSAGE
 597                                            ;ADVISING OPERATOR
 598                                            ;THAT THIS TEST WILL NOT BE RUN
 599 011260  001002                             BNE     Y11
 600 011262  000137  011672                     JMP     X11
 601
 602 011266                             Y11:
     011266  104401  011274                    TYPE    ,65$            ;;TYPE ASCIZ STRING
     011272  000426                            BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>/ALL DRIVES APPEAR TO BE ON LINE THEREFORE/
     011350                             64$:
 603 011350  104401  011356                    TYPE    ,67$            ;;TYPE ASCIZ STRING
     011354  000426                            BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  <CRLF>/THE NED NON-EXISTANT DRIVE TEST CANNOT BE/
     011432                             66$:
 604 011432  104401  011440                    TYPE    ,69$            ;;TYPE ASCIZ STRING
     011436  000424                            BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  <CRLF>/RUN. TO RUN THIS TEST TAKE ONE OR MORE/
     011510                             68$:
 605 011510  104401  011516                    TYPE    ,71$            ;;TYPE ASCIZ STRING
     011514  000425                            BR      70$             ;;GET OVER THE ASCIZ
                                        ;;71$:  .ASCIZ  <CRLF>/DRIVES OFF-LINE AND RESTART THE PROGRAM/
     011570                             70$:
 606
 607 011570  012706  001100             U11:    MOV     #STACK,SP       ;SET STACK POINTER.
 608
 609 011574  013702  001236                     MOV     RHCS2,R2        ;R2=RH11 RHCS2 ADDRESS.
 610 011600  010237  031620                     MOV     R2,REGADR
 611
 612 011604  013700  001420                     MOV     TOTALAT,R0
 613 011610  005001                             CLR     R1
 614 011612  006000                     S11:    ROR     R0
 615 011614  103423                             BCS     N11
 616
 617 011616  010137  001124             R11:    MOV     R1,$GDDAT       ;$GDDAT=S/B.
 618 011622  052737  010000  001124             BIS     #NED,$GDDAT
 619
 620 011630  013705  001240                     MOV     RHCS1,R5        ;ADDRESS OF A DEVICE REGISTER.
 621
 622 011634  010112                     L11:    MOV     R1,(R2)         ;LOAD A NON-EXISTANT DRIVE.
```

```
623
624 011636  011537  001176                  MOV     (R5),$TMP0      ;ATTEMPT TO READ FROM DEVICE REGISTER.
625
626 011642  011237  001126                  MOV     (R2),$BDDAT     ;$BDDAT=WAS.
627 011646  042737  167770  001126          BIC     #^C<NED!US4!US2!US1>,$BDDAT;$BDDAT=SAVED DATA.
628
629 011654  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
630 011662  001005                          BNE     E11
631
632 011664  005201                  N11:    INC     R1
633 011666  020127  000010                  CMP     R1,#8.          ;TESTED ALL DRIVES YET.
634 011672                          X11:
    011672  001402                          BEQ     TST22           ;NEXT TEST
635 011674  000746                          BR      S11
636
637 011676                          E11:
    011676  104050                          EMT     50
638
639                                 ;;*****************************************************************
                                    ;*TEST 22       BCTB (NED) NON-EXISTANT DRIVE TEST. (CLEARED)
                                    ;;*****************************************************************
    011700  000004                  TST22:  SCOPE
    011702  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
640 011710  012737  000022  003610          MOV     #22,TSTNM       ;MOVE #22 TO TEST NUMBER
641 011716  012777  000040  167312          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
642
643 011724  012706  001100                  MOV     #STACK,SP       ;SET STACK POINTER.
644
645 011730  013702  001236                  MOV     RHCS2,R2        ;R2=RH11 RHCS2 ADDRESS.
646 011734  010237  031620                  MOV     R2,REGADR
647 011740  013700  001420                  MOV     TOTALAT,R0
648 011744  005001                          CLR     R1
649 011746  006C00                  S12:    ROR     R0
650 011750  103020                          BCC     N12
651
652 011752  010137  001124                  MOV     R1,$GDDAT
653
654 011756  013705  001240                  MOV     RHCS1,R5        ;ADDRESS OF A DEVICE REGISTER.
655
656 011762  010112                  L12:    MOV     R1,(R2)         ;SELECT UNIT.
657
658 011764  011537  001176                  MOV     (R5),$TMP0      ;ATTEMPT TO READ FROM DEVICE.
659
660 011770  011237  001126                  MOV     (R2),$BDDAT     ;$BDDAT=WAS.
661 011774  042737  167770  001126          BIC     #^C<NED!US4!US2!US1>,$BDDAT     ;$BDDAT=SAVED DATA.
662
663 012002  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
664 012010  001005                          BNE     E12
665
666 012012  005201                  N12:    INC     R1
667 012014  020127  000010                  CMP     R1,#8.          ;TESTED ALL DRIVES.
668 012020  001402                          BEQ     TST23           ;NEXT TEST
669
670 012022  000751                          BR      S12
671 012024                          E12:
    012024  104050                          EMT     50
672
```

```
                                        ;;*****************************************************************
                                        ;*TEST 23       BCTB AS REGISTER TEST
                                        ;*TEST TO SEE IF WE CAN READ FROM THE "AS" REGISTER AND NOT CAUSE
                                        ;*A NED NON-EXISTANT DRIVE ERROR
                                        ;;*****************************************************************
012026  000004                          TST23:  SCOPE
012030  012737  000001  001212                  MOV     #1,$TIMES       ;;DO 1 ITERATION
678 012036  012737  000023  003610              MOV     #23,TSTNM       ;MOVE #23 TO TEST NUMBER
679 012044  012777  000040  167164              MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
680
681 012052  012706  001100                      MOV     #STACK,SP       ;SET STACK POINTER.
682
683 012056  013702  001236                      MOV     RHCS2,R2        ;R2=RH11 CS2 ADDRESS.
684 012062  012737  000007  001124              MOV     #US4!US2!US1,$GDDAT;$GDDAT=S/B.
685
686 012070  013705  001240                      MOV     RHCS1,R5        ;ADDRESS OF A DEVICE REGISTER.
687
688 012074  012712  000007          L13:        MOV     #US4!US2!US1,(R2);LOAD A NON-EXISTANT DEVICE.
689
690 012100  013702  001256                      MOV     RHAS,R2         ;R2= "AS".
691
692 012104  011237  001176                      MOV     (R2),$TMP0      ;ATTEMPT TO READ FROM DEVICE AS.
693
694 012110  005012                              CLR     (R2)            ;CLEAR AS REGISTER.
695 012112  013702  001236                      MOV     RHCS2,R2        ;R2=RH11 CS2 ADDRESS.
696 012116  010237  031620                      MOV     R2,REGADR
697
698 012122  011237  001126                      MOV     (R2),$BDDAT     ;$BDDAT=WAS.
699 012126  042737  167770  001126              BIC     #^C<NED!US4!US2!US1>,$BDDAT;SAVE NED AND DEVICE SELECTED.
700
701 012134  023737  001126  001124              CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
702 012142  001401                              BEQ     TST24           ;NEXT TEST
703 012144  104051                              EMT     51
704
710                                     ;;*****************************************************************
                                        ;*TEST 24       BCTC BUS ADDRESS REGISTER
                                        ;*THE NEXT THREE TESTS WILL TEST THE BUS ADDRESS REGISTER IN BOTH WORD
                                        ;*AND BYTE MODE. THE PATTERNS ARE CHOSEN TO PICK UP ANY DROPPED OR STUCK
                                        ;*BITS.
                                        ;;*****************************************************************
012146  000004                          TST24:  SCOPE
012150  012737  000001  001212                  MOV     #1,$TIMES       ;;DO 1 ITERATION
711 012156  012737  000024  003610              MOV     #24,TSTNM       ;MOVE #24 TO TEST NUMBER
712 012164  012777  000040  167044              MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
713
714 012172  012706  001100                      MOV     #STACK,SP       ;SET STACK POINTER.
715
716 012176  013702  001234                      MOV     RHBA,R2         ;R2=RH11 BA ADDRESS.
717 012202  010237  031620                      MOV     R2,REGADR
718 012206  012705  012246                      MOV     #LST14A,R5      ;R5=TEST LIST ADDRESS.
719
720 012212  012537  001124          L14:        MOV     (R5)+,$GDDAT    ;$GDDAT=S/B.
721
722 012216  013712  001124          I14:        MOV     $GDDAT,(R2)     ;SET BUS ADDRESS REGISTER.
723 012222  011237  001126                      MOV     (R2),$BDDAT     ;READ BUS ADDRESS REGISTER.
724
725 012226  023737  001126  001124              CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
```

```
                                                BEQ     N14             ;GET NEXT TEST DATA.
                                                EMT     52
                                        N14:    TST     (R5)            ;AT END OF LIST
                                                BNE     L14             ;NO!
                                                BR      TST25           ;NEXT TEST
                                        LST14A: 2
                                                4
                                                10
                                                20
                                                40
                                                100
                                                200
                                                400
                                                1000
                                                2000
                                                4000
                                                10000
                                                20000
                                                40000
                                                100000
                                                177776
                                                177774
                                                177772
                                                177766
                                                177756
                                                177736
                                                177676
                                                177576
                                                177376
                                                176776
                                                175776
                                                173776
                                                167776
                                                157776
                                                137776
                                                077776
                                                0

                                        ;;******************************************************************
                                        ;*TEST 25       BCTC BUS ADDRESS REGISTER LO-BYTE
                                        ;;******************************************************************
012346  000004                          TST25:  SCOPE
012350  012737  000001  001212                  MOV     #1,$TIMES       ;;DO 1 ITERATION
012356  012737  000025  003610                  MOV     #25,TSTNM       ;MOVE #25 TO TEST NUMBER
012364  012777  000040  166644                  MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER

012372  012706  001100                          MOV     #STACK,SP       ;RESET STACK.

012376  013702  001234                          MOV     RHBA,R2         ;R2=RH11 BA ADDRESS.
012402  010237  003620                          MOV     R2,REGADR
012406  012705  012450                          MOV     #LST15A,R5      ;R5=TEST LIST ADDRESS.

012412  005012                          L15:    CLR     (R2)            ;CLEAR RH11 BA REGISTER.

012414  012537  001124                          MOV     (R5)+,$GDDAT    ;$GDDAT=S/B.
```

```
                                I15:    MOVB    $GDDAT,(R2)     ;SET BUS ADDRESS REGISTER.
                                        MOV     (R2),$BDDAT     ;READ BUS ADDRESS REGISTER.

                                        CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
                                        BEQ     R15

                                        EMT     53

                                R15:    TST     (R5)            ;AT END OF TEST LIST.
                                        BNE     L15             ;NO!
                                        BR      TST26           ;NEXT TEST
                                ;THIS LIST WILL BE USED TO LOAD THE LOWER BYTE OF THE BA REGISTER.
                                ;
                                LST15A: 2
                                        4
                                        10
                                        20
                                        40
                                        100
                                        200
                                        376
                                        372
                                        366
                                        356
                                        336
                                        276
                                        176
                                        0

                                ;;**********************************************************************
                                ;*TEST 26        BCTC BUS ADDRESS REGISTER HI-BYTE
                                ;;**********************************************************************
 012506 000004                  TST26:  SCOPE
 012510 012737 000001 001212            MOV     #1,$TIMES       ;;DO 1 ITERATION
 012516 012737 000026 003610            MOV     #26,TSTNM       ;MOVE #26 TO TEST NUMBER
 012524 012777 000040 166504            MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER

 012532 012706 001100                   MOV     #STACK,SP       ;RESET STACK.

 012536 013702 001234                   MOV     RHBA,R2         ;R2=RH11 BA ADDRESS.
 012542 010237 031620                   MOV     R2,REGADR
 012546 012705 012620                   MOV     #LST16A,R5      ;R5=TEST LIST ADDRESS.

 012552 005012                  L16.    CLR     (R2)            ;CLEAR BA REGISTER.

 012554 012537 001124                   MOV     (R5)+,$GDDAT    ;$GDDAT=S/B.

 012560 005202                          INC     R2              ;R2=HI BYTE ADDRESS.

 012562 113712 001124           I16:    MOVB    $GDDAT,(R2)     ;SET BUS ADDRESS REGISTER HI-BYTE.

 012566 005302                          DEC     R2              ;R2=FULL WORD ADDRESS.

 012570 011237 001126                   MOV     (R2),$BDDAT     ;READ BUS ADDRESS.
 012574 000337 001124                   SWAB    $GDDAT          ;ADJUST DATA FOR HI BYTE.

 012600 023737 001126 001124            CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
```

```
                                        BEQ     R16             ;OKAY

                                        EMT     53

                                R16:    TST     (R5)            ;AT END OF TEST LIST.
                                        BNE     L16             ;NOPE
                                        BR      TST27           ;NEXT TEST
                                ;THIS LIST WILL BE USED TO LOAD THE UPPER BYTE OF THE BA REGISTER.
                                ;
                                LST16A: 2
                                        4
                                        10
                                        20
                                        40
                                        100
                                        200
                                        376
                                        374
                                        376
                                        366
                                        356
                                        336
                                        276
                                        176
                                        0

                                ;;*****************************************************************
                                ;*TEST 27        RH11 INTERRUPT TEST
                                ;*THIS TEST CAUSE AN RH11 INTERRUPT VIA THE SPECIAL COMMAND SEQUENCE
                                ;*'BIS  #IE,@RHCS1'. THIS TEST PROVES ONLY THAT THE HARDWARE CAN HANDLE
                                ;*INTERRUPTS PROPERLY
                                ;;*****************************************************************
 012660  000004                 TST27:  SCOPE
 012662  012737  000001  001212         MOV     #1,$TIMES       ;;DO 1 ITERATION
012670  012737  000027  003610         MOV     #27,TSTNM       ;MOVE #27 TO TEST NUMBER
012676  012777  000040  166332         MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER

012704  012706  001100                 MOV     #STACK,SP       ;SET STACK POINTER.

012710  013702  001240                 MOV     RHCS1,R2        ;R2=RH11 ADDRESS.
012714  010237  031620                 MOV     R2,REGADR
012720  005037  001126                 CLR     $BDDAT          ;$BDDAT=WAS.
012724  005037  001124                 CLR     $GDDAT          ;$GDDAT=S/B.

012730  017737  166272  001176         MOV     @RPVEC,$TMP0    ;SAVE RH11 INTERRUPT VECTOR.
012736  012777  012772  166262         MOV     #021,@RPVEC     ;SET RH11 INTERRUPT VECTOR TO 021.

012744  012706  001100         L21:    MOV     #STACK,SP       ;SET STACK POINTER.
012750  005037  177776                 CLR     PS

012754  052712  000100                 BIS     #IE,(R2)        ;CAUSE INTERRUPT.
012760  000240                         NOP                     ;WAIT FOR INTERRJPT.
012762  000240                         NOP
012764  011237  001124                 MOV     (R2),$GDDAT     ;SAVE CONTENTS OF REGISTER.

012770  104054                         EMT     54
```

```
    887 012772  013777  001176  166226  021:    MOV     $TMP0,@RPVEC    ;RESTORE RH11 INTERRUPT VECTOR.
    888
    893                                     ;;*****************************************************************
                                            ;*TEST 30        BCTD CLR L TEST
                                            ;*THE PROGRAM SETS THE PORT SELECT BIT .THEN DOES A RESET
                                            ;*IF THE SIGNAL CLR L WAS GENERATED THE PORT SELECT BIT WILL CLEAR.
                                            ;;*****************************************************************
        013000  000004                      TST30:  SCOPE
        013002  012737  000001  001212              MOV     #1,$TIMES       ;;DO 1 ITERATION
    894 013010  012737  000030  003610              MOV     #30,TSTNM       ;MOVE #30 TO TEST NUMBER
    895 013016  012777  000040  166212              MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
    896
    897 013024  013702  001240                      MOV     RHCS1,R2        ;R2=RH11 CS1 ADDRESS
    898 013030  010237  031620                      MOV     R2,REGADR
    899 013034  005037  001124                      CLR     $GDDAT          ;$GDDAT=S/B
    900
    901 013040  012706  001100              L22:    MOV     #STACK,SP       ;SET STACK POINTER.
    902 013044  012712  002000                      MOV     #PSEL,(R2)      ;SET PORT SELECT FLOP.
    903 013050  000005                              RESET                   ;DO A BUSA INIT.
    904
    905 013052  011237  001126                      MOV     (R2),$BDDAT     ;$BDDAT=WAS.
    906 013056  042737  177677  001126              BIC     #^C<IE>,$BDDAT  ;SAVE ONLY I.E. BIT.
    907
    908 013064  023737  001126  001124              CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
    909 013072  001401                              BEQ     TST31           ;NEXT TEST
    910 013074  104055                              EMT     55
    911
    915                                     ;;*****************************************************************
                                            ;*TEST 31        MXF TEST
                                            ;*SET THE MXF FLOP AND READ IT BACK.
                                            ;;*****************************************************************
        013076  000004                      TST31:  SCOPE
        013100  012737  000001  001212              MOV     #1,$TIMES       ;;DO 1 ITERATION
    916 013106  012737  000031  003610              MOV     #31,TSTNM       ;MOVE #31 TO TEST NUMBER
    917
                                            ;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

        013114  005737  003614                      TST     RH70            ;TEST FLAG FOR RH70 CONTROLLER
        013120  001402                              BEQ     64$             ;IF FLAG = 1, THIS TEST IS SKIPPED
        013122  000137  013206                      JMP     TST32           ;JUMP TO NEXT TEST
        013126                              64$:                            ;IF FLAG = 0, DO THIS TEST
    918 013126  012777  000040  166102              MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
    919
    920 013134  012706  001100                      MOV     #STACK,SP       ;SET STACK POINTER.
    921 013140  013702  001236                      MOV     RHCS2,R2        ;R2=RH11 CS2 ADDRESS.
    922 013144  010237  031620                      MOV     R2,REGADR
    923 013150  012737  001000  001124              MOV     #MXF,$GDDAT     ;$GDDAT=S/B.
    924
    925 013156  012712  001000              I24:    MOV     #MXF,(R2)       ;LOAD MXF
    926
    927 013162  011237  001126                      MOV     (R2),$BDDAT     ;$BDDAT=WAS.
    928
    929 013166  042737  176777  001126              BIC     #^C<MXF>,$BDDAT ;SAVE ONLY MXF
    930
    931 013174  023737  001126  001124              CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS
    932 013202  001401                              BEQ     TST32           ;NEXT TEST
    933 013204  104056                              EMT     56
```

```
                                ;;*****************************************************************
                                ;*TEST 32        CSRB UNIBUS PARITY ERROR SET TEST
                                ;*SET THE UNIBUS PARITY ERROR FLOP DIRECTLY VIA A MOV #UPE TO RHCS2
                                ;*ATTEMPT TO READ IT BACK.
                                ;;*****************************************************************
013206  000004                  TST32:  SCOPE
013210  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
940 013216  012737  000032  003610      MOV     #32,TSTNM       ;MOVE #32 TO TEST NUMBER
941
                                ;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

013224  005737  003614                  TST     RH70            ;TEST FLAG FOR RH70 CONTROLLER
013230  001402                  64$:    BEQ     64$             ;IF FLAG = 1, THIS TEST IS SKIPPED
013232  000137  013322                  JMP     TST33           ;JUMP TO NEXT TEST
013236                                                          ;IF FLAG = 0, DO THIS TEST
942
943 013236  012777  000040  165772      MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
944
945 013244  012706  001100              MOV     #STACK,SP       ;SET STACK POINTER.
946
947 013250  013702  001236              MOV     RHCS2,R2        ;R2=RHCS2 ADDRESS.
948 013254  010237  031620              MOV     R2,REGADR
949 013260  012737  020000  001124      MOV     #UPE,$GDDAT     ;$GDDAT=S/B.
950
951 013266  013712  001124      L30:    MOV     $GDDAT,(R2)     ;ATTEMPT TO SET UPE.
952 013272  000240                      NOP
953 013274  000240                      NOP
954
955 013276  011237  001126              MOV     (R2),$BDDAT     ;$BDDAT=ACTUAL DATA.
956 013302  042737  157777  001126      BIC     #^C<UPE>,$BDDAT ;SAVE ONLY UPE.
957
958 013310  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
959 013316  001401                      BEQ     TST33           ;NEXT TEST
960 013320  104057                      EMT     57
961
966                             ;;*****************************************************************
                                ;*TEST 33        CSRB UNIBUS PARITY ERROR CLEAR TEST
                                ;*SET THE UNIBUS PARITY ERROR FLOP AND ATTEMPT TO CLEAR IT WITH A
                                ;*CONTROLLER CLEAR.
                                ;;*****************************************************************
013322  000004                  TST33:  SCOPE
013324  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
967 013332  012737  000033  003610      MOV     #33,TSTNM       ;MOVE #33 TO TEST NUMBER
968 013340  012777  000040  165670      MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
969
970 013346  012706  001100              MOV     #STACK,SP       ;SET STACK POINTER.
971
972 013352  013702  001236              MOV     RHCS2,R2        ;R2=RHCS2 ADDRESS.
973 013356  010237  031620              MOV     R2,REGADR
974 013362  005037  001124              CLR     $GDDAT          ;$GDDAT=S/B.
975
976 013366  012712  020000      L31:    MOV     #UPE,(R2)       ;SET UNIBUS PARITY ERROR SET.
977 013372  000240                      NOP
978 013374  000240                      NOP
979
980 013376  012712  000040              MOV     #CLR,(R2)       ;CONTROLLER CLEAR.
```

```
013402  011237  001126                  MOV     (R2),$BDDAT      ;READ STATUS
013406  042737  157777  001126          BIC     #^C<UPE>,$BDDAT ;SAVE ERROR.

013414  023737  001126  001124          CMP     $BDDAT,$GDDAT    ;COMPARE RESULTS.
013422  001401                          BEQ     TST34            ;NEXT TEST
013424  104057                          EMT     57

                                ;;*****************************************************************
                                ;*TEST 34        CSRB UNIT SELECT CLEAR TEST
                                ;*SET THE UNIT SELECT REGISTER TO DRIVE #7 ALL BITS SET. GENERATE A
                                ;*CONTROLLER CLEAR AND VERIFY THAT THE UNIT SELECT REGISTER IS CLEARED.
                                ;;*****************************************************************
013426  000004                  TST34:  SCOPE
013430  012737  000001  001212          MOV     #1,$TIMES        ;;DO 1 ITERATION
013436  012737  000034  003610          MOV     #34,TSTNM        ;MOVE #34 TO TEST NUMBER
013444  012777  000040  165564          MOV     #CLR,@RHCS2      ;CLEAR RH11 CONTROLLER

013452  012706  001100                  MOV     #STACK,SP        ;SET STACK POINTER.

013456  013702  001236                  MOV     RHCS2,R2         ;R2=CS2 ADDRESS.
013462  010237  031620                  MOV     R2,REGADR
013466  005037  001124                  CLR     $GDDAT           ;$GDDAT=S/B.

013472  012712  000007          L34:    MOV     #US4!US2!US1,(R2)  ;LOAD RHCS2 DRIVE SELECT
013476  000240                          NOP
013500  000240                          NOP

013502  012712  000040                  MOV     #CLR,(R2)        ;GENERATE CLR.

013506  011237  001126                  MOV     (R2),$BDDAT      ;READ RHCS2
013512  042737  177770  001126          BIC     #^C<US4!US2!US1>,$BDDAT;SAVE DRIVE SELECT BITS.

013520  023737  001126  001124          CMP     $BDDAT,$GDDAT    ;COMPARE RESULTS.
013526  001401                          BEQ     TST35            ;NEXT TEST
013530  104060                          EMT     60

                                ;;*****************************************************************
                                ;*TEST 35        CSRB TRANSFER ERROR (TRE) - UPE
                                ;*SET THE UPE FLOP UNIBUS PARITY ERROR. VERIFY THAT THE SETTING OF (UPE)
                                ;*ALSO SETS THE TRANSFER ERROR (TRE) FLOP
                                ;;*****************************************************************
013532  000004                  TST35:  SCOPE
013534  012737  000001  001212          MOV     #1,$TIMES        ;;DO 1 ITERATION
013542  012737  000035  003610          MOV     #35,TSTNM        ;MOVE #35 TO TEST NUMBER

                                ;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

013550  005737  003614                  TST     RH70             ;TEST FLAG FOR RH70 CONTROLLER
013554  001402                          BEQ     64$              ;IF FLAG = 1, THIS TEST IS SKIPPED
013556  000137  013646                  JMP     TST36            ;JUMP TO NEXT TEST
013562                          64$:                             ;IF FLAG = 0, DO THIS TEST

013562  012777  000040  165446          MOV     #CLR,@RHCS2      ;CLEAR RH11 CONTROLLER

013570  012706  001100                  MOV     #STACK,SP        ;SET STACK POINTER.

```

```
1028 013574  013702  001236                MOV     RHCS2,R2        ;R2=RHCS2 ADDRESS.
1029 013600  012737  040000  001124        MOV     #TRE,$GDDAT     ;$GDDAT=S/B.
1030
1031 013606  012712  020000          L35:  MOV     #UPE,(R2)       ;SET UNIBUS PARITY ERROR.
1032
1033 013612  013702  001240                MOV     RHCS1,R2        ;R2=RHCS1 ADDRESS.
1034 013616  010237  031620                MOV     R2,REGADR
1035
1036 013622  011237  001126                MOV     (R2),$BDDAT     ;READ REGISTER
1037 013626  042737  137777  001126        BIC     #^C<TRE>,$BDDAT ;SAVE TRANSFER ERROR.
1038
1039 013634  023737  001126  001124        CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS
1040 013642  001401                        BEQ     TST36           ;NEXT TEST
1041 013644  104061                        EMT     61
1042
1046                                ;;*****************************************************************
                                    ;*TEST 36       CSRB TRANSFER ERROR (TRE) NED
                                    ;*SET THE NED BIT NON EXISTANT DRIVE VERIFY THAT THIS SETS THE (TRE) BIT.
                                    ;;*****************************************************************
     013646  000004                 TST36:  SCOPE
     013650  012737  000001  001212        MOV     #1,$TIMES       ;;DO 1 ITERATION
1047 013656  012737  000036  003610        MOV     #36,TSTNM       ;MOVE #36 TO TEST NUMBER
1048 013664  012777  000040  165344        MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
1049
1050 013672  012706  001100                MOV     #STACK,SP       ;SET STACK POINTER.
1051
1052 013676  013702  001236                MOV     RHCS2,R2        ;R2=CS2 ADDRESS.
1053 013702  012737  040000  001124        MOV     #TRE,$GDDAT     ;$GDDAT=S/B.
1054
1055 013710  013700  001420                MOV     TOTALAT,R0      ;GET ALL AVAILABLE DRIVES DATA
1056 013714  005001                        CLR     R1
1057 013716  006000                 S36:   ROR     R0
1058 013720  103420                        BCS     N36
1059 013722  010112                 L36:   MOV     R1,(R2)           ;SET NED.
1060
1061 013724  013702  001240                MOV     RHCS1,R2        ;R2=RHCS1 ADDRESS.
1062 013730  010237  031620                MOV     R2,REGADR
1063
1064 013734  011237  001126                MOV     (R2),$BDDAT     ;READ REGISTER.
1065 013740  042737  137777  001126        BIC     #^C<TRE>,$BDDAT ;SAVE TRE.
1066
1067 013746  023737  001126  001124        CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
1068 013754  001406                        BEQ     TST37           ;NEXT TEST
1069 013756  104062                        EMT     62
1070 013760  000404                        BR      TST37           ;NEXT TEST
1071
1072 013762  005201                 N36:   INC     R1
1073 013764  020127  000010                CMP     R1,#8.
1074 013770  001352                        BNE     S36
1075
1079                                ;;*****************************************************************
                                    ;*TEST 37       CSRA PSEL CLEAR TEST
                                    ;*SET THE PORT SELECT FLOP VERIFY THAT WE CAN READ IT BACK.
                                    ;;*****************************************************************
     013772  000004                 TST37:  SCOPE
     013774  012737  000001  001212        MOV     #1,$TIMES       ;;DO 1 ITERATION
1080 014002  012737  000037  003610        MOV     #37,TSTNM       ;MOVE #37 TO TEST NUMBER
```

```
 1081 014010  012777  000040  165220          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
 1082
 1083 014016  012706  001100                  MOV     #STACK,SP       ;SET STACK POINTER.
 1084
 1085 014022  013702  001240                  MOV     RHCS1,R2        ;R2=RHCS1 ADDRESS.
 1086 014026  010237  031620                  MOV     R2,REGADR
 1087 014032  005037  001124                  CLR     $GDDAT          ;$GDDAT=S/B.
 1088
 1089 014036  012712  002000          L40:    MOV     #PSEL,(R2)      ;SET P SELECT.
 1090
 1091 014042  012777  000040  165166          MOV     #CLR,@RHCS2     ;DO A CONTROLLER CLEAR.
 1092
 1093 014050  011237  001126                  MOV     (R2),$BDDAT     ;READ BACK P SELECT BIT.
 1094 014054  042737  175777  001126          BIC     #^C<PSEL>,$BDDAT;SAVE ONLY PORT SELECT.
 1095
 1096 014062  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
 1097 014070  001401                          BEQ     TST40           ;NEXT TEST
 1098 014072  104064                          EMT     64
 1099
 1103                                 ;;*****************************************************************
                                      ;*TEST 40       CSRB COMMAND REGISTER CLEAR TEST
                                      ;*VERIFY THAT CONTROLLER CLEAR WILL CLEAR THE COMMAND REGISTER.
                                      ;;*****************************************************************
      014074  000004                  TST40:  SCOPE
      014076  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
 1104 014104  012737  000040  003610          MOV     #40,TSTNM       ;MOVE #40 TO TEST NUMBER
 1105 014112  012777  000040  165116          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
 1106
 1107 014120  012706  001100                  MOV     #STACK,SP       ;SET STACK POINTER.
 1108
 1109 014124  013702  001240                  MOV     RHCS1,R2        ;R2=RHCS1 ADDRESS.
 1110 014130  010237  031620                  MOV     R2,REGADR
 1111 014134  005037  001124                  CLR     $GDDAT          ;$GDDAT=S/B.
 1112
 1113 014140  012712  000011          L41:    MOV     #11,(R2)        ;ISSUE A DRIVE CLR AND GO.
 1114
 1115 014144  012777  000040  165064          MOV     #CLR,@RHCS2     ;CONTROLLER CLEAR.
 1116
 1117 014152  011237  001126                  MOV     (R2),$BDDAT     ;READ COMMAND REGISTER.
 1118 014156  042737  177770  001126          BIC     #^C<7>,$BDDAT   ;SAVE ONLY THE COMMAND BITS.
 1119
 1120 014164  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
 1121 014172  001401                          BEQ     TST41           ;NEXT TEST
 1122 014174  104065                          EMT     65
 1123
 1131                                 ;;*****************************************************************
                                      ;*TEST 41       CSRB COMMAND REGISTER RESELECT CLEAR TEST
                                      ;*THE COMMAND REGISTER IS SUPPOSED TO BE SELF CLEARING THAT IS WHEN THE
                                      ;*RH11 IS SELECTED WHEATHER OR NOT WE ADDRESS IT DIRECTLY THE LOGIC
                                      ;*SHOULD CLEAR THE COMMAND REGISTER.
                                      ;*THIS IS TESTED BY LOADING A COMMAND INTO IT READING ANOTHER REGISTER
                                      ;*THAN RETURNING TO CHECK THE COMMAND REGISTER TO DETERMINE IF IT CLEARED.
                                      ;;*****************************************************************
      014176  000004                  TST41:  SCOPE
      014200  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
 1132 014206  012737  000041  003610          MOV     #41,TSTNM       ;MOVE #41 TO TEST NUMBER
 1133 014214  012777  000040  165014          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
```

```
1134
1135 014222  012706  001100                  MOV     #STACK,SP       ;SET STACK POINTER.
1136
1137 014226  013702  001240                  MOV     RHCS1,R2        ;R2=RHCS1 ADDRESS.
1138 014232  010237  031620                  MOV     R2,REGADR
1139 014236  005037  001124                  CLR     $GDDAT          ;$GDDAT=S/B.
1140
1141 014242  012737  000340  177776          MOV     #PR7,PS         ;SET PRIORITY TO #7.
1142
1143 014250  012712  000011          L42:    MOV     #11,(R2)        ;ISSUE A DRIVE CLR AND GO
1144
1145 014254  011237  001176                  MOV     (R2),$TMP0      ;DO A RESELECT OF THE REGISTER.
1146
1147 014260  011237  001126                  MOV     (R2),$BDDAT     ;READ THE COMMAND REGISTER.
1148 014264  042737  177770  001126          BIC     #^C<7>,$BDDAT   ;SAVE ONLY THE COMMAND BITS.
1149
1150 014272  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
1151 014300  001401                          BEQ     TST42           ;NEXT TEST
1152 014302  104066                          EMT     66
1153
1157                                ;;*****************************************************************
                                    ;*TEST 42       CSRB READY AND IE INTERRUPT TEST
                                    ;*HERE WE TEST TO SEE IF SETTING (IE) AND (RDY) WILL CAUSE AN INTERRUPT.
                                    ;;*****************************************************************
     014304  000004                 TST42:  SCOPE
     014306  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
1158 014314  012737  000042  003610          MOV     #42,TSTNM       ;MOVE #42 TO TEST NUMBER
1159 014322  012777  000040  164706          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
1160
1161 014330  013702  001240                  MOV     RHCS1,R2        ;R2=RPCS1 ADDRESS.
1162 014334  010237  031620                  MOV     R2,REGADR
1163 014340  005037  001126                  CLR     $BDDAT          ;$BDDAT=WAS.
1164 014344  005037  001124                  CLR     $GDDAT          ;$GDDAT=S/B.
1165
1166 014350  017737  164652  001176          MOV     @RPVEC,$TMP0    ;SAVE RH11 INTERRUPT VECTOR.
1167 014356  012777  014412  164642          MOV     #043,@RPVEC     ;SET RH11 INTERRUPT VECTOR 043
1168
1169 014364  012706  001100          L43:    MOV     #STACK,SP       ;SET STACK POINTER.
1170 014370  005037  177776                  CLR     PS
1171
1172 014374  052712  000300                  BIS     #IE!RDY,(R2)    ;THIS WILL CAUSE AN INTERRUPT.
1173 014400  000240                          NOP
1174 014402  000240                          NOP
1175
1176 014404  011237  001124                  MOV     (R2),$GDDAT
1177 014410  104067                          EMT     67
1178
1179 014412  013777  001176  164606  043:    MOV     $TMP0,@RPVEC    ;RESTORE RH11 INTERRUPT VECTOR.
1180
1185                                ;;*****************************************************************
                                    ;*TEST 43       CSRB BOES "IE" CLEAR AFTER AN INTERRUPT
                                    ;*THE (IE) BIT IS SET CAUSING AN INTERRUPT. AT THE COMPLETION OF THE INTERRUPT
                                    ;*WE CHECK TO SEE IF THE (IE) BIT HAS CLEARED.
                                    ;;*****************************************************************
     014420  000004                 TST43:  SCOPE
     014422  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
1186 014430  012737  000043  003610          MOV     #43,TSTNM       ;MOVE #43 TO TEST NUMBER
```

```
1187 014436  012777  000040  164572          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
1188
1189 014444  013702  001240                  MOV     RHCS1,R2        ;R2=RPCS1 ADDRESS.
1190 014450  010237  031620                  MOV     R2,REGADR
1191 014454  005037  001124                  CLR     $GDDAT          ;$GDDAT=S/B.
1192
1193 014460  017737  164542  001176          MOV     @RPVEC,$TMP0    ;SAVE RH11 INTERRUPT VECTOR.
1194 014466  012777  014516  164532          MOV     #044,@RPVEC     ;SET VECTOR TO 044.
1195
1196 014474  012706  001100          L44:    MOV     #STACK,SP       ;SET STACK POINTER.
1197 014500  005037  177776                  CLR     PS
1198
1199 014504  052712  000100                  BIS     #IE,(R2)        ;THIS WILL CAUSE AN INTERRUPT
1200 014510  000240                          NOP
1201 014512  000240                          NOP
1202
1203 014514  000411                          BR      E44             ;NO INTERRUPT.
1204
1205 014516  011237  001126          044:    MOV     (R2),$BDDAT     ;READ RHCS1.
1206 014522  042737  177677  001126          BIC     #^C<IE>,$BDDAT  ;SAVE ONLY THE "IE" BIT.
1207
1208 014530  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
1209 014536  001401                          BEQ     R44             ;OK!
1210
1211 014540                          E44:
     014540  104070                          EM      70
1212
1213 014542  013777  001176  164456  R44:    MOV     $TMP0,@RPVEC    ;RESTORE RH11 INTERRUPT VECTOR.
1214
1219                                 ;;*******  **********************************************************
                                     ;*TEST 44       BCTB "AS" WRITE TEST
                                     ;*HERE WE VERIFY THAT WRITING INTO THE "AS" REGISTER WILL NOT CAUSE
                                     ;*AN (NED) ERROR.
                                     ;;**********************************************************************
     014550  000004                  TST44:  SCOPE
     014552  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
1220 014560  012737  000044  003610          MOV     #44,TSTNM       ;MOVE #44 TO TEST NUMBER
1221 014566  012777  000040  164442          MOV     #CLR,@RHCS2     ;CLEAR RH11 CONTROLLER
1222
1223 014574  012706  001100                  MOV     #STACK,SP       ;SET STACK POINTER.
1224
1225 014600  013702  001236                  MOV     RHCS2,R2        ;R2=RH11 CS2 ADDRESS.
1226 014604  012737  000007  001124          MOV     #US4!US2!US1,$GDDAT;$GDDAT=S/B.
1227
1228 014612  013705  001240                  MOV     RHCS1,R5        ;ADDRESS OF A DEVICE REGISTER.
1229
1230 014616  012712  000007          L45:    MOV     #US4!US2!US1,(R2);LOAD A NON EXISTANT DEVICE.
1231
1232 014622  013702  001256                  MOV     RHAS,R2         ;R2="AS" ADDRESS.
1233
1234 014626  012712  000377                  MOV     #377,(R2)       ;ATTEMPT TO LOAD "AS".
1235
1236 014632  005012                          CLR     (R2)            ;CLEAR  "AS".
1237
1238 014634  013702  001236                  MOV     RHCS2,R2        ;R2=RH11 CS2 ADDRESS.
1239 014640  010237  031620                  MOV     R2,REGADR
1240
```

```
1241 014644  011237  001126                MOV     (R2),$BDDAT     ;$BDDAT=WAS.
1242 014650  042737  167770  001126        BIC     #^C<NED!US4!US2!US1>,$BDDAT;SAVE NED AND DEVICE SELECTED.
1243
1244 014656  023737  001126  001124        CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
1245 014664  001401                        BEQ     TST45           ;NEXT TEST
1246 014666  104071                        EMT     71
1247
1248                                ;;*****************************************************************
                                    ;*TEST 45       MAKE CURRENT CYLINDER = 0
                                    ;;*****************************************************************
     014670  000004                TST45:  SCOPE
     014672  012706  001100                MOV     #STACK,SP       ;RESET STACK
     014676  012737  000045  003610        MOV     #45,TSTNM       ;MOVE #45 TO TEST NUMBER
     014704  004737  032054                JSR     PC,CLDISK       ;INIT DRIVE
     014710  012777  000001  164342        MOV     #DMD,@RHMR      ;SET DIAGNOSTIC MODE
     014716  004037  034434                JSR     R0,MAKECYL      ;DO SEEK COMMAND FOLLOWED BY AN INIT TO
     014722  000000                        .WORD   0               ;CHANGE RHCC TO CYLINDER 0
1249
1256                                ;;*****************************************************************
                                    ;*TEST 46       RHCS1 - BITS 8 AND 9 - EXTENDED ADDR (A16 & A 17)
                                    ;*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
                                    ;;*****************************************************************
     014724  000004                TST46:  SCOPE
1257 014726  012706  001100                MOV     #STACK,SP       ;RESET STACK
1258 014732  012737  000046  003610        MOV     #46,TSTNM       ;MOVE #46 TO TEST NUMBER
1259
                                    ;CHECK TO SEE IF THE PROGRAM IS RUNNING WITH AN RH70

     014740  005737  003614                TST     RH70            ;TEST FLAG FOR RH70 CONTROLLER
     014744  001402                        BEQ     64$             ;IF FLAG = 1, THIS TEST IS SKIPPED
     014746  000137  015256                JMP     TST47           ;JUMP TO NEXT TEST
     014752                         64$:                           ;IF FLAG = 0, DO THIS TEST
1260
1261 014752  004037  031772                JSR     R0,CLAREA       ;CLEAR SIMULATED DISK
1262 014756  040626                        .WORD   DISK            ;FROM
1263 014760  041652                        .WORD   TOLGAP+16       ;TO
1264 014762  000000                        .WORD   0               ;DATA
1265
1266                                ;THESE ARE SETUP FOR DISKLESS USE ONLY
1267
1268 014764  012737  010000  036710        MOV     #FMT22,CYL      ;CYLINDER 0
1269                                                                ;16 BITS PER WORD
1270 014772  005037  036712                CLR     SECOTR          ;SECTOR 0 TRACK 0
1271 014776  005037  036714                CLR     KEY1            ;KEY1 0
1272 015002  005037  036716                CLR     KEY2            ;KEY2 0
1273 015006  012737  000400  036756        MOV     #256.,NOWORD    ;NO OF DATA WORDS
1274 015014  012737  000001  036720        MOV     #1,X            ;WRITE DATA
1275 015022  004537  033270                JSR     R5,CRC          ;GO TO CALCULATE CRC
1276 015026  036710                        CYL
1277 015030  040610                        WCRC
1278
1279                                ;THESE ARE REGULAR SETUPS
1280
1281 015032  004737  032054                JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
```

```
1282 015036  012777  177400  164166          MOV     #-256.,@RHWC    ;256 DATA WORDS
1283 015044  013777  001144  164162          MOV     $TKS,@RHBA      ;STARTING ADDRESS OF WRITE BUFFER
1284 015052  017737  164066  001422          MOV     @$TKS,TMPILL    ;TEMPORARY STORAGE OF DATA
1285 015060  005077  164160                  CLR     @RHDST          ;SECTOR 0 TRACK 0
1286 015064  012777  010000  164156          MOV     #FMT22,@RHOF    ;16 BITS PER WORD FORMAT
1287 015072  005077  164154                  CLR     @RHCA           ;CYLINDER 0
1288 015076  004737  032116                  JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
     015102  104401  053407                  TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                     ;TINUE TESTING IF BOTH AREN'T TRUE
     015106  000000                          HALT                    ;STOP THE TEST
1289 015110  013746  001452                  MOV     WRIDAT,-(SP)    ;WRITE DATA=60
1290 015114  052716  001400                  BIS     #A16!A17,(SP)   ;SET HIGH ORDER UNIBUS BITS
1291 015120  012611                          MOV     (SP)+,@R1       ;FILL RHCS1
1292 015122  052777  000010  164106          BIS     #BAI,@RHCS2     ;SET BUS ADDRESS INHIBIT
1293 015130  005037  001406                  CLR     ERFLG$          ;CLEAR ERROR FLAG
1294 015134  004737  036550                  JSR     PC,COMHD        ;WRITE DATA
1295
1296                                 ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
1297                                 ;FROM THE "COMHD" ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
1298                                 ;HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY
1299                                 ;AND SYNCS WERE CORRECTLY DETECTED.
1300                                 ;DATA IS TO BE CHECKED.
1301
1302 015140  004737  031554                  JSR     PC,PUTREG       ;SAVE REGISTERS
1303 015144  005737  001406                  TST     ERFLG$          ;HAVE ANY ERRORS OCCURED?
1304 015150  001042                          BNE     TST47           ;BRANCH OUT IF YES
1305 015152  013700  001422                  MOV     TMPILL,R0       ;GOOD DATA
1306 015156  012701  040626                  MOV     #DISK,R1        ;DATA WRITTEN INTO "DISK"
1307 015162  012702  000400                  MOV     #256.,R2        ;COUNTER
1308 015166  012737  000401  037030  1$:     MOV     #257.,ERWORD    ;FOR ERROR WORD
1309 015174  020021                          CMP     R0,(R1)+        ;COMPARE GOOD DATA WITH DATA ON DISK
1310 015176  001425                          BEQ     3$              ;BRANCH IF GOOD
1311 015200  013737  001422  001124          MOV     TMPILL,$GDDAT   ;GOOD DATA
1312 015206  014137  001126                  MOV     -(R1),$BDDAT    ;BAD DATA
1313 015212  160237  037030                  SUB     R2,ERWORD       ;ERROR WORD NO
1314 015216  005737  001406                  TST     ERFLG$          ;ANY ERRORS ALREADY THERE?
1315 015222  001002                          BNE     2$              ;BRANCH IF YES
1316 015224  104037                          EMT     37
1317                                                                 ;SEE NEXT ERROR COMMENTS
1318 015226  000401                          BR      1064$           ;BRANCH TO AVOID PRINTING NEXT ERROR
1319 015230                          2$:
     015230  104040                          EMT     40
1320                                                                 ;ERROR OCCURED WHILE WRITING
1321                                                                 ;WITH A16 A17 OF RHCS1 SET
1322 015232  005721                  1064$:  TST     (R1)+           ;UNDO -(R1) FOR BAD DATA
1323 015234  017746  163700                  MOV     @SWR,-(SP)      ;GET SWITCH SETTING
1324 015240  042716  177177                  BIC     #177177,(SP)    ;KEEP ONLY SWITCH 7 AND 8
1325 015244  022726  000200                  CMP     #SW07,(SP)+     ;IS 7 SET AND 8 RESET
1326 015250  001402                          BEQ     TST47           ;BRANCH OUT IF YES
1327 015252  005302                  3$:     DEC     R2              ;IF NOT COUNT 256 WORDS
1328 015254  001344                          BNE     1$              ;BRANCH IF 256 NOT DONE
1329
1341                                 ;;*****************************************************************
                                     ;*TEST 47       DRIVE TIMING ERROR
                                     ;*A READ HEADER AND DATA IS STARTED ON CYLINDER 0, SECTOR
                                     ;*0, TRACK 0, 260 WORDS.  AFTER THE HEADER IS READ IN CORRECTLY,
                                     ;*NO SYNC BYTE (DATA SYNC) IS GIVEN.
```

T47 DRIVE TIMING ERROR

```
                                        ;*THEN NORMAL DIAGNOSTIC DATA 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 WHICH SHOULD
                                        ;*SET 'DRIVE TIMING ERROR'' - 'DTE'
                                        ;;*****************************************************************
     015256  000004                     TST47:  SCOPE
1342 015260  012706  001100                     MOV     #STACK,SP       ;RESET STACK
1343 015264  012737  000047  003610             MOV     #47,TSTNM       ;MOVE #47 TO TEST NUMBER
1344
1345                                    ;THESE ARE TO SETUP FOR DISKLESS USE
1346
1347 015272  012737  010000  036710             MOV     #FMT22,CYL      ;16 BITS PER WORD
1348                                                                     ;CYLINDER 0
1349 015300  005037  036712                     CLR     SECOTR  ;SECTOR 0
1350                                                                     ;TRACK 0
1351 015304  005037  036714                     CLR     KEY1            ;KEY1 = 0
1352 015310  005037  036716                     CLR     KEY2            ;KEY2 = 0
1353 015314  005037  036720                     CLR     X               ;THIS IS A READ COMMAND
1354 015320  004537  033270                     JSR     R5,CRC  ;GO TO CALCULATE CRC
1355 015324  036710                             CYL
1356 015326  040610                             WCRC
1357
1358                                    ; THESE ARE REGULAR SETUPS
1359
1360 015330  004737  032054                     JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
1361 015334  012777  177374  163670             MOV     #-260.,@RHWC    ;256 DATA WORDS, 4 HEADER WORDS
1362 015342  012777  002544  163664             MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF BUFFER
1363 015350  005077  163670                     CLR     @RHDST          ;TRACK = 0
1364                                                                     ;SECTOR = 0
1365 015354  012777  014000  163666             MOV     #FMT22!ECI,@RHOF ;16 BITS PER WORD
1366                                                                     ;ECC CORRECTION INHIBITED
1367 015362  005077  163664                     CLR     @RHCA           ;CYLINDER = 0
1368 015366  004737  032116                     JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
     015372  104401  053407                     TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                     ;TINUE TESTING IF BOTH AREN'T TRUE
     015376  000000                             HALT                    ;STOP THE TEST
1369 015400  013711  001460                     MOV     REFOR,@R1       ;READ HEADER AND DATA = 72
1370
1371                                    ;READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'
1372
1373 015404  004037  032554                     JSR     R0,SAVER        ;READ IN SEQUENCE
1374 015410  001232                             RHWC                    ;FROM HARDWARE REGISTER
1375 015412  001306                             WC                      ;INTO CORE AT LOCATION
1376 015414  000023                             19.                     ;NUMBER OF REGISTERS TO READ
1377
1378
1379                                    ;NOW 'GO' WILL BE GIVEN, EVERYTHING WILL BE TREATED
1380                                    ;NORMALLY FOR THE HEADER. BUT WHEN IT IS TIME TO READ
1381                                    ;DATA, ONLY SECTOR CLOCKS WILL BE GIVEN. NO DIAGNOSTIC DATA
1382                                    ;CLOCKS WILL BE GIVEN. THIS SHOULD BRING SECTOR PULSE HIGH
1383                                    ;WITHOUT PUTTING 'READ'' DOWN HENCE 'DTE' WILL COME UP.
1384
1335 015416  012737  177777  001426             MOV     #-1,TESDTE      ;SET DTE TEST
1386 015424  012737  177777  001406             MOV     #-1,ERFLG$      ;THIS WILL BRING THE READ HEADER
1387                                                                     ;AND DATA PROCESS OUT AFTER THE
```

```
                                                                        ;HEADER HAS BEEN CORRECTLY READ
                                                                        ;
015432  004737  036550                  JSR     PC,COMHD              ;ISSUE 'GO', SEARCH FOR THE SECTOR
                                                                        ;AND READ THE HEADER
                                                                        ;
015436  017737  163576  001176  SETCK1: MOV     @RHCS1,$TMP0          ;READ CS1 TO CHECK FOR ANY READ ERRORS
015444  032737  100000  001176          BIT     #SC,$TMP0             ;TEST FOR "SPECIAL CONDITION" - 'SC'
015452  001405                        BEQ     8$                    ;CONTINUE WITH TEST IF 'SC' = 0 (NO ERRORS)
015454  004737  031554                JSR     PC,PUTREG             ;READ & SAVE ALL REGISTERS AGAIN IF AN
                                                                        ;UNDEFINED DATA TRANSFER ERROR OCCURRED
015460  104040                        EMT     40
015462  000137  015750                JMP     TST50                 ;'DTE' TEST - ABORT THE TEST
                                                                        ;
                                ;NOW THE HEADER HAS BEEN READ OK
                                ;NOW 560 SECTOR CLOCKS WILL BE GIVEN
                                ;GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
                                ;GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 2
                                ;
                                ;THESE 560 SECTOR CLOCKS ARE DIVIDED INTO TWO GROUPS
                                ;24 SECTOR CLOCKS WITH NORMAL DIAGNOSTIC DATA CLOCKS,
                                ;AND 536 SECTOR CLOCKS WITHOUT ANY DIAGNOSTIC DATA CLOCKS.
                                ;
                                ;THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC DATA CLOCKS
                                ;WHICH EQUALS 3 BYTES OF DATA
                                ;
015466  012701  000030          8$:     MOV     #24.,R1               ;LOAD COUNTER
015472  013700  001260                  MOV     RHMR,R0 ;GET RHMR ADDRESS
015476  012710  000001                  MOV     #DMD,@R0              ;SET DIAGNOSTIC MODE
015502  052710  000012          1$:     BIS     #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND DATA CLOCK
015506  042710  000012                  BIC     #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND DATA CLOCK
015512  012702  000007                  MOV     #7,R2                 ;LOAD COUNTER FOR DIAGNOSTIC CLOCKS
015516  052710  000002          4$:     BIS     #MCLK,@R0             ;SET CLOCK
015522  042710  000002                  BIC     #MCLK,@R0             ;CLEAR CLOCK
015526  005302                        DEC     R2                    ;COUNT TO 7
015530  001372                        BNE     4$                    ;BRANCH IF 7 NOT DONE
015532  005301                        DEC     R1                    ;COUNT TO 24
015534  001362                        BNE     1$                    ;BRANCH IF 24 NOT DONE
                                ;
                                ;THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
                                ;
015536  012701  001030                  MOV     #536.,R1              ;LOAD SECTOR CLOCK COUNTER
015542  052710  000010          5$:     BIS     #MSTCK,@R0            ;SET SECTOR CLOCK
015546  042710  000010                  BIC     #MSTCK,@R0            ;CLEAR SECTOR CLOCK
015552  005301                        DEC     R1                    ;COUNT
015554  001372                        BNE     5$                    ;BRANCH IF 536 NOT DONE
                                ;
                                ;
                                ;NOW 'DTE' SHOULD BE SET
                                ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
                                ;
015556  012737  177400  001306          MOV     #-256.,WC             ;SAVED RHWC
015564  012737  002554  001310          MOV     #REINTO+<4.*2>,BA ;SAVED RHBA
015572  052737  140000  001314          BIS     #SC!TRE,CS1           ;SAVED RHCS1
015600  052737  010000  001316          BIS     #DTE,ER1              ;SAVED RHER1
015606  012737  000401  001334          MOV     #401,MR ;SAVED RHMR
015614  052737  140000  001336          BIS     #ATA!ERR,DS1 ;SAVED RHDS1
015622  012737  000100  001350          MOV     #100,LA ;SAVED RHLA
```

```
1445 015630  012737  000001  001320          MOV     #1,DST  ;SAVED RHDST
1446 015636  013737  001416  001332          MOV     ATTENT,AS       ;SAVED RHAS
1447
1448
1449                                 ;NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS
1450                                 ;CAN BE DONE (USE THE 'WRFROM' SAVE BUFFER THIS TIME)
1451
1452 015644  004037  032554                  JSR     R0,SAVER        ;READ IN SEQUENCE
1453 015650  001232                          RHWC                    ;FROM HARDWARE REGISTER
1454 015652  001500                          WRFROM                  ;INTO CORE BUFFER
1455 015654  000023                          19.                     ;NUMBER OF REGISTERS TO READ
1456
1457                                 ;FOR RHAS UPPER BYTE
1458 015656  113737  001333  001525          MOVB    AS+1,WRFROM+25 ;UPPER RHAS
1459
1460                                 ;COMPARE THE HEADER READ
1461
1462 015664  004037  032756                  JSR     R0,COMFAR       ;COMPARE
1463 015670  036710                          CYL                     ;GOOD BUFFER
1464 015672  002544                          REINTO                  ;TEST BUFFER
1465 015674  000004                          4.                      ;NUMBER
1466 015676  015704                          6$                      ;RETURN FOR ERROR
1467 015700  015704                          6$                      ;SAME
1468 015702  015710                          7$                      ;RETURN FOR GOOD COMPARISON
1469 015704                          6$:
     015704  104010                          EMT     10
1470                                                                 ;IN ERROR
1471
1472 015706  000207                          RTS     PC              ;RETURN
1473
1474 015710                          7$:                             ;GOOD COMPARISON, CONTINUE
1475
1476
1477                                 ;COMPARE REGISTERS BEFORE COMMAND WITH REGISTERS AFTER COMMAND
1478
1479 015710  004037  032756                  JSR     R0,COMPAR       ;COMPARE
1480 015714  001306                          WC                      ;INITIAL SNAPSHOT BUFFER (CHANGED)
1481 015716  001500                          WRFROM                  ;TEST SNAPSHOT BUFFER
1482 015720  000022                          18.                     ;NUMBER OF REGISTERS
1483 015722  015730                          2$                      ;RETURN FOR ERROR
1484 015724  015730                          2$                      ;SAME
1485 015726  015750                          3$                      ;RETURN FOR GOOD COMPARISON
1486
1487 015730  013705  037030          2$:     MOV     ERWORD,R5       ;GETTING READY TO INDEX
1488 015734  060505                          ADD     R5,R5           ;DOUBLE ERROR WORD
1489 015736  016537  001230  031620          MOV     RHWC-2(R5),REGADR ;FAILING REGISTER
1490 015744  104001                          EMT     1
1491                                                                 ;CHANGE AFTER FORCING
1492                                                                 ;'DTE' ERROR
1493 015746  000207                          RTS     PC              ;RETURN
1494
1495 015750                          3$:                             ;GOOD - REGISTERS OK, GO ON TO NEXT TEST
1496
1511                                 ;;*****************************************************************
                                     ;*TEST 50        DRIVE TIMING ERROR
                                     ;*A WRITE DATA COMMAND IS STARTED ON CYLINDER 0, SECTOR
                                     ;*0, TRACK 0, 256 WORDS.
```

T50 DRIVE TIMING ERROR

```
                                        ;*
                                        ;*THE SECTOR IS SEARCHED FOR AND
                                        ;*AFTER THE HEADER IS READ IN CORRECTLY,
                                        ;*NO SYNC BYTE (DATA SYNC) IS GIVEN.
                                        ;*NORMAL DIAGNOSTIC DATA 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
                                        ;*WHICH SHOULD SET 'DRIVE TIMING ERROR'' - 'DTE'
                                        ;;*****************************************************************
     015750  000004                     TST50:  SCOPE
1512 015752  012706  001100                     MOV     #STACK,SP       ;RESET STACK
1513 015756  012737  000050  003610             MOV     #50,TSTNM       ;MOVE #50 TO TEST NUMBER
1514
1515                                    ;THESE ARE TO SETUP FOR DISKLESS USE
1516
1517 015764  012737  010000  036710             MOV     #FMT22,CYL      ;16 BITS PER WORD
1518                                                                     ;CYLINDER 0
1519 015772  005037  036712                     CLR     SECOTR  ;SECTOR 0
1520                                                                     ;TRACK 0
1521 015776  005037  036714                     CLR     KEY1            ;KEY1 = 0
1522 016002  005037  036716                     CLR     KEY2            ;KEY2 = 0
1523 016006  012737  177777  036720             MOV     #-1,X           ;THIS IS FOR WRITE DATA COMMAND
1524 016014  004537  033270                     JSR     R5,CRC  ;GO TO CALCULATE CRC
1525 016020  036710                             CYL
1526 016022  040610                             WCRC
1527
1528                                    ; THESE ARE REGULAR SETUPS & CHECKS
1529
1530 016024  004737  032054                     JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
1531 016030  012777  177400  163174             MOV     #-256.,@RHWC    ;256 DATA WORDS
1532 016036  012777  001500  163170             MOV     #WRFROM,@RHBA   ;STARTING ADDRESS OF BUFFER
1533 016044  005077  163174                     CLR     @RHDST          ;TRACK = 0
1534                                                                     ;SECTOR = 0
1535 016050  012777  010000  163172             MOV     #FMT22,@RHOF    ;16 BITS PER WORD
1536                                                                     ;ECC CORRECTION INHIBITED
1537 016056  005077  163170                     CLR     @RHCA           ;CYLINDER = 0
1538 016062  004737  032116                     JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
     016066  104401  053407                     TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                         ;TINUE TESTING IF BOTH AREN'T TRUE
     016072  000000                             HALT                    ;STOP THE TEST
1539 016074  013711  001452                     MOV     WRIDAT,@R1      ;WRITE DATA = 60
1540
1541                                    ;READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'
1542
1543 016100  004037  032554                     JSR     R0,SAVER        ;READ REGISTERS IN SEQUENCE
1544 016104  001232                             RHWC                    ;FROM HARDWARE REGISTER
1545 016106  001306                             WC                      ;INTO CORE BUFFER LOCATION
1546 016110  000023                             19.                     ;NUMBER OF REGISTERS TO READ
1547
1548                                    ;NOW 'GO' WILL BE GIVEN, EVERYTHING WILL BE TREATED
1549                                    ;NORMALLY FOR THE HEADER.  WHEN IT IS TIME TO WRITE
1550                                    ;DATA, ONLY SECTOR CLOCKS WILL BE GIVEN, NO DIAGNOSTIC DATA
1551                                    ;CLOCKS WILL BE GIVEN. THIS SHOULD BRING SECTOR PULSE HIGH
1552                                    ;WITHOUT PUTTING READ DOWN, HENCE 'DTE' WILL COME UP.
1553
```

T50 DRIVE TIMING ERROR

```
1554 016112  012737  177777  001426          MOV     #-1,TESDTE      ;SET DTE TEST
1555 016120  012737  177777  001406          MOV     #-1,ERFLG$      ;THIS WILL BRING THE READ HEADER
1556                                                                 ;AND DATA PROCESS OUT AFTER THE
1557                                                                 ;HEADER HAS BEEN CORRECTLY READ
1558
1559 016126  004737  036550                  JSR     PC,COMHD        ;ISSUE 'GO', SEARCH FOR SECTOR,
1560                                                                 ;READ HEADER AND DATA.
1561
1562 016132  017737  163102  001176  SETCK2: MOV     @RHCS1,$TMP0    ;READ CS1 TO CHECK FOR READ ERRORS
1563 016140  032737  100000  001176          BIT     #SC,$TMP0       ;TEST FOR "SPECIAL CONDITION" - 'SC'
1564 016146  001405                          BEQ     6$              ;CONTINUE TESTING IF NO ERROR
1565 016150  004737  031554                  JSR     PC,PUTREG       ;READ & SAVE REGISTERS AGAIN IF UNDE-
1566                                                                 ;FINED ERROR HAS OCCURRED
1567 016154  104040                          EMT     40
1568                                                                 ;DURING 'DTE' TEST SETUP
1569 016156  000137  016474                  JMP     TST51           ;ABORT THE TEST
1570
1571                                 ;NOW THE HEADER HAS BEEN READ,
1572                                 ;560 SECTOR CLOCKS WILL BE GIVEN -
1573                                 ;GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
1574                                 ;GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 2
1575
1576                                 ;THESE 560 SECTOR CLOCKS ARE DIVIDED INTO TWO GROUPS
1577                                 ;24 SECTOR CLOCKS WITH NORMAL DIAGNOSTIC DATA CLOCKS
1578                                 ;AND 536 SECTOR CLOCKS WITHOUT ANY DIAGNOSTIC DATA CLOCKS
1579
1580                                 ;THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC DATA CLOCKS
1581
1582 016162  012701  000030          6$:     MOV     #24.,R1         ;LOAD SECTOR CLOCK COUNTER
1583 016166  013700  001260                  MOV     RHMR,R0 ;GET RHMR ADDRESS
1584 016172  012710  000001                  MOV     #DMD,@R0        ;SET DIAGNOSTIC MODE
1585 016176  052710  000012          1$:     BIS     #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND DATA CLOCK
1586 016202  042710  000012                  BIC     #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND DATA CLOCK
1587 016206  012702  000007                  MOV     #7,R2           ;LOAD COUNTER FOR DIAGNOSTIC DATA CLOCKS
1588 016212  052710  000002          4$:     BIS     #MCLK,@R0       ;SET CLOCK (DATA)
1589 016216  042710  000002                  BIC     #MCLK,@R0       ;CLEAR CLOCK (DATA)
1590 016222  005302                          DEC     R2              ;COUNT
1591 016224  001372                          BNE     4$              ;BRANCH IF 7 NOT DONE
1592 016226  005301                          DEC     R1              ;COUNT
1593 016230  001362                          BNE     1$              ;BRANCH IF 24 NOT DONE
1594
1595                                 ;THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
1596
1597 016232  012701  001030                  MOV     #536.,R1        ;LOAD SECTOR CLOCK COUNTER
1598 016236  052710  000010          5$:     BIS     #MSTCK,@R0      ;SET SECTOR CLOCK
1599 016242  042710  000010                  BIC     #MSTCK,@R0      ;CLEAR SECTOR CLOCK
1600 016246  005301                          DEC     R1              ;COUNT
1601 016250  001372                          BNE     5$              ;BRANCH IF 536 NOT DONE
1602
1603                                 ;ECC PATTERN REGISTER IS NOT CHECKED
1604 016252  017737  163014  001346          MOV     @RHEC2,EC2      ;RHEC2 IS NOT CHECKED
1605
1606
1607                                 ;NOW 'DTE' SHOULD BE SET, CHANGE SAVED REGISTERS TO EXPECTED VALUES
1608
1609 016260  005737  003614                  TST     RH70            ;CHECK FOR RH70 CONTROLLER
1610 016264  001412                          BEQ     JP1             ;SKIP RH70 CODE AND DO RH11 IF NOT
```

```
                                        ;RHWC AND RHBA WILL BE MODIFIED FOR RH70C
016266  012737  177416  001306  VAR1:   MOV     #-242.,WC       ;SAVED RHWC
016274  012737  001534  001310  VAR2:   MOV     #WRFROM+<14.*2>,BA ;SAVED RHBA
016302  052737  000300  001312          BIS     #IR!OR,CS2      ;SAVED RHCS2
016310  000414                          BR      JP2             ;SKIP NEXT RH11 CODE

016312  012737  177511  001306  JP1:    MOV     #-183.,WC       ;SAVED RHWC
016320  012737  001722  001310          MOV     #WRFROM+<73.*2>,BA ;SAVED RHBA
016326  042737  000100  001312          BIC     #IR,CS2 ;SAVED RHCS2
016334  052737  000200  001312          BIS     #OR,CS2 ;SAVED RHCS2

016342  052737  140000  001314  JP2:    BIS     #SC!TRE,CS1     ;SAVED RHCS1
016350  052737  010000  001316          BIS     #DTE,ER1        ;SAVED RHER1
016356  012737  000201  001334          MOV     #DENVL!DMD,MR   ;SAVED RHMR
016364  052737  140000  001336          BIS     #ATA!ERR,DS1 ;SAVED RHDS1
016372  012737  000100  001350          MOV     #100,LA ;SAVED RHLA
016400  012737  000001  001320          MOV     #1,DST  ;SAVED RHDST
016406  013737  001416  001332          MOV     ATTENT,AS       ;SAVED RHAS

                                        ;NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS
                                        ;CAN BE DONE (USING 'WRFROM' BUFFER THIS TIME)

016414  004037  032554                 JSR     R0,SAVER        ;READ IN SEQUENCE
016420  001232                          RHWC                    ;FROM HARDWARE REGISTER
016422  002544                          REINTO                  ;INTO CORE BUFFER LOCATION
016424  000023                          19.                     ;NUMBER OF REGISTERS TO READ

                                        ;FOR RHAS UPPER BYTE
016426  113737  001333  002571          MOVB    AS+1,REINTO+25 ;UPPER RHAS


                                        ;COMPARE CHANGED REGISTER SNAPSHOT BEFORE COMMAND WITH
                                        ;SNAPSHOT AFTER COMMAND

016434  004037  032756                 JSR     R0,COMPAR       ;COMPARE
016440  001306                          WC                      ;CHANGED INITIAL SNAPSHOT BUFFER
016442  002544                          REINTO                  ;SNAPSHORT BUFFER AFTER COMMAND
016444  000022                          18.                     ;NUMBER OF REGISTERS TO COMPARE
016446  016454                          2$                      ;RETURN FOR ERROR
016450  016454                          2$                      ;SAME
016452  016474                          3$                      ;RETURN FOR GOOD COMPARISON

016454  013705  037030          2$:     MOV     ERWORD,R5       ;GETTING READY TO INDEX
016460  060505                          ADD     R5,R5           ;DOUBLE ERROR WORD
016462  016537  001230  031620          MOV     RHWC-2(R5),REGADR ;FAILING REGISTER
016470  104001                          EMT     1
                                                                ;CHANGE AFTER FORCING
                                                                ;'DTE' ERROR
016472  000207                          RTS     PC              ;RETURN

016474                          3$:                             ;GOOD, REGISTERS OK - GO ON TO NEXT TEST


                                        ;;*****************************************************************
                                        ;*TEST 51        DRIVE TIMING ERROR
                                        ;*A WRITE HEADER AND DATA COMMAND IS GIVEN
                                        ;*TO CYLINDER 0, SECTOR 0, TRACK 0, 256. WORDS.
```

T51 DRIVE TIMING ERROR

```
                                        ;*
                                        ;*AFTER SECTOR IS FOUND (THE SECTOR FOUND FLOP IS HIGH),
                                        ;*NO MORE DIAGNOSTIC DATA CLOCKS ARE GIVEN,
                                        ;*ONLY SECTOR CLOCKS ARE GIVEN TILL SECTOR PULSE IS HIGH.
                                        ;*THIS SHOULD SET 'DRIVE TIMING ERROR' - 'DTE'
                                        ;*****************************************************************
     016474  000004                     TST51:  SCOPE
1674 016476  012706  001100                     MOV     #STACK,SP       ;RESET STACK
1675 016502  012737  000051  003610             MOV     #51,TSTNM       ;MOVE #51 TO TEST NUMBER
1676
1677                                    ;THESE ARE TO SET UP FOR DISKLESS USE ONLY
1678
1679 016510  012737  010000  042046             MOV     #FMT22,WCYL     ;FORMAT 22=16 BITWORDS AND
1680                                                                     ;CYLINDER 0
1681 016516  005037  042050                     CLR     WSECTR  ;TRACK=0, SECTOR=0
1682 016522  005037  042052                     CLR     WKEY1           ;KEY1=0
1683 016526  005037  042054                     CLR     WKEY2           ;KEY2=0
1684 016532  012737  000400  042106             MOV     #256.,FNWORD    ;256 DATAWORDS
1685 016540  004537  033270                     JSR     R5,CRC  ;GO TO CALCULATE CRC
1686 016544  042046                            WCYL
1687 016546  042056                            GCRC
1688
1689                                    ; THESE ARE REGULAR SETUPS & CHECKS
1690
1691 016550  004737  032054                     JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
1692 016554  012777  177374  162450             MOV     #-260.,@RHWC    ;256 DATA WORDS & 4 HEADER WORDS
1693 016562  012777  001500  162444             MOV     #WRFROM,@RHBA   ;STARTING ADDRESS OF BUFFER
1694 016570  005077  162450                     CLR     @RHDST          ;TRACK = 0
1695                                                                    ;SECTOR = 0
1696 016574  012777  014000  162446             MOV     #FMT22!ECI,@RHOF ;16 BITS PER WORD
1697                                                                    ;ECC CORRECTION INHIBITED
1698 016602  005077  162444                     CLR     @RHCA           ;CYLINDER = 0
1699 016606  004737  032116                     JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
     016612  104401  053407                     TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                        ;TINUE TESTING IF BOTH AREN'T TRUE
     016616  000000                            HALT                    ;STOP THE TEST
1700 016620  013711  001454                     MOV     WRIFOR,@R1      ;WRITE HEADER AND DATA = 62
1701
1702                                    ;READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'
1703
1704 016624  004037  032554                     JSR     R0,SAVER        ;READ IN SEQUENCE
1705 016630  001232                            RHWC                    ;FROM HARDWARE REGISTER
1706 016632  001306                            WC                      ;INTO CORE BUFFER LOCATION
1707 016634  000023                            19.                     ;NUMBER OF REGISTERS TO READ
1708
1709                                    ;NOW 'GO' WILL BE GIVEN. EVERYTHING WILL BE TREATED
1710                                    ;NORMALLY TILL HEADER IS TO BE GIVEN, THEN ONLY
1711                                    ;SECTOR CLOCKS WILL BE GIVEN. NO DIAGNOSTIC DATA
1712                                    ;CLOCKS WILL BE GIVEN. THIS SHOULD BRING SECTOR PULSE HIGH
1713                                    ;WITHOUT PUTTING 'READ' DOWN, HENCE 'DTE' WILL COME UP.
1714
1715 016636  012737  177777  001426             MOV     #-1,TESDTE      ;SET DTE TEST
1716 016644  012737  177777  001406             MOV     #-1,ERFLG$      ;THIS WILL BRING THE READ HEADER
1717                                                                    ;AND DATA PROCESS OUT AFTER THE
1718                                                                    ;HEADER HAS BEEN CORRECTLY READ
1719
1720 016652  004737  041672                     JSR     PC,COMWHD       ;ISSUE 'GO', SEARCH FOR SECTOR,
```

```
                                                                        ;WRITE HEADER AND DATA.
                                                                        ;
016656  017737  162356  001176  SETCK3: MOV     @RHCS1,$TMP0          ;READ CS1 TO CHECK FOR ERRORS DURING WRITE
016664  032737  100000  001176          BIT     #SC,$TMP0             ;TEST FOR "SPECIAL CONDITION" - 'SC'
016672  001405                          BEQ     4$                    ;CONTINUE TEST IF NO ERROR ('SC' = 0)
016674  004737  031554                  JSR     PC,PUTREG             ;READ & SAVE REGISTERS AGAIN IF ERROR
016700  104040                          EMT     40
                                                                        ;DURING 'DTE' TEST SETUP
016702  000137  017134                  JMP     TST52                 ;ABORT THE TEST AT THAT POINT
                                ;NOW SECTOR HAS BEEN FOUND OK
                                ;
                                ;609 SECTOR CLOCKS WILL BE GIVEN,
                                ;39 BYTES FOR SECTOR GAP,
                                ;1 BYTE FOR HEADER SYNC,
                                ;8 BYTES FOR HEADER,
                                ;GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
                                ;GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 3
                                ;
                                ;THIS GIVES 609 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
                                ;
016706  012701  001141          4$:     MOV     #609.,R1              ;LOAD SECTOR CLOCK COUNTER
016712  052710  000010          5$:     BIS     #MSTCK,@R0            ;SET SECTOR CLOCK
016716  042710  000010                  BIC     #MSTCK,@R0            ;CLEAR SECTOR CLOCK
016722  005301                          DEC     R1                    ;COUNT
016724  001372                          BNE     5$                    ;BRANCH IF 536 NOT DONE
                                ;
                                ;NOW 'DTE' SHOULD BE SET, CHANGE SAVED REGISTERS
                                ;TO EXPECTED VALUES
                                ;
016726  005737  003614                  TST     RH70                  ;CHECK FOR RH70 CONTROLLER
016732  001407                          BEQ     JP3                   ;SKIP RH70 CODE AND DO RH11 IF NOT
                                ;
                                ;RHWC, RHBA AND IR BIT WILL BE MODIFIED FOR RH70C
                                ;
016734  012737  177404  001306  VAR3:   MOV     #-252.,WC             ;SAVED RHWC
016742  012737  001520  001310  VAR4:   MOV     #WRFROM+<8.*2>,BA ;SAVED RHBA
016750  000406                          BR      JP4                   ;SKIP NEXT RH11 CODE
                                ;
016752  012737  177477  001306  JP3:    MOV     #-193.,WC             ;SAVED RHWC
016760  012737  001706  001310          MOV     #WRFROM+<67.*2>,BA ;SAVED RHBA
                                ;
016766  052737  140000  001314  JP4:    BIS     #SC!TRE,CS1           ;SAVED RHCS1
016774  042737  000100  001312  VAR5:   BIC     #IR,CS2 ;SAVED RHCS2
017002  052737  000200  001312          BIS     #OR,CS2 ;SAVED RHCS2
017010  052737  010000  001316          BIS     #DTE,ER1              ;SAVED RHER1
017016  012737  000401  001334          MOV     #401,MR ;SAVED RHMR
017024  052737  140000  001336          BIS     #ATA!ERR,DS1          ;SAVED RHDS1
017032  012737  000100  001350          MOV     #100,LA ;SAVED RHLA
017040  012737  000001  001320          MOV     #1,DST  ;SAVED RHDST
017046  013737  001416  001332          MOV     ATTENT,AS             ;SAVED RHAS
                                ;
                                ;NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN BE DONE
                                ;
017054  004037  032554                  JSR     R0,SAVER              ;READ IN SEQUENCE
017060  001232                          RHWC                          ;FROM HARDWARE REGISTER
017062  002544                          REINTO                        ;INTO CORE BUFFER LOCATION
```

```
1778 017064  000023                        19.                     ;NUMBER OF REGISTERS TO READ
1779
1780                                ;FOR RHAS UPPER BYTE
1781 017066  113737  001333  002571        MOVB    AS+1,REINTO+25 ;UPPER RHAS
1782
1783                                ;COMPARE CHANGED REGISTER SNAPSHOT BEFORE COMMAND
1784                                ;WITH REGISTER SNAPSHOT AFTER COMMAND
1785
1786 017074  004037  032756            JSR     R0,COMPAR       ;COMPARE
1787 017100  001306                    WC                      ;CHANGED REGISTER SNAPSHOT
1788 017102  002544                    REINTO                  ;SNAPSHOT AFTER COMMAND
1789 017104  000022                    18.                     ;NUMBER OF REGISTERS TO COMPARE
1790 017106  017114                    2$                      ;RETURN FOR ERROR
1791 017110  017114                    2$                      ;SAME
1792 017112  017134                    3$                      ;RETURN FOR GOOD COMPARISON
1793
1794 017114  013705  037030    2$:     MOV     ERWORD,R5       ;GETTING READY TO INDEX
1795 017120  060505                    ADD     R5,R5           ;DOUBLE ERROR WORD
1796 017122  016537  001230  031620    MOV     RHWC-2(R5),REGADR ;FAILING REGISTER
1797 017130  104001                    EMT     1
1798                                                            ;CHANGE AFTER FORCING
1799                                                            ;'DTE' ERROR
1800 017132  000207                    RTS     PC              ;RETURN
1801
1802 017134                        3$:                         ;GOOD, REGISTERS COMPARE OK
1803                                                            ;GO ON TO THE NEXT TEST
1804
1815                                ;;*****************************************************************
                                    ;*TEST 52        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
                                    ;;*****************************************************************
     017134  000004                 TST52:  SCOPE
1816 017136  012706  001100                MOV     #STACK,SP       ;RESET STACK
1817 017142  012737  000052  003610        MOV     #52,TSTNM       ;MOVE #52 TO TEST NUMBER
1818 017150  004737  032054                JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS & CLEAR
1819                                                               ;THE DRIVE
1820 017154  012737  000026  001430        MOV     #22.,TAGDTE     ;22 SECTORS
1821                                                               ;THIS TEST REPEATS
1822                                                               ;ITSELF 22 TIMES
1823
1824                                ;THE FOLLOWING INITIALIZES FOR SECTOR 0
1825
1826 017162  005037  017266                CLR     SS3+2           ;HEADER (SECTOR)
1827 017166  012737  000025  017272        MOV     #21.,SS4+2      ;HEADER (KEY1)
1828 017174  012737  000025  017276        MOV     #21.,SS5+2      ;HEADER (KEY2)
1829 017202  005037  017324                CLR     SS7+2           ;DATA (SECTOR)
1830 017206  005037  017406                CLR     SS10+2  ;DATA
1831 017212  005037  017436                CLR     SS12+2  ;SECTOR (SIMULATED DISK)
1832 017216  012737  000025  017444        MOV     #21.,SS13+2     ;KEY1 (SIMULATED DISK)
1833 017224  012737  000025  017452        MOV     #21.,SS14+2     ;KEY2 (SIMULATED DISK)
```

```
1834 017232  005037  017512                    CLR     SS15+2  ;SECTOR (RHDST)
1835
1836                                 ;CLEAR SIMULATED DISK AREA
1837 017236                          SS1:
1838 017236  012700  040530          1$:     MOV     #SECGAP,R0      ;POINTER
1839 017242  012701  000460                  MOV     #304.,R1        ;COUNTER
1840 017246  005020                  2$:     CLR     (R0)+           ;CLEAR SIMULATED DISK AREA
1841 017250  005301                          DEC     R1              ;COUNT
1842 017252  001375                          BNE     2$
1843
1844
1845                                 ;SETUP WRITE FROM BUFFER
1846 017254  012700  001500                  MOV     #WRFROM,R0
1847
1848                                 ;HEADER
1849 017260  012720  010000                  MOV     #FMT22,(R0)+    ;FORMAT 16 BITS PER WORD
1850                                                                 ;CYLINDER 0
1851 017264  012720  000000          SS3:    MOV     #0,(R0)+        ;SECTOR TO VARY
1852 017270  012720  000025          SS4:    MOV     #21.,(R0)+      ;KEY1 TO VARY
1853 017274  012720  000025          SS5:    MOV     #21.,(R0)+      ;KEY2 TO VARY
1854
1855                                 ;DATA IN WRITE FROM BUFFER ALTHOUGH THIS IS DATA AND NOT
1856                                 ;HEADER, THE SECTOR WITH SYNC BYTES WILL BE GIVEN AS DATA.
1857
1858                                 ;DATA IS - 19 WORDS OF ZEROS - SYNC WORDS, 4 HEADER WORDS
1859                                 ;1 CRC WORD, 5 WORDS OF ZEROS, 1 SYNC WOPD, 100 ZEROS
1860                                 ;(DATA), 1 SYNC WORD, 70 SECTOR NUMBER TO VARY
1861
1862 017300  012705  000023                  MOV     #19.,R5         ;COUNTER
1863 017304  005020                  6$:     CLR     (R0)+           ;19 ZEROS
1864 017306  005305                          DEC     R5              ;COUNT
1865 017310  001375                          BNE     6$              ;19 DONE?
1866 017312  013720  037012                  MOV     RSYNC,(R0)+     ;SYNC = 14400
1867 017316  012720  010000                  MOV     #FMT22,(R0)+    ;CYLINDER 0
1868 017322  012720  000000          SS7:    MOV     #0,(R0)+        ;SECTOR TO VARY
1869 017326  005020                          CLR     (R0)+
1870 017330  005020                          CLR     (R0)+
1871 017332  004537  033270                  JSR     R5,CRC  ;CALCULATE CRC FOR ABOVE 4 WORDS
1872 017336  001550                          WRFROM+50               ;4 WORDS START FROM HERE
1873 017340  001560                          WRFROM+60               ;PUT CRC HERE
1874
1875 017342  005720                          TST     (R0)+           ;INCREMENT R0
1876
1877 017344  012705  000005                  MOV     #5.,R5
1878 017350  005020                  8$:     CLR     (R0)+           ;5 WORDS OF ZEROS
1879 017352  005305                          DEC     R5              ;COUNT
1880 017354  001375                          BNE     8$              ;BRANCH IF 5 NOT DONE
1881
1882 017356  013720  037012                  MOV     RSYNC,(R0)+     ;SYNC = 14400
1883
1884 017362  012705  000144                  MOV     #100.,R5
1885 017366  005020                  9$:     CLR     (R0)+           ;100 WORDS OF ZEROS
1886 017370  005305                          DEC     R5
1887 017372  001375                          BNE     9$
1888
1889 017374  013720  037012                  MOV     RSYNC,(R0)+     ;SYNC = 14400
1890 017400  012705  000106                  MOV     #70.,R5
```

```
1891 017404  012720  000000          SS10:   MOV     #0,(R0)+        ;SECTOR TO VARY
1892 017410  005305                          DEC     R5
1893 017412  001374                          BNE     SS10
1894
1895                                 ;CLEAR REST OF 256 WORDS THAT IS 54 WORDS OF ZEROS
1896
1897 017414  012705  000066                  MOV     #54.,R5
1898 017420  005020                  11$:    CLR     (R0)+
1899 017422  005305                          DEC     R5
1900 017424  001375                          BNE     11$
1901
1902                                 ;THESE ARE TO BE SET UP FOR DISKLESS USE ONLY
1903
1904 017426  012737  010000  042046          MOV     #FMT22,WCYL     ;FORMAT = 16 BIT WORDS
1905                                                                 ;CYLINDER = 0
1906 017434  012737  000000  042050  SS12:   MOV     #0,WSECTR       ;SECTOR TO VARY
1907 017442  012737  000025  042052  SS13:   MOV     #21.,WKEY1      ;KEY1 TO VARY
1908 017450  012737  000025  042054  SS14:   MOV     #21.,WKEY2      ;KEY2 TO VARY
1909 017456  012737  000312  042106          MOV     #202.,FNWORD    ;202 DATA WORDS
1910 017464  004537  033270                  JSR     R5,CRC  ;CALCULATE CRC
1911 017470  042046                          WCYL                    ;FIRST WORD
1912 017472  042056                          GCRC                    ;PUT HERE
1913
1914                                 ;THESE ARE REGULAR SETUPS
1915
1916 017474  012777  177400  161530          MOV     #-256.,@RHWC    ;202 DATA, 4 HEADER
1917 017502  012777  001500  161524          MOV     #WRFROM,@RHBA   ;FILL BUS ADDRESS
1918 017510  012777  000000  161526  SS15:   MOV     #0,@RHDST       ;SECTOR TO VARY
1919 017516  013777  001454  161514          MOV     WRIFOR,@RHCS1   ;GET READY TO DO
1920                                                                 ;WRITE HEADER AND DATA
1921                                                                 ;WITH 62 FUNCTION CODE IN RHCS1
1922 017524  012777  010000  161516          MOV     #FMT22,@RHOF    ;16 BITS PER WORD FORMAT
1923 017532  005077  161514                  CLR     @RHCA           ;CYLINDER = 0
1924
1925 017536  005037  001406                  CLR     ERFLG$  ;CLEAR ERROR FLAG
1926
1927 017542  004737  032116                  JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
     017546  104401  053407                  TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                     ;TINUE TESTING IF BOTH AREN'T TRUE
     017552  000000                          HALT                    ;STOP THE TEST
1928
1929 017554  004737  041672                  JSR     PC,COMWHD       ;ISSUE 'GO', COUNT SECTOR CLOCKS,
1930                                                                 ;WRITE HEADER AND DATA
1931 017560  005737  001406                  TST     ERFLG$  ;HAVE ANY ERRORS OCCURRED ?
1932 017564  001051                          BNE     TST53           ;EXIT IF YES----------------------)
1933
1934 017566  004737  032306                  JSR     PC,CHECKE       ;CHECK THAT BITS = 1
     017572  104401  053407                  TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF THEY DON'T
     017576  000000                          HALT                    ;STOP THE TEST
1935
1936                                 ;NOW COMPARE "DISK" BUFFER WITH "REINTO" BUFFER
1937 017600  004037  032756                  JSR     R0,COMPAR       ;CHECK
1938 017604  001510                          WRFROM+8.               ;GOOD BUFFER
1939 017606  040626                          DISK                    ;TEST BUFFER
1940 017610  000400                          256.                    ;NUMBER OF WORDS
1941 017612  017620                          16$                     ;RETURN POINT FOR ERROR  HEADER
1942 017614  017624                          17$                     ;RETURN POINT FOR ERROR DATA
```

```
1943 017616  017630                        18$                     ;RETURN FOR GOOD COMPARISON
1944 017620                          16$:
     017620  104007                        EMT     7
1945 017622  000207                        RTS     PC
1946 017624                          17$:
     017624  104010                        EMT     10
1947 017626  000207                        RTS     PC
1948
1949                                 ;THE FOLLOWING INCREMENTS ARE TO CHANGE THE ABOVE SET UP
1950                                 ;TO WRITE ON THE NEXT SECTOR
1951
1952 017630  005237  017266        18$:    INC     SS3+2           ;HEADER (SECTOR)
1953 017634  005337  017272              DEC     SS4+2           ;HEADER (KEY1)
1954 017640  005337  017276              DEC     SS5+2           ;HEADER (KEY2)
1955 017644  005237  017324              INC     SS7+2           ;DATA (SECTOR)
1956 017650  005237  017406              INC     SS10+2  ;DATA
1957 017654  005237  017436              INC     SS12+2  ;SECTOR (SIMULATED DISK)
1958 017660  005337  017444              DEC     SS13+2  ;KEY1 (SIMULATED DISK)
1959 017664  005337  017452              DEC     SS14+2  ;KEY2 (SIMULATED DISK)
1960 017670  005237  017512              INC     SS15+2  ;SECTOR (RHDST)
1961
1962 017674  005337  001430        SS2:    DEC     TAGDTE  ;COUNT DOWN FOR 22 SECTORS
1963 017700  001001                        BNE     1$              ;BRANCH IF 22 SECTORS NOT DONE
1964 017702  000402                        BR      TST53           ;ALL DONE - GO TO NEXT TEST
1965 017704  000137  017236        1$:     JMP     SS1             ;GO BACK TO NEXT SECTOR
1966
1973                                 ;;*****************************************************************
                                     ;*TEST 53        WRITE ECC TEST 1
                                     ;*THIS IS A WRITE ECC TEST
                                     ;*WRITE CYLINDER0, FORMAT 16 BITS PER WORD
                                     ;*TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
                                     ;*OF ALL ZEROS.
                                     ;;*****************************************************************
     017710  000004                  TST53:  SCOPE
1974 017712  012706  001100                  MOV     #STACK,SP       ;RESET STACK
1975
1976 017716  012737  000053  003610          MOV     #53,TSTNM       ;MOVE #53 TO TEST NUMBER
1977 017724  012700  040530                  MOV     #SECGAP,R0      ;POINTER
1978 017730  012701  000402                  MOV     #258.,R1        ;COUNTER
1979 017734  012720  177777        1$:       MOV     #-1,(R0)+       ;FILL SIMULATER DISK WITH ONES
1980 017740  005301                          DEC     R1
1981 017742  001374                          BNE     1$
1982 017744  004737  032054                  JSR     PC,CLDISK       ;THIS IS USED TO SET GENERAL REGISTERS
1983
1984                                 ;THESE ARE FOR ECC TEST ONLY
1985
1986 017750  012737  177777  001424          MOV     #-1,TSECC       ;THIS IS AN ECC TEST
1987 017756  005037  034570                  CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
1988 017762  013737  034564  034566          MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
1989 017770  013737  034572  034600          MOV     HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
1990 017776  005037  034556                  CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
1991 020002  005037  034560                  CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
1992 020006  005037  034574                  CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
1993 020012  005037  034576                  CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
1994
1995
1996
```

```
                                        ;THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
                                        ;
020016  012737  010000  042046          MOV     #FMT22,WCYL     ;FORMAT22=16BIT WORDS AND
                                                                ;CYLINDER 0
020024  012737  000001  042050          MOV     #1,WSECTR       ;TRACK=0, SECTOR=1
020032  005037  042052                  CLR     WKEY1           ;KEY1=0
020036  005037  042054                  CLR     WKEY2           ;KEY2=0
020042  012737  000400  042106          MOV     #256.,FNWORD    ;256 DATA WORDS
020050  004537  033270                  JSR     R5,CRC  ;GO TO CALCULATE CRC
020054  042046                          WCYL
020056  042056                          GCRC
                                        ;
                                        ;THESE ARE REGULAR SETUPS
                                        ;
020060  012777  177374  161144          MOV     #-260.,@RHWC    ;256 DATA WORDS 4 HEADER WORDS
020066  012700  001500                  MOV     #WRFROM,R0      ;THESE TWO INSTRUCTIONS GETS
020072  010077  161136                  MOV     R0,@RHBA        ;ADDR. OF WRFROM INTO R0 AND
                                                                ;BUS ADDRESS REGISTER
020076  012720  010000                  MOV     #FMT22,(R0)+    ;FORMAT=16 BIT WORDS
                                                                ;CYLINDER=0
020102  012720  000001          2$:     MOV     #1,(R0)+        ;TRACK=0, SECTOR=1, KEYS=0
020106  005020                          CLR     (R0)+           ;KEY1=0
020110  005020                          CLR     (R0)+           ;KEY2=0
020112  012705  000400                  MOV     #256.,R5        ;COUNTER
020116  012720  000000          3$:     MOV     #0,(R0)+        ;MOVE ALL ZEROS FOR DATA
020122  005305                          DEC     R5
020124  001374                          BNE     3$              ;BRANCH IF DATA NOT COMPLETE
020126  012777  000001  161110          MOV     #1,@RHDST       ;TRACK=0 SECTOR=1
                                        ;
020134  004737  032116                  JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
020140  104401  053407                  TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                ;TINUE TESTING IF BOTH AREN'T TRUE
020144  000000                          HALT                    ;STOP THE TEST
                                        ;
020146  013711  001454                  MOV     WRIFOR,@R1      ;GET READY FOR WRITE HEADER AND
                                                                ;DATA WITH 62 IN RHCS1
020152  005037  001406                  CLR     ERFLG$  ;CLEAR ERROR FLAG
020156  012777  010000  161064          MOV     #FMT22,@RHOF    ;FORMAT BIT=1 (16 BIT WORDS)
020164  005077  161062                  CLR     @RHCA           ;CYLINDER =0
020170  004737  041672                  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 ZEROS AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
                                        ;THEY ARE ALL ZEROS
                                        ;
020174  005737  001406                  TST     ERFLG$  ;HAS ANY ERRORS OCCURED?
                                                                ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
020200  001056                          BNE     TST54           ;;BRANCH IF YES
                                        ;
                                        ;
                                        ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
                                        ;
020202  023737  034556  041626          CMP     GECC1,WECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
020210  001402                          BEQ     6$              ;BRANCH IF GOOD
```

```
2051 020212  104031                          EMT     31
2052 020214  000405                          BR      7$                  ;BRANCH TO CONTINUE
2053 020216  023737  034560  041630  6$:     CMP     GECC2,WECC2;COMPARE SOFTWARE ECC WITH HARDWARE ECC
2054 020224  001401                          BEQ     7$                  ;BRANCH IF GOOD
2055 020226  104031                          EMT     31
2056
2057
2058 020230                          7$:
     020230  004737  032306                  JSR     PC,CHECKE           ;CHECK THAT BITS = 1
     020234  104401  053407                  TYPE    ,CPHALT             ;CANNOT CONTINUE TESTING IF THEY DON'T
     020240  000000                          HALT                        ;STOP THE TEST
2059
2060
2061
2062                                 ;FILL "REINTO" BUFFER WITH EXPECTED DATA
2063
2064 020242  004037  031772                  JSR     R0,CLAREA           ;FILL REINTO BUFFER
2065 020246  002544                          REINTO                      ;FROM
2066 020250  003542                          REINTO+<255.*2>             ;TO
2067 020252  000000                          .WORD   0                   ;DATA
2068
2069 020254  013737  034556  003544          MOV     GECC1,REINTO+<256.*2>;FILL ECC1
2070 020262  013737  034560  003546          MOV     GECC2,REINTO+<257.*2>;FILL ECC2
2071 020270  004037  031772                  JSR     R0,CLAREA           ;FILL REST
2072 020274  003550                          REINTO+<258.*2>             ;FROM
2073 020276  003604                          REINTO+<272.*2>             ;TO
2074 020300  000000                          0                           ;DATA
2075
2076
2077 020302  005037  001406                  CLR     ERFLG$  ;CLEAR ERROR FLAG
2078
2079
2080                                 ;NOW COMPARE "DISK" BUFFER WITH "REINTO"
2081
2082 020306  004037  032756                  JSR     R0,COMPAR           ;CHECK
2083 020312  002544                          REINTO                      ;GOOD BUFFER
2084 020314  040626                          DISK                        ;TEST BUFFER
2085 020316  000402                          258.                        ;NUMBER OF WORDS CHECKED
2086 020320  020326                          4$                          ;RETURN POINT FOR ERROR HEADER
2087 020322  020332                          5$                          ;RETURN POINT FOR ERROR DATA
2088 020324  020336                          TST54                       ;RETURN FOR GOOD COMPARISON
2089 020326                          4$:
     020326  104007                          EMT     7
2090 020330  000207                          RTS     PC                  ;RETURN TO COMPARE
2091 020332                          5$:
     020332  104010                          EMT     10
2092                                                                     ;DATA WORDS
2093                                                                     ;WORD NOS 257 AND 258
2094                                                                     ;ARE ECC WHICH ARE CHECKED
2095                                                                     ;WORD NOS 259
2096                                                                     ;IS DATA GAP
2097                                                                     ;WORD NOS 260 TO 273
2098                                                                     ;ARE TOLERANCE GAP
2099 020334  000207                          RTS     PC                  ;RETURN TO COMPARE
2100
2109                                 ;;*******************************************************************
                                     ;*TEST 54        READ ECC ENABLED 1A
```

```
                                        ;*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
                                        ;;*****************************************************************
    020336  000004                      TST54:  SCOPE
2110 020340  012706  001100                      MOV     #STACK,SP       ;RESET STACK
2111 020344  012737  000054  003610              MOV     #54,TSTNM       ;MOVE #54 TO TEST NUMBER
2112
2113                                    ;       SETUP FOR WHAT IS TO BE READ
2114                                    ;       HEADER CRC IS RESTORED FROM A SUBROUTINE
2115
2116 020352  012746  000000                      MOV     #0,-(SP)        ;DATA TO BE READ
2117 020356  012705  000400                      MOV     #256.,R5        ;COUNTER
2118 020362  012700  040626                      MOV     #DISK,R0        ;START OF SIMULATED DISK DATA
2119 020366  011620                      1$:     MOV     (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
2120 020370  005305                              DEC     R5              ;COUNT
2121 020372  001375                              BNE     1$              ;BRANCH IF 256 NOT COMPLETE
2122 020374  005726                              TST     (SP)+           ;UNDO -(SP)
2123 020376  022020                              CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
2124 020400  012705  000017                      MOV     #15.,R5         ;1 DATA GAP
2125                                                                     ;14 TOLERANCE GAP
2126 020404  005020                      2$:     CLR     (R0)+           ;CLEAR DATA GAP, AND
2127 020406  005305                              DEC     R5              ;TOLERANCE GAP
2128 020410  001375                              BNE     2$              ;BRANCH IF NOT COMPLETE
2129
2130
2131 020412  004737  035352                      JSR     PC,FILLEC       ;INSERT THE TWO ECC WORDS ON THE DISK
2132                                                                     ;IN THE CORRECT PLACE
2133
2134                                    ;THESE ARE FOR ECC TEST ONLY
2135
2136 020416  012737  177777  001424              MOV     #-1,TSECC       ;THIS IS AN ECC TEST
2137 020424  005037  034570                      CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
2138 020430  013737  034564  034566              MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
2139 020436  013737  034572  034600              MOV     HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
2140 020444  005037  034556                      CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
2141 020450  005037  034560                      CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
2142 020454  005037  034574                      CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
2143 020460  005037  034576                      CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
2144
2145
2146                                    ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
2147
2148 020464  012737  010000  036710              MOV     #FMT22,CYL      ;16 BITS PER WORD
2149                                                                     ;CYLINDER 0, FORMAT 16 BITS
2150 020472  112737  000000  036713              MOVB    #0,SECOTR+1     ;TRACK 0
2151 020500  112737  000000  036712              MOVB    #0,SECOTR       ;SECTOR 0
2152 020506  012737  000000  036714              MOV     #0,KEY1 ;KEY1=0
2153 020514  012737  000000  036716              MOV     #0,KEY2 ;KEY2=0
2154 020522  012737  000400  036 70              MOV     #256.,DAWORD ;NO. OF DATA WORDS
2155 020530  005037  036720                      CLR     X               ;THIS IS A READ COMMAND
2156 020534  004537  033270                      JSR     R5,CRC  ;GO TO CALCULATE CRC
2157 020540  036710                             CYL
2158 020542  040610                             WCRC
```

```
                                        ;THESE ARE REGULAR SETUPS
020544  004737  032054                        JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
020550  012777  177374  160454                MOV     #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
020556  012777  002544  160450                MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
020564  112746  000000                        MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
020570  112766  000000  000001                MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
020576  012677  160442                        MOV     (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
020602  012777  010000  160440                MOV     #FMT22,@RHOF    ;16 BITS PER WORD
                                                                        ;ECC CORRECTION NOT INHIBIT
                                                                        ;BECAUSE ECC IS NOT GOING
                                                                        ;TO BE CHECKED
020610  005077  160436                        CLR     @RHCA           ;CYLINDER 0
020614  004737  032116                        JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
020620  104401  053407                        TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                        ;TINUE TESTING IF BOTH AREN'T TRUE
020624  000000                                HALT                    ;STOP THE TEST
020626  013711  001460                        MOV     REFOR,@R1       ;READ HEADER AND DATA=72
020632  005037  001406                        CLR     ERFLG$  ;CLEAR ERROR FLAG
020636  004737  036550                        JSR     PC,COMHD        ;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 "COMHD" 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

020642  005737  001406                        TST     ERFLG$  ;ANY ERRORS ALREADY THERE
020646  001102                                BNE     TST55           ;BRANCH IF YES
020650  004737  031554                        JSR     PC,PUTREG       ;SAVE REGISTERS
020654  005737  001316                        TST     ER1             ;NO ERRORS SHOULD BE SET
020660  001401                                BEQ     6$              ;BRANCH IF NO ERRORS SET
020662  104032                                EMT     32
                                                                        ;ONLY 11 OF THE 32 BITS CAN BE SEEN
                                                                        ;IN THE PATERN REGISTER
                                                                        ;DCK SHOULD BE SET IN RHER1
020664  013746  034556                6$:     MOV     GECC1,-(SP)     ;GET PATTERN REGISTER
020670  042716  174000                        BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
020674  022637  001346                        CMP     (SP)+,EC2       ;COMPARE PATTERN REGISTER
020700  001401                                BEQ     7$              ;BRANCH IF GOOD
020702  104032                                EMT     32
```

```
                                        ;ADD 16 MAINTENANCE CLOCKS TO
                                        ;BRING EBL DOWN
                                
020704  012700  000020          7$:   MOV     #16.,R0           ;COUNTER
020710  052777  000002  160342  8$:   BIS     #MCLK,@RHMR       ;SET CLOCK
020716  042777  000002  160334        BIC     #MCLK,@RHMR       ;CLEAR CLOCK
020724  005300                        DEC     R0                ;COUNT
020726  001370                        BNE     8$                ;BRANCH IF 16 CLOCKS NOT DONE
020730  004737  032306                JSR     PC,CHECKE         ;CHECK THAT BITS = 1
020734  104401  053407                TYPE    ,CPHALT           ;CANNOT CONTINUE TESTING IF THEY DON'T
020740  000000                        HALT                      ;STOP THE TEST
020742  012700  001500                MOV     #WRFROM,R0        ;GETTING READY TO FILL EXPECTED DATA
020746  012720  010000                MOV     #0!FMT22,(R0)+    ;CYLINDER 0
020752  112746  000000                MOVB    #0,-(SP)          ;IN LOWER BYTE GET SECTOR
020756  112766  000000  000001        MOVB    #0,1(SP)          ;GET TRACK IN HIGHER BYTE
020764  012620                        MOV     (SP)+,(R0)+       ;GET TRACK/SECTOR IN BUFFER
020766  012720  000000                MOV     #0,(R0)+          ;KEY1 IN BUFFER
020772  012720  000000                MOV     #0,(R0)+          ;KEY2 IN BUFFER
020776  012701  000400                MOV     #256.,R1          ;DATA WORD COUNTER
021002  012702  000000                MOV     #0,R2             ;DATA
021006  010220          3$:   MOV     R2,(R0)+          ;DATA INTO BUFFER
021010  005301                        DEC     R1                ;COUNT
021012  001375                        BNE     3$                ;BRANCH IF 256 NOT DONE


021014  005037  001406                CLR     ERFLG$  ;CLEAR ERROR FLAG
021020  004737  031554                JSR     PC,PUTREG         ;SAVE REGISTERS


                                ;NOW READ DATA BUFFER WILL BE CHECKED

021024  004037  032756                JSR     R0,COMPAR         ;CHECK
021030  001500                        WRFROM                    ;GOOD BUFFER
021032  002544                        REINTO                    ;TEST BUFFER
021034  000404                        4+256.                    ;NUMBER OF WORDS CHECKED
021036  021044                        4$                        ;RETURN POINT FOR ERROR HEADER
021040  021050                        5$                        ;RETURN POINT FOR ERROR DATA
021042  021054                        TST55                     ;RETURN FOR GOOD COMPARISON
021044                          4$:
021044  104004                        EMT     4
021046  000207                        RTS     PC                ;RETURN TO ''COMPAR''
021050                          5$:
021050  104005                        EMT     5
                                                                  ;HEADER WORDS
                                                                  ;5 TO 260 ARE DATA WORDS
021052  000207                        RTS     PC                ;RETURN TO ''COMPAR''


                                ;;*****************************************************************
                                ;*TEST 55        READ ECC ENABLED 1B
                                ;*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
                                ;*TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
```

```
                                        ;;*****************************************************************
021054  000004                        TST55:  SCOPE
2267 021056  012706  001100                 MOV     #STACK,SP       ;RESET STACK
2268
2269 021062  012737  000055  003610         MOV     #55,TSTNM       ;MOVE #55 TO TEST NUMBER
2270
2271
2272                                    ;SETUP FOR WHAT IS TO BE READ
2273                                    ;HEADER CRC IS RESTORED FROM A SUBROUTINE
2274
2275 021070  012746  000000                 MOV     #0,-(SP)        ;DATA TO BE READ
2276 021074  012705  000400                 MOV     #256.,R5        ;COUNTER
2277 021100  012700  040626                 MOV     #DISK,R0        ;START OF SIMULATED DISK DATA
2278 021104  011620                 1$:     MOV     (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
2279 021106  005305                         DEC     R5              ;COUNT
2280 021110  001375                         BNE     1$              ;BRANCH IF 256 NOT COMPLETE
2281 021112  005726                         TST     (SP)+           ;UNDO -(SP)
2282 021114  022020                         CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
2283 021116  012705  000017                 MOV     #15.,R5 ;1 DATA GAP
2284                                                                 ;14 TOLERANCE GAP
2285 021122  005020                 2$:     CLR     (R0)+           ;CLEAR DATA GAP, AND
2286 021124  005305                         DEC     R5              ;TOLERANCE GAP
2287 021126  001375                         BNE     2$              ;BRANCH IF NOT COMPLETE
2288
2289
2290 021130  004737  035352                 JSR     PC,FILLEC       ;INSERT ECC IN PROPER PLACE ON DISK
2291
2292
2293
2294                                    ;THESE ARE FOR ECC TEST ONLY
2295
2296 021134  012737  177777  001424         MOV     #-1,TSECC       ;THIS IS AN ECC TEST
2297 021142  005037  034570                 CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
2298 021146  013737  034564  034566         MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
2299 021154  013737  034572  034600         MOV     HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
2300 021162  005037  034556                 CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
2301 021166  005037  034560                 CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
2302 021172  005037  034574                 CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
2303 021176  005037  034576                 CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
2304
2305
2306                                    ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
2307
2308 021202  012737  010000  036710         MOV     #FMT22,CYL      ;16 BITS PER WORD
2309                                                                 ;CYLINDER 0, FORMAT 16 BITS
2310 021210  112737  000000  036713         MOVB    #0,SECOTR+1     ;TRACK 0
2311 021216  112737  000000  036712         MOVB    #0,SECOTR       ;SECTOR 0
2312 021224  012737  000000  036714         MOV     #0,KEY1 ;KEY1=0
2313 021232  012737  000000  036716         MOV     #0,KEY2 ;KEY2=0
2314 021240  012737  000400  036770         MOV     #256.,DAWORD ;NO. OF DATA WORDS
2315 021246  005037  036720                 CLR     X               ;THIS IS A READ COMMAND
2316 021252  004537  033270                 JSR     R5,CRC  ;GO TO CALCULATE CRC
2317 021256  036710                         CYL
2318 021260  040610                         WCRC
2319
2320
2321                                    ;THIS IS TO INSERT ERROR
```

```
                                        ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
                                        ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
                                        ;THIS MOVE

021262  012737  100000  040630            MOV     #100000,DISK+2 ;FORCE ERROR ON BIT NUMBER 32
                                                                 ;SO ERROR POSITION REGISTER WILL SHOW
                                                                 ;22
021270  012737  000026  021444            MOV     #22.,8$ ;INSERT POSITION REG.


                                        ;THESE ARE REGULAR SETUPS

021276  004737  032054                    JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
021302  012777  177374  157722            MOV     #-256.*4.,@RHWC ;256. DATA 4 HEADER WORDS
021310  012777  002544  157716            MOV     #READ10,@RHBA   ;STARTING ADDRESS OF READ BUFFER
021316  112746  000000                    MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
021322  112766  000000  000001            MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
021330  012677  157710                    MOV     (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
021334  012777  010000  157706            MOV     #FMT22,@RHOF    ;16 BITS PER WORD
                                                                 ;ECC CORRECTION NOT INHIBIT
                                                                 ;BECAUSE ECC IS NOT GOING
                                                                 ;TO BE CHECKED
021342  005077  157704                    CLR     @RHCA           ;CYLINDER 0

021346  004737  032116                    JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
021352  104401  053407                    TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                 ;TINUE TESTING IF BOTH AREN'T TRUE
021356  000000                            HALT                    ;STOP THE TEST

021360  013711  001460                    MOV     REFOR,@R1       ;READ HEADER AND DATA=72
021364  005037  001406                    CLR     ERFLG$  ;CLEAR ERROR FLAG
021370  004737  036550                    JSR     PC,COMHD        ;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 "COMHD" 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

021374  005737  001406                    TST     ERFLG$  ;ANY ERRORS ALREADY THERE
021400  001077                            BNE     TST56           ;BRANCH IF YES
021402  004737  031554                    JSR     PC,PUTREG       ;SAVE REGISTERS
021406  022737  100000  001316            CMP     #DCK,ER1        ;ONLY DATA CHECK ERROR SHOULD BE SET
021414  001401                            BEQ     6$              ;BRANCH IF YES
021416  104032                            EMT     32
                                                                 ;ZERO
                                                                 ;ONLY 11 OF THE 32 BITS CAN BE SEEN
                                                                 ;IN THE PATERN REGISTER
                                                                 ;DCK SHOULD BE SET IN RHER1
021420  013746  034556            6$:     MOV     GECC1,-(SP)     ;GET PATTERN REGISTER
```

```
2376 021424  042716  174000                BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
2377 021430  022637  001346                CMP     (SP)+,EC2       ;COMPARE PATTERN REGISTER
2378 021434  001401                        BEQ     7$              ;BRANCH IF GOOD
2379 021436  104032                        EMT     32
2380
2381 021440  004037  035200          7$:   JSR     R0,ECORR        ;GO TO ECC CORRECTION PROCESS
2382 021444  000026                8$:   22.                     ;EXPECTED POSITION REG. WHEN CORRECTION
2383                                                              ;IS COMPLETE
2384
2385
2386
2387 021446  004737  032306                JSR     PC,CHECKE       ;CHECK THAT BITS = 1
     021452  104401  053407                TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF THEY DON'T
     021456  000000                        HALT                    ;STOP THE TEST
2388 021460  012700  001500                MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
2389 021464  012720  010000                MOV     #0!FMT22,(R0)+  ;CYLINDER 0
2390 021470  112746  000000                MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
2391 021474  112766  000000  000001        MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
2392 021502  012620                        MOV     (SP)+,(R0)+     ;GET TRACK/SECTOR IN BUFFER
2393 021504  012720  000000                MOV     #0,(R0)+        ;KEY1 IN BUFFER
2394 021510  012720  000000                MOV     #0,(R0)+        ;KEY2 IN BUFFER
2395 021514  012701  000400                MOV     #256.,R1        ;DATA WORD COUNTER
2396 021520  012702  000000                MOV     #0,R2           ;DATA
2397 021524  010220                3$:     MOV     R2,(R0)+        ;DATA INTO BUFFER
2398 021526  005301                        DEC     R1              ;COUNT
2399 021530  001375                        BNE     3$              ;BRANCH IF 256 NOT DONE
2400
2401                                ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
2402                                ;NOW THE INSERTED ERROR WILL BE PUT IN
2403 021532  012737  100000  001512        MOV     #100000,WRFROM+<5*2>    ;INSERTED ERROR
2404
2405
2406
2407 021540  005037  001406                CLR     ERFLG$  ;CLEAR ERROR FLAG
2408 021544  004737  031554                JSR     PC,PUTREG       ;SAVE REGISTERS
2409
2410
2411                              ;NOW READ DATA BUFFER WILL BE CHECKED
2412
2413 021550  004037  032756                JSR     R0,COMPAR       ;CHECK
2414 021554  001500                        WRFROM                  ;GOOD BUFFER
2415 021556  002544                        REINTO                  ;TEST BUFFER
2416 021560  000404                        4+256.                  ;NUMBER OF WORDS CHECKED
2417 021562  021570                        4$                      ;RETURN POINT FOR ERROR HEADER
2418 021564  021574                        5$                      ;RETURN POINT FOR ERROR DATA
2419 021566  021600                        TST56                   ;RETURN FOR GOOD COMPARISON
2420 021570                        4$:
     021570  104004                        EMT     4
2421 021572  000207                        RTS     PC              ;RETURN TO ''COMPAR''
2422 021574                        5$:
     021574  104005                        EMT     5
2423                                                              ;HEADER WORDS
2424                                                              ;5 TO 260 ARE DATA WORDS
2425 021576  000207                        RTS     PC              ;RETURN TO ''COMPAR''
2426
2435                              ;;*****************************************************************
                                  ;*TEST 56        READ ECC ENABLED 1C
```

T56 READ ECC ENABLED 1C

```
                                        ;*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
                                        ;;*****************************************************************
     021600  000004                     TST56:  SCOPE
2436 021602  012706  001100                     MOV     #STACK,SP       ;RESET STACK
2437 021606  012737  000056  003610             MOV     #56,TSTNM       ;MOVE #56 TO TEST NUMBER
2438
2439                                    ;       SETUP FOR WHAT IS TO BE READ
2440                                    ;       HEADER CRC IS RESTORED FROM A SUBROUTINE
2441
2442 021614  012746  000000                     MOV     #0,-(SP)        ;DATA TO BE READ
2443 021620  012705  000400                     MOV     #256.,R5        ;COUNTER
2444 021624  012700  040626                     MOV     #DISK,R0        ;START OF SIMULATED DISK DATA
2445 021630  011620                     1$:     MOV     (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
2446 021632  005305                             DEC     R5              ;COUNT
2447 021634  001375                             BNE     1$              ;BRANCH IF 256 NOT COMPLETE
2448 021636  005726                             TST     (SP)+           ;UNDO -(SP)
2449 021640  022020                             CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
2450 021642  012705  000017                     MOV     #15.,R5         ;1 DATA GAP
2451                                                                     ;14 TOLERANCE GAP
2452 021646  005020                     2$:     CLR     (R0)+           ;CLEAR DATA GAP, AND
2453 021650  005305                             DEC     R5              ;TOLERANCE GAP
2454 021652  001375                             BNE     2$              ;BRANCH IF NOT COMPLETE
2455
2456
2457 021654  004737  035352                     JSR     PC,FILLEC       ;INSERT THE TWO ECC WORDS ON THE DISK
2458                                                                     ;IN THE CORRECT PLACE
2459
2460                                    ;THESE ARE FOR ECC TEST ONLY
2461
2462 021660  012737  177777  001424             MOV     #-1,1SECC       ;THIS IS AN ECC TEST
2463 021666  005037  034570                     CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
2464 021672  013737  034564  034566             MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
2465 021700  013737  034572  034600             MOV     HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
2466 021706  005037  034556                     CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
2467 021712  005037  034560                     CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
2468 021716  005037  034574                     CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
2469 021722  005037  034576                     CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
2470
2471
2472                                    ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
2473
2474 021726  012737  010000  036710             MOV     #FMT22,CYL      ;16 BITS PER WORD
2475                                                                     ;CYLINDER 0, FORMAT 16 BITS
2476 021734  112737  000000  036713             MOVB    #0,SECOTR+1     ;TRACK 0
2477 021742  112737  000000  036712             MOVB    #0,SECOTR       ;SECTOR 0
2478 021750  012737  000000  036714             MOV     #0,KEY1 ;KEY1=0
2479 021756  012737  000000  036716             MOV     #0,KEY2 ;KEY2=0
2480 021764  012737  000400  036770             MOV     #256.,DAWORD    ;NO. OF DATA WORDS
2481 021772  005037  036720                     CLR     X               ;THIS IS A RFAD COMMAND
2482 021776  004537  033270                     JSR     R5,CRC  ;GO TO CALCULATE CRC
2483 022002  036710                             CYL
2484 022004  040610                             WCRC
```

```
                                        ;THIS IS TO INSERT ERROR
                                        ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
                                        ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
                                        ;THIS MOVE
022006  012737  177760  040630          MOV     #177760,DISK+2  ;FORCE ERROR ON BIT NUMBER 21 THRU 32
                                                                ;SO ERROR POSITION REGISTER WILL SHOW
                                                                ;22
022014  012737  010040  022170          MOV     #4128.,8$       ;INSERT POSITION REG.


                                        ;THESE ARE REGULAR SETUPS

022022  004737  032054                  JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
022026  012777  177374  157176          MOV     #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
022034  012777  002544  157172          MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
022042  112746  000000                  MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
022046  112766  000000  000001          MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
022054  012677  157164                  MOV     (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
022060  012777  010000  157162          MOV     #FMT22,@RHOF    ;16 BITS PER WORD
                                                                ;ECC CORRECTION NOT INHIBIT
                                                                ;BECAUSE ECC IS NOT GOING
                                                                ;TO BE CHECKED
022066  005077  157160                  CLR     @RHCA           ;CYLINDER 0
022072  004737  032116                  JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
022076  104401  053407                  TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                ;TINUE TESTING IF BOTH AREN'T TRUE
022102  000000                          HALT                    ;STOP THE TEST
022104  013711  001460                  MOV     REFOR,@R1       ;READ HEADER AND DATA=72
022110  005037  001406                  CLR     ERFLG$  ;CLEAR ERROR FLAG
022114  004737  036550                  JSR     PC,COMHD        ;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 "COMHD" 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

022120  005737  001406                  TST     ERFLG$  ;ANY ERRORS ALREADY THERE
022124  001106                          BNE     TST57           ;BRANCH IF YES
022126  004737  031554                  JSR     PC,PUTREG       ;SAVE REGISTERS
022132  022737  100000  001316          CMP     #DCK,ER1        ;ONLY DATA CHECK ERROR SHOULD BE SET
022140  001401                          BEQ     6$              ;BRANCH IF YES
022142  104032                          EMT     32
                                                                ;ZERO
                                                                ;ONLY 11 OF THE 32 BITS CAN BE SEEN
                                                                ;IN THE PATERN REGISTER
                                                                ;DCK SHOULD BE SET IN RHER1
022144  013746  034556          6$:     MOV     GECC1,-(SP)     ;GET PATTERN REGISTER
022150  042716  174000                  BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
```

```
2539 022154  022637  001346                CMP     (SP)+,EC2       ;COMPARE PATTERN REGISTER
2540 022160  001401                        BEQ     7$              ;BRANCH IF GOOD
2541 022162  104032                        EMT     32
2542
2543 022164  004037  035200          7$:   JSR     R0,ECORR        ;GO TO ECC CORRECTION PROCESS
2544 022170  000000                  8$:   .WORD                   ;EXPECTED POSITION REG. WHEN CORRECTION
2545                                                                ;IS COMPLETE
2546
2547
2548
2549 022172  004737  031554                JSR     PC,PUTREG       ;SAVE REGISTERS
2550 022176  022737  100100  001316        CMP     #DCK!ECH,ER1    ;WITH ERRORS INSERTED IN BIT POSITION 21
2551                                                                ;THRU 32 HARD ERROR BIT SHOULD SET
2552 022204  001401                        BEQ     9$              ;BRANCH IF GOOD
2553 022206  104036                        EMT     36
2554                                                                ;32 ECH SHOULD SET
2555
2556
2557
2558 022210                          9$:
     022210  004737  032306                JSR     PC,CHECKE       ;CHECK THAT BITS = 1
     022214  104401  053407                TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF THEY DON'T
     022220  000000                        HALT                    ;STOP THE TEST
2559 022222  012700  001500                MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
2560 022226  012720  010000                MOV     #0!FMT22.,(R0)+ ;CYLINDER 0
2561 022232  112746  000000                MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
2562 022236  112766  000000  000001        MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
2563 022244  012620                        MOV     (SP)+,(R0)+     ;GET TRACK/SECTOR IN BUFFER
2564 022246  012720  000000                MOV     #0,(R0)+        ;KEY1 IN BUFFER
2565 022252  012720  000000                MOV     #0,(R0)+        ;KEY2 IN BUFFER
2566 022256  012701  000400                MOV     #256.,R1        ;DATA WORD COUNTER
2567 022262  012702  000000                MOV     #0,R2           ;DATA
2568 022266  010220                  3$:   MOV     R2,(R0)+        ;DATA INTO BUFFER
2569 022270  005301                        DEC     R1              ;COUNT
2570 022272  001375                        BNE     3$              ;BRANCH IF 256 NOT DONE
2571
2572                                ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
2573                                ;NOW THE INSERTED ERROR WILL BE PUT IN
2574
2575 022274  012737  177760  001512        MOV     #177760,WRFROM+<5*2>    ;INSERTED ERROR
2576
2577
2578
2579 022302  005037  001406                CLR     ERFLG$          ;CLEAR ERROR FLAG
2580 022306  004737  031554                JSR     PC,PUTREG       ;SAVE REGISTERS
2581
2582
2583                                ;NOW READ DATA BUFFER WILL BE CHECKED
2584
2585 022312  004037  032756                JSR     R0,COMPAR       ;CHECK
2586 022316  001500                        WRFROM                  ;GOOD BUFFER
2587 022320  002544                        REINTO                  ;TEST BUFFER
2588 022322  000404                        4+256.                  ;NUMBER OF WORDS CHECKED
2589 022324  022332                        4$                      ;RETURN POINT FOR ERROR HEADER
2590 022326  022336                        5$                      ;RETURN POINT FOR ERROR DATA
2591 022330  022342                        TST57                   ;RETURN FOR GOOD COMPARISON
2592 022332                          6$:
```

```
T56     READ ECC ENABLED 1C

        022332  104004                          EMT     4
  2593  022334  000207                          RTS     PC              ;RETURN TO ''COMPAR''
  2594  022336                          5$:
        022336  104005                          EMT     5
  2595                                                                  ;HEADER WORDS
  2596                                                                  ;5 TO 260 ARE DATA WORDS
  2597  022340  000207                          RTS     PC              ;RETURN TO ''COMPAR''
  2598
  2605                                  ;;*************************************************************************
                                        ;*TEST 57        WRITE ECC TEST 2
                                        ;*THIS IS A WRITE ECC TEST
                                        ;*WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
                                        ;*TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
                                        ;*OF ALL ONES.
                                        ;;*************************************************************************
        022342  000004                  TST57:  SCOPE
  2606  022344  012706  001100                  MOV     #STACK,SP       ;RESET STACK
  2607
  2608  022350  012737  000057  003610          MOV     #57,TSTNM       ;MOVE #57 TO TEST NUMBER
  2609  022356  012700  040530                  MOV     #SECGAP,R0      ;POINTER
  2610  022362  012701  000460                  MOV     #304.,R1        ;COUNTER
  2611  022366  005020                  1$:     CLR     (R0)+           ;CLEAR SIMULATED DISK AREA
  2612  022370  005301                          DEC     R1
  2613  022372  001375                          BNE     1$
  2614  022374  004737  032054                  JSR     PC,CLDISK       ;THIS IS USED TO SET GENERAL REGISTERS
  2615
  2616                                  ;THESE ARE FOR ECC TEST ONLY
  2617
  2618  022400  012737  177777  001424          MOV     #-1,TSECC       ;THIS IS AN ECC TEST
  2619  022406  005037  034570                  CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
  2620  022412  013737  034564  034566          MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
  2621  022420  013737  034572  034600          MOV     HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
  2622  022426  005037  034556                  CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
  2623  022432  005037  034560                  CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
  2624  022436  005037  034574                  CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
  2625  022442  005037  034576                  CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
  2626
  2627
  2628
  2629
  2630                                  ;THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
  2631
  2632  022446  012737  010000  042046          MOV     #FMT22,WCYL     ;FORMAT22=16BIT WORDS AND
  2633                                                                  ;CYLINDER 0
  2634  022454  012737  000001  042050          MOV     #1,WSECTR       ;TRACK=0, SECTOR=1
  2635  022462  005037  042052                  CLR     WKEY1           ;KEY1=0
  2636  022466  005037  042054                  CLR     WKEY2           ;KEY2=0
  2637  022472  012737  000400  042106          MOV     #256.,FNWORD    ;256 DATA WORDS
  2638  022500  004537  033270                  JSR     R5,CRC  ;GO TO CALCULATE CRC
  2639  022504  042046                          WCYL
  2640  022506  042056                          GCRC
  2641
  2642                                  ;THESE ARE REGULAR SETUPS
  2643
  2644  022510  012777  177374  156514          MOV     #-260.,@RHWC    ;256 DATA WORDS 4 HEADER WORDS
  2645  022516  012700  001500                  MOV     #WRFROM,R0      ;THESE TWO INSTRUCTIONS GETS
  2646  022522  010077  156506                  MOV     R0,@RHBA        ;ADDR. OF WRFROM INTO R0 AND
```

```
                                                        ;BUS ADDRESS REGISTER
022526  012720  010000                MOV     #FMT22,(R0)+    ;FORMAT=16 BIT WORDS
                                                        ;CYLINDER=0
022532  012720  000001        2$:     MOV     #1,(R0)+        ;TRACK=0, SECTOR=1, KEYS=0
022536  005020                        CLR     (R0)+           ;KEY1=0
022540  005020                        CLR     (R0)+           ;KEY2=0
022542  012705  000400                MOV     #256.,R5        ;COUNTER
022546  012720  177777        3$:     MOV     #-1,(R0)+       ;MOVE ALL ONES FOR DATA
022552  005305                        DEC     R5
022554  001374                        BNE     3$              ;BRANCH IF DATA NOT COMPLETE
022556  012777  000001  156460        MOV     #1,@RHDST       ;TRACK=0 SECTOR=1

022564  004737  032116                JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
022570  104401  053407                TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                        ;TINUE TESTING IF BOTH AREN'T TRUE
022574  000000                        HALT                    ;STOP THE TEST

022576  013711  001454                MOV     WRIFOR,@R1      ;GET READY FOR WRITE HEADER AND
                                                        ;DATA WITH 62 IN RHCS1
022602  005037  001406                CLR     ERFLG$  ;CLEAR ERROR FLAG
022606  012777  010000  156434        MOV     #FMT22,@RHOF    ;FORMAT BIT=1 (16 BIT WORDS)
022614  005077  156432                CLR     @RHCA           ;CYLINDER =0
022620  004737  041672                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 ONES AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
                        ;THEY ARE ALL ZEROS

022624  005737  001406                TST     ERFLG$  ;HAS ANY ERRORS OCCURED?
                                                        ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
022630  001056                        BNE     TST60           ;;BRANCH IF YES


                        ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE

022632  023737  034556  041626        CMP     GECC1,WECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
022640  001402                        BEQ     6$              ;BRANCH IF GOOD
022642  104031                        EMT     31
022644  000405                        BR      7$              ;BRANCH TO CONTINUE
022646  023737  034560  041630  6$:   CMP     GECC2,WECC2;COMPARE SOFTWARE ECC WITH HARDWARE ECC
022654  001401                        BEQ     7$              ;BRANCH IF GOOD
022656  104031                        EMT     31


022660                          7$:
022660  004737  032306                JSR     PC,CHECKE       ;CHECK THAT BITS = 1
022664  104401  053407                TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF THEY DON'T
022670  000000                        HALT                    ;STOP THE TEST


                        ;FILL ''REINTO'' BUFFER WITH EXPECTED DATA

022672  004037  031772                JSR     R0,CLAREA       ;FILL REINTO BUFFER
```

```
                                                        REINTO          ;FROM
                                                        REINTO+<255.*2> ;TO
                                .WORD   -1              ;DATA

022704  013737  034556  003544        MOV     GECC1,REINTO+<256.*2>;FILL ECC1
022712  013737  034560  003546        MOV     GECC2,REINTO+<257.*2>;FILL ECC2
022720  004037  031772              JSR     R0,CLAREA       ;FILL REST
022724  003550                      REINTO+<258.*2>         ;FROM
022726  003604                      REINTO+<272.*2>         ;TO
022730  000000                      0                       ;DATA


022732  005037  001406              CLR     ERFLG$  ;CLEAR ERROR FLAG


                        ;NOW COMPARE "DISK" BUFFER WITH "REINTO"

022736  004037  032756              JSR     R0,COMPAR       ;CHECK
022742  002544                      REINTO                  ;GOOD BUFFER
022744  040626                      DISK                    ;TEST BUFFER
022746  000402                      258.                    ;NUMBER OF WORDS CHECKED
022750  022756                      4$                      ;RETURN POINT FOR ERROR HEADER
022752  022762                      5$                      ;RETURN POINT FOR ERROR DATA
022754  022766                      TST60                   ;RETURN FOR GOOD COMPARISON
022756                  4$:
022756  104007                      EMT     7
022760  000207                      RTS     PC              ;RETURN TO COMPARE
022762                  5$:
022762  104010                      EMT     10
                                                            ;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
022764  000207                      RTS     PC              ;RETURN TO COMPARE


                        ;;**************************************************************
                        ;*TEST 60        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
                        ;;**************************************************************
022766  000004         TST60:  SCOPE
022770  012706  001100              MOV     #STACK,SP       ;RESET STACK
022774  012737  000060  003610        MOV     #60,TSTNM       ;MOVE #60 TO TEST NUMBER


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

023002  012746  177777              MOV     #-1,-(SP)       ;DATA TO BE READ
023006  012705  000400              MOV     #256.,R5        ;COUNTER
023012  012700  040626              MOV     #DISK,R0        ;START OF SIMULATED DISK DATA
```

```
2752 023016  011620                1$:    MOV    (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
2753 023020  005305                       DEC    R5              ;COUNT
2754 023022  001375                       BNE    1$              ;BRANCH IF 256 NOT COMPLETE
2755 023024  005726                       TST    (SP)+           ;UNDO -(SP)
2756 023026  022020                       CMP    (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
2757 023030  012705  000017               MOV    #15.,R5         ;1 DATA GAP
2758                                                             ;14 TOLERANCE GAP
2759 023034  005020                2$:    CLR    (R0)+           ;CLEAR DATA GAP, AND
2760 023036  005305                       DEC    R5              ;TOLERANCE GAP
2761 023040  001375                       BNE    2$              ;BRANCH IF NOT COMPLETE
2762
2763
2764 023042  004737  035352               JSR    PC,FILLEC       ;INSERT THE TWO ECC WORDS ON THE DISK
2765                                                             ;IN THE CORRECT PLACE
2766
2767                                ;THESE ARE FOR ECC TEST ONLY
2768
2769 023046  012737  177777  001424       MOV    #-1,TSECC       ;THIS IS AN ECC TEST
2770 023054  005037  034570               CLR    POSITI  ;CLEAR ERROR POSITION COUNTER
2771 023060  013737  034564  034566       MOV    NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
2772 023066  013737  034572  034600       MOV    HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
2773 023074  005037  034556               CLR    GECC1           ;ECC LOW ORDER TO BE GENERATED
2774 023100  005037  034560               CLR    GECC2           ;ECC HIGH ORDER TO BE GENERATED
2775 023104  005037  034574               CLR    DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
2776 023110  005037  034576               CLR    ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
2777
2778
2779                                ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
2780
2781 023114  012737  010000  036710       MOV    #FMT22,CYL      ;16 BITS PER WORD
2782                                                             ;CYLINDER 0, FORMAT 16 BITS
2783 023122  112737  000000  036713       MOVB   #0,SECOTR+1     ;TRACK 0
2784 023130  112737  000000  036712       MOVB   #0,SECOTR       ;SECTOR 0
2785 023136  012737  000000  036714       MOV    #0,KEY1 ;KEY1=0
2786 023144  012737  000000  036716       MOV    #0,KEY2 ;KEY2=0
2787 023152  012737  000400  036770       MOV    #256.,DAWORD    ;NO. OF DATA WORDS
2788 023160  005037  036720               CLR    X               ;THIS IS A READ COMMAND
2789 023164  004537  033270               JSR    R5,CRC  ;GO TO CALCULATE CRC
2790 023170  036710                       CYL
2791 023172  040610                       WCRC
2792
2793
2794
2795
2796                                ;THESE ARE REGULAR SETUPS
2797
2798 023174  004737  032054               JSR    PC,CLDISK       ;SETUP GENERAL REGISTERS
2799 023200  012777  177374  156024       MOV    #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
2800 023206  012777  002544  156020       MOV    #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
2801 023214  112746  000000               MOVB   #0,-(SP)        ;IN LOWER BYTE GET SECTOR
2802 023220  112766  000000  000001       MOVB   #0,1(SP)        ;GET TRACK IN HIGHER BYTE
2803 023226  012677  016012               MOV    (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
2804 023232  012777  010000  156010       MOV    #FMT22,@RHOF    ;16 BITS PER WORD
2805                                                             ;ECC CORRECTION NOT INHIBIT
2806                                                             ;BECAUSE ECC IS NOT GOING
2807                                                             ;TO BE CHECKED
2808 023240  005077  156006               CLR    @RHCA           ;CYLINDER 0
```

```
2809
2810 023244  004737  032116                  JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
     023250  104401  053407                  TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                      ;TINUE TESTING IF BOTH AREN'T TRUE
     023254  000000                          HALT                    ;STOP THE TEST
2811
2812 023256  013711  001460                  MOV     REFOR,@R1       ;READ HEADER AND DATA=72
2813 023262  005037  001406                  CLR     ERFLG$  ;CLEAR ERROR FLAG
2814 023266  004737  036550                  JSR     PC,COMHD        ;READ HEADER AND DATA
2815                                                                  ;IF THERE ARE READ ERRORS THEN
2816                                                                  ;ECC WILL NOT BE CHECKED
2817
2818
2819                                 ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
2820                                 ;FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
2821                                 ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
2822                                 ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
2823                                 ;DETECTED
2824                                 ;HEADER AND DATA ARE TO BE CHECKED.
2825                                 ;IN CHECKING READ DATA THE WRITE FROM BUFFER
2826                                 ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
2827                                 ;COMPARISONS ARE MADE
2828
2829 023272  005737  001406                  TST     ERFLG$  ;ANY ERRORS ALREADY THERE
2830 023276  001102                          BNE     TST61           ;BRANCH IF YES
2831 023300  004737  031554                  JSR     PC,PUTREG       ;SAVE REGISTERS
2832 023304  005737  001316                  TST     ER1             ;NO ERRORS SHOULD BE SET
2833 023310  001401                          BEQ     6$              ;BRANCH IF NO ERRORS SET
2834 023312  104032                          EMT     32
2835                                                                  ;ONLY 11 OF THE 32 BITS CAN BE SEEN
2836                                                                  ;IN THE PATERN REGISTER
2837                                                                  ;DCK SHOULD BE SET IN RHER1
2838 023314  013746  034556          6$:     MOV     GECC1,-(SP)     ;GET PATTERN REGISTER
2839 023320  042716  174000                  BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
2840 023324  022637  001346                  CMP     (SP)+,EC2       ;COMPARE PATTERN REGISTER
2841 023330  001401                          BEQ     7$              ;BRANCH IF GOOD
2842 023332  104032                          EMT     32
2843
2844
2845
2846
2847                                 ;ADD 16 MAINTENANCE CLOCKS TO
2848                                 ;BRING EBL DOWN
2849
2850 023334  012700  000020          7$:     MOV     #16.,R0         ;COUNTER
2851 023340  052777  000002  155712  8$:     BIS     #MCLK,@RHMR     ;SET CLOCK
2852 023346  042777  000002  155704          BIC     #MCLK,@RHMR     ;CLEAR CLOCK
2853 023354  005300                          DEC     R0              ;COUNT
2854 023356  001370                          BNE     8$              ;BRANCH IF 16 CLOCKS NOT DONE
2855 023360  004737  032306                  JSR     PC,CHECKE       ;CHECK THAT BITS = 1
     023364  104401  053407                  TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF THEY DON'T
     023370  000000                          HALT                    ;STOP THE TEST
2856 023372  012700  001500                  MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
2857 023376  012720  010000                  MOV     #0!FMT22,(R0)+  ;CYLINDER 0
2858 023402  112746  000000                  MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
2859 023406  112766  000000  000001          MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
2860 023414  012620                          MOV     (SP)+,(R0)+     ;GET TRACK/SECTOR IN BUFFER
```

```
2861 023416  012720  000000                MOV     #0,(R0)+        ;KEY1 IN BUFFER
2862 023422  012720  000000                MOV     #0,(R0)+        ;KEY2 IN BUFFER
2863 023426  012701  000400                MOV     #256.,R1        ;DATA WORD COUNTER
2864 023432  012702  177777                MOV     #-1,R2  ;DATA
2865
2866 023436  010220                3$:     MOV     R2,(R0)+        ;DATA INTO BUFFER
2867 023440  005301                        DEC     R1              ;COUNT
2868 023442  001375                        BNE     3$              ;BRANCH IF 256 NOT DONE
2869 023444  005037  001406                CLR     ERFLG$  ;CLEAR ERROR FLAG
2870 023450  004737  031554                JSR     PC,PUTREG       ;SAVE REGISTERS
2871
2872                               ;NOW READ DATA BUFFER WILL BE CHECKED
2873
2874 023454  004037  032756                JSR     R0,COMPAR       ;CHECK
2875 023460  001500                        WRFROM                  ;GOOD BUFFER
2876 023462  002544                        REINTO                  ;TEST BUFFER
2877 023464  000404                        4+256.                  ;NUMBER OF WORDS CHECKED
2878 023466  023474                        4$                      ;RETURN POINT FOR ERROR HEADER
2879 023470  023500                        5$                      ;RETURN POINT FOR ERROR DATA
2880 023472  023504                        TST61                   ;RETURN FOR GOOD COMPARISON
2881 023474                        4$:
     023474  104004                        EMT     4
2882 023476  000207                        RTS     PC              ;RETURN TO ''COMPAR''
2883 023500                        5$:
     023500  104005                        EMT     5
2884                                                               ;HEADER WORDS
2885                                                               ;5 TO 260 ARE DATA WORDS
2886 023502  000207                        RTS     PC              ;RETURN TO ''COMPAR''
2887
2896                               ;;*****************************************************************
                                   ;*TEST 61        READ ECC ENABLED 2B
                                   ;*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
                                   ;;*****************************************************************
     023504  000004                TST61:  SCOPE
2897 023506  012706  001100                MOV     #STACK,SP       ;RESET STACK
2898
2899 023512  012737  000061  003610        MOV     #61,TSTNM       ;MOVE #61 TO TEST NUMBER
2900
2901
2902                               ;SETUP FOR WHAT IS TO BE READ
2903                               ;HEADER CRC IS RESTORED FROM A SUBROUTINE
2904
2905 023520  012746  177777                MOV     #-1,-(SP)       ;DATA TO BE READ
2906 023524  012705  000400                MOV     #256.,R5        ;COUNTER
2907 023530  012700  040626                MOV     #DISK,R0        ;START OF SIMULATED DISK DATA
2908 023534  011620                1$:     MOV     (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
2909 023536  005305                        DEC     R5              ;COUNT
2910 023540  001375                        BNE     1$              ;BRANCH IF 256 NOT COMPLETE
2911 023542  005726                        TST     (SP)+           ;UNDO -(SP)
2912 023544  022020                        CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
2913 023546  012705  000017                MOV     #15.,R5 ;1 DATA GAP
2914                                                               ;14 TOLERANCE GAP
```

```
2915 023552  005020                  2$:     CLR     (R0)+           ;CLEAR DATA GAP, AND
2916 023554  005305                          DEC     R5              ;TOLERANCE GAP
2917 023556  001375                          BNE     2$              ;BRANCH IF NOT COMPLETE
2918
2919
2920 023560  004737  035352                  JSR     PC,FILLEC       ;INSERT ECC IN PROPER PLACE ON DISK
2921
2922
2923
2924                                ;THESE ARE FOR ECC TEST ONLY
2925
2926 023564  012737  177777  001424          MOV     #-1,TSECC       ;THIS IS AN ECC TEST
2927 023572  005037  034570                  CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
2928 023576  013737  034564  034566          MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
2929 023604  013737  034572  034600          MOV     HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
2930 023612  005037  034556                  CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
2931 023616  005037  034560                  CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
2932 023622  005037  034574                  CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
2933 023626  005037  034576                  CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
2934
2935
2936                                ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
2937
2938 023632  012737  010000  036710          MOV     #FMT22,CYL      ;16 BITS PER WORD
2939                                                                  ;CYLINDER 0, FORMAT 16 BITS
2940 023640  112737  000000  036713          MOVB    #0,SECOTR+1     ;TRACK 0
2941 023646  112737  000000  036712          MOVB    #0,SECOTR       ;SECTOR 0
2942 023654  012737  000000  036714          MOV     #0,KEY1 ;KEY1=0
2943 023662  012737  000000  036716          MOV     #0,KEY2 ;KEY2=0
2944 023670  012737  000400  036770          MOV     #256.,DAWORD    ;NO. OF DATA WORDS
2945 023676  005037  036720                  CLR     X               ;THIS IS A READ COMMAND
2946 023702  004537  033270                  JSR     R5,CRC  ;GO TO CALCULATE CRC
2947 023706  036710                          CYL
2948 023710  040610                          WCRC
2949
2950
2951                                ;THIS IS TO INSERT ERROR
2952                                ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
2953                                ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
2954                                ;THIS MOVE
2955
2956 023712  012737  077777  040630          MOV     #77777,DISK+2   ;FORCE ERROR ON BIT NUMBER 32
2957                                                                  ;SO ERROR POSITION REGISTER WILL SHOW
2958                                                                  ;22
2959 023720  012737  000026  024074          MOV     #22.,8$ ;INSERT POSITION REG.
2960
2961
2962                                ;THESE ARE REGULAR SETUPS
2963
2964 023726  004737  032054                  JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
2965 023732  012777  177374  155272          MOV     #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
2966 023740  012777  002544  155266          MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
2967 023746  112746  000000                  MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
2968 023752  112766  000000  000001          MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
2969 023760  012677  155260                  MOV     (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
2970 023764  012777  010000  155256          MOV     #FMT22,@RHOF    ;16 BITS PER WORD
2971                                                                  ;ECC CORRECTION NOT INHIBIT
```

```
                                                                ;BECAUSE ECC IS NOT GOING
                                                                ;TO BE CHECKED
023772  005077  155254                CLR     @RHCA             ;CYLINDER 0
023776  004737  032116                JSR     PC,CHECKT         ;CHECK THAT DVA,RDY,DPR,DRY = 1
024002  104401  053407                TYPE    ,CPHALT           ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                ;TINUE TESTING IF BOTH AREN'T TRUE
024006  000000                        HALT                      ;STOP THE TEST
024010  013711  001460                MOV     REFOR,@R1         ;READ HEADER AND DATA=72
024014  005037  001406                CLR     ERFLG$  ;CLEAR ERROR FLAG
024020  004737  036550                JSR     PC,COMHD          ;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 "COMHD" 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

024024  005737  001406                TST     ERFLG$  ,ANY ERRORS ALREADY THERE
024030  001077                        BNE     TST62             ;BRANCH IF YES
024032  004737  031554                JSR     PC,PUTREG         ;SAVE REGISTERS
024036  022737  100000  001316        CMP     #DCK,ER1          ;ONLY DATA CHECK ERROR SHOULD BE SET
024044  001401                        BEQ     6$                ;BRANCH IF YES
024046  104032                        EMT     32
                                                                ;ZERO
                                                                ;ONLY 11 OF THE 32 BITS CAN BE SEEN
                                                                ;IN THE PATERN REGISTER
                                                                ;DCK SHOULD BE SET IN RHER1
024050  013746  034556        6$:     MOV     GECC1,-(SP)       ;GET PATTERN REGISTER
024054  042716  174000                BIC     #174000,(SP)      ;KEEP ONLY 11 BITS
024060  022637  001346                CMP     (SP)+,EC2         ;COMPARE PATTERN REGISTER
024064  001401                        BEQ     7$                ;BRANCH IF GOOD
024066  104032                        EMT     32

024070  004037  035200        7$:     JSR     R0,ECORR          ;GO TO ECC CORRECTION PROCESS
024074  000026                8$:     22.                       ;EXPECTED POSITION REG. WHEN CORRECTION
                                                                ;IS COMPLETE


024076  004737  032306                JSR     PC,CHECKE         ;CHECK THAT BITS = 1
024102  104401  053407                TYPE    ,CPHALT           ;CANNOT CONTINUE TESTING IF THEY DON'T
024106  000000                        HALT                      ;STOP THE TEST
024110  012700  001500                MOV     #WRFROM,R0        ;GETTING READY TO FILL EXPECTED DATA
024114  012720  010000                MOV     #0!FMT22,(R0)+    ;CYLINDER 0
024120  112746  000000                MOVB    #0,-(SP)          ;IN LOWER BYTE GET SECTOR
024124  112766  000000  000001        MOVB    #0,1(SP)          ;GET TRACK IN HIGHER BYTE
024132  012620                        MOV     (SP)+,(R0)+       ;GET TRACK/SECTOR IN BUFFER
024134  012720  000000                MOV     #0,(R0)+          ;KEY1 IN BUFFER
024140  012720  000000                MOV     #0,(R0)+          ;KEY2 IN BUFFER
024144  012701  000400                MOV     #256.,R1          ;DATA WORD COUNTER
```

```
                                        MOV     #-1,R2          ;DATA
                                3$:     MOV     R2,(R0)+        ;DATA INTO BUFFER
                                        DEC     R1              ;COUNT
                                        BNE     3$              ;BRANCH IF 256 NOT DONE

                                ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
                                ;NOW THE INSERTED ERROR WILL BE PUT IN

                                        MOV     #77777,WRFROM+<5*2>     ;INSERTED ERROR
                                        JSR     PC,PUTREG       ;SAVE REGISTERS
                                        CLR     ERFLG$  ;CLEAR ERROR FLAG


                                ;NOW READ DATA BUFFER WILL BE CHECKED

                                        JSR     R0,COMPAR       ;CHECK
                                        WRFROM                  ;GOOD BUFFER
                                        REINTO                  ;TEST BUFFER
                                        4+256.                  ;NUMBER OF WORDS CHECKED
                                        4$                      ;RETURN POINT FOR ERROR HEADER
                                        5$                      ;RETURN POINT FOR ERROR DATA
                                        TST62                   ;RETURN FOR GOOD COMPARISON
                                4$:
                                        EMT     4
                                        RTS     PC              ;RETURN TO ''COMPAR''
                                5$:
                                        EMT     5
                                                                ;HEADER WORDS
                                                                ;5 TO 260 ARE DATA WORDS
                                        RTS     PC              ;RETURN TO ''COMPAR''

                                ;;**********************************************************************
                                ;*TEST 62        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
                                ;*TRACK 0, SECTOR 0 KEYS 0  READ HEADER AND DATA
                                ;;**********************************************************************
                                TST62:  SCOPE
                                        MOV     #STACK,SP       ;RESET STACK
                                        MOV     #62,TSTNM       ;MOVE #62 TO 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
                                1$:     MOV     (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
                                        DEC     R5              ;COUNT
                                        BNE     1$              ;BRANCH IF 256 NOT COMPLETE
                                        TST     (SP)+           ;UNDO -(SP)
                                        CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
                                        MOV     #15.,R5         ;1 DATA GAP
                                                                ;14 TOLERANCE GAP
```

```
3078 024276  005020                  2$:     CLR     (R0)+           ;CLEAR DATA GAP, AND
3079 024300  005305                          DEC     R5              ;TOLERANCE GAP
3080 024302  001375                          BNE     2$              ;BRANCH IF NOT COMPLETE
3081
3082
3083 024304  004737  035352                  JSR     PC,FILLEC       ;INSERT THE TWO ECC WORDS ON THE DISK
3084                                                                  ;IN THE CORRECT PLACE
3085
3086                                 ;THESE ARE FOR ECC TEST ONLY
3087
3088 024310  012737  177777  001424          MOV     #-1,TSECC       ;THIS IS AN ECC TEST
3089 024316  005037  034570                  CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
3090 024322  013737  034564  034566          MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
3091 024330  013737  034572  034600          MOV     HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
3092 024336  005037  034556                  CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
3093 024342  005037  034560                  CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
3094 024346  005037  034574                  CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
3095 024352  005037  034576                  CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
3096
3097
3098                                 ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
3099
3100 024356  012737  010000  036710          MOV     #FMT22,CYL      ;16 BITS PER WORD
3101                                                                  ;CYLINDER 0, FORMAT 16 BITS
3102 024364  112737  000000  036713          MOVB    #0,SECOTR+1     ;TRACK 0
3103 024372  112737  000000  036712          MOVB    #0,SECOTR       ;SECTOR 0
3104 024400  012737  000000  036714          MOV     #0,KEY1 ;KEY1=0
3105 024406  012737  000000  036716          MOV     #0,KEY2 ;KEY2=0
3106 024414  012737  000400  036770          MOV     #256.,DAWORD ;NO. OF DATA WORDS
3107 024422  005037  036720                  CLR     X               ;THIS IS A READ COMMAND
3108 024426  004537  033270                  JSR     R5,CRC  ;GO TO CALCULATE CRC
3109 024432  036710                          CYL
3110 024434  040610                          WCRC
3111
3112
3113                                 ;THIS IS TO INSERT ERROR
3114                                 ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
3115                                 ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
3116                                 ;THIS MOVE
3117
3118 024436  012737  077757  040630          MOV     #77757,DISK+2   ;FORCE ERROR ON BIT NUMBER 32 AND 21
3119                                                                  ;SO ERROR POSITION REGISTER WILL SHOW
3120                                                                  ;22
3121 024444  012737  010040  024620          MOV     #4128.,8$       ;INSERT POSITION REG.
3122
3123
3124                                 ;THESE ARE REGULAR SETUPS
3125
3126 024452  004737  032054                  JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
3127 024456  012777  177374  154546          MOV     #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
3128 024464  012777  002544  154542          MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
3129 024472  112746  000000                  MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
3130 024476  112766  000000  000001          MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
3131 024504  012677  154534                  MOV     (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
3132 024510  012777  010000  154532          MOV     #FMT22,@RHOF    ;16 BITS PER WORD
3133                                                                  ;ECC CORRECTION NOT INHIBIT
3134                                                                  ;BECAUSE ECC IS NOT GOING
```

```
                                                                ;TO BE CHECKED
024516  005077  154530                  CLR     @RHCA           ;CYLINDER 0

024522  004737  032116                  JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
024526  104401  053407                  TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                ;TINUE TESTING IF BOTH AREN'T TRUE
024532  000000                          HALT                    ;STOP THE TEST

024534  013711  001460                  MOV     REFOR,@R1       ;READ HEADER AND DATA=72
024540  005037  001406                  CLR     ERFLG$  ;CLEAR ERROR FLAG
024544  004737  036550                  JSR     PC,COMHD        ;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 ''COMHD'' 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

024550  005737  001406                  TST     ERFLG$  ;ANY ERRORS ALREADY THERE
024554  001106                          BNE     TST63           ;BRANCH IF YES
024556  004737  031554                  JSR     PC,PUTREG       ;SAVE REGISTERS
024562  022737  100000  001316          CMP     #DCK,ER1        ;ONLY DATA CHECK ERROR SHOULD BE SET
024570  001401                          BEQ     6$              ;BRANCH IF YES
024572  104032                          EMT     32
                                                                ;ZERO
                                                                ;ONLY 11 OF THE 32 BITS CAN BE SEEN
                                                                ;IN THE PATERN REGISTER
                                                                ;DCK SHOULD BE SET IN RHER1
024574  013746  034556          6$:     MOV     GECC1,-(SP)     ;GET PATTERN REGISTER
024600  042716  174000                  BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
024604  022637  001346                  CMP     (SP)+,EC2       ;COMPARE PATTERN REGISTER
024610  001401                          BEQ     7$              ;BRANCH IF GOOD
024612  104032                          EMT     32

024614  004037  035200          7$:     JSR     R0,ECORR        ;GO TO ECC CORRECTION PROCESS
024620  000000                  8$:     .WORD                   ;EXPECTED POSITION REG. WHEN CORRECTION
                                                                ;IS COMPLETE

024622  004737  031554                  JSR     PC,PUTREG       ;SAVE REGISTERS


024626  022737  100100  001316          CMP     #DCK!ECH,ER1    ;WITH ERRORS INSERTED IN BIT POSITION 21
                                                                ;AND 32 HARD ERROR BIT SHOULD SET
024634  001401                          BEQ     9$              ;BRANCH IF GOOD
024636  104036                          EMT     36
                                                                ;32 HCE SHOULD SET
```

```
3189 024640                            9$:
     024640  004737  032306                    JSR     PC,CHECKE       ;CHECK THAT BITS = 1
     024644  104401  053407                    TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF THEY DON'T
     024650  000000                            HALT                    ;STOP THE TEST
3190 024652  012700  001500                    MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
3191 024656  012720  010000                    MOV     #0!FMT22,(R0)+  ;CYLINDER 0
3192 024662  112746  000000                    MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
3193 024666  112766  000000  000001            MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
3194 024674  012620                            MOV     (SP)+,(R0)+     ;GET TRACK/SECTOR IN BUFFER
3195 024676  012720  000000                    MOV     #0,(R0)+        ;KEY1 IN BUFFER
3196 024702  012720  000000                    MOV     #0,(R0)+        ;KEY2 IN BUFFER
3197 024706  012701  000400                    MOV     #256.,R1        ;DATA WORD COUNTER
3198 024712  012702  177777                    MOV     #-1,R2          ;DATA
3199 024716  010220                    3$:     MOV     R2,(R0)+        ;DATA INTO BUFFER
3200 024720  005301                            DEC     R1              ;COUNT
3201 024722  001375                            BNE     3$              ;BRANCH IF 256 NOT DONE
3202
3203                                   ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
3204                                   ;NOW THE INSERTED ERROR WILL BE PUT IN
3205
3206 024724  012737  077757  001512            MOV     #77757,WRFROM+<5*2>     ;INSERTED ERROR
3207 024732  004737  031554                    JSR     PC,PUTREG       ;SAVE REGISTERS
3208 024736  005037  001406                    CLR     ERFLG$  ;CLEAR ERROR FLAG
3209
3210
3211                                   ;NOW READ DATA BUFFER WILL BE CHECKED
3212
3213 024742  004037  032756                    JSR     R0,COMPAR       ;CHECK
3214 024746  001500                            WRFROM                  ;GOOD BUFFER
3215 024750  002544                            REINTO                  ;TEST BUFFER
3216 024752  000404                            4+256.                  ;NUMBER OF WORDS CHECKED
3217 024754  024762                            4$                      ;RETURN POINT FOR ERROR HEADER
3218 024756  024766                            5$                      ;RETURN POINT FOR ERROR DATA
3219 024760  024772                            TST63                   ;RETURN FOR GOOD COMPARISON
3220 024762                            4$:
     024762  104004                            EMT     4
3221 024764  000207                            RTS     PC              ;RETURN TO ''COMPAR''
3222 024766                            5$:
     024766  104005                            EMT     5
3223                                                                   ;HEADER WORDS
3224                                                                   ;5 TO 260 ARE DATA WORDS
3225 024770  000207                            RTS     PC              ;RETURN TO ''COMPAR''
3226
3233                                   ;;*****************************************************************
                                       ;*TEST 63       WRITE ECC TEST 3
                                       ;*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.
                                       ;;*****************************************************************
     024772  000004                    TST63:  SCOPE
3234 024774  012706  001100                    MOV     #STACK,SP       ;RESET STACK
3235
3236 025000  012737  000063  003610            MOV     #63,TSTNM       ;MOVE #63 TO TEST NUMBER
3237 025006  012700  040530                    MOV     #SECGAP,R0      ;POINTER
3238 025012  012701  000460                    MOV     #304.,R1        ;COUNTER
3239 025016  005020                    1$:     CLR     (R0)+           ;CLEAR SIMULATED DISK AREA
```

```
3240 025020  005301                          DEC     R1
3241 025022  001375                          BNE     1$
3242 025024  004737  032054                  JSR     PC,CLDISK       ;THIS IS USED TO SET GENERAL REGISTERS
3243
3244                                 ;THESE ARE FOR ECC TEST ONLY
3245
3246 025030  012737  177777  001424          MOV     #-1,TSECC       ;THIS IS AN ECC TEST
3247 025036  005037  034570                  CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
3248 025042  013737  034564  034566          MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
3249 025050  013737  034572  034600          MOV     HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
3250 025056  005037  034556                  CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
3251 025062  005037  034560                  CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
3252 025066  005037  034574                  CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
3253 025072  005037  034576                  CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
3254
3255
3256
3257
3258                                 ;THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
3259
3260 025076  012737  010000  042046          MOV     #FMT22,WCYL     ;FORMAT22=16BIT WORDS AND
3261                                                                  ;CYLINDER 0
3262 025104  012737  000001  042050          MOV     #1,WSECTR       ;TRACK=0, SECTOR=1
3263 025112  005037  042052                  CLR     WKEY1           ;KEY1=0
3264 025116  005037  042054                  CLR     WKEY2           ;KEY2=0
3265 025122  012737  000400  042106          MOV     #256.,FNWORD    ;256 DATA WORDS
3266 025130  004537  033270                  JSR     R5,CRC  ;GO TO CALCULATE CRC
3267 025134  042046                          WCYL
3268 025136  042056                          GCRC
3269
3270                                 ;THESE ARE REGULAR SETUPS
3271
3272 025140  012777  177374  154064          MOV     #-260.,@RHWC    ;256 DATA WORDS 4 HEADER WORDS
3273 025146  012700  001500                  MOV     #WRFROM,R0      ;THESE TWO INSTRUCTIONS GETS
3274 025152  010077  154056                  MOV     R0,@RHBA        ;ADDR. OF WRFROM INTO R0 AND
3275                                                                  ;BUS ADDRESS REGISTER
3276 025156  012720  010000                  MOV     #FMT22,(R0)+    ;FORMAT=16 BIT WORDS
3277                                                                  ;CYLINDER=0
3278 025162  012720  000001          2$:     MOV     #1,(R0)+        ;TRACK=0, SECTOR=1, KEYS=0
3279 025166  005020                          CLR     (R0)+           ;KEY1=0
3280 025170  005020                          CLR     (R0)+           ;KEY2=0
3281 025172  012705  000400                  MOV     #256.,R5        ;COUNTER
3282 025176  012720  052525          3$:     MOV     #52525,(R0)+    ;MOVE ALL 52525 FOR DATA
3283 025202  005305                          DEC     R5
3284 025204  001374                          BNE     3$              ;BRANCH IF DATA NOT COMPLETE
3285 025206  012777  000001  154030          MOV     #1,@RHDST       ;TRACK=0 SECTOR=1
3286
3287 025214  004737  032116                  JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
     025220  104401  053407                  TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                      ;TINUE TESTING IF BOTH AREN'T TRUE
     025224  000000                          HALT                    ;STOP THE TEST
3288
3289 025226  013711  001454                  MOV     WRIFOR,@R1      ;GET READY FOR WRITE HEADER AND
3290                                                                  ;DATA WITH 62 IN RHCS1
3291 025232  005037  001406                  CLR     ERFLG$  ;CLEAR ERROR FLAG
3292 025236  012777  010000  154004          MOV     #FMT22,@RHOF    ;FORMAT BIT=1 (16 BIT WORDS)
3293 025244  005077  154002                  CLR     @RHCA           ;CYLINDER =0
```

```
3294 025250  004737  041672                    JSR     PC,COMWHD       ;WRITE HEADER AND DATA
3295
3296                                ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
3297                                ;FROM THE ''COMWHD'' ROUTINE THAT MEANS ALL HEADER ON DISK
3298                                ;IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
3299                                ;ALL 52525 AND WRITE DATA GAP AND TOLERANCE GAP TO SFE IF
3300                                ;THEY ARE ALL ZEROS
3301
3302 025254  005737  001406                    TST     ERFLG$  ;HAS ANY ERRORS OCCURED?
3303                                                            ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
3304 025260  001056                            BNE     TST64           ;;BRANCH IF YES
3305
3306
3307                                ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
3308 025262  023737  034556  041626            CMP     GECC1,WECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
3309 025270  001402                            BEQ     6$              ;BRANCH IF GOOD
3310 025272  104031                            EMT     31
3311 025274  000405                            BR      7$              ;BRANCH TO CONTINUE
3312 025276  023737  034560  041630  6$:       CMP     GECC2,WECC2     ;COMPARE SOFTWARE ECC WITH HARDWARE ECC
3313 025304  001401                            BEQ     7$              ;BRANCH IF GOOD
3314 025306  104031                            EMT     31
3315
3316
3317 025310                             7$:
     025310  004737  032306                    JSR     PC,CHECKE       ;CHECK THAT BITS = 1
     025314  104401  053407                    TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF THEY DON'T
     025320  000000                            HALT                    ;STOP THE TEST
3318
3319
3320
3321
3322                                ;FILL ''REINTO'' BUFFER WITH EXPECTED DATA
3323
3324 025322  004037  031772                    JSR     R0,CLAREA       ;FILL REINTO BUFFER
3325 025326  002544                            REINTO                  ;FROM
3326 025330  003542                            REINTO+<255.*2>         ;TO
3327 025332  052525                            .WORD   52525           ;DATA
3328
3329 025334  013737  034556  003544            MOV     GECC1,REINTO+<256.*2>;FILL ECC1
3330 025342  013737  034560  003546            MOV     GECC2,REINTO+<257.*2>;FILL ECC2
3331 025350  004037  031772                    JSR     R0,CLAREA       ;FILL REST
3332 025354  003550                            REINTO+<258.*2>         ;FROM
3333 025356  003604                            REINTO+<272.*2>         ;TO
3334 025360  000000                            0                       ;DATA
3335
3336
3337 025362  005037  001406                    CLR     ERFLG$  ;CLEAR ERROR FLAG
3338
3339
3340                                ;NOW COMPARE ''DISK'' BUFFER WITH ''REINTO''
3341
3342 025366  004037  032756                    JSR     R0,COMPAR       ;CHECK
3343 025372  002544                            REINTO                  ;GOOD BUFFER
3344 025374  040626                            DISK                    ;TEST BUFFER
3345 025376  000402                            258.                    ;NUMBER OF WORDS CHECKED
3346 025400  025406                            4$                      ;RETURN POINT FOR ERROR HEADER
3347 025402  025412                            5$                      ;RETURN POINT FOR ERROR DATA
```

```
3348 025404  025416                    TST64                   ;RETURN FOR GOOD COMPARISON
3349 025406                      4$:
     025406  104007                    EMT     7
3350 025410  000207                    RTS     PC              ;RETURN TO COMPARE
3351 025412                      5$:
     025412  104010                    EMT     10
3352                                                            ;DATA WORDS
3353                                                            ;WORD NOS 257 AND 258
3354                                                            ;ARE ECC WHICH ARE CHECKED
3355                                                            ;WORD NOS 259
3356                                                            ;IS DATA GAP
3357                                                            ;WORD NOS 260 TO 273
3358                                                            ;ARE TOLERANCE GAP
3359 025414  000207                    RTS     PC              ;RETURN TO COMPARE
3360
3369                             ;;*****************************************************************
                                 ;*TEST 64       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
                                 ;;*****************************************************************
     025416  000004              TST64:  SCOPE
3370 025420  012706  001100              MOV     #STACK,SP       ;RESET STACK
3371 025424  012737  000064  003610      MOV     #64.,TSTNM      ;MOVE #64 TO TEST NUMBER
3372
3373                             ;SETUP FOR WHAT IS TO BE READ
3374                             ;HEADER CRC IS RESTORED FROM A SUBROUTINE
3375
3376 025432  012746  052525              MOV     #52525,-(SP)    ;DATA TO BE READ
3377 025436  012705  000400              MOV     #256.,R5        ;COUNTER
3378 025442  012700  040626              MOV     #DISK,R0        ;START OF SIMULATED DISK DATA
3379 025446  011620              1$:     MOV     (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
3380 025450  005305                      DEC     R5              ;COUNT
3381 025452  001375                      BNE     1$              ;BRANCH IF 256 NOT COMPLETE
3382 025454  005726                      TST     (SP)+           ;UNDO -(SP)
3383 025456  022020                      CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
3384 025460  012705  000017              MOV     #15.,R5         ;1 DATA GAP
3385                                                            ;14 TOLERANCE GAP
3386 025464  005020              2$:     CLR     (R0)+           ;CLEAR DATA GAP, AND
3387 025466  005305                      DEC     R5              ;TOLERANCE GAP
3388 025470  001375                      BNE     2$              ;BRANCH IF NOT COMPLETE
3389
3390 025472  004737  035352              JSR     PC,FILLEC       ;INSERT THE TWO ECC WORDS ON THE DISK
3391                                                            ;IN THE CORRECT PLACE
3392
3393                             ;THESE ARE FOR ECC TEST ONLY
3394
3395 025476  012737  177777  001424      MOV     #-1,TSECC       ;THIS IS AN ECC TEST
3396 025504  005037  034570              CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
3397 025510  013737  034564  034566      MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
3398 025516  013737  034572  034600      MOV     HORDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
3399 025524  005037  034556              CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
3400 025530  005037  034560              CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
3401 025534  005037  034574              CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
```

```
3402 025540  005037  034576                CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
3403
3404
3405                                ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
3406
3407 025544  012737  010000  036710        MOV     #FMT22,CYL      ;16 BITS PER WORD
3408                                                                 ;CYLINDER 0, FORMAT 16 BITS
3409 025552  112737  000000  036713        MOVB    #0,SECOTR+1     ;TRACK 0
3410 025560  112737  000000  036712        MOVB    #0,SECOTR       ;SECTOR 0
3411 025566  012737  000000  036714        MOV     #0,KEY1 ;KEY1=0
3412 025574  012737  000000  036716        MOV     #0,KEY2 ;KEY2=0
3413 025602  012737  000400  036770        MOV     #256.,DAWORD    ;NO. OF DATA WORDS
3414 025610  005037  036720                CLR     X               ;THIS IS A READ COMMAND
3415 025614  004537  033270                JSR     R5,CRC  ;GO TO CALCULATE CRC
3416 025620  036710                        CYL
3417 025622  040610                        WCRC
3418
3419
3420
3421
3422                                ;THESE ARE REGULAR SETUPS
3423
3424 025624  004737  032054                JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
3425 025630  012777  177374  153374        MOV     #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
3426 025636  012777  002544  153370        MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
3427 025644  112746  000000                MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
3428 025650  112766  000000  000001        MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
3429 025656  012677  153362                MOV     (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
3430 025662  012777  010000  153360        MOV     #FMT22,@RHOF ;16 BITS PER WORD
3431                                                                 ;ECC CORRECTION NOT INHIBIT
3432                                                                 ;BECAUSE ECC IS NOT GOING
3433                                                                 ;TO BE CHECKED
3434 025670  005077  153356                CLR     @RHCA           ;CYLINDER 0
3435
3436 025674  004737  032116                JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
     025700  104401  053407                TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                     ;TINUE TESTING IF BOTH AREN'T TRUE
     025704  000000                        HALT                    ;STOP THE TEST
3437
3438 025706  013711  001460                MOV     REFOR,@R1       ;READ HEADER AND DATA=72
3439 025712  005037  001406                CLR     ERFLG$  ;CLEAR ERROR FLAG
3440 025716  004737  036550                JSR     PC,COMHD        ;READ HEADER AND DATA
3441                                                                 ;IF THERE ARE READ ERRORS THEN
3442                                                                 ;ECC WILL NOT BE CHECKED
3443
3444
3445                            ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
3446                            ;FROM THE ''COMHD'' ROUTINE THAT MEANS SECTOR GAP,
3447                            ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
3448                            ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
3449                            ;DETECTED
3450                            ;HEADER AND DATA ARE TO BE CHECKED.
3451                            ;IN CHECKING READ DATA THE WRITE FROM BUFFER
3452                            ;''WRFROM'' IS FILLED WITH EXPECTED DATA AND
3453                            ;COMPARISONS ARE MADE
3454
3455 025722  005737  001406                TST     ERFLG$  ;ANY ERRORS ALREADY THERE
```

```
3456 025726  001102                        BNE     TST65           ;BRANCH IF YES
3457 025730  004737  031554                JSR     PC,PUTREG       ;SAVE REGISTERS
3458 025734  005737  001316                TST     ER1             ;NO ERRORS SHOULD BE SET
3459 025740  001401                        BEQ     6$              ;BRANCH IF NO ERRORS SET
3460 025742  104032                        EMT     32
3461                                                               ;ONLY 11 OF THE 32 BITS CAN BE SEEN
3462                                                               ;IN THE PATERN REGISTER
3463                                                               ;DCK SHOULD BE SET IN RHER1
3464 025744  013746  034556          6$:   MOV     GECC1,-(SP)     ;GET PATTERN REGISTER
3465 025750  042716  174000                BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
3466 025754  022637  001346                CMP     (SP)+,EC2       ;COMPARE PATTERN REGISTER
3467 025760  001401                        BEQ     7$              ;BRANCH IF GOOD
3468 025762  104032                        EMT     32
3469
3470
3471
3472
3473                                 ;ADD 16 MAINTENANCE CLOCKS TO
3474                                 ;BRING EBL DOWN
3475
3476 025764  012700  000020          7$:   MOV     #16.,R0         ;COUNTER
3477 025770  052777  000002  153262  8$:   BIS     #MCLK,@RHMR     ;SET CLOCK
3478 025776  042777  000002  153254        BIC     #MCLK,@RHMR     ;CLEAR CLOCK
3479 026004  005300                        DEC     R0              ;COUNT
3480 026006  001370                        BNE     8$              ;BRANCH IF 16 CLOCKS NOT DONE
3481 026010  004737  032306                JSR     PC,CHECKE       ;CHECK THAT BITS = 1
     026014  104401  053407                TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF THEY DON'T
     026020  000000                        HALT                    ;STOP THE TEST
3482 026022  012700  001500                MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
3483 026026  012720  010000                MOV     #0!FMT22,(R0)+  ;CYLINDER 0
3484 026032  112746  000000                MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
3485 026036  112766  000000  000001        MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
3486 026044  012620                        MOV     (SP)+,(R0)+     ;GET TRACK/SECTOR IN BUFFER
3487 026046  012720  000000                MOV     #0,(R0)+        ;KEY1 IN BUFFER
3488 026052  012720  000000                MOV     #0,(R0)+        ;KEY2 IN BUFFER
3489 026056  012701  000400                MOV     #256.,R1        ;DATA WORD COUNTER
3490 026062  012702  052525                MOV     #52525,R2       ;DATA
3491 026066  010220                  3$:   MOV     R2,(R0)+        ;DATA INTO BUFFER
3492 026070  005301                        DEC     R1              ;COUNT
3493 026072  001375                        BNE     3$              ;BRANCH IF 256 NOT DONE
3494 026074  005037  001406                CLR     ERFLG$  ;CLEAR ERROR FLAG
3495 026100  004737  031554                JSR     PC,PUTREG       ;SAVE REGISTERS
3496
3497                                 ;NOW READ DATA BUFFER WILL BE CHECKED
3498
3499 026104  004037  032756                JSR     R0,COMPAR       ;CHECK
3500 026110  001500                        WRFROM                  ;GOOD BUFFER
3501 026112  002544                        REINTO                  ;TEST BUFFER
3502 026114  000404                        4+256.                  ;NUMBER OF WORDS CHECKED
3503 026116  026124                        4$                      ;RETURN POINT FOR ERROR HEADER
3504 026120  026130                        5$                      ;RETURN POINT FOR ERROR DATA
3505 026122  026134                        TST65                   ;RETURN FOR GOOD COMPARISON
3506 026124                          4$:
     026124  104004                        EMT     4
3507 026126  000207                        RTS     PC              ;RETURN TO ''COMPAR''
3508 026130                          5$:
     026130  104005                        EMT     5
```

```
3509                                                            ;HEADER WORDS
3510                                                            ;5 TO 260 ARE DATA WORDS
3511 026132  000207                     RTS     PC              ;RETURN TO ''COMPAR''
3512
3522                                ;;**************************************************************
                                    ;*TEST 65       READ ECC ENABLED 3B
                                    ;*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
                                    ;;**************************************************************
     026134  000004                 TST65:  SCOPE
3523 026136  012706  001100                 MOV     #STACK,SP       ;RESET STACK
3524 026142  012737  000065  003610         MOV     #65,TSTNM       ;MOVE #65 TO TEST NUMBER
3525
3526                                ;SETUP FOR WHAT IS TO BE READ
3527                                ;HEADER CRC IS RESTORED FROM A SUBROUTINE
3528
3529 026150  012746  052525                 MOV     #52525,-(SP)    ;DATA TO BE READ
3530 026154  012705  000400                 MOV     #256.,R5        ;COUNTER
3531 026160  012700  040626                 MOV     #DISK,R0        ;START OF SIMULATED DISK DATA
3532 026164  011620                 1$:     MOV     (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
3533 026166  005305                         DEC     R5              ;COUNT
3534 026170  001375                         BNE     1$              ;BRANCH IF 256 NOT COMPLETE
3535 026172  005726                         TST     (SP)+           ;UNDO -(SP)
3536 026174  022020                         CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
3537 026176  012705  000017                 MOV     #15.,R5 ;1 DATA GAP
3538                                                                 ;14 TOLERANCE GAP
3539 026202  005020                 2$:     CLR     (R0)+           ;CLEAR DATA GAP, AND
3540 026204  005305                         DEC     R5              ;TOLERANCE GAP
3541 026206  001375                         BNE     2$              ;BRANCH IF NOT COMPLETE
3542
3543
3544 026210  004737  035352                 JSR     PC,FILLEC       ;INSERT ECC IN PROPER PLACE ON DISK
3545
3546
3547
3548                                ;THESE ARE FOR ECC TEST ONLY
3549
3550 026214  012737  177777  001424         MOV     #-1,TSECC       ;THIS IS AN ECC TEST
3551 026222  005037  034570                 CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
3552 026226  013737  034564  034566         MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
3553 026234  013737  034572  034600         MOV     HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
3554 026242  005037  034556                 CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
3555 026246  005037  034560                 CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
3556 026252  005037  034574                 CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
3557 026256  005037  034576                 CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
3558
3559
3560                                ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
3561
3562 026262  012737  010000  036710         MOV     #FMT22,CYL      ;16 BITS PER WORD
3563                                                                 ;CYLINDER 0, FORMAT 16 BITS
3564 026270  112737  000000  036713         MOVB    #0,SECOTR+1     ;TRACK 0
```

```
3565 026276  112737  000000  036712          MOVB    #0,SECOTR       ;SECTOR 0
3566 026304  012737  000000  036714          MOV     #0,KEY1 ;KEY1=0
3567 026312  012737  000000  036716          MOV     #0,KEY2 ;KEY2=0
3568 026320  012737  000400  036770          MOV     #256.,DAWORD    ;NO. OF DATA WORDS
3569 026326  005037  036720                  CLR     X               ;THIS IS A READ COMMAND
3570 026332  004537  033270                  JSR     R5,CRC  ;GO TO CALCULATE CRC
3571 026336  036710                          CYL
3572 026340  040610                          WCRC
3573
3574
3575                                ;THIS IS TO INSERT ERROR
3576                                ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
3577                                ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
3578                                ;THIS MOVE
3579                                ;THIS CHANGES THE LAST BIT OF THE ECC
3580
3581 026342  013746  041630                  MOV     WECC2,-(SP)     ;GET LAST ECC
3582 026346  005116                          COM     (SP)            ;INVERT ALL BITS OF WECC2
3583 026350  042716  077777                  BIC     #^C100000,(SP)  ;KEEP BIT 16
3584 026354  042737  100000  041630          BIC     #100000,WECC2   ;CLEAR BIT 16 IN ECC
3585 026362  052637  041630                  BIS     (SP)+,WECC2     ;THIS WILL SET BIT 16 IF IT WAS 0
3586                                                                 ;OR WILL SET NOTHING IF IT WAS A 1
3587
3588
3589 026366  012737  010026  026542          MOV     #4118.,8$       ;INSERT POSITION REG.
3590
3591
3592                                ;THESE ARE REGULAR SETUPS
3593
3594 026374  004737  032054                  JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
3595 026400  012777  177374  152624          MOV     #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
3596 026406  012777  002544  152620          MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
3597 026414  112746  000000                  MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
3598 026420  112766  000000  000001          MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
3599 026426  012677  152612                  MOV     (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
3600 026432  012777  010000  152610          MOV     #FMT22,@RHOF ;16 BITS PER WORD
3601                                                                 ;ECC CORRECTION NOT INHIBIT
3602                                                                 ;BECAUSE ECC IS NOT GOING
3603                                                                 ;TO BE CHECKED
3604 026440  005077  152606                  CLR     @RHCA           ;CYLINDER 0
3605
3606 026444  004737  032116                  JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
     026450  104401  053407                  TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                     ;TINUE TESTING IF BOTH AREN'T TRUE
     026454  000000                          HALT                    ;STOP THE TEST
3607
3608 026456  013711  001460                  MOV     REFOR,@R1       ;READ HEADER AND DATA=72
3609 026462  005037  001406                  CLR     ERFLG$  ;CLEAR ERROR FLAG
3610 026466  004737  036550                  JSR     PC,COMHD        ;READ HEADER AND DATA
3611                                                                 ;IF THERE ARE READ ERRORS THEN
3612                                                                 ;ECC WILL NOT BE CHECKED
3613
3614
3615                                ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
3616                                ;FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
3617                                ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
3618                                ;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

026472  005737  001406                  TST     ERFLG$  ;ANY ERRORS ALREADY THERE
026476  001074                          BNE     TST66           ;BRANCH IF YES
026500  004737  031554                  JSR     PC,PUTREG       ;SAVE REGISTERS
026504  022737  100000  001316          CMP     #DCK,ER1        ;ONLY DATA CHECK ERROR SHOULD BE SET
026512  001401                          BEQ     6$              ;BRANCH IF YES
026514  104032                          EMT     32
                                                                ;ZERO
                                                                ;ONLY 11 OF THE 32 BITS CAN BE SEEN
                                                                ;IN THE PATERN REGISTER
                                                                ;DCK SHOULD BE SET IN RHER1
026516  013746  034556          6$:     MOV     GECC1,-(SP)     ;GET PATTERN REGISTER
026522  042716  174000                  BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
026526  022637  001346                  CMP     (SP)+,EC2       ;COMPARE PATTERN REGISTER
026532  001401                          BEQ     7$              ;BRANCH IF GOOD
026534  104032                          EMT     32

026536  004037  035200          7$:     JSR     R0,ECORR        ;GO TO ECC CORRECTION PROCESS
026542  010026                  8$:     4118.                   ;EXPECTED POSITION REG. WHEN CORRECTION
                                                                ;IS COMPLETE


026544  004737  032306                  JSR     PC,CHECKE       ;CHECK THAT BITS = 1
026550  104401  053407                  TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF THEY DON'T
026554  000000                          HALT                    ;STOP THE TEST
026556  012700  001500                  MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
026562  012720  010000                  MOV     #0!FMT22,(R0)+  ;CYLINDER 0
026566  112746  000000                  MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
026572  112766  000000  000001          MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
026600  012620                          MOV     (SP)+,(R0)+     ;GET TRACK/SECTOR IN BUFFER
026602  012720  000000                  MOV     #0,(R0)+        ;KEY1 IN BUFFER
026606  012720  000000                  MOV     #0,(R0)+        ;KEY2 IN BUFFER
026612  012701  000400                  MOV     #256.,R1        ;DATA WORD COUNTER
026616  012702  052525                  MOV     #52525,R2       ;DATA
026622  010220                  3$:     MOV     R2,(R0)+        ;DATA INTO BUFFER
026624  005301                          DEC     R1              ;COUNT
026626  001375                          BNE     3$              ;BRANCH IF 256 NOT DONE

                                        ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
                                        ;NOW THE INSERTED ERROR WILL BE PUT IN
                                        ;BUT INSERTED ERROR IS IN ECC SO DATA IS NOT WRONG


026630  004737  031554                  JSR     PC,PUTREG       ;SAVE REGISTERS
026634  005037  001406                  CLR     ERFLG$  ;CLEAR ERROR FLAG


                                        ;NOW READ DATA BUFFER WILL BE CHECKED

026640  004037  032756                  JSR     R0,COMPAR       ;CHECK
026644  001500                          WRFROM                  ;GOOD BUFFER
```

```
3674 026646  002544                        REINTO                  ;TEST BUFFER
3675 026650  000404                        4+256.                  ;NUMBER OF WORDS CHECKED
3676 026652  026660                        4$                      ;RETURN POINT FOR ERROR HEADER
3677 026654  026664                        5$                      ;RETURN POINT FOR ERROR DATA
3678 026656  026670                        TST66                   ;RETURN FOR GOOD COMPARISON
3679 026660                          4$:
     026660  104004                        EMT     4
3680 026662  000207                        RTS     PC              ;RETURN TO ''COMPAR''
3681 026664                          5$:
     026664  104005                        EMT     5
3682                                                               ;HEADER WORDS
3683                                                               ;5 TO 260 ARE DATA WORDS
3684 026666  000207                        RTS     PC              ;RETURN TO ''COMPAR''
3685
3695                                 ;;*****************************************************************
                                     ;*TEST 66       READ ECC ENABLED 3C
                                     ;*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
                                     ;;*****************************************************************
     026670  000004                  TST66:  SCOPE
3696 026672  012706  001100                  MOV     #STACK,SP       ;RESET STACK
3697 026676  012737  000066  00361C          MOV     #66,TSTNM       ;MOVE #66 TO TEST NUMBER
3698
3699                                 ;SETUP FOR WHAT IS TO BE READ
3700                                 ;HEADER CRC IS RESTORED FROM A SUBROUTINE
3701
3702 026704  012746  052525                  MOV     #52525,-(SP)    ;DATA TO BE READ
3703 026710  012705  000400                  MOV     #256.,R5        ;COUNTER
3704 026714  012700  040626                  MOV     #DISK,R0        ;START OF SIMULATED DISK DATA
3705 026720  011620                  1$:     MOV     (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
3706 026722  005305                          DEC     R5              ;COUNT
3707 026724  001375                          BNE     1$              ;BRANCH IF 256 NOT COMPLETE
3708 026726  005726                          TST     (SP)+           ;UNDO -(SP)
3709 026730  022020                          CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
3710 026732  012705  000017                  MOV     #15.,R5 ;1 DATA GAP
3711                                                                 ;14 TOLERANCE GAP
3712 026736  005020                  2$:     CLR     (R0)+           ;CLEAR DATA GAP, AND
3713 026740  005305                          DEC     R5              ;TOLERANCE GAP
3714 026742  001375                          BNE     2$              ;BRANCH IF NOT COMPLETE
3715
3716
3717 026744  004737  035352                  JSR     PC,FILLEC       ;INSERT THE TWO ECC WORDS ON THE DISK
3718                                                                 ;IN THE CORRECT PLACE
3719
3720                                 ;THESE ARE FOR ECC TEST ONLY
3721
3722 026750  012737  177777  001424          MOV     #-1,TSECC       ;THIS IS AN ECC TEST
3723 026756  005037  034570                  CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
3724 026762  013737  034564  034566          MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
3725 026770  013737  034572  034600          MOV     HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
3726 026776  005037  034556                  CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
3727 027002  005037  034560                  CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
```

```
3728 027006  005037  034574                CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
3729 027012  005037  034576                CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
3730
3731
3732                                ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
3733
3734 027016  012737  010000  036710        MOV     #FMT22,CYL      ;16 BITS PER WORD
3735                                                                 ;CYLINDER 0, FORMAT 16 BITS
3736 027024  112737  000000  036713        MOVB    #0,SECOTR+1     ;TRACK 0
3737 027032  112737  000000  036712        MOVB    #0,SECOTR       ;SECTOR 0
3738 027040  012737  000000  036714        MOV     #0,KEY1 ;KEY1=0
3739 027046  012737  000000  036716        MOV     #0,KEY2 ;KEY2=0
3740 027054  012737  000400  036770        MOV     #256.,DAWORD    ;NO. OF DATA WORDS
3741 027062  005037  036720                CLR     X               ;THIS IS A READ COMMAND
3742 027066  004537  033270                JSR     R5,CRC  ;GO TO CALCULATE CRC
3743 027072  036710                        CYL
3744 027074  040610                        WCRC
3745
3746
3747                                ;THIS IS TO INSERT ERROR
3748                                ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
3749                                ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
3750                                ;THIS MOVE
3751
3752 027076  012737  152652  040672        MOV     #152652,DISK+44;INSERT ERROR IN POSITION 296 THRU 304
3753                                                                 ;IN WORD NUMBER 19
3754 027104  012737  052532  040674        MOV     #52532,DISK+46;INSERT ERROR IN POSITION 305 THRU 308
3755                                                                 ;IN WORD NUMBER 20
3756 027112  012737  010040  027266        MOV     #4128.,8$       ;INSERT POSITION REG.
3757
3758
3759                                ;THESE ARE REGULAR SETUPS
3760
3761 027120  004737  032054                JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
3762 027124  012777  177374  152100        MOV     #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
3763 027132  012777  002544  152074        MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
3764 027140  112746  000000                MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
3765 027144  112766  000000  000001        MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
3766 027152  012677  152066                MOV     (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
3767 027156  012777  010000  152064        MOV     #FMT22,@RHOF    ;16 BITS PER WORD
3768                                                                 ;ECC CORRECTION NOT INHIBIT
3769                                                                 ;BECAUSE ECC IS NOT GOING
3770                                                                 ;TO BE CHECKED
3771 027164  005077  152062                CLR     @RHCA           ;CYLINDER 0
3772 027170  004737  032116                JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
     027174  104401  053407                TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                     ;TINUE TESTING IF BOTH AREN'T TRUE
     027200  000000                        HALT                    ;STOP THE TEST
3773 027202  013711  001460                MOV     REFOR,@R1       ;READ HEADER AND DATA=72
3774 027206  005037  001406                CLR     ERFLG$  ;CLEAR ERROR FLAG
3775 027212  004737  036550                JSR     PC,COMHD        ;READ HEADER AND DATA
3776                                                                 ;IF THERE ARE READ ERRORS THEN
3777                                                                 ;ECC WILL NOT BE CHECKED
3778
3779
3780                                ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
3781                                ;FROM THE "COMHD" 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

027216  005737  001406                  TST     ERFLG$  ;ANY ERRORS ALREADY THERE
027222  001111                          BNE     TST67           ;BRANCH IF YES
027224  004737  031554                  JSR     PC,PUTREG       ;SAVE REGISTERS
027230  022737  100000  001316          CMP     #DCK,ER1        ;ONLY DATA CHECK ERROR SHOULD BE SET
027236  001401                          BEQ     6$              ;BRANCH IF YES
027240  104032                          EMT     32
                                                                ;ZERO
                                                                ;ONLY 11 OF THE 32 BITS CAN BE SEEN
                                                                ;IN THE PATERN REGISTER
                                                                ;DCK SHOULD BE SET IN RHER1
027242  013746  034556          6$:     MOV     CECC1,-(SP)     ;GET PATTERN REGISTER
027246  042716  174000                  BIC     #174000,(SP)    ;KEEP ONLY 11 BITS
027252  022637  001346                  CMP     (SP)+,EC2       ;COMPARE PATTERN REGISTER
027256  001401                          BEQ     7$              ;BRANCH IF GOOD
027260  104032                          EMT     32

027262  004037  035200          7$:     JSR     R0,ECORR        ;GO TO ECC CORRECTION PROCESS
027266  000000                  8$:     .WORD                   ;EXPECTED POSITION REG. WHEN CORRECTION
                                                                ;IS COMPLETE


027270  004737  031554                  JSR     PC,PUTREG       ;SAVE REGISTERS
027274  022737  100100  001316          CMP     #DCK!ECH,ER1    ;WITH ERRORS INSERTED IN BIT POSITION 21
                                                                ;THRU 32 HARD ERROR BIT SHOULD SET
027302  001401                          BEQ     9$              ;BRANCH IF GOOD
027304  104036                          EMT     36
                                                                ;32 HCE SHOULD SET


027306                          9$:
027306  004737  032306                  JSR     PC,CHECKE       ;CHECK THAT BITS = 1
027312  104401  053407                  TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF THEY DON'T
027316  000000                          HALT                    ;STOP THE TEST
027320  012700  001500                  MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
027324  012720  010000                  MOV     #0!FMT22,(R0)+  ;CYLINDER 0
027330  112746  000000                  MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
027334  112766  000000  000001          MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
027342  012620                          MOV     (SP)+,(R0)+     ;GET TRACK/SECTOR IN BUFFER
027344  012720  000000                  MOV     #0,(R0)+        ;KEY1 IN BUFFER
027350  012720  000000                  MOV     #0,(R0)+        ;KEY2 IN BUFFER
027354  012701  000400                  MOV     #256.,R1        ;DATA WORD COUNTER
027360  012702  052525                  MOV     #52525,R2       ;DATA
027364  010220                  3$:     MOV     R2,(R0)+        ;DATA INTO BUFFER
027366  005301                          DEC     R1              ;COUNT
027370  001375                          BNE     3$              ;BRANCH IF 256 NOT DONE

                                        ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
```

```
3836                                    ;NOW THE INSERTED ERROR WILL BE PUT IN
3837
3838 027372  012737  152652  001554          MOV     #152652,WRFROM+54;INSERT ERROR IN POSITION 296 THRU 304
3839                                                                    ;IN WORD NUMBER 19 IN DATA
3840 027400  012737  052532  001556          MOV     #52532,WRFROM+56;INSERT ERROR IN POSITION 305 THRU 308
3841                                                                    ;IN WORD NUMBER 20 IN DATA
3842
3843 027406  004737  031554                  JSR     PC,PUTREG       ;SAVE REGISTERS
3844 027412  005037  001406                  CLR     ERFLG$  ;CLEAR ERROR FLAG
3845
3846
3847                                    ;NOW READ DATA BUFFER WILL BE CHECKED
3848
3849 027416  004037  032756                  JSR     R0,COMPAR       ;CHECK
3850 027422  001500                          WRFROM                  ;GOOD BUFFER
3851 027424  002544                          REINTO                  ;TEST BUFFER
3852 027426  000404                          4+256.                  ;NUMBER OF WORDS CHECKED
3853 027430  027436                          4$                      ;RETURN POINT FOR ERROR HEADER
3854 027432  027442                          5$                      ;RETURN POINT FOR ERROR DATA
3855 027434  027446                          TST67                   ;RETURN FOR GOOD COMPARISON
3856 027436                             4$:
     027436  104004                          EMT     4
3857 027440  000207                          RTS     PC              ;RETURN TO ''COMPAR''
3858 027442                             5$:
     027442  104005                          EMT     5
3859                                                                    ;HEADER WORDS
3860                                                                    ;5 TO 260 ARE DATA WORDS
3861 027444  000207                          RTS     PC              ;RETURN TO ''COMPAR''
3862
3872                                    ;;*****************************************************************
                                        ;*TEST 67        READ ECC ENABLED 3D
                                        ;*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
                                        ;;*****************************************************************
     027446  000004                     TST67:  SCOPE
3873 027450  012706  001100                  MOV     #STACK,SP       ;RESET STACK
3874
3875 027454  012737  000067  003610          MOV     #67,TSTNM       ;MOVE #67 TO TEST NUMBER
3876
3877
3878                                    ;SETUP FOR WHAT IS TO BE READ
3879                                    ;HEADER CRC IS RESTORED FROM A SUBROUTINE
3880
3881 027462  012746  052525                  MOV     #52525,-(SP)    ;DATA TO BE READ
3882 027466  012705  000400                  MOV     #256.,R5        ;COUNTER
3883 027472  012700  040626                  MOV     #DISK,R0        ;START OF SIMULATED DISK DATA
3884 027476  011620                     1$:  MOV     (SP),(R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
3885 027500  005305                          DEC     R5              ;COUNT
3886 027502  001375                          BNE     1$              ;BRANCH IF 256 NOT COMPLETE
3887 027504  005726                          TST     (SP)+           ;UNDO -(SP)
3888 027506  022020                          CMP     (R0)+,(R0)+     ;JUMP OVER THE TWO ECC WORDS
3889 027510  012705  000017                  MOV     #15.,R5         ;1 DATA GAP
```

```
3890                                                    ;14 TOLERANCE GAP
3891 027514  005020                    2$:      CLR     (R0)+           ;CLEAR DATA GAP, AND
3892 027516  005305                             DEC     R5              ;TOLERANCE GAP
3893 027520  001375                             BNE     2$              ;BRANCH IF NOT COMPLETE
3894
3895
3896 027522  004737  035352                     JSR     PC,FILLEC       ;INSERT THE TWO ECC WORDS ON THE DISK
3897                                                    ;IN THE CORRECT PLACE
3898
3899                                    ;THESE ARE FOR ECC TEST ONLY
3900
3901 027526  012737  177777  001424             MOV     #-1,TSECC       ;THIS IS AN ECC TEST
3902 027534  005037  034570                     CLR     POSITI  ;CLEAR ERROR POSITION COUNTER
3903 027540  013737  034564  034566             MOV     NCODE,NCOUNT  ;TEMPORARY N-CODE COUNTER
3904 027546  013737  034572  034600             MOV     HARDER,HADTMP  ;TEMPORARY HARD ERROR COUNTER
3905 027554  005037  034556                     CLR     GECC1           ;ECC LOW ORDER TO BE GENERATED
3906 027560  005037  034560                     CLR     GECC2           ;ECC HIGH ORDER TO BE GENERATED
3907 027564  005037  034574                     CLR     DATENV  ;CLEAR DATA ENVELOPE CLOCK COUNT
3908 027570  005037  034576                     CLR     ZCODE           ;CLEAR LEADING ZEROS CLOCK COUNT
3909
3910
3911                                    ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
3912
3913 027574  012737  010000  036710             MOV     #FMT22,CYL      ;16 BITS PER WORD
3914                                                    ;CYLINDER 0, FORMAT 16 BITS
3915 027602  112737  000000  036713             MOVB    #0,SECOTR+1     ;TRACK 0
3916 027610  112737  000000  036712             MOVB    #0,SECOTR       ;SECTOR 0
3917 027616  012737  000000  036714             MOV     #0,KEY1 ;KEY1=0
3918 027624  012737  000000  036716             MOV     #0,KEY2 ;KEY2=0
3919 027632  012737  000400  036770             MOV     #256.,DAWORD    ;NO. OF DATA WORDS
3920 027640  005037  036720                     CLR     X               ;THIS IS A READ COMMAND
3921 027644  004537  033270                     JSR     R5,CRC  ;GO TO CALCULATE CRC
3922 027650  036710                            CYL
3923 027652  040610                            WCRC
3924
3925
3926                                    ;THIS IS TO INSERT ERROR
3927                                    ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
3928                                    ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
3929                                    ;THIS MOVE
3930
3931 027654  012737  152525  040630             MOV     #152525,DISK+2  ;FORCE ERROR ON BIT NUMBER 32
3932 027662  012737  152525  041624             MOV     #152525,DISK+<255.*2>;FORCE ERROR IN BIT 4096
3933 027670  012737  010040  030044             MOV     #4128.,8$       ;INSERT POSITION REG.
3934
3935
3936                                    ;THESE ARE REGULAR SETUPS
3937
3938 027676  004737  032054                     JSR     PC,CLDISK       ;SETUP GENERAL REGISTERS
3939 027702  012777  177374  151322             MOV     #-256.-4.,@RHWC ;256. DATA 4 HEADER WORDS
3940 027710  012777  002544  151316             MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
3941 027716  112746  000000                     MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
3942 027722  112766  000000  000001             MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
3943 027730  012677  151310                     MOV     (SP)+,@RHDST    ;TRACK/SECTOR IN RHDST
3944 027734  012777  010000  151306             MOV     #FMT22,@RHOF    ;16 BITS PER WORD
3945                                                    ;ECC CORRECTION NOT INHIBIT
3946                                                    ;BECAUSE ECC IS NOT GOING
```

```
3947                                                            ;TO BE CHECKED
3948 027742  005077  151304                CLR     @RHCA       ;CYLINDER 0
3949
3950 027746  004737  032116                JSR     PC,CHECKT   ;CHECK THAT DVA,RDY,DPR,DRY = 1
     027752  104401  053407                TYPE    ,CPHALT     ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                ;TINUE TESTING IF BOTH AREN'T TRUE
     027756  000000                       HALT                ;STOP THE TEST
3951
3952 027760  013711  001460                MOV     REFOR,@R1   ;READ HEADER AND DATA=72
3953 027764  005037  001406                CLR     ERFLG$  ;CLEAR ERROR FLAG
3954 027770  004737  036550                JSR     PC,COMHD    ;READ HEADER AND DATA
3955                                                            ;IF THERE ARE READ ERRORS THEN
3956                                                            ;ECC WILL NOT BE CHECKED
3957
3958
3959                              ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
3960                              ;FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
3961                              ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
3962                              ;SYNC BYTE HAVE GONE BY AND SYNCS WERE CORRECTLY
3963                              ;DETECTED
3964                              ;HEADER AND DATA ARE TO BE CHECKED.
3965                              ;IN CHECKING READ DATA THE WRITE FROM BUFFER
3966                              ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
3967                              ;COMPARISONS ARE MADE
3968
3969 027774  005737  001406                TST     ERFLG$  ;ANY ERRORS ALREADY THERE
3970 030000  001111                        BNE     TST70       ;BRANCH IF YES
3971 030002  004737  031554                JSR     PC,PUTREG   ;SAVE REGISTERS
3972 030006  022737  100000  001316        CMP     #DCK,ER1    ;ONLY DATA CHECK ERROR SHOULD BE SET
3973 030014  001401                        BEQ     6$          ;BRANCH IF YES
3974 030016  104032                        EMT     32
3975                                                            ;ZERO
3976                                                            ;ONLY 11 OF THE 32 BITS CAN BE SEEN
3977                                                            ;IN THE PATERN REGISTER
3978                                                            ;DCK SHOULD BE SET IN RHER1
3979 030020  013746  034556          6$:   MOV     GECC1,-(SP) ;GET PATTERN REGISTER
3980 030024  042716  174000                BIC     #174000,(SP);KEEP ONLY 11 BITS
3981 030030  022637  001346                CMP     (SP)+,EC2   ;COMPARE PATTERN REGISTER
3982 030034  001401                        BEQ     7$          ;BRANCH IF GOOD
3983 030036  104032                        EMT     32
3984
3985 030040  004037  035200          7$:   JSR     R0,ECORR    ;GO TO ECC CORRECTION PROCESS
3986 030044  000000                  8$:   .WORD               ;EXPECTED POSITION REG. WHEN CORRECTION
3987                                                            ;IS COMPLETE
3988
3989
3990
3991 030046  004737  031554                JSR     PC,PUTREG   ;SAVE REGISTERS
3992 030052  022737  100100  001316        CMP     #DCK!ECH,ER1;WITH ERRORS INSERTED IN BIT POSITION 32
3993                                                            ;AND 4096 HARD ERROR BIT SHOULD SET
3994 030060  001401                        BEQ     9$          ;BRANCH IF GOOD
3995 030062  104036                        EMT     36
3996                                                            ;32 HCE SHOULD SET
3997
3998
3999
4000
```

```
4001 030064                                    9$:
     030064  004737  032306                            JSR     PC,CHECKE       ;CHECK THAT BITS = 1
     030070  104401  053407                            TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF THEY DON'T
     030074  000000                                    HALT                    ;STOP THE TEST
4002 030076  012700  001500                            MOV     #WRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
4003 030102  012720  010000                            MOV     #0!FMT22.,(R0)+ ;CYLINDER 0
4004 030106  112746  000000                            MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
4005 030112  112766  000000  000001                    MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
4006 030120  012620                                    MOV     (SP)+,(R0)+     ;GET TRACK/SECTOR IN BUFFER
4007 030122  012720  000000                            MOV     #0,(R0)+        ;KEY1 IN BUFFER
4008 030126  012720  000000                            MOV     #0,(R0)+        ;KEY2 IN BUFFER
4009 030132  012701  000400                            MOV     #256.,R1        ;DATA WORD COUNTER
4010 030136  012702  052525                            MOV     #52525,R2       ;DATA
4011 030142  010220                            3$:     MOV     R2,(R0)+        ;DATA INTO BUFFER
4012 030144  005301                                    DEC     R1              ;COUNT
4013 030146  001375                                    BNE     3$              ;BRANCH IF 256 NOT DONE
4014
4015                                           ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
4016                                           ;NOW THE INSERTED ERROR WILL BE PUT IN
4017
4018 030150  012737  152525  001512                    MOV     #152525,WRFROM+<5*2>    ;INSERTED ERROR IN BIT 32
4019 030156  012737  152525  002506                    MOV     #152525,WRFROM+<259.*2>  ;INSERT ERROR IN BIT 4096
4020
4021 030164  005037  001406                            CLR     ERFLG$  ;CLEAR ERROR FLAG
4022 030170  004737  031554                            JSR     PC,PUTREG       ;SAVE REGISTERS
4023
4024                                           ;NOW READ DATA BUFFER WILL BE CHECKED
4025
4026 030174  004037  032756                            JSR     R0,COMPAR       ;CHECK
4027 030200  001500                                    WRFROM                  ;GOOD BUFFER (CHANGED)
4028 030202  002544                                    REINTO                  ;TEST BUFFER
4029 030204  000404                                    4+256.                  ;NUMBER OF WORDS CHECKED
4030 030206  030214                                    4$                      ;RETURN POINT FOR ERROR HEADER
4031 030210  030220                                    5$                      ;RETURN POINT FOR ERROR DATA
4032 030212  030224                                    TST70                   ;RETURN FOR GOOD COMPARISON
4033 030214                                    4$:
     030214  104004                                    EMT     4
4034 030216  000207                                    RTS     PC              ;RETURN TO ''COMPAR''
4035 030220                                    5$:
     030220  104005                                    EMT     5
4036                                                                           ;HEADER WORDS
4037                                                                           ;5 TO 260 ARE DATA WORDS
4038 030222  000207                                    RTS     PC              ;RETURN TO ''COMPAR''
4039
4046                                           ;;*****************************************************************
                                               ;*TEST 70       PROGRAM INTERRUPT
                                               ;*PROGRAM INTERRUPT IS TESTED BY SETTING RDY AND IE
                                               ;*IN RHCS1 AT THE SAME TIME
                                               ;*THIS SHOULD INTERRUPT THROUGH LOCATION 254
                                               ;*THE PROCESSOR PRIORITY IS SET TO 4
                                               ;;*****************************************************************
     030224  000004                            TST70:  SCOPE
4047 030226  012737  000070  003610                    MOV     #70,TSTNM       ;MOVE #70 TO TEST NUMBER
4048 030234  012706  001100                            MOV     #STACK,SP       ;RESET STACK
4049
4050 030240  004737  032054                            JSR     PC,CLDISK       ;CLEAR DISK
4051 030244  013700  001226                            MOV     RPVEC,R0        ;GET VECTOR ADDRESS
```

```
 4052 030250  012720  030316                MOV     #RPTRP1,(R0)+   ;SET INTERRUPT VECTOR
 4053 030254  012710  000340                MOV     #PR7,(R0)       ;SET SERVICE ROUTINE PRIORITY
 4054 030260  012737  000200  177776        MOV     #PR4,PS         ;SET PROCESSOR PRIORITY
 4055 030266  012711  000300                MOV     #RDY!IE,@R1     ;RDY, IE IN RHSC1 SHOULD CAUSE INTERRUPT
 4056 030272  013737  032432  001200        MOV     TIMCNT,$TMP1    ;COUNTER
 4057 030300  005337  001200          1$:   DEC     $TMP1           ;WAIT FOR INTERRUPT
 4058 030304  001375                       BNE     1$              ;BRANCH IF NOT ZERO
 4059                                                               ;BEFORE THIS IS ZERO INTERRUPT SHOULD
 4060                                                               ;OCCUR
 4061 030306  004737  031554                JSR     PC,PUTREG       ;SAVE REGISTERS
 4062 030312  104021                        EMT     21
 4063
 4064 030314  000410                        BR      TST71           ;BRANCH TO NEXT TEST
 4065
 4066 030316  022626                RPTRP1: CMP     (SP)+,(SP)+     ;RESTORE STACK
 4067 030320  004737  031554                JSR     PC,PUTREG       ;SAVE REGISTERS
 4068 030324  022737  004200  001314        CMP     #DVA!RDY,CS1    ;'IE' SHOULD BE LOW
 4069 030332  001401                        BEQ     TST71           ;BRANCH IF GOOD
 4070 030334  104021                        EMT     21
 4071                                                               ;'IE' FAILED TO RESET
 4072
 4077                                ;;*****************************************************************
                                     ;*TEST 71        INTERRUPT AT PROCESSOR AND DISK PRIORITY SAME
                                     ;*PROCESSOR PRIORITY IS SET AT 5 (SAME AS THE DISK)
                                     ;*IE AND RDY IS SET. THIS SHOULD NOT INTERRUPT
                                     ;;*****************************************************************
      030336  000004                 TST71:  SCOPE
 4078 030340  012737  000071  003610        MOV     #71,TSTNM       ;MOVE #71 TO TEST NUMBER
 4079 030346  012706  001100                MOV     #STACK,SP       ;RESET STACK
 4080
 4081 030352  004737  032054                JSR     PC,CLDISK       ;CLEAR DISK
 4082 030356  013700  001226                MOV     RPVEC,R0        ;GET VECTOR ADDRESS
 4083 030362  012720  030422                MOV     #RPTRP2,(R0)+   ;SET INTERRUPT VECTOR
 4084 030366  012710  000340                MOV     #PR7,(R0)       ;SET SERVICE ROUTINE PRIORITY
 4085 030372  012737  000240  177776        MOV     #PR5,PS         ;SET PROCESSOR PRIORITY
 4086 030400  012711  000300                MOV     #RDY!IE,@R1     ;RDY, IE IN RHSC1 SHOULD CAUSE INTERRUPT
 4087 030404  013737  032432  001200        MOV     TIMCNT,$TMP1;COUNTER
 4088 030412  005337  001200          1$:   DEC     $TMP1           ;WAIT FOR INTERRUPT
 4089 030416  001375                       BNE     1$              ;BRANCH IF NOT ZERO
 4090                                                               ;BEFORE THIS IS ZERO INTERRUPT SHOULD
 4091                                                               ;OCCUR
 4092 030420  000404                        BR      TST72           ;NO INTERRUPT SO BRANCH
 4093
 4094 030422  022626                RPTRP2: CMP     (SP)+,(SP)+     ;RESTORE STACK
 4095 030424  004737  031554                JSR     PC,PUTREG       ;SAVE REGISTERS
 4096 030430  104021                        EMT     21
 4097                                                               ;PROCESSOR STATUS SAME
 4098                                                               ;AS DISK
 4099
 4106                                ;;*****************************************************************
                                     ;*TEST 72        END OF DRIVE
                                     ;*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
                                     ;;*****************************************************************
      030432  000004                 TST72:  SCOPE
```

```
                030434  012737  000001  001212          MOV     #1,$TIMES       ;;DO 1 ITERATION
4107            030442  004737  032054                  JSR     PC,CLDISK
4108
4109            030446  012737  000000  177776          MOV     #0,PS           ;REINSTATE PS TO 0
4110            030454  104401  030462                  TYPE    ,65$            ;;TYPE ASCIZ STRING
                030460  000424                          BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  <CRLF>/TOTAL ERRORS FOR THIS PASS ON UNIT NO./
                030532                          64$:
4111            030532  013746  001374                  MOV     UNIT,-(SP)      ;GET READY TO TYPE UNIT NUMBER
4112            030536  104405                          TYPDS
4113            030540  104401  030546                  TYPE    ,67$            ;;TYPE ASCIZ STRING
                030544  000402                          BR      66$             ;;GET OVER THE ASCIZ
                                                ;;67$:  .ASCIZ  /= /
                030552                          66$:
4114            030552  013746  001112                  MOV     $ERTTL,-(SP)    ;GET READY TO TYPE NUMBER OF ERRORS
4115            030556  104405                          TYPDS
4116            030560  005037  001112                  CLR     $ERTTL          ;CLEAR TOTAL NUMBER OF ERRORS
4117            030564  005037  001102                  CLR     $TSTNM          ;CLEAR TEST NUMBER
4118            030570  005737  001402                  TST     SELECT          ;STARTING FROM 200 ?
4119            030574  001413                          BEQ     3$              ;TEST NEXT DRIVE IF SO
4120
4121            030576  005237  001100                  INC     $PASS           ;INCREASE PASS COUNT
4122            030602  104401  030765                  TYPE    ,$ENDMG         ;TYPE END PASS #
4123            030606  013746  001100                  MOV     $PASS,-(SP)
4124            030612  104405                          TYPDS
4125            030614  104401  030762                  TYPE    ,$ENULL
4126            030620  000137  006756                  JMP     TST5            ;CONTINUE TESTING THIS DRIVE
4127
4128            030624  005337  001376          3$:     DEC     NOUNITS         ;NO. OF UNITS PRESENT DECREMENTED
4129            030630  001413                          BEQ     $EOP            ;BRANCH IF ALL DRIVES COMPLETE
4130            030632  013700  001374                  MOV     UNIT,R0         ;UNIT UNDER TEST
4131            030636  012701  001354                  MOV     #UNITS,R1       ;TABLE
4132            030642  022100                  1$:     CMP     (R1)+,R0        ;IS THIS UNIT JUST TESTED
4133            030644  001401                          BEQ     2$              ;BRANCH IF YES
4134            030646  000775                          BR      1$              ;BRANCH IF NO
4135            030650  011137  001374          2$:     MOV     (R1),UNIT       ;THIS IS NEXT UNIT
4136            030654  000137  006756                  JMP     TST5            ;TEST NEXT DRIVE
```

END OF PASS ROUTINE

```
     1                                  .SBTTL  END OF PASS ROUTINE

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

     030660                             $EOP:
     030660  000004                             SCOPE
     030662  005037  001102                     CLR     $TSTNM          ;;ZERO THE TEST NUMBER
     030666  005037  001212                     CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
     030672  005237  001100                     INC     $PASS           ;;INCREMENT THE PASS NUMBER
     030676  042737  100000  001100           BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
     030704  005327                           DEC     (PC)+           ;;LOOP?
     030706  000001                     $EOPCT: .WORD   1
     030710  003022                             BGT     $DOAGN          ;;YES
     030712  012737                             MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
     030714  000001                     $ENDCT: .WORD   1
     030716  030706                             $EOPCT
     030720  104401  030765                     TYPE    ,$ENDMG         ;;TYPE "END PASS #"
     030724  013746  001100                     MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
     030730  104405                             TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
     030732  104401  030762                     TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
     030736  013700  000042           $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
     030742  001405                           BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
     030744  000005                           RESET                   ;;CLEAR THE WORLD
     030746  004710                     $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
     030750  000240                             NOP                     ;;SAVE ROOM
     030752  000240                             NOP                     ;;FOR
     030754  000240                             NOP                     ;;ACT11
     030756                             $DOAGN:
     030756  000137                             JMP     @(PC)+          ;;RETURN
     030760  005362                     $RTNAD: .WORD   TST1
     030762     377     377     000     $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
     030765     015     012     105     $ENDMG: .ASCIZ  <15><12>/END PASS #/
```

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

031002  000000                  TESTAD: 0                       ;FIRST ADDRESS OF TEST

031004                          OPERSEL:
031004  005037  177776                  CLR     PS              ;MAKE PROCESSOR STATUS ZERO
031010  104401  031016                  TYPE    ,65$            ;;TYPE ASCIZ STRING
031014  000421                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>/THE PROGRAM WAS IN TEST NUMBER /
031060                          64$:
031060  013746  003610                  MOV     TSTNM,-(SP)     ;GET READY TO TYPE TEST
031064  104402                          TYPOC                   ;NUMBER
031066  104401  031074                  TYPE    ,67$            ;;TYPE ASCIZ STRING
031072  000414                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <CRLF>/THE LOOP BACK PC WAS /
031124                          66$:
031124  013746  001110                  MOV     $LPERR,-(SP)    ;GET READY TO TYPE LOOP BACK PC
031130  104402                          TYPOC
031132  104401  001223                  TYPE    ,$CRLF
031136  104401  031144                  TYPE    ,69$            ;;TYPE ASCIZ STRING
031142  000425                          BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <CRLF>/SET LOOP ON ERROR OR LOOP ON TEST SWITCH/
031216                          68$:
031216  104401  031224                  TYPE    ,71$            ;;TYPE ASCIZ STRING
031222  000420                          BR      70$             ;;GET OVER THE ASCIZ
                                ;;71$:  .ASCIZ  <CRLF>/TYPE THE FIRST PC OF THE TEST/
031264                          70$:
031264  104401  031272                  TYPE    ,73$            ;;TYPE ASCIZ STRING
031270  000422                          BR      72$             ;;GET OVER THE ASCIZ
                                ;;73$:  .ASCIZ  <CRLF>/ FOLLOWED BY A CARRIAGE RETURN  /<CRLF>
031336                          72$:
031336  104412                          RDOCT
031340  062716  000002                  ADD     #2,(SP)         ;GET LPADR
031344  012637  001106                  MOV     (SP)+,$LPADR
031350  104401  031356                  TYPE    ,75$            ;;TYPE ASCIZ STRING
031354  000416                          BR      74$             ;;GET OVER THE ASCIZ
                                ;;75$:  .ASCIZ  <CRLF>/TYPE THE PC WHERE YOU WANT/
031412                          74$:
```

END OF PASS ROUTINE

```
     40 031412  104401  031420                 TYPE    ,77$            ;;TYPE ASCIZ STRING
        031416  000440                         BR      76$             ;;GET OVER THE ASCIZ
                                        ;;77$:  .ASCIZ  <CRLF>/ THE PROGRAM TO LOOP BACK TO FOLLOWED BY A CARRIAGE RETURN  /<CRLF>
        031520                          76$:
     41 031520  104412                         RDOCT
     42 031522  012637  001110                 MOV     (SP)+,$LPERR    ;GET LPERR
     43 031526  013746  001106                 MOV     $LPADR,-(SP)
     44
     45                                 ;THIS CLEARS UP GARBAGE
     46 031532  005037  037024                 CLR     NOSYNC          ;CLEAR FLAG FOR HEADER ERROR COMMANDS
     47 031536  005037  001424                 CLR     TSECC                   ;CLEAR FLAG FOR ECC TEST
     48                                                                         ;WHEN =177777 IT IS AN ECC TEST
     49                                                                         ;WHEN =0 IT IS NOT AN ECC TEST
     50
     51 031542  005037  034562                 CLR     TSECCG          ;EVEN IN AN ECC TEST EVERY CLOCK
     52                                                                         ;IS NOT TC GENERATE ECC
     53                                                                         ;IF =177777 GENERATE ECC
     54                                                                         ;IF =0 DO NOT GENERATE ECC
     55 031>46  005037  001426                 CLR     TESDTE          ;DRIVE TIMING ERROR TEST
     56 031552  000002                         RTI
```

SAVE REGISTERS ROUTINE

```
                                .SBTTL  SAVE REGISTERS ROUTINE

                                ;;*****************************************************************
                                ;THIS SAVES THE CONTENTS OF ALL HARDWARE REGISTERS
                                ;IN MEMORY LOCATIONS TAGED FROM 'WC' TO 'EC2'
                                ;
                                ;
                                ;THIS IS DONE SO THAT COMPARES ARE DONE WITH SAVED LOCATIONS
                                ;AND NOT THE REGISTERS THEMSELVES.   THIS WILL MAKE
                                ;ERROR PRINTOUTS FOR GOOD AND BAD DATA ALWAYS DIFFRENT
                                ;;*****************************************************************

 031554                         PUTREG:
 031554  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 031556  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 031560  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 031562  012700  001232                 MOV     #RHWC,R0        ;STARTING ADDRESS OF REG
 031566  012701  001306                 MOV     #WC,R1          ;STARTING ADDRESS OF SAVE LOCATIONS
 031572  012702  000023                 MOV     #RHCC-RHWC+2/2,R2 ;NUMBER OF REG. INTO R2
 031576  013021                 10$:    MOV     @(R0)+,(R1)+    ;SAVE HARDWARE REG.
 031600  005302                         DEC     R2
 031602  001375                         BNE     10$
 031604  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
 031606  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
 031610  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
 031612  000207                         RTS     PC
```

```
                                .SBTTL  FLOAT 1 AND 0

                                ;;*******************************************************************
                                ;*FLOAT A ONE AND A ZERO THRU A DESIGNATED REGISTER
                                ;*ABSOLUTE ADDRESS OF REG. UNDER TEST IS IN R4
                                ;;*******************************************************************

031614  000000                MASK:   0                       ;BITS UNDER TEST
031616  000000                LERR:   0                       ;ERROR HLT ADDRESS
031620  000000                REGADR: 0

031622  012537  031614        BITST:  MOV     (R5)+,MASK      ;FETCH DATA MASK
031626  012504                        MOV     (R5)+,R4        ;GET ADDRESS OF REG. UNDER TEST
031630  010437  031620                MOV     R4,REGADR
031634  010537  031616                MOV     R5,LERR ;GET ERROR RETURN ADDR.
031640  062705  000004                ADD     #4,R5   ;MODIFY RETURN ADDR. TO JUMP OVER RTS
031644  012703  000001                MOV     #1,R3   ;INITIALIZE DATA PATTERN
031650  004737  031672        BLT1:   JSR     PC,BLT2 ;OUTPUT FLOATING ZERO
031654  004737  031672                JSR     PC,BLT2 ;OUTPUT FLOATING ONE
031660  000241                        CLC
031662  006103                        ROL     R3              ;SHIFT PATTERN
031664  005703                        TST     R3
031666  001370                        BNE     BLT1            ;BRANCH IF NOT COMPLETE
031670  000205                        RTS     R5              ;RETURN TO TEST
031672  005103                BLT2:   COM     R3              ;COMPLEMENT PATTERN
031674  012737  031702  032034        MOV     #BLT3,LAD       ;SET SCOPE LOOP
031702  010337  001124        BLT3:   MOV     R3,$GDDAT       ;STORE GOOD DATA
031706  005137  031614                COM     MASK            ;AND MASK WITH PATTERN
031712  043737  031614  001124        BIC     MASK,$GDDAT;CLEAR THE REST
031720  005137  031614                COM     MASK            ;RESTORE MASK
031724  013714  001124                MOV     $GDDAT,(R4)     ;OUTPUT TO REGISTER
031730  011437  001126                MOV     (R4),$BDDAT     ;INPUT FROM REGISTER
031734  005137  031614                COM     MASK
031740  043737  031614  001126        BIC     MASK,$BDDAT     ;AND MASK OUT RECEIVED DATA
031746  005137  031614                COM     MASK            ;RESTORE MASK
031752  023737  001124  001126        CMP     $GDDAT,$BDDAT;IS DATA CORRECT
031760  001403                        BEQ     1$              ;BRANCH IF GOOD
031762  004777  177630                JSR     PC,@LERR        ;GO TO REPORT ERROR
031766  104413                        SCOP1                   ;LOCAL SCOPE LOOP
031770  000207                1$:     RTS     PC
```

```
                                .SBTTL  CLEAR MEMORY ROUTINE

                                ;THIS CLEARS ANY BLOCK OF MEMORY
                                ;FILLING IT WITH ANY DATA
                                ;
                                ;CALL
                                ;       JSR     R0,CLAREA
                                ;       .WORD   X               ;STARTING ADDRESS OF BLOCK
                                ;       .WORD   Y
                                ;       .WORD   Z               ;DATA TO BE FILLED
                                ;
                                ;R1 WILL HAVE STARTING ADDRESS OF BLOCK TO BE FILLED
                                ;R2 AFTER SUBTRACTION WILL HAVE TWICE NUMBER OF LOCATIONS
                                ;R3 WILL HAVE DATA TO BE FILLED
                                ;
                                ;TO AVOID DIVIDE ROUTINE TWO DECREMENT R2 WILL BE USED

 031772                         CLAREA:
 031772  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 031774  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 031776  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 032000  012001                        MOV     (R0)+,R1        ;FROM
 032002  012002                        MOV     (R0)+,R2        ;TO
 032004  012003                        MOV     (R0)+,R3        ;DATA
 032006  160102                        SUB     R1,R2           ;NO. OF LOCATIONS MINUS TWO
 032010  062702  000002                ADD     #2,R2           ;GET TWICE NO OF LOCATIONS
 032014  010321                1$:     MOV     R3,(R1)+        ;MOVE IN DATA
 032016  005302                        DEC     R2
 032020  005302                        DEC     R2
 032022  001374                        BNE     1$              ;BRANCH IF NOT COMPLETE
 032024  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
 032026  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 032030  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 032032  000200                        RTS     R0              ;RETURN
```

```
                                        .SBTTL  LOCAL TRAPS
032034  000000                  LAD:    0

032036  032777  001000  147074  T.SCOP: BIT     #SW09,@SWR
032044  001402                          BEQ     1$
032046  013716  032034                  MOV     LAD,(SP)
032052  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  CLEAR DISK ROUTINE

032054  013701  001240          CLDISK: MOV     RHCS1,R1        ;R1 WILL BE CONTROL AND STATUS1
032060  013702  001236                  MOV     RHCS2,R2        ;R2 WILL BE CONTROL AND STATUS2
032064  013703  001262                  MOV     RHDS1,R3        ;R3 WILL BE DISK STATUS REGISTER1
032070  013704  001242                  MOV     RHER1,R4        ;R4 WILL BE ERROR REGISTER #1

032074  012712  000040                  MOV     #CLR,@R2        ;CLEAR ALL REG.
032100  013712  001374                  MOV     UNIT,@R2        ;REINSTATE UNIT NO.
032104  005011                          CLR     @R1             ;CLEAR FUNCTION BITS
032106  012777  177777  147142          MOV     #-1,@RHAS       ;CLEAR ATTENTION BITS
032114  000207                          RTS     PC
```

```
                                .SBTTL  CHECK DISK STATUS ROUTINES

                                ;;*****************************************************************
                                ;*THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1
                                ;*AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1
                                ;*IT ALSO CHECKS THAT ALL OTHER BITS IN THESE REGISTERS = 0
                                ;;*****************************************************************

032116  011637  001414        CHECKT: MOV     (SP),PCJSR      ;SAVE PC OF JSR+4
032122  162737  000004  001414        SUB     #4,PCJSR        ;GET PC OF JSR
032130  004737  031554                JSR     PC,PUTREG       ;SAVE REGISTERS
032134  022737  004200  001314        CMP     #DVA!RDY,CS1    ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
                                                              ;AND BE READY
032142  001423                        BEQ     3$              ;BRANCH IF GOOD TO RHDS1 CHECK

032144  032737  004000  001314        BIT     #DVA,CS1        ;BAD SO TEST DEVICE AVAILABLE
032152  001004                        BNE     1$              ;TEST READY IF DVA THERE
032154  010137  001122                MOV     R1,$BDADR       ;ADDRESS OF BAD REGISTER (RHCS1)
032160  104026                        EMT     26
                                                              ;AVAILABLE AT START OF TEST
032162  000413                        BR      3$              ;BRANCH TO RHDS1 CHECK

032164  032737  000200  001314  1$:   BIT     #RDY,CS1        ;TEST READY
032172  001003                        BNE     2$              ;IF RDY THERE BRANCH
032174  010137  001122                MOV     R1,$BDADR       ;ADDRESS OF BAD REGISTER (RHCS1)
032200  104026                        EMT     26
                                                              ;AT THE START OF TEST
032202  000403                2$:     BR      3$              ;BRANCH TO NEXT COMPARE
032204  010137  001122                MOV     R1,$BDADR       ;ADDRESS OF BAD REGISTER (RHCS1)
032210  104026                        EMT     26
                                                              ;THAN DVA AND RDY SET
                                                              ;ALL OTHER BITS SHOULD BE 0
                                                              ;AT START OF TEST

032212  013746  001336          3$:     MOV     DS1,-(SP)       ;GET RHDS1
032216  042716  001100                BIC     #VV!PROG,(SP)   ;CLEAR VV AND PROGRAMABLE BIT
032222  022726  000600                CMP     #DPR!DRY,(SP)+  ;RHDS1 SHOULD HAVE THESE SET
032226  001424                        BEQ     8$              ;RETURN TO TEST IF GOOD

032230  032737  000400  001336  4$:   BIT     #DPR,DS1        ;BAD SO TEST DRIVE PRESENT
032236  001004                        BNE     5$              ;CHECK DRY IF GOOD
032240  010337  001122                MOV     R3,$BDADR       ;ADDRESS OF BAD REGISTER (RHDS1)
032244  104026                        EMT     26
032246  000413                        BR      7$              ;BRANCH OUT
032250  032737  000200  001336  5$:   BIT     #DRY,DS1        ;TEST DRIVE READY
032256  001004                        BNE     6$              ;IF DPR WAS THERE SO BRANCH
032260  010337  001122                MOV     R3,$BDADR       ;ADDRESS OF BAD REGISTER (RHDS1)
032264  104026                        EMT     26
032266  000403                        BR      7$              ;BRANCH OUT
032270  010337  001122          6$:   MOV     R3,$BDADR       ;ADDRESS OF BAD REGISTER (RHDS1)
032274  104026                        EMT     26
                                                              ;THAN MOL, DRY, DPR, SET
                                                              ;ALL OTHER BITS SHOULD BE 0
032276  000207                7$:     RTS     PC              ;RETURN TO TEST AND HALT - FATAL ERROR

032300  062716  000006          8$:   ADD     #6,(SP)         ;ADJUST STACK PTR TO GET OVER HALT IN TEST
032304  000207                        RTS     PC              ;RETURN TO TEST AND CONTINUE TESTING
```

```
                        ;;*********************************************************************
                        ;*THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1
                        ;*AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1
                        ;*IT DOES NOT CHECK THAT OTHER BITS IN THESE REGISTERS = 0
                        ;;*********************************************************************

032306  011637  001414        CHECKE: MOV     (SP),PCJSR      ;SAVE PC OF JSR+4
032312  162737  000004  001414        SUB     #4,PCJSR        ;GET PC OF JSR
032320  004737  031554                JSR     PC,PUTREG       ;SAVE REGISTERS
032324  032737  000200  001314        BIT     #RDY,CS1        ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
                                                              ;AND BE READY
032332  001004                        BNE     1$              ;BRANCH IF GOOD
032334  010137  001122                MOV     R1,$BDADR       ;FAILING REGISTER
032340  104026                        EMT     26
                                                              ;DOES NOT HAVE DVA, RDY
032342  000427                        BR      4$              ;BRANCH

032344  032737  004000  001314  1$:   BIT     #DVA,CS1        ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
                                                              ;AND BE READY
032352  001004                        BNE     2$              ;BRANCH IF GOOD
032354  010137  001122                MOV     R1,$BDADR       ;FAILING REGISTER
032360  104026                        EMT     26
                                                              ;DOES NOT HAVE DVA, RDY
032362  000417                        BR      4$              ;BRANCH OUT
032364  032737  000200  001336  2$:   BIT     #DRY,DS1        ;RHDS1 SHOULD HAVE DPR,DRY
032372  001004                        BNE     3$              ;BRANCH IF THERE
032374  010337  001122                MOV     R3,$BDADR       ;FAILING REGISTER RHDS1
032400  104026                        EMT     26
032402  000407                        BR      4$              ;BRANCH OUT
032404  032737  000400  001336  3$:   BIT     #DPR,DS1        ;RHDS1 SHOULD HAVE DPR,DRY
032412  001004                        BNE     5$              ;BRANCH IF THERE
032414  010337  001122                MOV     R3,$BDADR       ;FAILING REGISTER RHDS1
032420  104026                        EMT     26
032422  000207                4$:     RTS     PC              ;RETURN TO TEST AND HALT - FATAL ERROR

032424  062716  000006        5$:     ADD     #6,(SP)         ;ADJUST STACK TO GET OVER HALT IN TEST
032430  000207                        RTS     PC              ;RETURN TO TEST AND CONTINUE TESTING

                        .SBTTL  WAIT LOOP
                        ;;*********************************************************************
                        ;*WAIT LOOP
                        ;*ONE  LOOP OR ONE COUNT = 5.15 MICROSEC WITH BIPOLAR MEMORY (MIN)
                        ;*ONE LOOP OR ONE COUNT = 11.86 MICROSEC WITH CORE (MIN)
                        ;*WITH CORE ERROR IS INDICATED AFTER ABOUT 650 MILLISEC (MIN)
                        ;;*********************************************************************

032432  177777        TIMCNT: 177777                          ;WAITING COUNT

032434  010046        WAIT.T: MOV     R0,-(SP)                ;SAVE R0
032436  016600  000002        MOV     2(SP),R0                ;GET ADRESS OF REG. ADDRESS
032442  010037  001204        MOV     R0,$TMP3                ;WAT PC+2 IN $TMP3
032446  162737  000002  001204        SUB     #2,$TMP3        ;WAT PC FOR TYPEOUT
032454  012037  001176        MOV     (R0)+,$TMP0             ;WAIT REGISTER ADDRESS
032460  012037  001200        MOV     (R0)+,$TMP1             ;WAIT ON BIT
032464  010066  000002        MOV     R0,2(SP)                ;RESTORE RETURN ON STACK
032470  012600                MOV     (SP)+,R0                ;RESTORE R0
```

WAIT LOOP .

```
    115 032472  013737  032432  001202          MOV     TIMCNT,$TMP2;TEMPORARY COUNT
    116
    117 032500  033777  001200  146470  1$:     BIT     $TMP1,@$TMP0    ;IS REQUIRED BIT THERE?
    118 032506  001021                          BNE     2$              ;BRANCH IF YES
    119 032510  005337  001202                  DEC     $TMP2           ;COUNT
    120 032514  001371                          BNE     1$              ;BRANCH IF NOT TIME UP
    121 032516  013737  032432  001202          MOV     TIMCNT,$TMP2;TEMPORARY COUNT
    122 032524  033777  001200  146444  3$:     BIT     $TMP1,@$TMP0    ;IS REQUIRED BIT THERE?
    123 032532  001007                          BNE     2$              ;BRANCH IF YES
    124 032534  005337  001202                  DEC     $TMP2           ;COUNT
    125 032540  001371                          BNE     3$              ;BRANCH IF NOT TIME UP
    126 032542  017737  146430  001126          MOV     @$TMP0,$BDDAT   ;REGISTER CONTENTS
    127 032550  104016                          EMT     16
    128 032552  000002                  2$:     RTI
    129
    130                                 ;CALL FOR THE ABOVE WAITLOOP IS
    131                                 ;
    132                                 ;       MOV     @A,X$           ;A CONTAINS REGISTER ADDRESS
    133                                 ;       -       -       -       ;HENCE X$ WILL HAVE ABSOLUTE REG. ADR.
    134                                 ;       -       -       -
    135                                 ;       -       -       -
    136                                 ;       WAT
    137                                 ;X$:    0                       ;ABSOLUTE REG. ADDRESS UNDER WAIT
    138                                 ;       .WORD   0               ;BIT WAITED FOR
    139                                 ;                               ;CONTINUE
```

```
                                        .SBTTL  SAVE ROUTINE
                                        ;;*****************************************************************
                                        ;THIS IS A SUBROUTINE TO READ & SAVE REGISTERS
                                        ;IN THE REGISTER TABLE TO ANY LOCATION
                                        ;THE CALL IS
                                        ;JSR     R0,SAVER
                                        ;FROM
                                        ;TO
                                        ;NUMBER OF WORDS SAVED

032554                                  SAVER:
032554  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
032556  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
032560  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
032562  012001                                  MOV     (R0)+,R1        ;FROM
032564  012002                                  MOV     (R0)+,R2        ;TO
032566  012003                                  MOV     (R0)+,R3        ;NUMBER
032570  013122                          1$:     MOV     @(R1)+,(R2)+    ;SAVE REGISTER CONTENTS
032572  005303                                  DEC     R3              ;COUNT
032574  001375                                  BNE     1$              ;BRANCH IF NOT DONE
032576  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
032600  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
032602  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
032604  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

032606  012737  010000  036710  WRCHHD: MOV     #FMT22,CYL      ;CYLINDER 0 FORMAT 16 BIT WORDS
032614  112737  000001  036713          MOVB    #1,SECOTR+1     ;TRACK=1
032622  112737  000001  036712          MOVB    #1,SECOTR       ;SECTOR=1
032630  005037  036714                  CLR     KEY1            ;KEY1=0
032634  005037  036716                  CLR     KEY2            ;KEY2=0
032640  012737  000044  036770          MOV     #36.,DAWORD     ;NO OF DATA WORDS
032646  005037  036720                  CLR     X               ;THIS IS A READ OPERATION
032652  004537  033270                  JSR     R5,CRC  ;GO TO CALCULATE CRC
032656  036710                          CYL
032660  040610                          WCRC

                                        ;THESE ARE REGULAR SETUPS

032662  004737  032054                  JSR     PC,CLDISK       ;SET UP GENERAL REGISTERS
                                                                ;AND CLEAR DISK REGISTERS
032666  012777  177730  146336          MOV     #-40.,@RHWC     ;36 DATA WORDS 4 HEADER WORDS
032674  012777  002544  146332          MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
032702  112746  000001                  MOVB    #1,-(SP)        ;SECTOR=1
032706  112766  000001  000001          MOVB    #1,1(SP)        ;TRACK=1 IN UPPER BYTE
032714  012677  146324                  MOV     (SP)+,@RHDST    ;TRACK=1, SECTOR=1 IN RHDST
032720  012777  014000  146322          MOV     #FMT22!ECI,@RHOF ;16 BIT WORDS
                                                                ;ECC CORRECTION INHIBIT BECAUSE
```

WRITE CHECK ROUTINE

```
     53
     54 032726  005077  146320          CLR     @RHCA           ;ECC LOGIC IS NOT CHECKED YET
     55 032732  004737  032116          JSR     PC,CHECKT       ;CYLINDER=0
        032736  104401  053407          TYPE    ,CPHALT         ;CHECK THAT DVA,RDY,DPR,DRY = 1
                                                                ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                                ;TINUE TESTING IF BOTH AREN'T TRUE
        032742  000000                  HALT                    ;STOP THE TEST
     56 032744  013711  001450          MOV     WRCHDT,@R1      ;WRITE CHECK HEADER AND DATA=52
     57                                                         ;INTO RHCS1
     58 032750  004737  036550          JSR     PC,COMHD        ;WRITE CHECK HEADER AND DATA
     59                                                         ;SAME AS READ HEADER AND DATA
     60
     61 032754  000207                  RTS     PC              ;RETURN TO WRITE CHECK TEST
```

COMPARE ROUTINE

```
                                .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
                                ;       $TMP1 HAS ADDRESS OF RETURN ON ERROR TO PRINT DATA
                                ;       R3 HAS NUMBER OF WORDS TO BE COMPARED
                                ;       R4 HAS ONE MORE THAN NUMBER OF WORDS TO BE COMPARED
                                ;
032756                          COMPAR:
032756  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
032760  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
032762  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
032764  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
032766  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
032770  012001                          MOV     (R0)+,R1        ;ADDRESS OF GOOD DATA BUFFER
032772  012002                          MOV     (R0)+,R2        ;ADDRESS OF TEST DATA BUFFER
032774  012003                          MOV     (R0)+,R3        ;NO OF WORDS TO BE COMPARED
032776  012037  001176                  MOV     (R0)+,$TMP0     ;RETURN ON ERROR TO PRINT HEADER
033002  012037  001200                  MOV     (R0)+,$TMP1     ;RETURN ON ERROR TO PRINT DATA
033006  011000                          MOV     (R0),R0         ;RETURN ON NO ERROR
033010  010304                          MOV     R3,R4           ;NO OF WORDS TO BE COMPARED
033012  005204                          INC     R4
033014  010437  037030          1$:     MOV     R4,ERWORD       ;FOR ERROR WORD NO
033020  022122                          CMP     (R1)+,(R2)+     ;COMPARE GOOD WITH TEST DATA
033022  001426                          BEQ     3$              ;BRANCH IF GOOD

033024  014137  001124                  MOV     -(R1),$GDDAT    ;GOOD DATA
033030  014237  001126                  MOV     -(R2),$BDDAT    ;BAD DATA
033034  160337  037030                  SUB     R3,ERWORD       ;ERROR WORD NO.
033040  005737  001406                  TST     ERFLG$  ;ANY ERRORS ALREAY THERE
033044  001003                          BNE     2$              ;BRANCH IF YES
033046  004777  146124                  JSR     PC,@$TMP0       ;RETURN TO PRINT HEADER
033052  000402                          BR      5$              ;BRANCH TO AVOID PRINTING NEXT ERROR
033054  004777  146120          2$:     JSR     PC,@$TMP1       ;RETURN TO PRINT DATA
033060  022122                  5$:     CMP     (R1)+,(R2)+     ;UNDO -(R1) AND -(R2) FOR ERRORS
033062  017746  146052                  MOV     @SWR,-(SP)      ;GET SWITCH SETTING
033066  042716  177177                  BIC     #^C600,(SP)     ;KEEP ONLY SWITCH 7 AND 8
033072  022726  000200                  CMP     #SW07,(SP)+     ;IS 7 SET AND 8 RESET
033076  001402                          BEQ     4$              ;BRANCH OUT IF YES
033100  005303                  3$:     DEC     R3              ;COUNT
033102  001344                          BNE     1$              ;BRANCH IF ALL NOT DEVICE
033104                          4$:
033104  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
033106  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
033110  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
033112  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
033114  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
033116  000200                          RTS     R0              ;RETURN TO MAIN PROGRAM
```

```
                                .SBTTL  WRITE CHECK DATA

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

033120  012 37  010000  036710  WRCHDA: MOV     #FMT22,CYL      ;CYLINDER 0 FORMAT 16 BIT WORDS
033126  112737  000001  036713          MOVB    #1,SECOTR+1     ;TRACK=1
033134  112737  000001  036712          MOVB    #1,SECOTR       ;SECTOR=1
033142  005037  036714                  CLR     KEY1            ;KEY1=0
033146  005037  036716                  CLR     KEY2            ;KEY2=0
033152  012737  000040  036770          MOV     #32.,DAWORD     ;NO OF DATA WORDS
033160  005037  036720                  CLR     X               ;THIS IS A READ OPERATION

033164  004537  033270                  JSR     R5,CRC  ;GO TO CALCULATE CRC
033170  036710                          CYL
033172  040610                          WCRC

                                ;THESE ARE REGULAR SETUPS

033174  004737  032054                  JSR     PC,CLDISK       ;SET UP GENERAL REGISTERS
                                                                ;AND CLEAR DISK REGISTERS

033200  012777  177740  146024          MOV     #-32.,@RHWC     ;36 DATA WORDS 4 HEADER WORDS
033206  012777  002544  146020          MOV     #REINTO,@RHBA   ;STARTING ADDRESS OF READ BUFFER
033214  112746  000001                  MOVB    #1,-(SP)        ;SECTOR=1
033220  112766  000001  000001          MOVB    #1,1(SP)        ;TRACK=1 IN UPPER BYTE
033226  012677  146012                  MOV     (SP)+,@RHDST    ;TRACK=1, SECTOR=1 IN RHDST
033232  012777  014000  146010          MOV     #FMT22!ECI,@RHOF        ;16 BIT WORDS
                                                                ;ECC CORRECTION INHIBIT BECAUSE
                                                                ;ECC LOGIC IS NOT CHECKED YET
033240  005077  146006          CLR     @RHCA           ;CYLINDER=0
033244  004737  032116          JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
033250  104401  053407          TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                        ;TINUE TESTING IF BOTH AREN'T TRUE
033254  000000                  HALT                    ;STOP THE TEST
033256  013711  001446          MOV     WRCHEK,@R1      ;WRITE CHECK DATA=50 INTO RHCS1
033262  004737  036550          JSR     PC,COMHD        ;WRITE CHECK HEADER AND DATA
                                                        ;SAME AS READ HEADER AND DATA

033266  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
                                ;REPEAT 64 TIMES

                                ;CALL
                                ;       JSR     R5,CRC
                                ;       .WORD   X       ;FIRST LOCATION AT
                                ;       .WORD   Y       ;PUT CRC IN WCRC FOR READ GCRC FOR WRITE

 033270                         CRC:
 033270  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 033272  012500                         MOV     (R5)+,R0        ;GET POINTER TO CYL NO.
 033274  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 033276  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 033300  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 033302  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 033304  005001                         CLR     R1              ;CLEAR WORKING LOCATION
 033306  005037  001210                 CLR     $TMP5
 033312  012737  000004  001176         MOV     #4,$TMP0        ;WORD COUNT
 033320  012037  001204         16$:    MOV     (R0)+,$TMP3     ;TEMPORARY WORD STORAGE
 033324  012737  000020  001202         MOV     #16.,$TMP2      ;BIT COUNT
 033332  013737  001204  001206         MOV     $TMP3,$TMP4     ;TEMPORARY WORD STORAGE
 033340  006037  001204         15$:    ROR     $TMP3           ;GET LSB INTO "C"
 033344  006037  001210                 ROR     $TMP5           ;GET ABOVE "C" INTO $TMP5
 033350  032701  000001                 BIT     #BIT0,R1        ;IS POSITION 15 HIGH
 033354  001403                         BEQ     1$              ;BRANCH IF POSITION 16 LOW
 033356  012703  100000                 MOV     #BIT15,R3       ;GET POSITION 16
 033362  000401                         BR      2$
 033364  005003         1$:     CLR     R3              ;GET POSITION 16
 033366  063703  001210         2$:     ADD     $TMP5,R3        ;XOR POSITION 16 WITH D
                                                                ;TO GIVE B
 033372  032701  040000                 BIT     #BIT14,R1       ;IS POSITION 2 HIGH
 033376  001403                         BEQ     3$              ;BRANCH IF POSITION 2 LOW
 033400  012702  100000                 MOV     #BIT15,R2       ;GET POSITION 2
 033404  000401                         BR      4$
 033406  005002         3$:     CLR     R2              ;GET POSITION 2
 033410  060302         4$:     ADD     R3,R2           ;XOR B WITH POSITION 2
```

CRC GENERATION ROUTINE

```
                                                                ;TO GIVE C
033412  032701  000002          BIT     #BIT1,R1              ;IS POSITION 15 HIGH
033416  001403                  BEQ     5$                    ;BRANCH IF POSITION 15 LOW
033420  012704  100000          MOV     #BIT15,R4             ;GET POSITION 15
033424  000401                  BR      6$
033426  005004          5$:     CLR     R4                    ;GET POSITION 15
033430  060304          6$:     ADD     R3,R4                 ;XOR POSITION 15 WITH B
                                                                ;TO GIVE E
033432  006037  001206          ROR     $TMP4                 ;GET LSB INTO "C"
033436  006001                  ROR     R1                    ;GET ABOVE C INTO R1
033440  005703                  TST     R3                    ;TEST B
033442  100403                  BMI     7$                    ;BRANCH IF B=1
033444  042701  100000          BIC     #BIT15,R1             ;SET B IN POSITION 1
033450  000402                  BR      10$
033452  052701  100000  7$:     BIS     #BIT15,R1             ;SET B IN POSITION 1
033456  005702          10$:    TST     R2                    ;TEST C
033460  100403                  BMI     11$                   ;BRANCH IF C=1
033462  042701  020000          BIC     #BIT13,R1             ;GET C IN POSITION 3
033466  000402                  BR      12$
033470  052701  020000  11$:    BIS     #BIT13,R1             ;GET C IN POSITION 3
033474  005704          12$:    TST     R4                    ;TEST E
033476  100403                  BMI     13$                   ;BRANCH IF E=1
033500  042701  000001          BIC     #BIT0,R1              ;GET E IN POSITION 16
033504  000402                  BR      14$
033506  052701  000001  13$:    BIS     #BIT0,R1              ;GET E IN POSITION 16
033512  005337  001202  14$:    DEC     $TMP2                 ;BIT COUNTER
033516  001310                  BNE     15$                   ;BRANCH IF 16 NOT DONE
033520  005337  001176          DEC     $TMP0                 ;WORD COUNTER
033524  001275                  BNE     16$                   ;BRANCH IF 4 NOT DONE
033526  010135                  MOV     R1,@(R5)+             ;PUT CRC WHERE DESIRED
033530  012604                  MOV     (SP)+,R4              ;;POP STACK INTO R4
033532  012603                  MOV     (SP)+,R3              ;;POP STACK INTO R3
033534  012602                  MOV     (SP)+,R2              ;;POP STACK INTO R2
033536  012601                  MOV     (SP)+,R1              ;;POP STACK INTO R1
033540  012600                  MOV     (SP)+,R0              ;;POP STACK INTO R0
033542  000205                  RTS     R5
```

SIMULATED DISK SETUP

```
                                .SBTTL  SIMULATED DISK SETUP

                                ;THIS IS A SUBROUTINE TO SET UP THE SIMULATOR DISK FOR
                                ;CYLINDER 0 (16 BITS PER WORD)
                                ;TRACK 1, SECTOR 1
                                ;KEY1 1
                                ;KEY2 1
                                ;CRC THROUGH THE JSR R5,CRC
                                ;256 WORDS OF 177400

                                ;CALL
                                ;       JSR     PC,SETDSK

033544                          SETDSK:
033544  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
033546  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
033550  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
033552  012700  177400                  MOV     #177400,R0      ;DATA IN THE DISK
033556  012701  000400                  MOV     #256.,R1        ;COUNTER
033562  012702  040626                  MOV     #DISK,R2        ;START OF SIMULATOR DISK
033566  010022                  1$:     MOV     R0,(R2)+        ;MOVE IN DATA
033570  005301                          DEC     R1              ;COUNT FOR 256
033572  001375                          BNE     1$              ;BRANCH IF 256 NOT COMPLETE
033574  012701  000021                  MOV     #17.,R1         ;2 ECC WORDS, 1 DATA GAP
                                                                ;14 TOLERANCE GAP
033600  005022                  2$:     CLR     (R2)+           ;CLEAR ECC,DATA GAP AND
                                                                ;TOLERANCE GAP
033602  005301                          DEC     R1              ;COUNT
033604  001375                          BNE     2$              ;BRANCH IF NOT COMPLETE

                                ;NOW SET UP FOR DISKLESS USE

033606  012737  010000  036710          MOV     #FMT22,CYL      ;CYLINDER 0 (16 BIT WORDS)
033614  112737  000001  036713          MOVB    #1,SECOTR+1     ;TRACK=1
033622  112737  000001  036712          MOVB    #1,SECOTR       ;SECTOR=1
033630  012737  000001  036714          MOV     #1,KEY1 ;KEY1=1
033636  012737  000001  036716          MOV     #1,KEY2 ;KEY2=1
033644  013737  000400  036770          MOV     256.,DAWORD     ;NO. OF DATA WORDS
033652  004537  033270                  JSR     R5,CRC  ;GO TO CALCULATE CRC
033656  036710                          CYL                     ;FIRST CRC WORD
033660  040610                          WCRC                    ;PUT CALCULATED CRC
033662  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
033664  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
033666  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
033670  000207                          RTS     PC
```

```
                        .SBTTL  CHECK HCE ROUTINE

                        ;THIS IS A SUBROUTINE TO CHECK HEADER COMPARE ERROR
                        ;(BIT #7) AND CRC ERROR (BIT #8)

                        ;CALL
                        ;JSR R0,HCCRCE
                        ;       COM             ;COMMAND-READ HEADER AND DATA
                        ;                       ;       -WRITE DATA
                        ;       C               ;CYLINDER
                        ;       S               ;SECTOR
                        ;       T               ;TRACK
                        ;       -N.             ;WORD COUNT
                        ;       B               ;RHBA BUFFER START
                        ;       X               ;1=WRITE DATA 0=READ
                        ;       H               ;H=1 HEADER CHECK, H=0 CRC CHECK

033672  010037  001414  HCCRCE: MOV     R0,PCJSR        ;SAVE PC OF JSR+4
033676  162737  000004  001414  SUB     #4,PCJSR        ;GET PC OF JSR
033704  004737  032054          JSR     PC,CLDISK       ;INIT AND SETUP GENERAL REG.
033710  004737  032116          JSR     PC,CHECKT       ;CHECK THAT DVA,RDY,DPR,DRY = 1
033714  104401  053407          TYPE    ,CPHALT         ;AND THAT NO OTHERS = 1.  CANNOT CON-
                                                        ;TINUE TESTING IF BOTH AREN'T TRUE
033720  000000                  HALT                    ;STOP THE TEST
033722  011037  001210          MOV     (R0),$TMP5      ;SAVE COMMAND
033726  012011                  MOV     (R0)+,@R1       ;COMMAND
033730  012077  145316          MOV     (R0)+,@RHCA     ;CYLINDER
033734  112046                  MOVB    (R0)+,-(SP)     ;SECTOR
033736  105720                  TSTB    (R0)+           ;UP DATE R0
033740  112066  000001          MOVB    (R0)+,1(SP)     ;TRACK
033744  105720                  TSTB    (R0)+           ;UPDATE R0
033746  012677  145272          MOV     (SP)+,@RHDST    ;TRACK SECTOR
033752  012077  145254          MOV     (R0)+,@RHWC     ;NO. OF DATA WORDS +4 HEADER
                                                        ;IF A READ HEADER AND DATA
033756  012077  145252          MOV     (R0)+,@RHBA     ;STARTING ADDRESS OF BUFFER
033762  012037  036720          MOV     (R0)+,X ;X=0 READ HEADER AND DATA
                                                        ;X=1 WRITE DATA
033766  012777  014000  145254  MOV     #FMT22!ECI,@RHOF ;16 BITS PER WORD
                                                        ;ECC CORRECTION INHIBIT
033774  005037  001406          CLR     ERFLG$  ;CLEAR ERROR FLAG
034000  004737  036550          JSR     PC,COMHD        ;COMMAND

                        ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
                        ;FROM THE "COMHD" 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.

034004  004737  031554          JSR     PC,PUTREG       ;SAVE REGISTERS
034010  005737  001406          TST     ERFLG$  ;ANY ERRORS ALREADY THERE
034014  001034                  BNE     10$             ;BRANCH IF YES
034016  005737  036720          TST     X               ;IS THIS A READ
034022  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

034024  004037  032756                        JSR     R0,COMPAR       ;CHECK
034030  001500                                WRFROM                  ;GOOD DATA
034032  002544                                REINTO                  ;TEST BUFFER
034034  000400                                256.                    ;4 HEADER 252 DATA
034036  034044                                1$                      ;RETURN POINT FOR ERROR HEADER
034040  034050                                2$                      ;RETURN POINT FOR ERROR DATA
034042  034106                                10$                     ;RETURN FOR GOOD COMPARISON
034044                                  1$:
034044  104004                                EMT     4
034046  000207                                RTS     PC              ;RETURN TO COMPARISON SUBROUTINE
034050                                  2$:
034050  104005                                EMT     5
                                                                        ;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
034052  000207                                RTS     PC              ;RETURN TO COMPARISON

034054  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

034056  004037  032756                  3$:   JSR     R0,COMPAR       ;CHECK
034062  002544                                REINTO                  ;GOOD DATA BUFFER
034064  040626                                DISK                    ;TEST BUFFER
034066  000400                                256.
034070  034076                                4$                      ;RETURN POINT FOR ERROR HEADER
034072  034102                                5$                      ;RETURN POINT FOR ERROR DATA
034074  034106                                10$                     ;RETURN POINT FOR GOOD COMPARISON
034076                                  4$:
034076  104004                                EMT     4
034100  000207                                RTS     PC              ;RETURN TO COMPARISON SUBROUTINE
034102                                  5$:
034102  104005                                EMT     5
                                                                        ;WORDS THE SHOULD NOT
                                                                        ;HAVE BEEN CHANGED BY THE
                                                                        ;WRITE COMMAND
034104  000207                                RTS     PC              ;RETURN TO COMPARISON SUBROUTINE
034106  005720                          10$:  TST     (R0)+           ;IS THIS A HCRC ON HCE CHECK?
034110  001442                                BEQ     6$              ;BRANCH IF HCRC
034112  022737  000072  001210                CMP     #72,$TMP5       ;IS THIS A READ COMMAND
034120  001417                                BEQ     11$             ;BRANCH IF YES
034122  017737  145114  001126                MOV     @RHER1,$BDDAT   ;TEST DATA
034130  022737  000200  001126                CMP     #HCE,$BDDAT     ;ONLY HEADER COMPARE BIT?
                                                                        ;SHOULD BE SET
034136  001470                                BEQ     7$              ;BRANCH IF GOOD
034140  013737  001242  031620                MOV     RHER1,REGADR ;REGISTER ADDRESS RHER1
```

```
                                                MOV     #HCE,$GDDAT     ;GOOD DATA
                                                EMT     27
                                                                        ;HEADER ONLY HCE SHOULD
                                                BR      7$              ;BE SET
                                        11$:
                                                MOV     @RHER1,$BDDAT   ;TEST DATA
                                                CMP     #DCK!HCE,$BDDAT ;ONLY HEADER COMPARE BIT?
                                                                        ;SHOULD BE SET
                                                                        ;DCK IS SET BECAUSE ECC IS NOT READ
                                                BEQ     7$              ;BRANCH IF GOOD
                                                MOV     RHER1,REGADR ;REGISTER ADDRESS RHER1
                                                MOV     #DCK!HCE,$GDDAT ;GOOD DATA
                                                EMT     27
                                                                        ;HEADER ONLY HCE SHOULD
                                                BR      7$              ;BE SET
                                        6$:     CMP     #72,$TMP5       ;IS THIS A READ COMMAND?
                                                BEQ     12$             ;BRANCH IF A READ
                                                MOV     @RHER1,$BDDAT   ;TEST DATA
                                                CMP     #HCRC,$BDDAT    ;ONLY CRC ERROR SHOULD BE THERE
                                                BEQ     7$
                                                MOV     RHER1,REGADR ;REG. ADDR = RHER1
                                                MOV     #HCRC,$GDDAT    ;GOOD DATA
                                                EMT     27
                                                                        ;SHOULD BE SET
                                                BR      7$              ;BRANCH OUT
                                        12$:    MOV     @RHER1,$BDDAT   ;TEST DATA

                                                CMP     #DCK!HCRC,$BDDAT;HCRC AND DCK SHOULD BE SET
                                                                        ;DCK IS SET BECAUSE ECC IS NOT READ
                                                BEQ     7$              ;BRANCH IF GOOD
                                                MOV     #DCK!HCRC,$GDDAT;GOOD DATA
                                                MOV     RHER1,REGADR;FAILING REGISTER RHER1
                                                EMT     27

                                                                        ;DCK AND HCRC SHOULD BE SET
                                                                        ;DCK IS SET BECAUSE ECC IS NOT READ
                                        7$:     RTS     R0              ;RETURN TO MAIN TEST
```

EXIT WRT HEADER & DATA ROUTINE

```
                                .SBTTL  EXIT WRT HEADER & DATA ROUTINE

                                ;THIS IS A SUBROUTINE TO LEAVE AT THE MIDDLE OF
                                ;A WRITE HEADER AND DATA COMMAND
                                ;IT TRYS TO GET SECTOR 10, TRACK 0, CYLINDER 0
                                ;BUT COMES OUT AFTER ONE SECTOR
                                ;THE COMMAND OS JSR PC,MIDDLE
                                ;BAI IS SET

034322                          MIDDLE:
034322  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
034324  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
034326  013777  001454  144704          MOV     WRIFOR,@RHCS1   ;WRITE HEADER AND DATA=62
                                                                ;IN RHCS1
034334  012777  177766  144670          MOV     #-10.,@RHWC     ;10 WORDS
034342  012777  001500  144664          MOV     #WRFROM,@RHBA   ;BUS ADDRESS=WRFROM
034350  012777  000010  144666          MOV     #10,@RHDST      ;DESIRED TRACK=0 SECTOR=10
034356  052777  000010  144652          BIS     #BAI,@RHCS2     ;BUS ADDRESS INCREMENT INHIBIT
034364  012777  010000  144656          MOV     #FMT22,@RHOF    ;FORMAT 16 BIT WORDS
034372  005077  144654                  CLR     @RHCA           ;CYLINDER=0
034376  012737  000001  034424          MOV     #1,MID  ;SECTOR IS SET TO 1 SO THAT
                                                                ;WE CAN GET OUT AT THE
                                                                ;MIDDLE OF AN OPERATION
                                                                ;LOOKING FOR SECTOR 10
034404  012777  000001  144646          MOV     #DMD,@RHMR      ;SET DIAGNOSTIC MODE
034412  052777  000001  144620          BIS     #GO,@RHCS1              ;GO TO RHCS1 WITH 62
034420  004137  042756                  JSR     R1,SEARCH
034424  000000                  MID:    .WORD   0               ;SECTOR
034426  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
034430  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
034432  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
                                ;       JSR     R0,MAKECYL
                                ;       .WORD   XC              ;DESIRED VALUE OF CURRENT CYLINDER

034434                          MAKECYL:
034434  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
034436  010037  001414                  MOV     R0,PCJSR        ;PC OF JSR+4
034442  162737  000004  001414          SUB     #4,PCJSR        ;SAVE PC OF JSR
034450  012005                          MOV     (R0)+,R5        ;GETTING READY TO FILL DESIRED CYLINDER
034452  010577  144574                  MOV     R5,@RHCA        ;FILL DESIRED CYLINDER REGISTER
034456  005077  144562                  CLR     @RHDST          ;MAKE SURE DESIRED SECTOR TRACK IS NOT ILLEGAL
034462  013777  001462  144550          MOV     SEECOM,@RHCS1   ;FILL SEEK COMMAND
034470  012777  000001  144562          MOV     #DMD,@RHMR      ;SET DIAGNOSTIC MODE
034476  052777  000001  144534          BIS     #GO,@RHCS1      ;GO TO SEEK
034504  000240                          NOP                     ;ALLOW TIME FOR SEEK TO HANG UP
034506  000240                          NOP                     ;ALLOW TIME FOR SEEK TO HANG UP
034510  000240                          NOP                     ;ALLOW TIME FOR SEEK TO HANG UP
034512  000240                          NOP                     ;ALLOW TIME FOR SEEK TO HANG UP
034514  004737  032054                  JSR     PC,CLDISK       ;GIVE INIT
034520  017737  144552  001126          MOV     @RHCC,$BDDAT    ;TEST DATA
034526  020537  001126                  CMP     R5,$BDDAT       ;COMPARE CURRENT CYLINDER
034532  001406                          BEQ     1$              ;BRANCH IF GOOD
034534  010537  001124                  MOV     R5,$GDDAT       ;GOOD VALUE OF RHCC
034540  013737  001276  031620          MOV     RHCC,REGADR     ;FAILING REGISTER ADDRESS
034546  104030                          EMT     30
                                                                ;REGISTER AFTER A SEEK AND AN INIT
034550                          1$:
034550  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
034552  000200                          RTS     R0
```

ECC GENERATION AND COMPARISON ROUTINE

```
                                .SBTTL  ECC GENERATION AND COMPARISON ROUTINE
                                ;THIS SUBROUTINE GENERATES AND TESTS ECC
                                ;CALL
                                ;       JSR     PC,ECTEST
                        100000  PIE1    =100000
                        040000  PIE2    =40000
                        020000  PIE3    =20000
                        010000  PIE4    =10000
                        004000  PIE5    =4000
                        002000  PIE6    =2000
                        001000  PIE7    =1000
                        000400  PIE8    =400
                        000200  PIE9    =200
                        000100  PIE10   =100
                        000040  PIE11   =40
                        000020  PIE12   =20
                        000010  PIE13   =10
                        000004  PIE14   =4
                        000002  PIE15   =2
                        000001  PIE16   =1
                        100000  PIE17   =100000
                        040000  PIE18   =40000
                        020000  PIE19   =20000
                        010000  PIE20   =10000
                        004000  PIE21   =4000
                        002000  PIE22   =2000
                        001000  PIE23   =1000
                        000400  PIE24   =400
                        000200  PIE25   =200
                        000100  PIE26   =100
                        000040  PIE27   =40
                        000020  PIE28   =20
                        000010  PIE29   =10
                        000004  PIE30   =4
                        000002  PIE31   =2
                        000001  PIE32   =1

034554  000000  ECDATA: 0               ;DATA BIT FOR ECC
                                        ;IF ALL ONES THEN CURRENT BIT IS A ONE
                                        ;IF ZERO THEN CURRENT BIT IS A ZERO

034556  000000  GECC1:  0               ;LOW ORDER ECC WORD TO BE GENERATED HERE
                                        ;=R1

034560  000000  GECC2:  0               ;HIGH ORDER ECC WORD TO BE GENERATED HERE
                                        ;=R2

034562  000000  TSECCG: 0               ;IF =177777 GENERATE AND TEST ECC FOR THIS BIT
                                        ;IF =0 DO NOT GENERATE AND TEST ECC FOR THIS BIT

034564  113713  NCODE:  38859.          ;N-CODE WORD
034566  000000  NCOUNT: 0               ;TEMPORARY N CODE
034570  000000  POSITI: 0               ;POSITION REGISTER
034572  010041  HARDER: 4129.           ;HARD ERROR COUNT
                                        ;TRUE COUNT IS 4128 BUT AS COMPARES ARE
```

ECC GENERATION AND COMPARISON ROUTINE

```
                                                        ;DONE ONE STAGE LATER SO 4129
034574  000000          DATENV: 0                       ;DATA ENVELOPE FOR TYPE OUT
                                                        ;MAX FOR WRITE IS 4096
                                                        ;MAX FOR READ IS 4128
034576  000000          ZCODE:  0                       ;LEADING ZEROS ENVELOPE FOR TYPE OUT
                                                        ;THIS IS SHUT OFF WHEN POSITION COUNTER
                                                        ;IN ENABLED
                                                        ;MAX COUNT IS 38859


034600  000000          HADTMP: 0                       ;TEMPORARY HARD ERROR COUNT
034602  000000          P3:     0
034604  000000          P12:    0
034606  000000          P22:    0
034610  000000          P24:    0


034612                  ECTEST:
034612  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
034614  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
034616  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
034620  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
034622  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
034624  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
034626  013701  034556          MOV     GECC1,R1                ;ECC1 WORD
034632  013702  034560          MOV     GECC2,R2                ;ECC2 WORD
034636  005737  034554          TST     ECDATA          ;IS CURRENT BIT A ONE
034642  001406                  BEQ     2$                      ;BRANCH IF CURRENT DATA D=0

                        ;IF CARRY IS NOT ZERO THEN D=1
                        ;INVERT X32 TO GIVE R0

034644  010103          1$:     MOV     R1,R3
034646  052703  177776          BIS     #^CPIE32,R3
034652  005103                  COM     R3
034654  010300                  MOV     R3,R0
034656  000404                  BR      3$

                        ;IF CARRY IS ZERO THEN D=0, X32 BECOMES R0

034660  010103          2$:     MOV     R1,R3
034662  042703  177776          BIC     #^CPIE32,R3
034666  010300                  MOV     R3,R0

034670  000241          3$:     CLC
034672  006000                  ROR     R0
034674  006000                  ROR     R0
034676  005700                  TST     R0
034700  001462                  BEQ     10$             ;BRANCH IF R0=0

                        ;INVERT X2

034702  010203                  MOV     R2,R3
```

ECC GENERATION AND COMPARISON ROUTINE

```
034704  052703  137777                  BIS     #^CPIE2,R3
034710  005103                          COM     R3
034712  010337  034602                  MOV     R3,P3
034716  006237  034602                  ASR     P3

                                ;INVERT X11


034722  010203                          MOV     R2,R3
034724  052703  177737                  BIS     #^CPIE11,R3
034730  005103                          COM     R3
034732  010337  034604                  MOV     R3,P12
034736  006237  034604                  ASR     P12

                                ;INVERT X21

034742  010103                          MOV     R1,R3
034744  052703  173777                  BIS     #^CPIE21,R3
034750  005103                          COM     R3
034752  010337  034606                  MOV     R3,P22
034756  006237  034606                  ASR     P22

                                ;INVERT X23

034762  010103                          MOV     R1,R3
034764  052703  176777                  BIS     #^CPIE23,R3
034770  005103                          COM     R3
034772  010337  034610                  MOV     R3,P24
034776  006237  034610                  ASR     P24

                                ;NOW THAT R0 FOR POSITION 1
                                ;         P3 FOR POSITION 3
                                ;        P12 FOR POSITION 12
                                ;        P22 FOR POSITION 22
                                ;        P24 FOR POSITION 24
                                ;ARE KNOWN THE ROTATE WILL BE DONE AND
                                ;THESE BITS JAMED IN

035002  006002                          ROR     R2
035004  006001                          ROR     R1
035006  053700  034602                  BIS     P3,R0
035012  053700  034604                  BIS     P12,R0
035016  042702  120020                  BIC     #PIE1!PIE3!PIE12,R2
035022  050002                          BIS     R0,R2

035024  005000                          CLR     R0
035026  053700  034606                  BIS     P22,R0
035032  053700  034610                  BIS     P24,R0
035036  042701  002400                  BIC     #PIE22!PIE24,R1
035042  050001                          BIS     R0,R1
035044  000404                          BR      12$

                                ;THE PROGRAM COMES HERE IF R0=0
                                ;SO AFTER ROTATE R0 GETS PUT INTO POSITION 1

035046  006002                  10$:     ROR     R2
035050  006001                          ROR     R1
```

```
    166 035052  042702  100000              BIC     #PIE1,R2
    167 035056  010137  034556      12$:    MOV     R1,GECC1        ;SAVE ECC1
    168 035062  010237  034560              MOV     R2,GECC2        ;SAVE ECC2
    169 035066  005737  034562              TST     TSECCG  ;IS HARDWARE TO BE CHECKED
    170                                                             ;IF =1777777 TEST HARDWARE
    171                                                             ;IF = 0 DO NOT TEST HARDWARE
    172 035072  001432                      BEQ     14$             ;BRANCH IF HARDWARE NOT TO BE CHECKED
    173
    174
    175                             ;*CHECK HARDWARE
    176 035074  032777  000400  144036      BIT     #SW8,@SWR       ;IS SWITCH 8 SET
    177 035102  001005                      BNE     15$             ;BRANCH IF SW8 IS SET
    178 035104  032777  000100  144026      BIT     #SW6,@SWR       ;IS SWITCH 6 SET
    179 035112  001401                      BEQ     15$             ;BRANCH IF SW6 IS NOT SET
    180 035114  000421                      BR      14$             ;IF SWITCH 8 IS NOT SET AND
    181                                                             ;SWITCH 6 IS SET THEN
    182                                                             ;DO NOT DO COMPARES
    183 035116  010146          15$:        MOV     R1,-(SP)        ;GOOD PATTERN REGISTER
    184 035120  042716  174000              BIC     #174000,(SP)    ;GET ONLY PATTERN BITS
    185 035124  022677  144142              CMP     (SP)+,@RHEC2    ;COMPARE PATTERN REGISTER
    186 035130  001404                      BEQ     13$             ;BRANCH IF GOOD
    187
    188                             ;TO SAVE TIME
    189 035132  004737  031554              JSR     PC,PUTREG       ;SAVE REGISTERS
    190 035136  104035                      EMT     35
    191 035140  000407                      BR      14$             ;BRANCH OUT
    192 035142  023777  034570  144120  13$:CMP     POSITI,@RHEC1   ;COMPARE POISTION REGISTER
    193 035150  001403                      BEQ     14$             ;BRANCH IF GOOD
    194
    195                             ;TO SAVE TIME
    196 035152  004737  031554              JSR     PC,PUTREG       ;SAVE REGISTERS
    197 035156  104035                      EMT     35
    198                                                             ;"DATA ENVLOP" GIVES NUMBER OF CLOCK
    199                                                             ;PULSES FROM BEGINING OF COMMAND
    200                                                             ;THAT IS THE CLOCKS IN THE R/W DATA FIELD ENVELOPE
    201                                                             ;
    202                                                             ;IN A WRITE THERE ARE 10000 OCTAL CLOCKS
    203                                                             ;IN A READ THERE ARE 10040 OCTAL CLOCKS
    204                                                             ;
    205                                                             ;
    206                                                             ;"N-CODE ZEROS" GIVE THE NUMBER OF CLOCKS
    207                                                             ; GIVEN FOR THE LEADING ZEROS FIELD
    208                                                             ;MAX COUNT IS 113713 OCTAL
    209                                                             ;
    210                                                             ;"GOOD POSITION" GIVES NUMBER OF CLOCKS
    211                                                             ;GIVEN AFTER LEADING ZEROS WHICH IS FOR THE DATA
    212                                                             ;FIELD
    213                                                             ;MAX COUNT IS 10040 OR 10041 OCTAL
    214
    215
    216 035160                  14$:
        035160  012605                      MOV     (SP)+,R5        ;;POP STACK INTO R5
        035162  012604                      MOV     (SP)+,R4        ;;POP STACK INTO R4
        035164  012603                      MOV     (SP)+,R3        ;;POP STACK INTO R3
        035166  012602                      MOV     (SP)+,R2        ;;POP STACK INTO R2
        035170  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
        035172  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
035174  000207                          RTS     PC

                                .SBTTL  ECC GENERATION CONTROL ROUTINE

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

035176  000000                  ERPOS:  0                       ;POSITION REG. WHEN CORRECTION IS COMPLETE


035200  010037  001414          ECORR:  MOV     R0,PCJSR        ;SAVE PC OF JSR + 4
035204  162737  000004  001414          SUB     #4,PCJSR        ;SAVE PC OF JSR
035212  012037  035176                  MOV     (R0)+,ERPOS     ;GET POSITION REG. WHEN CORRECTION IS COMPLETE
035216  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
035220  013701  001260                  MOV     RHMR,R1 ;MAIMTENANCE REGISTER
035224  012711  000001                  MOV     #DMD,@R1        ;SET DIAGNOSTIC MODE BIT
035230  005037  034554                  CLR     ECDATA  ;ECC DATA IS ZERO


035234  005737  034570          1$:     TST     POSITI  ;IS  SOFTWARE POSITION NON ZERO
035240  001007                          BNE     2$              ;BRANCH IF N-CODE S COMPLETE
035242  005337  034566                  DEC     NCOUNT  ;DECREMENT N-CODE
035246  001001                          BNE     6$              ;BRANCH IF N-CODE IS NOT COMPLETE
035250  000403                          BR      2$              ;BRANCH  AS N-CODE IS COMPLETE
035252  005237  034576          6$:     INC     ZCODE           ;INCREMENT CLOCKS GIVEN FOR LEADING ZEROS
035256  000420                          BR      3$              ;BRANCH AS N-CODE IS NOT COMPLETE
                                                                ;GO TO GIVE CLOCK AND TEST ECC
035260  005237  034570          2$:     INC     POSITI  ;INCREMENT SOFTWARE POSITION
035264  023737  035176  034570          CMP     ERPOS,POSITI;HAVE ENOUGH CLOCKS BEEN GIVEN TO DETECT ERROR
035272  103012                          BHIS    3$              ;BRANCH IF MORE CLOCKS TO BE GIVEN
035274  023737  034600  034570          CMP     HADTMP,POSITI;HAVE ENOUGH CLOCKS BEEN GIVEN FOR HARD ERROR
                                                                ;THAT IS HAVE 4128 MORE CLOCKS BEEN GIVEN
035302  001415                          BEQ     5$              ;BRANC  IF YES
035304  032711  000400                  BIT     #ZER,@R1        ;CHECK   RO DETECT BIT IN RHMR
035310  001016                          BNE     4$              ;BRANCH IS ZER SET

                                ;TO SAVE TIME
035312  004737  031554                  JSR     PC,PUTREG       ;SAVE REGISTERS
035316  104034                          EMT     34
                                                                ;WHEN 21 BITS IN ECC 32 BIT REGISTER IS  0


035320  052711  000002          3$:     BIS     #MCLK,@R1       ;SET CLOCK
035324  042711  000002                  BIC     #MCLK,@R1       ;CLEAR CLOCK
035330  004737  034612                  JSR     PC,ECTEST       ;GO TO GENERATE AND TEST ECC
035334  000737                          BR      1$              ;CONTINUE

                                ;THIS EXTRA CLOCK IS TO BRING ECH HIGH
                                ;AFTER THIS CLOCK POSITION REGISTER MAY BE 10040 OR 10041 OCTAL

035336  052711  000002          5$:     BIS     #MCLK,@R1       ;SET CLOCK
035342  042711  000002                  BIC     #MCLK,@R1       ;CLEAR CLOCK
```

ECC GENERATION CONTROL ROUTINE

```
    274
    275 035346                          4$:
        035346  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
    276 035350  000200                          RTS     R0
```

```
                                        .SBTTL  SOFTWARE DISK DATA ECC GEN. ROUTINE
                                        ;
                                        ;THIS SUBROUTINE GENERATES THE ECC FOR WHAT IS ON DISK AND INSERTS THEM
                                        ;ON LOCATIONS 'DISK+1000" AND 'DISK+1002"
                                        ;
035352                                  FILLEC:
035352  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
035354  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
035356  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
035360  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
035362  010446                                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
035364  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
035366  005037  034570                         CLR     POSITI  ;CLEAR POSITION
035372  005037  034556                         CLR     GECC1           ;CLEAR GECC1
035376  005037  034560                         CLR     GECC2           ;CLEAR
035402  012701  040626                         MOV     #DISK,R1        ;POINTER TO DATA FOR ECC GENERATION
035406  012702  000400                         MOV     #256.,R2        ;COUNTER FOR NUMBER OF DATA WORDS
035412  012703  000020                 9$:     MOV     #16.,R3         ;COUNTER FOR NUMBER OF BITS PER WORD
035416  012104                         10$:    MOV     (R1)+,R4        ;DATA IN R4
035420  006004                                 ROR     R4              ;GET ONE DATA BIT IN CARRY
035422  103004                                 BCC     11$             ;BRANCH IF DATA BIT IS ZERO
035424  012737  177777  034554                 MOV     #-1,ECDATA      ;ECC DATA BIT IS A ONE
035432  000402                                 BR      12$             ;BRANCH TO GENERATE ECC
035434  005037  034554                 11$:    CLR     ECDATA  ;ECC DATA BIT IS A ZERO
035440  004737  034612                 12$:    JSR     PC,ECTEST       ;GO TO GENERATE ECC
035444  005303                                 DEC     R3              ;DECREMENT BIT COUNT
035446  001364                                 BNE     10$             ;BRANCH IF 16 BITS NOT DONE
035450  005302                                 DEC     R2              ;DECREMENT WORD COUNT
035452  001357                                 BNE     9$              ;BRANCH IF 256 WORDS NOT DONE
035454  013737  034556  041626                 MOV     GECC1,DISK+<256.*2>;INSERT ECC1 ON DISK
035462  013737  034560  041630                 MOV     GECC2,DISK+<257.*2>;INSERT ECC2 ON DISK
035470  012605                                 MOV     (SP)+,R5        ;;POP STACK INTO R5
035472  012604                                 MOV     (SP)+,R4        ;;POP STACK INTO R4
035474  012603                                 MOV     (SP)+,R3        ;;POP STACK INTO R3
035476  012602                                 MOV     (SP)+,R2        ;;POP STACK INTO R2
035500  012601                                 MOV     (SP)+,R1        ;;POP STACK INTO R1
035502  012600                                 MOV     (SP)+,R0        ;;POP STACK INTO R0
035504  000207                                 RTS     PC
```

RH BASE ADDRESS CHANGE ROUTINE

```
                                        .SBTTL  RH BASE ADDRESS CHANGE ROUTINE
                                        ;THIS ROUTINE WILL ALLOW THE CHANGE OF THE BASE
                                        ;ADDRESS FROM 176700 TO ANY TYPED VALUE
 035506  012706  001100         BASECH: MOV     #STACK,SP       ;RESET STACK
 035512  104401  035520                 TYPE    ,65$            ;;TYPE ASCIZ STRING
 035516  000415                         BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:   .ASCIZ  <CRLF>/PRESENT BASE ADDRESS IS /
                                64$:
 035552  013746  001240                 MOV     RHCS1,-(SP)     ;GET READY TO TYPE OLD BASE
 035556  104402                         TYPOC
 035560  104401  035566                 TYPE    ,67$            ;;TYPE ASCIZ STRING
 035564  000425                         BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:   .ASCIZ  <CRLF>/TYPE NEW BASE ADDRESS FOLLOWED BY 'CR' /
                                66$:
 035640  004737  044620                 JSR     PC,$TKINT       ;INITIALIZE THE TTY KEYBOARD
 035644  104412                         RDOCT
 035646  012700  001230                 MOV     #RHDB,R0        ;GET STARTING ADDRESS OF REGISTERS
 035652  012701  000026                 MOV     #22.,R1         ;NUMBER OF REGISTERS
 035656  012737  036436  000004         MOV     #ADTIMO,ERRVEC  ;SET UP TO CHECK NEW ADDRESS
 035664  021637  001240                 CMP     (SP),RHCS1      ;NEW CSR?
 035670  001407                         BEQ     1$              ;NO-SKIP NEXT
 035672  005776  000000                 TST     @(SP)           ;ACCESS THE NEW ADDRESS
 035676  163716  001240                 SUB     RHCS1,(SP)      ;GET THE ADDRESS OFFSET
 035702  061620                 2$:     ADD     (SP),(R0)+      ;AND PLUG IT IN
 035704  005301                         DEC     R1              ;DONE ALL OF THEM YET?
 035706  001375                         BNE     2$              ;NOT YET, SO DO MORE
                                1$:
 035710  104401  035716                 TYPE    ,69$            ;;TYPE ASCIZ STRING
 035714  000416                         BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:   .ASCIZ  <CRLF>/PRESENT VECTOR ADDRESS IS /
                                68$:
 035752  013746  001226                 MOV     RPVEC,-(SP)     ;GET READY TO TYPE OLD VECTOR ADDRESS
 035756  104402                         TYPOC
 035760  104401  035766                 TYPE    ,71$            ;;TYPE ASCIZ STRING
 035764  000437                         BR      70$             ;;GET OVER THE ASCIZ
                                ;;71$:   .ASCIZ  <CRLF>/TYPE NEW VECTOR ADDRESS OR RETYPE OLD ONE FOLLOWED BY 'CR' /
                                70$:
 036064  104412                         RDOCT
 036066  012637  001226                 MOV     (SP)+,RPVEC     ;SETUP VECTOR ADDRESS
 036072  104401  036100                 TYPE    ,73$            ;;TYPE ASCIZ STRING
 036076  000413                         BR      72$             ;;GET OVER THE ASCIZ
                                ;;73$:   .ASCIZ  <CRLF>/NEW BASE ADDRESS IS /
                                72$:
 036126  013746  001240                 MOV     RHCS1,-(SP)
 036132  104402                         TYPOC
 036134  104401  036142                 TYPE    ,75$            ;;TYPE ASCIZ STRING
 036140  000414                         BR      74$             ;;GET OVER THE ASCIZ
                                ;;75$:   .ASCIZ  <CRLF>/NEW VECTOR ADDRESS IS /
                                74$:
 036172  013746  001226                 MOV     RPVEC,-(SP)
 036176  104402                         TYPOC
 036200  104401  036206                 TYPE    ,77$            ;;TYPE ASCIZ STRING
 036204  000416                         BR      76$             ;;GET OVER THE ASCIZ
                                ;;77$:   .ASCIZ  <CRLF>/UNTIL PROGRAM IS RELOADED./
                                76$:
 036242
```

```
     36 036242  104401  036250                TYPE    ,79$            ;;TYPE ASCIZ STRING
        036246  000402                        BR      78$             ;;GET OVER THE ASCIZ
                                       ;;79$:  .ASCIZ  <CRLF>/ /
        036254                         78$:
     37 036254  104401  036262                TYPE    ,81$            ;;TYPE ASCIZ STRING
        036260  000424                        BR      80$             ;;GET OVER THE ASCIZ
                                       ;;81$:  .ASCIZ  <CRLF>/UNLESS HALTED AND MANUALLY RESTARTED,/
        036332                         80$:
     38 036332  104401  036340                TYPE    ,83$            ;;TYPE ASCIZ STRING
        036336  000425                        BR      82$             ;;GET OVER THE ASCIZ
                                       ;;83$:  .ASCIZ  <CRLF>/PROGRAM WILL AUTOMATICALLY RESTART FROM /
        036412                         82$:
     39 036412  012746  000200                MOV     #200,-(SP)
     40 036416  104402                        TYPOC
     41 036420  104401  036426                TYPE    ,85$            ;;TYPE ASCIZ STRING
        036424  000402                        BR      84$             ;;GET OVER THE ASCIZ
                                       ;;85$:  .ASCIZ  <CRLF>/ /
        036432                         84$:
     42 036432  000137  004620                JMP     BEGIN           ;OK, NOW START OVER WITH NEW ADDRESS
     43
     44 036436                         ADTIMO:
        036436  104401  036444                TYPE    ,65$            ;;TYPE ASCIZ STRING
        036442  000423                        BR      64$             ;;GET OVER THE ASCIZ
                                       ;;65$:  .ASCIZ  <CRLF><377>/SPECIFIED ADDRESS DID NOT RESPOND. /
        036512                         64$:
     45 036512  022626                        CMP     (SP)+,(SP)+     ;RESTORE THE STACK
     46 036514  000137  035506                JMP     BASECH          ;AND DO IT AGAIN.
     47
     48                                ;*THIS IS A LITTLE ROUTINE THAT TESTS NED BIT 11 IN PHCS2
     49                                ;*THIS LOOPS HERE FOR EVER
     50                                ;*TO BE USED ONLY IF DRIVES PRESENT LOOKING AT NED DOES NOT AGREE
     51                                ;*WITH WHAT IS REALY THERE
     52 036520  000000                 ERUNIT: 0                       ;UNIT UNDER MANUAL TEST
     53 036522  004737  032054         ERSTART:JSR     PC,CLDISK       ;SET GENERAL REG.
     54 036526  013712  036520                 MOV     ERUNIT,@R2      ;SELECT UNIT
     55 036532  005714                 1$:     TST     @R4             ;TEST RHER1
     56 036534  032712  010000                 BIT     #NED,@R2        ;TEST NED
     57 036540  001401                         BEQ     2$              ;BRANCH IF GOOD
     58 036542  000773                         BR      1$              ;NED NOT SET
     59 036544  000772                 2$:     BR      1$              ;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,COMHD
                        ;
                        ;
                        ;
                        ;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,COMHD
                        ;
                        ;
                        ;
                        ;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,COMHD
                        ;
```

```
                                ;;*****************************************************************
                                ;*WRITE DATA COMMAND
                                ;*OR READ COMMAND, I.E DATA ONLY, OR HEADER AND DATA
                                ;;*****************************************************************

                                ;*THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES

                                ;*IT ISSUES DIAGNOSTIC MODE, AND EXTRA DIAGNOSTIC INDEX, AND THE
                                ;*'GO' BIT

                                ;*IT THEN CALLS THREE OTHER SUBROUTINES, WHICH IN TURN CALL
                                ;*OTHER SUBROUTINES.  THE SUBROUTINES CALLED HERE ARE:

                                ;*SEARCH                 ;ISSUES SECTOR CLOCKS TO SET SECTOR FOUND FLOP
                                ;*RDHEAD                 ;READS THE SECTOR HEADER
                                ;*WRDATA                 ;WRITES THE SECTOR DATA (WRITE OPERATION)
                                ;*REDATA                 ;READS THE SECTOR DATA (READ OPERATION)

036546  000000                  RUNCTR: .WORD   0

036550  011637  001414          COMHD:  MOV     (SP),PCJSR      ;SAVE PC OF JSR + 4
036554  162737  000004  001414          SUB     #4,PCJSR        ;SAVE PC OF JSR
036562  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
036564  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
036566  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
036570  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
036572  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
036574  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK

036576  012777  000001  142454        MOV     #DMD,@RHMR      ;SET DIAGNOSTIC MODE
036604  052777  000004  142446        BIS     #MINX,@RHMR     ;SET DIAGNOSTIC INDEX
036612  042777  000004  142440        BIC     #MINX,@RHMR     ;CLEAR DIAGNOSTIC INDEX
036620  052777  000001  142412        BIS     #GO,@RHCS1      ;ISSUE 'GO' BIT & STALL 'TILL 'RUN'
                                                              ;FUNCTION CODE WAS ISSUED BY THE TEST
036626  012737  000113  036546  RUNWAT: MOV     #75.,RUNCTR     ;LOAD STALL COUNT = APPROX. 450US FOR 11/50 CPU
036634  005337  036546          1$:     DEC     RUNCTR  ;COUNT DOWN ONE
036640  001375                        BNE     1$              ;CONTINUE UNTIL = 0

036642  013746  036712                MOV     SECOTR,-(SP)    ;GET DESIRED SECTOR/TRACK
036646  042716  177740                BIC     #177740,(SP)    ;MAKE ONLY SECTOR
036652  012637  036662                MOV     (SP)+,TRK       ;SAVE SECTOR
036656  004137  042756                JSR     R1,SEARCH       ;ISSUE SECTOR CLOCKS <---------------->

036662  000000                  TRK:    .WORD   0
036664  012701  000240                MOV     #+NOP,R1        ;GOING TO MOVE NOPS
036670  010137  036722                MOV     R1,SSYN ;NOP INTO SSYN
036674  010137  036724                MOV     R1,HEDGAP       ;NOP INTO HEDGAP
036700  010137  036726                MOV     R1,HEDSYN       ;NOP INTO HEDSYN
036704  004137  037032                JSR     R1,RDHEAD       ;READ THE HEADER <------------->

036710  000000                  CYL:    .WORD   0               ;CYLINDER ADDRESS
036712  000000                  SECOTR: .WORD   0               ;SECTOR/TRACK ADDRESS
036714  000000                  KEY1:   .WORD   0               ;KEY1 WORD
036716  000000                  KEY2:   .WORD   0               ;KEY2 WORD
036720  000000                  X:      .WORD   0               ;X=1 WRITE COMMAND
                                                                ;X=0 READ COMMAND
```

```
                                ;DUMMY ERROR CALL LOCATIONS FOR THE READ HEADER OPERATION

036722  000240                  SSYN:   NOP                     ;IF "ERROR 2" INSERTED BY RDHEAD
                                                                ;SUBROUTINE, THEN THE FIRST SYNC
                                                                ;IS NOT DETECTED.  NO BAD DATA
                                                                ;IS GIVEN BECAUSE SYNC=144000
                                                                ;CANNOT BE READ.  WORD NUMBER
                                                                ;IS "1" BECAUSE THIS IS THE FIRST
                                                                ;WORD TESTED.

036724  000240                  HEDGAP: NOP                     ;IF "ERROR 3" INSERTED BY
                                                                ;RDHEAD SUBROUTINE, THEN THE
                                                                ;HEADER GAP 0'S WERE NOT
                                                                ;WRITTEN RIGHT.

                                                                ;IF "WORD NO" CONTAINS, SAY
                                                                ;3(8), THEN IT IS THE THIRD
                                                                ;WORD OF A 5 WORD HEADER
                                                                ;GAP THAT IS WRONG.

                                                                ;"BAD DATA" CONTAINS WHAT IS
                                                                ;GOING ON THE DISK.

036726  000240                  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 0,
                                                                ;LEFT BYTE IS SYNC.

                                                                ;"BAD DATA" HAS WHAT IS GOING
                                                                ;ON DISK.

036730  005737  001406                  TST     ERFLG$  ;WERE ANY ERRORS DETECTED ?
036734  001017                          BNE     OUT             ;IF YES, EXIT ---------------------->

036736  005737  036720                  TST     X               ;IS IT A DATA WRITE OPERATION ?
036742  001410                          BEQ     DAREAD          ;NO...THEN DO A DATA READ
036744  005737  037024                  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
036750  001011                          BNE     OUT             ;EXIT IF SET ---------------------->

036752  004137  040276                  JSR     R1,WRDATA       ;WRITE DATA <---------------------->
036756  000000          NOWORD: .WORD   0               ;NO OF WORDS TO BE WRITTEN
036760  000000          Y:      .WORD   0               ;
036762  000404                  BR      OUT             ;EXIT ---------------------------->

036764  004137  043232          DAREAD: JSR     R1,REDATA       ;READ DATA <---------------------->
036770  000000          DAWORD: .WORD   0               ;NO OF WORDS TO BE READ
036772  000000                  .WORD   0
```

```
    110
    111 036774                          OUT:
         036774  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
         036776  012604                         MOV     (SP)+,R4        ;;POP STACK INTO R4
         037000  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
         037002  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
         037004  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
         037006  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
    112 037010  000207                         RTS     PC              ;EXIT
```

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

 037012  014400                 RSYNC:  14400
 037014  000000                 RCYL:   0
 037016  000000                 RSETR:  0
 037020  000000                 RKEY1:  0
 037022  000000                 RKEY2:  0

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

                                ;*THERE ARE 256 WORDS OF DATA
                                ;*THERE ARE 2 WORDS FOR ECC GENERATED BY DCL
                                ;*15 WORDS OF 0'S FOR DATA GAP AND TOLERANCE GAP
                                ;;*****************************************************************
```

```
                                ;;*****************************************************************
                                ;*READ DISK HEADER
                                ;;*****************************************************************

037024  000000                  NOSYNC: 0                       ;FORCED HEADER ERROR = -1
                                                                ;NORMAL = 0
037026  000000                  TY:     0                       ;ERROR TYPE NO.
037030  000000                  ERWORD: 0                       ;ERROR WORD NO.

037032  012137  037014          RDHEAD: MOV     (R1)+,RCYL      ;STORE CYLINDER ADDRESS
037036  012137  037016                  MOV     (R1)+,RSETR     ;STORE SECTOR AND TRACK ADDRESS
037042  012137  037020                  MOV     (R1)+,RKEY1     ;STORE KEY1
037046  012137  037022                  MOV     (R1)+,RKEY2     ;STORE KEY2
037052  012137  037622                  MOV     (R1)+,COMPA     ;STORE COMPARE OR NOT
037056  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK

037060  013700  001260                  MOV     RHMR,R0 ;R0 CONTAINS MAINTANENCE REG.
037064  012705  000002                  MOV     #2,R5           ;R5 IS A COUNTER FOR WORDS
037070  012710  000001                  MOV     #DMD,@R0        ;SET DIAG. MODE
037074  052710  000010                  BIS     #MSTCK,@R0      ;SET SECTOR CLOCK FOR FIRST WORD
037100  052710  000002                  BIS     #MCLK,@R0       ;SET MAINT.CLOCK FOR FIRST WORD
037104  042710  000012                  BIC     #MSTCK!MCLK,@R0 ;RESET THEM

037110  000404                          BR      2$              ;DON'T GIVE SECTOR CLOCK FIRST TIME
037112  012710  000013          1$:     MOV     #MSTCK!MCLK!DMD,@R0;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
037116  042710  000012                  BIC     #MSTCK!MCLK,@R0 ;RESET SECTOR & MAINT.CLOCK

037122  012702  000007          2$:     MOV     #7,R2           ;LOAD BYTE COUNTER
037126  052710  000002          3$:     BIS     #MCLK,@R0       ;SET MAINT. CLOCK
037132  042710  000002                  BIC     #MCLK,@R0       ;RESET IT
037136  005302                          DEC     R2              ;BYTE COUNTER
037140  001372                          BNE     3$              ;CONTINUE IF BYTE NOT COMPLETE
037142  005305                          DEC     R5              ;WORD COUNTER
037144  001362                          BNE     1$              ;CONTINUE IF WORD NOT COMPLETE

037146  012702  000022                  MOV     #18.,R2         ;LOAD NO OF WORDS OF ZEROS
037152  005037  037620          4$:     CLR     WORD            ;DO 0'S
037156  004737  037624                  JSR     PC,READ ;READ 0'S <------------------->
037162  005302                          DEC     R2              ;COUNT DOWN WORDS
037164  001372                          BNE     4$              ;CONTINUE

037166  013737  037012  037620          MOV     RSYNC,WORD      ;SYNC. WORD
037174  004737  037624                  JSR     PC,READ ;READ IT <---------------------->
037200  032710  001000                  BIT     #DTSY,@R0       ;SYNC. BYTE DETECTED?
037204  001012                          BNE     5$              ;CONTINUE IF SYNC DETECTED
037206  012737  000001  037030          MOV     #1,ERWORD       ;ERROR WORD NO
037214  013737  037012  001124          MOV     RSYNC,$GDDAT;SYNC WORD
037222  012737  104002  036722          MOV     #104002,SSYN    ;INSERT 'ERROR 2'' IN SSYN
037230  000571                          BR      13$             ;BRANCH OUT ---------------------->

037232  013737  037014  037620  5$:     MOV     RCYL,WORD       ;SETUP CYLINDER
037240  004737  037624                  JSR     PC,READ ;READ IT <---------------------->
037244  013737  037016  037620          MOV     RSETR,WORD      ;SETUP SECTOR/TRACK
037252  004737  037624                  JSR     PC,READ ;READ THEM <------------------->
037256  013737  037020  037620          MOV     RKEY1,WORD      ;SETUP KEY1
037264  004737  037624                  JSR     PC,READ ;READ IT <---------------------->
037270  013737  037022  037620          MOV     RKEY2,WORD      ;SETUP KEY2
```

```
     58 037276  004737  037624                  JSR     PC,READ ;READ IT <---------------------->
     59 037302  013737  040610  037620          MOV     WCRC,WORD       ;SETUP CRC
     60 037310  004737  037624                  JSR     PC,READ ;READ IT <---------------------->
     61
     62 037314  005737  001426                  TST     TESDTE  ;IS THIS A DRIVE TIMING ERROR ?
     63 037320  001135                          BNE     13$             ;BRANCH OUT IF YES ------------>
     64 037322  005737  037622                  TST     COMPA           ;IS THIS A READ OR WRITE COMMAND ?
     65 037326  001472                          BEQ     11$             ;DO READ IF = 0
     66
     67                                 ;*CONTINUE WITH DIAGNOSTIC WRITE COMMAND
     68
     69 037330  012705  040612                  MOV     #HEGAP,R5       ;POINTER FOR HEADER GAP
     70 037334  012702  000005                  MOV     #5,R2           ;NO OF WORDS OF ZEROS
     71 037340  012737  000006  037030  6$:     MOV     #6,ERWORD       ;ERROR WORD NO SET
     72 037346  004737  040056                  JSR     PC,WRITE        ;FOR HEADER GAP
     73 037352  005737  040054                  TST     WWORD           ;TEST WRITTEN WORD
     74 037356  001413                          BEQ     7$              ;CONTINUE IF GOOD, THAT IS = 0
     75 037360  160237  037030                  SUB     R2,ERWORD       ;WORD NO IN ERROR
     76 037364  005037  001124                  CLR     $GDDAT  ;GOOD WORD SHOULD BE 0
     77 037370  013737  040054  001126          MOV     WWORD,$BDDAT    ;BAD DATA
     78 037376  012737  104003  036724          MOV     #104003,HEDGAP;''ERROR 2'' GOES IN HEDGAP
     79 037404  000503                          BR      13$             ;BRANCH OUT ------------------->
     80
     81 037406  013725  040054          7$:     MOV     WWORD,(R5)+     ;SAVE HEADER GAP
     82 037412  005302                          DEC     R2
     83 037414  001351                          BNE     6$
     84 037416  004737  040056                  JSR     PC,WRITE        ;WRITE HEADER (DATA) GAP SYNC
     85 037422  023737  037012  040054          CMP     RSYNC,WWORD
     86 037430  001426                          BEQ     10$
     87 037432  005737  037024                  TST     NOSYNC  ;IS THIS FORCED HEADER ERROR COMMAND ?
     88                                                         ;IF YES  NOSYNC=-1 THEN WRITE OR READ
     89                                                         ;IS SHUT OFF SO BRANCH OUT
     90                                                         ;IF NO NOSYNC=0 THEN CONTINUE
     91 037436  001406                          BEQ     14$             ;PRINT IT IF TRUE ERROR
     92
     93 037440  005737  040054                  TST     WWORD           ;IS IT = 0 ?
     94 037444  001420                          BEQ     10$             ;CONTINUE IF GOOD
     95 037446  005037  001124                  CLR     $GDDAT  ;IT SHOULD BE ZERO
     96 037452  000403                          BR      15$             ;BRANCH TO TYPE ERROR
     97 037454  013737  037012  001124  14$:     MOV     RSYNC,$GDDAT;GOOD DATA
     98 037462  013737  040054  001126  15$:     MOV     WWORD,$BDDAT;BAD DATA
     99 037470  012737  000006  037030          MOV     #6,ERWORD
    100 037476  012737  104003  036726          MOV     #104003,HEDSYN
    101 037504  000443                          BR      13$             ;BRANCH OUT ---------------------->
    102
    103 037506  013725  040054          10$:    MOV     WWORD,(R5)+     ;SAVE DATA SYNC.
    104 037512  000440                          BR      13$             ;EXIT ---------------------------->
    105
    106                                 ;*READ COMMAND START FROM HERE
    107
    108 037514  012702  000005          11$:    MOV     #5,R2
    109 037520  005037  037620          12$:    CLR     WORD
    110 037524  004737  037624                  JSR     PC,READ ;READ HEADER GAP <--------------->
    111 037530  005302                          DEC     R2              ;ARE 5 HEADER GAP ZEROS COMPLETE ?
    112 037532  001372                          BNE     12$             ;IF NOT CONTINUE
    113 037534  013737  037012  037620          MOV     RSYNC,WORD      ;SYNC WORD
    114 037542  004737  037624                  JSR     PC,READ ;READ HEADER (DATA) SYNC)
```

```
    115 037546  005737  037024                  TST     NOSYNC  ;FORCED SYNC ERROR ?
    116 037552  001404                           BEQ     16$             ;IF NOT ERROR COMMAND CONTINUE
    117 037554  032710  001000                   BIT     #DTSY,@R0       ;SYNC. DETECTED
    118 037560  001415                           BEQ     13$             ;IF ZERO BRANCH OUT ------------>
    119 037562  000403                           BR      17$             ;IF NOT ZERO BRANCH TO ERROR
    120
    121 037564  032710  001000           16$:    BIT     #DTSY,@R0       ;SYNC. DETECTED ?
    122 037570  001011                           BNE     13$             ;EXIT IF YES ------------------->
    123 037572  012737  000006  037030   17$:    MOV     #6,ERWORD       ;ERROR WORD NO.
    124 037600  013737  037012  001124           MOV     RSYNC,$GDDAT;SYNC WORD
    125 037606  012737  104002  036726           MOV     #104002,HEDSYN;MOVE 'ERROR 2''
    126 037614                           13$:
        037614  012601                           MOV     (SP)+,R1        ;;POP STACK INTO R1
    127 037616  000201                           RTS     R1              ;EXIT ------------------------->
```

```
                                ;;*****************************************************************
                                ;*READ ONE WORD IN 'WORD'
                                ;;*****************************************************************

037620  000000                WORD:   0
037622  000000                COMPA:  0

037624                          READ:
037624  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
037626  012705  000002                MOV     #2,R5           ;WORD COUNTER
037632  012710  000001                MOV     #DMD,@R0        ;SET DIAG. MODE
037636  006037  037620                ROR     WORD            ;CHECKING IF THERE IS A ONE
037642  103002                        BCC     1$              ;IF NO ONE BRANCH
037644  052710  000020                BIS     #MRD,@R0        ;SET BIT 4 IF DATA HAS ONE
037650  012702  000007          1$:   MOV     #7,R2           ;BYTE COUNTER
037654  052710  0C0012                BIS     #MSTCK!MCLK,@R0 ;SET CLOCK,DATA IF ANY, SECTOR
037660  005737  034562                TST     TSECCG  ;IS THIS BIT TO GENERATE AND TEST ECC ?
037664  001411                        BEQ     6$              ;BRANCH IF NO
037666  032710  000020                BIT     #MRD,@R0        ;IS DATA BIT A ONE ?
037672  001404                        BEQ     5$              ;BRANCH IF DATA BIT IS 0
037674  012737  177777  034554        MOV     #-1,ECDATA      ;ECC DATA BIT IS A ONE
037702  000402                        BR      6$              ;BRANCH
037704  005037  034554          5$:   CLR     ECDATA  ;ECC DATA BIT IS A 0
037710  012746  000001          6$:   MOV     #DMD,-(SP)      ;KEEP ONLY DIAG. MODE
037714  006037  037620                ROR     WORD            ;CHECKING IF THERE IS A ONE
037720  103002                        BCC     2$              ;IF NO ONE BRANCH
037722  012716  000021                MOV     #MRD!DMD,(SP)   ;KEEP DATA AND DIAG. MODE
037726  012610                  2$:   MOV     (SP)+,@R0       ;PUT IN DATA,RESET CLOCK, SECTOR
037730  005737  034562                TST     TSECCG  ;IS ECC TO BE GENERATED FOR THIS BIT
037734  001404                        BEQ     3$              ;BRANCH IF NO
037736  005237  034574                INC     DATENV  ;NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
037742  004737  034612                JSR     PC,ECTEST       ;GO TO GENERATE AND TEST ECC
037746  052710  000002          3$:   BIS     #MCLK,@R0       ;SET CLOCK
037752  005737  034562                TST     TSECCG  ;IS THIS BIT TO GENERATE ECC
037756  C01411                        BEQ     8$              ;BRANCH IF NO
037760  032710  000020                BIT     #MRD,@R0        ;IS DATA BIT A ONE
037764  001404                        BEQ     7$              ;BRANCH IF DATA BIT IS = 0
037766  012737  177777  034554        MOV     #-1,ECDATA      ;ECC DATA BIT IS A ONE
037774  000402                        BR      8$              ;BRANCH
037776  005037  034554          7$:   CLR     ECDATA  ;ECC DATA BIT IS = 0
040002  012746  000001          8$:   MOV     #DMD,-(SP)      ;KEEP DIAG. MODE
040006  006037  037620                ROR     WORD            ;CHECKING IF THERE IS A ONE
040012  103002                        BCC     4$              ;BRANCH IF NO ONE
040014  012716  000021                MOV     #MRD!DMD,(SP)   ;KEEP DIAG. MODE AND DATA
040020  012610                  4$:   MOV     (SP)+,@R0       ;SET DATA, DIAG. MODE, CLEAR CLOCK
040022  005737  034562                TST     TSECCG  ;IS THIS BIT TO GENERATE ECC
040026  001404                        BEQ     9$              ;BRANCH IF NO
040030  005237  034574                INC     DATENV  ;NUMBER OF CLOCKS FOR DATA ENVELOPE
040034  004737  034612                JSR     PC,ECTEST       ;GO TO GENERATE AND TEST ECC

040040  005302                  9$:   DEC     R2              ;BYTE COUNTER
040042  001341                        BNE     3$              ;CONTINUE IF ONE BYTE NOT COMPLETE
040044  005305                        DEC     R5              ;WORD COUNTER
040046  001300                        BNE     1$              ;CONTINUE IF ONE WORD NOT COMPLETE
040050  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
040052  000207                        RTS     PC              ;EXIT ---------------------------->
```

```
                                ;;*****************************************************************
                                ;*WRITE ONE WORD WHICH COMES BACK IN 'WWORD'
                                ;;*****************************************************************

040054  000000                  WWORD:  0

040056                          WRITE:
040056  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
040060  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
040062  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
040064  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
040066  012705  000002                  MOV     #2,R5           ;WORD COUNTER
040072  012710  000001                  MOV     #1,@R0          ;SET DIAG. MODE
040076  012702  000007          1$:     MOV     #7,R2           ;BYTE COUNTER
040102  012710  000013                  MOV     #MSTCK!MCLK!DMD,@R0;SET SECTOR & MANT. CLOCKS
040106  032710  000040                  BIT     #MWR,@R0        ;CHECK WRITEBIT IN MAINT. REG.
040112  001406                          BEQ     2$              ;BRANCH IF ZERO
040114  012737  177777  034554          MOV     #-1,ECDATA      ;ECC DATA BIT IS A ONE
040122  000261                          SEC                     ;SET CARRY
040124  006003                          ROR     R3              ;MOVE 1 FORWARD
040126  000404                          BR      3$
040130  005037  034554          2$:     CLR     ECDATA  ;ECC DATA BIT IS = 0
040134  000241                          CLC                     ;CLEAR CARRY
040136  006003                          ROR     R3              ;MOVE 0 FOR WWORD
040140  012710  000001          3$:     MOV     #DMD,@R0        ;CLEAR SECTOR AND CLOCK
040144  005737  034562                  TST     TSECCG  ;IS THIS BIT TO GENERATE ECC ?
040150  001404                          BEQ     4$              ;BRANCH IF NO
040152  005237  034574                  INC     DATENV  ;NUMBER OF CLOCKS FOR DATA ENVELOPE
040156  004737  034612                  JSR     PC,ECTEST       ;GO TO GENERATE AND TEST ECC <-------->

040162  052710  000002          4$:     BIS     #MCLK,@R0       ;SET CLOCK
040166  032710  000040                  BIT     #MWR,@R0        ;CHECK WRITE BIT IN MAINT. REG.
040172  001406                          BEQ     5$              ;BRANCH IF ZERO
040174  012737  177777  034554          MOV     #-1,ECDATA      ;ECC DATA BIT IS A ONE
040202  000261                          SEC                     ;SET CARRY
040204  006003                          ROR     R3              ;MOVE 1 FOR WWORD
040206  000404                          BR      6$
040210  005037  034554          5$:     CLR     ECDATA  ;ECC DATA BIT IS ZERO
040214  000241                          CLC                     ;CLEAR CARRY
040216  006003                          ROR     R3              ;MOVE 0 FOR WWORD
040220  012710  000001          6$:     MOV     #DMD,@R0        ;CLEAR CLOCK
040224  005737  034562                  TST     TSECCG  ;IS THIS BIT TO GENERATE ECC ?
040230  001404                          BEQ     7$              ;BRANCH IF NO
040232  005237  034574                  INC     DATENV  ;NUMBER OF CLOCKS FOR DATA ENVELOPE
040236  004737  034612                  JSR     PC,ECTEST       ;GO TO GENERATE AND TEST ECC <-------->

040242  005302                  7$:     DEC     R2              ;COUNT FOR BYTE END
040244  001346                          BNE     4$              ;IF NOT BYTE END CONTINUE
040246  005305                          DEC     R5              ;COUNT FOR WORD END
040250  001312                          BNE     1$              ;IF NOT WORD END CONTINUE
040252  010337  040054                  MOV     R3,WWORD        ;STORE THE WORD
040256  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
040260  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
040262  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
040264  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
040266  000207                          RTS     PC              ;EXIT ------------------------------>
```

```
                                ;;*****************************************************************
                                ;*WRITE DATA HOUSEKEEPING ROUTINE
                                ;;*****************************************************************

040270  000000                  COUNTD: 0
040272  000400                  FORMAT: 256.
040274  000000                  ZWORDS: 0
040276                          WRDATA:
040276  011137  040270                 MOV     (R1),COUNTD     ;STORE NO. OF WORDS TO BE WRITTEN
040302  012102                   MOV     (R1)+,R2        ;SAME IN R2
040304  012137  037622           MOV     (R1)+,COMPA     ;COMPARE OR NOT
040310  010046                   MOV     R0,-(SP)        ;;PUSH R0 ON STACK
040312  010146                   MOV     R1,-(SP)        ;;PUSH R1 ON STACK
040314  010246                   MOV     R2,-(SP)        ;;PUSH R2 ON STACK
040316  010346                   MOV     R3,-(SP)        ;;PUSH R3 ON STACK
040320  010446                   MOV     R4,-(SP)        ;;PUSH R4 ON STACK
040322  012701  000016           MOV     #14.,R1         ;NO. OF TOLERANCE GAP WORDS
040326  012703  041634           MOV     #TOLGAP,R3      ;START OF TOLERANCE GAP TABLE
040332  012723  177777   1$:     MOV     #-1,(R3)+       ;MAKE IT 177777
040336  005301                   DEC     R1              ;IS 14 COMPLETED ?
040340  001374                   BNE     1$              ;IF NOT, CONTINUE

040342  013700  001260           MOV     RHMR,R0 ;R0 CONTAINS MAINTANENCE REG.
040346  013746  040272           MOV     FORMAT,-(SP)
040352  163716  040270           SUB     COUNTD,(SP)
040356  011637  040274           MOV     (SP),ZWORDS     ;NO. OF ZERO WORDS TO BE WRITTEN
040362  012604                   MOV     (SP)+,R4
040364  005737  001424           TST     TSECC           ;IS THIS AN ECC TEST
040370  001403                   BEQ     7$              ;BRANCH IF NO
040372  012737  177777  034562   MOV     #-1,TSECCG      ;THESE BITS ARE TO GENERATE ECC
040400  012703  040626   7$:     MOV     #DISK,R3        ;SIMULATED DISK AREA
040404  004737  040056   2$:     JSR     PC,WRITE        ;WRITE ON SIMULATED DISK
040410  013723  040054           MOV     WWORD,(R3)+     ;STORE ON SIMULATED DISK
040414  005302                   DEC     R2
040416  001372                   BNE     2$
040420  005704                   TST     R4              ;ANY ZEROS TO BE WRITTEN ?
040422  001406                   BEQ     4$              ;BRANCH IF NONE TO BE WRITTEN

040424  004737  040056   3$:     JSR     PC,WRITE        ;WRITE ZEROS ON SIMULATED DISK
040430  013723  040054           MOV     WWORD,(R3)+     ;STORE THEM
040434  005304                   DEC     R4
040436  001372                   BNE     3$
040440  005037  034562   4$:     CLR     TSECCG  ;NO MORE ECC TO BE GENERATED
040444  012701  000002           MOV     #2,R1
040450  004737  040056   5$:     JSR     PC,WRITE        ;WRITE ECC1 AND ECC2 ON SIMULATED DISK
040454  013723  040054           MOV     WWORD,(R3)+     ;STORE IN WEEC1 AND WEEC2
040460  005301                   DEC     R1
040462  001372                   BNE     5$
040464  004737  040056           JSR     PC,WRITE        ;WRITE DATA GAP
040470  013723  040054           MOV     WWORD,(R3)+     ;STORE IT
040474  012701  000016           MOV     #14.,R1
040500  004737  040056   6$:     JSR     PC,WRITE        ;WRITE TOLERANCE GAP ZEROS
040504  013723  040054           MOV     WWORD,(R3)+     ;STORE THEM
040510  005301                   DEC     R1
040512  001372                   BNE     6$
040514  012604                   MOV     (SP)+,R4        ;;POP STACK INTO R4
040516  012603                   MOV     (SP)+,R3        ;;POP STACK INTO R3
```

```
        040520  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
        040522  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
        040524  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
     53 040526  000201                          RTS     R1              ;EXIT ---------------------------->
```

DISK SIMULATION

```
                                ;;*****************************************************************
                                ;*THIS IS THE SIMULATED DISK, ONLY ONE SECTOR OF SPACE IS ALLOCATED
                                ;;*****************************************************************

040530                          SECGAP: .BLKW   19.             ;SECTOR GAP 38 BYTES OF 0
040576                          WSSYNC: .BLKW   1               ;SECTOR GAP 1 BYTE OF 0 ONE SYNC BYTE
040600                          HEADER: .BLKW   4               ;HEADER = CYL, SECTOR/TRACK, KEY1, KEY2
040610                          WCRC:   .BLKW   1               ;CRC
040612                          HEGAP:  .BLKW   5               ;HEADER GAP 10 BYTES OF 0
040624                          HDWSYN: .BLKW   1               ;HEADER GAP 1 BYTE OF 0 ONE SYNC. BYTE
040626                          SILOTB:                         ;(ALSO USED IN SILO TEST AS SILO TABLE)
040626                          DISK:   .BLKW   256.            ;DATA SPACE
041626                          WECC1:  .BLKW   1               ;ECC1
041630                          WECC2:  .BLKW   1               ;ECC2
041632                          DTAGAP: .BLKW   1               ;DATA GAP 2 BYTES OF 0
041634                          TOLGAP: .BLKW   14.             ;TOLERANCE GAP 28 BYTES OF 0
```

```
                        ;;*****************************************************************
                        ;*WRITE HEADER AND DATA
                        ;;*****************************************************************

                        ;*THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES

                        ;*IT ISSUES DIAGNOSTIC MODE, AN EXTRA DIAGNOSTIC INDEX, AND THE
                        ;*'GO' BIT

                        ;*IT THEN CALLS THREE OTHER SUBROUTINES, WHICH IN TURN CALL OTHER
                        ;*SUBROUTINES.  THE THREE SUBROUTINES CALLED HERE ARE:

                        ;*SEARCH                ;ISSUES SECTOR CLOCKS TO SET SECTOR FOUND FLOP
                        ;*WRHEAD                ;WRITES THE SECTOR HEADER
                        ;*WRDATA                ;WRITES THE ACTUAL SECTOR DATA

                        ;*ALL OF THE ABOVE MENTIONED 'WRITING' IS ACTUALLY DONE INTO A CORE
                        ;*BUFFER AREA CALLED 'DISK' VIA THE MAINTENANCE REGISTER (RHMR)



 041670  000000         RNCTR1: .WORD   0               ;'RUN' LINE STALL COUNTER

 041672  011637  001414 COMWHD: MOV     (SP),PCJSR      ;SAVE PC OF JSR + 4
 041676  162737  000004 001414       SUB     #4,PCJSR        ;SAVE PC OF JSR
 041704  010046                     MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 041706  010146                     MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 041710  010246                     MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 041712  010346                     MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 041714  010446                     MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 041716  010546                     MOV     R5,-(SP)        ;;PUSH R5 ON STACK

 041720  012777  000001 137332       MOV     #DMD,@RHMR      ;SET DIAGNOSTIC MODE
 041726  052777  000004 137324       BIS     #MINX,@RHMR     ;SET DIAGNOSTIC INDEX
 041734  042777  000004 137316       BIC     #MINX,@RHMR     ;CLEAR DIAGNOSTIC INDEX
 041742  052777  000001 137270       BIS     #GO,@RHCS1      ;SET 'GO' BIT & STALL 'TILL 'RUN'
                                                         ;(FUNCTION CODE WAS SET UP BY THE TEST)
 041750  012737  000113 041670 RNWAT1: MOV     #75.,RNCTR1     ;LOAD STALL COUNTER = APPROX 450US
                                                         ;FOR 11/50 CPU
 041756  005337  041670       1$:     DEC     RNCTR1  ;COUNT DOWN 1 TIME
 041762  001375                     BNE     1$              ;CONTINUE UNTIL = 0

 041764  013746  042050             MOV     WSECTR,-(SP)    ;GET DESIRED SECTOR/TRACK
 041770  042716  177740             BIC     #177740,(SP)    ;MAKE ONLY SECTOR
 041774  012637  042004             MOV     (SP)+,WTRK      ;SAVE SECTOR
 042000  004137  042756       2$:     JSR     R1,SEARCH       ;ISSUE SECTOR CLOCKS TO GET TO
                                                         ;DESIRED SECTOR <------------------>

 042004  000000         WTRK:   .WORD   0               ;SECTOR NO.
 042006  012701  000240       MOV     #+NOP,R1        ;GOING TO MOVE NOPS
 042012  010137  042060       MOV     R1,SEGPER       ;NOP INTO SEGAP
 042016  010137  042062       MOV     R1,FSYNER       ;NOP INTO FSYNER
 042022  010137  042064       MOV     R1,ERHEAD       ;NOP INTO ERHEAD
 042026  010137  042066       MOV     R1,ERCRC        ;NOPINTO ERCRC
```

```
     53 042032  010137  042070                  MOV     R1,ERHDGP       ;NOP INTO ERHDGAP
     54 042036  010137  042072                  MOV     R1,HDESYN       ;NOP INTO HDESYN
     55
     56 042042  004137  042142                  JSR     R1,WRHEAD       ;WRITE THE HEADER <---------------->
     57
     58 042046  000000                  WCYL:   0                       ;CYLINDER
     59 042050  000000                  WSECTR: 0                       ;SECTOR AND TRACK
     60 042052  000000                  WKEY1:  0                       ;KEY1
     61 042054  000000                  WKEY2:  0                       ;KEY2
     62 042056  000000                  GCRC:   0                       ;GOOD CRC
     63
     64
     65                                 ;DUMMY ERROR CALL LOCATIONS FOR THE WRITE HEADER OPERATION
     66
     67 042060  000240                  SEGPER: NOP                     ;IF 'ERROR 6' INSERTED BY
     68                                                                 ;WRHEAD SUBROUTINE, THEN
     69                                                                 ;SECTOR GAP GOING ON DISK
     70                                                                 ;IS NOT RIGHT.
     71
     72                                                                 ;WORD NO. CONTAINS WHICH
     73                                                                 ;WORD IS WRONG, THAT IS
     74                                                                 ;FIRST OF TENTH, OR WHAT EVER.
     75
     76                                                                 ;BAD WORD IS WHAT IS GOING ON DISK
     77
     78 042062  000240                  FSYNER: NOP                     ;IF 'ERROR 6' INSERTED BY
     79                                                                 ;WRHEAD SUBROUTINE, THEN
     80                                                                 ;THE LAST 0 BYTE OF SECTOR
     81                                                                 ;GAP OF FIRST SYNC. BYTE
     82                                                                 ;AFTER SECTOR GAP IS IN
     83                                                                 ;ERROR.
     84
     85                                                                 ;WORD NO. CONTAINS 20
     86                                                                 ;RIGHT BYTE IS SECTOR GAP
     87                                                                 ;LEFT BYTE IS SYNC. BYTE.
     88
     89                                                                 ;BAD WORD IS WHAT IS GOING ON
     90                                                                 ;DISK
     91
     92 042064  000240                  ERHEAD: NOP
     93                                                                 ;IF 'ERROR 6' INSERTED BY
     94                                                                 ;WRHEAD SUBROUTINE, THEN
     95                                                                 ;HEADER GOING ON DISK
     96                                                                 ;IS WRONG.
     97
     98                                                                 ;WORD NO 1 = CYLINDER NO
     99                                                                 ;WORD NO 2 = SECTOR/TRACK
    100                                                                 ;WORD NO 3 = KEY1
    101                                                                 ;WORD NO 4 = KEY2
    102
    103                                                                 ;BAD WORD IS WHAT IS GOING ON
    104                                                                 ;DISK
    105
    106 042066  000240                  ERCRC:  NOP
    107                                                                 ;IF 'ERROR 6' INSERTED BY
    108                                                                 ;WRHEAD SUBROUTINE, THEN CRC WRITTEN
    109                                                                 ;ON DISK IS IN ERROR.
```

```
                                                        ;GOOD DATA IS WHAT SHOULD BE ON DISK
                                                        ;BAD DATA IS WHAT IS GOING ON DISK.

                                                        ;WORD NO IS 5

042070  000240                  ERHDGP: NOP
                                                        ;IF "ERROR 6" INSERTED BY
                                                        ;WRHEAD SUBROUTINE, THEN HEADER
                                                        ;GAP GOING ON DISK IS WRONG.

                                                        ;WORD NO GIVES WHICH OF
                                                        ;THE HEADER GAP WORDS
                                                        ;ARE WRONG. FOR EXAMPLE:

                                                        ;WORD NO 1 = FIRST HEADER
                                                        ;            GAP WORD
                                                        ;BAD WORD IS WHAT IS GOING ON DISK

042072  000240                  HDESYN: NOP
                                                        ;IF "ERROR 6" INSERTED BY
                                                        ;WRHEAD SUBROUTINE, THEN LAST
                                                        ;HEADER GAP BYTE OR HEADER
                                                        ;SYNC BYTE GOING ON DISK IS WRONG.

                                                        ;WORD NO = 5

                                                        ;BAD DATA IS WHAT IS GOING
                                                        ;ON DISK, RIGHT BYTE IS HEADER
                                                        ;GAP 0'S BYTE, LEFT BYTE IS HEADER
                                                        ;GAP SYNC.


042074  005737  001406          TST     ERFLG$  ;ARE ANY ERRORS DETECTED ?
042100  001004                  BNE     FOUT            ;IF YES BRANCH
042102  004137  040276          JSR     R1,WRDATA       ;WRITE THE DATA

042106  000000          FNWORD: .WORD   0               ;FORMAT COMMAND NO. OF DATA
042110  000000                  .WORD   0

042112                  FOUT:
042112  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
042114  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
042116  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
042120  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
042122  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
042124  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
042126  000207                  RTS     PC              ;EXIT
```

```
                                ;;*****************************************************************
                                ;*WRITE HEADER
                                ;;*****************************************************************

                                ;*R0 = MAINT.REG.
                                ;*R1 = SIMULATED DISK
                                ;*R2 = BYTE COUNT
                                ;*R3 = WRITE WORD
                                ;*R5 = WORD COUNT

042130  000000                  SCYL:   0                       ;
042132  000000                  SSECTR: 0                       ;
042134  000000                  SKEY1:  0                       ;
042136  000000                  SKEY2:  0                       ;
042140  000000                  SCRC:   0                       ;

042142  012137  042130          WRHEAD: MOV     (R1)+,SCYL
042146  012137  042132                  MOV     (R1)+,SSECTR
042152  012137  042134                  MOV     (R1)+,SKEY1
042156  012137  042136                  MOV     (R1)+,SKEY2
042162  012137  042140                  MOV     (R1)+,SCRC
042166  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK

042170  012701  040530                  MOV     #SECGAP,R1      ;SIMULATED DISK INDICATOR
042174  013700  001260                  MOV     RHMR,R0 ;R0 NOW HAS MAINT. REG. ADDR.
042200  012710  000001                  MOV     #DMD,@R0        ;SET DIAG. MODE
042204  012705  000002                  MOV     #2,R5           ;WORD COUNTER
042210  052710  000010                  BIS     #MSTCK,@R0      ;SET SECTOR FOR FIRST BYTE
042214  012710  000013          1$:     MOV     #MSTCK!MCLK!DMD,@R0;SET SECTOR, CLOCK, DIAG.
                                                                ;MODE, RESET INDEX
042220  032710  000040                  BIT     #MWR,@R0        ;CHECK WRITE BIT IN MAINT. REG.
042224  001403                          BEQ     2$
042226  000261                          SEC                     ;SET CARRY
042230  006003                          ROR     R3              ;MOVE ONE FORWARD
042232  000402                          BR      3$              ;
042234  000241                  2$:     CLC                     ;CLEAR CARRY
042236  006003                          ROR     R3              ;MOVE ZERO FORWARD
042240  012710  000001          3$:     MOV     #DMD,@R0        ;CLEAR CLOCK, SECTOR
042244  012702  000007                  MOV     #7,R2           ;BYTE COUNTER
042250  052710  000002          4$:     BIS     #MCLK,@R0       ;SET BIT CLOCK
042254  032710  000040                  BIT     #MWR,@R0        ;CHECK WRITE BIT IN MAINT.REG.
042260  001403                          BEQ     5$              ;BRANCH IF ZERO

042262  000261                          SEC                     ;SET CARRY
042264  006003                          ROR     R3              ;MOVE ONE FORWARD
042266  000402                          BR      6$
042270  000241                  5$:     CLC
042272  006003                          ROR     R3
042274  012710  000001          6$:     MOV     #DMD,@R0
042300  005302                          DEC     R2
042302  001362                          BNE     4$
042304  005305                          DEC     R5
042306  001342                          BNE     1$
042310  010321                          MOV     R3,(R1)+
042312  005703                          TST     R3
042314  001414                          BEQ     7$
```

```
042316  012737  000001  037030          MOV     #1,ERWORD
042324  005037  001124                  CLR     $GDDAT
042330  010337  001126                  MOV     R3,$BDDAT
042334  012737  104006  042060          MOV     #104006,SEGPER
042342  000137  042750                  JMP     17$             ;BRANCH OUT ---------------------->
042346  012702  000022          7$:     MOV     #18.,R2         ;COUNT NO. OF SECTOR GAP
042352  012737  000024  037030  10$:    MOV     #20.,ERWORD     ;COUNT TO GIVE ERROR WORD
042360  004737  040056                  JSR     PC,WRITE        ;WRITE SECTOR GAP
042364  013721  040054                  MOV     WWORD,(R1)+     ;STORE SECTOR GAP WORD
042370  001413                          BEQ     11$
042372  160237  037030                  SUB     R2,ERWORD       ;IF NOT GET ERROR WORD NO.
042376  005037  001124                  CLR     $GDDAT  ;GOOD WORD
042402  013737  040054  001126          MOV     WWORD,$BDDAT;BAD WORD
042410  012737  104006  042060          MOV     #104006,SEGPER;STORE "ERROR 6" IN SEGPER
042416  000554                          BR      17$             ;BRANCH OUT
042420  005302                  11$:    DEC     R2              ;HAVE 18 WORDS OF ZEROS BEEN WRITTEN ?
042422  001353                          BNE     10$             ;IF NOT CONTINUE
                                ;AT THIS POINT THE SECTOR FOUND FLOP SHOULD
                                ;BE HIGH,  SO THAT THE HEADER SYNC BYTE CAN BE GIVEN.
                                ;HOWEVER, IN THE DRIVE TIMING ERROR TEST THE REST OF THE ROUTINE
                                ;IS ABORTED
042424  005737  001426                  TST     TESDTE  ;IS THIS A DRIVE TIMING ERROR TEST ?
042430  001147                          BNE     17$             ;BRANCH OUT IF YES ------------------->
042432  004737  040056                  JSR     PC,WRITE        ;WRITE ONE SECTOR GAP 0 BYTE
                                                                ;AND ONE SYNC. BYTE = 230
042436  013711  040054                  MOV     WWORD,(R1)      ;SAVE 0 BYTE AND SYNC BYTE
042442  023721  037012                  CMP     RSYNC,(R1)+     ;IF SYNC. BYTE RIGHT
042446  001414                          BEQ     12$             ;IF YES CONTINUE OPERATION
042450  012737  000024  037030          MOV     #20.,ERWORD     ;IF NOT GET READY FOR ERROR PRINT
042456  013737  037012  001124          MOV     RSYNC,$GDDAT;GOOD WORD
042464  014137  001126                  MOV     -(R1),$BDDAT    ;BAD WORD
042470  012737  104006  042062          MOV     #104006,FSYNER;INSERT "ERROR 6" IN FSYNER
042476  000524                          BR      17$             ;BRANCH OUT ------------------------>
042500  012702  000004          12$:    MOV     #4,R2           ;FOUR HEADER WORDS
042504  012703  042130                  MOV     #SCYL,R3        ;POINTER FOR HEADER TABLE
042510  012737  000005  037030  13$:    MOV     #5,ERWORD       ;ERROR WORD NO SET
042516  004737  040056                  JSR     PC,WRITE        ;WRITE 4 HEADER WORDS
042522  013711  040054                  MOV     WWORD,(R1)      ;STORE WRITTEN WORD
042526  022321                          CMP     (R3)+,(R1)+     ;IS IT CORRECT ?
042530  001412                          BEQ     14$             ;IF GOOD CONTINUE OPERATION
                                                                ;IF NOT GET READY FOR ERROR PRINT
042532  160237  037030                  SUB     R2,ERWORD       ;WORD NO
042536  014337  001124                  MOV     -(R3),$GDDAT    ;GOOD DATA
042542  014137  001126                  MOV     -(R1),$BDDAT    ;BAD DATA
042546  012737  104006  042064          MOV     #104006,ERHEAD;INSERT "ERROR 6"
042554  000475                          BR      17$             ;BRANCH OUT ------------------------>
042556  005302                  14$:    DEC     R2              ;ARE 4 HEADER WORDS DONE?
```

```
                                                BNE     13$             ;IF NOT CONTINE
                                                JSR     PC,WRITE        ;WRITE CRC
                                                MOV     WWORD,(R1)      ;STORE CRC
                                                CMP     (R1)+,GCRC      ;COMPARE GOOD CRC
                                                BEQ     20$             ;IF GOOD CONTINUE OPERATION
                                                                        ;IF NOT GET READY FOR ERROR PRINT

                                                MOV     -(R1),$BDDATA   ;BAD CRC WRITTEN
                                                MOV     GCRC,$GDDAT     ;GOOD CRC
                                                MOV     #5,ERWORD       ;ERROR WORD NO
                                                MOV     #104006,ERCRC   ;INSERT ERROR 6
                                                BR      17$             ;EXIT ------------------------------->

                                        20$:    MOV     #5,R2           ;NO OF HEADER GAP
                                        15$:    MOV     #6,ERWORD       ;ERROR WORD NO SET
                                                JSR     PC,WRITE        ;WRITE HEADER GAP
                                                MOV     WWORD,(R1)+     ;STORE
                                                BEQ     16$             ;IF GOOD CONTINUE OPERATION
                                                                        ;IF NOT GET READY FOR PRINT

                                                SUB     R2,ERWORD       ;ERROR WORD NO
                                                CLR     $GDDAT  ;GOOD DATA
                                                MOV     -(R1),$BDDAT    ;BAD DATA
                                                MOV     #104006,ERHDGP;STORE "ERROR 6"
                                                BR      17$             ;BRANCH OUT ---------------------->

                                        16$:    DEC     R2              ;ARE 5 HEADER GAP ZEROS DONE ?
                                                BNE     15$             ;IF NOT CONTINUE
                                                JSR     PC,WRITE
                                                MOV     WWORD,(R1)
                                                CMP     RSYNC,(R1)+
                                                BEQ     17$             ;A-OK, EXIT ---------------------->

                                                MOV     #5,ERWORD       ;IF NOT GET READY FOR ERROR PRINT
                                                MOV     -(R1),$BDDAT
                                                MOV     RSYNC,$GDDAT
                                                MOV     #104006,HDESYN

                                        17$:
                                                MOV     (SP)+,R1        ;;POP STACK INTO R1
                                                RTS     R1              ;EXIT ---------------------------->
```

```
                                ;;*****************************************************************
                                ;*SEARCH SECTOR
                                ;;*****************************************************************

                                ;*R0=RHMR ADDRESS
                                ;*R1=PASSED ARGUMENT (SECTOR SEARCHED FOR)
                                ;*R2=CLOCK COUNT (PER BYTE)
                                ;*R3=SECTOR COUNTER FROM R1
                                ;*R5=BYTES PER WORD COUNT

                                ;*BEFORE INDEX IS GIVEN TWO SECTOR CLOCKS ARE GIVEN TO RESET
                                ;*SECTOR PULSE IN CASE IT IS SET.

                                ;*AT THE BEGINNING OF EACH SECTOR, ONE SECTOR CLOCK HAS TO RISE
                                ;*BEFORE MAINT. CLOCK, THEN EVERY EIGHT MAINT.CLOCKS, ONE SECTOR CLOCK
                                ;*IS IDENTICAL WITH THE MAINT. CLOCK
                                ;*THE SECTOR CLOCKS ARE NUMBERED AS FOLLOWS:

                                ;*THE SECTOR CLOCK UNDER INDEX - 0
                                ;*THE NEXT - 1
                                ;*THE NEXT - 2
                                ;*ETC.
                                ;*THEN THE LAST SECTOR CLOCK IN ONE SECTOR HAS NUMBER - 608
                                ;*THE NEXT SECTOR WILL HAVE 608 SECTOR CLOCKS
                                ;*THE NEXT SECTOR THEN HAS ANOTHER 608 SECTOR CLOCKS
                                ;*AND SO ON

042754  000000                  SECTR:  0                       ;SECTOR SEARCHED FOR

042756  012137  042754          SEARCH: MOV     (R1)+,SECTR ;SAVE SECTOR SEARCHED FOR
042762  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
042764  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
042766  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
042770  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
042772  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
042774  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
042776  013700  001260                  MOV     RHMR,R0 ;NOW R0 HAS MAINTENANCE REG. ADR.
043002  013703  042754                  MOV     SECTR,R3        ;LOAD SECTOR COUNTER

043006  012710  000001                  MOV     #DMD,@R0        ;SET DIAGNOSTIC MODE
043012  052710  000010                  BIS     #MSTCK,@R0      ;SET SECTOR CLOCK
043016  042710  000010                  BIC     #MSTCK,@R0      ;CLEAR SECTOR CLOCK
043022  052710  000010                  BIS     #MSTCK,@R0      ;SET SECTOR CLOCK
043026  042710  000010                  BIC     #MSTCK,@R0      ;CLEAR SECTOR CLOCK
                                                                ;THE ABOVE TWO SECTOR CLOCKS ARE GIVEN FOR
                                                                ;RESETING SECTOR PULSE
                                                                ;IN CASE IT STARTS SET
043032  052710  000014                  BIS     #MINX!MSTCK,@R0 ;SET INDEX AND SECTOR CLOCK
043036  042710  000014                  BIC     #MINX!MSTCK,@R0 ;RESET INDEX AND SECTOR CLOCK
043042  005703                          TST     R3              ;TEST FOR SECTOR 0
043044  001461                          BEQ     7$              ;IF SECTOR 0 IS REQUIRED, EXIT ------>

                                ;NOW 304 WORDS WILL START (608 BYTES)

                                ;*FOR FIRST BYTE MAINT. SECTOR CLOCK WILL GO HIGH, THEN MAINT. CLOCK WILL GO HIGH
                                ;*BOTH WILL COME DOWN TOGETHER, THEN SEVEN MAINT. CLOCKS WILL BE GIVEN
                                ;*FOR SECOND BYTE, AND ALL OTHER BYTES TILL NEXT SECTOR, SECTOR CLOCK
```

```
                                        ;*WILL BE IDENTICAL WITH ONE MAINT. CLOCK
                                        ;ONE WORD ONLY
                                        
043046  012702  000010          1$:     MOV     #8.,R2  ;BYTE COUNTER
043052  012705  000002                  MOV     #2,R5   ;BYTES PER WORD
043056  052710  000010                  BIS     #MSTCK,@R0      ;SET SECTOR CLOCK
043062  052710  000002                  BIS     #MCLK,@R0       ;SET MAINT. CLOCK
043066  000402                          BR      3$              ;BRANCH TO CLEAR SECTOR AND CLOCK
043070  052710  000012          2$:     BIS     #MSTCK!MCLK,@R0 ;SET BOTH CLOCKS
043074  042710  000012          3$:     BIC     #MSTCK!MCLK,@R0 ;CLEAR BOTH CLOCKS
043100  052710  000002          8$:     BIS     #MCLK,@R0       ;SET MAINT. CLOCK
043104  042710  000002                  BIC     #MCLK,@R0       ;CLEAR MAINT.CLOCK
043110  005302                          DEC     R2              ;BYTE COUNTER
043112  001372                          BNE     8$              ;CONTINUE IF BYTE NOT COMPLETE
                                        
043114  012702  000007                  MOV     #7,R2           ;SETUP FOR SECOND BYTE
043120  005305                          DEC     R5              ;IS WORD COMPLETE?
043122  001362                          BNE     2$              ;CONTINUE IF NOT COMPLETE
                                                                ;TO GIVE SECTOR CLOCK AND MAINT. CLOCK
                                        
                                        
                                        ;NOW 303 WORDS ARE LEFT AND ALL ARE IDENTICAL
                                        
043124  012701  000457                  MOV     #303.,R1        ;WORDS PER SECTOR COUNTER
043130  012705  000002          4$:     MOV     #2,R5           ;BYTES PER WORD COUNTER
043134  012702  000007          5$:     MOV     #7,R2           ;BIT COUNTER (MAINT.CLOCK COUNTER)
043140  052710  000012                  BIS     #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND MAINT. CLOCK
043144  042710  000012                  BIC     #MSTCK!MCLK,@R0 ;CLEAR CLOCKS
043150  052710  000002          6$:     BIS     #MCLK,@R0       ;SET MAINT. CLOCK
043154  042710  000002                  BIC     #MCLK,@R0       ;RESET MAINT. CLOCK
043160  005302                          DEC     R2              ;IS BYTE COMPLETE ?
043162  001372                          BNE     6$              ;CONTINUE IF NOT COMPLETE
                                        
043164  005305                          DEC     R5              ;IS WORD COMPLETE ?
043166  001362                          BNE     5$              ;CONTINUE IF NOT
                                        
043170  005301                          DEC     R1              ;IS SECTOR COMPLETE ?
043172  001356                          BNE     4$              ;CONTINUE IF NOT
                                        
043174  052710  000010                  BIS     #MSTCK,@R0      ;SET SECTOR CLOCK 1 MORE TIME (FOR 609)
043200  042710  000010                  BIC     #MSTCK,@R0      ;CLEAR SECTOR CLOCK
043204  005303                          DEC     R3              ;IS REQUIRED NO OF SECTORS COMPLETE ?
043206  001317                          BNE     1$              ;CONTINUE IF NOT
                                        
043210                          7$:
043210  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
043212  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
043214  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
043216  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
043220  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
043222  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
043224  000201                          RTS     R1
                                        
                                        ;;**************************************************************************
                                        ;*READ ONE SECTOR OF DATA
                                        ;;**************************************************************************
```

```
                                        RNO:    0                       ;NO. OF WORDS READ
                                        RCOM:   0                       ;EXTRA STORAGE

                                        REDATA: MOV     (R1)+,RNO       ;SAVE NO. OF WORDS ONLY FOR INFORMATION
                                                MOV     (R1)+,RCOM      ;EXTRA WORD ONLY FOR INFORMATION
                                                MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                                TST     TSECC           ;IS THIS AN ECC TEST
                                                BEQ     1$              ;BRANCH IF NO
                                                MOV     #-1,TSECCG      ;THESE BITS ARE TO GENERATE ECC
                                        1$:     MOV     #258.,R2        ;256 WORDS PER SECTOR
                                                                        ;PLUS 2 ECC WORDS
                                                MOV     #DISK,R3        ;POINTE TO DISK SIMULATION
                                        2$:     MOV     (R3)+,WORD      ;READY TO READ CONTENTS
                                                JSR     PC,READ         ;READ
                                                DEC     R2              ;IS 256 WORDS DONE?
                                                BNE     2$              ;IF NOT BRANCH
                                                TST     TSECC           ;IS THIS AN ECC TEST
                                                BNE     4$              ;BRANCH OUT IF YES
                                                CLR     TSECCG  ;NO MORE ECC BITS ARE TO BE GENERATED
                                                MOV     #15.,R2         ;ONE DATA GAP, 14 TOLERANCE GAP
                                        3$:     MOV     (R3)+,WORD      ;READY TO READ CONTENTS OF WORD
                                                JSR     PC,READ ;READ
                                                DEC     R2              ;COUNT
                                                BNE     3$              ;BRANCH IF 14 NOT DONE
                                        4$:
                                                MOV     (SP)+,R1        ;;POP STACK INTO R1
                                                RTS     R1              ;RETURN

                                        RPVECT:
                                                TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  /UNEXPECTED RP INTERRUPT @ PC = /
                                        64$:
                                                TYPOC                           ;TYPE FROM PC
                                                MOV     #RPVECT,@RPVEC          ;RESTORE TRAP VECTOR
                                                RTI                             ;CHANGE TO CONTINUE
```

```
      1                                 .SBTTL  SCOPE HANDLER ROUTINE

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

 043422                                 $SCOPE:
 043422  104407                                 CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
 043424  005037  037024                         CLR     NOSYNC          ;CLEAR FLAG FOR HEADER ERROR COMMANDS
 043430  005037  001424                         CLR     TSECC           ;CLEAR FLAG FOR ECC TEST
                                                                        ;WHEN =177777 IT IS AN ECC TEST
                                                                        ;WHEN =0 IT IS NOT AN ECC TEST

 043434  005037  034562                         CLR     TSECCG          ;EVEN IN AN ECC TEST EVERY CLOCK
                                                                        ;IS NOT TO GENERATE ECC
                                                                        ;IF =177777 GENERATE ECC
                                                                        ;IF =0 DO NOT GENERATE ECC
 043440  005037  001426                         CLR     TESDTE          ;DRIVE TIMING ERROR TEST
 043444                                 1$:
 043444  032777  040000  135466         1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
 043452  001402                                 BEQ     9$              ;;NO IF SW14=0
 043454  000137  044044                         JMP     $OVER           ;;JUMP OVER SCOPE ROUTINE
 043460                                 9$:
                                        ;#####START OF CODE FOR THE XOR TESTER#####
 043460  000416                         $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
 043462  013746  000004                         MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 043466  012737  043506  000004                 MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
 043474  005737  177060                         TST     @#177060        ;;TIME OUT ON XOR?
 043500  012637  000004                         MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 043504  000544                                 BR      $SVLAD          ;;GO TO THE NEXT TEST
 043506  022626                         5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
 043510  012637  000004                         MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 043514  000504                                 BR      7$              ;;LOOP ON THE PRESENT TEST
 043516                                 6$:;#####END OF CODE FOR THE XOR TESTER#####
 043516  032777  000400  135414                 BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
 043524  001404                                 BEQ     2$              ;;BR IF NO
 043526  127737  135406  001102                 CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
 043534  001543                                 BEQ     $OVER           ;;BR IF YES
 043536  105737  001103                 2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
 043542  001502                                 BEQ     3$              ;;BR IF NO
 043544  022737  177777  046430                 CMP     #-1,CPSAVE      ;;SEE IF TIMEOUT WAS PREVIOUSLY RECORDED
 043552  001455                                 BEQ     2003$           ;;KICK AROUND ROUTINE IF SO
 043554  013746  000004                         MOV     ERRVEC,-(SP)    ;;SAVE CONTENTS OF ERROR VECTOR
 043560  012737  043576  000004                 MOV     #2000$,ERRVEC   ;;SETUP 'TRAP' RETURN ADDRESS
 043566  013737  177766  046430                 MOV     177766,CPSAVE   ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
 043574  000406                                 BR      2001$
 043576  012737  177777  046430         2000$:  MOV     #-1,CPSAVE      ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
 043604  012716  043612                         MOV     #2001$,(SP)     ;;SETUP RETURN ADDRESS
```

SCOPE HANDLER ROUTINE

```
043610  000002                          RTI
043612  012637  000004          2001$:  MOV     (SP)+,ERRVEC    ;;RESTORE CONTENTS OF ERROR VECTOR

043616  022737  177777  046430  2002$:  CMP     #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
043624  001430                          BEQ     2003$           ;;BRANCH IF SO
043626  032737  000001  046430          BIT     #BIT00,CPSAVE   ;;SEE IF THE POWER MONITOR BIT IS ON
043634  001424                          BEQ     2003$           ;;BRANCH TO CONTINUE ROUTINE IF CLEAR
043636  042737  000001  177766          BIC     #BIT00,177766   ;;CLEAR THE BIT FOUND TO BE SET
043644  013746  001140                  MOV     SWR,-(SP)       ;;SAVE SWR ADDRESS
043650  017646  000000                  MOV     @(SP),-(SP)     ;;SAVE SWR VALUE
043654  012737  000176  001140          MOV     #176,SWR        ;;GET SOFTWARE SWR ADDRESS
043662  011677  135252                  MOV     (SP),@SWR       ;;GET CURRENT SWR VALUE
043666  042777  001000  135244          BIC     #BIT09,@SWR     ;;DON'T ALLOW LOOP ON ERROR ON THIS ERROR
043674  104177                          EMT     177             ;;CALL SPECIAL POWER FAIL BIT ERROR CALL
043676  012676  000C00                  MOV     (SP)+,@(SP)     ;;RESTORE SWR TO ORIGINAL VALUE
043702  012637  001140                  MOV     (SP)+,SWR       ;;RESTORE SWR ADDRESS
043706                          2003$:
043706  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
043714  101015                          BHI     3$              ;;BR IF NO
043716  032777  001000  135214          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
043724  001404                          BEQ     4$              ;;BR IF NO
043726  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
043734  000443                          BR      $OVER
043736  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
043742  005037  001212                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
043746  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
043750  032777  004000  135162  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
043756  001011                          BNE     1$              ;;BR IF YES
043760  005737  001100                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
043764  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
043766  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
043772  023737  001212  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
044000  002021                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
044002  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
044010  013737  044060  001212          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
044016  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
044022  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
044026  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
044032  005037  001214                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
044036  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
044044  013777  001102  135070  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
044052  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
044056  000002                          RTI                     ;;FIXES PS
044060  000004                  $MXCNT: 4                       ;;MAX. NUMBER OF ITERATIONS
```

```
                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

044062                          $TYPDS:
044062  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
044064  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
044066  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
044070  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
044072  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
044074  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
044100  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
044104  100004                          BPL     1$              ;;BR IF INPUT IS POS.
044106  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
044110  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
044116  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
044120  012703  044276                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
044124  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
044130  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
044132  016001  044266                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
044136  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
044140  002402                          BLT     4$              ;;BR IF DONE
044142  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
044144  000774                          BR      3$
044146  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
044150  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
044152  001002                          BNE     5$              ;;FALL THROUGH IF 0
044154  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
044156  100407                          BMI     7$              ;;BR IF YES
044160  106316                  5$:     ASLB    (SP)            ;;MSD?
044162  103003                          BCC     6$              ;;BR IF NO
044164  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
044172  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
044176  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
044202  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
044204  005720                          TST     (R0)+           ;;JUST INCREMENTING
044206  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
044212  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
044214  003002                          BGT     8$              ;;GO TO EXIT
044216  010502                          MOV     R5,R2           ;;GET THE LSD
044220  000764                          BR      6$              ;;GO CHANGE TO ASCII
044222  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
044224  100003                          BPL     9$              ;;BR IF NO
044226  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
044234  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
044236  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
044240  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
044242  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
044244  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
```

```
044246  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
044250  104401  044276                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
044254  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
044262  012616                          MOV     (SP)+,(SP)
044264  000002                          RTI                     ;;RETURN TO USER
044266  023420                  $DTBL:  10000.
044270  001750                          1000.
044272  000144                          100.
044274  000012                          10.
044276                          $DBLK:  .BLKW   4
```

```
                                .SBTTL  TYPE ROUTINE

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

044306  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
044312  100002                          BPL     1$              ;;BR IF YES
044314  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
044316  000407                          BR      3$              ;;LEAVE
044320  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
044322  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
044326  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
044330  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
044332  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
044334  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
044336  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
044342  000002                          RTI                     ;;RETURN
044344  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
044350  001430                          BEQ     8$
044352  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
044356  001006                          BNE     5$
044360  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
044362  104401                          TYPE                    ;;TYPE A CR AND LF
044364  001223                          $CRLF
044366  105037  044574                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
044372  000755                          BR      2$              ;;GET NEXT CHARACTER
044374  004737  044456          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
044400  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
044404  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
044406  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
044412  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
044416  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
044420  004737  044456                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
044424  105337  044574                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
044430  000770                          BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

044432  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
044436  004737  044456          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
044442  132737  000007  044574          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
044450  001372                          BNE     9$              ;;TAB STOP
044452  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
044454  000724                          BR      2$              ;;GET NEXT CHARACTER
```

```
044456                          $TYPEC:
044456  105777  134462                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?
044462  100022                  BPL     10$             ;;BR IF NOT
044464  017746  134456          MOV     @$TKB,-(SP)     ;;GET CHAR
044470  042716  177600          BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS
044474  122716  000023          CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF
044500  001012                  BNE     102$            ;;BR IF NOT
044502                          101$:
044502  105777  134436          TSTB    @$TKS           ;;WAIT FOR CHAR
044506  100375                  BPL     101$
044510  117716  134432          MOVB    @$TKB,(SP)      ;;GET CHAR
044514  042716  177600          BIC     #177600,(SP)    ;;STRIP IT
044520  122716  000021          CMPB    #$XON,(SP)      ;;WAS IT XON?
044524  001366                  BNE     101$            ;;BR IF NOT
044526                          102$:
044526  005726                  TST     (SP)+           ;;FIX STACK
044530                          10$:
044530  105777  134414          TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
044534  100375                  BPL     10$
044536  116677  000002  134406  MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
044544  122766  000015  000002  CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
044552  001003                  BNE     1$              ;;BRANCH IF NO
044554  105037  044574          CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
044560  000406                  BR      $TYPEX          ;;EXIT
044562  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
044570  001402                  BEQ     $TYPEX          ;;BRANCH IF YES
044572  105227                  INCB    (PC)+           ;;COUNT THE CHARACTER
044574  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
044576  000207          $TYPEX: RTS     PC
```

```
                                .SBTTL  TTY INPUT ROUTINE

                                ;;*******************************************************************
                                .ENABL  LSB
044600  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
044602  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
044604  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
044606                          $TKQSRT: .BLKB  9.              ;;TTY KEYBOARD QUEUE
        044617                  $TKQEND=.
                                .EVEN

                                ;*TK INITIALIZE ROUTINE
                                ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
                                ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
                                ;
                                ;*CALL:
                                ;*      JSR     PC,$TKINT
                                ;*      RETURN
                                ;
044620  005037  044600          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
044624  012737  044606  044602          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
044632  013737  044602  044604          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
044640  012737  044670  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
044646  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;'BR' LEVEL 4
044654  005777  134266                  TST     @$TKB           ;;CLEAR DONE FLAG
044660  012777  000100  134256          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
044666  000207                  .       RTS     PC              ;;RETURN TO CALLER

                                ;*TK SERVICE ROUTINE
                                ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
                                ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
                                ;*IT IN THE QUEUE.
                                ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
                                ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (OPERSEL)
                                ;
044670  117746  134252          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
044674  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
044700  021627  000021                  CMP     (SP),#$XON      ;;IS IT A RANDOM XON?
044704  001002                          BNE     30$             ;;BRANCH IF NO
044706  005726                          TST     (SP)+           ;;CLEAN RANDOM XON OFF STACK
044710  000002                          RTI                     ;;RETURN
044712                          30$:
044712  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
044716  001007                          BNE     1$              ;;BRANCH IF NO
044720  104401  045671                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
044724  004737  044620                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
044730  005726                          TST     (SP)+           ;;CLEAN UP STACK
044732  000137  031004                  JMP     OPERSEL         ;;CONTROL C RESTART
044736  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
044742  001004                          BNE     2$              ;;BRANCH IF NO
044744  022737  000176  001140          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
044752  001500                          BEQ     6$              ;;GO TO SWR CHANGE

044754                          2$:
044754  022737  000011  044600          CMP     #9.,$TKCNT      ;;IS THE QUEUE FULL?
044762  001004                          BNE     3$              ;;BRANCH IF NO
044764  104401  001216                  TYPE    ,$BELL          ;;RING THE TTY BELL
```

TTY INPUT ROUTINE

```
045770  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
045772  000451                          BR      5$              ;;EXIT
045774  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
045000  001021                          BNE     32$             ;;BRANCH IF NO
045002  005077  134136                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
045006  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
045010  105777  134130          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
045014  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
045016  117746  134124                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
045022  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
045026  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
045032  001366                          BNE     31$             ;;BRANCH IF NO
045034  012777  000100  134102          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
045042  000002                          RTI                     ;;RETURN
045044  005237  044600          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
045050  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
045054  002405                          BLT     4$              ;;BRANCH IF YES
045056  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
045062  003002                          BGT     4$              ;;BRANCH IF YES
045064  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
045070  112677  177506          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
045074  005237  044602                  INC     $TKQIN          ;;UPDATE THE POINTER
045100  023727  044602  044617          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
045106  001003                          BNE     5$              ;;BRANCH IF NO
045110  012737  044606  044602          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
045116  000002                  5$:     RTI                     ;;RETURN

                        ;;**********************************************************************
                        ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                        ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
                        ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
                        ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
045120  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
045126  001124                          BNE     15$             ;;EXIT IF NOT
045130  105777  134010                  TSTB    @$TKS           ;;IS A CHAR WAITING?
045134  100121                          BPL     15$             ;;IF NOT, EXIT
045136  117746  134004                  MOVB    @$TKB,-(SP)     ;;YES
045142  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
045146  022627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
045152  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
                                                                ;;AND EXIT

                        ;;**********************************************************************
                        ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                        ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                        ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
045154  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
045162  001674                          BEQ     2$              ;;BRANCH IF YES
045164  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
045166  004737  044620                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
045172  005077  133746                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
045176  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR

045204  104401  045703                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
045210  104401  045710  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
045214  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
045220  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
045222  104401  045721                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
045226  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
045230  005046                          CLR     -(SP)           ;;THE NEW SWR
045232  105777  133706          7$:     TSTB    @$TKS           ;;CHAR THERE?
045236  100375                          BPL     7$              ;;IF NOT TRY AGAIN

045240  117746  133702                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
045244  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII

045250  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
045254  001015                          BNE     9$              ;;BRANCH IF NOT
045256  104401  045671                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
045262  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
045266  123727  001135  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
045274  001003                          BNE     8$              ;;BRANCH IF NO
045276  012777  000100  133640          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
045304  000137  031004          8$:     JMP     OPERSEL         ;;CONTROL-C RESTART


045310  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
045314  001005                          BNE     10$             ;;BRANCH IF NOT
045316  104401  045676                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
045322  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
045326  000737                          BR      19$             ;;LET'S TRY IT AGAIN


045330  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
045334  001022                          BNE     16$             ;;BRANCH IF NO
045336  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
045342  001403                          BEQ     11$             ;;BRANCH IF YES
045344  016677  000002  133566          MOV     2(SP),@SWR      ;;SAVE NEW SWR
045352  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
045356  104401  001223          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
045362  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
045370  001003                          BNE     15$             ;;BRANCH IF NOT
045372  012777  000100  133544          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
045400  000002          15$:    RTI                             ;;RETURN
045402  004737  044456          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
045406  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
045412  002420                          BLT     18$             ;;BRANCH IF YES
045414  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
045420  003015                          BGT     18$             ;;BRANCH IF YES
045422  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
045426  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
045432  001403                          BEQ     17$             ;;BRANCH IF YES
045434  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
045436  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
045440  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
045442  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
045446  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
045452  000667                          BR      7$              ;;GET THE NEXT ONE
045454  104401  001222          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
045460  000720                          BR      20$             ;;SIMULATE CONTROL-U
                                .DSABL  LSB

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

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

045462  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
045464  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
045472  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
045476  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
045500  012746  045506                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
045504  000002                          RTI                     ;;POP NEW PC AND PS
045506                          64$:
045506  005737  044600          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
045512  001775                          BEQ     1$
045514  005337  044600                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
045520  117766  177060  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
045526  005237  044604                  INC     $TKQOUT         ;;UPDATE THE POINTER
045532  023727  044604  044617          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
045540  001003                          BNE     2$              ;;BRANCH IF NO
045542  012737  044606  044604          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
045550  000002                  2$:     RTI                     ;;RETURN
                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

045552  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
045554  012703  045660          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
045560  022703  045671          2$:     CMP     #$TTYIN+9.,R3   ;;BUFFER FULL?
045564  101405                          BLOS    4$              ;;BR IF YES
045566  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
045570  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
045572  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
045576  001003                          BNE     3$              ;;SKIP IF NOT
045600  104401  001222          4$:     TYPE    ,$QUES          ;;TYPE A '?'
045604  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
045606  111337  045656          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
045612  104401  045656                  TYPE    ,9$
045616  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
045622  001356                          BNE     2$              ;;LOOP IF NOT RETURN
045624  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
045630  104401  001224                  TYPE    ,$LF            ;;TYPE A LINE FEED
045634  012603                          MOV     (SP)+,R3        ;;RESTORE R3
045636  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
045640  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
045646  012766  045660  000004          MOV     #$TTYIN,4(SP)
045654  000002                          RTI                     ;;RETURN
045656     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
045657     000                          .BYTE   0               ;;TERMINATOR
045660                          $TTYIN: .BLKB   9.              ;;RESERVE 9. BYTES FOR TTY INPUT
045671     136      103      015 $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL "C"
045676     136      125      015 $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
045703     136      107      015 $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
```

```
045710  015  012  123  $MSWR:  .ASCIZ  <15><12>/SWR = /
045721  040  040  116  $MNEW:  .ASCIZ  /  NEW = /
```

```
                                .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY.
                                ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
                                ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
                                ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
                                ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
                                ;*CALL:
                                ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

045732  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
045734  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
045742  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
045744  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
045746  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
045750  104411                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
045752  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
045754  010037  046060                  MOV     R0,5$           ;;AND SAVE IT
045760  005001                          CLR     R1              ;;CLEAR DATA WORD
045762  005002                          CLR     R2
045764  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
045766  001420                          BEQ     3$              ;;IF ZERO GET OUT
045770  122716  000060                  CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
045774  003026                          BGT     4$              ;;IS AN OCTAL DIGIT
045776  122716  000067                  CMPB    #'7,(SP)
046002  002423                          BLT     4$
046004  006301                          ASL     R1              ;;*2
046006  006102                          ROL     R2
046010  006301                          ASL     R1              ;;*4
046012  006102                          ROL     R2
046014  006301                          ASL     R1              ;;*8
046016  006102                          ROL     R2
046020  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
046024  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
046026  000756                          BR      2$              ;;LOOP
046030  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
046032  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
046036  010237  046070                  MOV     R2,$HIOCT
046042  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
046044  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
046046  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
046050  000002                          RTI                     ;;RETURN
046052  005726                  4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
046054  105010                          CLRB    (R0)            ;;SET A TERMINATOR
046056  104401                          TYPE                    ;;TYPE UP THRU THE BAD CHAR.
046060  000000                  5$:     .WORD   0
046062  104401  001222                  TYPE    ,$QUES          ;;"?" "CR" & "LF"
046066  000730                          BR      1$              ;;TRY AGAIN
046070  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
```

```
                                .SBTTL  ERROR HANDLER ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO $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

046072  105037  046432         $ERROR: CLRB    IBSAVE          ;;CLEAR THE ITEM BYTE SAVE LOCATION
046076  104407                         CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
046100  012737  177777  001406         MOV     #-1,ERFLG$      ;SET ERROR FLAG
046106                          REGSA1:
046106  105237  001103         7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
046112  001775                         BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
046114  013777  001102  133020         MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
046122  032777  002000  133010         BIT     #BIT10,@SWR     ;;BELL ON ERROR?
046130  001402                         BEQ     1$              ;;NO - SKIP
046132  104401  001216                 TYPE    ,$BELL          ;;RING BELL
046136  005237  001112         1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
046142  011637  001116                 MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
046146  162737  000002  001116         SUB     #2,$ERRPC
046154  117737  132736  001114         MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
046162  032777  001000  132750         BIT     #BIT09,@SWR     ;;SEE IF LOOP ON ERROR IS SET
046170  001060                         BNE     1004$           ;;BRANCH AROUND ROUTINE IF SO
046172  122737  000177  001114         CMPB    #177,$ITEMB     ;;SEE IF THIS IS THE POWER FAIL CALL
046200  001454                         BEQ     1004$           ;;BRANCH AROUND ROUTINE IF IT IS
046202  105737  046432                 TSTB    IBSAVE          ;;SEE IF THIS IS THE 2ND ERROR CALL IN THIS ROUTINE
046206  001047                         BNE     1003$           ;;BRANCH IF SO
046210  022737  177777  046430         CMP     #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
046216  001445                         BEQ     1004$           ;;BRANCH IF SO
046220  013746  000004                 MOV     ERRVEC,-(SP)    ;;SAVE CONTENTS OF ERROR VECTOR
046224  012737  046242  000004         MOV     #1000$,ERRVEC   ;;SETUP 'TRAP' RETURN ADDRESS
046232  013737  177766  046430         MOV     177766,CPSAVE   ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
046240  000406                         BR      1001$
046242  012737  177777  046430 1000$:  MOV     #-1,CPSAVE      ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
046250  012716  046256                 MOV     #1001$,(SP)     ;;SETUP RETURN ADDRESS
046254  000002                         RTI
046256  012637  000004         1001$:  MOV     (SP)+,ERRVEC    ;;RESTORE CONTENTS OF ERROR VECTOR

046262  022737  177777  046430 1002$:  CMP     #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
046270  001420                         BEQ     1004$           ;;BRANCH IF SO
046272  032737  000001  046430         BIT     #BIT00,CPSAVE   ;;SEE IF POWER MONITOR BIT IS SET IN CPU ERR REG
046300  001414                         BEQ     1004$           ;;BRANCH IF OK
046302  042737  000001  177766         BIC     #BIT00,177766   ;;CLEAR THE BIT FOUND SET
046310  113737  001114  046432         MOVB    $ITEMB,IBSAVE   ;;MAKE IBSAVE NON-ZERO FOR DUAL ERROR CALL
046316  112737  000177  001114         MOVB    #177,$ITEMB     ;;SET $ITEMB TO SPECIAL POWER FAIL POINTER
046324  000402                         BR      1004$           ;;BRANCH OVER IBSAVE CLEARING

046326  105037  046432         1003$:  CLRB    IBSAVE          ;;CLEAR IBSAVE SO 2ND TIME THROUGH EXITS
046332                          1004$:
046332  032777  020000  132600         BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
```

ERROR HANDLER ROUTINE

```
046340  001004                          BNE     20$             ;;SKIP TYPEOUTS
046342  004737  046434                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
046346  1C4401  001223                  TYPE    ,$CRLF
046352                          20$:
046352  105737  046432          2$:     TSTB    IBSAVE          ;;SEE IF IBSAVE IS LOADED
046356  001005                          BNE     3$              ;;BRANCH IF NOT - NO HALT ON PWR MON BIT ERROR
046360  005777  132554                  TST     @SWR            ;;HALT ON ERROR
046364  100002                          BPL     3$              ;;SKIP IF CONTINUE
046366  000000                          HALT                    ;;HALT ON ERROR!
046370  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
046372                          3$:
046372  032777  001000  132540          BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
046400  001402                          BEQ     4$              ;;BR IF NO
046402  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
046406  005737  001214          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
046412  001402                          BEQ     5$              ;;BR IF NONE
046414  013716  001214                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
046420                          5$:
046420  105737  046432                  TSTB    IBSAVE          ;;SEE IF ITEM BYTE SAVE LOCATION HAS AN ERROR CALL
046424  001230                          BNE     7$              ;;BRANCH BACK TO CALL ORIGINAL ERROR
046426  000002                          RTI                     ;;RETURN
046430  000000                  CPSAVE: .WORD   0               ;;LOCATION TO SAVE CPU ERROR REG CONTENTS
046432  000000                  IBSAVE: .WORD   0               ;;LOCATION TO SAVE ITEM BYTE
```

```
                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

                                ;;*****************************************************************
                                ;*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.

046434                          $ERRTYP:
046434  104401  001223                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
046440  010046                          MOV     R0,-(SP)        ;;SAVE R0
046442  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
046444  153700  001114                  BISB    @#$ITEMB,R0
046450  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                ;;TYPE THE PC OF THE ERROR
046452  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                ;;ERROR ADDRESS
046456  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
046460  000456                          BR      10$             ;;GET OUT
046462  122700  000177          1$:     CMPB    #177,R0         ;;SEE IF THIS ERROR CALL IS SPECIAL POWER FAIL CALL
046466  001006                          BNE     1000$           ;;BRANCH IF NOT
046470  113737  001102  046772          MOVB    $TSTNM,PFTSTN   ;;GET TEST NUMBER
046476  012700  046632                  MOV     #PFECH,R0       ;;MOVE POWER FAIL ERROR CALL TABLE TO R0
046502  000406                          BR      1001$           ;;BRANCH TO CALL ERROR
046504  005300                  1000$:  DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
046506  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
046510  006300                          ASL     R0
046512  006300                          ASL     R0
046514  062700  003630                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
046520  012037  046530          1001$:  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
046524  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
046526  104401                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
046530  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
046532  104401  001223                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
046536  012037  046546          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
046542  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
046544  104401                          TYPE                    ;;TYPE THE "DATA HEADER"
046546  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
046550  104401  001223                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
046554  010146                  5$:     MOV     R1,-(SP)        ;;SAVE R1
046556  012001                          MOV     (R0)+,R1        ;;PICKUP "DATA TABLE" POINTER
046560  001415                          BEQ     9$              ;;BR IF NO DATA TO BE TYPED
046562  012000                          MOV     (R0)+,R0        ;;PICKUP "DATA FORMAT" POINTER
046564  105720                  6$:     TSTB    (R0)+           ;;"OCTAL" OR "DECIMAL"
046566  001003                          BNE     7$              ;;BR IF DECIMAL
046570  013146                          MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
046572  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
046574  000402                          BR      8$
046576                          7$:
046576  013146                          MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
046600  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
046602  005711                  8$:     TST     (R1)            ;;IS THERE ANOTHER NUMBER?
046604  001403                          BEQ     9$              ;;BR IF NO
046606  104401  046626                  TYPE    ,11$            ;;TYPE TWO(2) SPACES
046612  000764                          BR      6$              ;;LOOP

046614  012601                  9$:     MOV     (SP)+,R1        ;;RESTORE R1
046616  012600                  10$:    MOV     (SP)+,R0        ;;RESTORE R0
```

ERROR MESSAGE TYPEOUT ROUTINE

```
046620  104401  001223                  TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & 'LINE FEED''
046624  000207                          RTS     PC              ;;RETURN
046626     040     040     000  11$:     .ASCIZ  /  /            ;;TWO(2) SPACES
                                        .EVEN
046632  046642  046724  046756  PFECH:  PFECH1,PFECH2,PFECH3,PFECH4 ;;WORDS DEFINING TABLES BELOW
046642     120     117     127  PFECH1: .ASCIZ  ?POWER MONITOR BIT IN CPU ERROR REGISTER FOUND SET?
046724     124     105     123  PFECH2: .ASCIZ  ?TESTNO  ERR PC  CPUERREG?
                                        .EVEN
046756  046772  001116  046430  PFECH3: .WORD   PFTSTN,$ERRPC,CPSAVE,0
046766     000     000     000  PFECH4: .BYTE   0,0,0,0
046772  000000                  PFTSTN: .WORD   0               ;;CONTAINS TEST NUMBER FOR PF BIT ERROR
```

```
                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

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

046774  017646  000000        $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
047000  116637  000001  047217        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
047006  112637  047221              MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
047012  062716  000002              ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
047016  000406                      BR      $TYPON
047020  112737  000001  047217  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
047026  112737  000006  047221        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
047034  112737  000005  047216  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
047042  010346                      MOV     R3,-(SP)        ;;SAVE R3
047044  010446                      MOV     R4,-(SP)        ;;SAVE R4
047046  010546                      MOV     R5,-(SP)        ;;SAVE R5
047050  113704  047221              MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
047054  005404                      NEG     R4
047056  062704  000006              ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
047062  110437  047220              MOVB    R4,$OMODE       ;;SAVE IT FOR USE
047066  113704  047217              MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
047072  016605  000012              MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
047076  005003                      CLR     R3              ;;CLEAR THE OUTPUT WORD
047100  006105                1$:   ROL     R5              ;;ROTATE MSB INTO "C"
047102  000404                      BR      3$              ;;GO DO MSB
047104  006105                2$:   ROL     R5              ;;FORM THIS DIGIT
047106  006105                      ROL     R5
047110  006105                      ROL     R5
047112  010503                      MOV     R5,R3
047114  006103                3$:   ROL     R3              ;;GET LSB OF THIS DIGIT
047116  105337  047220              DECB    $OMODE          ;;TYPE THIS DIGIT?
047122  100016                      BPL     7$              ;;BR IF NO
047124  042703  177770              BIC     #177770,R3      ;;GET RID OF JUNK
047130  001002                      BNE     4$              ;;TEST FOR 0
047132  005704                      TST     R4              ;;SUPPRESS THIS 0?
047134  001403                      BEQ     5$              ;;BR IF YES
047136  005204                4$:   INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
```

BINARY TO OCTAL (ASCII) AND TYPE

```
047140  052703  000060              BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
047144  052703  000040      5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
047150  110337  047214              MOVB    R3,8$           ;;SAVE FOR TYPING
047154  104401  047214              TYPE    ,8$             ;;GO TYPE THIS DIGIT
047160  105337  047216      7$:     DECB    $OCNT           ;;COUNT BY 1
047164  003347                      BGT     2$              ;;BR IF MORE TO DO
047166  002402                      BLT     6$              ;;BR IF DONE
047170  005204                      INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
047172  000744                      BR      2$              ;;GO DO THE LAST DIGIT
047174  012605              6$:     MOV     (SP)+,R5        ;;RESTORE R5
047176  012604                      MOV     (SP)+,R4        ;;RESTORE R4
047200  012603                      MOV     (SP)+,R3        ;;RESTORE R3
047202  016666  000002  000004      MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
047210  012616                      MOV     (SP)+,(SP)
047212  000002                      RTI                     ;;RETURN
047214     000              8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
047215     000                      .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
047216     000              $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
047217     000              $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
047220  000000              $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                .SBTTL  TRAP DECODER

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

047222  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
047224  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
047230  005740                          TST     -(R0)           ;;BACKUP BY 2
047232  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
047234  006300                          ASL     R0              ;;POSITION FOR INDEXING
047236  016000  047256                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
047242  000200                          RTS     R0              ;;GO TO ROUTINE


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

047244  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
047246  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
047254  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
047256  047244                  $TRPAD: .WORD   $TRAP2
047260  044306                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
047262  047020                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
047264  046774                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
047266  047034                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
047270  044062                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

047272  045210                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

047274  045120                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
047276  045462                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
047300  045552                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
047302  045732                          $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
  2 047304  032036                      T.SCOP  ;;CALL=SCOP1    TRAP+13(104413) MY LOCAL SCOPES
  3 047306  032116                      CHECKT  ;;CALL=CHECKD   TRAP+14(104414) CHECK DVA,RDY,DPR,DRY
  4 047310  032434                      WAIT.T  ;;CALL=WAT      TRAP+15(104415) WAIT LOOP
```

```
      1                                 .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
047312  012737  047456  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
047320  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
047326  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
047330  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
047332  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
047334  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
047336  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
047340  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
047342  017746  131572                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
047346  010637  047462                  MOV     SP,$SAVR6       ;;SAVE SP
047352  012737  047364  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
047360  000000                          HALT
047362  000776                          BR      .-2             ;;HANG UP

                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
047364  012737  047456  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
047372  013706  047462                  MOV     $SAVR6,SP       ;;GET SP
047376  005037  047462                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
047402  005237  047462          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
047406  001375                          BNE     1$              ;;OF  WORD
047410  012677  131524                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
047414  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
047416  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
047420  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
047422  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
047424  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
047426  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
047430  012737  047312  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
047436  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
047444  104401                          TYPE                    ;REPORT THE POWER FAILURE
047446  047464          $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POINTER
047450  012716                          MOV     (PC)+,(SP)      ;;RESTART AT BEGIN
047452  004620          $PWRAD: .WORD   BEGIN                   ;;RESTART ADDRESS
047454  000002                          RTI
047456  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
047460  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
047462  000000          $SAVR6: 0                               ;;PUT THE SP HERE
047464     015     012     120  $POWER: .ASCIZ  <15><12>'POWER'
                                        .EVEN
```

ERROR AND MESSAGE TABLES

```
                                        .SBTTL  ERROR AND MESSAGE TABLES
     2
     3 047474  127  122  117  EM1:     .ASCIZ  /WRONG DATA IN READING OR WRITING HARDWARE REGISTER/
     4 047557  105  122  122  EM2:     .ASCIZ  /ERROR ON  DATA COMMAND/
     5 047606  105  122  122  EM6:     .ASCIZ  /ERROR ON WRITE HEADER AND DATA/
     6 047645  103  117  116  EM11:    .ASCIZ  /CONTROLLER OR DRIVE STATUS/
     7 047700  122  105  107  EM14:    .ASCIZ  /REGISTER FAILED/
     8 047720  116  117  116  EM15:    .ASCIZ  /NON EXISTENT REGISTER, PROGRAM ABORTED./
     9 047770  127  101  111  EM16:    .ASCIZ  /WAIT LOOP FAILED/
    10 050011  127  122  111  EM17:    .ASCIZ  /WRITE CHECK FAILING/
    11 050035  122  105  107  EM20:    .ASCIZ  /REGISTER FAILING/
    12 050056  111  11'  124  EM21:    .ASCIZ  /INTERRUPT FAILING/
    13 050100  105  12<  122  EM22:    .ASCII  /ERROR ON DRIVE PRESENT/<CRLF>
    14 050127  124  110  105            .ASCII  /THE UNIT NO'S FOUND BY SETTING RHAS/<CRLF>
    15 050173  104  117  040            .ASCII  /DO NOT AGREE WITH THE UNIT NO. FOUND FROM/<CRLF>
    16 050245  122  110  103            .ASCII  /RHCS2-'NED' BIT #12/<CRLF>
    17 050271  061  067  067            .ASCII  /177777-MEANS NO UNIT FOUND/<CRLF>
    18 050324  116  117  124            .ASCII  /NOTE: ON DUAL PORT SYSTEM, DRIVE ON OTHER PORT WILL NOT GIVE/<CRLF>
    19 050421  047  116  105            .ASCIZ  /'NED', HENCE THERE WILL BE AN EXTRA DRIVE/
    20 050473  114  117  117  EM24:    .ASCIZ  /LOOK AHEAD RFGISTER AT THE BEGINNING OF SECTOR IS IN ERROR/
    21 050566  114  117  117  EM25:    .ASCIZ  /LOOK AHEAD REGISTER IS IN ERROR/
    22 050626  103  125  122  EM30:    .ASCII  /CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER REGIHSTER/<CRLF>
    23 050721  101  106  124            .ASCIZ  /AFTER A SEEK AND INIT/
    24 050747  105  103  103  EM31:    .ASCII  /ECC GENERATED IS INCORRECT/<CRLF>
    25 051002  105  126  105            .ASCIZ  /EVERY WORD ON THIS SECTOR IS THAT GIVEN IN 'DATA USED''/
    26 051071  117  116  040  EM32:    .ASCII  /ON READ COMMAND, AFTER DATA AND ECC HAVE BEEN READ,/<CRLF>
    27 051155  105  103  103            .ASCII  /ECC REGISTERS OR RHER1 ARE IN ERROR/<CRLF>
    28 051221  117  116  114            .ASCII  /ONLY LOWER 11 BITS OF PATTERN REG. CAN BE READ/<CRLF>
    29 051300  124  110  111            .ASCIZ  /THIS SHOULD MATCH LOWER 11 BITS OF GOOD ECC1/
    30 051355  110  111  107  EM33:    .ASCIZ  /HIGH COUNT BIT NOT SET AFTER 38859 CLOCKS/
    31 051427  132  105  122  EM34:    .ASCIZ  /ZERO DETECT BIT NOT HIGH WHEN 32 BIT ECC REG. HAS 21 ZEROS/
    32 051522  120  117  123  EM35:    .ASCII  /POSITION REGISTER OR 11 BITS OF PATTERN REGISTER INCORRECT/<CRLF>
    33 051615  114  117  127            .ASCII  /LOWER 11 BITS OF PATTERN REGISTER SHOULD MATCH LOWER/<CRLF>
    34 051702  061  061  040            .ASCII  /11 BITS OF GOOD ECC1/<CRLF>
    35 051727  104  101  124            .ASCIZ  /DAT ENVLOP GOOD POSITION AND N-CODE ZEROS ARE IN OCTAL/
    36 052016  117  116  040  EM36:    .ASCII  /ON READ COMMAND WITH NON-CORRECTABLE ERROR DCK AND ECH SHOULD BE SET/<CRLF>
    37 052123  111  106  040            .ASCIZ  /IF POSITION REGISTER =10040 OR 10041 IT IS GOOD/
    38 052203  127  122  111  EM37:    .ASCIZ  /WRITING WITH BUS ADDRESS HIGHER THAN 28K CAUSED ERROR/
    39 052271  124  110  105  EM40:    .ASCII  /THERE WAS A READ-WRITE HEADER & DATA ERROR DURING 'DTE'/<CRLF>
    40 052361  124  105  123            .ASCIZ  /TEST SETUP - THE TEST WAS ABORTED AT THAT POINT/
```

ERROR AND MESSAGE TABLES

```
      1 052441      200      101      114  PREAMB: .ASCII  <CRLF>/ALL DCL'S UNDER TEST MUST BE LOCKED ON CORRECT PORT./
      2 052526      200      111      106          .ASCII  <CRLF>/IF CHANGES ARE REQUIRED ON PORT SWITCH, THEN A CYCLE/
      3 052613      200      125      120          .ASCII  <CRLF>/UP SEQUENCE IS REQUIRED FOR STROBING THE PORT SELECT/
      4 052700      200      106      114          .ASCII  <CRLF>/FLOP./
      5 052706      200                           .ASCII  <CRLF>
      6 052707      200      101      114          .ASCII  <CRLF>/ALL  DCL'S NOT  UNDER  TEST  MUST BE  SWITCHED  OFF./
      7 052774      200      111      106          .ASCII  <CRLF>/IF THIS IS NOT DONE, ERRORS WILL RESULT IN THE 'NED'/
      8 053061      200      124      105          .ASCIZ  <CRLF>/TESTS (T21 AND T36)./
      9
     10 053107      200      116      117  NDRVAS: .ASCII  <CRLF>/NO DRIVES PRESENT, RHAS=0/
     11 053141      200                           .ASCII  <CRLF>
     12 053142      200      127      122          .ASCII  <CRLF>/WRITTING ONES  INTO RHER1  FOR ALL UNIT NUMBERS DOES/
     13 053227      200      116      117          .ASCII  <CRLF>/NOT SET ANY BIT IN RHAS, SO ABORT PROGRAM./
     14 053302      200                           .ASCII  <CRLF>
     15 053303      200      124      117          .ASCII  <CRLF>/TO LOOP ON THIS TEST WITHOUT PRINTOUT,  SET SWITCHES/
     16 053370      200      061      063          .ASCIZ  <CRLF>/13, 8 AND 2./<CRLF>
     17
     18 053407      106      101      124  CPHALT: .ASCII  /FATAL ERROR - SEE DOCUMENT LISTING/<CRLF>
     19 053452      040      200      207          .ASCII  / /<CRLF><207><377><377><207><377><377><207><377><377>
     20 053465      124      110      105          .ASCII  /THE CONTROLLER OR DEVICE HAS GONE OFFLINE, LOST/<CRLF>
     21 053545      047      122      105          .ASCII  /'READY', BECOME UNAVAILABLE, OR HAS STATUS BITS/<CRLF>
     22 053625      127      110      111          .ASCIZ  /WHICH CANNOT BE CLEARED/
```

ERROR AND MESSAGE TABLES

```
 1 053655 120 103 040 DH1:   .ASCII /PC TEST REG.    GOOD    ACTUAL /<CRLF>
 2 053725 040 040 040        .ASCIZ /   NO ADDR.    DATA    DATA/
 3 053772 120 103 040 DH2:   .ASCII /PC TEST WORD    GOOD    BAD/<CRLF>
 4 054036 040 040 040        .ASCIZ /   NO NO       DATA    DATA/
 5 054103 120 103 040 DH11:  .ASCII /PC TEST FAILING CONT.   CONT.   CONT.   CONT./<CRLF>
 6 054171 040 040 040        .ASCIZ /   NO REG ADR  RHCS1   RHCS2   RHDS1   RHER1/
 7 054257 120 103 040 DH14:  .ASCII /PC TEST FAILING CONT.   CONT.   CONT.   CONT.   CONT.  /<CRLF>
 8 054356 040 040 040        .ASCIZ /   NO REG ADR  BAD REG RHCS1   RHCS2   RHDS1   RHER1  /
 9 054455 120 103 040 DH15:  .ASCIZ /PC TEST REG ADR/
10 054505 120 103 040 DH16:  .ASCII /PC TEST WAT    BIT     REG     REG/<CRLF>
11 054561 040 040 040        .ASCIZ /   NO PC       WANTED  ADDRESS CONTENTS/
12 054642 120 103 040 DH17:  .ASCII /PC TEST CONT.  CONT.   CONT.   CONT.   CONT./<CRLF>
13 054730 040 040 040        .ASCIZ /   NO RHBA     RHDB    RHWC    RHCS1   RHCS2/
14 055016 120 103 040 DH20:  .ASCII /PC TEST CONT   CONT    CONT    CONT    CONT/<CRLF>
15 055103 040 040 040        .ASCIZ /   NO RHER1    RHER2   RHER3   RHAS    RHDS1/
16 055171 120 103 040 DH21:  .ASCII /PC TEST CONT   CONT    CONT/<CRLF>
17 055236 040 040 040        .ASCIZ /   NO RHCS1    RHAS    RHDS1/
18 055304 120 103 040 DH22:  .ASCII /PC TEST RHAS   RHCS2/<CRLF>
19 055342 040 040 040        .ASCIZ /   NO UNIT     UNIT/
20 055377 120 103 040 DH24:  .ASCII /PC TEST RHDST  BAD     GOOD    SECTOR  SECTOR/<CRLF>
21 055466 040 040 040        .ASCIZ /   NO CONT.    RHLA    RHLA    NO      CLOCK/
22 055554 120 103 040 DH26:  .ASCII /PC TEST PC OF  FAILING CONT.   CONT.   CONT.   CONT./<CRLF>
23 055652 040 040 040        .ASCIZ /   NO JSR      REG ADD RHCS1   RHCS2   RHDS1   RHER1  /
24 055751 120 103 040 DH27:  .ASCII /PC TEST PC OF  FAILING GOOD    ACTUAL /<CRLF>
25 056031 040 040 040        .ASCIZ /   NO JSR      REG.    DATA    DATA/
26 056106 120 103 040 DH30:  .ASCII /PC TEST PC OF  REG.    GOOD    BAD/<CRLF>
27 056162 040 040 040        .ASCIZ /   NO JSR      ADDR    DATA    DATA/
28 056237 120 103 040 DH31:  .ASCII /PC TEST WORD   GOOD    BAD     CONT.   CONT.   CONT./<CRLF>
29 056335 040 040 040        .ASCIZ /   NO NO       DATA    DATA    RHCS1   RHDS1   RHER1  /
30 056434 120 103 040 DH32:  .ASCII /PC TEST PC OF  WORD    GOOD    BAD     CONT.   CONT.    CONT./<CRL
31 056542 040 040 040        .ASCIZ /   NO JSR      NO      DATA    DATA    RHCS1   RHDS1   RHER1/
32 056650 120 103 040 DH33:  .ASCII /PC TEST PC OF  WORD    GOOD    CONT.   CONT.   CONT.   /<CRLF>
33 056747 040 040 040        .ASCIZ /   NO JSR      NO      DATA    RHCS1   RHDS1   RHER1  /
34 057046 120 103 040 DH34:  .ASCII /PC TEST GOOD   GOOD    WRITTEN WRITTEN DATA/<CRLF>
35 057133 040 040 040        .ASCIZ /   NO ECC1     ECC2    ECC1    ECC2    USED/
36 057220 120 103 040 DH35:  .ASCII /PC TEST GOOD   GOOD    PATTERN POSITON GOOD    RHER1  /<CRLF>
37 057317 040 040 040        .ASCIZ /   NO ECC1     ECC2    REG.    REG.    POSITON REG./
38 057414 120 103 040 DH36:  .ASCII /PC TEST PC OF  RHMR    POSITON PATTERN/<CRLF>
39 057474 040 040 040        .ASCIZ /   NO JSR      CONT.   REG.    REG./
40 057551 120 103 040 DH37:  .ASCII /PC TEST POSITON POSITON GOOD    GOOD    PATTERN DATA    N-CODE/<CRL
41 057660 040 040 040        .ASCIZ /   NO ECC      GOOD    ECC1    ECC2    ECC     ENVLOPE ZEROS  /
42
43 057767 120 103 040 DH40:  .ASCII /PC TEST FAILING CONT.   CONT.   CONT.   CONT./<CRLF>
44 060055 040 040 040        .ASCIZ /   NO REG ADR  RHCS1   RHCS2   RHDS1   RHER1/
45                           .EVEN
```

ERROR AND MESSAGE TABLES

```
 1 060144  001116  003510  031620  DT1:    .WORD   $ERRPC,TSTNM,REGADR,$GDDAT,$BDDAT,0
 2 060160  001116  003610  037030  DT2:    .WORD   $ERRPC,TSTNM,ERWORD,$GDDAT,0
 3 060172  001116  003610  037030  DT3:    .WORD   $ERRPC,TSTNM,ERWORD,$GDDAT,$BDDAT,0
 4
 5 060206  001116  003610  001122  DT11:   .WORD   $ERRPC,TSTNM,$BDADR,CS1,CS2,DS1,ER1,0
 6 060226  001116  003610  001122  DT14:   .WORD   $ERRPC,TSTNM,$BDADR,$BDDAT,CS1,CS2,DS1,ER1,0
 7 060250  001116  003610  001200  DT15:   .WORD   $ERRPC,TSTNM,$TMP1,0
 8 060260  001116  003610  001204  DT16:   .WORD   $ERRPC,TSTNM,$TMP3,$TMP1,$TMP0,$BDDAT,0
 9 060276  001116  003610  001310  DT17:   .WORD   $ERRPC,TSTNM,BA,DB,WC,CS1,CS2,0
10
11 060316  001116  003610  001316  DT20:   .WORD   $ERRPC,TSTNM,ER1,ER2,ER3,AS,DS1,0
12 060336  001116  003610  001314  DT21:   .WORD   $ERRPC,TSTNM,CS1,AS,DS1,0
13 060352  001116  003610  001124  DT22:   .WORD   $ERRPC,TSTNM,$GDDAT,$BDDAT,0
14 060364  001116  003610  001320  DT24:   .WORD   $ERRPC,TSTNM,DST,$BDDAT,$TMP1,$TMP2,$TMP3,0
15 060404  001116  003610  001414  DT26:   .WORD   $ERRPC,TSTNM,PCJSR,$BDADR,CS1,CS2,DS1,ER1,0
16 060426  001116  003610  001414  DT27:   .WORD   $ERRPC,TSTNM,PCJSR,REGADR,$GDDAT,$BDDAT,0
17
18 060444  001116  003610  001414  DT30:   .WORD   $ERRPC,TSTNM,PCJSR,REGADR,$GDDAT,$BDDAT,0
19 060462  001116  003610  037030  DT31:   .WORD   $ERRPC,TSTNM,ERWORD,$GDDAT,$BDDAT,CS1,DS1,ER1,0
20 060504  001116  003610  001414  DT32:   .WORD   $ERRPC,TSTNM,PCJSR,ERWORD,$GDDAT,$BDDAT,CS1,DS1,ER1,0
21 060530  001116  003610  001414  DT33:   .WORD   $ERRPC,TSTNM,PCJSR,ERWORD,$GDDAT,CS1,DS1,ER1,0
22 060552  001116  003610  034556  DT34:   .WORD   $ERRPC,TSTNM,GECC1,GECC2,WECC1,WECC2,DISK,0
23 060572  001116  003610  034556  DT35:   .WORD   $ERRPC,TSTNM,GECC1,GECC2,EC2,EC1,POSITI,ER1,0
24 060614  001116  003610  001414  DT36:   .WORD   $ERRPC,TSTNM,PCJSR,MR,EC1,EC2,0
25 060632  001116  003610  001344  DT37:   .WORD   $ERRPC,TSTNM,EC1,POSITI,GECC1,GECC2,EC2,DATENV,ZCODE,0
26
27 060656  001116  003610  001122  DT40:   .WORD   $ERRPC,TSTNM,$BDADR,CS1,CS2,DS1,ER1,0
28
29 060676     000     000     000  DF1:    .BYTE   0,0,0,0,0
30 060703     000     000     001  DF2:    .BYTE   0,0,1,0
31 060707     000     000     001  DF3:    .BYTE   0,0,1,0,0
32
33 060714     000     000     000  DF11:   .BYTE   0,0,0,0,0,0,0
34 060723     000     000     000  DF14:   .BYTE   0,0,0,0,0,0,0,0
35 060733     000     000     000  DF15:   .BYTE   0,0,0
36 060736     000     000     000  DF16:   .BYTE   0,0,0,0,0
37 060743     000     000     000  DF17:   .BYTE   0,0,0,0,0,0,0
38 060752     000     000     000  DF20:   .BYTE   0,0,0,0,0,0,0
39
40 060761     000     000     000  DF21:   .BYTE   0,0,0,0,0
41 060766     000     000     000  DF22:   .BYTE   0,0,0,0
42 060772     000     000     000  DF24:   .BYTE   0,0,0,0,0,0,0
43 061001     000     000     000  DF26:   .BYTE   0,0,0,0,0,0,0,0
44 061011     000     000     000  DF27:   .BYTE   0,0,0,0,0,0
45 061017     000     000     000  DF30:   .BYTE   0,0,0,0,0,0
46
47 061025     000     000     001  DF31:   .BYTE   0,0,1,0,0,0,0,0
48 061035     000     000     000  DF32:   .BYTE   0,0,0,1,0,0,0,0,0
49 061046     000     000     000  DF33:   .BYTE   0,0,0,1,0,0,0,0
50 061056     000     000     000  DF34:   .BYTE   0,0,0,0,0,0,0
51 061065     000     000     000  DF35:   .BYTE   0,0,0,0,0,0,0,0
52 061075     000     000     000  DF36:   .BYTE   0,0,0,0,0,0
53 061103     000     000     000  DF37:   .BYTE   0,0,0,0,0,0,0,0,0
54 061114     000     000     000  DF40:   .BYTE   0,0,0,0,0,0,0
55                                 .EVEN
```

ERROR AND MESSAGE TABLES

```
 1 061124      122      110      061  EM42:   .ASCIZ  /RH11 REGISTER FAILED TO RESPOND TO A ''TST'' COMMAND/
 2 061207      122      110      061  EM43:   .ASCIZ  /RH11 ILLEGAL REGISTER RESPONSE TEST/
 3 061253      115      117      126  EM44:   .ASCIZ  /MOVE BYTE TO WORD COUNT TEST/
 4 061310      102      111      123  EM45:   .ASCIZ  /BIS BYTE TO WORD COUNT/
 5 061337      102      111      103  EM46:   .ASCIZ  /BIC BYTE TO WORD COUNT/
 6 061366      122      105      107  EM47:   .ASCIZ  /REGISTER ADDRESS SELECT TEST/
 7 061423      116      117      116  EM50:   .ASCIZ  /NON EXISTANT DRIVE (NED) TEST/
 8 061461      101      123      040  EM51:   .ASCIZ  /AS REGISTER TEST/
 9 061502      102      125      123  EM52:   .ASCIZ  /BUSS ADDRESS REGISTER DATA TEST/
10 061542      102      125      123  EM53:   .ASCIZ  /BUSS ADDRESS REGISTER MOVE BYTE/
11 061602      122      110      061  EM54:   .ASCIZ  /RH11 INTERRUPT FAILED TO OCCUR/
12 061641      122      105      123  EM55:   .ASCIZ  /RESET TEST/
13 061654      115      130      106  EM56:   .ASCIZ  /MXF BIT TEST/
14 061671      125      116      111  EM57:   .ASCIZ  /UNIBUS PARITY BIT TEST/
15 061720      104      117      105  EM60:   .ASCIZ  /DOES SELECTING THE RH11 CLEAR THE UNIT SELECT REGISTER/
16 062007      104      117      105  EM61:   .ASCIZ  /DOES TRE GET SET BY UNIBUS PARITY ERROR/
17 062057      116      117      116  EM62:   .ASCIZ  /NON EXISTANT DRIVE (NED) FAILED TO SET (TRE)/
18 062134      116      117      116  EM63:   .ASCIZ  /NON EXISTANT MEMORY (NEM) FAILED TO SET (TRE)/
19 062212      120      117      122  EM64:   .ASCIZ  /PORT SELECT FAILED TO CLEAR/
20 062246      103      117      116  EM65:   .ASCIZ  /CONTROLLER CLEAR FAILED TO CLEAR COMMAND REGISTER/
21 062330      122      105      123  EM66:   .ASCIZ  /RESELECT FAILED TO CLEAR COMMAND REGISTER/
22 062402      111      116      124  EM67:   .ASCIZ  /INTERRUPT CAUSED BY  RDY!IE BITS SET FAILED TO OCCUR/
23 062467      111      105      040  EM70:   .ASCIZ  /IE FAILED TO CLEAR FOLLOWING AN INTERRUPT/
24 062541      101      123      040  EM71:   .ASCIZ  /AS REGISTER WRITE TEST/
25
26 062570      120      103      040  DH42:   .ASCII  /PC      TEST    ILLEGAL/<CRLF>
27 062620      040      040      040          .ASCIZ  /        NO      ADDRESS/
28 062650      120      103      040  DH44:   .ASCII  /PC      TEST    REGISTR CONT    CONT/<CRLF>
29 062715      040      040      040          .ASCIZ  /        NO      ADDRESS GOOD    BAD/
30 062761      120      103      040  DH54:   .ASCII  /PC      TEST    REGISTR CONT/<CRLF>
31 063016      040      040      040          .ASCIZ  /        NO      ADDRESS/
32                                    .EVEN
33
34 063046   001116   003610   031620  DT42:   .WORD   $ERRPC,TSTNM,REGADR,0
35 063056   001116   003610   031620  DT44:   .WORD   $ERRPC,TSTNM,REGADR,$GDDAT,$BDDAT,0
36 063072   001116   003610   031620  DT54:   .WORD   $ERRPC,TSTNM,REGADR,$GDDAT,0
37
38 063104      000      000      000  DF42:   .BYTE   0,0,0
39 063107      000      000      000  DF44:   .BYTE   0,0,0,0,0
40 063114      000      000      000  DF54:   .BYTE   0,0,0,0
41
42          000200                    .END    200
```

SYMBOL TABLE

```
ABS    = 000200
ACL    = 000040
ACU    = 100000
ADTIMO  036436
AOE    = 001000
AS      001332
ATA    = 100000
ATABLE  003620
ATTENT  001416
AT0    = 000001
AT1    = 000002
AT2    = 000004
AT3    = 000010
AT4    = 000020
AT5    = 000040
AT6    = 000100
AT7    = 000200
A16    = 000400
A17    = 001000
BA      001310
BADTMO  004540
BAI    = 000010
BASECH  035506
BEGIN   004620
BEGIN2  004626
BITST   031622
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    031650
BLT2    031672
BLT3    031702
BPTVEC= 000014
CA      001326
CC      001352
CHECKD= 104414
CHECKE  032306
CHECKT  032116
CKSWR  = 104407
CLAREA  031772
CLDISK  032054
CLR    = 000040
COMHD   036550
COMPA   037622
COMPAR  032756
COMWHD  041672
COUNTD  040270
CPHALT  053407
CPSAVE  046430
CR     = 000015
CRC     033270
CRLF   = 000200
CSF    = 000002
CSU    = 000010
CS1     001314
CS2     001312
CYL     036710
DAREAD  036764
DATENV  034574
DAWORD  036770
DB      001304
DCK    = 100000
DCLEAR  001440
DDISP  = 177570
DENVL  = 000200
DE1    = 000040
DFF20  = 000002
DF1     060676
DF11    060714
DF14    060723
DF15    060733
DF16    060736
DF17    060743
DF2     060703
DF20    060752
DF21    060761
DF22    060766
DF24    060772
DF26    061001
DF27    061011
DF3     060707
DF30    061017
DF31    061025
DF32    061035
DF33    061046
DF34    061056
DF35    061065
DF36    061075
DF37    061103
DF40    061114
DF42    063104
DF44    063107
DF5    = 000001
DF54    063114
DH1     053655
DH11    054103
DH14    054257
DH15    054455
DH16    054505
DH17    054642
DH2     053772
DH20    055016
DH21    055171
DH22    055304
DH24    055377
DH26    055554
DH27    055751
DH30    056106
DH31    056237
DH32    056434
DH33    056650
DH34    057046
DH35    057220
DH36    057414
DH37    057551
DH40    057767
DH42    062570
DH44    062650
DH54    062761
DIGB   = 000004
DISK    040626
DISPLA  001142
DISPRE  000174
DLT    = 100000
DL64   = 000020
DMD    = 000001
DPR    = 000400
DRY    = 000200
DST     001320
DSWR   = 177570
DS1     001336
DT      001340
DTAGAP  041632
DTE    = 010000
DTSY   = 001000
DT1     060144
DT11    060206
DT14    060226
DT15    060250
DT16    060260
DT17    060276
DT2     060160
DT20    060316
DT21    060336
DT22    060352
DT24    060364
DT26    060404
DT27    060426
DT3     060172
DT30    060444
DT31    060462
DT32    060504
DT33    060530
DT34    060552
DT35    060572
DT36    060614
DT37    060632
DT40    060656
DT42    063046
DT44    063056
DT54    063072
DVA    = 004000
ECDATA  034554
ECH    = 000100
ECI    = 004000
ECORR   035200
ECTEST  034612
EC1     001344
EC2     001346
EMTVEC= 000030
EM1     047474
EM11    047645
EM14    047700
EM15    047720
EM16    047770
EM17    050011
EM2     047557
EM20    050035
EM21    050056
EM22    050100
EM24    050473
EM25    050566
EM30    050626
EM31    050747
EM32    051071
EM33    051355
EM34    051427
EM35    051522
EM36    052016
EM37    052203
EM40    052271
EM42    061024
EM43    061207
EM44    061253
EM45    061310
EM46    061337
EM47    061366
EM50    061423
EM51    061461
EM52    061502
EM53    061542
EM54    061602
EM55    061641
EM56    061654
EM57    061671
EM6     047606
EM60    061720
EM61    062007
EM62    062057
EM63    062134
EM64    062212
EM65    062246
EM66    062330
EM67    062402
EM70    062467
EM71    062541
ERCRC   042066
ERFLG$  001406
ERHDGP  042070
ERHEAD  042064
ERPOS   035176
ERR    = 040000
ERROR  = 104000
ERRVEC= 000004
ERSTAR  036522
ERUNIT  036520
ERWORD  037030
ER1     001316
ER2     001322
ER3     001330
EXT1   = 000001
EXT10  = 000010
EXT2   = 000002
EXT20  = 000020
EXT4   = 000004
EXT40  = 000040
E00     010272
E11     011676
E12     012024
E44     014540
FEN    = 000200
FER    = 000020
FILLEC  035352
FIRST   003612
FMT22  = 010000
FNWORD  042106
FORMAT  040272
FOUT    042112
FSYNER  042062
FUTABL  001432
GCRC    042056
GECC1   034556
GECC2   034560
GO     = 000001
GRV    = 000010
GTSWR  = 104406
HADTMP  034600
HARDER  034572
HCCRCE  033672
HCE    = 000200
HCI    = 002000
HCRC   = 000400
```

SYMBOL TABLE

HDESYN	042072	L00	010242	N14	012240	PIE32	= 000001	RHCA	001252
HDWSYN	040624	L02	010350	N36	013762	PIE4	= 010000	RHCC	001276
HEADER	040600	L03	010442	OCYL	= 100000	PIE5	= 004000	RHCS1	001240
HEDGAP	036724	L04	010540	OF	001324	PIE6	= 002000	RHCS2	001236
HEDSYN	036726	L05	010636	OFREV	= 000200	PIE7	= 001000	RHCS3	001300
HEGAP	040612	L06	010740	OFSETC	001464	PIE8	= 000400	RHDB	001230
HT	= 000011	L07	011036	OF100	= 000004	PIE9	= 000200	RHDST	001244
IAE	= 002000	L10	011172	OF200	= 000010	PIP	= 020000	RHDS1	001262
IBSAVE	046432	L11	011634	OF25	= 000001	PIRQ	= 177772	RHDT	001264
IE	= 000100	L12	011762	OF400	= 000020	PIRQVE	= 000240	RHEC1	001270
ILF	= 000001	L13	012074	OF50	= 000002	PKACK	001470	RHEC2	001272
ILLEGL	001474	L14	012212	OF800	= 000040	PLU	= 020000	RHER1	001242
ILR	= 000002	L15	012412	OPDCL	= 000100	POSITI	034570	RHER2	001246
IOTVEC	= 000020	L16	012552	OPERSE	031004	PRE	= 000020	RHER3	001254
IR	= 000100	L21	012744	OPI	= 020000	PREAMB	052441	RHLA	001274
IXE	= 004000	L22	013040	OR	= 000200	PROG	= 001000	RHMR	001260
I00	010246	L30	013266	OUT	036774	PR0	= 000000	RHOF	001250
I02	010352	L31	013366	O00	010274	PR1	= 000040	RHSN	001266
I03	010446	L34	013472	O10	011210	PR2	= 000100	RHWC	001232
I04	010546	L35	013606	O21	012772	PR3	= 000140	RH70	003614
I05	010646	L36	013722	O43	014412	PR4	= 000200	RH70CK	005246
I06	010746	L40	014036	O44	014516	PR5	= 000240	RKEY1	037020
I07	011046	L41	014140	PAR	= 000010	PR6	= 000300	RKEY2	037022
I14	012216	L42	014250	PAT	= 000020	PR7	= 000340	RMR	= 000004
I15	012420	L43	014364	PCJSR	001414	PS	= 177776	RNCTR1	041670
I16	012562	L44	014474	PFECH	046632	PSEL	= 002000	RNO	043226
I24	013156	L45	014616	PFECH1	046642	PSU	= 000001	RNWAT1	041750
JP1	016312	MAKECY	0344_4	PFECH2	046724	PSW	= 177776	RPTRP1	030316
JP2	016342	MASK	031614	PFECH3	046756	PUTREG	031554	RPTRP2	030422
JP3	016752	MCLK	= 000002	PFECH4	046766	PWRVEC	= 000024	RPVEC	001226
JP4	016766	MCPE	= 020000	PFTSTN	046772	P12	034604	RPVECT	043342
KEY1	036714	MHS	= 001000	PGE	= 002000	P22	03460c	RSETR	037016
KEY2	036716	MID	034424	PIE1	= 100000	P24	034610	RSYNC	037012
KIPAR0	= 172340	MIDDLE	034322	PIE10	= 000100	P3	034602	RUNCTR	036546
KIPAR1	= 172342	MINX	= 000004	PIE11	= 000040	RAW	= 000020	RUNWAT	036626
KIPAR2	= 172344	MMVEC	= 000250	PIE12	= 000020	RCOM	043230	R11	011616
KIPAR3	= 172346	MOL	= 010000	PIE13	= 000010	RCYL	037014	R15	012442
KIPAR4	= 172350	MPE	= 000400	PIE14	= 000004	RDCHR	= 104410	R16	012612
KIPAR5	= 172352	MR	001334	PIE15	= 000002	RDHEAD	037032	R44	014542
KIPAR6	= 172354	MRD	= 000020	PIE16	= 000001	RDLIN	= 104411	R6	=%000006
KIPAR7	= 172356	MSE	= 000020	PIE17	= 100000	RDOCT	= 104412	R7	=%000007
KIPDR0	= 172300	MSTCK	= 000010	PIE18	= 040000	RDY	= 000200	SAVDT	001410
KIPDR1	= 172302	MWR	= 000040	PIE19	= 020000	READ	037624	SAVER	032554
KIPDR2	= 172304	MXF	= 001000	PIE2	= 040000	READAT	001456	SAVSN	001412
KIPDR3	= 172306	NCODE	034564	PIE20	= 010000	READIN	001472	SC	= 100000
KIPDR4	= 172310	NCOUNT	034566	PIE21	= 004000	RECALI	001436	SCOPE	= 000004
KIPDR5	= 172312	NDRVAS	053107	PIE22	= 002000	REDATA	043232	SCOP1	= 104413
KIPDR6	= 172314	NED	= 010000	PIE23	= 001000	REFOR	001460	SCRC	042140
KIPDR7	= 172316	NEM	= 004000	PIE24	= 000400	REGADR	031620	SCYL	042130
LA	001350	NHS	= 002000	PIE25	= 000200	REGSAI	046106	SC1	= 000100
LAD	032034	NOPERA	001432	PIE26	= 000100	REINTO	002544	SC10	= 001000
LERR	031616	NOSYNC	037024	PIE27	= 000040	RELEAS	001442	SC2	= 000200
LF	= 000012	NOUNIT	001376	PIE28	= 000020	RESVEC	= 000010	SC20	= 002000
LST	= 002000	NOWORD	036756	PIE29	= 000010	RETCL	001466	SC4	= 000400
LST14A	012246	NUNIT	001400	PIE3	= 020000	RHAS	001256	SEARCH	042756
LST15A	012450	N11	011664	PIE30	= 000004	RHBA	001234	SECGAP	040530
LST16A	012620	N12	012012	PIE31	= 000002	RHBAE	001302	SECOTR	036712

SYMBOL TABLE

```
SECTR   042754        SW3    = 000010        TST34   013426        VAR3    016734        $CNTLG  045703
SEECOM  001462        SW4    = 000020        TST35   013532        VAR4    016742        $CNTLU  045676
SEGPER  042060        SW5    = 000040        TST36   013646        VAR5    016774        $CRLF   001223
SELECT  001402        SW6    = 000100        TST37   013772        VUF    = 000002       $DBLK   044276
SELTST  006742        SW7    = 000200        TST4    006216        VU30   = 010000       $DOAGN  030756
SERCH   001444        SW8    = 000400        TST40   014074        VV     = 000100       $DTBL   044266
SETCK1  015436        SW9    = 001000        TST41   014176        WAIT.T  032434        $ENDAD  030746
SETCK2  016132        S11      011612        TST42   014304        WAT    = 104415       $ENDCT  030714
SETCK3  016656        S12      011746        TST43   014420        WC       001306       $ENDMG  030765
SETDSK  033544        S36      013716        TST44   014550        WCE    = 040000       $ENULL  030762
SILOSZ  003616        TAGDTE   001430        TST45   014670        WCF    = 000040       $EOP    030660
SILOTB  040626        TBITVE=  000014        TST46   014724        WCRC     040610       $EOPCT  030706
SKEY1   042134        TDF    = 000040        TST47   015256        WCU    = 000001       $ERFLG  001103
SKEY2   042136        TESDTE   001426        TST5    006756        WCYL     042046       $ERMAX  001115
SKI    = 040000       TESTAD   031002        TST50   015750        WECC1    041626       $ERROR  046072
SN       001342       TIMCNT   032432        TST51   016474        WECC2    041630       $ERRPC  001116
SR0    = 177572       TIMOT  = 000004        TST52   017134        WKEY1    042052       $ERRTB  003630
SR1    = 177574       TKVEC  = 000060        TST53   017710        WKEY2    042054       $ERRTY  046434
SR2    = 177576       TMPILL   001422        TST54   020336        WLE    = 004000       $ERTTL  001112
SR3    = 172516       TOLGAP   041634        TST55   021054        WORD     037620       $ESCAP  001214
SSECTR  042132        TOTALA   001420        TST56   021600        WRCHDA   033120       $FILLC  001156
SSYN    036722        TPVEC  = 000064        TST57   022342        WRCHDT   001450       $FILLS  001155
SS1     017236        TRAPVE=  000034        TST6    007452        WRCHEK   001446       $GDADR  001120
SS10    017404        TRE    = 040000        TST60   022766        WRCHHD   032606       $GDDAT  001124
SS12    017434        TRK      036662        TST61   023504        WRDATA   040276       $GET42  030736
SS13    017442        TRK1   = 004000        TST62   024230        WRFROM   001500       $GTSWR  045210
SS14    017450        TRK10  = 040000        TST63   024772        WRHEAD   042142       $HD    = 000000
SS15    017510        TRK2   = 010000        TST64   025416        WRIDAT   001452       $HIOCT  046070
SS2     017674        TRK20  = 100000        TST65   026134        WRIFOR   001454       $ICNT   001104
SS3     017264        TRK4   = 020000        TST66   026670        WRITE    040056       $ILLUP  047456
SS4     017270        TRTVEC=  000014        TST67   027446        WRL    = 004000       $INTAG  001135
SS5     017274        TSECC    001424        TST7    010000        WRU    = 000400       $ITEMB  001114
SS7     017322        TSECCG   034562        TST70   030224        WSECTR   042050       $LF     001224
STACK  = 001100       TSTNM    003610        TST71   030336        WSSYNC   040576       $LPADR  001106
START   004634        TST1     005362        TST72   030432        WSU    = 000004       $LPERR  001110
STKLMT= 177774        TST10    010142        TUF    = 000100       WTRK     042004       $MNEW   045721
SWR     001140        TST11    010176        TY       037026       WWORD    040054       $MSWR   045710
SWREG   000176        TST12    010302        TYPDS  = 104405       X        036720       $MXCNT  044060
SW0    = 000001       TST13    010374        TYPE   = 104401       XE2      006414       $NULL   001154
SW00   = 000001       TST14    010472        TYPOC  = 104402       X11      011672       $NWTST= 000001
SW01   = 000002       TST15    010570        TYPON  = 104404       Y        036760       $OCNT   047216
SW02   = 000004       TST16    010672        TYPOS  = 104403       Y11      011266       $OMODE  047220
SW03   = 000010       TST17    010770        T.SCOP   032036       ZCODE    034576       $OVER   044044
SW04   = 000020       TST2     0C6066        UN       006134       ZEF    = 000400       $PASS   001100
SW05   = 000040       TST20    011072        UNIT     001374       ZWORDS   040274       $POWER  047464
SW06   = 000100       TST21    011216        UNITS    001354       $AUTOB   001134       $PWRAD  047452
SW07   = 000200       TST22    011700        UNITSL   001404       $BDADR   001122       $PWRDN  047312
SW08   = 000400       TST23    012026        UNLOAD   001434       $BDDAT   001126       $PWRMG  047446
SW09   = 001000       TST24    012146        UNS    = 040000       $BELL    001216       $PWRUP  047364
SW1    = 000002       TST25    012346        UPE    = 020000       $CHARC   044574       $QUES   001222
SW10   = 002000       TST26    012506        US1    = 000001       $CKSWR   045120       $RDCHR  045462
SW11   = 004000       TST27    012660        US2    = 000002       $CMTAG   001100       $RDLIN  045552
SW12   = 010000       TST3     006144        US4    = 000004       $CM1   = 000006       $RDOCT  045732
SW13   = 020000       TST30    013000        UWR    = 000010       $CM2   = 000014       $RDSZ  = 000011
SW14   = 040000       TST31    013076        U11      011570       $CM3   = 000006       $REGAD  001160
SW15   = 100000       TST32    013206        VAR1     016266       $CM4   = 000006       $REG0   001162
SW2    = 000004       TST33    013322        VAR2     016274       $CNTLC   045671       $REG1   001164
```

SYMBOL TABLE

```
$REG2   001166          $SVPC = 000214          $TKS    001144          $TPFLG  001157          $TYPEC  044456
$REG3   001170          $SWR  = 167770          $TKSRV  044670          $TPS    001150          $TYPEX  044576
$REG4   001172          $SWRMK= 000000          $TMP0   001176          $TRAP   047222          $TYPOC  047020
$REG5   001174          $TIMES  001212          $TMP1   001200          $TRAP2  047244          $TYPON  047034
$RTNAD  030760          $TKB    001146          $TMP2   001202          $TRP  = 000016          $TYPOS  046774
$SAVR6  047462          $TKCNT  044600          $TMP3   001204          $TRPAD  047256          $XOFF = 000023
$SCOPE  043422          $TKINT  044620          $TMP4   001206          $TSTNM  001102          $XON  = 000021
$SETUP= 000117          $TKQEN= 044617          $TMP5   001210          $TTYIN  045660          $XTSTR  043460
$SS1  = 000000          $TKQIN  044602          $TN   = 000073          $TYPDS  044062          $$GET4= 000000
$STUP = 177777          $TKQOU  044604          $TPB    001152          $TYPE   044306          $OFILL  047217
$SVLAD  044016          $TKQSR  044606

. ABS.  063120      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  56576 WORDS  ( 221 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  71 PAGES
CZRJHE.BIC,CZRJHE=CZRJHE.DOC,CZRJHE,[20,0]SYSMAC/M
```