

RP04/05/06

DISKLESS CONTROL PART 2
MD-11-DZRJH-A

EP DZRJH A DL A NOV 1976
COPYRIGHT 1976
FICHE 1 OF 2 MADE IN US

DZRJHA
SEQ

The image displays a grid of 100 small technical diagrams or data pages, arranged in 10 rows and 10 columns. Each cell contains a small-scale version of the technical content from the top-left cell, including text, tables, and diagrams. The diagrams appear to be related to the 'DZRJHA SEQ' header and likely represent various components or data points of a diskless control system. The overall layout is a dense, repeating pattern of these small technical elements.

RP04/05/06

DISKLESS CONTROL PART 2
MD-11-DZRJH-A

EP DZRJH A DL A NOV 1976
COPYRIGHT 1976
FICHE 2 OF 2 MADE IN US

This block contains a grid of 100 microfiche frames. Each frame displays a small, high-contrast image of a document page. The pages vary in content, including what appears to be technical specifications, diagrams, and text-heavy documents. The grid is organized into 10 rows and 10 columns. The right side of the microfiche card is blank.

11 12 13 14 15 16 17 18 19 20
21 22 23 24 25 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40
41 42 43 44 45 46 47 48 49 50
51 52 53 54 55 56 57 58 59 60
61 62 63 64 65 66 67 68 69 70
71 72 73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88 89 90
91 92 93 94 95 96 97 98 99 100

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZRJH-A-D
PRODUCT NAME: RPD4/5/6 DISKLESS CONTROLLER TEST-PART II
DATE CREATED: MAY 1976
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: PETE BLACKSTONE

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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

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

1. ABSTRACT
2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
 - 2.3 PRELIMINARY PROGRAMS
3. LOADING PROCEDURE
 - 3.1 METHOD
4. STARTING PROCEDURE
 - 4.1 CONTROL SWITCH SETTINGS
 - 4.2 STARTING ADDRESS OR ADDRESSES
 - 4.3 PROGRAM AND/OR OPERATOR ACTION
5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
 - 5.2 SUB-ROUTINE ABSTRACTS
6. ERRORS
 - 6.1 'FATAL' ERRORS
7. RESTRICTIONS
8. MISCELLANEOUS
 - 8.1 EXECUTION TIME
 - 8.2 STACK POINTER
 - 8.3 OPERATOR SELECTABLE SCOPE LOOPS
 - 8.4 PROGRAM REVISION HISTORY
9. PROGRAM DESCRIPTION

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 MDI. IT REQUIRES THAT THE DCL CABLE BE PLUGGED INTO THE MDI OR BE APPROPRIATELY 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 MDI, 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 SOFTWARE 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 210---FOR UNIT SELECTION

200 START
 ALL SWITCHES MUST BE DOWN FOR WORST CASE RUN. WITH THIS STARTING ADDRESS ALL THE RPO4/5/6'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.

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 RPO4/5/6 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN NEW ='

EACH TIME SWITCH SETTINGS ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING 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 APPROPRIATE 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 50.

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 RH.1 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 APPROPRIATE 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 MEANING 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 SUBSEQUENT 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 RPO4 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 CONTROL 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 SWITCH 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 EXPLANATION 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.

DOCUMENT

MNDEC-11-DZRJH-A,RP04/5/6 DSKLS CONTROLLER TST-PT 2

COPYRIGHT 1976
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

8

TABLE OF CONTENTS

40 OPERATIONAL SWITCH SETTINGS
56 BASIC DEFINITIONS
167 TRAP CATCHER
177 ACT11 HOOKS
188 STARTING ADDRESSES
203 MEMORY MANAGEMENT DEFINITIONS
240 COMMON TAGS
298 ERROR POINTER TABLE
2025 HARDWARE REGISTER BIT DEFINITIONS
2261 REGISTER ADDRESSES
2433
2434 ***PROGRAM SETUP & SETUP TESTS***
2435
2444 INITIALIZE THE COMMON TAGS
2539 GET VALUE FOR SOFTWARE SWITCH REGISTER
3034
3035 ***DIAGNOSTIC CODE***
3036
4107 EXTENDED MEMORY, DRIVE TIMING & SECTOR SELECT TESTS
4890 DATA TRANSFER TESTS USING ECC
7142 CURSORY INTERRUPT LOGIC TESTS
7265

TABLE OF CONTENTS

726E	***SUBROUTINES***
7267	
7272	END OF PASS ROUTINE
7393	SAVE REGISTERS ROUTINE
7420	FLOAT 1 AND 0
7461	CLEAR MEMORY ROUTINE
7496	LOCAL TRAPS
7513	CLEAR DISK ROUTINE
7526	CHECK DISK STATUS ROUTINES
7624	WAIT LOOP
7669	SAVE ROUTINE
7699	WRITE CHECK ROUTINE
7743	COMPARE ROUTINE
7795	WRITE CHECK DATA
7841	CRC GENERATION ROUTINE
7935	SIMULATED DISK SETUP
7982	CHECK HCE ROUTINE
8130	EXIT WRT HEADER & DATA ROUTINE
8164	JAM CURRENT CYLINDER ROUTINE
8203	ECC GENERATION AND COMPARISON ROUTINE
8433	ECC GENERATION CONTROL ROUTINE
8493	SOFTWARE DISK DATA ECC GEN. ROUTINE

TABLE OF CONTENTS

8535 RH BASE ADDRESS CHANGE ROUTINE
8615 DISK SIMULATION
9726
9727 ***SYSMAC LIBRARY ROUTINES***
9728
9732 SCOPE HANDLER ROUTINE
9806 CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
9873 TYPE ROUTINE
9943 TTY INPUT ROUTINE
10171 READ AN OCTAL NUMBER FROM THE TTY
10224 ERROR HANDLER ROUTINE
10272 ERROR MESSAGE TYPEOUT ROUTINE
10328 BINARY TO OCTAL (ASCII) AND TYPE
10405 TRAP DECODER
10421 TRAP TABLE
10444 POWER DOWN AND UP ROUTINES

3 COPYRIGHT (C) 1976
DIGITAL EQUIPMENT CORP.
MAYNARD, MASS. 01754

PROGRAM BY PETE BLACKSTONE

THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAIDEC SYSMAC
PACKAGE (MAIDEC-11-D20AC-C0), MAR 21, 1976.

29 *****

32 NOTE: ALL MACRO CALLS BEGINNING WITH ".S" ARE SUPPLIED FROM AN
EXTERNAL SYSMAC.SML PACKAGE WHICH MUST BE MADE AVAILABLE
TO THE SOURCE PROGRAM AT ASSEMBLY TIME.

40 *****
OPERATIONAL SWITCH SETTINGS

41

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

56 *****
BASIC DEFINITIONS

- 58 INITIAL ADDRESS OF THE STACK POINTER *** 1000 ***
- 63 MISCELLANEOUS DEFINITIONS
- 75 GENERAL PURPOSE REGISTER DEFINITIONS
- 87 PRIORITY LEVEL DEFINITIONS
- 97 "SWITCH REGISTER" SWITCH DEFINITIONS

125 DATA BIT DEFINITIONS (BIT00 TO BIT15)

153 BASIC "CPU" TRAP VECTOR ADDRESSES

167

TRAP CATCHER

170 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

177

ACT11 HOOKS

198

STARTING ADDRESSES

198 STARTING ADDRESS 200 FOR NORMAL STARTS
THIS WILL TEST ALL RPO4'S ON THE SYSTEM A SINGLE DRIVE AT A TIME
STARTING ADDRESS 210 WILL TEST ONLY ONE SPECIFIED DRIVE

203

MEMORY MANAGEMENT DEFINITIONS

205 KT11 VECTOR ADDRESS

209 KT11 STATUS REGISTER ADDRESSES

216 KERNEL "I" PAGE DESCRIPTOR REGISTERS

227 KERNEL "I" PAGE ADDRESS REGISTERS

243

COMMON TAGS

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

299

```

*****
ERROR POINTER TABLE
*****

```

```

300 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:

```

```

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

```

```

944 *****
*****

```

2025

```

*****
HARDWARE REGISTER BIT DEFINITIONS
*****

```

2261

```

*****
REGISTER ADDRESSES
*****

```

2433

```

*****
*****

```

2434

```

*****
***PROGRAM SETUP & SETUP TESTS***
*****

```

2435

```

*****
*****

```

2444

```

*****
INITIALIZE THE COMMON TAGS
*****

```


2539

 GET VALUE FOR SOFTWARE SWITCH REGISTER

2568 TEST 1 REFERENCE EACH REGISTER
 2570 REFERENCE EACH REGISTER BY A MOVE INSTRUCTION
 2613 TEST 2 RHCS2-CONTROL AND STATUS 2
 2615 THIS PARTIALLY TESTS RHCS2 TO ENABLE DETERMINATION
 OF THE NUMBER OF DRIVES PRESENT
 2624 CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70
 2644 TEST 3 PARTIAL TEST OF RHAS FOR UNIT NUMBERS PRESENT
 2662 TEST 4 TEST FOR DRIVES PRESENT USING RHAS AND RHCS2
 2718 SET UP UNITS TABLE
 2754 NO...IT'S NOT AN RPO4/RPO5/RPO6 DEVICE
 SO TYPE OUT THE DEVICE TYPE
 2795 TEST 5 TYPE SERIAL NUMBER AND DRIVE TYPE
 2797 READ SERIAL NUMBER REGISTER AND DRIVE TYPE REGISTER
 TYPE IT OUT AND PROCEED
 2800 TO LOOP HERE SET SWITCH 8 AND THIS TEST NO AND RESTART
 2810 TEST FOR UNIT #0
 2821 INCREMENT THE UNITS TABLE TO NEXT DRIVE (IF ANY)
 & DECREMENT THE "NOJNITS" PRESENT (TO BE TESTED)
 2920 TEST 6 CHECK MOL TO BE LOW
 2922 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
 2957 TEST 7 PACK ACKNOWLEDGE COMMAND TEST
 2959 THE PACK ACKNOWLEDGE COMMAND WILL BE LOADED INTO RHCS1 WITH GO
 THEN ALL REGISTERS WILL BE CHECKED
 RH CLEAR WILL BE GIVEN
 THEN ALL REGISTERS WILL BE CHECKED

3022 TEST 10 MAKE CURRENT CYLINDER = 0

3034

3035

DIAGNOSTIC CODE

3036

3047 TEST 11 BCTA LEGAL REGISTER RESPONSE TEST

3089 TEST 12 BCTA MOVB LO BYTE TO WC

3113 TEST 13 BCTA MOVB HI BYTE TO WC

3143 TEST 14 BCTA BISB LO BYTE TO WC

3170 TEST 15 BCTA BISB HI-BYTE TO WC

3201 TEST 16 BCTA BICB LO-BYTE TO WC

3229 TEST 17 BCTA BICB HI- BYTE TO WC

3272 TEST 20 BCTA ILLEGAL REGISTER TEST

3314 TEST 21 BCTB (NED) NON-EXISTANT DRIVE TEST. (SET)

3384 TEST 22 BCTB (NED) NON-EXISTANT DRIVE TEST. (CLEARED)

3428 TEST 23 BCTB AS REGISTER TEST

3467 TEST 24 BCTC BUS ADDRESS REGISTER

3527 TEST 25 BCTC BUS ADDRESS REGISTER LO-BYTE

3574 TEST 26 BCTC BUS ADDRESS REGISTER HI-BYTE

3633 TEST 27 RH11 INTERRUPT TEST

3657 TEST 30 BCTD CLR L TEST

3694 TEST 31 MXF TEST

3700 CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70

- 3729 TEST 32 CSRB UNIBUS PARITY ERROR SET TEST
- 3735 CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70
- 3768 TEST 33 CSRB UNIBUS PARITY ERROR CLEAR TEST
- 3901 TEST 34 CSRB UNIT SELECT CLEAR TEST
- 3834 TEST 35 CSRB TRANSFER ERROR (TRE) - UPE
- 3940 CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70
- 3872 TEST 36 CSRB TRANSFER ERROR (TRE) NED
- 3911 TEST 37 CSRA PSEL CLEAR TEST
- 3940 TEST 40 CSRB COMMAND REGISTER CLEAR TEST
- 3973 TEST 41 CSRB COMMAND REGISTER RESELECT CLEAR TEST
- 4004 TEST 42 CSRB READY AND IE INTERRUPT TEST
- 4036 TEST 43 CSRB BOES "IE" CLEAR AFTER AN INTERRUPT
- 4075 TEST 44 BCTB "AS" WRITE TEST

4107

 EXTENDED MEMORY, DRIVE TIMING & SECTOR SELECT TESTS

- 4111 TEST 45 MAKE CURRENT CYLINDER = 0
- 4129 TEST 46 RHCSI - BITS 8 AND 9 - EXTENDED ADDR (A16 & A 17)
- 4131 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
- 4142 CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70
- 4224 TEST 47 DRIVE TIMING ERROR
- 4226 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.
 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'

Use

4240 THESE ARE TO SETUP FOR DISKLESS USE

4253 THESE ARE REGULAR SETUPS

4269 READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'

4277 NOW 'GO' WILL BE GIVEN, EVERYTHING WILL BE TREATED NORMALLY FOR THE HEADER. BUT WHEN IT IS TIME TO READ DATA, ONLY SECTOR CLOCKS WILL BE GIVEN. NO DIAGNOSTIC DATA CLOCKS WILL BE GIVEN. THIS SHOULD BRING SECTOR PULSE HIGH WITHOUT PUTTING "READ" DOWN HENCE 'DTE' WILL COME UP.

4300 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

4304 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.

4309 THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC DATA CLOCKS
WHICH EQUALS 3 BYTES OF DATA

4325 THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS

4334 NOW 'DTE' SHOULD BE SET
CHANGE SAVED REGISTERS TO EXPECTED VALUES

4348 NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS
CAN BE DONE (USE THE 'WRFROM' SAVE BUFFER THIS TIME)

4356 FOR RHAS UPPER BYTE

4359 COMPARE THE HEADER READ

4376 COMPARE REGISTERS BEFORE COMMAND WITH REGISTERS AFTER COMMAND

4399 TEST 50 DRIVE TIMING ERROR

4401 A WRITE DATA COMMAND IS STARTED ON CYLINDER 0, SECTOR
0, TRACK 0, 256 WORDS.

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'

4417 THESE ARE TO SETUP FOR DISKLESS USE

4430 THESE ARE REGULAR SETUPS & CHECKS

4446 READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'

4453 NOW 'GO' WILL BE GIVEN, EVERYTHING WILL BE TREATED NORMALLY FOR THE HEADER. WHEN IT IS TIME TO WRITE DATA, ONLY SECTOR CLOCKS WILL BE GIVEN, NO DIAGNOSTIC DATA CLOCKS WILL BE GIVEN. THIS SHOULD BRING SECTOR PULSE HIGH WITHOUT PUTTING READ DOWN, HENCE 'DTE' WILL COME UP.

4477 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

4481 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

4486 THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC DATA CLOCKS

4501 THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS

4511 ECC PATTERN REGISTER IS NOT CHECKED

4515 NOW 'DTE' SHOULD BE SET, CHANGE SAVED REGISTERS TO EXPECTED VALUES

4538 NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS

4546 FOR RHAS UPPER BYTE

4550 COMPARE CHANGED REGISTER SNAPSHOT BEFORE COMMAND WITH
SNAPSHOT AFTER COMMAND

4574 TEST 51 DRIVE TIMING ERROR

4576 A WRITE HEADER AND DATA COMMAND IS GIVEN
TO CYLINDER 0, SECTOR 0, TRACK 0, 256. WORDS.

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'

4588 THESE ARE TO SET UP FOR DISKLESS USE ONLY

4600 THESE ARE REGULAR SETUPS & CHECKS

4616 READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'

4623 NOW 'GO' WILL BE GIVEN. EVERYTHING WILL BE TREATED
NORMALLY TILL HEADER IS TO BE GIVEN, THEN ONLY
SECTOR CLOCKS WILL BE GIVEN. NO DIAGNOSTIC DATA
CLOCKS WILL BE GIVEN. THIS SHOULD BRING SECTOR PULSE HIGH
WITHOUT PUTTING "READ" DOWN, HENCE 'DTE' WILL COME UP.

4647 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

4655 THIS GIVES 609 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS

4663 NOW 'DTE' SHOULD BE SET, CHANGE SAVED REGISTERS

4686 NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN BE DONE

4693 FOR RHAS UPPER BYTE

4696 COMPARE CHANGED REGISTER SNAPSHOT BEFORE COMMAND
WITH REGISTER SNAPSHOT AFTER COMMAND

4719 TEST 52 SECTOR SELECTION

4721 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

4740 THE FOLLOWING INITIALIZES FOR SECTOR 0

4752 CLEAR SIMULATED DISK AREA

4761 SETUP WRITE FROM BUFFER

4764 HEADER

4771 DATA IN WRITE FROM BUFFER ALTHOUGH THIS IS DATA AND NOT

4772 HEADER, THE SECTOR WITH SYNC BYTES WILL BE GIVEN AS DATA.



4774 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

4811 CLEAR REST OF 256 WORDS THAT IS 54 WORDS OF ZEROS

4818 THESE ARE TO BE SET UP FOR DISKLESS USE ONLY

4830 THESE ARE REGULAR SETUPS

4857 NOW COMPARE "DISK" BUFFER WITH "REINTO" BUFFER

4870 THE FOLLOWING INCREMENTS ARE TO CHANGE THE ABOVE SET UP
TO WRITE ON THE NEXT SECTOR

4890

DATA TRANSFER TESTS USING ECC

4895 TEST 53 WRITE ECC TEST 1

4897 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 ZEROS.

4914 THESE ARE FOR ECC TEST ONLY

4928 THESE ARE TO BE SETUP FOR DISKLESS USE ONLY

4940 THESE ARE REGULAR SETUPS

4969 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

4980 COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE

4998 FILL "REINTO" BUFFER WITH EXPECTED DATA

5016 NOW COMPARE "DISK" BUFFER WITH "REINTO"

5045 TEST 54 READ ECC ENABLED 1A

5047 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

5082 THESE ARE FOR ECC TEST ONLY

5094 THESE ARE TO SETUP FOR DISKLESS USE ONLY

5111 THESE ARE REGULAR SETUPS

5137 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 SYNC'S 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

5165 ADD 16 MAINTENANCE CLOCKS TO
BRING EBL DOWN

5195 NOW READ DATA BUFFER WILL BE CHECKED

5218 TEST 55 READ ECC ENABLED 18

5220 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

5234 SETUP FOR WHAT IS TO BE READ
HEADER CRC IS RESTORED FROM A SUBROUTINE

5256 THESE ARE FOR ECC TEST ONLY

5268 THESE ARE TO SETUP FOR DISKLESS USE ONLY

5283 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

5294 THESE ARE REGULAR SETUPS

5320 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 SYNC'S WERE CORRECTLY

5324 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

5368 ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
NOW THE INSERTED ERROR WILL BE PUT IN

5378 NOW READ DATA BUFFER WILL BE CHECKED

5398 TEST 56 READ ECC ENABLED 1C

5400 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

5435 THESE ARE FOR ECC TEST ONLY

5447 THESE ARE TO SETUP FOR DISKLESS USE ONLY

5462 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

5472 THESE ARE REGULAR SETUPS

5496 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 SYNC'S 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

5555 NOW THE INSERTED ERROR WILL BE PUT IN

5565 NOW READ DATA BUFFER WILL BE CHECKED

5590 TEST 57 WRITE ECC TEST 2

5592 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.

5609 THESE ARE FOR ECC TEST ONLY

5623 THESE ARE TO BE SETUP FOR DISKLESS USE ONLY

5635 THESE ARE REGULAR SETUPS

5664 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

5675 COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE

5694 FILL "REINTO" BUFFER WITH EXPECTED DATA

5712 NOW COMPARE "DISK" BUFFER WITH "REINTO"

5741 TEST 60 READ ECC ENABLED 2A

5743 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

5778 THESE ARE FOR ECC TEST ONLY

5790 THESE ARE TO SETUP FOR DISKLESS USE ONLY

5807 THESE ARE REGULAR SETUPS

5833 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 SYNC'S 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

5861 ADD 16 MAINTENANCE CLOCKS TO
BRING EBL DOWN

5911 TEST 61 READ ECC ENABLED 2B

5913 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

5927 SETUP FOR WHAT IS TO BE READ
HEADER CRC IS RESTORED FROM A SUBROUTINE

5949 THESE ARE FOR ECC TEST ONLY

5961 THESE ARE TO SETUP FOR DISKLESS USE ONLY

5976 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

5987 THESE ARE REGULAR SETUPS

6011 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 SYNC'S WERE CORRECTLY
DETECTED
HEADER AND DATA ARE TO BE CHECKED.

6017 IN CHECKING READ DATA THE WRITE FROM BUFFER
'WRFROM' IS FILLED WITH EXPECTED DATA AND
COMPARISONS ARE MADE

6059 ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
NOW THE INSERTED ERROR WILL BE PUT IN

6067 NOW READ DATA BUFFER WILL BE CHECKED

6087 TEST 62 READ ECC ENABLED 20

6099 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 17777?
COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA

6124 THESE ARE FOR ECC TEST ONLY

6136 THESE ARE TO SETUP FOR DISKLESS USE ONLY

6151 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

6162 THESE ARE REGULAR SETUPS

6189 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

6193 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

6247 ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
NOW THE INSERTED ERROR WILL BE PUT IN

6255 NOW READ DATA BUFFER WILL BE CHECKED

6274 TEST 63 WRITE ECC TEST 3

6276 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.

6293 THESE ARE FOR ECC TEST ONLY

6307 THESE ARE TO BE SETUP FOR DISKLESS USE ONLY

6319 THESE ARE REGULAR SETUPS

6348 IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
FROM THE "COMHD" ROUTINE THAT MEANS ALL HEADER ON DISK
IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
ALL 52525 AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
THEY ARE ALL ZEROS

6377 FILL "REINTO" BUFFER WITH EXPECTED DATA

6395 NOW COMPARE "DISK" BUFFER WITH "REINTO"

6419 TEST 64 READ ECC ENABLED 3A

6421 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

6434 SETUP FOR WHAT IS TO BE READ
HEADER CRC IS RESTORED FROM A SUBROUTINE

6454 THESE ARE FOR ECC TEST ONLY

6466 THESE ARE TO SETUP FOR DISKLESS USE ONLY

6493 THESE ARE REGULAR SETUPS

6509 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 SYNC'S 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

6537 ADD 16 MAINTENANCE CLOCKS TO
BRING EBL DOWN

6563 NOW READ DATA BUFFER WILL BE CHECKED

6582 TEST 65 READ ECC ENABLED 38

6584 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

6597 SETUP FOR WHAT IS TO BE READ
HEADER CRC IS RESTORED FROM A SUBROUTINE

6619 THESE ARE FOR ECC TEST ONLY

6631 THESE ARE TO SETUP FOR DISKLESS USE ONLY

6646 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
THIS CHANGES THE LAST BIT OF THE ECC

6663 THESE ARE REGULAR SETUPS

6689 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 SYNC'S 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

- 6737 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
- 6746 NOW READ DATA BUFFER WILL BE CHECKED
- 6766 TEST 66 READ ECC ENABLED 3C
- 6769 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
- 6782 SETUP FOR WHAT IS TO BE READ
HEADER CRC IS RESTORED FROM A SUBROUTINE
- 6803 THESE ARE FOR ECC TEST ONLY
- 6815 THESE ARE TO SETUP FOR DISKLESS USE ONLY
- 6830 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
- 6842 THESE ARE REGULAR SETUPS
- 6856 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 SYNC'S WERE CORRECTLY
DETECTED
HEADER AND DATA ARE TO BE CHECKED.
- 6872 IN CHECKING READ DATA THE WRITE FROM BUFFER
'WRFROM' IS FILLED WITH EXPECTED DATA AND
COMPARISONS ARE MADE
- 6924 ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
NOW THE INSERTED ERROR WILL BE PUT IN
- 6936 NOW READ DATA BUFFER WILL BE CHECKED
- 6956 TEST 67 READ ECC ENABLED 3D

6958 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

6973 SETUP FOR WHAT IS TO BE READ
 HEADER CRC IS RESTORED FROM A SUBROUTINE

6994 THESE ARE FOR ECC TEST ONLY

7006 THESE ARE TO SETUP FOR DISKLESS USE ONLY

7021 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

7031 THESE ARE REGULAR SETUPS

7057 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 SYNC'S WERE CORRECTLY
 DETECTED

7062 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

7116 ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
 NOW THE INSERTED ERROR WILL BE PUT IN

7125 NOW READ DATA BUFFER WILL BE CHECKED

7142

 CURSORY INTERRUPT LOGIC TESTS

7147 TEST 70 PROGRAM INTERRUPT

7149 PROGRAM INTERRUPT IS TESTED BY SETTING R0Y AND IE
 IN RHCS1 AT THE SAME TIME
 THIS SHOULD INTERRUPT THROUGH LOCATION 254
 THE PROCESSOR PRIORITY IS SET TO 4

7184 TEST 71 INTERRUPT AT PROCESSOR AND DISK PRIORITY SAME

7186 PROCESSOR PRIORITY IS SET AT 5 (SAME AS THE DISK)
IE AND RDY IS SET. THIS SHOULD NOT INTERRUPT

7219 TEST 72 END OF DRIVE

7220 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

7265

7266

SUBROUTINES

7267

7272

END OF PASS ROUTINE

7275 INCREMENT THE PASS NUMBER (\$PASS)
TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
IF THERES A MONITOR GO TO IT
IF THERE ISN'T JUMP TO TST1

7316 HERE IS A DETAILED EXPLANATION OF HOW THE LOOP ON ERROR WORKS.
ON HITTING AN ERROR IF THE LOOP ON ERROR SWITCH IS SET, THE

29

7318 PROGRAM GOES BACK - USUALLY BACK TO THE BEGINNING OF THE TEST.

7320 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.

7393

SAVE REGISTERS ROUTINE

7420

FLOAT 1 AND 0

7423 FLOAT A ONE AND A ZERO THRU A DESIGNATED REGISTER
 ABSOLUTE ADDRESS OF REG. UNDER TEST IS IN R4

7461

CLEAR MEMORY ROUTINE

7465 THIS CLEARS ANY BLOCK OF MEMORY
 FILLING IT WITH ANY DATA

 CALL
 JSR R0,CLAREA
 X ;STARTING ADDRESS OF BLOCK
 Y
 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

7496

LOCAL TRAPS

7504 EXAMPLE OF THE USE OF THE ABOVE
 THIS WILL LOOP BETWEEN X: AND SCOP1 PROVIDED THERE IS NO "NEWTST"

MOV #X, J#LAD
X: --- ---
 --- ---
 --- ---
 SCOP1

7513 *****
CLEAR DISK ROUTINE

7526 *****
CHECK DISK STATUS ROUTINES

7529 THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCSI = 1
AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1
IT ALSO CHECKS THAT ALL OTHER BITS IN THESE REGISTERS = 0

7587 THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCSI = 1
AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1
IT DOES NOT CHECK THAT OTHER BITS IN THESE REGISTERS = 0

7524 *****
WAIT LOOP

7526 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)

7559 CALL FOR THE ABOVE WAITLOOP IS

```
MOV    JA, JXS      ;A CONTAINS REGISTER ADDRESS
-      -            ;HENCE XS WILL HAVE ABSOLUTE REG. ADR.
-      -            -
-      -            -
XS:    WAT          ;ABSOLUTE REG. ADDRESS UNDER WAIT
      0             ;BIT WAITED FOR
      .WORD 0       ;CONTINUE
```

7569 *****
SAVE ROUTINE

7699 *****
WRITE CHECK ROUTINE

7742 *****
COMPARE ROUTINE

7746 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

7795 *****
WRITE CHECK DATA

7841 *****
CRC GENERATION ROUTINE

7935 *****
SIMULATED DISK SETUP

7982 *****
CHECK HCE ROUTINE

8130 *****
EXIT WRT HEADER & DATA ROUTINE

8164 *****
JAM CURRENT CYLINDER ROUTINE

8167 THIS SUBROUTINE WILL CHANGE THE CURRENT CYLINDER REGISTER
THIS IS DONE BY GIVING A SEEK COMMAND THEN AN INIT
WHICH WILL LOAD THE CURRENT CYLINDER WITH THE DESIRED CYLINDER VALUE

CALL IS:
 JSR RD, \$MAKECYL
 XC ; DESIRED VALUE OF CURRENT CYLINDER.

3

8203 *****
ECC GENERATION AND COMPARISON ROUTINE

8206 THIS SUBROUTINE GENERATES AND TESTS ECC
CALL JSR PC,ECTEST

8386 CHECK HARDWARE

8433 *****
ECC GENERATION CONTROL ROUTINE

8436 THIS SUBROUTINE WILL CONTROL THE ECC GENERATION ROUTINE
FOR ERROR CORRECTION PROCESS
CALL JSR, PC, @ECCORR
XP ;EXPECTED POSITION REGISTER WHEN CORRECTION IS COMPLETE

8493 *****
SOFTWARE DISK DATA ECC GEN. ROUTINE

8496 THIS SUBROUTINE GENERATES THE ECC FOR WHAT IS ON DISK AND INSERTS THEM
ON LOCATIONS "DISK+1000" AND "DISK+1002"

8535 *****
RH BASE ADDRESS CHANGE ROUTINE

8538 THIS ROUTINE WILL ALLOW THE CHANGE OF THE BASE
ADDRESS FROM 176700 TO ANY TYPED VALUE

8603 THIS IS A LITTLE ROUTINE THAT TESTS NED BIT 11 IN RHCS2
THIS LOOPS HERE FOR EVER
TO BE USED ONLY IF DRIVES PRESENT LOOKING AT NED DOES NOT AGREE
WITH WHAT IS REALY THERE

8615 *****
DISK SIMULATION

8618 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

8669

8677 WRITE DATA COMMAND
OR READ COMMAND, I.E DATA ONLY, OR HEADER AND DATA

8685 THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES

8687 IT ISSUES DIAGNOSTIC MODE, AND EXTRA DIAGNOSTIC INDEX, AND THE
'GO' BIT

8690 IT THEN CALLS THREE OTHER SUBROUTINES, WHICH IN TURN CALL
OTHER SUBROUTINES. THE SUBROUTINES CALLED HERE ARE:

8693 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)

8810 THE DISK SECTOR IS DEVIDED AS FOLLOWS

8812 19 WORDS OF 0, ONE WORD 144000
THESE MAKE 39 BYTES FOR SECTOR GAP AND ONE SYNC. BYTE

8823 5 WORDS OF 0'S, ONE WORD 144000
THESE MAKE 11 BYTES FOR HEADER GAP AND ONE SYNC. BYTE
THESE ARE DCL GENERATED

8829 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

8840 READ DISK HEADER

8912 CONTINUE WITH DIAGNOSTIC WRITE COMMAND

8951 READ COMMAND START FROM HERE

8983 READ ONE WORD IN "WORD"

9051 WRITE ONE WORD WHICH COMES BACK IN "WWORD"

9121 WRITE DATA HOUSEKEEPING ROUTINE

9196 THIS IS THE SIMULATED DISK
ONLY ONE SECTOR OF SPACE IS ALLOCATED

9226 WRITE HEADER AND DATA

9233 THIS SUBROUTINE IS THE FIRST IN A SERIES OF NESTED SUBROUTINES

9235 IT ISSUES DIAGNOSTIC MODE, AN EXTRA DIAGNOSTIC INDEX, AND THE
'GO' BIT

9238 IT THEN CALLS THREE OTHER SUBROUTINES, WHICH IN TURN CALL OTHER
SUBROUTINES. THE THREE SUBROUTINES CALLED HERE ARE:

9241 SEARCH ;ISSUES SECTOR CLOCKS TO SET SECTOR FOUND FLOP
WRHEAD ;WRITES THE SECTOR HEADER
WRDATA ;WRITES THE ACTUAL SECTOR DATA

- 9245 ALL OF THE ABOVE MENTIONED "WRITING" IS ACTUALLY DONE INTO A CORE BUFFER AREA CALLED 'DISK' VIA THE MAINTENANCE REGISTER (RHMR)
- 9393 WRITE HEADER
- 9400 R0 = MAINT. REG.
R1 = SIMULATED DISK
R2 = BYTE COUNT
R3 = WRITE WORD
R5 = WORD COUNT
- 9563 SEARCH SECTOR
- 9571 R0=RHMR ADDRESS
R1=PASSED ARGUMENT (SECTOR SEARCHED FOR)
R2=CLOCK COUNT (PER BYTE)
R3=SECTOR COUNTER FROM R1
R5=BYTES PER WORD COUNT
- 9577 BEFORE INDEX IS GIVEN TWO SECTOR CLOCKS ARE GIVEN TO RESET SECTOR PULSE IN CASE IT IS SET.
- 9580 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:
- 9585 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
- 9623 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
- 9683 READ ONE SECTOR OF DATA

9726

9727

SYSMAC LIBRARY ROUTINES

9728

9732

SCOPE HANDLER ROUTINE

9735 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

9806

CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

9809 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

9873

TYPE ROUTINE

9876 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

9943

TTY INPUT ROUTINE

9954 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

9971 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 (^PERSEL)

10023 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.

10038 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.

10109 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

10133 THIS ROUTINE WILL INPUT A STRING FROM THE TTY

CALL:

RDLIN
RETURN HERE

::INPUT A STRING FROM THE TTY
::ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
::TERMINATOR WILL BE A BYTE OF ALL 0'S

10171

READ AN OCTAL NUMBER FROM THE TTY

10174 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
RETURN HERE

::READ AN OCTAL NUMBER
::LOW ORDER BITS ARE ON TOP OF THE STACK
::HIGH ORDER BITS ARE IN \$HIOCT

10224

ERROR HANDLER ROUTINE

10227 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

10272

ERROR MESSAGE TYPEOUT ROUTINE

10275 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.

10329 *****
BINARY TO OCTAL (ASCII) AND TYPE

10331 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
```

10405 *****
TRAP DECODER

10408 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.



10421 *****
TRAP TABLE

10423 THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
BY THE "TRAP" INSTRUCTION.

10444

POWER DOWN AND UP ROUTINES

39 40 56 167 177 188 203 240 298 2025 2261 2433 2434 2435 244

TABLE OF CONTENTS

40	
41	***ERROR TABLE, BIT DEFINITIONS & STARTING ADDRESSES***
42	
44	OPERATIONAL SWITCH SETTINGS
60	BASIC DEFINITIONS
171	TRAP CATCHER
181	ACT11 HOOKS
192	STARTING ADDRESSES
207	MEMORY MANAGEMENT DEFINITIONS
244	COMMON TAGS
302	ERROR POINTER TABLE
2029	HARDWARE REGISTER BIT DEFINITIONS
2265	REGISTER ADDRESSES
2437	
2438	***PROGRAM SETUP & SETUP TESTS***
2439	
2448	INITIALIZE THE COMMON TAGS
2543	GET VALUE FOR SOFTWARE SWITCH REGISTER
2571	T1 REFERENCE EACH REGISTER
2617	T2 RHCS2-CONTROL AND STATUS 2
2649	T3 PARTIAL TEST OF RHAS FOR UNIT NUMBERS PRESENT
2668	T4 TEST FOR DRIVES PRESENT USING RHAS AND RHCS2
2802	T5 TYPE SERIAL NUMBER AND DRIVE TYPE
2928	T6 CHECK MOL TO BE LOW
2966	T7 PACK ACKNOWLEDGE COMMAND TEST
3032	T10 MAKE CURRENT CYLINDER = 0
3046	
3047	***DIAGNOSTIC CODE***
3048	
3058	T11 BCTA LEGAL REGISTER RESPONSE TEST
3101	T12 BCTA MOVB LO BYTE TO WC
3126	T13 BCTA MOVB HI BYTE TO WC
3157	T14 BCTA BISB LO BYTE TO WC
3185	T15 BCTA BISB HI-BYTE TO WC
3217	T16 BCTA BICB LO-BYTE TO WC
3246	T17 BCTA BICB HI- BYTE TO WC
3290	T20 BCTA ILLEGAL REGISTER TEST
3333	T21 BCTB (NED) NON-EXISTANT DRIVE TEST. (SET)
3404	T22 BCTB (NED) NON-EXISTANT DRIVE TEST. (CLEARED)
3449	T23 BCTB AS REGISTER TEST
3489	T24 BCTC BUS ADDRESS REGISTER
3550	T25 BCTC BUS ADDRESS REGISTER LO-BYTE
3598	T26 BCTC BUS ADDRESS REGISTER HI-BYTE
3658	T27 RH11 INTERRUPT TEST
3693	T30 BCTD CLR L TEST
3721	T31 MXF TEST
3757	T32 CSRB UNIBUS PARITY ERROR SET TEST
3797	T33 CSRB UNIBUS PARITY ERROR CLEAR TEST
3831	T34 CSRB UNIT SELECT CLEAR TEST
3865	T35 CSRB TRANSFER ERROR (TRE) - UPE
3904	T36 CSRB TRANSFER ERROR (TRE) NED
3944	T37 CSRA PSEL CLEAR TEST
3974	T40 CSRB COMMAND REGISTER CLEAR TEST
4008	T41 CSRB COMMAND REGISTER RESELECT CLEAR TEST

TABLE OF CONTENTS

4040	T42	CSRB READY AND IE INTERRUPT TEST
4073	T43	CSRB BOES "IE" CLEAR AFTER AN INTERRUPT
4113	T44	BCTB "AS" WRITE TEST
4147		EXTENDED MEMORY, DRIVE TIMING & SECTOR SELECT TESTS
4150	T45	MAKE CURRENT CYLINDER = 0
4169	T46	RHCS1 - BITS 8 AND 9 - EXTENDED ADDR (A16 & A 17)
4265	T47	DRIVE TIMING ERROR
4441	T50	DRIVE TIMING ERROR
4617	T51	DRIVE TIMING ERROR
4753	T52	SECTOR SELECTION
4936		DATA TRANSFER TESTS USING ECC
4940	T53	WRITE ECC TEST 1
5091	T54	READ ECC ENABLED 1A
5265	T55	READ ECC ENABLED 1B
5446	T56	READ ECC ENABLED 1C
5639	T57	WRITE ECC TEST 2
5791	T60	READ ECC ENABLED 2A
5962	T61	READ ECC ENABLED 2B
6139	T62	READ ECC ENABLED 2C
6327	T63	WRITE ECC TEST 3
6473	T64	READ ECC ENABLED 3A
6637	T65	READ ECC ENABLED 3B
6822	T66	READ ECC ENABLED 3C
7013	T67	READ ECC ENABLED 3D
7201		CURSORY INTERRUPT LOGIC TESTS
7205	T70	PROGRAM INTERRUPT
7243	T71	INTERRUPT AT PROCESSOR AND DISK PRIORITY SAME
7278	T72	END OF DRIVE

SUBROUTINES

7328		END OF PASS ROUTINE
7329		SAVE REGISTERS ROUTINE
7334		FLOAT 1 AND 0
7455		CLEAR MEMORY ROUTINE
7482		LOCAL TRAPS
7523		CLEAR DISK ROUTINE
7558		CHECK DISK STATUS ROUTINES
7575		WAIT LOOP
7588		SAVE ROUTINE
7686		WRITE CHECK ROUTINE
7731		COMPARE ROUTINE
7761		WRITE CHECK DATA
7805		CRC GENERATION ROUTINE
7857		SIMULATED DISK SETUP
7903		CHECK HCE ROUTINE
7997		EXIT WRT HEADER & DATA ROUTINE
8044		JAM CURRENT CYLINDER ROUTINE
8192		ECC GENERATION AND COMPARISON ROUTINE
8226		ECC GENERATION CONTROL ROUTINE
8265		SOFTWARE DISK DATA ECC GEN. ROUTINE
8495		RH BASE ADDRESS CHANGE ROUTINE
8555		DISK SIMULATION
8597		
8677		
9788		

SYSMAC LIBRARY ROUTINES

9789	SCOPE HANDLER ROUTINE
9790	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
9794	TYPE ROUTINE
9869	TTY INPUT ROUTINE
9936	READ AN OCTAL NUMBER FROM THE TTY
10006	ERROR HANDLER ROUTINE
10235	ERROR MESSAGE TYPEOUT ROUTINE
10288	BINARY TO OCTAL (ASCII) AND TYPE
10335	TRAP DECODER
10392	TRAP TABLE
10469	POWER DOWN AND UP ROUTINES
10485	
10509	

K04

MNDEC-11-DZRJH-A, RPD4/5/6 DSKLS CONTROLLER TST-PT 2
DZRJHA.ITM OPERATIONAL SWITCH SETTINGS

MACY11 27(655) 30-MAR-76 20:59 PAGE 2

SEQ 0048

55


```

56          .SBTTL  BASIC DEFINITIONS
57
58          ;*INITIAL ADDRESS OF THE STACK POINTER *** 1000 ***
59          001000  STACK= 1000
60          .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
61          .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
62
63          ;*MISCELLANEOUS DEFINITIONS
64          000011  HT= 11          ;;CODE FOR HORIZONTAL TAB
65          000012  LF= 12          ;;CODE FOR LINE FEED
66          000015  CR= 15          ;;CODE FOR CARRIAGE RETURN
67          000200  CRLF= 200       ;;CODE FOR CARRIAGE RETURN-LINE FEED
68          177776  PS= 177776     ;;PROCESSOR STATUS WORD
69          .EQUIV  PS,PSW
70          177774  STKLMT= 177774 ;;STACK LIMIT REGISTER
71          177772  PIRQ= 177772   ;;PROGRAM INTERRUPT REQUEST REGISTER
72          177570  DSWR= 177570   ;;HARDWARE SWITCH REGISTER
73          177570  DDISP= 177570  ;;HARDWARE DISPLAY REGISTER
74
75          ;*GENERAL PURPOSE REGISTER DEFINITIONS
76          000000  R0= %0          ;;GENERAL REGISTER
77          000001  R1= %1          ;;GENERAL REGISTER
78          000002  R2= %2          ;;GENERAL REGISTER
79          000003  R3= %3          ;;GENERAL REGISTER
80          000004  R4= %4          ;;GENERAL REGISTER
81          000005  R5= %5          ;;GENERAL REGISTER
82          000006  R6= %6          ;;GENERAL REGISTER
83          000007  R7= %7          ;;GENERAL REGISTER
84          .EQUIV  R6,SP          ;;STACK POINTER
85          .EQUIV  R7,PC          ;;PROGRAM COUNTER
86
87          ;*PRIORITY LEVEL DEFINITIONS
88          000000  PR0= 0          ;;PRIORITY LEVEL 0
89          000040  PR1= 40         ;;PRIORITY LEVEL 1
90          000100  PR2= 100        ;;PRIORITY LEVEL 2
91          000140  PR3= 140        ;;PRIORITY LEVEL 3
92          000200  PR4= 200        ;;PRIORITY LEVEL 4
93          000240  PR5= 240        ;;PRIORITY LEVEL 5
94          000300  PR6= 300        ;;PRIORITY LEVEL 6
95          000340  PR7= 340        ;;PRIORITY LEVEL 7
96
97          ;*"SWITCH REGISTER" SWITCH DEFINITIONS
98          100000  SW15= 100000
99          040000  SW14= 40000
100         020000  SW13= 20000
101         010000  SW12= 10000
102         004000  SW11= 4000
103         002000  SW10= 2000
104         001000  SW09= 1000
105         000400  SW08= 400
106         000200  SW07= 200
107         000100  SW06= 100
108         000040  SW05= 40
109         000020  SW04= 20

```

110 000010
111 000004
112 000002
113 000001

SW03= 10
SW02= 4
SW01= 2
SW00= 1
.EQUIV SW09,SW9
.EQUIV SW08,SW8
.EQUIV SW07,SW7
.EQUIV SW06,SW6
.EQUIV SW05,SW5
.EQUIV SW04,SW4
.EQUIV SW03,SW3
.EQUIV SW02,SW2
.EQUIV SW01,SW1
.EQUIV SW00,SW0

125
126 100000
127 040000
128 020000
129 010000
130 004000
131 002000
132 001000
133 000400
134 000200
135 000100
136 000040
137 000020
138 000010
139 000004
140 000002
141 000001

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)

BIT15= 100000
BIT14= 40000
BIT13= 20000
BIT12= 10000
BIT11= 4000
BIT10= 2000
BIT09= 1000
BIT08= 400
BIT07= 200
BIT06= 100
BIT05= 40
BIT04= 20
BIT03= 10
BIT02= 4
BIT01= 2
BIT00= 1
.EQUIV BIT09,BIT9
.EQUIV BIT08,BIT8
.EQUIV BIT07,BIT7
.EQUIV BIT06,BIT6
.EQUIV BIT05,BIT5
.EQUIV BIT04,BIT4
.EQUIV BIT03,BIT3
.EQUIV BIT02,BIT2
.EQUIV BIT01,BIT1
.EQUIV BIT00,BIT0

153
154 000004
155 000010
156 000014
157 000014
158 000014
159 000020
160 000024
161 000030
162 000034
163 000060

.*BASIC "CPU" TRAP VECTOR ADDRESSES

ERRVEC= 4 ;; TIME OUT AND OTHER ERRORS
RESVEC= 10 ;; RESERVED AND ILLEGAL INSTRUCTIONS
TBITVEC= 14 ;; "T" BIT
TRIVEC= 14 ;; TRACE TRAP
BPTVEC= 14 ;; BREAKPOINT TRAP (BPT)
IOTVEC= 20 ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
PWRVEC= 24 ;; POWER FAIL
EMTVEC= 30 ;; EMULATOR TRAP (EMT) **ERROR**
TRAPVEC= 34 ;; "TRAP" TRAP
TKVEC= 60 ;; TTY KEYBOARD VECTOR

N04

MNDEC-11-DZRJH-A, RPO4/5/6 DSKLC CONTROLLER TST-PT 2
DZRJHA.ITM BASIC DEFINITIONS

MACY11 27(655) 30-MAR-76 20:59 PAGE 5

SEQ 0051

164
165
166

000064
000240

TPVEC= 64
PIRQVEC=240

:::TTY PRINTER VECTOR
:::PROGRAM INTERRUPT REQUEST VECTOR



22

.SBTTL COMMON TAGS

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

```

001100
001100 000000
001102 000
001103 000
001104 000000
001106 000000
001110 000000
001112 000000
001114 000
001115 001
001116 000000
001120 000000
001122 000000
001124 000000
001126 000000
001130 000000
001132 000000
001134 000
001135 000
001136 000000
001140 177570
001142 177570
001144 177560
001146 177562
001150 177564
001152 177566
001154 000
001155 002
001156 012
001157 000
001160 000000
001162 000000
001164 000000
001166 000000
001170 000000
001172 000000
001174 000000
001176 000000
001200 000000
001202 000000
001204 000000
001206 000000
001210 000000
001212 000000
001214 000000
001216 177607 C00377

```

```

.=1100
SCMTAG:
$PASS: .WORD 0
$STNM: .BYTE 00
$ERFLG: .BYTE 00
$ICNT: .WORD 00
$LPADR: .WORD 00
$LPERR: .WORD 00
$ERTTL: .WORD 00
$ITEMB: .BYTE 0
$ERMAX: .BYTE 1
$ERRPC: .WORD 00
$GADR: .WORD 00
$BADR: .WORD 00
$GDADR: .WORD 00
$BDADR: .WORD 00
$GDDAT: .WORD 00
$BDDAT: .WORD 00
$AUTOB: .BYTE 0
$INTAG: .BYTE 0
$SWR: .WORD DSWR
$DISPLAY: .WORD DDISP
$TKS: 177560
$TKB: 177562
$TPS: 177564
$TPB: 177566
$NULL: .BYTE 0
$FILLS: .BYTE 2
$FILLC: .BYTE 12
$TPFLG: .BYTE 0
$REGAD: .WORD 0
$REG0: .WORD 0
$REG1: .WORD 0
$REG2: .WORD 0
$REG3: .WORD 0
$REG4: .WORD 0
$REG5: .WORD 0
$TMP0: .WORD 0
$TMP1: .WORD 0
$TMP2: .WORD 0
$TMP3: .WORD 0
$TMP4: .WORD 0
$TMP5: .WORD 0
$TIMES: 0
$ESCAPE: 0
$BELL: .ASCIIZ <20><37><37>

```

```

:: START OF COMMON TAGS
:: CONTAINS PASS COUNT
:: CONTAINS THE TEST NUMBER
:: CONTAINS ERROR FLAG
:: CONTAINS SUBTEST ITERATION COUNT
:: CONTAINS SCOPE LOOP ADDRESS
:: CONTAINS SCOPE RETURN FOR ERRORS
:: CONTAINS TOTAL ERRORS DETECTED
:: CONTAINS ITEM CONTROL BYTE
:: CONTAINS MAX. ERRORS PER TEST
:: CONTAINS PC OF LAST ERROR INSTRUCTION
:: CONTAINS ADDRESS OF 'GOOD' DATA
:: CONTAINS ADDRESS OF 'BAD' DATA
:: CONTAINS 'GOOD' DATA
:: CONTAINS 'BAD' DATA
:: RESERVED--NOT TO BE USED

:: AUTOMATIC MODE INDICATOR
:: INTERRUPT MODE INDICATOR

:: ADDRESS OF SWITCH REGISTER
:: ADDRESS OF DISPLAY REGISTER
:: TTY KBD STATUS
:: TTY KBD BUFFER
:: TTY PRINTER STATUS REG. ADDRESS
:: TTY PRINTER BUFFER REG. ADDRESS
:: CONTAINS NULL CHARACTER FOR FILLS
:: CONTAINS # OF FILLER CHARACTERS REQUIRED
:: INSERT FILL CHARS. AFTER A "LINE FEED"
:: "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
:: CONTAINS THE ADDRESS FROM
:: WHICH ($REG0) WAS OBTAINED
:: CONTAINS (($REGAD)+0)
:: CONTAINS (($REGAD)+2)
:: CONTAINS (($REGAD)+4)
:: CONTAINS (($REGAD)+6)
:: CONTAINS (($REGAD)+10)
:: CONTAINS (($REGAD)+12)
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: USER DEFINED
:: MAX. NUMBER OF ITERATIONS
:: ESCAPE ON ERROR ADDRESS
:: CODE FOR BELL

```

294 001222 077
295 001223 015
296 001224 000012
297

\$QUES: .ASCII /?/ ;:QUESTION MARK
\$CRLF: .ASCII <15> ;:CARRIAGE RETURN
\$LF: .ASCIZ <12> ;:LINE FEED
;:*****

298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351

.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

001226

\$ERRTB:

:ITEM1

001226 002136
001230 005452

EM1
DH1

:WRONG DATA IN READING OR WRITING HARDWARE REGISTER
:PC
:REG. ADDR.
:GOOD DATA
:RECEIVED DATA
:\$ERRPC,\$STNM,REGADR,\$GDDAT,\$BDDAT
:0,0,0,0

:ITEM2

001236 002221
001240 010464

EM2
DH33

:ERROR ON DATA COMMAND
:PC
:PC OF JSR
:TEST NO
:WORD NO.
:GOOD DATA
:CONTENTS OF RHCS1
:CONTENTS OF RHDS1
:CONTENTS OF RHER1
:\$ERRPC,PCJSR,\$STNM,ERWORD,\$GDDAT,CS1,DS1,ER1
:0,0,0,1,0,0,0,0

:ITEM3

001246 002221
001250 010247

EM2
DH32

:ERROR ON DATA COMMAND
:PC
:PC OF JSR
:TEST NO

352					:WORD NO.
353					:GOOD DATA
354					:BAD DATA
355					:CONTENTS OF RHCS1
356					:CONTENTS OF RHDS1
357					:CONTENTS OF RHER1
358					
359	001252	012326		DT32	:SERRPC,PCJSR,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
360	001254	012657		DF32	:0,0,0,1,0,0,0,0,0.
361					
362					
363					:ITEM4
364	001256	002221		EM2	:ERROR ON DATA COMMAND
365					
366	001260	010051		DH31	:PC
367					:TEST NO
368					:WORD NO.
369					:GOOD DATA
370					:BAD DATA
371					:CONTENTS OF RHCS1
372					:CONTENTS OF RHDS1
373					:CONTENTS OF RHER1
374					
375	001262	012304		DT31	:SERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
376	001264	012647		DF31	:0,0,1,0,0,0,0,C.
377					
378					
379					
380					:ITEM5
381	001266	000000		0	:
382	001270	000000		0	:
383	001272	012304		DT31	:SERRPC,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
384	001274	012647		DF31	:0,0,1,0,0,0,0,0.
385					
386					
387					
388	001276	002250		EM6	:ERROR ON WRITE HEADER AND DATA
389					
390	001300	010247		DH32	:PC
391					:PC OF JSR
392					:TEST NO
393					:WORD NO.
394					:GOOD DATA
395					:BAD DATA
396					:CONTENTS OF RHCS1
397					:CONTENTS OF RHDS1
398					:CONTENTS OF RHER1
399					
400	001302	012326		DT32	:SERRPC,PCJSR,\$STSTNM,ERWORD,\$GDDAT,\$BDDAT,CS1,DS1,ER1
401	001304	012657		DF32	:0,0,0,1,0,0,0,0,0.
402					
403					
404					
405					:ITEM7

406	001306	002250	EM6	: ERROR ON WRITE HEADER AND DATA
407	001310	005570	DH2	: PC
408				: TEST NO
409				: WORD NO.
410				: GOOD DATA
411				: BAD DATA
412	001312	012014	DT3	: \$ERRPC, \$STNM, ERWORD, \$GDDAT, \$BDDAT
413	001314	012531	DF3	: 0,0,1,0,0,
414				
415				
416				: ITEM10
417	001316	000000	0	:
418	001320	000000	0	:
419	001322	012014	DT3	: \$ERRPC, \$STNM, ERWORD, \$GDDAT, \$BDDAT
420	001324	012531	DF3	: 0,0,1,0,0,
421				
422				
423				: ITEM11
424	001326	002307	EM11	: CONTROLLER OR DRIVE STATUS
425	001330	005702	DH11	: PC
426				: TEST NO
427				: FAILING REG. ADDR
428				: CONTENTS OF RHCS1
429				: CONTENTS OF RHCS2
430				: CONTENTS OF RHDS1
431				: CONTENTS OF RHER1
432	001332	012030	DT11	: \$ERRPC, \$STNM, \$BCADR, CS1, CS2, DS1, ER1
433	001334	01253E	DF11	: 0,0,0,0,0,0
434				
435				
436				: ITEM12
437	001336	002307	EM11	: WRONG DATA FROM SILO
438				
439	001340	005452	DH1	: PC
440				: REG. ADDR
441				: GOOD DATA
442				: RECEIVED DATA
443	001342	011766	DT1	: \$ERRPC, REGADR, \$GDDAT, \$BDDAT
444	001344	012520	DF1	: 0,0,0,0
445				
446				
447				: ITEM13
448	001346	000000	0	:
449	001350	000000	0	:
450	001352	011766	DT1	: \$ERRPC, TSTNM, REGADR, \$GDDAT, \$BDDAT
451	001354	012520	DF1	: 0,0,0,0,0
452				
453				
454				: ITEM14
455	001356	002342	EM14	: REGISTER FAILED
456	001360	006057	DH14	: PC
457				: FAILING REG. ADDR
458				: CONTENTS OF FAILING REG.
459				: CONTENTS OF RHCS1

460				: CONTENTS OF RHCS2
461				: CONTENTS OF RHDS1
462				: CONTENTS OF RHER1
463	001362	012050	DT14	: \$ERRPC, \$BDADR, \$BDDAT, CS1, CS2, DS1, ER1
464	001264	012545	DF14	: 0, 0, 0, 0, 0, 0, 0
465				
466				
467				: ITEM15
468	001366	002362	EM15	: SPECIFIED REG. NON EXISTANT SO ABORT
469				: PROGRAM
470	001370	006256	DH15	: PC
471				: ADDR. OF REG
472	001372	012072	DT15	: \$ERRPC, TEMP1
473	001374	012555	DF15	: 0, 0
474				
475				
476				: ITEM16
477	001376	002433	EM16	: WAIT LOOP FAILED
478	001400	006306	DH16	: PC
479				: WAT PC
480				: BIT WANTED
481				: REG. ADR.
482				: REG. CONT.
483	001402	012102	DT16	: \$ERRPC, \$TMP3, \$TMP1, \$TMP0, \$BDDAT
484	001404	012560	DF16	: 0, 0, 0, 0
485				
486				
487				: ITEM17
488	001406	002454	EM17	: WRITE CHECK FAILING
489	001410	006444	DH17	: PC
490				: TEST NO
491				: CONTENTS OF RHBA
492				: CONTENTS OF RHDB
493				: CONTENTS OF RHWC
494				: CONTENTS OF RHCS1
495				: CONTENTS OF RHCS2
496	001412	012120	DT17	: \$ERRPC, \$TSTNM, \$BA, DB, WC, CS1, CS2
497	001414	012565	DF17	: 0, 0, 0, 0, 0, 0, 0
498				
499				
500				: ITEM20
501	001416	002500	EM20	: REGISTER FAILING
502	001420	006621	DH20	: PC
503				: TST NO
504				: CONTENTS OF RHER1
505				: CONTENTS OF RHER2
506				: CONTENTS OF RHER3
507				: CONTENTS OF RHAS
508				: CONTENTS OF RHDS1
509	001422	012140	DT20	: \$ERRPC, TSTNM ER1, ER2, ER3, AS, DS1
510	001424	012574	DF20	: 0, 0, 0, 0, 0, 0, 0
511				
512				: ITEM21
513				

514	001426	002521	EM21	: INTERRUPT FAILING
515	001430	006775	DH21	: PC
516				: TEST NO
517				: CONTENTS OF RHCS1
518				: CONTENTS OF RHAS
519				: CONTENTS OF RHDS1
520	001432	012160	DT21	: \$ERRPC, TSTNM, CS1, AS, DS1
521	001434	012603	DF21	: 0,0,0,0,0
522				
523				
524				
525	001436	002543	EM22	: ; ITEM22
526				: MISSMATCH IN DRIVE PRESENT
527				: LOCKING AT RHAS AND RHCS2-NED(BIT#12)
528				: DRIVE PRESENT DO NOT AGREE
529				: NOTE: ON DUAL PORT SYSTEM
530				: DRIVE ON OTHER PORT WILL NOT GIVE NED
531				: HENCE THERE WILL BE A MISSMATCH
532	001440	007111	DH22	: 17777-MEANS NOT PRESENT
533				: PC
534				: TEST NO
535				: RHAS UNIT
536				: RHCS2 UNIT
537	001442	012174	DT22	: \$ERRPC, TSTNMS, \$GDDAT, \$BDDAT
538	001444	012610	DF22	: 0,0,0,0
539				
540				
541				
542	001446	000000	0	: ; ITEM23
543				: MISSMATCH IN DRIVE PRESENT
544				: LOCKING AT RHAS AND RHCS2-NED(BIT#12)
545				: DRIVE PRESENT DO NOT AGREE
546	001450	000000	0	: 17777-MEANS NOT PRESENT
547				: PC
548				: TEST NO
549				: RHAS UNIT
550				: RHCS2 UNIT
551	001452	012174	DT22	: \$ERRPC, TSTNMS, \$GDDAT, \$BDDAT
552	001454	012610	DF22	: 0,0,0,0
553				
554				
555				
556				
557	001456	003144	EM24	: ; ITEM 24
558				: LOOK AHEAD REGISTER AT THE
559				: BEGINNING OF A SECTOR IS IN
560	001460	007205	DH24	: ERROR
561				: PC
562				: RHDST
563				: BAD RHLA
564				: GOOD RHLA
565				: SECTOR NO
566	001462	012206	DT24	: SECTOR CLOCK
567	001464	012614	DF24	: \$ERRPC, DST, \$BDDAT, \$TMP1, \$TMP2, \$TMP3
				: 0,0,0,0,0

Handwritten mark or signature.

568					
569					
570	001466	003237		; ITEM 25	; LOOK AHEAD REGISTER IS
571				EM25	; IN ERROR
572					
573	001470	007205		DH24	; PC
574					; RHDS1
575					; BAD RHLA
576					; GOOD RHLA
577					; SECTOR NO
578					; SECTOR CLOCK
579	001472	012206		DT24	; \$ERRPC, DST, \$BDDAT, \$TMP1, \$TMP2, \$TMP3
580	001474	012614		DF24	; 0,0,0,0,0
581					
582	001476	002307		; ITEM26	; CONTROLLER OR DRIVE STATUS
583				EM11	
584	001500	007363		DH26	; PC
585					; PC OF JSR
586					; FAILING REGISTER ADDRESS
587					; CONTENTS OF RHCS1
588					; CONTENTS OF RHCS2
589					; CONTENTS OF RHDS1
590					; CONTENTS OF RHER1
591					
592	001502	012226		DT26	; \$ERRPC, PCJSR, \$BDADR, CS1, CS2, DS1, ER1
593	001504	012623		DF26	; 0,0,0,0,0,0
594					
595					
596					
597					
598	001506	002136		; ITEM27	; ERROR IN READING OR WRITING HARDWARE REGISTER
599				EM1	
600	001510	007561		DH27	; PC
601					; PC OF JSR
602					; TEST NUMBER
603					; FAILING REGISTER
604					; GOOD DATA
605					; RECEIVED DATA
606					
607	001512	012250		DT27	; \$ERRPC, PCJSR, TSTNM, REGADR, \$GDDAT, \$BDDAT
608	001514	012633		DF27	; 0,0,0,0,0,0
609					
610					
611					
612					
613	001516	003277		; ITEM30	; CURRENT CYLINDER DOES NOT REFLECT DESIRED CYLINDER REG.
614	001520	007717		EM30	; PC
615				DH30	; PC OF JSR
616					; REGISTER ADDRESS
617					; GOOD DATA
618					; BAD DATA
619					
620	001522	012266		DT30	; \$ERRPC, PCJSR, REGADR, \$GDDAT, \$BDDAT
621	001524	012641		DF30	; 0,0,0,0,0

622					
623					
624					
625			; ITEM31		
626	001526	003421	EM31		; ECC GENERATED IS INCORRECT
627					; EVERY WORD IN THIS SECTOR IS GIVEN IN "DATA USED"
628					
629	001530	010663	DH34		; PC
630					; TEST NUMBER
631					; GOOD ECC1
632					; GOOD EC2C
633					; WRITTEN ECC1
634					; WRITTEN ECC2
635					; DATA USED
636					
637	001532	012374	DT34		; \$ERRPC, TSTNM, GECC1, GECC2, WECC1, WECC2, DISK
638					
639	001534	012700	DF34		; 0,0,0,0,0,0,0
640					
641					
642			; ITEM32		
643	001536	003544	EM32		; ON READ COMMAND AFTER DATA AND ECC HAVE BEEN READ
644					; ECC REGISTER OR RHER1 IS IN ERROR
645					; ONLY LOWER 11 BITS OF PATTERN REGISTER
646					; CAN BE READ
647					; THIS SHUOLD MATCH LOWER 11 BITS OF ECC1
648					
649	001540	011036	DH35		; PC
650					; TEST NUMBER
651					; GOOD ECC1
652					; GOOD ECC2
653					; PATTERN REGISTER
654					; RHER1
655					
656	001542	012414	DT35		; \$ERRPC, TSTNM, GECC1, GECC2, EC2, ER1
657					
658	001544	012707	DF35		; 0,0,0,0,0,0,0
659					
660					
661					
662			; ITEM33		
663	001546	004033	EM33		; HIGH COUNT BIT NOT HIGH AFTER 38859 CLOCKS
664	001550	011233	DH36		; PC
665					; PC OF JSR
666					; TEST NUMBER
667					; RHMR
668					; POSITION REG.
669					; PATTERN REGISTER
670					
671	001552	012436	DT36		; \$ERRPC, PCJSR, TSTNM, MR, EC1, EC2
672					
673	001554	012717	DF36		; 0,0,0,0,0,0,0
674					
675			; ITEM34		

676	001556	004105	EM34		
677					: ZERO DETECT BIT NOT HIGH WHEN THE
678					: 32 BIT ECC REGISTER HAS ITS 21 BITS
679					: OF ZEROS
680					: ERROR PRINTOUT WILL CONTINUE TILL
681	001560	011233	DH36		: ZERO DETECT BIT IS HIGH
682					: PC
683					: PC OF JSR
684					: TEST NUMBER
685					: RHMR
686					: POSITION REG.
687					: PATTERN REGISTER
688	001562	012436	DT36		: \$ERRPC, PCJSR, TSTNM, MR, EC1, EC2
689					
690	001564	012717	DF36		: 0,0,0,0,0,0
691					
692					
693					
694					
695	001566	004200	EM35	; ITEM35	: POSITION REGISTER OR 11 BITS OF
696					: PATTERN REGISTER INCORRECT
697					: LOWER 11 BITS OF PATTERN REGISTER
698					: SHOULD MATCH LOWER 11 BITS OF GOOD ECC1
699					: DATA ENVELOPE AND N-CODE ZEROS ARE IN DECIMAL
700					
701	001570	011371	DH37		: PC
702					: TEST NUMBER
703					: ECC POSITION
704					: GOOD POSITION
705					: GOOD ECC1
706					: GOOD ECC2
707					: ECC PATTERN
708					: DATA ENVELOPE
709					: N-CODE ZEROS
710					
711	001572	012454	DT37		: \$ERRPC, TSTNM, EC1, POSITI, GECC1, GECC2, EC2, DATENV, ZCODE
712					
713	001574	012725	DF37		: 0,0,0,0,0,0,0,0,0
714					
715					
716					
717					
718	001576	004477	EM36	; ITEM36	: ON A READ COMMAND WITH NON CORRECTABLE
719					: ERROR INSERTED DCK AND ECH SHOULD BE SET
720	001600	011036	DH35		: PC
721					: TEST NUMBER
722					: GOOD ECC1
723					: GOOD ECC2
724					: PATTERN REGISTER
725					: POSITION REGISTER
726					: RHER1
727					
728	001602	012414	DT35		: \$ERRPC, TSTNM, GECC1, GECC2, EC2, EC1, ER1
729					

730	001604	012707	DF35		;0,0,0,0,0,0,0
731					
732					
733					
734					
735					
736					
737				: ITEM37	
738	001606	004665	EM37		;ERROR ON DATA COMMAND ;WITH A16 A17 USED
739					
740					
741	001610	010051	DH31		;PC ;TEST NO ;WORD NO. ;GOOD DATA ;BAD DATA ;CONTENTS OF RHCS1 ;CONTENTS OF RHDS1 ;CONTENTS OF RHER1
742					
743					
744					
745					
746					
747					
748					
749					
750	001612	012304	DT31		;SERRPC, \$STSTNM, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1
751	001614	012647	DF31		;0,0,1,0,0,0,0,
752					
753					
754					
755				: ITEM40	
756	001616	000000	0		
757	001620	000000	0		
758	001622	012304	DT31		;SERRPC, \$STSTNM, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1
759	001624	012647	DF31		;0,0,1,0,0,0,0,
760					
761					
762					
763					
764				: ITEM41	
765	001626	004753	EM40		;THERE WAS A READ/WRITE HEADER & DATA ;ERROR DURING 'DTE' TEST SETUP - THE ;TEST IS ABORTED AT THAT POINT
766					
767					
768	001630	011610	DH40		;PC ;TEST NO ;FAILING REGISTER ADDRESS ;CONTENTS OF RHCS1 ;CONTENTS OF RHCS2 ;CONTENTS OF RHDS1 ;CONTENTS OF RHER1
769					
770					
771					
772					
773					
774					
775	001632	012500	DT40		;SERRPC, \$STSTNM, \$BDADR, CS1, CS2, DS1, ER1
776	001634	012736	DF40		;0,0,0,0,0,0

001700-001734-R RPO4 5 6 DSKLS CONTROLLER TST-PT 2
001700-001734-ITEM ERROR POINTER TABLE

001700
001701
001702
001703
001704
001705
001706
001707
001708
001709
001710
001711
001712
001713
001714
001715
001716
001717
001718
001719
001720
001721
001722
001723
001724
001725
001726
001727
001728
001729
001730
001731
001732
001733
001734

001636 012746
001640 014412
001642 014674
001644 014732

001646 013031
001650 014412
001652 014674
001654 014732

001656 013075
001660 014473
001662 014704
001664 014735

001666 013132
001670 014473
001672 014704
001674 014735

001676 013161
001700 014473
001702 014704
001704 014735

001706 013210
001710 014412
001712 014674
001714 014732

001716 013245
001720 014473
001722 014704
001724 014735

001726 013303
001730 014473
001732 014704
001734 014735

:ITEM 42
EM42
DH42
DT42
DF42

:ITEM 43
EM43
DH42
DT42
DF42

:ITEM 44
EM44
DH44
DT44
DF44

:ITEM 45
EM45
DH44
DT44
DF44

:ITEM 46
EM46
DH44
DT44
DF44

:ITEM 47
EM47
DH42
DT42
DF42

:ITEM 50
EM50
DH44
DT44
DF44

:ITEM 51
EM51
DH44
DT44
DF44

:ERROR PC,TEST NUMBER, REGISTER ADDRESS.

:ERROR PC,TEST NUMBER, REGISTER ADDRESS.

:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

:ERROR PC,TEST NUMBER, REGISTER ADDRESS.

:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

831			
832			: ITEM 52
833			
834	001736	013324	EM52
835	001740	014473	DH44
836	001742	014704	DT44
837	001744	014735	DF44
838			
839			: ITEM 53
840			
841	001746	013364	EM53
842	001750	014473	DH44
843	001752	014704	DT44
844	001754	014735	DF44
845			
846			: ITEM 54
847			
848	001756	013424	EM54
849	001760	014605	DH54
850	001762	014720	DT54
851	001764	014742	DF54
852			
853			: ITEM 55
854			
855	001766	013463	EM55
856	001770	014473	DH44
857	001772	014704	DT44
858	001774	014735	DF44
859			
860			: ITEM 56
861			
862	001776	013476	EM56
863	002000	014473	DH44
864	002002	014704	DT44
865	002004	014735	DF44
866			
867			: ITEM 57
868			
869	002006	013513	EM57
870	002010	014473	DH44
871	002012	014704	DT44
872	002014	014735	DF44
873			
874			: ITEM 60
875			
876	002016	013542	EM60
877	002020	014473	DH44
878	002022	014704	DT44
879	002024	014735	DF44
880			
881			: ITEM 61
882			
883	002026	013631	EM61
884	002030	014473	DH44

:ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

:ERROR PC, TEST NUMBER.

:ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

:ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

:ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

:ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

885	002032	014704	DT44	:ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
886	002034	014735	DF44	
887				
888				:ITEM 62
889				
890	002036	013701	EM62	
891	002040	014473	DH44	
892	002042	014704	DT44	:ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
893	002044	014735	DF44	
894				
895				:ITEM 63
896				
897	002046	013756	EM63	
898	002050	014473	DH44	
899	002052	014704	DT44	:ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
900	002054	014735	DF44	
901				
902				:ITEM 64
903				
904	002056	014034	EM64	
905	002060	014473	DH44	
906	002062	014704	DT44	:ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
907	002064	014735	DF44	
908				
909				:ITEM 65
910				
911	002066	014070	EM65	
912	002070	014473	DH44	
913	002072	014704	DT44	:ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
914	002074	014735	DF44	
915				
916				:ITEM 66
917				
918	002076	014152	EM66	
919	002100	014473	DH44	
920	002102	014704	DT44	:ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
921	002104	014735	DF44	
922				
923				:ITEM 67
924				
925	002106	014224	EM67	
926	002110	014605	DH54	
927	002112	014720	DT54	:ERROR PC, TEST NUMBER.
928	002114	014742	DF54	
929				
930				:ITEM 70
931				
932	002116	014311	EM70	
933	002120	014605	DH54	
934	002122	014720	DT54	:ERROR PC, TEST NUMBER.
935	002124	014742	DF54	
936				
937				:ITEM 71
938				

939	002126	014363	EM71
940	002130	014473	DM44
941	002132	014704	DT44
942	002134	014735	DF44

:ERROR PC,TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.

```

943
944
945
946
947
948
949
950
951
952
953 002136 051127 047117 020107
954 002144 040504 040524 044440
955 002152 020116 042522 042101
956 002160 047111 020107 051117
957 002166 053440 044522 044524
958 002174 043516 044040 051101
959 002202 053504 051101 020105
960 002210 042522 044507 052123
961 002216 051105 000
962 002221 105 051122 051117
963 002226 047440 020116 042040
964 002234 052101 020101 047503
965 002242 046515 047101 000104
966 002250 051105 047522 020122
967 002256 047117 053440 044522
968 002264 042524 044040 040505
969 002272 042504 020122 047101
970 002300 020104 040504 040524
971 002306 000
972 002307 103 047117 051124
973 002314 046117 042514 020122
974 002322 051117 042040 044522
975 002330 042526 051440 040524
976 002336 052524 000123
977 002342 042522 044507 052123
978 002350 051105 043040 044501
979 002356 042514 000104
980 002362 050123 041505 043111
981 002370 042511 020104 042522
982 002376 044507 052123 051105
983 002404 047040 047117 042440
984 002412 044530 052123 047101
985 002420 020124 047523 040440
986 002426 047502 052122 000
987 002433 127 044501 020124
988 002440 047514 050117 043040
989 002446 044501 042514 000104
990 002454 051127 052111 020105
991 002462 044103 041505 020113
992 002470 040506 046111 047111
993 002476 000107
994 002500 042522 044507 052123
995 002506 051105 043040 044501
996 002514 044514 043516 000

```

```

:*****
:ERROR AND MESSAGE TABLE CONDIMENTS
:*****

```

EM1: .ASCIZ /WRONG DATA IN READING OR WRITING HARDWARE REGISTER/

EM2: .ASCIZ /ERROR ON DATA COMMAND/

EM6: .ASCIZ /ERROR ON WRITE HEADER AND DATA/

EM11: .ASCIZ /CONTROLLER OR DRIVE STATUS/

EM14: .ASCIZ /REGISTER FAILED/

EM15: .ASCIZ /SPECIFIED REGISTER NON EXISTANT SO ABORT/

EM16: .ASCIZ /WAIT LOOP FAILED/

EM17: .ASCIZ /WRITE CHECK FAILING/

EM20: .ASCIZ /REGISTER FAILING/

997	002521	111	052116	051105	EM21:	.ASCIZ	/INTERRUPT FAILING/
998	002526	052522	052120	043040			
999	002534	044501	044514	043516			
1000	002542	000					
1001	002543	105	051122	051117	EM22:	.ASCII	ERROR ON DRIVE PRESENT/<<15><12>
1002	002550	047440	020116	051104			
1003	002556	053111	020105	051120			
1004	002564	051505	047105	006524			
1005	002572	012					
1006	002573	124	042510	052440		.ASCII	/THE UNIT NO'S FOUND BY SETTING RHAS/<<15><12>
1007	002600	044516	020124	047516			
1008	002606	051447	043040	052517			
1009	002614	042116	041040	020131			
1010	002622	042523	052124	047111			
1011	002630	020107	044122	051501			
1012	002636	005015					
1013	002640	047504	047040	052117		.ASCII	/DO NOT AGREE WITH THE UNIT NO. FOUND FROM/<<15><12>
1014	002646	040440	051107	042505			
1015	002654	053440	052111	020110			
1016	002662	044124	020105	047125			
1017	002670	052111	047040	027117			
1018	002676	043040	052517	042116			
1019	002704	043040	047522	006515			
1020	002712	012					
1021	002713	122	041510	031123		.ASCII	/RHCS2-'NED' BIT #12/<<15><12>
1022	002720	023455	042516	023504			
1023	002726	041040	052111	021440			
1024	002734	031061	005015				
1025	002740	033461	033467	033467		.ASCII	/177777-MEANS NO UNIT FOUND/<<15><12>
1026	002746	046455	040505	051516			
1027	002754	047040	020117	047125			
1028	002762	052111	043040	052517			
1029	002770	042116	005015				
1030	002774	047516	042524	020072		.ASCII	/NOTE: ON DUAL PORT SYSTEM, DRIVE ON OTHER PORT WILL NOT GIVE/<<15><12>
1031	003002	047117	042040	040525			
1032	003010	020114	047520	052122			
1033	003016	051440	051531	042524			
1034	003024	026115	042040	044522			
1035	003032	042526	047440	020116			
1036	003040	052117	042510	020122			
1037	003046	047520	052122	053440			
1038	003054	046111	020114	047516			
1039	003062	020124	044507	042526			
1040	003070	005015					
1041	003072	047047	042105	026047		.ASCIZ	/'NED', HENCE THERE WILL BE AN EXTRA DRIVE/
1042	003100	044040	047105	042503			
1043	003106	052040	042510	042522			
1044	003114	053440	046111	020114			
1045	003122	042502	040440	020116			
1046	003130	054105	051124	020101			
1047	003136	051104	053111	000105			
1048	003144	047514	045517	040440	EM24:	.ASCIZ	/LOOK AHEAD REGISTER AT THE BEGINNING OF SECTOR IS IN ERROR
1049	003152	042510	042101	051040			
1050	003160	043505	051511	042524			

1051	003166	020122	052101	052040	
1052	003174	042510	041040	043505	
1053	003202	047111	044516	043516	
1054	003210	047440	020106	042523	
1055	003216	052103	051117	044440	
1056	003224	020123	047111	042440	
1057	003232	051122	051117	000	
1058	003237	114	047517	020113	EM25: .ASCIZ /LOOK AHEAD REGISTER IS IN ERROR/
1059	003244	044101	040505	020104	
1060	003252	042522	044507	052123	
1061	003260	051105	044440	020123	
1062	003266	047111	042440	051122	
1063	003274	051117	000		
1064	003277	103	051125	042522	EM30: .ASCII /CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER REGISTER/<15><12>
1065	003304	052116	041440	046131	
1066	003312	047111	042504	020122	
1067	003320	047504	051505	047040	
1068	003326	052117	046440	052101	
1069	003334	044103	042040	051505	
1070	003342	051111	042105	041440	
1071	003350	046131	047111	042504	
1072	003356	020122	042522	044507	
1073	003364	051510	042524	006522	
1074	003372	012			
1075	003373	101	052106	051105	.ASCIZ /AFTER A SEEK AND INIT/
1076	003400	040440	051440	042505	
1077	003406	020113	047101	020104	
1078	003414	047111	052111	000	
1079	003421	105	041503	043440	EM31: .ASCII /ECC GENERATED IS INCORRECT/<15><12>
1080	003426	047105	051105	052101	
1081	003434	042105	044440	020123	
1082	003442	047111	047503	051122	
1083	003450	041505	006524	012	
1084	003455	105	042526	054522	.ASCIZ /EVERY WORD ON THIS SECTOR IS THAT GIVEN IN "DATA USED"/
1085	003462	053440	051117	020104	
1086	003470	047117	052040	044510	
1087	003476	020123	042523	052103	
1088	003504	051117	044440	020123	
1089	003512	044124	052101	043440	
1090	003520	053111	047105	044440	
1091	003526	020116	042042	052101	
1092	003534	020101	051525	042105	
1093	003542	000042			
1094	003544	047117	051040	040505	EM32: .ASCII /ON READ COMMAND, AFTER DATA AND ECC HAVE BEEN READ,/<15><12>
1095	003552	020104	047503	046515	
1096	003560	047101	026104	040440	
1097	003566	052106	051105	042040	
1098	003574	052101	020101	047101	
1099	003602	020104	041505	020103	
1100	003610	040510	042526	041040	
1101	003616	042505	020116	042522	
1102	003624	042101	006454	012	
1103	003631	105	041503	051040	.ASCII /ECC REGISTERS OR RHER1 ARE IN ERROR/<15><12>
1104	003636	043505	051511	042524	

1105	003644	051522	047440	020122	
1106	003652	044122	051105	020061	
1107	003660	051101	020105	047111	
1108	003666	042440	051122	051117	
1109	003674	005015			
1110	003676	047117	054514	046040	.ASCII /ONLY LOWER 11 BITS OF PATTERN REG. CAN BE READ/<15><12>
1111	003704	053517	051105	030440	
1112	003712	020061	044502	051524	
1113	003720	047440	020106	040520	
1114	003726	052124	051105	020116	
1115	003734	042522	027107	041440	
1116	003742	047101	041040	020105	
1117	003750	042522	042101	005015	
1118	003756	044124	051511	051440	.ASCIZ /THIS SHOULD MATCH LOWER 11 BITS OF GOOD ECC1/
1119	003764	047510	046125	020104	
1120	003772	040515	041524	020110	
1121	004000	047514	042527	020122	
1122	004006	030461	041040	052111	
1123	004014	020123	043117	043440	
1124	004022	047517	020104	041505	
1125	004030	030503	000		
1126	004033	110	043511	020110	EM33: .ASCIZ /HIGH COUNT BIT NOT SET AFTER 38859 CLOCKS/
1127	004040	047503	047125	020124	
1128	004046	044502	020124	047516	
1129	004054	020124	042523	020124	
1130	004062	043101	042524	020122	
1131	004070	034063	032470	020071	
1132	004076	046103	041517	051513	
1133	004104	000			
1134	004105	132	051105	020117	EM34: .ASCIZ /ZERO DETECT BIT NOT HIGH WHEN 32 BIT ECC REG. HAS 21 ZEROS/
1135	004112	042504	042524	052103	
1136	004120	041040	052111	047040	
1137	004126	052117	044040	043511	
1138	004134	020110	044127	047105	
1139	004142	031440	020062	044502	
1140	004150	020124	041505	020103	
1141	004156	042522	027107	044040	
1142	004164	051501	031040	020061	
1143	004172	042532	047522	000123	
1144	004200	047520	044523	044524	EM35: .ASCII /POSITION REGISTER OR 11 BITS OF PATTERN REGISTER INCORRECT/<15><12>
1145	004206	047117	051040	043505	
1146	004214	051511	042524	020122	
1147	004222	051117	030440	020061	
1148	004230	044502	051524	047440	
1149	004236	020106	040520	052124	
1150	004244	051105	020116	042522	
1151	004252	044507	052123	051105	
1152	004260	044440	041516	051117	
1153	004266	042522	052103	005015	
1154	004274	047514	042527	020122	.ASCII /LOWER 11 BITS OF PATTERN REGISTER SHOULD MATCH LOWER/<15><12>
1155	004302	030461	041040	052111	
1156	004310	020123	043117	050040	
1157	004316	052101	042524	047122	
1158	004324	051040	043505	051511	

1159	004332	042524	020122	044123	
1160	004340	052517	042114	046440	
1161	004346	052101	044103	046040	
1162	004354	053517	051105	005015	
1163	004362	030461	041040	052111	.ASCII /11 BITS OF GOOD ECC1/<15><12>
1164	004370	020123	043117	043440	
1165	004376	047517	020104	041505	
1166	004404	030503	005015		
1167	004410	040504	020124	047105	.ASCIZ /DAT ENVLOP GOOD POSITION AND N-CODE ZEROS ARE IN OCTAL/
1168	004416	046126	050117	043440	
1169	004424	047517	020104	047520	
1170	004432	044523	044524	047117	
1171	004440	040440	042116	047040	
1172	004446	041455	042117	020105	
1173	004454	042532	047522	020123	
1174	004462	051101	020105	047111	
1175	004470	047440	052103	046101	
1176	004476	000			
1177	004477	117	020116	042522	EM36: .ASCII /ON READ COMMAND WITH NON-CORRECTABLE ERROR DCK AND ECH SHOULD BE SET/<1
1178	004504	042101	041440	046517	
1179	004512	040515	042116	053440	
1180	004520	052111	020110	047516	
1181	004526	026516	047503	051122	
1182	004534	041505	040524	046102	
1183	004542	020105	051105	047522	
1184	004550	020122	041504	020113	
1185	004556	047101	020104	041505	
1186	004564	020110	044123	052517	
1187	004572	042114	041040	020105	
1188	004600	042523	006524	012	
1189	004605	111	020106	047520	.ASCIZ /IF POSITION REGISTER =10040 OR 10041 IT IS GOOD/
1190	004612	044523	044524	047117	
1191	004620	051040	043505	051511	
1192	004626	042524	020122	030475	
1193	004634	030060	030064	047440	
1194	004642	020122	030061	032060	
1195	004650	020061	052111	044440	
1196	004656	020123	047507	042117	
1197	004664	000			
1198	004665	127	044522	044524	EM37: .ASCIZ /WRITING WITH BUS ADDRESS HIGHER THAN 28K CAUSED ERROR/
1199	004672	043516	053440	052111	
1200	004700	020110	052502	020123	
1201	004706	042101	051104	051505	
1202	004714	020123	044510	044107	
1203	004722	051105	052040	040510	
1204	004730	020116	034062	020113	
1205	004736	040503	051525	042105	
1206	004744	042440	051122	051117	
1207	004752	000			
1208	004753	124	042510	042522	EM40: .ASCII /THERE WAS A READ-WRITE HEADER & DATA ERROR DURING 'DTE'/<15><12>
1209	004760	053440	051501	040440	
1210	004766	051040	040505	026504	
1211	004774	051127	052111	020105	
1212	005002	042510	042101	051105	

1213	005010	023040	042040	052101
1214	005016	020101	051105	047522
1215	005024	020122	052504	044522
1216	005032	043516	023440	052104
1217	005040	023505	005015	
1218	005044	042524	052123	051440
1219	005052	052105	050125	026440
1220	005060	052040	042510	052040
1221	005066	051505	020124	040527
1222	005074	020123	041101	051117
1223	005102	042524	020104	052101
1224	005110	052040	040510	020124
1225	005116	047520	047111	000124

.ASCIZ /TEST SETUP - THE TEST WAS ABORTED AT THAT POINT/

1226	005124	040506	040524	020114	CPHALT: .ASCII /FATAL ERROR - SEE DOCUMENT FOR BEST COURSE OF ACTION/<15><12>
1227	005132	051105	047522	020122	
1228	005140	020055	042523	020105	
1229	005146	047504	052503	042515	
1230	005154	052116	043040	051117	
1231	005162	041040	051505	020124	
1232	005170	047503	051125	042523	
1233	005176	047440	020106	041501	
1234	005204	044524	047117	005015	
1235	005212	006440	103412	177777	.ASCII / /<15><12><207><377><377><207><377><377><207><377><377>
1236	005220	177607	103777	177777	
1237	005226	044124	020105	051120	.ASCII /THE PROGRAM HAS HALTED DURING A TEST BECAUSE CONTROLLER/<15><12>
1238	005234	043517	040522	020115	
1239	005242	040510	020123	040510	
1240	005250	052114	042105	042040	
1241	005256	051125	047111	020107	
1242	005264	020101	042524	052123	
1243	005272	041040	041505	052501	
1244	005300	042523	041440	047117	
1245	005306	051124	046117	042514	
1246	005314	006522	012		
1247	005317	117	020122	042504	.ASCII /OR DEVICE HAS LOST 'READY', BECOME UNAVAILABLE,/<15><12>
1248	005324	044526	042503	044040	
1249	005332	051501	046040	051517	
1250	005340	020124	051047	040505	
1251	005346	054504	026047	041040	
1252	005354	041505	046517	020105	
1253	005362	047125	053101	044501	
1254	005370	040514	046102	026105	
1255	005376	005015			
1256	005400	047507	042516	047440	.ASCIZ /GONE OFFLINE, OR CANNOT CLEAR STATUS BITS/
1257	005406	043106	044514	042516	
1258	005414	020054	051117	041440	
1259	005422	047101	047516	020124	
1260	005430	046103	040505	020122	
1261	005436	052123	052101	051525	
1262	005444	041040	052111	000123	
1263					
1264	005452	041520	020040	020040	DH1: .ASCII /PC TEST REG. GOOD ACTUAL /<15><12>
1265	005460	020040	042524	052123	
1266	005466	020040	020040	042522	
1267	005474	027107	020040	020040	
1268	005502	047507	042117	020040	
1269	005510	020040	041501	052524	
1270	005516	046101	006411	012	
1271	005523	040	020040	020040	.ASCIZ / NC ADDR. DATA DATA/
1272	005530	020040	047040	020117	
1273	005536	020040	020040	040440	
1274	005544	042104	027122	020040	
1275	005552	042040	052101	020101	
1276	005560	020040	042040	052101	
1277	005566	000101			
1278	005570	041520	020040	020040	DH2: .ASCII /PC TEST WORD GOOD BAD/<15><12>
1279	005576	020040	042524	052123	

1442	007376	020124	020040	050040																
1443	007404	020103	043117	020040																
1444	007412	043040	044501	044514																
1445	007420	043516	041440	047117																
1446	007426	027124	020040	041440																
1447	007434	047117	027124	020040																
1448	007442	041440	047117	027124																
1449	007450	020040	041440	047117																
1450	007456	027124	005015																	
1451	007462	020040	020040	020040	.ASCIZ /	NO	JSR	REG	ADD	RHCS1	RHCS2	RHCS1	RHER1 /							
1452	007470	020040	047516	020040																
1453	007476	020040	020040	051512																
1454	007504	020122	020040	020040																
1455	007512	042522	020107	042101																
1456	007520	020104	044122	051503																
1457	007526	020061	020040	044122																
1458	007534	051503	020062	020040																
1459	007542	044122	051504	020061																
1460	007550	020040	044122	051105																
1461	007556	004461	000																	
1462	007561	120	020103	020040	DH27:	.ASCII /PC	TEST	PC	OF	FAILING	GOOD	ACTUAL	<15><12>							
1463	007566	020040	052040	051505																
1464	007574	020124	020040	050040																
1465	007602	020103	043117	020040																
1466	007610	043040	044501	044514																
1467	007616	043516	043440	047517																
1468	007624	020104	020040	040440																
1469	007632	052103	040525	004514																
1470	007640	005015																		
1471	007642	020040	020040	020040	.ASCIZ /	NO	JSR	REG.	DATA	DATA/										
1472	007650	020040	047516	020040																
1473	007656	020040	020040	051512																
1474	007664	020122	020040	020040																
1475	007672	042522	027107	020040																
1476	007700	020040	040504	040524																
1477	007706	020040	020040	040504																
1478	007714	040524	000																	
1479	007717	120	020103	020040	DH30:	.ASCII /PC	TEST	PC	OF	REG.	GOOD	BAD	<15><12>							
1480	007724	020040	052040	051505																
1481	007732	020124	020040	050040																
1482	007740	020103	043117	020040																
1483	007746	051040	043505	020056																
1484	007754	020040	043440	047517																
1485	007762	020104	020040	041040																
1486	007770	042101	005015																	
1487	007774	020040	020040	020040	.ASCIZ /	NO	JSR	ADDR	DATA	DATA/										
1488	010002	020040	047516	020040																
1489	010010	020040	020040	051512																
1490	010016	020122	020040	020040																
1491	010024	042101	051104	020040																
1492	010032	020040	040504	040524																
1493	010040	020040	020040	040504																
1494	010046	040524	000																	
1495	010051	120	020103	020040	DH31:	.ASCII /PC	TEST	WORD	GOOD	BAD	CONT.	CONT.	CONT. <15><12>							

1496 010056 020040 052040 051505
 1497 010064 020124 020040 053440
 1498 010072 051117 020104 020040
 1499 010100 043440 047517 020104
 1500 010106 020040 041040 042101
 1501 010114 020040 020040 041440
 1502 010122 047117 027124 020040
 1503 010130 041440 047117 027124
 1504 010136 020040 041440 047117
 1505 010144 027124 005015
 1506 010150 020040 020040 020040
 1507 010156 020040 047516 020040
 1508 010164 020040 020040 047516
 1509 010172 020040 020040 020040
 1510 010200 040504 040524 020040
 1511 010206 020040 040504 040524
 1512 010214 020040 020040 044122
 1513 010222 051503 020061 020040
 1514 010230 044122 051504 020061
 1515 010236 020040 044122 051105
 1516 010244 004461 000
 1517 010247 120 020103 020040
 1518 010254 020040 052040 051505
 1519 010262 020124 020040 050040
 1520 010270 020103 043117 020040
 1521 010276 053440 051117 020104
 1522 010304 020040 043440 047517
 1523 010312 020104 020040 041040
 1524 010320 042101 020040 020040
 1525 010326 041440 047117 027124
 1526 010334 020040 041440 047117
 1527 010342 027124 020011 041440
 1528 010350 047117 027124 005015
 1529 010356 020040 020040 020040
 1530 010364 020040 047516 020040
 1531 010372 020040 020040 051512
 1532 010400 020122 020040 020040
 1533 010406 047516 020040 020040
 1534 010414 020040 040504 040524
 1535 010422 020040 020040 040504
 1536 010430 040524 020040 020040
 1537 010436 044122 051503 020061
 1538 010444 020040 044122 051504
 1539 010452 020061 020040 044122
 1540 010460 051105 000061
 1541 010464 041520 020040 020040
 1542 010472 020040 042524 052123
 1543 010500 020040 020040 041520
 1544 010506 047440 020106 020040
 1545 010514 047527 042122 020040
 1546 010522 020040 047507 042117
 1547 010530 020040 020040 047503
 1548 010536 052116 020056 020040
 1549 010544 047503 052116 020056

.ASCIZ / NO NO DATA DATA RHCS1 RHDS1 RHER1 /
 DH32: .ASCII /PC TEST PC OF WORD GOOD BAD CONT. CONT. CONT. /
 .ASCIZ / NO JSR NO DATA DATA RHCS1 RHCS1 RHER1 /
 DH33: .ASCII /PC TEST PC OF WORD GOOD CONT. CONT. CONT. <15><12

1604	011233	120	020103	020040	DH36:	.ASCII	/PC	TEST	PC OF	RHMR	POSITON	PATTERN/	<15><12>
1605	011240	020040	052040	051505									
1606	011246	020124	020040	050040									
1607	011254	020103	043117	020040									
1608	011262	051040	046510	020122									
1609	011270	020040	050040	051517									
1610	011276	052111	047117	050040									
1611	011304	052101	042524	047122									
1612	011312	005015											
1613	011314	020040	020040	020040		.ASCIZ	/	NO	JSR	CONT.	REG.	REG./	
1614	011322	020040	047516	020040									
1615	011330	020040	020040	051512									
1616	011336	020122	020040	020040									
1617	011344	047503	052116	020056									
1618	011352	020040	042522	027107									
1619	011360	020040	020040	042522									
1620	011366	027107	000										
1621	011371	120	020103	020040	DH37:	.ASCII	/PC	TEST	POSITON	POSITON	GOOD	GOOD	PATTERN DATA N-CODE/
1622	011376	020040	052040	051505									
1623	011404	020124	020040	050040									
1624	011412	051517	052111	047117									
1625	011420	050040	051517	052111									
1626	011426	047117	043440	047517									
1627	011434	020104	020040	043440									
1628	011442	047517	020104	020040									
1629	011450	050040	052101	042524									
1630	011456	047122	042040	052101									
1631	011464	020101	020040	047040									
1632	011472	041455	042117	006505									
1633	011500	012											
1634	011501	040	020040	020040		.ASCIZ	/	NO	ECC	GOOD	ECC1	ECC2	ECC ENVELOPE ZEROS
1635	011506	020040	047040	020117									
1636	011514	020040	020040	042440									
1637	011522	041503	020040	020040									
1638	011530	043440	047517	020104									
1639	011536	020040	042440	041503									
1640	011544	020061	020040	042440									
1641	011552	041503	020062	020040									
1642	011560	042440	041503	020040									
1643	011566	020040	042440	053116									
1644	011574	047514	042520	055040									
1645	011602	051105	051517	000011									
1646													
1647	011610	041520	020040	020040	DH40:	.ASCII	/PC	TEST	FAILING	CONT.	CONT.	CONT.	CONT.<<15><12>
1648	011616	020040	042524	052123									
1649	011624	020040	020040	040506									
1650	011632	046111	047111	020107									
1651	011640	047503	052116	020056									
1652	011646	020040	047503	052116									
1653	011654	020056	020040	047503									
1654	011662	052116	020056	020040									
1655	011670	047503	052116	006456									
1656	011676	012											
1657	011677	040	020040	020040		.ASCIZ	/	NO	REG ADR	RHCS1	RHCS2	RHCS!	RHER!

1712	012304	001116	017330	052460	DT31:	.WORD	\$ERRPC, TSTNM, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1, 0
1713	012312	001124	001126	015034			
1714	012320	015056	015036	000000			
1715	012326	001116	017330	015134	DT32:	.WORD	\$ERRPC, TSTNM, PCJSR, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1, 0
1716	012334	052460	001124	001126			
1717	012342	015034	015056	015036			
1718	012350	000000					
1719	012352	001116	017330	015134	DT33:	.WORD	\$ERRPC, TSTNM, PCJSR, ERWORD, \$GDDAT, CS1, DS1, ER1, 0
1720	012360	052460	001124	015034			
1721	012366	015056	015036	000000			
1722	012374	001116	017330	050370	DT34:	.WORD	\$ERRPC, TSTNM, GECC1, GECC2, WECC1, WECC2, DISK, 0
1723	012402	050372	055256	055260			
1724	012410	054256	000000				
1725	012414	001116	017330	050370	DT35:	.WORD	\$ERRPC, TSTNM, GECC1, GECC2, EC2, EC1, POSITI, ER1, 0
1726	012422	050372	015066	015064			
1727	012430	050402	015036	000000			
1728	012436	001116	017330	015134	DT36:	.WORD	\$ERRPC, TSTNM, PCJSR, MR, EC1, EC2, 0
1729	012444	015054	015064	015066			
1730	012452	000000					
1731	012454	001116	017330	015064	DT37:	.WORD	\$ERRPC, TSTNM, EC1, POSITI, GECC1, GECC2, EC2, DATENV, ZCODE, 0
1732	012462	050402	050370	050372			
1733	012470	015066	050406	050410			
1734	012476	000000					
1735	012500	001116	017330	001122	DT40:	.WORD	\$ERRPC, TSTNM, \$BDADR, CS1, CS2, DS1, ER1, 0
1736	012506	015034	015032	015056			
1737	012514	015036	000000				
1738							
1739							
1740							
1741							
1742	012520	000	000	000	DF1:	.BYTE	0,0,0,0,0
1743	012523	000	000				
1744	012525	000	000	001	DF2:	.BYTE	0,0,1,0
1745	012530	000					
1746	012531	000	000	001	DF3:	.BYTE	0,0,1,0,0
1747	012534	000	000				
1748							
1749	012536	000	000	000	DF11:	.BYTE	0,0,0,0,0,0,0
1750	012541	000	000	000			
1751	012544	000					
1752	012545	000	000	000	DF14:	.BYTE	0,0,0,0,0,0,0,0
1753	012550	000	000	000			
1754	012553	000	000				
1755	012555	000	000	000	DF15:	.BYTE	0,0,0
1756	012560	000	000	000	DF16:	.BYTE	0,0,0,0,0
1757	012563	000	000				
1758	012565	000	000	000	DF17:	.BYTE	0,0,0,0,0,0,0
1759	012570	000	000	000			
1760	012573	000					
1761	012574	000	000	000	DF20:	.BYTE	0,0,0,0,0,0,0
1762	012577	000	000	000			
1763	012602	000					
1764							
1765	012603	000	000	000	DF21:	.BYTE	0,0,0,0,0

1766	012606	000	000				
1767	012610	000	000	000	DF22:	.BYTE	0,0,0,0
1768	012613	000					
1769	012614	000	000	000	DF24:	.BYTE	0,0,0,0,0,0,0
1770	012617	000	000	000			
1771	012622	000					
1772	012623	000	000	000	DF26:	.BYTE	0,0,0,0,0,0,0,0
1773	012626	000	000	000			
1774	012631	000	000				
1775	012633	000	000	000	DF27:	.BYTE	0,0,0,0,0,0
1776	012636	000	000	000			
1777	012641	000	000	000	DF30:	.BYTE	0,0,0,0,0,0
1778	012644	000	000	000			
1779							
1780	012647	000	000	001	DF31:	.BYTE	0,0,1,0,0,0,0,0
1781	012652	000	000	000			
1782	012655	000	000				
1783	012657	000	000	000	DF32:	.BYTE	0,0,0,1,0,0,0,0,0
1784	012662	001	000	000			
1785	012665	000	000	000			
1786	012670	000	000	000	DF33:	.BYTE	0,0,0,1,0,0,0,0
1787	012673	001	000	000			
1788	012676	000	000				
1789	012700	000	000	000	DF34:	.BYTE	0,0,0,0,0,0,0
1790	012703	000	000	000			
1791	012706	000					
1792	012707	000	000	000	DF35:	.BYTE	0,0,0,0,0,0,0,0
1793	012712	000	000	000			
1794	012715	000	000				
1795	012717	000	000	000	DF36:	.BYTE	0,0,0,0,0,0
1796	012722	000	000	000			
1797	012725	000	000	000	DF37:	.BYTE	0,0,0,0,0,0,0,0,0
1798	012730	000	000	000			
1799	012733	000	000	000			
1800	012736	000	000	000	DF40:	.BYTE	0,0,0,0,0,0,0
1801	012741	000	000	000			
1802	012744	000					
1803							
1804		012746				.EVEN	
1805	012746	044122	030461	051040	EM42:	.ASCIZ	/RH11 REGISTER FAILED TO RESPOND TO A "TST" COMMAND
1806	012754	043505	051511	042524			
1807	012762	020122	040506	046111			
1808	012770	042105	052040	020117			
1809	012776	042522	050123	047117			
1810	013004	020104	047524	040440			
1811	013012	021040	051524	021124			
1812	013020	041440	046517	040515			
1813	013026	042116	000				
1814							
1815	013031	122	030510	020061	EM43:	.ASCIZ	/RH11 ILLEGAL REGISTER RESPONSE TEST/
1816	013036	046111	042514	040507			
1817	013044	020114	042522	044507			
1818	013052	052123	051105	051040			
1819	013060	051505	047520	051516			

1820	013066	020105	042524	052123		
1821	013074	000				
1822						
1823	013075	115	053117	020105	EM44:	.ASCIZ /MOVE BYTE TO WORD COUNT TEST/
1824	013102	054502	042524	052040		
1825	013110	020117	047527	042122		
1826	013116	041440	052517	052116		
1827	013124	052040	051505	000124		
1828						
1829	013132	044502	020123	054502	EM45:	.ASCIZ /BIS BYTE TO WORD COUNT/
1830	013140	042524	052040	020117		
1831	013146	047527	042122	041440		
1832	013154	052517	052116	000		
1833						
1834	013161	102	041511	041040	EM46:	.ASCIZ /BIC BYTE TO WORD COUNT/
1835	013166	052131	020105	047524		
1836	013174	053440	051117	020104		
1837	013202	047503	047125	000124		
1838						
1839	013210	042522	044507	052123	EM47:	.ASCIZ /REGISTER ADDRESS SELECT TEST/
1840	013216	051105	040440	042104		
1841	013224	042522	051523	051440		
1842	013232	046105	041505	020124		
1843	013240	042524	052123	000		
1844						
1845	013245	116	047117	042440	EM50:	.ASCIZ /NON EXISTANT DRIVE (NED) TEST/
1846	013252	044530	052123	047101		
1847	013260	020124	051104	053111		
1848	013266	020105	047050	042105		
1849	013274	020051	042524	052123		
1850	013302	000				
1851						
1852	013303	101	020123	042522	EM51:	.ASCIZ /AS REGISTER TEST/
1853	013310	044507	052123	051105		
1854	013316	052040	051505	000124		
1855						
1856	013324	052502	051523	040440	EM52:	.ASCIZ /BUSS ADDRESS REGISTER DATA TEST/
1857	013332	042104	042522	051523		
1858	013340	051040	043505	051511		
1859	013346	042524	020122	040504		
1860	013354	040524	052040	051505		
1861	013362	000124				
1862						
1863	013364	052502	051523	040440	EM53:	.ASCIZ /BUSS ADDRESS REGISTER MOVE BYTE/
1864	013372	042104	042522	051523		
1865	013400	051040	043505	051511		
1866	013406	042524	020122	047515		
1867	013414	042526	041040	052131		
1868	013422	000105				
1869						
1870	013424	044122	030461	044440	EM54:	.ASCIZ /RH11 INTERRUPT FAILED TO OCCUR/
1871	013432	052116	051105	052522		
1872	013440	052120	043040	044501		
1873	013446	042514	020104	047524		

1874	013454	047440	041503	051125		
1875	013462	000				
1876						
1877	013463	122	051505	052105	EM55:	.ASCIZ /RESET TEST/
1878	013470	052040	051505	000124		
1879						
1880	013476	054115	020106	044502	EM56:	.ASCIZ /MXF BIT TEST/
1881	013504	020124	042524	052123		
1882	013512	000				
1883						
1884	013513	125	044516	052502	EM57:	.ASCIZ /UNIBUS PARITY BIT TEST/
1885	013520	020123	040520	044522		
1886	013526	054524	041040	052111		
1887	013534	052040	051505	000124		
1888						
1889	013542	047504	051505	051440	EM60:	.ASCIZ /DOES SELECTING THE RH11 CLEAR THE UNIT SELECT REGISTER/
1890	013550	046105	041505	044524		
1891	013556	043516	052040	042510		
1892	013564	051040	030510	020061		
1893	013572	046103	040505	020122		
1894	013600	044124	020105	047125		
1895	013606	052111	051440	046105		
1896	013614	041505	020124	042522		
1897	013622	044507	052123	051105		
1898	013630	000				
1899						
1900	013631	104	042517	020123	EM61:	.ASCIZ /DOES TRE GET SET BY UNIBUS PARITY ERROR/
1901	013636	051124	020105	042507		
1902	013644	020124	042523	020124		
1903	013652	054502	052440	044516		
1904	013660	052502	020123	040520		
1905	013666	044522	054524	042440		
1906	013674	051122	051117	000		
1907						
1908	013701	116	047117	042440	EM62:	.ASCIZ /NON EXISTANT DRIVE (NED) FAILED TO SET (TRE)/
1909	013706	044530	052123	047101		
1910	013714	020124	051104	053111		
1911	013722	020105	047050	042105		
1912	013730	020051	040506	046111		
1913	013736	042105	052040	020117		
1914	013744	042523	020124	052050		
1915	013752	042522	000051			
1916						
1917	013756	047516	020116	054105	EM63:	.ASCIZ /NON EXISTANT MEMORY (NEM) FAILED TO SET (TRE)/
1918	013764	051511	040524	052116		
1919	013772	046440	046505	051117		
1920	014000	020131	047050	046505		
1921	014006	020051	040506	046111		
1922	014014	042105	052040	020117		
1923	014022	042523	020124	052050		
1924	014030	042522	000051			
1925						
1926	014034	047520	052122	051440	EM64:	.ASCIZ /PORT SELECT FAILED TO CLEAR/
1927	014042	046105	041505	020124		

1928	014050	040506	046111	042105				
1929	014056	052040	020117	046103				
1930	014064	040505	000122					
1931								
1932	014070	047503	052116	047522	EM65:	.ASCIZ	/CONTROLLER CLEAR FAILED TO CLEAR COMMAND REGISTER/	
1933	014076	046114	051105	041440				
1934	014104	042514	051101	043040				
1935	014112	044501	042514	020104				
1936	014120	047524	041440	042514				
1937	014126	051101	041440	046517				
1938	014134	040515	042116	051040				
1939	014142	043505	051511	042524				
1940	014150	000122						
1941								
1942	014152	042522	042523	042514	EM66:	.ASCIZ	/RESELECT FAILED TO CLEAR COMMAND REGISTER/	
1943	014160	052103	043040	044501				
1944	014166	042514	020104	047524				
1945	014174	041440	042514	051101				
1946	014202	041440	046517	040515				
1947	014210	042116	051040	043505				
1948	014216	051511	042524	000122				
1949								
1950	014224	047111	042524	051122	EM67:	.ASCIZ	/INTERRUPT CAUSED BY ROY!IE BITS SET FAILED TO OCCUR/	
1951	014232	050125	020124	040503				
1952	014240	051525	042105	041040				
1953	014246	020131	051040	054504				
1954	014254	044441	020105	044502				
1955	014262	051524	051440	052105				
1956	014270	043040	044501	042514				
1957	014276	020104	047524	047440				
1958	014304	041503	051125	000				
1959								
1960	014311	111	020105	040506	EM70:	.ASCIZ	/IE FAILED TO CLEAR FOLLOWING AN INTERRUPT/	
1961	014316	046111	042105	052040				
1962	014324	020117	046103	040505				
1963	014332	020122	047506	046114				
1964	014340	053517	047111	020107				
1965	014346	047101	044440	052116				
1966	014354	051105	052522	052120				
1967	014362	000						
1968								
1969	014363	101	020123	042522	EM71:	.ASCIZ	/AS REGISTER WRITE TEST/	
1970	014370	044507	052123	051105				
1971	014376	053440	044522	042524				
1972	014404	052040	051505	000124				
1973								
1974								
1975								
1976	014412	041520	020040	020040	DH42:	.ASCII	/PC TEST ILLEGAL/<15><12>	
1977	014420	020040	042524	052123				
1978	014426	020040	020040	046111				
1979	014434	042514	040507	006514				
1980	014442	012						
1981	014443	040	020040	020040		.ASCIZ	/ NO ADDRESS/	

[Handwritten mark]

1982	014450	020040	047040	020117						
1983	014456	020040	020040	040440						
1984	014464	042104	042522	051523						
1985	014472	000								
1986	014473	120	020103	020040	DH44:	.ASCII	/PC	TEST	REGISTR	CONT CONT/<15><12>
1987	014500	020040	052040	051505						
1988	014506	020124	020040	051040						
1989	014514	043505	051511	051124						
1990	014522	041440	047117	020124						
1991	014530	020040	041440	047117						
1992	014536	006524	012							
1993	014541	040	020040	020040		.ASCIZ	/	NO	ADDRESS	GOOD BAD/
1994	014546	020040	047040	020117						
1995	014554	020040	020040	040440						
1996	014562	042104	042522	051523						
1997	014570	043440	047517	020104						
1998	014576	020040	041040	042101						
1999	014604	000								
2000	014605	120	020103	020040	DH54:	.ASCII	/PC	TEST	REGISTR	CONT/<15><12>
2001	014612	020040	052040	051505						
2002	014620	020124	020040	051040						
2003	014626	043505	051511	051124						
2004	014634	041440	047117	006524						
2005	014642	012								
2006	014643	040	020040	020040		.ASCIZ	/	NO	ADDRESS/	
2007	014650	020040	047040	020117						
2008	014656	020040	020040	040440						
2009	014664	042104	042522	051523						
2010	014672	000								
2011										
2012		014674				.EVEN				
2013										
2014	014674	001116	017330	045440	DT42:	.WORD	\$ERRPC,	TSTNM,	REGADR,	0
2015	014702	000000								
2016	014704	001116	017330	045440	DT44:	.WORD	\$ERRPC,	TSTNM,	REGADR,	\$GDDAT,\$BDDAT,0
2017	014712	001124	001126	000000						
2018	014720	001116	017330	045440	DT54:	.WORD	\$ERRPC,	TSTNM,	REGADR,	\$GDDAT,0
2019	014726	001124	000000							

2020	014732	000	000	000	DF42:	.BYTE	0,0,0
2021	014735	000	000	000	DF44:	.BYTE	0,0,0,0,0
2022	014740	000	000				
2023	014742	000	000	000	DF54:	.BYTE	0,0,0,0
2024	014745	000					

.SBTTL HARDWARE REGISTER BIT DEFINITIONS

:*****
:RH11 REGISTERS
:*****

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

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

:CONTROL AND STATUS REGISTER 2 (RHCS2)

2045 000001
2046 000002
2047 000004
2048 000010
2049 000020
2050 000040
2051 000100
2052 000200
2053 000400
2054 001000
2055 002000
2056 004000
2057 010000
2058 020000
2059 040000
2060 100000

US1= 1
LS2= 2
US4= 4
BAI= 10
PAT= 20
CLR= 40
IR= 100
OR= 200
MPE= 400
MXF= 1000
PGE= 2000
NEM= 4000
NED= 10000
JPE= 20000
WCE= 40000
DLT= 100000

:UNIT SELECT (BIT #0)
:UNIT SELECT (BIT #1)
:UNIT SELECT (BIT #2)
:BUS ADDRESS INCREMENT INHIBIT (BIT #3)
:INVERT PARITY ON MASS BUS TO EVEN (BIT #4)
:CLEAR (BIT #5)
:INPUT READY (BIT #6)
:OUTPUT READY (BIT #7)
:MASS BUS PARITY ERROR (BIT #8)
:MISSED TRANSFER ERROR (BIT #9)
:PROGRAM ERROR (BIT #10)
:NON EXISTANT MEMORY (BIT #11)
:NON EXISTANT DRIVE (BIT #12)
:UNIBUS PARITY ERROR (BIT #13)
:WRITE CHECK ERROR (BIT #14)
:DATA LATE (BIT #15)

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

2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063

2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117

::*****
:RPO4 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	DFS=	1	:DRIVE FORWARD 5"/SEC. (BIT #0)
000002	DF20=	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	PRCG=	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	CTE=	10000	:DRIVE TIMING ERROR (BIT #12)
020000	OPI=	20000	:OPERATION INCOMPLETE (BIT #13)
040000	UNS=	40000	:DRIVE UNSAFE (BIT #14)

```

2118      100000      DCK=      100000      ;DATA CHECK ERROR (BIT 15)
2119
2120      ;MAINTAINABILITY REGISTER (RHMR) (#03)
2121
2122      000001      DMD=      1      ;DIAGINOSTIC MODE (BIT #0)
2123      000002      MCLK=     2      ;MAINTAINABILITY CLOCK (BIT #1)
2124      000004      MINX=     4      ;MAINTAINABILITY INDEX (BIT #2)
2125      000010      MSTCK=    10     ;MAINTAINABILITY SECTOR CLOCK (BIT #3)
2126      000020      MRD=     20     ;MAINTAINABILITY READ (BIT #4)
2127      000040      MWR=     40     ;MAINTAINABILITY WRITE (BIT #5)
2128      000200      DENVL=   200    ;DATA ENVELOPE (BIT #7)
2129      000400      ZER=     400    ;ZERO DETECT (BIT #8)
2130      001000      DTSY=   1000   ;MAINTAINABILITY SYNC DETECTED (BIT #9)
2131
2132      ;ATTENTION SUMMARY PSEUDO-REGISTER (RHAS) (#04)
2133
2134      000001      AT0=      1      ;DEVICE 0 (BIT #0)
2135      000002      AT1=      2      ;DEVICE 1 (BIT #1)
2136      000004      AT2=      4      ;DEVICE 2 (BIT #2)
2137      000010      AT3=     10     ;DEVICE 3 (BIT #3)
2138      000020      AT4=     20     ;DEVICE 4 (BIT #4)
2139      000040      AT5=     40     ;DEVICE 5 (BIT #5)
2140      000100      AT6=    100    ;DEVICE 6 (BIT #6)
2141      000200      AT7=    200    ;DEVICE 7 (BIT #7)
2142
2143
2144
2145
2146
2147      ;DESIRED SECTOR/TRACK ADDRESS REGISTER (RHDST) (#1)
2148      ;EACH BIT IS CALLED BY BIT NUMBER
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161      ;DRIVE TYPE REGISTER (RHDT) (#06)
2162      ;EACH BIT IS CALLED BY BIT NUMBER
2163
2164
2165
2166
2167
2168
2169
2170
2171      ;LOOK-AHEAD REGISTER (RHLA) (#07)
2172
2173      000001      EXT1=     1      ;EXTENSION 1 (BIT #0)
2174      000002      EXT2=     2      ;EXTENSION 2 (BIT #1)
2175      000004      EXT4=     4      ;EXTENSION 3 (BIT #2)
2176      000010      EXT10=    10     ;EXTENSION 4 (BIT #3)
2177      000020      EXT20=    20     ;EXTENSION 5 (BIT #4)
2178      000040      EXT40=    40     ;EXTENSION 6 (BIT #5)
2179      000100      SC1=    100    ;SECTOR COUNT FIELD 0 (BIT #6)
2180      000200      SC2=    200    ;SECTOR COUNT FIELD 1 (BIT #7)
2181      000400      SC4=    400    ;SECTOR COUNT FIELD 2 (BIT #9)

```

```

2172 001000 SC10= 1000 ;SECTOR COUNT FIELD 3 (BIT #9)
2173 002000 SC20= 2000 ;SECTOR COUNT FIELD 4 (BIT #10)
2174 004000 TRK1= 4000 ;TRACK FIELD 1 (BIT #11)
2175 010000 TRK2= 10000 ;TRACK FIELD 2 (BIT #12)
2176 020000 TRK4= 20000 ;TRACK FIELD 3 (BIT #13)
2177 040000 TRK10= 40000 ;TRACK FIELD 4 (BIT #14)
2178 100000 TRK20= 100000 ;TRACK FIELD 5 (BIT #15)
2179
2180 ;ERROR REGISTER #2 (RHER2) (#10)
2181
2182 000001 WCU= 1 ;WRITE CURRENT UNSAFE (BIT #0)
2183 000002 CSF= 2 ;CURRENT SINK FAILURE (BIT #1)
2184 000004 WSL= 4 ;WRITE SELECT UNSAFE (BIT #2)
2185 000010 CSU= 10 ;CURRENT SWITCH UNSAFE (BIT #3)
2186 000020 MSE= 20 ;MOTOR SEQUENCE ERROR (BIT #4)
2187 000040 TDF= 40 ;TRANSITIONS DETECTOR FAILURE (BIT #5)
2188 000100 TUF= 100 ;TRANSITIONS UNSAFE (BIT #6)
2189 000200 FEN= 200 ;FAILSAFE ENABLED (BIT #7)
2190 000400 WRU= 400 ;WRITE READY UNSAFE (BIT #8)
2191 001000 MHS= 1000 ;MULTIPLE HEAD SELECT (BIT #9)
2192 002000 NMS= 2000 ;NO HEAD SELECTION (BIT #10)
2193 004000 IXE= 4000 ;INDEX ERROR (BIT #11)
2194 010000 VU30= 10000 ;30VOLT UNSAFE (BIT #12)
2195 020000 PLU= 20000 ;PLO UNSAFE (BIT #13)
2196 100000 ACU= 100000 ;ACUNSAFE (BIT #15)
2197
2198 ;OFFSET REGISTER (RHOF) (#11)
2199
2200 000001 OF25= 1 ;OFFSET 25 MICRO INCHES (BIT #0)
2201 000002 OF50= 2 ;OFFSET 50 MICRO INCHES (BIT #1)
2202 000004 OF100= 4 ;OFFSET 100 MICRO INCHES (BIT #2)
2203 000010 OF200= 10 ;OFFSET 200 MICRO INCHES (BIT #3)
2204 000020 OF400= 20 ;OFFSET 400 MICRO INCHES (BIT #4)
2205 000040 OF800= 40 ;OFFSET 800 MICRO INCHES (BIT #5)
2206
2207 000200 OFREV= 200 ;OFFSET NEGATIVE (REVERSE) (BIT #7)
2208 002000 HCI= 2000 ;HEADER COMPARE INHIBIT (BIT #10)
2209 004000 ECI= 4000 ;ERROR CORRECTION CODE INHIBIT (BIT #11)
2210 010000 FMT22= 10000 ;FORMAT BIT (BIT #12)
2211
2212
2213
2214
2215
2216 ;DESIRED CYLINDER ADDRESS (RHCA) (#12)
2217 ;EACH BIT IS CALLED BY BIT NUMBER.
2218
2219
2220
2221
2222
2223 ;CURRENT CYLINDER ADDRESS (RHCC) (#13)
2224 ;EACH BIT IS CALLED BY BIT NUMBER
2225

```


2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257

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

:ERROR REGISTER #03 (RHER3) (#15)

000001
000002
000010
000020
000040
000100
040000
100000

PSU= 1
VUF= 2
UWR= 10
PRE= 20
ACL= 40
DCL= 100
SKI= 40000
OCYL= 100000

:PACK SPEED UNSAFE (BIT #0)
:VELOCITY UNSAFE (BIT #1)
:ANY UNSAFE EXCEPT READ/WRITE (BIT #3)
:DISK PACK ROTATION ERROR (BIT #4)
:AC LOW (BIT #5)
:DC LOW (BIT #6)
:SEEK INCOMPLETE (BIT #14)
: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



```

2258
2259
2260
2261          .SBTTL REGISTER ADDRESSES
2262
2263
2264
2265
2266          :RP04/5/6 VECTOR ADDRESS
2267
2268 014746 000254          RPVEC: 254          :RP04/5/6 VECTOR ADDRESS
2269
2270
2271
2272
2273          ;RP04/5/6 DISK I/O REGISTERS LOCATED IN THE RH11 CONTROLLER
2274          ;NOTE: THE CONTENTS OF THESE LOCATIONS WILL BE DIFFERENT
2275          ;       IF THE "CHANGE BASE ADDRESS" ROUTINE IS USED.
2276          ;       THIS ROUTINE STARTS AT LOCATION TAGGED "BASECH"
2277
2278 014750 176722          RHDB: 176722          ;DATA BUFFER SEE NOTE ABOVE
2279 014752 176702          RHWC: 176702          ;WORD COUNT SEE NOTE ABOVE
2280 014754 176704          RHBA: 176704          ;BUS ADDRESS SEE NOTE ABOVE
2281 014756 176710          RHCS2: 176710          ;CONTROL AND STATUS 2 SEE NOTE ABOVE
2282
2283
2284          ;RP04/5/6 DISK I/O REGISTERS LOCATED IN THE DEVICE CONTROL LOGIC (DCL)
2285          ;NOTE: THE CONTENTS OF THESE LOCATIONS WILL BE DIFFERENT
2286          ;       IF THE "CHANGE BASE ADDRESS ROUTINE IS USED.
2287          ;       THIS ROUTINE STARTS AT LOCATION TAGED "BASECH"
2288
2289 014760 176700          RHCS1: 176700          ;CONTROL AND STATUS 1 SEE NOTE ABOVE
2290 014762 176714          RHER1: 176714          ;ERROR #1 SEE NOTE ABOVE
2291 014764 176706          RHDS1: 176706          ;DESIRED SECTOR/TRACK ADDRESS SEE NOTE ABOVE
2292 014766 176740          RHER2: 176740          ;ERROR #2 SEE NOTE ABOVE
2293 014770 176732          RHOF: 176732          ;OFFSET SEE NOTE ABOVE
2294 014772 176734          RHCA: 176734          ;DESIRED CYLINDER ADDRESS SEE NOTE ABOVE
2295 014774 176742          RHER3: 176742          ;ERROR #3 SEE NOTE ABOVE
2296 014776 176716          RHAS: 176716          ;ATTENTION SUMMARY SEE NOTE ABOVE
2297 015000 176724          RHMR: 176724          ;MAINTAINABILITY SEE NOTE ABOVE
2298 015002 176712          RHDS1: 176712          ;DRIVE STATUS SEE NOTE ABOVE
2299 015004 176726          RHDT: 176726          ;DRIVE TYPE SEE NOTE ABOVE
2300 015006 176730          RHSN: 176730          ;SERIAL NUMBER SEE NOTE ABOVE
2301 015010 176744          RHEC1: 176744          ;ECC POSITION SEE NOTE ABOVE
2302 015012 176746          RHEC2: 176746          ;ECC PATTERN SEE NOTE ABOVE
2303 015014 176720          RHLA: 176720          ;LOOK-AHEAD SEE NOTE ABOVE
2304 015016 176736          RHCC: 176736          ;CURRENT CYLINDER ADDRESS SEE NOTE ABOVE
2305
2306          ;ADDITIONAL REGISTERS LOCATED IN THE RH70 CONTROLLER
2307
2308 015020 176752          RHCS3: 176752          ;CONTROL AND STATUS REGISTER #3
2309 015022 176750          RHBAE: 176750          ;BUS ADDRESS EXTENSION REGISTER

```

```

2310
2311      :THE FOLLOWING LOCATIONS ARE RESERVED FOR REGISTER SAVES
2312      :ANY TIME THERE IS AN ERROR ALL THESE WILL BE FILLED
2313      :ONLY SOME MAY BE PRINTED BUT ALL WILL BE FILLED TRUE
2314      :FOR THE TIME JUST AFTER THE "ERROR" ERROR COMMAND
2315
2316 015024 000000      DB:      0      ;DATA BUFFER
2317 015026 000000      WC:      0      ;WORD COUNT
2318 015030 000000      BA:      0      ;BUS ADDRESS
2319 015032 000000      CS2:     0      ;CONTROL AND STATUS 2
2320
2321
2322 015034 000000      CS1:      0      ;CONTROL AND STATUS 1
2323 015036 000000      ER1:      0      ;ERROR #1
2324 015040 000000      DST:      0      ;DESIRED SECTOR/TRACK ADDRESS
2325 015042 000000      ER2:      0      ;ERROR #2
2326 015044 000000      OF:      0      ;OFFSET
2327 015046 000000      CA:      0      ;DESIRED CYLINDER ADDRESS
2328 015050 000000      ER3:      0      ;ERROR #3
2329 015052 000000      AS:      0      ;ATTENTION SUMMARY
2330 015054 000000      MR:      0      ;MAINTAINABILITY
2331 015056 000000      DS1:      0      ;DRIVE STATUS
2332 015060 000000      DT:      0      ;DRIVE TYPE
2333 015062 000000      SN:      0      ;SERIAL NUMBER
2334 015064 000000      EC1:      0      ;ECC POSITION
2335 015066 000000      EC2:      0      ;ECC PATTERN
2336 015070 000000      LA:      0      ;LOOK-AHEAD
2337 015072 000000      CC:      0      ;CURRENT CYLINDER ADDRESS
2338
2339

```

:FLAGS AND INTERNAL PROGRAM CONTROL WORDS

2340					
2341					
2342					
2343					
2344	015074	000010	UNITS: .BLKW	8.	; THIS IS FILLED WITH -1
2345	015114	000000	UNIT: .WORD	0	; UNIT UNDER TEST
2346	015116	000000	NUNIT: .WORD	0	; NUMBER OF UNITS PRESENT
2347					; USED TO KEEP TRACK OF UNIT UNDER TEST
2348	015120	000000	NUNIT: .WORD	0	; USED TO DETERMIN IF THERE ARE MORE
2349					; THAN ONE UNIT
2350	015122	000000	SELECT: .WORD	0	; ALL ONES INDICATE UNIT TO BE SELECTED
2351	015124	000000	UNITSL: .WORD	0	; UNIT NO. SELECTED
2352					
2353	015126	000000	ERFLGS: 0		; ERROR FLAG
2354					
2355	015130	000000	SAVDT: 0		; SAVE DRIVE TYPE REGISTER
2356					; FOR COMPARISON IN DRIVE CLEAR TEST
2357					; AND RH INIT TEST
2358	015132	000000	SAVSN: 0		; SAVE SERIAL NUMBER REGISTER
2359					; FOR COMPARISON IN DRIVE CLEAR TEST
2360					; AND RH INIT TEST
2361					
2362	015134	000000	PCJSR: 0		; SAVE PC OF JSR WHICH GAVE THE ERROR
2363					
2364	015136	000000	ATTENT: 0		; ATTENTION BIT FOR PRESENT UNIT
2365	015140	000000	TOTALAT: 0		; TOTAL ATTENTION BITS
2366					
2367	015142	000000	TMPILL: 0		; TEMPORARY ILLEGAL FUNCTION
2368					
2369	015144	000000	TSECC: 0		; FLAG TO SAY IF ECC TEST OR NOT
2370					; WHEN =177777 IT IS AN ECC TEST
2371					; WHEN =0IT IS NOT AN ECC TEST
2372					
2373	015146	000000	TESDTE: 0		; FLAG TO SAY IF DRIVE TIMING ERROR OR NOT
2374					; WHEN = 177777 IT IS A DTE TEST
2375					; WHEN = 0 IT IS NOT A DTE TEST
2376					
2377	015150	000000	TAGDTE: 0		; TEMPORARY TAG USED IN DRIVE TIMING
2378					; ERROR TEST
2379					

```

2380
2381
2382                ;FUNCTION EQUATES
2383
2384                ;TABLE OF COMMAND FUNCTIONS FOR RHCS1
2385                ;THEN "GO" BIT HAS TO BE SET
2386
2387 015152          FUTABL:
2388 015152 000000    NOPERA: 0                ;NO OPERATION
2389 015154 000002    UNLOAD: 2                ;UNLOAD (STAND BY)
2390 015156 000006    RECALI: 6                ;RECALIBRATE
2391 015160 000010    DCLEAR: 10               ;DRIVE CLEAR
2392 015162 000012    RELEAS: 12               ;RELEASE (DUAL-PORT OPERATION)
2393 015164 000030    SERCH: 30                ;SEARCH COMMAND
2394 015166 000050    WRCHK: 50                ;WRITE CHECK DATA
2395 015170 000052    WRCHDT: 52               ;WRITE CHECK HEADER AND DATA
2396 015172 000060    WRIDAT: 60               ;WRITE DATA
2397 015174 000062    WRIFOR: 62              ;WRITE HEADER AND DATA (FORMAT)
2398 015176 000070    READAT: 70               ;READ DATA
2399 015200 000072    REFOR: 72                ;READ HEADER AND DATA
2400 015202 000004    SEECOM: 4                ;SEEK COMMAND
2401 015204 000014    OFSETC: 14               ;OFFSET COMMAND
2402 015206 000016    RETCL: 16                ;RETURN TO CENTERLINE
2403 015210 000022    PKACK: 22                ;PACK ACKNOWLEDGE
2404 015212 000020    READIN: 20              ;READ IN
2405 015214 000000    ILLEGL: .WORD           ;COMPUTED ILLEGAL FUNCTION
2406
2407
2408
2409                ;DATA BUFFER FOR READ WRITE
2410
2411
2412 015220 000422    WRFROM: .BLKW 274.         ;WRITE FROM THIS BUFFER
2413 016264 000422    REINTO: .BLKW 274.        ;READ INTO THIS BUFFER
2414
2415
2416 017330 000000    TSTNM: 0                  ;TEST NUMBER
2417 017332 000000    FIRST: 0                 ;IF ZERO WILL TYPE HEADER
2418                                     ;IF ONES WILL NOT TYPE HEADER
2419
2420 017334 000000    RH70: 0                    ;FLAG = 1 FOR RH70 CONTROLLEF
2421                                     ;FLAG = 0 FOR RH11
2422
2423
2424
2425
2426                ;TABLE FOR ATTENTION BITS
2427                ;ATTENTION TABLE
2428
2429 017336 001 002 004 ATABLE: .BYTE 1,2,4,10,20,40,100,200
2430 017341 010 020 040
2431 017344 100 200

```

```

2432
2433
2434
2435
2436
2437 017346 012737 177777 015122 BEGIN2: MOV #-1, @#SELECT ;SELECT UNIT
2438 017354 000402 BR START
2439 017356 005037 015122 BEGIN: CLR @#SELECT ;DO NOT SELECT UNIT
2440 ;NORMAL RUN
2441
2442 017362 START:
2443 017362 000005 RESET
2444 .SBTTL INITIALIZE THE COMMON TAGS
2445 ;; CLEAR THE COMMON TAGS ($CMTAG) AREA
2446 017364 012706 001100 MOV #CMTAG, R6 ;; FIRST LOCATION TO BE CLEARED
2447 017370 005026 CLR (R6)+ ;; CLEAR MEMORY LOCATION
2448 017372 022706 001140 CMP #SWR, R6 ;; DONE?
2449 017376 001374 BNE -6 ;; LOOP BACK IF NO
2450 017400 012706 001000 MOV #STACK, SP ;; SETUP THE STACK POINTER
2451 ;; INITIALIZE A FEW VECTORS
2452 017404 012737 057060 000020 MOV #SCOPE, @#IOTVEC ;; IOT VECTOR FOR SCOPE ROUTINE
2453 017412 012737 000340 000022 MOV #340, @#IOTVEC+2 ;; LEVEL 7
2454 017420 012737 061276 000030 MOV #ERROR, @#EMTVEC ;; EMT VECTOR FOR ERROR ROUTINE
2455 017426 012737 000340 000032 MOV #340, @#EMTVEC+2 ;; LEVEL 7
2456 017434 012737 062046 000034 MOV #TRAP, @#TRAPVEC ;; TRAP VECTOR FOR TRAP CALLS
2457 017442 012737 000340 000036 MOV #340, @#TRAPVEC+2 ;; LEVEL 7
2458 017450 012737 062122 000024 MOV #SPWRDN, @#PWRVEC ;; POWER FAILURE VECTOR
2459 017456 012737 000340 000026 MOV #340, @#PWRVEC+2 ;; LEVEL 7
2460 017464 005037 001212 CLR $TIMES ;; INITIALIZE NUMBER OF ITERATIONS
2461 017470 005037 001214 CLR $ESCAPE ;; CLEAR THE ESCAPE ON ERROR ADDRESS
2462 017474 112737 000001 001115 MOVB #1, $ERMAX ;; ALLOW ONE ERROR PER TEST
2463 017502 012737 017502 001106 MOV #., $LPADR ;; INITIALIZE THE LOOP ADDRESS FOR SCOPE
2464 017510 012737 017510 001110 MOV #., $LPERR ;; SETUP THE ERROR LOOP ADDRESS
2465 ;; SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2466 ;; EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
2467 017516 013746 000004 MOV @#ERRVEC, -(SP) ;; SAVE ERROR VECTOR
2468 017522 012737 017556 000004 MOV #64$, @#ERRVEC ;; SET UP ERROR VECTOR
2469 017530 012737 177570 001140 MOV #DSWR, SWR ;; SETUP FOR A HARDWARE SWICH REGISTER
2470 017536 012737 177570 001142 MOV #DDISP, DISPLAY ;; AND A HARDWARE DISPLAY REGISTER
2471 017544 022777 177777 161366 CMP #-1, @SWR ;; TRY TO REFERENCE HARDWARE SWR
2472 017552 001012 BNE 66$ ;; BRANCH IF NO TIMEOUT TRAP OCCURRED
2473 ;; AND THE HARDWARE SWR IS NOT = -1
2474 017554 000403 BR 65$ ;; BRANCH IF NO TIMEOUT
2475 017556 012716 017564 64$: MOV #55$, (SP) ;; SET UP FOR TRAP RETURN
2476 017562 000002 RTI
2477 017564 012737 000176 001140 65$: MOV #SWREG, SWR ;; POINT TO SOFTWARE SWR
2478 017572 012737 000174 001142 MOV #DISPREG, DISPLAY
2479 017600 012637 000004 66$: MOV (SP)+, @#ERRVEC ;; RESTORE ERROR VECTOR
2480
2481
2482
2483
2484 017604 012737 000000 177776 STARTA: MOV #0, PS ;SET PROCESSOR STATUS TO 0
2485 017612 012777 056772 175126 MOV #RPVECT, @RPVEC ;THIS IS FOR UNTIMELY DRIVE INTERRUPTS

```

```

2486 017620 004737 060036 JSR PC, @#STKINT ; INITIALIZE TTY KEYBOARD
2487 017624 005737 017332 TST @#FIRST ; IS THIS FIRST TIME ROUND ?
2488 017630 001001 BNE 1$ ; DON'T TYPE HEADER IF NOT
2489 017632 000402 BR 2$ ; TYPE HEADER IF SO
2490
2491 017634 000137 020662 1$: JMP @#SND1
2492
2493 017640 2$:
2494 017640 104400 017646 TYPE 65$ ; TYPE ASCIZ STRING
2495 017644 000435 BR 64$ ; GET OVER THE ASCIZ
2496 ;:65$: .ASCIZ <15><12>'RPO4/5/6 DISKLESS CONTROLLER TEST - PART II - DZRJH-A?'
2497 017740 64$:
2498
2499 017740 104400 017746 TYPE 67$ ; TYPE ASCIZ STRING
2500 017744 000417 BR 66$ ; GET OVER THE ASCIZ
2501 ;:67$: .ASCIZ <15><12>'REVISION DATE: 21-MAR-76/'<15><12>
2502 020004 66$:
2503
2504 020004 104400 020012 TYPE 69$ ; TYPE ASCIZ STRING
2505 020010 000433 BR 68$ ; GET OVER THE ASCIZ
2506 ;:69$: .ASCIZ <15><12>'ALL DCL'S UNDER TEST MUST BE LOCKED ON CORRECT PORT/'
2507 020100 68$:
2508 020100 104400 020106 TYPE 71$ ; TYPE ASCIZ STRING
2509 020104 000433 BR 70$ ; GET OVER THE ASCIZ
2510 ;:71$: .ASCIZ <15><12>'IF CHANGES ARE REQUIRED ON PORT SWITCH, A CYCLE UP/'
2511 020174 70$:
2512 020174 104400 020202 TYPE 73$ ; TYPE ASCIZ STRING
2513 020200 000436 BR 72$ ; GET OVER THE ASCIZ
2514 ;:73$: .ASCIZ <15><12>'SEQUENCE IS REQUIRED FOR STROBING THE PORT SELECT FLOP/'<15><12>
2515 020276 72$:
2516 020276 104400 020304 TYPE 75$ ; TYPE ASCIZ STRING
2517 020302 000431 BR 74$ ; GET OVER THE ASCIZ
2518 ;:75$: .ASCIZ <15><12>'ALL DCL'S NOT UNDER TEST MUST BE SWITCHED OFF/'<15><12>
2519 020366 74$:
2520 020366 104400 020374 TYPE 77$ ; TYPE ASCIZ STRING
2521 020372 000427 BR 76$ ; GET OVER THE ASCIZ
2522 ;:77$: .ASCIZ <15><12>'*****'/
2523 020452 76$:
2524 020452 104400 020460 TYPE 79$ ; TYPE ASCIZ STRING
2525 020456 000427 BR 78$ ; GET OVER THE ASCIZ
2526 ;:79$: .ASCIZ <15><12>'IF THIS IS NOT DONE, ERRORS WILL RESULT ON/'
2527 020536 78$:
2528 020536 104400 020544 TYPE 81$ ; TYPE ASCIZ STRING
2529 020542 000415 BR 80$ ; GET OVER THE ASCIZ
2530 ;:81$: .ASCIZ <15><12>'NED' TESTS (T21 & T36)/'
2531 020576 80$:
2532 020576 104400 020604 TYPE 83$ ; TYPE ASCIZ STRING
2533 020602 000427 BR 82$ ; GET OVER THE ASCIZ
2534 ;:83$: .ASCIZ <15><12>'*****'/
2535 020662 82$:
2536
2537 020662 012737 177777 017332 SND1: MOV #-1, @#FIRST ; NEXT TIME DO NOT GIVE HEADER
2538 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
2539

```

```

2540 020670 005737 000042          TST      @#42          ;; ARE WE RUNNING UNDER XXDP/ACT?
2541 020674 001006                   BNE     64$          ;; BRANCH IF YES
2542 020676 023727 001140 000176    CMP     SWR, #SWREG ;; SOFTWARE SWITCH REG SELECTED?
2543 020704 001005                   BNE     65$          ;; BRANCH IF NO
2544 020706 104405                   GTSWR                    ;; GET SOFT-SWR SETTINGS
2545 020710 000403                   BR      65$
2546 020712 112737 000001 001134 64$:  MOVB   #1, $AUTOB    ;; SET AUTO-MODE INDICATOR
2547 020720 65$:
2548
2549 020720 032777 010000 160212 RH70CK: BIT   #SW12, @SWR      ; LOOK TO SEE IF USING RH70
2550 020726 001403                   BEQ    3$           ; IF SW12 = 0, SKIP NEXT
2551 020730 012737 177777 017334    MOV    #-1, @RH70  ; IF SW12 = 1, CU IS AN RH70
2552
2553 020736 005737 015122          3$:   TST      @#SELECT      ; WAS IT A 200 START
2554 020742 001434                   BEQ    TST1        ; GO TO FIRST TEST IF STARTING FROM 200 -----)
2555 020744 104400 020752                   TYPE  65$         ;; TYPE ASCIZ STRING
2556 020750 000422                   BR     64$         ;; GET OVER THE ASCIZ
2557                               ;; 65$: .ASCIZ <15><12>/SELECT UNIT NUMBER TO BE TESTED ?/
2558 021016 64$:
2559 021016 104411                   RDOCT
2560 021020 042716 177770          BIC    #177770, (SP) ; ONLY KEEP LAST 3 BITS
2561 021024 011637 015114          MOV    (SP), @#UNIT ; SAVE UNIT TO BE TESTED
2562 021030 012637 015124          MOV    (SP)+, @#UNITSL ; SAVE UNIT TO BE TESTED
2563
2564
2565
  
```



```

2566
2567      ;:*****
2568      ;*TEST 1      REFERENCE EACH REGISTER
2569
2570      ;*
2571      REFERENCE EACH REGISTER BY A MOVE INSTRUCTION
2572      ;:*****
2573 021034 000004      †ST1: SCOPE
2574 021036 012737 000001 001212      MOV #1,$TIMES      ;:DO 1 ITERATION
2575 021044 012706 001000      MOV #STACK,$SP      ;:SET UP STACK POINTER
2576 021050 012737 000001 017330      MOV #TTNO,$TSTNM      ;:THIS SAVES TEST NUMBER
2577 021056 012737 061306 000030      MOV #REGSA1,$EMTVEC ;:ERROR VECTOR SO THAT
2578      ;:NO REGISTERS ARE SAVED
2579 021064 012737 021112 000004      MOV #2,$ERRVEC      ;:SET UP FOR BUS TIMEOUT
2580 021072 012700 000024      MOV #24,$R0      ;:THERE ARE 24 REG TO TEST
2581 021076 012701 014750      MOV #RH0B,$R1      ;:R1 NOW HAS ADDR OF ADDR OF FIRST REG.
2582 021102 013102      1$: MOV @($R1)+,$R2      ;:READ HARDWARE REG.
2583 021104 005300      DEC $R0      ;:COUNT DOWN
2584 021106 001375      BNE 1$      ;:BRANCH IF 24 NOT DONE
2585 021110 000471      BR 3$      ;:BRANCH IF 24 DONE
2586 021112 012737 000006 000004 2$: MOV #ERRVEC+2,$ERRVEC ;:RESTORE TRAP CATCHER
2587 021120 022626      CMP ($SP)+,$($SP)+ ;:CLEAN STACK
2588 021122 016137 177776 001200      MOV -2($R1),$STMP1 ;:STORE FAILING REG ADDR
2589 021130 104015      ERROR 15      ;:REGISTER NON EXISTANT
2590 021132 032777 020000 160000      BIT #SW13,$SWR      ;:INHIBIT ERROR PRINTOUT ?
2591 021140 001053      BNE 4$      ;:BRANCH IF YES
2592 021142 104400 021150      TYPE 65$      ;:TYPE ASCIZ STRING
2593 021146 000431      BR 64$      ;:GET OVER THE ASCIZ
2594      ;:65$: .ASCIZ <15><12>/IF BASE ADDRESS IS TO BE CHANGED HALT PROGRAM /
2595      64$:
2596 021232      TYPE 67$      ;:TYPE ASCIZ STRING
2597 021236 104400 021240      BR 66$      ;:GET OVER THE ASCIZ
2598      ;:67$: .ASCIZ <15><12>/AND RESTART AT /
2599      66$:
2600 021262 012746 051320      MOV #BASECH,-($SP) ;:GET READY TO TYPE STARTING ADDRESS
2601      ;:OF "CHANGE OF BASE ADDRESS" ROUTINE
2602 021266 104401      TYPOC
2603 021270 000137 044470      4$: JMP @,$SEOP      ;:GO TO END OF PROGRAM ----->
2604
2605 021274 012737 061276 000030 3$: MOV #,$ERROR,$EMTVEC ;:RESTORE ERROR VECTOR
2606      ;:SO THAT REGISTERS ARE SAVED
2607 021302 012737 000006 000004      MOV #ERRVEC+2,$ERRVEC ;:RESTORE TRAP CATCHER
2608
2609
2610

```

```

2611
2612
2613
2614
2615
2616
2617
2618
2619 021310 000004
2620 021312 012737 000001 001212
2621 021320 012706 001000
2622 021324 012737 000002 017330
2623
2624
2625
2626 021332 005737 017334
2627 021336 001402
2628 021340 000137 021366
2629 021344
2630
2631 021344 013737 014756 021360
2632 021352 004537 045442
2633 021356 020017
2634 021360 000000
2635 021362 104001
2636 021364 000207
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646 021366 000004
2647 021370 012737 000001 001212
2648 021376 012737 000003 017330
2649
2650 021404 013701 014776
2651 021410 012711 177777
2652 021414 011137 001126
2653 021420 105737 001126
2654 021424 001405
2655 021426 005037 001124
2656 021432 010137 045440
2657 021436 104001
2658
2659

::*****
:*TEST 2          RHCS2-CONTROL AND STATUS 2
:*
:* THIS PARTIALLY TESTS RHCS2 TO ENABLE DETERMINATION
:* OF THE NUMBER OF DRIVES PRESENT
::*****
TST2: SCOPE
MOV      #1,STIMES          ;;DO 1 ITERATION
MOV      #STACK,SP         ;;RESET STACK
MOV      #TTNO,#TSTNM      ;;THIS SAVES TEST NUMBER

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

TST      #RH70             ;;TEST FOR RH70 CONTROLLER
BEQ      30$              ;;IF FLAG = 1, SKIP THIS TEST
JMP      TST3             ;;JUMP TO NEXT TEST -----)
30$:
;IF FLAG = 0, DO THIS TEST

MOV      #RHCS2,#UN+2
JSR      R5,#BITST        ;;TEST BITS IN REGISTER
UN:      20017             ;;ONLY THESE BITS ARE TEST READ/WRITE
        .WORD 0           ;;ADDRESS OF REG. BEING TESTED
        ERROR 1          ;;IN CORRECT DATA RECEIVED
        RTS  PC          ;;RETURN TO BLT3 ROUTINE

::*****
:*TEST 3          PARTIAL TEST OF RHAS FOR UNIT NUMBERS PRESENT
::*****
TST3: SCOPE
MOV      #1,STIMES          ;;DO 1 ITERATION
MOV      #TTNO,#TSTNM      ;;THIS SAVES TEST NUMBER

MOV      #RHAS,R1         ;;R1 HAS ADDRESS OF RHAS
MOV      #-1,#R1          ;;THIS CLEARS RHAS (SURPRISED!)
MOV      #R1,#$BDDAT      ;;TEST DATA
TSTB    #BDDAT
BEQ      TST4             ;;BRANCH IF GOOD
CLR      #SGDDAT          ;;GOOD DATA
MOV      R1,#REGADR       ;;FAILING REG. RHAS
ERROR 1                    ;;RHAS DOES NOT CLEAR
                        ;;WITH ONES MOVED INTO IT

```

```

2660
2661
2662
2663
2664 021440 000004
2665 021442 012737 000001 001212
2666 021450 000005
2667 021452 004737 060036
2668
2669 021456 032777 020000 157454
2670 021464 001026
2671 021456 104400 021474
2672 021472 000423
2673
2674 021542
2675 021542 013701 014776
2676 021546 013702 014756
2677 021552 005012
2678 021554 012700 000010
2679 021560 013704 014762
2680
2681 021564 012714 177777
2682 021570 005212
2683 021572 005300
2684 021574 001373
2685 021576 111137 015140
2686
2687 021602 105037 015141
2688 021606 105711
2689 021610 001402
2690 021612 000137 022202
2691
2692 021616 032777 020000 157314
2693 021624 001402
2694 021626 000137 022540
2695
2696
2697 021632
2698 021632 104400 021640
2699 021636 000412
2700
2701 021664
2702 021664 104400 021672
2703 021670 000436
2704
2705 021766
2706 021766 104400 021774
2707 021772 000441
2708
2709 022076
2710 022076 104400 022104
2711 022102 000435
2712
2713 022176

```

```

*****
:TEST 4 TEST FOR DRIVES PRESENT USING RHAS AND RHCS2
*****
TST4: SCOPE
MOV #1,STIMES ;;DO 1 ITERATION
RESET ;;START WITH AN INIT
JSR PC,@#STKINT ;;INITILIZE TTY KEYBOARD
BIT #SW13,@SWR ;;INHIBIT ERROR TYPEOUT ?
BNE 45 ;;SKIP NEXT IF 50
TYPE ,655 ;;TYPE ASCIZ STRING
BR 645 ;;GET OVER THE ASCIZ
;;655: .ASCIZ <15><12><15><12>/LOOKING AT RHAS - DRIVES PRESENT/
645: MOV @RHAS,R1 ;R1 HAS ADDR. OF RHAS
MOV @RHCS2,R2 ;R2 HAS ADDR. OF RHCS2
CLR @R2 ;CLEAR RHCS2
MOV #8,R0 ;COUNT
MOV @RHER1,R4 ;R4 HAS ADDR. OF RHER1
15: MOV #-1,@R4 ;MOVE ERRORS INTO RHER1
INC @R2 ;INCREMENT UNIT NO.
DEC R0 ;COUNT
BNE 15 ;BRANCH IF 8 NOT DONE
MOVB @R1,@TOTALAT ;SAVE TOTAL ATTENTION
;USED IN DRIVE CLEAR TEST
CLRB @TOTALAT+1 ;CLEAR UPPER BYTE
TSTB @R1 ;TEST FOR ANY DRIVES PRESENT
BEQ 25 ;NONE RESPONDING - TYPE THE MESSAGE
JMP XE2 ;SOME THERE - GO FILL "UNITS" TABLE
25: BIT #SW13,@SWR ;INHIBIT ERROR TYPE OUT?
BEQ 35 ;"NO DRIVES" MESSAGE IF NO
JMP SELTST ;CHECK FOR SELECTED UNIT START AND LOAD
;"UNITS" TABLE WITH DESIRED DRIVE IF SO
35: TYPE ,675 ;;TYPE ASCIZ STRING
BR 665 ;;GET OVER THE ASCIZ
;;675: .ASCIZ <15><12>/NO DRIVES-RHAS=0/
665: TYPE ,695 ;;TYPE ASCIZ STRING
BR 685 ;;GET OVER THE ASCIZ
;;695: .ASCIZ <15><12>/WRITING ONES INTO ERROR REGISTER #1 FOR ALL UNIT NUMBERS/
685: TYPE ,715 ;;TYPE ASCIZ STRING
BR 705 ;;GET OVER THE ASCIZ
;;715: .ASCIZ <15><12>/DOES NOT SET ANY BIT IN THE ATTENTION REGISTER SO ABORT PROGRAM
705: TYPE ,735 ;;TYPE ASCIZ STRING
BR 725 ;;GET OVER THE ASCIZ
;;735: .ASCIZ <15><12>/TO LOOP ON THIS TEST W0 PRINTOUT SET SWITCHES 13, 8 & 2
725:

```

```

2714
2715 022176 000137 044470 JMP 3#SEOP ;GO OUT-----,
2716
2717
2718 ;*SET UP UNITS TABLE
2719
2720 022202 XE2:
2721 022202 012700 000010 2$: MOV #8, R0 ;COUNTER
2722 022206 012703 015074 MOV #UNITS, R3 ;POINTER
2723 022212 012723 177777 3$: MOV #-1, (R3)+ ;PRESET BLOCK TO ALL ONES
2724 022216 005300 DEC R0 ;COUNT
2725 022220 001374 BNE 3$ ;BRANCH IF 8 NOT DONE
2726 022222 012703 015074 MOV #UNITS, R3 ;POINTER
2727 022226 005005 CLR R5 ;NO. OF UNITS PRESENT
2728 022230 005037 015116 CLR #NOUNIT ;COUNTER
2729 022234 012700 000010 MOV #8, R0 ;COUNTER
2730 022240 011137 001176 MOV #R1, #STMPD ;TEMPORARY STORAGE
2731 022244 006037 001176 4$: ROR #STMPD ;SET CARRY IF ONE IN 0 BIT
2732
2733 022250 103120 BCC 5$
2734 022252 010577 172500 MOV R5, #RHCS2 ;INSERT UNIT NUMBER
2735 022256 022777 024020 172520 CMP #24020, #RHDT ;IS THIS A DUAL PORT RPO4 ?
2736 022264 001503 BEQ 6$ ;TYPE DRIVE NO. IF SO
2737 022266 022777 020020 172510 CMP #20020, #RHDT ;IS THIS A SINGLE PORT RPO4 ?
2738 022274 001477 BEQ 6$ ;TYPE DRIVE NO. IF YES
2739
2740
2741 ;*****
2742 022276 022777 024021 172500 CMP #24021, #RHDT ;IS THIS A DUAL PORT RPO5 ?
2743 022304 001473 BEQ 6$ ;TYPE UNIT NO. OUT
2744 022306 022777 020021 172470 CMP #20021, #RHDT ;IS THIS A SINGLE PORT RPO5 ?
2745 022314 001467 BEQ 6$ ;TYPE UNIT NO. IF SO
2746
2747 022316 022777 024022 172460 CMP #24022, #RHDT ;IS THIS A DUAL PORT RPO6 ?
2748 022324 001463 BEQ 6$ ;TYPE THE NO IF SO
2749 022326 022777 020022 172450 CMP #20022, #RHDT ;IS THIS A SINGLE PORT RPO6 ?
2750 022334 001457 BEQ 6$ ;TYPE THE NO IF SO
2751 ;*****
2752
2753
2754 ;*NO...IT'S NOT AN RPO4/RPO5/RPO6 DEVICE
2755 ;*SO TYPE OUT THE DEVICE TYPE
2756
2757 022336 104400 022344 TYPE 65$ ;:TYPE ASCIZ STRING
2758 022342 000410 BR 64$ ;:GET OVER THE ASCIZ
2759 ;:65$: .ASCIZ <15><12>/UNIT NUMBER /
2760 64$:
2761 022364 010546 MOV R5, -(SP) ;GET READY TO TYPE UNIT NUMBER
2762 022366 104404 TYPDS
2763 022370 104400 022376 TYPE 67$ ;:TYPE ASCIZ STRING
2764 022374 000406 BR 66$ ;:GET OVER THE ASCIZ
2765 ;:67$: .ASCIZ /, RHDT = /
2766 66$:
2767 022412 017746 172366 MOV #RHDT, -(SP) ;GET READY TO TYPE RHDT

```

```

2768 022416 104401          TYP0C
2769 022420 104400 022426  TYPE      69$      ;;TYPE ASCIZ STRING
2770 022424 000422          BR        68$      ;;GET OVER THE ASCIZ
2771          ;;69$: .ASCIZ  ? - NOT AN RPO4/RPO5/RPO6 DEVICE !!?
2772          68$:
2773 022472 000407          BR        5$      ;NO RPO4/5/6 FOUND SO BRANCH
2774 022474 010523          6$: MOV     RS,(R3)+
2775 022476 104400 001223  TYPE     $CRLF
2776 022502 010546          MOV     RS,-(SP)  ;PUT DRIVE NO. ON STACK
2777 022504 104404          TYPDS
2778 022506 005237 015116  INC      @#NUNIT  ;TYPE DRIVE NO.
;INCR TOTAL NO. OF UNITS
2779
2780 022512 005205          5$: INC     RS      ;'RHCS2' UNIT ADDRESS
2781 022514 005300          DEC     RD      ;DRIVE COUNTER DOWN ONE
2782 022516 001252          BNE     4$      ;TEST AND DO NEXT UNIT IF 8 NOT DONE
2783
2784 022520 013737 015074 015114  MOV     @#UNITS,@#UNIT ;SET UNIT NO. TO FIRST ONE FOUND/CR 0
2785 022526 013737 015116 015120  MOV     @#NUNIT,@#NUNIT ;SAVE NO. OF UNITS
2786 022534 005337 015120          DEC     @#NUNIT  ;IF NUNIT = 0 THEN ONLY ONE UNIT
2787          ;IF NUNIT > 0 THEN MORE THAN ONE UNIT
2788
2789 022540 005737 015122          SELTST: TST    @#SELECT ;STARTING ADDRESS 200 ?
2790 022544 001403          BEQ
2791 022546 013737 015124 015114  MOV     @#UNITSL,@#UNIT ;BRANCH IF STARTING FROM 200
;CHANGE UNIT NUMBER TO SELECTED ONE
2792

```

```

2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803 022554 000004
2804 022556 012737 000001 001212
2805 022564 012737 023234 001106
2806
2807 022572 004737 045674
2808 022576 005037 015136
2809
2810
2811
2812 022602 005737 015114
2813 022606 001022
2814 022610 012700 000041
2815 022614 122710 000011
2816 022620 001015
2817 022622 005737 015122
2818
2819 022626 001012
2820
2821
2822
2823
2824 022630 012700 015074
2825 022634 005720
2826
2827 022636 022710 177777
2828 022642 001404
2829 022644 011037 015114
2830 022650 005337 015116
2831 022654 013700 015114
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846

```

```

;*****
;*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
;*****
TSTS: SCOPE
MOV #1,SIMES ;DO 1 ITERATION
MOV #1$,SLPADR ;SET SCOPE LOOP ADDRESS
JSR PC,@CLDISK ;FILL UNIT NO.
CLR @ATTENT ;CLEAR
;
; *TEST FOR UNIT #0
TST @UNIT ;IS UNIT #0 NEXT IN THE UNITS TABLE ?
BNE 10$ ;IF NOT, TEST THIS UNIT
MOV #41,RO ;IF SO, CHECK THE LOAD MEDIA LOCATION
CMPB #11,(RO) ;WAS IS AN RPO4/5/6 ?
BNE 10$ ;NO... GO AHEAD AND TEST UNIT #0
TST @SELECT ;WAS UNIT #0 SELECTED ?
; (IE. WAS IT A 210 START ?)
BNE 10$ ;IF SO...TEST UNIT #0
;
; *INCREMENT THE UNITS TABLE TO NEXT DRIVE (IF ANY)
; * & DECREMENT THE "NOUNITS" PRESENT (TO BE TESTED)
MOV #UNITS,RO ;LOAD UNITS TABLE POINTER
TST (RO)+ ;SELECT THE NEXT UNIT IN THE TABLE
; (DOUBLE INCREMENT THE POINTER)
CMP #-1,(RO) ;IS THERE ANOTHER TABLE ENTRY PRESENT ?
BEQ 10$ ;IF NOT (LOC = -1) ... MUST USE UNIT #0
MOV (RO),@UNIT ;SET UP TO BE THE UNIT UNDER TEST
DEC @NOUNITS ;DECREMENT BECAUSE UNIT #0 WON'T BE TESTED
10$: MOV @UNIT,RO ;RO CONTAINS THE UNIT UNDER TEST
;
;*****
CLR @RP06 ;CLEAR RP06 DEVICE TYPE FLAG
CMP #24022,@RHDT ;DUAL PORT RP06 ?
BEQ 2$ ;YES...SET THE FLAG
CMP #20022,@RHDT ;SINGLE PORT RP06 ?
BEQ 2$ ;YES...SET FLAG
BR 3$ ;DON'T SET THE RP06 FLAG
2$: MOV #-1,@RP06 ;SET THE FLAG
;
3$: ;ASSUME THE NEXT UNIT IS AN RPO4
;*****

```

```

2847
2848 022660 116037 017336 015136      MOVB  ATABLE(RO),@*ATTENT  ;SET APPROPRIATE ATTENTION BIT
2849 022666 104400 022674      TYPE  65$                ;:TYPE ASCIZ STRING
2850 022672 000414      BR    64$                ;:GET OVER THE ASCIZ
2851      ;:65$: .ASCIZ  <15><12>/TESTING DRIVE NUMBER/
2852 022724      64$:
2853 022724 013746 015114      MOV   @*UNIT,-(SP)       ;:UNIT NO. TO STACK
2854 022730 104404      TYPDS                ;:TYPE DRIVE NO.
2855 022732 104400 022740      TYPE  67$                ;:TYPE ASCIZ STRING
2856 022736 000410      BR    66$                ;:GET OVER THE ASCIZ
2857      ;:67$: .ASCIZ  <15><12>/SERIAL NO. = /
2858 022760      66$:
2859 022760 017746 172022      MOV   @RHSN,-(SP)       ;:SAVE @RHSN FOR TYPEOUT
2860 022764 104401      TYPOC                ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
2861 022766 104400 022774      TYPE  69$                ;:TYPE ASCIZ STRING
2862 022772 000410      BR    68$                ;:GET OVER THE ASCIZ
2863      ;:69$: .ASCIZ  <15><12>/DRIVE TYPE = /
2864 023014      68$:
2865 023014 017746 171764      MOV   @RHDT,-(SP)       ;:SAVE @RHDT FOR TYPEOUT
2866 023020 104401      TYPOC                ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
2867
2868
2869
2870
2871
2872      ;:*****
2873 023022 022777 024020 171754      CMP   #24020,@RHDT      ;:DUAL PORT RPO4 ?
2874 023030 001425      BEQ   4$                ;:TYPE ASCII MSG OUT
2875 023032 022777 020020 171744      CMP   #20020,@RHDT      ;:SINGLE PORT RPO4 ?
2876 023040 001421      BEQ   4$                ;:TYPE THE MESSAGE
2877
2878 023042 022777 024021 171734      CMP   #24021,@RHDT      ;:DUAL PORT RPO5 ?
2879 023050 001434      BEQ   5$                ;:TYPE THE MESSAGE
2880 023052 022777 020021 171724      CMP   #20021,@RHDT      ;:SINGLE PORT RPO5 ?
2881 023060 001430      BEQ   5$                ;:TYPE THE MESSAGE
2882
2883 023062 022777 024022 171714      CMP   #24022,@RHDT      ;:DUAL PORT RPO6 ?
2884 023070 001443      BEQ   6$                ;:TYPE THE MESSAGE
2885 023072 022777 020022 171704      CMP   #20022,@RHDT      ;:SINGLE PORT RPO6 ?
2886 023100 001437      BEQ   6$                ;:TYPE IT OUT
2887 023102 000454      BR    1$                ;:DRIVE IS NOT RPO4/5/6 - SO
2888      ;:DO NOT TYPE ANY MESSAGE OUT
2889
2890      ;:-SHOULD NEVER HAPPEN AT THIS POINT
2891      ;:UNLESS DRIVE GOT SICK WHILE TESTING
2892      ;:WAS IN PROGRESS
2893
2894 023104      4$:
2895 023104 104400 023112      TYPE  71$                ;:TYPE ASCIZ STRING
2896 023110 000413      BR    70$                ;:GET OVER THE ASCIZ
2897      ;:71$: .ASCIZ  <15><12>/DRIVE IS AN RPO4/<15><12>
2898 023140      70$:
2899 023140 000435      BR    1$                ;:SKIP NEXT ONES
2900 023142      5$:

```

H09

```

2901 023142 104400 023150      TYPE      73$      ;;TYPE ASCIZ STRING
2902 023146 000413              BR        72$      ;;GET OVER THE ASCIZ
2903              ;;73$: .ASCIZ <15><12>/DRIVE IS AN RP05/<15><12>
2904 023176              72$:
2905 023176 000416              BR        1$      ;SKIP NEXT
2906 023200              6$:
2907 023200 104400 023206      TYPE      75$      ;;TYPE ASCIZ STRING
2908 023204 000413              BR        74$      ;;GET OVER THE ASCIZ
2909              ;;75$: .ASCIZ <15><12>/DRIVE IS AN RP06/<15><12>
2910 023234              74$:
2911              ;;*****
2912
2913 023234 005777 171546      1$:  TST      @RHSN      ;READ SERIAL NO. AND DRIVE TYPE
2914 023240 005777 171540      TST      @RHDT      ;THESE TWO ARE TO HELP SCOPE LOOPS
2915 023244 017737 171536 015132  MOV      @RHSN,@#SAVSN ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
2916 023252 017737 171526 015130  MOV      @RHDT,@#SAVDT ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS

```

A-11

2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954

023260 000004
023262 012737 000006 017330
023270 004737 045674
023274 032713 010000
023300 001550
023302 104400 023310
023306 000421
023352
023352 104400 023360
023356 000424
023430
023430 104400 023436
023434 000430
023516
023516 032713 010000
023522 001375
023524 104400 023532
023530 000434
023622

```
*****
*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
*****
TST6:  SCOPE
      MOV      #TTNO,2#TSTNM ;THIS SAVES TEST NUMBER
      JSR      PC,2#CLDISK  ;GIVE INITILIZE
      BIT      #MOL,2R3     ;CHECK MOL IN RHDS1
      BEQ      TST7         ;BRANCH IF MOL LOW
      TYPE     65$          ;;TYPE ASCIZ STRING
      BR      64$          ;;GET OVER THE ASCIZ
65$:  .ASCIZ  <15><12>/DRIVE IS ON LINE - MOL IS HIGH/
      TYPE     67$          ;;TYPE ASCIZ STRING
      BR      66$          ;;GET OVER THE ASCIZ
67$:  .ASCIZ  <15><12>/HIT STOP ON DRIVE TO GET IT OFF LINE/
      TYPE     69$          ;;TYPE ASCIZ STRING
      BR      68$          ;;GET OVER THE ASCIZ
69$:  .ASCIZ  <15><12>/PROGRAM WILL HANG TESTING MOL TILL MOL IS LOW/
      BIT      #MOL,2R3     ;CHECK MOL IN RHDS1
      BNE     1$           ;BRANCH IF MOL IS HIGH
      TYPE     71$          ;;TYPE ASCIZ STRING
      BR      70$          ;;GET OVER THE ASCIZ
71$:  .ASCIZ  <15><12>/GOOD - MOL IS NOW LOW . PROGRAM WILL NOW BE EXECUTED/
70$:
```

2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008

: *TEST 7 PACK ACKNOWLEDGE COMMAND TEST

: * THE PACK ACKNOWLEDGE COMMAND WILL BE LOADED INTO RHCSI WITH GO
: * THEN ALL REGISTERS WILL BE CHECKED
: * RH CLEAR WILL BE GIVEN
: * THEN ALL REGISTERS WILL BE CHECKED

: *ST7: SCOPE

MOV #STACK, SP ; RESET STACK
MOV #TTNO, @#TSTNM ; THIS SAVES TEST NUMBER
JSR PC, @#CLDISK ; INIT AND SET UP GENERAL REG.
; AND UNIT NUMBER
MOV #DMD, @RHMR ; SET DIAGNOSTIC MODE
MOV @#PKACK, @RHCSI ; LOAD PACK ACKNOWLEDGE COMMAND INTO RHCSI

; SAVE REGISTERS FOR COMPARISON AFTER GO
JSR RO, @#SAVER ; SAVE
RHW ; FROM
REINTO ; TO
19. ; NUMBER OF REGISTERS SAVED

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

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

; AFTER GO HAS BEEN GIVEN TO PACK ACKNOWLEDGE COMMAND
; SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
; BE DONE
JSR RO, @#SAVER ; SAVE
RHW ; 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
MOVB @#REINTO+25, @#WRFROM+25 ; SAVE UPPER RHAS

; COMPARE REGISTERS BEFORE PACK ACKNOWLEDGE COMMAND
; WITH AFTER GO
JSR RO, @#COMPAR ; COMPARE
REINTO ; GOOD BUFFER
WRFROM ; TEST BUFFER
19. ; NUMBER
1\$; RETURN FOR ERROR
1\$; SAME

```

3009 023742 023764          2$          ;RETURN FOR GOOD COMPARISON
3010 023744 013705 052460 1$: MOV      @#ERWORD,R5 ;GETTING READY TO INDEX
3011 023750 060505          ADD      R5,R5 ;DOUBLE ERROR WORD
3012 023752 016537 014750 045440 MOV     RHC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
3013
3014 023760 104001          ERROR 1 ;IMPROPER REGISTER CHANGE
3015 ;AFTER PACK ACKNOWLEDGE COMMAND
3016 ;WITH GO IS GIVEN
3017 023762 000207          RTS      PC ;RETURN TO COMPARISION
3018
3019 023764          2$:
3020
3021 ;:*****
3022 ;:TEST 10 MAKE CURRENT CYLINDER = 0
3023 ;:*****
3024 023764 000004          ↑ST10: SCOPE
3025 023766 012706 001000 MOV     #STACK,SP ;RESET STACK
3026 023772 012737 000010 017330 MOV     #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
3027 024000 004737 045674 JSR     PC,@#CLDISK ;INIT DRIVE
3028 024004 012777 000001 170766 MOV     #DMD,@#RHMR ;SET DIAGNOSTIC MODE
3029 024012 004037 050246 JSR     RD,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK
3030 ;COMMAND FOLOWED BY AN INIT
3031 024016 000000          0 ;THIS SHUOLD CHANGE RHCC TO 0
3032
3033
3034 .SBTTL
3035 .SBTTL ***DIAGNOSTIC CODE***
3036 .SBTTL

```

```

3037      000004
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049 024020 000004
3050 024022 012737 000001 001212
3051 024030 012737 177777 017330
3052 024036 012777 000040 170712
3053
3054 024044 012705 014760
3055
3056 024050 013737 000004 001176
3057 024056 012737 024114 000004
3058
3059 024064 012706 001000
3060
3061 024070 011502
3062 024072 010237 045440
3063 024076 005712
3064
3065 024100 062705 000002
3066 024104 020527 015012
3067 024110 101767
3068 024112 000401
3069
3070 024114 104042
3071
3072 024116 013737 001176 000004

```

```

TIMOT=4
REM %
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 WILL
RESPOND TO I.E. NOT CAUSE A NON-EXISTANT MEMORY ERROR WHEN ACCESSED
BY A "TST" INSTRUCTION.
%
*****
;*TEST 11      BCTA LEGAL REGISTER RESPONSE TEST
*****
TST11:  SCOPE
        MOV      #1, $TIMES      ;:DO 1 ITERATION
        MOV      #TNO, @#TSTNM   ;↑THIS SAVES TEST NUMBER
        MOV      #CLR, @RHCS2    ;CLEAR RH11 CONTROLLER
        MOV      #RHCS1, R5      ;R5=LIST POINTER.
        MOV      @#TIMOT, $TMPD  ;SAVE TIMEOUT VECTOR.
        MOV      #E00, @#TIMOT  ;SET VECTOR TO E00
L00:    MOV      #STACK, SP      ;SET STACK POINTER.
I00:    MOV      (R5), R2        ;R2=RH11 ADDRESS.
        MOV      R2, REGADR      ;
        TST      (R2)           ;DOES THE RH11 REGISTER RESPOND?
        ADD      #2, R5          ;UPDATE ADDRESS
        CMP      R5, #RHEC2     ;AT THE END OF LEGAL RH11 REGISTERS?
        BLOS    I00            ;NOPE
        BR      000            ;YES! GOTO 000.
E00:    ERROR  42
000:    MOV      $TMPD, @#TIMOT  ;RESTORE TIMEOUT VECTOR.

```

M09

MNDEC-11-DZRJH-A RPO4/5/6 DSKLS CONTROLLER TST-PT 2 MACY11 27(655) 30-MAR-76 20:59 PAGE 69
FLTINI.P11 111 BCTA LEGAL REGISTER RESPONSE TEST

SEQ 0115

3073

3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110

REM %
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.

%

```

;*****
;TEST 12      BCTA MOV B LO BYTE TO WC
;*****
†TST12:  SCOPE
          MOV     #1, $TIMES      ;DO 1 ITERATION
          MOV     #TNO, 2#TSTNM  ;THIS SAVES TEST NUMBER
          MOV     #CLR, 2#RHCS2  ;CLEAR RH11 CONTROLLER

          MOV     #STACK, SP     ;SET STACK POINTER.

          MOV     RHWC, R2       ;R2=RH11 WC ADDRESS.
          MOV     R2, REGADR     ;REGISTER ADDRESS TO REGADR FOR TYPING.
          MOV     #377, $GDDAT   ;$GDDAT=S/B.

L02:     CLR     (R2)           ;CLEAR RH11 WC.

I02:     MOV B  #377, (R2)     ;SET WC TO LO BYTE.
          MOV     (R2), $BDDAT  ;$BDDAT=ACTUAL WAS

          CMP     $BDDAT, $GDDAT ;COMPARE RESULTS
          BEQ     TST13        ;NEXT TEST
          ERROR  44

```

024124	000004		
024126	012737	000001	001212
024134	012737	000012	017330
024142	012777	000040	170606
024150	012706	001000	
024154	013702	014752	
024160	010237	045440	
024164	012737	000377	001124
024172	005012		
024174	112712	000377	
024200	011237	001126	
024204	023737	001126	001124
024212	001401		
024214	104044		

B10

0111
0112
0113
0114
0115
0116
0117
0118
0119
0120
0121
0122
0123
0124
0125
0126
0127
0128
0129
0130
0131
0132
0133
0134
0135
0136
0137
0138
0139
0140

024216 000004
024220 012737 000001 001212
024226 012737 000013 017330
024234 012777 000040 170514

024242 012706 001000

024246 013702 014752
024252 010237 045440
024256 012737 177400 001124

024264 005012

024266 005202

024270 112712 000377

024274 005302

024276 011237 001126

024302 023737 001126 001124
024310 001401
024312 104044

```
*****  
: TEST 13 BCTA MOV B HI BYTE TO WC  
*****  
↑ST13: SCOPE  
MOV #1,STIMES ;DO 1 ITERATION  
MOV #TINO,STSTNM ;THIS SAVES TEST NUMBER  
MOV #CLR,RHCS2 ;CLEAR RH11 CONTROLLER  
  
MOV #STACK,SP ;SET STACK POINTER.  
  
MOV RHWC,R2 ;R2=RH11 WC HI BYTE ADDRESS.  
MOV R2,REGADR  
MOV #177400,$GDDAT ;$GDDAT=S'B.  
  
L03: CLR (R2) ;CLEAR RH11 WC.  
  
INC R2 ;R2=HI BYTE ADDRESS.  
  
I03: MOVB #377,(R2) ;SET WC HI BYTE.  
  
DEC R2 ;R2=FULL WORD ADDRESS.  
  
MOV (R2),$BDDAT ;$BDDAT=ACTUAL.  
  
CMP $BDDAT,$GDDAT ;COMPARE RESULTS.  
BEQ TST14 ;NEXT TEST  
ERROR 44
```

```

3141
3142
3143
3144
3145 024314 000004
3146 024316 012737 000001 001212
3147 024324 012737 000014 017330
3148 024332 012777 000040 170416
3149
3150 024340 012706 001000
3151
3152 024344 013702 014752
3153 024350 010237 045440
3154 024354 012737 000377 001124
3155
3156 024362 005012          L04: CLR (R2)          ;CLEAR RH11 WC.
3157
3158 024364 012712 000252          MOV #252,(R2)        ;SET UP WC
3159 024370 152712 000125          I04: BISB #125,(R2) ;DO A BISB
3160
3161 024374 011237 001126          MOV (R2),SBDDAT     ;SBDDAT=WAS.
3162
3163 024400 023737 001126 001124          CMP SBDDAT,$GDDAT   ;COMPARE RESULTS
3164 024406 001401          BEQ TST15           ;NEXT TEST
3165 024410 104045          ERROR 45
3166

```

```

:*****
:*TEST 14          BCTA BISB LO BYTE TO WC
:*****

```

```

†TST14: SCOPE
MOV #1,STIMES      ;DO 1 ITERATION
MOV #TNO,†TSTNM    ;THIS SAVES TEST NUMBER
MOV #CLR,†RHCS2    ;CLEAR RH11 CONTROLLER
MOV #STACK,SP      ;SET STACK POINTER.
MOV RHWC,R2        ;R2=RH11 WC ADDRESS
MOV R2,REGADR      ;
MOV #377,$GDDAT    ;$GDDAT=S/B.
L04: CLR (R2)      ;CLEAR RH11 WC.
I04: MOV #252,(R2) ;SET UP WC
      BISB #125,(R2);DO A BISB
MOV (R2),SBDDAT   ;SBDDAT=WAS.
CMP SBDDAT,$GDDAT;COMPARE RESULTS
BEQ TST15         ;NEXT TEST
ERROR 45

```



```

3168
3169
3170
3171
3172 024412 000004
3173 024414 012737 000001 001212
3174 024422 012737 000015 017330
3175 024430 012777 000040 170320
3176
3177 024436 012706 001000
3178
3179 024442 013702 014752
3180 024446 010237 045440
3181 024452 012737 177400 001124
3182
3183 024460 005012 LOS: CLR (R2) ;CLEAR RH11 WC.
3184
3185 024462 005202 INC R2 ;R2 =HI BYTE.
3186
3187 024464 112712 000125 MOVB #125,(R2) ;SET UP RH11 WC.
3188 024470 152712 000252 IOS: BISB #252,(R2) ;DO A BISB.
3189
3190 024474 005302 DEC R2 ;R2=FULL WORD ADDRESS.
3191
3192 024476 011237 001126 MOV (R2), $BDDAT ;$BDDAT=WAS.
3193
3194 024502 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3195 024510 001401 BEQ TST16 ;NEXT TEST
3196 024512 104045 ERROR 45
3197
3198

```

```

*****
;TEST 15 BCTA BISB HI-BYTE TO WC
*****

```

```

TST15: SCOPE
MOV #1,$TIMES ;DO 1 ITERATION
MOV #TINO,2#TSTNM ;THIS SAVES TEST NUMBER
MOV #CLR,RHCS2 ;CLEAR RH11 CONTROLLER
MOV #STACK,SP ;SET STACK POINTER.
MOV RHWC,R2 ;R2=RH11 WC ADDRESS.
MOV R2,REGADR
MOV #177400,$GDDAT ;$GDDAT=S/B.

```

E10

```

3199
3200
3201          ;*****
3202          ;*TEST 16      BCTA BICB LO-BYTE TO WC
3203          ;*****
3203 024514 000004          TST16: SCOPE
3204 024516 012737 000001 001212      MOV      #1,STIMES      ;;DO 1 ITERATION
3205 024524 012737 000016 017330      MOV      #TNO,STSTNM   ;THIS SAVES TEST NUMBER
3206 024532 012777 000040 170216      MOV      #CLR,RHCS2    ;CLEAR RH11 CONTROLLER
3207
3208 024540 012706 001000          MOV      #STACK,SP     ;SET STACK POINTER.
3209
3210 024544 013702 014752          MOV      RHWC,R2       ;R2=RH11 WC ADDRESS.
3211 024550 010237 045440          MOV      R2,REGADR
3212 024554 012737 000252 001124      MOV      #252,$GDDAT   ;$GDDAT=S/B.
3213
3214 024562 005012          L06:    CLR      (R2)    ;CLEAR RH11 WC.
3215
3216 024564 012712 000377          MOV      #377,(R2)    ;SET WC=0000377
3217
3218 024570 142712 000125          I06:    BICB     #125,(R2) ;DO A BICB
3219
3220 024574 011237 001126          MOV      (R2),$BDDAT   ;$BDDAT=WAS.
3221
3222 024600 023737 001126 001124      CMP      $BDDAT,$GDDAT ;COMPARE RESULTS.
3223 024606 001401          BEQ      TST17        ;NEXT TEST
3224 024610 104046          ERROR   46
3225
3226

```

F10

```

3227
3228
3229
3230
3231 024612 000004
3232 024614 012737 000001 001212
3233 024622 012737 000017 017330
3234 024630 012777 000040 170120
3235
3236 024636 012706 001000
3237
3238 024642 013702 014752
3239 024646 010237 045440
3240 024652 012737 125000 001124
3241
3242 024660 005012
3243
3244 024662 005202
3245
3246 024664 112712 000377
3247 024670 142712 000125
3248
3249 024674 005302
3250
3251 024676 011237 001126
3252
3253 024702 023737 001126 001124
3254 024710 001401
3255 024712 104046
3256
3257

;*****
;TEST 17 BCTA BICB HI- BYTE TO WC
;*****
TST17: SCOPE
MOV #1,STIMES ;DO 1 ITERATION
MOV #TNO,#TSTNM ;THIS SAVES TEST NUMBER
MOV #CLR,RHCS2 ;CLEAR RH11 CONTROLLER
MOV #STACK,SP ;SET STACK POINTER.
MOV RHWC,R2 ;R2=RH11 WC ADDRESS.
MOV R2,REGADR
MOV #125000,$GDDAT ;$GDDAT=S/B.
LO7: CLR (R2) ;CLEAR RH11 WC.
INC R2 ;R2=HI BYTE.
MOV #377,(R2) ;SET WC=177400
IO7: BICB #125,(R2) ;DO A BICB
DEC R2 ;R2=FULL WORD.
MOV (R2),$BDDAT ;$BDDAT=WAS.
CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
BEQ TST20 ;NEXT TEST
ERROR 46
  
```

G10

```

3258
3259 .REM %
3260 THIS TEST ATTEMPT TO ACCESS AN ILLEGAL REGISTER WITHIN THE RANGE
3261 OF ADDRESSES SELECTED BY THE XOR'S. THE RH11 ADDRESS DECODE LOGIC WILL ALLOW UP TO 32
3262 CONTIGUOUS REGISTERS TO EXIST. IN OUR CONFIGURATION ONLY 16 WILL REALLY
3263 EXIST. THIS TEST ATTEMPTS TO ACCESS THE 20(8) REGISTER - IF IT RESPONDS
3264 THE TEST WILL TYPE OUT AN ERROR.
3265
3266 FOR THE CASE OF THE RH70, THE CONFIGURATION ALLOWS 18 REGISTERS, SO
3267 WE WILL ATTEMPT TO ADDRESS REGISTER 23(8) OR AS BEFORE THE LAST
3268 REGISTER + 2.
3269 %
3270

```

```

3271
3272 ;*****
3273 ;*TEST 20 BCTA ILLEGAL REGISTER TEST
3274 ;*****
3275
3276 †ST20: SCOPE
3277 MOV #1,STIMES ;DO 1 ITERATION
3278 MOV #TTNO,‡TSTNM ;THIS SAVES TEST NUMBER
3279 MOV #CLR,‡RHCS2 ;CLEAR RH11 CONTROLLER
3280
3281 TST ‡RH70 ;CHECK TO SEE IF RUNNING WITH RH70
3282 BEQ 1$ ;IF NOT... SKIP NEXT & DO FOLLOWING
3283 MOV RHCS3,R2 ;R2 = LAST LEGAL RH70 REG ADDRESS
3284 BR 2$ ;SKIP NEXT
3285
3286 1$: MOV RHEC2,R2 ;R2 = LAST LEGAL RH11 REG ADDRESS.
3287 ADD #2,R2 ;R2 = FIRST ILLEGAL ADDRESS.
3288 MOV R2,REGADR
3289 CLR $BDDAT ;$BDDAT=WAS.
3290 CLR $GDDAT ;$GDDAT=S/B.
3291
3292 MOV ‡TIMOT,$TMPD ;SAVE TIMEOUT VECTOR.
3293 MOV #010,‡TIMOT ;SET TIMEOUT VECTOR TO 010.
3294
3295 L10: MOV #STACK,SP ;SET THE STACK POINTER.
3296
3297 TST (R2) ;TEST ADDRESS.
3298
3299 ERROR 47
3300
3301 010: MOV $TMPD,‡TIMOT ;RESTORE TIMEOUT VECTOR.

```

```

3301
3302      ; NOW WE ATTACK BCTB
3303
3304      .REM      %
3305      THE FOLLOWING TWO TESTS DETERMINE IF ALL NON DRIVES SET THE
3306      NED BIT AND THAT ALL EXISTING DRIVES CLEAR THE NED BIT.
3307      THE TESTS LOOK AT TOTALAT TO DETERMINE WHICH DRIVES EXIST AND TEST THAT
3308      THESE DRIVES WILL CLEAR THE NED BIT. AND ALSO THAT NON EXISTANT DRIVES
3309      WILL SET THE NED BIT.
3310      ON FIRST PASS ONLY, IF ALL DRIVES ARE ON LINE A MESSAGE WILL SO ADVISE THE
3311      OPERATOR. THE TEST WILL CONTINUE REGARDLESS OF THE OPERATORS ACTION.
3312      %
3313      ;*****
3314      ;*TEST 21      BCTB (NED) NON-EXISTANT DRIVE TEST. (SET)
3315      ;*****
3316      TST21:  SCOPE
3317      MOV      #1,STIMES      ;;DO 1 ITERATION
3318      MOV      #TNO, #TSTNM  ;THIS SAVES TEST NUMBER
3319      MOV      #CLR, #RHCS2  ;CLEAR RH11 CONTROLLER
3320
3321      CMPB     #TOTALAT, #377 ;ARE ALL DRIVES ON LINE.
3322      BNE      U11           ;NO GO RUN THE TEST.
3323
3324      CMP      $PASS, #0     ;IF PASS #0 THEN TYPE MESSAGE
3325                        ;ADVISING OPERATOR
3326                        ;THAT THIS TEST WILL NOT BE RUN
3327
3328      BNE      Y11
3329      JMP      X11
3330
3331      Y11:      TYPE      65$      ;;TYPE ASCIZ STRING
3332      BR       64$      ;;GET OVER THE ASCIZ
3333      ;;65$:   .ASCIZ  <15><12>/ALL DRIVES APPEAR TO BE ON LINE THEREFCRE/
3334      64$:
3335      TYPE      67$      ;;TYPE ASCIZ STRING
3336      BR       66$      ;;GET OVER THE ASCIZ
3337      ;;67$:   .ASCIZ  <15><12>/THE NED NON-EXISTANT DRIVE TEST CANNOT BE/
3338      66$:
3339      TYPE      69$      ;;TYPE ASCIZ STRING
3340      BR       68$      ;;GET OVER THE ASCIZ
3341      ;;69$:   .ASCIZ  <15><12>/RUN. TO RUN THIS TEST TAKE ONE OR MORE/
3342      68$:
3343      TYPE      71$      ;;TYPE ASCIZ STRING
3344      BR       70$      ;;GET OVER THE ASCIZ
3345      ;;71$:   .ASCIZ  <15><12>/DRIVES OFF-LINE AND RESTART THE PROGRAM/
3346      70$:
3347
3348      U11:     MOV      #STACK, SP      ;SET STACK POINTER.
3349
3350      MOV      RHCS2, R2      ;R2=RH11 RHCS2 ADDRESS.
3351      MOV      R2, REGADR
3352
3353      MOV      #TOTALAT, R0
3354      CLR      R1

```

```

3355 025430 006000          S11:  ROR  R0
3356 025432 103423          BCS  N11
3357
3358 025434 010137 001124  R11:  MOV  R1,$GDDAT ;$GDDAT=S/B.
3359 025440 052737 010000 001124 BIC  #NED,$GDDAT
3360
3361 025446 013705 014760          MOV  RHCS1,R5 ;ADDRESS OF A DEVICE REGISTER.
3362
3363 025452 010112          L11:  MOV  R1,(R2) ;LOAD A NON-EXISTANT DRIVE.
3364
3365 025454 011537 001176          MOV  (R5),$TMPO ;ATTEMPT TO READ FROM DEVICE REGISTER.
3366
3367 025460 011237 001126          MOV  (R2),$BDDAT ;$BDDAT=WAS.
3368 025464 042737 167770 001126 BIC  #↑C<NED!US4!US2!US1>,$BDDAT;$BDDAT=SAVED DATA.
3369
3370 025472 023737 001126 001124 CMP  $BDDAT,$GDDAT ;COMPARE RESULTS.
3371 025500 001005          BNE  E11
3372
3373 025502 005201          N11:  INC  R1
3374 025504 020127 000010          CMP  R1,#8. ;TESTED ALL DRIVES YET.
3375 025510          X11:
3376 025510 001402          BEQ  TST22 ;NEXT TEST
3377 025512 000746          BR   S11
3378
3379 025514 104050          E11:  ERROR 50
3380
3381

```

```

3382
3383
3384      ;:*****
3385      ;*TEST 22      BCTB (NED) NON-EXISTANT DRIVE TEST. (CLEARED)
3386      ;:*****
3386 025516 000004
3387 025520 012737 000001 001212  TST2: SCOPE
3388 025526 012737 000022 017330      MOV #1,$TIMES      ;;DO 1 ITERATION
3389 025534 012777 000040 167214      MOV #TNO,2#TSTNM  ;THIS SAVES TEST NUMBER
3390
3391 025542 012706 001000      MOV #CLR,2RHCS2  ;CLEAR RH11 CONTROLLER
3392
3393 025546 013702 014756      MOV #STACK,SP    ;SET STACK POINTER.
3394 025552 010237 045440      MOV RHCS2,R2     ;R2=RH11 RHCS2 ADDRESS.
3395 025556 013700 015140      MOV R2,REGADR
3396 025562 005001
3397 025564 006000      MOV #TOTALAT,R0
3398 025566 103020      CLR R1
3399
3400 025570 010137 001124      CLR R0
3401
3402 025574 013705 014760      BCC N12
3403
3404 025600 010112      MOV R1,$GDDAT
3405
3406 025602 011537 001176      MOV RHCS1,R5     ;ADDRESS OF A DEVICE REGISTER.
3407
3408 025606 011237 001126      L12: MOV R1,(R2)  ;SELECT UNIT.
3409 025612 042737 167770 001126      MOV (R5),$TMPD  ;ATTEMPT TO READ FROM DEVICE.
3410
3411 025620 023737 001126 001124      MOV (R2),$BDDAT ;$BDDAT=WAS.
3412 025626 001005      BIC #<NED!US4!US2!US1>,$BDDAT ;$BDDAT=SAVED DATA.
3413
3414 025630 005201      CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3415 025632 020127 000010      BNE E12
3416 025636 001402
3417
3418 025640 000751      N12: INC R1
3419 025642 104050      CMP R1,#8.      ;TESTED ALL DRIVES.
3420
3421      BEQ TST23     ;NEXT TEST
3421      BR S12
3421      ERROR S12
3421      50

```

3422
3423
3424
3425
3426
3427
3428
3429
3430
3431
3432
3433
3434
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459

```

.REM      %
TEST TO SEE IF WE CAN READ FROM THE "AS" REGISTER AND NOT CAUSE
A NED NON-EXISTANT DRIVE ERROR
%
;*****
; *TEST 23      BCTB AS REGISTER TEST
;*****
†ST23:  SCOPE
MOV      #1, $TIMES      ; DO 1 ITERATION
MOV      #TTNO, @†TSTNM ; THIS SAVES TEST NUMBER
MOV      #CLR, @RHCS2    ; CLEAR RH11 CONTROLLER
MOV      #STACK, SP     ; SET STACK POINTER.
MOV      RHCS2, R2       ; R2=RH11 CS2 ADDRESS.
MOV      #US4!US2!US1, $GDDAT; $GDDAT=S/B.
MOV      RHCS1, R5       ; ADDRESS OF A DEVICE REGISTER.
L13:    MOV      #US4!US2!US1, (R2); LOAD A NON-EXISTANT DEVICE.
MOV      RHAS, R2        ; R2= "AS".
MOV      (R2), $TMPO     ; ATTEMPT TO READ FROM DEVICE AS.
CLR      (R2)            ; CLEAR AS REGISTER.
MOV      RHCS2, R2       ; R2=RH11 CS2 ADDRESS.
MOV      R2, REGADR
MOV      (R2), $BDDAT    ; $BDDAT=WAS.
BIC      #†C<NED!US4!US2!US1>, $BDDAT; SAVE NED AND DEVICE SELECTED.
CMP      $BDDAT, $GDDAT  ; COMPARE RESULTS.
BEQ      TST24           ; NEXT TEST
ERROR    51

```

```

025644 000004
025646 012737 000001 001212
025654 012737 000023 017330
025652 012777 000040 167066
025670 012706 001000
025674 013702 014756
025700 012737 000007 001124
025706 013705 014760
025712 012712 000007
025716 013702 014776
025722 011237 001176
025726 005012
025730 013702 014756
025734 010237 045440
025740 011237 001126
025744 042737 167770 001126
025752 023737 001126 001124
025760 001401
025762 104051

```



```

3460          .REM          %
3461          THE NEXT THREE TESTS WILL TEST THE BUS ADDRESS REGISTER IN BOTH WORD
3462          AND BYTE MODE. THE PATTERNS ARE CHOSEN TO PICK UP ANY DROPPED OR STUCK
3463          BITS.
3464          %
3465
3466          ;*****
3467          ;*TEST 24          BCTC BUS ADDRESS REGISTER
3468          ;*****
3469          TST24:  SCOPE
3470          025764  000004          MOV          #1, $TIMES          ;; DO 1 ITERATION
3471          025774  012737  000001  001212  MOV          #TNO, #TSTNM          ; THIS SAVES TEST NUMBER
3472          026002  012777  000024  017330  MOV          #CLR, @RHCS2          ; CLEAR RH11 CONTROLLER
3473
3474          026010  012706  000040  166746  MOV          #STACK, SP          ; SET STACK POINTER.
3475
3476          026014  013702  001000          MOV          #R2BA, R2          ; R2=RH11 BA ADDRESS.
3477          026020  010237  014754          MOV          R2, REGADR
3478          026024  012705  045440          MOV          #LST14A, R5          ; R5=TEST LIST ADDRESS.
3479
3480          026030  012537  0026064          L14:  MOV          (R5)+, $GDDAT          ; $GDDAT=S/B.
3481
3482          026034  013712  001124          I14:  MOV          $GDDAT, (R2)          ; SET BUS ADDRESS REGISTER.
3483          026040  011237  001126          MOV          (R2), $BDDAT          ; READ BUS ADDRESS REGISTER.
3484
3485          026044  023737  001126  001124          CMP          $BDDAT, $GDDAT          ; COMPARE RESULTS.
3486          026052  001401          BEQ          N14          ; GET NEXT TEST DATA.
3487          026054  104052          ERROR          52
3488
3489          026056  005715          N14:  TST          (R5)          ; AT END OF LIST
3490          026060  001363          BNE          L14          ; NO!
3491          026062  000440          BR          TST25          ; NEXT TEST
3492          026064  000002          LST14A:  2
3493          026066  000004          4
3494          026070  000010          10
3495          026072  000020          20
3496          026074  000040          40
3497          026076  000100          100
3498          026100  000200          200
3499          026102  000400          400
3500          026104  001000          1000
3501          026106  002000          2000
3502          026110  004000          4000
3503          026112  010000          10000
3504          026114  020000          20000
3505          026116  040000          40000
3506          026120  100000          100000
3507          026122  177776          177776
3508          026124  177774          177774
3509          026126  177772          177772
3510          026130  177766          177766
3511          026132  177756          177756
3512          026134  177736          177736
3513          026136  177676          177676

```

M10

MNDEC-11-DZRJH-A.RP04/5/6 DSKLS CONTROLLER TST-PT 2
FLTINI.P11 ↑24 BCTC BUS ADDRESS REGISTER

MACY11 27(655) 30-MAR-76 20:59 PAGE 82

SEQ 0128

3514	026140	177576	177576
3515	026142	177376	177376
3516	026144	176776	176776
3517	026146	175776	175776
3518	026150	173776	173776
3519	026152	167776	167776
3520	026154	157776	157776
3521	026156	137776	137776
3522	026160	077776	077776
3523	026162	000000	0
3524			

```

3525
3526
3527
3528
3529 026164 000004
3530 026166 012737 000001 001212
3531 026174 012737 000025 017330
3532 026202 012777 000040 166546
3533
3534 026210 012706 001000
3535
3536 026214 013702 014754
3537 026220 010237 045440
3538 026224 012705 026266
3539
3540 026230 005012 L15: CLR (R2) ;CLEAR RH11 BA REGISTER.
3541
3542 026232 012537 001124 MOV (R5)+,$GDDAT ;$GDDAT=S/B.
3543
3544 026236 113712 001124 I15: MOVB $GDDAT,(R2) ;SET BUS ADDRESS REGISTER.
3545 026242 011237 001126 MOV (R2),$BDDAT ;READ BUS ADDRESS REGISTER.
3546
3547 026246 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3548 026254 001401 BEQ R15
3549
3550 026256 104053 ERROR 53
3551
3552 026260 005715 R15: TST (R5) ;AT END OF TEST LIST.
3553 026262 001362 BNE L15 ;NO!
3554 026264 000417 BR TST26 ;NEXT TEST
3555 ;THIS LIST WILL BE USED TO LOAD THE LOWER BYTE OF THE BA REGISTER.
3556
3557 026266 000002 LST15A: 2
3558 026270 000004 4
3559 026272 000010 10
3560 026274 000020 20
3561 026276 000040 40
3562 026300 000100 100
3563 026302 000200 200
3564 026304 000376 376
3565 026306 000372 372
3566 026310 000366 366
3567 026312 000356 356
3568 026314 000336 336
3569 026316 000276 276
3570 026320 000176 176
3571 026322 000000 0

```

```

;*****
;TEST 25 BCTC BUS ADDRESS REGISTER LO-BYTE
;*****

```

```

TST25: SCOPE
MOV #1,$TIMES ;DO 1 ITERATION
MOV #TTNO,2#TSTNM ;THIS SAVES TEST NUMBER
MOV #CLR,2#RHCS2 ;CLEAR RH11 CONTROLLER
MOV #STACK,SP ;RESET STACK.
MOV RHBA,R2 ;R2=RH11 BA ADDRESS.
MOV R2,REGADR
MOV #LST15A,R5 ;R5=TEST LIST ADDRESS.
L15: CLR (R2) ;CLEAR RH11 BA REGISTER.
MOV (R5)+,$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
ERROR 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

```

```

3573
3574
3575
3576 026324 000004
3577 026326 012737 000001 001212
3578 026334 012737 000026 017330
3579 026342 012777 000040 166406
3580
3581 026350 012706 001000
3582
3583 026354 013702 014754
3584 026360 010237 045440
3585 026364 012705 026436
3586
3587 026370 005012 L16: CLR (R2) ;CLEAR BA REGISTER.
3588
3589 026372 012537 001124 MOV (R5)+,$GDDAT ;$GDDAT=S/B.
3590
3591 026376 005202 INC R2 ;R2=HI BYTE ADDRESS.
3592
3593 026400 113712 001124 L16: MOV B $GDDAT,(R2) ;SET BUS ADDRESS REGISTER HI-BYTE.
3594
3595 026404 005302 DEC R2 ;R2=FULL WORD ADDRESS.
3596
3597 026406 011237 001126 MOV (R2),$BDDAT ;READ BUS ADDRESS.
3598 026412 000337 001124 SWAB $GDDAT ;ADJUST DATA FOR HI BYTE.
3599
3600 026416 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3601 026424 001401 BEQ R16 ;OKAY
3602
3603 026426 104053 ERROR 53
3604
3605 026430 005715 R16: TST (R5) ;AT END OF TEST LIST.
3606 026432 001356 BNE L16 ;NOPE
3607 026434 000420 BR TST27 ;NEXT TEST
3608 ;THIS LIST WILL BE USED TO LOAD THE UPPER BYTE OF THE BA REGISTER.
3609
3610 026436 000002 LST16A: 2
3611 026440 000004 4
3612 026442 000010 10
3613 026444 000020 20
3614 026446 000040 40
3615 026450 000100 100
3616 026452 000200 200
3617 026454 000376 376
3618 026456 000374 374
3619 026460 000376 376
3620 026462 000366 366
3621 026464 000356 356
3622 026466 000336 336
3623 026470 000276 276
3624 026472 000176 176
3625 026474 000000 0

```

C11

3526



```

3627 .REM %
3628 THIS TEST CAUSE AN RH11 INTERRUPT VIA THE SPECIAL COMMAND SEQUENCE
3629 BIS #IE,DRHCS2. THIS TEST PROVES ONLY THAT THE HARDWARE CAN HANDLE
3630 INTERRUPTS PROPERLY
3631 %
3632 ::*****
3633 :*TEST 27 RH11 INTERRUPT TEST
3634 ::*****
3635 †ST27: SCOPE
3636 026476 000004 MOV #1,$TIMES ;:DO 1 ITERATION
3637 026500 012737 000001 001212 MOV #TTNO,@TSTNM ;†THIS SAVES TEST NUMBER
3638 026506 012737 000027 017330 MOV #CLR,DRHCS2 ;:CLEAR RH11 CONTROLLER
3639 026514 012777 000040 166234
3640 026522 012706 001000 MOV #STACK,SP ;:SET STACK POINTER.
3641
3642 026526 013702 014760 MOV RHCS1,R2 ;:R2=RH11 ADDRESS.
3643 026532 010237 045440 MOV R2,REGADR
3644 026536 005037 001126 CLR $BDDAT ;:$BDDAT=WAS.
3645 026542 005037 001124 CLR $GDDAT ;:$GDDAT=S/B.
3646
3647 026546 017737 166174 001176 MOV DRPVEC,$TMPD ;:SAVE RH11 INTERRUPT VECTOR.
3648 026554 012777 026600 166164 MOV #021,DRPVEC ;:SET RH11 INTERRUPT VECTOR TO 021.
3649
3650 026562 052712 000100 L21: BIS #IE,(R2) ;:CAUSE INTERRUPT.
3651 026566 000240 NOP ;:WAIT FOR INTERRUPT.
3652 026570 000240 NOP
3653 026572 011237 001124 MOV (R2),$GDDAT ;:SAVE CONTENTS OF REGISTER.
3654
3655 026576 104054 ERROR 54
3656
3657 026600 013777 001176 166140 C21: MOV $TMPD,DRPVEC ;:RESTORE RH11 INTERRUPT VECTOR.
3658
3659

```

3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689

```

:TEST THAT RESET CAN GENERATE THE SIGNAL CLR L.
.REM %
      THE PROGRAM SETS THE PORT SELECT BIT .THEN DOES A RESET
      IF THE SIGNAL CLR L WAS GENERATED THE PORT SELECT BIT WILL CLEAR.
      %
:*****
:*TEST 30      BCTD CLR L TEST
:*****
TST30: SCOPE
      MOV      #1,STIMES      ;;DO 1 ITERATION
      MOV      #TNO,2#TSTNM   ;THIS SAVES TEST NUMBER
      MOV      #CLR,2RHCS2    ;CLEAR RH11 CONTROLLER
      MOV      RHCS1,R2       ;R2=RH11 CS1 ADDRESS
      MOV      R2,REGADR
      CLR      $GDDAT         ;$GDDAT=S/B
-22:  MOV      #STACK,SP      ;SET STACK POINTER.
      MOV      #PSEL,(R2)     ;SET PORT SELECT FLOP.
      RESET                      ;DO A BUSA INIT.
      MOV      (R2), $BDDAT    ;$BDDAT=WAS.
      BIC      #IC<IE>,$BDDAT ;SAVE ONLY I.E. BIT.
      CMP      $BDDAT,$GDDAT  ;COMPARE RESULTS.
      BEQ      TST31          ;NEXT TEST
      ERROR   55

```

```

026606 000004
026610 012737 000001 001212
026616 012737 000030 017330
026624 012777 000040 166124
026632 013702 014760
026636 010237 045440
026642 005037 001124
026646 012706 001000
026652 012712 002000
026656 000005
026660 011237 001126
026664 042737 177677 001126
026672 023737 001126 001124
026700 001401
026702 104055

```

F11

```

3690 .REM %
3691 SET THE MXF FLOP AND READ IT BACK.
3692 %
3693 ::*****
3694 :*TEST 31 MXF TEST
3695 ::*****
3696 †ST31: SCOPE
3697 026704 000004 MOV #1,STIMES ;DO 1 ITERATION
3698 026706 012737 000001 001212 MOV #TNO.2*STNM ;THIS SAVES TEST NUMBER
3699 026714 012737 000031 017330
3700 ;*CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70
3701
3702 026722 005737 017334 TST 2#RH70 ;TEST FOR RH70 CONTROLLER
3703 026726 001402 BEQ 30$ ;IF FLAG = 1, SKIP THIS TEST
3704 026730 000137 027014 JMP TST32 ;JUMP TO NEXT TEST -----)
3705 026734 30$: MOV #CLR,2RHCS2 ;IF FLAG = 0, DO THIS TEST
3706 026734 012777 000040 166014 ;CLEAR RH11 CONTROLLER
3707
3708 026742 012706 001000 MOV #STACK,SP ;SET STACK POINTER.
3709 026746 013702 014756 MOV RHCS2,R2 ;R2=RH11 CS2 ADDRESS.
3710 026752 010237 045440 MOV R2,REGADR
3711 026756 012737 001000 001124 MOV #MXF,$GDDAT ;$GDDAT=S/B.
3712
3713 026764 012712 001000 124: MOV #MXF,(R2) ;LOAD MXF
3714
3715 026770 011237 001126 MOV (R2),$BDDAT ;$BDDAT=WAS.
3716
3717 026774 042737 176777 001126 BIC #1<MXF>,$BDDAT ;SAVE ONLY MXF
3718
3719 027002 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS
3720 027010 001401 BEQ TST32 ;NEXT TEST
3721 027012 104056 ERROR 56
3722
3723

```



```

3724 .REM %
3725 SET THE UNIBUS PARITY ERROR FLOP DIRECTLY VIA A MOV #UPE TO RHCS2
3726 ATTEMPT TO READ IT BACK.
3727 %
3728 ::*****
3729 :*TEST 32 CSRB UNIBUS PARITY ERROR SET TEST
3730 :*****
3731 TST32: SCOPE
3732 MOV #1, $TIMES ; DO 1 ITERATION
3733 MOV #TNO, @TSTNM ; THIS SAVES TEST NUMBER
3734
3735 ;*CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70
3736
3737 TST @RH70 ; TEST FOR RH70 CONTROLLER
3738 BEQ 30$ ; IF FLAG = 1, SKIP THIS TEST
3739 JMP TST33 ; JUMP TO NEXT TEST -----)
3740 30$: ; IF FLAG = 0, DO THIS TEST
3741
3742 MOV #CLR, @RHCS2 ; CLEAR RH11 CONTROLLER
3743
3744 MOV #STACK, SP ; SET STACK POINTER.
3745
3746 MOV RHCS2, R2 ; R2=RHCS2 ADDRESS.
3747 MOV R2, REGADR
3748 MOV #UPE, $GDDAT ; $GDDAT=S/B.
3749
3750 L30: MOV $GDDAT, (R2) ; ATTEMPT TO SET UPE.
3751 NOP
3752 NOP
3753
3754 MOV (R2), $BDDAT ; $BDDAT=ACTUAL DATA.
3755 BIC #1C(UPE), $BDDAT ; SAVE ONLY UPE.
3756
3757 CMP $BDDAT, $GDDAT ; COMPARE RESULTS.
3758 BEQ TST33 ; NEXT TEST
3759 ERROR 57
3760
3761

```

```

3762
3763
3764
3765
3766
3767
3768
3769
3770 027130 000004
3771 027132 012737 000001 001212
3772 027140 012737 000033 017330
3773 027146 012777 000040 165602
3774
3775 027154 012706 001000
3776
3777 027160 013702 014756
3778 027164 010237 045440
3779 027170 005037 001124
3780
3781 027174 012712 020000 L31: MOV #UPE, (R2) ;SET UNIBUS PARITY ERROR SET.
3782 027200 000240 NOP
3783 027202 000240 NOP
3784
3785 027204 012712 000040 MOV #CLR, (R2) ;CONTROLLER CLEAR.
3786
3787 027210 011237 001126 MOV (R2), $BDDAT ;READ STATUS
3788 027214 042737 157777 001126 BIC #1C(UPE), $BDDAT ;SAVE ERROR.
3789
3790 027222 023737 001126 001124 CMP $BDDAT, $GDDAT ;COMPARE RESULTS.
3791 027230 001401 BEQ TST34 ;NEXT TEST
3792 027232 104057 ERROR 57
3793
3794

```

.REM %
SET THE UNIBUS PARITY ERROR FLOP AND ATTEMPT TO CLEAR IT WITH A CONTROLLER CLEAR.

: *TEST 33 CSRB UNIBUS PARITY ERROR CLEAR TEST

TST33: SCOPE
MOV #1, \$TIMES ;DO 1 ITERATION
MOV #TNO, #TSTNM ;THIS SAVES TEST NUMBER
MOV #CLR, #RHCS2 ;CLEAR RH11 CONTROLLER
MOV #STACK, SP ;SET STACK POINTER.
MOV RHCS2, R2 ;R2=RHCS2 ADDRESS.
MOV R2, REGADR
CLR \$GDDAT ;\$GDDAT=S/B.
L31: MOV #UPE, (R2) ;SET UNIBUS PARITY ERROR SET.
NOP
NOP
MOV #CLR, (R2) ;CONTROLLER CLEAR.
MOV (R2), \$BDDAT ;READ STATUS
BIC #1C(UPE), \$BDDAT ;SAVE ERROR.
CMP \$BDDAT, \$GDDAT ;COMPARE RESULTS.
BEQ TST34 ;NEXT TEST
ERROR 57

```

3795
3796
3797
3798
3799
3800
3801
3802
3803 027234 000004
3804 027236 012737 000001 001212
3805 027244 012737 000034 017330
3806 027252 012777 000040 165476
3807
3808 027260 012706 001000
3809
3810 027264 013702 014756
3811 027270 010237 045440
3812 027274 005037 001124
3813
3814 027300 012712 000007
3815 027304 000240
3816 027306 000240
3817
3818 027310 012712 000040
3819
3820 027314 011237 001126
3821 027320 042737 177770 001126
3822
3823 027326 023737 001126 001124
3824 027334 001401
3825 027336 104060
3826
3827

```

```

.REM %
SET THE UNIT SELECT REGISTER TO DRIVE #7 ALL BITS SET. GENERATE A
CONTROLLER CLEAR AND VERIFY THAT THE UNIT SELECT REGISTER IS CLEARED.
%
;*****
;TEST 34 CSRB UNIT SELECT CLEAR TEST
;*****
TST34: SCOPE
MOV #1,$TIMES ;DO 1 ITERATION
MOV #TNO,2#TSTNM ;THIS SAVES TEST NUMBER
MOV #CLR,2RHCS2 ;CLEAR RH11 CONTROLLER
MOV #STACK,SP ;SET STACK POINTER.
MOV RHCS2,R2 ;R2=CS2 ADDRESS.
MOV R2,REGADR
CLR $GDDAT ;$GDDAT=S/B.
L34: MOV #US4!US2!US1,(R2) ;LOAD RHCS2 DRIVE SELECT
NOP
NOP
MOV #CLR,(R2) ;GENERATE CLR.
MOV (R2),$BDDAT ;READ RHCS2
BIC #C<US4!US2!US1>,$BDDAT;SAVE DRIVE SELECT BITS.
CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
BEQ TST35 ;NEXT TEST
ERROR 60

```



```

3928
3929
3830          .REM          %
3831          SET THE UPE FLOP UNIBUS PARITY ERROR. VERIFY THAT THE SETTING OF (UPE)
3832          ALSO SETS THE TRANSFER ERROR (TRE) FLOP
3833          %
3834          ;*****
3835          ;*TEST 35          CSRB TRANSFER ERROR (TRE) - UPE
3836          ;*****
3836 027340 000004          TST35: SCOPE
3837 027342 012737 000001 001212          MOV          #1,$TIMES          ;;DO 1 ITERATION
3838 027350 012737 000035 017330          MOV          #TNO,@TSTNM          ;THIS SAVES TEST NUMBER
3839
3840          ;*CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70
3841
3842 027356 005737 017334          TST          @RH70          ;TEST FOR RH70 CONTROLLER
3843 027362 001402          BEQ          30$          ;IF FLAG = 1, SKIP THIS TEST
3844 027364 000137 027454          JMP          TST36          ;JUMP TO NEXT TEST -----)
3845 027370          30$:          ;IF FLAG = 0, DO THIS TEST
3846
3847 027370 012777 000040 165360          MOV          #CLR,@RHCS2          ;CLEAR RH11 CONTROLLER
3848
3849 027376 012706 001000          MOV          #STACK,SP          ;SET STACK POINTER.
3850
3851 027402 013702 014756          MOV          RHCS2,R2          ;R2=RHCS2 ADDRESS.
3852 027406 012737 040000 001124          MOV          #TRE,$GDDAT          ;$GDDAT=S/B.
3853
3854 027414 012712 020000          L35: MOV          #UPE,(R2)          ;SET UNIBUS PARITY ERROR.
3855
3856 027420 013702 014760          MOV          RHCS1,R2          ;R2=RHCS1 ADDRESS.
3857 027424 010237 045440          MOV          R2,REGADR
3858
3859 027430 011237 001126          MOV          (R2),$BDDAT          ;READ REGISTER
3860 027434 042737 137777 001126          BIC          #T<TRE>,$BDDAT          ;SAVE TRANSFER ERROR.
3861
3862 027442 023737 001126 001124          CMP          $BDDAT,$GDDAT          ;COMPARE RESULTS
3863 027450 001401          BEQ          TST36          ;NEXT TEST
3864 027452 104061          ERROR          61
3865
3866

```

```

3867
3868
3869      .REM      %
3870      SET THE NED BIT NON EXISTANT DRIVE VERIFY THAT THIS SETS THE (TRE) BIT.
3871      %
3872      ;*****
3873      ;*TEST 36      CSRB TRANSFER ERROR (TRE) NED
3874      ;*****
3875      †ST36:  SCOPE
3876      MOV      #1,STIMES      ;;DO 1 ITERATION
3877      MOV      #TNO,‡#TSTNM    ;THIS SAVES TEST NUMBER
3878      MOV      #CLR,‡RHCS2     ;CLEAR RH11 CONTROLLER
3879      MOV      #STACK,SP      ;SET STACK POINTER.
3880
3881      MOV      RHCS2,R2        ;R2=CS2 ADDRESS.
3882      MOV      #TRE,$GDDAT     ;$GDDAT=S/B.
3883
3884      MOV      ‡#TOTALAT,RO     ;GET ALL AVAILABLE DRIVES DATA
3885      CLR      R1
3886      S36:    ROR      RO
3887      BCS     N36
3888      L36:    MOV      R1,(R2)      ;SET NED.
3889
3890      MOV      RHCS1,R2        ;R2=RHCS1 ADDRESS.
3891      MOV      R2,REGADR
3892
3893      MOV      (R2),$BDDAT     ;READ REGISTER.
3894      BIC     #†C<TRE>,$BDDAT ;SAVE TRE.
3895
3896      CMP     $BDDAT,$GDDAT   ;COMPARE RESULTS.
3897      BEQ     TST37          ;NEXT TEST
3898      ERROR  62
3899      BR     TST37          ;NEXT TEST
3900
3901      N36:    INC     R1
3902      CMP     R1,#8.
3903      BNE     S36
3904
3905

```

```

3906
3907
3908      .REM      %
3909      SET THE PORT SELECT FLOP VERIFY THAT WE CAN READ IT BACK.
3910      %
3911      ;*****
3912      ;*TEST 37      CSRA PSEL CLEAR TEST
3913      ;*****
3913      TST37:  SCOPE
3914      MOV      #1, $TIMES      ;; DO 1 ITERATION
3915      MOV      #TNO, J#TSTNM   ;; THIS SAVES TEST NUMBER
3916      MOV      #CLR, J#RHCS2  ;; CLEAR RH11 CONTROLLER
3917
3918      MOV      #STACK, SP      ; SET STACK POINTER.
3919
3920      MOV      RHCS1, R2        ; R2=RHCS1 ADDRESS.
3921      MOV      R2, REGADR
3922      CLR      $GDDAT          ; $GDDAT=S/B.
3923
3924      L40:      MOV      #PSEL, (R2) ; SET P SELECT.
3925
3926      MOV      #CLR, J#RHCS2    ; DO A CONTROLLER CLEAR.
3927
3928      MOV      (R2), $BDDAT     ; READ BACK P SELECT BIT.
3929      BIC      #1<PSEL>, $BDDAT ; SAVE ONLY PORT SELECT.
3930
3931      CMP      $BDDAT, $GDDAT   ; COMPARE RESULTS.
3932      BEQ      TST40           ; NEXT TEST
3933      ERROR   64
3934

```

```

3935 .REM %
3936 VERIFY THAT CONTROLLER CLEAR WILL CLEAR THE COMMAND REGISTER.
3937 %
3938
3939 ;*****
3940 ;*TEST 40 CSRB COMMAND REGISTER CLEAR TEST
3941 ;*****
3942 TST40: SCOPE
3943 MOV #1, $TIMES ;:DO 1 ITERATION
3944 MOV #TNO, #TSTNM ;THIS SAVES TEST NUMBER
3945 MOV #CLR, #RHCS2 ;CLEAR RH11 CONTROLLER
3946
3947 MOV #STACK, SP ;SET STACK POINTER.
3948
3949 MOV RHCS1, R2 ;R2=RHCS1 ADDRESS.
3950 MOV R2, REGADR
3951 CLR $GDDAT ;$GDDAT=S/B.
3952
3953 L41: MOV #11, (R2) ;ISSUE A DRIVE CLR AND GO.
3954
3955 MOV #CLR, #RHCS2 ;CONTROLLER CLEAR.
3956
3957 MOV (R2), $BDDAT ;READ COMMAND REGISTER.
3958 BIC #1C<?>, $BDDAT ;SAVE ONLY THE COMMAND BITS.
3959
3960 CMP $BDDAT, $GDDAT ;COMPARE RESULTS.
3961 BEQ TST41 ;NEXT TEST
3962 ERROR 65
3963

```

```

3964
3965 .REM %
3966 THE COMMAND REGISTER IS SUPPOSED TO BE SELF CLEARING THAT IS WHEN THE
3967 RH11 IS SELECTED WHEATHER OR NOT WE ADDRESS IT DIRECTLY THE LOGIC
3968 SHOULD CLEAR THE COMMAND REGISTER.
3969 THIS IS TESTED BY LOADING A COMMAND INTO IT READING ANOTHER REGISTER
3970 THAN RETURNING TO CHECK THE COMMAND REGISTER TO DETERMINE IF IT CLEARED.
3971 %
3972 ;*****
3973 ;*TEST 41 CSRB COMMAND REGISTER RESELECT CLEAR TEST
3974 ;*****
3975 TST41: SCOPE
3976 030004 000004 MOV #1,$TIMES ;DO 1 ITERATION
3977 030006 012737 000001 001212 MOV #TINO,2*TSTNM ;THIS SAVES TEST NUMBER
3978 030014 012737 000041 017330 MOV #CLR,2RHCS2 ;CLEAR RH11 CONTROLLER
3979 030022 012777 000040 164726
3980 030030 012706 001000 MOV #STACK,SP ;SET STACK POINTER.
3981
3982 030034 013702 014760 MOV RHCS1,R2 ;R2=RHCS1 ADDRESS.
3983 030040 010237 045440 MOV R2,REGADR
3984 030044 005037 001124 CLR $GDDAT ;$GDDAT=S/B.
3985
3986 030050 012737 000340 177776 MOV #340,PS ;SET PRIORITY TO #7.
3987
3988 030056 012712 000011 L42: MOV #11,(R2) ;ISSUE A DRIVE CLR AND GO
3989
3990 030062 011237 001176 MOV (R2),$TMPD ;DO A RESELECT OF THE REGISTER.
3991
3992 030066 011237 001126 MOV (R2),$BDDAT ;READ THE COMMAND REGISTER.
3993 030072 042737 177770 001126 BIC #C<7>,$BDDAT ;SAVE ONLY THE COMMAND BITS.
3994
3995 030100 023737 001126 001124 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3996 030106 001401 BEQ TST42 ;NEXT TEST
3997 030110 104066 ERROR 66
3998

```


3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029

030112 000004
030114 012737 000001 001212
030122 012737 000042 017330
030130 012777 000040 164620

030136 013702 014760
030142 010237 045440
030146 005037 001126
030152 005037 001124

030156 017737 164564 001176
030164 012777 030220 164554

030172 012706 001000
030176 005037 177776

030202 052712 000300
030206 000240
030210 000240

030212 011237 001124
030216 104067

030220 013777 001176 164520

.REM %
HERE WE TEST TO SEE IF SETTING (IE) AND (RDY) WILL CAUSE AN INTERRUPT.
%
:*****
:TEST 42 CSRB READY AND IE INTERRUPT TEST
:*****
TST42: SCOPE
MOV #1,STIMES ;:DO 1 ITERATION
MOV #TNO,STSTNM ;:THIS SAVES TEST NUMBER
MOV #CLR,RHCS2 ;:CLEAR RH11 CONTROLLER

MOV RHCS1,R2 ;R2=RPCS1 ADDRESS.
MOV R2,REGADR
CLR \$BDDAT ;\$BDDAT=WAS.
CLR \$GDDAT ;\$GDDAT=S/B.

MOV @RPVEC,STMP0 ;SAVE RH11 INTERRUPT VECTOR.
MOV #043,@RPVEC ;SET RH11 INTERRUPT VECTOR 043

L43: MOV #STACK,SP ;SET STACK POINTER.
CLR PS

BIS #IE!RDY,(R2) ;THIS WILL CAUSE AN INTERRUPT.
NOP
NOP

MOV (R2), \$GDDAT
ERROR 67

MOV STMP0,@RPVEC ;RESTORE RH11 INTERRUPT VECTOR.

4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068

030226 000004
030230 012737 000001 001212
030236 012737 000043 017330
030244 012777 000040 164504

030252 013702 014760
030256 010237 045440
030262 005037 001124

030266 017737 164454 001176
030274 012777 030324 164444

030302 012706 001000
030306 005037 177776

030312 052712 000100
030316 000240
030320 000240

030322 000411

030324 011237 001126
030330 042737 177677 001126

030336 023737 001126 001124
030344 001401

030346 104070

030350 013777 001176 164370

.REM %
THE (IE) BIT IS SET CAUSING AN INTERRUPT. AT THE COMPLETION OF THE INTERRUPT
WE CHECK TO SEE IF THE (IE) BIT HAS CLEARED.

: *TEST 43 CSRB BOES "IE" CLEAR AFTER AN INTERRUPT

†ST43: SCOPE
MOV #1,STIMES ;DO 1 ITERATION
MOV #TTNO,STSTNM ;THIS SAVES TEST NUMBER
MOV #CLR,RHCS2 ;CLEAR RH11 CONTROLLER

MOV RHCS1,R2 ;R2=RPCS1 ADDRESS.
MOV R2,REGADR
CLR \$GDDAT ;\$GDDAT=S/B.

MOV JRPVEC,STMPD ;SAVE RH11 INTERRUPT VECTOR.
MOV #044,JRPVEC ;SET VECTOR TO 044.

L44: MOV #STACK,SP ;SET STACK POINTER.
CLR PS

BIS #IE,(R2) ;THIS WILL CAUSE AN INTERRUPT
NOP
NOP

BR E44 ;NO INTERRUPT.

044: MOV (R2),SBDDAT ;READ RHCS1.
BIC #1<IE>,SBDDAT ;SAVE ONLY THE "IE" BIT.

CMP SBDDAT,\$GDDAT ;COMPARE RESULTS.
BEG R44 ;OK!

E44: ERROR 70

R44: MOV STMPD,JRPVEC ;RESTORE RH11 INTERRUPT VECTOR.

```

4069
4070
4071
4072
4073
4074
4075
4076
4077 030356 000004
4078 030360 012737 000001 001212
4079 030366 012737 000044 017330
4080 030374 012777 000040 164354
4081
4082 030402 012706 001000
4083
4084 030406 013702 014756
4085 030412 012737 000007 001124
4086
4087 030420 013705 014760
4088
4089 030424 012712 000007 L45:
4090
4091 030430 013702 014776
4092
4093 030434 012712 000377
4094
4095 030440 005012
4096
4097 030442 013702 014756
4098 030446 010237 045440
4099
4100 030452 011237 001126
4101 030456 042737 167770 001126
4102
4103 030464 023737 001126 001124
4104 030472 001401
4105 030474 104071
4106

```

```

.REM %
HERE WE VERIFY THAT WRITING INTO THE "AS" REGISTER OWILL NOT CAUSE
AN (NED) ERROR.
%
*****
*TEST 44 BCTB "AS" WRITE TEST
*****
†TST44: SCOPE
MOV #1,STIMES ;:DO 1 ITERATION
MOV #TNO,†TSTNM ;THIS SAVES TEST NUMBER
MOV #CLR,†RHCS2 ;CLEAR RH11 CONTROLLER
MOV #STACK,SP ;SET STACK POINTER.
MOV RHCS2,R2 ;R2=RH11 CS2 ADDRESS.
MOV #US4!US2!US1,$GDDAT;$GDDAT=S/B.
MOV RHCS1,R5 ;ADDRESS OF A DEVICE REGISTER.
L45: MOV #US4!US2!US1,(R2);LOAD A NON EXISTANT DEVICE.
MOV RHAS,R2 ;R2="AS" ADDRESS.
MOV #377,(R2) ;ATTEMPT TO LOAD "AS".
CLR (R2) ;CLEAR "AS".
MOV RHCS2,R2 ;R2=RH11 CS2 ADDRESS.
MOV R2,REGADR
MOV (R2),$BDDAT ;$BDDAT=WAS.
BIC #†C<NED!US4!US2!US1>,$BDDAT;SAVE NED AND DEVICE SELECTED.
CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
BEQ TST45 ;NEXT TEST
ERROR 71

```

.SBTTL EXTENDED MEMORY, DRIVE TIMING & SECTOR SELECT TESTS

4107
4108
4109
4110
4111
4112
4113
4114
4115
4116
4117
4118
4119
4120
4121
4122
4123
4124
4125
4126
4127
4128
4129
4130
4131
4132
4133
4134
4135
4136
4137
4138
4139
4140
4141
4142
4143
4144
4145
4146
4147
4148
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4160

: *TEST 45 MAKE CURRENT CYLINDER = 0

```
↑ST45: SCOPE
MOV #STACK, SP ;RESET STACK
MOV #TTNO, #TSTNM ;THIS SAVES TEST NUMBER
JSR PC, #CLDISK ;INIT DRIVE
MOV #DMD, #RHMR ;SET DIAGNOSTIC MODE
JSR RO, #MAKECYL ;SUBROUTINE TO GIVE A SEEK
;COMMAND FOLOWED BY AN INIT
;THIS SHUOLD CHANGE RHCC TO 0
```

: *TEST 46 RHCSI - 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

↑ST46: SCOPE

```
MOV #STACK, SP ;RESET STACK
MOV #TTNO, #TSTNM ;THIS SAVES TEST NUMBER
; *CHECK TO SEE IF PROGRAM IS RUNNING WITH AN RH70
TST #RH70 ;TEST FOR RH70 CONTROLLER
BEQ 30$ ;IF FLAG = 1, SKIP THIS TEST
JMP TST47 ;JUMP TO NEXT TEST -----)
30$: ;IF FLAG = 0, DO THIS TEST
```

```
JSR RO, #CLAREA ;CLEAR SIMULATED DISK
.WORD DISK ;FROM
.WORD TOLGAP+16 ;TO
.WORD 0 ;DATA
```

;THESE ARE SETUP FOR DISKLESS USE ONLY

```
MOV #FMT22, #CYL;CYLINDER 0
;16 BITS PER WORD
CLR #SECTR ;SECTOR 0 TRACK 0
CLR #KEY1 ;KEY1 0
CLR #KEY2 ;KEY2 0
```

030476 000004
030500 012706 001000
030504 012737 000045 017330
030512 004737 045674
030516 012777 000001 164254
030524 004037 050246
030530 000000
030532 000004
030534 012706 001000
030540 012737 000045 017330
030546 005737 017334
030552 001402
030554 000137 031064
030560
030560 004037 045612
030564 054256
030566 055302
030570 000000
030572 012737 010000 052340
030600 005037 052342
030604 005037 052344
030610 005037 052346

```

4161 030614 012737 000400 052406      MOV      #256., @#NOWORD ;NO OF DATA WORDS
4162 030622 012737 000001 052350      MOV      #1, @#X ;WRITE DATA
4163 030630 004537 047102      JSR      R5, @#CRC ;GO TO CALCULATE CRC
4164 030634 052340      CYL
4165 030636 054240      WCRC
    
```

:THESE ARE REGULAR SETUPS

```

4169
4170 030640 004737 045674      JSR      PC, @#CLDISK ;SETUP GENERAL REGISTERS
4171 030644 012777 177400 164100      MOV      #-256., @RHWC ;256 DATA WORDS
4172 030652 013777 001144 164074      MOV      @#STKS, @RHBA ;STARTING ADDRESS OF WRITE BUFFER
4173 030660 017737 150260 015142      MOV      @#STKS, @#TMPILL ;TEMPORARY STORAGE OF DATA
4174 030666 005077 164072      CLR      @RHDS1 ;SECTOR 0 TRACK 0
4175 030672 012777 010000 164070      MOV      #FMT22, @RHOF ;16 BITS PER WORD FORMAT
4176 030700 005077 164066      CLR      @RHCA ;CYLINDER 0
4177 030704 004737 045730      JSR      PC, @#CHECKT ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
4178 030710 104400 005124      TYPE    , @PHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
4179 ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
4180 030714 000000      HALT ;STOP THE TEST
4181 030716 013746 015172      MOV      @#WRIDAT, -(SP) ;WRITE DATA=60
4182 030722 052716 001400      BIS      #A16!A17, (SP) ;SET HIGH ORDER UNIBUS BITS
4183 030726 012611      MOV      (SP)+, @R1 ;FILL RHCSI
4184 030730 052777 000010 164020      BIS      #BAI, @RHCS2 ;SET BUS ADDRESS INHIBIT
4185 030736 005037 015126      CLR      @#ERFLGS ;CLEAR ERROR FLAG
4186 030742 004737 052200      JSR      PC, @#COMHD ;WRITE DATA
    
```

4187
4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218
4219
4220
4221

030746 004737 045374
030752 005737 015126
030756 001042
030750 013700 015142
030764 012701 054256
030770 012702 000400
030774 012737 000401 052460 1\$:
031002 020021
031004 001425
031006 013737 015142 001124
031014 014137 001126
031020 160237 052460
031024 005737 015126
031032 104037
031034 000401
031036 104040 2\$:
031040 005721 64\$:
031042 017746 150072
031046 042716 177177
031052 022726 000200
031056 001402
031060 005302 3\$:
031062 001344

052460 1\$:
001124
2\$:
64\$:
3\$:

: IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
: FROM THE "COMHD" ROUTINE IT MEANS SECTOR GAP, SYNC BYTE
: HEADER, HEADER CRC, HEADER GAP AND SYNC BYTE HAVE GONE BY
: AND SYNCs WERE CORRECTLY DETECTED.

: DATA IS TO BE CHECKED.

JSR PC, @PUTREG ; SAVE REGISTERS
TST @ERFLGS ; HAVE ANY ERRORS OCCURED?
BNE TST47 ; BRANCH OUT IF YES
MOV @TMPILL, R0 ; GOOD DATA
MOV @DISK, R1 ; DATA WRITTEN INTO "DISK"
MOV @256, R2 ; COUNTER
MOV @257, @ERWORD ; FOR ERROR WORD
CMP R0, (R1)+ ; COMPARE GOOD DATA WITH DATA ON DISK
BEQ 3\$; BRANCH IF GOOD
MOV @TMPILL, @SGDDAT ; GOOD DATA
MOV -(R1), @SBDDAT ; BAD DATA
SUB R2, @ERWORD ; ERROR WORD NO
TST @ERFLGS ; ANY ERRORS ALREADY THERE?
BNE 2\$; BRANCH IF YES
ERROR 37 ; ERROR ON WRITE DATA COMMAND
BR 64\$; SEE NEXT ERROR COMMENTS
ERROR 40 ; BRANCH TO AVOID PRINTING NEXT ERROR
TST (R1)+ ; WORD NO GIVES WORD IN ERROR
MOV @SWR, -(SP) ; ERROR OCCURED WHILE WRITING
BIC #177177, (SP) ; WITH A16 A17 OF RHCSI SET
CMP #SW07, (SP)+ ; UNDO -(R1) FOR BAD DATA
BEQ TST47 ; GET SWITCH SETTING
DEC R2 ; KEEP ONLY SWITCH 7 AND 8
BNE 1\$; IS 7 SET AND 8 RESET
 ; BRANCH OUT IF YES
 ; IF NOT COUNT 256 WORDS
 ; BRANCH IF 256 NOT DONE

: SAVE REGISTERS
: HAVE ANY ERRORS OCCURED?
: BRANCH OUT IF YES
: GOOD DATA
: DATA WRITTEN INTO "DISK"
: COUNTER
: FOR ERROR WORD
: COMPARE GOOD DATA WITH DATA ON DISK
: BRANCH IF GOOD
: GOOD DATA
: BAD DATA
: ERROR WORD NO
: ANY ERRORS ALREADY THERE?
: BRANCH IF YES
: ERROR ON WRITE DATA COMMAND
: SEE NEXT ERROR COMMENTS
: BRANCH TO AVOID PRINTING NEXT ERROR
: WORD NO GIVES WORD IN ERROR
: ERROR OCCURED WHILE WRITING
: WITH A16 A17 OF RHCSI SET
: UNDO -(R1) FOR BAD DATA
: GET SWITCH SETTING
: KEEP ONLY SWITCH 7 AND 8
: IS 7 SET AND 8 RESET
: BRANCH OUT IF YES
: IF NOT COUNT 256 WORDS
: BRANCH IF 256 NOT DONE

4222
4223
4224
4225
4226
4227
4228
4229
4230
4231
4232
4233
4234
4235
4236
4237
4238
4239
4240
4241
4242
4243
4244
4245
4246
4247
4248
4249
4250
4251
4252
4253
4254
4255
4256
4257
4258
4259
4260
4261
4262
4263
4264
4265
4266
4267
4268
4269
4270
4271
4272
4273
4274
4275

031064 000004
031066 012706 001000
031072 012737 000047 017330

031100 012737 010000 052340
031106 005037 052342

031112 005037 052344
031116 005037 052346
031122 005037 052350
031126 004537 047102
031132 052340
031134 054240

031136 004737 045674
031142 012777 177374 163602
031150 012777 016264 163576
031156 005077 163602

031162 012777 014000 163600

031170 005077 163576
031174 004737 045730
031200 104400 005124

031204 000000
031206 013711 015200

031212 004037 046366
031216 014752
031220 015026
031222 000023

```
::*****  
:*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.  
:* 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 BRING SECTOR PULSE UP WHICH SHOULD  
:* SET "DRIVE TIMING ERROR" - 'DTE'  
:*****  
TST47: SCOPE  
MOV #STACK, SP ;RESET STACK  
MOV #TTNO, @#TSTNM ;THIS SAVES TEST NUMBER  
  
;*THESE ARE TO SETUP FOR DISKLESS USE  
  
MOV #FMT22, @#CYL ;16 BITS PER WORD  
;CYLINDER 0  
CLR @#SECOTR ;SECTOR 0  
;TRACK 0  
CLR @#KEY1 ;KEY1 = 0  
CLR @#KEY2 ;KEY2 = 0  
CLR @#X ;THIS IS A READ COMMAND  
JSR R5, @#CRC ;GO TO CALCULATE CRC  
CYL  
WCRC  
  
;* THESE ARE REGULAR SETUPS  
  
JSR PC, @#CLDISK ;SETUP GENERAL REGISTERS  
MOV #-260, @RHWC ;256 DATA WORDS, 4 HEADER WORDS  
MOV #REINT0, @RHBA ;STARTING ADDRESS OF BUFFER  
CLR @RHDSST ;TRACK = 0  
;SECTOR = 0  
MOV #FMT22!ECI, @RHOF ;16 BITS PER WORD  
;ECC CORRECTION INHIBITED  
CLR @RHCA ;CYLINDER = 0  
JSR PC, @#CHECKT ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T  
TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE  
;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1  
HALT ;STOP THE TEST  
MOV @#REFCR, @R1 ;READ HEADER AND DATA = 72  
  
;*READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'  
  
JSR R0, @#SAVER ;READ IN SEQUENCE  
RHWC ;FROM HARDWARE REGISTER  
WC ;INTO CORE AT LOCATION  
19. ;NUMBER OF REGISTERS TO READ
```

```

4276
4277
4278
4279
4280
4281
4282
4283 031224 012737 177777 015146 MOV # -1, D#TESDTE ;SET DTE TEST
4284 031232 012737 177777 015126 MOV # -1, D#ERFLG$ ;THIS WILL BRING THE READ HEADER
4285 ;AND DATA PROCESS OUT AFTER THE
4286 ;HEADER HAS BEEN CORRECTLY READ
4287
4288 031240 004737 052200 JSR PC, D#COMHD ;ISSUE 'GO', SEARCH FOR THE SECTOR
4289 ;AND READ THE HEADER
4290
4291 031244 017737 163510 001176 SETCK1: MOV D#RHCSI, D#$TMPD ;READ CS1 TO CHECK FOR ANY READ ERRORS
4292 031252 032737 100000 001176 BIT #SC, D#$TMPD ;TEST FOR "SPECIAL CONDITION" - 'SC'
4293 031260 001405 BEQ B$ ;CONTINUE WITH TEST IF 'SC' = 0 (NO ERRORS)
4294 031262 004737 045374 JSR PC, D#PUTREG ;READ & SAVE ALL REGISTERS AGAIN IF AN
4295 ;UNDEFINED DATA TRANSFER ERROR OCCURRED
4296 031266 104040 ERROR 40 ;READ/WRITE HEADER & DATA ERROR DURING
4297 031270 000137 031556 JMP TST50 ; 'DTE' TEST - ABORT THE TEST
4298
4299 031274 B$: ;NOW THE HEADER HAS BEEN READ OK
4300 ;*NOW 560 SECTOR CLOCKS WILL BE GIVEN
4301 ;*GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
4302 ;*GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 2
4303
4304 ;*THESE 560 SECTOR CLOCKS ARE DIVIDED INTO TWO GROUPS
4305 ;*24 SECTOR CLOCKS WITH NORMAL DIAGNOSTIC DATA CLOCKS.
4306 ;*AND 536 SECTOR CLOCKS WITHOUT ANY DIAGNOSTIC DATA CLOCKS.
4307
4308
4309 ;*THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC DATA CLOCKS
4310 ;*WHICH EQUALS 3 BYTES OF DATA
4311
4312 031274 012701 000030 MOV #24, R1 ;LOAD COUNTER
4313 031300 013700 015000 MOV D#RHMR, R0 ;GET RHMR ADDRESS
4314 031304 012710 000001 MOV #DMD, DRO ;SET DIAGNOSTIC MODE
4315 031310 052710 000012 1$: BIS #MSTCK!MCLK, DRO ;SET SECTOR CLOCK AND DATA CLOCK
4316 031314 042710 000012 BIC #MSTCK!MCLK, DRO ;CLEAR SECTOR CLOCK AND DATA CLOCK
4317 031320 012702 000007 MOV #7, R2 ;LOAD COUNTER FOR DIAGNOSTIC CLOCKS
4318 031324 052710 000002 4$: BIS #MCLK, DRO ;SET CLOCK
4319 031330 042710 000002 BIC #MCLK, DRO ;CLEAR CLOCK
4320 031334 005302 DEC R2 ;COUNT TO 7
4321 031336 001372 BNE 4$ ;BRANCH IF 7 NOT DONE
4322 031340 005301 DEC R1 ;COUNT TO 24
4323 031342 001362 BNE 1$ ;BRANCH IF 24 NOT DONE
4324
4325 ;*THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
4326
4327 031344 012701 001030 5$: MOV #536, R1 ;LOAD SECTOR CLOCK COUNTER
4328 031350 052710 000010 BIS #MSTCK, DRO ;SET SECTOR CLOCK
4329 031354 042710 000010 BIC #MSTCK, DRO ;CLEAR SECTOR CLOCK
    
```



```

4330 031360 005301          DEC      R1          ;COUNT
4331 031362 001372          BNE      S$          ;BRANCH IF 536 NOT DONE
4332
4333
4334                          ;*NOW 'DTE' SHOULD BE SET
4335                          ;*CHANGE SAVED REGISTERS TO EXPECTED VALUES
4336
4337 031364 012737 177400 015026  MOV      #-256, @#WC      ;SAVED RHWC
4338 031372 012737 016274 015030  MOV      #REINTO+<4.*2>, @#BA ;SAVED RHBA
4339 031400 052737 140000 015034  BIS      #SCITRE, @#CSI    ;SAVED RHCSI
4340 031406 052737 010000 015036  BIS      #DTE, @#ERI      ;SAVED RHERI
4341 031414 012737 000401 015054  MOV      #401, @#MR       ;SAVED RHMR
4342 031422 052737 140000 015056  BIS      #ATA!ERR, @#DS1  ;SAVED RHDS1
4343 031430 012737 000100 015070  MOV      #100, @#LA       ;SAVED RHLA
4344 031436 012737 000001 015040  MOV      #1, @#DST        ;SAVED RHDST
4345 031444 013737 015136 015052  MOV      @#ATTENT, @#AS   ;SAVED RHAS
4346
4347
4348                          ;*NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS
4349                          ;*CAN BE DONE (USE THE 'WRFROM' SAVE BUFFER THIS TIME)
4350
4351 031452 004037 046366      JSR      RO, @#SAVER      ;READ IN SEQUENCE
4352 031456 014752              RHWC                    ;FROM HARDWARE REGISTER
4353 031460 015220              WRFROM                  ;INTO CORE BUFFER
4354 031462 000023              19.                     ;NUMBER OF REGISTERS TO READ
4355
4356                          ;*FOR RHAS UPPER BYTE
4357 031464 113737 015053 015245  MOVB     @#AS+1, @#WRFROM+25 ;UPPER RHAS
4358
4359                          ;*COMPARE THE HEADER READ
4360
4361 031472 004037 046570      JSR      RO, @#COMPAR    ;COMPARE
4362 031476 052340              CYL                    ;GOOD BUFFER
4363 031500 016264              REINTO                  ;TEST BUFFER
4364 031502 000004              4.                      ;NUMBER
4365 031504 031512              6$                      ;RETURN FOR ERROR
4366 031506 031512              6$                      ;SAME
4367 031510 031516              7$                      ;RETURN FOR GOOD COMPARISON
4368 031512 104010              6$: ERROR 10            ;HEADER READ IN DURING THIS TEST IS
4369                                     ;IN ERROR
4370
4371 031514 000207              RTS      PC              ;RETURN
4372
4373 031516              7$:                                ;GOOD COMPARISON, CONTINUE
4374
4375
4376                          ;*COMPARE REGISTERS BEFORE COMMAND WITH REGISTERS AFTER COMMAND
4377
4378 031516 004037 046570      JSR      RO, @#COMPAR    ;COMPARE
4379 031522 015026              WC                      ;INITIAL SNAPSHOT BUFFER (CHANGED)
4380 031524 015220              WRFROM                  ;TEST SNAPSHOT BUFFER
4381 031526 000022              18.                     ;NUMBER OF REGISTERS
4382 031530 031536              2$                      ;RETURN FOR ERROR
4383 031532 031536              2$                      ;SAME

```

W

```

4384 031534 031556          3$:          ;RETURN FOR GOOD COMPARISON
4385
4386 031536 013705 052460    2$:      MOV      @#ERWORD,R5      ;GETTING READY TO INDEX
4387 031542 060505          ADD      R5,R5          ;DOUBLE ERROR WORD
4388 031544 016537 014750 045440  MOV      R#WC-2(R5),@#REGADR ;FAILING REGISTER
4389 031552 104001          ERROR    1              ;IMPROPER REGISTER
4390                                ;CHANGE AFTER FORCING
4391                                ;'DTE' ERROR
4392 031554 000207          RTS      PC              ;RETURN
4393
4394 031556          3$:          ;GOOD - REGISTERS OK, GO ON TO NEXT TEST
4395
4396

```

2.7

4397
4398
4399
4400
4401
4402
4403
4404
4405
4406
4407
4408
4409
4410
4411
4412
4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429
4430
4431
4432
4433
4434
4435
4436
4437
4438
4439
4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4450

: *TEST 50 DRIVE TIMING ERROR

: * A WRITE DATA COMMAND IS STARTED ON CYLINDER 0, SECTOR
: * 0, TRACK 0, 256 WORDS.

: * 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'

ST50: SCOPE
MOV #STACK, SP ; RESET STACK
MOV #TTNO, @#TSTNM ; THIS SAVES TEST NUMBER
; * THESE ARE TO SETUP FOR DISKLESS USE
MOV #FMT22, @#CYL ; 16 BITS PER WORD
; CYLINDER 0
CLR @#SECTR ; SECTOR 0
; TRACK 0
CLR @#KEY1 ; KEY1 = 0
CLR @#KEY2 ; KEY2 = 0
MOV #-1, @#X ; THIS IS FOR WRITE DATA COMMAND
JSR R5, @#CRC ; GO TO CALCULATE CRC
CYL
WCRC

; * THESE ARE REGULAR SETUPS & CHECKS

JSR PC, @#CLDISK ; SETUP GENERAL REGISTERS
MOV #-256, @RHWC ; 256 DATA WORDS
MOV #WRFROM, @RHBA ; STARTING ADDRESS OF BUFFER
CLR @RH DST ; TRACK = 0
; SECTOR = 0
MOV #FMT22, @RHOF ; 16 BITS PER WORD
; ECC CORRECTION INHIBITED
CLR @RHCA ; CYLINDER = 0
JSR PC, @#CHECKT ; CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
TYPE , CPHALT ; CANNOT CONTINUE TESTING IF ANY OF THE
; ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
HALT ; STOP THE TEST
MOV @#WRIDAT, @R1 ; WRITE DATA = 60

; * READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'

JSR R0, @#SAVER ; READ REGISTERS IN SEQUENCE
RHWC ; FROM HARDWARE REGISTER
WC ; INTO CORE BUFFER LOCATION

M12

MNDEC-11-DZRJH-A, RPO4/5/6 DSKLS CONTROLLER TST-PT 2
DZRJHA.TST ↑50 DRIVE TIMING ERROR

MACY11 27(655) 30-MAR-76 20:59 PAGE 108

SEQ 0154

4451 031716 000023

19.

;NUMBER OF REGISTERS TO READ

```

4452
4453
4454
4455
4456
4457
4458
4459 031720 012737 177777 015146 MOV #-1,@#TESDTE ;SET DTE TEST
4460 031726 012737 177777 015126 MOV #-1,@#ERFLG$ ;THIS WILL BRING THE READ HEADER
4461 ;AND DATA PROCESS OUT AFTER THE
4462 ;HEADER HAS BEEN CORRECTLY READ
4463
4464 031734 004737 052200 JSR PC,@#COMHD ;ISSUE 'GO', SEARCH FOR SECTOR,
4465 ;READ HEADER AND DATA.
4466
4467 031740 017737 163014 001176 SETCK2: MOV @RHCS1,@#$TMPD ;READ CS1 TO CHECK FOR READ ERRORS
4468 031746 032737 100000 001176 BIT #SC,@#$TMPD ;TEST FOR "SPECIAL CONDITION" - 'SC'
4469 031754 001405 BEQ 6$ ;CONTINUE TESTING IF NO ERROR
4470 031756 004737 045374 JSR PC,@#PUTREG ;READ & SAVE REGISTERS AGAIN IF UNDE-
4471 ;FINED ERROR HAS OCCURRED
4472 031762 104040 ERROR 40 ;THERE WAS A READ/WRITE HEADER ERROR
4473 ;DURING 'DTE' TEST SETUP
4474 031764 000137 032302 JMP TST51 ;ABORT THE TEST
4475
4476 031770 6$: ;NOW THE HEADER HAS BEEN READ.
4477 ;*560 SECTOR CLOCKS WILL BE GIVEN -
4478 ;*GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
4479 ;*GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 2
4480
4481 ;*THESE 560 SECTOR CLOCKS ARE DIVIDED INTO TWO GROUPS
4482 ;*24 SECTOR CLOCKS WITH NORMAL DIAGNOSTIC DATA CLOCKS
4483 ;*AND 536 SECTOR CLOCKS WITHOUT ANY DIAGNOSTIC DATA CLOCKS
4484
4485
4486 ;*THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC DATA CLOCKS
4487
4488 031770 012701 000030 MOV #24,R1 ;LOAD SECTOR CLOCK COUNTER
4489 031774 013700 015000 MOV @RHMR,R0 ;GET RHMR ADDRESS
4490 032000 012710 000001 MOV #DMD,@R0 ;SET DIAGNOSTIC MODE
4491 032004 052710 000012 1$: BIS #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND DATA CLOCK
4492 032010 042710 000012 BIC #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND DATA CLOCK
4493 032014 012702 000007 MOV #7,R2 ;LOAD COUNTER FOR DIAGNOSTIC DATA CLOCKS
4494 032020 052710 000002 4$: BIS #MCLK,@R0 ;SET CLOCK (DATA)
4495 032024 042710 000002 BIC #MCLK,@R0 ;CLEAR CLOCK (DATA)
4496 032030 005302 DEC R2 ;COUNT
4497 032032 001372 BNE 4$ ;BRANCH IF 7 NOT DONE
4498 032034 005301 DEC R1 ;COUNT
4499 032036 001362 BNE 1$ ;BRANCH IF 24 NOT DONE
4500
4501
4502 ;*THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
4503 032040 012701 001030 MOV #536,R1 ;LOAD SECTOR CLOCK COUNTER
4504 032044 052710 000010 5$: BIS #MSTCK,@R0 ;SET SECTOR CLOCK
4505 032050 042710 000010 BIC #MSTCK,@R0 ;CLEAR SECTOR CLOCK

```

```

4506 032054 005301          DEC      R1          :COUNT
4507 032056 001372          BNE      S$          :BRANCH IF 536 NOT DONE
4508
4509
4510
4511
4512 032060 017737 162726 015066      : *ECC PATTERN REGISTER IS NOT CHECKED
MOV      @RHEC2,@#EC2      ;RHEC2 IS NOT CHECKED
4513
4514
4515
4516
4517
4518 032066 005737 017334          TST      @#RH70       :CHECK FOR RH70 CONTROLLER
SEQ      7$              ;SKIP RH70 CODE AND DO RH11 IF NOT
4519
4520 032074 012737 177416 015026      MOV      #-242,@#WC    :SAVED RHWC
4521 032102 012737 015254 015030      MOV      @WRFROM+<14.*2>,@#BA :SAVED RHBA
4522 032110 052737 000300 015032      BIS      @IR!OR,@#CS2  :SAVED RHCS2
4523 032116 000414          BR        B$          ;SKIP NEXT RH11 CODE
4524
4525 032120 012737 177511 015026 7$: MOV      #-183,@#WC    :SAVED RHWC
4526 032126 012737 015442 015030      MOV      @WRFROM+<73.*2>,@#BA :SAVED RHBA
4527 032134 042737 000100 015032      BIC      @IR,@#CS2    :SAVED RHCS2
4528 032142 052737 000200 015032      BIS      @OR,@#CS2    :SAVED RHCS2
4529
4530 032150 052737 140000 015034 8$: BIS      @SC!TRE,@#CS1  :SAVED RHCS1
4531 032156 052737 010000 015036      BIS      @DTE,@#ER1   :SAVED RHER1
4532 032164 012737 000201 015054      MOV      @DENVL!DMD,@#MR :SAVED RHMR
4533 032172 052737 140000 015056      BIS      @ATA!ERR,@#DS1 :SAVED RHDS1
4534 032200 012737 000100 015070      MOV      @100,@#LA    :SAVED RHLA
4535 032206 012737 000001 015040      MOV      @1,@#DST     :SAVED RHDST
4536 032214 013737 015136 015052      MOV      @#ATTENT,@#AS :SAVED RHAS
4537
4538
4539
4540
4541
4542
4543
4544 032232 000023          JSR      RD,@#SAVER   :READ IN SEQUENCE
RHW      :FROM HARDWARE REGISTER
REINTC   :INTO CORE BUFFER LOCATION
19.      :NUMBER OF REGISTERS TO READ
4545
4546
4547 032234 113737 015053 016311      : *FOR RHAS UPPER BYTE
MOVB     @#AS+1,@#REINTO+25 ;UPPER RHAS
4548
4549
4550
4551
4552
4553
4554 032242 004037 046570          JSR      RD,@#COMPAR  :COMPARE
WC       :CHANGED INITIAL SNAPSHOT BUFFER
REINTC   :SNAPSHOT BUFFER AFTER COMMAND
18.      :NUMBER OF REGISTERS TO COMPARE
4555 032246 015026          2$      :RETURN FOR ERROR
4556 032250 016264          2$      :SAME
4557 032252 000022          3$      :RETURN FOR GOOD COMPARISON
4558 032254 032262
4559 032256 032262
4560 032260 032302

```

```

4560
4561 032262 013705 052460 2$: MOV 3#ERWORD,RS ;GETTING READY TO INDEX
4562 032266 060505 ;ADD R5,R5 ;DOUBLE ERROR WORD
4563 032270 016537 014750 045440 MOV RHWC-2(R5),3#REGADR ;FAILING REGISTER
4564 032276 104001 ERROR 1 ;IMPROPER REGISTER
4565 ;CHANGE AFTER FORCING
4566 ;'DTE' ERROR
4567 032300 000207 RTS PC ;RETURN
4568
4569 032302 3$: ;GOOD. REGISTERS OK - GO ON TO NEXT TEST
4570
4571

```

```

4572
4573
4574
4575
4576
4577
4578
4579
4580
4581
4582
4583
4584 032302 000004
4585 032304 012706 001000
4586 032310 012737 000051 017330
4587
4588
4589
4590 032316 012737 010000 055476
4591
4592 032324 005037 055500
4593 032330 005037 055502
4594 032334 005037 055504
4595 032340 012737 000400 055536
4596 032346 004537 047102
4597 032352 055476
4598 032354 055506
4599
4600
4601
4602 032356 004737 045674
4603 032362 012777 177374 162362
4604 032370 012777 015220 162356
4605 032376 005077 162362
4606
4607 032402 012777 014000 162360
4608
4609 032410 005077 162356
4610 032414 004737 045730
4611 032420 104400 005124
4612
4613 032424 000000
4614 032426 013711 015174
4615
4616
4617
4618 032432 004037 046366
4619 032436 014752
4620 032440 015026
4621 032442 000023

```

```

*****
*TEST 51 DRIVE TIMING ERROR
*****
* A WRITE HEADER AND DATA COMMAND IS GIVEN
* TO CYLINDER 0, SECTOR 0, TRACK 0, 256. WORDS.
*
* 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'
*****
↑ST51: SCOPE
MOV #STACK, SP ;RESET STACK
MOV #TTNO, #TSTNM ;THIS SAVES TEST NUMBER

*THESE ARE TO SET JP FOR DISKLESS USE ONLY

MOV #FMT22, #WCYL ;FORMAT 22=16 BITWORDS AND
;CYLINDER 0
CLR #WSECTR ;TRACK=0, SECTOR=0
CLR #WKEY1 ;KEY1=0
CLR #WKEY2 ;KEY2=0
MOV #256, #FNWORD ;256 DATAWORDS
JSR RS, #CRC ;GO TO CALCULATE CRC
WCVL
GCRC

* THESE ARE REGULAR SETUPS & CHECKS

JSR PC, #CLDISK ;SETUP GENERAL REGISTERS
MOV #-260, #RHWC ;256 DATA WORDS & 4 HEADER WORDS
MOV #WRFROM, #RHBA ;STARTING ADDRESS OF BUFFER
CLR #RHDSST ;TRACK = 0
;SECTOR = 0
MOV #FMT22!ECI, #RHOF ;16 BITS PER WORD
;ECC CORRECTION INHIBITED
CLR #RHCA ;CYLINDER = 0
JSR PC, #CHECKT ;CHECK OVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK & 1
;STOP THE TEST
MOV #WRIFOR, #RI ;WRITE HEADER AND DATA = 62

*READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'

JSR RO, #SAVER ;READ IN SEQUENCE
RHWC ;FROM HARDWARE REGISTER
WC ;INTO CORE BUFFER LOCATION
19. ;NUMBER OF REGISTERS TO READ

```



```

4622
4623
4624
4625
4626
4627
4628
4629 032444 012737 177777 015146 MOV # -1,2#TESDTE ;SET DTE TEST
4630 032452 012737 177777 015126 MOV # -1,2#ERFLGS ;THIS WILL BRING THE READ HEADER
4631 ;AND DATA PROCESS OUT AFTER THE
4632 ;HEADER HAS BEEN CORRECTLY READ
4633
4634 032460 004737 055322 JSR PC,2#COMWHD ;ISSUE 'GO', SEARCH FOR SECTOR,
4635 ;WRITE HEADER AND DATA.
4636
4637 032464 017737 162270 001176 SETCK3: MOV 2#RHCS1,2#STMPD ;READ CS1 TO CHECK FOR ERRORS DURING WRITE
4638 032472 032737 100000 001176 BIT #SC,2#STMPD ;TEST FOR "SPECIAL CONDITION" - 'SC'
4639 032500 001405 BEQ 4$ ;CONTINUE TEST IF NO ERROR ('SC' = 0)
4640 032502 004737 045374 JSR PC,2#PUTREG ;READ & SAVE REGISTERS AGAIN IF ERROR
4641 032506 104040 ERROR 40 ;THERE WAS A READ/WRITE HEADER ERROR
4642 ;DURING 'DTE' TEST SETUP
4643 032510 000137 032742 JMP TST52 ;ABORT THE TEST AT THAT POINT
4644
4645 032514 4$: ;NOW SECTOR HAS BEEN FOUND OK
4646
4647 ;*609 SECTOR CLOCKS WILL BE GIVEN,
4648 ;*39 BYTES FOR SECTOR GAP,
4649 ;*1 BYTE FOR HEADER SYNC,
4650 ;*8 BYTES FOR HEADER,
4651 ;*GAP 1 1 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
4652 ;*GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 3
4653
4654 ;*THIS GIVES 609 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
4655
4656
4657 032514 012701 001141 5$: MOV #609,R1 ;LOAD SECTOR CLOCK COUNTER
4658 032520 052710 000010 BIS #MSTCK,2#R0 ;SET SECTOR CLOCK
4659 032524 042710 000010 BIC #MSTCK,2#R0 ;CLEAR SECTOR CLOCK
4660 032530 005301 DEC R1 ;COUNT
4661 032532 001372 BNE 5$ ;BRANCH IF 536 NOT DONE
4662
4663 ;*NOW 'DTE' SHOULD BE SET, CHANGE SAVED REGISTERS
4664 ;TO EXPECTED VALUES
4665
4666 032534 005737 017334 TST 2#RH70 ;CHECK FOR RH70 CONTROLLER
4667 032540 001407 BEQ 6$ ;SKIP RH70 CODE AND DO RH11 IF NOT
4668
4669 032542 012737 177404 015026 MOV #-252,2#WC ;SAVED RHWC
4670 032550 012737 015240 015030 MOV #WFR0M+<8.*2>,2#BA ;SAVED RHBA
4671 032556 000406 BR 7$ ;SKIP NEXT RH11 CODE
4672
4673 032560 012737 177477 015026 6$: MOV #-193,2#WC ;SAVED RHWC
4674 032566 012737 015426 015030 MOV #WFR0M+<67.*2>,2#BA ;SAVED RHBA
4675

```

```

4676 032574 052737 140000 015034 7$: BIS      #SC!TRE, @#CS1 ;SAVED RHCS1
4677 032602 042737 000100 015032 BIC      #IR, @#CS2 ;SAVED RHCS2
4678 032610 052737 000200 015032 BIS      #OR, @#CS2 ;SAVED RHCS2
4679 032616 052737 010000 015036 BIS      #DTE, @#ER1 ;SAVED RHER1
4680 032624 012737 000401 015054 MOV      #401, @#MR ;SAVED RHMR
4681 032632 052737 140000 015056 BIS      #ATA!ERR, @#DS1 ;SAVED RHDS1
4682 032640 012737 000100 015070 MOV      #100, @#LA ;SAVED RHLA
4683 032646 012737 000001 015040 MOV      #1, @#DST ;SAVED RHDST
4684 032654 013737 015136 015052 MOV      @#ATTENT, @#AS ;SAVED RHAS
4685
4686 ;*NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN BE DONE
4687
4688 032652 004037 046366 JSR      RO, @#SAVER ;READ IN SEQUENCE
4689 032666 014752 RHW C ;FROM HARDWARE REGISTER
4690 032670 016264 REINTO ;INTO CORE BUFFER LOCATION
4691 032672 000023 19. ;NUMBER OF REGISTERS TO READ
4692
4693 ;*FOR RHAS UPPER BYTE
4694 032674 113737 015053 016311 MOV B @#AS+1, @#REINTO+25 ;UPPER RHAS
4695
4696 ;*COMPARE CHANGED REGISTER SNAPSHOT BEFORE COMMAND
4697 ;*WITH REGISTER SNAPSHOT AFTER COMMAND
4698
4699 032702 004037 046570 JSR      RO, @#COMPAR ;COMPARE
4700 032706 015026 W C ;CHANGED REGISTER SNAPSHOT
4701 032710 016264 REINTO ;SNAPSHOT AFTER COMMAND
4702 032712 000022 18. ;NUMBER OF REGISTERS TO COMPARE
4703 032714 032722 2$ ;RETURN FOR ERROR
4704 032716 032722 2$ ;SAME
4705 032720 032742 3$ ;RETURN FOR GOOD COMPARISON
4706
4707 032722 013705 052460 2$: MOV @#ERWORD, R5 ;GETTING READY TO INDEX
4708 032726 060505 ADD R5, R5 ;DOUBLE ERROR WORD
4709 032730 016537 014750 045440 MOV RHW C-2(R5), @#REGADR ;FAILING REGISTER
4710 032736 104001 ERROR 1 ;IMPROPER REGISTER
4711 ;CHANGE AFTER FORCING
4712 ;'DTE' ERROR
4713 032740 000207 RTS PC ;RETURN
4714
4715 032742 3$: ;GOOD, REGISTERS COMPARE OK
4716 ;GO ON TO THE NEXT TEST
4717

```

```

4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731 032742 000004
4732 032744 012706 001000
4733 032750 012737 000052 017330
4734 032756 004737 045674
4735
4736 032762 012737 000026 015150
4737
4738
4739
4740
4741
4742 032770 005037 033074
4743 032774 012737 000025 033100
4744 033002 012737 000025 033104
4745 033010 005037 033132
4746 033014 005037 033214
4747 033020 005037 033244
4748 033024 012737 000025 033252
4749 033032 012737 000025 033260
4750 033040 005037 033320
4751
4752
4753 033044
4754 033044 012700 054160
4755 033050 012701 000460
4756 033054 005020
4757 033056 005301
4758 033060 001375
4759
4760
4761
4762 033062 012700 015220
4763
4764
4765 033066 012720 010000
4766
4767 033072 012720 000000
4768 033076 012720 000025
4769 033102 012720 000025
4770
4771

```

```

*****
;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
*****
T52: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
JSR PC,@#CLDISK ;SETUP GENERAL REGISTERS & CLEAR
; THE DRIVE
MOV #22.,@#TAGDTE ;22 SECTORS
; THIS TEST REPEATS
; ITSELF 22 TIMES

; *THE FOLLOWING INITIALIZES FOR SECTOR 0
CLR @#SS3+2 ;HEADER (SECTOR)
MOV #21.,@#SS4+2 ;HEADER (KEY1)
MOV #21.,@#SS5+2 ;HEADER (KEY2)
CLR @#SS7+2 ;DATA (SECTOR)
CLR @#SS10+2 ;DATA
CLR @#SS12+2 ;SECTOR (SIMULATED DISK)
MOV #21.,@#SS13+2 ;KEY1 (SIMULATED DISK)
MOV #21.,@#SS14+2 ;KEY2 (SIMULATED DISK)
CLR @#SS15+2 ;SECTOR (RHDS)

; *CLEAR SIMULATED DISK AREA
SS1:
1$: MOV #SECGAP,R0 ;POINTER
MOV #304.,R1 ;COUNTER
2$: CLR (R0)+ ;CLEAR SIMULATED DISK AREA
DEC R1 ;COUNT
BNE 2$

; *SETUP WRITE FROM BUFFER
MOV #WRFROM,R0

; *HEADER
MOV #FMT22,(R0)+ ;FORMAT 16 BITS PER WORD
; CYLINDER 0
SS3: MOV #0,(R0)+ ;SECTOR TO VARY
SS4: MOV #21.,(R0)+ ;KEY1 TO VARY
SS5: MOV #21.,(R0)+ ;KEY2 TO VARY

; *DATA IN WRITE FROM BUFFER ALTHOUGH THIS IS DATA AND NOT

```

```

4772                                     ;*HEADER, THE SECTOR WITH SYNC BYTES WILL BE GIVEN AS DATA.
4773
4774                                     ;*DATA IS - 19 WORDS OF ZEROS - SYNC WORDS, 4 HEADER WORDS
4775                                     ;*1 CRC WORD, 5 WORDS OF ZEROS, 1 SYNC WORD, 100 ZEROS
4776                                     ;*(DATA), 1 SYNC WORD, 70 SECTOR NUMBER TO VARY
4777
4778 033106 012705 000023                6$: MOV #19.,R5 ;COUNTER
4779 033112 005020                      CLR (R0)+ ;19 ZEROS
4780 033114 005305                      DEC R5 ;COUNT
4781 033116 001375                      BNE 6$ ;19 DONE?
4782 033120 013720 052442                MOV @#RSYNC,(R0)+ ;SYNC = 14400
4783 033124 012720 010000                MOV @#FMT22,(R0)+ ;CYLINDER 0
4784 033130 012720 000000                SS7: MOV #0,(R0)+ ;SECTOR TO VARY
4785 033134 005020                      CLR (R0)+
4786 033136 005020                      CLR (R0)+
4787 033140 004537 047102                JSR R5,@#CRC ;CALCULATE CRC FOR ABOVE 4 WORDS
4788 033144 015270                      WRFROM+50 ;4 WORDS START FROM HERE
4789 033146 015300                      WRFROM+60 ;PUT CRC HERE
4790
4791 033150 005720                      TST (R0)+ ;INCREMENT R0
4792
4793 033152 012705 000005                8$: MOV #5.,R5
4794 033156 005020                      CLR (R0)+ ;5 WORDS OF ZEROS
4795 033160 005305                      DEC R5 ;COUNT
4796 033162 001375                      BNE 8$ ;BRANCH IF 5 NOT DONE
4797
4798 033164 013720 052442                MOV @#RSYNC,(R0)+ ;SYNC = 14400
4799
4800 033170 012705 000144                9$: MOV #100.,R5
4801 033174 005020                      CLR (R0)+ ;100 WORDS OF ZEROS
4802 033176 005305                      DEC R5
4803 033200 001375                      BNE 9$
4804
4805 033202 013720 052442                MOV @#RSYNC,(R0)+ ;SYNC = 14400
4806 033206 012705 000106                MOV #70.,R5
4807 033212 012720 000000                SS10: MOV #0,(R0)+ ;SECTOR TO VARY
4808 033216 005305                      DEC R5
4809 033220 001374                      BNE SS10
4810
4811                                     ;*CLEAR REST OF 256 WORDS THAT IS 54 WORDS OF ZEROS
4812
4813 033222 012705 000066                11$: MOV #54.,R5
4814 033226 005020                      CLR (R0)+
4815 033230 005305                      DEC R5
4816 033232 001375                      BNE 11$
4817
4818                                     ;*THESE ARE TO BE SET UP FOR DISKLESS USE ONLY
4819
4820 033234 012737 010000 055476          MOV @#FMT22,@#WCYL ;FORMAT = 16 BIT WORDS
4821                                     ;CYLINDER = 0
4822 033242 012737 000000 055500          SS12: MOV #0,@#WSECTR ;SECTOR TO VARY
4823 033250 012737 000025 055502          SS13: MOV #21.,@#WKEY1 ;KEY1 TO VARY
4824 033256 012737 000025 055504          SS14: MOV #21.,@#WKEY2 ;KEY2 TO VARY
4825 033264 012737 000312 055536          MOV #202.,@#FNWORD ;202 DATA WORDS

```

```

4826 033272 004537 047102      JSR    RS, @#CRC      ;CALCULATE CRC
4827 033276 055476              WCYL                ;FIRST WORD
4828 033300 055506              GCRC                 ;PUT HERE
4829
4830                               ;*THESE ARE REGULAR SETUPS
4831
4832 033302 012777 177400 161442  MOV    #-256, @RHWC   ;202 DATA, 4 HEADER
4833 033310 012777 015220 161436  MOV    @WRFROM, @RHBA ;FILL BUS ADDRESS
4834 033316 012777 000000 161440  SS15: MOV    #0, @RHDS1    ;SECTOR TO VARY
4835 033324 013777 015174 161426  MOV    @#WRIFOR, @RHCS1 ;GET READY TO DO
4836                                     ;WRITE HEADER AND DATA
4837                                     ;WITH 62 FUNCTION CODE IN RHCS1
4838 033332 012777 010000 161430  MOV    #FMT22, @RHOF  ;16 BITS PER WORD FORMAT
4839 033340 005077 161426          CLR    @RHCA          ;CYLINDER = 0
4840
4841 033344 005037 015126          CLR    @#ERFLGS      ;CLEAR ERROR FLAG
4842
4843 033350 004737 045730          JSR    PC, @#CHECKT   ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
4844 033354 104400 005124          TYPE  ,CPHALT        ;CANNOT CONTINUE TESTING IF ANY OF THE
4845                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
4846 033360 000000          HALT                ;STOP THE TEST
4847
4848 033362 004737 055322          JSR    PC, @#COMWHD   ;ISSUE 'GO', COUNT SECTOR CLOCKS,
4849                                     ;WRITE HEADER AND DATA
4850 033366 005737 015126          TST    @#ERFLGS      ;HAVE ANY ERRORS OCCURRED ?
4851 033372 001051          BNE    TST53         ;EXIT IF YES-----)
4852
4853 033374 004737 046120          JSR    PC, @#CHECKE   ;CHECK THAT DVA, RDY, DPR, DRY = 1
4854 033400 104400 005124          TYPE  ,CPHALT        ;CANNOT CONTINUE IF THEY DON'T
4855 033404 000000          HALT                ;STOP THE TEST AND RESTART PROGRAM
4856
4857                               ;*NOW COMPARE "DISK" BUFFER WITH "REINTO" BUFFER
4858 033406 004037 046570          JSR    RO, @#COMPAR   ;CHECK
4859 033412 015230          WRFROM+8.           ;GOOD BUFFER
4860 033414 054256          DISK                ;TEST BUFFER
4861 033416 000400          256.                ;NUMBER OF WORDS
4862 033420 033426          16$                 ;RETURN POINT FOR ERROR HEADER
4863 033422 033432          17$                 ;RETURN POINT FOR ERROR DATA
4864 033424 033436          18$                 ;RETURN FOR GOOD COMPARISON
4865 033426 104007          16$: ERROR 7
4866 033430 000207          RTS    PC
4867 033432 104010          17$: ERROR 10
4868 033434 000207          RTS    PC
4869
4870                               ;*THE FOLLOWING INCREMENTS ARE TO CHANGE THE ABOVE SET UP
4871                               ;*TO WRITE ON THE NEXT SECTOR
4872
4873 033436 005237 033074          18$: INC    @#SS3+2    ;HEADER (SECTOR)
4874 033442 005337 033100          DEC    @#SS4+2       ;HEADER (KEY1)
4875 033446 005337 033104          DEC    @#SS5+2       ;HEADER (KEY2)
4876 033452 005237 033132          INC    @#SS7+2       ;DATA (SECTOR)
4877 033456 005237 033214          INC    @#SS10+2      ;DATA
4878 033462 005237 033244          INC    @#SS12+2     ;SECTOR (SIMULATED DISK)
4879 033466 005337 033252          DEC    @#SS13+2     ;KEY1 (SIMULATED DISK)
    
```

4880	033472	005337	033260		DEC	2#SS14+2	;KEY2 (SIMULATED DISK)
4881	033476	005237	033320		INC	2#SS15+2	;SECTOR (RHDST)
4882							
4883	033502	005337	015150	SS2:	DEC	2#TAGDTE	;COUNT DOWN FOR 22 SECTORS
4884	033506	001001			BNE	1\$;BRANCH IF 22 SECTORS NOT DONE
4885	033510	000402			BR	TST53	;ALL DONE - GO TO NEXT TEST
4886	033512	000137	033044	1\$:	JMP	2#SS1	;GO BACK TO NEXT SECTOR
4887							
4888							
4889							

[Handwritten scribbles]

[Handwritten mark]

.SBTTL DATA TRANSFER TESTS USING ECC

4890
4891
4892
4893
4894
4895
4896
4897
4898
4899
4900
4901
4902
4903
4904
4905
4906
4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919
4920
4921
4922
4923
4924
4925
4926
4927
4928
4929
4930
4931
4932
4933
4934
4935
4936
4937
4938
4939
4940
4941
4942
4943

033516 000004
033520 012706 001000
033524 012737 000053 017330
033532 012700 054160
033536 012701 000402
033542 012720 177777
033546 005301
033550 001374
033552 004737 045674

033556 012737 177777 015144
033564 005037 050402
033570 013737 050376 050400
033576 013737 050404 050412
033604 005037 050370
033610 005037 050372
033614 005037 050406
033620 005037 050410

033624 012737 010000 055476
033632 012737 000001 055500
033640 005037 055502
033644 005037 055504
033650 012737 000400 055536
033656 004537 047102
033662 055476
033664 055506

033666 012777 177374 161056
033674 012700 015220

```
*****  
;*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.  
*****  
TST53: SCOPE  
MOV #STACK, SP ;RESET STACK  
  
MOV #TTNO, @#TSTNM ;THIS SAVES TEST NUMBER  
MOV #SECGAP, R0 ;POINTER  
MOV #256, R1 ;COUNTER  
1$: MOV #-1, (R0)+ ;FILL SIMULATER DISK WITH ONES  
DEC R1  
BNE 1$  
JSR PC, @#CLDISK ;THIS IS USED TO SET GENERAL REGISTERS  
  
;*THESE ARE FOR ECC TEST ONLY  
  
MOV #-1, @#TSECC ;THIS IS AN ECC TEST  
CLR @#POSITI ;CLEAR ERROR POSITION COUNTER  
MOV @#NCODE, @#NCOUNT ;TEMPORARY N-CODE COUNTER  
MOV @#HARDER, @#HADTMP ;TEMPORARY HARD ERROR COUNTER  
CLR @#GECCI ;ECC LOW ORDER TO BE GENERATED  
CLR @#GECC2 ;ECC HIGH ORDER TO BE GENERATED  
CLR @#DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT  
CLR @#ZCODE ;CLEAR LEADING ZEROS CLOCK COUNT  
  
;*THESE ARE TO BE SETUP FOR DISKLESS USE ONLY  
  
MOV #FMT22, @#WCYL ;FORMAT22=16BIT WORDS AND  
;CYLINDER 0  
MOV #1, @#WSECTR ;TRACK=0, SECTOR=1  
CLR @#WKEY1 ;KEY1=0  
CLR @#WKEY2 ;KEY2=0  
MOV #256, @#FNWORD ;256 DATA WORDS  
JSR RS, @#CRC ;GO TO CALCULATE CRC  
WCYL  
GCRC  
  
;*THESE ARE REGULAR SETUPS  
  
MOV #-260, @#RHWC ;256 DATA WORDS 4 HEADER WORDS  
MOV #WRFROM, R0 ;THESE TWO INSTRUCTIONS GETS
```

```

4944 033700 010077 161050      MOV      RO, @RHBA          ; ADDR. OF WRFROM INTO RO AND
4945                                ; BUS ADDRESS REGISTER
4946 033704 012720 010000      MOV      #FMT22, (RO)+     ; FORMAT=16 BIT WORDS
4947                                ; CYLINDER=0
4948 033710 012720 000001      2$: MOV      #1, (RO)+     ; TRACK=0, SECTOR=1, KEYS=0
4949 033714 005020              CLR      (RO)+             ; KEY1=0
4950 033716 005020              CLR      (RO)+             ; KEY2=0
4951 033720 012705 000400      MOV      #256, R5         ; COUNTER
4952 033724 012720 000000      3$: MOV      #0, (RO)+     ; MOVE ALL ZEROS FOR DATA
4953 033730 005305              DEC      R5
4954 033732 001374              BNE      3$
4955 033734 012777 000001 161022  MOV      #1, @RHDST        ; BRANCH IF DATA NOT COMPLETE
4956                                ; TRACK=0 SECTOR=1
4957 033742 004737 045730      JSR      PC, @#CHECKT     ; CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
4958 033746 104400 005124      TYPE    ,CPHALT          ; CANNOT CONTINUE TESTING IF ANY OF THE
4959                                ; ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
4960 033752 000000              HALT
4961                                ; STOP THE TEST
4962 033754 013711 015174      MOV      @#WRIFOR, @R1    ; GET READY FOR WRITE HEADER AND
4963                                ; DATA WITH 62 IN RHCSI
4964 033760 005037 015126      CLR      @#ERFLG$         ; CLEAR ERROR FLAG
4965 033764 012777 010000 160776  MOV      #FMT22, @RHOF    ; FORMAT BIT=1 (16 BIT WORDS)
4966 033772 005077 160774      CLR      @RHCA            ; CYLINDER =0
4967 033776 004737 055322      JSP      PC, @#COMWHD     ; WRITE HEADER AND DATA
4968
4969                                ; *IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
4970                                ; *FROM THE "COMWHD" ROUTINE THAT MEANS ALL HEADER ON DISK
4971                                ; *IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
4972                                ; *ALL ZEROS AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
4973                                ; *THEY ARE ALL ZEROS
4974
4975 034002 005737 015126      TST      @#ERFLG$         ; HAS ANY ERRORS OCCURED?
4976                                ; IF WRITE ERROR OCCURS ECC IS NOT CHECKED
4977 034006 001056              BNE      TST54            ; ; BRANCH IF YES
4978
4979                                ; *COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
4980
4981
4982 034010 023737 050370 055256  CMP      @#GECC1, @#WECC1 ; COMPARE SOFTWARE ECC WITH HARDWARE ECC
4983 034016 001402              BEQ      6$
4984 034020 104031              ERROR   31                ; BRANCH IF GOOD
4985 034022 000405              BR      7$                ; LOW ORDER ECC IN ERROR
4986 034024 023737 050372 055260 6$: CMP      @#GECC2, @#WECC2 ; COMPARE SOFTWARE ECC WITH HARDWARE ECC
4987 034032 001401              BEQ      7$
4988 034034 104031              ERROR   31                ; BRANCH IF GOOD
4989                                ; HIGH ORDER ECC IN ERROR
4990
4991                                7$:
4992 034036 004737 046120      JSR      PC, @#CHECKE     ; CHECK THAT DVA, RDY, DPR, DRY = 1
4993 034042 104400 005124      TYPE    ,CPHALT          ; CANNOT CONTINUE IF THEY DON'T
4994 034046 000000              HALT
4995                                ; STOP THE TEST AND RESTART PROGRAM
4996
4997

```



```

4998                                     ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
4999
5000 034050 004037 045612                JSR   RD,@#CLAREA      ;FILL REINTO BUFFER
5001 034054 016264                        REINTO                ;FROM
5002 034056 017262                        REINTO+<255.*2>      ;TO
5003 034060 000000                        .WORD 0               ;DATA
5004
5005 034062 013737 050370 017264        MOV   @#GECC1,@#REINTO+<256.*2>;FILL ECC1
5006 034070 013737 050372 017266        MOV   @#GECC2,@#REINTO+<257.*2>;FILL ECC2
5007 034076 004037 045612                JSR   RD,@#CLAREA      ;FILL REST
5008 034102 017270                        REINTO+<258.*2>      ;FROM
5009 034104 017324                        REINTO+<272.*2>      ;TO
5010 034106 000000                        0                     ;DATA
5011
5012
5013 034110 005037 015126                CLR   @#ERFLG$        ;CLEAR ERROR FLAG
5014
5015
5016                                     ;*NOW COMPARE "DISK" BUFFER WITH "REINTO"
5017
5018 034114 004037 046570                JSR   RD,@#COMPAR     ;CHECK
5019 034120 016264                        REINTO                ;GOOD BUFFER
5020 034122 054256                        DISK                  ;TEST BUFFER
5021 034124 000402                        258.                 ;NUMBER OF WORDS CHECKED
5022 034126 034134                        4$                   ;RETURN POINT FOR ERROR HEADER
5023 034130 034140                        5$                   ;RETURN POINT FOR ERROR DATA
5024
5025 034132 034144                        TST54                ;RETURN FOR GOOD COMPARISON
5026
5027 034134 104007                        4$: ERROR 7          ;READ ERROR 10 NEXT
5028 034136 000207                        RTS PC               ;RETURN TO COMPARE
5029 034140 104010                        5$: ERROR 10        ;WORD NOS 1 TO 256 ARE
5030                                     ;DATA WORDS
5031                                     ;WORD NOS 257 AND 258
5032                                     ;ARE ECC WHICH ARE CHECKED
5033                                     ;WORD NOS 259
5034                                     ;IS DATA GAP
5035                                     ;WORD NOS 260 TO 273
5036                                     ;ARE TOLERANCE GAP
5037 034142 000207                        RTS PC               ;RETURN TO COMPARE
5038
5039
5040
5041
5042

```

```

5043
5044 ;*****
5045 ;*TEST 54 READ ECC ENABLED 1A
5046
5047 ;* THIS IS AN ECC READ DATA TEST
5048 ;* ERROR CORRECTION IS ENABLED
5049 ;* NO ERROR IS INSERTED
5050 ;* GOOD DATA USED IS 256 WORDS OF 0
5051 ;* COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
5052 ;* TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA
5053
5054 ;*****
5055 T54: SCOPE
5056 034144 000004 MOV #STACK, SP ;RESET STACK
5057 034146 012706 001000
5058 034152 012737 000054 017330 MOV #TTNO, @#TSTNM ;THIS SAVES TEST NUMBER
5059
5060 ;
5061 ; SETUP FOR WHAT IS TO BE READ
5062 ; HEADER CRC IS RESTORED FROM A SUBROUTINE
5063
5064 034160 012746 000000 MOV #0, -(SP) ;DATA TO BE READ
5065 034164 012705 000400 MOV #256, R5 ;COUNTER
5066 034170 012700 054256 MOV #DISK, R0 ;START OF SIMULATED DISK DATA
5067 034174 011620 1$: MOV (SP), (R0)+ ;MOVE IN DATA ON TO SIMULATED DISK
5068 034176 005305 DEC R5 ;COUNT
5069 034200 001375 BNE 1$ ;BRANCH IF 256 NOT COMPLETE
5070 034202 005726 TST (SP)+ ;UNDO -(SP)
5071 034204 022020 CMP (R0)+, (R0)+ ;JUMP OVER THE TWO ECC WORDS
5072 034206 012705 000017 MOV #15, R5 ;1 DATA GAP
5073 ; ;14 TOLERANCE GAP
5074 034212 005020 2$: CLR (R0)+ ;CLEAR DATA GAP, AND
5075 034214 005305 DEC R5 ;TOLERANCE GAP
5076 034216 001375 BNE 2$ ;BRANCH IF NOT COMPLETE
5077
5078
5079 034220 004737 051164 JSR PC, @#FILLEC ;INSERT THE TWO ECC WORDS ON THE DISK
5080 ; ;IN THE CORRECT PLACE
5081
5082 ;*THESE ARE FOR ECC TEST ONLY
5083
5084 034224 012737 177777 015144 MOV #-1, @#TSECC ;THIS IS AN ECC TEST
5085 034232 005037 050402 CLR @#POSITI ;CLEAR ERROR POSITION COUNTER
5086 034236 013737 050376 050400 MOV @#NCODE, @#NCOUNT ;TEMPORARY N-CODE COUNTER
5087 034244 013737 050404 050412 MOV @#HARDER, @#HADTMP ;TEMPORARY HARD ERROR COUNTER
5088 034252 005037 050370 CLR @#GECC1 ;ECC LOW ORDER TO BE GENERATED
5089 034256 005037 050372 CLR @#GECC2 ;ECC HIGH ORDER TO BE GENERATED
5090 034262 005037 050406 CLR @#DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT
5091 034266 005037 050410 CLR @#ZCODE ;CLEAR LEADING ZEROS CLOCK COUNT
5092
5093
5094 ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
5095
5096 034272 012737 010000 052340 MOV #FMT22, @#CYL ;16 BITS PER WORD

```

```

S097      034300  112737  000000  052343      MOVB  #0,2#SECOTR+1      ;CYLINDER 0. FORMAT 16 BITS
S098      034306  112737  000000  052342      MOVB  #0,2#SECOTR      ;TRACK 0
S099      034314  012737  000000  052344      MOV   #0,2#KEY1        ;SECTOR 0
S100      034322  012737  000000  052346      MOV   #0,2#KEY2        ;KEY1=0
S101      034330  012737  000400  052420      MOV   #256., 2#DAWORD  ;KEY2=0
S102      034336  005037  052350      CLR   2#X              ;NO. OF DATA WORDS
S103      034342  004537  047102      JSR   R5,2#CRC         ;THIS IS A READ COMMAND
S104      034346  052340      CYL   ;GO TO CALCULATE CRC
S105      034350  054240      WCR   ;
S106
S107
S108
S109
S110
S111
S112      034352  004737  045674      JSR   PC,2#CLDISK     ;*THESE ARE REGULAR SETUPS
S113      034356  012777  177374  160366      MOV   #-256.-4,2#RHWC ;SETUP GENERAL REGISTERS
S114      034364  012777  016264  160362      MOV   #REINTO,2#RHBA  ;256. DATA 4 HEADER WORDS
S115      034372  112746  000000      MOVB  #0,-(SP)         ;STARTING ADDRESS OF READ BUFFER
S116      034376  112766  000000  000001      MOVB  #0,1(SP)        ;IN LOWER BYTE GET SECTOR
S117      034404  012677  160354      MOV   (SP)+,2#RH0ST   ;GET TRACK IN HIGHER BYTE
S118      034410  012777  010000  150352      MOV   #FMT22,2#RHOF  ;TRACK/SECTOR IN RH0ST
S119
S120
S121
S122
S123      034416  005077  160350      CLR   2#RHCA         ;16 BITS PER WORD
S124
S125      034422  004737  045730      JSR   PC,2#CHECKT     ;ECC CORRECTION NOT INHIBIT
S126      034426  104400  005124      TYPE  .CPHALT        ;BECAUSE ECC IS NOT GOING
S127
S128      034432  000000      HALT                ;TO BE CHECKED
S129
S130      034434  013711  015200      MOV   2#REFOR,2#RI    ;CYLINDER 0
S131      034440  005037  015126      CLR   2#ERFLG$       ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
S132      034444  004737  052200      JSR   PC,2#COMHD     ;CANNOT CONTINUE TESTING IF ANY OF THE
S133
S134
S135
S136
S137
S138
S139
S140
S141
S142
S143
S144
S145
S146
S147      034450  005737  015126      TST   2#ERFLG$       ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK & !
S148      034454  001102      BNE                    ;STOP THE TEST
S149      034456  004737  045374      JSR   PC,2#PUTREG    ;READ HEADER AND DATA=72
S150      034462  005737  015036      TST   2#ERI          ;CLEAR ERROR FLAG
;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 SYNC'S 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

S151      034450  005737  015126      TST   2#ERFLG$       ;ANY ERRORS ALREADY THERE
S152      034454  001102      BNE                    ;BRANCH IF YES
S153      034456  004737  045374      JSR   PC,2#PUTREG    ;SAVE REGISTERS
S154      034462  005737  015036      TST   2#ERI          ;NO ERRORS SHOULD BE SET

```

```

5151 034466 001401          BEQ      6$          ;BRANCH IF NO ERRORS SET
5152 034470 104032          ERROR   32          ;32 BIT ECC REGISTER SHOULD BE ZERO
5153                               ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5154                               ;IN THE PATERN REGISTER
5155                               ;DCK SHOULD BE SET IN RHER;
5156 034472 013746 050370      6$:   MOV      2#GECC1,-(SP) ;GET PATTERN REGISTER
5157 034476 042716 174000      SIC      #174000,(SP) ;KEEP ONLY 11 BITS
5158 034502 022637 015066      CMP      (SP)+,2#EC2 ;COMPARE PATTERN REGISTER
5159 034506 001401          BEQ      7$          ;BRANCH IF GOOD
5160 034510 104032          ERROR   32          ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5161
5162
5163
5164
5165                               ;*ADD 16 MAINTENANCE CLOCKS TO
5166                               ;*BRING EBL DOWN
5167
5168 034512 012700 000020      7$:   MOV      #16,R0          ;COUNTER
5169 034516 052777 000002 160254 9$:   BIS      #MCLK,2#RMR    ;SET CLOCK
5170 034524 042777 000002 160246      BIC      #MCLK,2#RMR    ;CLEAR CLOCK
5171 034532 005300          DEC      R0            ;COUNT
5172 034534 001370          BNE      8$          ;BRANCH IF 16 CLOCKS NOT DONE
5173 034536 004737 046120      JSR      PC,2#CHECKE    ;CHECK THAT DVA,RDY,DPR,DY = 1
5174 034542 104400 005124      TYPE    ,CPHALT        ;CANNOT CONTINUE IF THEY DON'T
5175 034546 000000          HALT                    ;STOP THE TEST AND RESTART PROGRAM
5176 034550 012700 015220      MOV      #WFRM,R0      ;GETTING READY TO FILL EXPECTED DATA
5177 034554 012720 010000      MOV      #0!FMT22,(R0)+ ;CYLINDER 0
5178 034560 112746 000000      MOV      #0,-(SP)      ;IN LOWER BYTE GET SECTOR
5179 034564 112766 000000 000001      MOV      #0,1(SP)      ;GET TRACK IN HIGHER BYTE
5180 034572 012620          MOV      (SP)+,(R0)+    ;GET TRACK/SECTOR IN BUFFER
5181 034574 012720 000000      MOV      #0,(R0)+      ;KEY1 IN BUFFER
5182 034600 012720 000000      MOV      #0,(R0)+      ;KEY2 IN BUFFER
5183 034604 012701 000400      MOV      #256,R1       ;DATA WORD COUNTER
5184 034610 012702 000000      MOV      #0,R2         ;DATA
5185 034614 010220          3$:   MOV      R2,(R0)+      ;DATA INTO BUFFER
5186 034616 005301          DEC      R1            ;COUNT
5187 034620 001375          BNE      3$          ;BRANCH IF 256 NOT DONE
5188
5189
5190 034622 005037 015126      CLR      2#ERFLG$      ;CLEAR ERROR FLAG
5191 034626 004737 045374      JSR      PC,2#PUTREG    ;SAVE REGISTERS
5192
5193
5194
5195                               ;*NOW READ DATA BUFFER WILL BE CHECKED
5196
5197 034632 004037 046570      JSR      R0,2#COMPAR    ;CHECK
5198 034636 015220          WFRM                    ;GOOD BUFFER
5199 034640 016264          REINTO                   ;TEST BUFFER
5200 034642 000404          4+256.                  ;NUMBER OF WORDS CHECKED
5201 034644 034652          4$                       ;RETURN POINT FOR ERROR HEADER
5202 034646 034656          5$                       ;RETURN POINT FOR ERROR DATA
5203
5204 034650 034662          TST5                    ;RETURN FOR GOOD COMPARISON
    
```

S205
S206
S207
S208
S209
S210
S211
S212
S213
S214
S215

034652 104004
034654 000207
034656 104005

034660 000207

48: ERROR 4
RTS PC
58: ERROR 5

RTS PC

;READ NEXT ERROR
;RETURN TO "COMPAR"
;WORD NOS 1 TO 4 ARE
;HEADER WORDS
;5 TO 260 ARE DATA WORDS
;RETURN TO "COMPAR"

E14

5216
5217
5218
5219
5220
5221
5222
5223
5224
5225
5226
5227
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241
5242
5243
5244
5245
5246
5247
5248
5249
5250
5251
5252
5253
5254
5255
5256
5257
5258
5259
5260
5261
5262
5263
5264
5265
5266
5267
5268
5269

: *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

†ST55: SCOPE
MOV #STACK, SP ; RESET STACK

MOV #TTNO, @TSTNM ; THIS SAVES TEST NUMBER

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

MOV #0, -(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
2\$: CLR (R0)+ ; CLEAR DATA GAP, AND
DEC R5 ; TOLERANCE GAP
BNE 2\$; BRANCH IF NOT COMPLETE

JSR PC, @FILLED ; INSERT ECC IN PROPER PLACE ON DISK

; *THESE ARE FOR ECC TEST ONLY

MOV #-1, @TSECC ; THIS IS AN ECC TEST
CLR @POSITI ; CLEAR ERROR POSITION COUNTER
MOV @NCODE, @NCOUNT ; TEMPORARY N-CODE COUNTER
MOV @HARDER, @HADTMP ; TEMPORARY HARD ERROR COUNTER
CLR @GECC1 ; ECC LOW ORDER TO BE GENERATED
CLR @GECC2 ; ECC HIGH ORDER TO BE GENERATED
CLR @DATENV ; CLEAR DATA ENVELOPE CLOCK COUNT
CLR @ZCODE ; CLEAR LEADING ZEROS CLOCK COUNT

; *THESE ARE TO SETUP FOR DISKLESS USE ONLY

```

5270 035010 012737 010000 052340      MOV      #FMT22, @#CYL      ;16 BITS PER WORD
5271                                     ;CYLINDER 0, FORMAT 16 BITS
5272 035016 112737 000000 052343      MOVVB   #0, @#SECOTR+1    ;TRACK 0
5273 035024 112737 000000 052342      MOVVB   #0, @#SECOTR     ;SECTOR 0
5274 035032 012737 000000 052344      MOV      #0, @#KEY1      ;KEY1=0
5275 035040 012737 000000 052346      MOV      #0, @#KEY2      ;KEY2=0
5276 035046 012737 000400 052420      MOV      #256., @#DAWORD ;NO. OF DATA WORDS
5277 035054 005037 052350                CLR      @#X              ;THIS IS A READ COMMAND
5278 035060 004537 047102                JSR      R5, @#CRC        ;GO TO CALCULATE CRC
5279 035064 052340                CYL
5280 035066 054240                WCRC

5281
5282
5283                                     ;*THIS IS TO INSERT ERROR
5284                                     ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
5285                                     ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
5286                                     ;*THIS MOVE
5287
5288 035070 012737 100000 054260      MOV      #100000, @#DISK+2 ;FORCE ERROR ON BIT NUMBER 32
5289                                     ;SO ERROR POSITION REGISTER WILL SHOW
5290                                     ;22
5291 035076 012737 000026 035252      MOV      #22., @#85      ;INSERT POSITION REG.
5292
5293
5294                                     ;*THESE ARE REGULAR SETUPS
5295
5296 035104 004737 045674                JSR      PC, @#CLDISK    ;SETUP GENERAL REGISTERS
5297 035110 012777 177374 157634      MOV      #-256.-4, @RHWC ;256. DATA 4 HEADER WORDS
5298 035116 012777 016264 157630      MOV      #REINTO, @RHBA  ;STARTING ADDRESS OF READ BUFFER
5299 035124 112746 000000                MOVVB   #0, -(SP)        ;IN LOWER BYTE GET SECTOR
5300 035130 112766 000000 000001      MOVVB   #0, 1(SP)        ;GET TRACK IN HIGHER BYTE
5301 035136 012677 157622                MOV      (SP)+, @RH DST  ;TRACK/SECTOR IN RH DST
5302 035142 012777 010000 157620      MOV      #FMT22, @RHOF   ;16 BITS PER WORD
5303                                     ;ECC CORRECTION NOT INHIBIT
5304                                     ;BECAUSE ECC IS NOT GOING
5305                                     ;TO BE CHECKED
5306 035150 005077 157616                CLR      @RHCA          ;CYLINDER 0
5307
5308 035154 004737 045730                JSR      PC, @#CHECKT    ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
5309 035160 104400 005124                TYPE    ,CPHALT         ;CANNOT CONTINUE TESTING IF ANY OF THE
5310                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5311 035164 000000                HALT                    ;STOP THE TEST
5312
5313 035166 013711 015200                MOV      @#REFOR, @RI    ;READ HEADER AND DATA=72
5314 035172 005037 015126                CLR      @#ERFLG$       ;CLEAR ERROR FLAG
5315 035176 004737 052200                JSR      PC, @#COMHD     ;READ HEADER AND DATA
5316                                     ;IF THERE ARE READ ERRORS THEN
5317                                     ;ECC WILL NOT BE CHECKED
5318
5319
5320
5321                                     ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5322                                     ;*FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP.
5323                                     ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5324                                     ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY

```

```

5324      ;*DETECTED
5325      ;*HEADER AND DATA ARE TO BE CHECKED.
5326      ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
5327      ;*"WRFROM" IS FILLED WITH EXPECTED DATA AND
5328      ;*COMPARISONS ARE MADE
5329
5330 035202 005737 015126      TST      @#ERFLGS      ;ANY ERRORS ALREADY THERE
5331 035206 001077      BNE      TST56      ;BRANCH IF YES
5332 035210 004737 045374      JSR      PC,@#PUTREG ;SAVE REGISTERS
5333 035214 022737 100000 015036  CMP      @#DCK,@#ERI ;ONLY DATA CHECK ERROR SHOULD BE SET
5334 035222 001401      BEQ      6$         ;BRANCH IF YES
5335 035224 104032      ERROR    32         ;32 BIT ECC REGISTER SHOULD BE NON
5336      ;ZERO
5337      ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5338      ;IN THE PATTERN REGISTER
5339      ;DCK SHOULD BE SET IN RHER1
5340 035226 013746 050370      6$:     MOV      @#GECC1,-(SP) ;GET PATTERN REGISTER
5341 035232 042716 174000      BIC      @#174000,(SP) ;KEEP ONLY 11 BITS
5342 035236 022637 015066      CMP      (SP)+,@#EC2 ;COMPARE PATTERN REGISTER
5343 035242 001401      BEQ      7$         ;BRANCH IF GOOD
5344 035244 104032      ERROR    32         ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5345
5346 035246 004037 051012      7$:     JSR      RC,@#ECORR ;GO TO ECC CORRECTION PROCESS
5347 035252 000026      8$:     22.          ;EXPECTED POSITION REG. WHEN CORRECTION
5348      ;IS COMPLETE
5349
5350
5351
5352 035254 004737 046120      JSR      PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
5353 035260 104400 005124      TYPE    ,CPHALT    ;CANNOT CONTINUE IF THEY DON'T
5354 035264 000000      HALT                    ;STOP THE TEST AND RESTART PROGRAM
5355 035266 012700 015220      MOV      @#WRFROM,RO ;GETTING READY TO FILL EXPECTED DATA
5356 035272 012720 010000      MOV      @#0!FMT22,(RO)+ ;CYLINDER 0
5357 035276 112746 000000      MOV      @#0,-(SP)    ;IN LOWER BYTE GET SECTOR
5358 035302 112766 000000 000001  MOV      @#0,1(SP)    ;GET TRACK IN HIGHER BYTE
5359 035310 012620      MOV      (SP)+,(RO)+ ;GET TRACK/SECTOR IN BUFFER
5360 035312 012720 000000      MOV      @#0,(RO)+   ;KEY1 IN BUFFER
5361 035316 012720 000000      MOV      @#0,(RO)+   ;KEY2 IN BUFFER
5362 035322 012701 000400      MOV      @#256.,R1   ;DATA WORD COUNTER
5363 035326 012702 000000      MOV      @#0,R2     ;DATA
5364 035332 010220      3$:     MOV      R2,(RO)+   ;DATA INTO BUFFER
5365 035334 005301      DEC      R1         ;COUNT
5366 035336 001375      BNE      3$         ;BRANCH IF 256 NOT DONE
5367
5368      ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
5369      ;*NOW THE INSERTED ERROR WILL BE PUT IN
5370 035340 012737 100000 015232  MOV      @#100000,@#WRFROM+<5*2> ;INSERTED ERROR
5371
5372
5373
5374 035346 005037 015126      CLR      @#ERFLGS    ;CLEAR ERROR FLAG
5375 035352 004737 045374      JSR      PC,@#PUTREG ;SAVE REGISTERS
5376
5377

```

211
282


```

5378 ;*NOW READ DATA BUFFER WILL BE CHECKED
5379
5380 035356 004037 046570 JSR RO,@#COMPAR ;CHECK
5381 035362 015220 WRFROM ;GOOD BUFFER
5382 035364 016264 REINTO ;TEST BUFFER
5383 035366 000404 4+256. ;NUMBER OF WORDS CHECKED
5384 035370 035376 4$ ;RETURN POINT FOR ERROR HEADER
5385 035372 035402 5$ ;RETURN POINT FOR ERROR DATA
5386
5387 035374 035406 TST56 ;RETURN FOR GOOD COMPARISON
5388
5389 035376 104004 4$: ERROR 4 ;READ NEXT ERROR
5390 035400 000207 RTS PC ;RETURN TO "COMPAR"
5391 035402 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE
5392 ;HEADER WORDS
5393 ;5 TO 260 ARE DATA WORDS
5394 035404 000207 RTS PC ;RETURN TO "COMPAR"
5395

```

```

5396
5397
5398
5399
5400
5401
5402
5403
5404
5405
5406
5407
5408 035406 000004
5409 035410 012706 001000
5410
5411 035414 012737 000056 017330
5412
5413
5414
5415
5416
5417 035422 012746 000000
5418 035426 012705 000400
5419 035432 012700 054256
5420 035436 011620
5421 035440 005305
5422 035442 001375
5423 035444 005726
5424 035446 022020
5425 035450 012705 000017
5426
5427 035454 005020
5428 035456 005305
5429 035460 001375
5430
5431
5432 035462 004737 051164
5433
5434
5435
5436
5437 035466 012737 177777 015144
5438 035474 005037 050402
5439 035500 013737 050376 050400
5440 035506 013737 050404 050412
5441 035514 005037 050370
5442 035520 005037 050372
5443 035524 005037 050406
5444 035530 005037 050410
5445
5446
5447
5448
5449 035534 012737 010000 052340

```

```

*****
*TEST 56      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

*****
†ST56:  SCOPE
        MOV      #STACK,SP      ;RESET STACK
        MOV      #TTNO,‡#TSTNM  ;THIS SAVES TEST NUMBER

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

        MOV      #0,-(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
2$:    CLR     (R0)+          ;CLEAR DATA GAP, AND
        DEC     R5            ;TOLERANCE GAP
        BNE     2$            ;BRANCH IF NOT COMPLETE

        JSR     PC,‡#FILLEC    ;INSERT THE TWO ECC WORDS ON THE DISK
                                ;IN THE CORRECT PLACE

;*THESE ARE FOR ECC TEST ONLY

        MOV     #-1,‡#TSECC    ;THIS IS AN ECC TEST
        CLR     ‡#POSITI      ;CLEAR ERROR POSITION COUNTER
        MOV     ‡#NCODE,‡#NCOUNT ;TEMPORARY N-CODE COUNTER
        MOV     ‡#HARDER,‡#HADTMP ;TEMPORARY HARD ERROR COUNTER
        CLR     ‡#GECC1       ;ECC LOW ORDER TO BE GENERATED
        CLR     ‡#GECC2       ;ECC HIGH ORDER TO BE GENERATED
        CLR     ‡#DATENV      ;CLEAR DATA ENVELOPE CLOCK COUNT
        CLR     ‡#ZCODE       ;CLEAR LEADING ZEROS CLOCK COUNT

;*THESE ARE TO SETUP FOR DISKLESS USE ONLY

        MOV     #FMT22,‡#CYL   ;16 BITS PER WORD

```

```

5450                                     ;CYLINDER 0, FORMAT 16 BITS
5451 035542 112737 000000 052343      MOVB  #0, @#SECOTR+1 ;TRACK 0
5452 035550 112737 000000 052342      MOVB  #0, @#SECOTR   ;SECTOR 0
5453 035556 012737 000000 052344      MOV   #0, @#KEY1    ;KEY1=0
5454 035564 012737 000000 052346      MOV   #0, @#KEY2    ;KEY2=0
5455 035572 012737 000400 052420      MOV   #256., @#DAWORD ;NO. OF DATA WORDS
5456 035600 005037 052350              CLR   @#X            ;THIS IS A READ COMMAND
5457 035604 004537 047102              JSR   R5, @#CRC     ;GO TO CALCULATE CRC
5458 035610 052340
5459 035612 054240
5460
5461
5462                                     ;*THIS IS TO INSERT ERROR
5463                                     ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
5464                                     ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
5465                                     ;*THIS MOVE
5466 035614 012737 177760 054260      MOV   #177760, @#DISK+2 ;FORCE ERROR ON BIT NUMBER 21 THRU 32
5467                                     ;S0 ERROR POSITION REGISTER WILL SHOW
5468                                     ;22
5469 035622 012737 010040 035776      MOV   #4128., @#8$    ;INSERT POSITION REG.
5470
5471
5472                                     ;*THESE ARE REGULAR SETUPS
5473
5474 035630 004737 045674              JSR   PC, @#CLDISK   ;SETUP GENERAL REGISTERS
5475 035634 012777 177374 157110      MOV   #-256.-4., @#RHWC ;256. DATA 4 HEADER WORDS
5476 035642 012777 016264 157104      MOV   @#REINTO, @#RHBA ;STARTING ADDRESS OF READ BUFFER
5477 035650 112746 000000              MOVB  #0, -(SP)      ;IN LOWER BYTE GET SECTOR
5478 035654 112766 000000 000001      MOVB  #0, 1(SP)     ;GET TRACK IN HIGHER BYTE
5479 035662 012677 157076              MOV   (SP)+, @#RHDST ;TRACK/SECTOR IN RHDST
5480 035666 012777 010000 157074      MOV   @#FMT22, @#RHOF ;16 BITS PER WORD
5481                                     ;ECC CORRECTION NOT INHIBIT
5482                                     ;BECAUSE ECC IS NOT GOING
5483                                     ;TO BE CHECKED
5484 035674 005077 157072              CLR   @#RHCA        ;CYLINDER 0
5485 035700 004737 045730              JSR   PC, @#CHECKT   ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
5486 035704 104400 005124              TYPE  , @#PHALT     ;CANNOT CONTINUE TESTING IF ANY OF THE
5487                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5488 035710 000000              HALT                ;STOP THE TEST
5489 035712 013711 015200              MOV   @#REFOR, @#R1 ;READ HEADER AND DATA=72
5490 035716 005037 015126              CLR   @#ERFLG$     ;CLEAR ERROR FLAG
5491 035722 004737 052200              JSR   PC, @#COMHD   ;READ HEADER AND DATA
5492                                     ;IF THERE ARE READ ERRORS THEN
5493                                     ;ECC WILL NOT BE CHECKED
5494

```



```

5495
5496
5497
5498
5499
5500
5501
5502
5503
5504
5505
5506 035726 005737 015126 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
5507 035732 001106 BNE TST57 ;BRANCH IF YES
5508 035734 004737 045374 JSR PC,@#PUTREG ;SAVE REGISTERS
5509 035740 022737 100000 015036 CMP #DCK,@#ER1 ;ONLY DATA CHECK ERROR SHOULD BE SET
5510 035746 001401 BEQ 6$ ;BRANCH IF YES
5511 035750 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE NON
5512 ;ZERO
5513 ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5514 ;IN THE PATERN REGISTER
5515 ;DCK SHOULD BE SET IN RHER1
5516 035752 013746 050370 6$: MOV @#GECC1,-(SP) ;GET PATTERN REGISTER
5517 035756 042716 174000 BIC #174000,(SP) ;KEEP ONLY 11 BITS
5518 035762 022637 015066 CMP (SP)+,@#EC2 ;COMPARE PATTERN REGISTER-
5519 035766 001401 BEQ 7$ ;BRANCH IF GOOD
5520 035770 104032 ERROR 32 ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5521
5522 035772 004037 051012 7$: JSR R0,@#ECORR ;GO TO ECC CORRECTION PROCESS
5523 035776 000000 8$: .WORD ;EXPECTED POSITION REG. WHEN CORRECTION
5524 ;IS COMPLETE
5525
5526
5527
5528 036000 004737 045374 JSR PC,@#PUTREG ;SAVE REGISTERS
5529 036004 022737 100100 015036 CMP #DCK!ECH,@#ER1 ;WITH ERRORS INSERTED IN BIT POSITION 21
5530 ;THRU 32 HARD ERROR BIT SHOULD SET
5531 036012 001401 BEQ 9$ ;BRANCH IF GOOD
5532 036014 104036 ERROR 36 ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
5533 ;32 ECH SHOULD SET
5534
5535
5536
5537 036016 9$: JSR PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
5538 036016 004737 046120 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
5539 036022 104400 005124 HALT ;STOP THE TEST AND RESTART PROGRAM
5540 036026 000000 MOV #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
5541 036030 012700 015220 MOV #0!FMT22,(R0)+ ;CYLINDER 0
5542 036034 012720 010000 MOVB #0,-(SP) ;IN LOWER BYTE GET SECTOR
5543 036040 112746 000000 MOVB #0,!(SP) ;GET TRACK IN HIGHER BYTE
5544 036044 112766 000000 000001 MOV (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
5545 036052 012620 000000 MOV #0,(R0)+ ;KEY1 IN BUFFER
5546 036054 012720 000000 MOV #0,(R0)+ ;KEY2 IN BUFFER
5547 036060 012720 000000 MOV #256.,R1 ;DATA WORD COUNTER
5548 036064 012701 000400
    
```

L14

MNDEC-11-DZRJH-A, RPO4/5/6 DSKLS CONTROLLER TST-PT 2
DZRJHA.TST T56 READ ECC ENABLED 1C

MACY11 27(655) 30-MAR-76 20:59 PAGE 133

SEQ 0179

5549	036070	012702	000000		MOV	#0, R2	; DATA
5550	036074	010220		3\$:	MOV	R2, (R0)+	; DATA INTO BUFFER
5551	036076	005301			DEC	R1	; COUNT
5552	036100	0C1375			BNE	3\$; BRANCH IF 256 NOT DONE
5553							

```

5554 ; ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
5555 ; *NOW THE INSERTED ERROR WILL BE PUT IN
5556
5557 036102 012737 177760 015232 MOV #177760, @WRFROM+<5*2> ; INSERTED ERROR
5558
5559
5560
5561 036110 005037 015126 CLR @ERFLG$ ; CLEAR ERROR FLAG
5562 036114 004737 045374 JSR PC, @PUTREG ; SAVE REGISTERS
5563
5564
5565 ; *NOW READ DATA BUFFER WILL BE CHECKED
5566
5567 036120 004037 046570 JSR RD, @COMPAR ; CHECK
5568 036124 015220 WRFROM ; GOOD BUFFER
5569 036126 016264 REINTO ; TEST BUFFER
5570 036130 000404 4+256. ; NUMBER OF WORDS CHECKED
5571 036132 036140 4$ ; RETURN POINT FOR ERROR HEADER
5572 036134 036144 5$ ; RETURN POINT FOR ERROR DATA
5573
5574 036136 036150 TST57 ; RETURN FOR GOOD COMPARISON
5575
5576 036140 104004 4$: ERROR 4 ; READ NEXT ERROR
5577 036142 000207 RTS PC ; RETURN TO "COMPAR"
5578 036144 104005 5$: ERROR 5 ; WORD NOS 1 TO 4 ARE
5579 ; HEADER WORDS
5580 ; 5 TO 260 ARE DATA WORDS
5581 036146 000207 RTS PC ; RETURN TO "COMPAR"
5582
5583
5584
5585
5586
5587

```

```

5588
5589
5590      ;:*****
5591      ;*TEST 57      WRITE ECC TEST 2
5592
5593      ;*      THIS IS A WRITE ECC TEST
5594      ;*      WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
5595      ;*      TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
5596      ;*      OF ALL ONES.
5597      ;:*****
5598      036150 000004
5599      036152 012706 001000      †TST57: SCOPE
5600
5601      036156 012737 000057 017330      MOV      #STACK, SP      ;RESET STACK
5602      036164 012700 054160      MOV      #TTNO, @#TSTNM      ;THIS SAVES TEST NUMBER
5603      036170 012701 000460      MOV      #SECGAP, R0      ;POINTER
5604      036174 005020      MOV      #304., R1      ;COUNTER
5605      036176 005301      1$: CLR      (R0)+      ;CLEAR SIMULATED DISK AREA
5606      036200 001375      DEC      R1
5607      036202 004737 045674      BNE     1$
5608      JSR     PC, CLDISK      ;THIS IS USED TO SET GENERAL REGISTERS
5609
5610      ;*THESE ARE FOR ECC TEST ONLY
5611      036206 012737 177777 015144      MOV      #-1, @#TSECC      ;THIS IS AN ECC TEST
5612      036214 005037 050402      CLR      @#POSITI      ;CLEAR ERROR POSITION COUNTER
5613      036220 013737 050376 050400      MOV      @#NCODE, @#NCOUNT      ;TEMPORARY N-CODE COUNTER
5614      036226 013737 050404 050412      MOV      @#HARDER, @#HADTMP      ;TEMPORARY HARD ERROR COUNTER
5615      036234 005037 050370      CLR      @#GECC1      ;ECC LOW ORDER TO BE GENERATED
5616      036240 005037 050372      CLR      @#GECC2      ;ECC HIGH ORDER TO BE GENERATED
5617      036244 005037 050406      CLR      @#DATENV      ;CLEAR DATA ENVELOPE CLOCK COUNT
5618      036250 005037 050410      CLR      @#ZCODE      ;CLEAR LEADING ZEROS CLOCK COUNT
5619
5620
5621
5622
5623      ;*THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
5624
5625      036254 012737 010000 055476      MOV      #FMT22, @#WCYL      ;FORMAT22=16BIT WORDS AND
5626
5627      036262 012737 000001 055500      MOV      #1, @#WSECTR      ;CYLINDER 0
5628      036270 005037 055502      CLR      @#WKEY1      ;TRACK=0, SECTOR=1
5629      036274 005037 055504      CLR      @#WKEY2      ;KEY1=0
5630      036300 012737 000400 055536      MOV      #256., @#FNWORD      ;KEY2=0
5631      036306 004537 047102      JSR     R5, @#CRC      ;256 DATA WORDS
5632      036312 055476      WCYL      ;GO TO CALCULATE CRC
5633      036314 055506      GCRC
5634
5635      ;*THESE ARE REGULAR SETUPS
5636
5637      036316 012777 177374 156426      MOV      #-260., @#RHWC      ;256 DATA WORDS 4 HEADER WORDS
5638      036324 012700 015220      MOV      #WRFROM, R0      ;THESE TWO INSTRUCTIONS GETS
5639      036330 010077 156420      MOV      R0, @#RHA      ;ADDR. OF WRFROM INTO R0 AND
5640
5641      036334 012720 010000      MOV      #FMT22, (R0)+      ;BUS ADDRESS REGISTER
5642
5643
5644
5645
5646
5647
5648
5649
5650
5651
5652
5653
5654
5655
5656
5657
5658
5659
5660
5661
5662
5663
5664
5665
5666
5667
5668
5669
5670
5671
5672
5673
5674
5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697
5698
5699
5700
5701
5702
5703
5704
5705
5706
5707
5708
5709
5710
5711
5712
5713
5714
5715
5716
5717
5718
5719
5720
5721
5722
5723
5724
5725
5726
5727
5728
5729
5730
5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773
5774
5775
5776
5777
5778
5779
5780
5781
5782
5783
5784
5785
5786
5787
5788
5789
5790
5791
5792
5793
5794
5795
5796
5797
5798
5799
5800

```

```

5643 036340 012720 000001 25: MOV #1,(R0)+ ;CYLINDER=0
5644 036344 005020 CLR (R0)+ ;TRACK=0, SECTOR=1, KEYS=0
5645 036346 005020 CLR (R0)+ ;KEY1=0
5646 036350 012705 000400 MOV #256,R5 ;KEY2=0
5647 036354 012720 177777 35: MOV #-1,(R0)+ ;COUNTER
5648 036360 005305 DEC R5 ;MOVE ALL ONES FOR DATA
5649 036362 001374 BNE 35 ;BRANCH IF DATA NOT COMPLETE
5650 036364 012777 000001 156372 MOV #1,BRHST ;TRACK=0 SECTOR=1
5651
5652 036372 004737 045730 JSR PC,B#CHECKT ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
5653 036376 104400 005124 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
5654 ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5655 036402 000000 HALT ;STOP THE TEST
5656
5657 036404 013711 015174 MOV B#WRIFOR,BR1 ;GET READY FOR WRITE HEADER AND
5658 ;DATA WITH 62 IN RHCS1
5659 036410 005037 015126 CLR B#ERFLG$ ;CLEAR ERROR FLAG
5660 036414 012777 010000 156346 MOV #FMT22,BRHOF ;FORMAT BIT=1 (16 BIT WORDS)
5661 036422 005077 156344 CLR BRMCA ;CYLINDER =0
5662 036426 004737 055322 JSR PC,B#COMWHD ;WRITE HEADER AND DATA
5663
5664 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5665 ;*FROM THE "COMWHD" ROUTINE THAT MEANS ALL HEADER ON DISK
5666 ;*IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
5667 ;*ALL ONES AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
5668 ;*THEY ARE ALL ZEROS
5669
5670 036432 005737 015126 TST B#ERFLG$ ;HAS ANY ERRORS OCCURED?
5671 BNE TST60 ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
5672 ;BRANCH IF YES
5673
5674 ;*COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
5675
5676
5677 036440 023737 050370 055256 CMP B#GECC1,B#WECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
5678 BEQ 65 ;BRANCH IF GOOD
5679 036450 104031 ERROR 31 ;LOW ORDER ECC IN ERROR
5680 036452 000405 BR 75 ;BRANCH TO CONTINUE
5681 036454 023737 050372 055260 65: CMP B#GECC2,B#WECC2;COMPARE SOFTWARE ECC WITH HARDWARE ECC
5682 BEQ 75 ;BRANCH IF GOOD
5683 036464 104031 ERROR 31 ;HIGH ORDER ECC IN ERROR
5684
5685
5686 036466 75:
5687 036466 004737 046120 JSR PC,B#CHECKE ;CHECK THAT DVA, RDY, DPR, DRY = 1
5688 036472 104400 005124 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
5689 036476 000000 HALT ;STOP THE TEST AND RESTART PROGRAM
5690
5691
5692
5693
5694 ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
5695

```



```

5696 036500 004037 045612 JSR RO,2#CLAREA ;FILL REINTO BUFFER
5697 036504 016264 REINTO ;FROM
5698 036506 017262 REINTO+(255.*2) ;TO
5699 036510 177777 .WORD -1 ;DATA
5700
5701 036512 013737 050370 017264 MOV 2#GECC1,2#REINTO+(256.*2);FILL ECC1
5702 036520 013737 050372 017265 MOV 2#GECC2,2#REINTO+(257.*2);FILL ECC2
5703 036526 004037 045612 JSR RO,2#CLAREA ;FILL REST
5704 036532 017270 REINTO+(258.*2) ;FROM
5705 036534 017324 REINTO+(272.*2) ;TO
5706 036536 000000 0 ;DATA
5707
5708
5709 036540 005037 015126 CLR 2#ERFLGS ;CLEAR ERROR FLAG
5710
5711
5712 ;*NOW COMPARE "DISK" BUFFER WITH "REINTO"
5713
5714 036544 004037 046570 JSR RO,2#COMPAR ;CHECK
5715 036550 016264 REINTO ;GOOD BUFFER
5716 036552 054256 DISK ;TEST BUFFER
5717 036554 000402 258. ;NUMBER OF WORDS CHECKED
5718 036556 036564 4$ ;RETURN POINT FOR ERROR HEADER
5719 036560 036570 5$ ;RETURN POINT FOR ERROR DATA
5720
5721 036562 036574 TST60 ;RETURN FOR GOOD COMPARISON
5722
5723 036564 104007 4$: ERROR 7 ;READ ERROR 10 NEXT
5724 036566 000207 RTS 5 PC ;RETURN TO COMPARE
5725 036570 104010 5$: ERROR 10 ;WORD NOS 1 TO 256 ARE
5726 ;DATA WORDS
5727 ;WORD NOS 257 AND 258
5728 ;ARE ECC WHICH ARE CHECKED
5729 ;WORD NOS 259
5730 ;IS DATA GAP
5731 ;WORD NOS 260 TO 273
5732 ;ARE TOLERANCE GAP
5733 036572 000207 RTS PC ;RETURN TO COMPARE
5734
5735
5736
5737
5738

```

5739
5740
5741
5742
5743
5744
5745
5746
5747
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773
5774
5775
5776
5777
5778
5779
5780
5781
5782
5783
5784
5785
5786
5787
5788
5789
5790
5791
5792

036574 000004
036576 012706 001000
036602 012737 000060 017330

036610 012746 177777
036614 012705 000400
036620 012700 054256
036624 011620
036626 005305
036630 001375
036632 005726
036634 022020
036636 012705 000017

036642 005020
036644 005305
036646 001375

036650 004737 051164

036654 012737 177777 015144
036662 005037 050402
036666 013737 050376 050400
036674 013737 050404 050412
036702 005037 050370
036706 005037 050372
036712 005037 050406
036716 005037 050410

036722 012737 010000 052340

```
::*****  
:*TEST 50 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  
  
::*****  
↑ST60: SCOPE  
MOV #STACK, SP ;RESET STACK  
MOV #TTNO, @TSTNM ;THIS SAVES TEST NUMBER  
  
: SETUP FOR WHAT IS TO BE READ  
: HEADER CRC IS RESTORED FROM A SUBROUTINE  
  
MOV #-1, -(SP) ;DATA TO BE READ  
MOV #256, R5 ;COUNTER  
MOV #DISK, R0 ;START OF SIMULATED DISK DATA  
15: MOV (SP), (R0)+ ;MOVE IN DATA ON TO SIMULATED DISK  
DEC R5 ;COUNT  
BNE 15 ;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  
25: CLR (R0)+ ;CLEAR DATA GAP, AND  
DEC R5 ;TOLERANCE GAP  
BNE 25 ;BRANCH IF NOT COMPLETE  
  
JSR PC, @#FILLEC ;INSERT THE TWO ECC WORDS ON THE DISK  
;IN THE CORRECT PLACE  
  
:*THESE ARE FOR ECC TEST ONLY  
MOV #-1, @#TSECC ;THIS IS AN ECC TEST  
CLR @#POSITI ;CLEAR ERROR POSITION COUNTER  
MOV @#NCODE, @#NCOUNT ;TEMPORARY N-CODE COUNTER  
MOV @#HARDER, @#HADTMP ;TEMPORARY HARD ERROR COUNTER  
CLR @#GECC1 ;ECC LOW ORDER TO BE GENERATED  
CLR @#GECC2 ;ECC HIGH ORDER TO BE GENERATED  
CLR @#DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT  
CLR @#ZCODE ;CLEAR LEADING ZEROS CLOCK COUNT  
  
:*THESE ARE TO SETUP FOR DISKLESS USE ONLY  
MOV #FMT22, @#CYL ;16 BITS PER WORD
```

```

5793                                     ;CYLINDER 0, FORMAT 16 BITS
5794 036730 112737 000000 052343      MOVB  #0, @#SECOTR+1 ;TRACK 0
5795 036736 112737 000000 052342      MOVB  #0, @#SECOTR  ;SECTOR 0
5796 036744 012737 000000 052344      MOV   #0, @#KEY1    ;KEY1=0
5797 036752 012737 000000 052346      MOV   #0, @#KEY2    ;KEY2=0
5798 036760 012737 000400 052420      MOV   #256., @#DAWORD ;NO. OF DATA WORDS
5799 036766 005037 052350              CLR   @#X           ;THIS IS A READ COMMAND
5800 036772 004537 047102              JSR   R5, @#CRC     ;GO TO CALCULATE CRC
5801 036776 052340                      CYL
5802 03700C 054240                      WCRC
5803
5804
5805
5806
5807                                     ;*THESE ARE REGULAR SETUPS
5808
5809 037002 004737 045674              JSR   PC, @#CLDISK ;SETUP GENERAL REGISTERS
5810 037006 012777 177374 155736      MOV   #-256.-4., @RHWC ;256. DATA 4 HEADER WORDS
5811 037014 012777 016264 155732      MOV   @#REINTO, @RHBA ;STARTING ADDRESS OF READ BUFFER
5812 037022 112746 000000              MOVB  #0, -(SP)     ;IN LOWER BYTE GET SECTOR
5813 037026 112766 000000 000001      MOVB  #0, 1(SP)    ;GET TRACK IN HIGHER BYTE
5814 037034 012677 155724              MOV   (SP)+, @RHST ;TRACK/SECTOR IN RHST
5815 037040 012777 010000 155722      MOV   @#FMT22, @RHOF ;16 BITS PER WORD
5816                                     ;ECC CORRECTION NOT INHIBIT
5817                                     ;BECAUSE ECC IS NOT GOING
5818                                     ;TO BE CHECKED
5819 037046 005077 155720              CLR   @RHCA       ;CYLINDER 0
5820
5821 037052 004737 045730              JSR   PC, @#CHECKT ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
5822 037056 104400 005124              TYPE  .CPHALT     ;CANNOT CONTINUE TESTING IF ANY OF THE
5823                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK & 1
5824 037062 000000                      HALT              ;STOP THE TEST
5825
5826 037064 013711 015200              MOV   @#REFOR, @R1 ;READ HEADER AND DATA=72
5827 037070 005037 015126              CLR   @#ERFLG$    ;CLEAR ERROR FLAG
5828 037074 004737 052200              JSR   PC, @#COMHD  ;READ HEADER AND DATA
5829                                     ;IF THERE ARE READ ERRORS THEN
5830                                     ;ECC WILL NOT BE CHECKED
5831
5832
5833                                     ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5834                                     ;*FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP.
5835                                     ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5836                                     ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
5837                                     ;*DETECTED
5838                                     ;*HEADER AND DATA ARE TO BE CHECKED.
5839                                     ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
5840                                     ;*"WRFRM" IS FILLED WITH EXPECTED DATA AND
5841                                     ;*COMPARISONS ARE MADE
5842
5843 037100 005737 015126              TST   @#ERFLG$    ;ANY ERRORS ALREADY THERE
5844 037104 001102                      BNE   TST61       ;BRANCH IF YES
5845 037106 004737 045374              JSR   PC, @#PUTREG ;SAVE REGISTERS
5846 037112 005737 015036              TST   @#ERI       ;NO ERRORS SHOULD BE SET

```

```

5847 037116 001401          BEQ      6$          ;BRANCH IF NO ERRORS SET
5848 037120 104032          ERROR   32          ;32 BIT ECC REGISTER SHOULD BE ZERO
5849                                     ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5850                                     ;IN THE PATERN REGISTER
5851                                     ;DCK SHOULD BE SET IN RHER;
5852 037122 013746 050370    6$:   MOV      @#GECC1,-(SP) ;GET PATTERN REGISTER
5853 037126 042716 174000    BIC      #174000,(SP) ;KEEP ONLY 11 BITS
5854 037132 022637 015066    CMP      (SP)+,@#EC2 ;COMPARE PATTERN REGISTER
5855 037136 001401          BEQ      7$          ;BRANCH IF GOOD
5856 037140 104032          ERROR   32          ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5857
5858
5859
5860
5861                                     ;*ADD 16 MAINTENANCE CLOCKS TO
5862                                     ;*BRING EBL DOWN
5863
5864 037142 012700 000020    7$:   MOV      #16,R0      ;COUNTER
5865 037146 052777 000002    8$:   BIS      #MCLK,@RHMR ;SET CLOCK
5866 037154 042777 000002    BIC      #MCLK,@RHMR ;CLEAR CLOCK
5867 037162 005300          DEC      R0          ;COUNT
5868 037164 001370          BNE      8$          ;BRANCH IF 16 CLOCKS NOT DONE
5869 037166 004737 046120    JSR      PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
5870 037172 104400 005124    TYPE    ,CPHALT    ;CANNOT CONTINUE IF THEY DON'T
5871 037176 000000          HALT                    ;STOP THE TEST AND RESTART PROGRAM
5872 037200 012700 015220    MOV      #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
5873 037204 012720 010000    MOV      #0!FMT22,(R0)+ ;CYLINDER 0
5874 037210 112746 000000    MOV      #0,-(SP)    ;IN LOWER BYTE GET SECTOR
5875 037214 112766 000000    MOV      #0,1(SP)   ;GET TRACK IN HIGHER BYTE
5876 037222 012620          MOV      (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
5877 037224 012720 000000    MOV      #0,(R0)+  ;KEY1 IN BUFFER
5878 037230 012720 000000    MCV      #0,(R0)+  ;KEY2 IN BUFFER
5879 037234 012701 000400    MOV      #256.,R1   ;DATA WORD COUNTER
5880 037240 012702 177777    MOV      #-1,R2     ;DATA
5881
5882 037244 010220          3$:   MOV      R2,(R0)+  ;DATA INTO BUFFER
5883 037246 005301          DEC      R1          ;COUNT
5884 037250 001375          BNE      3$          ;BRANCH IF 256 NOT DONE
5885 037252 005037 015126    CLR      @#ERFLG$   ;CLEAR ERROR FLAG
5886 037256 004737 045374    JSR      PC,@#PUTREG ;SAVE REGISTERS
5887
5888                                     ;NOW READ DATA BUFFER WILL BE CHECKED
5889
5890 037262 004037 046570    JSR      R0,@#COMPAR ;CHECK
5891 037266 015220          WRFROM ;GOOD BUFFER
5892 037270 016264          REINTO ;TEST BUFFER
5893 037272 000404          4+256. ;NUMBER OF WORDS CHECKED
5894 037274 037302          4$      ;RETURN POINT FOR ERROR HEADER
5895 037276 037306          5$      ;RETURN POINT FOR ERROR DATA
5896
5897 037300 037312          TST61 ;RETURN FOR GOOD COMPARISON
5898
5899 037302 104004          4$:   ERROR   4          ;READ NEXT ERROR
5900 037304 000207          RTS      PC         ;RETURN TO "COMPAR"
    
```

G15

MIDEC-11-DZRJH-A.RP04-5 6 DSKLS CONTROLLER TST-PT 2
DZRJHA.TST 160 READ ECC ENABLED 2A

MACY11 27(655) 30-MAR-76 20:59 PAGE 141

SEQ C187

5901	037306	104905	SS:	ERROR	S
5902					
5903					
5904	037310	000207		RTS	PC
5905					
5906					
5907					
5908					

;WORD NOS 1 TO 4 ARE
;HEADER WORDS
;5 TO 260 ARE DATA WORDS
;RETURN TO "COMPAR"

Handwritten notes and signatures in the bottom right corner.

```

5909
5910 ::*****
5911 :*TEST 61 READ ECC ENABLED 2B
5912
5913 :* THIS IS AN ECC READ DATA TEST
5914 :* ERROR CORRECTION IS ENABLED
5915 :* A CORRECTABLE ERROR IS INSERTED IN BIT POSITION 32
5916 :* GOOD DATA USED IS 256 WORDS OF 177777
5917 :* COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
5918 :* TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA
5919
5920 ::*****
5921 037312 000004 †ST61: SCOPE
5922 037314 012706 001000 MOV #STACK,SP ;RESET STACK
5923
5924 037320 012737 000061 017330 MOV #TTNO,‡#TSTNM ;THIS SAVES TEST NUMBER
5925
5926
5927 ;*SETUP FOR WHAT IS TO BE READ
5928 ;*HEADER CRC IS RESTORED FROM A SUBROUTINE
5929
5930 037326 012746 177777 MOV #-1,-(SP) ;DATA TO BE READ
5931 037332 012705 000400 MOV #256,R5 ;COUNTER
5932 037336 012700 054256 MOV #DISK,R0 ;START OF SIMULATED DISK DATA
5933 037342 011620 15: MOV (SP),(R0)+ ;MOVE IN DATA ON TO SIMULATED DISK
5934 037344 005305 DEC R5 ;COUNT
5935 037346 001375 BNE 15 ;BRANCH IF 256 NOT COMPLETE
5936 037350 005726 TST (SP)+ ;UNDO -(SP)
5937 037352 022020 CMP (R0)+,(R0)+ ;JUMP OVER THE TWO ECC WORDS
5938 037354 012705 000017 MOV #15,R5 ;1 DATA GAP
5939 ;14 TOLERANCE GAP
5940 037360 005020 25: CLR (R0)+ ;CLEAR DATA GAP, AND
5941 037362 005305 DEC R5 ;TOLERANCE GAP
5942 037364 001375 BNE 25 ;BRANCH IF NOT COMPLETE
5943
5944
5945 037366 004737 051164 JSR PC,‡#FILLEC ;INSERT ECC IN PROPER PLACE ON DISK
5946
5947
5948
5949 ;*THESE ARE FOR ECC TEST ONLY
5950
5951 037372 012737 177777 015144 MOV #-1,‡#TSECC ;THIS IS AN ECC TEST
5952 037400 005037 050402 CLR ‡#POSITI ;CLEAR ERROR POSITION COUNTER
5953 037404 013737 050376 050400 MOV ‡#NCODE,‡#NCOUNT ;TEMPORARY N-CODE COUNTER
5954 037412 013737 050404 050412 MOV ‡#HARDER,‡#HADTMP ;TEMPORARY HARD ERROR COUNTER
5955 037420 005037 050370 CLR ‡#GECC1 ;ECC LOW ORDER TO BE GENERATED
5956 037424 005037 050372 CLR ‡#GECC2 ;ECC HIGH ORDER TO BE GENERATED
5957 037430 005037 050406 CLR ‡#DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT
5958 037434 005037 050410 CLR ‡#ZCODE ;CLEAR LEADING ZEROS CLOCK COUNT
5959
5960
5961 ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY -
5962

```

```

5963 037440 012737 010000 052340      MOV      #FMT22, @#CYL      ;16 BITS PER WORD
5964                                     ;CYLINDER 0, FORMAT 16 BITS
5965 037446 112737 000000 052343      MOV      @0, @#SECOTR+1    ;TRACK 0
5966 037454 112737 000000 052342      MOV      @0, @#SECOTR      ;SECTOR 0
5967 037462 012737 000000 052344      MOV      @0, @#KEY1        ;KEY1=0
5968 037470 012737 000000 052346      MOV      @0, @#KEY2        ;KEY2=0
5969 037476 012737 000400 052420      MOV      #256., @#DAWORD   ;NO. OF DATA WORDS
5970 037504 005037 052350                CLR      @#X                ;THIS IS A READ COMMAND
5971 037510 004537 047102                JSR      R5, @#CRC          ;GO TO CALCULATE CRC
5972 037514 052340                CYL
5973 037516 054240                WCR
5974
5975
5976                                     ;*THIS IS TO INSERT ERROR
5977                                     ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
5978                                     ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
5979                                     ;*THIS MOVE
5980
5981 037520 012737 077777 054260      MOV      #77777, @#DISK+2 ;FORCE ERROR ON BIT NUMBER 32
5982                                     ;SO ERROR POSITION REGISTER WILL SHOW
5983                                     ;22
5984 037526 012737 000026 037702      MOV      #22., @#85        ;INSERT POSITION REG.
5985
5986
5987                                     ;*THESE ARE REGULAR SETUPS
5988
5989 037534 004737 045674                JSR      PC, @#CLDISK      ;SETUP GENERAL REGISTERS
5990 037540 012777 177374 155204      MOV      #-256.-4, @#RHW  ;256. DATA 4 HEADER WORDS
5991 037546 012777 016264 155200      MOV      @#REINTO, @#RBA  ;STARTING ADDRESS OF READ BUFFER
5992 037554 112746 000000                MOV      @0, -(SP)        ;IN LOWER BYTE GET SECTOR
5993 037560 112766 000000 000001      MOV      @0, 1(SP)        ;GET TRACK IN HIGHER BYTE
5994 037566 012677 155172                MOV      (SP)+, @#RDST    ;TRACK/SECTOR IN RDST
5995 037572 012777 010000 155170      MOV      #FMT22, @#RHF   ;16 BITS PER WORD
5996                                     ;ECC CORRECTION NOT INHIBIT
5997                                     ;BECAUSE ECC IS NOT GOING
5998                                     ;TO BE CHECKED
5999 037600 005077 155166                CLR      @#RHC            ;CYLINDER 0
6000 037504 004737 045730                JSR      PC, @#CHECKT     ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
6001 037610 104400 005124                TYPE      ,CPHALT        ;CANNOT CONTINUE TESTING IF ANY OF THE
6002                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
6003 037614 000000                HALT
6004 037616 013711 015200      MOV      @#REFOR, @#R1    ;READ HEADER AND DATA=72
6005 037622 005037 015126      CLR      @#ERFLG$        ;CLEAR ERROR FLAG
6006 037626 004737 052200      JSR      PC, @#COMHD      ;READ HEADER AND DATA
6007                                     ;IF THERE ARE READ ERRORS THEN
6008                                     ;ECC WILL NOT BE CHECKED
6009
6010
6011                                     ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6012                                     ;*FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
6013                                     ;*FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
6014                                     ;*SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
6015                                     ;*DETECTED
6016                                     ;*HEADER AND DATA ARE TO BE CHECKED.
    
```

```

6017                                     ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
6018                                     ;*"WRFROM" IS FILLED WITH EXPECTED DATA AND
6019                                     ;*COMPARISONS ARE MADE
6020
6021 037632 005737 015126                TST      @#ERFLGS      ;ANY ERRORS ALREADY THERE
6022 037636 001077                        BNE      TST62        ;BRANCH IF YES
6023 037640 004737 045374                JSR      PC,@#PUTREG  ;SAVE REGISTERS
6024 037644 022737 100000 015036        CMP      #DCK,@#ERI  ;ONLY DATA CHECK ERROR SHOULD BE SET
6025 037652 001401                        BEQ      6$          ;BRANCH IF YES
6026 037654 104032                        ERROR    32          ;32 BIT ECC REGISTER SHOULD BE NON
6027                                     ;ZERO
6028                                     ;ONLY 11 OF THE 32 BITS CAN BE SEEN
6029                                     ;IN THE PATERN REGISTER
6030                                     ;DCK SHOULD BE SET IN RHER1
6031 037656 013746 050370                5$:     MOV      @#GECC1,-(SP) ;GET PATTERN REGISTER
6032 037662 042716 174000                BIC      #174000,(SP) ;KEEP ONLY 11 BITS
6033 037666 022637 015066                CMP      (SP)+,@#EC2 ;COMPARE PATTERN REGISTER
6034 037672 001401                        BEQ      7$          ;BRANCH IF GOOD
6035 037674 104032                        ERROR    32          ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6036
6037 037676 004037 051012                7$:     JSR      R0,@#ECORR ;GO TO ECC CORRECTION PROCESS
6038 037702 000026                8$:     22.          ;EXPECTED POSITION REG. WHEN CORRECTION
6039                                     ;IS COMPLETE
6040
6041
6042
6043 037704 004737 046120                JSR      PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
6044 037710 104400 005124                TYPE    ,CPHALT    ;CANNOT CONTINUE IF THEY DON'T
6045 037714 000000                        HALT                                     ;STOP THE TEST AND RESTART PROGRAM
6046 037716 012700 015220                MOV      #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
6047 037722 012720 010000                MOV      #0!FMT22,(R0)+ ;CYLINDER 0
6048 037726 112746 000000                MOVVB   #0,-(SP)    ;IN LOWER BYTE GET SECTOR
6049 037732 112766 000000 000001        MOVVB   #0,1(SP)    ;GET TRACK IN HIGHER BYTE
6050 037740 012620                        MOV      (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
6051 037742 012720 000000                MOV      #0,(R0)+  ;KEY1 IN BUFFER
6052 037746 012720 000000                MOV      #0,(R0)+  ;KEY2 IN BUFFER
6053 037752 012701 000400                MOV      #256,R1    ;DATA WORD COUNTER
6054 037756 012702 177777                MOV      #-1,R2    ;DATA
6055 037762 010220                3$:     MOV      R2,(R0)+ ;DATA INTO BUFFER
6056 037764 005301                        DEC      R1          ;COUNT
6057 037766 001375                        BNE     3$          ;BRANCH IF 256 NOT DONE
6058
6059                                     ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6060                                     ;*NOW THE INSERTED ERROR WILL BE PUT IN
6061
6062 037770 012737 077777 015232        MOV      #77777,@#WRFROM+(5*2) ;INSERTED ERROR
6063 037776 004737 045374                JSR      PC,@#PUTREG ;SAVE REGISTERS
6064 040002 005037 015126                CLR      @#ERFLGS   ;CLEAR ERROR FLAG
6065
6066
6067                                     ;*NOW READ DATA BUFFER WILL BE CHECKED
6068
6069 040006 004037 046570                JSR      R0,@#COMPAR ;CHECK
6070 040012 015220                WRFROM .          ;GOOD BUFFER
    
```


6071	040014	016264		REINTO		; TEST BUFFER
6072	040016	000404		4+256.		; NUMBER OF WORDS CHECKED
6073	040020	040026		4\$; RETURN POINT FOR ERROR HEADER
6074	040022	040032		5\$; RETURN POINT FOR ERROR DATA
6075						
6076	040024	040036		TST62		; RETURN FOR GOOD COMPARISON
6077						
6078	040026	104004	4\$:	ERROR	4	; READ NEXT ERROR
6079	040030	000207		RTS	PC	; RETURN TO "COMPAR"
6080	040032	104005	5\$:	ERROR	5	; WORD NOS 1 TO 4 ARE
6081						; HEADER WORDS
6082						; 5 TO 260 ARE DATA WORDS
6083	040034	000207		RTS	PC	; RETURN TO "COMPAR"
6084						

6085
6086
6087
6088
6089
6090
6091
6092
6093
6094
6095
6096
6097
6098
6099
6100
6101
6102
6103
6104
6105
6106
6107
6108
6109
6110
6111
6112
6113
6114
6115
6116
6117
6118
6119
6120
6121
6122
6123
6124
6125
6126
6127
6128
6129
6130
6131
6132
6133
6134
6135
6136
6137
6138

040036 000004
040040 012706 001000
040044 012737 000062 017330
040052 012746 177777
040056 012705 000400
040062 012700 054256
040066 011620
040070 005305
040072 001375
040074 005726
040076 022020
040100 012705 000017
040104 005020
040106 005305
040110 001375
040112 004737 051164
040116 012737 177777 015144
040124 005037 050402
040130 013737 050376 050400
040136 013737 050404 050412
040144 005037 050370
040150 005037 050372
040154 005037 050405
040160 005037 050410
040164 012737 010000 052340

```
*****  
*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 #TTNO, @#TSTNM ;THIS SAVES TEST NUMBER  
  
; SETUP FOR WHAT IS TO BE READ  
; HEADER CRC IS RESTORED FROM A SUBROUTINE  
  
MOV #-1, -(SP) ;DATA TO BE READ  
MOV #256, R5 ;COUNTER  
MOV #DISK, R0 ;START OF SIMULATED DISK DATA  
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  
2$: CLR (R0)+ ;CLEAR DATA GAP, AND  
DEC R5 ;TOLERANCE GAP  
BNE 2$ ;BRANCH IF NOT COMPLETE  
  
JSR PC, @#FILLEC ;INSERT THE TWO ECC WORDS ON THE DISK  
;IN THE CORRECT PLACE  
  
;*THESE ARE FOR ECC TEST ONLY  
  
MOV #-1, @#TSECC ;THIS IS AN ECC TEST  
CLR @#POSITI ;CLEAR ERROR POSITION COUNTER  
MOV @#NCODE, @#NCOUNT ;TEMPORARY N-CODE COUNTER  
MOV @#HARDER, @#HADTMP ;TEMPORARY HARD ERROR COUNTER  
CLR @#GECC1 ;ECC LOW ORDER TO BE GENERATED  
CLR @#GECC2 ;ECC HIGH ORDER TO BE GENERATED  
CLR @#DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT  
CLR @#ZCODE ;CLEAR LEADING ZEROS CLOCK COUNT  
  
;*THESE ARE TO SETUP FOR DISKLESS USE ONLY  
  
MOV #FMT22, @#CYL ;16 BITS PER WORD
```

M15

```

6139
6140 040172 112757 000000 052343      MOVB    #0, @#SECOTR+1      ;CYLINDER 0, FORMAT 16 BITS
6141 040200 112757 000000 052342      MOVB    #0, @#SECOTR      ;TRACK 0
6142 040206 012737 000000 052344      MOV     #0, @#KEY1        ;SECTOR 0
6143 040214 012737 000000 052346      MOV     #0, @#KEY2        ;KEY1=0
6144 040222 012737 000400 052420      MOV     #256., @#DAWORD    ;KEY2=0
6145 040230 005037 052350                CLR     @#X                ;NO. OF DATA WORDS
6146 040234 004537 047102                JSR     R5, @#CRC          ;THIS IS A READ COMMAND
6147 040240 052340                CYL     ;GO TO CALCULATE CRC
6148 040242 054240                WCRC
6149
6150
6151
6152
6153
6154
6155
6156 040244 012737 077757 054260      MOV     #77757, @#DISK+2   ;*THIS IS TO INSERT ERROR
6157
6158
6159 040252 012737 010040 040426      MOV     #4128., @#85      ;*THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
6160
6161
6162
6163
6164 040260 004737 045674                JSR     PC, @#CLDISK      ;*THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
6165 040264 012777 177374 154460      MOV     #-256.-4., @#RHWC ;*THIS MOVE
6166 040272 012777 016264 154454      MOV     @#REINTO, @#RHBA  ;FORCE ERROR ON BIT NUMBER 32 AND 21
6167 040300 112746 000000                MOVB    #0, -(SP)         ;SO ERROR POSITION REGISTER WILL SHOW
6168 040304 112766 000000 000001      MOVB    #0, 1(SP)        ;22
6169 040312 012677 154446                MOV     (SP)+, @#RHDST    ;INSERT POSITION REG.
6170 040316 012777 010000 154444      MOV     #FMT22, @#RHOF   ;*THESE ARE REGULAR SETUPS
6171
6172
6173
6174 040324 005077 154442                CLR     @#RHCA           ;SETUP GENERAL REGISTERS
6175
6176 040330 004737 045730                JSR     PC, @#CHECKT     ;256. DATA 4 HEADER WORDS
6177 040334 104400 005124                TYPE    ,CPHALT         ;STARTING ADDRESS OF READ BUFFER
6178
6179 040340 000000                HALT
6180
6181 040342 013711 015200                MOV     @#REFOR, @#R1    ;IN LOWER BYTE GET SECTOR
6182 040346 005037 015126                CLR     @#ERFLG$        ;GET TRACK IN HIGHER BYTE
6183 040352 004737 052200                JSR     PC, @#COMHD      ;TRACK/SECTOR IN RHDST
6184
6185
6186
6187
6188
6189
6190
6191
6192
    ;*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 SYNC'S WERE CORRECTLY
    ;*DETECTED
    ;ECC CORRECTION NOT INHIBIT
    ;BECAUSE ECC IS NOT GOING
    ;TO BE CHECKED
    ;CYLINDER 0
    ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
    ;CANNOT CONTINUE TESTING IF ANY OF THE
    ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
    ;STOP THE TEST
    ;READ HEADER AND DATA=72
    ;CLEAR ERROR FLAG
    ;READ HEADER AND DATA
    ;IF THERE ARE READ ERRORS THEN
    ;ECC WILL NOT BE CHECKED
    
```

```

6193 ;*HEADER AND DATA ARE TO BE CHECKED.
6194 ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
6195 ;*"WRFROM" IS FILLED WITH EXPECTED DATA AND
6196 ;*COMPARISONS ARE MADE
6197
6198 040356 005737 015126 TST @#ERFLG$ ;ANY ERRORS ALREADY THERE
6199 040362 001106 BNE TST63 ;BRANCH IF YES
6200 040364 004737 045374 JSR PC,@#PUTREG ;SAVE REGISTERS
6201 040370 022737 100000 015036 CMP #DCK,@#ER1 ;ONLY DATA CHECK ERROR SHOULD BE SET
6202 040376 001401 BEQ 6$ ;BRANCH IF YES
6203 040400 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE NON
6204 ;ZERO
6205 ;ONLY 11 OF THE 32 BITS CAN BE SEEN
6206 ;IN THE PATERN REGISTER
6207 ;DCK SHOULD BE SET IN RHER1
6208 040402 013746 050370 6$: MOV @#GECC1,-(SP) ;GET PATTERN REGISTER
6209 040406 042716 174000 BIC #174000,(SP) ;KEEP ONLY 11 BITS
6210 040412 022637 015066 CMP (SP)+,@#EC2 ;COMPARE PATTERN REGISTER
6211 040416 001401 BEQ 7$ ;BRANCH IF GOOD
6212 040420 104032 ERROR 32 ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6213
6214 040422 004037 051012 7$: JSR RO,@#ECORR ;GO TO ECC CORRECTION PROCESS
6215 040426 000000 8$: .WORD ;EXPECTED POSITION REG. WHEN CORRECTION
6216 ;IS COMPLETE
6217
6218 040430 004737 045374 JSR PC,@#PUTREG ;SAVE REGISTERS
6219
6220
6221 040434 022737 100100 015036 CMP #DCK!ECH,@#ER1 ;WITH ERRORS INSERTED IN BIT POSITION 2!
6222 ;AND 32 HARD ERROR BIT SHOULD SET
6223 040442 001401 BEQ 9$ ;BRANCH IF GOOD
6224 040444 104036 ERROR 36 ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
6225 ;32 HCE SHOULD SET
6226
6227
6228
6229
6230 040446 9$: JSR PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
6231 040446 004737 046120 ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
6232 040452 104400 005124 HALT ;STOP THE TEST AND RESTART PROGRAM
6233 040456 000000 MOV #WRFROM,RO ;GETTING READY TO FILL EXPECTED DATA
6234 040460 012700 015220 MOV #0!FMT22,(RO)+ ;CYLINDER 0
6235 040464 012720 010000 MOV #0,-(SP) ;IN LOWER BYTE GET SECTOR
6236 040470 112746 000000 MOV #0,1(SP) ;GET TRACK IN HIGHER BYTE
6237 040474 112766 000000 000001 MOV (SP)+,(RO)+ ;GET TRACK/SECTOR IN BUFFER
6238 040502 012620 MOV #0,(RO)+ ;KEY1 IN BUFFER
6239 040504 012720 000000 MOV #0,(RO)+ ;KEY2 IN BUFFER
6240 040510 012720 000000 MOV #256,R1 ;DATA WORD COUNTER
6241 040514 012701 000400 MOV #-1,R2 ;DATA
6242 040520 012702 177777 3$: MOV R2,(RO)+ ;DATA INTO BUFFER
6243 040524 010220 DEC R1 ;COUNT
6244 040526 005301 BNE 3$ ;BRANCH IF 256 NOT DONE
6245 040530 001375
6246
    
```

5 6 DSX 5 CONTROLLER 'S'-PT 2
READ ECC ENABLED 20

040533
040540
040544
040550
040554
040556
040560
040562
040564
040566
040570
040572
040574
040576

012737
004737
005037
004037
015320
016264
000404
040570
040574
040600
104004
000207
104005
000207

:*ONLY GOOD DATA HAS BEEN PUT IN 'WFRM'
:*NOW THE INSERTED ERROR WILL BE PUT IN

MOV 077757,0#WFRM+(5*2) :INSERTED ERROR
JSR PC,0#PUTREG :SAVE REGISTERS
CLR 0#ERFLG :CLEAR ERROR FLAG

:*NOW READ DATA BUFFER WILL BE CHECKED

JSR RD,0#COMPAR :CHECK
WFRM :GOOD BUFFER
REINTO :TEST BUFFER
4+256. :NUMBER OF WORDS CHECKED
4\$:RETURN POINT FOR ERROR HEADER
5\$:RETURN POINT FOR ERROR DATA
TSTB3 :RETURN FOR GOOD COMPARISON
4\$: ERROR 4 :READ NEXT ERROR
RTS PC :RETURN TO "COMPAR"
5\$: ERROR 5 :WORD NOS 1 TO 4 ARE
 :HEADER WORDS
 :5 TO 260 ARE DATA WORDS
RTS PC :RETURN TO "COMPAR"

```

6272
6273
6274
6275
6276
6277
6278
6279
6280
6291
6292 040600 000004
6293 040602 012706 001000
6294
6295 040606 012737 000063 017330
6296 040614 012700 054160
6297 040620 012701 00046C
6298 040624 005020
6299 040626 005301
6290 040630 001375
6291 040632 004737 045674
6292
6293
6294
6295 040636 012737 177777 015144
6296 040644 005037 050402
6297 040650 013737 050376 050400
6298 040656 013737 050404 050412
6299 040664 005037 050370
6300 040670 005037 050372
6301 040674 005037 050406
6302 040700 005037 05041C
6303
6304
6305
6306
6307
6308
6309 040704 012737 010000 055476
6310
6311 040712 012737 000001 055500
6312 040720 005037 055502
6313 040724 005037 055504
6314 040730 012737 000400 055536
6315 040736 004537 047102
6316 040742 055476
6317 040744 055506
6318
6319
6320
6321 040746 012777 177374 153776
6322 040754 012700 015220
6323 040760 010077 153770
6324
6325 040764 012720 010000

```

```

*****
*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 $2525.
*****
TST63:  SCOPE
        MOV      #STACK,SP      ;RESET STACK
        MOV      #TTNO, @TSTNM  ;THIS SAVES TEST NUMBER
        MOV      #SECGAP,RO     ;POINTER
        MOV      #304.,R1      ;COUNTER
1$:     CLR      (RO)+          ;CLEAR SIMULATED DISK AREA
        DEC      R1
        BNE     1$
        JSR     PC,CLDISK      ;THIS IS USED TO SET GENERAL REGISTERS
*****
*THESE ARE FOR ECC TEST ONLY
        MOV      #-1, @TSECC    ;THIS IS AN ECC TEST
        CLR     @POSITI        ;CLEAR ERROR POSITION COUNTER
        MOV     @NCODE, @NJCUNT ;TEMPORARY N-CODE COUNTER
        MOV     @HARDER, @HADTMP ;TEMPORARY HARD ERROR COUNTER
        CLR     @GECC1         ;ECC LOW ORDER TO BE GENERATED
        CLR     @GECC2         ;ECC HIGH ORDER TO BE GENERATED
        CLR     @DATENV        ;CLEAR DATA ENVELOPE CLOCK COUNT
        CLR     @ZCODE         ;CLEAR LEADING ZEROS CLOCK COUNT
*****
*THESE ARE TO BE SETUP FOR DISKLESS USE ONLY
        MOV     #FMT22, @WCYL   ;FORMAT22=16BIT WORDS AND
                                ;CYLINDER 0
        MOV     #1, @WSECTR     ;TRACK=0, SECTOR=1
        CLR     @WKEY1         ;KEY1=0
        CLR     @WKEY2         ;KEY2=0
        MOV     #256., @FNWORD  ;256 DATA WORDS
        JSR     R5, @CRC        ;GO TO CALCULATE CRC
        WCYL
        GCRC
*****
*THESE ARE REGULAR SETUPS
        MOV     #-260., @RHWC   ;256 DATA WORDS 4 HEADER WORDS
        MOV     #WRFROM, RO     ;THESE TWO INSTRUCTIONS GETS
        MOV     RO, @RHBA      ;ADDR. OF WRFROM INTO RO AND
                                ;BUS ADDRESS REGISTER
        MOV     #FMT22, (RO)+   ;FORMAT=16 BIT WORDS

```

```

6325
6327 040770 012720 000001 2$: MOV #1,(R0)+ ;CYLINDER=0
6328 040774 005020 CLR (R0)+ ;TRACK=0, SECTOR=1, KEYS=0
6329 040776 005020 CLR (R0)+ ;KEY1=0
6330 041000 012705 000400 MOV #256,R5 ;KEY2=0
6331 041004 012720 052525 3$: MOV #52525,(R0)+ ;COUNTER
6332 041010 005305 DEC R5 ;MOVE ALL 52525 FOR DATA
6333 041012 001374 BNE 3$ ;BRANCH IF DATA NOT COMPLETE
6334 041014 012777 000001 153742 MOV #1,RHDST ;TRACK=0 SECTOR=1
6335
6336 041022 004737 045730 JSR PC,R#CHECKT ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
6337 041026 104400 005124 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
6338 ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
6339 041032 000000 HALT ;STOP THE TEST
6340
6341 041034 013711 015174 MOV R#WRIFOR,R#1 ;GET READY FOR WRITE HEADER AND
6342 ;DATA WITH 62 IN RHCS1
6343 041040 005037 015126 CLR R#ERFLG$ ;CLEAR ERROR FLAG
6344 041044 012777 C10000 153716 MOV #FMT22,R#HOF ;FORMAT BIT=1 (16 BIT WORDS)
6345 041052 005077 153714 CLR R#HCA ;CYLINDER =0
6346 041056 004737 055322 JSR PC,R#COMWHD ;WRITE HEADER AND DATA
6347
6348 ;*IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6349 ;*FROM THE "COMWHD" ROUTINE THAT MEANS ALL HEADER ON DISK
6350 ;*IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
6351 ;*ALL 52525 AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
6352 ;*THEY ARE ALL ZEROS
6353
6354 041062 005737 015126 TST R#ERFLG$ ;HAS ANY ERRORS OCCURED?
6355 ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
6356 041066 001056 BNE TST64 ;:BRANCH IF YES
6357
6358
6359 ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
6360 041070 023737 050370 055256 CMP R#GECC1,R#WECC1 ;COMPARE SOFTWARE ECC WITH HARDWARE ECC
6361 041076 001402 BEQ 6$ ;BRANCH IF GOOD
6362 041100 104031 ERROR 31 ;LOW ORDER ECC IN ERROR
6363 041102 000405 BR 7$ ;BRANCH TO CONTINUE
6364 041104 023737 050372 055260 6$: CMP R#GECC2,R#WECC2 ;COMPARE SOFTWARE ECC WITH HARDWARE ECC
6365 041112 001401 BEQ 7$ ;BRANCH IF GOOD
6366 041114 104031 ERROR 31 ;HIGH ORDER ECC IN ERROR
6367
6368
6369 041116 7$: JSR PC,R#CHECKE ;CHECK THAT DVA, RDY, DPR, DRY = 1
6370 041116 004737 046120 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
6371 041122 104400 005124 HALT ;STOP THE TEST AND RESTART PROGRAM
6372 041126 000000
6373
6374
6375
6376
6377 ;*FILL "REINTO" BUFFER WITH EXPECTED DATA
6378
6379 041130 004037 045612 JSR R0,R#CLAREA ;FILL REINTO BUFFER

```

E16

```

6380 041134 016264 REINTO ;FROM
6381 041136 017262 REINTO+(255.*2) ;TO
6382 041140 052525 .WORD 52525 ;DATA
6383
6384 041142 013737 050370 017264 MOV J#GECC1,J#REINTO+(256.*2);FILL ECC1
6385 041150 013737 050372 017266 MOV J#GECC2,J#REINTO+(257.*2);FILL ECC2
6386 041156 004037 045612 JSR RD,J#CLAREA ;FILL REST
6387 041162 017270 REINTO+(258.*2) ;FROM
6388 041164 017324 REINTO+(272.*2) ;TO
6389 041166 000000 0 ;DATA
6390
6391
6392 041170 005037 015126 CLR J#ERFLGS ;CLEAR ERROR FLAG
6393
6394
6395 ;*NOW COMPARE "DISK" BUFFER WITH "REINTO"
6396
6397 041174 004037 045570 JSR RD,J#COMPAR ;CHECK
6398 041200 016264 REINTO ;GOOD BUFFER
6399 041202 054256 DISK ;TEST BUFFER
6400 041204 000402 258. ;NUMBER OF WORDS CHECKED
6401 041206 041214 4$ ;RETURN POINT FOR ERROR HEADER
6402 041210 041220 5$ ;RETURN POINT FOR ERROR DATA
6403
6404 041212 041224 TST64 ;RETURN FOR GOOD COMPARISON
6405
6406 041214 104007 4$: ERROR 7 ;READ ERROR 10 NEXT
6407 041216 000207 RTS PC ;RETURN TO COMPARE
6408 041220 104010 5$: ERROR 10 ;WORD NOS 1 TO 256 ARE
6409 ;DATA WORDS
6410 ;WORD NOS 257 AND 258
6411 ;ARE ECC WHICH ARE CHECKED
6412 ;WORD NOS 259
6413 ;IS DATA GAP
6414 ;WORD NOS 260 TO 273
6415 ;ARE TOLERANCE GAP
6416 041222 000207 RTS PC ;RETURN TO COMPARE

```



6417
6418
6419
6420
6421
6422
6423
6424
6425
6426
6427
6428
6429
6430
6431
6432
6433
6434
6435
6436
6437
6438
6439
6440
6441
6442
6443
6444
6445
6446
6447
6448
6449
6450
6451
6452
6453
6454
6455
6456
6457
6458
6459
6460
6461
6462
6463
6464
6465
6466
6467
6468
6469
6470

::*****
:*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

::*****

↑ST64: SCOPE
MOV #STACK, SP ;RESET STACK
MOV #TTNO, @#TSTNM ;THIS SAVES TEST NUMBER

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

MOV #52525, -(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
2\$: CLR (R0)+ ;CLEAR DATA GAP, AND
DEC R5 ;TOLERANCE GAP
BNE 2\$;BRANCH IF NOT COMPLETE
JSR PC, @#FILLEC ;INSERT THE TWO ECC WORDS ON THE DISK
;IN THE CORRECT PLACE

;*THESE ARE FOR ECC TEST ONLY

MOV #-1, @#TSECC ;THIS IS AN ECC TEST
CLR @#POSITI ;CLEAR ERROR POSITION COUNTER
MOV @#NCODE, @#NCOUNT ;TEMPORARY N-CODE COUNTER
MOV @#HARDER, @#HADTMP ;TEMPORARY HARD ERROR COUNTER
CLR @#GECC1 ;ECC LOW ORDER TO BE GENERATED
CLR @#GECC2 ;ECC HIGH ORDER TO BE GENERATED
CLR @#DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT
CLR @#ZCODE ;CLEAR LEADING ZEROS CLOCK COUNT

;*THESE ARE TO SETUP FOR DISKLESS USE ONLY

MOV #FMT22, @#CYL ;16 BITS PER WORD
;CYLINDER 0, FORMAT 16 BITS
MOV #0, @#SECOTR+1 ;TRACK 0

```

6471 041366 112737 000000 052342
6472 041374 012737 000000 052344
6473 041402 012737 000000 052346
6474 041410 012737 000400 052420
6475 041416 005037 052350
6476 041422 004537 047102
6477 041426 052340
6478 041430 054240

```

```

MOVVB #0, @#SECOTR ; SECTOR 0
MOV #0, @#KEY1 ; KEY1=0
MOV #0, @#KEY2 ; KEY2=0
MOV #256., @#DAWORD ; NO. OF DATA WORDS
CLR @#X ; THIS IS A READ COMMAND
JSR RS, @#CRC ; GO TO CALCULATE CRC
CYL
WCRC

```

;*THESE ARE REGULAR SETUPS

```

6485 041432 004737 045674
6486 041436 012777 177374 153306
6487 041444 012777 016264 153302
6488 041452 112746 000000
6489 041456 112766 000000 000001
6490 041464 012677 153274
6491 041470 012777 010000 153272

```

```

JSR PC, @#CLDISK ; SETUP GENERAL REGISTERS
MOV #-256.-4., @RHWC ; 256. DATA 4 HEADER WORDS
MOV @#REINTO, @RHBA ; STARTING ADDRESS OF READ BUFFER
MOVVB #0, -(SP) ; IN LOWER BYTE GET SECTOR
MOVVB #0, 1(SP) ; GET TRACK IN HIGHER BYTE
MOV (SP)+, @RHST ; TRACK/SECTOR IN RHST
MOV @#FMT22, @RHOF ; 16 BITS PER WORD

```

```

6492
6493
6494
6495 041476 005077 153270
6496
6497 041502 004737 045730
6498 041506 104400 005124
6499
6500 041512 000000
6501

```

```

CLR @RHCA ; ECC CORRECTION NOT INHIBIT
; BECAUSE ECC IS NOT GOING
; TO BE CHECKED
; CYLINDER 0
JSR PC, @#CHECKT ; CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
TYPE , @PHALT ; CANNOT CONTINUE TESTING IF ANY OF THE
; ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
HALT ; STOP THE TEST

```

```

6502 041514 013711 015200
6503 041520 005037 015126
6504 041524 004737 052200
6505
6506
6507
6508
6509

```

```

MOV @#REFOR, @R1 ; READ HEADER AND DATA=72
CLR @#ERFLG$ ; CLEAR ERROR FLAG
JSR PC, @#COMHD ; READ HEADER AND DATA
; IF THERE ARE READ ERRORS THEN
; ECC WILL NOT BE CHECKED

```

```

6510
6511
6512
6513
6514
6515
6516
6517
6518

```

```

;*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 SYNC'S 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

```

```

6519 041530 005737 015126
6520 041534 001102
6521 041536 004737 045374
6522 041542 005737 015036
6523 041546 001401
6524 041550 104032

```

```

TST @#ERFLG$ ; ANY ERRORS ALREADY THERE
BNE TST65 ; BRANCH IF YES
JSR PC, @#PUTREG ; SAVE REGISTERS
TST @#ER1 ; NO ERRORS SHOULD BE SET
BEQ 65 ; BRANCH IF NO ERRORS SET
ERROR 32 ; 32 BIT ECC REGISTER SHOULD BE ZERO

```


MNDEC-11-DZRJH-A, RPO4/5/6 DSKLS CONTROLLER TST-PT 2
DZRJHA.TST 164 READ ECC ENABLED 3A

MACY11 27(655) 30-MAR-76 20:59 PAGE 156

SEQ 0202

6579 041740 000207

RIS

PC

;RETURN TO "CCMPAR"

```

6580
6581      ;:*****
6582      ;*TEST 65      READ ECC ENABLED 3B
6583
6584      ;*
6585      ;*      THIS IS AN ECC READ DATA TEST
6586      ;*      ERROR CORRECTION IS ENABLED
6587      ;*      A CORRECTABLE ERROR IS INSERTED IN BIT POSITION 4128
6588      ;*      THIS IS THE LAST BIT OF THE ECC
6589      ;*      GOOD DATA USED IS 256 WORDS OF 52525
6590      ;*      COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
6591      ;*      TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA
6592      ;:*****
6593      041742 000004      †ST65: SCOPE
6594      041744 012706 001000      MOV      #STACK,SP      ;RESET STACK
6595      041750 012737 000065 017330      MOV      #TTNO,‡#TSTNM      ;THIS SAVES TEST NUMBER
6596
6597      ;*SETUP FOR WHAT IS TO BE READ
6598      ;*HEADER CRC IS RESTORED FROM A SUBROUTINE
6599
6600      041756 012746 052525      MOV      #52525, -(SP)      ;DATA TO BE READ
6601      041762 012705 000400      MOV      #256, R5      ;COUNTER
6602      041766 012700 054256      MOV      #DISK, R0      ;START OF SIMULATED DISK DATA
6603      041772 011620      1$: MOV      (SP), (R0)+      ;MOVE IN DATA ON TO SIMULATED DISK
6604      041774 005305      DEC      R5      ;COUNT
6605      041776 001375      BNE     1$      ;BRANCH IF 256 NOT COMPLETE
6606      042000 005726      TST     (SP)+      ;UNDO -(SP)
6607      042002 022020      CMP     (R0)+, (R0)+      ;JUMP OVER THE TWO ECC WORDS
6608      042004 012705 000017      MOV     #15., R5      ;1 DATA GAP
6609
6610      042010 005020      2$: CLR     (R0)+      ;14 TOLERANCE GAP
6611      042012 005305      DEC     R5      ;CLEAR DATA GAP, AND
6612      042014 001375      BNE     2$      ;TOLERANCE GAP
6613
6614
6615      042016 004737 051164      JSR     PC,‡#FILLEC      ;INSERT ECC IN PROPER PLACE ON DISK
6616
6617
6618
6619      ;*THESE ARE FOR ECC TEST ONLY
6620
6621      042022 012737 177777 015144      MOV     #-1,‡#TSECC      ;THIS IS AN ECC TEST
6622      042030 005037 050402      CLR     ‡#POSITI      ;CLEAR ERROR POSITION COUNTER
6623      042034 013737 050376 050400      MOV     ‡#NCODE,‡#NCOUNT      ;TEMPORARY N-CODE COUNTER
6624      042042 013737 050404 050412      MOV     ‡#HARDER,‡#HADTMP      ;TEMPORARY HARD ERROR COUNTER
6625      042050 005037 050370      CLR     ‡#GECC1      ;ECC LOW ORDER TO BE GENERATED
6626      042054 005037 050372      CLR     ‡#GECC2      ;ECC HIGH ORDER TO BE GENERATED
6627      042060 005037 050406      CLR     ‡#DATENV      ;CLEAR DATA ENVELOPE CLOCK COUNT
6628      042064 005037 050410      CLR     ‡#ZCODE      ;CLEAR LEADING ZEROS CLOCK COUNT
6629
6630
6631      ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY
6632
6633      042070 012737 010000 052340      MOV     #FMT22,‡#CYL      ;16 BITS PER WORD

```

Handwritten mark


```

6688
6689
6690
6691
6692
6693
6694
6695
6696
6697
6698
6699 042300 005737 015126      TST      @#ERFLG$      ;ANY ERRORS ALREADY THERE
6700 042304 001074      SNE      TST66        ;BRANCH IF YES
6701 042306 0074737 045374      JSR      PC,@#PUTREG  ;SAVE REGISTERS
6702 042312 0022737 100000 015036  CMP      @#DCK,@#ER1  ;ONLY DATA CHECK ERROR SHOULD BE SET
6703 042320 0001401      BEQ      6$          ;BRANCH IF YES
6704 042322 104032      ERROR   32          ;32 BIT ECC REGISTER SHOULD BE NON
6705                                     ;ZERO
6706                                     ;ONLY 11 OF THE 32 BITS CAN BE SEEN
6707                                     ;IN THE PATERN REGISTER
6708                                     ;DCK SHOULD BE SET IN RHER1
6709 042324 013746 050370 6$:  MOV      @#GECC1,-(SP) ;GET PATTERN REGISTER
6710 042330 042716 174000      BIC      @#174000,(SP) ;KEEP ONLY 11 BITS
6711 042334 022637 015066      CMP      (SP)+,@#EC2  ;COMPARE PATTERN REGISTER
6712 042340 001401      BEQ      7$          ;BRANCH IF GOOD
6713 042342 104032      ERROR   32          ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6714
6715 042344 004037 051012 7$:  JSR      R0,@#ECORR   ;GO TO ECC CORRECTION PROCESS
6716 042350 010026 8$:  4118.        ;EXPECTED POSITION REG. WHEN CORRECTION
6717                                     ;IS COMPLETE
6718
6719
6720
6721 042352 004737 046120      JSR      PC,@#CHECKE  ;CHECK THAT DVA,RDY,DPR,DYR = 1
6722 042356 104400 005124      TYPE    ,CPHALT     ;CANNOT CONTINUE IF THEY DON'T
6723 042362 000000      HALT                                     ;STOP THE TEST AND RESTART PROGRAM
6724 042364 012700 015220      MOV      @#WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
6725 042370 012720 010000      MOV      @#0!FMT22,(R0)+ ;CYLINDER 0
6726 042374 112746 000000      MOV      @#0,-(SP)   ;IN LOWER BYTE GET SECTOR
6727 042400 112766 000000 000001  MOV      @#0,1(SP)   ;GET TRACK IN HIGHER BYTE
6728 042406 012620      MOV      (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
6729 042410 012720 000000      MOV      @#0,(R0)+  ;KEY1 IN BUFFER
6730 042414 012720 000000      MOV      @#0,(R0)+  ;KEY2 IN BUFFER
6731 042420 012701 000400      MOV      @#256,R1   ;DATA WORD COUNTER
6732 042424 012702 052525      MOV      @#52525,R2 ;DATA
6733 042430 010220 3$:  MOV      R2,(R0)+   ;DATA INTO BUFFER
6734 042432 005301      DEC      R1         ;COUNT
6735 042434 001375      BNE     3$         ;BRANCH IF 256 NOT DONE
6736
6737 ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6738 ;*NOW THE INSERTED ERROR WILL BE PUT IN
6739 ;*BUT INSERTED ERROR IS IN ECC SO DATA IS NOT WRONG
6740
6741
    
```

M16

```

6742 042436 004737 045374 JSR PC, @#PUTREG ;SAVE REGISTERS
6743 042442 005037 015126 CLR @#ERFLG$ ;CLEAR ERROR FLAG
6744
6745
6746 ;*NOW READ DATA BUFFER WILL BE CHECKED
6747
6748 042446 004037 046570 JSR RD, @#COMPAR ;CHECK
6749 042452 015220 WRFROM ;GOOD BUFFER
6750 042454 016264 REINTO ;TEST BUFFER
6751 042456 000404 4+256. ;NUMBER OF WORDS CHECKED
6752 042460 042466 4$ ;RETURN POINT FOR ERROR HEADER
6753 042462 042472 5$ ;RETURN POINT FOR ERROR DATA
6754
6755 042464 042476 TST66 ;RETURN FOR GOOD COMPARISON
6756
6757 042466 104004 4$: ERROR 4 ;READ NEXT ERROR
6758 042470 000207 RTS PC ;RETURN TO "COMPAR"
6759 042472 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE
6760 ;HEADER WORDS
6761 ;5 TO 260 ARE DATA WORDS
6762 042474 000207 RTS PC ;RETURN TO "COMPAR"
6763

```

: *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

↑ST66: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #TTNO, @TSTNM ;THIS SAVES TEST NUMBER

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

012706 000004
012707 001000
012708 000065 017330

MOV #52525, -(SP) ;DATA TO BE READ
MOV #256, R5 ;COUNTER
MOV #DISK, RO ;START OF SIMULATED DISK DATA
15: MOV (SP), (RO)+ ;MOVE IN DATA ON TO SIMULATED DISK
DEC R5 ;COUNT
BNE 15 ;BRANCH IF 256 NOT COMPLETE
TST (SP)+ ;UNDO -(SP)
CMP (RO)+, (RO)+ ;JUMP OVER THE TWO ECC WORDS
MOV #15, R5 ;1 DATA GAP
25: CLR (RO)+ ;14 TOLERANCE GAP
DEC R5 ;CLEAR DATA GAP, AND
BNE 25 ;TOLERANCE GAP
;BRANCH IF NOT COMPLETE

012746 052525
012706 000400
012708 054255
011620
005305
001375
005726
022020
012705 000017
005020
005305
001375

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

042552 004737 051164

: *THESE ARE FOR ECC TEST ONLY

MOV #-1, @TSECC ;THIS IS AN ECC TEST
CLR @POSITI ;CLEAR ERROR POSITION COUNTER
MOV @NCODE, @NCOUNT ;TEMPORARY N-CODE COUNTER
MOV @HARDER, @HADTMP ;TEMPORARY HARD ERROR COUNTER
CLR @GECC1 ;ECC LOW ORDER TO BE GENERATED
CLR @GECC2 ;ECC HIGH ORDER TO BE GENERATED
CLR @DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT
CLR @ZCODE ;CLEAR LEADING ZEROS CLOCK COUNT

012737 177777 015144
005037 050402
013737 050376 050400
013737 050404 050412
005037 050370
005037 050372
005037 050406
005037 050410

: *THESE ARE TO SETUP FOR DISKLESS USE ONLY

MOV #FM*22, @CYL ;16 BITS PER WORD

012737 010000 052340

012706
012707
012708
012709
012710
012711
012712
012713
012714
012715
012716
012717
012718
012719
012720
012721
012722
012723
012724
012725
012726
012727
012728
012729
012730
012731
012732
012733
012734
012735
012736
012737
012738
012739
012740
012741
012742
012743
012744
012745
012746
012747
012748
012749
012750
012751
012752
012753
012754
012755
012756
012757
012758
012759
012760
012761
012762
012763
012764
012765
012766
012767
012768
012769
012770
012771
012772
012773
012774
012775
012776
012777
012778
012779
012780
012781
012782
012783
012784
012785
012786
012787
012788
012789
012790
012791
012792
012793
012794
012795
012796
012797
012798
012799
012800
012801
012802
012803
012804
012805
012806
012807
012808
012809
012810
012811
012812
012813
012814
012815
012816
012817
012818
012819
012820
012821
012822
012823
012824
012825
012826
012827
012828
012829
012830
012831
012832
012833
012834
012835
012836
012837
012838
012839
012840
012841
012842
012843
012844
012845
012846
012847
012848
012849
012850
012851
012852
012853
012854
012855
012856
012857
012858
012859
012860
012861
012862
012863
012864
012865
012866
012867
012868
012869
012870
012871
012872
012873
012874
012875
012876
012877
012878
012879
012880
012881
012882
012883
012884
012885
012886
012887
012888
012889
012890
012891
012892
012893
012894
012895
012896
012897
012898
012899
012900
012901
012902
012903
012904
012905
012906
012907
012908
012909
012910
012911
012912
012913
012914
012915
012916
012917
012918
012919
012920
012921
012922
012923
012924
012925
012926
012927
012928
012929
012930
012931
012932
012933
012934
012935
012936
012937
012938
012939
012940
012941
012942
012943
012944
012945
012946
012947
012948
012949
012950
012951
012952
012953
012954
012955
012956
012957
012958
012959
012960
012961
012962
012963
012964
012965
012966
012967
012968
012969
012970
012971
012972
012973
012974
012975
012976
012977
012978
012979
012980
012981
012982
012983
012984
012985
012986
012987
012988
012989
012990
012991
012992
012993
012994
012995
012996
012997
012998
012999
013000

```

6818 042632 112737 000000 052343
6819 042640 112737 000000 052342
6820 042646 012737 000000 052344
6821 042654 012737 000000 052346
6822 042662 012737 000400 052420
6823 042670 005037 052350
6824 042674 004537 047102
6825 042680 052340
6826 042682 054240

```

```

MOV #0, @SECOTR+1 ; CYLINDER 0, FORMAT 16 BITS
MOV #0, @SECOTR ; TRACK 0
MOV #0, @KEY1 ; SECTOR 0
MOV #0, @KEY2 ; KEY1=0
MOV #256, @DAWORD ; KEY2=0
CLR @X ; NO. OF DATA WORDS
JSR RS, @CRC ; THIS IS A READ COMMAND
CYL ; GO TO CALCULATE CRC
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

```

```

6835 042704 012737 152652 054322
6836 042712 012737 052532 054324
6837 042720 012737 010040 043074

```

```

MOV #152652, @DISK+44 ; INSERT ERROR IN POSITION 296 THRU 304
; IN WORD NUMBER 19
MOV #52532, @DISK+46 ; INSERT ERROR IN POSITION 305 THRU 308
; IN WORD NUMBER 20
MOV #4128, @B$ ; INSERT POSITION REG.

```

```

; *THESE ARE REGULAR SETUPS

```

```

6844 042726 004737 045674
6845 042732 012777 177374 152012
6846 042740 012777 016264 152006
6847 042746 112746 000000
6848 042752 112766 000000 000001
6849 042760 012677 152000
6850 042764 012777 010000 151776

```

```

JSR PC, @CLDISK ; SETUP GENERAL REGISTERS
MOV #-256, @RHWC ; 256. DATA 4 HEADER WORDS
MOV @REINTO, @RHEA ; STARTING ADDRESS OF READ BUFFER
MOV #0, -(SP) ; IN LOWER BYTE GET SECTOR
MOV #0, 1(SP) ; GET TRACK IN HIGHER BYTE
MOV (SP)+, @RHST ; TRACK/SECTOR IN RHST
MOV @FMT22, @RHOF ; 16 BITS PER WORD
; ECC CORRECTION NOT INHIBIT
; BECAUSE ECC IS NOT GOING
; TO BE CHECKED
CLR @RHCA ; CYLINDER 0
JSR PC, @CHECKT ; CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
TYPE ,CPHALT ; CANNOT CONTINUE TESTING IF ANY OF THE
; ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
HALT ; STOP THE TEST
MOV @REFOR, @RI ; READ HEADER AND DATA=72
CLR @ERFLG$ ; CLEAR ERROR FLAG
JSR PC, @COMHD ; READ HEADER AND DATA
; IF THERE ARE READ ERRORS THEN
; ECC WILL NOT BE CHECKED

```

```

6854 042772 005077 151774
6855 042776 004737 045730
6856 043002 104400 005124
6857
6858 043006 000000
6859 043010 013711 015200
6860 043014 005037 015126
6861 043020 004737 052200
6862
6863
6864
6865
6866
6867
6868
6869
6870
6871

```

```

; *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 SYNC'S WERE CORRECTLY
; *DETECTED
; *HEADER AND DATA ARE TO BE CHECKED.

```



```

6872 ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
6873 ;*"WRFROM" IS FILLED WITH EXPECTED DATA AND
6874 ;*COMPARISONS ARE MADE
6875
6876 043024 005737 015126 TST 2#ERFLG$ ;ANY ERRORS ALREADY THERE
6877 043030 001111 BNE TST67 ;BRANCH IF YES
6878 043032 004737 045374 JSR PC,2#PUTREG ;SAVE REGISTERS
6879 043036 022737 100000 015036 CMP 2#DCK,2#ER1 ;ONLY DATA CHECK ERROR SHOULD BE SET
6880 043044 001401 BEQ 6$ ;BRANCH IF YES
6881 043046 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE NON
6882 ;ZERO
6883 ;ONLY 11 OF THE 32 BITS CAN BE SEEN
6884 ;IN THE PATERN REGISTER
6885 ;DCK SHOULD BE SET IN RHER1
6886 043050 013746 050370 6$: MOV 2#GECC1,-(SP) ;GET PATTERN REGISTER
6887 043054 042716 174000 BIC 2#174000,(SP) ;KEEP ONLY 11 BITS
6888 043060 022637 015066 CMP (SP)+,2#EC2 ;COMPA PATTERN REGISTER
6889 043064 001401 BEQ 7$ ;BRANCH IF GOOD
6890 043066 104032 ERROR 32 ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6891
6892 043070 004037 051012 7$: JSR RD,2#ECORR ;GO TO ECC CORRECTION PROCESS
6893 043074 000000 9$: .WORD ;EXPECTED POSITION REG. WHEN CORRECTION
6894 ;IS COMPLETE
6895
6896
6897
6898 043076 004737 045374 JSR PC,2#PUTREG ;SAVE REGISTERS
6899 043102 022737 100100 015036 CMP 2#DCK!ECH,2#ER1 ;WITH ERRORS INSERTED IN BIT POSITION 21
6900 ;THRU 32 HARD ERROR BIT SHOULD SET
6901 043110 001401 BEQ 9$ ;BRANCH IF GOOD
6902 043112 104036 ERROR 36 ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
6903 ;32 HCE SHOULD SET
6904
6905
6906
6907 043114 9$: JSR PC,2#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
6908 043114 004737 046120 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
6909 043120 104400 005124 HALT ;STOP THE TEST AND RESTART PROGRAM
6910 043124 000000 MOV 2#WRFROM,RD ;GETTING READY TO FILL EXPECTED DATA
6911 043126 012700 015220 MOV 2#0!FMT22,(RD)+ ;CYLINDER 0
6912 043132 012720 010000 MOV 2#0,-(SP) ;IN LOWER BYTE GET SECTOR
6913 043136 112746 000000 MOV 2#0,1(SP) ;GET TRACK IN HIGHER BYTE
6914 043142 112766 000000 000001 MOV (SP)+,(RD)+ ;GET TRACK/SECTOR IN BUFFER
6915 043150 012620 MOV 2#0,(RD)+ ;KEY1 IN BUFFER
6916 043152 012720 000000 MOV 2#0,(RD)+ ;KEY2 IN BUFFER
6917 043156 012720 000000 MOV 2#256,R1 ;DATA WORD COUNTER
6918 043162 012701 000400 MOV 2#52525,R2 ;DATA
6919 043166 012702 052525 3$: MOV R2,(RD)+ ;DATA INTO BUFFER
6920 043172 010220 DEC R1 ;COUNT
6921 043174 005301 BNE 3$ ;BRANCH IF 256 NOT DONE
6922 043176 001375
6923
6924 ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6925 ;*NOW THE INSERTED ERROR WILL BE PUT IN

```

E01

MNDEC-11-DZRJM-A, RPO4/5 6 DSKLS CONTROLLER TST-PT 2
 DZRJMA.TST T66 READ ECC ENABLED 3C

MACY11 27(655) 30-MAR-76 20:59 PAGE 164

SEQ 0210

```

6926
6927 043200 012737 152652 015274      MOV      #152652,2#WRFROM+54;INSERT ERROR IN POSITION 296 THRU 304
6928                                     ;IN WORD NUMBER 19 IN DATA
6929 043206 012737 052532 015276      MOV      #52532,2#WRFROM+56;INSERT ERROR IN POSITION 305 THRU 308
6930                                     ;IN WORD NUMBER 20 IN DATA
6931
6932 043214 004737 045374      JSR      PC,2#PUTREG      ;SAVE REGISTERS
6933 043220 005037 015126      CLR      2#ERFLGS        ;CLEAR ERROR FLAG
6934
6935
6936                                     ;*NOW READ DATA BUFFER WILL BE CHECKED
6937
6938 043224 004037 046570      JSR      RD,2#COMPAR      ;CHECK
6939 043230 015220      WRFROM      ;GOOD BUFFER
6940 043232 016264      REINTO      ;TEST BUFFER
6941 043234 000404      4+256.      ;NUMBER OF WORDS CHECKED
6942 043236 043244      4$          ;RETURN POINT FOR ERROR HEADER
6943 043240 043250      5$          ;RETURN POINT FOR ERROR DATA
6944
6945 043242 043254      TST67      ;RETURN FOR GOOD COMPARISON
6946
6947 043244 104004      4$: ERROR 4      ;READ NEXT ERROR
6948 043246 000207      RTS PC      ;RETURN TO "COMPAR"
6949 043250 104005      5$: ERROR 5      ;WORD NOS 1 TO 4 ARE
6950                                     ;HEADER WORDS
6951                                     ;5 TO 260 ARE DATA WORDS
6952 043252 000207      RTS PC      ;RETURN TO "COMPAR"
6953
    
```

```

6954
6955
6956
6957
6958
6959
6960
6961
6962
6963
6964
6965
6966
6967 043254 000004
6968 043255 012706 001000
6969
6970 043262 012737 000067 017330
6971
6972
6973
6974
6975
6976 043270 012746 052525
6977 043274 012705 000400
6978 043300 012700 054256
6979 043304 011620
6980 043306 005305
6981 043310 001375
6982 043312 005726
6983 043314 022020
6984 043316 012705 000017
6985
6986 043322 005020
6987 043324 005305
6988 043326 001375
6989
6990
6991 043330 004737 051164
6992
6993
6994
6995
6996 043334 012737 177777 015144
6997 043342 005037 050402
6998 043346 013737 050376 050400
6999 043354 013737 050404 050412
7000 043362 005037 050370
7001 043366 005037 050372
7002 043372 005037 050406
7003 043376 005037 050410
7004
7005
7006
7007

```

```

:*****
:*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

:*****
†T67: SCOPE
      MOV      #STACK,SP      ;RESET STACK
      MOV      #TTNO,‡#TSTNM ;THIS SAVES TEST NUMBER

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

      MOV      #52525, -(SP)  ;DATA TO BE READ
      MOV      #256.,R5      ;COUNTER
      MOV      #DISK,RD      ;START OF SIMULATED DISK DATA
1$:  MOV      (SP),(RD)+      ;MOVE IN DATA ON TO SIMULATED DISK
      DEC      R5            ;COUNT
      BNE     1$            ;BRANCH IF 256 NOT COMPLETE
      TST     (SP)+         ;UNDO -(SP)
      CMP     (RD)+,(RD)+   ;JUMP OVER THE TWO ECC WORDS
      MOV     #15.,R5      ;1 DATA GAP
                          ;14 TOLERANCE GAP
2$:  CLR     (RD)+         ;CLEAR DATA GAP, AND
      DEC     R5            ;TOLERANCE GAP
      BNE     2$            ;BRANCH IF NOT COMPLETE

      JSR     PC,‡#FILLEC   ;INSERT THE TWO ECC WORDS ON THE DISK
                          ;IN THE CORRECT PLACE

      ;*THESE ARE FOR ECC TEST ONLY

      MOV     #-1,‡#TSECC   ;THIS IS AN ECC TEST
      CLR     ‡#POSITI     ;CLEAR ERFJR POSITION COUNTER
      MOV     ‡#NCODE,‡#NCOUNT ;TEMPORARY N-CODE COUNTER
      MOV     ‡#HARDER,‡#HADTMP ;TEMPORARY HARD ERROR COUNTER
      CLR     ‡#GECC1      ;ECC LOW ORDER TO BE GENERATED
      CLR     ‡#GECC2      ;ECC HIGH ORDER TO BE GENERATED
      CLR     ‡#DATENV     ;CLEAR DATA ENVELOPE CLOCK COUNT
      CLR     ‡#ZCODE      ;CLEAR LEADING ZEROS CLOCK COUNT

      ;*THESE ARE TO SETUP FOR DISKLESS USE ONLY

```

GO1

MNDEC-11-DZRJH-A.RP04/5/6 DSKLS CONTROLLER TST-PT 2
 DZRJHA.TST 167 READ ECC ENABLED 30

MACY11 27(655) 30-MAR-76 20:59 PAGE 166

SEQ 0212

```

7008 043402 012737 010000 052340      MOV      #FMT22, @#CYL      ; 16 BITS PER WORD
7009                                     ; CYLINDER 0, FORMAT 16 BITS
7010 043410 112737 000000 052343      MOV      #0, @#SECOTR+1    ; TRACK 0
7011 043416 112737 000000 052342      MOV      #0, @#SECOTR      ; SECTOR 0
7012 043424 012737 000000 052344      MOV      #0, @#KEY1        ; KEY1=0
7013 043432 012737 000000 052346      MOV      #0, @#KEY2        ; KEY2=0
7014 043440 012737 000400 052420      MOV      #256., @#DAWORD   ; NO. OF DATA WORDS
7015 043446 005037 052350                                     CLR      @#X                ; THIS IS A READ COMMAND
7016 043452 004537 047102                                     JSR      RS, @#CRC          ; GO TO CALCULATE CRC
7017 043456 052340                                     CYL
7018 043460 054240                                     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
7026 043462 012737 152525 054260      MOV      #152525, @#DISK+2 ; FORCE ERROR ON BIT NUMBER 32
7027 043470 012737 152525 055254      MOV      #152525, @#DISK+(255.*2); FORCE ERROR IN BIT 4096
7028 043476 012737 010040 043652      MOV      #4128., @#B5      ; INSERT POSITION REG.

; *THESE ARE REGULAR SETUPS
7033 043504 004737 045674                                     JSR      PC, @#CLDISK      ; SETUP GENERAL REGISTERS
7034 043510 012777 177374 151234      MOV      #-256.-4, @RHWC   ; 256. DATA 4 HEADER WORDS
7035 043516 012777 016264 151230      MOV      #REINTO, @RHBA    ; STARTING ADDRESS OF READ BUFFER
7036 043524 112746 000000                                     MOV      #0, -(SP)         ; IN LOWER BYTE GET SECTOR
7037 043530 112766 000000 000001      MOV      #0, 1(SP)        ; GET TRACK IN HIGHER BYTE
7038 043536 012677 151222                                     MOV      (SP)+, @RHDSST    ; TRACK/SECTOR IN RHDSST
7039 043542 012777 010000 151220      MOV      #FMT22, @RHOF    ; 16 BITS PER WORD
7040                                     ; ECC CORRECTION NOT INHIBIT
7041                                     ; BECAUSE ECC IS NOT GOING
7042                                     ; TO BE CHECKED
7043 043550 005077 151216                                     CLR      @RHCA            ; CYLINDER 0
7044
7045 043554 004737 045730                                     JSR      PC, @#CHECKT     ; CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
7046 043560 104400 005124                                     TYPE     ,CPHALT         ; CANNOT CONTINUE TESTING IF ANY OF THE
7047                                     ; ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
7048 043564 000000                                     HALT                    ; STOP THE TEST
7049
7050 043566 013711 015200      MOV      @#REFOR, @R1     ; READ HEADER AND DATA=72
7051 043572 005037 015126      CLR      @#ERFLG$        ; CLEAR ERROR FLAG
7052 043576 004737 052200      JSR      PC, @#COMHD     ; READ HEADER AND DATA
7053                                     ; IF THERE ARE READ ERRORS THEN
7054                                     ; ECC WILL NOT BE CHECKED
7055
7056
7057                                     ; *IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
7058                                     ; *FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
7059                                     ; *FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
7060                                     ; *SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
7061                                     ; *DETECTED

```

H01

```

7062 ;*HEADER AND DATA ARE TO BE CHECKED.
7063 ;*IN CHECKING READ DATA THE WRITE FROM BUFFER
7064 ;*"WRFROM" IS FILLED WITH EXPECTED DATA AND
7065 ;*COMPARISONS ARE MADE
7066
7067 043602 005737 015126 TST 2#ERFLG$ ;ANY ERRORS ALREADY THERE
7068 043606 001111 BNE TST70 ; BRANCH IF YES
7069 043610 004737 045374 JSR PC,2#PUTREG ;SAVE REGISTERS
7070 043614 022737 100000 015036 CMP #DCK,2#ER1 ;ONLY DATA CHECK ERROR SHOULD BE SET
7071 043622 001401 BEQ 6$ ;BRANCH IF YES
7072 043624 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE NON
7073 ;ZERO
7074 ;ONLY 11 OF THE 32 BITS CAN BE SEEN
7075 ;IN THE PATERN REGISTER
7076 ;DCK SHOULD BE SET IN RHER1
7077 043626 013746 050370 6$: MOV 2#GECC1,-(SP) ;GET PATTERN REGISTER
7078 043632 042716 174000 BIC #174000,(SP) ;KEEP ONLY 11 BITS
7079 043636 022637 015066 CMP (SP)+,2#EC2 ;COMPARE PATTERN REGISTER
7080 043642 001401 BEQ 7$ ;BRANCH IF GOOD
7081 043644 104032 ERROR 32 ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
7082
7083 043646 004037 051012 7$: JSR RD,2#ECORR ;GO TO ECC CORRECTION PROCESS
7084 043652 000000 8$: .WORD ;EXPECTED POSITION REG. WHEN CORRECTION
7085 ;IS COMPLETE
7086
7087
7088
7089 043654 004737 045374 JSR PC,2#PUTREG ;SAVE REGISTERS
7090 043660 022737 100100 015036 CMP #DCK!ECH,2#ER1 ;WITH ERRORS INSERTED IN BIT POSITION 32
7091 ;AND 4096 HARD ERROR BIT SHOULD SET
7092 043666 001401 BEQ 9$ ;BRANCH IF GOOD
7093 043670 104036 ERROR 36 ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
7094 ;32 HCE SHOULD SET
7095
7096
7097
7098
7099 043672 9$:
7100 043672 004737 046120 JSR PC,2#CHECKE ;CHECK THAT DVA, RDY, DPR, DRY = 1
7101 043676 104400 005124 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
7102 043702 000000 HALT ;STOP THE TEST AND RESTART PROGRAM
7103 043704 012700 015220 MOV #WRFROM,RO ;GETTING READY TO FILL EXPECTED DATA
7104 043710 012720 010000 MOV #0!FMT22,(RO)+ ;CYLINDER 0
7105 043714 112746 000000 MOVB #0,-(SP) ;IN LOWER BYTE GET SECTOR
7106 043720 112766 000000 000001 MOVB #0,1(SP) ;GET TRACK IN HIGHER BYTE
7107 043726 012620 MOV (SP)+,(RO)+ ;GET TRACK/SECTOR IN BUFFER
7108 043730 012720 000000 MOV #0,(RO)+ ;KEY1 IN BUFFER
7109 043734 012720 000000 MOV #0,(RO)+ ;KEY2 IN BUFFER
7110 043740 012701 000400 MOV #256,R1 ;DATA WORD COUNTER
7111 043744 012702 052525 MOV #52525,R2 ;DATA
7112 043750 010220 3$: MOV R2,(RO)+ ;DATA INTO BUFFER
7113 043752 005301 DEC R1 ;COUNT
7114 043754 001375 BNE 3$ ;BRANCH IF 256 NOT DONE
7115

```

19

I01

MMDEC-11-DZRJH-A.RP04/5/6 DSKLS CONTROLLER TST-PT 2
 DZRJHA.TST 167 READ ECC ENABLED 3D

MACY11 27(655) 30-MAR-76 20:59 PAGE 168

SEQ 0214

```

7116                                     ;*ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
7117                                     ;*NOW THE INSERTED ERROR WILL BE PUT IN
7118
7119 043756 012737 152525 015232      MOV   #152525,@WRFROM+(<5*2>  ;INSERTED ERROR IN BIT 32
7120 043764 012737 152525 016226      MOV   #152525,@WRFROM+(<259.*2> ;INSERT ERROR IN BIT 4096
7121
7122 043772 005037 015126              CLR   @ERFLGS                ;CLEAR ERROR FLAG
7123 043776 004737 045374              JSR   PC,@PUTREG            ;SAVE REGISTERS
7124
7125                                     ;*NOW READ DATA BUFFER WILL BE CHECKED
7126
7127 044002 004037 046570              JSR   RD,@COMPAR           ;CHECK
7128 044006 015220                      WRFROM                      ;GOOD BUFFER (CHANGED)
7129 044010 016264                      REINTO                      ;TEST BUFFER
7130 044012 000404                      4+256.                      ;NUMBER OF WORDS CHECKED
7131 044014 044022                      4$                          ;RETURN POINT FOR ERROR HEADER
7132 044016 044026                      5$                          ;RETURN POINT FOR ERROR DATA
7133
7134 044020 044032                      TST70                       ;RETURN FOR GOOD COMPARISON
7135
7136 044022 104004                      4$: ERROR 4                 ;READ NEXT ERROR
7137 044024 000207                      RTS PC                       ;RETURN TO "COMPAR"
7138 044026 104005                      5$: ERROR 5                 ;WORD NOS 1 TO 4 ARE
7139                                     ;HEADER WORDS
7140                                     ;5 TO 260 ARE DATA WORDS
7141 044030 000207                      RTS PC                       ;RETURN TO "COMPAR"
  
```


.SBTTL CURSORY INTERRUPT LOGIC TESTS

7142
7143
7144
7145
7146
7147
7148
7149
7150
7151
7152
7153
7154
7155
7156
7157
7158
7159
7160
7161
7162
7163
7164
7165
7166
7167
7168
7169
7170
7171
7172
7173
7174
7175
7176
7177
7178
7179
7180

044032 000004
044034 012737 000070 017330
044042 012706 001000
044046 004737 045674
044052 013700 014746
044056 012720 044124
044062 012710 000340
044066 012737 000200 177776
044074 012711 000300
044100 013737 046244 001200
044106 005337 001200
044112 001375
044114 004737 045374
044120 104021
044122 000410
044124 022626
044126 004737 045374
044132 022737 004200 015034
044140 001401
044142 104021

```

*****
*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
*****
TST70: SCOPE
MOV #TTNO, @TSTNM ; THIS SAVES TEST NUMBER
MOV #STACK, SP ; RESET STACK
JSR PC, @CLDISK ; CLEAR DISK
MOV @RPVEC, R0 ; GET VECTOR ADDRESS
MOV #RPTRP1, (R0)+ ; SET INTERRUPT VECTOR
MOV #340, (R0) ; SET SERVICE ROUTINE PRIORITY
MOV #200, PS ; SET PROCESSOR PRIORITY
MOV #RDY!IE, @R1 ; RDY, IE IN RHCS1 SHOULD CAUSE INTERRUPT
MOV @TIMCNT, @STMP1 ; COUNTER
15: DEC @STMP1 ; WAIT FOR INTERRUPT
BNE 15 ; BRANCH IF NOT ZERO
; BEFORE THIS IS ZERO INTERRUPT SHOULD
; OCCUR
JSR PC, @PUTREG ; SAVE REGISTERS
ERROR 21 ; INTERRUPT DID NOT OCCUR
BR TST71 ; BRANCH TO NEXT TEST
RPTRP1: CMP (SP)+, (SP)+ ; RESTORE STACK
JSR PC, @PUTREG ; SAVE REGISTERS
CMP #DVA!RDY, @CS1 ; 'IE' SHOULD BE LOW
BEQ TST71 ; BRANCH IF GOOD
ERROR 21 ; INTERRUPT OCCURRED BUT
; 'IE' FAILED TO RESET

```

K01

7181
7182
7183
7184
7185
7186
7187
7188
7189
7190
7191
7192
7193
7194
7195
7196
7197
7198
7199
7200
7201
7202
7203
7204
7205
7206
7207
7208
7209
7210
7211
7212
7213
7214
7215

; *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

TST71: SCOPE

044144 000004

044146 012737 000071 017330

044154 312706 001000

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

MOV #STACK, SP ; RESET STACK

044160 004737 045674

JSR PC, @#CLDISK ; CLEAR DISK

044164 013700 014746

MOV @#RPVEC, R0 ; GET VECTOR ADDRESS

044170 012720 044230

MOV #RPTRP2, (R0)+ ; SET INTERRUPT VECTOR

044174 012710 000340

MOV #340, (R0) ; SET SERVICE ROUTINE PRIORITY

044200 012737 000240 177776

MOV #240, PS ; SET PROCESSOR PRIORITY

044206 012711 000300

MOV #RDY!IE, @R1 ; RDY, IE IN RHSC1 SHOULD CAUSE INTERRUPT

044212 013737 046244 001200

MOV @#TIMCNT, @#STMP1 ; COUNTER

044220 005337 001200

1\$: DEC @#STMP1 ; WAIT FOR INTERRUPT

044224 001375

BNE 1\$; BRANCH IF NOT ZERO

044226 000404

BR TST72 ; NO INTERRUPT SO BRANCH

044230 022626

RPTRP2: CMP (SP)+, (SP)+ ; RESTORE STACK

044232 004737 045374

JSR PC, @#PUTREG ; SAVE REGISTERS

044236 104021

ERROR 21 ; INTERRUPT OCCURRED WITH

21 ; PROCESSOR STATUS SAME

AS DISK

```

7216 ::*****
7217 ::*****
7218 ;*TEST 72      END OF DRIVE
7219
7220 ;*          THIS IS THE END OF TEST FOR ONE DRIVE
7221 ;*          IF THERE ARE MORE DRIVES THEN THE PROGRAM
7222 ;*          JUMPS TO TEST 5 FOR NEXT DRIVE TEST
7223 ;*          END PASS IS REACHED ONLY AFTER ALL DRIVES ARE COMPLETE
7224
7225 ::*****
7226 044240 000004  TST72: SCOPE
7227 044242 012737 000001 001212  MOV      #1,$TIMES      ;;DO 1 ITERATION
7228 044250 004737 045674      JSR      PC,$#CLDISK
7229
7230 044254 012737 000000 177776  MOV      #0,PS          ;REINSTATE PS TO 0
7231 044262 104400 044270      TYPE     ,65$          ;;TYPE ASCIZ STRING
7232 044266 000425      BR       64$          ;;GET OVER THE ASCIZ
7233 ;:65$: .ASCIZ <15><12>/TOTAL ERRORS ON THIS PASS ON UNIT NO./
7234 044342 ;64$:
7235 044342 013746 015114  MOV      $#UNIT,-(SP)  ;GET READY TO TYPE UNIT NUMBER
7236 044346 104404      TYPDS
7237 044350 104400 044356  TYPE     ,67$          ;;TYPE ASCIZ STRING
7238 044354 000402      BR       66$          ;;GET OVER THE ASCIZ
7239 ;:67$: .ASCIZ /= /
7240 044362 ;66$:
7241 044362 013746 001112  MOV      $#SERTTL,-(SP) ;GET READY TO TYPE NUMBER OF ERRORS
7242 044366 104404      TYPDS
7243 044370 005037 001112  CLR      $#SERTTL     ;CLEAR TOTAL NUMBER OF ERRORS
7244 044374 005037 001102  CLR      $#STSTNM    ;CLEAR TEST NUMBER
7245 044400 005737 015122  TST      $#SELECT    ;STARTING FROM 200 ?
7246 044404 001413      BEQ     3$           ;TEST NEXT DRIVE IF SO
7247
7248 044406 005237 001100  INC      $#SPASS     ;INCREASE PASS COUNT
7249 044412 104400 044575  TYPE     ,SENDMG     ;TYPE END PASS #
7250 044416 013746 001100  MOV      $#SPASS,-(SP)
7251 044422 104404      TYPDS
7252 044424 104400 044572  TYPE     ,SENULL
7253 044430 000137 022554  JMP      $#TSTS      ;CONTINUE TESTING THIS DRIVE ----->
7254
7255 044434 005337 015116  3$: DEC      $#NUNITS   ;NO. OF UNITS PRESENT DECREMENTED
7256 044440 001413      BEQ     $EOP         ;BRANCH IF ALL DRIVES COMPLETE
7257 044442 013700 015114  MOV      $#UNIT,R0   ;UNIT UNDER TEST
7258 044446 012701 015074  MOV      #UNITS,R1  ;TABLE
7259 044452 022100  1$: CMP      (R1)+,R0   ;IS THIS UNIT JUST TESTED
7260 044454 001401      BEQ     2$         ;BRANCH IF YES
7261 044456 000775      BR     1$         ;BRANCH IF NO
7262 044460 011137 015114  2$: MOV      (R1), $#UNIT ;THIS IS NEXT UNIT
7263 044464 000137 022554  JMP      $#TSTS     ;TEST NEXT DRIVE ----->

```

7264
7265
7266
7267
7268
7269
7270
7271
7272
7273
7274
7275
7276
7277
7278
7279
7280
7281
7282
7283
7284
7285
7286
7287
7288
7289
7290
7291
7292
7293
7294
7295
7296
7297
7298
7299
7300
7301
7302
7303
7304
7305
7306
7307
7308
7309
7310
7311
7312
7313
7314
7315
7316
7317

.SBTTL
.SBTTL ***SUBROUTINES***
.SBTTL

.SBTTL END OF PASS ROUTINE

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

\$EOP:

SCOPE
CLR \$TSTNM ;: ZERO THE TEST NUMBER
CLR \$TIMES ;: ZERC THE NUMBER OF ITERATIONS
INC \$PASS ;: INCREMENT THE PASS NUMBER
BIC #100000,\$PASS ;: DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;: LOOP?

\$EOPCT: .WORD 1

BGT \$DOAGN ;: YES
MOV (PC)+,2(PC)+ ;: RESTORE COUNTER

\$ENDCT: .WORD 1

\$EOPCT
TYPE \$ENDMG ;: TYPE "END PASS #"
MOV \$PASS,-(SP) ;: SAVE \$PASS FOR TYPEOUT
TYPDS ;: GO TYPE--DECIMAL ASCII WITH SIGN

\$GET42: MOV #42,R0 ;: TYPE A NULL CHARACTER
BEQ \$DOAGN ;: GET MONITOR ADDRESS
;: BRANCH IF NO MONITOR

\$ENDAD: RESET ;: CLEAR THE WORLD
JSR PC,(R0) ;: GO TO MONITOR

NOP ;: SAVE ROOM
NOP ;: FOR
NOP ;: ACT11

\$DOAGN: JMP 2(PC)+ ;: RETURN

\$RTNAD: .WORD TST1 ;: NULL CHARACTER STRING
\$NULL: .BYTE -1,-1,0 ;: NULL CHARACTER STRING

\$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

```

7318
7319
7320
7321
7322
7323
7324
7325
7326
7327
7328
7329
7330
7331
7332
7333
7334
7335
7336
7337 044612 000000
7338
7339 044614
7340 044614 005037 177776
7341 044620 104400 044626
7342 044624 000421
7343
7344 044670
7345 044670 013746 017330
7346 044674 104401
7347 044676 104400 044704
7348 044702 000414
7349
7350 044734
7351 044734 013746 001110
7352 044740 104401
7353 044742 104400 001223
7354 044746 104400 044754
7355 044752 000426
7356
7357 045030
7358 045030 104400 045036
7359 045034 000420
7360
7361 045076
7362 045076 104400 045104
7363 045102 000423
7364
7365 045152
7366 045152 104411
7367 045154 062716 000002
7368 045160 012637 001106
7369 045164 104400 045172
7370 045170 000417
7371

```

```

;*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.
;*****

```

```

TESTAD: 0 ;FIRST ADDRESS OF TEST
OPERSEL:
CLR PS ;MAKE PROCESSOR STATUS ZERO
TYPE ,65$ ;;TYPE ASCIZ STRING
BR ,64$ ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <15><12>/THE PROGRAM WAS IN TEST NUMBER /
64$:
MOV @#TSTNM,-(SP) ;GET READY TO TYPE TEST
TYPOC ;NUMBER
TYPE ,67$ ;;TYPE ASCIZ STRING
BR ,66$ ;;GET OVER THE ASCIZ
;;67$: .ASCIZ <15><12>/THE LOOP BACK PC WAS /
66$:
MOV @#SLPERR,-(SP) ;GET READY TO TYPE LOOP BACK PC
TYPOC
TYPE ,69$ ;;TYPE ASCIZ STRING
BR ,68$ ;;GET OVER THE ASCIZ
;;69$: .ASCIZ <15><12>/SET LOOP ON ERROR OR LOOP ON TEST SWITCH/
68$:
TYPE ,71$ ;;TYPE ASCIZ STRING
BR ,70$ ;;GET OVER THE ASCIZ
;;71$: .ASCIZ <15><12>/TYPE THE FIRST PC OF THE TEST/
70$:
TYPE ,73$ ;;TYPE ASCIZ STRING
BR ,72$ ;;GET OVER THE ASCIZ
;;73$: .ASCIZ <15><12>/ FOLLOWED BY A CARRIAGE RETURN /<15><12>
72$:
RDOCT
ADD #2,(SP) ;GET LPADR
MOV (SP)+,@#SLPADR
TYPE ,75$ ;;TYPE ASCIZ STRING
BR ,74$ ;;GET OVER THE ASCIZ
;;75$: .ASCIZ <15><12>/TYPE THE PC WHERE YOU WANT/

```

```

748:
045230 104400 045236 TYPE 77$ ;;TYPE ASCIZ STRING
045230 000441 BR 46$ ;;GET OVER THE ASCIZ
045234 000441 .ASCIZ <15><12> THE PROGRAM TO LOOP BACK TO FOLLOWED BY A CARRIAGE RETURN <1
749:
045240 104411 RDOCT
045240 012637 001110 MOV (SP)+,2*SLPERR ;GET LPERR
045242 013746 001105 MOV 2*SLPADR,-(SP)
045246 013746 001105 ;THIS CLEARS UP GARBAGE
045352 005037 052454 CLR 2*NO SYNC ;CLEAR FLAG FOR HEADER ERROR COMMANDS
045356 005037 015144 CLR 2*TSECC ;CLEAR FLAG FOR ECC TEST
;WHEN =177777 IT IS AN ECC TEST
;WHEN =0 IT IS NOT AN ECC TEST
045362 005037 050374 CLP 2*TSECCG ;EVEN IN AN ECC TEST EVERY CLOCK
;IS NOT TO GENERATE ECC
;IF =177777 GENERATE ECC
;IF =0 DO NOT GENERATE ECC
045366 005037 015146 CLR 2*TESDTE ;DRIVE TIMING ERROR TEST
045372 000002 RTY

```

7392
7393
7394
7395
7396
7397
7398
7399
7400
7401
7402
7403
7404
7405
7406
7407
7408
7409
7410
7411
7412
7413
7414
7415
7416
7417
7418

045374
045374 010046
045376 010146
045400 010246
045402 012700 014752
045406 012701 015026
045412 012702 000023
045416 013021
045420 005302
045422 001375
045424 012602
045426 012501
045430 012500
045432 000207

.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
::~*****

```
PUTREG:  MOV    R0,-(SP)      ;; PUSH R0 ON STACK
         MOV    R1,-(SP)      ;; PUSH R1 ON STACK
         MOV    R2,-(SP)      ;; PUSH R2 ON STACK
         MOV    *RWC,R0      ;; STARTING ADDRESS OF REG
         MOV    *WC,R1       ;; STARTING ADDRESS OF SAVE LOCATIONS
         MOV    *RHC-RWC+2/2,R2 ;; NUMBER OF REG. INTO R2
10$:    MOV    2(R0)+,(R1)+    ;; SAVE HARDWARE REG.
         DEC    R2
         BNE   10$
         MOV    (SP)+,R2     ;; POP STACK INTO R2
         MOV    (SP)+,R1     ;; POP STACK INTO R1
         MOV    (SP)+,R0     ;; POP STACK INTO R0
         RTS   PC
```

7419
7420
7421
7422
7423
7424
7425
7426
7427
7428
7429
7430
7431
7432
7433
7434
7435
7436
7437
7438
7439
7440
7441
7442
7443
7444
7445
7446
7447
7448
7449
7450
7451
7452
7453
7454
7455
7456
7457
7458
7459

045434 000000
045436 000000
045440 000000

045442 012537 045434
045446 012504
045450 010437 045440
045454 010537 045436
045460 062705 000004
045464 012703 000001
045470 004737 045512
045474 004737 045512
045500 000241
045502 006103
045504 005703
045506 001370
045510 000205
045512 005103
045514 012737 045522 045654
045522 010337 001124
045526 005137 045434
045532 043737 045434 001124
045540 005137 045434
045544 013714 001124
045550 011437 001126
045554 005137 045434
045560 043737 045434 001126
045566 005137 045434
045572 023737 001124 001126
045600 001403
045602 004777 177630
045606 104412
045610 000207

.SBTTL FLOAT 1 AND 0

```

:*****
:*FLOAT A ONE AND A ZERO THRU A DESIGNATED REGISTER
:*ABSOLUTE ADDRESS OF REG. UNDER TEST IS IN R4
:*****
MASK: 0 ;BITS UNDER TEST
LERR: 0 ;ERROR HLT ADDRESS
REGADR: 0

BITST: MOV (R5)+, MASK ;FETCH DATA MASK
        MOV (R5)+, R4 ;GET ADDRESS OF REG. UNDER TEST
        MOV R4, REGADR
        MOV R5, LERR ;GET ERROR RETURN ADDR.
        ADD #4, R5 ;MODIFY RETURN ADDR. TO JUMP OVER RTS
        MOV #1, R3 ;INITIALIZE DATA PATTERN
BLT1: JSR PC, BLT2 ;OUTPUT FLOATING ZERO
      JSR PC, BLT2 ;OUTPUT FLOATING ONE
      CLC
      ROL R3 ;SHIFT PATTERN
      TST R3
      BNE BLT1 ;BRANCH IF NOT COMPLETE
      RTS R5 ;RETURN TO TEST
BLT2: COM R3 ;COMPLEMENT PATTERN
      MOV #BLT3, @LAD ;SET SCOPE LOOP
BLT3: MOV R3, @SGDDAT ;STORE GOOD DATA
      COM @MASK ;AND MASK WITH PATTERN
      BIC @MASK, @SGDDAT ;CLEAR THE REST
      COM @MASK ;RESTORE MASK
      MOV @SGDDAT, (R4) ;OUTPUT TO REGISTER
      MOV (R4), @SBDDAT ;INPUT FROM REGISTER
      COM @MASK
      BIC @MASK, @SBDDAT ;AND MASK OUT RECEIVED DATA
      COM @MASK ;RESTORE MASK
      CMP @SGDDAT, @SBDDAT ;IS DATA CORRECT
      BEQ IS ;BRANCH IF GOOD
      JSR PC, @LERR ;GO TO REPORT ERROR
      SCOP1 ;LOCAL SCOPE LOOP
IS: RTS PC
```


.SBTTL CLEAR MEMORY ROUTINE

7460
7461
7462
7463
7464
7465
7466
7467
7468
7469
7470
7471
7472
7473
7474
7475
7476
7477
7478
7479
7480
7481
7482
7483
7484
7485
7486
7487
7488
7489
7490
7491
7492
7493
7494
7495

045612
045612 010146
045614 010246
045616 010346
045620 012001
045622 012002
045624 012003
045626 160102
045630 062702
045634 010321
045636 005302
045640 005302
045642 001374
045644 012603
045646 012602
045650 012601
045652 000200

000002

```
*****
* THIS CLEARS ANY BLOCK OF MEMORY
* FILLING IT WITH ANY DATA
*
* CALL
* JSR      RO,CLAREA
* X
* Y
* Z
*
* *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
* *****
```

```
CLAREA:
MOV      R1,-(SP)      ;; PUSH R1 ON STACK
MOV      R2,-(SP)      ;; PUSH R2 ON STACK
MOV      R3,-(SP)      ;; PUSH R3 ON STACK
MOV      (R0)+,R1      ;; FROM
MOV      (R0)+,R2      ;; TO
MOV      (R0)+,R3      ;; DATA
SUB      R1,R2          ;; NO. OF LOCATIONS MINUS TWO
ADD      #2,R2          ;; GET TWICE NO OF LOCATIONS
1$:      MOV      R3,(R1)+ ;; MOVE IN DATA
DEC      R2
DEC      R2
BNE      1$            ;; BRANCH IF NOT COMPLETE
MOV      (SP)+,R3      ;; POP STACK INTO R3
MOV      (SP)+,R2      ;; POP STACK INTO R2
MOV      (SP)+,R1      ;; POP STACK INTO R1
RTS      R0            ;; RETURN
```

```

7496:
7497: 045654 000000          LAD: 0          .SBTTL LOCAL TRAPS
7498:
7499: 045656 032777 001000 133254 T.SCOPI: BIT      @SW09, @SWR
7500: 045664 001402          BEQ          IS
7501: 045666 013716 045654          MOV          @LAD, (SP)
7502: 045672 000002          IS: RTI
7503:

```

```

;*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

```

7514:
7515: 045674 013701 014760          CLDISK: MOV      @RHCS1,R1      ;R1 WILL BE CONTROL AND STATUS1
7516: 045700 013702 014756          MOV      @RHCS2,R2      ;R2 WILL BE CONTROL AND STATUS2
7517: 045704 013703 015002          MOV      @RHDS1,R3      ;R3 WILL BE DISK STATUS REGISTER1
7518: 045710 013704 014762          MOV      @RHER1,R4      ;R4 WILL BE ERROR REGISTER #1
7519:
7520: 045714 012712 000040          MOV      @CLR,@R2      ;CLEAR ALL REG.
7521: 045720 013712 015114          MOV      @UNIT,@R2     ;REINSTATE UNIT NO.
7522: 045724 005011          CLR      @R1           ;CLEAR FUNCTION BITS
7523: 045726 000207          RTS      PC
7524:

```

7525
7526
7527
7528
7529
7530
7531
7532
7533
7534
7535
7536
7537
7538
7539
7540
7541
7542
7543
7544
7545
7546
7547
7548
7549
7550
7551
7552
7553
7554
7555
7556
7557
7558
7559
7560
7561
7562
7563
7564
7565
7566
7567
7568
7569
7570
7571
7572
7573
7574
7575
7576
7577
7578

.SBTTL CHECK DISK STATUS ROUTINES

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

```

CHECKT: MOV      (SP), @#PCJSR      ;SAVE PC OF JSR+4
          SUB      #4, @#PCJSR      ;GET PC OF JSR
          JSR      PC, @#PUTREG     ;SAVE REGISTERS
          CMP      #DVA!RDY, @#CS1  ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
          BEQ      3$              ;AND BE READY
          ;BRANCH IF GOOD TO RHDS1 CHECK

          BIT      #DVA, @#CS1      ;BAD SO TEST DEVICE AVAILABLE
          BNE      1$              ;TEST READY IF DVA THERE
          MOV      R1, @#SBDADR     ;ADDRESS OF BAD REGISTER (RHCS1)
          ERROR   26               ;RHCS1 DID NOT HAVE DEVICE
          ;AVAILABLE AT START OF TEST
          BR       3$              ;BRANCH TO RHDS1 CHECK

          BIT      #RDY, @#CS1      ;TEST READY
          BNE      2$              ;IF RDY THERE BRANCH
          MOV      R1, @#SBDADR     ;ADDRESS OF BAD REGISTER (RHCS1)
          ERROR   26               ;RHCS1 DID NOT HAVE READY
          ;AT THE START OF TEST
          BR       3$              ;BRANCH TO NEXT COMPARE
          MOV      R1, @#SBDADR     ;ADDRESS OF BAD REGISTER (RHCS1)
          ERROR   26               ;RHCS1 HAD SOME BITS OTHER
          ;THAN DVA AND RDY SET
          ;ALL OTHER BITS SHOULD BE 0
          ;AT START OF TEST

          MOV      @#DS1, -(SP)     ;GET RHDS1
          BIC      #VV!PROG, (SP)   ;CLEAR VV AND PROGRAMABLE BIT
          CMP      #DPR!DRY, (SP)+ ;RHDS1 SHOULD HAVE THESE SET
          BEQ      8$              ;RETURN TO TEST IF GOOD

          BIT      #DPR, @#DS1      ;BAD SO TEST DRIVE PRESENT
          BNE      5$              ;CHECK DRY IF GOOD
          MOV      R3, @#SBDADR     ;ADDRESS OF BAD REGISTER (RHDS1)
          ERROR   26               ;RHDS1 DOES NOT HAVE DPR
          BR       7$              ;BRANCH OUT
          BIT      #DRY, @#DS1      ;TEST DRIVE READY
          BNE      6$              ;IF DPR WAS THERE SO BRANCH
          MOV      R3, @#SBDADR     ;ADDRESS OF BAD REGISTER (RHDS1)
          ERROR   26               ;RHDS1 DOES NOT HAVE DRY
          BR       7$              ;BRANCH OUT
          MOV      R3, @#SBDADR     ;ADDRESS OF BAD REGISTER (RHDS1)
          ERROR   26               ;RHDS1 HAS SOME BITS OTHER
          ;THAN MOL, DRY, DPR, SET

```

```

7579                                     ;ALL OTHER BITS SHOULD BE 0
7580 046110 000207           7$:   RTS   PC           ;RETURN TO TEST AND HALT - FATAL ERROR
7581
7582 046112 062716 000006   8$:   ADD   #6,(SP)      ;ADJUST STACK PTR TO GET OVER HALT IN TEST
7583 046116 000207           RTS   PC           ;RETURN TO TEST AND CONTINUE TESTING
7584
7585
7586                                     ;*****
7587                                     ;*THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCSI = 1
7588                                     ;*AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDSI = 1
7589                                     ;*IT DOES NOT CHECK THAT OTHER BITS IN THESE REGISTERS = 0
7590                                     ;*****
7591
7592 046120 011637 015134   CHECKE: MOV   (SP),@#PCJSR      ;SAVE PC OF JSR+4
7593 046124 162737 000004 015134   SUB   #4,@#PCJSR      ;GET PC OF JSR
7594 046132 004737 045374           JSR   PC,@#PUTREG     ;SAVE REGISTERS
7595 046136 032737 000200 015034   BIT   #RDY,@#CSI     ;RHCSI SHOULD HAVE DEVICE AVAILABLE
7596                                     ;AND BE READY
7597 046144 001004           BNE   1$             ;BRANCH IF GOOD
7598 046146 010137 001122           MOV   R1,@#SBDADR    ;FAILING REGISTER
7599 046152 104026           ERROR 26             ;RHCSI IS IN ERROR
7600                                     ;DOES NOT HAVE DVA, RDY
7601 046154 000427           BR    4$             ;BRANCH
7602
7603 046156 032737 004000 015034 1$:   BIT   #DVA,@#CSI     ;RHCSI SHOULD HAVE DEVICE AVAILABLE
7604                                     ;AND BE READY
7605 046164 001004           BNE   2$             ;BRANCH IF GOOD
7606 046166 010137 001122           MOV   R1,@#SBDADR    ;FAILING REGISTER
7607 046172 104026           ERROR 26             ;RHCSI IS IN ERROR
7608                                     ;DOES NOT HAVE DVA, RDY
7609 046174 000417           BR    4$             ;BRANCH OUT
7610 046176 032737 000200 015056 2$:   BIT   #DRY,@#DSI     ;RHDSI SHOULD HAVE DPR.DRY
7611 046204 001004           BNE   3$             ;BRANCH IF THERE
7612 046206 010337 001122           MOV   R3,@#SBDADR    ;FAILING REGISTER RHDSI
7613 046212 104026           ERROR 26             ;RHDSI DOES NOT HAVE DPR.DRY
7614 046214 000407           PC    4$             ;BRANCH OUT
7615 046216 032737 000400 015056 3$:   BIT   #DPR,@#DSI     ;RHDSI SHOULD HAVE DPR,DRY
7616 046224 001004           BNE   5$             ;BRANCH IF THERE
7617 046226 010337 001122           MOV   R3,@#SBDADR    ;FAILING REGISTER RHDSI
7618 046232 104026           ERROR 26             ;RHDSI DOES NOT HAVE DPR,DRY
7619 046234 000207           4$:   RTS   PC           ;RETURN TO TEST AND HALT - FATAL ERROR
7620
7621 046236 062716 000006   5$:   ADD   #5,(SP)      ;ADJUST STACK TO GET OVER HALT IN TEST
7622 046242 000207           RTS   PC           ;RETURN TO TEST AND CONTINUE TESTING

```

```

7623
7624
7625
7626
7627
7628
7629
7630
7631
7632 046244 177777
7633
7634 046246 010046
7635 046250 016600 000002
7636 046254 010037 001204
7637 046260 162737 000002 001204
7638 046266 012037 001176
7639 046272 012037 001200
7640 046276 010066 000002
7641 046302 012600
7642 046304 013737 046244 001202
7643
7644 046312 033777 001200 132656 1$:
7645 046320 001021
7646 046322 005337 001202
7647 046326 001371
7648 046330 013737 046244 001202
7649 046336 033777 001200 132632 3$:
7650 046344 001007
7651 046346 005337 001202
7652 046352 001371
7653 046354 017737 132616 001126
7654 046362 104016
7655 046364 000002 2$:
7656
7657
7658
7659
7660
7661
7662
7663
7664
7665
7666
7667

.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)
;*****
TIMCNT: 177777 ;WAITING COUNT
WAIT.T: MOV R0, -(SP) ;SAVE R0
MOV 2(SP), R0 ;GET ADDRESS OF REG. ADDRESS
MOV R0, @#STMP3 ;WAT PC+2 IN STMP3
SUB #2, @#STMP3 ;WAT PC FOR TYPEOUT
MOV (R0)+, @#STMP0 ;WAIT REGISTER ADDRESS
MOV (R0)+, @#STMP1 ;WAIT ON BIT
MOV R0, 2(SP) ;RESTORE RETURN ON STACK
MOV (SP)+, R0 ;RESTORE R0
MOV @#TIMCNT, @#STMP2 ;TEMPORARY COUNT
BIT @#STMP1, @#STMP0 ;IS REQUIRED BIT THERE?
BNE 2$ ;BRANCH IF YES
DEC @#STMP2 ;COUNT
BNE 1$ ;BRANCH IF NOT TIME UP
MOV @#TIMCNT, @#STMP2 ;TEMPORARY COUNT
BIT @#STMP1, @#STMP0 ;IS REQUIRED BIT THERE?
BNE 2$ ;BRANCH IF YES
DEC @#STMP2 ;COUNT
BNE 3$ ;BRANCH IF NOT TIME UP
MOV @#STMP0, @#SBDDAT ;REGISTER CONTENTS
ERROR 16 ;WAITED ON BIT FAILED TO SET
2$: RTI

;* CALL FOR THE ABOVE WAITLOOP IS
;*
;* MOV @A, @#XS ;A CONTAINS REGISTER ADDRESS
;* - - - ;HENCE XS WILL HAVE ABSOLUTE REG. ADR.
;* - - -
;*
;* WAT
;* XS: 0 ;ABSOLUTE REG. ADDRESS UNDER WAIT
;* .WORD 0 ;BIT WAITED FOR
;* ;CONTINUE

```


7742
7743
7744
7745
7746
7747
7748
7749
7750
7751
7752
7753
7754
7755
7756
7757
7758
7759
7760
7761
7762
7763
7764
7765
7766
7767
7768
7769
7770
7771
7772
7773
7774
7775
7776
7777
7778
7779
7780
7781
7782
7783
7784
7785
7786
7787
7788
7789
7790
7791
7792
7793
7794

.SBTTL COMPARE ROUTINE

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

COMPAR:

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

001176
001200

052460

001124

001126

052460

015126

001003

132312

000402

132306

022122

132240

042716

177177

022726

000200

001402

005303

001344

046716

012605

012604

012603

012602

012601

000200

7795
7796
7797
7798
7799
7800
7801
7802
7803
7804
7805
7806
7807
7808
7809
7810
7811
7812
7813
7814
7815
7816
7817
7818
7819
7820
7821
7822
7823
7824
7825
7826
7827
7828
7829
7830
7831
7832
7833
7834
7835
7836
7837
7838
7839
7840

.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

```
WRCHDA: MOV    #FMT22, @#CYL    ; CYLINDER 0 FORMAT 16 BIT WORDS
        MOVB   #1, @#SECOTR+1 ; TRACK=1
        MOVB   #1, @#SECOTR    ; SECTOR=1
        CLR    @#KEY1          ; KEY1=0
        CLR    @#KEY2          ; KEY2=0
        MOV    #32., @#DAWORD   ; NO OF DATA WORDS
        CLR    @#X              ; THIS IS A READ OPERATION

        JSR    R5, @#CRC        ; GO TO CALCULATE CRC
        CYL
        WCRC
```

; THESE ARE REGULAR SETUPS

```
JSR    PC, @#CLDISK          ; SET UP GENERAL REGISTERS
                                   ; AND CLEAR DISK REGISTERS

MOV    #-32., @RHWC          ; 36 DATA WORDS 4 HEADER WORDS
MOV    #REINTO, @RHBA        ; STARTING ADDRESS OF READ BUFFER
MOVE   #1, -(SP)              ; SECTOR=1
MOVE   #1, 1(SP)              ; TRACK=1 IN UPPER BYTE
MOV    (SP)+, @RHDST          ; TRACK=1, SECTOR=1 IN RHDST
MOV    #FMT22!ECI, @RHOF     ; 16 BIT WORDS
                                   ; ECC CORRECTION INHIBIT BECAUSE
                                   ; ECC LOGIC IS NOT CHECKED YET
CLR    @RHCA                  ; CYLINDER=0
JSR    PC, @#CHECKT          ; CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
TYPE   , CPHALT               ; CANNOT CONTINUE TESTING IF ANY OF THE
                                   ; ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
HALT
MOV    @#WRCHK, @R1          ; WRITE CHECK DATA=50 INTO RHCS1
JSR    PC, @#COMHD           ; WRITE CHECK HEADER AND DATA
                                   ; SAME AS READ HEADER AND DATA

RTS    PC                      ; RETURN TO WRITE CHECK TEST
```

.SBTTL CRC GENERATION ROUTINE

7841
7842
7843
7844
7845
7846
7847
7848
7849
7850
7851
7852
7853
7854
7855
7856
7857
7858
7859
7860
7861
7862
7863
7864
7865
7866
7867
7868
7869
7870
7871
7872
7873
7874
7875
7876
7877
7878
7879
7880
7881
7882
7883
7884
7885
7886
7887
7888
7889
7890
7891
7892
7893
7894

047102
047102 010046
047104 012500
047106 010146
047110 010246
047112 010346
047114 010446
047116 005001
047120 005037 001210
047124 012737 000004 001176
047132 012037 001204
047136 012737 000020 001202
047144 013737 001204 001206
047152 006037 001204
047156 006037 001210
047162 032701 000001
047166 001403
047170 012703 100000
047174 000401
047176 005003
047200 063703 001210
047204 032701 040000
047210 001403

```

;*****
;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
;STMP0 - NUMBER OF WORDS
;STMP2 - NUMBER OF BITS PER WORD = 16
;STMP3 - TEMPORARY REG.
;STMP4 - TEMPORARY REG TO TRANSFER CARRY
;STMP5 - 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
;X ;FIRST LOCATION AT
;Y ;PUT CRC IN WCRC FOR READ GCRC FOR WRITE
;*****

CRC:
MOV R0,-(SP) ;:PUSH R0 ON STACK
MOV (R5)+,R0 ;:GET POINTER TO CYL NO.
MOV R1,-(SP) ;:PUSH R1 ON STACK
MOV R2,-(SP) ;:PUSH R2 ON STACK
MOV R3,-(SP) ;:PUSH R3 ON STACK
MOV R4,-(SP) ;:PUSH R4 ON STACK
CLR R1 ;:CLEAR WORKING LOCATION
CLR @#STMP5
MOV #4,@#STMP0 ;:WORD COUNT
MOV (R0)+,@#STMP3 ;:TEMPORARY WORD STORAGE
MOV #16,@#STMP2 ;:BIT COUNT
MOV @#STMP3,@#STMP4 ;:TEMPORARY WORD STORAGE
16$: ROR @#STMP3 ;:GET LSB INTO "C"
ROR @#STMP5 ;:GET ABOVE "C" INTO STMP5
BIT #BIT0,R1 ;:IS POSITION 15 HIGH
BEQ 1$ ;:BRANCH IF POSITION 16 LOW
MOV #BIT15,R3 ;:GET POSITION 16
BR 2$
1$: CLR R3 ;:GET POSITION 16
2$: ADD @#STMP5,R3 ;:XOR POSITION 16 WITH D
;:TO GIVE B
BIT #BIT14,R1 ;:IS POSITION 2 HIGH
BEQ 3$ ;:BRANCH IF POSITION 2 LOW

```

7995	047212	012702	100000		MOV	#BIT15,R2	:GET POSITION 2
7996	047216	000401			BR	4\$	
7997	047220	005002		3\$:	CLR	R2	:GET POSITION 2
7998	047222	060302		4\$:	ADD	R3,R2	:XOR B WITH POSITION 2
7999							:TO GIVE C
8000	047224	032701	000002		BIT	#BIT1,R1	:IS POSITION 15 HIGH
8001	047230	001403			BEQ	5\$:BRANCH IF POSITION 15 LOW
8002	047232	012704	100000		MOV	#BIT15,R4	:GET POSITION 15
8003	047236	000401			BR	6\$	
8004	047240	005004		5\$:	CLR	R4	:GET POSITION 15
8005	047242	060304		6\$:	ADD	R3,R4	:XOR POSITION 15 WITH B
8006							:TO GIVE E
8007	047244	006037	001206		ROR	2#STMP4	:GET LSB INTO "C"
8008	047250	006001			ROR	R1	:GET ABOVE C INTO R1
8009	047252	005703			TST	R3	:TEST B
8010	047254	100403			BMI	7\$:BRANCH IF B=1
8011	047256	042701	100000		BIC	#BIT15,R1	:SET B IN POSITION 1
8012	047262	000402			BR	10\$	
8013	047264	052701	100000	7\$:	BIS	#BIT15,R1	:SET B IN POSITION 1
8014	047270	005702		10\$:	TST	R2	:TEST C
8015	047272	100403			BMI	11\$:BRANCH IF C=1
8016	047274	042701	020000		BIC	#BIT13,R1	:GET C IN POSITION 3
8017	047300	000402			BR	12\$	
8018	047302	052701	020000	11\$:	BIS	#BIT13,R1	:GET C IN POSITION 3
8019	047306	005704		12\$:	TST	R4	:TEST E
8020	047310	100403			BMI	13\$:BRANCH IF E=1
8021	047312	042701	000001		BIC	#BIT0,R1	:GET E IN POSITION 16
8022	047316	000402			BR	14\$	
8023	047320	052701	000001	13\$:	BIS	#BIT0,R1	:GET E IN POSITION 16
8024	047324	005337	001202	14\$:	DEC	2#STMP2	:BIT COUNTER
8025	047330	001310			BNE	15\$:BRANCH IF 16 NOT DONE
8026	047332	005337	001176		DEC	2#STMP0	:WORD COUNTER
8027	047336	001275			BNE	16\$:BRANCH IF 4 NOT DONE
8028	047340	010135			MOV	R1,2(R5)+	:PUT CRC WHERE DESIRED
8029	047342	012604			MOV	(SP)+,R4	:POP STACK INTO R4
8030	047344	012603			MOV	(SP)+,R3	:POP STACK INTO R3
8031	047346	012602			MOV	(SP)+,R2	:POP STACK INTO R2
8032	047350	012601			MOV	(SP)+,R1	:POP STACK INTO R1
8033	047352	012600			MOV	(SP)+,R0	:POP STACK INTO R0
8034	047354	000205			RTS	R5	

7935
7936
7937
7938
7939
7940
7941
7942
7943
7944
7945
7946
7947
7948
7949
7950
7951
7952
7953
7954
7955
7956
7957
7958
7959
7960
7961
7962
7963
7964
7965
7966
7967
7968
7969
7970
7971
7972
7973
7974
7975
7976
7977
7978
7979
7980
7981

.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

```
::*****
```

SETDSK:

```
MOV R0, -(SP) ;: PUSH R0 ON STACK  
MOV R1, -(SP) ;: PUSH R1 ON STACK  
MOV R2, -(SP) ;: PUSH R2 ON STACK  
MOV #177400, R0 ;: DATA IN THE DISK  
MOV #256, R1 ;: COUNTER  
MOV #DISK, R2 ;: START OF SIMULATOR DISK  
1$: MOV R0, (R2)+ ;: MOVE IN DATA  
DEC R1 ;: COUNT FOR 256  
BNE 1$ ;: BRANCH IF 256 NOT COMPLETE  
MOV #17, R1 ;: 2 ECC WORDS, 1 DATA GAP  
;: 14 TOLERANCE GAP  
2$: CLR (R2)+ ;: CLEAR ECC, DATA GAP AND  
;: TOLERANCE GAP  
DEC R1 ;: COUNT  
BNE 2$ ;: BRANCH IF NOT COMPLETE
```

:NOW SET UP FOR DISKLESS USE

```
MOV #FMT22, @#CYL ;: CYLINDER 0 (16 BIT WORDS)  
MOVB #1, @#SECOTR+1 ;: TRACK=1  
MOVB #1, @#SECOTR ;: SECTOR=1  
MOV #1, @#KEY1 ;: KEY1=1  
MOV #1, @#KEY2 ;: KEY2=1  
MOV 256, @#DAWORD ;: NO. OF DATA WORDS  
JSR R5, @#CRC ;: GO TO CALCULATE CRC  
CYL ;: FIRST CRC WORD  
WCRC ;: PUT CALCULATED CRC  
MOV (SP)+, R2 ;: POP STACK INTO R2  
MOV (SP)+, R1 ;: POP STACK INTO R1  
MOV (SP)+, R0 ;: POP STACK INTO R0  
RTS PC
```

7982
7983
7984
7985
7986
7987
7988
7989
7990
7991
7992
7993
7994
7995
7996
7997
7998
7999
8000
8001
8002
8003
8004
8005
8006
8007
8008
8009
8010
8011
8012
8013
8014
8015
8016
8017
8018
8019
8020
8021
8022
8023
8024
8025
8026
8027
8028
8029
8030
8031
8032
8033
8034
8035

.SBTTL CHECK HCE ROUTINE

```

:*****
:THIS IS A SUBROUTINE TO CHECK HEADER COMPARE ERROR
:(BIT #7) AND CRC ERROR (BIT #8)
:CALL JSR RO, @#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
:*****

```

```

HCCRCE: MOV   RO, @#PCJSR      ;SAVE PC OF JSR+4
        SUB   #4, @#PCJSR    ;GET PC OF JSR
        JSR   PC, @#CLDISK   ;INIT AND SETUP GENERAL REG.
        JSR   PC, @#CHECKT   ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
        TYPE  ,CPHALT        ;CANNOT CONTINUE TESTING IF ANY OF THE
                               ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
        HALT
        MOV   (RO), @#STMP5   ;SAVE COMMAND
        MOV   (RO)+, @R1      ;COMMAND
        MOV   (RO)+, @RHCA    ;CYLINDER
        MOV   (RO)+, -(SP)    ;SECTOR
        TST  (RO)+            ;UP DATE RO
        MOV   (RO)+, 1(SP)    ;TRACK
        TST  (RO)+            ;UPDATE RO
        MOV   (SP)+, @RHDSST  ;TRACK SECTOR
        MOV   (RO)+, @RHWC    ;NO. OF DATA WORDS +4 HEADER
                               ;IF A READ HEADER AND DATA
        MOV   (RO)+, @RHBA    ;STARTING ADDRESS OF BUFFER
        MOV   (RO)+, @#X      ;X=0 READ HEADER AND DATA
                               ;X=1 WRITE DATA
        MOV   #FMT22!ECI, @RHOF ;16 BITS PER WORD
                               ;ECC CORRECTION INHIBIT
        CLR   @#ERFLGS       ;CLEAR ERROR FLAG
        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 SYNC'S WERE CORRECTLY
                               ;DETECTED
                               ;HEADER AND DATA ARE TO BE CHECKED.
        JSR   PC, @#PUTREG    ;SAVE REGISTERS
        TST   @#ERFLGS        ;ANY ERRORS ALREADY THERE
        BNE  10$             ;BRANCH IF YES

```

015134

145162

```

047504 010037 015134
047510 162737 000004
047516 004737 045674
047522 004737 045730
047526 104400 005124
047532 000000
047534 011037 001210
047540 012011
047542 012077 145224
047546 112046
047550 105720
047552 112066 000001
047556 105720
047560 012677 145200
047564 012077 145162
047570 012077 145160
047574 012037 052350
047600 012777 014000
047606 005037 015126
047612 004737 052200
047616 004737 045374
047622 005737 015126
047626 001034

```

E03

```

8036 047630 005737 052350      TST      2#X      ; IS THIS A READ
8037 047634 001015              BNE      3$      ; IF A WRITE DATA BRANCH
8038
8039                          ; NOW THE READ BUFFER WILL BE CHECKED
8040                          ; HEADER SHOULD BE COMPLETELY READ AS WRITTEN
8041                          ; NO DATA WORDS SHOULD BE READ
8042                          ; REINTO BUFFER HAS BEEN FILLED WITH 0
8043                          ; WRFROM BUFFER HAS BEEN FILLED WITH EXPECTED DATA
8044
8045 047636 004037 046570      JSR      RO,2#COMPAR ; CHECK
8046 047642 015220              WRFROM      ; GOOD DATA
8047 047644 016264              REINTO      ; TEST BUFFER
8048 047646 000400              256.      ; 4 HEADER 252 DATA
8049 047650 047656              1$        ; RETURN POINT FOR ERROR HEADER
8050 047652 047662              2$        ; RETURN POINT FOR ERROR DATA
8051 047654 047720              10$       ; RETURN FOR GOOD COMPARISON
8052 047656 104004              1$:      ERROR 4 ; READ NEXT ERROR 5
8053 047660 000207              RTS      PC   ; RETURN TO COMPARISON SUBROUTINE
8054 047662 104005              2$:      ERROR 5 ; WORD NO 1 THRU 4 ARE
8055                          ; HEADER WORDS AND HENCE
8056                          ; SHOULD BE READ AS WRITTEN ON
8057                          ; DISK, WORD NOS. 5 ONWARDS
8058                          ; SHOULD NOT BE READ AND HENCE
8059                          ; READ INTO BUFFER
8060                          ; SHOULD BE UNCHANGED
8061 047664 000207              RTS      PC   ; RETURN TO COMPARISON
8062
8063 047666 000414              BR       10$   ; JUMP OUT
8064
8065                          ; NOW THE DISK WILL BE CHECKED
8066                          ; NO DATA SHOULD BE WRITTEN
8067                          ; REINTO BUFFER HAS BEEN FILLED WITH EXPECTED DATA
8068                          ; DISK HAS BEEN FILLED WITH 177400
8069                          ; WRFROM HAS BEEN FILLED WITH 125252
8070
8071 047670 004037 046570      3$:      JSR      RO,2#COMPAR ; CHECK
8072 047674 016264              REINTO      ; GOOD DATA BUFFER
8073 047676 054256              DISK      ; TEST BUFFER
8074 047700 000400              256.
8075 047702 047710              4$        ; RETURN POINT FOR ERROR HEADER
8076 047704 047714              5$        ; RETURN POINT FOR ERROR DATA
8077 047706 047720              10$       ; RETURN POINT FOR GOOD COMPARISON
8078 047710 104004              4$:      ERROR 4 ; READ NEXT ERROR 5
8079 047712 000207              RTS      PC   ; RETURN TO COMPARISON SUBROUTINE
8080 047714 104005              5$:      ERROR 5 ; WORD NO ARE ALL DATA
8081                          ; WORDS THE SHOULD NOT
8082                          ; HAVE BEEN CHANGED BY THE
8083                          ; WRITE COMMAND
8084 047716 000207              RTS      PC   ; RETURN TO COMPARISON SUBROUTINE
8085 047720 005720              10$:     TST      (RO)+ ; IS THIS A HCRC ON HCE CHECK
8086 047722 001442              BEQ      6$    ; BRANCH IF HCRC
8087 047724 022737 000072 001210  CMP      #72,2#STMP5 ; IS THIS A READ COMMAND
8088 047732 001417              BEQ      11$   ; BRANCH IF YES
8089 047734 017737 145022 001126  MOV      @RHER1,2#SBDDAT ; TEST DATA

```

F03

MNDEC-11-DZRJH-A, RPO4/5-6 DSKLS CONTROLLER TST-PT 2
 DZRJHA.SUB CHECK HCE ROUTINE

MACY11 27(655) 30-MAR-76 20:59 PAGE 191

SEQ 0237

8090	047742	022737	000200	001126		CMP	#HCE, @#SBDDAT	; ONLY HEADER COMPARE BIT?
8091								; SHOULD BE SET
8092	047750	001470				BEG	7\$; BRANCH IF GOOD
8093	047752	013737	014762	045440		MOV	@#RHER1, @#REGADR	; REGISTER ADDRESS RHER1
8094	047760	012737	000200	001124		MOV	#HCE, @#SGDDAT	; GOOD DATA
8095	047766	104027				ERROR	27	; AFTER AN ERROR ON THE
8096								; HEADER ONLY HCE SHOULD
8097	047770	000460				BR	7\$; BE SET
8098	047772				11\$:			
8099	047772	017737	144764	001126		MOV	@#RHER1, @#SBDDAT	; TEST DATA
8100	050000	022737	100200	001126		CMP	#DCK!HCE, @#SBDDAT	; ONLY HEADER COMPARE BIT?
8101								; SHOULD BE SET
8102								; DCK IS SET BECAUSE ECC IS NOT READ
8103	050006	001451				BEG	7\$; BRANCH IF GOOD
8104	050010	013737	014762	045440		MOV	@#RHER1, @#REGADR	; REGISTER ADDRESS RHER1
8105	050016	012737	100200	001124		MOV	#DCK!HCE, @#SGDDAT	; GOOD DATA
8106	050024	104027				ERROR	27	; AFTER AN ERROR ON THE
8107								; HEADER ONLY HCE SHOULD
8108	050026	000441				BR	7\$; BE SET
8109	050030	022737	000072	001210	6\$:	CMP	#72, @#STMP5	; IS THIS A READ COMMAND?
8110	050036	001417				BEG	12\$; BRANCH IF A READ
8111	050040	017737	144716	001126		MOV	@#RHER1, @#SBDDAT	; TEST DATA
8112	050046	022737	000400	001126		CMP	#HCRC, @#SBDDAT	; ONLY CRC ERROR SHOULD BE THERE
8113	050054	001426				BEG	7\$	
8114	050056	013737	014762	045440		MOV	@#RHER1, @#REGADR	; REG. ADDR = RHER1
8115	050064	012737	000400	001124		MOV	#HCRC, @#SGDDAT	; GOOD DATA
8116	050072	104027				ERROR	27	; AFTER A CRC ERROR ONLY CRC
8117								; SHOULD BE SET
8118	050074	000416				BR	7\$; BRANCH OUT
8119	050076	017737	144660	001126	12\$:	MOV	@#RHER1, @#SBDDAT	; TEST DATA
8120								
8121	050104	022737	100400	001126		CMP	#DCK!HCRC, @#SBDDAT	; HCRC AND DCK SHOULD BE SET
8122								; DCK IS SET BECAUSE ECC IS NOT READ
8123	050112	001407				BEG	7\$; BRANCH IF GOOD
8124	050114	012737	100400	001124		MOV	#DCK!HCRC, @#SGDDAT	; GOOD DATA
8125	050122	013737	014762	045440		MOV	@#RHER1, @#REGADR	; FAILING REGISTER RHER1
8126	050130	104027				ERROR	27	; AFTER A CRC ERROR ON A READ
8127								; DCK AND HCRC SHOULD BE SET
8128								; DCK IS SET BECAUSE ECC IS NOT READ
8129	050132	000200			7\$:	RTS	RD	; RETURN TO MAIN TEST

.SBTTL EXIT WRT HEADER & DATA ROUTINE

8130
8131
8132
8133
8134
8135
8136
8137
8138
8139
8140
8141
8142
8143
8144
8145
8146
8147
8148
8149
8150
8151
8152
8153
8154
8155
8156
8157
8158
8159
8160
8161
8162

050134
050134 010046
050136 010146
050140 013777 015174 144612
050146 012777 177766 144576
050154 012777 015220 144572
050162 012777 000010 144574
050170 052777 000010 144560
050176 012777 010000 144564
050204 005077 144562
050210 012737 000001 050236
050216 012777 000001 144554
050224 052777 000001 144526
050232 004137 056406
050236 000000
050240 012601
050242 012600
050244 000207

::*****
;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,0#MIDDLE
;BAI IS SET
::*****

MIDDLE:

MOV RO,-(SP) ;:PUSH RO ON STACK
MOV R1,-(SP) ;:PUSH R1 ON STACK
MOV @#WRIFOR,@RHCS1 ;:WRITE HEADER AND DATA=62
;:IN RHCS1
MOV #-10,@RHWC ;:10 WORDS
MOV #WRFROM,@RHBA ;:BUS ADDRESS=WRFROM
MOV #10,@RH0ST ;:DESIRED TRACK=0 SECTOR=10
BIS #BAI,@RHCS2 ;:BUS ADDRESS INCREMENT INHIBIT
MOV #FMT22,@RHOF ;:FORMAT 16 BIT WORDS
CLR @RHCA ;:CYLINDER=0
MOV #1,@#MID ;:SECTOR IS SET TO 1 SO THAT
;:WE CAN GET OUT AT THE
;:MIDDLE OF AN OPERATION
;:LOOKING FOR SECTOR 10
;:SET DIAGNOSTIC MODE
MOV #DMD,@RHMR ;:GO TO RHCS1 WITH 52
BIS #GO,@RHCS1 ;GO TO RHCS1 WITH 52
JSR R1,@#SEARCH
MID: .WORD 0 ;SECTOR
MOV (SP)+,R1 ;:POP STACK INTO R1
MOV (SP)+,RO ;:POP STACK INTO RO
RTS PC

.SBTTL JAM CURRENT CYLINDER ROUTINE

8163
8164
8165
8166
8167
8168
8169
8170
8171
8172
8173
8174
8175
8176
8177
8178
8179
8180
8181
8182
8183
8184
8185
8186
8187
8188
8189
8190
8191
8192
8193
8194
8195
8196
8197
8198
8199
8200
8201

```

*****
*THIS SUBROUTINE WILL CHANGE THE CURRENT CYLINDER REGISTER
*THIS IS DONE BY GIVING A SEEK COMMAND THEN AN INIT
*WHICH WILL LOAD THE CURRENT CYLINDER WITH THE DESIRED CYLINDER VALUE
*
*CALL IS:
*   JSR   RO, @MAKECYL
*   XC    ; DESIRED VALUE OF CURRENT CYLINDER
*****
MAKECYL:
MOV     R5, -(SP)      ; PUSH R5 ON STACK
MOV     RO, @PCJSR    ; PC OF JSR+4
SUB     #4, @PCJSR    ; SAVE PC OF JSR
MOV     (RO)+, R5     ; GETTING READY TO FILL DESIRED CYLINDER
MOV     R5, @RHCA     ; FILL DESIRED CYLINDER REGISTER
CLR     @RHST        ; MAKE SURE DESIRED SECTOR TRACK IS NOT ILLEGAL
MOV     @SEECOM, @RHCSI ; FILL SEEK COMMAND
MOV     #DMD, @RHMR   ; SET DIAGNOSTIC MODE
BIS     #GO, @RHCSI   ; GO TO SEEK
NOP     ; ALLOW TIME FOR SEEK TO HANG UP
NOP     ; ALLOW TIME FOR SEEK TO HANG UP
NOP     ; ALLOW TIME FOR SEEK TO HANG UP
NOP     ; ALLOW TIME FOR SEEK TO HANG UP
JSR     PC, @CLDISK   ; GIVE INIT
MOV     @RHCC, @SBDDAT ; TEST DATA
CMP     R5, @SBDDAT   ; COMPARE CURRENT CYLINDER
BEQ     IS           ; BRANCH IF GOOD
MOV     R5, @SGDDAT   ; GOOD VALUE OF RHCC
MOV     @RHCC, @REGADR ; FAILING REGISTER ADDRESS
ERROR   30           ; CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER
                        ; REGISTER AFTER A SEEK AND AN INIT
IS:
MOV     (SP)+, R5     ; POP STACK INTO R5
RTS     RO

```

```

050246
050246 010546
050250 010037 015134
050254 162737 000004 015134
050262 012005
050264 010577 144502
050270 005077 144470
050274 013777 015202 144456
050302 012777 000001 144470
050310 052777 000001 144442
050316 000240
050320 000240
050322 000240
050324 000240
050326 004737 045674
050332 017737 144460 001126
050340 020537 001126
050344 001406
050346 010537 001124
050352 013737 015016 045440
050360 104030
050362
050362 012605
050364 000200

```

.SBTTL ECC GENERATION AND COMPARISON ROUTINE

```

*****
*THIS SUBROUTINE GENERATES AND TESTS ECC
*CALL JSR PC,ECTEST
*****

```

8202
8203
8204
8205
8206
8207
8208
8209
8210
8211
8212
8213
8214
8215
8216
8217
8218
8219
8220
8221
8222
8223
8224
8225
8226
8227
8228
8229
8230
8231
8232
8233
8234
8235
8236
8237
8238
8239
8240
8241
8242
8243
8244
8245
8246
8247
8248
8249
8250
8251
8252
8253
8254
8255

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001
100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

PIE1 =100000
PIE2 =40000
PIE3 =20000
PIE4 =10000
PIE5 =4000
PIE6 =2000
PIE7 =1000
PIE8 =400
PIE9 =200
PIE10 =100
PIE11 =40
PIE12 =20
PIE13 =10
PIE14 =4
PIE15 =2
PIE16 =1
PIE17 =100000
PIE18 =40000
PIE19 =20000
PIE20 =10000
PIE21 =4000
PIE22 =2000
PIE23 =1000
PIE24 =400
PIE25 =200
PIE26 =100
PIE27 =40
PIE28 =20
PIE29 =10
PIE30 =4
PIE31 =2
PIE32 =1

050366 000000

050370 000000

050372 000000

```

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

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

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

```

```

8256 050374 000000      TSECCG: 0          ;IF =177777 GENERATE AND TEST ECC FOR THIS BIT
8257                                     ;IF =0 DO NOT GENERATE AND TEST ECC FOR THIS BIT
8258
8259 050376 113713      NCODE: 38859.     ;N-CODE WORD
8260 050400 000000      NCOUNT: 0       ;TEMPORARY N CODE
8261 050402 000000      POSITI: 0        ;POSITION REGISTER
8262 050404 010041      HARDER: 4129.    ;HARD ERROR COUNT
8263                                     ;TRUE COUNT IS 4128 BUT AS COMPARES ARE
8264                                     ;DONE ONE STAGE LATER SO 4129
8265 050406 000000      DATENV: 0        ;DATA ENVELOPE FOR TYPE OUT
8266                                     ;MAX FOR WRITE IS 4096
8267                                     ;MAX FOR READ IS 4128
8268 050410 000000      ZCODE: 0         ;LEADING ZEROS ENVELOPE FOR TYPE OUT
8269                                     ;THIS IS SHUT OFF WHEN POSITION COUNTER
8270                                     ;IN ENABLED
8271                                     ;MAX COUNT IS 38859
8272
8273
8274
8275 050412 000000      HADTMP: 0        ;TEMPORARY HARD ERROR COUNT
8276 050414 000000      P3: 0
8277 050416 000000      P12: 0
8278 050420 000000      P22: 0
8279 050422 000000      P24: 0
8280
8281
8282
8283
8284
8285 050424      ECTEST:
8286 050424 010046      MOV      R0, -(SP)      ;; PUSH R0 ON STACK
8287 050426 010146      MOV      R1, -(SP)      ;; PUSH R1 ON STACK
8288 050430 010246      MOV      R2, -(SP)      ;; PUSH R2 ON STACK
8289 050432 010346      MOV      R3, -(SP)      ;; PUSH R3 ON STACK
8290 050434 010446      MOV      R4, -(SP)      ;; PUSH R4 ON STACK
8291 050436 010546      MOV      R5, -(SP)      ;; PUSH R5 ON STACK
8292 050440 013701 050370  MOV      @#GECC1, R1      ;ECC1 WORD
8293 050444 013702 050372  MOV      @#GECC2, R2      ;ECC2 WORD
8294 050450 005737 050366  TST      @#ECDATA        ;IS CURRENT BIT A ONE
8295 050454 001406      BEQ      2$             ;BRANCH IF CURRENT DATA D=0
8296
8297                                     ;IF CARRY IS NOT ZERO THEN D=1
8298                                     ;INVERT X32 TO GIVE R0
8299
8300 050456 010103      1$:  MOV      R1, R3
8301 050460 052703 177776  BIS      #1CPIE32, R3
8302 050464 005103      COM      R3
8303 050466 010300      MOV      R3, R0
8304 050470 000404      BR      3$
8305
8306                                     ;IF CARRY IS ZERO THEN D=0
8307                                     ;X32 BECOMES R0
8308 050472 010103      2$:  MOV      R1, R3
8309 050474 042703 177776  BIC      #1CPIE32, R3

```

```

8310 050500 010300          MOV    R3,R0
8311
8312 050502 000241          3$:   CLC
8313 050504 006000          ROR    R0
8314 050506 006000          ROR    R0
8315 050510 005700          TST    R0
8316 050512 001462          BEQ    10$          ;BRANCH IF R0=0
8317
8318
8319 050514 010203          MOV    R2,R3
8320 050516 052703 137777    BIS    #↑CPIE2,R3
8321 050522 005103          COM    R3
8322 050524 010337 050414    MOV    R3,@#P3
8323 050530 006237 050414    ASR    @#P3
8324
8325
8326
8327
8328 050534 010203          MOV    R2,R3
8329 050536 052703 177737    BIS    #↑CPIE11,R3
8330 050542 005103          COM    R3
8331 050544 010337 050416    MOV    R3,@#P12
8332 050550 006237 050416    ASR    @#P12
8333
8334
8335
8336 050554 010103          MOV    R1,R3
8337 050556 052703 173777    BIS    #↑CPIE21,R3
8338 050562 005103          COM    R3
8339 050564 010337 050420    MOV    R3,@#P22
8340 050570 006237 050420    ASR    @#P22
8341
8342
8343
8344 050574 010103          MOV    R1,R3
8345 050576 052703 176777    BIS    #↑CPIE23,R3
8346 050602 005103          COM    R3
8347 050604 010337 050422    MOV    R3,@#P24
8348 050610 006237 050422    ASR    @#P24
8349
8350
8351
8352
8353
8354
8355
8356
8357
8358 050614 006002          ;NOW THAT R0 FOR POSITION 1
8359 050616 006001          ;      P3 FOR POSITION 3
8360 050620 053700 050414    ;      P12 FOR POSITION 12
8361 050624 053700 050416    ;      P22 FOR POSITION 22
8362 050630 042702 120020    ;      P24 FOR POSITION 24
8363 050634 050002          ;ARE KNOWN THE ROTATE WILL BE DONE AND
                                ;THESE BITS JAMED IN
                                ROR    R2
                                ROR    R1
                                BIS    @#P3,R0
                                BIS    @#P12,R0
                                BIC    #PIE1!PIE3!PIE12,R2
                                BIS    R0,R2

```

```

8364
8365 050636 005000          CLR      RO
8366 050640 053700 050420  BIS      @#P22,RO
8367 050644 053700 050422  BIS      @#P24,RO
8368 050650 042701 002400  BIC      #PIE22!PIE24,R1
8369 050654 050001          BIS      RO,R1
8370 050656 000404          BR       12$
8371
8372                          ;THE PROGRAM COMES HERE IF RO=0
8373                          ;SO AFTER ROTATE RO GETS PUT INTO POSITION 1
8374
8375 050660 006002          10$:  ROR      R2
8376 050662 006001          ROR      R1
8377 050664 042702 100000  BIC      #PIE1,R2
8378 050670 010137 050370  12$:  MOV      R1,@#GECC1      ;SAVE ECC1
8379 050674 010237 050372  MOV      R2,@#GECC2      ;SAVE ECC2
8380 050700 005737 050374  TST      @#TSECCG        ;IS HARDWARE TO BE CHECKED
8381                          ;IF =1777777 TEST HARDWARE
8382                          ;IF = 0 DO NOT TEST HARDWARE
8383 050704 001432          BEQ      14$      ;BRANCH IF HARDWARE NOT TO BE CHECKED
8384
8385
8386                          ;*CHECK HARDWARE
8387 050706 032777 000400 130224 BIT      #SW8,@SWR        ;IS SWITCH 8 SET
8388 050714 001005          BNE      15$      ;BRANCH IF SW8 IS SET
8389 050716 032777 000100 130214 BIT      #SW6,@SWR        ;IS SWITCH 6 SET
8390 050724 001401          BEQ      15$      ;BRANCH IF SW6 IS NOT SET
8391 050726 000421          BR       14$      ;IF SWITCH 8 IS NOT SET AND
8392                          ;SWITCH 6 IS SET THEN
8393                          ;DO NOT DO COMPARES
8394 050730 010146          15$:  MOV      R1,-(SP)      ;GOOD PATTERN REGISTER
8395 050732 042716 174000  BIC      #174000,(SP)    ;GET ONLY PATTERN BITS
8396 050736 022677 144050  CMP      (SP)+,@RHEC2    ;COMPARE PATTERN REGISTER
8397 050742 001404          BEQ      13$      ;BRANCH IF GOOD
8398                          ;TO SAVE TIME
8399 050744 004737 045374  JSR      PC,@#PUTREG     ;SAVE REGISTERS
8400 050750 104035          ERROR   35      ;PATTERN REGISTER IN 11 BITS IN ERROR
8401 050752 000407          BR       14$      ;BRANCH OUT
8402 050754 023777 050402 144026 13$:  CMP      @#POSIT1,@RHEC1 ;COMPARE POSITION REGISTER
8403 050762 001403          BEQ      14$      ;BRANCH IF GOOD
8404                          ;TO SAVE TIME
8405 050764 004737 045374  JSR      PC,@#PUTREG     ;SAVE REGISTERS
8406 050770 104035          ERROR   35      ;POSITION REGISTER IN ERROR
8407                          ;"DATA ENVELOP" GIVES NUMBER OF CLOCK
8408                          ;PULSES FROM BEGINING OF COMMAND
8409                          ;THAT IS THE CLOCKS IN THE R/W DATA FIELD ENVELOPE
8410
8411                          ;IN A WRITE THERE ARE 10000 OCTAL CLOCKS
8412                          ;IN A READ THERE ARE 10040 OCTAL CLOCKS
8413
8414
8415                          ;"N-CODE ZEROS" GIVE THE NUMBER OF CLOCKS
8416                          ;GIVEN FOR THE LEADING ZEROS FIELD
8417                          ;MAX COUNT IS 113713 OCTAL

```

```

8418
8419
8420
8421
8422
8423
8424
8425 050772
8426 050772 012605
8427 050774 012604
8428 050776 012603
8429 051000 012602
8430 051002 012601
8431 051004 012600
8432 051006 000207

```

145:

```

MOV (SP)+,R5
MOV (SP)+,R4
MOV (SP)+,R3
MOV (SP)+,R2
MOV (SP)+,R1
MOV (SP)+,R0
RTS PC

```

```

; "GOOD POSITION" GIVES NUMBER OF CLOCKS
; GIVEN AFTER LEADING ZEROS WHICH IS FOR THE DATA
; FIELD
; MAX COUNT IS 10040 OR 10041 OCTAL

```

```

;; POP STACK INTO R5
;; POP STACK INTO R4
;; POP STACK INTO R3
;; POP STACK INTO R2
;; POP STACK INTO R1
;; POP STACK INTO R0

```

.SBTTL ECC GENERATION CONTROL ROUTINE

8433
8434
8435
8436
8437
8438
8439
8440
8441
8442
8443
8444
8445
8446
8447
8448
8449
8450
8451
8452
8453
8454
8455
8456
8457
8458
8459
8460
8461
8462
8463
8464
8465
8466
8467
8468
8469
8470
8471
8472
8473
8474
8475
8476
8477
8478
8479
8480
8481
8482
8483
8484
8485
8486

```

;*****
;THIS SUBROUTINE WILL CONTROL THE ECC GENERATION ROUTINE
;FOR ERROR CORRECTION PROCESS
;CALL JSR, PC,@#ECORR
;* XP ;EXPECTED POSITION REGISTER WHEN CORRECTION IS COMPLETE
;*****
051010 000000 ERPOS: 0 ;POSITION REG. WHEN CORRECTION IS COMPLETE

051012 010037 015134 ECORR: MOV R0,@#PCJSR ;SAVE PC OF JSR + 4
051016 162737 000004 015134 SUB #4,@#PCJSR ;SAVE PC OF JSR
051024 012037 051010 MOV (R0)+,@#ERPOS ;GET POSITION REG. WHEN CORRECTION IS COMPLETE
051030 010146 MOV R1,-(SP) ;PUSH R1 ON STACK
051032 013701 015000 MOV @#RHMR,R1 ;MAINTENANCE REGISTER
051036 012711 000001 MOV #DMD,@R1 ;SET DIAGNOSTIC MODE BIT
051042 005037 050366 CLR @#ECDATA ;ECC DATA IS ZERO

051046 005737 050402 1$: TST @#POSITI ;IS SOFTWARE POSITION NON ZERO
051052 001007 BNE 2$ ;BRANCH IF N-CODE S COMPLETE
051054 005337 050400 DEC @#NCOUNT ;DECREMENT N-CODE
051060 001001 BNE 6$ ;BRANCH IF N-CODE IS NOT COMPLETE
051062 000403 BR 2$ ;BRANCH AS N-CODE IS COMPLETE
051064 005237 050410 6$: INC @#ZCODE ;INCREMENT CLOCKS GIVEN FOR LEADING ZEROS
051070 000420 BR 3$ ;BRANCH AS N-CODE IS NOT COMPLETE
;GO TO GIVE CLOCK AND TEST ECC
051072 005237 050402 2$: INC @#POSITI ;INCREMENT SOFTWARE POSITION
051076 023737 051010 050402 CMP @#ERPOS,@#POSITI ;HAVE ENOUGH CLOCKS BEEN GIVEN TO DETECT ERROR
051104 103012 BHS 3$ ;BRANCH IF MORE CLOCKS TO BE GIVEN
051106 023737 050412 050402 CMP @#HADTMP,@#POSITI ;HAVE ENOUGH CLOCKS BEEN GIVEN FOR HARD ERROR
;THAT IS HAVE 4128 MORE CLOCKS BEEN GIVEN
051114 001415 BEQ 5$ ;BRANCH IF YES
051116 032711 000400 BIT #ZER,@R1 ;CHECK ZERO DETECT BIT IN RHMR
051122 001016 BNE 4$ ;BRANCH IS ZER SET
;TO SAVE TIME
051124 004737 045374 JSR PC,@#PUTREG ;SAVE REGISTERS
051130 104034 ERROR 34 ;ZERO DETECT BIT NOT HIGH
;WHEN 21 BITS IN ECC 32 BIT REGISTER IS 0

051132 052711 000002 3$: BIS #MCLK,@R1 ;SET CLOCK
051136 042711 000002 BIC #MCLK,@R1 ;CLEAR CLOCK
051142 004737 050424 JSR PC,@#ECTEST ;GO TO GENERATE AND TEST ECC
051146 000737 BR 1$ ;CONTINUE

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

```

8487	051150	052711	000002	55:	BIS	#MCLK,R1	:SET CLOCK
8488	051154	042711	000002		BIC	#MCLK,R1	:CLEAR CLOCK
8489				45:			
8490	051160						
8491	051160	012601			MOV	.SP)+.R1	::POP STACK INTO R1
8492	051162	000200			RTS	RD	

.SBTTL SOFTWARE DISK DATA ECC GEN. ROUTINE

8493
8494
8495
8496
8497
8498
8499
8500
8501
8502
8503
8504
8505
8506
8507
8508
8509
8510
8511
8512
8513
8514
8515
8516
8517
8518
8519
8520
8521
8522
8523
8524
8525
8526
8527
8528
8529
8530
8531
8532
8533

051164
051164 010046
051166 010146
051170 010246
051172 010346
051174 010446
051176 010546
051200 005037 050402
051204 005037 050370
051210 005037 050372
051214 012701 054256
051220 012702 000400
051224 012703 000020
051230 012104
051232 006004
051234 103004
051236 012737 177777 050366
051244 000402
051246 005037 050366
051252 004737 050424
051256 005303
051260 001364
051262 005302
051264 001357
051266 013737 050370 055256
051274 013737 050372 055260
051302 012605
051304 012604
051306 012603
051310 012602
051312 012601
051314 012600
051316 000207

*THIS SUBROUTINE GENERATES THE ECC FOR WHAT IS ON DISK AND INSERTS THEM
*ON LOCATIONS "DISK+1000" AND "DISK+1002"

FILLEC:

MOV R0,-(SP) ;: PUSH R0 ON STACK
MOV R1,-(SP) ;: PUSH R1 ON STACK
MOV R2,-(SP) ;: PUSH R2 ON STACK
MOV R3,-(SP) ;: PUSH R3 ON STACK
MOV R4,-(SP) ;: PUSH R4 ON STACK
MOV R5,-(SP) ;: PUSH R5 ON STACK
CLR @#POSITI ;: CLEAR POSITION
CLR @#GECC1 ;: CLEAR GECC1
CLR @#GECC2 ;: CLEAR
MOV #DISK,R1 ;: POINTER TO DATA FOR ECC GENERATION
MOV #256,R2 ;: COUNTER FOR NUMBER OF DATA WORDS
95: MOV #16,R3 ;: COUNTER FOR NUMBER OF BITS PER WORD
MOV (R1)+,R4 ;: DATA IN R4
105: ROR R4 ;: GET ONE DATA BIT IN CARRY
BCC 115 ;: BRANCH IF DATA BIT IS ZERO
MOV #-1,@#ECDATA ;: ECC DATA BIT IS A ONE
BR 125 ;: BRANCH TO GENERATE ECC
115: CLR @#ECDATA ;: ECC DATA BIT IS A ZERO
125: JSR PC,@#ECTEST ;: GO TO GENERATE ECC
DEC R3 ;: DECREMENT BIT COUNT
BNE 105 ;: BRANCH IF 16 BITS NOT DONE
DEC R2 ;: DECREMENT WORD COUNT
BNE 95 ;: BRANCH IF 256 WORDS NOT DONE
MOV @#GECC1,@#DISK+(256.*2); INSERT ECC1 ON DISK
MOV @#GECC2,@#DISK+(257.*2); INSERT ECC2 ON DISK
MOV (SP)+,R5 ;: POP STACK INTO R5
MOV (SP)+,R4 ;: POP STACK INTO R4
MOV (SP)+,R3 ;: POP STACK INTO R3
MOV (SP)+,R2 ;: POP STACK INTO R2
MOV (SP)+,R1 ;: POP STACK INTO R1
MOV (SP)+,R0 ;: POP STACK INTO R0
RTS PC

```

8534
8535
8536
8537
8538
8539
8540
8541
8542 051320
8543 051320 104400 051326
8544 051324 000425
8545
8546 051400
8547 051400 013746 014760
8548 051404 104401
8549 051406 104400 051414
8550 051412 000425
8551
8552 051466
8553 051466 004737 060036
8554 051472 104411
8555 051474 012700 014750
8556 051500 012701 000024
8557 051504 042710 177700
8558 051510 051620
8559 051512 005301
8560 051514 001373
8561 051516 104400 051524
8562 051522 000417
8563
8564 051562
8565 051562 013746 014746
8566 051566 104401
8567 051570 104400 051576
8568 051574 000437
8569
8570 051674
8571 051674 104411
8572 051676 012637 014746
8573 051702 104400 051710
8574 051706 000421
8575
8576 051752
8577 051752 104400 051760
8578 051756 000416
8579
8580 052014
8581 052014 013746 014760
8582 052020 104401
8583 052022 104400 052030
8584 052026 000416
8585
8586 052064
8587 052064 013746 014746

```

.SBTTL RH BASE ADDRESS CHANGE ROUTINE

```

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

```

```

BASECH:
        TYPE      65$          ;;TYPE ASCIZ STRING
        BR        64$          ;;GET OVER THE ASCIZ
        65$: .ASCIZ <15><12>/PRESENT BASE ADDRESS OF REGISTERS IS /
        64$:
        MOV      2#RHCS1,-(SP) ;GET READY TO TYPE OLD BASE
        TYPOC
        TYPE      67$          ;;TYPE ASCIZ STRING
        BR        66$          ;;GET OVER THE ASCIZ
        67$: .ASCIZ <15><12>/TYPE NEW BASE ADDRESS FOLLOWED BY 'CR' /
        66$:
        JSR      PC,2#STKINT  ;INITIALIZE THE TTY KEYBOARD
        RDOCT
        MOV      #RHDB,RO      ;GET STARTING ADDRESS OF REGISTERS
        MOV      #20,R1        ;NUMBER OF REGISTERS
        1$: BIC   #1C77,(RO)    ;CLEAR OLD BASE ADDRESS
        BIS     (SP),(RO)+     ;SET NEW BASE
        DEC     R1             ;COUNT
        BNE     1$            ;BRANCH IF 20 NOT DONE
        TYPE      69$          ;;TYPE ASCIZ STRING
        BR        68$          ;;GET OVER THE ASCIZ
        69$: .ASCIZ <15><12>/PRESENT VECTOR ADDRESS IS /
        68$:
        MOV      2#RPVEC,-(SP) ;GET READY TO TYPE OLD VECTOR ADDRESS
        TYPOC
        TYPE      71$          ;;TYPE ASCIZ STRING
        BR        70$          ;;GET OVER THE ASCIZ
        71$: .ASCIZ <15><12>/TYPE NEW VECTOR ADDRESS OR RETYPE OLD ONE FOLLOWED BY "CR" /
        70$:
        RDOCT
        MOV      (SP)+,2#RPVEC ;SETUP VECTOR ADDRESS
        TYPE      73$          ;;TYPE ASCIZ STRING
        BR        72$          ;;GET OVER THE ASCIZ
        73$: .ASCIZ <15><12>/RESTART PROGRAM FROM 200 OR 210/
        72$:
        TYPE      75$          ;;TYPE ASCIZ STRING
        BR        74$          ;;GET OVER THE ASCIZ
        75$: .ASCIZ <15><12>/NEW BASE WILL REMAIN - /
        74$:
        MOV      2#RHCS1,-(SP)
        TYPOC
        TYPE      77$          ;;TYPE ASCIZ STRING
        BR        76$          ;;GET OVER THE ASCIZ
        77$: .ASCIZ <15><12>/NEW VECTOR WILL REMAIN - /
        76$:
        MOV      2#RPVEC,-(SP)

```

E04

MNDEC-11-DZRJH-A, RP04/5/6 DSKLS CONTROLLER TST-PT 2
 DZRJHA.SUB RH BASE ADDRESS CHANGE ROUTINE

MACY11 27(655) 30-MAR-76 20:59 PAGE 203

SEQ 0249

```

9588 052070 104401          TYP0C
9589 052072 104400 052100  TYPE      79$          ;;TYPE ASCIZ STRING
8590 052076 000402          BR        78$          ;;GET OVER THE ASCIZ
8591          ;;79$: .ASCIZ <15><12> / /
8592          78$:
9593 052104 104400 052112  TYPE      81$          ;;TYPE ASCIZ STRING
8594 052110 000416          BR        80$          ;;GET OVER THE ASCIZ
8595          ;;81$: .ASCIZ <15><12>/UNTIL PROGRAM IS RELOADED/
8596 052146          80$:
8597 052146 000000          HALT
8598
8599
8600
8601
8602
8603
8604
8605
8606
8607 052150 000000
8608 052152 004737 045674
8609 052156 013712 052150
8610 052162 005714
8611 052164 032712 010000
8612 052170 001401
8613 052172 000773
8614 052174 000772
  
```

```

;*THIS IS A LITTLE ROUTINE THAT TESTS NED BIT 11 IN RHCS2
;*THIS LOOPS HERE FOR EVER
;*TO BE USED ONLY IF DRIVES PRESENT LOOKING AT NED DOES NOT AGREE
;*WITH WHAT IS REALY THERE
ERUNIT: 0 ;UNIT UNDER MANUAL TEST
ERSTART: JSR PC, @#CLDISK ;SET GENERAL REG.
          MOV @#ERUNIT, @R2 ;SELECT UNIT
1$:      TST @R4 ;TEST RHER1
          BIT #NED, @R2 ;TEST NED
          BEQ 2$ ;BRANCH IF GOOD
          BR 1$ ;NED NOT SET
2$:      BR 1$ ;NED SET
  
```

8615
8616
8617
8618
8619
8620
8621
8622
8623
8624
8625
8626
8627
8628
8629
8630
8631
8632
8633
8634
8635
8636
8637
8638
8639
8640
8641
8642
8643
8644
8645
8646
8647
8648
8649
8650
8651
8652
8653
8654
8655
8656
8657
8658
8659
8660
8661
8662
8663
8664
8665
8666
8667
8668

```
.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
*SECTR= 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,COMPD
*
*
*
*IN A READ HEADER AND DATA COMMAND FILL THE FOLLOWING
*CYL= WITH CYLINDER TO BE FOUND ON DISK
*SECTR= 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
*SECTR= 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
```

GO4

MNDEC-11-DZRJH-A, RPO4/S & DSKLS CONTROLLER TST-PT 2
DZRJHA.SUB DISK SIMULATION

MAY11 27(655) 30-MAR-63 20:59 PAGE 205

SEQ 0251

9669

:*

8670
8671
8672
8673
8674
8675
8676
8677
8678
8679
8680
8681
8682
8683
8684
8685
8686
8687
8688
8689
8690
8691
8692
8693
8694
8695
8696
8697
8698
8699
8700
8701
8702
8703
8704
8705
8706
8707
8708
8709
8710
8711
8712
8713
8714
8715
8716
8717
8718
8719
8720
8721
8722
8723

```
*****  
: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  
;* R0HEAD ;READS THE SECTOR HEADER  
;* WRDATA ;WRITES THE SECTOR DATA (WRITE OPERATION)  
;* REDATA ;READS THE SECTOR DATA (READ OPERATION)
```

```
052176 000000  
052200 011637 015134  
052204 162737 000004 015134  
052212 010046  
052214 010146  
052216 010246  
052220 010346  
052222 010446  
052224 010546  
  
052226 012777 000001 142544  
052234 052777 000004 142536  
052242 042777 000004 142530  
052250 052777 000001 142502  
  
052256 012737 000113 052176  
052264 005337 052176  
052270 001375  
  
052272 013746 052342  
052276 042716 177740  
052302 012637 052312
```

```
RUNCTR: .WORD 0  
COMMD: MOV (SP), @#PCJSR ;SAVE PC OF JSR + 4  
SUB #4, @#PCJSR ;SAVE PC OF JSR  
MOV R0, -(SP) ;PUSH R0 ON STACK  
MOV R1, -(SP) ;PUSH R1 ON STACK  
MOV R2, -(SP) ;PUSH R2 ON STACK  
MOV R3, -(SP) ;PUSH R3 ON STACK  
MOV R4, -(SP) ;PUSH R4 ON STACK  
MOV R5, -(SP) ;PUSH R5 ON STACK  
  
MOV #DMD, @RHMR ;SET DIAGNOSTIC MODE  
BIS #MINX, @RHMR ;SET DIAGNOSTIC INDEX  
BIC #MINX, @RHMR ;CLEAR DIAGNOSTIC INDEX  
BIS #GO, @RHCSI ;ISSUE 'GO' BIT & STALL 'TILL 'RUN'  
;FUNCTION CODE WAS ISSUED BY THE TEST  
RUNWAT: MOV #75, @#RUNCTR ;LOAD STALL COUNT = APPROX. 450US FOR 11.50 CPU  
IS: DEC @#RUNCTR ;COUNT DOWN ONE  
BNE IS ;CONTINUE UNTIL = 0  
  
MOV SECOTR, -(SP) ;GET DESIRED SECTOR/TRACK  
BIC #177740, (SP) ;MAKE ONLY SECTOR  
MOV (SP)+, @#TRK ;SAVE SECTOR
```



```

8778 052360 005737 015126      TST      @#ERFLGS      ;WERE ANY ERRORS DETECTED ?
8779 052364 001017              BNE      OUT          ;IF YES, EXIT ----->
8780
8781 052366 005737 052350      TST      @#X          ;IS IT A DATA WRITE OPERATION ?
8782 052372 001410      BEQ      DAREAD      ;NO... THEN DO A DATA READ
8783 052374 005737 052454      TST      @#NOSYNC    ;IS THIS FORCED HEADER ERROR COMMAND ?
8784                          ;IF YES NOSYNC=-1 THEN WRITE OR READ
8785                          ;IS SHUT OFF, SO BRANCH OUT
8786                          ;IF NOSYNC=0 THEN CONTINUE
8787 052400 001017              BNE      CJT          ;EXIT IF SET ----->
8788
8789 052402 004137 053726      JSR      R1,@#WRDATA ;WRITE DATA <----->
8790 052406 000000      NOWORD: .WORD      0 ;NO OF WORDS TO BE WRITTEN
8791 052410 000000      Y:      .WORD      0 ;
8792 052412 000404      BR      OUT          ;EXIT ----->
8793
8794 052414 004137 056662      DAREAD: JSR      R1,@#REDATA ;READ DATA <----->
8795 052420 000000      DAWORD: .WORD      0 ;NO OF WORDS TO BE READ
8796 052422 000000      .WORD      0 ;
8797
8798                          OUT:
8799 052424 012605      MOV      (SP)+,R5    ;: POP STACK INTO R5
8800 052426 012604      MOV      (SP)+,R4    ;: POP STACK INTO R4
8801 052430 012603      MOV      (SP)+,R3    ;: POP STACK INTO R3
8802 052432 012602      MOV      (SP)+,R2    ;: POP STACK INTO R2
8803 052434 012601      MOV      (SP)+,R1    ;: POP STACK INTO R1
8804 052436 012600      MOV      (SP)+,R0    ;: POP STACK INTO R0
8805 052440 000207      RTS      PC          ;EXIT
8806

```


8807
8808
8809
8810
8811
8812
8813
8814
8815
8816
8817
8818
8819
8820
8821
8822
8823
8824
8825
8826
8827
8828
8829
8830
8831
8832
8833
8834
8835
8836
8837
8838

052442 014400
052444 000000
052446 000000
052450 000000
052452 000000

;*****

;*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

RSYNC: 14400
RCYL: 0
RSETR: 0
RKEY1: 0
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
;*****

```

8839      ;*****
8840      ;*READ DISK HEADER
8841      ;*****
8842
8843
8844
8845
8846
8847 052454 000000      NOSYNC: 0      ;FORCED HEADER ERROR = -1
8848      ;NORMAL = 0
8849 052456 000000      TY: 0      ;ERROR TYPE NO.
8850 052450 000000      ERWORD: 0      ;ERROR WORD NO.
8851
8852
8853
8854
8855 052462 012137 052444      RDHEAD: MOV (R1)+, @#RCYL      ;STORE CYLINDER ADDRESS
8856 052466 012137 052446      MOV (R1)+, @#RSETR      ;STORE SECTOR AND TRACK ADDRESS
8857 052472 012137 052450      MOV (R1)+, @#RKEY1      ;STORE KEY1
8858 052476 012137 052452      MOV (R1)+, @#RKEY2      ;STORE KEY2
8859 052502 012137 053252      MOV (R1)+, @#COMPA      ;STORE COMPARE OR NOT
8860 052506 010146      MOV R1, -(SP)      ;PUSH R1 ON STACK
8861
8862 052510 013700 015000      MOV @#RHRM, R0      ;R0 CONTAINS MAINTANENCE REG.
8863 052514 012705 000002      MOV #2, R5      ;R5 IS A COUNTER FOR WORDS
8864 052520 012710 000001      MOV #DMD, @R0      ;SET DIAG. MODE
8865 052524 052710 000010      BIS #MSTCK, @R0      ;SET SECTOR CLOCK FOR FIRST WORD
8866 052530 052710 000002      BIS #MCLK, @R0      ;SET MAINT. CLOCK FOR FIRST WORD
8867 052534 042710 000012      BIC #MSTCK!MCLK, @R0 ;RESET THEM
8868
8869 052540 000404      BR 2$      ;DON'T GIVE SECTOR CLOCK FIRST TIME
8870 052542 012710 000013      1$: MOV #MSTCK!MCLK!DMD, @R0 ;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
8871 052546 042710 000012      BIC #MSTCK!MCLK, @R0 ;RESET SECTOR & MAINT. CLOCK
8872
8873 052552 012702 000007      2$: MOV #7, R2      ;LOAD BYTE COUNTER
8874 052556 052710 000002      3$: BIS #MCLK, @R0      ;SET MAINT. CLOCK
8875 052562 042710 000002      BIC #MCLK, @R0      ;RESET IT
8876 052566 005302      DEC R2      ;BYTE COUNTER
8877 052570 001372      BNE 3$      ;CONTINUE IF BYTE NOT COMPLETE
8878 052572 005305      DEC R5      ;WORD COUNTER
8879 052574 001362      BNE 1$      ;CONTINUE IF WORD NOT COMPLETE
8880
8881 052576 012702 000022      4$: MOV #18, R2      ;LOAD NO OF WORDS OF ZEROS
8882 052602 005037 053250      CLR @#WORD      ;DO 0'S
8883 052606 004737 053254      JSR PC, @#READ      ;READ 0'S <----->
8884 052612 005302      DEC R2      ;COUNT DOWN WORDS
8885 052614 001372      BNE 4$      ;CONTINUE
8886
8887 052616 013737 052442 053250      MOV @#RSYNC, @#WORD ;SYNC. WORD
8888 052624 004737 053254      JSR PC, @#READ      ;READ IT <----->
8889 052630 032710 001000      BIT #DTSY, @R0      ;SYNC. BYTE DETECTED?
8890 052634 001012      BNE 5$      ;CONTINUE IF SYNC DETECTED
8891 052636 012737 000001 052460      MOV #1, @#ERWORD      ;ERROR WORD NO
8892 052644 013737 052442 001124      MOV @#RSYNC, @#$GDDAT; SYNC WORD

```

```

8893 052652 012737 104002 052352      MOV      #104002, @#SSYN ;INSERT "ERROR 2" IN SSYN
8894 052660 000571                    BR       13$           ;BRANCH OUT ----->
8895
8896 052662 013737 052444 053250 5$:  MOV      @#RCYL, @#WORD ;SETUP CYLINDER
8897 052670 004737 053254                    JSR      PC, @#READ    ;READ IT <----->
8898 052674 013737 052446 053250      MOV      @#RSETR, @#WORD ;SETUP SECTOR/TRACK
8899 052702 004737 053254                    JSR      PC, @#READ    ;READ THEM <----->
8900 052706 013737 052450 053250      MOV      @#RKEY1, @#WORD ;SETUP KEY1
8901 052714 004737 053254                    JSR      PC, @#READ    ;READ IT <----->
8902 052720 013737 052452 053250      MOV      @#RKEY2, @#WORD ;SETUP KEY2
8903 052726 004737 053254                    JSR      PC, @#READ    ;READ IT <----->
8904 052732 013737 054240 053250      MOV      @#WCRC, @#WORD ;SETUP CRC
8905 052740 004737 053254                    JSR      PC, @#READ    ;READ IT <----->
8906
8907 052744 005737 015146                    TST      @#TESDTE      ;IS THIS A DRIVE TIMING ERROR ?
8908 052750 001135                    BNE      13$           ;BRANCH OUT IF YES ----->
8909 052752 005737 053252                    TST      @#COMPA      ;IS THIS A READ OR WRITE COMMAND ?
8910 052756 001472                    BEQ      11$           ;DO READ IF = 0
8911
8912                                ;*CONTINUE WITH DIAGNOSTIC WRITE COMMAND
8913
8914 052760 012705 054242                    MOV      #HEGAP, R5   ;POINTER FOR HEADER GAP
8915 052764 012702 000005                    MOV      #5, R2       ;NO OF WORDS OF ZEROS
8916 052770 012737 000006 052460 6$:  MOV      #6, @#ERWORD ;ERROR WORD NO SET
8917 052776 004737 053506                    JSR      PC, @#WRITE   ;FOR HEADER GAP
8918 053002 005737 053504                    TST      @#WWORD      ;TEST WRITTEN WORD
8919 053006 001413                    BEQ      7$           ;CONTINUE IF GOOD, THAT IS = 0
8920 053010 160237 052460                    SUB      R2, @#ERWORD ;WORD NO IN ERROR
8921 053014 005037 001124                    CLR      @#$GDDAT     ;GOOD WORD SHOULD BE 0
8922 053020 013737 053504 001126      MOV      @#WWORD, $BDDAT ;BAD DATA
8923 053026 012737 104003 052354      MOV      #104003, @#HEDGAP ;"ERROR 2" GOES IN HEDGAP
8924 053034 000503                    BR       13$           ;BRANCH OUT ----->
8925
8926 053036 013725 053504                    7$:  MOV      @#WWORD, (R5)+ ;SAVE HEADER GAP
8927 053042 005302                    DEC      R2
8928 053044 001351                    BNE      6$
8929 053046 004737 053506                    JSR      PC, @#WRITE   ;WRITE HEADER (DATA) GAP SYNC
8930 053052 023737 052442 053504      CMP      @#RSYNC, @#WWORD
8931 053060 001426                    BEQ      10$
8932 053062 005737 052454                    TST      @#NOSYNC     ;IS THIS FORCED HEADER ERROR COMMAND ?
8933                                ;IF YES NOSYNC=-1 THEN WRITE OR READ
8934                                ;IS SHUT OFF SO BRANCH OUT
8935                                ;IF NO NOSYNC=0 THEN CONTINUE
8936 053066 001406                    BEQ      14$           ;PRINT IT IF TRUE ERROR
8937
8938 053070 005737 053504                    TST      @#WWORD      ;IS IT = 0 ?
8939 053074 001420                    BEQ      10$           ;CONTINUE IF GOOD
8940 053076 005037 001124                    CLR      @#$GDDAT     ;IT SHOULD BE ZERO
8941 053102 000403                    BR       15$           ;BRANCH TO TYPE ERROR
8942 053104 013737 052442 001124 14$:  MOV      @#RSYNC, @#$GDDAT ;GOOD DATA
8943 053112 013737 053504 001126 15$:  MOV      @#WWORD, @#$BDDAT ;BAD DATA
8944 053120 012737 000006 052460      MOV      #6, @#ERWORD
8945 053126 012737 104003 052356      MOV      #104003, @#HEDSYN
8946 053134 000443                    BR       13$           ;BRANCH OUT ----->

```

```

8947
8948 053136 013725 053504      10$:  MOV    @#WORD,(R5)+ ;SAVE DATA SYNC.
8949 053142 000440                BR      13$           ;EXIT ----->
8950
8951                ;*READ COMMAND START FROM HERE
8952
8953 053144 012702 000005      11$:  MOV    #5,R2
8954 053150 005037 053250      12$:  CLR    WORD
8955 053154 004737 053254      JSR    PC,@#READ ;READ HEADER GAP <----->
8956 053160 005302                DEC    R2          ;ARE 5 HEADER GAP ZEROS COMPLETE ?
8957 053162 001372                BNE    12$        ;IF NOT CONTINUE
8958 053164 013737 052442 053250  MOV    @#RSYNC,@#WORD ;SYNC WORD
8959 053172 004737 053254      JSR    PC,@#READ ;READ HEADER (DATA) SYNC)
8960 053176 005737 052454      TST    @#NOSYNC   ;FORCED SYNC ERROR ?
8961 053202 001404                BEQ    16$        ;IF NOT ERROR COMMAND CONTINUE
8962 053204 032710 001000      BIT    #DTSY,@R0 ;SYNC. DETECTED
8963 053210 001415                BEQ    13$        ;IF ZERO BRANCH OUT ----->
8964 053212 000403                BR     17$        ;IF NOT ZERO BRANCH TO ERROR
8965
8966 053214 032710 001000      16$:  BIT    #DTSY, @R0 ;SYNC. DETECTED ?
8967 053220 001011                BNE    13$        ;EXIT IF YES ----->
8968 053222 012737 000006 052460  17$:  MOV    #6,@#ERWORD ;ERROR WORD NO.
8969 053230 013737 052442 001124  MOV    @#RSYNC,@#$GDDAT;SYNC WORD
8970 053236 012737 104002 052356  MOV    #104002,@#HEDSYN;MOVE "ERROR 2"
8971 053244
8972 053244 012601                13$:  MOV    (SP)+,R1 ;POP STACK INTO R1
8973 053246 000201                RTS    R1          ;EXIT ----->
8974
8975
8976
8977
8978
8979
8980
8981

```

```

09982          ::*****
09983          ::*READ ONE WORD IN "WORD"
09984          ::*****
09985
09986
09987
09988
09989 053250 000000          WORD: 0
09990 053252 000000          COMPA: C
09991
09992
09993
09994
09995          READ:
09996 053254          MOV      R2,-(SP)          ;;PUSH R2 ON STACK
09997 053254 010246          MOV      #2,R5          ;;WORD COUNTER
09998 053256 012705 000002          MOV      #DMD,DR0          ;;SET DIAG. MODE
09999 053262 012710 000001          MOV      @#WORD          ;;CHECKING IF THERE IS A ONE
09999 053266 006037 053250          ROR      @#WORD          ;;IF NO ONE BRANCH
90000 053272 103002          BCC      1$          ;;SET BIT 4 IF DATA HAS ONE
90001 053274 052710 000020          BIS      #MRD,DR0          ;;BYTE COUNTER
90002 053300 012702 000007          1$: MOV      #7,R2          ;;SET CLOCK DATA IF ANY SECTOR
90003 053304 052710 000012          BIS      #MSTCK!MCLK,DR0  ;;IS THIS BIT TO GENERATE AND TEST ECC ?
90004 053310 005737 050374          TST      @#TSECCG          ;;BRANCH IF NO
90005 053314 001411          BEQ      6$          ;;IS DATA BIT A ONE ?
90006 053316 032710 000020          BIT      #MRD,DR0          ;;BRANCH IF DATA BIT IS 0
90007 053322 001404          BEQ      5$          ;;ECC DATA BIT IS A ONE
90008 053324 012737 177777 050366          MOV      #-1,@#ECDATA          ;;BRANCH
90009 053332 000402          BR       6$          ;;ECC DATA BIT IS A 0
90010 053334 005037 050366          5$: CLR      @#ECDATA          ;;KEEP ONLY DIAG. MODE
90011 053340 012746 000001          6$: MOV      #DMD,-(SP)          ;;CHECKING IF THERE IS A ONE
90012 053344 006037 053250          ROR      @#WORD          ;;IF NO ONE BRANCH
90013 053350 103002          BCC      2$          ;;KEEP DATA AND DIAG. MODE
90014 053352 012716 000021          MOV      #MRD!DMD,(SP)      ;;PUT IN DATA,RESET CLOCK, SECTOR
90015 053356 012610          2$: MOV      (SP)+,DR0          ;;IS ECC TO BE GENERATED FOR THIS BIT
90016 053360 005737 050374          TST      @#TSECCG          ;;BRANCH IF NO
90017 053364 001404          BEQ      3$          ;;NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
90018 053366 005237 050406          INC      @#DATENV          ;;GO TO GENERATE AND TEST ECC
90019 053372 004737 050424          JSR      PC,@#ECTEST          ;;SET CLOCK
90020 053376 052710 000002          3$: BIS      #MCLK,DR0          ;;IS THIS BIT TO GENERATE ECC
90021 053402 005737 050374          TST      @#TSECCG          ;;BRANCH IF NO
90022 053406 001411          BEQ      8$          ;;IS DATA BIT A ONE
90023 053410 032710 000020          BIT      #MRD,DR0          ;;BRANCH IF DATA BIT IS = 0
90024 053414 001404          BEQ      7$          ;;ECC DATA BIT IS A ONE
90025 053416 012737 177777 050366          MOV      #-1,@#ECDATA          ;;BRANCH
90026 053424 000402          BR       8$          ;;ECC DATA BIT IS = 0
90027 053426 005037 050366          7$: CLR      @#ECDATA          ;;KEEP DIAG. MODE
90028 053432 012746 000001          8$: MOV      #DMD,-(SP)          ;;CHECKING IF THERE IS A ONE
90029 053436 006037 053250          ROR      @#WORD          ;;BRANCH IF NO ONE
90030 053442 103002          BCC      4$          ;;KEEP DIAG. MODE AND DATA
90031 053444 012716 000021          MOV      #MRD!DMD,(SP)      ;;SET DATA, DIAG. MODE, CLEAR CLOCK
90032 053450 012610          4$: MOV      (SP)+,DR0          ;;IS THIS BIT TO GENERATE ECC
90033 053452 005737 050374          TST      @#TSECCG          ;;BRANCH IF NO
90034 053456 001404          BEQ      9$          ;;NUMBER OF CLOCKS FOR DATA ENVELOPE
90035 053460 005237 050406          INC      @#DATENV

```

9036	053464	004737	050424		JSR	PC, DNECTEST	;GO TO GENERATE AND TEST ECC
9037							
9038	053470	005302		9S:	DEC	R2	:BYTE COUNTER
9039	053472	001341			BNE	3\$:CONTINUE IF ONE BYTE NOT COMPLETE
9040	053474	005305			DEC	R5	:WORD COUNTER
9041	053476	001300			BNE	4\$:CONTINUE IF ONE WORD NOT COMPLETE
9042	053500	012602			MOV	(SP)+,R2	:POP STACK INTO R2
9043	053502	000207			RTS	PC	:EXIT -----
9044							
9045							
9046							
9047							
9048							
9049							
9050							

```

9050
9051
9052
9053
9054
9055
9056
9057
9058
9059 053504 000000          WWORD: 0
9060
9061
9062
9063
9064 053506          WRITE:
9065 053506 010046          MOV R0, -(SP)          ;: PUSH R0 ON STACK
9066 053510 010246          MOV R2, -(SP)          ;: PUSH R2 ON STACK
9067 053512 010346          MOV R3, -(SP)          ;: PUSH R3 ON STACK
9068 053514 010546          MOV R5, -(SP)          ;: PUSH R5 ON STACK
9069 053516 012705 000002          MOV #2, R5          ;: WORD COUNTER
9070 053522 012710 000001          MOV #1, JRO          ;: SET DIAG. MODE
9071 053526 012702 000007          1$: MOV #7, R2          ;: BYTE COUNTER
9072 053532 012710 000013          MOV #MSTCK!MCLK!DMD, JRO ;: SET SECTOR & MANT. CLOCKS
9073 053536 032710 000040          BIT #MWR, JRO          ;: CHECK WRITEBIT IN MAINT. REG.
9074 053542 001406          BEQ 2$                ;: BRANCH IF ZERO
9075 053544 012737 177777 050366          MOV #-1, J#ECDATA          ;: ECC DATA BIT IS A ONE
9076 053552 000261          SEC                ;: SET CARRY
9077 053554 006003          ROR R3                ;: MOVE 1 FORWARD
9078 053556 000404          BR 3$                ;:
9079 053560 005037 050366          2$: CLR J#ECDATA          ;: ECC DATA BIT IS = 0
9080 053564 000241          CLC                ;: CLEAR CARRY
9081 053566 006003          ROR R3                ;: MOVE 0 FOR WWORD
9082 053570 012710 000001          3$: MOV #DMD, JRO          ;: CLEAR SECTOR AND CLOCK
9083 053574 005737 050374          TST J#TSECCG          ;: IS THIS BIT TO GENERATE ECC ?
9084 053600 001404          BEQ 4$                ;: BRANCH IF NO
9085 053602 005237 050406          INC J#DATENV          ;: NUMBER OF CLOCKS FOR DATA ENVELOPE
9086 053606 004737 050424          JSR PC, J#ECTEST          ;: GO TO GENERATE AND TEST ECC (<----->)
9087
9088 053612 052710 000002          4$: BIS #MCLK, JRO          ;: SET CLOCK
9089 053616 032710 000040          BIT #MWR, JRO          ;: CHECK WRITE BIT IN MAINT. REG.
9090 053622 001406          BEQ 5$                ;: BRANCH IF ZERO
9091 053624 012737 177777 050366          MOV #-1, J#ECDATA          ;: ECC DATA BIT IS A ONE
9092 053632 000261          SEC                ;: SET CARRY
9093 053634 006003          ROR R3                ;: MOVE 1 FOR WWORD
9094 053636 000404          BR 6$                ;:
9095 053640 005037 050366          5$: CLR J#ECDATA          ;: ECC DATA BIT IS ZERO
9096 053644 000241          CLC                ;: CLEAR CARRY
9097 053646 006003          ROR R3                ;: MOVE 0 FOR WWORD
9098 053650 012710 000001          6$: MOV #DMD, JRO          ;: CLEAR CLOCK
9099 053654 005737 050374          TST J#TSECCG          ;: IS THIS BIT TO GENERATE ECC ?
9100 053660 001404          BEQ 7$                ;: BRANCH IF NO
9101 053662 005237 050406          INC J#DATENV          ;: NUMBER OF CLOCKS FOR DATA ENVELOPE
9102 053666 004737 050424          JSR PC, J#ECTEST          ;: GO TO GENERATE AND TEST ECC (<----->)
9103

```

```

9104 053672 005302          75:  DEC      R2      ;COUNT FOR BYTE END
9105 053674 001346          BNE     4$     ;IF NOT BYTE END CONTINUE
9106 053676 005305          DEC     R5     ;COUNT FOR WORD END
9107 053700 001312          BNE     1$     ;IF NOT WORD END CONTINUE
9108 053702 010337 053504  MOV     R3, @#WORD ;STORE THE WORD
9109 053706 012605          MOV     (SP)+, R5 ;POP STACK INTO R5
9110 053710 012603          MOV     (SP)+, R3 ;POP STACK INTO R3
9111 053712 012602          MOV     (SP)+, R2 ;POP STACK INTO R2
9112 053714 012600          MOV     (SP)+, R0 ;POP STACK INTO R0
9113 053716 000207          RTS     PC     ;EXIT ----->
9114
9115
9116
9117
9118
9119
9120
9121
9122
9123
9124
9125
9126
9127
9128
9129
9130
9131
9132
9133
9134
9135
9136
9137
9138
9139
9140
9141
9142
9143
9144
9145
9146
9147
9148
9149
9150
9151
9152
9153
9154
9155
9156
9157
9158
9159
9160
9161
9162
9163
9164
9165
9166
9167
9168
9169
9170
9171
9172
9173
9174
9175
9176
9177
9178
9179
9180
9181
9182
9183
9184
9185
9186
9187
9188
9189
9190
9191
9192
9193
9194
9195
9196
9197
9198
9199
9200

```



```

9120          ;:*****
9121          ;WRITE DATA HOUSEKEEPING ROUTINE
9122          ;:*****
9123
9124
9125
9126
9127
9128
9129
9130 053720 000000          COUNTD: 0
9131 053722 000400          FORMAT: 256.
9132 053724 000000          ZWORDS: 0
9133 053726          WRDATA:
9134 053726 011137 053720          MOV      (R1), @#COUNTD ;STORE NO. OF WORDS TO BE WRITTEN
9135 053732 012102          MOV      (R1)+, R2 ;SAME IN R2
9136 053734 012137 053252          MOV      (R1)+, @#COMPA ;COMPARE OR NOT
9137 053740 010046          MOV      RO, -(SP) ;PUSH RO ON STACK
9138 053742 010146          MOV      R1, -(SP) ;PUSH R1 ON STACK
9139 053744 010246          MOV      R2, -(SP) ;PUSH R2 ON STACK
9140 053746 010346          MOV      R3, -(SP) ;PUSH R3 ON STACK
9141 053750 010446          MOV      R4, -(SP) ;PUSH R4 ON STACK
9142 053752 012701 000016          MOV      #14, R1 ;NO. OF TOLERANCE GAP WORDS
9143 053756 012703 055264          MOV      #TOLGAP, R3 ;START OF TOLERANCE GAP TABLE
9144 053762 012723 177777          15: MOV      #-1, (R3)+ ;MAKE IT 177777
9145 053766 005301          DEC      R1 ;IS 14 COMPLETED ?
9146 053770 001374          BNE     15 ;IF NOT, CONTINUE
9147
9148 053772 013700 015000          MOV      @#RHMR, RO ;RO CONTAINS MAINTANENCE REG.
9149 053776 013746 053722          MOV      @#FORMAT, -(SP)
9150 054002 163716 053720          SUB     @#COUNTD, (SP)
9151 054006 011637 053724          MOV      (SP), @#ZWORDS ;NO. OF ZERO WORDS TO BE WRITTEN
9152 054012 012604          MOV      (SP)+, R4
9153 054014 005737 015144          TST     @#TSECC ;IS THIS AN ECC TEST
9154 054020 001403          BEQ     75 ;BRANCH IF NO
9155 054022 012737 177777 050374          MOV      #-1, @#TSECCG ;THESE BITS ARE TO GENERATE ECC
9156 054030 012703 054256          75: MOV      #DISK, R3 ;SIMULATED DISK AREA
9157 054034 004737 053506          25: JSR     PC, @#WRITE ;WRITE ON SIMULATED DISK
9158 054040 013723 053504          MOV      @#WORD, (R3)+ ;STORE ON SIMULATED DISK
9159 054044 005302          DEC      R2
9160 054046 001372          BNE     25
9161 054050 005704          TST     R4 ;ANY ZEROS TO BE WRITTEN ?
9162 054052 001406          BEQ     45 ;BRANCH IF NONE TO BE WRITTEN
9163
9164 054054 004737 053506          35: JSR     PC, @#WRITE ;WRITE ZEROS ON SIMULATED DISK
9165 054060 013723 053504          MOV      @#WORD, (R3)+ ;STORE THEM
9166 054064 005304          DEC      R4
9167 054066 001372          BNE     35
9168 054070 005037 050374          45: CLR     @#TSECCG ;NO MORE ECC TO BE GENERATED
9169 054074 012701 000002          MOV      #2, R1
9170 054100 004737 053506          55: JSR     PC, @#WRITE ;WRITE ECC1 AND ECC2 ON SIMULATED DISK
9171 054104 013723 053504          MOV      @#WORD, (R3)+ ;STORE IN WEEC1 AND WEEC2
9172 054110 005301          DEC      R1
9173 054112 001372          BNE     55

```

9174	054114	004737	053506		JSR	PC,@WRITE	:WRITE DATA GAP
9175	054120	013723	053504		MOV	@WORD,(R3)+	:STORE IT
9176	054124	012701	000016		MOV	#14,R1	
9177	054130	004737	053506	6\$:	JSR	PC,@WRITE	:WRITE TOLERANCE GAP ZEROS
9178	054134	013723	053504		MOV	@WORD,(R3)+	:STORE THEM
9179	054140	005301			DEC	R1	
9180	054142	001372			SNE	6\$	
9181	054144	012604			MOV	(SP)+,R4	::POP STACK INTO R4
9182	054146	012603			MOV	(SP)+,R3	::POP STACK INTO R3
9183	054150	012602			MOV	(SP)+,R2	::POP STACK INTO R2
9184	054152	012601			MOV	(SP)+,R1	::POP STACK INTO R1
9185	054154	012600			MOV	(SP)+,R0	::POP STACK INTO R0
9186	054156	000201			RTS	R1	:EXIT -----
9187							
9188							
9189							
9190							
9191							
9192							
9193							
9194							

9195
9196
9197
9199
9199
9200
9201
9202
9203
9204
9205
9206
9207
9208
9209
9210
9211
9212
9213
9214
9215
9216
9217
9218
9219
9220
9221
9222
9223
9224

054160 000023
054226 000001
054230 000004
054240 000001
054242 000005
054254 000001
054256 000400
055256 000001
055260 000001
055262 000001
055264 000016

:*THIS IS THE SIMULATED DISK
:*ONLY ONE SECTOR OF SPACE IS ALLOCATED
:*****
:*****

SECGAP: .BLKW 19.
WSSYNC: .BLKW 1
HEADER: .BLKW 4
WCRC: .BLKW 1
HEGAP: .BLKW 5
HDWSYN: .BLKW 1
SILOTB:
DISK: .BLKW 256.
WECC1: .BLKW 1
WECC2: .BLKW 1
DTAGAP: .BLKW 1
TOLGAP: .BLKW 14.

:SECTOR GAP 38 BYTES OF 0
:SECTOR GAP 1 BYTE OF 0 ONE SYNC BYTE
:HEADER = CYL, SECTOR/TRACK, KEY1, KEY2
:CRC
:HEADER GAP 10 BYTES OF 0
:HEADER GAP 1 BYTE OF 0 ONE SYNC. BYTE
:(ALSO USED IN SILO TEST AS SILO TABLE)
:DATA SPACE
:ECC1
:ECC2
:DATA GAP 2 BYTES OF 0
:TOLERANCE GAP 28 BYTES OF 0


```

9333                                     ;WORD NO 3 = KEY1
9334                                     ;WORD NO 4 = KEY2
9335
9336                                     ;BAD WORD IS WHAT IS GOING ON
9337                                     ;DISK
9338
9339 055516 000240          ERCRC: NOP
9340
9341                                     ; IF "ERROR 6" INSERTED BY
9342                                     ; WRHEAD SUBROUTINE, THEN CRC WRITTEN
9343                                     ; ON DISK IS IN ERROR.
9344
9345                                     ;GOOD DATA IS WHAT SHOULD BE ON DISK
9346                                     ;BAD DATA IS WHAT IS GOING ON DISK.
9347
9348                                     ;WORD NO IS 5
9349 055520 000240          ERHDGP: NOP
9350
9351                                     ; IF "ERROR 6" INSERTED BY
9352                                     ; WRHEAD SUBROUTINE, THEN HEADER
9353                                     ; GAP GOING ON DISK IS WRONG.
9354
9355                                     ;WORD NO GIVES WHICH OF
9356                                     ; THE HEADER GAP WORDS
9357                                     ; ARE WRONG. FOR EXAMPLE:
9358
9359                                     ;WORD NO 1 = FIRST HEADER
9360                                     ;          GAP WORD
9361                                     ;BAD WORD IS WHAT IS GOING ON DISK
9362 055522 000240          HDESYN: NOP
9363
9364                                     ; IF "ERROR 6" INSERTED BY
9365                                     ; WRHEAD SUBROUTINE, THEN LAST
9366                                     ; HEADER GAP BYTE OR HEADER
9367                                     ; SYNC BYTE GOING ON DISK IS WRONG.
9368
9369                                     ;WORD NO = 5
9370
9371                                     ;BAD DATA IS WHAT IS GOING
9372                                     ; ON DISK, RIGHT BYTE IS HEADER
9373                                     ; GAP 0'S BYTE, LEFT BYTE IS HEADER
9374                                     ; GAP SYNC.
9375
9376 055524 005737 015126          TST    @#ERFLGS      ;ARE ANY ERRORS DETECTED ?
9377 055530 001004                BNE    FOUT          ;IF YES BRANCH
9378 055532 004137 053726          JSR    R1,@#WRDATA ;WRITE THE DATA
9379
9380 055536 000000          FNWORD: .WORD 0      ;FORMAT COMMAND NO. OF DATA
9381 055540 000000          .WORD 0
9382
9383 055542          FOUT:
9384 055542 012605          MOV    (SP)+,R5      ;;POP STACK INTO R5
9385 055544 012604          MOV    (SP)+,R4      ;;POP STACK INTO R4
9386 055546 012603          MOV    (SP)+,R3      ;;POP STACK INTO R3

```

9387	055550	012602	MOV	(SP)+,R2	::POP STACK INTO R2
9388	055552	012601	MOV	(SP)+,R1	::POP STACK INTO R1
9389	055554	012600	MOV	(SP)+,R0	::POP STACK INTO R0
9390	055556	0C0207	RTS	PC	::EXIT

```

9391
9392
9393
9394
9395
9396
9397
9398
9399
9400
9401
9402
9403
9404
9405
9406
9407
9408
9409 055560 000000
9410 055562 000000
9411 055564 000000
9412 055566 000000
9413 055570 000000
9414
9415
9416 055572 012137 055560
9417 055576 012137 055562
9418 055602 012137 055564
9419 055606 012137 055566
9420 055612 012137 055570
9421 055616 010146
9422
9423 055620 012701 054160
9424 055624 013700 015000
9425 055630 012710 000001
9426 055634 012705 000002
9427 055640 052710 000010
9428 055644 012710 000013
9429
9430 055650 032710 000040
9431 055654 001403
9432 055656 000261
9433 055660 006003
9434 055662 000402
9435 055664 000241
9436 055666 006003
9437 055670 012710 000001
9438 055674 012702 000007
9439 055700 052710 000002
9440 055704 032710 000040
9441 055710 001403
9442
9443 055712 000261
9444 055714 006003

```

```

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

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

SCYL: 0
SSECTR: 0
SKEY1: 0
SKEY2: 0
SCRC: 0

WRHEAD: MOV (R1)+, @#SCYL
MOV (R1)+, @#SSECTR
MOV (R1)+, @#SKEY1
MOV (R1)+, @#SKEY2
MOV (R1)+, @#SCRC
MOV R1, -(SP) ;; PUSH R1 ON STACK

MOV #SECGAP, R1 ;; SIMULATED DISK INDICATOR
MOV @#RHMR, R0 ;; R0 NOW HAS MAINT. REG. ADDR.
MOV #DMD, @R0 ;; SET DIAG. MODE
MOV #2, R5 ;; WORD COUNTER
BIS #MSTCK, @R0 ;; SET SECTOR FOR FIRST BYTE
1$: MOV #MSTCK!MCLK!DMD, @R0 ;; SET SECTOR, CLOCK, DIAG.
MODE, RESET INDEX
;; CHECK WRITE BIT IN MAINT. REG.
BIT #MWR, @R0
BEQ 2$
SEC ;; SET CARRY
ROR R3 ;; MOVE ONE FORWARD
BR 3$
2$: CLC ;; CLEAR CARRY
ROR R3 ;; MOVE ZERO FORWARD
3$: MOV #DMD, @R0 ;; CLEAR CLOCK, SECTOR
MOV #7, R2 ;; BYTE COUNTER
4$: BIS #MCLK, @R0 ;; SET BIT CLOCK
BIT #MWR, @R0 ;; CHECK WRITE BIT IN MAINT.REG.
BEQ 5$
5$: SEC ;; SET CARRY
ROR R3 ;; MOVE ONE FORWARD

```



```

9445 055716 000402          BR      6$
9446 055720 000241          5$:    CLC
9447 055722 006003          ROR      R3
9448 055724 012710 000001  6$:    MOV      #DMD, @R0
9449 055730 005302          DEC      R2
9450 055732 001362          BNE     4$
9451 055734 005305          DEC      R5
9452 055736 001342          BNE     1$
9453 055740 010321          MOV      R3, (R1)+
9454 055742 005703          TST     R3
9455 055744 001414          BEQ     7$
9456
9457 055746 012737 000001 052460  MOV     #1, @#ERWORD
9458 055754 005037 001124          CLR     @#$GDDAT
9459 055760 010337 001126          MOV     R3, @#$BDDAT
9460 055764 012737 104006 055510  MOV     #104006, @#SEGP
9461 055772 000137 056400          JMP     @#17$ ; BRANCH OUT ----->
9462
9463 055776 012702 000022  7$:    MOV     #18., R2 ; COUNT NO. OF SECTOR GAP
9464 056002 012737 000024 052460 10$:   MOV     #20., @#ERWORD ; COUNT TO GIVE ERROR WORD
9465 056010 004737 053506          JSR     PC, @#WRITE ; WRITE SECTOR GAP
9466 056014 013721 053504          MOV     @#WWORD, (R1)+ ; STORE SECTOR GAP WORD
9467 056020 001413          BEQ     11$
9468 056022 160237 052460          SUB     R2, @#ERWORD ; IF NOT GET ERROR WORD NO.
9469 056026 005037 001124          CLR     @#$GDDAT ; GOOD WORD
9470 056032 013737 053504 001126          MOV     @#WWORD, @#$BDDAT ; BAD WORD
9471 056040 012737 104006 055510  MOV     #104006, @#SEGP ; STORE "ERROR 6" IN SEGP
9472 056046 000554          BR      17$ ; BRANCH OUT
9473 056050 005302          11$:   DEC     R2 ; HAVE 18 WORDS OF ZEROS BEEN WRITTEN ?
9474 056052 001353          BNE     10$ ; IF NOT CONTINUE
9475
9476          ; AT THIS POINT THE SECTOR FOUND FLOP SHOULD
9477          ; BE HIGH, SO THAT THE HEADER SYNC BYTE CAN BE GIVEN.
9478
9479          ; HOWEVER, IN THE DRIVE TIMING ERROR TEST THE REST OF THE ROUTINE
9480          ; IS ABORTED
9481
9482 056054 005737 015146          TST     @#TESDTE ; IS THIS A DRIVE TIMING ERROR TEST ?
9483 056060 001147          BNE     17$ ; BRANCH OUT IF YES ----->
9484
9485 056062 004737 053506          JSR     PC, @#WRITE ; WRITE ONE SECTOR GAP 0 BYTE
9486          ; AND ONE SYNC. BYTE = 230
9487 056066 013711 053504          MOV     @#WWORD, (R1) ; SAVE 0 BYTE AND SYNC BYTE
9488 056072 023721 052442          CMP     @#RSYNC, (R1)+ ; IF SYNC. BYTE RIGHT
9489 056076 001414          BEQ     12$ ; IF YES CONTINUE OPERATION
9490 056100 012737 000024 052460  MOV     #20., @#ERWORD ; IF NOT GET READY FOR ERROR PRINT
9491
9492 056106 013737 052442 001124          MOV     @#RSYNC, @#$GDDAT ; GOOD WORD
9493 056114 014137 001126          MOV     -(R1), @#$BDDAT ; BAD WORD
9494 056120 012737 104006 055512  MOV     #104006, @#FSYNER ; INSERT "ERROR 6" IN FSYNER
9495 056126 000524          BR      17$ ; BRANCH OUT ----->
9496
9497 056130 012702 000004          12$:   MOV     #4, R2 ; FOUR HEADER WORDS
9498 056134 012703 055560          MOV     #SCYL, R3 ; POINTER FOR HEADER TABLE

```

9499	056140	012737	000005	052460	13:	MOV	#5, @#ERWORD	; ERROR WORD NO SET
9500	056146	004737	053506			JSR	PC, @#WRITE	; WRITE 4 HEADER WORDS
9501	056152	013711	053504			MOV	@#WORD, (R1)	; STORE WRITTEN WORD
9502	056156	022321				CMP	(R3)+, (R1)+	; IS IT CORRECT ?
9503	056160	001412				BEG	14:	; IF GOOD CONTINUE OPERATION
9504								; IF NOT GET READY FOR ERROR PRINT
9505								
9506	056162	160237	052460			SUB	R2, @#ERWORD	; WORD NO
9507	056166	014337	001124			MOV	-(R3), @#SGDDAT	; GOOD DATA
9508	056172	014137	001126			MOV	-(R1), @#SBDDAT	; BAD DATA
9509	056176	012737	104006	055514		MOV	#104006, @#ERHEAD	; INSERT "ERROR 6"
9510	056204	000475				BR	17:	; BRANCH OUT -----
9511								
9512	056206	005302			14:	DEC	R2	; ARE 4 HEADER WORDS DONE ?
9513	056210	001353				BNE	13:	; IF NOT CONTINUE
9514	056212	004737	053506			JSR	PC, @#WRITE	; WRITE CRC
9515	056216	013711	053504			MOV	@#WORD, (R1)	; STORE CRC
9516	056222	022137	055506			CMP	(R1)+, @#GCRC	; COMPARE GOOD CRC
9517	056226	001414				BEG	20:	; IF GOOD CONTINUE OPERATION
9518								; IF NOT GET READY FOR ERROR PRINT
9519								
9520	056230	014137	001126			MOV	-(R1), @#SBDDATA	; BAD CRC WRITTEN
9521	056234	013737	055506	001124		MOV	@#GCRC, @#SGDDAT	; GOOD CRC
9522	056242	012737	000005	052460		MOV	#5, @#ERWORD	; ERROR WORD NO
9523	056250	012737	104006	055516		MOV	#104006, @#ERCRC	; INSERT ERROR 6
9524	056256	000450				BR	17:	; EXIT -----
9525								
9526	056260	012702	000005		20:	MOV	#5, R2	; NO OF HEADER GAP
9527	056264	012737	000006	052460	15:	MOV	#6, @#ERWORD	; ERROR WORD NO SET
9528	056272	004737	053506			JSR	PC, @#WRITE	; WRITE HEADER GAP
9529	056276	013721	053504			MOV	@#WORD, (R1)+	; STORE
9530	056302	001412				BEG	16:	; IF GOOD CONTINUE OPERATION
9531								; IF NOT GET READY FOR PRINT
9532								
9533	056304	160237	052460			SUB	R2, @#ERWORD	; ERROR WORD NO
9534	056310	005337	001124			CLR	@#SGDDAT	; GOOD DATA
9535	056314	014137	001126			MOV	-(R1), @#SBDDAT	; BAD DATA
9536	056320	012737	104006	055520		MOV	#104006, @#ERHDP	; STORE "ERROR 6"
9537	056326	000424				BR	17:	; BRANCH OUT -----
9538								
9539	056330	005302			16:	DEC	R2	; ARE 5 HEADER GAP ZEROS DONE ?
9540	056332	001354				BNE	15:	; IF NOT CONTINUE
9541	056334	004737	053506			JSR	PC, @#WRITE	
9542	056340	013711	053504			MOV	@#WORD, (R1)	
9543	056344	023721	052442			CMP	@#RSYNC, (R1)+	
9544	056350	001413				BEG	17:	; A-OK, EXIT -----
9545								
9546	056352	012737	000005	052460		MOV	#5, @#ERWORD	; IF NOT GET READY FOR ERROR PRINT
9547	056360	014137	001126			MOV	-(R1), @#SBDDAT	
9548	056364	013737	052442	001124		MOV	@#RSYNC, @#SGDDAT	
9549	056372	012737	104006	055522		MOV	#104006, @#HDESYN	
9550								
9551	056400				17:			
9552	056400	012601				MOV	(SP)+, R1	; POP STACK INTO R1

C06

MNDEC-11-DZRJH-A, RPO4-S 6 DSKLS CONTROLLER TST-PT 2
DZRJH.SUB DISK SIMULATION

MACY11 27(655) 30-MAR-76 20:59 PAGE 227

SEQ 0273

056402 000201
RIS RI
;EXIT ----->

056402 000201

RIS

RI

;EXIT ----->

95
96
97
98
99
00
01
02
03
04
05
06
07
08
09
10
11
12
13
14
15

:SEARCH SECTOR

* RO=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

056404 000000
056406 012137 056404
056412 010046
056414 010146
056416 010246
056420 010346
056422 010446
056424 010546
056426 013700 015000
056432 013703 056404
056436 012710 000001
056442 052710 000010
056446 042710 000010
056452 052710 000010
056456 042710 000010

SECTR: 0 ;SECTOR SEARCHED FOR
SEARCH: MOV (R1)+, 2#SECTR ;SAVE SECTOR SEARCHED FOR
MOV RO, -(SP) ;PUSH RO ON STACK
MOV R1, -(SP) ;PUSH R1 ON STACK
MOV R2, -(SP) ;PUSH R2 ON STACK
MOV R3, -(SP) ;PUSH R3 ON STACK
MOV R4, -(SP) ;PUSH R4 ON STACK
MOV R5, -(SP) ;PUSH R5 ON STACK
MOV 2#RHMR, RO ;NOW RO HAS MAINTENANCE REG. ADR.
MOV 2#SECTR, R3 ;LOAD SECTOR COUNTER
MOV #DMD, 2RO ;SET DIAGNOSTIC MODE
BIS #MSTCK, 2RO ;SET SECTOR CLOCK
BIC #MSTCK, 2RO ;CLEAR SECTOR CLOCK
BIS #MSTCK, 2RO ;SET SECTOR CLOCK
BIC #MSTCK, 2RO ;CLEAR SECTOR CLOCK
;THE ABOVE TWO SECTOR CLOCKS ARE GIVEN FOR
;RESETTING SECTOR PULSE
;IN CASE IT STARTS SET

```

9616 056462 052710 000014      BIS      #MINX!MSTCK, @R0 ;SET INDEX AND SECTOR CLOCK
9617 056466 042710 000014      BIC      #MINX!MSTCK, @R0 ;RESET INDEX AND SECTOR CLOCK
9618 056472 005703                TST      R3                ;TEST FOR SECTOR 0
9619 056474 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

```

```

9630
9631 056476 012702 000010      1$:     MOV      #8, R2          ;BYTE COUNTER
9632 056502 012705 000002      MOV      #2, R5          ;BYTES PER WORD
9633 056506 052710 000010      BIS      #MSTCK, @R0     ;SET SECTOR CLOCK
9634 056512 052710 000002      BIS      #MCLK, @R0     ;SET MAINT. CLOCK
9635 056516 000402                BR       3$              ;BRANCH TO CLEAR SECTOR AND CLOCK
9636 056520 052710 000012      2$:     BIS      #MSTCK!MCLK, @R0 ;SET BOTH CLOCKS
9637 056524 042710 000012      3$:     BIC      #MSTCK!MCLK, @R0 ;CLEAR BOTH CLOCKS
9638 056530 052710 000002      8$:     BIS      #MCLK, @R0   ;SET MAINT. CLOCK
9639 056534 042710 000002      BIC      #MCLK, @R0     ;CLEAR MAINT. CLOCK
9640 056540 005302                DEC      R2              ;BYTE COUNTER
9641 056542 001372                BNE      8$              ;CONTINUE IF BYTE NOT COMPLETE
9642
9643 056544 012702 000007      MOV      #7, R2          ;SETUP FOR SECOND BYTE
9644 056550 005305                DEC      R5              ;IS WORD COMPLETE?
9645 056552 001362                BNE      2$              ;CONTINUE IF NOT COMPLETE
9646
9647
9648
9649
9650
9651 056554 012701 000457                MOV      #303, R1        ;WORDS PER SECTOR COUNTER
9652 056560 012705 000002      4$:     MOV      #2, R5        ;BYTES PER WORD COUNTER
9653 056564 012702 000007      5$:     MOV      #7, R2        ;BIT COUNTER (MAINT. CLOCK COUNTER)
9654 056570 052710 000012      BIS      #MSTCK!MCLK, @R0 ;SET SECTOR CLOCK AND MAINT. CLOCK
9655 056574 042710 000012      BIC      #MSTCK!MCLK, @R0 ;CLEAR CLOCKS
9656 056600 052710 000002      6$:     BIS      #MCLK, @R0   ;SET MAINT. CLOCK
9657 056604 042710 000002      BIC      #MCLK, @R0     ;RESET MAINT. CLOCK
9658 056610 005302                DEC      R2              ;IS BYTE COMPLETE ?
9659 056612 001372                BNE      6$              ;CONTINUE IF NOT COMPLETE
9660
9661 056614 005305                DEC      R5              ;IS WORD COMPLETE ?
9662 056616 001362                BNE      5$              ;CONTINUE IF NOT
9663
9664 056620 005301                DEC      R1              ;IS SECTOR COMPLETE ?
9665 056622 001356                BNE      4$              ;CONTINUE IF NOT
9666
9667 056624 052710 000010      BIS      #MSTCK, @R0     ;SET SECTOR CLOCK 1 MORE TIME (FOR 509)
9668 056630 042710 000010      BIC      #MSTCK, @R0     ;CLEAR SECTOR CLOCK
9669 056634 005303                DEC      R3              ;IS REQUIRED NO OF SECTORS COMPLETE '

```

```

;NOW 303 WORDS ARE LEFT AND ALL ARE IDENTICAL

```

```

9670 056636 001317          BNE      1$          ;CONTINUE IF NOT
9671
9672 056640          7$:
9673 056640 012605          MOV      (SP)+,R5          ;;POP STACK INTO R5
9674 056642 012604          MOV      (SP)+,R4          ;;POP STACK INTO R4
9675 056644 012603          MOV      (SP)+,R3          ;;POP STACK INTO R3
9676 056646 012602          MOV      (SP)+,R2          ;;POP STACK INTO R2
9677 056650 012601          MOV      (SP)+,R1          ;;POP STACK INTO R1
9678 056652 012600          MOV      (SP)+,R0          ;;POP STACK INTO R0
9679 056654 000201          RTS      R1
9680
9681
9682          ::*****
9683          ;*READ ONE SECTOR OF DATA
9684          ;:*****
9685
9686 056656 000000          RNO:     0          ;NO. OF WORDS READ
9687 056660 000000          RCOM:    0          ;EXTRA STORAGE
9688
9689
9690
9691 056662 012137 056656          REDATA: MOV      (R1)+,RNO          ;SAVE NO. OF WORDS ONLY FOR INFORMATION
9692 056666 012137 056660          MOV      (R1)+,RCOM          ;EXTRA WORD ONLY FOR INFORMATION
9693 056672 010146          MOV      R1,-(SP)          ;;PUSH R1 ON STACK
9694 056674 005737 015144          TST      R1,TSECC          ;IS THIS AN ECC TEST
9695 056700 001403          BEQ      1$          ;BRANCH IF NO
9696 056702 012737 177777 050374          MOV      R1,-1,R2,TSECCG          ;THESE BITS ARE TO GENERATE ECC
9697 056710 012702 000402          1$:     MOV      R1,258,R2          ;256 WORDS PER SECTOR
9698          ;PLUS 2 ECC WORDS
9699 056714 012703 054256          MOV      R3,DISK          ;POINTE TO DISK SIMULATION
9700 056720 012337 053250          2$:     MOV      (R3)+,R2          ;READY TO READ CONTENTS
9701 056724 004737 053254          JSR      PC,R2          ;READ
9702 056730 005302          DEC      R2          ;IS 256 WORDS DONE?
9703 056732 001372          BNE      2$          ;IF NOT BRANCH
9704 056734 005737 015144          TST      R2,TSECC          ;IS THIS AN ECC TEST
9705 056740 001012          BNE      4$          ;BRANCH OUT IF YES
9706 056742 005037 050374          CLR      R2,TSECCG          ;NO MORE ECC BITS ARE TO BE GENERATED
9707 056746 012702 000017          MOV      R2,15          ;ONE DATA GAP, 14 TOLERANCE GAP
9708 056752 012337 053250          3$:     MOV      (R3)+,R2          ;READY TO READ CONTENTS OF WORD
9709 056756 004737 053254          JSR      PC,R2          ;READ
9710 056762 005302          DEC      R2          ;COUNT
9711 056764 001372          BNE      3$          ;BRANCH IF 14 NOT DONE
9712          4$:
9713 056766 012601          MOV      (SP)+,R1          ;;POP STACK INTO R1
9714 056770 000201          RTS      R1          ;RETURN

```


DZRJHA.SUB

9726
9727
9728
9729
9730
9731
9732
9733
9734
9735
9736
9737
9738
9739
9740
9741
9742
9743
9744
9745
9746
9747
9748
9749
9750
9751
9752
9753
9754
9755
9756
9757
9758
9759
9760
9761
9762
9763
9764
9765
9766
9767
9768
9769
9770
9771
9772
9773
9774
9775
9776
9777
9778
9779

.SBTTL
.SBTTL ***SYSMAC LIBRARY ROUTINES***
.SBTTL

.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

SSCOPE:
CKSWR ::TEST FOR CHANGE IN SOFT-SWR
CLR @#NOSYNC ::CLEAR FLAG FOR HEADER ERROR COMMANDS
CLR @#TSECC ::CLEAR FLAG FOR ECC TEST
WHEN =177777 IT IS AN ECC TEST
WHEN =0 IT IS NOT AN ECC TEST
CLR @#TSECCG ;EVEN IN AN ECC TEST EVERY CLOCK
IS NOT TO GENERATE ECC
IF =177777 GENERATE ECC
IF =0 DO NOT GENERATE ECC
CLR @#TESDTE ;DRIVE TIMING ERROR TEST
1S: BIT #BIT14,@SWR ::LOOP ON PRESENT TEST?
1S: BNE \$COVER ::YES IF SW14=1
*****START OF CODE FOR THE XOR TESTER*****
\$XTSTR: BR 6S
MOV @#ERRVEC, -(SP) ;IF RUNNING ON THE "XOR" TESTER CHANGE
MOV #55,@#ERRVEC ;THIS INSTRUCTION TO A "NOP" (NOP=240)
TST @#177060 ;SAVE THE CONTENTS OF THE ERROR VECTOR
MOV (SP)+,@#ERRVEC ;SET FOR TIMEOUT
BR \$SVLAD ;TIME OUT ON XOR?
5S: CMP (SP)+,(SP)+ ;RESTORE THE ERROR VECTOR
MOV (SP)+,@#ERRVEC ;GO TO THE NEXT TEST
BR 7S ;CLEAR THE STACK AFTER A TIME OUT
6S: *****END OF CODE FOR THE XOR TESTER*****
BIT #BIT08,@SWR ;LOOP ON SPEC. TEST?
BEQ 2S ;BR IF NO
CMPB @SWR,\$TSTNM ;ON THE RIGHT TEST? SWR<7:0>
BEQ \$COVER ;BR IF YES
2S: TSTB \$ERFLG ;HAS AN ERROR OCCURRED?
BEQ 3S ;BR IF NO
CMPB \$ERMAX,\$ERFLG ;MAX. ERRORS FOR THIS TEST OCCURRED?

057060
057060 104406
057062 005037 052454
057066 005037 015144
057072 005037 050374
057076 005037 015146
057102
057102 032777 040000 122030
057110 001111
057112 000416
057114 013746 000004
057120 012737 057140 000004
057126 005737 177060
057132 012637 000004
057136 000463
057140 022626
057142 012637 000004
057146 000423
057150
057150 032777 000400 121762
057156 001404
057160 127737 121754 001102
057166 001462
057170 105737 001103
057174 001421
057176 123737 001115 001103

9780	057204	101015			BHI	3\$:: BR IF NO
9781	057206	032777	001000	121724	BIT	#BIT09, @SWR	:: LOOP ON ERROR?
9782	057214	001404			BEQ	4\$:: BR IF NO
9783	057216	013737	001110	001106	7\$: MOV	\$LPERR, \$LPADR	:: SET LOOP ADDRESS TO LAST SCOPE
9784	057224	000443			BR	\$OVER	
9785	057226	105037	001103		4\$: CLRB	\$ERFLG	:: ZERO THE ERROR FLAG
9786	057232	005037	001212		CLR	\$TIMES	:: CLEAR THE NUMBER OF ITERATIONS TO MAKE
9787	057236	000415			BR	1\$:: ESCAPE TO THE NEXT TEST
9788	057240	032777	004000	121672	3\$: BIT	#BIT11, @SWR	:: INHIBIT ITERATIONS?
9789	057246	001011			BNE	1\$:: BR IF YES
9790	057250	005737	001100		TST	\$PASS	:: IF FIRST PASS OF PROGRAM
9791	057254	001406			BEQ	1\$:: INHIBIT ITERATIONS
9792	057256	005237	001104		INC	\$ICNT	:: INCREMENT ITERATION COUNT
9793	057262	023737	001212	001104	CMP	\$TIMES, \$ICNT	:: CHECK THE NUMBER OF ITERATIONS MADE
9794	057270	002021			BGE	\$OVER	:: BR IF MORE ITERATION REQUIRED
9795	057272	012737	000001	001104	1\$: MOV	#1, \$ICNT	:: REINITIALIZE THE ITERATION COUNTER
9796	057300	013737	057350	001212	MOV	\$MXCNT, \$TIMES	:: SET NUMBER OF ITERATIONS TO DO
9797	057306	105237	001102		\$SVLAD: INCB	\$TSTNM	:: COUNT TEST NUMBERS
9798	057312	011637	001106		MOV	(SP), \$LPADR	:: SAVE SCOPE LOOP ADDRESS
9799	057316	011637	001110		MOV	(SP), \$LPERR	:: SAVE ERROR LOOP ADDRESS
9800	057322	005037	001214		CLR	\$ESCAPE	:: CLEAR THE ESCAPE FROM ERROR ADDRESS
9801	057326	112737	000001	001115	MOVB	#1, \$ERMAX	:: ONLY ALLOW ONE(1) ERROR ON NEXT TEST
9802	057334	013777	001102	121600	\$OVER: MOV	\$TSTNM, @DISPLAY	:: DISPLAY TEST NUMBER
9803	057342	013716	001106		MOV	\$LPADR, (SP)	:: FUDGE RETURN ADDRESS
9804	057346	000002			RTI		:: FIXES PS
9805	057350	000004			\$MXCNT: 4		:: MAX. NUMBER OF ITERATIONS

9806
9807
9808
9809
9810
9811
9812
9813
9814
9815
9816
9817
9818
9819
9820
9821
9822
9823
9824
9825
9826
9827
9828
9829
9830
9831
9832
9833
9834
9835
9836
9837
9838
9839
9840
9841
9842
9843
9844
9845
9846
9847
9848
9849
9850
9851
9852
9853
9854
9855
9856
9857
9858
9859

057352
057352 010046
057354 010146
057356 010246
057360 010346
057362 010546
057364 012746 020200
057370 016605 000020
057374 100004
057376 005405
057400 112766 000055 000001
057406 005000
057410 012703 057566
057414 112723 000040
057420 005002
057422 016001 057556
057426 160105
057430 002402
057432 005202
057434 000774
057436 060105
057440 005702
057442 001002
057444 105716
057446 100407
057450 106316
057452 103003
057454 116663 000001 177777
057462 052702 000060
057466 052702 000040
057472 110223
057474 005720
057476 020027 000010
057502 002746
057504 003002
057506 010502
057510 000764
057512 105726
057514 100003
057516 116663 177777 177776
057524 105013
057526 012605

.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

\$TYPDS:

MOV R0,-(SP) ;;PUSH R0 ON STACK
MOV R1,-(SP) ;;PUSH R1 ON STACK
MOV R2,-(SP) ;;PUSH R2 ON STACK
MOV R3,-(SP) ;;PUSH R3 ON STACK
MOV R5,-(SP) ;;PUSH R5 ON STACK
MOV #20200,-(SP) ;;SET BLANK SWITCH AND SIGN
MOV 20(SP),R5 ;;GET THE INPUT NUMBER
BPL 1\$;;BR IF INPUT IS POS.
NEG R5 ;;MAKE THE BINARY NUMBER POS.
MOVB #'-,1(SP) ;;MAKE THE ASCII NUMBER NEG.
1\$: CLR R0 ;;ZERO THE CONSTANTS INDEX
MOV #SDBLK,R3 ;;SETUP THE OUTPUT POINTER
MOVB #' ,(R3)+ ;;SET THE FIRST CHARACTER TO A BLANK
2\$: CLR R2 ;;CLEAR THE BCD NUMBER
MOV \$DTBL(R0),R1 ;;GET THE CONSTANT
3\$: SUB R1,R5 ;;FORM THIS BCD DIGIT
BLT 4\$;;BR IF DONE
INC R2 ;;INCREASE THE BCD DIGIT BY 1
BR 3\$
4\$: ADD R1,R5 ;;ADD BACK THE CONSTANT
TST R2 ;;CHECK IF BCD DIGIT=0
BNE 5\$;;FALL THROUGH IF 0
TSTB (SP) ;;STILL DOING LEADING 0'S?
BMI 7\$;;BR IF YES
5\$: ASLB (SP) ;;MSD?
BCC 6\$;;BR IF NO
MOVB 1(SP),-1(R3) ;;YES--SET THE SIGN
6\$: BIS #'0,R2 ;;MAKE THE BCD DIGIT ASCII
7\$: BIS #' ,R2 ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
MOVB R2,(R3)+ ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
TST (R0)+ ;;JUST INCREMENTING
CMP R0,#10 ;;CHECK THE TABLE INDEX
BLT 2\$;;GO DO THE NEXT DIGIT
BGT 8\$;;GO TO EXIT
MOV R5,R2 ;;GET THE LSD
BR 6\$;;GO CHANGE TO ASCII
8\$: TSTB (SP)+ ;;WAS THE LSD THE FIRST NON-ZERO?
BPL 9\$;;BR IF NO
9\$: MOVB -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
CLRB (R3) ;;SET THE TERMINATOR
MOV (SP)+,R5 ;;POP STACK INTO R5

9860	057530	012603		MOV	(SP)+,R3	;;POP STACK INTO R3
9861	057532	012602		MOV	(SP)+,R2	;;POP STACK INTO R2
9862	057534	012601		MOV	(SP)+,R1	;;POP STACK INTO R1
9863	057536	012600		MOV	(SP)+,R0	;;POP STACK INTO R0
9864	057540	104400	057566	TYPE	\$DBLK	;;NOW TYPE THE NUMBER
9865	057544	016666	000002 000004	MOV	2(SP),4(SP)	;;ADJUST THE STACK
9866	057552	012616		MOV	(SP)+,(SP)	
9867	057554	000002		RTI		;;RETURN TO USER
9868	057556	023420		\$DTBL:	10000.	
9869	057560	001750			1000.	
9870	057562	000144			100.	
9871	057554	000012			10.	
9872	057556	000004		\$DBLK:	.BLKW 4	

```

9873 .SBTTL TYPE ROUTINE
9874
9875 *****
9876 *ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
9877 *THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
9878 *NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
9879 *NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
9880 *NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
9881 *
9882 *CALL:
9883 *1) USING A TRAP INSTRUCTION
9884 * TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
9885 *OR
9886 * TYPE
9887 * MESADR
9888 *
9889
9890 057576 105737 001157 $TYPE: TSTB $TFPLG ;; IS THERE A TERMINAL?
9891 057602 100002 BPL 1$ ;; BR IF YES
9892 057604 000000 HALT ;; HALT HERE IF NO TERMINAL
9893 057606 000407 BR 3$ ;; LEAVE
9894 057610 010046 1$: MOV RO, -(SP) ;; SAVE RO
9895 057612 017600 000002 MOV 22(SP), RO ;; GET ADDRESS OF ASCIZ STRING
9896 057616 112046 2$: MOV 2(SP), -(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
9897 057620 001005 BNE 4$ ;; BR IF IT ISN'T THE TERMINATOR
9898 057622 005726 TST (SP)+ ;; IF TERMINATOR POP IT OFF THE STACK
9899 057624 012600 60$: MOV (SP)+, RO ;; RESTORE RO
9900 057626 062716 000002 3$: ADD #2, (SP) ;; ADJUST RETURN PC
9901 057632 000002 RTI ;; RETURN
9902 057634 122716 000011 4$: CMPB #HT, (SP) ;; BRANCH IF <HT>
9903 057640 001430 BEQ 8$
9904 057642 122716 000200 CMPB #CRLF, (SP) ;; BRANCH IF NOT <CRLF>
9905 057646 001006 BNE 5$
9906 057650 005726 TST (SP)+ ;; POP <CR><LF> EQUIV
9907 057652 104400 TYPE ;; TYPE A CR AND LF
9908 057654 001223 $CRLF
9909 057656 105037 060012 CLR 2$ ;; CLEAR CHARACTER COUNT
9910 057662 000755 BR 2$ ;; GET NEXT CHARACTER
9911 057664 004737 057746 5$: JSR PC, $TYPEC ;; GO TYPE THIS CHARACTER
9912 057670 123726 001156 6$: CMPB $FILLC, (SP)+ ;; IS IT TIME FOR FILLER CHARS.?
9913 057674 001350 BNE 2$ ;; IF NO GO GET NEXT CHAR.
9914 057676 013746 001154 MOV $NULL, -(SP) ;; GET # OF FILLER CHARS. NEEDED
9915 ;; AND THE NULL CHAR.
9916 057702 105366 000001 7$: DECB 1(SP) ;; DOES A NULL NEED TO BE TYPED?
9917 057706 002770 BLT 6$ ;; BR IF NO--GO POP THE NULL OFF OF STACK
9918 057710 004737 057746 JSR PC, $TYPEC ;; GO TYPE A NULL
9919 057714 105337 060012 DECB $CHARCNT ;; DO NOT COUNT AS A COUNT
9920 057720 000770 BR 7$ ;; LOOP
9921
9922 ;HORIZONTAL TAB PROCESSOR
9923
9924 057722 112716 000040 8$: MOV 8'(SP) ;; REPLACE TAB WITH SPACE
9925 057726 004737 057746 9$: JSR PC, $TYPEC ;; TYPE A SPACE
9926 057732 132737 000007 060012 BITB #7, $CHARCNT ;; BRANCH IF NOT AT

```

9927	057740	001372			BNE	9\$:: TAB STOP
9928	057742	005726			TST	(SP)+	:: POP SPACE OFF STACK
9929	057744	000724			BR	2\$:: GET NEXT CHARACTER
9930	057746	105777	121176		\$TYPEC: TSTB	2\$TPS	:: WAIT UNTIL PRINTER IS READY
9931	057752	100375			BPL	\$TYPEC	
9932	057754	116677	000002	121170	MOVB	2(SP), 2\$TPB	:: LOAD CHAR TO BE TYPED INTO DATA REG.
9933	057762	122766	000015	000002	CMPB	#CR, 2(SP)	:: IS CHARACTER A CARRIAGE RETURN?
9934	057770	001003			BNE	1\$:: BRANCH IF NO
9935	057772	105037	060012		CLRB	\$CHARCNT	:: YES--CLEAR CHARACTER COUNT
9936	057776	000406			BR	\$TYPEX	:: EXIT
9937	060000	122766	000012	000002	1\$: CMPB	#LF, 2(SP)	:: IS CHARACTER A LINE FEED?
9938	060006	001402			BEQ	\$TYPEX	:: BRANCH IF YES
9939	060010	105227			INCB	(PC)+	:: COUNT THE CHARACTER
9940	060012	000000			\$CHARCNT: .WORD	0	:: CHARACTER COUNT STORAGE
9941	060014	000207			\$TYPEX: RTS	PC	
9942							

```

9943      .SBTTL  TTY INPUT ROUTINE
9944
9945      ;*****
9946      .ENABL  LSB
9947      060C16 000000 $TKCNT: .WORD 0      ;;NUMBER OF ITEMS IN QUEUE
9948      060020 000000 $TKQIN: .WORD 0      ;;INPUT POINTER
9949      060022 000000 $TKQOUT: .WORD 0     ;;OUTPUT POINTER
9950      060024 000011 $TKQSRT: .BLKB 9.    ;;TTY KEYBOARD QUEUE
9951      060035 $TKQEND=.
9952      060036 .EVEN
9953
9954      ;*TK INITIALIZE ROUTINE
9955      ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
9956      ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
9957
9958      ;*CALL:
9959      ;*      JSR      PC,$TKINT
9960      ;*      RETURN
9961
9962      060036 005037 060016 $TKINT: CLR      $TKCNT      ;;CLEAR COUNT OF ITEMS IN QUEUE
9963      060042 012737 060024 060020 MOV      # $TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
9964      060050 013737 060020 060022 MOV      $TKQIN,$TKQOUT ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
9965      060056 012737 060106 000060 MOV      # $TKSRV,$TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
9966      060064 012737 000200 000062 MOV      #200,$TKVEC+2 ;;"BR" LEVEL 4
9967      060072 005777 121050 TST      $TKB      ;;CLEAR DONE FLAG
9968      060076 012777 000100 121040 MOV      #100,$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
9969      060104 000207 PC      RTS      PC      ;;RETURN TO CALLER
9970
9971      ;*TK SERVICE ROUTINE
9972      ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
9973      ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
9974      ;*IT IN THE QUEUE.
9975      ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
9976      ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (OPERSEL)
9977
9978      060106 117746 121034 $TKSRV: MOVB    $TKB,-(SP) ;;PICKUP THE CHARACTER
9979      060112 042716 177600 BIC      #↑C177,(SP) ;;STRIP THE JUNK
9980      060116 021627 000003 CMP      (SP),#3 ;;IS IT A CONTROL C?
9981      060122 001007 BNE      1$ ;;BRANCH IF NO
9982      060124 104400 061075 TYPE     ,SCNTLC ;;TYPE A CONTROL-C (↑C)
9983      060130 004737 060036 JSR      PC,$TKINT ;;INIT THE KEYBOARD
9984      060134 005726 TST      (SP)+ ;;CLEAN UP STACK
9985      060136 000137 044614 JMP      OPERSEL ;;CONTROL C RESTART
9986      060142 021627 000007 1$: CMP      (SP),#7 ;;IS IT A CONTROL G?
9987      060146 001004 BNE      2$ ;;BRANCH IF NO
9988      060150 022737 000176 001140 CMP      #SWREG,SWR ;;IS SOFT-SWR SELECTED?
9989      060156 001500 BEQ      6$ ;;GO TO SWR CHANGE
9990
9991      060160 2$: CMP      #9,$TKCNT ;;IS THE QUEUE FULL?
9992      060160 022737 000011 060016 BNE      3$ ;;BRANCH IF NO
9993      060166 001004 TYPE     ,SBELL ;;RING THE TTY BELL
9994      060170 104400 001216 TST      (SP)+ ;;CLEAN CHARACTER OFF OF STACK
9995      060174 005726 BR      5$ ;;EXIT
9996      060176 000451

```

```

9997 060200 021627 000023 3$: CMP (SP),#23 ;: IS IT A CONTROL-S?
9998 060204 001021 ;: JNE 32$ ;: BRANCH IF NO
9999 060206 005077 120732 ;: CLR $STKS ;: DISABLE TTY KEYBOARD INTERRUPTS
10000 060212 005726 ;: TST (SP)+ ;: CLEAN CHAR OFF STACK
10001 060214 105777 120724 31$: TSTB $STKS ;: WAIT FOR A CHAR
10002 060220 100375 ;: BPL 31$ ;: LOOP UNTIL ITS THERE
10003 060222 117746 120720 ;: MOVB $STKB,-(SP) ;: GET THE CHARACTER
10004 060226 042716 177600 ;: BIC #1C177,(SP) ;: MAKE IT 7-BIT ASCII
10005 060232 022627 000021 ;: CMP (SP)+,#21 ;: IS IT A CONTROL-Q?
10006 060236 001366 ;: BNE 31$ ;: BRANCH IF NO
10007 060240 012777 000100 120676 ;: MOV #100,$STKS ;: REENABLE TTY KEYBOARD INTERRUPTS
10008 060246 000002 ;: RTI ;: RETURN
10009 060250 005237 060016 32$: INC $STCNT ;: COUNT THIS CHARACTER
10010 060254 021627 000140 ;: CMP (SP),#140 ;: IS IT UPPER CASE?
10011 060260 002405 ;: BLT 4$ ;: BRANCH IF YES
10012 060262 021627 000175 ;: CMP (SP),#175 ;: IS IT A SPECIAL CHAR?
10013 060266 003002 ;: BGT 4$ ;: BRANCH IF YES
10014 060270 042716 000040 ;: BIC #40,(SP) ;: MAKE IT UPPER CASE
10015 060274 112677 177520 4$: MOVB (SP)+,$STKQIN ;: AND PUT IT IN QUEUE
10016 060300 005237 060020 ;: INC $STKQIN ;: UPDATE THE POINTER
10017 060304 023727 060020 060035 ;: CMP $STKQIN,$STKQEND ;: GO OFF THE END?
10018 060312 001003 ;: BNE 5$ ;: BRANCH IF NO
10019 060314 012737 060024 060020 ;: MOV #STKQSRST,$STKQIN ;: RESET THE POINTER
10020 060322 000002 5$: RTI ;: RETURN

```

```

10021
10022 ;:*****
10023 ;: *SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
10024 ;: *ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
10025 ;: *SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
10026 ;: *CALL WHEN OPERATING IN TTY INTERRUPT MODE.

```

```

10027 060324 022737 000176 001140 $CKSWR: CMP #SWREG,SWR ;: IS THE SOFT-SWR SELECTED
10028 060332 001124 ;: BNE 15$ ;: EXIT IF NOT
10029 060334 105777 120604 ;: TSTB $STKS ;: IS A CHAR WAITING?
10030 060340 100121 ;: BPL 15$ ;: IF NOT, EXIT
10031 060342 117746 120600 ;: MOVB $STKB,-(SP) ;: YES
10032 060346 042716 177600 ;: BIC #1C177,(SP) ;: MAKE IT 7-BIT ASCII
10033 060352 021627 000007 ;: CMP (SP),#7 ;: IS IT A CONTROL-G?
10034 060356 001300 ;: BNE 2$ ;: IF NOT, PUT IT IN THE TTY QUEUE
10035 ;: ;: AND EXIT

```

```

10036
10037 ;:*****
10038 ;: *CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
10039 ;: *ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
10040 ;: *CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.

```

```

10041 060360 123727 001134 000001 6$: CMPB $AUTOB,#1 ;: ARE WE RUNNING IN AUTO-MODE?
10042 060366 001674 ;: BEQ 2$ ;: BRANCH IF YES
10043 060370 005726 ;: TST (SP)+ ;: CLEAR CONTROL-G OFF STACK
10044 060372 004737 060036 ;: JSR PC,$TKINT ;: FLUSH THE TTY INPUT QUEUE
10045 060376 005077 120542 ;: CLR $STKS ;: DISABLE TTY KEYBOARD INTERRUPTS
10046 060402 112737 000001 001135 ;: MOVB #1,$INTAG ;: SET INTERRUPT MODE INDICATOR
10047
10048 060410 104400 061107 ;: TYPE $CNTLG ;: ECHO THE CONTROL-G (+G)
10049 060414 104400 061114 ;: TYPE $MSWR ;: TYPE CURRENT CONTENTS
10050 060420 013746 000176 ;: MOV $WREG,-(SP) ;: SAVE SWREG FOR TYPEOUT

```

10051	060424	104401			TYPOC		:: GO TYPE--OCTAL ASCII(ALL DIGITS)
10052	060426	104400	061125		TYPE	, \$MNEW	:: PROMPT FOR NEW SWR
10053	060432	005046		19\$:	CLR	-(SP)	:: CLEAR COUNTER
10054	060434	005046			CLR	-(SP)	:: THE NEW SWR
10055	060436	105777	120502	7\$:	TSTB	2\$TKS	:: CHAR THERE?
10056	060442	100375			BPL	7\$:: IF NOT TRY AGAIN
10057							
10058	060444	117746	120476		MOVB	2\$TKB, -(SP)	:: PICK UP CHAR
10059	060450	042716	177600		BIC	#1C177, (SP)	:: MAKE IT 7-BIT ASCII
10060							
10061	060454	021627	000003		CMP	(SP), #3	:: IS IT A CONTROL-C?
10062	060450	001015			BNE	9\$:: BRANCH IF NOT
10063	060462	104400	061075		TYPE	, \$CNTLC	:: YES, ECHO CONTROL-C (↑C)
10064	060466	062706	000006		ADD	#6, SP	:: CLEAN UP STACK
10065	060472	123727	001135	000001	CMPB	\$INTAG, #1	:: REENABLE TTY KEYBOARD INTERRUPTS?
10066	060500	001003			BNE	8\$:: BRANCH IF NO
10067	060502	012777	000100	120434	MOV	#100, 2\$TKS	:: ALLOW TTY KEYBOARD INTERRUPTS
10068	060510	000137	044614	8\$:	JMP	OPERSEL	:: CONTROL-C RESTART
10069							
10070							
10071	060514	021627	000025	9\$:	CMP	(SP), #25	:: IS IT A CONTROL-U?
10072	060520	001005			BNE	10\$:: BRANCH IF NOT
10073	060522	104400	061102		TYPE	, \$CNTLU	:: YES, ECHO CONTROL-U (↑U)
10074	060526	062706	000006	20\$:	ADD	#6, SP	:: IGNORE PREVIOUS INPUT
10075	060532	000737			BR	19\$:: LET'S TRY IT AGAIN
10076							
10077							
10078	060534	021627	000015	10\$:	CMP	(SP), #15	:: IS IT A <CR>?
10079	060540	001022			BNE	16\$:: BRANCH IF NO
10080	060542	005766	000004		TST	4(SP)	:: YES, IS IT THE FIRST CHAR?
10081	060546	001403			BEQ	11\$:: BRANCH IF YES
10082	060550	016677	000002	120362	MOV	2(SP), 2\$SWR	:: SAVE NEW SWR
10083	060556	062706	000006	11\$:	ADD	#6, SP	:: CLEAR UP STACK
10084	060562	104400	001223	14\$:	TYPE	, \$CRLF	:: ECHO <CR> AND <LF>
10085	060566	123727	001135	000001	CMPB	\$INTAG, #1	:: RE-ENABLE TTY KBD INTERRUPTS?
10086	060574	001003			BNE	15\$:: BRANCH IF NOT
10087	060576	012777	000100	120340	MOV	#100, 2\$TKS	:: RE-ENABLE TTY KBD INTERRUPTS
10088	060604	000002		15\$:	RTI		:: RETURN
10089	060606	004737	057746	16\$:	JSR	PC, \$TYPEC	:: ECHO CHAR
10090	060612	021627	000060		CMP	(SP), #60	:: CHAR < 0?
10091	060616	002420			BLT	18\$:: BRANCH IF YES
10092	060620	021627	000067		CMP	(SP), #67	:: CHAR > 7?
10093	060624	003015			BGT	18\$:: BRANCH IF YES
10094	060626	042726	000060		BIC	#60, (SP)+	:: STRIP-OFF ASCII
10095	060632	005766	000002		TST	2(SP)	:: IS THIS THE FIRST CHAR
10096	060636	001403			BEQ	17\$:: BRANCH IF YES
10097	060640	006316			ASL	(SP)	:: NO, SHIFT PRESENT
10098	060642	006316			ASL	(SP)	:: CHAR OVER TO MAKE
10099	060644	006316			ASL	(SP)	:: ROOM FOR NEW ONE.
10100	060646	005266	000002	17\$:	INC	2(SP)	:: KEEP COUNT OF CHAR
10101	060652	056616	177776		BIS	-2(SP), (SP)	:: SET IN NEW CHAR
10102	060656	000667			BR	7\$:: GET THE NEXT ONE
10103	060660	104400	001222	18\$:	TYPE	, \$QUES	:: TYPE ?<CR><LF>
10104	060664	000720			BR	20\$:: SIMULATE CONTROL-U


```

10105          .DSABL LSB
10106
10107
10108          ;*****
10109          ;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
10110          ;CALL:
10111          ;*      RDCHR          ;;GET A CHARACTER FROM THE QUEUE
10112          ;*      RETURN HERE      ;;CHARACTER IS ON THE STACK
10113          ;*                               ;;WITH PARITY BIT STRIPPED OFF
10114
10115
10116 060656 011646          SRDCHR: MOV      (SP),-(SP)      ;;PUSH DOWN THE PC AND
10117 060670 016666 000004 000002      MOV      4(SP),2(SP)      ;;THE PS
10118 060676 005066 000004          CLR      4(SP)          ;;GET READY FOR A CHARACTER
10119 060702 005046          CLR      -(SP)          ;;PUT NEW PS ON STACK
10120 060704 012746 060712          MOV      #64$,-(SP)      ;;PUT NEW PC ON STACK
10121 060710 000002          RTI          ;;POP NEW PC AND PS
10122 060712
10123 060712 005737 060016          64$:   TST      STKCNT      ;;WAIT ON A CHARACTER
10124 060716 001775          1$:   BEQ      1$
10125 060720 005337 060016          DEC      STKCNT      ;;DECREMENT THE COUNTER
10126 060724 117766 177072 000004      MOVB    2$STKQOUT,4(SP)  ;;GET ONE CHARACTER
10127 060732 005237 060022          INC      STKQOUT      ;;UPDATE THE POINTER
10128 060736 023727 060022 060035      CMP     STKQOUT,#STKQEND ;;DID IT GO OFF OF THE END?
10129 060744 001003          BNE     2$          ;;BRANCH IF NO
10130 060746 012737 060024 060022      MOV     #STKQSR,STKQOUT ;;RESET THE POINTER
10131 060754 000002          RTI          ;;RETURN
10132          ;*****
10133          ;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
10134          ;CALL:
10135          ;*      RDLIN          ;;INPUT A STRING FROM THE TTY
10136          ;*      RETURN HERE      ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
10137          ;*                               ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
10138
10139 060756 010346          SRDLIN: MOV     R3, -(SP)      ;;SAVE R3
10140 060760 012703 061064          1$:   MOV     #STTYIN,R3      ;;GET ADDRESS
10141 060764 022703 061075          2$:   CMP     #STTYIN+9.,R3    ;;BUFFER FULL?
10142 060770 101405          BLOS   4$          ;;BR IF YES
10143 060772 104407          RDCHR  ;;GO READ ONE CHARACTER FROM THE TTY
10144 060774 112613          MOVB   (SP)+,(R3)      ;;GET CHARACTER
10145 060776 122713 000177          10$:  CMPB   #177,(R3)      ;;IS IT A RUBOUT
10146 061002 001003          BNE   3$          ;;SKIP IF NOT
10147 061004 104400 001222          4$:   TYPE   $QUES      ;;TYPE A '?'
10148 061010 000763          BR     1$          ;;CLEAR THE BUFFER AND LOOP
10149 061012 111337 061062          3$:   MOVB   (R3),9$      ;;ECHO THE CHARACTER
10150 061016 104400 061062          TYPE   9$
10151 061022 122723 000015          CMPB   #15,(R3)+      ;;CHECK FOR RETURN
10152 061026 001356          BNE   2$          ;;LOOP IF NOT RETURN
10153 061030 105063 177777          CLRB  -1(R3)        ;;CLEAR RETURN (THE 15)
10154 061034 104400 001224          TYPE   $LF          ;;TYPE A LINE FEED
10155 061040 012603          MOV   (SP)+,R3      ;;RESTORE R3
10156 061042 011646          MOV   (SP),-(SP)    ;;ADJUST THE STACK AND PUT ADDRESS OF THE
10157 061044 016666 000004 000002      MOV   4(SP),2(SP)    ;;FIRST ASCII CHARACTER ON IT
10158 061052 012766 061064 000004          MOV   #STTYIN,4(SP)

```

```

10159 061060 000002          RTI          ::RETURN
10160 061062      000          9$: .BYTE 0      ::STORAGE FOR ASCII CHAR. TO TYPE
10161 061063      000          .BYTE 0      ::TERMINATOR
10162 061064 000011          $TTYIN: .BLKB 9.  ::RESERVE 9. BYTES FOR TTY INPUT
10163 061075      136 005503 000012 $CNTLC: .ASCIZ /C/<15><12> ::CONTROL "C"
10164 061102 052536 005015 000 $CNTLU: .ASCIZ /U/<15><12> ::CONTROL "U"
10165 061107      136 006507 000012 $CNTLG: .ASCIZ /G/<15><12> ::CONTROL "G"
10166 061114 005015 053523 020122 $MSWR: .ASCIZ /15<12>/SWR = /
10167 061122 020075 000
10168 061125      040 047040 053505 $MNEW: .ASCIZ NEW = /
10169 061132 036440 000040
10170

```

;FROM THE TTY

```

10171 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
10172
10173
10174 ::*****
10175 ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
10176 ;*CHANGE IT TO BINARY.
10177 ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
10178 ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
10179 ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
10180 ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
10181 ;*CALL:
10182 ;*      RDOCT          ;; READ AN OCTAL NUMBER
10183 ;*      RETURN HERE   ;; LOW ORDER BITS ARE ON TOP OF THE STACK
10184 ;*                   ;; HIGH ORDER BITS ARE IN $HIOCT
10185 061136 011646 $RDOCT: MOV      (SP), -(SP)      ;; PROVIDE SPACE FOR THE
10186 061140 016666 000004 000002 MOV      4(SP), 2(SP)      ;; INPUT NUMBER
10187 061146 010046 MOV      RO, -(SP)      ;; PUSH RO ON STACK
10188 061150 010146 MOV      R1, -(SP)      ;; PUSH R1 ON STACK
10189 061152 010246 MOV      R2, -(SP)      ;; PUSH R2 ON STACK
10190 061154 104410 1$:      RDLIN      ;; READ AN ASCII LINE
10191 061156 012600 MOV      (SP)+, RO      ;; GET ADDRESS OF 1ST CHARACTER
10192 061160 010037 061264 MOV      RO, 5$      ;; AND SAVE IT
10193 061164 005001 CLR      R1          ;; CLEAR DATA WORD
10194 061166 005002 CLR      R2
10195 061170 112046 2$:      MOVVB     (RO)+, -(SP)      ;; PICKUP THIS CHARACTER
10196 061172 001420 BEQ      3$          ;; IF ZERO GET OUT
10197 061174 122716 000060 CMPB     #'0, (SP)      ;; MAKE SURE THIS CHARACTER
10198 061200 003026 BGT      4$          ;; IS AN OCTAL DIGIT
10199 061202 122716 000067 CMPB     #'7, (SP)
10200 061206 002423 BLT      4$
10201 061210 006301 ASL      R1          ;; *2
10202 061212 006102 RCL      R2
10203 061214 006301 ASL      R1          ;; *4
10204 061216 006102 ROL      R2
10205 061220 006301 ASL      R1          ;; *8
10206 061222 006102 ROL      R2
10207 061224 042716 177770 BIC      #'C7, (SP)      ;; STRIP THE ASCII JUNK
10208 061230 062601 ADD      (SP)+, R1      ;; ADD IN THIS DIGIT
10209 061232 000756 BR       2$          ;; LOOP
10210 061234 005726 3$:      TST      (SP)+      ;; CLEAN TERMINATOR FROM STACK
10211 061236 010166 000012 MOV      R1, 12(SP)      ;; SAVE THE RESULT
10212 061242 010237 061274 MOV      R2, $HIOCT
10213 061246 012602 MOV      (SP)+, R2      ;; POP STACK INTO R2
10214 061250 012601 MOV      (SP)+, R1      ;; POP STACK INTO R1
10215 061252 012600 MOV      (SP)+, RO      ;; POP STACK INTO RO
10216 061254 000002 RTI
10217 061256 005726 4$:      TST      (SP)+      ;; CLEAN PARTIAL FROM STACK
10218 061260 105010 CLRB     (RO)          ;; SET A TERMINATOR
10219 061262 104400 TYPE
10220 061264 000000 5$:      .WORD     0      ;; TYPE UP THRU THE BAD CHAR.
10221 061266 104400 001222 TYPE     $QUES      ;; "?" "CR" & "LF"
10222 061272 000730 BR       1$          ;; TRY AGAIN
10223 061274 000000 $HIOCT: .WORD     0      ;; HIGH ORDER BITS GO HERE

```

```

10224      .SBTTL  ERROR HANDLER ROUTINE
10225
10226      ;;*****
10227      ;;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
10228      ;;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
10229      ;;*AND GO TO $ERRTYP ON ERROR
10230      ;;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
10231      ;;*SW15=1      HALT ON ERROR
10232      ;;*SW13=1      INHIBIT ERROR TYPEOUTS
10233      ;;*SW10=1      BELL ON ERROR
10234      ;;*SW09=1      LOOP ON ERROR
10235      ;;*CALL
10236      ;;*      ERROR      N      ;;:ERROR=EMT AND N=ERROR ITEM NUMBER
10237
10238      061276      $ERROR:      CKSWR      ;;:TEST FOR CHANGE IN SOFT-SWR
10239      061276      104406
10240
10241      061300      012737      177777      015126      REGSA1:      MOV      #-1, @#ERFLG$      ;;:SET ERROR FLAG
10242      061306
10243
10244
10245      061306      105237      001103      7$:      INCB      $ERFLG      ;;:SET THE ERROR FLAG
10246      061312      001775      BEQ      7$      ;;:DON'T LET THE FLAG GO TO ZERO
10247      061314      013777      001102      117620      MOV      $STNM, @DISPLAY      ;;:DISPLAY TEST NUMBER AND ERROR FLAG
10248      061322      032777      002000      117610      BIT      #BIT10, @SWR      ;;:BELL ON ERROR?
10249      061330      001402      BEQ      1$      ;;:NO - SKIP
10250      061332      104400      001216      TYPE      $BELL      ;;:RING BELL
10251      061336      005237      001112      1$:      INC      $ERTTL      ;;:COUNT THE NUMBER OF ERRORS
10252      061342      011637      001116      MOV      (SP), $ERRPC      ;;:GET ADDRESS OF ERROR INSTRUCTION
10253      061346      162737      000002      001116      SUB      #2, $ERRPC
10254      061354      117737      117536      001114      MOVB      @ $ERRPC, $ITEMB      ;;:STRIP AND SAVE THE ERROR ITEM CODE
10255      061362      032777      020000      117550      BIT      #BIT13, @SWR      ;;:SKIP TYPEOUT IF SET
10256      061370      001004      BNE      20$      ;;:SKIP TYPEOUTS
10257      061372      004737      061444      JSR      PC, $ERRTYP      ;;:GO TO USER ERROR ROUTINE
10258      061376      104400      001223      TYPE      , $CRLF
10259      061402
10260      061402      005777      117532      20$:
10261      061406      100002      2$:      TST      @SWR      ;;:HALT ON ERROR
10262      061410      000000      BPL      3$      ;;:SKIP IF CONTINUE
10263      061412      104406      HALT      ;;:HALT ON ERROR!
10264      061414      032777      001000      117516      3$:      CKSWR      ;;:TEST FOR CHANGE IN SOFT-SWR
10265      061422      001402      BIT      #BIT09, @SWR      ;;:LOOP ON ERROR SWITCH SET?
10266      061424      013716      001110      BEQ      4$      ;;:BR IF NO
10267      061430      005737      031214      4$:      MOV      $LPERR, (SF)      ;;:FUDGE RETURN FOR LOOPING
10268      061434      001402      TST      $ESCAPE      ;;:CHECK FOR AN ESCAPE ADDRESS
10269      061436      013716      001214      BEQ      5$      ;;:BR IF NONE
10270      061442      MOV      $ESCAPE, (SP)      ;;:FUDGE RETURN ADDRESS FOR ESCAPE
10271      061442      000002      5$:      RTI      ;;:RETURN

```

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

```

10272
10273
10274
10275
10276
10277
10278
10279 061444
10280 061444 104400 001223
10291 061450 010046
10282 061452 005000
10283 061454 153700 001114
10284 061460 001004
10285
10286 061462 013746 001116
10287
10288 061466 104401
10289 061470 000445
10290 061472 005300
10291 061474 006300
10292 061476 006300
10293 061500 006300
10294 061502 062700 001226
10295 061506 012037 061516
10296 061512 001404
10297 061514 104400
10298 061516 000000
10299 061520 104400 001223
10300 061524 012037 061534
10301 061530 001404
10302 061532 104400
10303 061534 000000
10304 061536 104400 001223
10305 061542 010146
10306 061544 012001
10307 061546 001415
10308 061550 012000
10309 061552 105720
10310 061554 001003
10311 061556 013146
10312 061560 104401
10313 061562 000402
10314 061564
10315 061564 013146
10316 061566 104404
10317 061570 005711
10318 061572 001403
10319 061574 104400 061614
10320 061600 000764
10321
10322 061602 012601
10323 061604 012600
10324 061606 104400 001223
10325 061612 000207

```

```

*****
; *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.
$ERRTYP:
      TYPE      $CRLF      ;; "CARRIAGE RETURN" & "LINE FEED"
      MOV      RO, -(SP)   ;; SAVE RO
      CLR      RO         ;; PICKUP THE ITEM INDEX
      BISB     @($ITEMB, RO
      BNE     1$         ;; IF ITEM NUMBER IS ZERO, JUST
                          ;; TYPE THE PC OF THE ERROR
      MOV      $ERRPC, -(SP) ;; SAVE $ERRPC FOR TYPEOUT
                          ;; ERROR ADDRESS
                          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
      TYPD     10$        ;; GET OUT
      DEC      RO         ;; ADJUST THE INDEX SO THAT IT WILL
                          ;; WORK FOR THE ERROR TABLE
      ASL      RO
      ASL      RO
      ASL      RO
      ADD      @($ERRTB, RO ;; FORM TABLE POINTER
      MOV      (RO)+, 2$   ;; PICKUP "ERROR MESSAGE" POINTER
      BEQ     3$         ;; SKIP TYPEOUT IF NO POINTER
      TYPE     .WORD      ;; TYPE THE "ERROR MESSAGE"
      .WORD   0           ;; "ERROR MESSAGE" POINTER GOES HERE
      TYPE     $CRLF      ;; "CARRIAGE RETURN" & "LINE FEED"
      MOV      (RO)+, 4$   ;; PICKUP "DATA HEADER" POINTER
      BEQ     5$         ;; SKIP TYPEOUT IF 0
      TYPE     .WORD      ;; TYPE THE "DATA HEADER"
      .WORD   0           ;; "DATA HEADER" POINTER GOES HERE
      TYPE     $CRLF      ;; "CARRIAGE RETURN" & "LINE FEED"
      MOV      R1, -(SP)  ;; SAVE R1
      MOV      (RO)+, R1  ;; PICKUP "DATA TABLE" POINTER
      BEQ     9$         ;; BR IF NO DATA TO BE TYPED
      MOV      (RO)+, RO  ;; PICKUP "DATA FORMAT" POINTER
      TSTB    (RO)+      ;; "OCTAL" OR "DECIMAL"
      BNE     7$         ;; BR IF DECIMAL
      MOV      @ (R1)+, -(SP) ;; SAVE @ (R1)+ FOR TYPEOUT
      TYPD     8$        ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
      BR      8$
      MOV      @ (R1)+, -(SP) ;; SAVE @ (R1)+ FOR TYPEOUT
      TYPD     8$        ;; GO TYPE--DECIMAL ASCII WITH SIGN
      TST     (R1)       ;; IS THERE ANOTHER NUMBER?
      BEQ     9$         ;; BR IF NO
      TYPE     , 11$     ;; TYPE TWO(2) SPACES
      BR      6$        ;; LOOP
      MOV      (SP)+, R1  ;; RESTORE R1
      MOV      (SP)+, RO  ;; RESTORE RO
      TYPE     $CRLF      ;; "CARRIAGE RETURN" & "LINE FEED"
      RTS     PC         ;; RETURN
1$:
2$:
3$:
4$:
5$:
6$:
7$:
8$:
9$:
10$:

```

10326 061614 020040 000 115: .ASCIZ / / ;;TWO(2) SPACES
10327 061620 .EVEN

10328
10329
10330
10331
10332
10333
10334
10335
10336
10337
10338
10339
10340
10341
10342
10343
10344
10345
10346
10347
10348
10349
10350
10351
10352
10353
10354
10355
10356
10357
10358
10359
10360
10361
10362
10363
10364
10365
10366
10367
10368
10369
10370
10371
10372
10373
10374
10375
10376
10377
10378
10379
10380
10381

061620 017646 000000
061624 116637 000001 062043
061632 112637 062045
061636 062716 000002
061642 000406
061644 112737 000001 062043
061652 112737 000006 062045
061660 112737 000005 062042
061666 010346
061670 010446
061672 010546
061674 113704 062045
061700 005404
061702 062704 000006
061706 110437 062044
061712 113704 062043
061716 016605 000012
061722 005003
061724 006105
061726 000404
061730 006105
061732 006105
061734 006105
061736 010503
061740 006103
061742 105337 062044
061746 100016
061750 042703 177770
061754 001002

```
.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.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPOS    ;;CALL FOR TYPEOUT
*      .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*      .BYTE   M              ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS
*$STYPON----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
*$TYPOS: MOV      2(SP),-(SP)    ;;PICKUP THE MODE
        MOV      1(SP),%OFILL    ;;LOAD ZERO FILL SWITCH
        MOV      (SP)+,%SOMODE+1 ;;NUMBER OF DIGITS TO TYPE
        ADD     #2,(SP)          ;;ADJUST RETURN ADDRESS
        BR      $TYPON
*$TYPOC: MOV      #1,%OFILL      ;;SET THE ZERO FILL SWITCH
        MOV      #6,%SOMODE+1    ;;SET FOR SIX(6) DIGITS
*$TYPON: MOV      #5,%SOCNT      ;;SET THE ITERATION COUNT
        MOV      R3,-(SP)        ;;SAVE R3
        MOV      R4,-(SP)        ;;SAVE R4
        MOV      R5,-(SP)        ;;SAVE R5
        MOV      %SOMODE+1,R4    ;;GET THE NUMBER OF DIGITS TO TYPE
        NEG     R4
        ADD     #6,R4            ;;SUBTRACT IT FOR MAX. ALLOWED
        MOV      R4,%SOMODE      ;;SAVE IT FOR USE
        MOV      %OFILL,R4      ;;GET THE ZERO FILL SWITCH
        MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
        CLR     R3              ;;CLEAR THE OUTPUT WORD
        ROL    R5               ;;ROTATE MSB INTO "C"
        BR     3$              ;;GO DO MSB
        ROL    R5               ;;FORM THIS DIGIT
        ROL    R5
        ROL    R5
        MOV     R5,R3
        ROL    R3               ;;GET LSB OF THIS DIGIT
        DECB   %SOMODE          ;;TYPE THIS DIGIT?
        BPL    7$              ;;BR IF NO
        BIC    #177770,R3      ;;GET RID OF JUNK
        BNE    4$              ;;TEST FOR 0
        1$: ROL    R5
        2$: ROL    R5
        3$: ROL    R3
        4$: BNE
```

10382	061756	005704		TST	R4	:: SUPPRESS THIS 0'
10383	061760	001403		BEQ	5\$:: BR IF YES
10384	061762	005204	4\$:	INC	R4	:: DON'T SUPPRESS ANYMORE 0'S
10395	061764	052703	000060	BIS	#'0,R3	:: MAKE THIS DIGIT ASCII
10386	061770	052703	000040	5\$:	#',R3	:: MAKE ASCII IF NOT ALREADY
10387	061774	110337	062040	MOVB	R3,8\$:: SAVE FOR TYPING
10388	062000	104400	062040	TYPE	8\$:: GO TYPE THIS DIGIT
10389	062004	105337	062042	7\$:	\$OCNT	:: COUNT BY 1
10390	062010	003347		BGT	2\$:: BR IF MORE TO DO
10391	062012	002402		BLT	6\$:: BR IF DONE
10392	062014	005204		INC	R4	:: INSURE LAST DIGIT ISN'T A BLANK
10393	062016	000744		BR	2\$:: GO DO THE LAST DIGIT
10394	062020	012605	6\$:	MOV	(SP)+,R5	:: RESTORE R5
10395	062022	012604		MOV	(SP)+,R4	:: RESTORE R4
10396	062024	012603		MOV	(SP)+,R3	:: RESTORE R3
10397	062026	016666	000002 000004	MOV	2(SP),4(SP)	:: SET THE STACK FOR RETURNING
10398	062034	012616		MOV	(SP)+,(SP)	
10399	062036	000002		RTI		:: RETURN
10400	062040	000	8\$:	.BYTE	0	:: STORAGE FOR ASCII DIGIT
10401	062041	000		.BYTE	0	:: TERMINATOR FOR TYPE ROUTINE
10402	062042	000	\$OCNT:	.BYTE	0	:: OCTAL DIGIT COUNTER
10403	062043	000	\$OFILL:	.BYTE	0	:: ZERO FILL SWITCH
10404	062044	000000	\$OMODE:	.WORD	0	:: NUMBER OF DIGITS TO TYPE


```

10405
10406
10407
10408
10409
10410
10411
10412
10413 062046 010046
10414 062050 016600 000002
10415 062054 005740
10416 062056 111000
10417 062060 006300
10418 062062 016000 062070
10419 062066 000200
10420
10421
10422
10423
10424
10425
10426
10427
10428 062070
10429 062070 057576
10430 062072 061644
10431 062074 061620
10432 062076 061660
10433 062100 057352
10434
10435 062102 060414
10436
10437 062104 060324
10438 062106 060666
10439 062110 060756
10440 062112 061136
10441 062114 045656
10442 062116 045730
10443 062120 046246

```

.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.

```

```

$TRAP:  MOV    RO, -(SP)           ;;SAVE RO
        MOV    2(SP),RO         ;;GET TRAP ADDRESS
        TST    -(RO)            ;;BACKUP BY 2
        MOVB   (RO),RO          ;;GET RIGHT BYTE OF TRAP
        ASL    RO               ;;POSITION FOR INDEXING
        MOV    $TRPAD(RO),RO    ;;INDEX TO TABLE
        RTS    RO               ;;GO TO ROUTINE

```

.SBTTL TRAP TABLE

```

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

```

```

; ROUTINE
;-----
$TRPAD: $TYPE   ;;CALL=TYPE   TRAP+0(104400) TTY TYPEOUT ROUTINE
        $TYPOC  ;;CALL=TYPOC  TRAP+1(104401) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        $TYPOS  ;;CALL=TYPOS  TRAP+2(104402) TYPE OCTAL NUMBER (NO LEADING ZEROS)
        $TYPON  ;;CALL=TYPON  TRAP+3(104403) TYPE OCTAL NUMBER (AS PER LAST CALL)
        $TYPDS  ;;CALL=TYPDS  TRAP+4(104404) TYPE DECIMAL NUMBER (WITH SIGN)

        $GTSWR  ;;CALL=GTSWR  TRAP+5(104405) GET SOFT-SWR SETTING

        $CKSWR  ;;CALL=CKSWR  TRAP+6(104406) TEST FOR CHANGE IN SOFT-SWR
        $RDCHR  ;;CALL=RDCHR  TRAP+7(104407) TTY TYPEIN CHARACTER ROUTINE
        $RDLIN  ;;CALL=RDLIN  TRAP+10(104410) TTY TYPEIN STRING ROUTINE
        $RDOCT  ;;CALL=RDOCT  TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
        T.SCOPI ;;CALL=SCOPI  TRAP+12(104412) MY LOCAL SCOPES
        CHECKD  ;;CALL=CHECKD TRAP+13(104413) CHECK DVA, RDY, DPR, DRY
        WAIT.T  ;;CALL=WAT    TRAP+14(104414) WAIT LOOP

```

.SBTTL POWER DOWN AND UP ROUTINES

```

10444                                     .SBTTL POWER DOWN AND UP ROUTINES
10445
10446                                     ;:*****
10447                                     ;:POWER DOWN ROUTINE
10448 062122 012737 062266 000024 $PWRDN: MOV    $SILLUP,@#PWRVEC ;:SET FOR FAST UP
10449 062130 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;:PRIO:7
10450 062136 010046      MOV    R0,-(SP) ;:PUSH R0 ON STACK
10451 062140 010146      MOV    R1,-(SP) ;:PUSH R1 ON STACK
10452 062142 010246      MOV    R2,-(SP) ;:PUSH R2 ON STACK
10453 062144 010346      MOV    R3,-(SP) ;:PUSH R3 ON STACK
10454 062146 010446      MOV    R4,-(SP) ;:PUSH R4 ON STACK
10455 062150 010546      MOV    R5,-(SP) ;:PUSH R5 ON STACK
10456 062152 017746 116762      MOV    @SWR,-(SP) ;:PUSH @SWR ON STACK
10457 062156 010637 062272      MOV    SP,$SAVR6 ;:SAVE SP
10458 062162 012737 062174 000024      MOV    $PWRUP,@#PWRVEC ;:SET UP VECTOR
10459 062170 000000      HALT
10460 062172 000776      BR      .-2 ;:HANG UP
10461
10462                                     ;:*****
10463                                     ;:POWER UP ROUTINE
10464 062174 012737 062266 000024 $PWRUP: MOV    $SILLUP,@#PWRVEC ;:SET FOR FAST DOWN
10465 062202 013706 062272      MOV    $SAVR6,SP ;:GET SP
10466 062206 005037 062272      CLR    $SAVR6 ;:WAIT LOOP FOR THE TTY
10467 062212 005237 062272      1$: INC    $SAVR6 ;:WAIT FOR THE INC
10468 062216 001375      BNE    1$ ;:OF WORD
10469 062220 012677 116714      MOV    (SP)+,@SWR ;:POP STACK INTO @SWR
10470 062224 012605      MOV    (SP)+,R5 ;:POP STACK INTO R5
10471 062226 012604      MOV    (SP)+,R4 ;:POP STACK INTO R4
10472 062230 012603      MOV    (SP)+,R3 ;:POP STACK INTO R3
10473 062232 012602      MOV    (SP)+,R2 ;:POP STACK INTO R2
10474 062234 012601      MOV    (SP)+,R1 ;:POP STACK INTO R1
10475 062236 012600      MOV    (SP)+,R0 ;:POP STACK INTO R0
10476 062240 012737 062122 000024      MOV    $PWRDN,@#PWRVEC ;:SET UP THE POWER DOWN VECTOR
10477 062246 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;:PRIO:7
10478 062254 104400      TYPE ;:REPORT THE POWER FAILURE
10479 062256 062274      $PWRMG: .WORD $POWER ;:POWER FAIL MESSAGE POINTER
10480 062260 012716      MOV    (PC)+,(SP) ;:RESTART AT BEGIN
10481 062262 017356      $PWRAD: .WORD BEGIN ;:RESTART ADDRESS
10482 062264 000002      RTI
10483 062266 000000      $SILLUP: HALT ;:THE POWER UP SEQUENCE WAS STARTED
10484 062270 000776      BR      .-2 ;:BEFORE THE POWER DOWN WAS COMPLETE
10485 062272 000000      $SAVR6: 0 ;:PUT THE SP HERE
10486 062274 005015 047520 042527 $POWER: .ASCIZ <15><12>"POWER"
10487 062302 000122
10488                                     .EVEN
10489
10490                                     .END
000001

```

ACL	=	000040	2243#								
ACU	=	100000	2196#								
AOE	=	001000	2112#								
AS	=	015052	1692	1695	2329#	4345*	4357	4536*	4547	4684*	4694
ATA	=	100000	2100#	4342	4533	4681					
ATABLE	=	017336	2429#	2848							
ATTENT	=	015136	2364#	2808*	2848*	4345	4536	4684			
ATO	=	000001	2134#								
AT1	=	000002	2135#								
AT2	=	000004	2136#								
AT3	=	000010	2137#								
AT4	=	000020	2138#								
AT5	=	000040	2139#								
AT6	=	000100	2140#								
AT7	=	000200	2141#								
A16	=	000400	2075#	4182							
A17	=	001000	2076#	4182							
BA	=	015030	1688	2318#	4338*	4521*	4526*	4670*	4674*		
BA1	=	000010	2048#	4184	8149						
BASECH	=	051320	2600	8542#							
BEGIN	=	017356	191	2439#	10481						
BEGIN2	=	017346	194	2437#							
BITST	=	045442	2632	7431#							
BIT0	=	000001	151#	7886	7921	7923					
BIT00	=	000001	141#	151							
BIT01	=	000002	140#	150							
BIT02	=	000004	139#	149							
BIT03	=	000010	138#	148							
BIT04	=	000020	137#	147							
BIT05	=	000040	136#	146							
BIT06	=	000100	135#	145							
BIT07	=	000200	134#	144							
BIT08	=	000400	133#	143	9773						
BIT09	=	001000	132#	142	9781	10264					
BIT1	=	000002	150#	7900							
BIT10	=	002000	131#	10248							
BIT11	=	004000	130#	9788							
BIT12	=	010000	129#								
BIT13	=	020000	128#	7916	7918	10255					
BIT14	=	040000	127#	7893	9759						
BIT15	=	100000	126#	7888	7895	7902	7911	7913			
BIT2	=	000004	149#								
BIT3	=	000010	148#								
BIT4	=	000020	147#								
BIT5	=	000040	146#								
BIT6	=	000100	145#								
BIT7	=	000200	144#								
BIT8	=	000400	143#								
BIT9	=	001000	142#								
BLT1	=	045470	7437#	7442							
BLT2	=	045512	7437	7438	7444#						
BLT3	=	045522	7445	7446#							
BPTVEC	=	000014	158#								
CA	=	015046	2327#								

ERPOS	051010	9443#	8449*	8466										
ERR	= 040000	2099#	4342	4533	4681									
ERRVEC	= 000004	154#	2467	2468*	2479*	2579*	2586*	2607*	9764	9765*	9767*	9770*		
ERSTAR	052152	8608#												
ERUNIT	052150	8607#	8609											
ERWORD	052460	1672	1674	1712	1715	1719	3010	4201*	4206*	4386	4551	4707	7769*	7775*
		8850#	8891*	8916*	8920*	8944*	8968*	9457*	9464*	9468*	9490*	9499*	9506*	9522*
		9527*	9533*	9546*										
ER1	015036	1677	1680	1692	1702	1712	1715	1719	1725	1735	2323#	4340*	4531*	4679*
		5150	5333	5509	5529	5846	6024	6201	6221	6522	6702	6879	6899	7070
		7090												
ER2	015042	1692	2325#											
ER3	015050	1692	2328#											
EXT1	= 000001	2163#												
EXT10	= 000010	2166#												
EXT2	= 000002	2164#												
EXT20	= 000020	2167#												
EXT4	= 000004	2165#												
EXT40	= 000040	2168#												
ECO	024114	3057	3070#											
E11	025514	3371	3379#											
E12	025642	3412	3419#											
E44	030346	4057	4065#											
FEN	= 000200	2189#												
FER	= 000020	2107#												
FILLEC	051164	5079	5252	5432	5775	5945	6121	6451	6615	6800	6991	9501#		
FIRST	017332	2417#	2487	2537*										
FMT22	= 010000	2210#	4156	4175	4242	4260	4419	4437	4590	4607	4765	4783	4820	4838
		4930	4946	4965	5096	5119	5177	5270	5302	5356	5449	5480	5542	5625
		5641	5660	5792	5815	5873	5963	5995	6047	6138	6170	6235	6309	6325
		6344	6468	6491	6549	6633	6671	6725	6817	6850	6912	7008	7039	7104
		7708	7728	7805	7827	7968	8021	8150						
FNWORD	055536	4595*	4825*	4935*	5630*	6314*	9380#							
FORMAT	053722	9131#	9149											
FOUT	055542	9377	9383#											
FSYNER	055512	9283*	9311#	9494*										
FUTABL	015152	2387#												
GCRC	055506	4598	4828	4938	5633	6317	9295#	9516	9521					
GECC1	050373	1722	1725	1731	4920*	4982	5005	5088*	5156	5262*	5340	5441*	5516	5615*
		5677	5701	5784*	5852	5955*	6031	6130*	6208	6299*	6360	6384	6460*	6528
		6625*	6709	6809*	6886	7000*	7077	8250#	8292	8378*	9509*	8525		
GECC2	050372	1722	1725	1731	4921*	4986	5006	5089*	5263*	5442*	5616*	5681	5702	5785*
		5956*	6131*	6300*	6364	6385	6461*	6626*	6810*	7001*	8253#	8293	8379*	8510*
		8526												
GNS	= ***** U	173	2496	2501	2506	2510	2514	2518	2522	2526	2530	2534	2557	2594
		2598	2573	2700	2704	2708	2712	2759	2765	2771	2851	2857	2863	2897
		2903	2909	2935	2939	2943	2950	3333	3337	3341	3345	7233	7239	7343
		7349	7356	7360	7364	7371	7375	8545	8551	9563	9569	8575	9579	9585
		8591	8595	9719	10429	10430	10431	10432	10433	10435	10437	10438	10439	10440
		10441	10442	10443										
GC	= 000001	2072#	2982	8157	8185	8715	9267							
GRV	= 000010	2088#												
GTSWR	= 104405	2544	10435#											
HACTMP	050412	4919*	5087*	5261*	5440*	5614*	5783*	5954*	6129*	6299*	6459*	6624*	6809*	6999*

MWR = 000040	2127*	9073	9089	4430	9440									
MXF = 001000	2054*	3711	3713	3717										
NCODE 050376	4918	5086	5260	5439	5613	5782	5953	6128	6297	6458	6623	6807	6998	
	8259*													
NCOUNT 050400	4918*	5086*	5260*	5439*	5613*	5782*	5953*	6128*	6297*	6458*	6623*	6807*	6998*	
	8260*	8459*												
NED = 010000	2057*	3359	3368	3409	3453	4101	8611							
NEM = 004000	2056*													
NHS = 002000	2192*													
NOPERA 015152	2388*													
NOSYNC 052454	7381*	8793	8847*	8932	8960	9748*								
NOLNIT 015116	2346*	2728*	2778*	2785	2830*	7255*								
NOWORD 052406	4161*	8790*												
NUNIT 015120	2348*	2785*	2786*											
N11 025502	3356	3373*												
N12 025630	3398	3414*												
N14 026056	3486	3489*												
N36 027570	3887	3901*												
OCYL = 100000	2246*													
OF 015044	2326*													
OFREV = 000200	2207*													
OFSETC 015204	2401*													
OF100 = 000004	2202*													
OF200 = 000010	2203*													
OF25 = 000001	2200*													
OF400 = 000020	2204*													
OF50 = 000002	2201*													
OF800 = 000040	2205*													
OPERSE 044614	7339*	9985	10068											
OPI = 020000	2116*													
OR = 000200	2052*	4522	4528	4678										
OUT 052424	8779	8787	8792	8798*										
000 024116	3068	3072*												
010 025024	3291	3299*												
021 026600	3648	3657*												
043 030220	4017	4029*												
044 030324	4048	4059*												
PAR = 000010	2106*													
PAT = 000020	2049*													
PC =%000007	85*	2486*	2636*	2667*	2807*	2929*	2969*	3017*	3027*	4116*	4170*	4177*	4186*	
	4195*	4255*	4263*	4288*	4294*	4371*	4392*	4432*	4440*	4464*	4470*	4567*	4602*	
	4610*	4634*	4640*	4713*	4734*	4843*	4848*	4853*	4866*	4868*	4912*	4957*	4967*	
	4992*	5028*	5037*	5079*	5113*	5125*	5132*	5149*	5173*	5191*	5207*	5211*	5252*	
	5296*	5308*	5315*	5332*	5352*	5375*	5390*	5394*	5432*	5474*	5485*	5491*	5508*	
	5528*	5538*	5562*	5577*	5581*	5607*	5652*	5662*	5687*	5724*	5733*	5775*	5809*	
	5821*	5828*	5845*	5869*	5886*	5900*	5904*	5945*	5989*	6000*	6006*	6023*	6043*	
	6063*	6079*	6083*	6121*	6164*	6176*	6183*	6200*	6218*	6231*	6251*	6267*	6271*	
	6291*	6336*	6346*	6370*	6407*	6416*	6451*	6485*	6497*	6504*	6521*	6545*	6561*	
	6575*	6579*	6615*	6665*	6677*	6684*	6701*	6721*	6742*	6758*	6762*	6800*	6844*	
	6855*	6861*	6878*	6898*	6908*	6932*	6948*	6952*	6991*	7033*	7045*	7052*	7069*	
	7089*	7100*	7123*	7137*	7141*	7159*	7170*	7176*	7194*	7208*	7228*	7286*	7289*	
	7299*	7304	7418*	7437*	7438*	7457*	7459*	7523*	7537*	7580*	7583*	7594*	7619*	
	7622*	7721*	7732*	7738*	7741*	7778*	7780*	7819*	7831*	7836*	7839*	7980*	8003*	
	8004*	8024*	8033*	8053*	8061*	8079*	8084*	8162*	8190*	8399*	8405*	8432*	8474*	

PR6 = 000300	94#													
PR7 = 000340	95#													
PS = 177776	68#	69	2484*	3986*	4020*	4051*	7163*	7198*	7230*	7340*				
PSEL = 002000	2077#	3679	3924	3929										
PSU = 000001	2239#													
PSW = 177776	69#													
PUTREG 045374	4195	4294	4470	4640	5149	5191	5332	5375	5508	5528	5562	5845	5886	
	6023	6063	6200	6218	6251	6521	6561	6701	6742	6878	6898	6932	7069	
	7089	7123	7170	7176	7208	7405#	7537	7594	8033	8399	8405	8474		
PWRVEC= 000024	160#	2458*	2459*	10448*	10449*	10458*	10464*	10476*	10477*					
P12 050416	8277#	8331*	8332*	8361										
P22 050420	8278#	8339*	8340*	8366										
P24 050422	8279#	8347*	8348*	8367										
P3 050414	8276#	8322*	8323*	8360										
RCOM 056660	9687#	9692*												
RCYL 052444	8817#	8855*	8896											
RDCHR = 104407	10143	10438#												
RDHEAD 052462	8731	8855#												
RDLIN = 104410	10190	10439#												
RDOCT = 104411	2559	7366	7377	8554	8571	10440#								
RDY = 000200	2074#	4022	7164	7177	7199	7538	7549	7595						
READ 053254	8883	8888	8897	8899	8901	8903	8905	8955	8959	8995#	9701	9709		
READAT 015176	2398#													
READIN 015212	2404#													
RECALI 015156	2390#													
REDATA 056662	8794	9691#												
REFOR 015200	2399#	4267	5130	5313	5489	5826	6004	6181	6502	6682	6859	7050		
REGADR 045440	1670	1705	1708	2014	2016	2018	2656*	3012*	3062*	3099*	3123*	3153*	3180*	
	3211*	3239*	3286*	3351*	3394*	3450*	3477*	3537*	3584*	3643*	3675*	3710*	3747*	
	3778*	3811*	3857*	3891*	3921*	3950*	3983*	4012*	4044*	4098*	4388*	4563*	4709*	
	7429#	7433*	8093*	8104*	8114*	8125*	8195*							
REGSA1 061306	2577	10242#												
REINTO 016264	2413#	2978	2985*	2998	3004	4257	4338	4363	4543	4547*	4555	4690	4694*	
	4701	5001	5002	5005*	5006*	5008	5009	5019	5115	5199	5298	5382	5476	
	5569	5697	5698	5701*	5702*	5704	5705	5715	5811	5892	5991	6071	6166	
	6259	6380	6381	6384*	6385*	6387	6388	6398	6487	6567	6667	6750	6846	
	6940	7035	7129	7724	7823	8047	9072							
RELEAS 015162	2392#													
RESVEC= 000010	155#													
RETCL 015206	2402#													
RHAS 014776	2296#	2650	2675	3444	4091									
RHBA 014754	2280#	3476	3536	3583	4172*	4257*	4434*	4604*	4833*	4944*	5115*	5298*	5476*	
	5639*	5811*	5991*	6166*	6323*	6487*	6667*	6846*	7035*	7724*	7823*	8018*	8147*	
	2309#													
RHBAE 015022	2294#	4176*	4262*	4439*	4609*	4839*	4966*	5123*	5306*	5484*	5661*	5819*	5999*	
RHCA 014772	6174*	6345*	6495*	6675*	6854*	7043*	7731*	7830*	8010*	8151*	8181*			
	2304#	7411	8191	8195										
RHCC 015016	2289#	2973*	2982*	3054	3361	3402	3440	3642	3674	3856	3890	3920	3949	
RHCS1 014760	3982	4011	4043	4087	4291	4467	4637	4835*	7515	8144*	8157*	8183*	8185*	
	8547	8581	8715*	9267*										
RHCS2 014756	2281#	2631	2676	2734*	3052*	3094*	3118*	3148*	3175*	3206*	3234*	3277*	3319*	
	3350	3389*	3393	3433*	3437	3449	3472*	3532*	3579*	3638*	3672*	3706*	3709	
	3742*	3746	3773*	3777	3806*	3810	3847*	3851	3877*	3881	3916*	3926*	3945*	
	3955*	3978*	4009*	4041*	4080*	4084	4097	4184*	7516	8149*				

RHCS3	015020	2308*	3281											
RHDB	014750	2278*	2581	8555										
RHDST	014764	2291*	4174*	4258*	4435*	4605*	4834*	4955*	5118*	5301*	5479*	5650*	5814*	5994*
		6169*	6334*	6490*	6670*	6849*	7038*	7227*	7826*	8015*	8148*	8182*		
RHDS1	015002	2298*	7517											
RHDT	015004	2299*	2735	2737	2742	2744	2747	2749	2767	2865	2873	2875	2878	2890
		2883	2885	2914	2916									
RHEC1	015010	2301*	8402											
RHEC2	015012	2302*	3066	3284	4512	8396								
RHER1	014762	2290*	2679	7518	8089	8093	8099	8104	8111	8114	8119	8125		
RHER2	014766	2292*												
RHER3	014774	2295*												
RHLA	015014	2303*												
RHMR	015000	2297*	2971*	3028*	4117*	4313	4489	5169*	5170*	5865*	5866*	6541*	6542*	8156*
		8184*	8451	8712*	8713*	8714*	8862	9148	9264*	9265*	9266*	9424	9605	
RHOF	014770	2293*	4175*	4260*	4437*	4607*	4838*	4965*	5119*	5302*	5480*	5660*	5815*	5995*
		6170*	6344*	6491*	6671*	6850*	7039*	7728*	7827*	8021*	8150*			
RHSN	015006	2300*	2859	2913	2915									
RHWC	014752	2279*	2977	2991	3012	3098	3122	3152	3179	3210	3238	4171*	4256*	4272
		4352	4388	4433*	4449	4542	4563	4603*	4619	4689	4709	4832*	4942*	5114*
		5297*	5475*	5637*	5810*	5990*	6165*	6321*	6486*	6666*	6845*	7034*	7409	7411
		7723*	7822*	8016*	8146*									
RH70	017334	2420*	2551*	2626	3279	3702	3737	3842	4144	4517	4666			
RH70CK	020720	2549*												
RKEY1	052450	8819*	8857*	8900										
RKEY2	052452	8820*	8858*	8902										
RMR	= 000004	2105*												
RNCTR1	055320	9253*	9269*	9271*										
RNO	056656	9686*	9691*											
RNWAT1	055400	9269*												
RPTRP1	044124	7161	7175*											
RPTRP2	044230	7196	7207*											
RPVEC	014746	2268*	2485*	3647	3648*	3657*	4016	4017*	4029*	4047	4048*	4067*	7160	7195
		8565	8572*	8587	9722*									
RPVECT	056772	2485	9716*	9722										
RSETR	052446	8818*	8856*	8898										
RSYNC	052442	4782	4798	4805	8816*	8887	8892	9930	8942	8958	8969	9488	9492	9543
		9548												
RUNCTR	052176	8702*	8717*	8718*										
RUNWAT	052256	8717*												
RO	=%000000	76*	2580*	2583*	2678*	2683*	2721*	2724*	2729*	2781*	2814*	2815	2824*	2825
		2827	2829	2831*	2848	2976*	2990*	3003*	3029*	3353*	3355*	3395*	3397*	3884*
		3886*	4118*	4149*	4198*	4202	4271*	4313*	4314*	4315*	4316*	4318*	4319*	4328*
		4329*	4351*	4361*	4378*	4448*	4489*	4490*	4491*	4492*	4494*	4495*	4504*	4505*
		4541*	4553*	4618*	4658*	4659*	4688*	4699*	4754*	4756*	4762*	4765*	4767*	4768*
		4769*	4779*	4782*	4783*	4784*	4785*	4786*	4791	4794*	4798*	4801*	4805*	4807*
		4814*	4858*	4907*	4909*	4943*	4944	4946*	4948*	4949*	4950*	4952*	5000*	5007*
		5018*	5066*	5067*	5071	5074*	5168*	5171*	5176*	5177*	5180*	5181*	5182*	5185*
		5197*	5239*	5240*	5244	5247*	5346*	5355*	5356*	5359*	5360*	5361*	5364*	5380*
		5419*	5420*	5424	5427*	5522*	5541*	5542*	5545*	5546*	5547*	5550*	5567*	5602*
		5604*	5638*	5639	5641*	5643*	5644*	5645*	5647*	5696*	5703*	5714*	5762*	5763*
		5767	5770*	5864*	5867*	5872*	5873*	5876*	5877*	5878*	5882*	5890*	5932*	5933*
		5937	5940*	6037*	6046*	6047*	6050*	6051*	6052*	6055*	6069*	6108*	6109*	6113
		6116*	6214*	6234*	6235*	6238*	6239*	6240*	6243*	6257*	6286*	6288*	6322*	6323

6325*	6327*	6328*	6329*	6331*	6379*	6386*	6397*	6439*	6440*	6444	6447*	6540*	
6543*	6548*	6549*	6552*	6553*	6554*	6557*	6565*	6602*	6603*	6607	6610*	6715*	
6724*	6725*	6728*	6729*	6730*	6733*	6748*	6787*	6788*	6792	6795*	6892*	6911*	
6912*	6915*	6916*	6917*	6920*	6938*	6978*	6979*	6983	6986*	7083*	7103*	7104*	
7107*	7108*	7109*	7112*	7127*	7160*	7161*	7162*	7195*	7196*	7197*	7257*	7259	
7296*	7299	7406	7409*	7412	7417*	7483	7484	7485	7495*	7634	7635*	7636	
7638	7639	7640	7641*	7685	7686	7687	7694*	7761	7762	7763	7764	7765	
7766*	7794*	7872	7873*	7881	7933*	7950	7953*	7956	7979*	8001	8008	8009	
8010	8011	8012	8013	8014	8016	8018	8019	8045*	8071*	8085	8129*	8142	
8161*	8178	8180	8200*	8286	8303*	8310*	8313*	8314*	8315	8360*	8361*	8363	
8365*	8366*	8367*	8369	8431*	8447	8449	8492*	8502	8532*	8555*	8557*	8558*	
8705	8804*	8862*	8864*	8865*	8866*	8867*	8870*	8871*	8874*	8875*	8889	8962	
8966	8998*	9001*	9003*	9006	9015*	9020*	9023	9032*	9065	9070*	9072*	9073	
9082*	9088*	9089	9098*	9112*	9137	9148*	9185*	9257	9389*	9424*	9425*	9427*	
9428*	9430	9437*	9439*	9440	9448*	9599	9605*	9608*	9609*	9610*	9611*	9612*	
9616*	9617*	9633*	9634*	9636*	9637*	9638*	9639*	9654*	9655*	9656*	9657*	9667*	
9668*	9678*	9819	9829*	9833	9849	9850	9863*	9894	9895*	9896	9899*	10187	
10191*	10192	10195	10215*	10218*	10281	10282*	10283*	10290*	10291*	10292*	10293*	10294*	
10295	10300	10306	10308*	10309	10323*	10413	10414*	10415	10416*	10417*	10418*	10419*	
10450	10475*												
R1	77*	2581*	2582	2588	2650*	2651*	2652	2656	2675*	2685	2688	2730	3354*
	3358	3363	3373*	3374	3396*	3400	3404	3414*	3415	3885*	3888	3901*	3902
	4183*	4199*	4202	4205	4215	4267*	4312*	4322*	4327*	4330*	4444*	4488*	4498*
	4503*	4506*	4614*	4657*	4660*	4755*	4757*	4908*	4910*	4962*	5130*	5183*	5186*
	5313*	5362*	5365*	5489*	5548*	5551*	5603*	5605*	5657*	5826*	5879*	5883*	6004*
	6053*	6056*	6181*	6241*	6244*	6287*	6289*	6341*	6502*	6555*	6558*	6682*	6731*
	6734*	6859*	6918*	6921*	7050*	7110*	7113*	7164*	7199*	7258*	7259	7262	7407
	7410*	7412*	7416*	7480	7483*	7486	7488*	7494*	7515*	7522*	7544	7551	7555
	7598	7606	7682	7685*	7688	7693*	7736*	7756	7761*	7770	7773	7781	7793*
	7835*	7874	7878*	7886	7893	7900	7908*	7911*	7913*	7916*	7918*	7921*	7923*
	7928	7932*	7951	7954*	7957*	7959*	7963*	7978*	8009*	8143	8158*	8160*	8287
	8292*	8300	8308	8336	8344	8359*	8368*	8369*	8376*	8378	8394	8430*	8450
	8451*	8452*	8471	8479*	8480*	8487*	8488*	8491*	8503	8511*	8514	8531*	8556*
	8559*	8706	8724*	8727*	8728	8729	8730	8731*	8789*	8794*	8803*	8855	8856
	8857	8858	8859	8860	8972*	8973*	9134	9135	9136	9138	9142*	9145*	9169*
	9172*	9176*	9179*	9184*	9186*	9258	9277*	9281*	9282	9283	9284	9285	9286
	9287	9289*	9378*	9388*	9416	9417	9418	9419	9420	9421	9423*	9453*	9466*
	9487*	9488	9493	9501*	9502	9508	9515*	9516	9520	9529*	9535	9542*	9543
	9547	9552*	9553*	9598	9600	9651*	9664*	9677*	9679*	9691	9692	9693	9713*
	9714*	9820	9833*	9834	9838	9862*	10188	10193*	10201*	10203*	10205*	10208*	10211
	10214*	10305	10306*	10311	10315	10317	10322*	10451	10474*				
R11	025434												
R15	026260												
R16	026430												
R2	=:000002												
	73*	2582*	2676*	2677*	2682*	3061*	3062	3063	3098*	3099	3102*	3104*	3105
	3122*	3123	3126*	3128*	3130*	3132*	3134	3152*	3153	3156*	3158*	3159*	3161
	3179*	3180	3183*	3185*	3187*	3188*	3190*	3192	3210*	3211	3214*	3216*	3218*
	3220	3238*	3239	3242*	3244*	3246*	3247*	3249*	3251	3281*	3284*	3285*	3286
	3295	3350*	3351	3363*	3367	3393*	3394	3404*	3408	3437*	3442*	3444*	3446
	3448*	3449*	3450	3452	3476*	3477	3482*	3483	3536*	3537	3540*	3544*	3545
	3583*	3584	3587*	3591*	3593*	3595*	3597	3642*	3643	3650*	3653	3674*	3675
	3679*	3682	3709*	3710	3713*	3715	3746*	3747	3750*	3754	3777*	3778	3781*
	3785*	3787	3810*	3811	3814*	3818*	3820	3851*	3854*	3856*	3857	3859	3881*
	3888*	3890*	3891	3893	3920*	3921	3924*	3928	3949*	3950	3953*	3957	3982*

TST14	024314	3137	3145#																	
TST15	024412	3164	3172#																	
TST16	024514	3195	3203#																	
TST17	024612	3223	3231#																	
TST2	021210	2619#																		
TST20	024714	3254	3274#																	
TST21	025032	3316#																		
TST22	025516	3376	3386#																	
TST23	025644	3416	3430#																	
TST24	025764	3456	3469#																	
TST25	026164	3491	3529#																	
TST26	026324	3554	3576#																	
TST27	026476	3607	3635#																	
TST3	021366	2628	2646#																	
TST30	026606	3669#																		
TST31	026704	3686	3696#																	
TST32	027014	3704	3720	3731#																
TST33	027130	3739	3758	3770#																
TST34	027234	3791	3803#																	
TST35	027340	3824	3836#																	
TST36	027454	3844	3863	3874#																
TST37	027600	3897	3899	3913#																
TST4	021440	2654	2664#																	
TST40	027702	3932	3942#																	
TST41	030004	3961	3975#																	
TST42	030112	3996	4006#																	
TST43	030226	4038#																		
TST44	030356	4077#																		
TST45	030476	4104	4113#																	
TST46	030532	4137#																		
TST47	031064	4146	4197	4219	4236#															
TST5	022554	2790	2803#	7253	7263															
TST50	031556	4297	4414#																	
TST51	032302	4474	4584#																	
TST52	032742	4643	4731#																	
TST53	033516	4851	4885	4903#																
TST54	034144	4977	5025	5055#																
TST55	034662	5148	5204	5228#																
TST56	035406	5331	5387	5409#																
TST57	036150	5507	5574	5598#																
TST6	023260	2927#																		
TST60	036574	5672	5721	5751#																
TST61	037312	5844	5897	5921#																
TST62	040036	6022	6076	6097#																
TST63	040600	6199	6264	6282#																
TST64	041224	6356	6404	6429#																
TST65	041742	6520	6572	6593#																
TST66	042476	6700	6755	6777#																
TST67	043254	6877	6945	6967#																
TST7	023622	2931	2965#																	
TST70	044032	7068	7134	7154#																
TST71	044144	7173	7178	7189#																
TST72	044240	7205	7226#																	
T*4C = C0007!		2576#	2622#	2648#	2928#	2967#	3026#	3051#	3093#	3117#	3147#	3174#	3205#	3233#						

\$CNTLU	061102	10073	10164#											
\$CRLF	001223	295#	2775	7353	9908	9943	10084	10163	10224	10258	10272	10280	10299	10304
		10324												
\$DBLK	057566	9830	9864	9872#										
\$DOAGN	044566	7288	7297	7303#										
\$DTBL	057556	9833	9868#											
\$ENDAD	044556	183	7299#											
\$ENDCT	044524	7290#												
\$ENDMG	044575	7249	7292	7307#										
\$ENULL	044572	7252	7295	7306#										
\$EOP	044470	2603	2715	7256	7280#									
\$ECPCT	044516	7287#	7291											
\$ERFLG	001103	250#	9737	9777	9779	9785*	9806	10245*	10272					
\$ERMAX	001115	256#	2462*	9779	9801*	9806								
\$ERROR	061276	2454	2605	10238#										
\$ERRPC	001116	257#	1670	1672	1674	1677	1680	1683	1685	1698	1692	1695	1697	1699
		1702	1705	1708	1712	1715	1719	1722	1725	1728	1731	1735	2014	2016
		2018	10252*	10253*	10254	10272	10286							
\$ERRTB	001226	312#	10294											
\$ERTY	061444	10257	10279#											
\$ERTTL	001112	254#	7241	7243*	10251*	10272								
\$ESCAP	001214	292#	2461*	9800*	10267	10269	10272							
\$FILLC	001156	275#	9912	9943										
\$FILLS	001155	274#	9943											
\$GADR	001120	258#												
\$GODAT	001124	260#	1670	1672	1674	1697	1705	1708	1712	1715	1719	2016	2018	2655*
		3100*	3107	3124*	3136	3154*	3163	3191*	3194	3212*	3222	3240*	3253	3288*
		3358*	3359*	3370	3400*	3411	3438*	3455	3480*	3482	3485	3542*	3544	3547
		3589*	3593	3598*	3600	3645*	3653*	3676*	3685	3711*	3719	3748*	3750	3757
		3779*	3790	3812*	3823	3852*	3862	3882*	3896	3922*	3931	3951*	3960	3984*
		3995	4014*	4026*	4045*	4062	4085*	4103	4204*	7446*	7448*	7450	7455	7773*
		8094*	8105*	8115*	8124*	8194*	8892*	8921*	8940*	8942*	8969*	9458*	9469*	9492*
		9507*	9521*	9534*	9548*									
		7296#												
\$GET42	044546													
\$GTSWR	060414	10049#	10435											
\$HD	= 000000	12												
\$HIOCT	061274	10212*	10223#											
\$ICNT	001104	251#	9792*	9793	9795*	9805								
\$ILLUP	062256	10448	10464	10483#										
\$INTAG	001135	265#	10046*	10065	10085	10170								
\$ITEMB	001114	255#	10254*	10272	10283									
\$LF	001224	296#	9943	10154	10163	10224	10272							
\$LPADR	001106	252#	2463*	2805*	7368*	7379	9783*	9798*	9803	9805				
\$LPERR	001110	253#	2464*	7351	7378*	9783	9799*	9805	10266					
\$MAIL	= ***** U	2481	2542	9798	9896	10260								
\$MNEW	061125	10052	10168#											
\$MSWR	061114	10049	10166#											
\$MXCNT	057350	9796	9805#											
\$NULL	001154	273#	9914	9943										
\$NWTST	= 000001	2567#	2569	2612#	2614	2643#	2661#	2794#	2796	2919#	2921	2956#	2958	3021#
		3046#	3088#	3112#	3142#	3169#	3200#	3228#	3271#	3313#	3383#	3427#	3466#	3526#
		3573#	3632#	3666#	3693#	3728#	3767#	3800#	3833#	3871#	3910#	3939#	3972#	4003#
		4035#	4074#	4110#	4128#	4130	4223#	4225	4398#	4400	4573#	4575	4718#	4720
		4894#	4896	5044#	5046	5217#	5219	5397#	5399	5589#	5591	5740#	5742	5910#

		5912	6086#	6088	6273#	6275	6418#	6420	6581#	6583	6765#	6767	6955#	6957
		7146#	7148	7183#	7185	7217#	7219							
\$OCNT	062042	10360*	10389*	10402#										
\$OMODE	062044	10355*	10359*	10364	10367*	10378*	10404#							
\$OVER	057234	9760	9776	9784	9794	9802#								
\$PASS	001100	248#	3324	7248*	7250	7284*	7285*	7293	7306	9790	9876			
\$PCWER	062274	10479	10486#											
\$PWRAD	062262	10481#												
\$PWRDN	062122	2458	10448#	10476										
\$PWRMG	062256	10479#												
\$PWRUP	062174	10458	10464#											
\$QLES	001222	294#	9943	10103	10147	10163	10221	10224	10272					
\$RDCHR	060666	10116#	10438											
\$RDDEC=	***** U	10441												
\$RDLIN	060756	10139#	10439											
\$RDOCT	061136	10185#	10440											
\$RDSZ =	000011	10132#												
\$REGAD	001160	277#												
\$REGO	001162	279#												
\$REG1	001164	280#												
\$REG2	001166	281#												
\$REG3	001170	282#												
\$REG4	001172	283#												
\$REG5	001174	284#												
\$RTNAD	044570	7305#												
\$R2A =	***** U	10441												
\$SAVRE=	***** U	10441												
\$SAVR6	062272	10457*	10465	10466*	10467*	10485#								
\$SCOPE	057060	2452	9746#											
\$SETUP=	000117	2443#	2451	2452	2454	2456	2458	2460	2461	2463	2539	7282	9747	9986
		9991	9992	10022	10170	10239	10263	10271						
\$SS1 =	000000	2482#												
\$STUP =	177777	2443#												
\$SVLAD	057306	9768	9797#											
\$SVPC =	000200	181#	186											
\$SWR =	167770	2#	12	44	45	46	47	48	49	50	51	291	292	293
		2460	2461	2463	2464	2574	2620	2647	2665	2804	2928	2966	3025	3050
		3092	3116	3146	3173	3204	3232	3275	3317	3387	3431	3470	3530	3577
		3636	3670	3697	3732	3771	3804	3837	3875	3914	3943	3976	4007	4039
		4078	4114	4138	4237	4415	4585	4732	4904	5056	5229	5409	5599	5752
		5922	6098	6283	6430	6594	6778	6968	7155	7190	7227	7277	7283	7298
		7304	7306	9738	9739	9740	9741	9742	9759	9771	9773	9774	9777	9779
		9779	9786	9787	9788	9799	9802	9805	10230	10231	10232	10233	10234	10249
\$SWRMK=	000000	10255	10260	10264	10272	10482								
\$TIMES	001212	51	52	9742	9743	9775								
		291#	2460*	2574*	2620*	2647*	2665*	2804*	3050*	3092*	3116*	3146*	3173*	3204*
		3232*	3275*	3317*	3387*	3431*	3470*	3530*	3577*	3636*	3670*	3697*	3732*	3771*
		3804*	3837*	3875*	3914*	3943*	3976*	4007*	4039*	4078*	7227*	7283*	9786*	9793
		9796*	9805											
\$TKB	001146	270#	9346	9967	9978	10003	10031	10058						
\$TKCNT	060016	9947#	9962*	9992	10009*	10123	10125*							
\$TKINT	060036	2486	2667	8553	9962#	9983	10044							
\$TKQEN=	060035	9951#	10017	10128										
\$TKQIN	060020	9948#	9963*	9964	10015*	10016*	10017	10019*						

STKQOU	060022	9949#	9964*	10126	10127*	10128	10130*							
STKQSR	060024	9950#	9963	10019	10130									
STKS	001144	269#	4172	4173	9946	9968*	9999*	10001	10007*	10029	10045*	10055	10067*	10087*
STKSRV	060106	9965	9978#											
STMPO	001176	285#	1685	2730*	2731*	3056*	3072	3290*	3299	3365*	3406*	3446*	3647*	3657
		3990*	4016*	4029	4047*	4067	4291*	4292	4467*	4468	4637*	4638	7638*	7644
		7649	7653	7764*	7778	7880*	7926*							
STMP1	001200	286#	1683	1685	1699	2588*	7165*	7166*	7200*	7201*	7639*	7644	7649	7765*
		7780												
STMP2	001202	287#	1699	7642*	7646*	7648*	7651*	7882*	7924*					
STMP3	001204	288#	1685	1699	7636*	7637*	7881*	7883	7884*					
STMP4	001206	289#	7883*	7907*										
STMP5	001210	290#	7879*	7885*	7891	9008*	8087	8109						
STN =	000073	2#	12	2554	2567	2574#	2576	2612	2620#	2622	2628	2643	2647#	2648
		2654	2661	2665#	2790	2794	2804#	2919	2928#	2931	2956	2966#	2967	3021
		3025#	3026	3046	3050#	3088	3092#	3093	3108	3112	3116#	3117	3137	3142
		3146#	3147	3164	3169	3173#	3174	3195	3200	3204#	3205	3223	3228	3232#
		3233	3254	3271	3275#	3276	3313	3317#	3318	3375	3383	3387#	3388	3416
		3427	3431#	3432	3456	3466	3470#	3471	3491	3526	3530#	3531	3554	3573
		3577#	3578	3607	3632	3636#	3637	3666	3670#	3671	3686	3693	3697#	3698
		3704	3720	3728	3732#	3733	3739	3758	3767	3771#	3772	3791	3800	3804#
		3805	3824	3833	3837#	3838	3844	3863	3871	3875#	3876	3897	3899	3910
		3914#	3915	3932	3939	3943#	3944	3961	3972	3976#	3977	3996	4003	4007#
		4008	4035	4039#	4040	4074	4078#	4079	4104	4110	4114#	4115	4128	4138#
		4140	4146	4197	4219	4223	4237#	4238	4297	4398	4415#	4416	4474	4573
		4585#	4586	4643	4718	4732#	4733	4851	4885	4894	4904#	4906	4977	5024
		5044	5056#	5058	5148	5203	5217	5229#	5231	5331	5386	5397	5409#	5411
		5507	5573	5589	5599#	5601	5672	5720	5740	5752#	5754	5844	5896	5910
		5922#	5924	6022	6075	6086	6098#	6100	6199	6263	6273	6283#	6285	6356
		6403	6418	6430#	6431	6520	6571	6581	6594#	6595	6700	6754	6765	6778#
		6779	6877	6944	6955	6968#	6970	7068	7133	7146	7155#	7156	7173	7178
		7183	7190#	7191	7205	7217	7227#							
STPB	001152	272#	9932*	9943										
STPFLG	001157	276#	9890	9943										
STPS	001150	271#	9930	9943										
STRAP	062046	2456	10413#											
STRP =	000015	10421#	10430#	10431#	10432#	10433#	10434#	10435	10436#	10437	10438#	10439#	10440#	10441#
		10442#	10443#	10444#										
STRPAD	062070	10418	10428#											
STSTNM	001102	249#	7244*	7282*	9737	9775	9797*	9802	9806	10247	10272			
STTYIN	061064	10140	10141	10158	10162#									
STYPBN=	***** U	10434												
STYPDS	057352	9818#	10433											
STYPE	057576	9890#	10421	10429										
STYPEC	057746	9911	9918	9925	9930#	9931	10089							
STYPEX	060014	9936	9938	9941#										
STYPOC	061644	10358#	10430											
STYPON	061660	10357	10360#	10432										
STYPOS	061620	10353#	10431											
SXTSTR	057112	9762#												
SSGET4=	000000	7298#												
SOFILL	062043	10354*	10358*	10368	10403#									
S4OCAT=	***** U	9759	10257											
.	= 062304	169#	173#	181	182#	184#	186#	190#	193#	239#	246#	297	1669#	1804#

2012#	2344#	2411#	2412#	2413#	2449	2463	2464	2497#	2502#	2511#	2515#	2527#
2595#	2674#	2701#	2705#	2760#	2766#	2772#	2852#	2898#	2904#	2910#	2936#	2940#
2951#	3342#	7234#	7240#	7306	7310	7357#	7365#	7372#	7376#	8564#	9207#	9208#
9209#	9210#	9211#	9212#	9214#	9215#	9216#	9217#	9218#	9720#	9805	9806	9872#
9943	9946	9950#	9951	9952#	10162#	10163	10170	10224	10272	10327#	10460	10484

.SCATC	2#	167
.SCMTA	2#	240
.SEOP	2#	7272
.SERRO	2#	10224
.SERRT	2#	10272
.SPOWE	2#	10444
.SRDOC	2#	10171
.SREAD	2#	9943
.SSCOP	2#	9732
.STRAP	2#	10405
.STYPD	2#	9806
.STYPE	2#	9873
.STYPO	2#	10328

ADD	3011	3065	3285	4397	4562	4708	7367	7435	7487	7582	7621	7891	7898	7905	9838
AS	9900	10064	10074	10093	10209	10294	10356	10366							
ASUB	10097	10098	10099	10201	10203	10205	10291	10292	10293	10417					
ASUB	9843														
ASUB	8323	8332	8340	8348											
ASUB	3355	3398	8516	9000	9013	9030	9844								
ASUB	3550	3897													
ASUB	2876	2554	2627	2654	2689	2693	2736	2738	2743	2745	2748	2750	2790	2828	2974
ASUB	3456	2879	2881	2884	2986	2931	3108	3137	3164	3195	3223	3254	3280	3376	3416
ASUB	3961	3496	3548	3601	3686	3703	3720	3738	3759	3791	3824	3843	3863	3897	3932
ASUB	5159	3996	4063	4104	4145	4203	4219	4293	4463	4518	4639	4667	4983	4987	5151
ASUB	5351	5334	5343	5510	5519	5531	5678	5682	5847	5855	6025	6034	6202	6211	6223
ASUB	7260	6365	6523	6531	6703	6712	6880	6889	6901	7071	7080	7092	7179	7246	7256
ASUB	8110	7297	7456	7500	7540	7564	7771	7785	7887	7894	7901	8086	8088	8092	8103
ASUB	8931	8113	8123	8193	8295	8316	8383	8390	8397	8403	8470	8612	8782	8910	8919
ASUB	9154	8936	8939	8961	8963	9005	9007	9017	9022	9024	9034	9074	9084	9090	9100
ASUB	9778	9162	9431	9441	9455	9467	9489	9503	9517	9530	9544	9619	9595	9774	9776
ASUB	10296	9782	9791	9903	9938	9989	10042	10081	10096	10124	10196	10246	10249	10265	10268
ASUB	9794	10301	10307	10318	10383										
ASUB	7288	9852	10013	10093	10198	10390									
ASUB	9780														
ASUB	8457														
ASUB	2560	3368	3409	3453	3683	3717	3755	3788	3821	3860	3894	3929	3958	3993	4060
ASUB	4101	4217	4316	4319	4329	4492	4495	4505	4527	4659	4677	5157	5170	5341	5517
ASUB	5953	5866	6032	6209	6529	6542	6654	6655	6710	6887	7078	7285	7448	7453	7562
ASUB	7783	7911	7916	7921	8309	8362	9368	8377	8395	8480	8488	8557	8714	8722	9867
ASUB	9971	8975	9266	9275	9610	9612	9617	9637	9639	9655	9657	9568	9979	10004	10014
ASUB	10032	10059	10094	10207	10380										
ASUB	3218	3247													
ASUB	2982	2985	3359	3650	4022	4053	4192	4194	4315	4318	4329	4339	4340	4342	4491
ASUB	4494	4504	4522	4528	4530	4531	4533	4658	4676	4678	4679	4681	5169	5865	6541
ASUB	6656	7913	7918	7923	8149	8157	8195	8301	8320	8329	8337	8345	8360	8361	8363
ASUB	6366	8367	8369	8479	8487	8558	8713	8715	8865	8866	8974	9001	9003	9020	9098
ASUB	9255	9267	9427	9439	9609	9611	9616	9633	9634	9636	9638	9654	9656	9667	9846
ASUB	9847	10101	10385	10386											
ASUB	3159	3188	10283												
ASUB	2549	2590	2669	2692	2930	2946	4292	4468	4638	7499	7542	7549	7566	7571	7595
ASUB	7603	7610	7615	7644	7649	7886	7893	7900	8387	8389	8471	8611	8899	8962	8966
ASUB	9006	9023	9073	9089	9430	9440	9759	9773	9781	9788	10248	10255	10264		
ASUB	9926														
ASUB	3067	10142													
ASUB	9835	9851	9917	10011	10091	10200	10391								
ASUB	7910	7915	7920	9842											
ASUB	2449	2472	2488	2541	2543	2584	2591	2670	2684	2725	2782	2913	2816	2819	2947
ASUB	3322	3327	3371	3412	3490	3553	3606	3603	4197	4208	4221	4321	4323	4331	4497
ASUB	4499	4507	4661	4758	4781	4796	4803	4809	4816	4851	4884	4911	4954	4977	5069
ASUB	5076	5148	5172	5187	5242	5249	5331	5361	5422	5429	5507	5552	5606	5649	5672
ASUB	5765	5772	5844	5968	5884	5935	5942	6022	6057	6111	6118	6199	6245	6290	6323
ASUB	6356	6442	6449	6520	6544	6559	6605	6612	6700	6735	6790	6797	6877	6922	6981
ASUB	6988	7068	7114	7167	7202	7414	7442	7491	7543	7550	7567	7572	7597	7605	7611
ASUB	7616	7645	7647	7650	7652	7690	7777	7787	7925	7927	7958	7964	8035	8037	8388
ASUB	8458	8460	8472	8522	8524	8560	8719	8779	8787	8877	8879	8885	8990	8908	8928
ASUB	8957	8967	9039	9041	9105	9107	9146	9160	9167	9173	9180	9272	9377	9450	9452

	9474	9483	9513	9540	9641	9645	9659	9662	9665	9670	9703	9705	9711	9760	9789
	9840	9897	9905	9913	9927	9934	9981	9987	9993	9998	10006	10018	10028	10034	10062
BR	10066	10072	10079	10086	10129	10146	10152	10256	10284	10310	10381	10468			
	9826	9856	9891	9931	10002	10030	10056	10261	10379						
	2438	2474	2489	2495	2500	2505	2509	2513	2517	2521	2525	2529	2533	2545	2556
	2595	2593	2597	2672	2699	2703	2707	2711	2758	2764	2770	2773	2850	2856	2862
	2887	2896	2899	2902	2905	2908	2934	2938	2942	2949	3068	3282	3332	3336	3340
	3344	3377	3418	3491	3554	3607	3899	4057	4211	4523	4671	4885	4985	5680	6363
	7173	7205	7232	7238	7261	7342	7348	7355	7359	7363	7370	7374	7547	7554	7570
	7573	7601	7609	7614	7779	7889	7896	7903	7912	7917	7922	8063	8097	8108	8118
	8304	8370	8391	8401	8461	8463	8482	8518	8544	8550	8562	8569	8574	8579	8584
	9590	9594	8613	8614	8792	8869	8894	8924	8941	8946	8949	8964	9009	9026	9078
	9094	9434	9445	9472	9495	9510	9524	9537	9635	9718	9762	9768	9771	9784	9787
	9837	9854	9893	9910	9920	9929	9936	9996	10075	10102	10104	10148	10209	10222	10289
CLC	10313	10320	10357	10372	10393	10460	10484								
	7439	8312	9080	9095	9435	9446									
	2439	2447	2460	2461	2655	2677	2727	2728	2808	3102	3126	3156	3183	3214	3242
	3287	3288	3354	3396	3448	3540	3587	3644	3645	3676	3779	3812	3885	3922	3951
	3984	4013	4014	4020	4045	4051	4095	4158	4159	4160	4174	4175	4185	4244	4246
	4247	4248	4258	4262	4421	4423	4424	4435	4439	4592	4593	4594	4605	4609	4742
	4745	4746	4747	4750	4756	4779	4785	4786	4794	4801	4814	4839	4841	4917	4920
	4921	4922	4923	4933	4934	4949	4950	4964	4966	5013	5074	5085	5088	5089	5090
	5091	5103	5123	5131	5190	5247	5259	5262	5263	5264	5265	5277	5306	5314	5374
	5427	5438	5441	5442	5443	5444	5456	5484	5490	5561	5604	5612	5615	5616	5617
	5518	5628	5629	5644	5645	5659	5661	5709	5770	5781	5784	5785	5786	5787	5799
	5819	5827	5885	5940	5952	5955	5956	5957	5958	5970	5999	6005	6064	6116	6127
	6130	6131	6132	6133	6145	6174	6182	6252	6288	6296	6299	6300	6301	6302	6312
	6313	6328	6329	6343	6345	6392	6447	6457	6460	6461	6462	6463	6475	6495	6503
	6560	6610	6622	6625	6626	6627	6628	6640	6675	6683	6743	6795	6806	6809	6810
	6811	6812	6824	6854	6860	6933	6986	6997	7000	7001	7002	7003	7015	7043	7051
	7122	7243	7244	7282	7283	7340	7381	7392	7386	7390	7522	7711	7712	7714	7731
	7808	7909	7811	7830	7878	7879	7890	7897	7904	7961	8023	8151	8182	8365	8453
	8508	8509	8510	8519	8882	8921	8940	8954	9010	9027	9079	9095	9168	9458	9469
	9534	9705	9748	9749	9753	9757	9796	9800	9829	9832	9962	9999	10045	10053	10054
CLRB	10118	10119	10193	10194	10282	10370	10466								
CMP	2687	9785	9858	9909	9935	10153	10218								
	2448	2471	2542	2587	2735	2737	2742	2744	2747	2749	2827	2873	2875	2878	2880
	2883	2885	3066	3107	3136	3163	3194	3222	3253	3324	3370	3374	3411	3415	3455
	3485	3547	3600	3685	3719	3757	3790	3823	3862	3896	3902	3931	3960	3995	4062
	4103	4202	4218	4982	4986	5071	5158	5244	5333	5342	5424	5509	5518	5529	5577
	5581	5767	5854	5937	6024	6033	6113	6201	6210	6221	6360	6364	6444	6530	6507
	6702	6711	6792	6879	6888	6899	6983	7070	7079	7090	7175	7177	7207	7259	7455
	7538	7563	7770	7781	7784	8087	8090	8100	8109	8112	8121	8192	8396	8402	8466
	8468	8930	9488	9502	9516	9543	9769	9793	9850	9980	9986	9988	9992	9997	10005
CMPB	10010	10012	10017	10027	10033	10061	10071	10078	10090	10092	10128	10141			
	2815	3321	9775	9779	9902	9904	9912	9933	9937	10041	10065	10085	10145	10151	10197
CCM	10199														
DEC	6653	7444	7447	7449	7452	7454	8302	8321	8330	8338	8346				
	2583	2683	2724	2781	2786	2830	3132	3190	3249	3595	4320	4320	4322	4330	4496
	4498	4506	4660	4757	4780	4795	4802	4808	4815	4874	4875	4879	4880	4893	4910
	4953	5068	5075	5171	5186	5241	5248	5365	5421	5428	5551	5605	5648	5764	5771
	5867	5883	5934	5941	6056	6110	6117	6244	6289	6332	6441	6448	6543	6558	6604
	6611	6734	6789	6796	6921	6980	6987	7113	7166	7201	7255	7296	7413	7489	7490
	7646	7651	7689	7786	7924	7926	7957	7963	8459	8521	8523	9559	8718	8875	8878

	8994	8927	8956	9038	9040	9104	9106	9145	9159	9166	9172	9179	9271	9449	9451
	9473	9512	9539	9640	9644	9658	9661	9664	9669	9702	9710	10125	10290		
DECB	9916	9919	10378	10389											
EMT	60														
MALT	173	4180	4266	4443	4613	4946	4855	4960	4994	5128	5175	5311	5354	5488	5540
	5655	5689	5824	5871	6003	6045	6179	6233	6339	6372	6500	6547	6680	6723	6858
	6910	7048	7102	7735	7834	8007	8597	9723	9992	10262	10459	10483			
INC	2682	2778	2780	3128	3185	3244	3373	3414	3591	3901	4873	4876	4877	4878	4881
	7248	7284	7768	8462	8465	9018	9035	9085	9101	9792	9836	10002	10016	10100	10127
	10251	10384	10392	10467											
INCB	9797	9939	10245												
TOT	61														
JMP	191	194	2491	2603	2628	2690	2694	2715	3328	3704	3739	3844	4146	4297	4474
	4643	4886	7253	7263	7304	9461	9985	10068							
JSR	2486	2632	2667	2807	2929	2969	2976	2990	3003	3027	3029	4116	4118	4149	4163
	4170	4177	4186	4195	4249	4255	4263	4271	4288	4294	4351	4361	4378	4426	4432
	4440	4448	4464	4470	4541	4553	4596	4602	4610	4618	4634	4640	4688	4699	4734
	4787	4826	4943	4848	4853	4858	4912	4936	4957	4967	4992	5000	5007	5018	5079
	5104	5113	5125	5132	5149	5173	5191	5197	5252	5278	5296	5308	5315	5332	5346
	5352	5375	5380	5432	5457	5474	5485	5491	5508	5522	5528	5538	5562	5557	5607
	5631	5652	5662	5687	5696	5703	5714	5775	5800	5809	5821	5828	5845	5869	5886
	5890	5945	5971	5989	6000	6006	6023	6037	6043	6063	6069	6121	6146	6164	6176
	6183	6200	6214	6218	6231	6251	6257	6291	6315	6336	6346	6370	6379	6386	6397
	6451	6476	6485	6497	6504	6521	6545	6561	6565	6615	6641	6665	6677	6684	6701
	6715	6721	6742	6748	6800	6825	6844	6855	6861	6878	6892	6898	6908	6932	6938
	6991	7016	7033	7045	7052	7069	7083	7089	7100	7123	7127	7159	7170	7176	7194
	7208	7228	7299	7437	7438	7457	7537	7594	7715	7721	7732	7738	7778	7780	7813
	7819	7831	7836	7974	8003	8004	8024	8033	8045	8071	8158	8190	8399	8405	8474
	8491	8520	8553	8608	8724	8731	8789	8794	8883	8888	8897	8899	8901	8903	8905
	8917	8929	8955	8959	9019	9036	9086	9102	9157	9164	9170	9174	9177	9277	9289
	9378	9465	9485	9500	9514	9528	9541	9701	9709	9911	9918	9925	9983	10044	10089
	10257														
MOV	2437	2446	2450	2452	2453	2454	2455	2456	2457	2458	2459	2463	2464	2467	2468
	2469	2470	2475	2477	2478	2479	2484	2485	2537	2551	2561	2562	2574	2575	2576
	2577	2579	2580	2581	2582	2586	2588	2600	2605	2607	2620	2621	2622	2631	2647
	2648	2650	2651	2652	2656	2665	2675	2676	2678	2679	2681	2721	2722	2723	2726
	2729	2730	2734	2761	2767	2774	2776	2784	2785	2791	2804	2805	2814	2824	2829
	2831	2853	2859	2865	2915	2916	2928	2966	2967	2971	2973	3010	3012	3025	3026
	3028	3050	3051	3052	3054	3056	3057	3059	3061	3062	3072	3092	3093	3094	3096
	3098	3099	3100	3105	3116	3117	3118	3120	3122	3123	3124	3134	3146	3147	3148
	3150	3152	3153	3154	3158	3161	3173	3174	3175	3177	3179	3180	3181	3192	3204
	3205	3206	3208	3210	3211	3212	3216	3220	3232	3233	3234	3236	3239	3239	3240
	3251	3275	3276	3277	3281	3284	3286	3290	3291	3293	3299	3317	3318	3319	3348
	3350	3351	3353	3358	3361	3363	3365	3367	3387	3388	3389	3391	3393	3394	3395
	3400	3402	3404	3406	3408	3431	3432	3433	3435	3437	3438	3440	3442	3444	3446
	3449	3450	3452	3470	3471	3472	3474	3476	3477	3478	3480	3482	3483	3530	3531
	3532	3534	3536	3537	3538	3542	3545	3577	3578	3579	3581	3593	3594	3585	3589
	3597	3636	3637	3638	3640	3642	3643	3647	3648	3653	3657	3670	3671	3672	3674
	3675	3678	3679	3682	3697	3698	3706	3708	3709	3710	3711	3713	3715	3732	3733
	3742	3744	3746	3747	3748	3750	3754	3771	3772	3773	3775	3777	3778	3781	3795
	3787	3804	3805	3806	3808	3810	3811	3814	3818	3820	3837	3838	3847	3849	3851
	3852	3854	3856	3857	3859	3875	3876	3877	3879	3881	3882	3884	3899	3890	3891
	3893	3914	3915	3916	3918	3920	3921	3924	3926	3928	3943	3944	3945	3947	3949
	3950	3953	3955	3957	3976	3977	3978	3980	3982	3983	3986	3989	3990	3992	4007

4008	4009	4011	4012	4016	4017	4019	4026	4029	4039	4040	4041	4043	4044	4047
4048	4050	4059	4067	4078	4079	4080	4082	4084	4085	4087	4089	4091	4093	4097
4098	4100	4114	4115	4117	4139	4140	4156	4161	4162	4171	4172	4173	4175	4181
4183	4198	4199	4200	4201	4204	4205	4216	4237	4238	4242	4256	4257	4260	4267
4283	4284	4291	4312	4313	4314	4317	4327	4337	4338	4341	4343	4344	4345	4386
4388	4415	4416	4419	4425	4433	4434	4437	4444	4459	4460	4457	4488	4489	4490
4493	4503	4512	4520	4521	4525	4526	4532	4534	4535	4536	4561	4563	4585	4586
4590	4595	4603	4604	4607	4614	4629	4630	4637	4657	4669	4670	4673	4674	4680
4682	4683	4684	4707	4709	4732	4733	4736	4743	4744	4748	4749	4754	4755	4762
4765	4767	4768	4769	4778	4782	4783	4784	4793	4798	4800	4805	4806	4807	4813
4820	4822	4823	4824	4825	4832	4833	4834	4835	4838	4904	4906	4907	4908	4909
4915	4918	4919	4930	4932	4935	4942	4943	4944	4946	4948	4951	4952	4955	4962
4965	5005	5006	5056	5058	5064	5065	5066	5067	5072	5084	5086	5087	5096	5100
5101	5102	5114	5115	5118	5119	5130	5156	5169	5176	5177	5180	5181	5182	5183
5194	5185	5229	5231	5237	5238	5239	5240	5245	5258	5260	5261	5270	5274	5275
5276	5288	5291	5297	5298	5301	5302	5313	5340	5355	5356	5359	5360	5361	5362
5363	5364	5370	5409	5411	5417	5418	5419	5420	5425	5437	5439	5440	5449	5453
5454	5455	5466	5469	5475	5476	5479	5480	5489	5516	5541	5542	5545	5546	5547
5548	5549	5550	5557	5599	5601	5602	5603	5611	5613	5614	5625	5627	5630	5637
5638	5639	5641	5643	5646	5647	5650	5657	5660	5701	5702	5752	5754	5750	5761
5762	5763	5768	5780	5782	5783	5792	5796	5797	5798	5810	5811	5814	5815	5826
5852	5864	5872	5873	5876	5877	5878	5879	5880	5882	5922	5924	5930	5931	5932
5933	5938	5951	5953	5954	5963	5967	5968	5969	5981	5984	5990	5991	5994	5995
6004	6031	6046	6047	6050	6051	6052	6053	6054	6055	6062	6098	6100	6106	6107
6108	6109	6114	6126	6128	6129	6138	6142	6143	6144	6156	6159	6165	6166	6169
6170	6181	6208	6234	6235	6238	6239	6240	6241	6242	6243	6250	6283	6285	6286
6287	6295	6297	6298	6309	6311	6314	6321	6322	6323	6325	6327	6330	6331	6334
6341	6344	6384	6385	6430	6431	6437	6438	6439	6440	6445	6456	6458	6459	6468
6472	6473	6474	6486	6487	6490	6491	6502	6528	6540	6549	6549	6552	6553	6554
6555	6556	6557	6594	6595	6600	6601	6602	6603	6608	6621	6623	6624	6633	6637
6638	6639	6652	6660	6666	6667	6670	6671	6682	6709	6724	6725	6728	6729	6730
6731	6732	6733	6778	6779	6785	6786	6787	6788	6793	6805	6807	6808	6817	6821
6822	6823	6835	6837	6839	6845	6846	6849	6850	6859	6886	6911	6912	6915	6916
6917	6918	6919	6920	6927	6929	6968	6970	6976	6977	6978	6979	6984	6996	6998
6999	7008	7012	7013	7014	7026	7027	7028	7034	7035	7038	7039	7050	7077	7103
7104	7107	7108	7109	7110	7111	7112	7119	7120	7156	7157	7160	7161	7162	7163
7164	7165	7191	7192	7195	7196	7197	7198	7199	7200	7227	7230	7235	7241	7250
7257	7258	7262	7289	7293	7296	7345	7351	7368	7378	7379	7406	7407	7408	7409
7410	7411	7412	7415	7416	7417	7431	7432	7433	7434	7436	7445	7446	7450	7451
7480	7481	7482	7483	7484	7485	7488	7492	7493	7494	7501	7515	7516	7517	7518
7520	7521	7535	7544	7551	7555	7561	7568	7573	7576	7592	7598	7606	7612	7617
7634	7635	7636	7638	7639	7640	7641	7642	7648	7653	7682	7683	7684	7685	7686
7687	7688	7691	7692	7693	7708	7713	7723	7724	7727	7728	7736	7756	7757	7759
7759	7760	7761	7762	7763	7764	7765	7766	7767	7769	7773	7774	7782	7789	7790
7791	7792	7793	7805	7810	7822	7823	7826	7827	7835	7872	7873	7874	7875	7876
7877	7880	7881	7882	7883	7888	7895	7902	7928	7929	7930	7931	7932	7933	7950
7951	7952	7953	7954	7955	7956	7959	7968	7971	7972	7973	7977	7978	7979	8001
8008	8009	8010	8015	8016	8018	8019	8021	8089	8093	8094	8099	8104	8105	8111
8114	8115	8119	8124	8125	8142	8143	8144	8146	8147	8148	8150	8152	8156	8160
8161	8177	8178	8180	8181	8183	8184	8191	8194	8195	8199	8286	8287	8288	8289
8290	8291	8292	8293	8300	8303	8308	8310	8319	8322	8328	8331	8336	8339	8344
8347	8378	8379	8394	8426	8427	8428	8429	8430	8431	8447	8449	8450	8451	8452
8491	8502	8503	8504	8505	8506	8507	8511	8512	8513	8514	8517	8525	8526	8527
8528	8529	8530	8531	8532	8547	8555	8556	8565	8572	8591	8587	8609	8703	8705

	8706	8707	8708	8709	8710	8712	8717	8721	8723	8727	8728	8729	8730	8799	8800
	8801	8802	8803	8804	8855	8856	8957	8858	8859	8860	8862	8863	8864	8870	8873
	8881	8887	8891	8892	8893	8896	8998	8900	8902	8904	8914	8915	8916	8922	8923
	8926	8942	8943	8944	8945	8948	8953	8958	8968	8969	8970	8972	8996	8997	8998
	9002	9009	9011	9014	9015	9025	9028	9031	9032	9042	9065	9066	9067	9068	9069
	9070	9071	9072	9075	9082	9091	9098	9108	9109	9110	9111	9112	9134	9135	9136
	9137	9138	9139	9140	9141	9142	9143	9144	9148	9149	9151	9152	9155	9156	9158
	9165	9169	9171	9175	9176	9178	9181	9182	9183	9184	9185	9255	9257	9258	9259
	9260	9261	9262	9264	9269	9274	9276	9281	9282	9283	9284	9285	9286	9287	9384
	9385	9386	9387	9388	9389	9416	9417	9418	9419	9420	9421	9423	9424	9425	9426
	9428	9437	9438	9448	9453	9457	9459	9460	9463	9464	9466	9470	9471	9487	9490
	9492	9493	9494	9497	9498	9499	9501	9507	9508	9509	9515	9520	9521	9522	9523
	9526	9527	9529	9535	9536	9542	9546	9547	9548	9549	9552	9598	9599	9600	9601
	9602	9603	9604	9605	9606	9608	9631	9632	9643	9651	9652	9653	9673	9674	9675
	9676	9677	9678	9691	9692	9693	9696	9697	9699	9700	9707	9708	9713	9722	9764
	9765	9767	9770	9783	9795	9796	9798	9799	9802	9803	9819	9820	9821	9822	9823
	9824	9825	9830	9833	9853	9859	9860	9861	9862	9863	9865	9866	9894	9895	9899
	9914	9963	9964	9965	9966	9968	10007	10019	10050	10067	10082	10087	10116	10117	10120
	10130	10139	10140	10155	10156	10157	10158	10185	10186	10187	10188	10189	10191	10192	10211
	10212	10213	10214	10215	10241	10247	10252	10266	10269	10281	10286	10295	10300	10305	10306
	10308	10311	10315	10322	10323	10353	10361	10362	10363	10369	10376	10394	10395	10396	10397
	10398	10413	10414	10418	10448	10449	10450	10451	10452	10453	10454	10455	10456	10457	10458
	10464	10465	10469	10470	10471	10472	10473	10474	10475	10476	10477	10480			
MOV8	2462	2546	2685	2848	2998	3104	3130	3187	3246	3544	3593	4357	4547	4694	5098
	5099	5116	5117	5178	5179	5272	5273	5299	5300	5357	5358	5451	5452	5477	5479
	5543	5544	5794	5795	5812	5813	5874	5875	5965	5966	5992	5993	6048	6049	6140
	6141	6167	6168	6236	6237	6470	6471	6488	6489	6550	6551	6635	6636	6668	6669
	6726	6727	6819	6820	6847	6848	6913	6914	7010	7011	7036	7037	7105	7106	7709
	7710	7725	7726	7806	7807	7824	7825	7969	7970	8011	8013	9801	9828	9831	9845
	9848	9857	9896	9924	9932	9978	10003	10015	10031	10046	10058	10126	10144	10149	10195
	10254	10354	10355	10358	10359	10360	10364	10367	10368	10387	10416				
NEG	9827	10365													
NOP	3651	3652	3751	3752	3782	3783	3815	3816	4023	4024	4054	4055	7300	7301	7302
	8186	8187	8188	8189	8727	8743	8751	8764	9281	9300	9311	9325	9339	9349	9362
RESET	2443	2666	3680	7298											
ROL	7440	10202	10204	10206	10371	10373	10374	10375	10377						
ROR	2731	3355	3397	3886	7884	7885	7907	7908	9313	8314	8358	8359	8375	8376	8515
	8999	9012	9029	9077	9081	9093	9097	9433	9436	9444	9447				
RTI	2476	7391	7502	7655	9804	9867	9901	10008	10020	10088	10121	10131	10159	10216	10271
	10399	10482													
RTS	2636	3017	4371	4392	4567	4713	4866	4868	5028	5037	5207	5211	5390	5394	5577
	5581	5724	5733	5900	5904	6079	6083	6267	6271	6407	6416	6575	6579	6759	6762
	6948	6952	7137	7141	7418	7443	7459	7495	7523	7580	7583	7619	7622	7694	7741
	7794	7839	7934	7980	8053	8061	8079	8084	8129	8162	8200	8432	8492	8533	8805
	8973	9043	9113	9186	9390	9553	9679	9714	9941	9969	10325	10419			
SEC	9076	9092	9432	9443											
SUB	4206	7486	7536	7593	7637	7775	8002	8179	8443	8704	8920	9150	9256	9468	9506
	9533	9834	10253												
SWAB	3598														
TRAP	10421	10430	10431	10432	10433	10435	10437	10438	10439	10440	10441	10442	10443		
TS*	2487	2540	2553	2626	2789	2812	2817	2825	2913	2914	3063	3279	3295	3499	3552
	3605	3702	3737	3842	4144	4196	4207	4215	4517	4666	4791	4950	4975	5070	5147
	5150	5243	5330	5423	5506	5670	5766	5843	5846	5936	6021	6112	6198	6354	6443
	6519	6522	6606	6699	6791	6876	6982	7067	7245	7441	7776	7909	7914	7919	9034

	9036	9095	9294	8315	9380	8457	8610	8778	8791	8783	8907	8909	8918	8932	8939
	9960	9004	9016	9021	9033	9083	9099	9153	9161	9376	9454	9482	9618	9694	9704
	9766	9790	9839	9849	9898	9906	9928	9967	9984	9995	10000	10043	10080	10095	10123
	10210	10217	10260	10267	10317	10382	10415								
TSTB	2653	2689	8012	8014	9777	9841	9855	9890	9930	10001	10029	10055	10309		
.ASCII	294	295	1001	1006	1013	1021	1025	1030	1054	1079	1094	1103	1110	1144	1154
	1163	1177	1208	1226	1235	1237	1247	1264	1278	1292	1312	1338	1355	1375	1395
	1409	1420	1440	1462	1479	1495	1517	1541	1563	1582	1604	1621	1647	1976	1986
	2000														
.ASCII2	293	296	953	962	966	972	977	980	987	990	994	997	1041	1048	1058
	1075	1084	1118	1126	1134	1167	1189	1198	1218	1256	1271	1285	1302	1323	1334
	1346	1365	1385	1402	1415	1430	1451	1471	1487	1506	1529	1552	1573	1593	1613
	1634	1657	1805	1815	1823	1829	1834	1839	1845	1852	1856	1863	1870	1877	1880
	1884	1889	1900	1908	1917	1926	1932	1942	1950	1960	1969	1981	1993	2006	2497
	2502	2507	2511	2515	2519	2523	2527	2531	2535	2558	2595	2599	2674	2701	2705
	2709	2713	2760	2766	2772	2852	2858	2864	2898	2904	2910	2936	2940	2944	2951
	3334	3338	3342	3346	7234	7240	7307	7344	7350	7357	7361	7365	7372	7376	8546
	8552	8564	8570	8576	8580	8586	8592	8596	9720	10163	10164	10165	10166	10168	10326
	10486														
.BLKB	9950	10162													
.BLKW	2344	2412	2413	9207	9208	9209	9210	9211	9212	9214	9215	9216	9217	9218	9872
.BYTE	249	250	255	256	264	265	273	274	275	276	1742	1744	1746	1749	1752
	1755	1756	1758	1761	1765	1767	1769	1772	1775	1777	1780	1783	1786	1789	1792
	1795	1797	1800	2020	2021	2023	2429	7306	10160	10161	10400	10401	10402	10403	
.DSABL	10105														
.ENABL	2	9946													
.END	10490														
.ENDC	7														
	186	18	24	32	38	48	50	51	52	60	152	166	176	180	184
	2031	2065	2067	2411	2443	2450	2451	2454	2456	2458	2460	2461	2463	2465	2481
	2497	2502	2507	2511	2515	2519	2523	2527	2531	2535	2542	2548	2558	2568	2569
	2572	2573	2574	2575	2595	2599	2613	2614	2618	2619	2620	2621	2644	2645	2646
	2647	2648	2662	2663	2664	2665	2666	2674	2701	2705	2709	2713	2742	2752	2760
	2766	2772	2795	2796	2802	2803	2804	2805	2806	2836	2846	2852	2858	2864	2873
	2898	2904	2910	2912	2920	2921	2926	2927	2928	2936	2940	2944	2951	2957	2958
	2964	2965	2966	3022	3023	3024	3025	3047	3048	3049	3050	3051	3089	3090	3091
	3092	3093	3113	3114	3115	3116	3117	3143	3144	3145	3146	3147	3170	3171	3172
	3173	3174	3201	3202	3203	3204	3205	3229	3230	3231	3232	3233	3272	3273	3274
	3275	3276	3314	3315	3316	3317	3318	3334	3338	3342	3346	3384	3385	3386	3387
	3388	3428	3429	3430	3431	3432	3467	3468	3469	3470	3471	3527	3528	3529	3530
	3531	3574	3575	3576	3577	3578	3633	3634	3635	3636	3637	3667	3668	3669	3670
	3671	3694	3695	3696	3697	3698	3729	3730	3731	3732	3733	3758	3769	3770	3771
	3772	3801	3802	3803	3804	3805	3834	3835	3836	3837	3838	3872	3873	3874	3875
	3876	3911	3912	3913	3914	3915	3940	3941	3942	3943	3944	3973	3974	3975	3976
	3977	4004	4005	4006	4007	4008	4036	4037	4038	4039	4040	4075	4076	4077	4079
	4079	4111	4112	4113	4114	4129	4130	4136	4137	4138	4224	4225	4235	4236	4237
	4399	4400	4413	4414	4415	4574	4575	4583	4584	4585	4719	4720	4730	4731	4732
	4895	4896	4902	4903	4904	5045	5046	5054	5055	5056	5218	5219	5227	5228	5229
	5398	5399	5407	5408	5409	5590	5591	5597	5598	5599	5741	5742	5750	5751	5752
	5911	5912	5920	5921	5922	6087	6088	6096	6097	6098	6274	6275	6281	6282	6283
	6419	6420	6428	6429	6430	6582	6583	6592	6593	6594	6766	6767	6776	6777	6778
	6956	6957	6966	6967	6968	7147	7148	7153	7154	7155	7184	7185	7188	7189	7190
	7217	7218	7219	7225	7226	7227	7228	7234	7240	7275	7276	7277	7279	7282	7289
	7291	7292	7296	7298	7304	7306	7307	7310	7316	7335	7344	7350	7357	7361	7365

	7372	7376	7396	7404	7423	7426	7465	7478	7529	7533	7587	7591	7626	7631	7672
	7680	7702	7705	7746	7754	7798	7801	7844	7870	7938	7948	7985	8000	8133	8140
	8167	8175	8206	8209	8436	8441	8496	8499	8538	8541	8546	8552	8564	8570	8576
	8580	8586	8592	8596	8617	8618	8677	8680	8809	8833	8840	8842	8983	8985	8951
	9053	9121	9123	9199	9200	9226	9228	9393	9395	9563	9565	9583	9685	9716	9720
	9725	9735	9738	9743	9759	9761	9772	9775	9776	9777	9779	9791	9788	9792	9797
	9798	9802	9805	9806	9809	9876	9896	9946	9977	9986	9990	10021	10023	10038	10069
	10105	10109	10120	10132	10133	10140	10142	10145	10147	10163	10164	10170	10174	10180	10224
	10227	10230	10245	10252	10257	10258	10259	10260	10271	10272	10275	10290	10328	10331	10408
	10414	10417	10429	10430	10431	10432	10433	10434	10435	10436	10437	10438	10439	10440	10441
	10442	10443	10447	10456	10457	10463	10469	10470	10480	10482	10489				
.EQUIV	60	61	69	84	85	114	115	116	117	118	119	120	121	122	123
	142	143	144	145	146	147	148	149	150	151					
.EVEN	1668	1804	2012	2497	2502	2507	2511	2515	2519	2523	2527	2531	2535	2558	2595
	2599	2674	2701	2705	2709	2713	2760	2766	2772	2852	2858	2864	2898	2904	2910
	2936	2940	2944	2951	3334	3338	3342	3346	7234	7240	7344	7350	7357	7361	7365
	7372	7376	8546	8552	8564	8570	8576	8580	8586	8592	8596	9720	9952	10327	10488
.IF	3	15	21	31	37	47	49	50	51	52	58	124	152	176	179
	182	184	215	226	237	242	246	248	277	285	291	292	293	297	298
	2028	2030	2064	2066	2411	2443	2445	2450	2452	2454	2456	2458	2460	2461	2463
	2481	2496	2501	2506	2510	2514	2518	2522	2526	2530	2534	2539	2542	2557	2567
	2569	2572	2574	2575	2594	2598	2612	2614	2618	2620	2621	2643	2645	2647	2648
	2661	2663	2665	2666	2673	2700	2704	2708	2712	2741	2751	2759	2765	2771	2794
	2796	2802	2804	2805	2806	2835	2845	2851	2857	2863	2872	2897	2903	2909	2911
	2919	2921	2926	2928	2935	2939	2943	2950	2956	2958	2964	2966	3021	3023	3025
	3046	3048	3050	3051	3088	3090	3092	3093	3112	3114	3116	3117	3142	3144	3146
	3147	3169	3171	3173	3174	3200	3202	3204	3205	3228	3230	3232	3233	3271	3273
	3275	3276	3313	3315	3317	3318	3333	3337	3341	3345	3383	3385	3387	3388	3427
	3429	3431	3432	3466	3468	3470	3471	3526	3528	3530	3531	3573	3575	3577	3578
	3632	3634	3636	3637	3666	3668	3670	3671	3693	3695	3697	3698	3728	3730	3732
	3733	3767	3769	3771	3772	3800	3802	3804	3805	3833	3835	3837	3838	3871	3873
	3875	3876	3910	3912	3914	3915	3939	3941	3943	3944	3972	3974	3976	3977	4003
	4005	4007	4008	4035	4037	4039	4040	4074	4076	4078	4079	4110	4112	4114	4128
	4130	4136	4138	4223	4225	4235	4237	4398	4400	4413	4415	4573	4575	4583	4585
	4718	4720	4730	4732	4894	4896	4902	4904	5044	5046	5054	5056	5217	5219	5227
	5229	5397	5399	5407	5409	5589	5591	5597	5599	5740	5742	5750	5752	5910	5912
	5920	5922	6086	6088	6096	6098	6273	6275	6281	6283	6418	6420	6428	6430	6581
	6583	6592	6594	6765	6767	6776	6778	6955	6957	6966	6968	7146	7148	7153	7155
	7183	7185	7188	7190	7216	7217	7219	7225	7227	7228	7233	7239	7274	7275	7276
	7277	7278	7279	7281	7287	7290	7292	7296	7298	7304	7306	7307	7315	7334	7343
	7349	7356	7360	7364	7371	7375	7395	7403	7422	7425	7464	7477	7528	7532	7586
	7590	7625	7630	7671	7679	7701	7704	7745	7753	7797	7800	7843	7869	7937	7947
	7984	7999	8132	8139	8166	8174	8205	8208	8435	8440	8495	8498	8537	8540	8545
	8551	8563	8569	8575	8579	8585	8591	8595	8616	8617	8676	8679	9808	8832	8839
	8841	8982	8984	9050	9052	9120	9122	9198	9199	9225	9227	9392	9394	9562	9564
	9682	9684	9715	9719	9724	9734	9737	9742	9748	9759	9771	9773	9774	9775	9777
	9778	9779	9788	9790	9798	9799	9804	9805	9806	9808	9875	9896	9945	9947	9975
	9980	9986	10022	10023	10037	10061	10108	10109	10119	10132	10140	10141	10145	10146	10162
	10163	10170	10173	10176	10192	10226	10229	10240	10248	10255	10257	10258	10260	10264	10271
	10272	10274	10289	10305	10330	10407	10413	10417	10421	10430	10431	10432	10433	10434	10435
	10437	10438	10439	10440	10441	10442	10443	10446	10456	10457	10462	10469	10470	10478	10480
	10482	10486													
.IFF	32	38	47	50	51	52	60	180	184	243	246	248	277	298	2029
	2031	2065	2067	2450	2568	2569	2573	2574	2575	2613	2614	2619	2620	2621	2644

	2645	2646	2647	2648	2662	2663	2664	2665	2666	2742	2752	2795	2796	2803	2804
	2805	2836	2846	2873	2912	2920	2921	2927	2928	2957	2958	2965	2966	3022	3023
	3024	3025	3047	3048	3049	3050	3051	3089	3090	3091	3092	3093	3113	3114	3115
	3116	3117	3143	3144	3145	3146	3147	3170	3171	3172	3173	3174	3201	3202	3203
	3204	3205	3229	3230	3231	3232	3233	3272	3273	3274	3275	3276	3314	3315	3316
	3317	3318	3384	3385	3386	3387	3388	3428	3429	3430	3431	3432	3467	3468	3469
	3470	3471	3527	3528	3529	3530	3531	3574	3575	3576	3577	3578	3633	3634	3635
	3636	3637	3667	3668	3669	3670	3671	3694	3695	3696	3697	3698	3729	3730	3731
	3732	3733	3768	3769	3770	3771	3772	3801	3802	3803	3804	3805	3834	3835	3836
	3837	3838	3872	3873	3874	3875	3876	3911	3912	3913	3914	3915	3940	3941	3942
	3943	3944	3973	3974	3975	3976	3977	4004	4005	4006	4007	4008	4036	4037	4038
	4039	4040	4075	4076	4077	4078	4079	4111	4112	4113	4114	4129	4130	4137	4138
	4224	4225	4236	4237	4399	4400	4414	4415	4574	4575	4584	4585	4719	4720	4731
	4732	4895	4896	4903	4904	5045	5046	5055	5056	5218	5219	5228	5229	5398	5399
	5408	5409	5590	5591	5598	5599	5741	5742	5751	5752	5911	5912	5921	5922	6037
	6088	6097	6098	6274	6275	6282	6283	6419	6420	6429	6430	6582	6583	6593	6594
	6766	6767	6777	6778	6956	6957	6967	6968	7147	7148	7154	7155	7184	7185	7189
	7190	7217	7218	7219	7226	7227	7228	7275	7278	7282	7288	7291	7306	7316	7335
	7396	7404	7423	7426	7465	7478	7529	7533	7587	7591	7626	7631	7672	7680	7702
	7705	7746	7754	7798	7801	7844	7870	7938	7948	7985	8000	8133	8140	8157	8175
	8206	8209	8436	8441	8496	8499	8538	8541	8617	8618	8677	8680	8809	8833	8840
	8842	8983	8985	9051	9053	9121	9123	9199	9200	9226	9228	9393	9395	9563	9565
	9683	9685	9716	9725	9735	9772	9775	9776	9779	9805	9809	9876	9946	10023	10038
	10048	10109	10112	10120	10132	10133	10141	10146	10162	10174	10227	10229	10248	10271	10272
.IFT	10275	10289	10305	10331	10408	10414	10447	10463	10480						
	2497	2502	2507	2511	2515	2519	2523	2527	2531	2535	2558	2595	2599	2674	2701
	2705	2709	2713	2760	2766	2772	2852	2858	2864	2898	2904	2910	2936	2940	2944
	2951	3334	3338	3342	3346	7234	7240	7344	7350	7357	7361	7365	7372	7376	9546
	8552	8564	8570	8576	8580	8586	8592	8596	9720	9787	10111	10116	10197	10217	10224
.IFTF	10258														
	2497	2502	2507	2511	2515	2519	2523	2527	2531	2535	2558	2595	2599	2674	2701
	2705	2709	2713	2760	2766	2772	2852	2858	2864	2898	2904	2910	2936	2940	2944
	2951	3334	3338	3342	3346	7234	7240	7344	7350	7357	7361	7365	7372	7376	8546
	8552	8564	8570	8576	8580	8586	8592	8596	9720	9785	10048	10109	10112	10193	10201
.IIF	10223	10257													
	2	7	12	44	45	46	48	51	52	53	54	173	297	2451	2454
	2460	2461	2463	2464	2860	2866	7276	7282	7283	7294	7306	7310	9738	9739	9740
	9741	9742	9743	9747	3786	9787	9802	9805	9806	9943	9946	9952	9991	9992	10051
	10155	10163	10170	10224	10230	10231	10232	10233	10234	10239	10263	10271	10272	10297	10312
.IRP	10316	10429	10430	10431	10432	10433	10435	10437	10438	10439	10440	10441	10442	10443	
	2443	2567	2612	2643	2661	2794	2919	2956	3021	3046	3088	3112	3142	3169	3200
	3228	3271	3313	3383	3427	3466	3526	3573	3632	3666	3693	3728	3767	3800	3833
	3871	3910	3939	3972	4003	4035	4074	4110	4128	4223	4398	4573	4718	4894	5044
	5217	5397	5589	5740	5910	6086	6273	6418	6581	6765	6955	7146	7183	7217	7406
	7415	7480	7492	7632	7691	7756	7789	7872	7874	7929	7950	7977	8142	8160	8177
	8199	8286	8426	8450	8491	8502	8527	8705	8799	8860	8972	8996	9042	9065	9109
	9137	9181	9257	9384	9421	9552	9599	9673	9693	9713	9748	9819	9859	10187	10213
.LIST	10240	10450	10456	10469	10470										
	2	14	25	39	51	166	173	277	279	280	281	282	283	284	285
	286	287	288	289	290	291	2411	2443	2465	2482	2497	2502	2507	2511	2515
	2519	2523	2527	2531	2535	2558	2567	2574	2576	2595	2599	2612	2620	2622	2643
	2647	2648	2661	2665	2674	2701	2705	2709	2713	2760	2766	2772	2794	2804	2852
	2858	2864	2898	2904	2910	2919	2928	2936	2940	2944	2951	2956	2966	2967	3021
	3025	3026	3046	3050	3051	3088	3092	3093	3112	3116	3117	3142	3146	3147	3169

	3173	3174	3200	3204	3205	3228	3232	3233	3271	3275	3276	3313	3317	3318	3334
	3338	3342	3346	3383	3387	3388	3427	3431	3432	3466	3470	3471	3526	3530	3531
	3573	3577	3578	3632	3636	3637	3666	3670	3671	3693	3697	3698	3728	3732	3733
	3767	3771	3772	3800	3804	3805	3833	3837	3838	3871	3875	3876	3910	3914	3915
	3939	3943	3944	3972	3976	3977	4003	4007	4008	4035	4039	4040	4074	4078	4079
	4110	4114	4115	4128	4138	4140	4223	4237	4238	4398	4415	4416	4573	4585	4586
	4718	4732	4733	4894	4904	4906	5044	5056	5058	5217	5229	5231	5397	5409	5411
	5589	5599	5601	5740	5752	5754	5910	5922	5924	6086	6098	6100	6273	6283	6285
	6418	6430	6431	6581	6594	6595	6765	6778	6779	6955	6968	6970	7146	7155	7156
	7183	7190	7191	7217	7227	7234	7240	7282	7298	7344	7350	7357	7361	7365	7372
	7376	8546	8552	8564	8570	8576	8580	8586	8592	8596	9720	9742	10132	10271	10421
	10429	10430	10431	10432	10433	10434	10435	10436	10437	10438	10439	10440	10441	10442	10443
	10444														
.MACRO	39	52	240	2566	2611	2793	2919	2955	4128	4222	4397	4572	4718	4893	5043
	5216	5396	5588	5739	5909	6085	6272	6417	6580	6764	6954	7145	7182	7217	10421
.MCALL	2	166	2465												
.NLIST	2	14	25	39	51	166	173	277	279	280	281	282	283	284	285
	286	287	288	289	290	291	2411	2443	2465	2482	2497	2502	2507	2511	2515
	2519	2523	2527	2531	2535	2558	2567	2574	2576	2595	2599	2612	2620	2622	2643
	2647	2648	2661	2665	2674	2701	2705	2709	2713	2760	2766	2772	2794	2804	2852
	2858	2864	2898	2904	2910	2919	2928	2936	2940	2944	2951	2956	2966	2967	3021
	3025	3026	3046	3050	3051	3088	3092	3093	3112	3116	3117	3142	3146	3147	3169
	3173	3174	3200	3204	3205	3228	3232	3233	3271	3275	3276	3313	3317	3318	3334
	3338	3342	3346	3383	3387	3388	3427	3431	3432	3466	3470	3471	3526	3530	3531
	3573	3577	3578	3632	3636	3637	3666	3670	3671	3693	3697	3698	3728	3732	3733
	3767	3771	3772	3800	3804	3805	3833	3837	3838	3871	3875	3876	3910	3914	3915
	3939	3943	3944	3972	3976	3977	4003	4007	4008	4035	4039	4040	4074	4078	4079
	4110	4114	4115	4128	4138	4140	4223	4237	4238	4398	4415	4416	4573	4585	4586
	4718	4732	4733	4894	4904	4906	5044	5056	5058	5217	5229	5231	5397	5409	5411
	5589	5599	5601	5740	5752	5754	5910	5922	5924	6086	6098	6100	6273	6283	6285
	6418	6430	6431	6581	6594	6595	6765	6778	6779	6955	6968	6970	7146	7155	7156
	7183	7190	7191	7217	7227	7234	7240	7282	7298	7344	7350	7357	7361	7365	7372
	7376	8546	8552	8564	8570	8576	8580	8586	8592	8596	9720	9742	10132	10271	10421
	10429	10430	10431	10432	10433	10434	10435	10436	10437	10438	10439	10440	10441	10442	10443
	10444														
.PAGE	56	167	203	240	298	7496	8670	9806	9873	9943	10171	10224	10272	10328	10405
	10444														
.REM	3038	3074	3259	3304	3423	3460	3627	3662	3690	3724	3763	3796	3829	3868	3907
	3935	3965	4000	4031	4070										
.REPT	15	21	173	279	285										
.SBTTL	39	40	56	167	177	188	203	240	298	2025	2261	2433	2434	2435	2444
	2539	2567	2612	2643	2661	2734	2919	2956	3021	3034	3035	3036	3046	3088	3112
	3142	3169	3200	3228	3271	3313	3383	3427	3466	3526	3573	3632	3666	3693	3729
	3767	3800	3833	3871	3910	3939	3972	4003	4035	4074	4107	4110	4128	4223	4398
	4573	4718	4890	4894	5044	5217	5397	5589	5740	5910	6086	6273	6418	6581	6765
	6955	7142	7146	7183	7217	7265	7266	7267	7272	7393	7420	7461	7496	7513	7526
	7624	7669	7699	7743	7795	7841	7935	7982	8130	8164	8203	8433	8493	8535	8615
	9726	9727	9728	9732	9806	9873	9943	10171	10224	10272	10328	10405	10421	10444	
.TITLE	2														
.WORD	173	174	175	185	248	251	252	253	254	257	258	259	260	261	262
	263	266	267	268	277	279	280	281	282	283	284	285	286	287	288
	289	290	1670	1672	1674	1677	1680	1683	1685	1688	1692	1695	1697	1699	1702
	1705	1708	1712	1715	1719	1722	1725	1728	1731	1735	2014	2016	2018	2345	2346
	2348	2350	2351	2405	2634	4150	4151	4152	5003	5523	5699	6215	6382	6893	7084

7287	7290	7305	8159	8702	8726	8733	8734	8735	8736	8737	8790	8791	8795	8796
9253	9280	9380	9381	9940	9947	9948	9949	10220	10223	10298	10303	10404	10479	10481

ERRORS DETECTED: 0

*DZRJHA, DZRJHA/SOL/CRF+DZRJHA.MAC, DZRJHA.ITM, FLTINS.ITM, DZRJHA.ERR, FLTINS.ERR, DZRJHA.DEF, DZRJHA.SET, FLTINI.P11, DZRJHA.TST, DZRJH
RUN-TIME: 120 104 16 SECONDS
CORE USED: 29K

