

# RH11-RP04

MASSBUS INTERFACE/DCL  
MD-11-DZRPT-D

EP-DZRPT-D-DL-B

SEP 1975

Copyright © 1975



FICHE 1 OF 2

Made In U.S.A.

# RH11-RP04

MASSBUS INTERFACE/DCL  
MD-11-DZRPT-D

EP-DZRPT-D-DL-B

Copyright © 1975  
FICHE 2 OF 2

SEP 1975  
**digital**  
Made In U.S.A.

This block contains a grid of 100 microfiche frames, arranged in 10 rows and 10 columns. Each frame contains a small, high-contrast image of a document page, likely a technical manual or report. The pages are too small to read clearly but appear to contain text and possibly diagrams or tables. The frames are separated by thin white lines.





## IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZRPT-D-D  
PRODUCT NAME: RJPO4 DISKLESS CONTROLLER TEST-PART II (STATIC 1B)  
DATE CREATED: OCT 1974, MAR 1975, MAY 1975, AUG 1975  
MAINTAINER: DIAGNOSTIC ENGINEERING  
AUTHOR: S. MALLICK, H. BLACKSTONE, J. KELLY

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 RPO4 SUBSYSTEM. IT DOES NOT USE THE DISK SURFACE OR ANY SIGNALS FROM THE MDLI. IT REQUIRES THAT THE DCL CABLE BE PLUGGED INTO THE MDLI OR BE 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.

## 2.0 REQUIREMENTS

## 2.1 EQUIPMENT

PDP-11 COMPUTER WITH CONSOLE TELETYPE, AND A RPO4 DISK SYSTEM. THE RPO4 DISK SYSTEM WILL CONSIST OF AN RH11 CONTROLLER, A DISK CONTROL LOGIC (DCL), THE CABLE FROM THE DCL CAN BE CONNECTED TO THE MDLI BUT IF NOT THAT CABLE MUST BE PROPERLY TERMINATED.

## 2.2 STORAGE

THIS PROGRAM REQUIRES 16K WORDS OF MEMORY.

## 2.3 PRELIMINARY PROGRAMS

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

## 3.0 LOADING PROCEDURE

USE STANDARD PROCEDURE FOR LOADING .ABS TAPES

## 4.0 STARTING PROCEDURE

## 4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1

## 4.2 STARTING ADDRESS

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

200 START  
ALL SWITCHES MUST BE DOWN FOR WORST CASE RUN. WITH THIS STARTING ADDRESS ALL THE RPO4S ON THE SYSTEM WILL BE TESTED ONE AT A TIME BEFORE "END PASS" IS PRINTED OUT. TESTING WILL START WITH THE LOWEST UNIT NUMBER DRIVE THAT IS POWERED UP (THAT IS THE LOWEST UNIT NUMBER RHAS REGISTER THAT RESPONDS) THEN GO ON TO THE NEXT HIGHER UNIT NUMBER THAT IS POWERED UP.

210 START  
ALL SWITCHES MUST BE DOWN FOR WORST CASE RUN. WITH THIS STARTING ADDRESS THE CONSOLE TELETYPE WILL ASK FOR THE UNIT NUMBER TO BE TESTED. THEN ONLY THAT UNIT WILL BE TESTED

FOR EACH PASS OF THE PROGRAM.

#### 4.3 PROGRAM AND/OR OPERATOR ACTION

1. LOAD THE PROGRAM INTO MEMORY.
2. SET STARTING ADDRESS ON THE SWITCH REGISTER
3. PRESS "LOAD ADDRESS".
4. SET "OPERATIONAL SWITCH SETTINGS" (SEE SECTION 5.1)  
WORST CASE IS ALL SWITCHES DOWN.
5. PRESS "START".
6. FOR THE FIRST PASS EACH TEST WILL BE EXECUTED ONCE ON THE DRIVES PRESENT OR DRIVE SELECTED BEFORE "END PASS" IS PRINTED. THE FIRST PASS WILL REQUIRE OPERATOR INTERVENTION IF THE PROGRAM IS NOT RUN UNDER AN "ACT-11" MONITOR. THE SECOND AND SUBSEQUENT PASSES WILL EXECUTE EACH TEST FOUR TIMES ON EACH DRIVES PRESENT OR DRIVE SELECTED BEFORE "END PASS" IS PRINTED. THE SECOND AND SUBSEQUENT PASSED DO NOT NEED ANY OPERATOR INTERVENTION.

#### 5.0 OPERATING PROCEDURE

##### 5.1 OPERATIONAL SWITCH SETTINGS

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

SWITCH 15 - HALT ON ERROR  
WHEN THIS SWITCH IS SET, IF THE PROGRAM FINDS AN ERROR THEN THE 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 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 USED 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 EID 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

SEQ 0006



SWITCH 7 - STOP FURTHER COMPARES IF SMOB 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 SMOB 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 TEST 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.

## 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

## REVISION B:

1. ADDED READ/WRITE SETUP CHECKS TO THE 'DTE' TESTS (T12, T13 & T14) AND FIXED A BUG IN THE REGISTER SAVE PORTION OF THEIR READ/WRITE ROUTINES.
2. MODIFIED DVA, DPR, RDY, DRY & VV STATUS BIT CHECKS IN ALL TESTS TO HALT THE PROGRAM IF THEY FAIL.
3. ADDED DELAY BEFORE ISSUING DIAGNOSTIC MODE SECTOR PULSES (SP) TO ACCOUNT FOR THE SILO FILLING TIME DELAY BETWEEN ISSUING GO AND THE RUN LINE BECOMING HIGH.

## REVISION C:

1. ADDED 28 NEW TESTS TO THIS PROGRAM. THESE /

TESTS WERE DEVELOPED FROM FAULT INSERTION TESTING ON THE RH11 USING "DZRPS". JO1

2. ADDED REVISION DATE TO THE PROGRAM HEADER TYPE OUT.
3. CHANGED UNEXPECTED DRIVE INTERRUPT TYPE OUT MESSAGE.

REVISION D:

1. MODIFIED THE PROGRAM SO THAT WITH A LOAD AND START AT LOCATION 210, THE PROGRAM WOULD LOOP ON TESTING THE SELECTED UNIT EVEN IF THAT UNIT SHOULD FAIL TO RESPOND.
2. MODIFIED THE NON-STANDARD ADDRESS RESTART ROUTINE SO THAT THE TTY INTERRUPT BIT WOULD BE ENABLED.

9.0 PROGRAM DESCRIPTION

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

DOCUMENT  
\*\*\*\*\*  
MAINDEC-11-DZRPT-D, RJPO4 DISKLESS RH11 TEST-PART 2  
\*\*\*\*\*

COPYRIGHT 1975  
DIGITAL EQUIPMENT CORPORATION  
MAYNARD, MASS. 01754

TABLE OF CONTENTS  
\*\*\*\*\*

40 OPERATIONAL SWITCH SETTINGS  
55 BASIC DEFINITIONS  
161 TRAP CATCHER  
168 STARTING ADDRESS(ES)  
181 MEMORY MANAGEMENT DEFINITIONS  
220 COMMON TAGS  
276 ERROR POINTER TABLE  
2001 HARDWARE REGISTER BIT DEFINITIONS  
2237 REGISTER ADDRESSES  
2397  
2398 \*\*\*PROGRA. SETUP & SETUP TESTS\*\*\*  
2399  
2866  
2867 \*\*\*DIAGNOSTIC CODE\*\*\*  
2868  
3907 EXTENDED MEMORY, DRIVE TIMING & SECTOR SELECT TESTS  
4665 DATA TRANSFER TESTS USING ECC  
6917 CURSORY INTERRUPT LOGIC TESTS  
7037  
7038 \*\*\*SUBROUTINES\*\*\*  
7039  
7046 END OF PASS ROUTINE

TABLE OF CONTENTS  
\*\*\*\*\*

7168	SAVE REGISTERS ROUTINE
7195	FLOAT 1 AND 0
7236	CLEAR MEMORY ROUTINE
7271	LOCAL TRAPS
7288	CLEAR DISK ROUTINE
7301	CHECK DISK STATUS ROUTINES
7399	WAIT LOOP
7444	SAVE ROUTINE
7474	WRITE CHECK ROUTINE
7518	COMPARE ROUTINE
7570	WRITE CHECK DATA
7616	CRC GENERATION ROUTINE
7710	SIMULATED DISK SETUP
7757	CHECK HCE ROUTINE
7905	EXIT MRT HEADER & DATA ROUTINE
7939	JAM CURRENT CYLINDER ROUTINE
7978	ECC GENERATION AND COMPARISON ROUTINE
8208	ECC GENERATION CONTROL ROUTINE
8268	SOFTWARE DISK DATA ECC GEN. ROUTINE
8310	RH BASE ADDRESS CHANGE ROUTINE
8390	DISK SIMULATION
9394	

TABLE OF CONTENTS  
\*\*\*\*\*

9395 \*\*\*SYSMAC LIBRARY ROUTINES\*\*\*  
9396  
9402 SCOPE HANDLER ROUTINE  
9476 CONVERT BINARY TO DECIMAL AND TYPE ROUTINE  
9544 TYPE ROUTINE  
9591 TTY INPUT ROUTINE  
9694 READ AN OCTAL NUMBER FROM THE TTY  
9748 ERROR HANDLER ROUTINE  
9794 ERROR MESSAGE TYPEOUT ROUTINE  
9851 BINARY TO OCTAL (ASCII) AND TYPE  
9929 TRAP DECODER  
9944 TRAP TABLE  
9965 POWER DOWN AND UP ROUTINES

2 COPYRIGHT (C) 1975  
DIGITAL EQUIPMENT CORP.  
MAYNARD, MASS. 01754

PROGRAM BY SUB MALLICK, PETE BLACKSTONE, JIM KELLY

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

28 \*\*\*\*\*

30 \*\*\*\*\*

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
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST IN SMR<7:0>
7	STOP FURTHER COMPARES IF SMOB IS LOW
6	ECC TEST-COMPARE END RESULTS ONLY IF SMOB IS LOW

55 \*\*\*\*\*  
BASIC DEFINITIONS  
\*\*\*\*\*

57 INITIAL ADDRESS OF THE STACK POINTER \*\*\* 1000 \*\*\*

68 GENERAL PURPOSE REGISTER DEFINITIONS

80 PRIORITY LEVEL DEFINITIONS

90 "SWITCH REGISTER" SWITCH DEFINITIONS



118 DATA BIT DEFINITIONS (BIT00 TO BIT15)

146 BASIC "CPU" TRAP VECTOR ADDRESSES

161 \*\*\*\*\*  
TRAP CATCHER  
\*\*\*\*\*

164 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

168 \*\*\*\*\*  
STARTING ADDRESS(ES)  
\*\*\*\*\*

175 STARTING ADDRESS 200 FOR NORMAL STARTS  
THIS WILL TEST ALL R-74'S ON THE SYSTEM A SINGLE DRIVE AT A TIME  
STARTING ADDRESS 210 WILL TEST ONLY ONE SPECIFIED DRIVE

181 \*\*\*\*\*  
MEMORY MANAGEMENT DEFINITIONS  
\*\*\*\*\*

183 KT11 VECTOR ADDRESS

187 KT11 STATUS REGISTER ADDRESSES

194 KERNAL "I" PAGE DESCRIPTOR REGISTERS

205 KERNAL "I" PAGE ADDRESS REGISTERS

218 \*\*\*\*\*

220 \*\*\*\*\*  
COMMON TAGS  
\*\*\*\*\*

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

274 \*\*\*\*\*

276

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

278 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:

284 .           EM            ::POINTS TO THE ERROR MESSAGE  
                  DH            ::POINTS TO THE DATA HEADER  
                  DT            ::POINTS TO THE DATA  
                  DF            ::POINTS TO THE DATA FORMAT

922 \*\*\*\*\*  
\*\*\*\*\*

2001

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

2004 \*\*\*\*\*  
\*\*\*\*\*

2040 \*\*\*\*\*  
\*\*\*\*\*

2237

\*\*\*\*\*  
REGISTER ADDRESSES  
\*\*\*\*\*

2397

\*\*\*\*\*  
\*\*\*\*\*

2398

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

2399

\*\*\*\*\*  
\*\*\*\*\*

2513 \*\*\*\*\*  
TEST 1 REFERENCE EACH REGISTER  
REFERENCE EACH REGISTER BY A MOVE INSTRUCTION  
\*\*\*\*\*

2555 \*\*\*\*\*  
TEST 2 RHCS2-CONTROL AND STATUS 2

2558 THIS PARTIALLY TESTS RHCS2 TO ENABLE DETERMINATION  
OF THE NUMBER OF DRIVES PRESENT

2561 \*\*\*\*\*

2579 \*\*\*\*\*  
TEST 3 PARTIAL TEST FOR RHAS FOR UNIT NUMBERS PRESENT  
\*\*\*\*\*

2596 \*\*\*\*\*  
TEST 4 TEST FOR DRIVES PRESENT USING RHAS AND RHCS2  
\*\*\*\*\*

2711 \*\*\*\*\*  
TEST 5 TEST SERIAL NUMBER AND DRIVE TYPE  
READ SERIAL NUMBER REGISTER AND DRIVE TYPE REGISTER  
TYPE IT OUT AND PROCEED  
TO LOOP HERE SET SWITCH 8 AND THIS TEST NO AND RESTART

2717 \*\*\*\*\*

2752 \*\*\*\*\*  
TEST 6 CHECK MOL TO BE LOW

2755 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

2759 \*\*\*\*\*

2789 \*\*\*\*\*  
TEST 7 PACK ACKNOWLEDGE COMMAND TEST

2792 THE PACK ACKNOWLEDGE COMMAND WILL BE LOADED INTO F:ACS1 WITH GO  
THEN ALL REGISTERS WILL BE CHECKED  
RH CLEAR WILL BE GIVEN  
THEN ALL REGISTERS WILL BE CHECKED

2797 \*\*\*\*\*

2854 \*\*\*\*\*  
TEST 10 MAKE CURRENT CYLINDER = 0  
\*\*\*\*\*

2866 \*\*\*\*\*  
\*\*\*\*\*

2867 \*\*\*\*\*  
\*\*\*DIAGNOSTIC CODE\*\*\*  
\*\*\*\*\*

2868 \*\*\*\*\*  
\*\*\*\*\*

2871 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.

2878 \*\*\*\*\*  
TEST 11 BCTA LEGAL REGISTER RESPONSE TEST  
\*\*\*\*\*

2907 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.

2920 \*\*\*\*\*  
TEST 12 BCTA MOVB LO BYTE TO MC  
\*\*\*\*\*

2944 \*\*\*\*\*  
TEST 13 BCTA MOVB HI BYTE TO MC  
\*\*\*\*\*

2974 \*\*\*\*\*  
TEST 14 BCTA BISO LO BYTE TO MC  
\*\*\*\*\*

3001 \*\*\*\*\*  
TEST 15 BCTA BISO HI-BYTE TO MC  
\*\*\*\*\*

3032 \*\*\*\*\*  
TEST 16 BCTA BICB LO-BYTE TO MC  
\*\*\*\*\*

3060 \*\*\*\*\*  
TEST 17 BCTA BICB HI- BYTE TO MC  
\*\*\*\*\*

3092 THIS TEST ATTEMPT TO ACCESS AN ILLEGAL REGISTER WITHIN THE RANGE  
OF ADDRESSES SELECTED BY THE XOR'S. THE RH11 ADDRESS DECODE LOGIC WILL ALLOW UP TO 32  
CONTIGOUS REGISTERS TO EXIST IN OUR CONFIGURATION ONLY 16 WILL REALLY  
EXIST THIS TEST ATTEMPTS TO ACCESS THE 20(8) REGISTER IF IT RESPONDS  
THE TEST WILL TYPE OUT AN ERROR

3099 \*\*\*\*\*  
TEST 20 BCTA ILLEGAL REGISTER TEST  
\*\*\*\*\*

3128 THE FOLLOWING TWO TESTS DETERMINE IF ALL NON DRIVES SET THE  
NED BIT AND THAT ALL EXISTING DRIVES CLEAR THE NED BIT.  
THE TESTS LOOK AT TOTALAT TO DETERMINE WHICH DRIVES EXIST AND TEST THAT  
THESE DRIVES WILL CLEAR THE NED BIT. AND ALSO THAT NON EXISTANT DRIVES  
WILL SET THE NED BIT.  
ON FIRST PASS ONLY, IF ALL DRIVES ARE ON LINE A MESSAGE WILL SO ADVISE THE  
OPERATOR. THE TEST WILL CONTINUE REGARDLESS OF THE OPERATORS ACTION.

3136 \*\*\*\*\*  
TEST 21 BCTB (NED) NON-EXISTANT DRIVE TEST. (SET)  
\*\*\*\*\*

3206 \*\*\*\*\*  
TEST 22 BCTB (NED) NON-EXISTANT DRIVE TEST. (CLEARED)  
\*\*\*\*\*

3247 TEST TO SEE IF WE CAN READ FROM THE "AS" REGISTER AND NOT CAUSE  
A NED NON-EXISTANT DRIVE ERROR

3250 \*\*\*\*\*  
TEST 23 BCTB AS REGISTER TEST  
\*\*\*\*\*

3284 THE NEXT THREE TESTS WILL TEST THE BUS ADDRESS REGISTER IN BOTH WORD AND BYTE MODE. THE PATTERNS ARE CHOSEN TO PICK UP ANY DROPPED OR STUCK BITS.

3289 \*\*\*\*\*  
TEST 24 BCTC BUS ADDRESS REGISTER  
\*\*\*\*\*

3349 \*\*\*\*\*  
TEST 25 BCTC BUS ADDRESS REGISTER LO-BYTE  
\*\*\*\*\*

3396 \*\*\*\*\*  
TEST 26 BCTC BUS ADDRESS REGISTER HI-BYTE  
\*\*\*\*\*

3451 THIS TEST CAUSE AN RH11 INTERRUPT VIA THE SPECIAL COMMAND SEQUENCE  
BIS #IE,2RHCS2. THIS TEST PROVES ONLY THAT THE HARDWARE CAN HANDLE  
INTERRUPTS PROPERLY

3455 \*\*\*\*\*  
TEST 27 RH11 INTERRUPT TEST  
\*\*\*\*\*

3486 THE PROGRAM SETS THE PORT SELECT BIT .THEN DOES A RESET  
IF THE SIGNAL CLR L WAS GENERATED THE PORT SELECT BIT WILL CLEAR.

3489 \*\*\*\*\*  
TEST 30 BCTD CLR L TEST  
\*\*\*\*\*

3514 SET THE MXF FLOP AND READ IT BACK.

3516 \*\*\*\*\*  
TEST 31 MXF TEST  
\*\*\*\*\*

3541 SET THE UNIBUS PARITY ERROR FLOP DIRECTLY VIA A MOV #UPE TO RHCS2  
ATTEMPT TO READ IT BACK.

3544 \*\*\*\*\*  
TEST 32 CSRB UNIBUS PARITY ERROR SET TEST  
\*\*\*\*\*

3572 SET THE UNIBUS PARITY ERROR FLOP AND ATTEMPT TO CLEAR IT WITH A  
CONTROLLER CLEAR.

3575 \*\*\*\*\*  
TEST 33 CSRB UNIBUS PARITY ERROR CLEAR TEST  
\*\*\*\*\*

3605 SET THE UNIT SELECT REGISTER TO DRIVE #7 ALL BITS SET. GENERATE A CONTROLLER CLEAR AND VERIFY THAT THE UNIT SELECT REGISTER IS CLEARED.

3608 \*\*\*\*\*  
TEST 34 CSRB UNIT SELECT CLEAR TEST  
\*\*\*\*\*

3638 SET THE UPE FLOP UNIBUS PARITY ERROR. VERIFY THAT THE SETTING OF (UPE) ALSO SETS THE TRANSFER ERROR (TRE) FLOP

3641 \*\*\*\*\*  
TEST 35 CSRB TRANSFER ERROR (TRE) - UPE  
\*\*\*\*\*

3669 SET THE NED BIT NON EXISTANT DRIVE VERIFY THAT THIS SETS THE (TRE) BIT.

3671 \*\*\*\*\*  
TEST 36 CSRB TRANSFER ERROR (TRE) NED  
\*\*\*\*\*

3708 SET THE PORT SELECT FLOP VERIFY THAT WE CAN READ IT BACK.

3710 \*\*\*\*\*  
TEST 37 CSRA PSEL CLEAR TEST  
\*\*\*\*\*

3736 VERIFY THAT CONTROLLER CLEAR WILL CLEAR THE COMMAND REGISTER.

3739 \*\*\*\*\*  
TEST 40 CSRB COMMAND REGISTER CLEAR TEST  
\*\*\*\*\*

3766 THE COMMAND REGISTER IS SUPPOSED TO BE SELF CLEARING THAT IS WHEN THE RH11 IS SELECTED WHEATHER OR NOT WE ADDRESS IT DIRECTLY THE LOGIC SHOULD CLEAR THE COMMAND REGISTER. THIS IS TESTED BY LOADING A COMMAND INTO IT READING ANOTHER REGISTER THAN RETURNING TO CHECK THE COMMAND REGISTER TO DETERMINE IF IT CLEARED.

3772 \*\*\*\*\*  
TEST 41 CSRB COMMAND REGISTER RESELECT CLEAR TEST  
\*\*\*\*\*

3801 HERE WE TEST TO SEE IF SETTING (IE) AND (RDY) WILL CAUSE AN INTERRUPT.

3803 \*\*\*\*\*  
TEST 42 CSRB READY AND IE INTERRUPT TEST  
\*\*\*\*\*

3832 THE (IE) BIT IS SET CAUSING AN INTERRUPT. AT THE COMPLETION OF THE INTERRUPT WE CHECK TO SEE IF THE (IE) BIT HAS CLEARED.

3835 \*\*\*\*\*  
TEST 43 CSRB BOES "IE" CLEAR AFTER AN INTERRUPT  
\*\*\*\*\*

3871 HERE WE VERIFY THAT WRITING INTO THE "AS" REGISTER WILL NOT CAUSE  
AN (NED) ERROR.

3874 \*\*\*\*\*  
TEST 44 BCTB "AS" WRITE TEST  
\*\*\*\*\*

3907 \*\*\*\*\*  
EXTENDED MEMORY, DRIVE TIMING & SECTOR SELECT TESTS  
\*\*\*\*\*

3910 \*\*\*\*\*  
TEST 45 MAKE CURRENT CYLINDER = 0  
\*\*\*\*\*

3927 \*\*\*\*\*  
TEST 46 RHCSI - BITS 8 AND 9 - EXTENDED ADDR (A16 & A 17)

3930 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

3935 \*\*\*\*\*

4014 \*\*\*\*\*  
TEST 47 DRIVE TIMING ERROR

4017 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'

\*\*\*\*\*

4189 \*\*\*\*\*  
TEST 50 DRIVE TIMING ERROR

4192 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'

\*\*\*\*\*

4355 \*\*\*\*\*  
TEST 51 DRIVE TIMING ERROR

4358 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'

\*\*\*\*\*

4492 \*\*\*\*\*  
TEST 52 SECTOR SELECTION

4495 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

4504 \*\*\*\*\*

4665

\*\*\*\*\*  
DATA TRANSFER TESTS USING ECC  
\*\*\*\*\*

4669 \*\*\*\*\*  
TEST 53 WRITE ECC TEST 1

4672 THIS IS A WRITE ECC TEST  
WRITE CYLINDER0, FORMAT 16 BITS PER WORD  
TRACK 0, SECTOR 1, KEYS C, NUMBER OF WORDS 256  
OF ALL ZEROS.

4677 \*\*\*\*\*

4819 \*\*\*\*\*  
TEST 54 READ ECC ENABLED 1A

4822 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

4829 \*\*\*\*\*

4992 \*\*\*\*\*  
TEST 55 READ ECC ENABLED 1B

4995 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

5002 \*\*\*\*\*

5172 \*\*\*\*\*  
TEST 56 READ ECC ENABLED 1C

5175 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

5182 \*\*\*\*\*

5364 \*\*\*\*\*  
TEST 57 WRITE ECC TEST 2

5367 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.

5372 \*\*\*\*\*

5515 \*\*\*\*\*  
TEST 60 READ ECC ENABLED 2A

5518 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

5525 \*\*\*\*\*

5685 \*\*\*\*\*  
TEST 61 READ ECC ENABLED 28

5688 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

5695 \*\*\*\*\*

5861 \*\*\*\*\*  
TEST 62 READ ECC ENABLED 2C

5864 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

5871 \*\*\*\*\*

6048 \*\*\*\*\*  
TEST 63 WRITE ECC TEST 3

6051 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.

6056 \*\*\*\*\*

6193 \*\*\*\*\*  
TEST 64 READ ECC ENABLED 3A

6196 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

6203 \*\*\*\*\*

6356 \*\*\*\*\*  
TEST 65 READ ECC ENABLED 3B

6359 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

6367 \*\*\*\*\*

6540 \*\*\*\*\*  
 TEST 66 READ ECC ENABLED 3C

6543 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

6551 \*\*\*\*\*

6730 \*\*\*\*\*  
 TEST 67 READ ECC ENABLED 3D

6733 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

6741 \*\*\*\*\*

6917 \*\*\*\*\*  
 CURSORY INTERRUPT LOGIC TESTS  
 \*\*\*\*\*

6921 \*\*\*\*\*  
 TEST 70 PROGRAM INTERRUPT

6924 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  
 \*\*\*\*\*

6958 \*\*\*\*\*  
TEST 71 INTERRUPT AT PROCESSOR AND DISK PRIORITY SAME

6961 PROCESSOR PRIORITY IS SET AT 5 (SAME AS THE DISK)  
IE AND RDY IS SET. THIS SHOULD NOT INTERRUPT  
\*\*\*\*\*

6991 \*\*\*\*\*  
\*\*\*\*\*  
TEST 72 END OF DRIVE

6995 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

7000 \*\*\*\*\*

7037

\*\*\*\*\*  
\*\*\*\*\*

7038

\*\*\*\*\*  
\*\*\*SUBROUTINES\*\*\*  
\*\*\*\*\*

7039

\*\*\*\*\*  
\*\*\*\*\*

7044 \*\*\*\*\*

7046

\*\*\*\*\*  
END OF PASS ROUTINE  
\*\*\*\*\*

7048 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 TST!

7090 \*\*\*\*\*  
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  
PROGRAM GOES BACK - USUALLY BACK TO THE BEGINNING OF THE TEST.

7095 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.

\*\*\*\*\*

7168 \*\*\*\*\*  
SAVE REGISTERS ROUTINE  
\*\*\*\*\*

7170 \*\*\*\*\*  
\*\*\*\*\*

7195 \*\*\*\*\*  
FLOAT 1 AND 0  
\*\*\*\*\*

7197 \*\*\*\*\*  
FLOAT A ONE AND A ZERO THRU A DESIGNATED REGISTER  
ABSOLUTE ADDRESS OF REG UNDER TEST IS IN R4  
\*\*\*\*\*

7236 \*\*\*\*\*  
CLEAR MEMORY ROUTINE  
\*\*\*\*\*

7239 \*\*\*\*\*  
THIS CLEARS ANY BLOCK OF MEMORY  
FILLING IT WITH ANY DATA

```
CALL
JSR   R0,CLAREA
X           ;STARTING ADDRESS OF BLOCK
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  
\*\*\*\*\*

7271 \*\*\*\*\*  
 LOCAL TRAPS  
 \*\*\*\*\*

7279 EXAMPLE OF THE USE OF THE ABOVE  
 THIS WILL LOOP BETWEEN X: AND SCOP1 PROVIDED THERE IS NO "NEWTST"  
 MOV     #X,     @#LAD  
 X:     ---     ---  
        ---     ---  
        ---     ---  
                SCOP1

7288 \*\*\*\*\*  
 CLEAR DISK ROUTINE  
 \*\*\*\*\*

7301 \*\*\*\*\*  
 CHECK DISK STATUS ROUTINES  
 \*\*\*\*\*

7303 \*\*\*\*\*  
 THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1  
 AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1  
 IT ALSO CHECKS THAT ALL OTHER BITS IN THESE REGISTERS = 0  
 \*\*\*\*\*

7361 \*\*\*\*\*  
 THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1  
 AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1  
 IT DOES NOT CHECK THAT OTHER BITS IN THESE REGISTERS = 0  
 \*\*\*\*\*

7399 \*\*\*\*\*  
 WAIT LOOP  
 \*\*\*\*\*

7400 \*\*\*\*\*  
 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)  
 \*\*\*\*\*

7433 CALL FOR THE ABOVE WAITLOOP IS

```

MOV     2A,2#XS             ;A CONTAINS REGISTER ADDRESS
-     -                     ;HENCE XS WILL HAVE ABSOLUTE REG. ADR.
-     -                     -
-     -                     -
X$:     MAT                   ;ABSOLUTE REG. ADDRESS UNDER WAIT
       0                     ;BIT WAITED FOR
       .WORD 0               ;CONTINUE
```

7444 \*\*\*\*\*  
SAVE ROUTINE  
\*\*\*\*\*

7446 \*\*\*\*\*  
\*\*\*\*\*

7474 \*\*\*\*\*  
WRITE CHECK ROUTINE  
\*\*\*\*\*

7476 \*\*\*\*\*  
\*\*\*\*\*

7518 \*\*\*\*\*  
COMPARE ROUTINE  
\*\*\*\*\*

7520 \*\*\*\*\*  
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  
\*\*\*\*\*

7570 \*\*\*\*\*  
WRITE CHECK DATA  
\*\*\*\*\*

7572 \*\*\*\*\*  
\*\*\*\*\*

7616 \*\*\*\*\*  
CRC GENERATION ROUTINE  
\*\*\*\*\*

7618 \*\*\*\*\*

7644 \*\*\*\*\*



7710 \*\*\*\*\*  
SIMULATED DISK SETUP  
\*\*\*\*\*

7712 \*\*\*\*\*

7722 \*\*\*\*\*

7757 \*\*\*\*\*  
CHECK HCE ROUTINE  
\*\*\*\*\*

7759 \*\*\*\*\*

7774 \*\*\*\*\*

7905 \*\*\*\*\*  
EXIT WRT HEADER & DATA ROUTINE  
\*\*\*\*\*

7907 \*\*\*\*\*  
\*\*\*\*\*

7939 \*\*\*\*\*  
JAM CURRENT CYLINDER ROUTINE  
\*\*\*\*\*

7941 \*\*\*\*\*  
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  
\*\*\*\*\*

7978 \*\*\*\*\*  
ECC GENERATION AND COMPARISON ROUTINE  
\*\*\*\*\*

7980 \*\*\*\*\*  
THIS SUBROUTINE GENERATES AND TESTS ECC  
CALL      JSR PC, ECTEST  
\*\*\*\*\*

8161 CHECK HARDWARE

8208 \*\*\*\*\*  
 ECC GENERATION CONTROL ROUTINE  
 \*\*\*\*\*

8210 \*\*\*\*\*  
 THIS SUBROUTINE WILL CONTROL THE ECC GENERATION ROUTINE  
 FOR ERROR CORRECTION PROCESS  
 CALL JSR, PC,@ECCORR  
 XP ;EXPECTED POSITION REGISTER WHEN CORRECTION IS COMPLETE  
 \*\*\*\*\*

8268 \*\*\*\*\*  
 SOFTWARE DISK DATA ECC GEN. ROUTINE  
 \*\*\*\*\*

8270 \*\*\*\*\*  
 THIS SUBROUTINE GENERATES THE ECC FOR WHAT IS ON DISK AND INSERTS THEM  
 ON LOCATIONS "DISK+1000" AND "DISK+1002"  
 \*\*\*\*\*

8310 \*\*\*\*\*  
 RH BASE ADDRESS CHANGE ROUTINE  
 \*\*\*\*\*

8312 \*\*\*\*\*  
 THIS ROUTINE WILL ALLOW THE CHANGE OF THE BASE  
 ADDRESS FROM 176700 TO ANY TYPED VALUE  
 \*\*\*\*\*

8378 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

8390 \*\*\*\*\*  
 DISK SIMULATION  
 \*\*\*\*\*

8391 \*\*\*\*\*  
 \*\*\*\*\*  
 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 . . . 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,COM4D

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  
DANORD= WITH NUMBER OF WORDS TO BE FOUND ON DISK  
:X=0 MUST BE ZERO  
THE COMMAND THEN IS JSR PC,COM4D

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  
DANORD= WITH NUMBER OF WORDS TO BE FOUND ON DISK  
:X=0 MUST BE ZERO  
THE COMMAND THEN IS JSR PC,COM4D

8444

8451

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

8554

\*\*\*\*\*

8556

THE DISK SECTOR IS DEVIDED AS FOLLOWS

8558 19 WORDS OF 0, ONE WORD 144000  
THESE MAKE 39 BYTES FOR SECTOR GAP AND ONE SYNC. BYTE

8569 5 WORDS OF 0 ONE WORD 144000  
THESE MAKE 11 BYTES FOR HEADER GAP AND ONE SYNC. BYTE  
THESE ARE DCL GENERATED

8575 THERE ARE 256 WORDS OF DATA  
THERE ARE 2 WORDS FOR ECC GENERATED BY DCL  
15 WORDS OF 0 FOR DATA GAP AND TOLERANCE GAP  
\*\*\*\*\*

8585 \*\*\*\*\*  
READ DISK HEADER  
\*\*\*\*\*

8683 READ COMMAND START FROM HERE

8712 \*\*\*\*\*  
READ ONE WORD IN "WORD"  
\*\*\*\*\*

8779 \*\*\*\*\*  
WRITE ONE WORD WHICH COMES BACK IN "WORD"  
\*\*\*\*\*

8847 \*\*\*\*\*  
WRITE DATA  
\*\*\*\*\*

8920 \*\*\*\*\*  
WRITE HEADER AND DATA

THIS IS THE SIMULATED DISK  
ONLY ONE SECTOR OF SPACE IS ALLOWED  
\*\*\*\*\*  
\*\*\*\*\*

8953 \*\*\*\*\*  
WRITE HEADER AND DATA  
\*\*\*\*\*

9088 \*\*\*\*\*  
WRITE HEADER  
\*\*\*\*\*

9096 R0 = MAINT.REG.  
R1 = SIMULATED DISK  
R2 = BYTE COUNT  
R3 = WRITE WORD  
R5 = WORD COUNT

9237 \*\*\*\*\*  
SEARCH SECTOR  
\*\*\*\*\*

9246 R0=RHM 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 BEGINNING OF EACH SECTOR ONE SECTOR CLOCK HAS TO RISE  
BEFORE CLOCK THEN EVERY EIGHT CLOCKS ONE SECTOR CLOCK IS  
IDENTICAL WITH CLOCK  
NUMBERING THE SECTOR CLOCKS 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 THEN HAS 608 SECTOR CLOCKS  
THE NEXT SECTOR THEN HAS ANOTHER 608 SECTOR CLOCKS  
AND SO ON

9295 FOR FIRST BYTE SECTOR CLOCK WILL GO HIGH THEN CLOCK WILL GO HIGH  
BOTH WILL COME DOWN TOGETHER THEN SEVEN CLOCKS WILL BE GIVEN  
;FOR SECOND BYTE AND ALL OTHER BYTES TILL NEXT SECTOR SECTOR CLOCK  
WILL BE IDENTICAL WITH ONE CLOCK

9350 \*\*\*\*\*  
READ ONE SECTOR OF DATA  
\*\*\*\*\*

9383 \*\*\*\*\*

9392 \*\*\*\*\*

9394

\*\*\*\*\*  
\*\*\*\*\*

9395 \*\*\*\*\*  
 \*\*\*SYSMAC LIBRARY ROUTINES\*\*\*  
 \*\*\*\*\*

9396 \*\*\*\*\*  
 \*\*\*\*\*

9400 \*\*\*\*\*

9402 \*\*\*\*\*  
 SCOPE HANDLER ROUTINE  
 \*\*\*\*\*

9404 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 (\$ERFLW) INTO DISPLAY<15:08>  
 THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:  
 SM14=1 LOOP ON TEST  
 SM11=1 INHIBIT ITERATIONS  
 SM09=1 LOOP ON ERROR  
 SM08=1 LOOP ON TEST IN SMR<7:0>  
 CALL SCOPE ;;SCOPE=IOT

9474 \*\*\*\*\*

9476 \*\*\*\*\*  
 CONVERT BINARY TO DECIMAL AND TYPE ROUTINE  
 \*\*\*\*\*

9478 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

9542 \*\*\*\*\*



```

9692 *****
*****
9694 READ AN OCTAL NUMBER FROM THE TTY
*****
9696 THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
      CHANGE IT TO BINARY.
      THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
      OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
      FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
      THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
      CALL:
            RDOCT                ;;READ AN OCTAL NUMBER
            RETURN HERE          ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;;HIGH ORDER BITS ARE IN $HIOCT

9746 *****
*****
9748 ERROR HANDLER ROUTINE
*****
9750 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 $ERRTY: 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=ENT AND N=ERROR ITEM NUMBER

9792 *****
*****
9794 ERROR MESSAGE TYPEOUT ROUTINE
*****
9796 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.

9849 *****

```



9851

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

9853

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

```

9927

\*\*\*\*\*

9929

\*\*\*\*\*  
TRAP DECODER  
\*\*\*\*\*

9931

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.

9944

\*\*\*\*\*  
TRAP TABLE  
\*\*\*\*\*

9946

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

9963

\*\*\*\*\*

9965

\*\*\*\*\*  
POWER DOWN AND UP ROUTINES  
\*\*\*\*\*

39	
40	***ERROR TABLE, BIT DEFINITIONS & STARTING ADDRESSES***
41	
44	OPERATIONAL SWITCH SETTINGS
59	BASIC DEFINITIONS
165	TRAP CATCHER
172	STARTING ADDRESS(ES)
185	MEMORY MANAGEMENT DEFINITIONS
224	COMMON TAGS
277	ERROR POINTER TABLE
2002	HARDWARE REGISTER BIT DEFINITIONS
2238	REGISTER ADDRESSES
2398	
2399	***PROGRAM SETUP & SETUP TESTS***
2400	
2514	T1 REFERENCE EACH REGISTER
2557	T2 RHCS2-CONTROL AND STATUS 2
2582	T3 PARTIAL TEST FOR RHAS FOR UNIT NUMBERS PRESENT
2600	T4 TEST FOR DRIVES PRESENT USING RHAS AND RHCS2
2716	T5 TEST SERIAL NUMBER AND DRIVE TYPE1
2758	T6 CHECK MOL TO BE LOW
2796	T7 PACK ACKNOWLEDGE COMMAND TEST
2862	T10 MAKE CURRENT CYLINDER = 0
2875	
2876	***DIAGNOSTIC CODE***
2877	
2887	T11 BCTA LEGAL REGISTER RESPONSE TEST
2930	T12 BCTA MOV8 LO BYTE TO MC
2955	T13 BCTA MOV8 HI BYTE TO MC
2985	T14 BCTA BIS8 LO BYTE TO MC
3014	T15 BCTA BIS8 HI-BYTE TO MC
3046	T16 BCTA BIC8 LO-BYTE TO MC
3075	T17 BCTA BIC8 HI-BYTE TO MC
3115	T20 BCTA ILLEGAL REGISTER TEST
3153	T21 BCTB (MED) NON-EXISTANT DRIVE TEST. (SET)
3224	T22 BCTB (MED) NON-EXISTANT DRIVE TEST. (CLEARED)
3269	T23 BCTB AS REGISTER TEST
3309	T24 BCTC BUS ADDRESS REGISTER
3370	T25 BCTC BUS ADDRESS REGISTER LO-BYTE
3418	T26 BCTC BUS ADDRESS REGISTER HI-BYTE
3478	T27 RH11 INTERRUPT TEST
3513	T30 BCTD CLR L TEST
3541	T31 MOF TEST
3570	T32 CSRB UNIBUS PARITY ERROR SET TEST
3602	T33 CSRB UNIBUS PARITY ERROR CLEAR TEST
3636	T34 CSRB UNIT SELECT CLEAR TEST
3670	T35 CSRB TRANSFER ERROR (T2) - UPE
3701	T36 CSRB TRANSFER ERROR (TRE) MED
3741	T37 CSRA PSEL CLEAR TEST
3771	T40 CSRB COMMAND REGISTER CLEAR TEST
3805	T41 CSRB COMMAND REGISTER RESELECT CLEAR TEST
3837	T42 CSRB READY AND IE INTERRUPT TEST
3870	T43 CSRB BOES "IE" CLEAR AFTER AN INTERRUPT
3910	T44 BCTB "AS" WRITE TEST

## TABLE OF CONTENTS

3944	EXTENDED MEMORY, DRIVE TIMING & SECTOR SELECT TESTS
3947	T45 MAKE CURRENT CYLINDER = 0
3965	T46 RHCSI - BITS 8 AND 9 - EXTENDED ADDR (A16 & A 17)
4053	T47 DRIVE TIMING ERROR
4229	T50 DRIVE TIMING ERROR
4396	T51 DRIVE TIMING ERROR
4534	T52 SECTOR SELECTION
4708	DATA TRANSFER TESTS USING ECC
4712	T53 WRITE ECC TEST 1
4863	T54 READ ECC ENABLED 1A
5037	T55 READ ECC ENABLED 1B
5218	T56 READ ECC ENABLED 1C
5411	T57 WRITE ECC TEST 2
5563	T60 READ ECC ENABLED 2A
5734	T61 READ ECC ENABLED 2B
5911	T62 READ ECC ENABLED 2C
6099	T63 WRITE ECC TEST 3
6245	T64 READ ECC ENABLED 3A
6409	T65 READ ECC ENABLED 3B
6594	T66 READ ECC ENABLED 3C
6785	T67 READ ECC ENABLED 3D
6973	CURSORY INTERRUPT LOGIC TESTS
6977	T70 PROGRAM INTERRUPT
7015	T71 INTERRUPT AT PROCESSOR AND DISK PRIORITY SAME
7050	T72 END OF DRIVE
7096	
7097	***SUBROUTINES***
7098	
7105	END OF PASS ROUTINE
7227	SAVE REGISTERS ROUTINE
7254	FLOAT 1 AND 0
7295	CLEAR MEMORY ROUTINE
7330	LOCAL TRAPS
7347	CLEAR DISK ROUTINE
7360	CHECK DISK STATUS ROUTINES
7458	WAIT LOOP
7503	SAVE ROUTINE
7533	WRITE CHECK ROUTINE
7577	COMPARE ROUTINE
7629	WRITE CHECK DATA
7675	CRC GENERATION ROUTINE
7769	SIMULATED DISK SETUP
7816	CHECK HCE ROUTINE
7964	EXIT WRT HEADER & DATA ROUTINE
7998	JAM CURRENT CYLINDER ROUTINE
8037	ECC GENERATION AND COMPARISON ROUTINE
8267	ECC GENERATION CONTROL ROUTINE
8327	SOFTWARE DISK DATA ECC GEN. ROUTINE
8369	RH BASE ADDRESS CHANGE ROUTINE
8449	DISK SIMULATION
9453	
9454	***SYSMAC LIBRARY ROUTINES***
9455	
9461	SCOPE HANDLER ROUTINE

E04

MAINDEC-11-DZRPT-D, RJPO4 DISKLESS RH11 TEST-PART 2  
DZRPTD.SUB

MACY11 27(663) 7-OCT-75 17:33

SEQ 0042

TABLE OF CONTENTS

9536	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
9604	TYPE ROUTINE
9651	TTY INPUT ROUTINE
9754	READ AN OCTAL NUMBER FROM THE TTY
9808	ERROR HANDLER ROUTINE
9854	ERROR MESSAGE TIMEOUT ROUTINE
9911	BINARY TO OCTAL (ASCII) AND TYPE
9989	TRAP DECODER
10004	TRAP TABLE
10025	POWER DOWN AND UP ROUTINES

10  
11  
12

```

.TITLE MAINDEC-11-DZRPT-D, RJP04 DISKLESS RH11 TEST-PART 2
;#COPYRIGHT (C) 1975
;#DIGITAL EQUIPMENT CORP.
;#MAYNARD, MASS. 01754
;
;PROGRAM BY SUB MALLICK, PETE BLACKSTONE, JIM KELLY
;
;THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;PACKAGE (MAINDEC-11-DZQAC-A3).
;

```

```

;\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:
;\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:
;\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:\/:\:

```

;DRIVE MUST BE LOCKED ON PORT A OR PORT B

```

;:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\
;:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\
;:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\:\

```

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

```

;INTERNAL PROGRAM MACROS BEGIN HERE
;*****

```

```

;*****
;#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.
;*****

```

```

.SBTTL OPERATIONAL SWITCH SETTINGS
;
; SWITCH USE
;-----
; 15 HALT ON ERROR
; 14 LOOP ON TEST
; 13 INHIBIT ERROR TYPEOUTS
; 11 INHIBIT ITERATIONS
; 10 BELL ON ERROR
; 9 LOOP ON ERROR
; 8 LOOP ON TEST IN SMR(7:0)
; 7 STOP FURTHER COMPARES IF SM08 IS LOW
; 6 ECC TEST-COMPARE END RESULTS ONLY IF SM08 IS LOW

```

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  
101  
102  
103  
104  
105  
106  
107

```

.SBTTL BASIC DEFINITIONS

:*INITIAL ADDRESS OF THE STACK POINTER *** 1000 ***
001000 STACK= 1000
.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
177776 PS= 177776 ;;PROCESSOR STATUS WORD
.EQUIV PS,PSW
177774 STKLMT= 177774 ;;STACK LIMIT REGISTER
177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
177570 SMR= 177570 ;;SWITCH REGISTER
177570 DISPLAY=SMR

:*GENERAL PURPOSE REGISTER DEFINITIONS
000000 R0= X0 ;;GENERAL REGISTER
000001 R1= X1 ;;GENERAL REGISTER
000002 R2= X2 ;;GENERAL REGISTER
000003 R3= X3 ;;GENERAL REGISTER
000004 R4= X4 ;;GENERAL REGISTER
000005 R5= X5 ;;GENERAL REGISTER
000006 R6= X6 ;;GENERAL REGISTER
000007 R7= X7 ;;GENERAL REGISTER
.EQUIV R6,SP ;;STACK POINTER
.EQUIV R7,PC ;;PROGRAM COUNTER

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

:*"SWITCH REGISTER" SWITCH DEFINITIONS
100000 SW15= 100000
040000 SW14= 40000
020030 SW13= 20000
010000 SW12= 10000
004000 SW11= 4000
002000 SW10= 2000
001000 SW09= 1000
000400 SW08= 400
000200 SW07= 200
000100 SW06= 100
000040 SW05= 40
000020 SW04= 20
000010 SW03= 10
000004 SW02= 4
000002 SW01= 2
000001 SW00= 1
.EQUIV SW09,SW9
  
```

108		.EQUIV	S408,S408
109		.EQUIV	S 07,S417
110		.EQUIV	S 06,S416
111		.EQUIV	S405,S415
112		.EQUIV	S404,S414
113		.EQUIV	S403,S413
114		.EQUIV	S 02,S412
115		.EQUIV	S 01,S411
116		.EQUIV	S400,S40

118		. #DATA BIT DEFINITIONS (BIT00 TO BIT15)	
119	100000	BIT15=	100000
120	040000	BIT14=	40000
121	020000	BIT13=	20000
122	010000	BIT12=	10000
123	004000	BIT11=	4000
124	002000	BIT10=	2000
125	001000	BIT09=	1000
126	000400	BIT08=	400
127	000200	BIT07=	200
128	000100	BIT06=	100
129	000040	BIT05=	40
130	000020	BIT04=	20
131	000010	BIT03=	10
132	000004	BIT02=	4
133	000002	BIT01=	2
134	000001	BIT00=	1

135		.EQUIV	BIT09,BIT9
136		.EQUIV	BIT08,BIT8
137		.EQUIV	BIT07,BIT7
138		.EQUIV	BIT06,BIT6
139		.EQUIV	BIT05,BIT5
140		.EQUIV	BIT04,BIT4
141		.EQUIV	BIT03,BIT3
142		.EQUIV	BIT02,BIT2
143		.EQUIV	BIT01,BIT1
144		.EQUIV	BIT00,BIT0

146		. #BASIC "CPU" TRAP VECTOR ADDRESSES	
147	000004	ERRVEC= 4	:: TIME OUT AND OTHER ERRORS
148	000010	RESVEC= 10	:: RESERVED AND ILLEGAL INSTRUCTIONS
149	000014	TBITVEC=14	:: "T" BIT
150	000014	TRTVEC= 14	:: TRACE TRAP
151	000014	BPTVEC= 14	:: BREAKPOINT TRAP (BPT)
152	000020	IOTVEC= 20	:: INPUT/OUTPUT TRAP (IOT) **SCOPE**
153	000024	PMRVEC= 24	:: POWER FAIL
154	000030	EMTVEC= 30	:: EMULATOR TRAP (EMT) **ERROR**
155	000034	TRAPVEC=34	:: "TRAP" TRAP
156	000060	TKVEC= 60	:: TTY KEYBOARD VECTOR
157	000064	TPVEC= 64	:: TTY PRINTER VECTOR
158	000240	PIRQVEC=240	:: PROGRAM INTERRUPT REQUEST VECTOR
159			



```

160
161      .SBTTL TRAP CATCHER
162
163      000000      .=0
164      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
165      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
166      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
167
168      .SBTTL STARTING ADDRESS(ES)
169      000200      .=200
170
171      000200 000137 017346      JMP      @#BEGIN      ;;JUMP TO STARTING ADDRESS OF PROGRAM
172      ;& STARTING ADDRESSES
173      000210 000137 017336      JMP      @#BEGIN2      ;JUMP SELECT TEST
174
175      ;*STARTING ADDRESS 200 FOR NORMAL STARTS
176      ;*THIS WILL TEST ALL RPO4'S ON THE SYSTEM A SINGLE DRIVE AT A TIME
177      ;*
178      ;*STARTING ADDRESS 210 WILL TEST ONLY ONE SPECIFIED DRIVE
179

```

```

180
181      .SBTTL MEMORY MANAGEMENT DEFINITIONS
182
183      ;#KT11 VECTOR ADDRESS
184
185      000250      MMVEC= 250
186
187      ;#KT11 STATUS REGISTER ADDRESSES
188
189      177572      SR0= 177572
190      177574      SR1= 177574
191      177576      SR2= 177576
192      172516      SR3= 172516
193
194      ;#KERNAL "I" PAGE DESCRIPTOR REGISTERS
195
196      172300      KIPDR0= 172300
197      172302      KIPDR1= 172302
198      172304      KIPDR2= 172304
199      172306      KIPDR3= 172306
200      172310      KIPDR4= 172310
201      172312      KIPDR5= 172312
202      172314      KIPDR6= 172314
203      172316      KIPDR7= 172316
204
205      ;#KERNAL "I" PAGE ADDRESS REGISTERS
206
207      172340      KIPAR0= 172340
208      172342      KIPAR1= 172342
209      172344      KIPAR2= 172344
210      172346      KIPAR3= 172346
211      172350      KIPAR4= 172350
212      172352      KIPAR5= 172352
213      172354      KIPAR6= 172354
214      172356      KIPAR7= 172356
215
216      001110      .=1110
217

```

```

218 ;*****
219
220 .SBTTL COMMON TAGS
221
222 ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
223 ;*USED IN THE PROGRAM.
224
225 000046 000046 .=46
226 000046 044034 $ENDAD ;;LOGICAL END OF PROGRAM
227
228 001100 .=1100
229
230 $CHTAG: ; START OF COMMON TAGS
231 $PASS: .WORD 0 ; CONTAINS PASS COUNT
232 $STNM: .BYTE 00 ; CONTAINS THE TEST NUMBER
233 $ERFLG: .BYTE 00 ; CONTAINS ERROR FLAG
234 $ICNT: .WORD 00 ; CONTAINS SUBTEST ITERATION COUNT
235 $LPADR: .WORD 00 ; CONTAINS SCOPE LOOP
236 $LPERR: .WORD 00 ; CONTAINS SCOPE RETURN FOR ERRORS
237 $ERTTL: .WORD 00 ; CONTAINS TOTAL ERRORS DETECTED
238 $ITEMB: .BYTE 00 ; CONTAINS ITEM CONTROL BYTE
239 $ERMAX: .BYTE 001 ; CONTAINS MAX. ERRORS PER TEST
240 $ERRPC: .WORD 00 ; CONTAINS PC OF LAST ERROR INSTRUCTION
241 $GDADR: .WORD 00 ; CONTAINS OF 'GOOD' DATA
242 $BDADR: .WORD 00 ; CONTAINS OF 'BAD' DATA
243 $GDADR: .WORD 00 ; CONTAINS 'GOOD' DATA
244 $BDADR: .WORD 00 ; CONTAINS 'BAD' DATA
245 $WDRD: .WORD 0,0,0 ; RESERVED--NOT TO BE USED
246 $TKS: 177560 ; TTY KBD STATUS
247 $TKB: 177562 ; TTY KBD BUFFER
248 $TPS: 177564 ; TTY PRINTER STATUS REG.
249 $TPB: 177566 ; TTY PRINTER BUFFER REG.
250 $NULL: .BYTE 0 ; CONTAINS NULL CHARACTER FOR FILLS
251 $FILLS: .BYTE 002 ; CONTAINS # OF FILLER CHARACTERS REQUIRED
252 $FILLC: .BYTE 012 ; INSERT FILL CHARS. AFTER A "LINE FEED"
253 $TPFLG: .BYTE 00 ; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
254 $REGAD: .WORD 0 ; CONTAINS THE FROM
255 ; WHICH ($REGO) WAS OBTAINED
256 $REG0: .WORD 0 ; CONTAINS (($REGAD)+0)
257 $REG1: .WORD 00 ; CONTAINS (($REGAD)+2)
258 $REG2: .WORD 00 ; CONTAINS (($REGAD)+4)
259 $REG3: .WORD 00 ; CONTAINS (($REGAD)+6)
260 $REG4: .WORD 00 ; CONTAINS (($REGAD)+10)
261 $REG5: .WORD 00 ; CONTAINS (($REGAD)+12)
262 $TMP0: .WORD 00 ; USER DEFINED
263 $TMP1: .WORD 00 ; USER DEFINED
264 $TMP2: .WORD 00 ; USER DEFINED
265 $TMP3: .WORD 00 ; USER DEFINED
266 $TMP4: .WORD 00 ; USER DEFINED
267 $TMP5: .WORD 00 ; USER DEFINED
268 $TIMES: 0 ; MAX. NUMBER OF ITERATIONS
269 $ESCAPE: 0 ; ESCAPE ON ERROR
270 $BELL: .ASCIZ <207><377><377> ; CODE FOR BELL
271 $QUES: .ASCII /?/ ; QUESTION MARK

```

L04

MAINDEC-11-DZRPT-D, RJPO4 DISKLESS RH11 TEST-PART 2  
DZRPTC.ITM COMMON TAGS

MACY11 27(663) 7-OCT-75 17:33 PAGE 7

SEQ 0049

272 001215 015  
273 001216 000012

SCRLF: .ASCII <15>  
SLF: .ASCIZ <12>

:::CARRIAGE RETURN  
:::LINE FEED

274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
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

\*\*\*\*\*  
.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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.  
; \*NOTE1: IF SITEMB 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

001220 \$ERRTB:

; ITEM1  
EM1 ;: WRONG DATA IN READING OR WRITING HARDWARE REGISTER  
DH1 ;: PC  
;: REG. ADDR.  
;: GOOD DATA  
;: RECEIVED DATA  
DT1 ;: \$ERRPC, \$STNM, REGADR, \$GDDAT, \$BDDAT  
DF1 ;: 0,0,0,0,0

001220 002130  
001222 005444

; ITEM2  
EM2 ;: ERROR ON DATA COMMAND  
DH33 ;: PC  
;: PC OF JSR  
;: TEST NO  
;: WORD NO.  
;: GOOD DATA  
;: CONTENTS OF RHCS1  
;: CONTENTS OF RHDS1  
;: CONTENTS OF RHER1  
DT33 ;: \$ERRPC, PCJSR, \$STNM, ERWORD, \$GDDAT, CS1, DS1, ER1  
DF33 ;: 0,0,0,1,0,0,0,0

001234 012344  
001236 012662

; ITEM3  
EM2 ;: ERROR ON DATA COMMAND  
DH32 ;: PC

001240 002213  
001242 010241

328					: PC OF JSR
329					: TEST NO
330					: WORD NO.
331					: GOOD DATA
332					: BAD DATA
333					: CONTENTS OF RHCS1
334					: CONTENTS OF RHDS1
335					: CONTENTS OF RHER1
336					
337	001244	012320		DT32	: SERRPC, PCJSR, \$STNM, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1
338	001246	012651		DF32	: 0,0,0,1,0,0,0,0,0,
339					
340					
341					
342	001250	002213		EM2	: ERROR ON DATA COMMAND
343					
344	001252	010043		DH31	
345					: PC
346					: TEST NO
347					: WORD NO.
348					: GOOD DATA
349					: BAD DATA
350					: CONTENTS OF RHCS1
351					: CONTENTS OF RHDS1
352					: CONTENTS OF RHER1
353	001254	012276		DT31	: SERRPC, \$STNM, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1
354	001256	012641		DF31	: 0,0,1,0,0,0,0,0,
355					
356					
357					
358					
359	001260	000000		0	:
360	001262	000000		0	:
361	001264	012276		DT31	: SERRPC, \$STNM, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1
362	001266	012641		DF31	: 0,0,1,0,0,0,0,0,
363					
364					
365					
366	001270	002242		EM6	: ERROR ON WRITE HEADER AND DATA
367					
368	001272	010241		DH32	
369					: PC
370					: PC OF JSR
371					: TEST NO
372					: WORD NO.
373					: GOOD DATA
374					: BAD DATA
375					: CONTENTS OF RHCS1
376					: CONTENTS OF RHDS1
377					: CONTENTS OF RHER1
378	001274	012320		DT32	: SERRPC, PCJSR, \$STNM, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1
379	001276	012651		DF32	: 0,0,0,1,0,0,0,0,0,
380					
381					

382					
383					
384	001300	002242		EM6	; ERROR ON WRITE HEADER AND DATA ; PC ; TEST NO ; WORD NO. ; GOOD DATA ; BAD DATA ; SERRPC, \$TSTNM, ERWORD, \$GDDAT, \$BDDAT ; 0,0,1,0,0, /
385	001302	005562		DH2	
386					
387					
388					
389					
390	001304	012006		DT3	; SERRPC, \$TSTNM, ERWORD, \$GDDAT, \$BDDAT ; 0,0,1,0,0,
391	001306	012523		DF3	
392					
393					
394					
395	001310	000000		0	; SERRPC, \$TSTNM, ERWORD, \$GDDAT, \$BDDAT ; 0,0,1,0,0,
396	001312	000000		0	
397	001314	012006		DT3	
398	001316	012523		DF3	
399					
400					
401					
402	001320	002301		EM11	; CONTROLLER OR DRIVE STATUS ; PC ; TEST NO ; FAILING REG. ADDR ; CONTENTS OF RHCS1 ; CONTENTS OF RHCS2 ; CONTENTS OF RHDS1 ; CONTENTS OF RHER1 ; SERRPC, \$TSTNM, \$BDADR, CS1, CS2, DS1, ER1 ; 0,0,0,0,0,0
403	001322	005674		DH11	
404					
405					
406					
407					
408					
409					
410	001324	012022		DT11	
411	001326	012530		DF11	
412					
413					
414					
415	001330	002301		EM11	; WRONG DATA FROM SILO ; PC ; REG. ADDR ; GOOD DATA ; RECEIVED DATA ; SERRPC, REGADR, \$GDDAT, \$BDDAT ; 0,0,0,0
416					
417	001332	005444		DH1	
418					
419					
420					
421	001334	011760		DT1	; SERRPC, REGADR, \$GDDAT, \$BDDAT ; 0,0,0,0
422	001336	012512		DF1	
423					
424					
425					
426	001340	000000		0	; SERRPC, \$TSTNM, REGADR, \$GDDAT, \$BDDAT ; 0,0,0,0,0
427	001342	000000		0	
428	001344	011760		DT1	
429	001346	012512		DF1	
430					
431					
432					
433	001350	002334		EM14	; REGISTER FAILED ; PC ; FAILING REG. ADDR
434	001352	006051		DH14	
435					

436				: CONTENTS OF FAILING REG.
437				: CONTENTS OF RHCS1
438				: CONTENTS OF RHCS2
439				: CONTENTS OF RHDS1
440				: CONTENTS OF RHER1
441	001354	012042	DT14	: SERRPC, \$BODR, \$BODAT, CS1, CS2, DS1, ER1
442	001356	012537	DF14	: 0,0,0,0,0,0,0
443				
444				
445			; ITEM15	
446	001360	002354	EM15	: SPECIFIED REG. NON EXISTANT SO ABORT
447				: PROGRAM
448	001362	006250	DM15	: PC
449				: ADDR. OF REG
450	001364	012064	DT15	: SERRPC, TEMP1
451	001366	012547	DF15	: 0,0
452				
453				
454			; ITEM16	
455	001370	002425	EM16	: WAIT LOOP FAILED
456	001372	006300	DM16	: PC
457				: MAT PC
458				: BIT WANTED
459				: REG. ADR.
460				: REG. CONT.
461	001374	012074	DT16	: SERRPC, STMP3, STMP1, STMP0, \$BODAT
462	001376	012552	DF16	: 0,0,0,0
463				
464				
465			; ITEM17	
466	001400	002446	EM17	: WRITE CHECK FAILING
467	001402	006436	DM17	: PC
468				: TEST NO
469				: CONTENTS OF RHBA
470				: CONTENTS OF RHDB
471				: CONTENTS OF RHMC
472				: CONTENTS OF RHCS1
473				: CONTENTS OF RHCS2
474	001404	012112	DT17	: SERRPC, \$TSTNM, \$BA, DB, MC, CS1, CS2
475	001406	012557	DF17	: 0,0,0,0,0,0,0
476				
477				
478			; ITEM20	
479	001410	002472	EM20	: REGISTER FAILING
480	001412	006613	DM20	: PC
481				: TST NO
482				: CONTENTS OF RHER1
483				: CONTENTS OF RHER2
484				: CONTENTS OF RHER3
485				: CONTENTS OF RHAS
486				: CONTENTS OF RHDS1
487	001414	012132	DT20	: SERRPC, TSTNM ER1, ER2, ER3, AS, DS1
488	001416	012566	DF20	: 0,0,0,0,0,0,0
489				



490			;ITEM21		
491					
492	001420	002513		EM21	: INTERRUPT FAILING
493	001422	006767		DH21	: PC
494					: TEST NO
495					: CONTENTS OF RHCS1
496					: CONTENTS OF RHAS
497					: CONTENTS OF RHDS1
498	001424	012152		DT21	: \$ERRPC, TSTNM, CS1, AS, DS1
499	001426	012575		DF21	: 0,0,0,0,0
500					
501					
502			;ITEM22		
503	001430	002535		EM22	: MISMATCH IN DRIVE PRESENT
504					: LOOKING AT RHAS AND RHCS2-NED(BIT#12)
505					: DRIVE PRESENT DO NOT AGREE
506					: NOTE: ON DUAL PORT SYSTEM
507					: DRIVE ON OTHER PORT WILL NOT GIVE NED
508					: HENCE THERE WILL BE A MISMATCH
509					: 17777-MEANS NOT PRESENT
510	001432	007103		DH22	: PC
511					: TEST NO
512					: RHAS UNIT
513					: RHCS2 UNIT
514					
515	001434	012166		DT22	: \$ERRPC, TSTNMS, SGDDAT, SBDDAT
516	001436	012602		DF22	: 0,0,0,0
517					
518					
519			;ITEM23		
520	001440	000000		0	: MISMATCH IN DRIVE PRESENT
521					: LOOKING AT RHAS AND RHCS2-NED(BIT#12)
522					: DRIVE PRESENT DO NOT AGREE
523					: 17777-MEANS NOT PRESENT
524	001442	000000		0	: PC
525					: TEST NO
526					: RHAS UNIT
527					: RHCS2 UNIT
528					
529	001444	012166		DT22	: \$ERRPC, TSTNMS, SGDDAT, SBDDAT
530	001446	012602		DF22	: 0,0,0,0
531					
532					
533			;ITEM 24		
534					
535	001450	003136		EM24	: LOOK AHEAD REGISTER AT THE
536					: BEGINNING OF A SECTOR IS IN
537					: ERROR
538	001452	007177		DH24	: PC
539					: RHDST
540					: BAD RHLA
541					: GOOD RHLA
542					: SECTOR NO
543					: SECTOR CLOCK

544	001454	012200	DT24	:SERRPC,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3
545	001456	012606	DF24	:0,0,0,0,0
546				
547			;ITEM 25	
548	001460	003231	EM25	:LOOK AHEAD REGISTER IS
549				:IN ERROR
550				
551	001462	007177	DH24	:PC
552				:RHDST
553				:BAD RHLA
554				:GOOD RHLA
555				:SECTOR NO
556				:SECTOR CLOCK
557	001464	012200	DT24	:SERRPC,DST,\$BDDAT,\$TMP1,\$TMP2,\$TMP3
558	001466	012606	DF24	:0,0,0,0,0
559			;ITEM26	
560	001470	002301	EM11	:CONTROLLER OR DRIVE STATUS
561				
562	001472	007355	DH26	:PC
563				:PC OF JSR
564				:FAILING REGISTER ADDRESS
565				:CONTENTS OF RHCS1
566				:CONTENTS OF RHCS2
567				:CONTENTS OF RHDS1
568				:CONTENTS OF RHER1
569				
570	001474	012220	DT26	:SERRPC,PCJSR,\$BDADR,CS1,CS2,DS1,ER1
571	001476	012615	DF26	:0,0,0,0,0,0,
572				
573				
574				
575			;ITEM27	
576	001500	002130	EM1	:ERROR IN READING OR WRITING HARDWARE REGISTER
577				
578	001502	007553	DH27	:PC
579				:PC OF JSR
580				:TEST NUMBER
581				:FAILING REGISTER
582				:GOOD DATA
583				:RECEIVED DATA
584				
585	001504	012242	DT27	:SERRPC,PCJSR,TSTNM,REGADR,\$GDDAT,\$BDDAT
586	001506	012625	DF27	:0,0,0,0,0,0
587				
588				
589				
590			;ITEM30	
591	001510	003271	EM30	:CURRENT CYLINDER DOES NOT REFLECT DESIRED CYLINDER REG.
592	001512	007711	DH30	:PC
593				:PC OF JSR
594				:REGISTER ADDRESS
595				:GOOD DATA
596				:BAD DATA
597				

598	001514	012260	DT30	:SERRPC, PCJSR, REGADR, \$GDDAT, \$BDDAT
599	001516	012633	DF30	:0,0,0,0,0
600				
601				
602				
603				
604	001520	003413	EM31	:ECC GENERATED IS INCORRECT
605				:EVERY WORD IN THIS SECTOR IS GIVEN IN "DATA USED"
606				
607	001522	010655	DH34	:PC
608				:TEST NUMBER
609				:GOOD ECC1
610				:GOOD EC2C
611				:WRITTEN ECC1
612				:WRITTEN ECC2
613				:DATA USED
614				
615	001524	012366	DT34	:SERRPC, TSTNM, GECC1, GECC2, MECC1, MECC2, DISK
616				
617	001526	012672	DF34	:0,0,0,0,0,0,0
618				
619				
620				
621	001530	003536	EM32	:ON READ COMMAND AFTER DATA AND ECC HAVE BEEN READ
622				:ECC REGISTER OR RHER1 IS IN ERROR
623				:ONLY LOWER 11 BITS OF PATTERN REGISTER
624				:CAN BE READ
625				:THIS SHUOLD MATCH LOWER 11 BITS OF ECC1
626				
627	001532	011030	DH35	:PC
628				:TEST NUMBER
629				:GOOD ECC1
630				:GOOD ECC2
631				:PATTERN REGISTER
632				:RHER1
633				
634	001534	012406	DT35	:SERRPC, TSTNM, GECC1, GECC2, EC2, ER1
635				
636	001536	012701	DF35	:0,0,0,0,0,0
637				
638				
639				
640				
641	001540	004025	EM33	:HIGH COUNT BIT NOT HIGH AFTER 38859 CLOCKS
642	001542	011225	DH36	:PC
643				:PC OF JSR
644				:TEST NUMBER
645				:RHMR
646				:POSITION REG.
647				:PATTERN REGISTER
648				
649	001544	012430	DT36	:SERRPC, PCJSR, TSTNM, MR, EC1, EC2
650				
651	001546	012711	DF36	:0,0,0,0,0,0

652					
653					
654	001550	004077		EM34	
655					: ZERO DETECT BIT NOT HIGH WHEN THE
656					: 32 BIT ECC REGISTER HAS ITS 21 BITS
657					: OF ZEROS
658					: ERROR PRINTOUT WILL CONTINUE TILL
659	001552	011225		DH36	: ZERO DETECT BIT IS HIGH
660					: PC
661					: PC OF JSR
662					: TEST NUMBER
663					: RHMR
664					: POSITION REG.
665					: PATTERN REGISTER
666	001554	012430		DT36	: \$ERRPC, PCJSR, TSTNM, MR, EC1, EC2
667					
668	001556	012711		DF36	: 0,0,0,0,0,0
669					
670					
671					
672					
673	001560	004172		EM35	: POSITION REGISTER OR 11 BITS OF
674					: PATTERN REGISTER INCORRECT
675					: LOWER 11 BITS OF PATTERN REGISTER
676					: SHOULD MATCH LOWER 11 BITS OF GOOD ECC1
677					: DATA ENVELOPE AND N-CODE ZEROS ARE IN DECIMAL
678					
679	001562	011363		DH37	: PC
680					: TEST NUMBER
681					: ECC POSITION
682					: GOOD POSITION
683					: GOOD ECC1
684					: GOOD ECC2
685					: ECC PATTERN
686					: DATA ENVELOPE
687					: N-CODE ZEROS
688					
689	001564	012446		DT37	: \$ERRPC, TSTNM, EC1, POSITI, GECC1, GECC2, EC2, DATENV, ZCODE
690					
691	001566	012717		DF37	: 0,0,0,0,0,0,0,0,0
692					
693					
694					
695					
696	001570	004471		EM36	: ON A READ COMMAND WITH NON CORRECTABLE
697					: ERROR INSERTED DCK AND ECH SHOULD BE SET
698	001572	011030		DH35	: PC
699					: TEST NUMBER
700					: GOOD ECC1
701					: GOOD ECC2
702					: PATTERN REGISTER
703					: POSITION REGISTER
704					: RHER1
705					

706	001574	012406	DT35	;SERRPC, TSTNM, GECC1, GECC2, EC2, EC1, ER1
707				
708	001576	012701	DF35	;0,0,0,0,0,0,0
709				
710				
711				
712				
713				
714				
715				;ITEM37
716	001600	004657	EM37	;ERROR ON DATA COMMAND ;WITH A16 A17 USED
717				
718				
719	001602	010043	DM31	;PC ;TEST NO ;WORD NO. ;GOOD DATA ;BAD DATA ;CONTENTS OF RHCS1 ;CONTENTS OF RHDS1 ;CONTENTS OF RHER1
720				
721				
722				
723				
724				
725				
726				
727				
728	001604	012276	DT31	;SERRPC, \$TSTNM, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1
729	001606	012641	DF31	;0,0,1,0,0,0,0,0,
730				
731				
732				
733				;ITEM40
734	001610	000000	0	:
735	001612	000000	0	:
736	001614	012276	DT31	;SERRPC, \$TSTNM, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1
737	001616	012641	DF31	;0,0,1,0,0,0,0,0,
738				
739				
740				
741				
742				;ITEM41
743	001620	004745	EM40	;THERE WAS A READ/WRITE HEADER & DATA ;ERROR DURING 'DTE' TEST SETUP - THE ;TEST IS ABORTED AT THAT POINT
744				
745				
746	001622	011602	DM40	;PC ;TEST NO ;FAILING REGISTER ADDRESS ;CONTENTS OF RHCS1 ;CONTENTS OF RHCS2 ;CONTENTS OF RHDS1 ;CONTENTS OF RHER1
747				
748				
749				
750				
751				
752				
753	001624	012472	DT40	;SERRPC, \$TSTNM, \$BODR, CS1, CS2, DS1, ER1
754	001626	012730	DF40	;0,0,0,0,0,0

755				
756			; ITEM 42	
757	001630	012740	EM42	
758	001632	014405	DH42	
759	001634	014674	DT42	; ERROR PC, TEST NUMBER, REGISTER ADDRESS.
760	001636	014732	DF42	
761				
762			; ITEM 43	
763				
764	001640	013023	EM43	
765	001642	014405	DH42	
766	001644	014674	DT42	; ERROR PC, TEST NUMBER, REGISTER ADDRESS.
767	001646	014732	DF42	
768				
769			; ITEM 44	
770				
771	001650	013067	EM44	
772	001652	014462	DH44	
773	001654	014704	DT44	; ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
774	001656	014735	DF44	
775				
776			; ITEM 45	
777	001660	013124	EM45	
778	001662	014462	DH44	
779	001664	014704	DT44	; ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
780	001666	014735	DF44	
781				
782			; ITEM 46	
783				
784	001670	013154	EM46	
785	001672	014462	DH44	
786	001674	014704	DT44	; ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
787	001676	014735	DF44	
788				
789			; ITEM 47	
790				
791	001700	013203	EM47	
792	001702	014405	DH42	
793	001704	014674	DT42	; ERROR PC, TEST NUMBER, REGISTER ADDRESS.
794	001706	014732	DF42	
795				
796			; ITEM 50	
797				
798	001710	013235	EM50	
799	001712	014462	DH44	
800	001714	014704	DT44	; ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
801	001716	014735	DF44	
802				
803			; ITEM 51	
804				
805	001720	013273	EM51	
806	001722	014462	DH44	
807	001724	014704	DT44	; ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
808	001726	014735	DF44	

809			
810			;ITEM 52
811			
812	001730	013314	EM52
813	001732	014462	DH44
814	001734	014704	DT44
815	001736	014735	DF44
816			
817			;ITEM 53
818			
819	001740	013354	EM53
820	001742	014462	DH44
821	001744	014704	DT44
822	001746	014735	DF44
823			
824			;ITEM 54
825			
826	001750	013416	EM54
827	001752	014576	DH54
828	001754	014720	DT54
829	001756	014742	DF54
830			
831			;ITEM 55
832			
833	001760	013455	EM55
834	001762	014462	DH44
835	001764	014704	DT44
836	001766	014735	DF44
837			
838			;ITEM 56
839			
840	001770	013470	EM56
841	001772	014462	DH44
842	001774	014704	DT44
843	001776	014735	DF44
844			
845			;ITEM 57
846			
847	002000	013506	EM57
848	002002	014462	DH44
849	002004	014704	DT44
850	002006	014735	DF44
851			
852			;ITEM 60
853			
854	002010	013535	EM60
855	002012	014462	DH44
856	002014	014704	DT44
857	002016	014735	DF44
858			
859			;ITEM 61
860			
861	002020	013535	EM60
862	002022	014462	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.

863	002024	014704	DT44	;ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
864	002026	014735	DF44	
865				
866				;ITEM 62
867				
868	002030	013674	EM62	
869	002032	014462	DH44	
870	002034	014704	DT44	;ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
871	002036	014735	DF44	
872				
873				;ITEM 63
874				
875	002040	013751	EM63	
876	002042	014462	DH44	
877	002044	014704	DT44	;ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
878	002046	014735	DF44	
879				
880				;ITEM 64
881				
882	002050	014027	EM64	
883	002052	014462	DH44	
884	002054	014704	DT44	;ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
885	002056	014735	DF44	
886				
887				;ITEM 65
888				
889	002060	014063	EM65	
890	002062	014462	DH44	
891	002064	014704	DT44	;ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
892	002066	014735	DF44	
893				
894				;ITEM 66
895				
896	002070	014145	EM66	
897	002072	014462	DH44	
898	002074	014704	DT44	;ERROR PC, TEST NUMBER, REGISTER ADDRESS CORRECT, ACTUAL.
899	002076	014735	DF44	
900				
901				;ITEM 67
902				
903	002100	014217	EM67	
904	002102	014576	DH54	
905	002104	014720	DT54	;ERROR PC, TEST NUMBER.
906	002106	014742	DF54	
907				
908				;ITEM 70
909				
910	002110	014304	EM70	
911	002112	014576	DH54	
912	002114	014720	DT54	;ERROR PC, TEST NUMBER.
913	002116	014742	DF54	
914				
915				;ITEM 71
916				



917	002120	014356
918	002122	014462
919	002124	014704
920	002126	014735

EM71  
DM44  
DT44  
DF44

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

```

921
922 ;*****
923 ;
924 ;ERROR AND MESSAGE TABLE CONDIMENTS
925 ;
926 ;*****
927
928
929
930
931 002130 051127 047117 020107 EM1: .ASCIZ /WRONG DATA IN READING OR WRITING HARDWARE REGISTER/
932 002136 040504 040524 044440
933 002144 020116 042522 042101
934 002152 047111 020107 051117
935 002160 053440 044522 044524
936 002166 043516 044040 051101
937 002174 053504 051101 020105
938 002202 042522 044507 052123
939 002210 051105 000
940 002213 105 051122 051117 EM2: .ASCIZ /ERROR ON DATA COMMAND/
941 002220 047440 020116 042040
942 002226 052101 020101 047503
943 002234 046515 047101 000104
944 002242 051105 047522 020122 EM6: .ASCIZ /ERROR ON WRITE HEADER AND DATA/
945 002250 047117 053440 044522
946 002256 042524 044040 040505
947 002264 042504 020122 047101
948 002272 020104 040504 040524
949 002300 000
950 002301 103 047117 051124 EM11: .ASCIZ /CONTROLLER OR DRIVE STATUS/
951 002306 046117 042514 020122
952 002314 051117 042040 044522
953 002322 042526 051440 040524
954 002330 052524 000123
955 002334 042522 044507 052123 EM14: .ASCIZ /REGISTER FAILED/
956 002342 051105 043040 044501
957 002350 042514 000104
958 002354 050123 041505 043111 EM15: .ASCIZ /SPECIFIED REGISTER NON EXISTANT SO ABORT/
959 002362 042511 020104 042522
960 002370 044507 052123 051105
961 002376 047040 047117 042440
962 002404 044530 052123 047101
963 002412 020124 047523 040440
964 002420 047502 052122 000
965 002425 127 044501 020124 EM16: .ASCIZ /WAIT LOOP FAILED/
966 002432 047514 050117 043040
967 002440 044501 042514 000104
968 002446 051127 052111 020105 EM17: .ASCIZ /WRITE CHECK FAILING/
969 002454 044103 041505 020113
970 002462 040506 046111 047111
971 002470 000107
972 002472 042522 044507 052123 EM20: .ASCIZ /REGISTER FAILING/
973 002500 051105 043040 044501
974 002506 044514 043516 000

```

975	002513	111	052116	051105	EM21: .ASCIZ /INTERRUPT FAILING/
976	002520	052522	052120	043040	
977	002526	044501	044514	043516	
978	002534	000			
979	002535	105	051122	051117	EM22: .ASCII /ERROR ON DRIVE PRESENT/<15><12>
980	002542	047440	020116	051104	
981	002550	053111	020105	051120	
982	002556	051505	047105	006524	
983	002564	012			
984	002565	124	042510	052440	.ASCII /THE UNIT NO'S FOUND BY SETTING RHAS/<15><12>
985	002572	044516	020124	047516	
986	002600	051447	043040	052517	
987	002606	042116	041040	020131	
988	002614	042523	052124	047111	
989	002622	020107	044122	051501	
990	002630	005015			
991	002632	047504	047040	052117	.ASCII /DO NOT AGREE WITH THE UNIT NO. FOUND FROM/<15><12>
992	002640	040440	051107	042505	
993	002646	053440	052111	020110	
994	002654	044124	020105	047125	
995	002662	052111	047040	027117	
996	002670	043040	052517	042116	
997	002676	043040	047522	006515	
998	002704	012			
999	002705	122	041510	031123	.ASCII /RHCS2-'NED' BIT #12/<15><12>
1000	002712	023455	042516	023504	
1001	002720	041040	052111	021440	
1002	002726	031061	005015		
1003	002732	033461	033467	033467	.ASCII /177777-MEANS NO UNIT FOUND/<15><12>
1004	002740	046455	040505	051516	
1005	002746	047040	020117	047125	
1006	002754	052111	043040	052517	
1007	002762	042116	005015		
1008	002766	047516	042524	020072	.ASCII /NOTE: ON DUAL PORT SYSTEM, DRIVE ON OTHER PORT WILL NOT GIVE/<15><12>
1009	002774	047117	042040	040525	
1010	003002	020114	047520	052122	
1011	003010	051440	051531	042524	
1012	003016	026115	042040	044522	
1013	003024	042526	047440	020116	
1014	003032	052117	042510	020122	
1015	003040	047520	052122	053440	
1016	003046	046111	020114	047516	
1017	003054	020124	044507	042526	
1018	003062	005015			
1019	003064	047047	042105	026047	.ASCIZ /'NED', HENCE THERE WILL BE AN EXTRA DRIVE/
1020	003072	044040	047105	042503	
1021	003100	052040	042510	042522	
1022	003106	053440	046111	020114	
1023	003114	042502	040440	020116	
1024	003122	054105	051124	020101	
1025	003130	051104	053111	000105	
1026	003136	047514	045517	040440	EM24: .ASCIZ /LOOK AHEAD REGISTER AT THE BEGINNING OF SECTOR IS IN ERROR/
1027	003144	042510	042101	051040	
1028	003152	043505	051511	042524	

1029	003160	020122	052101	052040	
1030	003166	042510	041040	043505	
1031	003174	047111	044516	043516	
1032	003202	047440	020106	042523	
1033	003210	052103	051117	044440	
1034	003216	020123	047111	042440	
1035	003224	051122	051117	000	
1036	003231	114	047517	020113	EM25: .ASCIZ /LOOK AHEAD REGISTER IS IN ERROR/
1037	003236	044101	040505	020104	
1038	003244	042522	044507	052123	
1039	003252	051105	044440	020123	
1040	003260	047111	042440	051122	
1041	003266	051117	000		
1042	003271	103	051125	042522	EM30: .ASCII /CURRENT CYLINDER DOES NOT MATCH DESIRED CYLINDER REGISTER/<15><12>
1043	003276	052116	041440	046131	
1044	003304	047111	042504	020122	
1045	003312	047504	051505	047040	
1046	003320	052117	046440	052101	
1047	003326	044103	042040	051505	
1048	003334	051111	042105	041440	
1049	003342	046131	047111	042504	
1050	003350	020122	042522	04407	
1051	003356	051510	042524	006522	
1052	003364	012			
1053	003365	101	052106	051105	.ASCIZ /AFTER A SEEK AND INIT/
1054	003372	040440	051440	042505	
1055	003400	020113	047101	020104	
1056	003406	047111	052111	000	
1057	003413	105	041503	043440	EM31: .ASCII /ECC GENERATED IS INCORRECT/<15><12>
1058	003420	047105	051105	052101	
1059	003426	042105	044440	020123	
1060	003434	047111	047503	051122	
1061	003442	041505	006524	012	
1062	003447	105	042526	054522	.ASCIZ /EVERY WORD ON THIS SECTOR IS THAT GIVEN IN "DATA USED"/
1063	003454	053440	051117	020104	
1064	003462	047117	052040	044510	
1065	003470	020123	042523	052103	
1066	003476	051117	044440	020123	
1067	003504	044124	052101	043440	
1068	003512	053111	047105	044440	
1069	003520	020116	042042	052101	
1070	003526	020101	051525	042105	
1071	003534	000042			
1072	003536	047117	051040	040505	EM32: .ASCII /ON READ COMMAND, AFTER DATA AND ECC HAVE BEEN READ,<15><12>
1073	003544	020104	047503	046515	
1074	003552	047101	026104	040440	
1075	003560	052106	051105	042040	
1076	003566	052101	020101	047101	
1077	003574	020104	041505	020103	
1078	003602	040510	042526	041040	
1079	003610	042505	020116	042522	
1080	003616	042101	006454	012	
1081	003623	105	041503	051040	.ASCII /ECC REGISTERS OR RHER1 ARE IN ERROR/<15><12>
1082	003630	043505	051511	042524	

1083	003636	051522	047440	020122	
1084	003644	044122	051105	020061	
1085	003652	051101	020105	047111	
1086	003660	042440	051122	051117	
1087	003666	005015			
1088	003670	047117	054514	046040	.ASCII /ONLY LOWER 11 BITS OF PATTERN REG. CAN BE READ/<15><12>
1089	003676	053517	051105	030440	
1090	003704	020061	044502	051524	
1091	003712	047440	020106	040520	
1092	003720	052124	051105	020116	
1093	003726	042522	027107	041440	
1094	003734	047101	041040	020105	
1095	003742	042522	042101	005015	
1096	003750	044124	051511	051440	.ASCIZ /THIS SHOULD MATCH LOWER 11 BITS OF GOOD ECC1/
1097	003756	047510	046125	020104	
1098	003764	040515	041524	020110	
1099	003772	047514	042527	020122	
1100	004000	030461	041040	052111	
1101	004006	020123	043117	043440	
1102	004014	047517	020104	041505	
1103	004022	030503	000		
1104	004025	110	043511	020110	EM33: .ASCIZ /HIGH COUNT BIT NOT SET AFTER 38859 CLOCKS/
1105	004032	047503	047125	020124	
1106	004040	044502	020124	047516	
1107	004046	020124	042523	020124	
1108	004054	043101	042524	020122	
1109	004062	034063	032470	020071	
1110	004070	046103	041517	051513	
1111	004076	000			
1112	004077	132	051105	020117	EM34: .ASCIZ /ZERO DETECT BIT NOT HIGH WHEN 32 BIT ECC REG. HAS 21 ZEROS/
1113	004104	042504	042524	052103	
1114	004112	041040	052111	047040	
1115	004120	052117	044040	043511	
1116	004126	020110	044127	047105	
1117	004134	031440	020062	044502	
1118	004142	020124	041505	020103	
1119	004150	042522	027107	044040	
1120	004156	051501	031040	020061	
1121	004164	042532	047522	050123	
1122	004172	047520	044523	044524	EM35: .ASCII /POSITION REGISTER OR 11 BITS OF PATTERN REGISTER INCORRECT/<15><12>
1123	004200	047117	051040	043505	
1124	004206	051511	042524	020122	
1125	004214	051117	030440	020061	
1126	004222	044502	051524	047440	
1127	004230	020106	040520	052124	
1128	004236	051105	020116	042522	
1129	004244	044507	052123	051105	
1130	004252	044440	041516	051117	
1131	004260	042522	052103	005015	
1132	004266	047514	042527	020122	.ASCII /LOWER 11 BITS OF PATTERN REGISTER SHOULD MATCH LOWER/<15><12>
1133	004274	030461	041040	052111	
1134	004302	020123	043117	050040	
1135	004310	052101	042524	047122	
1136	004316	051040	043505	051511	

1137	004324	042524	020122	044123
1138	004332	052517	042114	046440
1139	004340	052101	044103	046040
1140	004346	053517	051105	005015
1141	004354	030461	041040	052111
1142	004362	020123	043117	043440
1143	004370	047517	020104	041505
1144	004376	030503	005015	
1145	004402	040504	020124	047105
1146	004410	046126	050117	043440
1147	004416	047517	020104	047520
1148	004424	044523	044524	047117
1149	004432	040440	042116	047040
1150	004440	041455	042117	020105
1151	004446	042532	047522	020123
1152	004454	051101	020105	047111
1153	004462	047440	052103	046101
1154	004470	000		
1155	004471	117	020116	042522
1156	004476	042101	041440	046517
1157	004504	040515	042116	053440
1158	004512	052111	020110	047516
1159	004520	026516	047503	051122
1160	004526	041505	040524	046102
1161	004534	020105	051105	047522
1162	004542	020122	041504	020113
1163	004550	047101	020104	041505
1164	004556	020110	044123	052517
1165	004564	042114	041040	020105
1166	004572	042523	006524	012
1167	004577	111	020106	047520
1168	004604	044523	044524	047117
1169	004612	051040	043505	051511
1170	004620	042524	020122	030475
1171	004626	030060	030064	047440
1172	004634	020122	030061	032060
1173	004642	020061	052111	044440
1174	004650	020123	047507	042117
1175	004656	000		
1176	004657	127	044522	044524
1177	004664	043516	053440	052111
1178	004672	020110	052502	020123
1179	004700	042101	051104	051505
1180	004706	020123	044510	044107
1181	004714	051105	052040	040510
1182	004722	020116	034062	020113
1183	004730	040503	051525	042105
1184	004736	042440	051122	051117
1185	004744	000		
1186	004745	124	042510	042522
1187	004752	053440	051501	040440
1188	004760	051040	040505	026504
1189	004766	051127	052111	020105
1190	004774	042510	042101	051105

.ASCII /11 BITS OF GOOD ECC1/&lt;15&gt;&lt;12&gt;

.ASCIZ /DAT ENVELOP GOOD POSITION AND N-CODE ZEROS ARE IN OCTAL/

EM36: .ASCII /ON READ COMMAND WITH NON-CORRECTABLE ERROR DCK AND ECH SHOULD BE SET/&lt;1

.ASCIZ /IF POSITION REGISTER =10040 OR 10041 IT IS GOOD/

EM37: .ASCIZ /WRITING WITH BUS ADDRESS HIGHER THAN 28K CAUSED ERROR/

EM40: .ASCII /THERE WAS A READ-WRITE HEADER &amp; DATA ERROR DURING 'DTE'/&lt;15&gt;&lt;12&gt;

1191	005002	023040	042040	052101
1132	005010	020101	051105	047522
1193	005016	020122	052504	044522
1194	005024	043516	023440	052104
1195	005032	023505	005015	
1196	005036	042524	052123	051440
1197	005044	052105	050125	026440
1198	005052	052040	042510	052040
1199	005060	051505	020124	040527
1200	005066	020123	041101	051117
1201	005074	042524	020104	052101
1202	005102	052040	040510	020124
1203	005110	047520	047111	000124

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

1204 005116 040506 040524 020114 CPHALT: .ASCII /FATAL ERROR - SEE DOCUMENT FOR BEST COURSE OF ACTION/<15><12>

1205 005124 051105 047522 020122

1206 005132 020055 042523 020105

1207 005140 047504 052503 042515

1208 005146 052116 043040 051117

1209 005154 041040 051505 020124

1210 005162 047503 051125 042523

1211 005170 047440 020106 041501

1212 005176 044524 047117 005015

1213 005204 006440 103412 177777

.ASCII / <15><12><207><377><377><207><377><377><207><377><377>

1214 005212 177607 103777 177777

1215 005220 044124 020105 051120

.ASCII /THE PROGRAM HAS HALTED DURING A TEST BECAUSE CONTROLLER/<15><12>

1216 005226 043517 040522 020115

1217 005234 040510 020123 040510

1218 005242 052114 042105 042040

1219 005250 051125 047111 020107

1220 005256 020101 042524 052123

1221 005264 041040 041505 052501

1222 005272 042523 041440 047117

1223 005300 051124 046117 042514

1224 005306 006522 012

1225 005311 117 020122 042504

.ASCII /OR DEVICE HAS LOST 'READY', BECOME UNAVAILABLE,/<15><12>

1226 005316 044526 042503 044040

1227 005324 051501 046040 051517

1228 005332 020124 051047 040505

1229 005340 054504 026047 041040

1230 005346 041505 046517 020105

1231 005354 047125 053101 044501

1232 005362 040514 046102 026105

1233 005370 005015

1234 005372 047507 042516 047440

.ASCIZ /GONE OFFLINE, OR CANNOT CLEAR STATUS BITS/

1235 005400 043106 044514 042516

1236 005406 020054 051117 041440

1237 005414 047101 047516 020124

1238 005422 046103 040505 020122

1239 005430 052123 052101 051525

1240 005436 041040 052111 000123

1241

1242 005444 041520 020040 020040

DH1: .ASCII /PC TEST REG. GOOD ACTUAL /<15><12>

1243 005452 020040 042524 052123

1244 005460 020040 020040 042522

1245 005466 027107 020040 020040

1246 005474 047507 042117 020040

1247 005502 020040 041501 752524

1248 005510 046101 006411 012

1249 005515 040 020040 020040

.ASCIZ / NO ADDR. DATA DATA/

1250 005522 020040 047040 020117

1251 005530 020040 020040 040440

1252 005536 042104 027122 020040

1253 005544 042040 052101 020101

1254 005552 020040 042040 052101

1255 005560 000101

DH2: .ASCII /PC TEST WORD GOOD BAD/<15><12>

1256 005562 041520 020040 020040

1257 005570 020040 042524 052123







1366	006722	042510	030522	020040
1367	006730	051040	042510	031122
1368	006736	020040	051040	042510
1369	006744	031522	020040	051040
1370	006752	040510	020123	020040
1371	006760	051040	042110	030523
1372	006766	000		
1373	006767	120	020103	020040
1374	006774	020040	052040	051505
1375	007002	020124	020040	041440
1376	007010	047117	020124	020040
1377	007016	041440	047117	020124
1378	007024	020040	041440	047117
1379	007032	006524	012	
1380	007035	040	020040	020040
1381	007042	020040	047040	020117
1382	007050	020040	020040	051040
1383	007056	041510	030523	020040
1384	007064	051040	040510	020123
1385	007072	020040	051040	042110
1386	007100	030523	000	
1387	007103	120	020103	020040
1388	007110	020040	052040	051505
1389	007116	020124	020040	051040
1390	007124	040510	020123	020040
1391	007132	051040	041510	031123
1392	007140	005015		
1393	007142	020040	020040	020040
1394	007150	020040	047516	020040
1395	007156	020040	020040	047125
1396	007164	052111	020040	020040
1397	007172	047125	052111	000
1398	007177	120	020103	020040
1399	007204	020040	052040	051505
1400	007212	020124	020040	051040
1401	007220	042110	052123	020040
1402	007226	041040	042101	020040
1403	007234	020040	043440	047517
1404	007242	020104	020040	051440
1405	007250	041505	047524	020122
1406	007256	051440	041505	047524
1407	007264	006522	012	
1408	007267	040	020040	020040
1409	007274	020040	047040	020117
1410	007302	020040	020040	041440
1411	007310	047117	027124	020040
1412	007316	051040	046110	020101
1413	007324	020040	051040	046110
1414	007332	020101	020040	047040
1415	007340	020117	020040	020040
1416	007346	041440	047514	045503
1417	007354	000		
1418	007355	120	020103	020040
1419	007362	020040	052040	051505

DH21: .ASCII /PC TEST CONT CONT CONT/<15><12>

.ASCIZ / NO RHCS1 RHAS RHDS1/

DH22: .ASCII /PC TEST RHAS RHCS2/<15><12>

.ASCIZ / NO UNIT UNIT/

DH24: .ASCII /PC TEST RMDST BAD GOOD SECTOR SECTOR/<15><12>

.ASCIZ / NO CONT. RHLA RHLA NO CLOCK/

DH26: .ASCII /PC TEST PC OF FAILING CONT. CONT. CONT. CONT./<15><12>

1420	007370	020124	020040	050040
1421	007376	020103	043117	020040
1422	007404	043040	044501	044514
1423	007412	043516	041440	047117
1424	007420	027124	020040	041440
1425	007426	047117	027124	020040
1426	007434	041440	047117	027124
1427	007442	020040	041440	047117
1428	007450	027124	005015	
1429	007454	020040	020040	020040
1430	007462	040040	047516	020040
1431	007470	040040	020040	051512
1432	007476	040122	020040	020040
1433	007504	042522	020107	042101
1434	007512	020104	044122	051503
1435	007520	020061	020040	044122
1436	007526	051503	020062	020040
1437	007534	044122	051504	020061
1438	007542	020040	044122	051105
1439	007550	004461	000	
1440	007553	120	020103	020040
1441	007560	020040	052040	051505
1442	007566	020124	020040	050040
1443	007574	020103	043117	020040
1444	007602	043040	044501	044514
1445	007610	043516	043440	047517
1446	007616	020104	020040	040440
1447	007624	052103	040525	004514
1448	007632	005015		
1449	007634	020040	020040	020040
1450	007642	020040	047516	020040
1451	007650	020040	020040	051512
1452	007656	020122	020040	020040
1453	007664	042522	027107	020040
1454	007672	020040	040504	040524
1455	007700	020040	020040	040504
1456	007706	040524	000	
1457	007711	120	020103	020040
1458	007716	020040	052040	051505
1459	007724	020124	020040	050040
1460	007732	020103	043117	020040
1461	007740	051040	043505	020056
1462	007746	020040	043440	047517
1463	007754	020104	020040	041040
1464	007762	042101	005015	
1465	007766	020040	020040	020040
1466	007774	020040	047516	020040
1467	010002	020040	020040	051512
1468	010010	020122	020040	020040
1469	010016	042101	051104	020040
1470	010024	020040	040504	040524
1471	010032	020040	020040	040504
1472	010040	040524	000	
1473	010043	120	020103	020040

.ASCIZ / NO JSR REG ADD RHCS1 RHCS2 RHDS1 RHER1 /

DH27: .ASCII /PC TEST PC OF FAILING GOOD ACTUAL /<15><12>

.ASCIZ / NO JSR REG. DATA DATA/

DH30: .ASCII /PC TEST PC OF REG. GOOD BAD/<15><12>

.ASCIZ / NO JSR ADDR DATA DATA/

DH31: .ASCII /PC TEST WORD GOOD BAD CONT. CONT. CONT./<15><12>

1474	010050	020040	052040	051505
1475	010056	020124	020040	053440
1476	010064	051117	020104	020040
1477	010072	043440	047517	020104
1478	010100	020040	041040	042101
1479	010106	020040	020040	041440
1480	010114	047117	027124	020040
1481	010122	041440	047117	027124
1482	010130	020040	041440	047117
1483	010136	027124	005015	
1484	010142	020040	020040	020040
1485	010150	020040	047516	020040
1486	010156	020040	020040	047516
1487	010164	020040	020040	020040
1488	010172	040504	040524	020040
1489	010200	020040	040504	040524
1490	010206	020040	020040	044122
1491	010214	051503	020061	020040
1492	010222	044122	051504	020061
1493	010230	020040	044122	051105
1494	010236	004461	000	
1495	010241	120	020103	020040
1496	010246	020040	052040	051505
1497	010254	020124	020040	050040
1498	010262	020103	043117	020040
1499	010270	053440	051117	020104
1500	010276	020040	043440	047517
1501	010304	020104	020040	041040
1502	010312	042101	020040	020040
1503	010320	041440	047117	027124
1504	010326	020040	041440	047117
1505	010334	027124	020011	041440
1506	010342	047117	027124	005015
1507	010350	020040	020040	020040
1508	010356	020040	047516	04040
1509	010364	020040	020040	051512
1510	010372	020122	020040	020040
1511	010400	047516	020040	020040
1512	010406	020040	040504	040524
1513	010414	020040	020040	040504
1514	010422	040524	020040	020040
1515	010430	044122	051503	020061
1516	010436	020040	044122	051504
1517	010444	020061	020040	044122
1518	010452	051105	000061	
1519	010456	041520	020040	020040
1520	010464	020040	042524	052123
1521	010472	020040	020040	041520
1522	010500	047440	020106	020040
1523	010506	047527	042122	020040
1524	010514	020040	047507	042117
1525	010522	020040	020040	047503
1526	010530	052116	020056	020040
1527	010536	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	RHDS1	RHER1/
DH33:	.ASCII /PC	TEST	PC OF	WORD	GOOD	CONT.	CONT.	CONT.	<15><12



1582	011225	120	020103	020040	DH36:	.ASCII	/PC	TEST	PC OF	RHMR	POSITON PATTERN/<15><12>			
1583	011232	020040	052040	051505										
1584	011240	020124	020040	050040										
1585	011246	020103	043117	020040										
1586	011254	051040	046510	020122										
1587	011262	020040	050040	051517										
1588	011270	052111	047117	050040										
1589	011276	052101	042524	047122										
1590	011304	005015												
1591	011306	020040	020040	020040		.ASCIZ	/	NO	JSR	CONT.	REG.	REG./		
1592	011314	020040	047516	020040										
1593	011322	020040	020040	051512										
1594	011330	020122	020040	020040										
1595	011336	047503	052116	020056										
1596	011344	020040	042522	027107										
1597	011352	020040	020040	042522										
1598	011360	027107	000											
1599	011363	120	020103	020040	DH37:	.ASCII	/PC	TEST	POSITON	POSITON	GOOD	GOOD	PATTERN DATA	N-CODE/
1600	011370	020040	052040	051505										
1601	011376	020124	020040	050040										
1602	011404	051517	052111	047117										
1603	011412	050040	051517	052111										
1604	011420	047117	043440	047517										
1605	011426	020104	020040	043440										
1606	011434	047517	020104	020040										
1607	011442	050040	052101	042524										
1608	011450	047122	042040	052101										
1609	011456	020101	020040	047040										
1610	011464	041455	042117	006505										
1611	011472	012												
1612	011473	040	020040	020040		.ASCIZ	/	NO	ECC	GOOD	ECC1	ECC2	ECC	ENVELOPE ZEROS
1613	011500	020040	047040	020117										
1614	011506	020040	020040	042440										
1615	011514	041503	020040	020040										
1616	011522	043440	047517	020104										
1617	011530	020040	042440	041503										
1618	011536	020061	020040	042440										
1619	011544	041503	020062	020040										
1620	011552	042440	041503	020040										
1621	011560	020040	042440	053116										
1622	011566	047514	042520	055040										
1623	011574	051105	051517	000011										
1624														
1625	011602	041520	020040	020040	DH40:	.ASCII	/PC	TEST	FAILING	CONT.	CONT.	CONT.	CONT./<15><12>	
1626	011610	020040	042524	052123										
1627	011616	020040	020040	040506										
1628	011624	046111	047111	020107										
1629	011632	047503	052116	020056										
1630	011640	020040	047503	052116										
1631	011646	020056	020040	047503										
1632	011654	052116	020056	020040										
1633	011662	047503	052116	006456										
1634	011670	012												
1635	011671	040	020040	020040		.ASCIZ	/	NO	REG ADR	RHCS1	RHCS2	RHDS1	RHER1/	





1690	012276	001116	017322	051722	DT31:	.WORD	\$ERRPC, TSTNM, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1, 0
1691	012304	001124	001126	015030			
1692	012312	015052	015032	000000			
1693	012320	001116	017322	015130	DT32:	.WORD	\$ERRPC, TSTNM, PCJSR, ERWORD, \$GDDAT, \$BDDAT, CS1, DS1, ER1, 0
1694	012326	051722	001124	001126			
1695	012334	015030	015052	015032			
1696	012342	000000					
1697	012344	001116	017322	015130	DT33:	.WORD	\$ERRPC, TSTNM, PCJSR, ERWORD, \$GDDAT, CS1, DS1, ER1, 0
1698	012352	051722	001124	015030			
1699	012360	015052	015032	000000			
1700	012366	001116	017322	047646	DT34:	.WORD	\$ERRPC, TSTNM, GECC1, GECC2, MECC1, MECC2, DISK, 0
1701	012374	047650	054520	054522			
1702	012402	053520	000000				
1703	012406	001116	017322	047646	DT35:	.WORD	\$ERRPC, TSTNM, GECC1, GECC2, EC2, EC1, POSITI, ER1, 0
1704	012414	047650	015062	015060			
1705	012422	047660	015032	000000			
1706	012430	001116	017322	015130	DT36:	.WORD	\$ERRPC, TSTNM, PCJSR, MR, EC1, EC2, 0
1707	012436	015050	015060	015062			
1708	012444	000000					
1709	012446	001116	017322	015060	DT37:	.WORD	\$ERRPC, TSTNM, EC1, POSITI, GECC1, GECC2, EC2, DATENV, ZCODE, 0
1710	012454	047660	047646	047650			
1711	012462	015062	047664	047666			
1712	012470	000000					
1713	012472	001116	017322	001122	DT40:	.WORD	\$ERRPC, TSTNM, \$BDADR, CS1, CS2, DS1, ER1, 0
1714	012500	015030	015026	015052			
1715	012506	015032	000000				
1716							
1717							
1718							
1719							
1720	012512	000	000	000	DF1:	.BYTE	0,0,0,0,0
1721	012515	000	000				
1722	012517	000	000	001	DF2:	.BYTE	0,0,1,0
1723	012522	000					
1724	012523	000	000	001	DF3:	.BYTE	0,0,1,0,0
1725	012526	000	000				
1726							
1727	012530	000	000	000	DF11:	.BYTE	0,0,0,0,0,0,0
1728	012533	000	000	000			
1729	012536	000					
1730	012537	000	000	000	DF14:	.BYTE	0,0,0,0,0,0,0,0
1731	012542	000	000	000			
1732	012545	000	000				
1733	012547	000	000	000	DF15:	.BYTE	0,0,0
1734	012552	000	000	000	DF16:	.BYTE	0,0,0,0,0
1735	012555	000	000				
1736	012557	000	000	000	DF17:	.BYTE	0,0,0,0,0,0,0
1737	012562	000	000	000			
1738	012565	000					
1739	012566	000	000	000	DF20:	.BYTE	0,0,0,0,0,0,0
1740	012571	000	000	000			
1741	012574	000					
1742							
1743	012575	000	000	000	DF21:	.BYTE	0,0,0,0,0

1744	012600	000	000				
1745	012602	000	000	000	DF22:	.BYTE	0,0,0,0
1746	012605	000					
1747	012606	000	000	000	DF24:	.BYTE	0,0,0,0,0,0,0
1748	012611	000	000	000			
1749	012614	000					
1750	012615	000	000	000	DF26:	.BYTE	0,0,0,0,0,0,0,0
1751	012620	000	000	000			
1752	012623	000	000				
1753	012625	000	000	000	DF27:	.BYTE	0,0,0,0,0,0
1754	012630	000	000	000			
1755	012633	000	000	000	DF30:	.BYTE	0,0,0,0,0,0
1756	012636	000	000	000			
1757							
1758	012641	000	000	001	DF31:	.BYTE	0,0,1,0,0,0,0,0
1759	012644	000	000	000			
1760	012647	000	000				
1761	012651	000	000	000	DF32:	.BYTE	0,0,0,1,0,0,0,0,0
1762	012654	001	000	000			
1763	012657	000	000	000			
1764	012662	000	000	000	DF33:	.BYTE	0,0,0,1,0,0,0,0
1765	012665	001	000	000			
1766	012670	000	000				
1767	012672	000	000	000	DF34:	.BYTE	0,0,0,0,0,0,0
1768	012675	000	000	000			
1769	012700	000					
1770	012701	000	000	000	DF35:	.BYTE	0,0,0,0,0,0,0,0
1771	012704	000	000	000			
1772	012707	000	000				
1773	012711	000	000	000	DF36:	.BYTE	0,0,0,0,0,0
1774	012714	000	000	000			
1775	012717	000	000	000	DF37:	.BYTE	0,0,0,0,0,0,0,0,0
1776	012722	000	000	000			
1777	012725	000	000	000			
1778	012730	000	000	000	DF40:	.BYTE	0,0,0,0,0,0,0
1779	012733	000	000	000			
1780	012736	000					
1781							
1782		012740				.EVEN	
1783	012740	044122	030461	051040	EM42:	.ASCIZ	/RH11 REGISTER FAILED TO RESPOND TO A "TST" COMMAND/
1784	012746	043505	051511	042524			
1785	012754	020122	040506	046111			
1786	012762	042105	052040	020117			
1787	012770	042522	050123	047117			
1788	012776	020104	047524	040440			
1789	013004	021040	051524	021124			
1790	013012	041440	046517	040515			
1791	013020	042116	000				
1792							
1793	013023	122	030510	020061	EM43:	.ASCIZ	/RH11 ILLEGAL REGISTER RESPONSE TEST/
1794	013030	046111	042514	040507			
1795	013036	020114	042522	044507			
1796	013044	052123	051105	051040			
1797	013052	051505	047520	051516			

1798	013060	020105	042524	052123			
1799	013066	000					
1800							
1801	013067	115	053117	020105	EM44:	.ASCIZ	/MOVE BYTE TO WORD COUNT TEST/
1802	013074	054502	042524	052040			
1803	013102	020117	047527	042122			
1804	013110	041440	052517	052116			
1805	013116	052040	051505	000124			
1806							
1807	013124	044502	020123	054502	EM45:	.ASCIZ	/BIS BYTE TO WORD COUNT /
1808	013132	042524	052040	020117			
1809	013140	047527	042122	041440			
1810	013146	052517	052116	000040			
1811							
1812	013154	044502	020103	054502	EM46:	.ASCIZ	/BIC BYTE TO WORD COUNT/
1813	013162	042524	052040	020117			
1814	013170	047527	042122	041440			
1815	013176	052517	052116	000			
1816							
1817	013203	101	042104	042522	EM47:	.ASCIZ	/ADDRESS SELECT XOR'S TEST/
1818	013210	051523	051440	046105			
1819	013216	041505	020124	047530			
1820	013224	023522	020123	042524			
1821	013232	052123	000				
1822							
1823	013235	116	047117	042440	EM50:	.ASCIZ	/NON EXISTANT DRIVE (NED) TEST/
1824	013242	044530	052123	047101			
1825	013250	020124	051104	053111			
1826	013256	020105	047050	042105			
1827	013264	020051	042524	052123			
1828	013272	000					
1829							
1830	013273	101	020123	042522	EM51:	.ASCIZ	/AS REGISTER TEST/
1831	013300	044507	052123	051105			
1832	013306	052040	051505	000124			
1833							
1834	013314	052502	051523	040440	EM52:	.ASCIZ	/BUSS ADDRESS REGISTER DATA TEST/
1835	013322	042104	042522	051523			
1836	013330	051040	043505	051511			
1837	013336	042524	020122	040504			
1838	013344	040524	052040	051505			
1839	013352	000124					
1840							
1841	013354	052502	051523	040440	EM53:	.ASCIZ	/BUSS ADDRESS REGISTER MOVE BYTE /
1842	013362	042104	042522	051523			
1843	013370	051040	043505	051511			
1844	013376	042524	020122	047515			
1845	013404	042526	020040	054502			
1846	013412	042524	000040				
1847							
1848	013416	044122	030461	044440	EM54:	.ASCIZ	/RH11 INTERRUPT FAILED TO OCCUR/
1849	013424	052116	051105	052522			
1850	013432	052120	043040	044501			
1851	013440	042514	020104	047524			

1852	013446	047440	041503	051125		
1853	013454	000				
1854						
1855	013455	122	051505	052105	EM55:	.ASCIZ /RESET TEST/
1856	013462	052040	051505	000124		
1857						
1858	013470	054115	020106	041040	EM56:	.ASCIZ /MXF BIT TEST/
1859	013476	052111	052040	051505		
1860	013504	000124				
1861						
1862	013506	047125	041111	051525	EM57:	.ASCIZ /UNIBUS PARITY BIT TEST/
1863	013514	050040	051101	052111		
1864	013522	020131	044502	020124		
1865	013530	042524	052123	000		
1866						
1867	013535	104	042517	020123	EM60:	.ASCIZ /DOES SELECTING THE RH11 CLEAR THE UNIT SELECT REGISTER/
1868	013542	042523	042514	052103		
1869	013550	047111	020107	044124		
1870	013556	020105	044122	030461		
1871	013564	041440	042514	051101		
1872	013572	052040	042510	052440		
1873	013600	044516	020124	042523		
1874	013606	042514	052103	051040		
1875	013614	043505	051511	042524		
1876	013622	000122				
1877						
1878	013624	047504	051505	052040	EM61:	.ASCIZ /DOES TRE GET SET BY UNIBUS PARITY ERROR/
1879	013632	042522	043440	052105		
1880	013640	051440	052105	041040		
1881	013646	020131	047125	041111		
1882	013654	051525	050040	051101		
1883	013662	052111	020131	051105		
1884	013670	047522	000122			
1885						
1886	013674	047516	020116	054105	EM62:	.ASCIZ /NON EXISTANT DRIVE (NED) FAILED TO SET (TRE)/
1887	013702	051511	040524	052116		
1888	013710	042040	044522	042526		
1889	013716	024040	042516	024504		
1890	013724	043040	044501	042514		
1891	013732	020104	047524	051440		
1892	013740	052105	024040	051124		
1893	013746	024505	000			
1894						
1895	013751	116	047117	042440	EM63:	.ASCIZ /NON EXISTANT MEMORY (NEM) FAILED TO SET (TRE)/
1896	013756	044530	052123	047101		
1897	013764	020124	042515	047515		
1898	013772	054522	024040	042516		
1899	014000	024515	043040	044501		
1900	014006	042514	020104	047524		
1901	014014	051440	052105	024040		
1902	014022	051124	024505	000		
1903						
1904	014027	120	051117	020124	EM64:	.ASCIZ /PORT SELECT FAILED TO CLEAR/
1905	014034	042523	042514	052103		

1906	014042	043040	044501	042514				
1907	014050	020104	047524	041440				
1908	014056	042514	051101	000				
1909								
1910	014063	103	047117	051124	EM65:	.ASCIZ	/CONTROLLER CLEAR FAILED TO CLEAR COMMAND REGISTER/	
1911	014070	046117	042514	020122				
1912	014076	046103	040505	020122				
1913	014104	040506	046111	042105				
1914	014112	052040	020117	046103				
1915	014120	040505	020122	047503				
1916	014126	046515	047101	020104				
1917	014134	042522	044507	052123				
1918	014142	051105	000					
1919								
1920	014145	122	051505	046105	EM66:	.ASCIZ	/RESELECT FAILED TO CLEAR COMMAND REGISTER/	
1921	014152	041505	020124	040506				
1922	014160	046111	042105	052040				
1923	014166	020117	046103	040505				
1924	014174	020122	047503	046515				
1925	014202	047101	020104	042522				
1926	014210	044507	052123	051105				
1927	014216	000						
1928								
1929	014217	111	052116	051105	EM67:	.ASCIZ	/INTERRUPT CAUSED BY RDY!IE BITS SET FAILED TO OCCUR/	
1930	014224	052522	052120	041440				
1931	014232	052501	042523	020104				
1932	014240	054502	020040	042122				
1933	014246	020531	042511	041040				
1934	014254	052111	020123	042523				
1935	014262	020124	040506	046111				
1936	014270	042105	052040	020117				
1937	014276	041517	052503	000122				
1938								
1939	014304	042511	043040	044501	EM70:	.ASCIZ	/IE FAILED TO CLEAR FOLLOWING AN INTERRUPT/	
1940	014312	042514	020104	047524				
1941	014320	041440	042514	051101				
1942	014326	043040	046117	047514				
1943	014334	044527	043516	040440				
1944	014342	020116	047111	042524				
1945	014350	051122	050125	000124				
1946								
1947	014356	051501	051040	043505	EM71:	.ASCIZ	/AS REGISTER WRITE TEST/	
1948	014364	051511	042524	020122				
1949	014372	051127	052111	020105				
1950	014400	042524	052123	000				
1951								
1952								
1953								
1954	014405	120	020103	020040	DM42:	.ASCII	/PC TEST ILLEGAL/<15><12>	
1955	014412	020040	052040	051505				
1956	014420	004524	046111	042514				
1957	014426	040507	006514	012				
1958	014433	040	020040	020040		.ASCIZ	/ NO ADDRESS/	
1959	014440	020040	047040	020117				

1960	014446	020040	020040	042101					
1961	014454	051104	051505	000123					
1962	014462	041520	020040	020040	DM44:	.ASCII	/PC	TEST	REGISTER CONT CONT/<15><12>
1963	014470	020040	042524	052123					
1964	014476	020040	020040	042522					
1965	014504	044507	052123	051105					
1966	014512	041440	047117	020124					
1967	014520	020040	041440	047117					
1968	014526	006524	012						
1969	014531	040	020040	020040		.ASCIZ	/	NO	ADDRESS GOOD BAD/
1970	014536	020040	047040	020117					
1971	014544	020040	020040	040440					
1972	014552	042104	042522	051523					
1973	014560	020040	047507	042117					
1974	014566	020040	020040	040502					
1975	014574	000104							
1976	014576	041520	020040	020040	DM54:	.ASCII	/PC	TEST	REGISTER CONTS/<15><12>
1977	014604	020040	042524	052123					
1978	014612	020040	051040	043505					
1979	014620	051511	042524	020122					
1980	014626	020040	020040	020040					
1981	014634	041440	047117	051524					
1982	014642	005015							
1983	014644	020040	020040	020040		.ASCIZ	/	NO	ADDRESS/
1984	014652	020040	047516	020040					
1985	014660	020040	040440	042104					
1986	014666	042522	051523	000					
1987									
1988		014674				.EVEN			
1989									
1990	014674	001116	017322	044716	DT42:	.WORD		SERRPC, TSTNM, REGADR, 0	
1991	014702	000000							
1992	014704	001116	017322	044716	DT44:	.WORD		SERRPC, TSTNM, REGADR, SGDDAT, SBDDAT, 0	
1993	014712	001124	001126	000000					
1994	014720	001116	017322	044716	DT54:	.WORD		SERRPC, TSTNM, REGADR, SGDDAT, 0	
1995	014726	001124	000000						

H07

1996	014732	000	000	000	DF42:	.BYTE	0,0,0
1997	014735	000	000	000	DF44:	.BYTE	0,0,0,0,0
1998	014740	000	000				
1999	014742	000	000	000	DFS4:	.BYTE	0,0,0,0
2000	014745	000					

2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039

.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)

000001	US1=	1	;UNIT SELECT (BIT #0)
000002	US2=	2	;UNIT SELECT (BIT #1)
000004	US4=	4	;UNIT SELECT (BIT #2)
000010	BAI=	10	;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
000020	PAT=	20	;INVERT PARITY ON MASS BUS TO EVEN (BIT #4)
000040	CLR=	40	;CLEAR (BIT #5)
000100	IR=	100	;INPUT READY (BIT #6)
000200	OR=	200	;OUTPUT READY (BIT #7)
000400	MPE=	400	;MASS BUS PARITY ERROR (BIT #8)
001000	MXF=	1000	;MISSED TRANSFER ERROR (BIT #9)
002000	PGE=	2000	;PROGRAM ERROR (BIT #10)
004000	NEM=	4000	;NON EXISTANT MEMORY (BIT #11)
010000	NED=	10000	;NON EXISTANT DRIVE (BIT #12)
020000	LPE=	20000	;UNIBUS PARITY ERROR (BIT #13)
040000	MCE=	40000	;WRITE CHECK ERROR (BIT #14)
100000	DLT=	100000	;DATA LATE (BIT #15)

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



```

2040 ;*****
2041 ;RPO4 REGISTERS
2042 ;*****
2043
2044
2045
2046 ;CONTROL AND STATUS 1 REGISTER. (#00)
2047
2048 000001 GO= 1 ;GO (BIT #0)
2049 000100 IE= 100 ;INTERRUPT ENABLE (BIT #6)
2050 000200 RDY= 200 ;READY (BIT #7)
2051 000400 A16= 400 ;HIGH ORDER UNIBUS BITS (BIT #8)
2052 001000 A17= 1000 ;HIGH ORDER UNIBUS BITS (BIT #9)
2053 002000 PSEL= 2000 ;PORT SELECT (BIT #10)
2054 004000 DVA= 4000 ;DEVICE AVAILABLE (BIT #11)
2055 020000 MCPE= 20000 ;MASSBUSS PARITY ERROR (BIT #13)
2056 040000 TRE= 40000 ;TRANSFER ERROR (BIT #14)
2057 100000 SC= 100000 ;SPECIAL CONDITION (BIT #15)
2058
2059 ;STATUS REGISTER (RHDS1) (#01)
2060
2061 000001 DFF5= 1 ;DRIVE FORWARD 5"/SEC. (BIT #0)
2062 000002 DFF20= 2 ;DRIVE FORWARD 20"/SEC. (BIT #1)
2063 000004 DIG8= 4 ;DRIVE TO INNER GAVRD BAND (BIT #2)
2064 000010 GRV= 10 ;GO REVERSE (BIT #3)
2065 000020 DL64= 20 ;DIFFERENCE LESS THAN 64 (BIT #4)
2066 000040 DE1= 40 ;DIFFERENCE EQUALS 1 (BIT #5)
2067 000100 WV= 100 ;VOLUME VALID (BIT #6)
2068 000200 DRY= 200 ;DRIVE READY (BIT #7)
2069 000400 DPR= 400 ;DRIVE PRESENT (BIT #8)
2070 001000 PROG= 1000 ;PROGRAMABLE (BIT #9)
2071 002000 LST= 2000 ;LAST SECTOR TRANSFERRED (BIT #10)
2072 004000 WRL= 4000 ;WRITE LOCK (BIT #11)
2073 010000 MOL= 10000 ;MEDIUM ON-LINE (BIT #12)
2074 020000 PIP= 20000 ;POSITIONING OPERATION IN PROGRESS (BIT #13)
2075 040000 ERR= 40000 ;COMPOSIT ERROR. (BIT #14)
2076 100000 ATA= 100000 ;ATTENTION ACTIVE (BIT #15)
2077
2078 ;ERROR REGISTER #01 (RHER1) (#02)
2079 000001 ILF= 1 ;ILLEGAL FUNCTION (BIT #0)
2080 000002 ILR= 2 ;ILLEGAL REGISTER (BIT #1)
2081 000004 RMR= 4 ;REGISTER MODIFICATION REFUSED (BIT #2)
2082 000010 PAR= 10 ;PARITY ERROR (BIT #3)
2083 000020 FER= 20 ;FORMAT ERROR (BIT #4)
2084 000040 WCF= 40 ;WRITE CLOCK FAIL (BIT #5)
2085 000100 ECH= 100 ;ECC HARD ERROR (BIT #6)
2086 000200 HCE= 200 ;HEADER COMPARE ERROR (BIT #7)
2087 000400 HCRC= 400 ;HEADER CRC ERROR (BIT #8)
2088 001000 AOE= 1000 ;ADDRESS OVERFLOW ERROR (BIT #9)
2089 002000 IAE= 2000 ;INVALID ADDRESS ERROR (BIT #10)
2090 004000 WLE= 4000 ;WRITE LOCK ERROR (BIT #11)
2091 010000 DTE= 10000 ;DRIVE TIMING ERROR (BIT #12)
2092 020000 OPT= 20000 ;OPERATION INCOMPLETE (BIT #13)
2093 040000 UNS= 40000 ;DRIVE UNSAFE (BIT #14)

```

```

2094      100000      DCK=      100000      ;DATA CHECK ERROR (BIT 15)
2095
2096      ;MAINTAINABILITY REGISTER (RHMR)(#03)
2097
2098      000001      DMD=      1      ;DIAGINOSTIC MODE (BIT #0)
2099      000002      MCLK=     2      ;MAINTAINABILITY CLOCK (BIT #1)
2100      000004      MINX=     4      ;MAINTAINABILITY INDEX (BIT #2)
2101      000010      MSTCK=    10     ;MAINTAINABILITY SECTOR CLOCK (BIT #3)
2102      000020      MRD=     20     ;MAINTAINABILITY READ (BIT #4)
2103      000040      MWR=     40     ;MAINTAINABILITY WRITE (BIT #5)
2104      000200      DENVL=   200    ;DATA ENVELOPE (BIT #7)
2105      000400      ZER=     400    ;ZERO DETECT (BIT #8)
2106      001000      DTSY=    1000   ;MAINTAINABILITY SYNC DETECTED (BIT #9)
2107
2108      ;ATTENTION SUMMARY PSEUDO-REGISTER (RHAS) (#04)
2109
2110      000001      AT0=      1      ;DEVICE 0 (BIT #0)
2111      000002      AT1=      2      ;DEVICE 1 (BIT #1)
2112      000004      AT2=      4      ;DEVICE 2 (BIT #2)
2113      000010      AT3=     10     ;DEVICE 3 (BIT #3)
2114      000020      AT4=     20     ;DEVICE 4 (BIT #4)
2115      000040      AT5=     40     ;DEVICE 5 (BIT #5)
2116      000100      AT6=    100    ;DEVICE 6 (BIT #6)
2117      000200      AT7=    200    ;DEVICE 7 (BIT #7)
2118
2119
2120
2121
2122
2123      ;DESIRED SECTOR/TRACK ADDRESS REGISTER (RHDST) (#1)
2124      ;EACH BIT IS CALLED BY BIT NUMBER
2125
2126
2127
2128
2129
2130
2131      ;DRIVE TYPE REGISTER (RHDT) (#06)
2132      ;EACH BIT IS CALLED BY BIT NUMBER
2133
2134
2135
2136
2137
2138
2139      ;LOOK-AHEAD REGISTER (RHLA) (#07)
2140      000001      EXT1=     1      ;EXTENSION 1 (BIT #0)
2141      000002      EXT2=     2      ;EXTENSION 2 (BIT #1)
2142      000004      EXT4=     4      ;EXTENSION 3 (BIT #2)
2143      000010      EXT10=    10     ;EXTENSION 4 (BIT #3)
2144      000020      EXT20=    20     ;EXTENSION 5 (BIT #4)
2145      000040      EXT40=    40     ;EXTENSION 6 (BIT #5)
2146      000100      SC1=    100    ;SECTOR COUNT FIELD 0 (BIT #6)
2147      000200      SC2=    200    ;SECTOR COUNT FIELD 1 (BIT #7)
          000400      SC4=    400    ;SECTOR COUNT FIELD 2 (BIT #8)

```

2148	001000	SC10=	1000	;SECTOR COUNT FIELD 3 (BIT #9)
2149	002000	SC20=	2000	;SECTOR COUNT FIELD 4 (BIT #10)
2150	004000	TRK1=	4000	;TRACK FIELD 1 (BIT #11)
2151	010000	TRK2=	10000	;TRACK FIELD 2 (BIT #12)
2152	020000	TRK4=	20000	;TRACK FIELD 3 (BIT #13)
2153	040000	TRK10=	40000	;TRACK FIELD 4 (BIT #14)
2154	100000	TRK20=	100000	;TRACK FIELD 5 (BIT #15)
2155				
2156		;ERROR REGISTER #2 (RHER2) (#10)		
2157				
2158	000001	WCU=	1	;WRITE CURRENT UNSAFE (BIT #0)
2159	000002	CSF=	2	;CURRENT SINK FAILURE (BIT #1)
2160	000004	WSU=	4	;WRITE SELECT UNSAFE (BIT #2)
2161	000010	CSU=	10	;CURRENT SWITCH UNSAFE (BIT #3)
2162	000020	MSE=	20	;MOTOR SEQUENCE ERROR (BIT #4)
2163	000040	TDF=	40	;TRANSITIONS DETECTOR FAILURE (BIT #5)
2164	000100	TUF=	100	;TRANSITIONS UNSAFE (BIT #6)
2165	000200	FEN=	200	;FAILSAFE ENABLED (BIT #7)
2166	000400	WRU=	400	;WRITE READY UNSAFE (BIT #8)
2167	001000	MHS=	1000	;MULTIPLE HEAD SELECT (BIT #9)
2168	002000	NHS=	2000	;NO HEAD SELECTION (BIT #10)
2169	004000	IXE=	4000	;INDEX ERROR (BIT #11)
2170	010000	VU30=	10000	;30VOLT UNSAFE (BIT #12)
2171	020000	PLU=	20000	;PLO UNSAFE (BIT #13)
2172	100000	ACU=	100000	;ACUNSAFE (BIT #15)
2173				
2174		;OFFSET REGISTER (RHOF) (#11)		
2175				
2176	000001	OF25=	1	;OFFSET 25 MICRO INCHES (BIT #0)
2177	000002	OF50=	2	;OFFSET 50 MICRO INCHES (BIT #1)
2178	000004	OF100=	4	;OFFSET 100 MICRO INCHES (BIT #2)
2179	000010	OF200=	10	;OFFSET 200 MICRO INCHES (BIT #3)
2180	000020	OF400=	20	;OFFSET 400 MICRO INCHES (BIT #4)
2181	000040	OF800=	40	;OFFSET 800 MICRO INCHES (BIT #5)
2182				
2183	000200	OFREV=	200	;OFFSET NEGATIVE (REVERSE) (BIT #7)
2184	002000	HCI=	2000	;HEADER COMPARE INHIBIT (BIT #10)
2185	004000	ECI=	4000	;ERROR CORRECTION CODE INHIBIT (BIT #11)
2186	010000	FMT22=	10000	;FORMAT BIT (BIT #12)
2187				
2188				
2189				
2190				
2191				
2192		;DESIRED CYLINDER ADDRESS (RHCA) (#12)		
2193		;EACH BIT IS CALLED BY BIT NUMBER.		
2194				
2195				
2196				
2197				
2198				
2199		;CURRENT CYLINDER ADDRESS (RHCC) (#13)		
2200		;EACH BIT IS CALLED BY BIT NUMBER		
2201				

2202  
2203  
2204  
2205  
2206  
2207  
2208  
2209  
2210  
2211  
2212  
2213  
2214  
2215  
2216  
2217  
2218  
2219  
2220  
2221  
2222  
2223  
2224  
2225  
2226  
2227  
2228  
2229  
2230  
2231  
2232  
2233

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

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

000001	PSU=	1	;PACK SPEED UNSAFE (BIT #0)
000002	VUF=	2	;VELOCITY UNSAFE (BIT #1)
000010	UMR=	10	;ANY UNSAFE EXCEPT READ/WRITE (BIT #3)
000020	PRE=	20	;DISK PACK ROTATION ERROR (BIT #4)
000040	ACL=	40	;AC LOW (BIT #5)
000100	DCL=	100	;DC LOW (BIT #6)
040000	SKI=	40000	;SEEK INCOMPLETE (BIT #14)
100000	OCYL=	100000	;OFF CYLINDER (BIT #15)

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

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

2234  
 2235  
 2236  
 2237  
 2238  
 2239  
 2240  
 2241  
 2242  
 2243  
 2244  
 2245  
 2246  
 2247  
 2248  
 2249  
 2250  
 2251  
 2252  
 2253  
 2254  
 2255  
 2256  
 2257  
 2258  
 2259  
 2260  
 2261  
 2262  
 2263  
 2264  
 2265  
 2266  
 2267  
 2268  
 2269  
 2270  
 2271  
 2272  
 2273  
 2274  
 2275  
 2276  
 2277  
 2278  
 2279  
 2280

.SBTTL REGISTER ADDRESSES

;RPO4 VECTOR ADDRESS

014746 000254

RPVEC: 254

;RPO4 VECTOR ADDRESS

;RPO4 DISK I/O REGISTERS LOCATED IN THE RH11 CONTROLLER  
 ;NOTE: THE CONTENTS OF THESE LOCATIONS WILL BE DIFFRENT  
 ; IF THE "CHANGE BASE ADDRESS" ROUTINE IS USED.  
 ; THIS ROUTINE STARTS AT LOCATION TAGED "BASECH"

014750 176722  
 014752 176702  
 014754 176704  
 014756 176710

RHDB: 176722  
 RHMC: 176702  
 RHBA: 176704  
 RHCS2: 176710

;DATA BUFFER SEE NOTE ABOVE  
 ;WORD COUNT SEE NOTE ABOVE  
 ;BUS ADDRESS SEE NOTE ABOVE  
 ;CONTROL AND STATUS 2 SEE NOTE ABOVE

;RPO4 DISK I/O REGISTERS LOCATED IN THE RPO4 DEVICE LOGIC  
 ;NOTE: THE CONTENTS OF THESE LOCATIONS WILL BE DIFFRENT  
 ; IF THE "CHANGE BASE ADDRESS ROUTINE IS USED.  
 ; THIS ROUTINE STARTS AT LOCATION TAGED "BASECH"

014760 176700  
 014762 176714  
 014764 176706  
 014766 176740  
 014770 176732  
 014772 176734  
 014774 176742  
 014776 176716  
 015000 176724  
 015002 176712  
 015004 176726  
 015006 176730  
 015010 176744  
 015012 176746  
 015014 176720  
 015016 176736

RHCS1: 176700  
 RHER1: 176714  
 RHDS1: 176706  
 RHER2: 176740  
 RHOF: 176732  
 RHCA: 176734  
 RHER3: 176742  
 RHAS: 176716  
 RHMR: 176724  
 RHDS1: 176712  
 RHDT: 176726  
 RHSN: 176730  
 RHEC1: 176744  
 RHEC2: 176746  
 RHLA: 176720  
 RHCC: 176736

;CONTROL AND STATUS 1 SEE NOTE ABOVE  
 ;ERROR #1 SEE NOTE ABOVE  
 ;DESIRED SECTOR/TRACK ADDRESS SEE NOTE ABOVE  
 ;ERROR #2 SEE NOTE ABOVE  
 ;OFFSET SEE NOTE ABOVE  
 ;DESIRED CYLINDER ADDRESS SEE NOTE ABOVE  
 ;ERROR #3 SEE NOTE ABOVE  
 ;ATTENTION SUMMARY SEE NOTE ABOVE  
 ;MAINTAINABILITY SEE NOTE ABOVE  
 ;DRIVE STATUS SEE NOTE ABOVE  
 ;DRIVE TYPE SEE NOTE ABOVE  
 ;SERIAL NUMBER SEE NOTE ABOVE  
 ;ECC POSITION SEE NOTE ABOVE  
 ;ECC PATTERN SEE NOTE ABOVE  
 ;LOOK-AHEAD SEE NOTE ABOVE  
 ;CURRENT CYLINDER ADDRESS SEE NOTE ABOVE

2281  
 2282  
 2283  
 2284  
 2285  
 2286  
 2287 015020 000000  
 2288 015022 000000  
 2289 015024 000000  
 2290 015026 000000  
 2291  
 2292  
 2293 015030 000000  
 2294 015032 000000  
 2295 015034 000000  
 2296 015036 000000  
 2297 015040 000000  
 2298 015042 000000  
 2299 015044 000000  
 2300 015046 000000  
 2301 015050 000000  
 2302 015052 000000  
 2303 015054 000000  
 2304 015056 000000  
 2305 015060 000000  
 2306 015062 000000  
 2307 015064 000000  
 2308 015066 000000  
 2309  
 2310

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

DB:	0	; DATA BUFFER
WC:	0	; WORD COUNT
BA:	0	; BUS ADDRESS
CS2:	0	; CONTROL AND STATUS 2
CS1:	0	; CONTROL AND STATUS 1
ER1:	0	; ERROR #1
DST:	0	; DESIRED SECTOR/TRACK ADDRESS
ER2:	0	; ERROR #2
OF:	0	; OFFSET
CA:	0	; DESIRED CYLINDER ADDRESS
ER3:	0	; ERROR #3
AS:	0	; ATTENTION SUMMARY
MR:	0	; MAINTAINABILITY
DS1:	0	; DRIVE STATUS
DT:	0	; DRIVE TYPE
SN:	0	; SERIAL NUMBER
EC1:	0	; ECC POSITION
EC2:	0	; ECC PATTERN
LA:	0	; LOOK-AHEAD
CC:	0	; CURRENT CYLINDER ADDRESS

```

2311                ;FLAGS AND INTERNAL PROGRAM CONTROL WORDS
2312
2313
2314
2315 015070 000010   UNITS:  .BLKW  8.   ; THIS IS FILLED WITH -1
2316 015110 000000   UNIT:   .WORD  0   ; UNIT UNDER TEST
2317 015112 000000   NUNIT: .WORD  0   ; NUMBER OF UNITS PRESENT
2318                ; USED TO KEEP TRACK OF UNIT UNDER TEST
2319 015114 000000   NUNIT: .WORD  0   ; USED TO DETERMIN IF THERE ARE MORE
2320                ; THAN ONE UNIT
2321 015116 000000   SELECT: .WORD  0   ; ALL ONES INDICATE UNIT TO BE SELECTED
2322 015120 000000   UNITSL: .WORD  0   ; UNIT NO. SELECTED
2323
2324 015122 000000   ERFLGS: 0           ; ERROR FLAG
2325
2326 015124 000000   SAVDT:  0           ; SAVE DRIVE TYPE REGISTER
2327                ; FOR COMPARISON IN DRIVE CLEAR TEST
2328                ; AND RH INIT TEST
2329 015126 000000   SAVSN:  0           ; SAVE SERIAL NUMBER REGISTER
2330                ; FOR COMPARISON IN DRIVE CLEAR TEST
2331                ; AND RH INIT TEST
2332
2333 015130 000000   PCJSR:  0           ; SAVE PC OF JSR WHICH GAVE THE ERROR
2334
2335 015132 000000   ATTENT: 0           ; ATTENTION BIT FOR PRESENT UNIT
2336 015134 000000   TOTALAT: 0         ; TOTAL ATTENTION BITS
2337
2338 015136 000000   TMPILL: 0           ; TEMPORARY ILLEGAL FUNCTION
2339
2340 015140 000000   TSECC:  0           ; FLAG TO SAY IF ECC TEST OR NOT
2341                ; WHEN =177777 IT IS AN ECC TEST
2342                ; WHEN =0 IT IS NOT AN ECC TEST
2343
2344 015142 000000   TESDTE: 0           ; FLAG TO SAY IF DRIVE TIMING ERROR OR NOT
2345                ; WHEN = 177777 IT IS A DTE TEST
2346                ; WHEN = 0 IT IS NOT A DTE TEST
2347
2348 015144 000000   TAGDTE: 0           ; TEMPORARY TAG USED IN DRIVE TIMING
2349                ; ERROR TEST
2350

```

2351  
 2352  
 2353  
 2354  
 2355  
 2356  
 2357  
 2358 015146  
 2359 015146 000000  
 2360 015150 000002  
 2361 015152 000006  
 2362 015154 000010  
 2363 015156 000012  
 2364 015160 000030  
 2365 015162 000050  
 2366 015164 000052  
 2367 015166 000060  
 2368 015170 000062  
 2369 015172 000070  
 2370 015174 000072  
 2371 015176 000004  
 2372 015200 000014  
 2373 015202 000016  
 2374 015204 000022  
 2375 015206 000020  
 2376 015210 000000

;FUNCTION EQUATES

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

FUTABL:  
 NOPERA: 0 ;NO OPERATION  
 UNLOAD: 2 ;UNLOAD (STAND BY)  
 RECALI: 6 ;RECALIBRATE  
 DCLEAR: 10 ;DRIVE CLEAR  
 RELEAS: 12 ;RELEASE (DUAL-PORT OPERATION)  
 SERCH: 30 ;SEARCH COMMAND  
 WRCHK: 50 ;WRITE CHECK DATA  
 WRCHDT: 52 ;WRITE CHECK HEADER AND DATA  
 WRIDAT: 60 ;WRITE DATA  
 WRIFOR: 62 ;WRITE HEADER AND DATA (FORMAT)  
 READAT: 70 ;READ DATA  
 REFOR: 72 ;READ HEADER AND DATA  
 SEECOM: 4 ;SEEK COMMAND  
 OFSETC: 14 ;OFFSET COMMAND  
 RETCL: 16 ;RETURN TO CENTERLINE  
 PKACK: 22 ;PACK ACKNOWLEDGE  
 READIN: 20 ;READ IN  
 ILLEGL: .WORD ;COMPUTED ILLEGAL FUNCTION

;DATA BUFFER FOR READ WRITE

2381  
 2382 015212 000422  
 2383 016256 000422  
 2384 017322 000000  
 2385 017324 000000

WRFROM: .BLKW 274. ;WRITE FROM THIS BUFFER  
 REINTO: .BLKW 274. ;READ INTO THIS BUFFER  
 TSTNM: 0 ;TEST NUMBER  
 FIRST: 0 ;IF ZERO WILL TYPE HEADER  
 ;IF ONES WILL NOT TYPE HEADER

;TABLE FOR ATTENTION BITS  
;ATTENTION TABLE

2393  
 2394 017326 001 002 004  
 2395 017331 010 020 040  
 2396 017334 100 200

ATABLE: .BYTE 1,2,4,10,20,40,100,200



```

2397 .SBTTL
2398 .SBTTL ***PROGRAM SETUP & SETUP TESTS***
2399 .SBTTL
2400
2401 017336 012737 177777 015116 BEGIN2: MOV # -1, @#SELECT ; SELECT UNIT
2402 017344 000402 BR START
2403 017346 005037 015116 BEGIN: CLR @#SELECT ; DO NOT SELECT UNIT
2404 ; NORMAL RUN
2405
2406 017352 START:
2407 017352 012737 000340 177776 MOV #340, @#PS ; LOCK OUT ALL INTERRUPTS
2408 017360 012706 001100 MOV #SCMTAG, R6 ; FIRST LOCATION TO BE CLEARED
2409 017364 005026 CLR (R6)+ ; CLEAR MEMORY LOCATION
2410 017366 022706 001136 CMP #STKS, R6 ; DONE?
2411 017372 001374 BNE -6 ; LOOP BACK IF NO
2412 017374 012706 001000 MOV #STACK, SP ; SETUP THE STACK POINTER
2413 017400 012737 056306 000020 MOV #SCOPE, @#IOTVEC ; IOT VECTOR FOR SCOPE ROUTINE
2414 017406 012737 000340 000022 MOV #340, @#IOTVEC+2 ; LEVEL 7
2415 017414 012737 057674 000030 MOV #ERROR, @#EMTVEC ; EMT VECTOR FOR ERROR ROUTINE
2416 017422 012737 000340 000032 MOV #340, @#EMTVEC+2 ; LEVEL 7
2417 017430 012737 060440 000034 MOV #TRAP, @#TRAPVEC ; TRAP VECTOR FOR TRAP CALLS
2418 017436 012737 000340 000036 MOV #340, @#TRAPVEC+2 ; LEVEL 7
2419 017444 012737 060506 000024 MOV #SPWRDN, @#PMRVEC ; POWER FAILURE VECTOR
2420 017452 012737 000340 000026 MOV #340, @#PMRVEC+2 ; LEVEL 7
2421 017460 005067 161520 CLR $TIMES ; INITIALIZE NUMBER OF ITERATIONS
2422 017464 005067 161516 CLR $ESCAPE ; CLEAR THE ESCAPE ON ERROR ADDRESS
2423 017470 112767 000001 161417 MOVB #1, $ERMAX ; ALLOW ONE ERROR PER TEST
2424 017476 012767 017476 161402 MOV #, $I_PADR ; INITIALIZE THE LOOP ADDRESS FOR SCOPE
2425 017504 012767 017504 161376 MOV #, $PERR ; SETUP THE ERROR LOOP ADDRESS
2426
2427
2428
2429 017512 012767 000000 160256 MOV #0, PS ; SET PROCESSOR STATUS TO 0
2430 017520 012777 056220 175220 MOV #RPVECT, @#RPVEC ; THIS IS FOR UNTIMELY RPO4 INTERRUPTS
2431 017526 004737 057146 JSR PC, @#STKINT ; INITILIZE TTY KEYBOARD
2432 017532 005737 017324 TST @#FIRST ; IS THIS FIRST TIME ROUND ?
2433 017536 001001 BNE 1$ ; DON'T TYPE HEADER IF NOT
2434 017540 000402 BR 2$ ; TYPE HEADER IF SO
2435 017542 000137 020636 1$: JMP @#SND1
2436
2437 017546 2$:
2438 017546 104400 017554 TYPE ,.+4 ; TYPE ASCIZ STRING
2439 017552 000435 BR 64$ ; GET OVER THE ASCIZ
2440 ;;.ASCIZ <15><12>/RJPO4 DISKLESS RH11 TEST-PART II (STATIC 18) - DZRPT-D/
2441 017646 64$:
2442 017646 104400 017654 TYPE ,.+4 ; TYPE ASCIZ STRING
2443 017652 000416 BR 65$ ; GET OVER THE ASCIZ
2444 ;;.ASCIZ <15><12>/REVISION DATE: 21-JULY-75/
2445 017710 65$:
2446 017710 104400 017716 TYPE ,.+4 ; TYPE ASCIZ STRING
2447 017714 000402 BR 66$ ; GET OVER THE ASCIZ
2448 ;;.ASCIZ <15><12>/ /
2449 017722 66$:
2450 017722 104400 017730 TYPE ,.+4 ; TYPE ASCIZ STRING

```

```

2451 017726 000432 BR 67$ ;;GET OVER THE ASCIZ
2452 ;;.ASCIZ <15><12>/ALL DCL UNDER TEST MUST BE LOCKED ON CORRECT PORT/
2453 020014 67$: TYPE +4 ;;TYPE ASCIZ STRING
2454 020014 104400 020022 BR 68$ ;;GET OVER THE ASCIZ
2455 020020 000425 ;;.ASCIZ <15><12>/IF CHANGES ARE REQUIRED ON PORT SWITCH,/
2456 020074 68$: TYPE +4 ;;TYPE ASCIZ STRING
2457 020074 104400 020102 BR 69$ ;;GET OVER THE ASCIZ
2458 020074 000430 ;;.ASCIZ <15><12>/A CYCLE UP SEQUENCE IS REQUIRED FOR STROBING/
2459 020100 69$: TYPE +4 ;;TYPE ASCIZ STRING
2460 020162 BR 70$ ;;GET OVER THE ASCIZ
2461 020162 104400 020170 ;;.ASCIZ <15><12>/THE PORT SELECT FLOP/
2462 020162 70$: TYPE +4 ;;TYPE ASCIZ STRING
2463 020166 000414 BR 71$ ;;GET OVER THE ASCIZ
2464 020220 71$: ;;.ASCIZ <15><12>/ /
2465 020220 104400 020226 BR 72$ ;;GET OVER THE ASCIZ
2466 020224 000402 72$: TYPE +4 ;;TYPE ASCIZ STRING
2467 020224 73$: BR 73$ ;;GET OVER THE ASCIZ
2468 020232 73$: ;;.ASCIZ <15><12>/ /
2469 020232 104400 020240 BR 74$ ;;GET OVER THE ASCIZ
2470 020232 000427 74$: TYPE +4 ;;TYPE ASCIZ STRING
2471 020236 75$: BR 75$ ;;GET OVER THE ASCIZ
2472 020316 75$: ;;.ASCIZ <15><12>/ALL DCL NOT UNDER TEST MUST BE SWITCHED OFF/
2473 020316 104400 020324 BR 76$ ;;GET OVER THE ASCIZ
2474 020322 000402 76$: TYPE +4 ;;TYPE ASCIZ STRING
2475 020330 77$: BR 77$ ;;GET OVER THE ASCIZ
2476 020330 104400 020336 77$: ;;.ASCIZ <15><12>/ /
2477 020330 000427 78$: TYPE +4 ;;TYPE ASCIZ STRING
2478 020334 79$: BR 79$ ;;GET OVER THE ASCIZ
2479 020414 79$: ;;.ASCIZ <15><12>/***** /
2480 020414 104400 020422 80$: TYPE +4 ;;TYPE ASCIZ STRING
2481 020420 000427 81$: BR 81$ ;;GET OVER THE ASCIZ
2482 020500 81$: ;;.ASCIZ <15><12>/IF THIS IS NOT DONE, ERRORS WILL RESULT ON/
2483 020500 104400 020506 82$: TYPE +4 ;;TYPE ASCIZ STRING
2484 020504 000415 83$: BR 83$ ;;GET OVER THE ASCIZ
2485 020540 83$: ;;.ASCIZ <15><12>/'RED' TESTS (T21 & T36)/
2486 020540 104400 020546 84$: TYPE +4 ;;TYPE ASCIZ STRING
2487 020544 000427 85$: BR 85$ ;;GET OVER THE ASCIZ
2488 020624 85$: ;;.ASCIZ <15><12>/***** /
2489 020624 104400 020632 86$: TYPE +4 ;;TYPE ASCIZ STRING
2490 020630 000402 87$: BR 87$ ;;GET OVER THE ASCIZ
2491 020636 87$: ;;.ASCIZ <15><12>/ /
2492 020636 012737 177777 017324 SND1: MOV #-1,2#FIRST ;NEXT TIME DO NOT GIVE HEADER
2493 020644 005737 015116 TST 2#SELECT ;WAS IT A 200 START
2494 020650 001435 BEQ TST1 ;BRANCH IF STARTING FROM 200
2495 020652 104400 020660 TYPE +4 ;;TYPE ASCIZ STRING
2496 020656 000423 BR 64$ ;;GET OVER THE ASCIZ
2497 020636 88$: ;;.ASCIZ <15><12>/SELECT UNIT NUMBER TO BE TESTED ? /
2498
2499
2500
2501
2502
2503
2504

```



# H08

```
2513 ;*****
2514 ;*TEST 1 REFERENCE EACH REGISTER
2515 ;* REFERENCE EACH REGISTER BY A MOVE INSTRUCTION
2516 ;*****
2517 020744 000004 TST1: SCOPE
2518 020746 012767 000001 160230 MOV #1,STIMES ;DO 1 ITERATION
2519 020754 012706 001000 MOV #STACK,SP ;SET UP STACK POINTER
2520 020760 012737 000001 017322 MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
2521 020766 012737 057702 000030 MOV #REGSA1,@#EMTVEC ;ERROR VECTOR SO THAT
2522 ; NO REGISTERS ARE SAVED
2523 020774 012737 021022 000004 MOV #2$, @#ERRVEC ;SET UP FOR BUS TIMEOUT
2524 021002 012700 000024 MOV #24, RO ;THERE ARE 24 REG TO TEST
2525 021006 012701 014750 MOV #RH08, R1 ;R1 NOW HAS ADDR OF ADDR OF FIRST REG.
2526 021012 013102 1$: MOV @ (R1)+, R2 ;READ HARDWARE REG.
2527 021014 005300 DEC RO ;COUNT DOWN
2528 021016 001375 BNE 1$ ;BRANCH IF 24 NOT DONE
2529 021020 000471 BR 3$ ;BRANCH IF 24 DONE
2530 021022 012737 000006 000004 2$: MOV #ERRVEC+2,@#ERRVEC ;RESTORE TRAP CATCHER
2531 021030 022626 CMP (SP)+, (SP)+ ;CLEAN STACK
2532 021032 016167 177776 160132 MOV -2(R1), STMP1 ;STORE FAILING REG ADDR
2533 021040 104015 ERROR 1$ ;REGISTER NON EXISTANT
2534 021042 032737 020000 177570 BIT #SW13,@#SWR ;INHIBIT ERROR PRINTOUT ?
2535 021050 001053 BNE 4$ ;BRANCH IF YES
2536 021052 104400 021060 TYPE +4 ;TYPE ASCIZ STRING
2537 021056 000431 BR 64$ ;GET OVER THE ASCIZ
2538 ;:ASCIZ <15><12>/IF BASE ADDRESS IS TO BE CHANGED HALT PROGRAM /
2539 021142 64$: TYPE +4 ;TYPE ASCIZ STRING
2540 021142 104400 021150 BR 65$ ;GET OVER THE ASCIZ
2541 021146 000411 ;:ASCIZ <15><12>/AND RESTART AT /
2542 65$:
2543 021172
2544 021172 012746 050576 MOV #BASECH,-(SP) ;GET READY TO TYPE STARTING ADDRESS
2545 ; OF "CHANGE OF BASE ADDRESS" ROUTINE
2546 021176 104402 TYPOC
2547 021200 000137 043730 4$: JMP @#SEOP ;GO TO END OF PROGRAM
2548 021204 012737 057674 000030 3$: MOV #SEORR,@#EMTVEC ;RESTORE ERROR VECTOR
2549 ; SO THAT REGISTERS ARE SAVED
2550 021212 012737 000006 000004 MOV #ERRVEC+2,@#ERRVEC ;RESTORE TRAP CATCHER
2551
2552
2553
```

```

2554
2555 ;*****
2556 ;*TEST 2          RHCS2-CONTROL AND STATUS 2
2557
2558 ;*          THIS PARTIALLY TESTS RHCS2 TO ENABLE DETERMINATION
2559 ;*          OF THE NUMBER OF DRIVES PRESENT
2560
2561 ;*****
2562 021220 000004      TST2: SCOPE
2563 021222 012767 000001 157754      MOV      #1,STIMES          ;;DO 1 ITERATION
2564 021230 012706 001000              MOV      #STACK,SP          ;RESET STACK
2565
2566 021234 012737 000002 017322      MOV      #TTNO,@TSTNM      ;THIS SAVES TEST NUMBER
2567 021242 013737 014756 021256      MOV      @RHCS2,@UN+2
2568 021250 004537 044720              JSR      RS,@BITST
2569 021254 020017      UN:          ;TEST BITS IN REGISTER
2570 021256 000000      .WORD    0                ;ONLY THESE BITS ARE TEST READ/WRITE
2571 021260 104001      ERROR    1                ;ADDRESS OF REG. BEING TESTED
2572 021262 000207      RTS      PC                ;IN CORRECT DATA RECEIVED
2573                                     ;RETURN TO BLT3 ROUTINE
2574
2575
2576
2577
2578
2579 ;*****
2580 ;*TEST 3          PARTIAL TEST FOR RHAS FOR UNIT NUMBERS PRESENT
2581 ;*****
2582 021264 000004      TST3: SCOPE
2583 021266 012767 000001 157710      MOV      #1,STIMES          ;;DO 1 ITERATION
2584 021274 012737 000003 017322      MOV      #TTNO,@TSTNM      ;THIS SAVES TEST NUMBER
2585 021302 013701 014776              MOV      @RHAS,R1          ;R1 HAS ADDRESS OF RHAS
2586 021306 012711 177777              MOV      #-1,@R1          ;THIS CLEARS RHAS (SURPRISED!)
2587 021312 011137 001126              MOV      @R1,@$BDDAT      ;TEST DATA
2588 021316 105737 001126              TSTB   @$BDDAT
2589 021322 001405              BEQ    TST4                ;BRANCH IF GOOD
2590 021324 005037 001124              CLR    @$SGDDAT           ;GOOD DATA
2591 021330 010137 044716              MOV    R1,@REGADR        ;FAILING REG. RHAS
2592 021334 104001      ERROR    1                ;RHAS DOES NOT CLEAR
2593                                     ;WITH ONES
2594

```

# J08

```
2595  
2596 ;*****  
2597 ;*TEST 4 TEST FOR DRIVES PRESENT USING RHAS AND RHCS2  
2598 ;*****  
2599 021336 000004 TST4: SCOPE  
2600 021340 012767 000001 157636 MOV #1,STIMES ;DO 1 ITERATION  
2601 021346 000005 RESET ;START WITH AN INIT  
2602 021350 004737 057146 JSR PC,@#STKINT ;INITILIZE TK  
2603 021354 032737 020000 177570 BIT #SW13,@#SWR ;INHIBIT ERROR TYPEOUT?  
2604 021362 001030 BNE 45 ;BRANCH IF YES  
2605 021364 104400 021372 TYPE ;TYPE ASCIZ STRING  
2606 021370 000425 BR 645 ;GET OVER THE ASCIZ  
2607 ;;.ASCIZ <15><12>/LOOKING AT RHAS - RPO4 DRIVES PRESENT /  
2608 021444 645:  
2609 021444 013701 014776 45: MOV @#RHAS,R1 ;R1 HAS ADDR. OF RHAS  
2610 021450 013702 014756 MOV @#RHCS2,R2 ;R2 HAS ADDR. OF RHCS2  
2611 021454 005012 CLR @R2 ;CLEAR RHCS2  
2612 021456 012700 000010 MOV #8,R0 ;COUNT  
2613 021462 013704 014762 MOV @#RHER1,R4 ;R4 HAS ADDR. OF RHER1  
2614 021466 012714 177777 15: MOV #-1,@R4 ;MOVE ERRORS INTO RHER1  
2615 021472 005212 INC @R2 ;INCREMENT UNIT NO.  
2616 021474 005300 DEC R0 ;COUNT  
2617 021476 001373 BNE 15 ;BRANCH IF 0 NOT DONE  
2618 021500 111137 015134 MOV @R1,@#TOTALAT ;SAVE TOTAL ATTENTION  
2619 ;USED IN DRIVE CLEAR TEST  
2620 021504 105037 015135 CLRB @#TOTALAT+1 ;CLEAR UPPER BYTE  
2621 021510 105711 TSTB @R1 ;TEST FOR ANY DRIVES PRESENT  
2622 021512 001402 BEQ 25 ;NONE RESPONDING - TYPE THE MESSAGE  
2623 021514 000167 000430 JMP XE2 ;SOME THERE - GO FILL "UNITS" TABLE  
2624  
2625 021520 032737 020000 177570 25: BIT #SW13,@#SWR ;INHIBIT ERROR TYPE OUT?  
2626 021526 001402 BEQ 35 ;"NO DRIVES" MESSAGE IF NO  
2627 021530 000167 000670 JMP SELTST ;CHECK FOR SELECTED UNIT START AND LOAD  
2628 ;"UNITS" TABLE WITH DESIRED DRIVE IF SO  
2629  
2630 021534 35:  
2631 021534 104400 021542 TYPE ;TYPE ASCIZ STRING  
2632 021540 000412 BR 655 ;GET OVER THE ASCIZ  
2633 ;;.ASCIZ <15><12>/NO DRIVES-RHAS=0/  
2634 655:  
2635 021566 104400 021574 TYPE ;TYPE ASCIZ STRING  
2636 021572 000436 BR 665 ;GET OVER THE ASCIZ  
2637 ;;.ASCIZ <15><12>/WRITING ONES INTO ERROR REGISTER #1 FOR ALL UNIT NUMBERS/  
2638 665:  
2639 021670 104400 021676 TYPE ;TYPE ASCIZ STRING  
2640 021674 000441 BR 675 ;GET OVER THE ASCIZ  
2641 ;;.ASCIZ <15><12>/DOES NOT SET ANY BIT IN THE ATTENTION REGISTER SO ABORT P  
2642 675:  
2643 022000 104400 022006 TYPE ;TYPE ASCIZ STRING  
2644 022004 000435 BR 685 ;GET OVER THE ASCIZ  
2645 ;;.ASCIZ <15><12>/TO LOOP ON THIS TEST NO PRINTOUT SET SWITCHES 13, 8 & 2/  
2646 685:  
2647  
2648 022100 000137 043730 JMP @#SEOP ;GO OUT----->
```

## K08

MAINDEC-11-DZRPT-D, RJPO4 DISKLESS RH11 TEST-PART 2 MACY11 27(663) 7-OCT-75 17:33 PAGE 58  
 DZRPTD.SET T4 TEST FOR DRIVES PRESENT USING RHAS AND RHCS2

SEQ 0100

```

2649
2650 022104 104400 022112 TYPE .+4 ;:TYPE ASCIZ STRING
2651 022110 000410 BR 69$ ;:GET OVER THE ASCIZ
2652 ;:ASCIZ <15><12>/TEST DRIVE 0/
2653 022132 69$: CLR @#UNITS
2654 022132 005037 015070 MOV #1,NOUNIT ;NO. UNITS PRESENT=1
2655 022136 012767 000001 172746 CLR @#UNIT
2656 022144 005037 015110 XE2:
2657 022150 2$: MOV #8, RO ;COUNTER
2658 022150 012700 000010 MOV #UNITS,R3 ;POINTER
2659 022154 012703 015070 MOV #-1,(R3)+ ;PRESET BLOCK TO ALL ONES
2660 022160 012723 177777 3$: DEC RO ;COUNT
2661 022164 005300 BNE 3$ ;BRANCH IF 8 NOT DONE
2662 022166 001374 MOV #UNITS,R3 ;POINTER
2663 022170 012703 015070 CLR R5
2664 022174 005005 CLR @#NOUNIT ;NO. OF UNITS PRESENT
2665 022176 005037 015112 MOV #8, RO ;COUNTER
2666 022202 012700 000010 MOV @R1,@$STMPD ;TEMPORARY STORAGE
2667 022206 011137 001170 4$: ROR @$STMPD ;SET CARRY IF ONE IN 0 BIT
2668 022212 006037 001170
2669
2670 022216 103067 BCC 5$
2671 022220 010577 :72532 MOV R5,@RHCS2 ;INSERT UNIT NUMBER
2672 022224 022777 024020 172552 CMP #24020,@RHDT ;IS THIS A DUAL PORT RPO4
2673 022232 001452 BEQ 6$ ;BRANCH IF YES
2674 022234 022777 020020 172542 CMP #20020,@RHDT ;IS THIS A SINGLE PORT RPO4
2675 022242 001446 BEQ 6$ ;BRANCH IF YES
2676 022244 104400 022252 TYPE .+4 ;:TYPE ASCIZ STRING
2677 022250 000410 BR 64$ ;:GET OVER THE ASCIZ
2678 ;:ASCIZ <15><12>/UNIT NUMBER /
2679 64$: MOV R5,-(SP) ;GET READY TO TYPE UNIT NUMBER
2680 022272 010546 TYPDS
2681 022274 104410 TYPE
2682 022276 104400 022304 BR 65$ ;:TYPE ASCIZ STRING
2683 022302 000406 ;:GET OVER THE ASCIZ
2684 ;:ASCIZ / , RHDT = /
2685 65$: MOV @RHDT,-(SP) ;GET READY TO TYPE RHDT
2686 022320 017746 172460 TYPDC
2687 022324 104402 TYPE
2688 022326 104400 022334 BR 66$ ;:TYPE ASCIZ STRING
2689 022332 000411 ;:GET OVER THE ASCIZ
2690 ;:ASCIZ / --- NOT AN RPO4/
2691 66$: BR 5$ ;NO RPO4 FOUND SO BRANCH
2692 022356 000407 5$:
2693 022360 010523 MOV R5,(R3)+
2694 022362 104400 001215 TYPE $CRLF
2695 022366 010546 MOV R5,-(SP) ;PUT DRIVE NO. ON STACK
2696 022370 104410 TYPDS ;TYPE DRIVE NO.
2697 022372 005237 015112 INC @#NOUNIT ;INCR TOTAL NO. OF UNITS
2698
2699 022376 005205 5$: INC R5 ;'RHCS2' UNIT ADDRESS
2700 022400 005300 DEC RO ;DRIVE COUNTER DOWN ONE
2701 022402 001303 BNE 4$ ;TEST AND DO NEXT UNIT IF 8 NOT DONE
2702

```

2703	022404	013737	015070	015110	MOV	@#UNITS,@#UNIT	;SET UNIT NO. TO FIRST ONE FOUND/OR 0
2704	022412	013737	015112	015114	MOV	@#NUNIT,@#NUNIT	;SAVE NO. OF UNITS
2705	022420	005337	015114		DEC	@#NUNIT	;IF NUNIT = 0 THEN ONLY ONE UNIT
2706							;IF NUNIT > 0 THEN MORE THAN ONE UNIT
2707	022424	005737	015116		SELTST: TST	@#SELECT	;STARTING ADDRESS 200 ?
2708	022430	001403			BEG	TST5	;BRANCH IF STARTING FROM 200
2709	022432	013737	015120	015110	MOV	@#UNITSL,@#UNIT	;CHANGE UNIT NUMBER TO SELECTED ONE
2710							



MO8

```

2711 *****
2712 TEST 5 TEST SERIAL NUMBER AND DRIVE TYPEI
2713 READ SERIAL NUMBER REGISTER AND DRIVE TYPE REGISTER
2714 ;* TYPE IT OUT AND PROCEED
2715 ;* TO LOOP HERE SET SWITCH 8 AND THIS TEST NO AND RESTART
2716 *****
2717
2718 TESTS: SCOPE
2719 MOV #1,STIMES ;DO 1 ITERATION
2720 MOV #15,SLPADR ;SET SCOPE LOOP ADDRESS
2721 JSR PC,@CLDISK ;FILL UNIT NO.
2722 CLR @ATTENT ;CLEAR
2723 MOV @UNIT,RO ;RO CONTAINS UNIT NO
2724 MOV @ATTENT,@ATTENT ;SET APPROPRIATE ATTENTION BIT
2725 TYPE +4 ;TYPE ASCIZ STRING
2726 BR 64$ ;GET OVER THE ASCIZ
2727 ;;.ASCIZ <15><12>/TESTING DRIVE NUMBER /
2728 64$:
2729 MOV @UNIT,-(SP) ;UNIT NO. TO STACK
2730 TYPOC ;TYPE DRIVE NO.
2731 TYPE ,SCLF
2732 TYPE +4 ;TYPE ASCIZ STRING
2733 BR 65$ ;GET OVER THE ASCIZ
2734 ;;.ASCIZ <15><12>/SERIAL NO. = /
2735 65$:
2736 MOV @RHSN,-(SP) ;SAVE @RHSN FOR TYPEOUT
2737 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
2738 TYPE ,SCLF
2739 TYPE +4 ;TYPE ASCIZ STRING
2740 BR 66$ ;GET OVER THE ASCIZ
2741 ;;.ASCIZ <15><12>/DRIVE TYPE = /
2742 66$:
2743 MOV @RHDT,-(SP) ;SAVE @RHDT FOR TYPEOUT
2744 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
2745 TYPE ,SCLF
2746 TST @RHSN ;READ SERIAL NO. AND DRIVE TYPE
2747 TST @RHDT ;THESE TWO ARE TO HELP SCOPE LOOPS
2748 MOV @RHSN,@SAVSN ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS
2749 MOV @RHDT,@SAVDT ;SAVE TO CHECK IF CLR RHCS2 BIT 5 CLEARS ANY BITS

```

```

2750
2751
2752 ;*****
2753 ;*TEST 6 CHECK MOL TO BE LOW
2754
2755 ;* MAKE SURE THAT DRIVE IS OFF LINE BEFORE STARTING PROGRAM
2756 ;* IF DRIVE IS ON LINE THEN AFTER TYPE OUT THE PROGRAM WILL
2757 ;* HANG FOR EVER WAITING FOR DRIVE TO GO OFF LINE
2758
2759 ;*****
2760 022676 000004 TST6: SCOPE
2761 022700 012737 000006 017322 MOV #TTNO,@TSTNM ;THIS SAVES TEST NUMBER
2762 022706 004737 045152 JSR PC,@CLDISK ;GIVE INITILIZE
2763 022712 032713 010000 BIT #MOL,@R3 ;CHECK MOL IN RHDS1
2764 022716 001550 BEQ TST7 ;BRANCH IF MOL LOW
2765
2766 022720 104400 022726 TYPE +4 ;:TYPE ASCIZ STRING
2767 022724 000421 BR 64$ ;:GET OVER THE ASCIZ
2768 ;:ASCIZ <15><12>/DRIVE IS ON LINE - MOL IS HIGH/
2769 022770 64$:
2770 022770 104400 022776 TYPE +4 ;:TYPE ASCIZ STRING
2771 022774 000424 BR 65$ ;:GET OVER THE ASCIZ
2772 ;:ASCIZ <15><12>/HIT STOP ON DRIVE TO GET IT OFF LINE/
2773 023046 65$:
2774 023046 104400 023054 TYPE +4 ;:TYPE ASCIZ STRING
2775 023052 000430 BR 66$ ;:GET OVER THE ASCIZ
2776 ;:ASCIZ <15><12>/PROGRAM WILL HANG TESTING MOL TILL MOL IS LOW/
2777 023134 66$:
2778
2779 023134 032713 010000 1$: BIT #MOL,@R3 ;CHECK MOL IN RHDS1
2780 023140 001375 BNE 1$ ;BRANCH IF MOL IS HIGH
2781 023142 104400 023150 TYPE +4 ;:TYPE ASCIZ STRING
2782 023146 000434 BR 67$ ;:GET OVER THE ASCIZ
2783 ;:ASCIZ <15><12>/GOOD - MOL IS NOW LOW . PROGRAM WILL NOW BE EXECUTED/
2784 023240 67$:
2785
2786
2787

```

```

2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798 023240 000004
2799 023242 012706 001000
2800 023246 012737 000007 017322
2801
2802 023254 004737 045152
2803
2804 023260 012777 000001 171512
2805
2806 023266 013777 015204 171464
2807
2808
2809 023274 004037 045644
2810 023300 014752
2811 023302 016256
2812 023304 000023
2813
2814
2815 023306 052777 000001 171444
2816
2817
2818 023314 052737 000100 016306
2819
2820
2821
2822
2823 023322 004037 045644
2824 023326 014752
2825 023330 015212
2826 023332 000023
2827
2828
2829
2830
2831 023334 113737 016303 015237
2832
2833
2834
2835
2836 023342 004037 046046
2837 023346 016256
2838 023350 015212
2839 023352 000023
2840 023354 023362
2841 023356 023362

```

```

;*****
;*TEST 7          PACK ACKNOWLEDGE COMMAND TEST
;
;*   THE PACK ACKNOWLEDGE COMMAND WILL BE LOADED INTO RHC 31 WITH GO
;*   THEN ALL REGISTERS WILL BE CHECKED
;*   RH CLEAR WILL BE GIVEN
;*   THEN ALL REGISTERS WILL BE CHECKED
;*****
TST7:  SCOPE
      MOV      #STACK,SP      ;RESET STACK
      MOV      #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, @RHCS1 ;LOAD PACK ACKNOWLEDGE COMMAND INTO RHCS1
      ;SAVE REGISTERS FOR COMPARISON AFTER GO
      JSR      RO, @#SAVER    ;SAVE
      RHMIC                               ;FROM
      REINTO                               ;TO
      19.                                ;NUMBER OF REGISTERS SAVED
      ;GIVE GO TO PACK ACKNOWLEDGE COMMAND
      BIS      #GO, @RHCS1    ;GO TO PACK ACKNOWLEDGE COMMAND
      ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
      BIS      #VV, @#REINTO+30 ;SAVED RHDS1
      ;AFTER GO HAS BEEN GIVEN TO PACK ACKNOWLEDGE COMMAND
      ;SAVE REGISTERS AGAIN SO THAT COMPARISONS CAN
      ;BE DONE
      JSR      RO, @#SAVER    ;SAVE
      RHMIC                               ;FROM
      WRFROM                               ;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

```

```

2842 023360 023402          2S          ;RETURN FOR GOOD COMPARISON
2843 023362 013705 051722 1S:  MOV     @#ERWORD,R5 ;GETTING READY TO INDEX
2844 023366 060505          ADD     R5,R5 ;DOUBLE ERROR WORD
2845 023370 016537 014750 044716 MOV     RHCC-2(R5),@#REGADR ;FAILING REGISTER ADDRESS
2846
2847 023376 104001          ERROR 1 ;IMPROPER REGISTER CHANGE
2848
2849
2850 023400 000207          RTS     PC ;AFTER PACK ACKNOWLEDGE COMMAND
2851
2852 023402          2S: ;WITH GO IS GIVEN
2853
2854 ;*****
2855 ;*TEST 10 MAKE CURRENT CYLINDER = 0
2856 ;*****
2857 023402 000004          TST10: SCOPE
2858 023404 012706 001000    MOV     #STACK,SP ;RESET STACK
2859 023410 004737 045152    JSR     PC,@#CLDISK ;INIT DRIVE
2860 023414 012777 000001 171356 MOV     @CMD,@RHMR ;SET DIAGNOSTIC MODE
2861 023422 004037 047524    JSR     RO,@#MAKECYL ;SUBROUTINE TO GIVE A SEEK
2862
2863 023426 000000          0 ;COMMAND FOLLOMED BY AN INIT
2864
2865
2866 ;SBTTL
2867 ;SBTTL ***DIAGNOSTIC CODE***
2868 ;SBTTL

```

2869 000004

TIMOT=4

2870

.REM %

2871

THE FOLLOWING TESTS WILL ATTEMPT TO CATCH THOSE BUGS WHICH FAULT  
INSERTION HAS SHOWN US WE MISSED IN THE FIRST PART OF THIS TEST.

2872

2873

2874

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" INSRUCTION.

2875

2876

2877

X

2878

\*\*\*\*\*

2879

TEST 11 BCTA LEGAL REGISTER RESPONSE TEST

2880

\*\*\*\*\*

2881 023430 000004

TST11: SCOPE

2882 023432 012767 000001 155544

MOV #1,STIMES ;DO 1 ITERATION

2883 023440 012737 177777 017322

MOV #TIMO,#TSTNM ;THIS SAVES TEST NUMBER

2884 023446 012777 000040 171302

MOV #CLR,#RHCS2 ;CLEAR RH11 CONTROLLER

2885

2886 023454 012705 014760

MOV #RHCS1,R5 ;R5=LIST POINTER.

2887

2888 023460 013767 000004 155502

MOV #TIMOT,\$TMPD ;SAVE TIMEOUT VECTOR.

2889 023466 012737 023524 000004

MOV #EOD,#TIMOT ;SET VECTOR TO EOD

2890

2891 023474 012706 001000

LOL: MOV #STACK,SP ;SET STACK POINTER.

2892

2893 023500 011502

I00: MOV (R5),R2 ;R2=RH11 ADDRESS.

2894 023502 010267 021210

MOV R2,REGADR

2895 023506 005712

TST (R2) ;DOES THE RH11 REGISTER RESPOND?

2896

2897 023510 062705 000002

ADD #2,R5 ;UPDATE ADDRESS

2898 023514 020527 015012

CMP R5,#RHEC2 ;AT THE END OF LEGAL RH11 REGISTERS?

2899 023520 101767

BLOS I00 ;NOPE

2900 023522 000401

BR 000 ;YES! GOTO 000.

2901

2902 023524 104042

E00: ERROR 42

2903

2904 023526 016737 155436 000004

000: MOV \$TMPD,#TIMOT ;RESTORE TIMEOUT VECTOR.

E09

MAINDEC-11-DZRPT-D, RJPO4 DISKLESS RH11 TEST-PART 2 MACY11 27(663) 7-OCT-75 17:33 PAGE 65  
FLTINS.P11 T11 BCTA LEGAL REGISTER RESPONSE TEST

SEQ 0107

2905

2906  
2907  
2908  
2909  
2910  
2911  
2912  
2913  
2914  
2915  
2916  
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

REM X  
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.

X

\*\*\*\*\*  
; \*TEST 12 BCTA MOV8 LO BYTE TO MC  
\*\*\*\*\*

```

TST12: SCOPE
        MOV     #1,STIMES      ;;DO 1 ITERATION
        MOV     #TINO,#TSTNM   ;;THIS SAVES TEST NUMBER
        MOV     #CLR,#RHCS2    ;;CLEAR RH11 CONTROLLER
        MOV     #STACK,SP     ;;SET STACK POINTER.
        MOV     R1MC,R2        ;;R2=RH11 MC ADDRESS.
        MOV     R2,REGADR      ;;REGISTER ADDRESS TO REGADR FOR TYPING.
        MOV     #377,$GDDAT    ;;$GDDAT=S/B.
L02:   CLR     (R2)           ;;CLEAR RH11 MC.
I02:   MOV8   #377,(R2)      ;;SET MC TO LO BYTE.
        MOV     (R2),$BDDAT    ;;$BDDAT=ACTUAL WAS
        CMP     $BDDAT,$GDDAT  ;;COMPARE RESULTS
        BEQ    TST13          ;;NEXT TEST
        ERROR  44

```

023534	000004		
023536	012767	000001	155440
023544	012737	000012	017322
023552	012777	000040	171176
023560	012706	001000	
023564	016702	171162	
023570	010267	021122	
023574	012767	000377	155322
023602	005012		
023604	112712	000377	
023610	011267	155312	
023614	026767	155306	155302
023622	001401		
023624	104044		

```

2943
2944
2945 ;*****
2946 ;*TEST 13 BCTA MOVB HI BYTE TO MC
2947 ;*****
2947 023626 000004 TST13: SCOPE
2948 023630 012767 000001 155346 MOV #1,STIMES ;DO 1 ITERATION
2949 023636 012737 000013 017322 MOV #TNO,STSTM ;THIS SAVES TEST NUMBER
2950 023644 012777 000040 171104 MOV #CLR,RHCS2 ;CLEAR RH11 CONTROLLER
2951
2952 023652 012706 001000 MOV #STACK,SP ;SET STACK POINTER.
2953
2954 023656 016702 171070 MOV RHMC,R2 ;R2=RH11 MC HI BYTE ADDRESS.
2955 023662 010267 021030 MOV R2,REGADR
2956 023666 012767 177400 155230 MOV #177400,$GDDAT ;$GDDAT=S/B.
2957
2958 023674 005012 L03: CLR (R2) ;CLEAR RH11 MC.
2959
2960 023676 005202 INC R2 ;R2=HI BYTE ADDRESS.
2961
2962 023700 112712 000377 I03: MOVB #377,(R2) ;SET MC HI BYTE.
2963
2964 023704 005302 DEC R2 ;R2=FULL WORD ADDRESS.
2965
2966 023706 011267 155214 MOV (R2),$BDDAT ;$BDDAT=ACTUAL.
2967
2968 023712 026767 155210 155204 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
2969 023720 001401 BEQ TST14 ;NEXT TEST
2970 023722 104044 ERROR 44
2971
2972

```



```

2973
2974
2975
2976
2977 023724 000004
2978 023726 012767 000001 155250
2979 023734 012737 000014 017322
2980 023742 012777 000040 171006
2981
2982 023750 012706 001000
2983
2984 023754 016702 170772
2985 023760 010267 020732
2986 023764 012767 000377 155132
2987
2988 023772 005012
2989
2990 023774 012712 000252
2991 024000 152712 000125
2992
2993 024004 011267 155116
2994
2995 024010 026767 155112 155106
2996 024016 001401
2997 024020 104045
2998
2999

```

```

:*****
:*TEST 14      BCTA BIS8 LO BYTE TO MC
:*****
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      RMC,R2         ;;R2=RH11 MC ADDRESS
      MOV      R2,REGADR      ;;
      MOV      #377,$GDDAT    ;;$GDDAT=S/B.
L04:  CLR      (R2)           ;;CLEAR RH11 MC.
      MOV      #252,(R2)      ;;SET UP MC
I04:  BISB     #125,(R2)     ;;DO A BISB
      MOV      (R2),$BDDAT    ;;$BDDAT=MAS.
      CMP      $BDDAT,$GDDAT  ;;COMPARE RESULTS
      BEQ     TST15          ;;NEXT TEST
      ERROR   45

```

```

3000
3001
3002
3003
3004 024022 000004
3005 024024 012767 000001 155152
3006 024032 012737 000015 017322
3007 024040 012777 000040 170710
3008
3009 024046 012706 001000
3010
3011 024052 016702 170674
3012 024056 010267 020634
3013 024062 012767 177400 155034
3014
3015 024070 005012
3016
3017 024072 005202
3018
3019 024074 112712 000125
3020 024100 152712 000252
3021
3022 024104 005302
3023
3024 024106 011267 155014
3025
3026 024112 026767 155010 155004
3027 024120 001401
3028 024122 104045
3029
3030

```

```

:*****
:*TEST 15      BCTA BISB HI-BYTE TO MC
:*****
TST15: 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 R1MC,R2       ;;R2=RH11 MC ADDRESS.
MOV R2,REGADR
MOV #177400,$GDDAT ;$GDDAT=S/B.
LOS: CLR (R2)     ;;CLEAR RH11 MC.
INC R2            ;;R2 =HI BYTE.
IOS: MOVB #125,(R2) ;SET UP RH11 MC.
BISB #252,(R2)   ;;DO A BISB.
DEC R2           ;;R2=FULL WORD ADDRESS.
MOV (R2),$BDDAT  ;;$BDDAT=NAS.
CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
BEQ TST16        ;;NEXT TEST
ERROR 45

```

```

3031
3032
3033
3034
3035 024124 000004
3036 024126 012767 000001 155050
3037 024134 012737 000016 017322
3038 024142 012777 000040 170606
3039
3040 024150 012706 001000
3041
3042 024154 016702 170572
3043 024160 010267 020532
3044 024164 012767 000252 154732
3045
3046 024172 005012          L06:  CLR      (R2)          ;CLEAR RH11 MC.
3047
3048 024174 012712 000377          MOV      #377,(R2)       ;SET MC=0000377
3049
3050 024200 142712 000125          I06:  BICB   #125,(R2)   ;DO A BICB
3051
3052 024204 011267 154716          MOV      (R2), $BODAT   ;$BODAT=MAS.
3053
3054 024210 026767 154712 154706      CMP      $BODAT,$GDDAT  ;COMPARE RESULTS.
3055 024216 001401          BEQ     TST17          ;NEXT TEST
3056 024220 104046          ERROR  46
3057
3058

```

```

;*****
;TEST 16      BCTA BICB LO-BYTE TO MC
;*****
TST16:  SCOPE

```

```

MOV      #1,STIMES      ;;DO 1 ITERATION
MOV      #TNO,2#TSTNM  ;THIS SAVES TEST NUMBER
MOV      #CLR,2RHCS2   ;CLEAR RH11 CONTROLLER
MOV      #STACK,SP    ;SET STACK POINTER.
MOV      RHMC,R2       ;R2=RH11 MC ADDRESS.
MOV      R2,REGADR     ;
MOV      #252,$GDDAT   ;$GDDAT=S/B.
CLR      (R2)          ;CLEAR RH11 MC.
MOV      #377,(R2)    ;SET MC=0000377
BICB    #125,(R2)     ;DO A BICB
MOV      (R2),$BODAT  ;$BODAT=MAS.
CMP      $BODAT,$GDDAT;COMPARE RESULTS.
BEQ     TST17        ;NEXT TEST
ERROR  46

```

```

3059
3060 ;*****
3061 ;*TEST 17      BCTA BICB HI- BYTE TO MC
3062 ;*****
3063 TST17: SCOPE
3064 024222 000004          MOV      #1,STIMES      ;;DO 1 ITERATION
3065 024224 012767 000001 154752  MOV      #TNO,@TSTNM   ;THIS SAVES TEST NUMBER
3066 024232 012737 000017 017322  MOV      #CLR,@RHCS2   ;CLEAR RH11 CONTROLLER
3067 024240 012777 000040 170510  MOV      #STACK,SP    ;SET STACK POINTER.
3068 024246 012706 001000          MOV      RHMC,R2      ;R2=RH11 MC ADDRESS.
3069 024252 016702 170474          MOV      R2,REGADR
3070 024256 010267 020434          MOV      #125000,$GDDAT ;$GDDAT=S/B.
3071 024262 012767 125000 154634  MOV
3072 024270 005012          L07:    CLR      (R2)    ;CLEAR RH11 MC.
3073 024272 005202          INC      R2           ;R2=HI BYTE.
3074 024274 112712 000377          MOVB    #377,(R2)    ;SET MC=177400
3075 024300 142712 000125          I07:    BICB    #125,(R2) ;DO A BICB
3076 024304 005302          DEC      R2           ;R2=FULL WORD.
3077 024306 011267 154614          MOV      (R2),$BDDAT ;$BDDAT=MAS.
3078 024312 026767 154610 154604  CMP     $BDDAT,$GDDAT ;COMPARE RESULTS.
3079 024320 001401          BEQ     TST20        ;NEXT TEST
3080 024322 104046          ERROR   46
3081
3082
3083
3084
3085
3086
3087
3088
3089
    
```

```

3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102 024324 000004
3103 024326 012767 000001 154650
3104 024334 012737 000020 017322
3105 024342 012777 000040 170406
3106
3107 024350 016702 170436
3108 024354 062702 000002
3109 024360 010267 020332
3110 024364 005067 154536
3111 024370 005067 154530
3112
3113 024374 013767 000004 154566
3114 024402 012737 024420 000004
3115
3116 024410 012706 001000 L10:
3117
3118 024414 005712
3119
3120 024416 104047
3121
3122 024420 016737 154544 000004 010:
3123

```

```

.REM X
THIS TEST ATTEMPT TO ACCESS AN ILLEGAL REGISTER WITHIN THE RANGE
OF ADDRESSES SELECTED BY THE XOR'S. THE RH11 ADDRESS DECODE LOGIC WILL ALLOW UP TO 32
CONTIGOUS REGISTERS TO EXIST IN OUR CONFIGURATION ONLY 16 WILL REALLY
EXIST THIS TEST ATTEMPTS TO ACCESS THE 20(B) REGISTER IF IT RESPONDS
THE TEST WILL TYPE OUT AN ERROR
X

```

```

;*****
;*TEST 20 BCTA ILLEGAL REGISTER TEST
;*****
†ST20: SCOPE
MOV #1,STIMES ;:DO 1 ITERATION
MOV #TNO,#TSTNM ;:THIS SAVES TEST NUMBER
MOV #CLR,#RHCS2 ;:CLEAR RH11 CONTROLLER
MOV RHEC2,R2 ;:R2=RH11 LAST POINTER ADDRESS.
ADD #2,R2 ;:R2=FIRST ILLEGAL ADDRESS.
MOV R2,REGADR
CLR $BDDAT ;:$BDDAT=NAS.
CLR $GDDAT ;:$GDDAT=S/B.
MOV #TIMOT,STMP0 ;:SAVE TIMEOUT VECTOR.
MOV #010,#TIMOT ;:SET TIMEOUT VECTOR TO 010.
L10: MOV #STACK,SP ;:SET THE STACK POINTER.
TST (R2) ;:TEST ADDRESS.
ERROR 47
010: MOV STMP0,#TIMOT ;:RESTORE TIMEOUT VECTOR.

```

```

3124 ;NOW WE ATTACK BTCB
3125 ;
3126 ;
3127 ;REM X
3128 THE FOLLOWING TWO TESTS DETERMINE IF ALL NON DRIVES SET THE
3129 NED BIT AND THAT ALL EXISTING DRIVES CLEAR THE NED BIT.
3130 THE TESTS LOOK AT TOTALAT TO DETERMINE WHICH DRIVES EXIST AND TEST THAT
3131 THESE DRIVES WILL CLEAR THE NED BIT. AND ALSO THAT NON EXISTANT DRIVES
3132 WILL SET THE NED BIT.
3133 ON FIRST PASS ONLY, IF ALL DRIVES ARE ON LINE A MESSAGE WILL SO ADVISE THE
3134 OPERATOR. THE TEST WILL CONTINUE REGARDLESS OF THE OPERATORS ACTION.
3135 ;
3136 ;*****
3137 ;*TEST 21 BCTB (NED) NON-EXISTANT DRIVE TEST. (SET)
3138 ;*****
3139 TST21: SCOPE
3140 MOV #1,STIMES ;DO 1 ITERATION
3141 MOV #TNO,STSTNM ;THIS SAVES TEST NUMBER
3142 MOV #CLR,RHCS2 ;CLEAR RH11 CONTROLLER
3143 ;
3144 CMPB #TOTALAT,#377 ;ARE ALL DRIVES ON LINE.
3145 BNE U11 ;NO GO RUN THE TEST.
3146 ;
3147 CMP SPASS,#0 ;IF PASS #0 THEN TYPE MESSAGE
3148 ;ADVISING OPERATOR
3149 ;THAT THIS TEST WILL NOT BE RUN
3150 BNE Y11
3151 JMP X11
3152 ;
3153 Y11:
3154 TYPE #+4 ;TYPE ASCIZ STRING
3155 BR 64$ ;GET OVER THE ASCIZ
3156 ;;.ASCIZ <15><12>/ALL DRIVES APPEAR TO BE ON LINE THEREFORE/
3157 64$:
3158 TYPE #+4 ;TYPE ASCIZ STRING
3159 BR 65$ ;GET OVER THE ASCIZ
3160 ;;.ASCIZ <15><12>/THE NED NON-EXISTANT DRIVE TEST CANNOT BE/
3161 65$:
3162 TYPE #+4 ;TYPE ASCIZ STRING
3163 BR 66$ ;GET OVER THE ASCIZ
3164 ;;.ASCIZ <15><12>/RUN. TO RUN THIS TEST TAKE ONE OR MORE/
3165 66$:
3166 TYPE #+4 ;TYPE ASCIZ STRING
3167 BR 67$ ;GET OVER THE ASCIZ
3168 ;;.ASCIZ <15><12>/DRIVES OFF-LINE AND RESTART THE PROGRAM/
3169 67$:
3170 ;
3171 U11: MOV #STACK,SP ;SET STACK POINTER.
3172 ;
3173 MOV RHCS2,R2 ;R2=RH11 RHCS2 ADDRESS.
3174 MOV R2,REGADR
3175 ;
3176 MOV #TOTALAT,RO
3177 CLR R1

```

```

3178 025024 006000          S11:  ROR    R0
3179 025026 103423          BCS    N11
3180
3181 025030 010167 154070      R11:  MOV    R1,$GDDAT ;SGDDAT=S/B.
3182 025034 052767 010000 154062  BIS    #NED,$GDDAT
3183
3184 025042 016705 167712          MOV    RHCS1,R5 ;ADDRESS OF A DEVICE REGISTER.
3185
3186 025046 010112          L11:  MOV    R1,(R2) ;LOAD A NON-EXISTANT DRIVE.
3187
3188 025050 011567 154114          MOV    (R5),STMP0 ;ATTEMPT TO READ FROM DEVICE REGISTER.
3189
3190 025054 011267 154046          MOV    (R2),SBDDAT ;SBDDAT=NAS.
3191 025060 042767 167770 154040  BIC    #↑C<NED!US4!US2!US1>,$BDDAT;SBDDAT=SAVED DATA.
3192
3193 025066 026767 154034 154030  CMP    $BDDAT,$GDDAT ;COMPARE RESULTS.
3194 025074 001005          BNE    E11
3195
3196 025076 005201          N11:  INC    R1
3197 025100 020127 000010          CMP    R1,#8. ;TESTED ALL DRIVES YET.
3198 025104          X11:
3199 025104 001402          BEQ    TST22 ;NEXT TEST
3200 025106 000746          BR    S11
3201
3202 025110 104050          E11:  ERROR 50
3203
3204
    
```

```

3205
3206
3207
3208
3209 025112 000004
3210 025114 012767 000001 154062
3211 025122 012737 000022 017322
3212 025130 012777 000040 167620
3213
3214 025136 012706 001000
3215
3216 025142 016702 167610
3217 025146 010267 017544
3218 025152 013700 015134
3219 025156 005001
3220 025160 006000
3221 025162 103020
3222
3223 025164 010167 153734
3224
3225 025170 016705 167564
3226
3227 025174 010112
3228
3229 025176 011567 153766
3230
3231 025202 011267 153720
3232 025206 042767 167770 153712
3233
3234 025214 026767 153706 153702
3235 025222 001005
3236
3237 025224 005201
3238 025226 020127 000010
3239 025232 001402
3240
3241 025234 000751
3242 025236 104050
3243
3244

```

```

;*****
;*TEST 22 BCTE (NED) NON-EXISTANT DRIVE TEST. (CLEARED)
;*****
TST22: SCOPE
MOV #1,STIMES ;DO 1 ITERATION
MOV #TIMO,#TSTNM ;THIS SAVES TEST NUMBER
MOV #CLR,#RHCS2 ;CLEAR RH11 CONTROLLER
MOV #STACK,SP ;SET STACK POINTER.
MOV RHCS2,R2 ;R2=RH11 RHCS2 ADDRESS.
MOV R2,REGADR
MOV #TOTALAT,R0
CLR R1
S12: ROR R0
BCC N12
MOV R1,$GDDAT
MOV RHCS1,R5 ;ADDRESS OF A DEVICE REGISTER.
L12: MOV R1,(R2) ;SELECT UNIT.
MOV (R5),$TMPD ;ATTEMPT TO READ FROM DEVICE.
MOV (R2),$BDDAT ;$BDDAT=MAS.
BIC #1C(NED!US4!US2!US1),$BDDAT ;$BDDAT=SAVED DATA.
CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
BNE E12
N12: INC R1
CMP R1,#8 ;TESTED ALL DRIVES.
BEQ TST23 ;NEXT TEST
E12: BR $12
ERROR $0

```



3245  
3246  
3247  
3248  
3249  
3250  
3251  
3252  
3253  
3254  
3255  
3256  
3257  
3258  
3259  
3260  
3261  
3262  
3263  
3264  
3265  
3266  
3267  
3268  
3269  
3270  
3271  
3272  
3273  
3274  
3275  
3276  
3277  
3278  
3279  
3280  
3281  
3282

```

      REM      X
      TEST TO SEE IF WE CAN READ FROM THE "AS" REGISTER AND NOT CAUSE
      A MED NON-EXISTANT DRIVE ERROR
      X
      *****
      : *TEST 23      BCTB AS REGISTER TEST
      : *****
      TST23: 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.
      L13: MOV      #US4!US2!US1, (R2);:LOAD A NON-EXISTANT DEVICE.
      MOV      RHAS, R2      ;:R2= "AS".
      MOV      (R2), $TMPD   ;: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=MAS.
      BIC      #1C(NED!US4!US2!US1), $BDDAT;:SAVE MED AND DEVICE SELECTED.
      CMP      $BDDAT, $GDDAT ;:COMPARE RESULTS.
      BEQ      TST24        ;:NEXT TEST
      ERROR   51

```

```

3283
3284
3285
3286
3287
3288
3289
3290
3291
3292 025360 000004
3293 025362 012767 000001 153614
3294 025370 012737 000024 017322
3295 025376 012777 000040 167352
3296
3297 025404 012706 001000
3298
3299 025410 016702 167340
3300 025414 010267 017276
3301 025420 012705 025460
3302
3303 025424 012567 153474
3304
3305 025430 016712 153470
3306 025434 011267 153466
3307
3308 025440 026767 153462 153456
3309 025446 001401
3310 025450 104052
3311
3312 025452 005715
3313 025454 001363
3314 025456 000440
3315 025460 000002
3316 025462 000004
3317 025464 000010
3318 025466 000020
3319 025470 000040
3320 025472 000100
3321 025474 000200
3322 025476 000400
3323 025500 001000
3324 025502 002000
3325 025504 004000
3326 025506 010000
3327 025510 020000
3328 025512 040000
3329 025514 100000
3330 025516 177776
3331 025520 177774
3332 025522 177772
3333 025524 177766
3334 025526 177756
3335 025530 177736
3336 025532 177676

```

```

.REM X
THE NEXT THREE TESTS WILL TEST THE BUS ADDRESS REGISTER IN BOTH WORD
AND BYTE MODE. THE PATTERNS ARE CHOSEN TO PICK UP ANY DROPPED OR STUCK
BITS.

```

```

X
;*****
;*TEST 24 BCTC BUS ADDRESS REGISTER
;*****
†ST24: 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 RHBA,R2 ;:R2=RH11 BA ADDRESS.
MOV R2,REGADR
MOV #LST14A,R5 ;:R5=TEST LIST ADDRESS.
L14: MOV (R5)+,$GDDAT ;:$GDDAT=S/B.
I14: MOV $GDDAT,(R2) ;:SET BUS ADDRESS REGISTER.
MOV (R2),$BDDAT ;:READ BUS ADDRESS REGISTER.
CMP $BDDAT,$GDDAT ;:COMPARE RESULTS.
BEQ N14 ;:GET NEXT TEST DATA.
ERROR 52
N14: TST (R5) ;:AT END OF LIST
BNE L14 ;:NO!
BR TST25 ;:NEXT TEST
LST14A: 2
4
10
20
40
100
200
400
1000
2000
4000
10000
20000
40000
100000
177776
177774
177772
177766
177756
177736
177676

```

3337	025534	177576	177576
3338	025536	177376	177376
3339	025540	176776	176776
3340	025542	175776	175776
3341	025544	173776	173776
3342	025546	167776	167776
3343	025550	157776	157776
3344	025552	137776	137776
3345	025554	077776	077776
3346	025556	000000	0
3347			

```

3348
3349
3350 ;*****
3351 ;*TEST 25 BCTC BUS ADDRESS REGISTER LO-BYTE
3352 ;*****
3352 025560 000004 TST25: SCOPE
3353 025562 012767 000001 153414 MOV #1,STIMES ;:DO 1 ITERATION
3354 025570 012737 000025 017322 MOV #TNO,STSTNM ;↑THIS SAVES TEST NUMBER
3355 025576 012777 000040 167152 MOV #CLR,RHCS2 ;CLEAR RH11 CONTROLLER
3356
3357 025604 012706 001000 MOV #STACK,SP ;RESET STACK.
3358
3359 025610 016702 167140 MOV RHBA,R2 ;R2=RH11 BA ADDRESS.
3360 025614 010267 017076 MOV R2,REGADR
3361 025620 012705 025662 MOV #LST15A,R5 ;R5=TEST LIST ADDRESS.
3362
3363 025624 005012 L15: CLR (R2) ;CLEAR RH11 BA REGISTER.
3364
3365 025626 012567 153272 MOV (R5)+,$GDDAT ;$GDDAT=S/B.
3366
3367 025632 116712 153266 I15: MOVB $GDDAT,(R2) ;SET BUS ADDRESS REGISTER.
3368 025636 011267 153264 MOV (R2),$BDDAT ;READ BUS ADDRESS REGISTER.
3369
3370 025642 026767 153260 153254 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3371 025650 001401 BEQ R15
3372
3373 025652 104053 ERROR 53
3374
3375 025654 005715 R15: TST (R5) ;AT END OF TEST LIST.
3376 025656 001362 BNE L15 ;NO!
3377 025660 000417 BR TST26 ;NEXT TEST
3378 ;THIS LIST WILL BE USED TO LOAD THE LOWER BYTE OF THE BA REGISTER.
3379
3380 025662 000002 LST15A: 2
3381 025664 000004 4
3382 025666 000010 10
3383 025670 000020 20
3384 025672 000040 40
3385 025674 000100 100
3386 025676 000200 200
3387 025700 000376 376
3388 025702 000372 372
3389 025704 000366 366
3390 025706 000356 356
3391 025710 000336 336
3392 025712 000276 276
3393 025714 000176 176
3394 025716 000000 0

```

```

3395
3396
3397
3398
3399 025720 000004
3400 025722 012767 000001 153254
3401 025730 012737 000026 017322
3402 025736 012777 000040 167012
3403
3404 025744 012706 001000
3405
3406 025750 016702 167000
3407 025754 010267 016736
3408 025760 012705 026032
3409
3410 025764 005012 L16: CLR (R2) ;CLEAR BA REGISTER.
3411
3412 025766 012567 153132 MOV (R5)+, $GDDAT ;$GDDAT=S/B.
3413
3414 025772 005202 INC R2 ;R2=HI BYTE ADDRESS.
3415
3416 025774 116712 153124 I16: MOV $GDDAT, (R2) ;SET BUS ADDRESS REGISTER HI-BYTE.
3417
3418 026000 005302 DEC R2 ;R2=FULL WORD ADDRESS.
3419
3420 026002 011267 153120 MOV (R2), $BDDAT ;READ BUS ADDRESS.
3421 026006 000367 153112 SWAB $GDDAT ;ADJUST DATA FOR HI BYTE.
3422
3423 026012 026767 153110 153104 CMP $BDDAT, $GDDAT ;COMPARE RESULTS.
3424 026020 001401 BEQ R16 ;OKAY
3425
3426 026022 104053 ERROR 53
3427
3428 026024 005715 R16: TST (R5) ;AT END OF TEST LIST.
3429 026026 001356 BNE L16 ;NOPE
3430 026030 000420 BR TST27 ;NEXT TEST
3431 ;THIS LIST WILL BE USED TO LOAD THE UPPER BYTE OF THE BA REGISTER.
3432
3433 026032 000002 LST16A: 2
3434 026034 000004 4
3435 026036 000010 10
3436 026040 000020 20
3437 026042 000040 40
3438 026044 000100 100
3439 026046 000200 200
3440 026050 000376 376
3441 026052 000374 374
3442 026054 000376 376
3443 026056 000366 366
3444 026060 000356 356
3445 026062 000336 336
3446 026064 000276 276
3447 026066 000176 176
3448 026070 000000 0

```

H10

MAINDEC-11-DZRPT-D, RJPO4 DISKLESS RH11 TEST-PART 2 MACY11 27(663) 7-OCT-75 17:33 PAGE 81  
FLTINS.P11 T26 BCTC BUS ADDRESS REGISTER HI-BYTE

SEQ 0123

3449

```

3450 .REM X
3451 THIS TEST CAUSE AN RH11 INTERRUPT VIA THE SPECIAL COMMAND SEQUENCE
3452 BIS #IE,@RHCS2. THIS TEST PROVES ONLY THAT THE HARDWARE CAN HANDLE
3453 INTERRUPTS PROPERLY
3454 X
3455 ;*****
3456 ;*TEST 27 RH11 INTERRUPT TEST
3457 ;*****
3458 †ST27: SCOPE
3459 026072 000004 MOV #1,STIMES ;DO 1 ITERATION
3460 026074 012767 000001 153102 MOV #TNO,@TSTNM ;THIS SAVES TEST NUMBER
3461 026102 012737 000027 017322 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
3462 026110 012777 000040 166640
3463 026116 012706 001000 MOV #STACK,SP ;SET STACK POINTER.
3464
3465 026122 016702 166632 MOV RHCS1,R2 ;R2=RH11 ADDRESS.
3466 026126 010267 016564 MOV R2,REGADR
3467 026132 005067 152770 CLR $BODAT ;$BODAT=MAS.
3468 026136 005067 152762 CLR $GDDAT ;$GDDAT=S/B.
3469
3470 026142 017767 166600 153020 MOV @RPVEC,$TMPD ;SAVE RH11 INTERRUPT VECTOR.
3471 026150 012777 026174 166570 MOV #021,@RPVEC ;SET RH11 INTERRUPT VECTOR TO 021.
3472
3473 026156 052712 000100 L21: BIS #IE,(R2) ;CAUSE INTERRUPT.
3474 026162 000240 NOP ;WAIT FOR INTERRUPT.
3475 026164 000240 NOP
3476 026166 011267 152732 MOV (R2),$GDDAT ;SAVE CONTENTS OF REGISTER.
3477
3478 026172 104054 ERROR 54
3479
3480 026174 016777 152770 166544 021: MOV $TMPD,@RPVEC ;RESTORE RH11 INTERRUPT VECTOR.
3481
3482

```

```

3483
3484
3485
3486
3487
3488
3489
3490
3491
3492 026202 000004
3493 026204 012767 000001 152772
3494 026212 012737 000030 017322
3495 026220 012777 000040 166530
3496
3497 026226 016702 166526
3498 026232 010267 016460
3499 026236 005067 152662
3500
3501 026242 012706 001000
3502 026246 012712 002000
3503 026252 000005
3504
3505 026254 011267 152646
3506 026260 042767 177677 152640
3507
3508 026266 026767 152634 152630
3509 026274 001401
3510 026276 104055
3511
3512

```

```

;TEST THAT RESET CAN GENERATE THE SIGNAL CLR L.
.REM      X
          X
          THE PROGRAM SETS THE PORT SELECT BIT .THEN DOES A RESET
          IF THE SIGNAL CLR L WAS GENERATED THE PORT SELECT BIT WILL CLEAR.
          X
;*****
;TEST 30      BCTD CLR L TEST
;*****
TST30:  SCOPE
        MOV      #1,STIMES      ;;DO 1 ITERATION
        MOV      #TNO,2#TSTNM   ;;THIS SAVES TEST NUMBER
        MOV      #CLR,2#RHCS2   ;;CLEAR RH11 CONTROLLER
        MOV      RHCS1,R2      ;;R2=RH11 CS1 ADDRESS
        MOV      R2,REGADR
        CLR      $GDDAT        ;;$GDDAT=S/B
L22:    MOV      #STACK,SP      ;;SET STACK POINTER.
        MOV      #PSEL,(R2)    ;;SET PORT SELECT FLOP.
        RESET                    ;;DO A BUSA INIT.
        MOV      (R2), $BDDAT   ;;$BDDAT=MAS.
        BIC      #1C<IE>,$BDDAT ;;SAVE ONLY I.E. BIT
        CMP      $BDDAT,$GDDAT ;;COMPARE RESULTS.
        BEQ      TST31         ;;NEXT TEST
        ERROR   S5

```



3513  
3514  
3515  
3516  
3517  
3518  
3519  
3520  
3521  
3522  
3523  
3524  
3525  
3526  
3527  
3528  
3529  
3530  
3531  
3532  
3533  
3534  
3535  
3536  
3537  
3538  
3539

```

.REM      X
SET THE  MXF FLOP AND READ IT BACK.
          X
:*****
:*TEST 31      MXF TEST
:*****
TST31:  SCOPE
        MOV      #1,STIMES      ;;DO 1 ITERATION
        MOV      #TNO,STSTNM    ;;THIS SAVES TEST NUMBER
        MOV      #CLR,RHCS2     ;;CLEAR RH11 CONTROLLER
        MOV      #STACK,SP      ;;SET STACK POINTER.
        MOV      RHCS2,R2       ;;R2=RH11 CS2 ADDRESS.
        MOV      R2,REGADR
        MOV      #MXF,$GDDAT    ;;SGDDAT=S/B.
I24:    MOV      #MXF,(R2)      ;;LOAD MXF
        MOV      (R2),$BDDAT    ;;$BDDAT=MAS.
        BIC      #1<MXF>,$BDDAT ;SAVE ONLY MXF
        CMP      $BDDAT,$GDDAT  ;;COMPARE RESULTS
        BEQ      TST32          ;;NEXT TEST
        ERROR   56

```

```

026300 000004
026302 012767 000001 152674
026310 012737 000031 017322
026316 012777 000040 166432
026324 012706 001000
026330 016702 166422
026334 010267 016356
026340 012767 001000 152556
026346 012712 001000
026352 011267 152550
026356 042767 176777 152542
026364 026767 152536 152532
026372 001401
026374 104056

```

```

3540 .REM X
3541 SET THE UNIBUS PARITY ERROR FLOP DIRECTLY VIA A MOV #UPE TO RHCS2
3542 ATTEMPT TO READ IT BACK.
3543 X
3544 ;*****
3545 ;*TEST 32 CSR8 UNIBUS PARITY ERROR SET TEST
3546 ;*****
3547 TST32: SCOPE
3548 MOV #1,STIMES ;DO 1 ITERATION
3549 MOV #TNO,#TSTNM ;THIS SAVES TEST NUMBER
3550 MOV #CLR,#RHCS2 ;CLEAR RH11 CONTROLLER
3551
3552 MOV #STACK,SP ;SET STACK POINTER.
3553
3554 MOV RHCS2,R2 ;R2=RHCS2 ADDRESS.
3555 MOV R2,REGADR
3556 MOV #UPE,$GDDAT ;$GDDAT=S/B.
3557
3558 L30: MOV $GDDAT,(R2) ;ATTEMPT TO SET UPE.
3559 NOP
3560 NOP
3561
3562 MOV (R2),$BDDAT ;$BDDAT=ACTUAL DATA.
3563 BIC #C(UPE),$BDDAT ;SAVE ONLY UPE.
3564
3565 MOV $BDDAT,$GDDAT ;COMPARE RESULTS.
3566 BEQ TST33 ;NEXT TEST
3567 ERROR 57
3568
3569

```

3570  
3571  
3572  
3573  
3574  
3575  
3576  
3577  
3578 026500 000004  
3579 026502 012767 000001 152474  
3580 026510 012737 000033 017322  
3581 026516 012777 000040 166232  
3582  
3583 026524 012706 001000  
3584  
3585 026530 016702 166222  
3586 026534 010267 016156  
3587 026540 005067 152360  
3588  
3589 026544 012712 020000  
3590 026550 000240  
3591 026552 000240  
3592  
3593 026554 012712 000040  
3594  
3595 026560 011267 152342  
3596 026564 042767 157777 152334  
3597  
3598 026572 026767 152330 152324  
3599 026600 001401  
3600 026602 104057  
3601  
3602

```

.REM      X
SET THE UNIBUS PARITY ERROR FLOP AND ATTEMPT TO CLEAR IT WITH A
CONTROLLER CLEAR.
      X
;*****
;*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      #1<UPE>, $BDDAT ;; SAVE ERROR.
        CMP      $BDDAT, $GDDAT ;; COMPARE RESULTS.
        BEQ      TST34         ;; NEXT TEST
        ERROR   57

```

3603  
3604  
3605  
3606  
3607  
3608  
3609  
3610  
3611 026604 000004  
3612 026606 012767 000001 152370  
3613 026614 012737 000034 017322  
3614 026622 012777 000040 166126  
3615  
3616 026630 012706 001000  
3617  
3618 026634 016702 166116  
3619 026640 010267 016052  
3620 026644 005067 152254  
3621  
3622 026650 012712 000007  
3623 026654 000240  
3624 026656 000240  
3625  
3626 026660 012712 000040  
3627  
3628 026664 011267 152236  
3629 026670 042767 177770 152230  
3630  
3631 026676 026767 152224 152220  
3632 026704 001401  
3633 026706 104060  
3634  
3635

```

      REM      X
      SET THE UNIT SELECT REGISTER TO DRIVE #7 ALL BITS SET. GENERATE A
      CONTROLLER CLEAR AND VERIFY THAT THE UNIT SELECT REGISTER IS CLEARED.
      X
      ;*****
      ;*TEST 34      CSRB UNIT SELECT CLEAR TEST
      ;*****
      TST34:  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=CS2 ADDRESS.
              MOV      R2,REGADR
              CLR      $GDDAT         ;$GDDAT=5/B.
      L34:    MOV      #US4!US2!US1,(R2) ;LOAD RHCS2 DRIVE SELECT
              NOP
              NOP
              MOV      #CLR,(R2)      ;GENERATE CLR.
              MOV      (R2), $BDDAT    ;READ RHCS2
              BIC      #1C<US4!US2!US1>,$BDDAT;SAVE DRIVE SELECT BITS.
              CMP      $BDDAT,$GDDAT  ;COMPARE RESULTS.
              BEQ      TST35          ;NEXT TEST
              ERROR   60

```

```

3636
3637
3638
3639
3640
3641
3642
3643
3644 026710 000004
3645 026712 012757 000001 152264
3646 026720 012737 000035 017322
3647 026726 012777 000040 166022
3648
3649 026734 012706 001000
3650
3651 026740 016702 166012
3652 026744 012767 040000 152152
3653
3654 026752 012712 020000
3655
3656 026756 016702 165776
3657 026762 010267 015730
3658
3659 026766 011267 152134
3660 026772 042767 137777 152126
3661
3662 027000 026767 152122 152116
3663 027006 001401
3664 027010 104061
3665
3666

```

```

.REM X
SET THE UPE FLOP UNIBUS PARITY ERROR. VERIFY THAT THE SETTING OF (UPE)
ALSO SETS THE TRANSFER ERROR (TRE) FLOP
X

```

```

;*****
;TEST 35 CSR8 TRANSFER ERROR (TRE) - UPE
;*****
TST35: 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=RHCS2 ADDRESS.
MOV #TRE,$GDDAT ;$GDDAT=S/B.

L35: MOV #UPE,(R2) ;SET UNIBUS PARITY ERROR.

MOV RHCS1,R2 ;R2=RHCS1 ADDRESS.
MOV R2,REGADR

MOV (R2), $BDDAT ;READ REGISTER
BIC #1<TRE>,$BDDAT ;SAVE TRANSFER ERROR.

CMP $BDDAT,$GDDAT ;COMPARE RESULTS
BEQ TST36 ;NEXT TEST
ERROR 61

```

```

3667
3668
3669 .REM X
3670 SET THE NED BIT NON EXISTANT DRIVE VERIFY THAT THIS SETS THE (TRE) BIT.
3671 X
3672 ;*****
3673 ;*TEST 36 CSR8 TRANSFER ERROR (TRE) NED
3674 ;*****
3675 TST36: SCOPE
3676 MOV #1,STIMES ;:DO 1 ITERATION
3677 MOV #TNO,#TSTNM ;THIS SAVES TEST NUMBER
3678 MOV #CLR,#RHCS2 ;CLEAR RH11 CONTROLLER
3679 MOV #STACK,SP ;SET STACK POINTER.
3680
3681 MOV RHCS2,R2 ;R2=CS2 ADDRESS.
3682 MOV #TRE,$GDDAT ;$GDDAT=S/B.
3683
3684 MOV #TOTALAT,RO ;GET ALL AVAILABLE DRIVES DATA
3685 CLR R1
3686 S36: ROR RO
3687 BCS N36
3688 L36: MOV R1,(R2) ;SET NED.
3689
3690 MOV RHCS1,R2 ;R2=RHCS1 ADDRESS.
3691 MOV R2,REGADR
3692
3693 MOV (R2),$BDDAT ;READ REGISTER.
3694 BIC #1<TRE>,$BDDAT ;SAVE TRE.
3695
3696 CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
3697 BEQ TST37 ;NEXT TEST
3698 ERROR 62
3699 BR TST37 ;NEXT TEST
3700
3701 N36: INC R1
3702 CMP R1,#8.
3703 BNE S36
3704
3705

```

```

3706
3707
3708
3709
3710
3711
3712
3713 027136 000004
3714 027140 012767 000001 152036
3715 027146 012737 000037 017322
3716 027154 012777 000040 165574
3717
3718 027162 012706 001000
3719
3720 027166 016702 165566
3721 027172 010267 015520
3722 027176 005067 151722
3723
3724 027202 012712 002000 L40:
3725
3726 027206 012777 000040 165542
3727
3728 027214 011267 151706
3729 027220 042767 175777 151700
3730
3731 027226 026767 151674 151670
3732 027234 001401
3733 027236 104064
3734

```

```

.REM X
SET THE PORT SELECT FLOP VERIFY THAT WE CAN READ IT BACK.
X
;*****
;#TEST 37 CSRA PSEL CLEAR TEST
;*****
TST37: 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 RHCS1,R2 ;R2=RHCS1 ADDRESS.
MOV R2,REGADR
CLR $GDDAT ;$GDDAT=S/B.
L40: MOV #PSEL,(R2) ;SET P SELECT.
MOV #CLR,#RHCS2 ;DO A CONTROLLER CLEAR.
MOV (R2),$BDDAT ;READ BACK P SELECT BIT.
BIC #1<PSEL>,$BDDAT ;SAVE ONLY PORT SELECT.
CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
BEQ TST40 ;NEXT TEST
ERROR 64

```

```

3735
3736
3737
3738
3739
3740
3741
3742 027240 000004
3743 027242 012767 000001 151734
3744 027250 012737 000040 017322
3745 027256 012777 000040 165472
3746
3747 027264 012706 001000
3748
3749 027270 016702 165464
3750 027274 010267 015416
3751 027300 005067 151620
3752
3753 027304 012712 000011
3754
3755 027310 012777 000040 165440
3756
3757 027316 011267 151604
3758 027322 042767 177770 151576
3759
3760 027330 026767 151572 151566
3761 027336 001401
3762 027340 104065
3763

```

```

.REM X
VERIFY THAT CONTROLLER CLEAR WILL CLEAR THE COMMAND REGISTER.
X

```

```

;*****
;TEST 40 CSRB COMMAND REGISTER CLEAR TEST
;*****

```

```

TST40: SCOPE
MOV #1,STIMES ;:DO 1 ITERATION
MOV #TNO,2#TSTNM ;THIS SAVES TEST NUMBER
MOV #CLR,2RHCS2 ;CLEAR RH11 CONTROLLER
MOV #STACK,SP ;SET STACK POINTER.
MOV RHCS1,R2 ;R2=RHCS1 ADDRESS.
MOV R2,REGADR
CLR $GDDAT ;$GDDAT=S/B.
L41: MOV #11,(R2) ;ISSUE A DRIVE CLR AND GO.
MOV #CLR,2RHCS2 ;CONTROLLER CLEAR.
MOV (R2),$BDDAT ;READ COMMAND REGISTER.
BIC #1C<7>,$BDDAT ;SAVE ONLY THE COMMAND BITS.
CMP $BDDAT,$GDDAT ;COMPARE RESULTS.
BEQ TST41 ;NEXT TEST
ERROR 65

```



```

3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775 027342 000004
3776 027344 012767 000001 151632
3777 027352 012737 000041 017322
3778 027360 012777 000040 165370
3779
3780 027366 012706 001000
3781
3782 027372 016702 165362
3783 027376 010267 015314
3784 027402 005067 151516
3785
3786 027406 012767 000340 150362
3787
3788 027414 012712 000011
3789
3790 027420 011267 151544
3791
3792 027424 011267 151476
3793 027430 042767 177770 151470
3794
3795 027436 026767 151464 151460
3796 027444 001401
3797 027446 104066
3798

```

```

.REM      X
THE COMMAND REGISTER IS SUPPOSED TO BE SELF CLEARING THAT IS WHEN THE
RH11 IS SELECTED WHETHER OR NOT WE ADDRESS IT DIRECTLY THE LOGIC
SHOULD CLEAR THE COMMAND REGISTER.
THIS IS TESTED BY LOADING A COMMAND INTO IT READING ANOTHER REGISTER
THAN RETURNING TO CHECK THE COMMAND REGISTER TO DETERMINE IF IT CLEARED.
      X
;*****
;*TEST 41      CSR8 COMMAND REGISTER RESELECT CLEAR TEST
;*****
TST41:  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      RHCS1,R2       ;;R2=RHCS1 ADDRESS.
        MOV      R2,REGADR
        CLR      $GDDAT        ;;$GDDAT=S/B.
        MOV      #340,PS        ;;SET PRIORITY TO #7.
L42:    MOV      #11,(R2)       ;;ISSUE A DRIVE CLR AND GO
        MOV      (R2),$TMPD     ;;DO A RESELECT OF THE REGISTER.
        MOV      (R2),$BDDAT    ;;READ THE COMMAND REGISTER.
        BIC      #1C(7),$BDDAT  ;;SAVE ONLY THE COMMAND BITS.
        CMP      $BDDAT,$GDDAT  ;;COMPARE RESULTS.
        BEQ      TST42         ;;NEXT TEST
        ERROR   66

```

```

3799
3800 .REM X
3801 HERE WE TEST TO SEE IF SETTING (IE) AND (RDY) WILL CAUSE AN INTERRUPT.
3802 X
3803 ;*****
3804 ;*TEST 42 CSR8 READY AND IE INTERRUPT TEST
3805 ;*****
3806 TST42: SCOPE
3807 MOV #1,STIMES ;DO 1 ITERATION
3808 MOV #TNO,@TSTNM ;THIS SAVES TEST NUMBER
3809 MOV #CLR,@RHCS2 ;CLEAR RH11 CONTROLLER
3810
3811 MOV RHCS1,R2 ;R2=RPCS1 ADDRESS.
3812 MOV R2,REGADR
3813 CLR $BODAT ;$BODAT=MAS.
3814 CLR $GDDAT ;$GDDAT=S/B.
3815
3816 MOV @RPVEC,$TMPD ;SAVE RH11 INTERRUPT VECTOR.
3817 MOV #043,@RPVEC ;SET RH11 INTERRUPT VECTOR 043
3818
3819 L43: MOV #STACK,SP ;SET STACK POINTER.
3820 CLR PS
3821
3822 BIS #IE!RDY,(R2) ;THIS WILL CAUSE AN INTERRUPT.
3823 NOP
3824 NOP
3825
3826 MOV (R2),$GDDAT
3827 ERROR 67
3828
3829 MOV $TMPD,@RPVEC ;RESTORE RH11 INTERRUPT VECTOR.

```

# H11

3830  
3831  
3832  
3833  
3834  
3835  
3836  
3837  
3838  
3839  
3840  
3841  
3842  
3843  
3844  
3845  
3846  
3847  
3848  
3849  
3850  
3851  
3852  
3853  
3854  
3855  
3856  
3857  
3858  
3859  
3860  
3861  
3862  
3863  
3864  
3865  
3866  
3867  
3868

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

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

TST43: SCOPE  
MOV #1,STIMES ;DO 1 ITERATION  
MOV #TNO,#TSTNM ;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 @RPVEC,\$TMPD ;SAVE RH11 INTERRUPT VECTOR.  
MOV #044,@RPVEC ;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.  
O44: MOV (R2),\$BDDAT ;READ RHCS1.  
BIC #IC<IE>,\$BDDAT ;SAVE ONLY THE "IE" BIT.  
CMP \$BDDAT,\$GDDAT ;COMPARE RESULTS.  
BEQ R44 ;OK!  
E44: ERROR 70  
R44: MOV \$TMPD,@RPVEC ;RESTORE RH11 INTERRUPT VECTOR.

```

3869
3870
3871 .REM X
3872 HERE WE VERIFY THAT WRITING INTO THE "AS" REGISTER OWILL NOT CAUSE
3873 AN (NED) ERROR.
3874 X
3875 :*****
3876 :*TEST 44 BCTB "AS" WRITE TEST
3877 :*****
3878 †ST44: SCOPE
3879 MOV #1,STIMES ;:DO 1 ITERATION
3880 MOV #TNO,#TSTNM ;:THIS SAVES TEST NUMBER
3881 MOV #CLR,#RHCS2 ;:CLEAR RH11 CONTROLLER
3882 MOV #STACK,SP ;:SET STACK POINTER.
3883
3884 MOV RHCS2,R2 ;:R2=RH11 CS2 ADDRESS.
3885 MOV #US4!US2!US1,$GDDAT;$GDDAT=S/B.
3886
3887 MOV RHCS1,RS ;:ADDRESS OF A DEVICE REGISTER.
3888
3889 L45: MOV #US4!US2!US1,(R2);LOAD A NON EXISTANT DEVICE.
3890
3891 MOV RHAS,R2 ;:R2="AS" ADDRESS.
3892
3893 MOV #377,(R2) ;:ATTEMPT TO LOAD "AS".
3894
3895 CLR (R2) ;:CLEAR "AS".
3896
3897 MOV RHCS2,R2 ;:R2=RH11 CS2 ADDRESS.
3898 MOV R2,REGADR
3899
3900 MOV (R2),$BDDAT ;:SBDDAT=MAS.
3901 BIC #1C<NED!US4!US2!US1>,$BDDAT;SAVE NED AND DEVICE SELECTED.
3902
3903 CMP $BDDAT,$GDDAT ;:COMPARE RESULTS.
3904 BEQ TST45 ;:NEXT TEST
3905 ERROR 71
3906

```

.SBTTL EXTENDED MEMORY, DRIVE TIMING & SECTOR SELECT TESTS

3907  
3908  
3909  
3910  
3911  
3912  
3913 030034 000004  
3914 030036 012706 001000  
3915 030042 004737 045152  
3916 030046 012777 000001 164724  
3917 030054 004037 047524  
3918  
3919 030060 000000  
3920  
3921  
3922  
3923  
3924  
3925  
3926  
3927  
3928  
3929  
3930  
3931  
3932  
3933  
3934  
3935  
3936 030062 000004  
3937  
3938 030064 012706 001000  
3939 030070 012737 000046 017322  
3940 030076 004037 045070  
3941 030102 053520  
3942 030104 054544  
3943 030106 000000  
3944  
3945  
3946  
3947 030110 012737 010000 051602  
3948  
3949 030116 005037 051604  
3950 030122 005037 051606  
3951 030126 005037 051610  
3952 030132 012737 000400 051650  
3953 030140 012737 000001 051612  
3954 030146 004537 046360  
3955 030152 051602  
3956 030154 053502  
3957  
3958  
3959  
3960

```
*****  
;*TEST 45 MAKE CURRENT CYLINDER = 0  
*****  
↑ST45: SCOPE  
MOV #STACK, SP ;RESET STACK  
JSR PC, @#CLDISK ;INIT DRIVE  
MOV #DMD, @RHMR ;SET DIAGNOSTIC MODE  
JSR RD, @#MAKECYL ;SUBROUTINE TO GIVE A SEEK  
;COMMAND FOLLOMED BY AN INIT  
;THIS SHUOLD CHANGE RHCC TO 0  
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  
JSR RD, @#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 @#SECOTR ;SECTOR 0 TRACK 0  
CLR @#KEY1 ;KEY1 0  
CLR @#KEY2 ;KEY2 0  
MOV #256, @#NOWORD ;NO OF DATA WORDS  
MOV #1, @#X ;WRITE DATA  
JSR RS, @#CRC ;GO TO CALCULATE CRC  
CYL  
MCRC
```

;THESE ARE REGULAR SETUPS

# K11

3961	030156	004737	045152			JSR	PC, @#CLDISK	; SETUP GENERAL REGISTERS
3962	030162	012777	177400	164562		MOV	#-256, @RHMC	; 256 DATA WORDS
3963	030170	013777	001136	164556		MOV	@#STKS, @RHBA	; STARTING ADDRESS OF WRITE BUFFER
3964	030176	017737	150734	015136		MOV	@#STKS, @TMPILL	; TEMPORARY STORAGE OF DATA
3965	030204	005077	164554			CLR	@RHDS1	; SECTOR 0 TRACK 0
3966	030210	012777	010000	164552		MOV	#FMT22, @RHOF	; 16 BITS PER WORD FORMAT
3967	030216	005077	164550			CLR	@RHCA	; CYLINDER 0
3968	030222	004737	045206			JSR	PC, @#CHECKT	; CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
3969	030226	104400	005116			TYPE	, CPHALT	; CANNOT CONTINUE TESTING IF ANY OF THE
3970								; ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
3971	030232	000000				HALT		; STOP THE TEST
3972	030234	013746	015166			MOV	@#WRIDAT, -(SP)	; WRITE DATA=60
3973	030240	052716	001400			BIS	#A16!A17, (SP)	; SET HIGH ORDER UNIBUS BITS
3974	030244	012611				MOV	(SP)+, @R1	; FILL RHCSI
3975	030246	052777	000010	164502		BIS	#BA1, @RHCS2	; SET BUS ADDRESS INHIBIT
3976	030254	005037	015122			CLR	@#ERFLG	; CLEAR ERROR FLAG
3977	030260	004737	051456			JSR	PC, @#COMHD	; WRITE DATA

```

3978
3979
3980
3981
3982
3983
3984
3985
3986 030264 004737 044652 JSR PC, @PUTREG ;SAVE REGISTERS
3987 030270 005737 015122 TST @ERFLGS ;HAVE ANY ERRORS OCCURED?
3988 030274 001042 BNE TST47 ; BRANCH OUT IF YES
3989 030276 013700 015136 MOV @TMPILL, R0 ;GOOD DATA
3990 030302 012701 053520 MOV @DISK, R1 ;DATA WRITTEN INTO "DISK"
3991 030306 012702 000400 MOV #256, R2 ;COUNTER
3992 030312 012737 000401 051722 1$: MOV @257, @ERWORD ;FOR ERROR WORD
3993 030320 020021 CMP R0, (R1)+ ;COMPARE GOOD DATA WITH DATA ON DISK
3994 030322 001425 BEQ 35 ;BRANCH IF GOOD
3995 030324 013737 015136 001124 MOV @TMPILL, @SGDDAT ;GOOD DATA
3996 030332 014137 001126 MOV -(R1), @SBDDAT ;BAD DATA
3997 030336 160237 051722 SUB R2, @ERWORD ;ERROR WORD NO
3998 030342 005737 015122 TST @ERFLGS ;ANY ERRORS ALREADY THERE?
3999 030346 001002 BNE 25 ;BRANCH IF YES
4000 030350 104037 ERROR 37 ;ERROR ON WRITE DATA COMMAND
4001 ;SEE NEXT ERROR COMMENTS
4002 030352 000401 BR 645 ;BRANCH TO AVOID PRINTING NEXT ERROR
4003 030354 104040 2$: ERROR 40 ;WORD NO GIVES WORD IN ERROR
4004 ;ERROR OCCURED WHILE WRITING
4005 ;WITH A16 A17 OF RHCSI SET
4006 030356 005721 64$: TST (R1)+ ;UNDO -(R1) FOR BAD DATA
4007 030360 013746 177570 MOV @SWR, -(SP) ;GET SWITCH SETTING
4008 030364 042716 177177 BIC #177177, (SP) ;KEEP ONLY SWITCH 7 AND 8
4009 030370 022726 000200 CMP #SW07, (SP)+ ;IS 7 SET AND 8 RESET
4010 030374 001402 BEQ TST47 ; BRANCH OUT IF YES
4011 030376 005302 3$: DEC R2 ;IF NOT COUNT 256 WORDS
4012 030400 001344 BNE 15 ;BRANCH IF 256 NOT DONE
    
```

; 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 SYNC'S WERE CORRECTLY DETECTED.

; DATA IS TO BE CHECKED.

; 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

4013  
4014  
4015  
4016  
4017  
4018  
4019  
4020  
4021  
4022  
4023  
4024  
4025  
4026  
4027 030402 000004  
4028 030404 012706 001000  
4029 030410 012737 000047 017322  
4030  
4031  
4032  
4033 030416 012737 010000 051602  
4034  
4035 030424 005037 051604  
4036  
4037 030430 005037 051606  
4038 030434 005037 051610  
4039 030440 005037 051612  
4040 030444 004537 046360  
4041 030450 051602  
4042 030452 053502  
4043  
4044  
4045  
4046 030454 004737 045152  
4047 030460 012777 177374 164264  
4048 030466 012777 016256 164260  
4049 030474 005077 164264  
4050  
4051 030500 012777 014000 164262  
4052  
4053 030506 005077 164260  
4054 030512 004737 045206  
4055 030516 104400 005116  
4056  
4057 030522 000000  
4058 030524 013711 015174  
4059  
4060  
4061  
4062 030530 004037 045644  
4063 030534 014752  
4064 030536 015022  
4065 030540 000023  
4066

\*\*\*\*\*  
;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 #SECTR ;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  
MCRC

; THESE ARE REGULAR SETUPS

JSR PC, #CLDISK ;SETUP GENERAL REGISTERS  
MOV #-260, #RHMC ;256 DATA WORDS, 4 HEADER WORDS  
MOV #REINT0, #RHBA ;STARTING ADDRESS OF BUFFER  
CLR #RHDS ;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 #REFOR, #R1 ;READ HEADER AND DATA = 72

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

JSR R0, #SAVER ;READ IN SEQUENCE  
RHMC ;FROM HARDWARE REGISTER  
MC ;INTO CORE AT LOCATION  
19. ;NUMBER OF REGISTERS TO READ



```

4067
4068
4069
4070
4071
4072
4073
4074 030542 012737 177777 015142 MOV # -1, @#TESDTE ;SET DTE TEST
4075 030550 012737 177777 015122 MOV # -1, @#ERFLGS ;THIS WILL BRING THE READ HEADER
4076 ;AND DATA PROCESS OUT AFTER THE
4077 ;HEADER HAS BEEN CORRECTLY READ
4078
4079 030556 004737 051456 JSR PC, @#COMHD ;ISSUE 'GO', SEARCH FOR THE SECTOR
4080 ;AND READ THE HEADER
4081
4082 030562 017737 164172 001170 SETCK1: MOV @RHCS1, @#STMPD ;READ CS1 TO CHECK FOR ANY READ ERRORS
4083 030570 032737 100000 001170 BIT #SC, @#STMPD ;TEST FOR "SPECIAL CONDITION" - 'SC'
4084 030576 001405 BEQ BS ;CONTINUE WITH TEST IF 'SC' = 0 (NO ERRORS)
4085 030600 004737 044652 JSR PC, @#PUTREG ;READ & SAVE ALL REGISTERS AGAIN IF AN
4086 ;UNDEFINED DATA TRANSFER ERROR OCCURRED
4087 030604 104040 ERROR 40 ;READ/WRITE HEADER & DATA ERROR DURING
4088 030606 000167 000262 JMP TST50 ; 'DTE' TEST - ABORT THE TEST
4089
4090 030612 BS: ;NOW THE HEADER HAS BEEN READ OK
4091 ;NOW 560 SECTOR CLOCKS WILL BE GIVEN
4092 ;GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
4093 ;GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 2
4094
4095 ;THESE 560 SECTOR CLOCKS ARE DIVIDED INTO TWO GROUPS
4096 ;24 SECTOR CLOCKS WITH NORMAL DIAGNOSTIC DATA CLOCKS,
4097 ;AND 536 SECTOR CLOCKS WITHOUT ANY DIAGNOSTIC DATA CLOCKS.
4098
4099
4100 ;THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC DATA CLOCKS
4101 ;WHICH EQUALS 3 BYTES OF DATA
4102
4103 030612 012701 000030 MOV #24, R1 ;LOAD COUNTER
4104 030616 013700 015000 MOV @#RHMR, R0 ;GET RHMR ADDRESS
4105 030622 012710 000001 MOV #DMD, @RO ;SET DIAGNOSTIC MODE
4106 030626 052710 000012 15: BIS #MSTCK!MCLK, @RO ;SET SECTOR CLOCK AND DATA CLOCK
4107 030632 042710 000012 BIC #MSTCK!MCLK, @RO ;CLEAR SECTOR CLOCK AND DATA CLOCK
4108 030636 012702 000007 MOV #7, R2 ;LOAD COUNTER FOR DIAGNOSTIC CLOCKS
4109 030642 052710 000002 45: BIS #MCLK, @RO ;SET CLOCK
4110 030646 042710 000002 BIC #MCLK, @RO ;CLEAR CLOCK
4111 030652 005302 DEC R2 ;COUNT TO 7
4112 030654 001372 BNE 45 ;BRANCH IF 7 NOT DONE
4113 030656 005301 DEC R1 ;COUNT TO 24
4114 030660 001362 BNE 15 ;BRANCH IF 24 NOT DONE
4115
4116 ;THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
4117
4118 030662 012701 001030 MOV #536, R1 ;LOAD SECTOR CLOCK COUNTER
4119 030666 052710 000010 55: BIS #MSTCK, @RO ;SET SECTOR CLOCK
4120 030672 042710 000010 BIC #MSTCK, @RO ;CLEAR SECTOR CLOCK

```

```

4121 030676 005301 DEC R1 ;COUNT
4122 030700 001372 BNE 55 ;BRANCH IF 536 NOT DONE
4123
4124
4125 ;NOW 'DTE' SHOULD BE SET
4126 ;CHANGE SAVED REGISTERS TO EXPECTED VALUES
4127
4128 030702 012737 177400 015022 MOV #-256, @#MC ;SAVED RHMC
4129 030710 012737 016266 015024 MOV #REINT0+(4.*2), @#BA ;SAVED RHBA
4130 030716 052737 140000 015030 BIS #SC!TRE, @#CSI ;SAVED RHCSI
4131 030724 052737 010000 015032 BIS #DTE, @#ERI ;SAVED RHERI
4132 030732 012737 000401 015050 MOV #401, @#MR ;SAVED RHMR
4133 030740 052737 140000 015052 BIS #ATA!ERR, @#DSI ;SAVED RHDSI
4134 030746 012737 000100 015064 MOV #100, @#LA ;SAVED RHLA
4135 030754 012737 000001 015034 MOV #1, @#DST ;SAVED RHDST
4136 030762 013737 015132 015046 MOV @#ATTENT, @#AS ;SAVED RHAS
4137
4138
4139 ;NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS
4140 ;CAN BE DONE (USE THE 'MRFROM' SAVE BUFFER THIS TIME)
4141
4142 030770 004037 045644 JSR RO, @#SAVER ;READ IN SEQUENCE
4143 030774 014752 RMC ;FROM HARDWARE REGISTER
4144 030776 015212 MRFROM ;INTO CORE BUFFER
4145 031000 000023 19. ;NUMBER OF REGISTERS TO READ
4146
4147 ;FOR RHAS UPPER BYTE
4148 031002 113737 015047 015237 MOVB @#AS+1, @#MRFROM+25 ;UPPER RHAS
4149
4150 ;COMPARE THE HEADER READ
4151
4152 031010 004037 046046 JSR RO, @#COMPAR ;COMPARE
4153 031014 051602 CYL ;GOOD BUFFER
4154 031016 016256 REINT0 ;TEST BUFFER
4155 031020 000004 4. ;NUMBER
4156 031022 031030 65 ;RETURN FOR ERROR
4157 031024 031030 65 ;SAME
4158 031026 031034 75 ;RETURN FOR GOOD COMPARISON
4159 031030 104010 65: ERROR 10 ;HEADER READ IN DURING THIS TEST IS
4160 ;IN ERROR
4161
4162 031032 000207 RTS PC ;RETURN
4163
4164 031034 75: ;GOOD COMPARISON, CONTINUE
4165
4166
4167 ;COMPARE REGISTERS BEFORE COMMAND WITH REGISTERS AFTER COMMAND
4168
4169 031034 004037 046046 JSR RO, @#COMPAR ;COMPARE
4170 031040 015022 MC ;INITIAL SNAPSHOT BUFFER (CHANGED)
4171 031042 015212 MRFROM ;TEST SNAPSHOT BUFFER
4172 031044 000022 18. ;NUMBER OF REGISTERS
4173 031046 031054 25 ;RETURN FOR ERROR
4174 031050 031054 25 ;SAME

```

```

4175 031052 031074          3$          ;RETURN FOR GOOD COMPARISON
4176
4177 031054 013705 051722    2$:      MOV      2#ERWORD,R5      ;GETTING READY TO INDEX
4178 031060 060505          ADD      R5,R5          ;DOUBLE ERROR WORD
4179 031062 016537 014750 044716  MOV      R#C-2(R5),2#REGADR ;FAILING REGISTER
4180 031070 104001          ERROR    1              ;IMPROPER REGISTER
4181                                     ;CHANGE AFTER FORCING
4182                                     ;'DTE' ERROR
4183 031072 000207          RTS      PC              ;RETURN
4184
4185 031074          3$:          ;GOOD - REGISTERS OK, GO ON TO NEXT TEST
4186
4187

```

```

4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199
4200
4201
4202
4203
4204
4205 031074 000004
4206 031076 012706 001000
4207 031102 012737 000050 017322
4208
4209
4210 031110 012737 010000 051602
4211
4212 031116 005037 051604
4213
4214 031122 005037 051606
4215 031126 005037 051610
4216 031132 012737 177777 051612
4217 031140 004537 046360
4218 031144 051602
4219 031146 053502
4220
4221
4222
4223 031150 004737 045152
4224 031154 012777 177400 163570
4225 031162 012777 015212 163564
4226 031170 005077 163570
4227
4228 031174 012777 010000 163566
4229
4230 031202 005077 163564
4231 031206 004737 045206
4232 031212 104400 005116
4233
4234 031216 000000
4235 031220 013711 015166
4236
4237
4238
4239 031224 004037 045644
4240 031230 014752
4241 031232 015022

```

```

;*****
;*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
CLP #SECTOR ;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
MCRC
; THESE ARE REGULAR SETUPS & CHECKS
JSR PC,#CLDISK ;SETUP GENERAL REGISTERS
MOV #-256,#RHMC ;256 DATA WORDS
MOV #RFRM,#RHB ;STARTING ADDRESS OF BUFFER
CLR #RHST ;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 #MRIDAT,#R1 ;WRITE DATA = 60
;READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'
JSR R0,#SAVER ;READ REGISTERS IN SEQUENCE
RHMC ;FROM HARDWARE REGISTER
MC ;INTO CORE BUFFER LOCATION

```

E12

MAINDEC-11-DZRPT-D RJPO4 DISKLESS RH11 TEST-PART 2  
DZRPTC.TS7 T50 DRIVE TIMING ERROR

MACY11 27(663) 7-OCT-75 17:33 PAGE 104

SEQ 0146

4242 031234 000023

19.

;NUMBER OF REGISTERS TO READ

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  
4276  
4277  
4278  
4279  
4280  
4281  
4282  
4283  
4284  
4285  
4286  
4287  
4288  
4289  
4290  
4291  
4292  
4293  
4294  
4295  
4296

031236 012737 177777 015142  
031244 012737 177777 015122  
  
031252 004737 051456  
  
031256 017737 163476 001170  
031264 032737 100000 001170  
031272 001405  
031274 004737 044652  
  
031300 104040  
  
031302 000167 000260  
  
031306  
  
  
  
  
  
  
  
  
031306 012701 000030  
031312 013700 015000  
031316 012710 000001  
031322 052710 000012  
031326 042710 000012  
031332 012702 000007  
031336 052710 000002  
031342 042710 000002  
031346 005302  
031350 001372  
031352 005301  
031354 001362  
  
  
  
031356 012701 001030  
031362 052710 000010  
031366 042710 000010

```

;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.

MOV    #-1,2#TESDTE ;SET DTE TEST
MOV    #-1,2#ERFLGS ;THIS WILL BRING THE READ HEADER
                    ;AND DATA PROCESS OUT AFTER THE
                    ;HEADER HAS BEEN CORRECTLY READ

JSR    PC,2#COMHD  ;ISSUE 'GO', SEARCH FOR SECTOR,
                    ;READ HEADER AND DATA.

SETCK2: MOV    2#RHCSI,2#STMPD ;READ CSI TO CHECK FOR READ ERRORS
        BIT    #SC,2#STMPD  ;TEST FOR "SPECIAL CONDITION" - 'SC'
        BEQ    65           ;CONTINUE TESTING IF NO ERROR
        JSR    PC,2#PUTREG  ;READ & SAVE REGISTERS AGAIN IF UNDE-
                    ;FINED ERROR HAS OCCURRED
        ERROR  40           ;THERE WAS A READ/WRITE HEADER ERROR
                    ;DURING 'DTE' TEST SETUP
        JMP    TST51       ;ABORT THE TEST

65:     ;NOW THE HEADER HAS BEEN READ,
        ;560 SECTOR CLOCKS WILL BE GIVEN -
        ;GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
        ;GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 2

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

        ;THIS GIVES 24 SECTOR CLOCKS WITH DIAGNOSTIC DATA CLOCKS

MOV    #24, R1        ;LOAD SECTOR CLOCK COUNTER
MOV    2#RHMR, R0     ;GET RHMR ADDRESS
MOV    #DMD, 2#RO     ;SET DIAGNOSTIC MODE
1$:    BIS    #MSTCK!MCLK, 2#RO ;SET SECTOR CLOCK AND DATA CLOCK
        BIC    #MSTCK!MCLK, 2#RO ;CLEAR SECTOR CLOCK AND DATA CLOCK
MOV    #7, R2        ;LOAD COUNTER FOR DIAGNOSTIC DATA CLOCKS
4$:    BIS    #MCLK, 2#RO      ;SET CLOCK (DATA)
        BIC    #MCLK, 2#RO    ;CLEAR CLOCK (DATA)
        DEC    R2            ;COUNT
        BNE    4$           ;BRANCH IF 7 NOT DONE
        DEC    R1            ;COUNT
        BNE    1$           ;BRANCH IF 24 NOT DONE

        ;THIS GIVES 536 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS

MOV    #536, R1      ;LOAD SECTOR CLOCK COUNTER
5$:    BIS    #MSTCK, 2#RO   ;SET SECTOR CLOCK
        BIC    #MSTCK, 2#RO ;CLEAR SECTOR CLOCK
    
```

```

4297 031372 005301          DEC      R1          ;COUNT
4298 031374 001372          BNE      S$          ;BRANCH IF 536 NOT DONE
4299
4300
4301
4302          ;ECC PATTERN REGISTER IS NOT CHECKED
4303 031376 017737 163410 015062  MOV      @RHEC2,@#EC2 ;RHEC2 IS NOT CHECKED
4304
4305
4306          ;NOW 'DTE' SHOULD BE SET, CHANGE SAVED REGISTERS TO EXPECTED VALUES
4307
4308 031404 012737 177511 015022  MOV      #-183,@#MC   ;SAVED RHMC
4309 031412 012737 015434 015024  MOV      @#RFROM+(<73.*2),@#BA ;SAVED RHBA
4310 031420 052737 140000 015030  BIS      @#SC!TRE,@#CS1 ;SAVED RHCS1
4311 031426 042737 000100 015026  BIC      @#IR,@#CS2   ;SAVED RHCS2
4312 031434 052737 000200 015026  BIS      @#OR,@#CS2   ;SAVED RHCS2
4313 031442 052737 010000 015032  BIS      @#DTE,@#ER1  ;SAVED RHER1
4314 031450 012737 000201 015050  MOV      @#DENVL!DMD,@#MR ;SAVED RHMR
4315 031456 052737 140000 015052  BIS      @#ATA!ERR,@#DS1 ;SAVED RHDS1
4316 031464 012737 000100 015064  MOV      #100,@#LA    ;SAVED RHLA
4317 031472 012737 000001 015034  MOV      #1,@#DST    ;SAVED RHDST
4318 031500 013737 015132 015046  MOV      @#ATTENT,@#AS ;SAVED RHAS
4319
4320          ;NOW READ & SAVE REGISTERS AGAIN SO THAT COMPARISONS
4321          ;CAN BE DONE (USING 'RFROM' BUFFER THIS TIME)
4322
4323 031506 004037 045644          JSR      RO,@#SAVER   ;READ IN SEQUENCE
4324 031512 014752          RHC      ;FROM HARDWARE REGISTER
4325 031514 016256          REINTO   ;INTO CORE BUFFER LOCATION
4326 031516 000023          19.     ;NUMBER OF REGISTERS TO READ
4327
4328          ;FOR RHAS UPPER BYTE
4329 031520 113737 015047 016303  MOV8     @#AS+1,@#REINTO+25 ;UPPER RHAS
4330
4331
4332          ;COMPARE CHANGED REGISTER SNAPSHOT BEFORE COMMAND WITH
4333          ;SNAPSHOT AFTER COMMAND
4334
4335 031526 004037 046046          JSR      RO,@#COMPAR  ;COMPARE
4336 031532 015022          MC      ;CHANGED INITIAL SNAPSHOT BUFFER
4337 031534 016256          REINTO   ;SNAPSHOT BUFFER AFTER COMMAND
4338 031536 000022          18.     ;NUMBER OF REGISTERS TO COMPARE
4339 031540 031546          2$      ;RETURN FOR ERROR
4340 031542 031546          2$      ;SAME
4341 031544 031566          3$      ;RETURN FOR GOOD COMPARISON
4342
4343 031546 013705 051722          2$:     MOV      @#ERWORD,R5   ;GETTING READY TO INDEX
4344 031552 060505          ADD      R5,R5        ;DOUBLE ERROR WORD
4345 031554 016537 014750 044716  MOV      RHMC-2(R5),@#REGADR ;FAILING REGISTER
4346 031562 104001          ERROR   1           ;IMPROPER REGISTER
4347          ;CHANGE AFTER FORCING
4348          ;'DTE' ERROR
4349 031564 000207          RTS      PC         ;RETURN
4350

```

H12

MAINDEC-11-DZRPT-D, RJPO4 DISKLESS RH11 TEST-PART 2  
DZRPTC.TST TSO DRIVE TIMING ERROR

MACY11 27(663) 7-OCT-75 17:33 PAGE 107

SEQ 0149

4351 031566  
4352  
4353

3\$:

;GOOD, REGISTERS OK - GO ON TO NEXT TEST



```

4354
4355
4356
4357
4358
4359
4360
4361
4362
4363
4364
4365
4366 031566 000004
4367 031570 012706 001000
4368 031574 012737 000051 017322
4369
4370
4371
4372 031602 012737 010000 054724
4373
4374 031610 005037 054726
4375 031614 005037 054730
4376 031620 005037 054732
4377 031624 012737 000400 054764
4378 031632 004537 046360
4379 031636 054724
4380 031640 054734
4381
4382
4383
4384 031642 004737 045152
4385 031646 012777 177374 163076
4386 031654 012777 015212 163072
4387 031662 005077 163076
4388
4389 031666 012777 014000 163074
4390
4391 031674 005077 163072
4392 031700 004737 045206
4393 031704 104400 005116
4394
4395 031710 000000
4396 031712 013711 015170
4397
4398
4399
4400 031716 004037 045644
4401 031722 014752
4402 031724 015022
4403 031726 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 UP FOR DISKLESS USE ONLY
MOV #FMT22,#MCYL ;FORMAT 22=16 BITWORDS AND
;CYLINDER 0
CLR #MSECTR ;TRACK=0, SECTOR=0
CLR #MKEY1 ;KEY1=0
CLR #MKEY2 ;KEY2=0
MOV #256,#FNWORD ;256 DATAWORDS
JSR RS,#CRC ;GO TO CALCULATE CRC
MCYL
GCRC
; THESE ARE REGULAR SETUPS & CHECKS
JSR PC,#CLDISK ;SETUP GENERAL REGISTERS
MOV #-260,#RHMC ;256 DATA WORDS & 4 HEADER WORDS
MOV #RIFROM,#RHBA ;STARTING ADDRESS OF BUFFER
CLR #RHST ;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,DYR = 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 #RIFOR,#RI ;WRITE HEADER AND DATA = 62
;READ & SAVE REGISTERS FOR COMPARISON AFTER SIMULATED 'DTE'
JSR RO,#SAVER ;READ IN SEQUENCE
RHMC ;FROM HARDWARE REGISTER
MC ;INTO CORE BUFFER LOCATION
19. ;NUMBER OF REGISTERS TO READ

```

```

4404
4405 ;NOW 'GO' WILL BE GIVEN. EVERYTHING WILL BE TREATED
4406 ;NORMALLY TILL HEADER IS TO BE GIVEN, THEN ONLY
4407 ;SECTOR CLOCKS WILL BE GIVEN. NO DIAGNOSTIC DATA
4408 ;CLOCKS WILL BE GIVEN. THIS SHOULD BRING SECTOR PULSE HIGH
4409 ;WITHOUT PUTTING "READ" DOWN, HENCE 'DTE' WILL COME UP.
4410
4411 031730 012737 177777 015142 MOV #1, @#TESDTE ;SET DTE TEST
4412 031736 012737 177777 015122 MOV #1, @#ERFLGS ;THIS WILL BRING THE READ HEADER
4413 ;AND DATA PROCESS OUT AFTER THE
4414 ;HEADER HAS BEEN CORRECTLY READ
4415
4416 031744 004737 054564 JSR PC, @#COMHHD ;ISSUE 'GO', SEARCH FOR SECTOR,
4417 ;WRITE HEADER AND DATA.
4418
4419 031750 017737 163004 001170 SETCK3: MOV @RHCS1, @#STMPD ;READ CS1 TO CHECK FOR ERRORS DURING WRITE
4420 031756 032737 100000 001170 BIT #SC, @#STMPD ;TEST FOR "SPECIAL CONDITION" - 'SC'
4421 031764 001405 BEQ 45 ;CONTINUE TEST IF NO ERROR ('SC' = 0)
4422 031766 004737 044652 JSR PC, @#PUTREG ;READ & SAVE REGISTERS AGAIN IF ERROR
4423 031772 104040 ERROR 40 ;THERE WAS A READ/WRITE HEADER ERROR
4424 ;DURING 'DTE' TEST SETUP
4425 031774 000167 000202 JMP TST52 ; ABORT THE TEST AT THAT POINT
4426
4427 032000 45: ;NOW SECTOR HAS BEEN FOUND OK
4428
4429 ;609 SECTOR CLOCKS WILL BE GIVEN,
4430 ;39 BYTES FOR SECTOR GAP,
4431 ;1 BYTE FOR HEADER SYNC,
4432 ;8 BYTES FOR HEADER,
4433 ;GAP 11 BYTES, SYNC 1 BYTE, DATA 512, ECC 4 BYTES
4434 ;GAP 2 BYTES, TOLERANCE 28 BYTES, EXTRA 3
4435
4436 ;THIS GIVES 609 SECTOR CLOCKS WITHOUT DIAGNOSTIC DATA CLOCKS
4437
4438
4439 032000 012701 001141 MOV #609, R1 ;LOAD SECTOR CLOCK COUNTER
4440 032004 052710 000010 55: BIS #MSTCK, @RO ;SET SECTOR CLOCK
4441 032010 042710 000010 BIC #MSTCK, @RO ;CLEAR SECTOR CLOCK
4442 032014 005301 DEC R1 ;COUNT
4443 032016 001372 BNE 55 ;BRANCH IF 536 NOT DONE
4444
4445 ;NOW 'DTE' SHOULD BE SET, CHANGE SAVED REGISTERS
4446 ;TO EXPECTED VALUES
4447
4448 032020 012737 177477 015022 MOV #-193, @#MC ;SAVED RHMC
4449 032026 012737 015420 015024 MOV @#RFRM+(67.*2), @#BA ;SAVED RHBA
4450 032034 052737 140000 015030 BIS #SC!TRE, @#CS1 ;SAVED RHCS1
4451 032042 042737 000100 015026 BIC #IR, @#CS2 ;SAVED RHCS2
4452 032050 052737 000200 015026 BIS #OR, @#CS2 ;SAVED RHCS2
4453 032056 052737 010000 015032 BIS #DTE, @#ER1 ;SAVED RHER1
4454 032064 012737 000401 015050 MOV #401, @#MR ;SAVED RHMR
4455 032072 052737 140000 015052 BIS #ATA!ERR, @#DS1 ;SAVED RHDS1
4456 032100 012737 000100 015064 MOV #100, @#LA ;SAVED RHLA
4457 032106 012737 000001 015034 MOV #1, @#DST ;SAVED RHDST

```



```

4492 ;*****
4493 ;*TEST 52 SECTOR SELECTION
4494
4495 ;* THE SECTOR SELECTION LOGIC IS CHECKED HERE
4496 ;* EACH SECTOR ON TRACK ZERO IS WRITTEN INTO.
4497
4498 ;* DATA IS - 19 WORDS OF ZEROS - SYNC WORDS, 4 HEADER WORDS
4499 ;* 1 CRC WORD, 5 WORDS OF ZEROS, 1 SYNC WORD, 100 ZEROS
4500 ;* (DATA), 1 SYNC WORD, 70 SECTOR NUMBER TO VARY
4501 ;*
4502 ;* THE WRITTEN DATA IS CHECKED IN MEMORY
4503
4504 ;*****
4505 TST52: SCOPE
4506 032202 000004 MOV #STACK,SP ;RESET STACK
4507 032204 012706 001000 MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
4508 032210 012737 000052 017322
4509 032216 012737 000026 015144 MOV #22.,@#TAGDTE ;22 SECTORS
4510 ;THIS TEST REPEATS
4511 ;ITSELF 22 TIMES
4512
4513 ;THE FOLLOWING INITIALIZES FOR SECTOR 0
4514
4515 032224 005037 032334 CLR @#SS3+2 ;HEADER (SECTOR)
4516 032230 012737 000025 032340 MOV #21.,@#SS4+2 ;HEADER (KEY1)
4517 032236 012737 000025 032344 MOV #21.,@#SS5+2 ;HEADER (KEY2)
4518 032244 005037 032372 CLR @#SS7+2 ;DATA (SECTOR)
4519 032250 005037 032454 CLR @#SS10+2 ;DATA
4520 032254 005037 032504 CLR @#SS12+2 ;SECTOR (SIMULATED DISK)
4521 032260 012737 000025 032512 MOV #21.,@#SS13+2 ;KEY1 (SIMULATED DISK)
4522 032266 012737 000025 032520 MOV #21.,@#SS14+2 ;KEY2 (SIMULATED DISK)
4523 032274 005037 032560 CLR @#SS15+2 ;SECTOR (RHDST)
4524
4525 ;CLEAR SIMULATED DISK AREA
4526 032300 SS1:
4527 032300 012700 053422 1$: MOV #SECGAP,RO ;POINTER
4528 032304 012701 000460 MOV #304.,R1 ;COUNTER
4529 032310 005020 2$: CLR (RO)+ ;CLEAR SIMULATED DISK AREA
4530 032312 005301 DEC R1 ;COUNT
4531 032314 001375 BNE 2$
4532
4533 ;SETUP GENERAL REGISTERS
4534 032316 004737 045152 JSR PC,@#CLDISK
4535
4536 ;SETUP WRITE FROM BUFFER
4537 032322 012700 015212 MOV #WRFROM,RO
4538
4539 ;HEADER
4540 032326 012720 010000 MOV #FMT22,(RO)+ ;FORMAT 16 BITS PER WORD
4541 ;CYLINDER 0
4542 032332 012720 000000 SS3: MOV #0,(RO)+ ;SECTOR TO VARY
4543 032336 012720 000025 SS4: MOV #21.,(RO)+ ;KEY1 TO VARY
4544 032342 012720 000025 SS5: MOV #21.,(RO)+ ;KEY2 TO VARY
4545

```

```

4546                                     ;DATA IN WRITE FROM BUFFER ALTHOUGH THIS IS DATA AND NOT
4547                                     ;HEADER, THE SECTOR WITH SYNC BYTES WILL BE GIVEN AS DATA.
4548
4549                                     ;DATA IS - 19 WORDS OF ZEROS - SYNC WORDS, 4 HEADER WORDS
4550                                     ;1 CRC WORD, 5 WORDS OF ZEROS, 1 SYNC WORD, 100 ZEROS
4551                                     ;(DATA), 1 SYNC WORD, 70 SECTOR NUMBER TO VARY
4552
4553 032346 012705 000023                6$: MOV    #19, R5          ;COUNTER
4554 032352 005020                    CLR    (R0)+          ;19 ZEROS
4555 032354 005305                    DEC    R5             ;COUNT
4556 032356 001375                    BNE   6$             ;19 DONE?
4557 032360 013720 051704              MOV    @#RSYNC, (R0)+ ;SYNC = 14400
4558 032364 012720 010000              MOV    #FMT22, (R0)+ ;CYLINDER 0
4559 032370 012720 000000              SS7:  MOV    #0, (R0)+ ;SECTOR TO VARY
4560 032374 005020                    CLR    (R0)+
4561 032376 005020                    CLR    (R0)+
4562 032400 004537 046360              JSR   R5, @#CRC      ;CALCULATE CRC FOR ABOVE 4 WORDS
4563 032404 015262                    MRFROM+50            ;4 WORDS START FROM HERE
4564 032406 015272                    MRFROM+60            ;PUT CRC HERE
4565
4566 032410 005720                    TST   (R0)+          ;INCREMENT R0
4567
4568 032412 012705 000005                8$: MOV    #5, R5          ;5 WORDS OF ZEROS
4569 032416 005020                    CLR    (R0)+          ;COUNT
4570 032420 005305                    DEC    R5             ;BRANCH IF 5 NOT DONE
4571 032422 001375                    BNE   8$
4572
4573 032424 013720 051704              MOV    @#RSYNC, (R0)+ ;SYNC = 14400
4574
4575 032430 012705 000144                9$: MOV    #100, R5       ;100 WORDS OF ZEROS
4576 032434 005020                    CLR    (R0)+
4577 032436 005305                    DEC    R5
4578 032440 001375                    BNE   9$
4579
4580 032442 013720 051704              MOV    @#RSYNC, (R0)+ ;SYNC = 14400
4581 032446 012705 000106              MOV    #70, R5
4582 032452 012720 000000              SS10: MOV    #0, (R0)+ ;SECTOR TO VARY
4583 032456 005305                    DEC    R5
4584 032460 001374                    BNE   SS10
4585
4586                                     ;CLEAR REST OF 256 WORDS THAT IS 54 WORDS OF ZEROS
4587
4588 032462 012705 000066                11$: MOV    #54, R5
4589 032466 005020                    CLR    (R0)+
4590 032470 005305                    DEC    R5
4591 032472 001375                    BNE   11$
4592
4593                                     ;THESE ARE TO BE SET UP FOR DISKLESS USE ONLY
4594
4595 032474 012737 010000 054724          MOV    #FMT22, @#WCVL ;FORMAT = 16 BIT WORDS
4596                                     ;CYLINDER = 0
4597 032502 012737 000000 054726          SS12: MOV    #0, @#WSECTR ;SECTOR TO VARY
4598 032510 012737 000025 054730          SS13: MOV    #21, @#WKEY1 ;KEY1 TO VARY
4599 032516 012737 000025 054732          SS14: MOV    #21, @#WKEY2 ;KEY2 TO VARY

```

```

4600 032524 012737 000312 054764  MOV    #202, @#FNWORD    ;202 DATA WORDS
4601 032532 004537 046360          JSR    RS, @#CRC        ;CALCULATE CRC
4602 032536 054724          MCVL           ;FIRST WORD
4603 032540 054734          GCRC           ;PUT HERE
4604
4605                                     ; THESE ARE REGULAR SETUPS
4606
4607 032542 012777 177400 162202  MOV    #-256, @RHMC     ;202 DATA, 4 HEADER
4608 032550 012777 015212 162176  MOV    @RFROM, @RHBA    ;FILL BUS ADDRESS
4609 032556 012777 000000 162200  SS15:  MOV    #0, @RHDS        ;SECTOR TO VARY
4610 032564 013777 015170 162166  MOV    @#WRIFOR, @RHCS1 ;GET READY TO DO
4611                                     ;WRITE HEADER AND DATA
4612                                     ;WITH 62 FUNCTION CODE IN RHCS1
4613 032572 012777 010600 162170  MOV    #FMT22, @RHOF    ;16 BITS PER WORD FORMAT
4614 032600 005077 162166          CLR    @RHCA           ;CYLINDER = 0
4615
4616 032604 005037 015122          CLR    @#ERFLGS        ;CLEAR ERROR FLAG
4617
4618 032610 004737 045206          JSR    PC, @#CHECKT     ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
4619 032614 104400 005116          TYPE   , CPHALT        ;CANNOT CONTINUE TESTING IF ANY OF THE
4620                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
4621 032620 000000          HALT                   ;STOP THE TEST
4622
4623 032622 004737 054564          JSR    PC, @#COMMD      ;ISSUE 'GO', SEARCH FOR SECTOR,
4624                                     ;WRITE HEADER AND DATA
4625 032626 005737 015122          TST    @#ERFLGS        ;HAVE ANY ERRORS OCCURRED ?
4626 032632 001051          BNE    TST53           ; BRANCH IF YES*****
4627
4628 032634 004737 045376          JSR    PC, @#CHECKE     ;CHECK THAT DVA, RDY, DPR, DRY = 1
4629 032640 104400 005116          TYPE   , CPHALT        ;CANNOT CONTINUE IF THEY DON'T
4630 032644 000000          HALT                   ;STOP THE TEST AND RESTART PROGRAM
4631
4632                                     ;NON COMPARE "DISK" BUFFER WITH "REINTO" BUFFER
4633 032646 004037 046046          JSR    RO, @#COMPAR     ;CHECK
4634 032652 015222          MRFROM+8.             ;GOOD BUFFER
4635 032654 053520          DISK                  ;TEST BUFFER
4636 032656 000400          256.                  ;NUMBER OF WORDS
4637 032660 032666          16$                   ;RETURN POINT FOR ERROR HEADER
4638 032662 032672          17$                   ;RETURN POINT FOR ERROR DATA
4639 032664 032676          18$                   ;RETURN FOR GOOD COMPARISON
4640 032666 104007          16$: ERROR 7
4641 032670 000207          RTS    PC
4642 032672 104010          17$: ERROR 10
4643 032674 000207          RTS    PC
4644
4645                                     ;THE FOLLOWING INCREMENTS ARE TO CHANGE THE ABOVE SET UP
4646                                     ;TO WRITE ON THE NEXT SECTOR
4647
4648 032676 005237 032334          18$: INC    @#SS3+2     ;HEADER (SECTOR)
4649 032702 005337 032340          DEC    @#SS4+2         ;HEADER (KEY1)
4650 032706 005337 032344          DEC    @#SS5+2         ;HEADER (KEY2)
4651 032712 005237 032372          INC    @#SS7+2         ;DATA (SECTOR)
4652 032716 005237 032454          INC    @#SS10+2        ;DATA
4653 032722 005237 032504          INC    @#SS12+2        ;SECTOR (SIMULATED DISK)

```

*Handwritten signature or mark*

4654	032726	005337	032512		DEC	@#SS13+2	;KEY1 (SIMULATED DISK)
4655	032732	005337	032520		DEC	@#SS14+2	;KEY2 (SIMULATED DISK)
4656	032736	005237	032560		INC	@#SS15+2	;SECTOR (RH0ST)
4657							
4658	032742	005337	015144	SS2:	DEC	@#TAGDTE	;COUNT DOWN FOR 22 SECTORS
4659	032746	001001			BNE	IS	;BRANCH IF 22 SECTORS NOT NONE
4660	032750	000402			BR	TST53	;ALL DONE - GO TO NEXT TEST
4661	032752	000137	032300	IS:	JMP	@#SS1	;GO BACK TO NEXT SECTOR
4662							
4663							
4664							

.SBTTL DATA TRANSFER TESTS USING ECC

4665  
4666  
4667  
4668  
4669  
4670  
4671  
4672  
4673  
4674  
4675  
4676  
4677  
4678  
4679  
4680  
4681  
4682  
4683  
4684  
4685  
4686  
4687  
4688  
4689  
4690  
4691  
4692  
4693  
4694  
4695  
4696  
4697  
4698  
4699  
4700  
4701  
4702  
4703  
4704  
4705  
4706  
4707  
4708  
4709  
4710  
4711  
4712  
4713  
4714  
4715  
4716  
4717  
4718

032756 000004  
032760 012706 001000  
032764 012737 000053 017322  
032772 012700 053422  
032776 012701 000402  
033002 012720 177777  
033006 005301  
033010 001374  
033012 004767 012134  
  
033016 012737 177777 015140  
033024 005037 047660  
033030 013737 047654 047656  
033036 013737 047662 047670  
033044 005037 047646  
033050 005037 047650  
033054 005037 047664  
033060 005037 047666  
  
033064 012737 010000 054724  
033072 012737 000001 054726  
033100 005037 054730  
033104 005037 054732  
033110 012737 000400 054764  
033116 004537 046360  
033122 054724  
033124 054734  
  
033126 012777 177374 161616  
033134 012700 015212

```
*****  
;*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.  
  
*****  
†ST53: SCOPE  
MOV #STACK, SP ;RESET STACK  
  
MOV #TTNO, @TSTNM ;THIS SAVES TEST NUMBER  
MOV #SECCAP, R0 ;POINTER  
MOV #256, R1 ;COUNTER  
15: MOV #-1, (R0)+ ;FILL SIMULATOR DISK WITH ONES  
DEC R1  
BNE 15  
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 @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, @MNCYL ;FORMAT22=16BIT WORDS AND  
;CYLINDER 0  
MOV #1, @MSECTR ;TRACK=0, SECTOR=1  
CLR @KEY1 ;KEY1=0  
CLR @KEY2 ;KEY2=0  
MOV #256, @FNWORD ;256 DATA WORDS  
JSR R5, @CRC ;GO TO CALCULATE CRC  
MNCYL  
GCRC  
  
;THESE ARE REGULAR SETUPS  
  
MOV #-260, @RHMC ;256 DATA WORDS 4 HEADER WORDS  
MOV #MRFROM, R0 ;THESE TWO INSTRUCTIONS GETS
```



```

4719 033140 010077 161610      MOV      RO,ARH2A      ;ADDR. OF WRFROM INTO RO AND
4720                                ;BUS ADDRESS REGISTER
4721 033144 012720 010000      MOV      #FMT22,(RO)+ ;FORMAT=16 BIT WORDS
4722                                ;CYLINDER=0
4723 033150 012720 000001      2$: MOV    #1,(RO)+    ;TRACK=0, SECTOR=1, KEYS=0
4724 033154 005020              CLR      (RO)+        ;KEY1=0
4725 033156 005020              CLR      (RO)+        ;KEY2=0
4726 033160 012705 000400      MOV      #256,R5     ;COUNTER
4727 033164 012720 000000      3$: MOV    #0,(RO)+    ;MOVE ALL ZEROS FOR DATA
4728 033170 005305              DEC      R5           ;
4729 033172 001374              BNE     3$           ;BRANCH IF DATA NOT COMPLETE
4730 033174 012777 000001 161562 MOV      #1,ARH0ST    ;TRACK=0 SECTOR=1
4731
4732 033202 004737 045206      JSR     PC,ARCHECKT  ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
4733 033206 104400 005116      TYPE   ,CPHALT      ;CANNOT CONTINUE TESTING IF ANY OF THE
4734                                ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
4735 033212 000000              HALT                ;STOP THE TEST
4736
4737 033214 013711 015170      MOV      ARIFOR,ARI  ;GET READY FOR WRITE HEADER AND
4738                                ;DATA WITH 62 IN RHCS1
4739 033220 005037 015122      CLR     ARERFLGS     ;CLEAR ERROR FLAG
4740 033224 012777 010000 161536 MOV      #FMT22,ARHOF ;FORMAT BIT=1 (16 BIT WORDS)
4741 033232 005077 161534      CLR     ARHCA        ;CYLINDER =0
4742 033236 004737 054564      JSR     PC,ARCOMAND  ;WRITE HEADER AND DATA
4743
4744                                ; IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
4745                                ; FROM THE "COMAND" ROUTINE THAT MEANS ALL HEADER ON DISK
4746                                ; IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
4747                                ; ALL ZEROS AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
4748                                ; THEY ARE ALL ZEROS
4749
4750 033242 005737 015122      TST     ARERFLGS     ;HAS ANY ERRORS OCCURED?
4751                                ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
4752 033246 001056              BNE     TST54       ;;BRANCH IF YES
4753
4754                                ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
4755
4756
4757 033250 023737 047646 054520  CMP     ARGECC1,ARECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
4758 033256 001402              BEQ     6$          ;BRANCH IF GOOD
4759 033260 104031              ERROR   31         ;LOW ORDER ECC IN ERROR
4760 033262 000405              BR      7$          ;BRANCH TO CONTINUE
4761 033264 023737 047650 054522 6$: CMP     ARGECC2,ARECC2;COMPARE SOFTWARE ECC WITH HARDWARE ECC
4762 033272 001401              BEQ     7$          ;BRANCH IF GOOD
4763 033274 104031              ERROR   31         ;HIGH ORDER ECC IN ERROR
4764
4765
4766                                7$:
4767 033276 004737 045376      JSR     PC,ARCHECKE ;CHECK THAT DVA, RDY, DPR, DRY = 1
4768 033302 104400 005116      TYPE   ,CPHALT      ;CANNOT CONTINUE IF THEY DON'T
4769 033306 000000              HALT                ;STOP THE TEST AND RESTART PROGRAM
4770
4771
4772

```

```

4773 ;FILL "REINTO" BUFFER WITH EXPECTED DATA
4774
4775 033310 004037 045070 JSR RD,@#CLAREA ;FILL REINTO BUFFER
4776 033314 016256 REINTO ;FROM
4777 033316 017254 REINTO+(255.*2) ;TO
4778 033320 000000 .WORD 0 ;DATA
4779
4780 033322 013737 047646 017256 MOV @#GECC1,@#REINTO+(256.*2);FILL ECC1
4781 033330 013737 047650 017260 MOV @#GECC2,@#REINTO+(257.*2);FILL ECC2
4782 033336 004037 045070 JSR RD,@#CLAREA ;FILL REST
4783 033342 017262 REINTO+(258.*2) ;FROM
4784 033344 017316 REINTO+(272.*2) ;TO
4785 033346 000000 0 ;DATA
4786
4787
4788 033350 005037 015122 CLR @#ERFLG$ ;CLEAR ERROR FLAG
4789
4790
4791 ;NOW COMPARE "DISK" BUFFER WITH "REINTO"
4792
4793 033354 004037 046046 JSR RD,@#COMPAR ;CHECK
4794 033360 016256 REINTO ;GOOD BUFFER
4795 033362 053520 DISK ;TEST BUFFER
4796 033364 000402 258. ;NUMBER OF WORDS CHECKED
4797 033366 033374 4$ ;RETURN POINT FOR ERROR HEADER
4798 033370 033400 5$ ;RETURN POINT FOR ERROR DATA
4799
4800 033372 033404 TST54 ;RETURN FOR GOOD COMPARISON
4801
4802 033374 104007 4$: ERROR 7 ;READ ERROR 10 NEXT
4803 033376 000207 RTS PC ;RETURN TO COMPARE
4804 033400 104010 5$: ERROR 10 ;WORD NOS 1 TO 256 ARE
4805 ;DATA WORDS
4806 ;WORD NOS 257 AND 258
4807 ;ARE ECC WHICH ARE CHECKED
4808 ;WORD NOS 259
4809 ;IS DATA GAP
4810 ;WORD NOS 260 TO 273
4811 ;ARE TOLERANCE GAP
4812 033402 000207 RTS PC ;RETURN TO COMPARE
4813
4814
4815
4816
4817

```

```

4818
4819 ;*****
4820 ;*TEST 54 READ ECC ENABLED 1A
4821
4822 ;* THIS IS AN ECC READ DATA TEST
4823 ;* ERROR CORRECTION IS ENABLED
4824 ;* NO ERROR IS INSERTED
4825 ;* GOOD DATA USED IS 256 WORDS OF 0
4826 ;* COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
4827 ;* TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA
4828
4829 ;*****
4830 033404 000004          †ST54: SCOPE
4831 033406 012706 001000      MOV      #STACK,SP      ;RESET STACK
4832
4833 033412 012737 000054 017322  MOV      #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
4834
4835
4836 ;
4837 ; SETUP FOR WHAT IS TO BE READ
4838 ; HEADER CRC IS RESTORED FROM A SUBROUTINE
4839 033420 012746 000000      MOV      #0, -(SP)      ;DATA TO BE READ
4840 033424 012705 000400      MOV      #256, R5      ;COUNTER
4841 033430 012700 053520      MOV      #DISK, R0     ;START OF SIMULATED DISK DATA
4842 033434 011620          1$: MOV      (SP), (R0)+    ;MOVE IN DATA ON TO SIMULATED DISK
4843 033436 005305          DEC      R5            ;COUNT
4844 033440 001375          BNE     1$            ;BRANCH IF 256 NOT COMPLETE
4845 033442 005726          TST     (SP)+         ;UNDO -(SP)
4846 033444 022020          CMP     (R0)+, (R0)+  ;JUMP OVER THE TWO ECC WORDS
4847 033446 012705 000017      MOV      #15, R5      ;1 DATA GAP
4848
4849 033452 005020          2$: CLR      (R0)+         ;14 TOLERANCE GAP
4850 033454 005305          DEC     R5            ;CLEAR DATA GAP, AND
4851 033456 001375          BNE     2$            ;TOLERANCE GAP
4852
4853
4854 033460 004737 050442      JSR     PC,@#FILLEC   ;INSERT THE TWO ECC WORDS ON THE DISK
4855
4856
4857
4858 ;THESE ARE FOR ECC TEST ONLY
4859 033464 012737 177777 015140  MOV     #-1,@#TSECC   ;THIS IS AN ECC TEST
4860 033472 005037 047660          CLR     @#POSITI     ;CLEAR ERROR POSITION COUNTER
4861 033476 013737 047654 047656  MOV     @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
4862 033504 013737 047662 047670  MOV     @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
4863 033512 005037 047646          CLR     @#GECC1     ;ECC LOW ORDER TO BE GENERATED
4864 033516 005037 047650          CLR     @#GECC2     ;ECC HIGH ORDER TO BE GENERATED
4865 033522 005037 047664          CLR     @#DATENV    ;CLEAR DATA ENVELOPE CLOCK COUNT
4866 033526 005037 047666          CLR     @#ZCODE     ;CLEAR LEADING ZEROS CLOCK COUNT
4867
4868
4869 ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
4870
4871 033532 012737 010006 051602  MOV     #FMT22,@#CYL ;16 BITS PER WORD

```

```

4872                                     ;CYLINDER 0, FORMAT 16 BITS
4873 033540 112737 000000 051605      MOVB  #0,@#SECOTR+1 ;TRACK 0
4874 033546 112737 000000 051604      MOVB  #0,@#SECOTR  ;SECTOR 0
4875 033554 012737 000000 051606      MOV   #0,@#KEY1   ;KEY1=0
4876 033562 012737 000000 051610      MOV   #0,@#KEY2   ;KEY2=0
4877 033570 012737 000400 051662      MOV   #256., @#DANORD ;NO. OF DATA WORDS
4878 033576 005037 051612                CLR   @#X          ;THIS IS A READ COMMAND
4879 033602 004537 046360                JSR   RS,@#CRC     ;GO TO CALCULATE CRC
4880 033606 051602
4881 033610 053502
4882
4883
4884
4885
4886                                     ;THESE ARE REGULAR SETUPS
4887
4888 033612 004737 045152                JSR   PC,@#CLDISK ;SETUP GENERAL REGISTERS
4889 033616 012777 177374 161126      MOV   #-256.-4,@#RHMC ;256. DATA 4 HEADER WORDS
4890 033624 012777 016256 161122      MOV   #REINTO,@#RHA ;STARTING ADDRESS OF READ BUFFER
4891 033632 112746 000000                MOVB  #0,-(SP)     ;IN LOWER BYTE GET SECTOR
4892 033636 112766 000000 000001      MOVB  #0,1(SP)    ;GET TRACK IN HIGHER BYTE
4893 033644 012677 161114                MOV   (SP)+,@#RHST ;TRACK/SECTOR IN RHST
4894 033650 012777 010000 161112      MOV   #FMT22,@#RHOF ;16 BITS PER WORD
4895                                     ;ECC CORRECTION NOT INHIBIT
4896                                     ;BECAUSE ECC IS NOT GOING
4897                                     ;TO BE CHECKED
4898 033656 005077 161110                CLR   @#RHCA      ;CYLINDER 0
4899
4900 033662 004737 045206                JSR   PC,@#CHECKT ;CHECK DVA,ROY,DPR,DRY = 1 AND OTHERS DON'T
4901 033666 104400 005116                TYPE  ,CPHALT    ;CANNOT CONTINUE TESTING IF ANY OF THE
4902                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
4903 033672 000000                HALT             ;STOP THE TEST
4904
4905 033674 013711 015174                MOV   @#REFOR,@#R1 ;READ HEADER AND DATA=72
4906 033700 005037 015122                CLR   @#ERFLG$   ;CLEAR ERROR FLAG
4907 033704 004737 051456                JSR   PC,@#COMHD  ;READ HEADER AND DATA
4908                                     ;IF THERE ARE READ ERRORS THEN
4909                                     ;ECC WILL NOT BE CHECKED
4910
4911
4912                                     ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
4913                                     ;FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
4914                                     ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
4915                                     ;SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
4916                                     ;DETECTED
4917                                     ;HEADER AND DATA ARE TO BE CHECKED.
4918                                     ;IN CHECKING READ DATA THE WRITE FROM BUFFER
4919                                     ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
4920                                     ;COMPARISONS ARE MADE
4921
4922 033710 005737 015122                TST   @#ERFLG$   ;ANY ERRORS ALREADY THERE
4923 033714 001102                BNE   TST55      ;BRANCH IF YES
4924 033716 004737 044652                JSR   PC,@#PUTREG ;SAVE REGISTERS
4925 033722 005737 015032                TST   @#ERI      ;NO ERRORS SHOULD BE SET

```

```

4926 033726 001401          BEQ      6$          ; BRANCH IF NO ERRORS SET
4927 033730 104032          ERROR    32          ; 32 BIT ECC REGISTER SHOULD BE ZERO
4928                                     ; ONLY 11 OF THE 32 BITS CAN BE SEEN
4929                                     ; IN THE PATERN REGISTER
4930                                     ; DCK SHOULD BE SET IN RHER1
4931 033732 013746 047646    6$:   MOV      @#GECC1, -(SP) ; GET PATTERN REGISTER
4932 033736 042716 174000    BIC      #174000, (SP) ; KEEP ONLY 11 BITS
4933 033742 022637 015062    CMP      (SP)+, @#EC2 ; COMPARE PATTERN REGISTER
4934 033746 001401          BEQ      7$          ; BRANCH IF GOOD
4935 033750 104032          ERROR    32          ; 11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
4936
4937
4938
4939
4940                                     ; ADD 16 MAINTENANCE CLOCKS TO
4941                                     ; BRING EBL DOWN
4942
4943 033752 012700 000020    7$:   MOV      #16., R0      ; COUNTER
4944 033756 052777 000002 161014 8$:   BIS      #MCLK, @RHMR ; SET CLOCK
4945 033764 042777 000002 161006    BIC      #MCLK, @RHMR ; CLEAR CLOCK
4946 033772 005300          DEC      R0          ; COUNT
4947 033774 001370          BNE      8$          ; BRANCH IF 16 CLOCKS NOT DONE
4948 033776 004737 045376    JSR      PC, @#CHECKE ; CHECK THAT DVA, RDY, DPR, DRY = 1
4949 034002 104400 005116    TYPE    , CPHALT    ; CANNOT CONTINUE IF THEY DON'T
4950 034006 000000          HALT                    ; STOP THE TEST AND RESTART PROGRAM
4951 034010 012700 015212    MOV      #MRFROM, R0 ; GETTING READY TO FILL EXPECTED DATA
4952 034014 012720 010000    MOV      #0!FMT22, (R0)+ ; CYLINDER 0
4953 034020 112746 000000    MOV      #0, -(SP)    ; IN LOWER BYTE GET SECTOR
4954 034024 112766 000000 000001    MOV      #0, 1(SP)   ; GET TRACK IN HIGHER BYTE
4955 034032 012620          MOV      (SP)+, (R0)+ ; GET TRACK/SECTOR IN BUFFER
4956 034034 012720 000000    MOV      #0, (R0)+   ; KEY1 IN BUFFER
4957 034040 012720 000000    MOV      #0, (R0)+   ; KEY2 IN BUFFER
4958 034044 012701 000400    MOV      #256., R1   ; DATA WORD COUNTER
4959 034050 012702 000000    MOV      #0, R2     ; DATA
4960 034054 010220          3$:   MOV      R2, (R0)+   ; DATA INTO BUFFER
4961 034056 005301          DEC      R1          ; COUNT
4962 034060 001375          BNE      3$          ; BRANCH IF 256 NOT DONE
4963
4964
4965 034062 005037 015122    CLR      @#ERFLG$    ; CLEAR ERROR FLAG
4966 034066 004737 044652    JSR      PC, @#PUTREG ; SAVE REGISTERS
4967
4968
4969
4970                                     ; NOW READ DATA BUFFER WILL BE CHECKED
4971
4972 034072 004037 046046    JSR      R0, @#COMPAR ; CHECK
4973 034076 015212          MRFROM ; GOOD BUFFER
4974 034100 016256          REINTO ; TEST BUFFER
4975 034102 000404          4+256. ; NUMBER OF WORDS CHECKED
4976 034104 034112          4$      ; RETURN POINT FOR ERROR HEADER
4977 034106 034116          5$      ; RETURN POINT FOR ERROR DATA
4978
4979 034110 034122          TST55 ; RETURN FOR GOOD COMPARISON

```

4980						
4981	034112	104004	4\$:	ERROR	4	: READ NEXT ERROR
4982	034114	000207		RTS	PC	: RETURN TO "COMPAR"
4983	034116	104005	5\$:	ERROR	5	: WORD NOS 1 TO 4 ARE
4984						: HEADER WORDS
4985						: 5 TO 260 ARE DATA WORDS
4986	034120	000207		RTS	PC	: RETURN TO "COMPAR"
4987						
4988						
4989						
4990						

4991  
4992  
4993  
4994  
4995  
4996  
4997  
4998  
4999  
5000  
5001  
5002  
5003 034122 000004  
5004 034124 012706 001000  
5005  
5006 034130 012737 000055 017322  
5007  
5008  
5009  
5010  
5011  
5012 034136 012746 000000  
5013 034142 012705 000400  
5014 034146 012700 053520  
5015 034152 011620  
5016 034154 005305  
5017 034156 001375  
5018 034160 005726  
5019 034162 022020  
5020 034164 012705 000017  
5021  
5022 034170 005020  
5023 034172 005305  
5024 034174 001375  
5025  
5026  
5027 034176 004737 050442  
5028  
5029  
5030  
5031  
5032  
5033 034202 012737 177777 015140  
5034 034210 005037 047660  
5035 034214 013737 047654 047656  
5036 034222 013737 047662 047670  
5037 034230 005037 047646  
5038 034234 005037 047650  
5039 034240 005037 047664  
5040 034244 005037 047666  
5041  
5042  
5043  
5044

\*\*\*\*\*  
\*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

\*\*\*\*\*

TST55: 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  
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 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

```

5045 034250 012737 010000 051602      MOV      #FMT22, @#CYL      ;16 BITS PER WORD
5046                                     ;CYLINDER 0, FORMAT 16 BITS
5047 034256 112737 000000 051605      MOVB     #0, @#SECOTR+1    ;TRACK 0
5048 034264 112737 000000 051604      MOVB     #0, @#SECOTR     ;SECTOR 0
5049 034272 012737 000000 051606      MOV      #0, @#KEY1       ;KEY1=0
5050 034300 012737 000000 051610      MOV      #0, @#KEY2       ;KEY2=0
5051 034306 012737 000400 051662      MOV      #256., @#DAMWORD ;NO. OF DATA WORDS
5052 034314 005037 051612              CLR      @#X              ;THIS IS A READ COMMAND
5053 034320 004537 046360              JSR      R5, @#CRC        ;GO TO CALCULATE CRC
5054 034324 051602
5055 034326 053502
5056
5057
5058                                     ;THIS IS TO INSERT ERROR
5059                                     ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
5060                                     ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
5061                                     ;THIS MOVE
5062
5063 034330 012737 100000 053522      MOV      #100000, @#DISK+2 ;FORCE ERROR ON BIT NUMBER 32
5064                                     ;50 ERROR POSITION REGISTER WILL SHOW
5065                                     ;22
5066 034336 012737 000026 034512      MOV      #22., @#85       ;INSERT POSITION REG.
5067
5068
5069                                     ;THESE ARE REGULAR SETUPS
5070
5071 034344 004737 045152              JSR      PC, @#CLDISK     ;SETUP GENERAL REGISTERS
5072 034350 012777 177374 160374      MOV      #-256.-4, @#HMC  ;256. DATA 4 HEADER WORDS
5073 034356 012777 016256 160370      MOV      #REINTO, @#HBA   ;STARTING ADDRESS OF READ BUFFER
5074 034364 112746 000000              MOVB     #0, -(SP)        ;IN LOWER BYTE GET SECTOR
5075 034370 112766 000000 000001      MOVB     #0, 1(SP)       ;GET TRACK IN HIGHER BYTE
5076 034376 012677 160362              MOV      (SP)+, @#RHST    ;TRACK/SECTOR IN RHST
5077 034402 012777 010000 160360      MOV      #FMT22, @#RHOF  ;16 BITS PER WORD
5078                                     ;ECC CORRECTION NOT INHIBIT
5079                                     ;BECAUSE ECC IS NOT GOING
5080                                     ;TO BE CHECKED
5081 034410 005077 160356              CLR      @#RHCA          ;CYLINDER C
5082
5083 034414 004737 045206              JSR      PC, @#CHECKT     ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
5084 034420 104400 005116              TYPE     , CPHALT        ;CANNOT CONTINUE TESTING IF ANY OF THE
5085                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5086 034424 000000
5087                                     ;STOP THE TEST
5088 034426 013711 015174              MOV      @#REFOR, @#RI    ;READ HEADER AND DATA=72
5089 034432 005037 015122              CLR      @#ERFLG$        ;CLEAR ERROR FLAG
5090 034436 004737 051456              JSR      PC, @#COMHD     ;READ HEADER AND DATA
5091                                     ;IF THERE ARE READ ERRORS THEN
5092                                     ;ECC WILL NOT BE CHECKED
5093
5094
5095                                     ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5096                                     ;FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
5097                                     ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5098                                     ;SYNC BYTE HAVE GCNE BY AND SYNC'S WERE CORRECTLY

```



```

5099          ;DETECTED
5100          ;HEADER AND DATA ARE TO BE CHECKED.
5101          ;IN CHECKING READ DATA THE WRITE FROM BUFFER
5102          ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
5103          ;COMPARISONS ARE MADE
5104
5105 034442 005737 015122          TST      2#ERFLGS          ;ANY ERRORS ALREADY THERE
5106 034446 001077                BNE      TST56           ;BRANCH IF YES
5107 034450 004737 044652          JSR      PC,2#PUTREG     ;SAVE REGISTERS
5108 034454 022737 100000 015032  CMP      #DCK,2#ER1     ;ONLY DATA CHECK ERROR SHOULD BE SET
5109 034462 001401                BEQ      65             ;BRANCH IF YES
5110 034464 104032                ERROR   32             ;32 BIT ECC REGISTER SHOULD BE NON
5111                                     ZERO
5112                                     ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5113                                     ;IN THE PATERN REGISTER
5114                                     ;DCK SHOULD BE SET IN RHER1
5115 034466 013746 047646          65:     MOV      2#GECC1,-(SP) ;GET PATTERN REGISTER
5116 034472 042716 174000          BIC      #174000,(SP)   ;KEEP ONLY 11 BITS
5117 034476 022637 015062          CMP      (SP)+,2#EC2   ;COMPARE PATTERN REGISTER
5118 034502 001401                BEQ      75             ;BRANCH IF GOOD
5119 034504 104032                ERROR   32             ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5120
5121 034506 004037 050270          75:     JSR      RD,2#ECORR ;GO TO ECC CORRECTION PROCESS
5122 034512 000026                85:     22.             ;EXPECTED POSITION REG. WHEN CORRECTION
5123                                     ;IS COMPLETE
5124
5125
5126
5127 034514 004737 045376          JSR      PC,2#CHECKE    ;CHECK THAT DVA,RDY,DPR,DRY = 1
5128 034520 104400 005116          TYPE    ,CPHALT       ;CANNOT CONTINUE IF THEY DON'T
5129 034524 000000                HALT                    ;STOP THE TEST AND RESTART PROGRAM
5130 034526 012700 015212          MOV      #WRFROM,R0    ;GETTING READY TO FILL EXPECTED DATA
5131 034532 012720 010000          MOV      #0!FMT22,(R0)+ ;CYLINDER 0
5132 034536 112746 000000          MOVVB   #0,-(SP)       ;IN LOWER BYTE GET SECTOR
5133 034542 112766 000000 000001  MOVVB   #0,1(SP)       ;GET TRACK IN HIGHER BYTE
5134 034550 012620                MOV      (SP)+,(R0)+   ;GET TRACK/SECTOR IN BUFFER
5135 034552 012720 000000          MOV      #0,(R0)+     ;KEY1 IN BUFFER
5136 034556 012720 000000          MOV      #0,(R0)+     ;KEY2 IN BUFFER
5137 034562 012701 000400          MOV      #256.,R1     ;DATA WORD COUNTER
5138 034566 012702 000000          MOV      #0,R2        ;DATA
5139 034572 010220                35:     MOV      R2,(R0)+   ;DATA INTO BUFFER
5140 034574 005301                DEC      R1            ;COUNT
5141 034576 001375                BNE     35             ;BRANCH IF 256 NOT DONE
5142
5143          ;ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
5144          ;NOW THE INSERTED ERROR WILL BE PUT IN
5145 034600 012737 100000 015224  MOV      #100000,2#WRFROM+<5*2> ;INSERTED ERROR
5146
5147
5148
5149 034606 005037 015122          CLR     2#ERFLGS       ;CLEAR ERROR FLAG
5150 034612 004737 044652          JSR     PC,2#PUTREG     ;SAVE REGISTERS
5151
5152

```

```

5153 ;NOW READ DATA BUFFER WILL BE CHECKED
5154
5155 034616 007037 046046 JSR RO,2#COMPAR ;CHECK
5156 034622 015212 MRFROM ;GOOD BUFFER
5157 034624 016256 REINTO ;TEST BUFFER
5158 034626 000404 4+256. ;NUMBER OF WORDS CHECKED
5159 034630 034636 4$ ;RETURN POINT FOR ERROR HEADER
5160 034632 034642 5$ ;RETURN POINT FOR ERROR DATA
5161
5162 034634 034646 TST56 ;RETURN FOR GOOD COMPARISON
5163
5164 034636 104004 4$: ERROR 4 ;READ NEXT ERROR
5165 034640 000207 RTS PC ;RETURN TO "COMPAR"
5166 034642 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE
5167 ;HEADER WORDS
5168 ;5 TO 260 ARE DATA WORDS
5169 034644 000207 RTS PC ;RETURN TO "COMPAR"
5170

```

```

5171
5172
5173
5174
5175
5176
5177
5178
5179
5180
5181
5182
5183 034646 000004
5184 034630 012706 001000
5185
5186 034654 012737 000056 017322
5187
5188
5189
5190
5191
5192 034662 012746 000000
5193 034666 012705 000400
5194 034672 012700 053520
5195 034676 011620
5196 034700 005305
5197 034702 001375
5198 034704 005726
5199 034706 022020
5200 034710 012705 000017
5201
5202 034714 005020
5203 034716 005305
5204 034720 001375
5205
5206
5207 034722 004737 050442
5208
5209
5210
5211
5212 034726 012737 177777 015140
5213 034734 005037 047660
5214 034740 013737 047654 047656
5215 034746 013737 047662 047670
5216 034754 005037 047646
5217 034760 005037 047650
5218 034764 005037 047664
5219 034770 005037 047666
5220
5221
5222
5223
5224 034774 012737 010000 051602

```

```

*****
;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,2#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,2#FILLEC  ;INSERT THE TWO ECC WORDS ON THE DISK
                           ;IN THE CORRECT PLACE
;THESE ARE FOR ECC TEST ONLY
        MOV     #-1,2#TSECC  ;THIS IS AN ECC TEST
        CLR     2#POSITI    ;CLEAR ERROR POSITION COUNTER
        MOV     2#NCODE,2#NCOUNT ;TEMPORARY N-CODE COUNTER
        MOV     2#HARDER,2#HADTMP ;TEMPORARY HARD ERROR COUNTER
        CLR     2#GECC1     ;ECC LOW ORDER TO BE GENERATED
        CLR     2#GECC2     ;ECC HIGH ORDER TO BE GENERATED
        CLR     2#DATENV    ;CLEAR DATA ENVELOPE CLOCK COUNT
        CLR     2#ZCODE     ;CLEAR LEADING ZEROS CLOCK COUNT
;THESE ARE TO SETUP FOR DISKLESS USE ONLY
        MOV     #FMT22,2#CYL ;16 BITS PER WORD

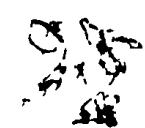
```



```

5270
5271 ; IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5272 ; FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
5273 ; FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5274 ; SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
5275 ; DETECTED
5276 ; HEADER AND DATA ARE TO BE CHECKED.
5277 ; IN CHECKING READ DATA THE WRITE FROM BUFFER
5278 ; "WRFROM" IS FILLED WITH EXPECTED DATA AND
5279 ; COMPARISONS ARE MADE
5280
5281 035166 005737 015122 TST @#ERFLG$ ; ANY ERRORS ALREADY THERE
5282 035172 001106 BNE TST57 ; BRANCH IF YES
5283 035174 004737 044652 JSR PC,@#PUTREG ; SAVE REGISTERS
5284 035200 022737 100000 015032 CMP #DCK,@#ER1 ; ONLY DATA CHECK ERROR SHOULD BE SET
5285 035206 001401 BEQ 6$ ; BRANCH IF YES
5286 035210 104032 ERROR 32 ; 32 BIT ECC REGISTER SHOULD BE NON
; ZERO
; ONLY 11 OF THE 32 BITS CAN BE SEEN
; IN THE PATERN REGISTER
; DCK SHOULD BE SET IN RHER1
5291 035212 013746 047646 6$: MOV @#GECC1,-(SP) ; GET PATTERN REGISTER
5292 035216 042716 174000 BIC #174000,(SP) ; KEEP ONLY 11 BITS
5293 035222 022637 015062 CMP (SP)+,@#EC2 ; COMPARE PATTERN REGISTER
5294 035226 001401 BEQ 7$ ; BRANCH IF GOOD
5295 035230 104032 ERROR 32 ; 11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5296
5297 035232 004037 050270 7$: JSR RO,@#ECORR ; GO TO ECC CORRECTION PROCESS
5298 035236 000000 8$: .WORD ; EXPECTED POSITION REG. WHEN CORRECTION
; IS COMPLETE
5299
5300
5301
5302
5303 035240 004737 044652 JSR PC,@#PUTREG ; SAVE REGISTERS
5304 035244 022737 100100 015032 CMP #DCK!ECH,@#ER1 ; WITH ERRORS INSERTED IN BIT POSITION 21
; THRU 32 HARD ERROR BIT SHOULD SET
5305
5306 035252 001401 BEQ 9$ ; BRANCH IF GOOD
5307 035254 104036 ERROR 36 ; WITH ERROR INSERTED IN BIT POSITION 21 THRU
; 32 ECH SHOULD SET
5308
5309
5310
5311
5312 035256 9$: JSR PC,@#CHECKE ; CHECK THAT DVA,RDY,DPR,DYR = 1
5313 035256 004737 045376 TYPE ,CPHALT ; CANNOT CONTINUE IF THEY DON'T
5314 035262 104400 005116 HALT ; STOP THE TEST AND RESTART PROGRAM
5315 035266 000000
5316 035270 012700 015212 MOV #WRFROM,RO ; GETTING READY TO FILL EXPECTED DATA
5317 035274 012720 010000 MOV #0!FMT22,(RO)+ ; CYLINDER C
5318 035300 112746 000000 MOVB #0,-(SP) ; IN LOWER BYTE GET SECTOR
5319 035304 112766 000000 000001 MOVB #0,1(SP) ; GET TRACK IN HIGHER BYTE
5320 035312 012620 MOV (SP)+,(RO)+ ; GET TRACK/SECTOR IN BUFFER
5321 035314 012720 000000 MOV #0,(RO)+ ; KEY1 IN BUFFER
5322 035320 012720 000000 MOV #0,(RO)+ ; KEY2 IN BUFFER
5323 035324 012701 000400 MOV #256.,R1 ; DATA WORD COUNTER

```



5324	035330	012702	000000		MOV	#0,R2	;DATA
5325	035334	010220		3\$:	MOV	R2,(R0)+	;DATA INTO BUFFER
5326	035336	005301			DEC	R1	;COUNT
5327	035340	001375			BNE	3\$	;BRANCH IF 256 NOT DONE
5328							

```

5329 ; ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
5330 ; NOW THE INSERTED ERROR WILL BE PUT IN
5331
5332 035342 012737 177760 015224 MOV #177760, @WRFROM+(<5*2) ; INSERTED ERROR
5333
5334
5335
5336 035350 005037 015122 CLR @ERFLGS ; CLEAR ERROR FLAG
5337 035354 004737 044652 JSR PC, @PUTREG ; SAVE REGISTERS
5338
5339
5340 ; NOW READ DATA BUFFER WILL BE CHECKED
5341
5342 035360 004037 046046 JSR RO, @COMPAR ; CHECK
5343 035364 015212 WRFROM ; GOOD BUFFER
5344 035366 016256 REINTO ; TEST BUFFER
5345 035370 000404 4+256. ; NUMBER OF WORDS CHECKED
5346 035372 035400 4$ ; RETURN POINT FOR ERROR HEADER
5347 035374 035404 5$ ; RETURN POINT FOR ERROR DATA
5348
5349 035376 035410 TST57 ; RETURN FOR GOOD COMPARISON
5350
5351 035400 104004 4$: ERROR 4 ; READ NEXT ERROR
5352 035402 000207 RTS PC ; RETURN TO "COMPAR"
5353 035404 104005 5$: ERROR 5 ; WORD NOS 1 TO 4 ARE
5354 ; HEADER WORDS
5355 ; 5 TO 260 ARE DATA WORDS
5356 035406 000207 RTS PC ; RETURN TO "COMPAR"
5357
5358
5359
5360
5361
5362

```

```

5363
5364
5365
5366
5367
5368
5369
5370
5371
5372
5373 035410 000004
5374 035412 012706 001000
5375
5376 035416 012737 000057 017322
5377 035424 012700 053422
5378 035430 012701 000460
5379 035434 005020
5380 035436 005301
5381 035440 001375
5382 035442 004767 007504
5383
5384
5385
5386 035446 012737 177777 015140
5387 035454 005037 047660
5388 035460 013737 047654 047656
5389 035466 013737 047662 047670
5390 035474 005037 047646
5391 035500 005037 047650
5392 035504 005037 047664
5393 035510 005037 047666
5394
5395
5396
5397
5398
5399
5400 035514 012737 010000 054724
5401
5402 035522 012737 000001 054726
5403 035530 005037 054730
5404 035534 005037 054732
5405 035540 012737 000400 054764
5406 035546 004537 046360
5407 035552 054724
5408 035554 054734
5409
5410
5411
5412 035556 012777 177374 157166
5413 035564 012700 015212
5414 035570 010077 157160
5415
5416 035574 012720 010000

```

```

*****
*TEST 57      WRITE ECC TEST 2
*****
*      THIS IS A WRITE ECC TEST
*      WRITE CYLINDER 0, FORMAT 16 BITS PER WORD
*      TRACK 0, SECTOR 1, KEYS 0, NUMBER OF WORDS 256
*      OF ALL ONES.
*****
†ST57: SCOPE
MOV      #STACK, SP      ;RESET STACK
MOV      #TTNO, @#TSTNM  ;THIS SAVES TEST NUMBER
MOV      #SECCAP, R0     ;POINTER
MOV      #304., R1      ;COUNTER
15: CLR   (R0)+          ;CLEAR SIMULATED DISK AREA
DEC      R1
BNE     15
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 ERPOR 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 BE SETUP FOR DISKLESS USE ONLY
MOV      #FMT22, @#WCYL  ;FORMAT22=16BIT WORDS AND
                        ;CYLINDER 0
MOV      #1, @#WSECTR   ;TRACK=0, SECTOR=1
CLR      @#KEY1         ;KEY1=0
CLR      @#KEY2         ;KEY2=0
MOV      #256., @#FNWORD ;256 DATA WORDS
JSR      RS, @#CRC      ;GO TO CALCULATE CRC
WCYL
GCRC
;THESE ARE REGULAR SETUPS
MOV      #-260., @#RHMC  ;256 DATA WORDS 4 HEADER WORDS
MOV      @#WRFROM, R0    ;THESE TWO INSTRUCTIONS GETS
MOV      R0, @#RBA      ;ADDR. OF WRFROM INTO R0 AND
                        ;BUS ADDRESS REGISTER
MOV      #FMT22, (R0)+  ;FORMAT=16 BIT WORDS

```



```

5417
5418 035600 012720 000001 25: MOV #1 (R0)+ ;CYLINDER=0
5419 035604 005020 CLR (R0)+ ;TRACK=0, SECTOR=1, KEYS=0
5420 035606 005020 CLR (R0)+ ;KEY1=0
5421 035610 012705 000400 MOV #256, R5 ;KEY2=0
5422 035614 012720 177777 35: MOV #-1, (R0)+ ;COUNTER
5423 035620 005305 DEC R5 ;MOVE ALL ONES FOR DATA
5424 035622 001374 BNE 35 ;BRANCH IF DATA NOT COMPLETE
5425 035624 012777 000001 157132 MOV #1, @RH0ST ;TRACK=0 SECTOR=1
5426
5427 035632 004737 045206 JSR PC, @#CHECKT ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
5428 035636 104400 005116 TYPE ,CPHALT ;CANNOT CONTINUE TESTING IF ANY OF THE
5429 ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5430 035642 000000 HALT ;STOP THE TEST
5431
5432 035644 013711 015170 MOV @#WRIFOR, @R1 ;GET READY FOR WRITE HEADER AND
5433 ;DATA WITH 62 IN RHCS1
5434 035650 005037 015122 CLR @#ERFLG$ ;CLEAR ERROR FLAG
5435 035654 012777 010000 157106 MOV #FMT22, @RHOF ;FORMAT BIT=1 (16 BIT WORDS)
5436 035662 005077 157104 CLR @RHCA ;CYLINDER =0
5437 035666 004737 054564 JSR PC, @#COMMD ;WRITE HEADER AND DATA
5438
5439 ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5440 ;FROM THE "COMMD" ROUTINE THAT MEANS ALL HEADER ON DISK
5441 ;IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
5442 ;ALL ONES AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
5443 ;THEY ARE ALL ZEROS
5444
5445 035672 005737 015122 TST @#ERFLG$ ;HAS ANY ERRORS OCCURED?
5446 ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
5447 035676 001056 BNE TST60 ;;BRANCH IF YES
5448
5449
5450 ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
5451
5452 035700 023737 047646 054520 CMP @#GECC1, @#HECC1;COMPARE SOFTWARE ECC WITH HARDWARE ECC
5453 035706 001402 BEQ 65 ;BRANCH IF GOOD
5454 035710 104031 ERROR 31 ;LOW ORDER ECC IN ERROR
5455 035712 000405 BR 75 ;BRANCH TO CONTINUE
5456 035714 023737 047650 054522 65: CMP @#GECC2, @#HECC2;COMPARE SOFTWARE ECC WITH HARDWARE ECC
5457 035722 001401 BEQ 75 ;BRANCH IF GOOD
5458 035724 104031 ERROR 31 ;HIGH ORDER ECC IN ERROR
5459
5460
5461 035726 75: JSR PC, @#CHECKE ;CHECK THAT DVA, RDY, DPR, DRY = 1
5462 035726 004737 045376 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
5463 035732 104400 005116 HALT ;STOP THE TEST AND RESTART PROGRAM
5464 035736 000000
5465
5466
5467
5468
5469 ;FILL "REINTO" BUFFER WITH EXPECTED DATA
5470

```

```

5471 035740 004037 045070 JSR RO, @#CLAREA ;FILL REINTO BUFFER
5472 035744 016256 REINTO ;FROM
5473 035746 017254 REINTO+(255.*2) ;TO
5474 035750 177777 .WORD -1 ;DATA
5475
5476 035752 013737 047646 017256 MOV @#GECC1, @#REINTO+(256.*2);FILL ECC1
5477 035760 013737 047650 017260 MOV @#GECC2, @#REINTO+(257.*2);FILL ECC2
5478 035766 004037 045070 JSR RO, @#CLAREA ;FILL REST
5479 035772 017262 REINTO+(258.*2) ;FROM
5480 035774 017316 REINTO+(272.*2) ;TO
5481 035776 000000 0 ;DATA
5482
5483
5484 036000 005037 015122 CLR @#ERFLGS ;CLEAR ERROR FLAG
5485
5486
5487 ;NOW COMPARE "DISK" BUFFER WITH "REINTO"
5488
5489 036004 004037 046046 JSR RO, @#COMPAR ;CHECK
5490 036010 016256 REINTO ;GOOD BUFFER
5491 036012 053520 DISK ;TEST BUFFER
5492 036014 000402 258. ;NUMBER OF WORDS CHECKED
5493 036016 036024 4$ ;RETURN POINT FOR ERROR HEADER
5494 036020 036030 5$ ;RETURN POINT FOR ERROR DATA
5495
5496 036022 036034 TST60 ;RETURN FOR GOOD COMPARISON
5497
5498 036024 104007 4$: ERROR 7 ;READ ERROR 10 NEXT
5499 036026 000207 RTS PC ;RETURN TO COMPARE
5500 036030 104010 5$: ERROR 10 ;WORD NOS 1 TO 256 ARE
5501 ;DATA WORDS
5502 ;WORD NOS 257 AND 258
5503 ;ARE ECC WHICH ARE CHECKED
5504 ;WORD NOS 259
5505 ;IS DATA GAP
5506 ;WORD NOS 260 TO 273
5507 ;ARE TOLERANCE GAP
5508 036032 000207 RTS PC ;RETURN TO COMPARE
5509
5510
5511
5512
5513

```

```

5514
5515
5516 ;*****
5517 ;*TEST 60 READ ECC ENABLED 2A
5518
5519 ;* THIS IS AN ECC READ DATA TEST
5520 ;* ERROR CORRECTION IS ENABLED
5521 ;* NO ERROR IS INSERTED
5522 ;* GOOD DATA USED IS 256 WORDS OF 177777
5523 ;* COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
5524 ;* TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA
5525 ;*****
5526 036034 000004 ;TST60: SCOPE
5527 036036 012706 001000 MOV #STACK,SP ;RESET STACK
5528
5529 036042 012737 000060 017322 MOV #TTNO,#TSTNM ;THIS SAVES TEST NUMBER
5530
5531
5532 ; SETUP FOR WHAT IS TO BE READ
5533 ; HEADER CRC IS RESTORED FROM A SUBROUTINE
5534
5535 036050 012746 177777 MOV #-1,-(SP) ;DATA TO BE READ
5536 036054 012705 000400 MOV #256,R5 ;COUNTER
5537 036060 012700 053520 MOV #DISK,RO ;START OF SIMULATED DISK DATA
5538 036064 011620 15: MOV (SP),(RO)+ ;MOVE IN DATA ON TO SIMULATED DISK
5539 036066 005305 DEC R5 ;COUNT
5540 036070 001375 BNE 15 ;BRANCH IF 256 NOT COMPLETE
5541 036072 005726 TST (SP)+ ;UNDO -(SP)
5542 036074 022020 CMP (RO)+,(RO)+ ;JUMP OVER THE TWO ECC WORDS
5543 036076 012705 000017 MOV #15,R5 ;1 DATA GAP
5544 ;14 TOLERANCE GAP
5545 036102 005020 25: CLR (RO)+ ;CLEAR DATA GAP, AND
5546 036104 005305 DEC R5 ;TOLERANCE GAP
5547 036106 001375 BNE 25 ;BRANCH IF NOT COMPLETE
5548
5549
5550 036110 004737 050442 JSR PC,#FILLEC ;INSERT THE TWO ECC WORDS ON THE DISK
5551 ;IN THE CORRECT PLACE
5552
5553 ;THESE ARE FOR ECC TEST ONLY
5554
5555 036114 012737 177777 015140 MOV #-1,#TSECC ;THIS IS AN ECC TEST
5556 036122 005037 047660 CLR #POSITI ;CLEAR ERROR POSITION COUNTER
5557 036126 013737 047654 047656 MOV #NCODE,#NCOUNT ;TEMPORARY N-CODE COUNTER
5558 036134 013737 047662 047670 MOV #HARDER,#HADTMP ;TEMPORARY HARD ERROR COUNTER
5559 036142 005037 047646 CLR #GECC1 ;ECC LOW ORDER TO BE GENERATED
5560 036146 005037 047650 CLR #GECC2 ;ECC HIGH ORDER TO BE GENERATED
5561 036152 005037 047664 CLR #DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT
5562 036156 005037 047666 CLR #ZCODE ;CLEAR LEADING ZEROS CLOCK COUNT
5563
5564
5565 ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
5566
5567 036162 012737 010000 051602 MOV #FMT22,#CYL ;16 BITS PER WORD

```

```

5568
5569 036170 112737 000000 051605      MOVB    #0,@#SECOTR+1      ;CYLINDER 0, FORMAT 16 BITS
5570 036176 112737 000000 051604      MOVB    #0,@#SECOTR      ;TRACK 0
5571 036204 012737 000000 051606      MOV     #0,@#KEY1        ;SECTOR 0
5572 036212 012737 000000 051610      MOV     #0,@#KEY2        ;KEY1=0
5573 036220 012737 000400 051662      MOV     #256.,@#DANWORD  ;KEY2=0
5574 036226 005037 051612      CLR     @#X              ;NO. OF DATA WORDS
5575 036232 004537 046360      JSR     R5,@#CRC         ;THIS IS A READ COMMAND
5576 036236 051602      CYL     @#X              ;GO TO CALCULATE CRC
5577 036240 053502      WCRRC
5578
5579
5580
5581
5582                                     ;THESE ARE REGULAR SETUPS
5583
5584 036242 004737 045152      JSR     PC,@#CLDISK     ;SETUP GENERAL REGISTERS
5585 036246 012777 177374 156476      MOV     #-256.-4,@#RMC  ;256. DATA 4 HEADER WORDS
5586 036254 012777 016256 156472      MOV     @#REINTO,@#RBA  ;STARTING ADDRESS OF READ BUFFER
5587 036262 112746 000000      MOVB    #0,-(SP)        ;IN LOWER BYTE GET SECTOR
5588 036266 112766 000000 000001      MOVB    #0,1(SP)        ;GET TRACK IN HIGHER BYTE
5589 036274 012677 156464      MOV     (SP)+,@#RHDST   ;TRACK/SECTOR IN RHDST
5590 036300 012777 010000 156462      MOV     @#FMT22,@#RHOF  ;16 BITS PER WORD
5591                                     ;ECC CORRECTION NOT INHIBIT
5592                                     ;BECAUSE ECC IS NOT GOING
5593                                     ;TO BE CHECKED
5594 036306 005077 156460      CLR     @#RHCA         ;CYLINDER 0
5595
5596 036312 004737 045206      JSR     PC,@#CHECKT    ;CHECK DVA,ROY,DPR,DRY = 1 AND OTHERS DON'T
5597 036316 104400 005116      TYPE    ,CPHALT        ;CANNOT CONTINUE TESTING IF ANY OF THE
5598                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5599 036322 000000      HALT                    ;STOP THE TEST
5600
5601 036324 013711 015174      MOV     @#REFOR,@#RI    ;READ HEADER AND DATA=72
5602 036330 005037 015122      CLR     @#ERFLG$       ;CLEAR ERROR FLAG
5603 036334 004737 051456      JSR     PC,@#COMHD     ;READ HEADER AND DATA
5604                                     ;IF THERE ARE READ ERRORS THEN
5605                                     ;ECC WILL NOT BE CHECKED
5606
5607
5608                                     ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5609                                     ;FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
5610                                     ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5611                                     ;SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
5612                                     ;DETECTED
5613                                     ;HEADER AND DATA ARE TO BE CHECKED.
5614                                     ;IN CHECKING READ DATA THE WRITE FROM BUFFER
5615                                     ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
5616                                     ;COMPARISONS ARE MADE
5617
5618 036340 005737 015122      TST     @#ERFLG$       ;ANY ERRORS ALREADY THERE
5619 036344 001102      BNE     TST61          ;BRANCH IF YES
5620 036346 004232 044652      JSR     PC,@#PUTREG    ;SAVE REGISTERS
5621 036352 005737 015032      TST     @#ER1         ;NO ERRORS SHOULD BE SET
    
```

```

5622 036356 001401          BEQ      6$          ;BRANCH IF NO ERRORS SET
5623 036360 104032          ERROR    32          ;32 BIT ECC REGISTER SHOULD BE ZERO
5624                                     ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5625                                     ;IN THE PATERN REGISTER
5626                                     ;DCK SHOULD BE SET IN RHER1
5627 036362 013746 047646    6$:   MOV      @#GECC1,-(SP) ;GET PATTERN REGISTER
5628 036366 042716 174000    BIC      #174000,(SP) ;KEEP ONLY 11 BITS
5629 036372 022637 015062    CMP      (SP)+,@#EC2 ;COMPARE PATTERN REGISTER
5630 036376 001401          BEQ      7$          ;BRANCH IF GOOD
5631 036400 104032          ERROR    32          ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5632
5633
5634
5635
5636                                     ;ADD 16 MAINTENANCE CLOCKS TO
5637                                     ;BRING EBL DOWN
5638
5639 036402 012700 000020    7$:   MOV      #16.,R0          ;COUNTER
5640 036406 052777 000002    8$:   BIS      #MCLK,@RHMR      ;SET CLOCK
5641 036414 042777 000002    BIC      #MCLK,@RHMR      ;CLEAR CLOCK
5642 036422 005300          DEC      R0              ;COUNT
5643 036424 001370          BNE      8$              ;BRANCH IF 16 CLOCKS NOT DONE
5644 036426 004737 045376    JSR      PC,@#CHECKE      ;CHECK THAT DVA,RDY,DPR,DRY = 1
5645 036432 104400 005116    TYPE    ,CPHALT          ;CANNOT CONTINUE IF THEY DON'T
5646 036436 000000          HALT                    ;STOP THE TEST AND RESTART PROGRAM
5647 036440 012700 015212    MOV      #MRFROM,R0      ;GETTING READY TO FILL EXPECTED DATA
5648 036444 012720 010000    MOV      #0!FMT22,(R0)+ ;CYLINDER 0
5649 036450 112746 000000    MOV      #0,-(SP)        ;IN LOWER BYTE GET SECTOR
5650 036454 112766 000000    MOV      #0,1(SP)        ;GET TRACK IN HIGHER BYTE
5651 036462 012620          MOV      (SP)+,(R0)+     ;GET TRACK/SECTOR IN BUFFER
5652 036464 012720 000000    MOV      #0,(R0)+        ;KEY1 IN BUFFER
5653 036470 012720 000000    MOV      #0,(R0)+        ;KEY2 IN BUFFER
5654 036474 012701 000400    MOV      #256.,R1        ;DATA WORD COUNTER
5655 036500 012702 177777    MOV      #-1,R2          ;DATA
5656
5657 036504 010220          3$:   MOV      R2,(R0)+        ;DATA INTO BUFFER
5658 036506 005301          DEC      R1              ;COUNT
5659 036510 001375          BNE      3$              ;BRANCH IF 256 NOT DONE
5660 036512 005037 015122    CLR      @#ERFLGS        ;CLEAR ERROR FLAG
5661 036516 004737 044652    JSR      PC,@#PUTREG      ;SAVE REGISTERS
5662
5663                                     ;NOW READ DATA BUFFER WILL BE CHECKED
5664
5665 036522 004037 046846    JSR      R0,@#COMPAR      ;CHECK
5666 036526 015212          MRFROM ;GOOD BUFFER
5667 036530 016256          REINTO ;TEST BUFFER
5668 036532 000404          4+256. ;NUMBER OF WORDS CHECKED
5669 036534 036542          4$      ;RETURN POINT FOR ERROR HEADER
5670 036536 036546          5$      ;RETURN POINT FOR ERROR DATA
5671
5672 036540 036552          TST61 ;RETURN FOR GOOD COMPARISON
5673
5674 036542 104004          4$:   ERROR    4          ;READ NEXT ERROR
5675 036544 000207          RTS     PC              ;RETURN TO "COMPAR"

```

L14

MAINDEC-11-DZRPT-D RJPO4 DISKLESS RH11 TEST-PART 2  
DZRPTC.TST T60 READ ECC ENABLED 2A

MACY11 27(663) 7-OCT-75 17:33 PAGE 137

SEQ 0179

5676 036546 104005  
5677  
5678  
5679 036550 000207  
5680  
5681  
5682  
5683

SS: ERROR 5  
RTS PC

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

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

\*\*\*\*\*  
;\*TEST 61 READ ECC ENABLED 2B

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

\*\*\*\*\*

†ST61: 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 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 @#HARDEA, @#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 SETUP FOR DISKLESS USE ONLY





```

5792 ; IN CHECKING READ DATA THE WRITE FROM BUFFER
5793 ; "WRFROM" IS FILLED WITH EXPECTED DATA AND
5794 ; COMPARISONS ARE MADE
5795
5796 037072 005737 015122 TST @#ERFLGS ; ANY ERRORS ALREADY THERE
5797 037076 001077 BNE TST62 ; BRANCH IF YES
5798 037100 004737 044652 JSR PC,@#PUTREG ; SAVE REGISTERS
5799 037104 022737 100000 015032 CMP #DCK,@#ERI ; ONLY DATA CHECK ERROR SHOULD BE SET
5800 037112 001401 BEQ 65 ; BRANCH IF YES
5801 037114 104032 ERROR 32 ; 32 BIT ECC REGISTER SHOULD BE NON
5802 ; ZERO
5803 ; ONLY 11 OF THE 32 BITS CAN BE SEEN
5804 ; IN THE PATTERN REGISTER
5805 ; DCK SHOULD BE SET IN RHER1
5806 037116 013746 047646 65: MOV @#GECC1,-(SP) ; GET PATTERN REGISTER
5807 037122 042716 174000 BIC #174000,(SP) ; KEEP ONLY 11 BITS
5808 037126 022637 015062 CMP (SP)+,@#EC2 ; COMPARE PATTERN REGISTER
5809 037132 001401 BEQ 75 ; BRANCH IF GOOD
5810 037134 104032 ERROR 32 ; 11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5811
5812 037136 004037 050270 75: JSR R0,@#ECORR ; GO TO ECC CORRECTION PROCESS
5813 037142 000026 85: 22. ; EXPECTED POSITION REG. WHEN CORRECTION
5814 ; IS COMPLETE
5815
5816
5817
5818 037144 004737 045376 JSR PC,@#CHECKE ; CHECK THAT DVA,RDY,DPR,DRY = 1
5819 037150 104400 005116 TYPE ,CPHALT ; CANNOT CONTINUE IF THEY DON'T
5820 037154 000000 HALT ; STOP THE TEST AND RESTART PROGRAM
5821 037156 012700 015212 MOV #WRFROM,R0 ; GETTING READY TO FILL EXPECTED DATA
5822 037162 012720 010000 MOV #0:FMT22,(R0)+ ; CYLINDER 0
5823 037166 112746 000000 MOVB #0,-(SP) ; IN LOWER BYTE GET SECTOR
5824 037172 112766 000000 000001 MOVB #0,1(SP) ; GET TRACK IN HIGHER BYTE
5825 037200 012620 MOV (SP)+,(R0)+ ; GET TRACK/SECTOR IN BUFFER
5826 037202 012720 000000 MOV #0,(R0)+ ; KEY1 IN BUFFER
5827 037206 012720 000000 MOV #0,(R0)+ ; KEY2 IN BUFFER
5828 037212 012701 000400 MOV #256,R1 ; DATA WORD COUNTER
5829 037216 012702 177777 MOV #-1,R2 ; DATA
5830 037222 010220 35: MOV R2,(R0)+ ; DATA INTO BUFFER
5831 037224 005301 DEC R1 ; COUNT
5832 037226 001375 BNE 35 ; BRANCH IF 256 NOT DONE
5833
5834 ; ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
5835 ; NOW THE INSERTED ERROR WILL BE PUT IN
5836
5837 037230 012737 077777 015224 MOV #77777,@#WRFROM+(5*2) ; INSERTED ERROR
5838 037236 004737 044652 JSR PC,@#PUTREG ; SAVE REGISTERS
5839 037242 005037 015122 CLR @#ERFLGS ; CLEAR ERROR FLAG
5840
5841
5842 ; NOW READ DATA BUFFER WILL BE CHECKED
5843
5844 037246 004037 046046 JSR R0,@#COMPAR ; CHECK
5845 037252 015212 WRFROM ; GOOD BUFFER

```

5846	037254	016256		REINTO	
5847	037256	000404		4+256.	
5848	037260	037266		4\$	
5849	037262	037272		5\$	
5850					
5851	037264	037276		TST62	
5852					
5853	037266	104004	4\$:	ERROR	4
5854	037270	000207		RTS	PC
5855	037272	104005	5\$:	ERROR	5
5856					
5857					
5858	037274	000207		RTS	PC
5859					

```
; TEST BUFFER
; NUMBER OF WORDS CHECKED
; RETURN POINT FOR ERROR HEADER
; RETURN POINT FOR ERROR DATA

; RETURN FOR GOOD COMPARISON

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

```

5860
5861 ;*****
5862 ;*TEST 62 READ ECC ENABLED 2C
5863
5864 ;* THIS IS AN ECC READ DATA TEST
5865 ;* ERROR CORRECTION IS ENABLED
5866 ;* A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 32 AND 21
5867 ;* GOOD DATA USED IS 256 WORDS OF 177777
5868 ;* COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
5869 ;* TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA
5870
5871 ;*****
5872 037276 000004 TST62: SCOPE
5873 037300 012706 001000 MOV #STACK,SP ;RESET STACK
5874
5875 037304 012737 000062 017322 MOV #TTNO,@#TSTNM ;THIS SAVES TEST NUMBER
5876
5877
5878 ; SETUP FOR WHAT IS TO BE READ
5879 ; HEADER CRC IS RESTORED FROM A SUBROUTINE
5880
5881 037312 012746 177777 MOV #-1,-(SP) ;DATA TO BE READ
5882 037316 012705 000400 MOV #256,R5 ;COUNTER
5883 037322 012700 053520 MOV #DISK,RO ;START OF SIMULATED DISK DATA
5884 037326 011620 15: MOV (SP),(RO)+ ;MOVE IN DATA ON TO SIMULATED DISK
5885 037330 005305 DEC R5 ;COUNT
5886 037332 001375 BNE 15 ;BRANCH IF 256 NOT COMPLETE
5887 037334 005726 TST (SP)+ ;UNDO -(SP)
5888 037336 022020 CMP (RO)+,(RO)+ ;JUMP OVER THE TWO ECC WORDS
5889 037340 012705 000017 MOV #15,R5 ;1 DATA GAP
5890
5891 037344 005020 25: CLR (RO)+ ;14 TOLERANCE GAP
5892 037346 005305 DEC R5 ;CLEAR DATA GAP, AND
5893 037350 001375 BNE 25 ;TOLERANCE GAP
5894 ;BRANCH IF NOT COMPLETE
5895
5896 037352 004737 050442 JSR PC,@#FILLEC ;INSERT THE TWO ECC WORDS ON THE DISK
5897 ;IN THE CORRECT PLACE
5898
5899 ;THESE ARE FOR ECC TEST ONLY
5900
5901 037356 012737 177777 015140 MOV #-1,@#TSECC ;THIS IS AN ECC TEST
5902 037364 005037 047660 CLR @#POSITI ;CLEAR ERROR POSITION COUNTER
5903 037370 013737 047654 047656 MOV @#NCODE,@#NCOUNT ;TEMPORARY N-CODE COUNTER
5904 037376 013737 047662 047670 MOV @#HARDER,@#HADTMP ;TEMPORARY HARD ERROR COUNTER
5905 037404 005037 047646 CLR @#GECC1 ;ECC LOW ORDER TO BE GENERATED
5906 037410 005037 047650 CLR @#GECC2 ;ECC HIGH ORDER TO BE GENERATED
5907 037414 005037 047664 CLR @#DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT
5908 037420 005037 047666 CLR @#ZCODE ;CLEAR LEADING ZEROS CLOCK COUNT
5909
5910
5911 ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
5912
5913 037424 012737 010000 051602 MOV #FMT22,@#CYL ;16 BITS PER WORD

```

```

5914                                     ;CYLINDER 0, FORMAT 16 BITS
5915 037432 112737 000000 051605      MOVB  #0,@#SECOTR+1 ;TRACK 0
5916 037440 112737 000000 051604      MOVB  #0,@#SECOTR   ;SECTOR 0
5917 037446 012737 000000 051606      MOV   #0,@#KEY1    ;KEY1=0
5918 037454 012737 000000 051610      MOV   #0,@#KEY2    ;KEY2=0
5919 037462 012737 000400 051662      MOV   #256.,@#DANORD ;NO. OF DATA WORDS
5920 037470 005037 051612              CLR   @#X           ;THIS IS A READ COMMAND
5921 037474 004537 046360              JSR   RS,@#CRC      ;GO TO CALCULATE CRC
5922 037500 051602
5923 037502 053502
5924
5925
5926                                     ;THIS IS TO INSERT ERROR
5927                                     ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
5928                                     ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
5929                                     ;THIS MOVE
5930
5931 037504 012737 077757 053522      MOV   #77757,@#DISK+2 ;FORCE ERROR ON BIT NUMBER 32 AND 21
5932                                     ;50 ERROR POSITION REGISTER WILL SHOW
5933                                     ;22
5934 037512 012737 010040 037666      MOV   #4128.,@#85    ;INSERT POSITION REG.
5935
5936
5937                                     ;THESE ARE REGULAR SETUPS
5938
5939 037520 004737 045152              JSR   PC,@#CLDISK   ;SETUP GENERAL REGISTERS
5940 037524 012777 177374 155220      MOV   #-256.-4,@#RHMC ;256. DATA 4 HEADER WORDS
5941 037532 012777 016256 155214      MOV   @#REINTO,@#RMB ;STARTING ADDRESS OF READ BUFFER
5942 037540 112746 000000              MOVB  #0,-(SP)      ;IN LOWER BYTE GET SECTOR
5943 037544 112766 000000 000001      MOVB  #0,1(SP)      ;GET TRACK IN HIGHER BYTE
5944 037552 012677 155206              MOV   (SP)+,@#RHST  ;TRACK/SECTOR IN RHST
5945 037556 012777 010000 155204      MOV   #FMT22,@#RHOF ;16 BITS PER WORD
5946                                     ;ECC CORRECTION NOT INHIBIT
5947                                     ;BECAUSE ECC IS NOT GOING
5948                                     ;TO BE CHECKED
5949 037564 005077 155202              CLR   @#RHCA       ;CYLINDER 0
5950
5951 037570 004737 045206              JSR   PC,@#CHECKT   ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
5952 037574 104400 005116              TYPE  ,CPHALT      ;CANNOT CONTINUE TESTING IF ANY OF THE
5953                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
5954 037600 000000                      HALT                ;STOP THE TEST
5955
5956 037602 013711 015174              MOV   @#REFOR,@#R1  ;READ HEADER AND DATA=72
5957 037606 005037 015122              CLR   @#ERFLG$     ;CLEAR ERROR FLAG
5958 037612 004737 051456              JSR   PC,@#COMHD    ;READ HEADER AND DATA
5959                                     ;IF THERE ARE READ ERRORS THEN
5960                                     ;ECC WILL NOT BE CHECKED
5961
5962
5963                                     ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
5964                                     ;FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
5965                                     ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
5966                                     ;SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
5967                                     ;DETECTED

```

# F15

```

5968 ;HEADER AND DATA ARE TO BE CHECKED.
5969 ;IN CHECKING READ DATA THE WRITE FROM BUFFER
5970 ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
5971 ;COMPARISONS ARE MADE
5972
5973 037616 005737 015122 TST @#ERFLG5 ;ANY ERRORS ALREADY THERE
5974 037622 001106 BNE TST63 ;BRANCH IF YES
5975 037624 004737 044652 JSR PC,@#PUTREG ;SAVE REGISTERS
5976 037630 022737 100000 015032 CMP #DCK,@#ER1 ;ONLY DATA CHECK ERROR SHOULD BE SET
5977 037636 001401 BEQ 6$ ;BRANCH IF YES
5978 037640 104032 ERROR 32 ;32 BIT ECC REGISTER SHOULD BE NON
5979 ;ZERO
5980 ;ONLY 11 OF THE 32 BITS CAN BE SEEN
5981 ;IN THE PATERN REGISTER
5982 ;DCK SHOULD BE SET IN RHER1
5983 037642 013746 047646 6$: MOV @#GECC1,-(SP) ;GET PATTERN REGISTER
5984 037646 042716 174000 BIC #174000,(SP) ;KEEP ONLY 11 BITS
5985 037652 022637 015062 CMP (SP)+,@#EC2 ;COMPARE PATTERN REGISTER
5986 037656 001401 BEQ 7$ ;BRANCH IF GOOD
5987 037660 104032 ERROR 32 ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
5988
5989 037662 004037 050270 7$: JSR R0,@#ECORR ;GO TO ECC CORRECTION PROCESS
5990 037666 000000 8$: .WORD ;EXPECTED POSITION REG. WHEN CORRECTION
5991 ;IS COMPLETE
5992
5993 037670 004737 044652 JSR PC,@#PUTREG ;SAVE REGISTERS
5994
5995
5996 037674 022737 100100 015032 CMP #DCK!ECH,@#ER1 ;WITH ERRORS INSERTED IN BIT POSITION 21
5997 ;AND 32 HARD ERROR BIT SHOULD SET
5998 037702 001401 BEQ 9$ ;BRANCH IF GOOD
5999 037704 104036 ERROR 36 ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
6000 ;32 HCE SHOULD SET
6001
6002
6003
6004
6005 037706 9$: JSR PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DYR = 1
6006 037706 004737 045376 TYPE ,CPHALT ;CANNOT CONTINUE IF THEY DON'T
6007 037712 104400 005116 HALT ;STOP THE TEST AND RESTART PROGRAM
6008 037716 000000 MOV #WRFROM,R0 ;GETTING READY TO FILL EXPECTED DATA
6009 037720 012700 015212 MOV #0!FMT22,(R0)+ ;CYLINDER 0
6010 037724 012720 010000 MOV #0,-(SP) ;IN LOWER BYTE GET SECTOR
6011 037730 112746 000000 MOV #0,1(SP) ;GET TRACK IN HIGHER BYTE
6012 037734 112766 000000 000001 MOV (SP)+,(R0)+ ;GET TRACK/SECTOR IN BUFFER
6013 037742 012620 MOV #0,(R0)+ ;KEY1 IN BUFFER
6014 037744 012720 000000 MOV #0,(R0)+ ;KEY2 IN BUFFER
6015 037750 012720 000000 MOV #256,R1 ;DATA WORD COUNTER
6016 037754 012701 000400 MOV #-1,R2 ;DATA
6017 037760 012702 177777 3$: MOV R2,(R0)+ ;DATA INTO BUFFER
6018 037764 010220 MOV R1 ;COUNT
6019 037766 005301 DEC ;BRANCH IF 256 NOT DONE
6020 037770 001375 BNE 3$
6021

```

```

6022                                     ; ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6023                                     ; NOW THE INSERTED ERROR WILL BE PUT IN
6024
6025 037772 012737 077757 015224      MOV    #77757,2#WRFROM+(5*2) ; INSERTED ERROR
6026 040000 004737 044652              JSR    PC,2#PUTREG           ; SAVE REGISTERS
6027 040004 005037 015122              CLR    2#ERFLGS             ; CLEAR ERROR FLAG
6028
6029
6030                                     ; NOW READ DATA BUFFER WILL BE CHECKED
6031
6032 040010 004037 046046              JSR    R0,2#COMPAR          ; CHECK
6033 040014 015212                      WRFROM                      ; GOOD BUFFER
6034 040016 016256                      REINTO                      ; TEST BUFFER
6035 040020 000404                      4+256.                      ; NUMBER OF WORDS CHECKED
6036 040022 040030                      4$                          ; RETURN POINT FOR ERROR HEADER
6037 040024 040034                      5$                          ; RETURN POINT FOR ERROR DATA
6038
6039 040026 040040                      TST63                       ; RETURN FOR GOOD COMPARISON
6040
6041 040030 104004                      4$: ERROR 4                 ; READ NEXT ERROR
6042 040032 000207                      RTS PC                       ; RETURN TO "COMPAR"
6043 040034 104005                      5$: ERROR 5                 ; WORD NOS 1 TO 4 ARE
6044                                     ; HEADER WORDS
6045                                     ; 5 TO 260 ARE DATA WORDS
6046 040036 000207                      RTS PC                       ; RETURN TO "COMPAR"
    
```

# H15

MAINDEC-11-DZRPT-D, RJPO4 DISKLESS RH11 TEST-PART 2  
DZRPTC.TST T62 READ ECC ENABLED 2C

MACY11 27(663) 7-OCT-75 17:33 PAGE 146

SEQ 0188

6047  
6048  
6049  
6050  
6051  
6052  
6053  
6054  
6055  
6056  
6057  
6058  
6059  
6060  
6061  
6062  
6063  
6064  
6065  
6066  
6067  
6068  
6069  
6070  
6071  
6072  
6073  
6074  
6075  
6076  
6077  
6078  
6079  
6080  
6081  
6082  
6083  
6084  
6085  
6086  
6087  
6088  
6089  
6090  
6091  
6092  
6093  
6094  
6095  
6096  
6097  
6098  
6099  
6100

\*\*\*\*\*  
;TEST 63 WRITE ECC TEST 3

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

\*\*\*\*\*

†ST63: SCOPE  
MOV #STACK, SP ;RESET STACK  
MOV #TTNO, @TSTNM ;THIS SAVES TEST NUMBER  
MOV #SECGAP, R0 ;POINTER  
MOV #304., R1 ;COUNTER  
1S: CLR (R0)+ ;CLEAR SIMULATED DISK AREA  
DEC R1  
BNE 1S  
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, @MNCYL ;FORMAT22=16BIT WORDS AND  
CYLINDER 0  
MOV #1, @MSECTR ;TRACK=0, SECTOR=1  
CLR @KEY1 ;KEY1=0  
CLR @KEY2 ;KEY2=0  
MOV #256., @FNWORD ;256 DATA WORDS  
JSR R5, @CRC ;GO TO CALCULATE CRC  
MNCYL  
GCRC

;THESE ARE REGULAR SETUPS

MOV #-260., @RHMC ;256 DATA WORDS 4 HEADER WORDS  
MOV #WRFROM, R0 ;THESE TWO INSTRUCTIONS GETS  
MOV R0, @R4BA ;ADDR. OF WRFROM INTO R0 AND  
BUS ADDRESS REGISTER  
MOV #FMT22, (R0)+ ;FORMAT=16 BIT WORDS

```

6101                                     ;CYLINDER=0
6102 040230 012720 000001      2S:  MOV    #1,(R0)+      ;TRACK=0, SECTOR=1, KEYS=0
6103 040234 005020              CLR    (R0)+      ;KEY1=0
6104 040236 005020              CLR    (R0)+      ;KEY2=0
6105 040240 012705 000400      MOV    #256,R5    ;COUNTER
6106 040244 012720 052525      3S:  MOV    #52525,(R0)+ ;MOVE ALL 52525 FOR DATA
6107 040250 005305              DEC    R5
6108 040252 001374              BNE    3S        ;BRANCH IF DATA NOT COMPLETE
6109 040254 012777 000001 154502 MOV    #1,@RHDS1 ;TRACK=0 SECTOR=1
6110
6111 040262 004737 045206      JSR    PC,@#CHECKT ;CHECK DVA,RDY,DPR,DRY = 1 AND OTHERS DON'T
6112 040266 104400 005116      TYPE  ,CPHALT    ;CANNOT CONTINUE TESTING IF ANY OF THE
6113                                ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
6114 040272 000000              HALT            ;STOP THE TEST
6115
6116 040274 013711 015170      MOV    @#WRIFOR,@R1 ;GET READY FOR WRITE HEADER AND
6117                                ;DATA WITH 62 IN RHCS1
6118 040300 005037 015122      CLR    @#ERFLG$   ;CLEAR ERROR FLAG
6119 040304 012777 010000 154456 MOV    #FMT22,@RHOF ;FORMAT BIT=1 (16 BIT WORDS)
6120 040312 005077 154454      CLR    @RHCA     ;CYLINDER =0
6121 040316 004737 054564      JSR    PC,@#COMHHD ;WRITE HEADER AND DATA
6122
6123                                ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6124                                ;FROM THE "COMHHD" ROUTINE THAT MEANS ALL HEADER ON DISK
6125                                ;IS GOOD IE. ONLY DATA IS TO BE CHECKED TO SEE IF THEY ARE
6126                                ;ALL 52525 AND WRITE DATA GAP AND TOLERANCE GAP TO SEE IF
6127                                ;THEY ARE ALL ZEROS
6128
6129 040322 005737 015122      TST    @#ERFLG$   ;HAS ANY ERRORS OCCURED?
6130                                ;IF WRITE ERROR OCCURS ECC IS NOT CHECKED
6131 040326 001056              BNE    TST64    ;;BRANCH IF YES
6132
6133                                ;COMPARE SOFTWARE GENERATED ECC WITH THAT GENERATED BY HARDWARE
6134                                ;COMPARE SOFTWARE ECC WITH HARDWARE ECC
6135 040330 023737 047646 054520 CMP    @#GECC1,@#HECC1 ;COMPARE SOFTWARE ECC WITH HARDWARE ECC
6136 040336 001402              BEQ    6S        ;BRANCH IF GOOD
6137 040340 104031              ERROR  31        ;LOW ORDER ECC IN ERROR
6138 040342 000405              BR     7S        ;BRANCH TO CONTINUE
6139 040344 023737 047650 054522 6S:  CMP    @#GECC2,@#HECC2 ;COMPARE SOFTWARE ECC WITH HARDWARE ECC
6140 040352 001401              BEQ    7S        ;BRANCH IF GOOD
6141 040354 104031              ERROR  31        ;HIGH ORDER ECC IN ERROR
6142
6143
6144                                7S:
6145 040356 004737 045376      JSR    PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
6146 040362 104400 005116      TYPE  ,CPHALT    ;CANNOT CONTINUE IF THEY DON'T
6147 040366 000000              HALT            ;STOP THE TEST AND RESTART PROGRAM
6148
6149
6150
6151                                ;FILL "REINTO" BUFFER WITH EXPECTED DATA
6152
6153
6154 040370 004037 045070      JSR    R0,@#CLAREA ;FILL REINTO BUFFER

```



```

6155 040374 016256 REINTO ;FROM
6156 040376 017254 REINTO+(255.*2) ;TO
6157 040400 052525 .WORD 52525 ;DATA
6158
6159 040402 013737 047646 017256 MOV @#GECC1,@#REINTO+(256.*2);FILL ECC1
6160 040410 013737 047650 017260 MOV @#GECC2,@#REINTO+(257.*2);FILL ECC2
6161 040416 004037 045070 JSR RO,@#CLAREA ;FILL REST
6162 040422 017262 REINTO+(258.*2) ;FROM
6163 040424 017316 REINTO+(272.*2) ;TO
6164 040426 000000 0 ;DATA
6165
6166
6167 040430 005037 015122 CLR @#ERFLGS ;CLEAR ERROR FLAG
6168
6169
6170 ;NOW COMPARE "DISK" BUFFER WITH "REINTO"
6171
6172 040434 004037 046046 JSR RO,@#COMPAR ;CHECK
6173 040440 016256 REINTO ;GOOD BUFFER
6174 040442 053520 DISK ;TEST BUFFER
6175 040444 000402 258. ;NUMBER OF WORDS CHECKED
6176 040446 040454 4$ ;RETURN POINT FOR ERROR HEADER
6177 040450 040460 5$ ;RETURN POINT FOR ERROR DATA
6178
6179 040452 040464 TST64 ;RETURN FOR GOOD COMPARISON
6180
6181 040454 104007 4$: ERROR 7 ;READ ERROR 10 NEXT
6182 040456 000207 RTS PC ;RETURN TO COMPARE
6183 040460 104010 5$: ERROR 10 ;WORD NOS 1 TO 256 ARE
6184 ;DATA WORDS
6185 ;WORD NOS 257 AND 258
6186 ;ARE ECC WHICH ARE CHECKED
6187 ;WORD NOS 259
6188 ;IS DATA GAP
6189 ;WORD NOS 260 TO 273
6190 ;ARE TOLERANCE GAP
6191 040462 000207 RTS PC ;RETURN TO COMPARE

```

```

6192
6193
6194
6195
6196
6197
6198
6199
6200
6201
6202
6203
6204 040464 000004
6205 040466 012706 001000
6206 040472 012737 000064 017322
6207
6208
6209
6210
6211
6212 040500 012746 052525
6213 040504 012705 030400
6214 040510 012700 053520
6215 040514 011620
6216 040516 005305
6217 040520 001375
6218 040522 005726
6219 040524 022020
6220 040526 012705 000017
6221
6222 040532 005020
6223 040534 005305
6224 040536 001375
6225
6226 040540 004737 050442
6227
6228
6229
6230
6231 040544 012737 177777 015140
6232 040552 005037 047660
6233 040556 013737 047654 047656
6234 040564 013737 047662 047670
6235 040572 005037 047646
6236 040576 005037 047650
6237 040602 005037 047664
6238 040606 005037 047666
6239
6240
6241
6242
6243 040612 012737 010000 051602
6244
6245 040620 112737 000000 051605

```

```

;*****
;#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
;*****
TST64: 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
MOVB #0, @#SECOTR+1 ;TRACK 0

```

6246	040626	112737	000000	051604	MOVB	#0, @#SECOTR	; SECTOR 0
6247	040634	012737	000000	051606	MOV	#0, @#KEY1	; KEY1=0
6248	040642	012737	000000	051610	MOV	#0, @#KEY2	; KEY2=0
6249	040650	012737	000400	051662	MOV	#256., @#DAMORD	; NO. OF DATA WORDS
6250	040656	005037	051612		CLR	@#X	; THIS IS A READ COMMAND
6251	040662	004537	046360		JSR	RS, @#CRC	; GO TO CALCULATE CRC
6252	040666	051602			CYL		
6253	040670	053502			MCRC		
6254							
6255							
6256							
6257							
6258							
6259							
6260	040672	004737	045152		JSR	PC, @#CLDISK	; SETUP GENERAL REGISTERS
6261	040676	012777	177374	154046	MOV	#-256.-4., @#RHMC	; 256. DATA 4 HEADER WORDS
6262	040704	012777	016256	154042	MOV	#REINTO, @#RHBA	; STARTING ADDRESS OF READ BUFFER
6263	040712	112746	000000		MOVB	#0, -(SP)	; IN LOWER BYTE GET SECTOR
6264	040716	112766	000000	000001	MOVB	#0, 1(SP)	; GET TRACK IN HIGHER BYTE
6265	040724	012677	154034		MOV	(SP)+, @#RHST	; TRACK/SECTOR IN RHDST
6266	040730	012777	010000	154032	MOV	#FMT22, @#RHOF ; 16	; BITS PER WORD
6267							; ECC CORRECTION NOT INHIBIT
6268							; BECAUSE ECC IS NOT GOING
6269							; TO BE CHECKED
6270	040736	005077	154030		CLR	@#RHCA	; CYLINDER 6
6271							
6272	040742	004737	045206		JSR	PC, @#CHECKT	; CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
6273	040746	104400	005116		TYPE	, CPHALT	; CANNOT CONTINUE TESTING IF ANY OF THE
6274							; ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
6275	040752	000000			HALT		; STOP THE TEST
6276							
6277	040754	013711	015174		MOV	@#REFOR, @#RI	; READ HEADER AND DATA=72
6278	040760	005037	015122		CLR	@#ERFLG\$	; CLEAR ERROR FLAG
6279	040764	004737	051456		JSR	PC, @#CCMHD	; READ HEADER AND DATA
6280							; IF THERE ARE READ ERRORS THEN
6281							; ECC WILL NOT BE CHECKED
6282							
6283							
6284							
6285							
6286							
6287							
6288							
6289							
6290							
6291							
6292							
6293							
6294	040770	005737	015122		TST	@#ERFLG\$	; ANY ERRORS ALREADY THERE
6295	040774	001102			BNE	TST65	; BRANCH IF YES
6296	040776	004737	044652		JSR	PC, @#PUTREG	; SAVE REGISTERS
6297	041002	005737	015032		TST	@#ERI	; NO ERRORS SHOULD BE SET
6298	041006	001401			BEG	65	; BRANCH IF NO ERRORS SET
6299	041010	104032			ERROR	32	; 32 BIT ECC REGISTER SHOULD BE ZERO

; THESE ARE REGULAR SETUPS

; 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



N15

MAINDEC-11-DZRPT-D, RJPO4 DISKLESS RH11 TEST-PART 2  
DZRPTC.TST T64 READ ECC ENABLED 3A

MACY11 27(663) 7-OCT-75 17:33 PAGE 152

SEQ 0194

6354 041200 000207

RTS

PC

;RETURN TO "COMPAR"

```

6355
6356
6357
6358
6359
6360
6361
6362
6363
6364
6365
6366
6367
6368 041202 000004
6369 041204 012706 001000
6370 041210 012737 000065 017322
6371
6372
6373
6374
6375 041216 012746 052525
6376 041222 012705 000400
6377 041226 012700 053520
6378 041232 011620
6379 041234 005305
6380 041236 001375
6381 041240 005726
6382 041242 022020
6383 041244 012705 000017
6384
6385 041250 005020
6386 041252 005305
6387 041254 001375
6388
6389
6390 041256 004737 050442
6391
6392
6393
6394
6395
6396 041262 012737 177777 015140
6397 041270 005037 047660
6398 041274 013737 047654 047656
6399 041302 013737 047662 047670
6400 041310 005037 047646
6401 041314 005037 047650
6402 041320 005037 047664
6403 041324 005037 047666
6404
6405
6406
6407
6408 041330 012737 010000 051602

```

```

*****
; *TEST 65 READ ECC ENABLED 3B
; *
; * THIS IS AN ECC READ DATA TEST
; * ERROR CORRECTION IS ENABLED
; * A CORRECTABLE ERROR IS INSERTED IN BIT POSITION 4128
; * THIS IS THE LAST BIT OF THE ECC
; * GOOD DATA USED IS 256 WORDS OF 52525
; * COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
; * TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA
*****
†T65: 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,RO ;START OF SIMULATED DISK DATA
1$: MOV (SP),(RO)+ ;MOVE IN DATA ON TO SIMULATED DISK
DEC R5 ;COUNT
BNE 1$ ;BRANCH IF 256 NOT COMPLETE
TST (SP)+ ;UNDO -(SP)
CMP (RO)+,(RO)+ ;JUMP OVER THE TWO ECC WORDS
MOV #15,R5 ;1 DATA GAP
;14 TOLERANCE GAP
2$: CLR (RO)+ ;CLEAR DATA GAP, AND
DEC R5 ;TOLERANCE GAP
BNE 2$ ;BRANCH IF NOT COMPLETE

JSR PC,#FILLEC ;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 #HARDEA,#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

```



6463  
6464  
6465  
6466  
6467  
6468  
6469  
6470  
6471  
6472  
6473  
6474 041540 005737 015122  
6475 041544 001074  
6476 041546 004737 044652  
6477 041552 022737 100000 015032  
6478 041560 001401  
6479 041562 104032  
6480  
6481  
6482  
6483  
6484 041564 013746 047646 65:  
6485 041570 042716 174000  
6486 041574 022637 015062  
6487 041600 001401  
6488 041602 104032  
6489  
6490 041604 004037 050270 75:  
6491 041610 010026 85:  
6492  
6493  
6494  
6495  
6496 041612 004737 045376  
6497 041616 104400 005116  
6498 041622 000000  
6499 041624 012700 015212  
6500 041630 012720 010000  
6501 041634 112746 000000  
6502 041640 112766 000000 000001  
6503 041646 012620  
6504 041650 012720 000000  
6505 041654 012720 000000  
6506 041660 012701 000400  
6507 041664 012702 052525  
6508 041670 010220 35:  
6509 041672 005301  
6510 041674 001375  
6511  
6512  
6513  
6514  
6515  
6516

```

; 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

```

```

TST      2#ERFLG$      ; ANY ERRORS ALREADY THERE
BNE      TST66         ; BRANCH IF YES
JSR      PC, 2#PUTREG  ; SAVE REGISTERS
CMP      2#DCK, 2#ER1  ; ONLY DATA CHECK ERROR SHOULD BE SET
BEQ      65           ; BRANCH IF YES
ERROR    32           ; 32 BIT ECC REGISTER SHOULD BE NON
                        ; ZERO
                        ; ONLY 11 OF THE 32 BITS CAN BE SEEN
                        ; IN THE PATTERN REGISTER
                        ; DCK SHOULD BE SET IN RHER1
65:      MOV      2#GECC1, -(SP) ; GET PATTERN REGISTER
        BIC      2#174000, (SP) ; KEEP ONLY 11 BITS
        CMP      (SP)+, 2#EC2   ; COMPARE PATTERN REGISTER
        BEQ      75           ; BRANCH IF GOOD
        ERROR    32           ; 11 BITS OF THE 32 BIT ECC REGISTER INCORRECT

75:      JSR      RD, 2#ECCORR   ; GO TO ECC CORRECTION PROCESS
85:      4118.                ; EXPECTED POSITION REG. WHEN CORRECTION
                        ; IS COMPLETE

JSR      PC, 2#CHECKE   ; CHECK THAT DVA, RDY, DPR, DRY = !
TYPE     ,CPHALT       ; CANNOT CONTINUE IF THEY DON'T
HALT     ; STOP THE TEST AND RESTART PROGRAM
MOV      2#WRFROM, RD   ; GETTING READY TO FILL EXPECTED DATA
MOV      2#0!FMT22, (RD)+ ; CYLINDER 0
MOVB     2#0, -(SP)     ; IN LOWER BYTE GET SECTOR
MOVB     2#0, 1(SP)     ; GET TRACK IN HIGHER BYTE
MOV      (SP)+, (RD)+   ; GET TRACK/SECTOR IN BUFFER
MOV      2#0, (RD)+    ; KEY1 IN BUFFER
MOV      2#0, (RD)+    ; KEY2 IN BUFFER
MOV      2#256, R1      ; DATA WORD COUNTER
MOV      2#52525, R2   ; DATA
35:      MOV      R2, (RD)+   ; DATA INTO BUFFER
        DEC      R1         ; COUNT
        BNE     35         ; BRANCH IF 256 NOT DONE

```

```

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

```



```

6517 041676 004737 044652 JSR PC,@#PUTREG ;SAVE REGISTERS
6518 041702 005037 015122 CLR @#ERFLGS ;CLEAR ERROR FLAG
6519
6520
6521 ;NOW READ DATA BUFFER WILL BE CHECKED
6522
6523 041706 004037 046046 JSR RO,@#COMPAR ;CHECK
6524 041712 015212 MRFROM ;GOOD BUFFER
6525 041714 016256 REINTO ;TEST BUFFER
6526 041716 000404 4+256. ;NUMBER OF WORDS CHECKED
6527 041720 041726 4$ ;RETURN POINT FOR ERROR HEADER
6528 041722 041732 5$ ;RETURN POINT FOR ERROR DATA
6529
6530 041724 041736 TST66 ;RETURN FOR GOOD COMPARISON
6531
6532 041726 104004 4$: ERROR 4 ;READ NEXT ERROR
6533 041730 000207 RTS PC ;RETURN TO "COMPAR"
6534 041732 104005 5$: ERROR 5 ;WORD NOS 1 TO 4 ARE
6535 ;HEADER WORDS
6536 ;5 TO 260 ARE DATA WORDS
6537 041734 000207 RTS PC ;RETURN TO "COMPAR"
6538

```

```

6539
6540 ;*****
6541 ;#TEST 66 READ ECC ENABLED 3C
6542
6543 ;* THIS IS AN ECC READ DATA TEST
6544 ;* ERROR CORRECTION IS ENABLED
6545 ;* A NON CORRECTABLE ERROR IS INSERTED IN BIT POSITION 296 THRU 308
6546 ;* THIS IS IN WORD NUMBER 19 AND 20
6547 ;* GOOD DATA USED IS 256 WORDS OF 52525
6548 ;* COMMAND IS GIVEN FOR CYLINDER 0 FORMAT 16 BITS PER WORD
6549 ;* TRACK 0, SECTOR 0 KEYS 0 READ HEADER AND DATA
6550
6551 ;*****
6552 †ST66: SCOPE
6553 041736 000004 MOV #STACK,SP ;RESET STACK
6554 041740 012706 001000 MOV #ATTNO,#TSTNM ;THIS SAVES TEST NUMBER
6555 041744 012737 000066 017322
6556
6557 ;SETUP FOR WHAT IS TO BE READ
6558 ;HEADER CRC IS RESTORED FROM A SUBROUTINE
6559
6560 041752 012746 052525 MOV #52525,-(SP) ;DATA TO BE READ
6561 041756 012705 000400 MOV #256,R5 ;COUNTER
6562 041762 012700 053520 MOV #DISK,RO ;START OF SIMULATED DISK DATA
6563 041766 011620 15: MOV (SP),(RO)+ ;MOVE IN DATA ON TO SIMULATED DISK
6564 041770 005305 DEC R5 ;COUNT
6565 041772 001375 BNE 1$ ;BRANCH IF 256 NOT COMPLETE
6566 041774 005726 TST (SP)+ ;UNDO -(SP)
6567 041776 022020 CMP (RO)+,(RO)+ ;JUMP OVER THE TWO ECC WORDS
6568 042000 012705 000017 MOV #15., R5 ;1 DATA GAP
6569 ;14 TOLERANCE GAP
6570 042004 005020 2$: CLR (RO)+ ;CLEAR DATA GAP, AND
6571 042006 005305 DEC R5 ;TOLERANCE GAP
6572 042010 001375 BNE 2$ ;BRANCH IF NOT COMPLETE
6573
6574
6575 042012 004737 050442 JSR PC,#FILLEC ;INSERT THE TWO ECC WORDS ON THE DISK
6576 ;IN THE CORRECT PLACE
6577
6578 ;THESE ARE FOR ECC TEST ONLY
6579
6580 042016 012737 177777 015140 MOV #-1,#TSECC ;THIS IS AN ECC TEST
6581 042024 005037 047660 CLR #POSITI ;CLEAR ERROR POSITION COUNTER
6582 042030 013737 047654 047656 MOV #NCODE,#NCOUNT ;TEMPORARY N-CODE COUNTER
6583 042036 013737 047662 047670 MOV #HARDER,#HAOTMP ;TEMPORARY HARD ERROR COUNTER
6584 042044 005037 047646 CLR #GECCI ;ECC LOW ORDER TO BE GENERATED
6585 042050 005037 047650 CLR #GECC2 ;ECC HIGH ORDER TO BE GENERATED
6586 042054 005037 047664 CLR #DATENV ;CLEAR DATA ENVELOPE CLOCK COUNT
6587 042060 005037 047666 CLR #ZCODE ;CLEAR LEADING ZEROS CLOCK COUNT
6588
6589
6590 ;THESE ARE TO SETUP FOR DISKLESS USE ONLY
6591
6592 042064 012737 010000 051602 MOV #FMT22,#CYL ;16 BITS PER WORD

```



```

6647 ; IN CHECKING READ DATA THE WRITE FROM BUFFER
6648 ; "WRFROM" IS FILLED WITH EXPECTED DATA AND
6649 ; COMPARISONS ARE MADE
6650
6651 042264 005737 015122 TST @#ERFLG$ ; ANY ERRORS ALREADY THERE
6652 042270 001111 BNE TST67 ; BRANCH IF YES
6653 042272 004737 044652 JSR PC,@#PUTREG ; SAVE REGISTERS
6654 042276 022737 100000 015032 CMP #DCK,@#ER1 ; ONLY DATA CHECK ERROR SHOULD BE SET
6655 042304 001401 BEQ 6$ ; BRANCH IF YES
6656 042306 104032 ERROR 32 ; 32 BIT ECC REGISTER SHOULD BE NON
6657 ; ZERO
6658 ; ONLY 11 OF THE 32 BITS CAN BE SEEN
6659 ; IN THE PATTERN REGISTER
6660 ; DCK SHOULD BE SET IN RHER1
6661 042310 013746 047646 6$: MOV @#GECC1,-(SP) ; GET PATTERN REGISTER
6662 042314 042716 174000 BIC #174000,(SP) ; KEEP ONLY 11 BITS
6663 042320 022637 015062 CMP (SP)+,@#EC2 ; COMPARE PATTERN REGISTER
6664 042324 001401 BEQ 7$ ; BRANCH IF GOOD
6665 042326 104032 ERROR 32 ; 11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6666
6667 042330 004037 050270 7$: JSR R0,@#ECORR ; GO TO ECC CORRECTION PROCESS
6668 042334 000000 8$: .WORD ; EXPECTED POSITION REG. WHEN CORRECTION
6669 ; IS COMPLETE
6670
6671
6672
6673 042336 004737 044652 JSR PC,@#PUTREG ; SAVE REGISTERS
6674 042342 022737 100100 015032 CMP #DCK!ECH,@#ER1 ; WITH ERRORS INSERTED IN BIT POSITION 21
6675 ; THRU 32 HARD ERROR BIT SHOULD SET
6676 042350 001401 BEQ 9$ ; BRANCH IF GOOD
6677 042352 104036 ERROR 36 ; WITH ERROR INSERTED IN BIT POSITION 21 THRU
6678 ; 32 HCE SHOULD SET
6679
6680
6681
6682 042354 9$: JSR PC,@#CHECKE ; CHECK THAT DVA,RDY,DPR,DRY = 1
6683 042354 004737 045376 TYPE ,CPHALT ; CANNOT CONTINUE IF THEY DON'T
6684 042360 104400 005116 HALT ; STOP THE TEST AND RESTART PROGRAM
6685 042364 000000 ; GETTING READY TO FILL EXPECTED DATA
6686 042366 012700 015212 MOV #WRFROM,R0 ; CYLINDER 0
6687 042372 012720 010000 MOV #0!FMT22,(R0)+ ; IN LOWER BYTE GET SECTOR
6688 042376 112746 000000 MOVB #0,-(SP) ; GET TRACK IN HIGHER BYTE
6689 042402 112766 000000 000001 MOVB #0,1(SP) ; GET TRACK/SECTOR IN BUFFER
6690 042410 012620 MOV (SP)+,(R0)+ ; KEY1 IN BUFFER
6691 042412 012720 000000 MOV #0,(R0)+ ; KEY2 IN BUFFER
6692 042416 012720 000000 MOV #0,(R0)+ ; DATA WORD COUNTER
6693 042422 012701 000400 MOV #256,R1 ; DATA
6694 042426 012702 052525 MOV #52525,R2 ; DATA INTO BUFFER
6695 042432 010220 3$: MOV R2,(R0)+ ; COUNT
6696 042434 005301 DEC R1 ; BRANCH IF 256 NOT DONE
6697 042436 001375 BNE 3$
6698
6699 ; ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6700 ; NOW THE INSERTED ERROR WILL BE PUT IN

```

```

6701
6702 042440 012737 152652 015266      MOV      #152652,2#WRFROM+54;INSERT ERROR IN POSITION 296 THRU 304
6703                                     ;IN WORD NUMBER 19 IN DATA
6704 042446 012737 052532 015270      MOV      #52532,2#WRFROM+56;INSERT ERROR IN POSITION 305 THRU 308
6705                                     ;IN WORD NUMBER 20 IN DATA
6706
6707 042454 004737 044652      JSR      PC,2#PUTREG      ;SAVE REGISTERS
6708 042460 005037 015122      CLR      2#ERFLGS        ;CLEAR ERROR FLAG
6709
6710
6711                                     ;NOW READ DATA BUFFER WILL BE CHECKED
6712
6713 042464 004037 046046      JSR      RD,2#COMPAR      ;CHECK
6714 042470 015212      WRFROM      ;GOOD BUFFER
6715 042472 016256      REINTO      ;TEST BUFFER
6716 042474 000404      4+256.      ;NUMBER OF WORDS CHECKED
6717 042476 042504      4$          ;RETURN POINT FOR ERROR HEADER
6718 042500 042510      5$          ;RETURN POINT FOR ERROR DATA
6719
6720 042502 042514      TST67      ;RETURN FOR GOOD COMPARISON
6721
6722 042504 104004      4$: ERROR 4      ;READ NEXT ERROR
6723 042506 000207      RTS PC      ;RETURN TO "COMPAR"
6724 042510 104005      5$: ERROR 5      ;WORD NOS 1 TO 4 ARE
6725                                     ;HEADER WORDS
6726                                     ;5 TO 260 ARE DATA WORDS
6727 042512 000207      RTS PC      ;RETURN TO "COMPAR"
6728

```

6729  
6730  
6731  
6732  
6733  
6734  
6735  
6736  
6737  
6738  
6739  
6740  
6741  
6742 042514 000004  
6743 042516 012706 001000  
6744  
6745 042522 012737 000067 017322  
6746  
6747  
6748  
6749  
6750  
6751 042530 012746 052525  
6752 042534 012705 000400  
6753 042540 012700 053520  
6754 042544 011620  
6755 042546 005305  
6756 042550 001375  
6757 042552 005726  
6758 042554 022020  
6759 042556 012705 000017  
6760  
6761 042562 005020  
6762 042564 005305  
6763 042566 001375  
6764  
6765  
6766 042570 004737 050442  
6767  
6768  
6769  
6770  
6771 042574 012737 177777 015140  
6772 042602 005037 047660  
6773 042606 013737 047654 047656  
6774 042614 013737 047662 047670  
6775 042622 005037 047646  
6776 042626 005037 047650  
6777 042632 005037 047664  
6778 042636 005037 047666  
6779  
6780  
6781  
6782

```

*****
;TEST 67      READ ECC ENABLED 30

```

```

; 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

```

```

*****

```

```

TST67:  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

```

```

6783 042642 012737 010000 051602      MOV      #FMT22, @#CYL      ;16 BITS PER WORD
6784                                     ;CYLINDER 0, FORMAT 16 BITS
6785 042650 112737 000000 051605      MOVB     #0, @#SECOTR+1    ;TRACK 0
6786 042656 112737 000000 051604      MOVB     #0, @#SECOTR      ;SECTOR 0
6787 042664 012737 000000 051606      MOV      #0, @#KEY1        ;KEY1=0
6788 042672 012737 000000 051610      MOV      #0, @#KEY2        ;KEY2=0
6789 042700 012737 000400 051662      MOV      #256., @#DANWORD  ;NO. OF DATA WORDS
6790 042706 005037 051612                CLR      @#X                ;THIS IS A READ COMMAND
6791 042712 004537 046360                JSR      R5, @#CRC          ;GO TO CALCULATE CRC
6792 042716 051602
6793 042720 053502
6794
6795
6796                                     ;THIS IS TO INSERT ERROR
6797                                     ;THE DISK DATA IS IN LOCATION STARTING FROM 'DISK'
6798                                     ;THE POSITION OF THE ERROR CAN BE CHANGED BY CHANGING
6799                                     ;THIS MOVE
6800
6801 042722 012737 152525 053522      MOV      #152525, @#DISK+2 ;FORCE ERROR ON BIT NUMBER 32
6802 042730 012737 152525 054516      MOV      #152525, @#DISK+(255.*2);FORCE ERROR IN BIT 4096
6803 042736 012737 010040 043112      MOV      #4128., @#BS      ;INSERT POSITION REG.
6804
6805
6806                                     ;THESE ARE REGULAR SETUPS
6807
6808 042744 004737 045152                JSR      PC, @#CLDISK      ;SETUP GENERAL REGISTERS
6809 042750 012777 177374 151774      MOV      #-256.-4., @#RHMC ;256. DATA 4 HEADER WORDS
6810 042756 012777 016256 151770      MOV      #REINTO, @#RHB   ;STARTING ADDRESS OF READ BUFFER
6811 042764 112746 000000                MOVB     #0, -(SP)         ;IN LOWER BYTE GET SECTOR
6812 042770 112766 000000 000001      MOVB     #0, 1(SP)        ;GET TRACK IN HIGHER BYTE
6813 042776 012677 151762                MOV      (SP)+, @#RHST    ;TRACK/SECTOR IN RHST
6814 043002 012777 010000 151760      MOV      #FMT22, @#RHOF   ;16 BITS PER WORD
6815                                     ;ECC CORRECTION NOT INHIBIT
6816                                     ;BECAUSE ECC IS NOT GOING
6817                                     ;TO BE CHECKED
6818 043010 005077 151756                CLR      @#RHCA           ;CYLINDER 0
6819
6820 043014 004737 045206                JSR      PC, @#CHECKT     ;CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
6821 043020 104400 005116                TYPE     ,CPHALT         ;CANNOT CONTINUE TESTING IF ANY OF THE
6822                                     ;ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
6823 043024 000000                HALT                    ;STOP THE TEST
6824
6825 043026 013711 015174                MOV      @#REFOR, @#R1    ;READ HEADER AND DATA=72
6826 043032 005037 015122                CLR      @#ERFLG$        ;CLEAR ERROR FLAG
6827 043036 004737 051456                JSR      PC, @#COMHD      ;READ HEADER AND DATA
6828                                     ;IF THERE ARE READ ERRORS THEN
6829                                     ;ECC WILL NOT BE CHECKED
6830
6831
6832                                     ;IF THE PROGRAM COMES BACK HERE WITHOUT ERROR PRINTOUTS
6833                                     ;FROM THE "COMHD" ROUTINE THAT MEANS SECTOR GAP,
6834                                     ;FIRST SYNC, HEADER, HEADER CRC, HEADER GAP AND
6835                                     ;SYNC BYTE HAVE GONE BY AND SYNC'S WERE CORRECTLY
6836                                     ;DETECTED

```

```

6837                                     ;HEADER AND DATA ARE TO BE CHECKED.
6838                                     ;IN CHECKING READ DATA THE WRITE FROM BUFFER
6839                                     ;"WRFROM" IS FILLED WITH EXPECTED DATA AND
6840                                     ;COMPARISONS ARE MADE
6841
6842 043042 005737 015122                TST      @#ERFLGS      ;ANY ERRORS ALREADY THERE
6843 043046 001111                        BNE     TST70         ;BRANCH IF YES
6844 043050 004737 044652                JSR     PC,@#PUTREG  ;SAVE REGISTERS
6845 043054 022737 100000 015032        CMP     @DCK,@#ER1  ;ONLY DATA CHECK ERROR SHOULD BE SET
6846 043062 001401                        BEQ     6$           ;BRANCH IF YES
6847 043064 104032                        ERROR   32           ;32 BIT ECC REGISTER SHOULD BE NON
6848                                     ;ZERO
6849                                     ;ONLY 11 OF THE 32 BITS CAN BE SEEN
6850                                     ;IN THE PATTERN REGISTER
6851                                     ;DCK SHOULD BE SET IN RHER1
6852 043066 013746 047646                6$:    MOV     @#GECC1,-(SP) ;GET PATTERN REGISTER
6853 043072 042716 174000                BIC     @174000,(SP) ;KEEP ONLY 11 BITS
6854 043076 022637 015062                CMP     (SP)+,@#EC2 ;COMPARE PATTERN REGISTER
6855 043102 001401                        BEQ     7$           ;BRANCH IF GOOD
6856 043104 104032                        ERROR   32           ;11 BITS OF THE 32 BIT ECC REGISTER INCORRECT
6857
6858 043106 004037 050270                7$:    JSR     RD,@#ECORR ;GO TO ECC CORRECTION PROCESS
6859 043112 000000                        8$:    .WORD   ;EXPECTED POSITION REG. WHEN CORRECTION
6860                                     ;IS COMPLETE
6861
6862
6863
6864 043114 004737 044652                JSR     PC,@#PUTREG  ;SAVE REGISTERS
6865 043120 022737 100100 015032        CMP     @DCK!ECH,@#ER1 ;WITH ERRORS INSERTED IN BIT POSITION 32
6866                                     ;AND 4096 HARD ERROR BIT SHOULD SET
6867 043126 001401                        BEQ     9$           ;BRANCH IF GOOD
6868 043130 104036                        ERROR   36           ;WITH ERROR INSERTED IN BIT POSITION 21 THRU
6869                                     ;32 HCE SHOULD SET
6870
6871
6872
6873
6874 043132                                9$:
6875 043132 004737 045376                JSR     PC,@#CHECKE ;CHECK THAT DVA,RDY,DPR,DRY = 1
6876 043136 104400 005116                TYPE   ,CPHALT     ;CANNOT CONTINUE IF THEY DON'T
6877 043142 000000                        HALT                               ;STOP THE TEST AND RESTART PROGRAM
6878 043144 012700 015212                MOV     @WRFROM,RO  ;GETTING READY TO FILL EXPECTED DATA
6879 043150 012720 010000                MOV     @0!FMT22,(RO)+ ;CYLINDER 0
6880 043154 112746 000000                MOVB   @0,-(SP)    ;IN LOWER BYTE GET SECTOR
6881 043160 112766 000000 000001        MOVB   @0,1(SP)    ;GET TRACK IN HIGHER BYTE
6882 043166 012620                        MOV     (SP)+,(RO)+ ;GET TRACK/SECTOR IN BUFFER
6883 043170 012720 000000                MOV     @0,(RO)+   ;KEY1 IN BUFFER
6884 043174 012720 000000                MOV     @0,(RO)+   ;KEY2 IN BUFFER
6885 043200 012701 000400                MOV     @256,R1    ;DATA WORD COUNTER
6886 043204 012702 052525                MOV     @52525,R2  ;DATA
6887 043210 010220                        3$:    MOV     R2,(RO)+   ;DATA INTO BUFFER
6888 043212 005301                        DEC     R1          ;COUNT
6889 043214 001375                        BNE     3$         ;BRANCH IF 256 NOT DONE
6890

```



M16

```

6891 ; ONLY GOOD DATA HAS BEEN PUT IN 'WRFROM'
6892 ; NOW THE INSERTED ERROR WILL BE PUT IN
6893
6894 043216 012737 152525 015224 MOV #152525, @WRFROM+(5*2) ; INSERTED ERROR IN BIT 32
6895 043224 012737 152525 016220 MOV #152525, @WRFROM+(259.*2) ; INSERT ERROR IN BIT 4096
6896
6897 043232 005037 015122 CLR @ERFLGS ; CLEAR ERROR FLAG
6898 043236 004737 044652 JSR PC, @PUTREG ; SAVE REGISTERS
6899
6900 ; NOW READ DATA BUFFER WILL BE CHECKED
6901
6902 043242 004037 046046 JSR RD, @COMPAR ; CHECK
6903 043246 015212 WRFROM ; GOOD BUFFER (CHANGED)
6904 043250 016256 REINTO ; TEST BUFFER
6905 043252 000404 4+256. ; NUMBER OF WORDS CHECKED
6906 043254 043262 4$ ; RETURN POINT FOR ERROR HEADER
6907 043256 043266 5$ ; RETURN POINT FOR ERROR DATA
6908
6909 043260 043272 TST70 ; RETURN FOR GOOD CO:PARISON
6910
6911 043262 104004 4$: ERROR 4 ; READ NEXT ERROR
6912 043264 000207 RTS PC ; RETURN TO "COMPAR"
6913 043266 104005 5$: ERROR 5 ; WORD NOS 1 TO 4 ARE
6914 ; HEADER WORDS
6915 ; 5 TO 260 ARE DATA WORDS
6916 043270 000207 RTS PC ; RETURN TO "COMPAR"

```

.SBTTL CURSORY INTERRUPT LOGIC TESTS

6917  
6918  
6919  
6920  
6921  
6922  
6923  
6924  
6925  
6926  
6927  
6928  
6929  
6930  
6931  
6932  
6933  
6934  
6935  
6936  
6937  
6938  
6939  
6940  
6941  
6942  
6943  
6944  
6945  
6946  
6947  
6948  
6949  
6950  
6951  
6952  
6953  
6954  
6955

043272 000004  
043274 012737 000070 017322  
043302 012706 001000  
043306 004737 045152  
043312 013700 014746  
043316 012720 043364  
043322 012710 000340  
043326 012767 000200 134442  
043334 012711 000300  
043340 013737 045522 001172  
043346 005337 001172  
043352 001375  
043354 004737 044652  
043360 104021  
043362 000410  
043364 022626  
043366 004737 044652  
043372 022737 004200 015030  
043400 001401  
043402 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 #RPTVEC,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
IS: DEC #STMP1 ;WAIT FOR INTERRUPT
BNE IS ;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 2! ;INTERRUPT OCCURRED BUT
;'IE' FAILED TO RESET
```



```

6991
6992
6993
6994
6995
6996
6997
6998
6999
7000
7001 043500 000004
7002 043502 012767 000001 135474
7003 043510 004737 045152
7004 043514 012767 000000 134254
7005 043522 104400 043530
7006 043526 000425
7007
7008 043602
7009 043602 013746 015110
7010 043606 104410
7011 043610 104400 043616
7012 043614 000402
7013
7014 043622
7015 043622 013746 001112
7016 043626 104410
7017 043630 005037 001112
7018 043634 005737 015116
7019 043640 001415
7020 043642 005067 135234
7021 043646 005237 001100
7022 043652 104400 044050
7023 043656 013746 001100
7024 043662 104410
7025 043664 104400 044065
7026 043670 000137 022440
7027 043674 005337 015112
7028 043700 001413
7029 043702 013700 015110
7030 043706 012701 015070
7031 043712 022100
7032 043714 001401
7033 043716 000775
7034 043720 011137 015110
7035 043724 000137 022440

```

```

;*****
;*****
;*TEST 72      END OF DRIVE

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

;*****
†ST72: SCOPE
MOV      #1, $TIMES      ;;DO 1 ITERATION
JSR      PC, @#CLDISK
MOV      #0, PS          ;REINSTATE PS TO 0
TYPE     #+4             ;;TYPE ASCIZ STRING
BR       64$            ;;GET OVER THE ASCIZ
;::.ASCIZ      <15><12>/TOTAL ERRORS ON THIS PASS ON UNIT NO. /

64$: MOV      @#UNIT, -(SP) ;GET READY TO TYPE UNIT NUMBER
      TYPDS
      TYPE     #+4             ;;TYPE ASCIZ STRING
      BR       65$            ;;GET OVER THE ASCIZ
;::.ASCIZ      /= /

65$: MOV      @#$ERTTL, -(SP) ;GET READY TO TYPE NUMBER OF ERRORS
      TYPDS
      CLR      @#$ERTTL      ;CLEAR TOTAL NUMBER OF ERRORS
      TST      @#$SELECT    ;STARTING FROM 200
      BEQ      3$           ;BRANCH IF YES
      CLR      $TSTNM       ;CLEAR TEST NUMBER
      INC      @#$PASS      ;INCREASE PASS COUNT
      TYPE     $SENDMG      ;TYPE END PASS
      MOV      @#$PASS, -(SP)

      JMP      @#TST5
3$: DEC      @#NOUNITS      ;NO. OF UNITS PRESENT DECREMENT
      BEQ      $EOP        ;BRANCH IF ALL DRIVES COMPLETE
      MOV      @#UNIT, R0   ;UNIT UNDER TEST
      MOV      @#UNITS, R1  ;TABLE
1$: CMP      (R1)+, R0      ;IS THIS UNIT JUST TESTED
      BEQ      2$          ;BRANCH IF YES
      BR       1$          ;BRANCH IF NO
2$: MOV      (R1), @#UNIT   ;THIS IS NEXT UNIT
      JMP      @#TST5      ;GO FOR NEXT TESTS.

```

7036  
7037  
7038  
7039  
7040  
7041  
7042  
7043  
7044  
7045  
7046  
7047  
7048  
7049  
7050  
7051  
7052  
7053  
7054  
7055  
7056  
7057  
7058  
7059  
7060  
7061  
7062  
7063  
7064  
7065  
7066  
7067  
7068  
7069  
7070  
7071  
7072  
7073  
7074  
7075  
7076  
7077  
7078  
7079  
7080  
7081  
7082  
7083  
7084  
7085  
7086  
7087  
7088  
7089

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

\*\*\*\*\*

.SBTTL END OF PASS ROUTINE

;\*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

SEOP:

SCOPE CLR \$TSTNM ;; ZERO THE TEST NUMBER  
CLR \$TIMES ;; ZERO THE NUMBER OF ITERATIONS  
INC \$PASS ;; INCREMENT THE PASS NUMBER  
BIC #10000, \$PASS ;; DON'T ALLOW A NEG. NUMBER  
DEC (PC)+ ;; LOOP?

SEOPCT: .WORD 1

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

SENDCT: .WORD 1

SENDMG \$PASS, -(SP) ;; TYPE "END PASS #"  
MOV \$PASS, -(SP) ;; SAVE \$PASS FOR TYPEOUT  
TYPDS TYPE ;; GO TYPE--DECIMAL ASCII WITH SIGN  
TYPE \$NULL ;; TYPE A NULL CHARACTER

SGET42: MOV #42, R0

\$DOAGN \$DOAGN ;; GET MONITOR ADDRESS  
BEQ \$SENDAD, R0 ;; BRANCH IF NO MONITOR  
CMP #SENDAD, R0 ;; IS MONITOR ACT11?  
BNE \$RESET ;; NO--BRANCH (IT'S X00P)  
CMP #-1, 2(R0) ;; YES--IS THIS THE LAST PASS?  
BNE \$SENDAD ;; NO--MAKE ANOTHER PASS

\$RESET: RESET

\$SENDAD: JSR PC, (R0) ;; CLEAR THE WORLD  
NOP ;; GO TO MONITOR  
NOP ;; SAVE ROOM  
NOP ;; FOR

\$DOAGN: JMP #TST1

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

\$SENDMG: .ASCIZ <15><12>/END PASS #/

SENULL: .BYTE -1, -1, 0 ;; NULL CHARACTER STRING

```

7090
7091
7092
7093
7094
7095
7096
7097
7098
7099
7100
7101
7102
7103
7104
7105
7106
7107
7108
7109
7110
7111
7112 044070 000000
7113
7114 044072
7115 044072 005067 133700
7116 044076 104400 044104
7117 044102 000421
7118
7119 044146
7120 044146 013746 017322
7121 044152 104402
7122 044154 104400 044162
7123 044160 000414
7124
7125 044212
7126 044212 013746 001110
7127 044216 104402
7128 044220 104400 001215
7129 044224 104400 044232
7130 044230 000426
7131
7132 044306
7133 044306 104400 044314
7134 044312 000420
7135
7136 044354
7137 044354 104400 044362
7138 044360 000423
7139
7140 044430
7141 044430 104416
7142 044432 062716 000002
7143 044436 012637 001106

```

```

;*****
;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
;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 ,+4 ;TYPE ASCIZ STRING
BR 64$ ;GET OVER THE ASCIZ
;;.ASCIZ <15><12>/THE PROGRAM WAS IN TEST NUMBER /

64$:
MOV @#TSTNM,-(SP) ;GET READY TO TYPE TEST
TYPOC ;NUMBER
TYPE ,+4 ;TYPE ASCIZ STRING
BR 65$ ;GET OVER THE ASCIZ
;;.ASCIZ <15><12>/THE LOOP BACK PC WAS /

65$:
MOV @#SLPERR,-(SP) ;GET READY TO TYPE LOOP BACK PC
TYPOC
TYPE ,SCLF
TYPE ,+4 ;TYPE ASCIZ STRING
BR 66$ ;GET OVER THE ASCIZ
;;.ASCIZ <15><12>/SET LOOP ON ERROR OR LOOP ON TEST SWITCH/

66$:
TYPE ,+4 ;TYPE ASCIZ STRING
BR 67$ ;GET OVER THE ASCIZ
;;.ASCIZ <15><12>/TYPE THE FIRST PC OF THE TEST/

67$:
TYPE ,+4 ;TYPE ASCIZ STRING
BR 68$ ;GET OVER THE ASCIZ
;;.ASCIZ <15><12>/ FOLLOWED BY A CARRIAGE RETURN /<15><12>

68$:
RDOCT
ADD #2,(SP) ;GET LPADR
MOV (SP)+,@#SLPADR

```

# GO1

7144	044442	104400	044450		TYPE	+4	::TYPE ASCIZ STRING
7145	044446	000417			BR	69\$	::GET OVER THE ASCIZ
7146					::.ASCIZ		<15><12>/TYPE THE PC WHERE YOU WANT/
7147	044506			69\$:			
7148	044506	104400	044514		TYPE	+4	::TYPE ASCIZ STRING
7149	044512	000441			BR	70\$	::GET OVER THE ASCIZ
7150					::.ASCIZ		<15><12>/ THE PROGRAM TO LOOP BACK TO FOLLOWED BY A CARRIAGE RETUR
7151	044616			70\$:			
7152	044616	104416			RDOCT		
7153	044620	012637	001110		MOV	(SP)+, @#SLPERR	;GET LPERR
7154	044624	013746	001106		MOV	@#SLPADR -(SP)	
7155					:THIS	CLEAR UP GARBAGE	
7156	044630	005037	051716		CLR	@#NOSYNC	;CLEAR FLAG FOR HEADER ERROR COMMANDS
7157	044634	005037	015140		CLR	@#TSECC	;CLEAR FLAG FOR ECC TEST
7158							;WHEN =177777 IT IS AN ECC TEST
7159							;WHEN =0 IT IS NOT AN ECC TEST
7160							
7161	044640	005037	047652		CLR	@#TSECCG	;EVEN IN AN ECC TEST EVERY CLOCK
7162							;IS NOT TO GENERATE ECC
7163							;IF =177777 GENERATE ECC
7164							;IF =0 DO NOT GENERATE ECC
7165	044644	005037	015142		CLR	@#TESDTE	;DRIVE TIMING ERROR TEST
7166	044650	000002			RTI		

7167  
7168  
7169  
7170  
7171  
7172  
7173  
7174  
7175  
7176  
7177  
7178  
7179  
7180  
7181  
7182  
7183  
7184  
7185  
7186  
7187  
7188  
7189  
7190  
7191  
7192  
7193

044652  
044652 010046  
044654 010146  
044656 010246  
044660 012700 014752  
044664 012701 015022  
044670 012702 000023  
044674 013021  
044676 005302  
044700 001375  
044702 012602  
044704 012601  
044706 012600  
044710 000207

.SBTTL SAVE REGISTERS ROUTINE

\*\*\*\*\*  
: THIS SAVES THE CONTENTS OF ALL HARDWARE REGISTERS  
: IN MEMORY LOCATIONS TAGED FROM "MC" 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    #RHMC,R0      ;; STARTING ADDRESS OF REG
        MOV    #MC,R1        ;; STARTING ADDRESS OF SAVE LOCATIONS
        MOV    #RHCC-RHMC+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
```



```

7194
7195
7196
7197
7198
7199
7200
7201
7202 044712 000000
7203 044714 000000
7204 044716 000000
7205
7206 044720 012567 177766
7207 044724 012504
7208 044726 010467 177764
7209 044732 010567 177756
7210 044736 062705 000004
7211 044742 012703 000001
7212 044746 004767 000016
7213 044752 004767 000012
7214 044756 000241
7215 044760 006103
7216 044762 005703
7217 044764 001370
7218 044766 000205
7219 044770 005103
7220 044772 012737 045000 045132
7221 045000 010337 001124
7222 045004 005137 044712
7223 045010 043737 044712 001124
7224 045016 005137 044712
7225 045022 013714 001124
7226 045026 011437 001126
7227 045032 005137 044712
7228 045036 043737 044712 001126
7229 045044 005137 044712
7230 045050 023737 001124 001126
7231 045056 001403
7232 045060 004777 177630
7233 045064 104420
7234 045066 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
R3
TST R3
BNE BLT1 ;BRANCH IF NOT COMPLETE
RTS ;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
SCOPI ;LOCAL SCOPE LOOP
IS: RTS PC

```

7235  
7236  
7237  
7238  
7239  
7240  
7241  
7242  
7243  
7244  
7245  
7246  
7247  
7248  
7249  
7250  
7251  
7252  
7253  
7254  
7255  
7256  
7257  
7258  
7259  
7260  
7261  
7262  
7263  
7264  
7265  
7266  
7267  
7268  
7269  
727C

.SBTTL CLEAR MEMORY ROUTINE

```

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

```

```

045070
045070 010146
045072 010246
045074 010346
045076 012001
045100 012002
045102 012003
045104 160102
045106 062702 000002
045112 010321
045114 005302
045116 005302
045120 001374
045122 012603
045124 012602
045126 012601
045130 000200

```

```

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
15:      MOV      R3, (R1)+ ;; MOVE IN DATA
DEC      R2
DEC      R2
BNE      15             ;; 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

```

```

7271          .SBTTL LOCAL TRAPS
7272 045132 000000          LAD:      0
7273
7274 045134 032737 001000 177570 T.SCOF: BIT      @SW09, @SWR
7275 045142 001402          BEQ      15
7276 045144 013716 045132          MOV      @LAD, (SP)
7277 045150 000002          IS:      RTI
7278
7279          ;*EXAMPLE OF THE USE OF THE ABOVE
7280          ;*THIS WILL LOOP BETWEEN X: AND SCOP1 PROVIDED THERE IS NO "NEWST"
7281          ;*MOV      @X, @LAD
7282          ;*X:      ---
7283          ;*      ---
7284          ;*      ---
7285          ;*      SCOP1
7286
7287          .SBTTL CLEAR DISK ROUTINE
7288
7289          CLDISK: MOV      @RHCS1,R1      ;R1 WILL BE CONTROL AND STATUS1
7290 045152 013701 014760          MOV      @RHCS2,R2      ;R2 WILL BE CONTROL AND STATUS2
7291 045156 013702 014756          MOV      @RHDS1,R3      ;R3 WILL BE DISK STATUS REGISTER1
7292 045162 013703 015002          MOV      @RHER1,R4      ;R4 WILL BE ERROR REGISTER #1
7293 045166 013704 014762
7294
7295          MOV      @CLR,@R2      ;CLEAR ALL REG.
7296 045172 012712 000040          MOV      @UNIT,@R2      ;REINSTATE UNIT NO.
7297 045176 013712 015110          CLR      @R1      ;CLEAR FUNCTION BITS
7298 045202 005011          RTS      PC
7299 045204 000207
    
```

7300  
7301  
7302  
7303  
7304  
7305  
7306  
7307  
7308  
7309  
7310  
7311  
7312  
7313  
7314  
7315  
7316  
7317  
7318  
7319  
7320  
7321  
7322  
7323  
7324  
7325  
7326  
7327  
7328  
7329  
7330  
7331  
7332  
7333  
7334  
7335  
7336  
7337  
7338  
7339  
7340  
7341  
7342  
7343  
7344  
7345  
7346  
7347  
7348  
7349  
7350  
7351  
7352  
7353

.SBTTL CHECK DISK STATUS ROUTINES

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

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  
;AND BE READY  
BEQ 3\$ ;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  
3\$: 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  
4\$: 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  
5\$: 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  
6\$: MOV R3,@#SBDADR ;ADDRESS OF BAD REGISTER (RHDS1)  
ERROR 26 ;RHDS1 HAS SOME BITS OTHER  
;THAN MCL, DRY, DPR, SET

# MO1

```

7354                                     ;ALL OTHER BITS SHOULD BE 0
7355 045366 000207                       7$:   RTS   PC   ;RETURN TO TEST AND HALT - FATAL ERROR
7356
7357 045370 062716 000006                 8$:   ADD   #6,(SP) ;ADJUST STACK PTR TO GET OVER HALT IN TEST
7358 045374 000207                       RTS   PC   ;RETURN TO TEST AND CONTINUE TESTING
7359
7360
7361                                     ;*****
7362                                     ;*THIS CHECKS THAT DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1 = 1
7363                                     ;*AND CHECKS THAT DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1 = 1
7364                                     ;*IT DOES NOT CHECK THAT OTHER BITS IN THESE REGISTERS = 0
7365                                     ;*****
7366
7367 045376 011637 015130                 CHECKE: MOV   (SP),@#PCJSR ;SAVE PC OF JSR+4
7368 045402 162737 000004 015130         SUB   #4,@#PCJSR ;GET PC OF JSR
7369 045410 004737 044652                 JSR   PC,@#PUTREG ;SAVE REGISTERS
7370 045414 032737 000200 015030         BIT   #RDY,@#CS1 ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
7371                                     ;AND BE READY
7372 045422 001004                         BNE   1$ ;BRANCH IF GOOD
7373 045424 010137 001122                 MOV   R1,@#SBDADR ;FAILING REGISTER
7374 045430 104026                         ERROR 26 ;RHCS1 IS IN ERROR
7375                                     ;DOES NOT HAVE DVA, RDY
7376 045432 000427                         BR    4$ ;BRANCH
7377
7378 045434 032737 004000 015030 1$:   BIT   #DVA,@#CS1 ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
7379                                     ;AND BE READY
7380 045442 001004                         BNE   2$ ;BRANCH IF GOOD
7381 045444 010137 001122                 MOV   R1,@#SBDADR ;FAILING REGISTER
7382 045450 104026                         ERROR 26 ;RHCS1 IS IN ERROR
7383                                     ;DOES NOT HAVE DVA, RDY
7384 045452 000417                         BR    4$ ;BRANCH OUT
7385 045454 032737 000200 015052 2$:   BIT   #DRY,@#DS1 ;RHDS1 SHOULD HAVE DPR DRY
7386 045462 001004                         BNE   3$ ;BRANCH IF THERE
7387 045464 010337 001122                 MOV   R3,@#SBDADR ;FAILING REGISTER RHDS1
7388 045470 104026                         ERROR 26 ;RHDS1 DOES NOT HAVE DPR, DRY
7389 045472 000407                         BR    4$ ;BRANCH OUT
7390 045474 032737 000400 015052 3$:   BIT   #DPR,@#DS1 ;RHDS1 SHOULD HAVE DPR, DRY
7391 045502 001004                         BNE   5$ ;BRANCH IF THERE
7392 045504 010337 001122                 MOV   R3,@#SBDADR ;FAILING REGISTER RHDS1
7393 045510 104026                         ERROR 26 ;RHDS1 DOES NOT HAVE DPR, DRY
7394 045512 000207                       4$:   RTS   PC   ;RETURN TO TEST AND HALT - FATAL ERROR
7395
7396 045514 062716 000006                 5$:   ADD   #6,(SP) ;ADJUST STACK TO GET OVER HALT IN TEST
7397 045520 000207                       RTS   PC   ;RETURN TO TEST AND CONTINUE TESTING

```

```

7398
7399
7400 .SBTTL WAIT LOOP
7401 ;*****
7402 ;* WAIT LOOP
7403 ;* ONE LOOP OR ONE COUNT = 5.15 MICROSEC WITH BIPOLAR MEMORY (MIN)
7404 ;* ONE LOOP OR ONE COUNT = 11.86 MICROSEC WITH CORE (MIN)
7405 ;* WITH CORE ERROR IS INDICATED AFTER ABOUT 650 MILLISEC (MIN)
7406 ;*****
7407 045522 177777 TIMCNT: 177777 ;WAITING COUNT
7408
7409 045524 010046 WAIT.T: MOV RO, -(SP) ;SAVE RO
7410 045526 016600 000002 MOV 2(SP), RO ;GET ADDRESS OF REG. ADDRESS
7411 045532 010037 001176 MOV RO, @#STMP3 ;WAT PC+2 IN STMP3
7412 045536 162737 000002 001176 SUB #2, @#STMP3 ;WAT PC FOR TIMEOUT
7413 045544 012037 001170 MOV (RO)+, @#STMP0 ;WAIT REGISTER ADDRESS
7414 045550 012037 001172 MOV (RO)+, @#STMP1 ;WAIT ON BIT
7415 045554 010066 000002 MOV RO, 2(SP) ;RESTORE RETURN ON STACK
7416 045560 012600 MOV (SP)+, RO ;RESTORE RO
7417 045562 013737 045522 001174 MOV @#TIMCNT, @#STMP2 ;TEMPORARY COUNT
7418
7419 045570 033777 001172 133372 15: BIT @#STMP1, @#STMP0 ;IS REQUIRED BIT THERE?
7420 045576 001021 BNE 25 ;BRANCH IF YES
7421 045600 005337 001174 DEC @#STMP2 ;COUNT
7422 045604 001371 BNE 15 ;BRANCH IF NOT TIME UP
7423 045606 013737 045522 001174 MOV @#TIMCNT, @#STMP2 ;TEMPORARY COUNT
7424 045614 033777 001172 133346 35: BIT @#STMP1, @#STMP0 ;IS REQUIRED BIT THERE?
7425 045622 001007 BNE 25 ;BRANCH IF YES
7426 045624 005337 001174 DEC @#STMP2 ;COUNT
7427 045630 001371 BNE 35 ;BRANCH IF NOT TIME UP
7428 045632 017737 133332 001126 MOV @#STMP0, @#SBDDAT ;REGISTER CONTENTS
7429 045640 104016 ERROR 16 ;WAITED ON BIT FAILED TO SET
7430 045642 000002 25: RTI
7431
7432
7433 ;* CALL FOR THE ABOVE WAITLOOP IS
7434 ;*
7435 ;* MOV @A, @#XS ;A CONTAINS REGISTER ADDRESS
7436 ;* - - - ;HENCE XS WILL HAVE ABSOLUTE REG. ADR.
7437 ;* - - -
7438 ;*
7439 ;* WAT
7440 ;*XS: 0 ;ABSOLUTE REG. ADDRESS UNDER WAIT
7441 ;* .WORD 0 ;BIT WAITED FOR
7442 ;* ;CONTINUE

```

7443  
7444  
7445  
7446  
7447  
7448  
7449  
7450  
7451  
7452  
7453  
7454  
7455  
7456  
7457  
7458  
7459  
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  
7496

045644  
045644 010146  
045646 010246  
045650 010346  
045652 012001  
045654 012002  
045656 012003  
045660 013122  
045662 005303  
045664 001375  
045666 012603  
045670 012602  
045672 012601  
045674 000200

.SBTTL SAVE ROUTINE

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

```
SAVER:
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 ;: NUMBER
IS: MOV @R1+, (R2)+ ;: SAVE REGISTER CONTENTS
DEC R3 ;: COUNT
BNE IS ;: BRANCH IF NOT DONE
MOV (SP)+, R3 ;: POP STACK INTO R3
MOV (SP)+, R2 ;: POP STACK INTO R2
MOV (SP)+, R1 ;: POP STACK INTO R1
RTS R0
```

.SBTTL WRITE CHECK ROUTINE

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

; THESE ARE TO SET UP FOR DISKLESS USE ONLY

012737 010000 051602  
112737 000001 051605  
112737 000001 051604  
005037 051606  
005037 051610  
012767 000044 003724  
005037 051612  
004537 046360  
051602  
053502

```
MRCHHD: MOV #FMT22, @CYL ;: CYLINDER 0 FORMAT 16 BIT WORDS
MOV @1, @SECOTR+1 ;: TRACK=1
MOV @1, @SECOTR ;: SECTOR=1
CLR @KEY1 ;: KEY1=0
CLR @KEY2 ;: KEY2=0
MOV @36., @DANORD ;: NO OF DATA WORDS
CLR @X ;: THIS IS A READ OPERATION
JSR R5, @CRC ;: GO TO CALCULATE CRC
CYL
MCRC
```

; THESE ARE REGULAR SETUPS

```
JSR PC, @CLDISK ;: SET UP GENERAL REGISTERS
```





```

7517
7518
7519
7520
7521
7522
7523
7524
7525
7526
7527
7528
7529
7530 046046
7531 046046 010146
7532 046050 010246
7533 046052 010346
7534 046054 010446
7535 046056 010546
7536 046060 012001
7537 046062 012002
7538 046064 012003
7539 046066 012067 133076
7540 046072 012067 133074
7541 046076 011000
7542 046100 010304
7543 046102 005204
7544 046104 010437 051722
7545 046110 022122
7546 046112 001426
7547
7548 046114 014137 001124
7549 046120 014237 001126
7550 046124 160337 051722
7551 046130 005737 015122
7552 046134 001003
7553 046136 004777 133026
7554 046142 000402
7555 046144 004777 133022
7556 046150 022122
7557 046152 013746 177570
7558 046156 042716 177177
7559 046162 022726 000200
7560 046166 001402
7561 046170 005303
7562 046172 001344
7563 046174
7564 046174 012605
7565 046176 012604
7566 046200 012603
7567 046202 012602
7568 046204 012601
7569 046206 000200

```

.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
*STMP0 HAS ADDRESS OF RETURN ON ERROR TO PRINT HEADER
*STMP1 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), @SGDDAT ;: GOOD DATA
MOV -(R2), @SBDDAT ;: BAD DATA
SUB R3, @ERWORD ;: ERROR WORD NO.
TST @ERFLGS ;: ANY ERRORS ALREADY THERE
BNE 2$ ;: BRANCH IF YES
JSR PC, @STMP0 ;: RETURN TO PRINT HEADER
BR 5$ ;: BRANCH TO AVOID PRINTING NEXT ERROR
2$: JSR PC, @STMP1 ;: RETURN TO PRINT DATA
5$: CMP (R1)+, (R2)+ ;: UNDO -(R1) AND -(R2) FOR ERRORS
MOV @SWR, -(SP) ;: GET SWITCH SETTING
BIC #C600, (SP) ;: KEEP ONLY SWITCH 7 AND 8
CMP #SM07, (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

```

```

7570          .SBTTL WRITE CHECK DATA
7571
7572          ;*****
7573          ; THIS IS A SUBROUTINE TO DO WRITE CHECK DATA
7574          ; CYLINDER 0, TRACK 1, SECTOR 1, KEYS 0
7575          ;*****
7576
7577
7578          ; THESE ARE TO SET UP FOR DISKLESS USE ONLY
7579
7580 046210 012737 010000 051602 MRCHDA: MOV      #FMT22, @#CYL      ; CYLINDER 0 FORMAT 16 BIT WORDS
7581 046216 112737 000001 051605      MOV      #1, @#SECOTR+1      ; TRACK=1
7582 046224 112737 000001 051604      MOV      #1, @#SECOTR      ; SECTOR=1
7583 046232 005037 051606      CLR      @#KEY1      ; KEY1=0
7584 046236 005037 051610      CLR      @#KEY2      ; KEY2=0
7585 046242 012737 000040 051652      MOV      #32., @#DADWORD      ; NO OF DATA WORDS
7586 046250 005037 051612      CLR      @#X      ; THIS IS A READ OPERATION
7587
7588 046254 004537 046360      JSR      RS, @#CRC      ; GO TO CALCULATE CRC
7589 046260 051602
7590 046262 053502
7591
7592          ; THESE ARE REGULAR SETUPS
7593
7594 046264 004737 045152      JSR      PC, @#CLDISK      ; SET UP GENERAL REGISTERS
7595          ; AND CLEAR DISK REGISTERS
7596
7597 046270 012777 177740 146454      MOV      #-32., @#RHMC      ; 36 DATA WORDS 4 HEADER WORDS
7598 046276 012777 016256 146450      MOV      #REINTO, @#RHBA      ; STARTING ADDRESS OF READ BUFFER
7599 046304 112746 000001      MOV      #1, -(SP)      ; SECTOR=1
7600 046310 112766 000001 000001      MOV      #1, 1(SP)      ; TRACK=1 IN UPPER BYTE
7601 046316 012677 146442      MOV      (SP)+, @#RHST      ; TRACK=1, SECTOR=1 IN RHST
7602 046322 012777 014000 146440      MOV      #FMT22!ECI, @#RHOF      ; 16 BIT WORDS
7603          ; ECC CORRECTION INHIBIT BECAUSE
7604          ; ECC LOGIC IS NOT CHECKED YET
7605 046330 005077 146436      CLR      @#RHCA      ; CYLINDER=0
7606 046334 004737 045206      JSR      PC, @#CHECKT      ; CHECK DVA, RDY, DPR, DRY = 1 AND OTHERS DON'T
7607 046340 104400 005116      TYPE      .CPHALT      ; CAN'T CONTINUE TESTING IF ANY OF THE
7608          ; ABOVE BITS DON'T = 1 OR OTHERS ARE STUCK @ 1
7609          ; STOP THE TEST
7610 046346 013711 015162      MOV      @#MRCHK, @#R1      ; WRITE CHECK DATA=50 INTO RHCS1
7611 046352 004737 051456      JSR      PC, @#COMHD      ; WRITE CHECK HEADER AND DATA
7612          ; SAME AS READ HEADER AND DATA
7613
7614 046356 000207      RTS      PC      ; RETURN TO WRITE CHECK TEST
7615

```

.SBTTL CRC GENERATION ROUTINE

7616  
7617  
7618  
7619  
7620  
7621  
7622  
7623  
7624  
7625  
7626  
7627  
7628  
7629  
7630  
7631  
7632  
7633  
7634  
7635  
7636  
7637  
7638  
7639  
7640  
7641  
7642  
7643  
7644  
7645  
7646  
7647  
7648  
7649  
7650  
7651  
7652  
7653  
7654  
7655  
7656  
7657  
7658  
7659  
7660  
7661  
7662  
7663  
7664  
7665  
7666  
7667  
7668  
7669

```

;*****
; THIS IS A SUBROUTINE TO CALCULATE CRC FOR THE FOUR
; HEADER WORDS AND STORE THEM IN "MCRC" 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 MCRC FOR READ GCRC FOR WRITE
;*****

```

```

CRC:
046360 010046
046360 012500
046362 010146
046364 010246
046366 010346
046370 010446
046372 005001
046374 005037 001202
046376 012737 000004 001170
046402 012037 001176
046410 012767 000020 132552
046414 013737 001176 001200
046422 006037 001176
046430 006037 001202
046434 032701 000001
046440 001403
046444 012703 100000
046446 000401
046452 005003
046454 063703 001202
046456
046462 032701 040000
046466 001403

```

```

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 ;: WORD COUNT
MOV #4,@STMP0 ;: TEMPORARY WORD STORAGE
15: MOV (R0)+,@STMP3 ;: BIT COUNT
MOV @16,@STMP2 ;: TEMPORARY WORD STORAGE
MOV @STMP3,@STMP4 ;: GET LSB INTO "C"
15: ROR @STMP3 ;: GET ABOVE "C" INTO STMP5
ROR @STMP5 ;: IS POSITION 15 HIGH
BIT @BIT0,R1 ;: BRANCH IF POSITION 16 LOW
BEQ 15 ;: GET POSITION 16
MOV @BIT15,R3 ;: XOR POSITION 16 WITH D
BR 25 ;: TO GIVE B
15: CLR R3 ;: IS POSITION 2 HIGH
25: ADD @STMP5,R3 ;: BRANCH IF POSITION 2 LOW
BIT @BIT14,R1
BEQ 35

```

7670	046470	012702	100000		MOV	#BIT15,R2	;GET POSITION 2
7671	046474	000401			BR	4\$	
7672	046476	005002		3\$:	CLR	R2	;GET POSITION 2
7673	046500	060302		4\$:	ADD	R3,R2	;XOR B WITH POSITION 2
7674							;TO GIVE C
7675	046502	032701	000002		BIT	#BIT1,R1	;IS POSITION 15 HIGH
7676	046506	001403			BEG	5\$	;BRANCH IF POSITION 15 LOW
7677	046510	012704	100000		MOV	#BIT15,R4	;GET POSITION 15
7678	046514	000401			BR	6\$	
7679	046516	005004		5\$:	CLR	R4	;GET POSITION 15
7680	046520	060304		6\$:	ADD	R3,R4	;XOR POSITION 15 WITH B
7681							;TO GIVE E
7682	046522	005037	001200		ROR	@\$TMP4	;GET LSB INTO "C"
7683	046526	006001			ROR	R1	;GET ABOVE C INTO R1
7684	046530	005703			TST	R3	;TEST B
7685	046532	100403			BMI	7\$	;BRANCH IF B=1
7686	046534	042701	100000		BIC	#BIT15,R1	;SET B IN POSITION 1
7687	046540	000402			BR	10\$	
7688	046542	052701	100000	7\$:	BIS	#BIT15,R1	;SET B IN POSITION 1
7689	046546	005702		10\$:	TST	R2	;TEST C
7690	046550	100403			BMI	11\$	;BRANCH IF C=1
7691	046552	042701	020000		BIC	#BIT13,R1	;GET C IN POSITION 3
7692	046556	000402			BR	12\$	
7693	046560	052701	020000	11\$:	BIS	#BIT13,R1	;GET C IN POSITION 3
7694	046564	005704		12\$:	TST	R4	;TEST E
7695	046566	100403			BMI	13\$	;BRANCH IF E=1
7696	046570	042701	000001		BIC	#BIT0,R1	;GET E IN POSITION 16
7697	046574	000402			BR	14\$	
7698	046576	052701	000001	13\$:	BIS	#BIT0,R1	;GET E IN POSITION 16
7699	046602	005337	001174	14\$:	DEC	@\$TMP2	;BIT COUNTER
7700	046606	001310			BNE	15\$	;BRANCH IF 16 NOT DONE
7701	046610	005337	001170		DEC	@\$TMP0	;WORD COUNTER
7702	046614	001275			BNE	16\$	;BRANCH IF 4 NOT DONE
7703	046616	010135			MOV	R1,@(R5)+	;PUT CRC WHERE DESIRED
7704	046620	012604			MOV	(SP)+,R4	::POP STACK INTO R4
7705	046622	012603			MOV	(SP)+,R3	::POP STACK INTO R3
7706	046624	012602			MOV	(SP)+,R2	::POP STACK INTO R2
7707	046626	012601			MOV	(SP)+,R1	::POP STACK INTO R1
7708	046630	012600			MOV	(SP)+,R0	::POP STACK INTO R0
7709	046632	000205			RTS	R5	

```

7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720
7721
7722
7723
7724 046634
7725 046634 010046
7726 046636 010146
7727 046640 010246
7728 046642 012700 177400
7729 046646 012701 000400
7730 046652 012702 053520
7731 046656 010022
7732 046660 005301
7733 046662 001375
7734 046664 012701 000021
7735
7736 046670 005022
7737
7738 046672 005301
7739 046674 001375
7740
7741
7742
7743 046676 012737 010000 051602
7744 046704 112737 000001 051605
7745 046712 112737 000001 051604
7746 046720 012737 000001 051606
7747 046726 012737 000001 051610
7748 046734 016737 131440 051662
7749 046742 004537 046360
7750 046746 051602
7751 046750 053502
7752 046752 012602
7753 046754 012601
7754 046756 012600
7755 046760 000207
7756

```

```

.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.,@#DATAWORD ;:NO. OF DATA WORDS
JSR R5,@#CRC ;:GO TO CALCULATE CRC
CYL ;:FIRST CRC WORD
MCRC ;: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

```

.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
;*****

```

```

7757
7758
7759
7760
7761
7762
7763
7764
7765
7766
7767
7768
7769
7770
7771
7772
7773
7774
7775
7776 046762 010037 015130
7777 045766 162737 000004 015130
7778 046774 004737 045152
7779 047000 004737 045206
7780 047004 104400 005116
7781
7782 047010 000000
7783 047012 011037 001202
7784 047016 012011
7785 047020 012077 145746
7786 047024 112046
7787 047026 105720
7788 047030 112066 000001
7789 047034 105720
7790 047036 012677 145722
7791 047042 012077 145704
7792
7793 047046 012077 145702
7794 047052 012037 051612
7795
7796 047056 012777 014000 145704
7797
7798 047064 005037 015122
7799 047070 004737 051456
7800
7801
7802
7803
7804
7805
7806
7807
7808 047074 004737 044652
7809 047100 005737 015122
7810 047104 001034

```

```

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
                                ;STOP THE TEST
        HALT
        MOV     (RO), @#STMP5    ;SAVE COMMAND
        MOV     (RO)+, @R1      ;COMMAND
        MOV     (RO)+, @RHCA    ;CYLINDER
        MOV     (RO)+, @R1      ;SECTOR
        MOV     (RO)+, @R1      ;UP DATE RO
        TST    (RO)+            ;TRACK
        MOV     (RO)+, 1(SP)    ;UPDATE RO
        TST    (RO)+            ;TRACK SECTOR
        MOV     (SP)+, @RHDS    ;NO. OF DATA WORDS +4 HEADER
        MOV     (RO)+, @RHBA    ;IF A READ HEADER AND DATA
                                ;STARTING ADDRESS OF BUFFER
        MOV     (RO)+, @#X      ;X=0 READ HEADER AND DATA
                                ;X=1 WRITE DATA
                                ;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    IOS             ;BRANCH IF YES

```

```

7811 047106 005737 051612      TST      @#X      ; IS THIS A READ
7812 047112 001015              BNE      3$      ; IF A WRITE DATA BRANCH
7813
7814                          ; NOW THE READ BUFFER WILL BE CHECKED
7815                          ; HEADER SHOULD BE COMPLETELY READ AS WRITTEN
7816                          ; NO DATA WORDS SHOULD BE READ
7817                          ; REINTO BUFFER HAS BEEN FILLED WITH 0
7818                          ; MRFROM BUFFER HAS BEEN FILLED WITH EXPECTED DATA
7819
7820 047114 004037 046046      JSR      RD,@#COMPAR ; CHECK
7821 047120 015212              MRFROM          ; GOOD DATA
7822 047122 016256              REINTO          ; TEST BUFFER
7823 047124 000400              256.          ; 4 HEADER 252 DATA
7824 047126 047134              1$           ; RETURN POINT FOR ERROR HEADER
7825 047130 047140              2$           ; RETURN POINT FOR ERROR DATA
7826 047132 047176              10$          ; RETURN FOR GOOD COMPARISON
7827 047134 104004              1$: ERROR 4   ; READ NEXT ERROR 5
7828 047136 000207              RTS      PC    ; RETURN TO COMPARISON SUBROUTINE
7829 047140 104005              2$: ERROR 5   ; WORD NO 1 THRU 4 ARE
7830                          ; HEADER WORDS AND HENCE
7831                          ; SHOULD BE READ AS WRITTEN ON
7832                          ; DISK, WORD NOS. 5 ONWARDS
7833                          ; SHOULD NOT BE READ AND HENCE
7834                          ; READ INTO BUFFER
7835                          ; SHOULD BE UNCHANGED
7836 047142 000207              RTS      PC    ; RETURN TO COMPARISON
7837
7838 047144 000414              BR       10$   ; JUMP OUT
7839
7840                          ; NOW THE DISK WILL BE CHECKED
7841                          ; NO DATA SHOULD BE WRITTEN
7842                          ; REINTO BUFFER HAS BEEN FILLED WITH EXPECTED DATA
7843                          ; DISK HAS BEEN FILLED WITH 177400
7844                          ; MRFROM HAS BEEN FILLED WITH 125252
7845
7846 047146 004037 046046      3$: JSR      RD,@#COMPAR ; CHECK
7847 047152 016256              REINTO          ; GOOD DATA BUFFER
7848 047154 053520              DISK           ; TEST BUFFER
7849 047156 000400              256.          ;
7850 047160 047166              4$           ; RETURN POINT FOR ERROR HEADER
7851 047162 047172              5$           ; RETURN POINT FOR ERROR DATA
7852 047164 047176              10$          ; RETURN POINT FOR GOOD COMPARISON
7853 047166 104004              4$: ERROR 4   ; READ NEXT ERROR 5
7854 047170 000207              RTS      PC    ; RETURN TO COMPARISON SUBROUTINE
7855 047172 104005              5$: ERROR 5   ; WORD NO ARE ALL DATA
7856                          ; WORDS THE SHOULD NOT
7857                          ; HAVE BEEN CHANGED BY THE
7858                          ; WRITE COMMAND
7859 047174 000207              RTS      PC    ; RETURN TO COMPARISON SUBROUTINE
7860 047176 005720              10$: TST     (RD)+ ; IS THIS A HCRC ON HCE CHECK?
7861 047200 001442              BEQ     6$     ; BRANCH IF HCRC
7862 047202 022737 000072 001202  CMP     #72,@#STMP5 ; IS THIS A READ COMMAND
7863 047210 001417              BEQ     11$    ; BRANCH IF YES
7864 047212 017737 145544 001126  MOV     @RHER1,@#SBCDAT ; TEST DATA
    
```

7865	047220	022737	000200	001126		CMP	#HCE, @#SBDDAT	: ONLY HEADER COMPARE BIT?
7866								: SHOULD BE SET
7867	047226	001470				BEQ	75	: BRANCH IF GOOD
7868	047230	013737	014762	044716		MOV	@#RHER1, @#REGADR	: REGISTER ADDRESS RHER1
7869	047236	012737	000200	001124		MOV	#HCE, @#SGDDAT	: GOOD DATA
7870	047244	104027				ERROR	27	: AFTER AN ERROR ON THE
7871								: HEADER ONLY HCE SHOULD
7872	047246	000460				BR	75	: BE SET
7873	047250				115:			
7874	047250	017737	145506	001126		MOV	@#RHER1, @#SBDDAT	: TEST DATA
7875	047255	022737	100200	001126		CMP	#DCK!HCE, @#SBDDAT	: ONLY HEADER COMPARE BIT?
7876								: SHOULD BE SET
7877								: DCK IS SET BECAUSE ECC IS NOT READ
7878	047264	001451				BEQ	75	: BRANCH IF GOOD
7879	047266	013737	014762	044716		MOV	@#RHER1, @#REGADR	: REGISTER ADDRESS RHER1
7880	047274	012737	100200	001124		MOV	#DCK!HCE, @#SGDDAT	: GOOD DATA
7881	047302	104027				ERROR	27	: AFTER AN ERROR ON THE
7882								: HEADER ONLY HCE SHOULD
7883	047304	000441				BR	75	: BE SET
7884	047306	022737	000072	001202	65:	CMP	#72, @#STMP5	: IS THIS A READ COMMAND?
7885	047314	001417				BEQ	125	: BRANCH IF A READ
7886	047316	017737	145440	001126		MOV	@#RHER1, @#SBDDAT	: TEST DATA
7887	047324	022737	000400	001126		CMP	#HCRC, @#SBDDAT	: ONLY CRC ERROR SHOULD BE THERE
7888	047332	001426				BEQ	75	
7889	047334	013737	014762	044716		MOV	@#RHER1, @#REGADR	: REG. ADDR = RHER1
7890	047342	012737	000400	001124		MOV	#HCRC, @#SGDDAT	: GOOD DATA
7891	047350	104027				ERROR	27	: AFTER A CRC ERROR ONLY CRC
7892								: SHOULD BE SET
7893	047352	000416				BR	75	: BRANCH OUT
7894	047354	017737	145402	001126	125:	MOV	@#RHER1, @#SBDDAT	: TEST DATA
7895								
7896	047362	022737	100400	001126		CMP	#DCK!HCRC, @#SBDDAT	: HCRC AND DCK SHOULD BE SET
7897								: DCK IS SET BECAUSE ECC IS NOT READ
7898	047370	001407				BEQ	75	: BRANCH IF GOOD
7899	047372	012737	100400	001124		MOV	#DCK!HCRC, @#SGDDAT	: GOOD DATA
7900	047400	013737	014762	044716		MOV	@#RHER1, @#REGADR	: FAILING REGISTER RHER1
7901	047406	104027				ERROR	27	: AFTER A CRC ERROR ON A READ
7902								: DCK AND HCRC SHOULD BE SET
7903								: DCK IS SET BECAUSE ECC IS NOT READ
7904	047410	000200			75:	RTS	RO	: RETURN TO MAIN TEST



.SBTTL EXIT WRT HEADER & DATA ROUTINE

7905  
7906  
7907  
7908  
7909  
7910  
7911  
7912  
7913  
7914  
7915  
7916  
7917  
7918  
7919  
7920  
7921  
7922  
7923  
7924  
7925  
7926  
7927  
7928  
7929  
7930  
7931  
7932  
7933  
7934  
7935  
7936  
7937

047412  
047412 010046  
047414 010146  
047416 013777 015170 145334  
047424 012777 177766 145320  
047432 012777 015212 145314  
047440 012777 000010 145316  
047446 052777 000010 145302  
047454 012777 010000 145306  
047462 005077 145304  
047466 012737 000001 047514  
047474 012777 000001 145276  
047502 052777 000001 145250  
047510 004137 055634  
047514 000000  
047516 012601  
047520 012600  
047522 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, @MIDDLE  
: BAI IS SET  
\*\*\*\*\*

MIDDLE:

MOV RO, -(SP) ;: PUSH RO ON STACK  
MOV R1, -(SP) ;: PUSH R1 ON STACK  
MOV @RIFOR, @RHCSI ;: WRITE HEADER AND DATA=62  
;: IN RHCSI  
MOV #-10, @RHMC ;: 10 WORDS  
MOV @RFROM, @RHBA ;: BUS ADDRESS=@RFROM  
MOV #10, @RHCS1 ;: 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 RHCSI WITH 62  
BIS @GO, @RHCSI  
JSR R1, @SEARCH  
MID: .WORD 0 ;: SECTOR  
MOV (SP)+, R1 ;: POP STACK INTO R1  
MOV (SP)+, RO ;: POP STACK INTO RO  
RTS PC

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  
7375  
7976

.SBTTL JAM CURRENT CYLINDER ROUTINE

```

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

```

MAKECYL:

```

MOV   RS,-(SP)           ;.PUSH RS ON STACK
MOV   RO,@#PCJSR        ;PC OF JSR+4
SUB   #4,@#PCJSR        ;SAVE PC OF JSR
MOV   RS,@#RHCA         ;GETTING READY TO FILL DESIRED CYLINDER
CLR   @#RHST            ;FILL DESIRED CYLINDER REGISTER
MOV   @#SEECOM,@#RHCS1 ;MAKE SURE DESIRED SECTOR TRACK IS NOT ILLEGAL
MOV   @#DMD,@#RHMR      ;FILL SEEK COMMAND
BIS   @#GO,@#RHCS1      ;SET DIAGNOSTIC MODE
NOP   ;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   RS,@#SBDDAT      ;COMPARE CURRENT CYLINDER
BEQ   IS               ;BRANCH IF GOOD
MOV   RS,@#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)+,RS        ;;POP STACK INTO RS
RTS   RO

```

```

047524
047524 010546
047526 010037 015130
047532 162737 000004 015130
047540 012005
047542 010577 145224
047546 005077 145212
047552 013777 015176 145200
047560 012777 000001 145212
047566 052777 000001 145164
047574 000240
047576 000240
047600 000240
047602 000240
047604 004737 045152
047610 017737 145202 001126
047616 020537 001126
047622 001406
047624 010537 001124
047630 013737 015016 044716
047636 104030
047640
047640 012605
047642 000200

```

.SBTTL ECC GENERATION AND COMPARISON ROUTINE

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

7977			
7978			
7979			
7980			
7981			
7982			
7983			
7984			
7985			
7986			
7987			
7988	100000	PIE1	=100000
7989	040000	PIE2	=40000
7990	020000	PIE3	=20000
7991	010000	PIE4	=10000
7992	004000	PIE5	=4000
7993	002000	PIE6	=2000
7994	001000	PIE7	=1000
7995	000400	PIE8	=400
7996	000200	PIE9	=200
7997	000100	PIE10	=100
7998	000040	PIE11	=40
7999	000020	PIE12	=20
8000	000010	PIE13	=10
8001	000004	PIE14	=4
8002	000002	PIE15	=2
8003	000001	PIE16	=1
8004	100000	PIE17	=100000
8005	040000	PIE18	=40000
8006	020000	PIE19	=20000
8007	010000	PIE20	=10000
8008	004000	PIE21	=4000
8009	002000	PIE22	=2000
8010	001000	PIE23	=1000
8011	000400	PIE24	=400
8012	000200	PIE25	=200
8013	000100	PIE26	=100
8014	000040	PIE27	=40
8015	000020	PIE28	=20
8016	000010	PIE29	=10
8017	000004	PIE30	=4
8018	000002	PIE31	=2
8019	000001	PIE32	=1

8020			
8021	047644	000000	ECDATA: 0
8022			
8023			
8024			
8025	047646	000000	GECC1: 0
8026			
8027			
8028	047650	000000	GECC2: 0
8029			
8030			

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

;LOW ORDER ECC WORD TO BE GENERATED HERE  
;=R1

;HIGH ORDER ECC WORD TO BE GENERATED HERE  
;=R2

8031 047652 000000  
8032  
8033  
8034 047654 113713  
8035 047656 000000  
8036 047660 000000  
8037 047662 010041

TSECCG: 0  
NCODE: 38859.  
NCOUNT: 0  
POSITI: 0  
HARDER: 4129.

; IF =177777 GENERATE AND TEST ECC FOR THIS BIT  
; IF =0 DO NOT GENERATE AND TEST ECC FOR THIS BIT  
; N-CODE WORD  
; TEMPORARY N CODE  
; POSITION REGISTER  
; HARD ERROR COUNT  
; TRUE COUNT IS 4128 BUT AS COMPARES ARE  
; DONE ONE STAGE LATER SO 4129  
; DATA ENVELOPE FOR TYPE OUT  
; MAX FOR WRITE IS 4096  
; MAX FOR READ IS 4128  
; LEADING ZEROS ENVELOPE FOR TYPE OUT  
; THIS IS SHUT OFF WHEN POSITION COUNTER  
; IN ENABLED  
; MAX COUNT IS 38859

8040 047664 000000  
8041  
8042  
8043 047666 000000  
8044  
8045

DATENV: 0  
ZCODE: 0

8046  
8047  
8048  
8049  
8050 047670 000000  
8051 047672 000000  
8052 047674 000000  
8053 047676 000000  
8054 047700 000000

HADTMP: 0  
P3: 0  
P12: 0  
P22: 0  
P24: 0

; TEMPORARY HARD ERROR COUNT

8055  
8056  
8057  
8058  
8059

8060 047702  
8061 047702 010046  
8062 047704 010146  
8063 047706 010246  
8064 047710 010346  
8065 047712 010446  
8066 047714 010546  
8067 047716 013701 047646  
8068 047722 013702 047650  
8069 047726 005737 047644  
8070 047732 001406

ECTEST:

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 @#ECC1, R1 ;: ECC1 WORD  
MOV @#ECC2, R2 ;: ECC2 WORD  
TST @#ECDATA ;: IS CURRENT BIT A ONE  
BEQ 2\$ ;: BRANCH IF CURRENT DATA D=0

8071  
8072  
8073  
8074

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

8075 047734 010103  
8076 047736 052703 177776  
8077 047742 005103  
8078 047744 010300  
8079 047746 000404

1\$: MOV R1, R3  
BIS #1CPIE32, R3  
COM R3  
MOV R3, R0  
BR 3\$

8080  
8081  
8082  
8083 047750 010103  
8084 047752 042703 177776

; IF CARRY IS ZERO THEN D=0  
; X32 BECOMES R0  
2\$: MOV R1, R3  
BIC #1CPIE32, R3

```

8085 047756 010300          MOV      R3,R0
8086
8087 047760 000241          3$:    CLC
8088 047762 006000          ROR      R0
8089 047764 006000          ROR      R0
8090 047766 005700          TST      R0
8091 047770 001462          BEQ      10$          ;BRANCH IF R0=0
8092                          ;INVERT X2
8093
8094 047772 010203          MOV      R2,R3
8095 047774 052703 137777    BIS      #1CPIE2,R3
8096 050000 005103          COM      R3
8097 050002 010337 047672    MOV      R3,@#P3
8098 050006 006237 047672    ASR      @#P3
8099
8100                          ;INVERT X11
8101
8102
8103 050012 010203          MOV      R2,R3
8104 050014 052703 177737    BIS      #1CPIE11,R3
8105 050020 005103          COM      R3
8106 050022 010337 047674    MOV      R3,@#P12
8107 050026 006237 047674    ASR      @#P12
8108
8109                          ;INVERT X21
8110
8111 050032 010103          MOV      R1,R3
8112 050034 052703 173777    BIS      #1CPIE21,R3
8113 050040 005103          COM      R3
8114 050042 010337 047676    MOV      R3,@#P22
8115 050046 006237 047676    ASR      @#P22
8116
8117                          ;INVERT X23
8118
8119 050052 010103          MOV      R1,R3
8120 050054 052703 176777    BIS      #1CPIE23,R3
8121 050060 005103          COM      R3
8122 050062 010337 047700    MOV      R3,@#P24
8123 050066 006237 047700    ASR      @#P24
8124
8125                          ;NOW THAT R0 FOR POSITION 1
8126                          ;      P3 FOR POSITION 3
8127                          ;      P12 FOR POSITION 12
8128                          ;      P22 FOR POSITION 22
8129                          ;      P24 FOR POSITION 24
8130                          ;ARE KNOWN THE ROTATE WILL BE DONE AND
8131                          ;THESE BITS JAMED IN
8132
8133 050072 006002          ROR      R2
8134 050074 006001          ROR      R1
8135 050076 053700 047672    BIS      @#P3,R0
8136 050102 053700 047674    BIS      @#P12,R0
8137 050106 042702 120020    BIC      #PIE1!PIE3!PIE12,R2
8138 050112 050002          BIS      R0,R2

```



```

8193
8194
8195
8196
8197
8198
8199
8200 050250
8201 050250 012605
8202 050252 012604
8203 050254 012603
8204 050256 012602
8205 050260 012601
8206 050262 012600
8207 050264 000207

```

148:

```

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

```

8208
8209
8210
8211
8212
8213
8214
8215
8216
8217
8218 050266 000000 ERPOS: 0 ; POSITION REG. WHEN CORRECTION IS COMPLETE
8219
8220
8221
8222 050270 010037 015130 ECORR: MOV R0, @PCJSR ; SAVE PC OF JSR + 4
8223 050274 162737 000004 015130 SUB #4, @PCJSR ; SAVE PC OF JSR
8224 050302 012037 050266 MOV (R0)+, @ERPOS ; GET POSITION REG. WHEN CORRECTION IS COMPLETE
8225 050306 010146 MOV R1, -(SP) ; PUSH R1 ON STACK
8226 050310 013701 015000 MOV @RHMR, R1 ; MAINTENANCE REGISTER
8227 050314 012711 000001 MOV @DMD, @R1 ; SET DIAGNOSTIC MODE BIT
8228 050320 005037 047644 CLR @ECDATA ; ECC DATA IS ZERO
8229
8230
8231
8232 050324 005737 047660 1$: TST @POSITI ; IS SOFTWARE POSITION NON ZERO
8233 050330 001007 BNE 2$ ; BRANCH IF N-CODE S COMPLETE
8234 050332 005337 047656 DEC @NCOUNT ; DECREMENT N-CODE
8235 050336 001001 BNE 6$ ; BRANCH IF N-CODE ; NOT COMPLETE
8236 050340 000403 BR 2$ ; BRANCH AS N-CODE IS COMPLETE
8237 050342 005237 047666 6$: INC @ZCODE ; INCREMENT CLOCKS GIVEN FOR LEADING ZEROS
8238 050346 000420 BR 3$ ; BRANCH AS N-CODE IS NOT COMPLETE
8239
8240 050350 005237 047660 2$: INC @POSITI ; INCREMENT SOFTWARE POSITION
8241 050354 023737 050266 047660 CMP @ERPOS, @POSITI ; HAVE ENOUGH CLOCKS BEEN GIVEN TO DETECT ERROR
8242 050362 103012 BHS 3$ ; BRANCH IF MORE CLOCKS TO BE GIVEN
8243 050364 023737 047670 047660 CMP @HADTMP, @POSITI ; HAVE ENOUGH CLOCKS BEEN GIVEN FOR HARD ERROR
8244 ; THAT IS HAVE 4128 MORE CLOCKS BEEN GIVEN
8245 BEQ 5$ ; BRANCH IF YES
8246 050374 032711 000400 BIT #ZER, @R1 ; CHECK ZERO DETECT BIT IN RHMR
8247 050400 001016 BNE 4$ ; BRANCH IS ZER SET
8248 ; TO SAVE TIME
8249 050402 004737 044652 JSR PC, @PUTREG ; SAVE REGISTERS
8250 050406 104034 ERROR 34 ; ZERO DETECT BIT NOT HIGH
8251 ; WHEN 21 BITS IN ECC 32 BIT REGISTER IS 0
8252
8253
8254 050410 052711 000002 3$: BIS #CLK, @R1 ; SET CLOCK
8255 050414 042711 000002 BIC #CLK, @R1 ; CLEAR CLOCK
8256 050420 004737 047702 JSR PC, @ECTEST ; GO TO GENERATE AND TEST ECC
8257 050424 000737 BR 1$ ; CONTINUE
8258
8259
8260
8261

```

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



```
8262 050426 052711 000002      5S:  BIS      #MCLK,R1      ;SET CLOCK
8263 050432 042711 000002      BIC      #MCLK,R1      ;CLEAR CLOCK
8264
8265 050436                                4S:
8266 050436 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
8267 050440 000200      RTS      RO
```

.SBTTL SOFTWARE DISK DATA ECC GEN. ROUTINE

8268  
8269  
8270  
8271  
8272  
8273  
8274  
8275  
8276  
8277  
8278  
8279  
8280  
8281  
8282  
8283  
8284  
8285  
8286  
8287  
8288  
8289  
8290  
8291  
8292  
8293  
8294  
8295  
8296  
8297  
8298  
8299  
8300  
8301  
8302  
8303  
8304  
8305  
8306  
8307  
8308

050442  
050442 010046  
050444 010146  
050446 010246  
050450 010346  
050452 010446  
050454 010546  
050456 005037 047660  
050462 005037 047646  
050466 005037 047650  
050472 012701 053520  
050476 012702 000400  
050502 012703 000020  
050506 012104  
050510 006004  
050512 103004  
050514 012737 177777 047644  
050522 000402  
050524 005037 047644  
050530 004737 047702  
050534 005303  
050536 001364  
050540 005302  
050542 001357  
050544 013737 047646 054520  
050552 013737 047650 054522  
050560 012605  
050562 012604  
050564 012603  
050566 012602  
050570 012601  
050572 012600  
050574 000207

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

FILEEC:

```
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
    CLR @#ECDATA ;; ECC DATA BIT IS A ZERO
115: JSR PC,@#ECTEST ;; GO TO GENERATE ECC
125: 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
```

.SBTTL RH BASE ADDRESS CHANGE ROUTINE

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

8309  
8310  
8311  
8312  
8313  
8314  
8315  
8316  
8317 050576  
8318 050576 104400 050604  
8319 050602 000425  
8320  
8321 050656  
8322 050656 013746 014760  
8323 050662 104402  
8324 050664 104400 050672  
8325 050670 000425  
8326  
8327 050744  
8328 050744 004737 057146  
8329 050750 04416  
8330 050752 012700 014750  
8331 050756 012701 000024  
8332 050762 042710 177700  
8333 050766 051620  
8334 050770 005301  
8335 050772 001373  
8336 050774 104400 051002  
8337 051000 000417  
8338  
8339 051040  
8340 051040 013746 014746  
8341 051044 104402  
8342 051046 104400 051054  
8343 051052 000437  
8344  
8345 051152  
8346 051152 104416  
8347 051154 012637 014746  
8348 051160 104400 051166  
8349 051164 000421  
8350  
8351 051230  
8352 051230 104400 051236  
8353 051234 000416  
8354  
8355 051272  
8356 051272 013746 014760  
8357 051276 104402  
8358 051300 104400 051306  
8359 051304 000416  
8360  
8361 051342  
8362 051342 013746 014746

BASECH:  
TYPE +4 ;:TYPE ASCIZ STRING  
BR 64\$ ;:GET OVER THE ASCIZ  
;:ASCIZ <15><12>/PRESENT BASE ADDRESS OF REGISTERS IS /  
64\$:  
MOV @#RHCS1,-(SP) ;GET READY TO TYPE OLD BASE  
TYPOC  
TYPE +4 ;:TYPE ASCIZ STRING  
BR 65\$ ;:GET OVER THE ASCIZ  
;:ASCIZ <15><12>/TYPE NEW BASE ADDRESS FOLLOWED BY 'CR' /  
65\$:  
JSR PC,@#STKINT ;INITIALIZE THE TTY KEYBOARD  
RDOCT  
MOV @#RHDB,R0 ;GET STARTING ADDRESS OF REGISTERS  
MOV @#20,R1 ;NUMBER OF REGISTERS  
1\$: BIC @#1C77,(R0) ;CLEAR OLD BASE ADDRESS  
BIS (SP),(R0)+ ;SET NEW BASE  
DEC R1 ;COUNT  
BNE 1\$ ;BRANCH IF 20 NOT DONE  
TYPE +4 ;:TYPE ASCIZ STRING  
BR 66\$ ;:GET OVER THE ASCIZ  
;:ASCIZ <15><12>/PRESENT VECTOR ADDRESS IS /  
66\$:  
MOV @#RPVEC,-(SP) ;GET READY TO TYPE OLD VECTOR ADDRESS  
TYPOC  
TYPE +4 ;:TYPE ASCIZ STRING  
BR 67\$ ;:GET OVER THE ASCIZ  
;:ASCIZ <15><12>/TYPE NEW VECTOR ADDRESS OR RETYPE OLD ONE FOLLOWED BY "CR"  
67\$:  
RDOCT  
MOV (SP)+,@#RPVEC ;SETUP VECTOR ADDRESS  
TYPE +4 ;:TYPE ASCIZ STRING  
BR 68\$ ;:GET OVER THE ASCIZ  
;:ASCIZ <15><12>/RESTART PROGRAM FROM 200 OR 210/  
68\$:  
TYPE +4 ;:TYPE ASCIZ STRING  
BR 69\$ ;:GET OVER THE ASCIZ  
;:ASCIZ <15><12>/NEW BASE WILL REMAIN - /  
69\$:  
MOV @#RHCS1,-(SP)  
TYPOC  
TYPE +4 ;:TYPE ASCIZ STRING  
BR 70\$ ;:GET OVER THE ASCIZ  
;:ASCIZ <15><12>/NEW VECTOR WILL REMAIN - /  
70\$:  
MOV @#RPVEC,-(SP)



8390  
8391  
8392  
8393  
8394  
8395  
8396  
8397  
8398  
8399  
8400  
8401  
8402  
8403  
8404  
8405  
8406  
8407  
8408  
8409  
8410  
8411  
8412  
8413  
8414  
8415  
8416  
8417  
8418  
8419  
8420  
8421  
8422  
8423  
8424  
8425  
8426  
8427  
8428  
8429  
8430  
8431  
8432  
8433  
8434  
8435  
8436  
8437  
8438  
8439  
8440  
8441  
8442  
8443

```
.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
*WNWORD= NO OF DATA WORDS TO BE WRITTEN ON DISK
*THE COMMAND THEN IS JSR PC,COMHD
*
*
*
*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,COMHD
*
*
*
*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
*DNWORD= 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
*DNWORD= WITH NUMBER OF WORDS TO BE FOUND ON DISK
*X=0 MUST BE ZERO
*THE COMMAND THEN IS JSR PC,COMHD
```

L03

MAINDEC-11-DZRPT-D, RJPO4 DISKLESS RH11 TEST-PART 2  
DZRPTD.SUB DISK SIMULATION

MACY11 27(663) 7-OCT-75 17:33 PAGE 201

SEQ 0243

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  
8487  
8488  
8489  
8490  
8491  
8492  
8493  
8494  
8495  
8496  
8497  
8498

051454 000000  
051456 011637 015130  
051462 162737 000004 015130  
051470 010046  
051472 010146  
051474 010246  
051476 010346  
051500 010446  
051502 010546  
051504 012777 000001 143266  
051512 052777 000001 143240  
051520 012737 000113 051454  
051526 005337 051454  
051532 001375  
051534 016746 000044  
051540 042716 177740  
051544 012637 051554  
051550 004137 055634  
051554 000000  
051556 012701 000240  
051562 010137 051614  
051566 010137 051616  
051572 010137 051620  
051576 004137 051724  
051602 000000  
051604 000000  
051606 000000  
051610 000000  
051612 000000  
051614 000240

\*\*\*\*\*  
:WRITE DATA COMMAND  
:OR READ COMMAND, I.E DATA ONLY, OR HEADER AND DATA  
:\*\*\*\*\*

RUNCTR: .WORD 0  
COMHD: 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 #GO, @RHCS1 ;ISSUE 'GO' BIT & STALL 'TILL 'RUN'  
;FUNCTION CODE IS ISSUED BY THE TEST  
RUNMAT: 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  
JSR R1, @SEARCH ;DO SEARCH SECTOR & ISSUE SECTOR CLOCKS  
TRK: .WORD 0  
MOV #+NOP, R1 ;GOING TO MOVE NOPS  
MOV R1, @SSYN ;NOP INTO SSYN  
MOV R1, @HEDGAP ;NOP INTO HEDGAP  
MOV R1, @HEDSYN ;NOP INTO HEDSYN  
JSR R1, @RDHEAD ;READ THE HEADER  
CYL: .WORD 0 ;CYLINDER ADDRESS  
SECOTR: .WORD 0 ;SECTOR/TRACK ADDRESS  
KEY1: .WORD 0 ;KEY1 WORD  
KEY2: .WORD 0 ;KEY2 WORD  
X: .WORD 0 ;X=1 WRITE COMMAND  
;X=0 READ COMMAND  
SSYN: NOP ;IF "ERROR 2" INSERTED BY RDHEAD  
;SUBROUTINE, THEN THE FIRST SYNC  
;IS NOT DETECTED. NO BAD DATA  
;IS GIVEN BECAUSE SYNC=144000  
;CANNOT BE READ. WORD NUMBER  
;IS "1" BECAUSE THIS IS THE FIRST  
;WORD TESTED.





8553  
8554  
8555  
8556  
8557  
8558  
8559  
8560  
8561  
8562  
8563  
8564  
8565  
8566  
8567  
8568  
8569  
8570  
8571  
8572  
8573  
8574  
8575  
8576  
8577  
8578  
8579  
8580  
8581  
8582  
8583  
8584

\*\*\*\*\*

;\*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

051704 014400  
051706 000000  
051710 000000  
051712 000000  
051714 000000

RSYNC: 14400  
RCYL: 0  
RSETR: 0  
RKEY1: 0  
RKEY2: 0

;\*5 WORDS OF 0 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 FOR DATA GAP AND TOLERANCE GAP  
\*\*\*\*\*

```

8585 :*****
8586 :*READ DISK HEADER
8587 :*****
8588
8589
8590
8591
8592
8593 051716 000000 NOSYNC: 0 ;FORCED HEADER ERROR = -1
8594 ;NORMAL = 0
8595 051720 000000 TY: 0 ;ERROR TYPE NO.
8596 051722 000000 ERMWORD: 0 ;ERROR WORD NO.
8597
8598
8599
8600
8601 051724 012137 051706 RDHEAD: MOV (R1)+, @#RCYL ;STORE CYLINDER ADDRESS
8602 051730 012137 051710 MOV (R1)+, @#RSETR ;STORE SECTOR AND TRACK ADDRESS
8603 051734 012137 051712 MOV (R1)+, @#RKEY1 ;STORE KEY1
8604 051740 012137 051714 MOV (R1)+, @#RKEY2 ;STORE KEY2
8605 051744 012137 052514 MOV (R1)+, @#COMPA ;STORE COMPARE OR NOT
8606 051750 010146 MOV R1, -(SP) ;PUSH R1 ON STACK
8607 051752 013700 015000 MOV @#AHMR, R0 ;R0 CONTAINS MAINTANENCE REG.
8608 051756 012705 000002 MOV #2, R5 ;R5 IS A COUNTER FOR WORDS
8609 051762 012710 000001 MOV @#DMD, @R0 ;DIAG. MODE
8610 051766 052710 000010 BIS @#MSTCK, @R0 ;SET SECTOR FOR FIRST WORD
8611 051772 052710 000002 BIS @#MCLK, @R0 ;SET CLOCK FOR FIRST WORD
8612 051776 042710 000012 BIC @#MSTCK!MCLK, @R0 ;RESET SECTOR AND CLOCK
8613 052002 000404 BR 2$ ;BRANCH OVER GIVING SECTOR FOR FIRST TIME
8614 052004 012710 000013 1$: MOV @#MSTCK!MCLK!DMD, @R0 ;SET SECTOR, CLOCK, DIAG. MODE, RESET INDEX
8615 052010 042710 000012 BIC @#MSTCK!MCLK, @R0 ;RESET SECTOR, CLOCK
8616 052014 012702 000007 2$: MOV #7, R2 ;R2 IS A COUNTER FOR BYTES
8617 052020 052710 000002 3$: BIS @#MCLK, @R0 ;SET CLOCK
8618 052024 042710 000002 BIC @#MCLK, @R0 ;RESET CLOCK
8619 052030 005302 DEC R2 ;BYTE COUNTER
8620 052032 001372 BNE 3$ ;BRANCH IF BYTE NOT COMPLETE
8621 052034 005305 DEC R5 ;WORD COUNTER
8622 052036 001362 BNE 1$ ;BRANCH IF WORD NOT COMPLETE
8623 052040 012702 000022 MOV #18, R2 ;NO OF WORDS OF ZEROS
8624 052044 005037 052512 4$: CLR @#WORD ;READ 0
8625 052050 004737 052516 JSR PC, @#READ ;GO TO READ
8626 052054 005302 DEC R2 ;COUNT
8627 052056 001372 BNE 4$
8628 052060 013737 051704 052512 MOV @#RSYNC, @#WORD ;SYNC. WORD
8629 052066 004737 052516 JSR PC, @#READ
8630 052072 032710 001000 BIT @#01SY, @R0 ;SYNC. BYTE DETECTED?
8631 052076 001012 BNE 5$ ;BRANCH IF SYNC DETECTED
8632 052100 012737 000001 051722 MOV #1, @#ERMWORD ;ERROR WORD NO
8633 052106 013737 051704 001124 MOV @#RSYNC, @#SGDDAT ;SYNC WORD
8634 052114 012737 104002 051614 MOV #104002, @#SSYN ;INSERT "ERROR 2" IN SSYN
8635 052122 000571 BR 13$ ;BRANCH OUT
8636 052124 013737 051706 052512 5$: MOV @#RCYL, @#WORD ;SETUP CYLINDER
8637 052132 004737 052516 JSR PC, @#READ ;READ
8638 052136 013737 051710 052512 MOV @#RSETR, @#WORD ;SETUP SECTOR/TRACK

```

8639	052144	004737	052516			JSR	PC, @#READ	: READ
8640	052150	013737	051712	052512		MOV	@#RKEY1, @#WORD	: SETUP KEY1
8641	052156	004737	052516			JSR	PC, @#READ	: READ
8642	052162	013737	051714	052512		MOV	@#RKEY2, @#WORD	: SETUP KEY2
8643	052170	004737	052516			JSR	PC, @#READ	: READ
8644	052174	013737	053502	052512		MOV	@#CRC, @#WORD	: SETUP CRC
8645	052202	004737	052516			JSR	PC, @#READ	: READ
8646	052206	005737	015142			TST	@#TESOTE	: IS THIS A DRIVE TIMING ERROR
8647	052212	001135				BNE	13\$	: BRANCH OUT IF YES
8648	052214	005737	052514			TST	@#COMPA	: IS THIS A READ OR WRITE COMMAND
8649	052220	001472				BEQ	11\$	
8650	052222	012705	053504			MOV	#HEGAP, R5	: POINTER FOR HEADER GAP
8651	052226	012702	000005			MOV	#5, R2	: NO OF WORDS OF ZEROS
8652	052232	012737	000006	051722	6\$:	MOV	@#ERWORD	: ERROR WORD NO SET
8653	052240	004737	052750			JSR	PC, @#WRITE	: FOR HEADER GAP
8654	052244	005737	052746			TST	@#WORD	: TEST WRITTEN WORD
8655	052250	001413				BEQ	7\$	: BRANCH IF GOOD THAT IS 0
8656	052252	160237	051722			SUB	R2, @#ERWORD	: WORD NO IN ERROR
8657	052256	005037	001124			CLR	@#SGDDAT	: GOOD WORD SHOULD BE 0
8658	052262	013767	052746	126636		MOV	@#WORD, @#BDDAT	: BAD DATA
8659	052270	012737	104003	051616		MOV	#104003, @#HEDGAP	: "ERROR 2" GOES IN HEDGAP
8660	052276	000503				BR	13\$	: BRANCH OUT
8661	052300	013725	052746		7\$:	MOV	@#WORD, (R5)+	: SAVE HEADER GAP
8662	052304	005302				DEC	R2	
8663	052306	001351				BNE	6\$	
8664	052310	004737	052750			JSR	PC, @#WRITE	: WRITE HEADER (DATA) GAP SYNC
8665	052314	023737	051704	052746		CMF	@#RSYNC, @#WORD	
8666	052322	001426				BEQ	10\$	
8667	052324	005737	051716			TST	@#NOSYNC	: IS THIS FORCED HEADER ERROR COMMAND
8668								: IF YES NOSYNC=-1 THEN WRITE OR READ
8669								: IS SHUT OFF SO BRANCH OUT
8670								: IF NO NOSYNC=0 THEN CONTINUE
8671	052330	001406				BEQ	14\$	: BRANCH IF TRUE ERROR
8672	052332	005737	052746			TST	@#WORD	
8673	052336	001420				BEQ	10\$	: BRANCH IF GOOD
8674	052340	005037	001124			CLR	@#SGDDAT	: IT SHOULD BE ZERO
8675	052344	000403				BR	15\$	: BRANCH TO TYPE ERROR
8676	052346	013737	051704	001124	14\$:	MOV	@#RSYNC, @#SGDDAT	: GOOD DATA
8677	052354	013737	052746	001126	15\$:	MOV	@#WORD, @#BDDAT	: BAD DATA
8678	052362	012737	000006	051722		MOV	#6, @#ERWORD	
8679	052370	012737	104003	051620		MOV	#104003, @#HEDSYN	
8680	052376	000443				BR	13\$	: BRANCH OUT
8681	052400	013725	052746		10\$:	MOV	@#WORD, (R5)+	: SAVE DATA SYNC.
8682	052404	000440				BR	13\$	
8683								: *READ COMMAND START FROM HERE
8684	052406	012702	000005		11\$:	MOV	#5, R2	
8685	052412	005067	000074		12\$:	CLR	WORD	
8686	052416	004767	000074			JSR	PC, READ	: READ HEADER GAP
8687	052422	005302				DEC	R2	: IS 5 HEADER GAP ZEROS COMPLETE
8688	052424	001372				BNE	12\$	: IF NOT BRANCH
8689	052426	013737	051704	052512		MOV	@#RSYNC, @#WORD	: SYNC WORD
8690	052434	004767	000056			JSR	PC, READ	: READ HEADER (DATA) SYNC
8691	052440	005737	051716			TST	@#NOSYNC	
8692	052444	001404				BEQ	16\$	: IF NOT ERROR COMMAND BRANCH

8693	052446	032710	001000			BIT	#DTSY, @R0	: SYNC. DETECTED
8694	052452	001415				BEQ	13\$	: IF ZERO BRANCH OUT
8695	052454	000403				BR	17\$	: IF NOT ZERO BRANCH TO ERROR
8696	052456	032710	001000		16\$:	BIT	#DTSY, @R0	: SYNC. DETECTED?
8697	052462	001011				BNE	13\$	: BRANCH IF YES
8698	052464	012737	000006	051722	17\$:	MOV	#5, @#ERWORD	: ERROR WORD NO.
8699	052472	013737	051704	001124		MOV	@#RSYNC, @#SGDDAT	: SYNC WORD
8700	052500	012737	104002	051620		MOV	#104002, @#HEDSYN	
8701	052506				13\$:			
8702	052506	012601				MOV	(SP)+, R1	: ; POP STACK INTO R1
8703	052510	000201				RTS	R1	
8704								
8705								
8706								
8707								
8708								
8709								
8710								
8711								

```

8712
8713
8714
8715
8716
8717
8718
8719 052512 000000
8720 052514 000000
8721
8722
8723
8724
8725 052516
8726 052516 010246
8727 052520 012705 000002
8728 052524 012710 000001
8729 052530 006037 052512
8730 052534 103002
8731 052536 052710 000020
8732 052542 012702 000007
8733 052546 052710 000012
8734 052552 005737 047652
8735 052556 001411
8736 052560 032710 000020
8737 052564 001404
8738 052566 012737 177777 047644
8739 052574 000402
8740 052576 005037 047644
8741 052602 012746 000001
8742 052606 006037 052512
8743 052612 103002
8744 052614 012716 000021
8745 052620 012610
8746 052622 005737 047652
8747 052626 001404
8748 052630 005237 047664
8749 052634 004737 047702
8750 052640 052710 000002
8751 052644 005737 047652
8752 052650 001411
8753 052652 032710 000020
8754 052656 001404
8755 052660 012737 177777 047644
8756 052666 000402
8757 052670 005037 047644
8758 052674 012746 000001
8759 052700 006037 052512
8760 052704 103002
8761 052706 012716 000021
8762 052712 012610
8763 052714 005737 047652
8764 052720 001404
8765 052722 005237 047664

```

```

*****
*READ ONE WORD IN "WORD"
*****

```

```

WORD: 0
COMPA: 0

```

READ:

```

MOV R2,-(SP) ;: PUSH R2 ON STACK
MOV #2,R5 ;: WORD COUNTER
MOV #DMD,@R0 ;: SET DIAG. MODE
ROR @#WORD ;: CHECKING IF THERE IS A ONE
BCC 1$ ;: IF NO ONE BRANCH
BIS #MRD,@R0 ;: SET BIT 4 IF DATA HAS ONE
1$: MOV #7,R2 ;: BYTE COUNTER
BIS #MSTCK!MCLK,@R0 ;: SET CLOCK, DATA IF ANY, SECTOR
TST @#TSECCG ;: IS THIS BIT TO GENERATE AND TEST ECC
BEQ 6$ ;: BRANCH IF NO
BIT #MRD,@R0 ;: IS DATA BIT A ONE
BEQ 5$ ;: BRANCH IF DATA BIT IS 0
MOV #-1,@#ECDATA ;: ECC DATA BIT IS A ONE
BR 6$ ;: BRANCH
5$: CLR @#ECDATA ;: ECC DATA BIT IS A 0
6$: MOV #DMD,-(SP) ;: KEEP ONLY DIAG. MODE
ROR @#WORD ;: CHECKING IF THERE IS A ONE
BCC 2$ ;: IF NO ONE BRANCH
MOV #MRD!DMD,(SP)+@R0 ;: (SP) : KEEP DATA AND DIAG. MODE
TST @#TSECCG ;: PUT IN DATA, RESET CLOCK, SECTOR
BEQ 3$ ;: IS ECC TO BE GENERATED FOR THIS BIT
INC @#DATENV ;: BRANCH IF NO
JSR PC,@#ECTEST ;: NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
3$: BIS #MCLK,@R0 ;: GO TO GENERATE AND TEST ECC
TST @#TSECCG ;: SET CLOCK
BEQ 8$ ;: IS THIS BIT TO GENERATE ECC
BIT #MRD,@R0 ;: BRANCH IF NO
BEQ 7$ ;: IS DATA BIT A ONE
MOV #-1,@#ECDATA ;: BRANCH IF DATA BIT IS = 0
BR 8$ ;: ECC DATA BIT IS A ONE
7$: CLR @#ECDATA ;: BRANCH
8$: MOV #DMD,-(SP) ;: ECC DATA BIT IS = 0
ROR @#WORD ;: KEEP DIAG. MODE
BCC 4$ ;: CHECKING IF THERE IS A ONE
MOV #MRD!DMD,(SP)+@R0 ;: BRANCH IF NO ONE
TST @#TSECCG ;: KEEP DIAG. MODE AND DATA
BEQ 9$ ;: SET DATA, DIAG. MODE, CLEAR CLOCK
INC @#DATENV ;: IS THIS BIT TO GENERATE ECC
;: BRANCH IF NO
;: NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE

```

8766	052726	004737	047702
8767	052732	005302	
8768	052734	001341	
8769	052736	005305	
8770	052740	001300	
8771	052742	012602	
8772	052744	000207	
8773			
8774			
8775			
8776			
8777			
8778			

95:

JSR  
DEC  
BNE  
DEC  
BNE  
MOV  
RTS

PC, @#ECTEST  
R2  
3\$  
R5  
1\$  
(SP)+, R2  
PC

; GO TO GENERATE AND TEST ECC  
; BYTE COUNTER  
; BRANCH IF ONE BYTE NOT COMPLETE.  
; WORD COUNTER  
; BRANCH IF ONE WORD NOT COMPLETE  
; ; POP STACK INTO R2

```

8779
8780
8791
8782
8783
8784
8785
8786
8787
8788 052746 000000
8789
8790
8791
8792
8793 052750
8794 052750 010046
8795 052752 010246
8796 052754 010346
8797 052756 010546
8798 052760 012705 000002
8799 052764 012710 000001
8800 052770 012702 000007
8801 052774 012710 000013
8802 053000 032710 000040
8803 053004 001406
8804 053006 012737 177777 047644
8805 053014 000261
8806 053016 006003
8807 053020 000404
8808 053022 005037 047644
8809 053026 000241
8810 053030 006003
8811 053032 012710 000001
8812 053036 005737 047652
8813 053042 001404
8814 053044 005237 047664
8815 053050 004737 047702
8816 053054 052710 000002
8817 053060 032710 000040
8818 053064 001406
8819 053066 012737 177777 047644
8820 053074 000261
8821 053076 006003
8822 053100 000404
8823 053102 005037 047644
8824 053106 000241
8825 053110 006003
8826 053112 012710 000001
8827 053116 005737 047652
8828 053122 001404
8829 053124 005237 047664
8830 053130 004737 047702
8831 053134 005302
8832 053136 001346

```

```

*****
;WRITE ONE WORD WHICH COMES BACK IN "MMWORD"
*****

```

MMWORD: 0

WRITE:

```

MOV R0,-(SP) ;: PUSH R0 ON STACK
MOV R2,-(SP) ;: PUSH R2 ON STACK
MOV R3,-(SP) ;: PUSH R3 ON STACK
MOV R5,-(SP) ;: PUSH R5 ON STACK
MOV #2, R5 ;: WORD COUNTER
MOV #1, @R0 ;: SET DIAG. MODE
MOV #7, R2 ;: BYTE COUNTER
1S: MOV @MSTCK!MCLK!DMD, @R0 ;: ARC: SET SECTOR AND CLOCK
BIT @MWR, @R0 ;: CHECK WRITEBIT IN MAINT. REG.
BEQ 2S ;: BRANCH IF ZERO
MOV #-1, @#ECDATA ;: ECC DATA BIT IS A ONE
SEC ;: SET CARRY
ROR R3 ;: MOVE 1 FORWARD
BR 3S
2S: CLR @#ECDATA ;: ECC DATA BIT IS = 0
CLC ;: CLEAR CARRY
ROR R3 ;: MOVE 0 FOR MMWORD
3S: MOV @DMD, @R0 ;: CLEAR SECTOR AND CLOCK
TST @#TSECCG ;: IS THIS BIT TO GENERATE ECC
BEQ 4S ;: BRANCH IF NO
INC @#DATENV ;: NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
JSR PC, @#ECTEST ;: GO TO GENERATE AND TEST ECC
4S: BIS @MCLK, @R0 ;: SET CLOCK
BIT @MWR, @R0 ;: CHECK WRITE BIT IN MAINT. REG.
BEQ 5S ;: BRANCH IF ZERO
MOV #-1, @#ECDATA ;: ECC DATA BIT IS A ONE
SEC ;: SET CARRY
ROR R3 ;: MOVE 1 FOR MMWORD
BR 6S
5S: CLR @#ECDATA ;: ECC DATA BIT IS ZERO
CLC ;: CLEAR CARRY
ROR R3 ;: MOVE 0 FOR MMWORD
6S: MOV @DMD, @R0 ;: CLEAR CLOCK
TST @#TSECCG ;: IS THIS BIT TO GENERATE ECC
BEQ 7S ;: BRANCH IF NO
INC @#DATENV ;: NUMBER OF CLOCKS GIVEN FOR DATA ENVELOPE
JSR PC, @#ECTEST ;: GO TO GENERATE AND TEST ECC
7S: DEC R2 ;: COUNT FOR BYTE END
BNE 4S ;: IF NOT BYTE END BRANCH

```

```

8833 053140 005305 DEC R5 ;COUNT FOR WORD END
8834 053142 001312 BNE 15 ;IF NOT WORD END BRANCH
8835 053144 010337 MOV R3, @#WORD ;STORE WORD
8836 053150 012605 MOV (SP)+,R5 ;POP STACK INTO R5
8837 053152 012603 MOV (SP)+,R3 ;POP STACK INTO R3
8838 053154 012602 MOV (SP)+,R2 ;POP STACK INTO R2
8839 053156 012600 MOV (SP)+,R0 ;POP STACK INTO R0
8840 053160 000207 RTS PC
8841
8842
8843
8844
8845
8846

```



```

8847
8848
8849
8850
8851
8852
8853
8854
8855
8856
8857 053162 000000
8858 053164 000400
8859 053166 000000
8860 053170
8861 053170 011137 053162
8862 053174 012102
8863 053176 012137 052514
8864 053202 010046
8865 053204 010146
8866 053206 010246
8867 053210 010346
8868 053212 010446
8869 053214 012701 000016
8870 053220 012703 054526
8871 053224 012723 177777
8872 053230 005301
8873 053232 001374
8874 053234 013700 015000
8875 053240 013746 053164
8876 053244 163716 053162
8877 053250 011637 053166
8878 053254 012604
8879 053256 005737 015140
8880 053262 001403
8881 053264 012737 177777 047652
8882 053272 012703 053520
8883 053276 004737 052750
8884 053302 013723 052746
8885 053306 005302
8886 053310 001372
8887 053312 005704
8888 053314 001406
8889 053316 004737 052750
8890 053322 013723 052746
8891 053326 005304
8892 053330 001372
8893 053332 005037 047652
8894 053336 012701 000002
8895 053342 004767 177402
8896 053346 013723 052746
8897 053352 005301
8898 053354 001372
8899 053356 004767 177366
8900 053362 013723 052746

```

```

*****
*WRITE DATA
*****

```

```

COUNT: 0
FORMAT: 256.
ZWORDS: 0
MRDATA:
MOV (R1), @#COUNTD ;STORE NO. OF WORDS TO BE WRITTEN
MOV (R1)+, R2 ;SAME IN R2
MOV (R1)+, @#COMPA ;COMPARE OR NOT
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 #14, R1 ;NO. OF TOLERANCE GAP WORDS
MOV #TOLGAP, R3 ;START OF TOLERANCE GAP TABLE
1$: MOV #-1, (R3)+ ;MAKE IT 177777
DEC R1 ;IS 14 COMPLETED
BNE 1$ ;IF NO BRANCH
MOV @#RHMR, RO ;RO CONTAINS MAINTANENCE REG.
MOV @#FORMAT, -(SP)
SUB @#COUNTD, (SP)
MOV (SP), @#ZWORDS ;NO. OF ZERO WORDS TO BE WRITTEN
MOV (SP)+, R4
TST @#TSECC ;IS THIS AN ECC TEST
BEQ 7$ ;BRANCH IF NO
MOV #-1 @#TSECCG ;THESE BITS ARE TO GENERATE ECC
7$: MOV #DISK, R3 ;SIMULATED DISK AREA
2$: JSR PC, @#WRITE ;WRITE ON SIMULATED DISK
MOV @#WORD, (R3)+ ;STORE ON SIMULATED DISK
DEC R2
BNE 2$
TST R4 ;ANY ZEROS TO BE WRITTEN
BEQ 4$ ;BRANCH IF NONE TO BE WRITTEN
3$: JSR PC, @#WRITE ;WRITE ZEROS ON SIMULATED DISK
MOV @#WORD, (R3)+ ;STORE
DEC R4
BNE 3$
4$: CLR @#TSECCG ;NO MORE ECC TO BE GENERATED
MOV #2, R1
5$: JSR PC, WRITE ;WRITE ECC1 AND ECC2 ON SIMULATED DISK
MOV @#WORD, (R3)+ ;STORE ON WEEC1 AND WEEC2
DEC R1
BNE 5$
JSR PC, WRITE ;WRITE DATA GAP
MOV @#WORD, (R3)+ ;STORE

```

# K04

MAINDEC-11-DZRPT-D, RJF04 DISKLESS RH11 TEST-PART 2  
DZRPTD.SUB DISK SIMULATION

MACY11 27(663) 7-OCT-75 17:33 PAGE 213

SEQ 0255

8901	053366	012701	000016		MOV	#14., R1	
8902	053372	004737	052750	6\$:	JSR	PC, @#WRITE	;WRITE TOLERANCE GAP ZEROS
8903	053376	013723	052746		MOV	@#WORD, (R3)+	;STORE
8904	053402	005301			DEC	R1	
8905	053404	001372			BNE	6\$	
8906	053406	012604			MOV	(SP)+, R4	::POP STACK INTO R4
8907	053410	012603			MOV	(SP)+, R3	::POP STACK INTO R3
8908	053412	012602			MOV	(SP)+, R2	::POP STACK INTO R2
8909	053414	012601			MOV	(SP)+, R1	::POP STACK INTO R1
8910	053416	012600			MOV	(SP)+, R0	::POP STACK INTO R0
8911	053420	000201			RTS	R1	
8912							
8913							
8914							
8915							
8916							
8917							
8918							
8919							

8920  
8921  
8922  
8923  
8924  
8925  
8926  
8927  
8928  
8929  
8930  
8931  
8932  
8933  
8934  
8935  
8936  
8937  
8938  
8939  
8940  
8941  
8942  
8943  
8944  
8945  
8946  
8947  
8948  
8949  
8950  
8951  
8952

053422 000023  
053470 000001  
053472 000004  
053502 000001  
053504 000005  
053516 000001  
053520  
053520 000400  
054520 000001  
054522 000001  
054524 000001  
054526 000016

```
*****  
:WRITE HEADER AND DATA  
:  
:  
:THIS IS THE SIMULATED DISK  
:ONLY ONE SECTOR OF SPACE IS ALLOWED  
:*****  
:*****
```

SECGAP: .BLKM 19.  
MSSYNC: .BLKM 1  
HEADER: .BLKM 4  
MCRC: .BLKM 1  
HEGAP: .BLKM 5  
HDMSYN: .BLKM 1  
SILOTB:  
DISK: .BLKM 256.  
MECC1: .BLKM 1  
MECC2: .BLKM 1  
DTAGAP: .BLKM 1  
TOLGAP: .BLKM 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  
; (USED IN SILO TEST AS SILO TABLE)  
: DATA SPACE  
: ECC1  
: ECC2  
: DATA GAP 2 BYTES OF 0  
: TOLERANCE GAP 28 BYTES OF 0

```

8953 ;*****
8954 ;WRITE HEADER AND DATA
8955 ;*****
8956
8957
8958
8959 054562 000000 RNCTR1: .WORD 0 ;'RUN' LINE STALL COUNTER
8960 054564 011637 015130 COMMD: MOV (SP),@#PCJSR ;SAVE PC OF JSR + 4
8961 054570 162737 000004 015130 SUB #4,@#PCJSR ;SAVE PC OF JSR
8962 054576 010046 MOV R0,-(SP) ;PUSH R0 ON STACK
8963 054600 010146 MOV R1,-(SP) ;PUSH R1 ON STACK
8964 054602 010246 MOV R2,-(SP) ;PUSH R2 ON STACK
8965 054604 010346 MOV R3,-(SP) ;PUSH R3 ON STACK
8966 054606 010446 MOV R4,-(SP) ;PUSH R4 ON STACK
8967 054610 010546 MOV R5,-(SP) ;PUSH R5 ON STACK
8968 054612 012777 000001 140160 MOV #DMD,@RHMR ;SET DIAGNOSTIC MODE BIT
8969 054620 052777 000001 140132 BIS #GO,@RHCSI ;SET 'GO' BIT & STALL 'TILL 'RUN'
8970 ;FUNCTION CODE IS SET UP BY THE TEST
8971 054626 012737 000113 054562 RNCTR1: MOV #75,@#RNCTR1 ;LOAD STALL COUNTER = APPROX 450US FOR 11/50 CPU
8972 054634 005337 054562 IS: DEC @#RNCTR1 ;COUNT DOWN 1 TIME
8973 054640 001375 BNE IS ;CONTINUE UNTIL = 0
8974
8975 054642 013746 054726 MOV @#WSECTR,-(SP) ;GET DESIRED SECTOR/TRACK
8976 054646 042716 177740 BIC #177740,(SP) ;MAKE ONLY SECTOR
8977 054652 012637 054662 MOV (SP)+,@#WTRK ;SAVE SECTOR
8978 054656 004137 055634 JSR R1,@#SEARCH ;SEARCH SECTOR & ISSUE SECTOR CLOCKS
8979
8980 054662 000000 WTRK: .WORD 0 ;SECTOR NO.
8981 054664 012701 000240 MOV #+NOP,R1 ;GOING TO MOVE NOPS
8982 054670 010137 054736 MOV R1,@#SEGPER ;NOP INTO SEGAP
8983 054674 010137 054740 MOV R1,@#FSYNER ;NOP INTO FSYNER
8984 054700 010137 054742 MOV R1,@#ERHEAD ;NOP INTO ERHEAD
8985 054704 010137 054744 MOV R1,@#ERCRC ;NOP INTO ERCRC
8986 054710 010137 054746 MOV R1,@#ERHDGP ;NOP INTO ERHDGAP
8987 054714 010137 054750 MOV R1,@#HDESYN ;NOP INTO HDESYN
8988 054720 004137 055020 JSR R1,@#WRHEAD ;WRITE THE HEADER
8989
8990 054724 000000 WCYL: 0 ;CYLINDER
8991 054726 000000 WSECTR: 0 ;SECTOR AND TRACK
8992 054730 000000 WKEY1: 0 ;KEY1
8993 054732 000000 WKEY2: 0 ;KEY2
8994 054734 000000 GCRC: 0 ;GOOD CRC
8995
8996 054736 000240 SEGPER: NOP ;IF "ERROR 6" INSERTED BY
8997 ;WRHEAD SUBROUTINE, THEN
8998 ;SECTOR GAP GOING ON DISK
8999 ;IS NOT RIGHT.
9000
9001 ;WORD NO. CONTAINS WHICH
9002 ;WORD IS WRONG, THAT IS
9003 ;FIRST OF TENTH, OR WHAT EVER.
9004
9005 ;BAD WORD IS WHAT IS GOING ON DISK
9006

```

```

9007 054740 000240          FSYNER: NOP
9008
9009
9010
9011
9012
9013
9014
9015
9016
9017
9018
9019
9020
9021 054742 000240          ERHEAD: NOP
9022
9023
9024
9025
9026
9027
9028
9029
9030
9031
9032
9033
9034
9035 054744 000240          ERCRC:  NOP
9036
9037
9038
9039
9040
9041
9042
9043
9044
9045 054746 000240          ERHDGP: NOP
9046
9047
9048
9049
9050
9051
9052
9053
9054
9055
9056
9057
9058 054750 000240          HDESYN: NOP
9059
9060
    
```

```

;IF "ERROR 6" INSERTED BY
;WRHEAD SUBROUTINE, THEN
;THE LAST 0 BYTE OF SECTOR
;GAP OF FIRST SYNC. BYTE
;AFTER SECTOR GAP IS IN
;ERROR.
    
```

```

;WORD NO. CONTAINS 20
;RIGHT BYTE IS SECTOR GAP
;LEFT BYTE IS SYNC. BYTE.

;BAD WORD IS WHAT IS GOING ON
;DISK
    
```

```

;IF "ERROR 6" INSERTED BY
;WRHEAD SUBROUTINE, THEN
;HEADER GOING ON DISK
;IS WRONG.
    
```

```

;WORD NO 1 = CYLINDER NO
;WORD NO 2 = SECTOR/TRACK
;WORD NO 3 = KEY1
;WORD NO 4 = KEY2
    
```

```

;BAD WORD IS WHAT IS GOING ON
;DISK
    
```

```

;IF "ERROR 6" INSERTED BY
;WRHEAD SUBROUTINE, THEN CRC WRITTEN
;ON DISK IS IN ERROR.
    
```

```

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

;WORD NO IS 5
    
```

```

;IF "ERROR 6" INSERTED BY
;WRHEAD SUBROUTINE, THEN HEADER
;GAP GOING ON DISK IS WRONG.
    
```

```

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

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

```

;IF "ERROR 6" INSERTED BY
;WRHEAD SUBROUTINE, THEN LAST
    
```

```

9061 ;HEADER GAP BYTE OR HEADER
9062 ;SYNC BYTE GOING ON DISK IS WRONG.
9063
9064 ;WORD NO = 5
9065
9066 ;BAD DATA IS WHAT IS GOING
9067 ;ON DISK, RIGHT BYTE IS HEADER
9068 ;GAP 0'S BYTE, LEFT BYTE IS HEADER
9069 ;GAP SYNC.
9070
9071
9072 054752 005737 015122 TST @#ERFLGS ;ARE ANY ERRORS DETECTED ?
9073 054756 001034 BNE FOUT ;IF YES BRANCH
9074 054760 004137 053170 JSR R1,@#MRDATA ;WRITE THE DATA
9075
9076 054764 000000 FNMWORD: .WORD 0 ;FORMAT COMMAND NO. OF DATA
9077 054766 000000 .WORD 0
9078
9079 054770 FOUT: MOV (SP)+,R5 ;: POP STACK INTO R5
9080 054770 012605 MOV (SP)+,R4 ;: POP STACK INTO R4
9081 054772 012604 MOV (SP)+,R3 ;: POP STACK INTO R3
9082 054774 012603 MOV (SP)+,R2 ;: POP STACK INTO R2
9083 054776 012602 MOV (SP)+,R1 ;: POP STACK INTO R1
9084 055000 012601 MOV (SP)+,R0 ;: POP STACK INTO R0
9085 055002 012600 RTS PC ;EXIT
9086 055004 000207

```

9087  
9088  
9089  
9090  
9091  
9092  
9093  
9094  
9095  
9096  
9097  
9098  
9099  
9100  
9101  
9102  
9103  
9104  
9105  
9106  
9107  
9108  
9109  
9110  
9111  
9112  
9113  
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

055006 000000  
055010 000000  
055012 000000  
055014 000000  
055016 000000  
  
055020 012137 055006  
055024 012137 055010  
055030 012137 055012  
055034 012137 055014  
055040 012137 055016  
055044 010146  
055046 012701 053422  
055052 013700 015000  
055056 012710 000001  
055062 012705 000002  
055066 052710 000010  
055072 012710 000013  
055076 032710 000040  
055102 001403  
055104 000261  
055106 006003  
055110 000402  
055112 000241  
055114 006003  
055116 012710 000001  
055122 012702 000007  
055126 052710 000002  
055132 032710 000040  
055136 001403  
055140 000261  
055142 006003  
055144 000402  
055146 000241  
055150 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  
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  
BIT #MWR, @R0 ;: CHECK WRITE BIT IN MAINT. REG.  
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 CLOCK  
BIT #MWR, @R0 ;: CHECK WRITE BIT IN MAINT.REG.  
BEQ 5\$ ;: BRANCH IF ZERO  
SEC ;: SET CARRY  
ROR R3 ;: MOVE ONE FORWARD  
BR 6\$  
5\$: CLC  
ROR R3

```

9141 055152 012710 000001      6S:  MOV      #0MD,  @R0
9142 055156 005302             DEC      R2
9143 055160 001362             BNE     4S
9144 055162 005305             DEC      R5
9145 055164 001342             BNE     1S
9146 055166 010321             MOV      R3,  (R1)+
9147 055170 005703             TST     R3
9148 055172 001414             BEQ     7S
9149 055174 012737 000001 051722  MOV      @1,@#ERWORD
9150 055202 005037 001124             CLR      @#SGDDAT
9151 055206 010337 001126             MOV      R3,  @#SBDDAT
9152 055212 012737 104006 054736  MOV      #104006,@#SEGP
9153 055220 000137 055626             JMP      @#17S ; BRANCH OUT
9154 055224 012702 000022      7S:  MOV      #18.,  R2 ; COUNT NO. OF SECTOR GAP
9155 055230 012737 000024 051722 10S:  MOV      #20.,@#ERWORD ; COUNT TO GIVE ERROR WORD
9156 055236 004737 052750             JSR     PC,  @#WRITE ; WRITE SECTOR GAP
9157 055242 013721 052746             MOV      @#WORD,(R1)+ ; STORE SECTOR GAP WORD
9158 055246 001413             BEQ     11S
9159 055250 160237 051722             SUB     R2,@#ERWORD ; IF NOT GET ERROR WORD NO.
9160 055254 005037 001124             CLR      @#SGDDAT ; GOOD WORD
9161 055260 013737 052746 001126             MOV      @#WORD,@#SBDDAT ; BAD WORD
9162 055266 012737 104006 054736  MOV      #104006,@#SEGP ; STORE "ERROR 6" IN SEGP
9163 055274 000554             BR      17S ; BRANCH OUT
9164 055276 005302      11S:  DEC      R2 ; HAVE 18 WORDS OF ZEROS BEEN WRITTEN
9165 055300 001353             BNE     10S ; IF NOT BRANCH
9166 ; AT THIS POINT THE SECTOR FOUND FLOP SHOULD
9167 ; BE HIGH. SO THAT THE HEADER SYNC BYTE CAN BE GIVEN
9168 ; HOWEVER IN THE DRIVE TIMING ERROR TEST THE REST OF THE ROUTINE
9169 ; IS ABORTED
9170 055302 005737 015142             TST     @#TESDTE ; IS THIS A DRIVE TIMING ERROR
9171 055306 001147             BNE     17S ; BRANCH OUT IF YES
9172 055310 004737 052750             JSR     PC,  @#WRITE ; WRITE ONE SECTOR GAP 0 BYTE
9173 ; AND ONE SYNC. BYTE = 230
9174 055314 013711 052746             MOV      @#WORD,(R1) ; SAVE 0 BYTE AND SYNC BYTE
9175 055320 023721 051704             CMP     @#RSYNC,(R1)+ ; IF SYNC. BYTE RIGHT
9176 055324 001414             BEQ     12S ; IF YES BRANCH
9177 055326 012737 000024 051722  MOV      #20.,@#ERWORD ; IF NOT GET READY FOR ERROR
9178 055334 013737 051704 001124  MOV      @#RSYNC,@#SGDDAT ; GOOD WORD
9179 055342 014137 001126             MOV      -(R1),@#SBDDAT ; BAD WORD
9180 055346 012737 104006 054740  MOV      #104006,@#FSYNER ; INSERT "ERROR 6" IN FSYNER
9181 055354 000524             BR      17S ; BRANCH OUT
9182 055356 012702 000004      12S:  MOV      #4,  R2 ; FOUR HEADER WORDS
9183 055362 012703 055006             MOV      #SCYL, R3 ; POINTER FOR HEADER TABLE
9184 055366 012737 000005 051722 13S:  MOV      #5,@#ERWORD ; ERROR WORD NO SET
9185 055374 004737 052750             JSR     PC,  @#WRITE ; WRITE 4 HEADER WORDS
9186 055400 013711 052746             MOV      @#WORD,(R1) ; STORE WRITTEN WORD
9187 055404 022321             CMP     (R3)+, (R1)+ ; IS IT RIGHT?
9188 ; IF GOOD BRANCH
9189 ; IF NOT GET READY FOR PRINT
9190 055410 160237 051722             SUB     R2,@#ERWORD ; WORD NO
9191 055414 014337 001124             MOV      -(R3),@#SGDDAT ; GOOD DATA
9192 055420 014137 001126             MOV      -(R1),@#SBDDAT ; BAD DATA
9193 055424 012737 104006 054742  MOV      #104006,@#ERHEAD ; INSERT "ERROR 6"
9194 055432 000475             BR      17S ; BRANCH OUT

```





9237  
9238  
9239  
9240  
9241  
9242  
9243  
9244  
9245  
9246  
9247  
9248  
9249  
9250  
9251  
9252  
9253  
9254  
9255  
9256  
9257  
9258  
9259  
9260  
9261  
9262  
9263  
9264  
9265  
9266  
9267  
9268  
9269  
9270  
9271  
9272  
9273  
9274  
9275  
9276  
9277  
9278  
9279  
9280  
9281  
9282  
9283  
9284  
9285  
9286  
9287  
9288  
9289  
9290

055632 000000  
055634 012137 055632  
055640 010046  
055642 010146  
055644 010246  
055646 010346  
055650 010446  
055652 010546  
055654 013700 015000  
055660 013703 055632  
055664 012710 000001  
055670 052710 000010  
055674 042710 000010  
055700 052710 000010  
055704 042710 000010  
055710 052710 000014  
055714 012710 000001  
055720 005703  
055722 001461

\*\*\*\*\*  
:SEARCH SECTOR  
\*\*\*\*\*

\* R0=RHMR ADDRESS  
\* R1=PASSED ARGUMENT (SECTOR SEARCHED FOR)  
\* R2=CLOCK COUNT (PER BYTE)  
\* R3=SECTOR COUNTER FROM R1  
\* R5=BYTES PER WORD COUNT  
\*BEFORE INDEX IS GIVEN TWO SECTOR CLOCKS ARE GIVEN TO RESET  
\*SECTOR PULSE IN CASE IT IS SET  
\*AT BEGINNING OF EACH SECTOR ONE SECTOR CLOCK HAS TO RISE  
\*BEFORE CLOCK THEN EVERY EIGHT CLOCKS ONE SECTOR CLOCK IS  
\*IDENTICAL WITH CLOCK  
\*NUMBERING THE SECTOR CLOCKS 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 THEN HAS 608 SECTOR CLOCKS  
\*THE NEXT SECTOR THEN HAS ANOTHER 608 SECTOR CLOCKS  
\*AND SO ON

SECTR: 0 ;SECTOR SEARCHED FOR  
SEARCH: MOV (R1)+, @#SECTR ;SAVE SECTOR SEARCHED FOR  
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 @#RHMR, R0 ;NOW R0 HAS MAINTENANCE REG. ADR.  
MOV @#SECTR, R3 ;SECTOR COUNTER  
MOV #DMD, @RD ;SET DIAGNOSTIC MODE  
BIS #MSTCK, @RD ;SET SECTOR CLOCK  
BIC #MSTCK, @RD ;CLEAR SECTOR CLOCK  
BIS #MSTCK, @RD ;SET SECTOR CLOCK  
BIC #MSTCK, @RD ;CLEAR SECTOR CLOCK  
;THE ABOVE TWO SECTOR CLOCKS ARE GIVEN FOR  
;RESETTING SECTOR PULSE  
;IN CASE IT STARTS SET  
BIS #MINX!MSTCK, @RD ;SET INDEX AND SECTOR CLOCK  
MOV #DMD, @RD ;RESET INDEX AND SECTOR CLOCK  
TST R3 ;IF SECTOR REQUIRED JUMP OUT  
BEQ 7\$ ;BRANCH OF SECTOR ZERO REQUIRED

```

9291                                     ;NOW THE 304 WORDS WILL START
9292
9293
9294
9295                                     ;*FOR FIRST BYTE SECTOR CLOCK WILL GO HIGH THEN CLOCK WILL GO HIGH
9296                                     ;*BOTH WILL COME DOWN TOGETHER THEN SEVEN CLOCKS WILL BE GIVEN
9297                                     ;*FOR SECOND BYTE AND ALL OTHER BYTES TILL NEXT SECTOR SECTOR CLOCK
9298                                     ;*WILL BE IDENTICAL WITH ONE CLOCK
9299
9300
9301                                     ;ONE WORD ONLY
9302
9303 055724 012702 000010 1$: MOV #8., R2 ;BYTE COUNTER
9304 055730 012705 000002 MOV #2, R5 ;BYTES PER WORD
9305 055734 052710 000010 BIS #MSTCK,@R0 ;SET SECTOR CLOCK
9306 055740 052710 000002 BIS #MCLK,@R0 ;SET CLOCK
9307 055744 000402 BR 3$ ;BRANCH TO CLEAR SECTOR AND CLOCK
9308 055746 052710 000012 2$: BIS #MSTCK!MCLK,@R0 ;SET SECTOR AND CLOCK
9309 055752 042710 000012 3$: BIC #MSTCK!MCLK,@R0 ;CLEAR SECTOR AND CLOCK
9310 055756 052710 000002 8$: BIS #MCLK,@R0 ;SET CLOCK
9311 055762 042710 000002 BIC #MCLK,@R0 ;CLEAR CLOCK
9312 055766 005302 DEC R2 ;BYTE COUNTER
9313 055770 001372 BNE 8$ ;BRANCH IF BYTE NOT COMPLETE
9314 055772 012702 000007 MOV #7, R2 ;SETUP FOR SECOND BYTE
9315 055776 005305 DEC R5 ;IS WORD COMPLETE?
9316 056000 001362 BNE 2$ ;BRANCH IF NOT COMPLETE
9317                                     ;TO GIVE SECTOR CLOCK AND CLOCK
9318
9319
9320                                     ;NOW 303 WORDS ARE LEFT AND ALL ARE IDENTICAL
9321
9322 056002 012701 000457 MOV #303., R1 ;WORDS PER SECTOR COUNTER
9323 056006 012705 000002 4$: MOV #2, R5 ;BYTES PER WORD COUNTER
9324 056012 012702 000007 5$: MOV #7, R2 ;BYTE COUNTER (CLOCK COUNTER)
9325 056016 052710 000012 BIS #MSTCK!MCLK,@R0 ;SET SECTOR CLOCK AND CLOCK
9326 056022 042710 000012 BIC #MSTCK!MCLK,@R0 ;CLEAR SECTOR CLOCK AND CLOCK
9327 056026 052710 000002 6$: BIS #MCLK,@R0 ;SET CLOCK
9328 056032 042710 000002 BIC #MCLK,@R0 ;RESET CLOCK
9329 056036 005302 DEC R2 ;IS BYTE COMPLETE?
9330 056040 001372 BNE 6$ ;BRANCH IF NOT COMPLETE
9331 056042 005305 DEC R5 ;IS WORD COMPLETE?
9332 056044 001362 BNE 5$ ;BRANCH IF NOT
9333 056046 005301 DEC R1 ;IS SECTOR COMPLETE
9334 056050 001356 BNE 4$ ;BRANCH IF NOT
9335 056052 052710 000010 BIS #MSTCK,@R0 ;SET SECTOR
9336 056056 042710 000010 BIC #MSTCK,@R0 ;CLEAR SECTOR
9337 056062 005303 DEC R3 ;IS REQUIRED NO OF SECTORS COMPLETE
9338 056064 001317 BNE 1$ ;BRANCH IF NOT
9339
9340 7$:
9341 056066 012605 MOV (SP)+,R5 ;POP STACK INTO R5
9342 056070 012604 MOV (SP)+,R4 ;POP STACK INTO R4
9343 056072 012603 MOV (SP)+,R3 ;POP STACK INTO R3
9344 056074 012602 MOV (SP)+,R2 ;POP STACK INTO R2

```

# H05

```
9345 056076 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
9346 056100 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
9347 056102 000201      RTS      R1
9348
9349
9350                      ;*****
9351                      ;*READ ONE SECTOR OF DATA
9352                      ;*****
9353
9354 056104 000000      RNO:      0              ;NO. OF WORDS READ
9355 056106 000000      RCOM:     0              ;EXTRA STORAGE
9356
9357
9358
9359 056110 012137 056104  REDATA:  MOV      (R1)+,@#RNO      ;SAVE NO. OF WORDS ONLY FOR INFORMATION
9360 056114 012137 056106      MOV      (R1)+,@#RCOM   ;EXTRA WORD ONLY FOR INFORMATION
9361 056120 010146      MOV      R1,-(SP)      ;PUSH R1 ON STACK
9362 056122 005737 015140      TST      @#TSECC       ;IS THIS AN ECC TEST
9363 056126 001403      BEQ      1$           ;BRANCH IF NO
9364 056130 012737 177777 047652  MOV      #-1,@#TSECCG   ;THESE BITS ARE TO GENERATE ECC
9365 056136 012702 000402 1$:      MOV      @256.,R2      ;256 WORDS PER SECTOR
9366                                     PLUS 2 ECC W JS
9367 056142 012703 053520      MOV      @DISK,R3      ;POINTE TO DISK SIMULATION
9368 056146 012337 052512 2$:      MOV      (R3)+,@#WORD   ;READY TO READ CONTENTS
9369 056152 004737 052516      JSR      PC,@#READ     ;READ
9370 056156 005302      DEC      R2           ;IS 256 WORDS DONE?
9371 056160 001372      BNE      2$          ;IF NOT BRANCH
9372 056162 005737 015140      TST      @#TSECC       ;IS THIS AN ECC TEST
9373 056166 001012      BNE      4$          ;BRANCH OUT IF YES
9374 056170 005037 047652      CLR      @#TSECCG      ;NO MORE ECC BITS ARE TO BE GENERATED
9375 056174 012702 000017 3$:      MOV      @15.,R2      ;ONE DATA GAP, 14 TOLERANCE GAP
9376 056200 012337 052512      MOV      (R3)+,@#WORD   ;READY TO READ CONTENTS OF WORD
9377 056204 004737 052516      JSR      PC,@#READ     ;READ
9378 056210 005302      DEC      R2           ;COUNT
9379 056212 001372      BNE      3$          ;BRANCH IF 14 NOT DONE
9380 056214
9381 056214 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
9382 056216 000201      RTS      R1           ;RETURN
```

```

9383
9384 056220
9385 056220 104400 056226
9386 056224 000423
9387
9388 056274
9389 056274 104402
9390 056276 012777 056220 136442
9391 056304 000000
9392
9393

```

```

*****
RPVECT:
        TYPE      +4      ;:TYPE ASCIZ STRING
        BR        645     ;:GET OVER THE ASCIZ
        ;:ASCIZ      /UNEXPECTED RPO4 INTERRUPT FROM PC = /
645:
        TYPOC      ;:TYPE FROM PC
        MOV        @RPVECT,@RPVEC ;:RESTORE TRAP RPO4 VECTOR
        HALT       ;:CHANGE TO CONTINUE
*****

```

```

9394
9395
9396
9397
9398
9399
9400
9401
9402
9403
9404
9405
9406
9407
9408
9409
9410
9411
9412
9413
9414
9415 056306
9416 056306 005037 051716
9417 056312 005037 015140
9418
9419
9420
9421 056316 005037 047652
9422
9423
9424
9425 056322 005037 015142
9426 056326
9427 056326 006137 177570
9428 056332 100511
9429
9430 056334 000416
9431
9432 056336 013746 000004
9433 056342 012737 056362 000004
9434 056350 005737 177060
9435 056354 012637 000004
9436 056360 000463
9437 056362 022626
9438 056364 012637 000004
9439 056370 000423
9440 056372
9441 056372 032737 000400 177570
9442 056400 001404
9443 056402 123767 177570 122472
9444 056410 001462
9445 056412 105767 122465
9446 056416 001421
9447 056420 126767 122471 122455

```

```

.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(STSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
; *AND LOAD THE ERROR FLAG (SERFLG) INTO DISPLAY<15:08>
; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
; *SW14=1 LOOP ON TEST
; *SW11=1 INHIBIT ITERATIONS
; *SW09=1 LOOP ON ERROR
; *SW08=1 LOOP ON TEST IN SWR<7:0>
; *CALL
; * SCOPE ;;SCOPE=IOT

$SCOPE:
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
; DRIVE TIMING ERROR TEST

15: CLR @#TESDTE

ROL @#SWR ; LOOP ON PRESENT TEST?
BMI $OVER ; YES IF SW14=1

; *****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR 6$ ; IF RUNNING ON THE "XOR" TESTER CHANGE
; THIS INSTRUCTION TO A "NOP" (NOP=240)
; SAVE THE CONTENTS OF THE ERROR VECTOR
MOV @#ERRVEC, -(SP) ; SET FOR TIMEOUT
MOV @#SS, @#ERRVEC ; TIME OUT ON XOR?
TST @#177060 ; RESTORE THE ERROR VECTOR
MOV (SP)+, @#ERRVEC ; GO TO THE NEXT TEST
BR $SVLAD ; CLEAR THE STACK AFTER A TIME OUT
5$: CMP (SP)+, (SP)+ ; RESTORE THE ERROR VECTOR
MOV (SP)+, @#ERRVEC ; LOOP ON THE PRESENT TEST
BR 7$

6$; *****END OF CODE FOR THE XOR TESTER*****
BIT @#BIT08, @#SWR ; LOOP ON SPEC. TEST?
BEQ 2$ ; BR IF NO
CMPB @#SWR, STSTNM ; ON THE RIGHT TEST? SWR<7:0>
BEQ $OVER ; BR IF YES
2$: TSTB SERFLG ; HAS AN ERROR OCCURRED?
BEQ 3$ ; BR IF NO
CMPB SERMAX, SERFLG ; MAX. ERRORS FOR THIS TEST OCCURRED?

```

# K05

9448	056426	101015					BHI	3\$		:: BR IF NO
9449	056430	032737	001000	177570			BIT	#BIT09, @#SWR		:: LOOP ON ERROR?
9450	056436	001404					BEQ	4\$		:: BR IF NO
9451	056440	016767	122444	122440	7\$:		MOV	\$LPERR, \$LPADR		:: SET LOOP ADDRESS TO LAST SCOPE
9452	056446	000443					BR	\$OVER		
9453	056450	105067	122427		4\$:		CLRB	\$ERFLG		:: ZERO THE ERROR FLAG
9454	056454	005067	122524				CLR	\$TIMES		:: CLEAR THE NUMBER OF ITERATIONS TO MAKE
9455	056460	000415					BR	1\$		:: ESCAPE TO THE NEXT TEST
9456	056462	032737	004000	177570	3\$:		BIT	#BIT11, @#SWR		:: INHIBIT ITERATIONS?
9457	056470	001011					BNE	1\$		:: BR IF YES
9458	056472	005767	122402				TST	\$PASS		:: IF FIRST PASS OF PROGRAM
9459	056476	001406					BEQ	1\$		:: INHIBIT ITERATIONS
9460	056500	005267	122400				INC	\$ICNT		:: INCREMENT ITERATION COUNT
9461	056504	026767	122474	122372			CMP	\$TIMES, \$ICNT		:: CHECK THE NUMBER OF ITERATIONS MADE
9462	056512	002021					BGE	\$OVER		:: BR IF MORE ITERATION REQUIRED
9463	056514	012767	000001	122362	1\$:		MOV	#1, \$ICNT		:: REINITIALIZE THE ITERATION COUNTER
9464	056522	016767	000044	122454			MOV	\$MXCNT, \$TIMES		:: SET NUMBER OF ITERATIONS TO DO
9465	056530	105267	122346		\$SVLAD:		INCB	\$STNM		:: COUNT TEST NUMBERS
9466	056534	011667	122346				MOV	(SP), \$LPADR		:: SAVE SCOPE LOOP ADDRESS
9467	056540	011667	122344				MOV	(SP), \$LPERR		:: SAVE ERROR LOOP ADDRESS
9468	056544	005067	122436				CLR	\$ESCAPE		:: CLEAR THE ESCAPE FROM ERROR ADDRESS
9469	056550	112767	000001	122337			MOVB	#1, \$ERMAX		:: ONLY ALLOW ONE(1) ERROR ON NEXT TEST
9470	056556	016737	122320	177570	\$OVER:		MOV	\$STNM, @#DISPLAY		:: DISPLAY TEST NUMBER
9471	056564	016716	122316				MOV	\$LPADR, (SP)		:: FUDGE RETURN ADDRESS
9472	056570	000002					RTI			:: FIXES PS
9473	056572	000004					\$MXCNT:	4		:: MAX. NUMBER OF ITERATIONS

```

9474 ;*****
9475
9476 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
9477
9478 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
9479 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
9480 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
9481 ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
9482 ;*REPLACED WITH SPACES.
9483 ;*CALL:
9484 ;*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
9485 ;*      TYPDS                    ;;GO TO THE ROUTINE
9486
9487 $TYPDS:
9488 056574 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
9489 056576 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
9490 056600 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
9491 056602 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
9492 056604 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
9493 056606 012746 020200      MOV      #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
9494 056612 016605 000020      MOV      20(SP),R5    ;;GET THE INPUT NUMBER
9495 056616 100004      BPL      1$           ;;BR IF INPUT IS POS.
9496 056620 005405      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
9497 056622 112766 000055 000001      MOV      #'-,1(SP)    ;;MAKE THE ASCII NUMBER NEG.
9498 056630 005000      CLR      R0           ;;ZERO THE CONSTANTS INDEX
9499 056632 012703 057010      MOV      #SDBLK,R3    ;;SETUP THE OUTPUT POINTER
9500 056636 112723 000040      MOV      #' ,(R3)+    ;;SET THE FIRST CHARACTER TO A BLANK
9501 056642 005002      CLR      R2           ;;CLEAR THE BCD NUMBER
9502 056644 016001 057000      MOV      $DTBL(R0),R1 ;;GET THE CONSTANT
9503 056650 160105      SUB      R1,R5        ;;FORM THIS BCD DIGIT
9504 056652 002402      BLT     4$           ;;BR IF DONE
9505 056654 005202      INC      R2           ;;INCREASE THE BCD DIGIT BY 1
9506 056656 000774      BR      3$
9507 056660 060105      4$:      ADD      R1,R5        ;;ADD BACK THE CONSTANT
9508 056662 005702      TST     R2           ;;CHECK IF BCD DIGIT=0
9509 056664 001002      BNE     5$           ;;FALL THROUGH IF 0
9510 056666 105716      TST     (SP)         ;;STILL DOING LEADING 0'S?
9511 056670 100407      BMI     7$           ;;BR IF YES
9512 056672 106316      5$:      ASLB   (SP)         ;;MSD?
9513 056674 103003      BCC     6$           ;;BR IF NO
9514 056676 116663 000001 177777      MOV      1(SP),-1(R3) ;;YES--SET THE SIGN
9515 056704 052702 000060      6$:      BIS   #'0,R2        ;;MAKE THE BCD DIGIT ASCII
9516 056710 052702 000040      7$:      BIS   #' ,R2        ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
9517 056714 110223      MOV      R2,(R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
9518 056716 005720      TST     (R0)+        ;;JUST INCREMENTING
9519 056720 020027 000010      CMP     R0,#10      ;;CHECK THE TABLE INDEX
9520 056724 002746      BLT     2$           ;;GO DO THE NEXT DIGIT
9521 056726 003002      BGT     8$           ;;GO TO EXIT
9522 056730 010502      MOV     R5,R2        ;;GET THE LSD
9523 056732 000764      BR      6$           ;;GO CHANGE TO ASCII
9524 056734 105726      8$:      TST     (SP)+        ;;WAS THE LSD THE FIRST NON-ZERO?
9525 056736 100003      BPL     9$           ;;BR IF NO
9526 056740 116663 177777 177776      MOV      -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
9527 056746 105013      9$:      CLAB   (R3)        ;;SET THE TERMINATOR

```



# M05

9528	056750	012605		MOV	(SP)+,R5	::POP STACK INTO R5
9529	056752	012603		MOV	(SP)+,R3	::POP STACK INTO R3
9530	056754	012602		MOV	(SP)+,R2	::POP STACK INTO R2
9531	056756	012601		MOV	(SP)+,R1	::POP STACK INTO R1
9532	056760	012600		MOV	(SP)+,R0	::POP STACK INTO R0
9533	056762	104400	057010	TYPE	\$DBLK	::NOW TYPE THE NUMBER
9534	056766	016666	000002 000004	MOV	2(SP),4(SP)	::ADJUST THE STACK
9535	056774	012616		MOV	(SP)+,(SP)	
9536	056776	000002		RTI		::RETURN TO USER
9537	057000	023420	\$DTBL:	10000.		
9538	057002	001750		1000.		
9539	057004	000144		100.		
9540	057006	000012		10.		
9541	057010	000004	\$DBLK:	.BLKW 4		

```

9542 ;*****
9543
9544 .SBTTL TYPE ROUTINE
9545
9546 ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
9547 ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
9548 ;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
9549 ;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
9550 ;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
9551 ;*
9552 ;*CALL:
9553 ;*1) USING A TRAP INSTRUCTION
9554 ;* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
9555 ;*OR
9556 ;* TYPE
9557 ;* MESADR
9558 ;*
9559 ;*2) USING A JSR INSTRUCTION
9560 ;* MOV PS,-(SP) ;;PUSH PROCESSOR STATUS WORD ON THE STACK
9561 ;* JSR PC,$TYPE ;;CALL TYPE ROUTINE
9562 ;* MESADDR ;;FIRST ADDRESS OF MESSAGE
9563
9564 057020 105767 122125 $TYPE: TSTB $STPFLG ;; IS THERE A TERMINAL?
9565 057024 100002 BPL 1$ BR IF YES
9566 057026 000000 HALT HALT HERE IF NO TERMINAL
9567 057030 000407 BR 3$ LEAVE
9568 057032 010046 1$: MOV RO,-(SP) SAVE RO
9569 057034 017600 000002 MOV 2(SP),RO GET ADDRESS OF ASCIZ STRING
9570 057040 112046 2$: MOVB (RO)+,-(SP) PUSH CHARACTER TO BE TYPED ONTO STACK
9571 057042 001005 BNE 4$ BR IF IT ISN'T THE TERMINATOR
9572 057044 005726 TST (SP)+ IF TERMINATOR POP IT OFF THE STACK
9573 057046 012600 MOV (SP)+,RO RESTORE RO
9574 057050 062716 000002 3$: ADD #2,(SP) ADJUST RETURN PC
9575 057054 000002 RTI RETURN
9576 057056 004767 000026 4$: JSR PC,$TYPEC GO TYPE THIS CHARACTER
9577 057062 126726 122062 5$: CMPB $FILLC,(SP)+ IS IT TIME FOR FILLER CHARS.?
9578 057066 001364 BNE 2$ IF NO GO GET NEXT CHAR.
9579 057070 016746 122052 MOV $NULL,-(SP) GET # OF FILLER CHARS. NEEDED
9580 AND THE NULL CHAR.
9581 057074 105366 000001 6$: DECB 1(SP) DOES A NULL NEED TO BE TYPED?
9582 057100 032770 BLT 5$ BR IF --GO POP THE NULL OFF OF STACK
9583 057102 004767 000002 JSR PC,$TYPEC GO TYPE A NULL
9584 057106 000772 BR 6$ LOOP
9585 057110 105777 122026 $TYPEC: TSTB 2$STPC ;;WAIT UNTIL PRINTER IS READY
9586 057114 100375 BPL $TYPEC
9587 057116 116677 000002 122020 MOVB 2(SP),2$TPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
9588 057124 000207 RTS PC
    
```

9589  
9590  
9591  
9592  
9593  
9594  
9595  
9596  
9597  
9598  
9599  
9600  
9601  
9602  
9603  
9604  
9605  
9606  
9607  
9608  
9609  
9610  
9611  
9612  
9613  
9614  
9615  
9616  
9617  
9618  
9619  
9620  
9621  
9622  
9623  
9624  
9625  
9626  
9627  
9628  
9629  
9630  
9631  
9632  
9633  
9634  
9635  
9636  
9637  
9638  
9639  
9640  
9641  
9642

057126 000000  
057130 000000  
057132 000000  
057134 000011  
057145  
057146  
  
005067 177754  
012767 057134 177750  
016767 177744 177744  
012737 057216 000060  
012737 000200 000062  
005777 121732  
012777 000100 121722  
000207  
  
117746 121716  
042716 177600  
021627 000003  
001006  
104400 057527  
004767 177702  
000157 164622  
022767 000011 177650 1S:  
001004  
104400 001210  
005726  
000002  
005267 177632 2S:  
112677 177630  
005267 177624  
026727 177620 057145  
001003  
012767 057134 177606  
000002 3S:

```
*****  
;SBTTL TTY INPUT ROUTINE  
STKCNT: .WORD 0 ;: NUMBER OF ITEMS IN QUEUE  
STKQIN: .WORD 0 ;: INPUT POINTER  
STKQOUT: .WORD 0 ;: OUTPUT POINTER  
STKQSRV: .BLKB 9. ;: TTY KEYBOARD QUEUE  
STKQEND=.  
.EVEN  
  
;*TK INITIALIZE ROUTINE  
;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE  
;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT  
;CALL  
;* JSR PC,STKINT  
;* RETURN  
;*  
$TKINT: CLR STKCNT ;: CLEAR COUNT OF ITEMS IN QUEUE  
MOV $STKQSRV,STKQIN ;: MOVE THE STARTING ADDRESS OF THE  
MOV STKQIN,STKQOUT ;: QUEUE INTO THE INPUT & OUTPUT POINTERS.  
MOV $STKSRV,$$TKVEC ;: INITIALIZE THE KEYBOARD VECTOR  
MOV $200,$$TKVEC+2 ;: "BR" LEVEL 4  
TST $STKB ;: CLEAR DONE FLAG  
MOV $BIT06,$$STKS ;: ENABLE INTERRUPT  
RTS PC ;: RETURN TO CALLER  
  
;*TK SERVICE ROUTINE  
;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT  
;*  
$TKSRV: MOV $STKB,-(SP) ;: PICKUP THE CHARACTER  
BIC $1C177,(SP) ;: STRIP THE JUNK  
CMP (SP),#3 ;: IS IT A CONTROL C?  
BNE 1S ;: BRANCH IF NO  
TYPE $CNTLC ;: TYPE A CONTROL-C (1C)  
JSR PC,STKINT ;: INIT THE KEYBOARD  
JMP OPSEL ;: CONTROL C RESTART  
CMP #9.,STKCNT ;: IS THE QUEUE FULL?  
BNE 2S ;: BRANCH IF NO  
TYPE $BELL ;: RING THE TTY BELL  
TST (SP)+ ;: CLEAN CHARACTER OFF OF STACK  
RTI ;: RETURN  
2S: INC STKCNT ;: COUNT THIS CHARACTER  
MOV (SP)+,$$STKQIN ;: AND PUT IT IN QUEUE  
INC STKQIN ;: UPDATE THE POINTER  
CMP STKQIN,$$STKQEND ;: GO OFF THE END?  
BNE 3S ;: BRANCH IF NO  
MOV $STKQSRV,STKQIN ;: RESET THE POINTER  
3S: RTI ;: RETURN  
;*****  
;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY  
;CALL  
;* R0CHR ;: INPUT A SINGLE CHARACTER FROM THE TTY
```

```

9643          ;*      RETURN HERE          ;; CHARACTER IS ON THE STACK
9644          ;
9645
9646 057324 011646          $RDCHR: MOV      (SP), -(SP)          ;; PUSH DOWN THE PC AND
9647 057326 016666 000004 000002      MOV      4(SP), 2(SP)          ;; THE PS
9648 057334 005066 000004          CLR      4(SP)          ;; GET READY FOR A CHARACTER
9649 057340 005037 177776          CLR      2#PS          ;; ALLOW INTERRUPTS
9650 057344 005767 177556          1$: TST     $TKCNT          ;; WAIT ON A CHARACTER
9651 057350 001775          BEQ      1$
9652 057352 005367 177550          DEC     $TKCNT          ;; DECREMENT THE COUNTER
9653 057356 117766 177550 000004      MOVB   2$TKQOUT, 4(SP)          ;; GET ONE CHARACTER
9654 057364 005267 177542          INC     $TKQOUT          ;; UPDATE THE POINTER
9655 057370 026727 177536 057145      CMP     $TKQOUT, # $TKQEND          ;; DID IT GO OFF OF THE END?
9656 057376 001003          BNE     2$          ;; BRANCH IF NO
9657 057400 012767 057134 177524      MOV     # $TKQSRT, $TKQOUT          ;; RESET THE POINTER
9658 057406 000002          2$: RTI          ;; RETURN
9659          ;*****
9660          ; THIS ROUTINE WILL INPUT A STRING FROM THE TTY
9661          ; CALL:
9662          ; *      RDLIN          ;; INPUT A STRING FROM THE TTY
9663          ; *      RETURN HERE          ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
9664          ; *          ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
9665
9666 057410 010346          $RDLIN: MOV     R3, -(SP)          ;; SAVE R3
9667 057412 012703 057516          1$: MOV     # $TTYIN, R3          ;; GET ADDRESS
9668 057416 022703 057527          2$: CMP     # $TTYIN+9., R3          ;; BUFFER FULL?
9669 057422 101405          BLOS    4$          ;; BR IF YES
9670 057424 104412          RDCHR          ;; GO READ ONE CHARACTER FROM THE TTY
9671 057426 112613          MOVB   (SP)+, (R3)          ;; GET CHARACTER
9672 057430 122713 000177          CMPB   #177, (R3)          ;; IS IT A RUBOUT
9673 057434 001003          BNE     3$          ;; SKIP IF NOT
9674 057436 104400 001214          4$: TYPE   $QUES          ;; TYPE A '?'
9675 057442 000763          BR     1$          ;; CLEAR THE BUFFER AND LOOP
9676 057444 111367 000044          3$: MOVB   (R3), 9$          ;; ECHO THE CHARACTER
9677 057450 104400 057514          TYPE   9$
9678 057454 122723 000015          CMPB   #15, (R3)+          ;; CHECK FOR RETURN
9679 057460 001356          BNE     2$          ;; LOOP IF NOT RETURN
9680 057462 105063 177777          CLRB   -1(R3)          ;; CLEAR RETURN (THE 15)
9681 057466 104400 001216          TYPE   $LF          ;; TYPE A LINE FEED
9682 057472 012603          MOV     (SP)+, R3          ;; RESTORE R3
9683 057474 011646          MOV     (SP), -(SP)          ;; ADJUST THE STACK AND PUT ADDRESS OF THE
9684 057476 016666 000004 000002      MOV     4(SP), 2(SP)          ;; FIRST ASCII CHARACTER ON IT
9685 057504 012766 057516 000004      MOV     # $TTYIN, 4(SP)
9686 057512 000002          RTI          ;; RETURN
9687 057514          9$: .BYTE 0          ;; STORAGE FOR ASCII CHAR. TO TYPE
9688 057515          .BYTE 0          ;; TERMINATOR
9689 057516 000011          $TTYIN: .BLKB 9.          ;; RESERVE 9. BYTES FOR TTY INPUT
9690 057527          136 006503 000012 $CNTLC: .ASCIZ /TC/<15><12>          ;; CONTROL "C"
9691          ; FROM THE TTY

```

9692  
9693  
9694  
9695  
9696  
9697  
9698  
9699  
9700  
9701  
9702  
9703  
9704  
9705  
9706  
9707  
9708  
9709  
9710  
9711  
9712  
9713  
9714  
9715  
9716  
9717  
9718  
9719  
9720  
9721  
9722  
9723  
9724  
9725  
9726  
9727  
9728  
9729  
9730  
9731  
9732  
9733  
9734  
9735  
9736  
9737  
9738  
9739  
9740  
9741  
9742  
9743  
9744  
9745

057534 011646  
057536 016666 000004 000002  
057544 010746  
057546 010146  
057550 010246  
057552 104414  
057554 012600  
057556 010067 000100  
057562 005001  
057564 005002  
057566 112046  
057570 001420  
057572 122716 000160  
057576 003026  
057600 122716 000067  
057604 002423  
057606 006301  
057610 006102  
057612 006301  
057614 006102  
057616 006301  
057620 006102  
057622 042716 177770  
057626 062601  
057630 000756  
057632 005726  
057634 010166 000012  
057640 010267 000026  
057644 012602  
057646 012601  
057650 012600  
057652 000002  
057654 005726  
057656 105010  
057660 104400  
057662 000000  
057664 104400 001214  
057670 000730  
057672 000000

```

;*****
.SBTTL READ AN OCTAL NUMBER FROM THE TTY

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

SRDOCT: MOV      (SP), -(SP)      ;; PROVIDE SPACE FOR THE
        MOV      4(SP), 2(SP)    ;; INPUT NUMBER
        MOV      R0, -(SP)      ;; PUSH R0 ON STACK
        MOV      R1, -(SP)      ;; PUSH R1 ON STACK
        MOV      R2, -(SP)      ;; PUSH R2 ON STACK
1$:     RDLIN                      ;; READ AN ASCII LINE
        MOV      (SP)+, R0      ;; GET ADDRESS OF 1ST CHARACTER
        MOV      R0, 5$        ;; AND SAVE IT
        CLR      R1            ;; CLEAR DATA WORD
        CLR      R2
2$:     MOVB     (R0)+, -(SP)    ;; PICKUP THIS CHARACTER
        BEQ      3$            ;; IF ZERO GET OUT
        CMPB    #'0, (SP)      ;; MAKE SURE THIS CHARACTER
        BGT     4$            ;; IS AN OCTAL DIGIT
        CMPB    #'7, (SP)
        BLT     4$
        ASL     R1              ;; *2
        ROL     R2
        ASL     R1              ;; *4
        ROL     R2
        ASL     R1              ;; *8
        ROL     R2
        BIC     #'C7, (SP)     ;; STRIP THE ASCII JUNK
        ADD     (SP)+, R1      ;; ADD IN THIS DIGIT
        BR      2$            ;; LOOP
3$:     TST     (SP)+          ;; CLEAN TERMINATOR FROM STACK
        MOV     R1, 12(SP)     ;; SAVE THE RESULT
        MOV     R2, SHIOCT
        MOV     (SP)+, R2      ;; POP STACK INTO R2
        MOV     (SP)+, R1      ;; POP STACK INTO R1
        MOV     (SP)+, R0      ;; POP STACK INTO R0
        RTI                      ;; RETURN
4$:     TST     (SP)+          ;; CLEAN PARTIAL FROM STACK
        CLRB   (R0)           ;; SET A TERMINATOR
        TYPE                    ;; TYPE UP THRU THE BAD CHAR.
5$:     .WORD   0
        TYPE   $QUES          ;; "?" "CR" & "LF"
        BR     1$            ;; TRY AGAIN
SHIOCT: .WORD   0            ;; HIGH ORDER BITS GO HERE

```

```

9746 ;*****
9747
9748 .SBTTL ERROR HANDLER ROUTINE
9749
9750 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
9751 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
9752 ;*AND GO TO $ERRTYP ON ERROR
9753 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
9754 ;*SW15=1 HALT ON ERROR
9755 ;*SW13=1 INHIBIT ERROR TYPEOUTS
9756 ;*SW10=1 BELL ON ERROR
9757 ;*SW09=1 LOOP ON ERROR
9758 ;*CALL
9759 ;* ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER
9760
9761 057674 $ERROR:
9762
9763 057674 012737 177777 015122 REGSA1: MOV #-1,$ERFLG ;SET ERROR FLAG
9764 057702
9765
9766
9767 057702 105267 121175 7$: INCB $ERFLG ;;SET THE ERROR FLAG
9768 057706 001775 BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
9769 057710 016737 121166 177570 MOV $STNM,$DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
9770 057716 032737 002000 177570 BIT #BIT10,$SWR ;;BELL ON ERROR?
9771 057724 001402 BEQ 1$ ;;NO - SKIP
9772 057726 104400 001210 TYPE $BELL ;;RING BELL
9773 057732 005267 121154 1$: INC $ERTTL ;;COUNT THE NUMBER OF ERRORS
9774 057736 011667 121154 MOV ($,$ERRPC) ;;GET ADDRESS OF ERROR INSTRUCTION
9775 057742 162767 000002 121146 SUB #2,$ERRPC
9776 057750 117767 121142 121136 MOVB $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
9777 057756 032737 020000 177570 BIT #BIT13,$SWR ;;SKIP TYPEOUT IF SET
9778 057764 001004 BNE 2$ ;;SKIP TYPEOUTS
9779 057766 004737 060036 JSR PC,$ERRTYP ;;GO TO USER ERROR ROUTINE
9780 057772 104400 001215 TYPE $CRLF
9781 057776 005737 177570 2$: TST $SWR ;;HALT ON ERROR
9782 060002 100001 BPL 3$ ;;SKIP IF CONTINUE
9783 060004 000000 HALT ;;HALT ON ERROR!
9784 060006 032737 001000 177570 3$: BIT #BIT09,$SWR ;;LOOP ON ERROR SWITCH SET?
9785 060014 001402 BEQ 4$ ;;BR IF NO
9786 060016 016716 121066 MOV $LPERR,$SP ;;FUDGE RETURN FOR LOOPING
9787 060022 005767 121160 4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
9788 060026 001402 BEQ 5$ ;;BR IF NONE
9789 060030 016716 121152 MOV $ESCAPE,$SP ;;FUDGE RETURN ADDRESS FOR ESCAPE
9790 060034
9791 060034 000002 5$: RTI ;;RETURN

```

```

9792
9793
9794
9795
9796
9797
9798
9799
9800 060036
9801 060036 104400 001215
9802 060042 010046
9803 060044 005000
9804 060046 153700 001114
9805 060052 001004
9806
9807 060054 016746 121036
9808
9809 060060 104402
9810 060062 000445
9811 060064 005300
9812 060066 006300
9813 060070 006300
9814 060072 006300
9815 060074 062700 001220
9816 060100 012067 000004
9817 060104 001404
9818 060106 104400
9819 060110 000000
9820 060112 104400 001215
9821 060116 012067 000004
9822 060122 001404
9823 060124 104400
9824 060126 000000
9825 060130 104400 001215
9826 060134 010146
9827 060136 012001
9828 060140 001415
9829 060142 012000
9830 060144 105720
9831 060146 061003
9832 060150 013146
9833 060152 104402
9834 060154 000402
9835 060156
9836 060156 013146
9837 060160 104410
9838 060162 005711
9839 060164 001403
9840 060166 104400 060206
9841 060172 000764
9842
9843 060174 012601
9844 060176 012600
9845 060200 104400 001215
    
```

\*\*\*\*\*

.SBTTL ERROR MESSAGE TIMEOUT ROUTINE

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

\$ERRTYP:

```

          TYPE      $CRLF      ;; "CARRIAGE RETURN" & "LINE FEED"
          MOV      R0,-(SP)    ;; SAVE R0
          CLR      R0          ;; PICKUP THE ITEM INDEX
          BISB     @($ITEMB,R0
          BNE      1$         ;; IF ITEM NUMBER IS ZERO, JUST
                                ;; TYPE THE PC OF THE ERROR
          MOV      $ERRPC,-(SP) ;; SAVE $ERRPC FOR TIMEOUT
                                ;; ERROR ADDRESS
          TYPDC     ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
          BR      10$         ;; GET OUT
          1$:      DEC      R0   ;; ADJUST THE INDEX SO THAT IT WILL
                                ;; WORK FOR THE ERROR TABLE
          ASL      R0
          ASL      R0
          ASL      R0
          ADD      @($ERRTB,R0 ;; FORM TABLE POINTER
          MOV      (R0)+,2$    ;; PICKUP "ERROR MESSAGE" POINTER
          BEQ      3$         ;; SKIP TIMEOUT IF NO POINTER
          TYPE     ;; TYPE THE "ERROR MESSAGE"
          .WORD   0           ;; "ERROR MESSAGE" POINTER GOES HERE
          2$:      $CRLF      ;; "CARRIAGE RETURN" & "LINE FEED"
          MOV      (R0)+,4$    ;; PICKUP "DATA HEADER" POINTER
          BEQ      5$         ;; SKIP TIMEOUT IF 0
          TYPE     ;; TYPE THE "DATA HEADER"
          .WORD   0           ;; "DATA HEADER" POINTER GOES HERE
          3$:      $CRLF      ;; "CARRIAGE RETURN" & "LINE FEED"
          MOV      R1,-(SP)    ;; SAVE R1
          MOV      (R0)+,R1    ;; PICKUP "DATA TABLE" POINTER
          BEQ      9$         ;; BR IF NO DATA TO BE TYPED
          MOV      (R0)+,R0    ;; PICKUP "DATA FORMAT" POINTER
          4$:      TSTB     (R0)+ ;; "OCTAL" OR "DECIMAL"
          BNE      7$         ;; BR IF DECIMAL
          MOV      @ (R1)+,-(SP) ;; SAVE @ (R1)+ FOR TIMEOUT
          TYPDC     ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
          BR      8$         ;;
          5$:      MOV      @ (R1)+,-(SP) ;; SAVE @ (R1)+ FOR TIMEOUT
          TYPDS     ;; GO TYPE--DECIMAL ASCII WITH SIGN
          6$:      TST      (R1)  ;; IS THERE ANOTHER NUMBER?
          BEQ      9$         ;; BR IF NO
          TYPE     11$        ;; TYPE TWO(2) SPACES
          BR      6$         ;; LOOP
          7$:      MOV      @ (R1)+,-(SP) ;; SAVE @ (R1)+ FOR TIMEOUT
          8$:      TYPDS     ;; 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
          9$:      MOV      (SP)+,R1 ;; RESTORE R1
          10$:     MOV      (SP)+,R0 ;; RESTORE R0
          TYPE     , $CRLF    ;; "CARRIAGE RETURN" & "LINE FEED"
    
```

G06

MAINDEC-11-DZRPT-D, RJPO4 DISKLESS RH11 TEST-PART 2  
DZRPTD.SUB ERROR MESSAGE TYPEOUT ROUTINE

MACY11 27(663) 7-OCT-75 17:33 PAGE 235

SEQ 0277

9846	060204	000207			RTS	PC	::RETURN
9847	060206	020040	000	115:	.ASCIZ	/ /	::TWO(2) SPACES
9848		060212			.EVEN		



9849  
9850  
9851  
9852  
9853  
9854  
9855  
9856  
9857  
9858  
9859  
9860  
9861  
9862  
9863  
9864  
9865  
9866  
9867  
9868  
9869  
9870  
9871  
9872  
9873  
9874  
9875  
9876  
9877  
9878  
9879  
9880  
9881  
9882  
9883  
9884  
9885  
9886  
9887  
9888  
9889  
9890  
9891  
9892  
9893  
9894  
9895  
9896  
9897  
9898  
9899  
9900  
9901  
9902

060212 017546 000000  
050216 116667 000001 000211  
060224 112667 000207  
060230 062716 000002  
060234 000406  
060236 112767 000001 000171  
060244 112767 000006 000165  
060252 112767 000005 000154  
060260 010346  
060262 010446  
060264 010546  
060266 116704 000145  
060272 005404  
060274 062704 000006  
060300 110467 000132  
060304 116704 000125  
060310 016605 000012  
060314 005003  
060316 006105 15:  
060320 000404  
060322 006105 25:  
060324 006105  
060326 006105  
060330 010503  
060332 006103 35:  
060334 105367 000076  
060340 100016  
060342 042703 177770

```
*****  
.SBTTL BINARY TO OCTAL (ASCII) AND TYPE  
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT  
*OCTAL (ASCII) NUMBER AND TYPE IT.  
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE  
*CALL:  
*   MOV     NUM,-(SP)      ;; NUMBER TO BE TYPED  
*   TYPOS   ;; CALL FOR TYPEOUT  
*   .BYTE  N              ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE  
*   .BYTE  M              ;; M=1 OR 0  
*                               ;; 1=TYPE LEADING ZEROS  
*                               ;; 0=SUPPRESS LEADING ZEROS  
*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST  
*$TYPOS OR $TYPOC  
*CALL:  
*   MOV     NUM,-(SP)      ;; NUMBER TO BE TYPED  
*   TYPON   ;; CALL FOR TYPEOUT  
*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER  
*CALL:  
*   MOV     NUM,-(SP)      ;; NUMBER TO BE TYPED  
*   TYPOC   ;; CALL FOR TYPEOUT  
$TYPOS: MOV     2(SP),-(SP)  ;; PICKUP THE MODE  
        MOV     1(SP),%OFILL ;; LOAD ZERO FILL SWITCH  
        MOV     (SP)+,%OMODE+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,%OMODE+1 ;; SET FOR SIX(6) DIGITS  
$TYPON: MOV     #5,%OCNT    ;; SET THE ITERATION COUNT  
        MOV     R3,-(SP)    ;; SAVE R3  
        MOV     R4,-(SP)    ;; SAVE R4  
        MOV     R5,-(SP)    ;; SAVE R5  
        MOV     %OMODE+1,R4 ;; GET THE NUMBER OF DIGITS TO TYPE  
        NEG     R4  
        ADD     #6,R4      ;; SUBTRACT IT FOR MAX. ALLOWED  
        MOV     R4,%OMODE  ;; 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  
15:    ROL     R5        ;; ROTATE MSB INTO "C"  
        BR     35        ;; GO DO MSB  
25:    ROL     R5        ;; FORM THIS DIGIT  
        ROL     R5  
        ROL     R5  
        MOV     R5,R3  
35:    ROL     R3        ;; GET LSB OF THIS DIGIT  
        DECB   %OMODE    ;; TYPE THIS DIGIT?  
        BPL    75        ;; BR IF NO  
        BIC   #177770,R3 ;; GET RID OF JUNK
```

9903	060346	001002		BNE	4\$	:: TEST FOR 0
9904	060350	005704		TST	R4	:: SUPPRESS THIS 0'
9905	060352	001403		BEQ	5\$	:: BR IF YES
9906	060354	005204	4\$:	INC	R4	:: DON'T SUPPRESS ANYMORE 0'S
9907	060356	052703	000060	BIS	#'0,R3	:: MAKE THIS DIGIT ASCII
9908	060362	052703	000040	5\$:	BIS	:: MAKE ASCII IF NOT ALREADY
9909	060366	110367	000040	MOVB	R3,8\$	:: SAVE FOR TYPING
9910	060372	104400	060432	TYPE	8\$	:: GO TYPE THIS DIGIT
9911	060376	105367	000032	7\$:	SOCNT	:: COUNT BY 1
9912	060402	003347		BGT	2\$	:: BR IF MORE TO DO
9913	060404	002402		BLT	6\$	:: BR IF DONE
9914	060406	005204		INC	R4	:: INSURE LAST DIGIT ISN'T A BLANK
9915	060410	000744		BR	2\$	:: GO DO THE LAST DIGIT
9916	060412	012605		6\$:	MOV	:: RESTORE R5
9917	060414	012604		MOV	(SP)+,R4	:: RESTORE R4
9918	060416	012603		MOV	(SP)+,R3	:: RESTORE R3
9919	060420	016666	000002 000004	MOV	2(SP),4(SP)	:: SET THE STACK FOR RETURNING
9920	060426	012616		MOV	(SP)+,(SP)	
9921	060430	000002		RTI		:: RETURN
9922	060432	000	8\$:	.BYTE	0	:: STORAGE FOR ASCII DIGIT
9923	060433	000		.BYTE	0	:: TERMINATOR FOR TYPE ROUTINE
9924	060434	000	SOCNT:	.BYTE	0	:: OCTAL DIGIT COUNTER
9925	060435	000	SOFILL:	.BYTE	0	:: ZERO FILL SWITCH
9926	060436	000000	SOMODE:	.WORD	0	:: NUMBER OF DIGITS TO TYPE

```

9927
9928
9929
9930
9931
9932
9933
9934
9935
9936 060440 010046
9937 060442 016600 000002
9938 060446 005740
9939 060450 111000
9940 060452 016000 060460
9941 060456 000200
9942
9943
9944
9945
9946
9947
9948
9949
9950
9951 060460
9952 060460 057020
9953 060462 060236
9954 060464 060212
9955 060466 060252
9956 060470 056574
9957 060472 057324
9958 060474 057410
9959 060476 057534
9960 060500 045134
9961 060502 045206
9962 060504 045524

```

\*\*\*\*\*

.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
        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+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        $TYPOS     ;; CALL=TYPOS     TRAP+4(104404) TYPE OCTAL NUMBER (NO LEADING ZEROS)
        $TYPON     ;; CALL=TYPON     TRAP+6(104406) TYPE OCTAL NUMBER (AS PER LAST CALL)
        $TYPDS     ;; CALL=TYPDS     TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
        $RDCHR     ;; CALL=RDCHR     TRAP+12(104412) TTY TYPEIN CHARACTER ROUTINE
        $ROLIN     ;; CALL=ROLIN     TRAP+14(104414) TTY TYPEIN STRING ROUTINE
        $RDOCT     ;; CALL=RDOCT     TRAP+16(104416) READ AN OCTAL NUMBER FROM TTY
        T.SCOPE   ;; CALL=SCOPE     TRAP+20(104420) MY LOCAL SCOPES
        CHECKT    ;; CALL=CHECKD    TRAP+22(104422) CHECK DVA, RDY, DPR, DRY
        WAIT.T    ;; CALL=WAT       TRAP+24(104424) WAIT LOOP

```

```

9963 ;*****
9964
9965 .SBTTL POWER DOWN AND UP ROUTINES
9966
9967 :POWER DOWN ROUTINE
9968 060506 012737 060634 000024 $PWRDN: MOV $SILLUP,2#PWRVEC ;;SET FOR FAST UP
9969 060514 012737 000340 000026 MOV #340,2#PWRVEC+2 ;;PRIO:7
9970 060522 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
9971 060524 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
9972 060526 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
9973 060530 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
9974 060532 010446 MOV R4,-(SP) ;;PUSH R4 ON STACK
9975 060534 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
9976 060536 010667 000076 MOV SP,$SAVR6 ;;SAVE SP
9977 060542 012737 060554 000024 MOV $PWRUP,2#PWRVEC ;;SET UP VECTOR
9978 060550 000000 HALT
9979 060552 000776 BR .-2 ;;HANG UP
9980
9981 :POWER UP ROUTINE
9982 060554 016706 000060 $PWRUP: MOV $SAVR6,SP ;;GET SP
9983 060560 005067 000054 CLR $SAVR6 ;;WAIT LOOP FOR THE TTY
9984 060564 005267 000050 1$: INC $SAVR6 ;;WAIT FOR THE INC
9985 060570 001375 BNE 1$ ;;OF WORD
9986 060572 012605 MOV (SP)+,R5 ;;POP STACK INTO R5
9987 060574 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
9988 060576 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
9989 060600 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
9990 060602 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
9991 060604 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
9992 060606 012737 060506 000024 MOV $PWRDN,2#PWRVEC ;;SET UP THE POWER DOWN VECTOR
9993 060614 012737 000340 000026 MOV #340,2#PWRVEC+2 ;;PRIO:7
9994 060622 104400 TYPE $POWER ;;REPORT THE POWER FAILURE
9995 060624 060642 $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
9996 060626 012716 MOV (PC)+,(SP) ;;RESTART AT BEGIN
9997 060630 017346 $PWRAD: .WORD BEGIN ;;RESTART ADDRESS
9998 060632 000002 RTI
9999 060634 000000 $SILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
10000 060636 000776 BR .-2 ;;BEFORE THE POWER DOWN WAS COMPLETE
10001 060640 000000 $SAVR6: 0 ;;PUT THE SP HERE
10002 060642 005015 047520 042527 $POWER: .ASCIZ <15><12>"POWER"
10003 060650 000122 .EVEN
10004
10005
10006 000001 .END

```

ACL	=	000040	2219#							
ACU	=	100000	2172#							
AOE	=	001000	2088#							
AS	=	015046	1670	1673	2300#	4136*	4148	4318*	4329	4458* 4468
ATA	=	100000	2076#	4133	4315	4455				
ATABLE	=	017326	2394#	2724						
ATTENT	=	015132	2335#	2722*	2724*	4136	4318	4458		
ATO	=	000001	2110#							
AT1	=	000002	2111#							
AT2	=	000004	2112#							
AT3	=	000010	2113#							
AT4	=	000020	2114#							
AT5	=	000040	2115#							
AT6	=	000100	2116#							
AT7	=	000200	2117#							
AIS	=	000400	2051#	3973						
A17	=	001000	2052#	3973						
BA	=	015024	1666	2289#	4129*	4309*	4449*			
BAI	=	000010	2024#	3975	7924					
BASECH	=	050576	2544	8317#						
BEGIN	=	017346	171	2403#	9997					
BEGIN2	=	017336	173	2401#						
BITST	=	044720	2568	7206#						
BIT0	=	000001	144#	7661	7696	7698				
BIT00	=	000001	134#	144						
BIT01	=	000002	133#	143						
BIT02	=	000004	132#	142						
BIT03	=	000010	131#	141						
BIT04	=	000020	130#	140						
BIT05	=	000040	129#	139						
BIT06	=	000100	128#	138	9614					
BIT07	=	000200	127#	137						
BIT08	=	000400	126#	136	9441					
BIT09	=	001000	125#	135	9449	9784				
BIT1	=	000002	143#	7675						
BIT10	=	002000	124#	9770						
BIT11	=	004000	123#	9456						
BIT12	=	010000	122#							
BIT13	=	020000	121#	7691	7693	9777				
BIT14	=	040000	120#	7668						
BIT15	=	100000	119#	7663	7670	7677	7686	7688		
BIT2	=	000004	142#							
BIT3	=	000010	141#							
BIT4	=	000020	140#							
BIT5	=	000040	139#							
BIT6	=	000100	138#							
BIT7	=	000200	137#							
BIT8	=	000400	136#							
BIT9	=	001000	135#							
BLT1	=	044746	7212#	7217						
BLT2	=	044770	7212	7213	7219#					
BLT3	=	045000	7220	7221#						
BPTVEC	=	000014	151#							
CA	=	015042	2298#							



DF21	012575	499	1743#															
DF22	012602	516	530	1745#														
DF24	012606	545	558	1747#														
DF26	012615	571	1750#															
DF27	012625	586	1753#															
DF3	012523	391	398	1724#														
DF30	012633	599	1755#															
DF31	012641	354	362	729	737	1758#												
DF32	012651	338	379	1761#														
DF33	012662	321	1764#															
DF34	012672	617	1767#															
DF35	012701	636	708	1770#														
DF36	012711	651	668	1773#														
DF37	012717	691	1775#															
DF40	012730	754	1778#															
DF42	014732	760	767	794	1996#													
DF44	014735	774	780	787	801	808	815	822	836	843	850	857	864	871				
		878	885	892	899	920	1997#											
DF5	= 000001	2061#																
DF54	014742	829	906	913	1999#													
DH1	005444	298	417	1242#														
DH11	005674	403	1270#															
DH14	006051	434	1290#															
DH15	006250	448	1312#															
DH16	006300	456	1316#															
DH17	006436	467	1333#															
DH2	005562	385	1256#															
DH20	006613	480	1353#															
DH21	006767	493	1373#															
DH22	007103	510	1387#															
DH24	007177	538	551	1398#														
DH26	007355	562	1418#															
DH27	007553	578	1440#															
DH30	007711	592	1457#															
DH31	010043	344	719	1473#														
DH32	010241	327	368	1495#														
DH33	010456	312	1519#															
DH34	010655	607	1541#															
DH35	011030	627	698	1560#														
DH36	011225	642	659	1582#														
DH37	011363	679	1599#															
DH40	011602	746	1625#															
DH42	014405	758	765	792	1954#													
DH44	014462	772	778	785	799	806	813	820	834	841	848	855	862	869				
		876	883	890	897	918	1962#											
DH54	014576	827	904	911	1976#													
DIG8	= 000004	2063#																
DISK	053520	1700	3941	3990	4635	4795	4841	5014	5063*	5194	5241*	5491	5537	5707				
		5756#	5883	5931*	6174	6214	6377	6562	6610*	6612*	6753	6801*	6802*	7730				
		7248	8286	8300*	8301*	8882	8942*	9367										
		66#	9470*	9769*														
DISPLA=	177570																	
DLT	= 100000	2036#																
DL64	= 000020	2065#																
DMD	= 000001	2098#	2804	2860	3916	4105	4281	4314	7931	7959	8227	8465	8609	8614				





EM2	002213	310	325	342	940*															
EM20	002472	479	972*																	
EM21	002513	492	975*																	
EM22	002535	503	979*																	
EM24	003136	535	1026*																	
EM25	003231	548	1036*																	
EM30	003271	591	1042*																	
EM31	003413	604	1057*																	
EM32	003536	621	1072*																	
EM33	004025	641	1104*																	
EM34	004077	654	1112*																	
EM35	004172	673	1122*																	
EM36	004471	696	1155*																	
EM37	004657	716	1176*																	
EM40	004745	743	1186*																	
EM42	012740	757	1783*																	
EM43	013023	764	1793*																	
EM44	013067	771	1801*																	
EM45	013124	777	1807*																	
EM46	013154	784	1812*																	
EM47	013203	791	1817*																	
EM50	013235	798	1823*																	
EM51	013273	805	1830*																	
EM52	013314	812	1834*																	
EM53	013354	819	1841*																	
EM54	013416	826	1848*																	
EM55	013455	833	1855*																	
EM56	013470	840	1858*																	
EM57	013506	847	1862*																	
EM6	002242	366	384	944*																
EM60	013535	854	861	1867*																
EM61	013624	1878*																		
EM62	013674	868	1886*																	
EM63	013751	875	1895*																	
EM64	014027	882	1904*																	
EM65	014063	889	1910*																	
EM66	014145	896	1920*																	
EM67	014217	903	1929*																	
EM70	014304	910	1939*																	
EM71	014356	917	1947*																	
ERCRC	054744	8985*	9035*	9204*																
ERFLGS	015122	2324*	3976*	3987	3998	4075*	4251*	4412*	4616*	4625	4739*	4750	4788*	4906*						
		4922	4965*	5089*	5105	5149*	5265*	5281	5336*	5434*	5445	5484*	5602*	5618						
		5660*	5780*	5796	5839*	5957*	5973	6027*	6118*	6129	6167*	6278*	6294	6335*						
		6458*	6474	6518*	6635*	6651	6708*	6826*	6842	6897*	7551	7798*	7809	8526						
		9072	9763*																	
ERMGP	054746	8986*	9045*	9214*																
ERHEAD	054742	8984*	9021*	9193*																
ERPOS	050266	8218*	8224*	8241																
ERR	= 040000	2075*	4133	4315	4455															
EPRVEC	= 000004	147*	2523*	2530*	2550*	9432	9433*	9435*	9438*											
ERSTAR	051430	8383*																		
ERUNIT	051426	8382*	8384																	
ERMORD	051722	1650	1652	1690	1693	1697	2843	3992*	3997*	4177	4343	4481	7544*	7550*						









PIE11 = 000040	7998#	8104												
PIE12 = 000020	7999#	8137												
PIE13 = 000010	8000#													
PIE14 = 000004	8001#													
PIE15 = 000002	8002#													
PIE16 = 000001	8003#													
PIE17 = 100000	8004#													
PIE18 = 040000	8005#													
PIE19 = 020000	8006#													
PIE2 = 040000	7989#	8095												
PIE20 = 010000	8007#													
PIE21 = 004000	8008#	8112												
PIE22 = 002000	8009#	8143												
PIE23 = 001000	8010#	8120												
PIE24 = 000400	8011#	8143												
PIE25 = 000200	8012#													
PIE26 = 000100	8013#													
PIE27 = 000040	8014#													
PIE28 = 000020	8015#													
PIE29 = 000010	8016#													
PIE3 = 020000	7990#	8137												
PIE30 = 000004	8017#													
PIE31 = 000002	8018#													
PIE32 = 000001	8019#	8076	8084											
PIE4 = 010000	7991#													
PIE5 = 004000	7992#													
PIE6 = 002000	7993#													
PIE7 = 001000	7994#													
PIE8 = 000400	7995#													
PIE9 = 000200	7996#													
PIP = 020000	2074#													
PIRQ = 177772	64#													
PIRQVE = 000240	158#													
PKACK = 015204	2374#	2806												
PLU = 020000	2171#													
POSITI = 047660	1703	1709	4692*	4860*	5034*	5213*	5387*	5556*	5727*	5902*	6071*	6232*	6397*	
	6581*	6772*	8036*	8177	8232	8240*	8241	8243	8283*					
	2218#													
PRE = 000020	2070#	7337												
PROG = 001000	81#													
PRO = 000000	82#													
PR1 = 000040	83#													
PR2 = 000100	84#													
PR3 = 000140	85#													
PR4 = 000200	86#													
PR5 = 000240	87#													
PR6 = 000300	88#													
PR7 = 000340	61#	62	2407*	2429*	3786*	3820*	3851*	6938*	6973*	7004*	7115*	9649*		
PS = 177776	2053#	3502	3724	3729										
PSEL = 002000	2215#													
PSU = 000001	62#													
PSW = 177776	3986	4085	4261	4422	4924	4966	5107	5150	5283	5303	5337	5620	5661	
PUTREG = 044652	5798	5838	5975	5993	6026	6296	6336	6476	6517	6653	6673	6707	6844	
	6864	6898	6945	6951	6983	7180#	7312	7369	7808	8174	8180	8249		



RHLA	015014	2278*												
RHMR	015000	2272*	2804*	2860*	3916*	4104	4280	4944*	4945*	5640*	5641*	6316*	6317*	7931*
		7959*	8226	8465*	8607	8874	8968*	9119	9277					
RMOF	014770	2268*	3966*	4051*	4228*	4389*	4613*	4740*	4894*	5077*	5255*	5435*	5590*	5770*
		5945*	6119*	6266*	6446*	6625*	6814*	7503*	7602*	7796*	7925*			
RHSN	015006	2275*	2736	2746	2748									
RHMC	014752	2255*	2810	2824	2845	2930	2954	2984	3011	3042	3070	3962*	4047*	4063
		4143	4179	4224*	4240	4324	4345	4385*	4401	4463	4463	4607*	4717*	4889*
		5072*	5250*	5412*	5585*	5765*	5940*	6096*	6261*	6441*	6620*	6809*	7184	7186
		7498*	7597*	7791*	7921*									
RKEY1	051712	8565*	8603*	8640										
RKEY2	051714	8566*	8604*	8642										
RMP	= 000004	2081*												
RNCTR1	054562	8959*	8971*	8972*										
RNO	056104	9354*	9359*											
RNMAT1	054626	8971*												
RPTRP1	043364	6936	6950*											
RPTRP2	043470	6971	6982*											
RPVEC	014746	2244*	2430*	3470	3471*	3480*	3816	3817*	3829*	3847	3848*	3867*	6935	6970
		8340	8347*	8362	9390*									
RPVECT	056220	2430	9384*	9390										
RSETR	051710	8564*	8602*	8638										
RSYNC	051704	4557	4573	4580	8562*	8628	8633	8665	8676	8689	8699	9175	9178	9220
		9224												
RUNCTR	051454	8456*	8468*	8469*										
RUNMAT	051520	8468*												
RO	=%000000	69*	2524*	2527*	2612*	2616*	2658*	2661*	2666*	2700*	2723*	2724	2809*	2823*
		2836*	2861*	3176*	3178*	3218*	3220*	3684*	3686*	3917*	3940*	3989*	3993	4062*
		4104*	4105*	4106*	4107*	4109*	4110*	4119*	4120*	4142*	4152*	4169*	4239*	4280*
		4281*	4282*	4283*	4285*	4286*	4295*	4296*	4323*	4335*	4400*	4440*	4441*	4462*
		4473*	4527*	4529*	4537*	4540*	4542*	4543*	4544*	4554*	4557*	4558*	4559*	4560*
		4561*	4566	4569*	4573*	4576*	4580*	4582*	4589*	4633*	4682*	4684*	4718*	4719
		4721*	4723*	4724*	4725*	4727*	4775*	4782*	4793*	4841*	4842*	4846	4849*	4943*
		4946*	4951*	4952*	4955*	4956*	4957*	4960*	4972*	5014*	5015*	5019	5022*	5121*
		5130*	5131*	5134*	5135*	5136*	5139*	5155*	5194*	5195*	5199	5202*	5297*	5316*
		5317*	5320*	5321*	5322*	5325*	5342*	5377*	5379*	5413*	5414	5416*	5418*	5419*
		5420*	5422*	5471*	5478*	5489*	5537*	5538*	5542	5545*	5639*	5642*	5647*	5648*
		5651*	5652*	5653*	5657*	5665*	5707*	5708*	5712	5715*	5812*	5821*	5822*	5825*
		5826*	5827*	5830*	5844*	5883*	5884*	5888	5891*	5989*	6009*	6010*	6013*	6014*
		6015*	6018*	6032*	6061*	6063*	6097*	6098	6100*	6102*	6103*	6104*	6106*	6154*
		6161*	6172*	6214*	6215*	6219	6222*	6315*	6318*	6323*	6324*	6327*	6328*	6329*
		6332*	6340*	6377*	6378*	6382	6385*	6490*	6499*	6500*	6503*	6504*	6505*	6508*
		6523*	6562*	6563*	6567	6570*	6667*	6686*	6687*	6690*	6691*	6692*	6695*	6713*
		6753*	6754*	6758	6761*	6858*	6878*	6879*	6882*	6883*	6884*	6887*	6902*	6935*
		6936*	6937*	6970*	6971*	6972*	7029*	7031	7069*	7071	7073	7076	7181	7184*
		7187	7192*	7258	7259	7260	7270*	7409	7410*	7411	7413	7414	7415	7416*
		7460	7461	7462	7469*	7536	7537	7538	7539	7540	7541*	7569*	7647	7648*
		7656	7708*	7725	7728*	7731	7754*	7776	7783	7784	7785	7786	7787	7788
		7789	7791	7793	7794	7820*	7846*	7860	7904*	7917	7936*	7953	7955	7975*
		8061	8078*	8085*	8088*	8089*	8090	8135*	8136*	8138	8140*	8141*	8142*	8144
		8206*	8222	8224	8267*	8277	8307*	8330*	8332*	8333*	8459	8550*	8607*	8609*
		8610*	8611*	8612*	8614*	8615*	8617*	8618*	8630	8693	8696	8728*	8731*	8733*
		8736	8745*	8750*	8753	8753	8762*	8794	8799*	8801*	8802	8811*	8816*	8826*
		8839*	8864	8874*	8910*	8962	9085*	9119*	9120*	9122*	9123*	9124	9131*	9133*



		9134	9141*	9271	9277*	9279*	9280*	9281*	9282*	9283*	9287*	9288*	9305*	9306*
		9308*	9309*	9310*	9311*	9325*	9326*	9327*	9328*	9335*	9336*	9346*	9488	9498*
		9502	9518	9519	9532*	9568	9569*	9570	9573*	9709	9713*	9714	9717	9737*
		9740*	9802	9803*	9804*	9811*	9812*	9813*	9814*	9815*	9816	9821	9827	9829*
R1	=%000001	9830	9844*	9936	9937*	9938	9939*	9940*	9941*	9970	9991*			
		70#	2525*	2526	2532	2585*	2586*	2587	2591	2609*	2618	2621	2667	3177*
		3181	3186	3196*	3197	3219*	3223	3227	3237*	3238	3685*	3688	3701*	3702
		3974*	3990*	3993	3996	4006	4058*	4103*	4113*	4118*	4121*	4235*	4279*	4289*
		4294*	4297*	4396*	4439*	4442*	4528*	4530*	4683*	4685*	4737*	4905*	4958*	4961*
		5088*	5137*	5140*	5264*	5323*	5326*	5378*	5380*	5432*	5601*	5654*	5658*	5779*
		5828*	5831*	5956*	6016*	6019*	6062*	6064*	6116*	6277*	6330*	6333*	6457*	6506*
		6509*	6634*	6693*	6696*	6825*	6885*	6888*	6939*	6974*	7030*	7031	7034	7182
		7185*	7187*	7191*	7255	7258*	7261	7263*	7269*	7290*	7297*	7319	7326	7330
		7373	7381	7457	7460*	7453	7468*	7511*	7531	7536*	7545	7548	7556	7568*
		7610*	7649	7653*	7661	7668	7675	7683*	7686*	7688*	7691*	7693*	7696*	7698*
		7703	7707*	7726	7729*	7732*	7734*	7738*	7753*	7784*	7918	7933*	7935*	8062
		8067*	8075	8083	8111	8119	8134*	8143*	8144*	8151*	8153	8169	8205*	8225
		8226*	8227*	8246	8254*	8255*	8262*	8263*	8266*	8278	8286*	8289	8306*	8331*
		8334*	8460	8475*	8478*	8479	8480	8481	8482*	8535*	8540*	8549*	8601	8602
		8603	8604	8605	8606	8702*	8703*	8861	8862	8863	8865	8869*	8872*	8894*
		8897*	8901*	8904*	8909*	8911*	8963	8978*	8981*	8982	8983	8984	8985	8986
		8987	8988*	9074*	9084*	9112	9113	9114	9115	9116	9117	9118*	9146*	9157*
		9174*	9175	9179	9186*	9187	9192	9198*	9199	9201	9209*	9213	9219*	9220
		9223	9227*	9228*	9270	9272	9322*	9333*	9345*	9347*	9359	9360	9361	9381*
		9382*	9489	9502*	9503	9507	9531*	9710	9715*	9723*	9725*	9727*	9730*	9733
		9736*	9826	9827*	9832	9836	9838	9843*	9971	9990*				
R11	025030	3181*												
R15	025654	3371	3375*											
R16	026024	3424	3428*											
R2	=%000002	71#	2526*	2610*	2611*	2615*	2893*	2894	2895	2930*	2931	2934*	2936*	2937
		2954*	2955	2958*	2960*	2962*	2964*	2966	2984*	2985	2988*	2990*	2991*	2993
		3011*	3012	3015*	3017*	3019*	3020*	3022*	3024	3042*	3043	3046*	3048*	3050*
		3052	3070*	3071	3074*	3076*	3078*	3079*	3081*	3083	3107*	3108*	3109	3118
		3173*	3174	3186*	3190	3216*	3217	3227*	3231	3260*	3265*	3267*	3269	3271*
		3272*	3273	3275	3299*	3300	3305*	3306	3359*	3360	3363*	3367*	3368	3406*
		3407	3410*	3414*	3416*	3418*	3420	3465*	3466	3473*	3476	3497*	3498	3502*
		3505	3525*	3526	3529*	3531	3554*	3555	3558*	3562	3585*	3586	3589*	3593*
		3595	3618*	3619	3622*	3626*	3628	3651*	3654*	3656*	3657	3659	3681*	3688*
		3690*	3691	3693	3720*	3721	3724*	3728	3749*	3750	3753*	3757	3782*	3783
		3788*	3790	3792	3811*	3812	3822*	3826	3843*	3844	3853*	3959	3884*	3889*
		3891*	3893*	3895*	3897*	3898	3900	3991*	3997	4011*	4108*	4111*	4284*	4287*
		4959*	4960	5138*	5139	5324*	5325	5655*	5657	5829*	5830	6017*	6018	6331*
		6332	6507*	6508	6694*	6695	6886*	6887	7183	7186*	7188*	7190*	7256	7259*
		7261*	7262*	7264*	7265*	7268*	7291*	7295*	7296*	7458	7461*	7463*	7467*	7532
		7537*	7545	7549	7556	7567*	7650	7670*	7672*	7673*	7689	7706*	7727	7730*
		7731*	7736*	7752*	8063	8068*	8094	8103	8133*	8137*	8138*	8150*	8152*	8154
		8204*	8279	8287*	8298*	8305*	8384*	8386	8461	8548*	8616*	8619*	8623*	8626*
		8651*	8656	8662*	8684*	8687*	8726	8732*	8767*	8771*	8795	8800*	8831*	8838*
		8862*	8866	8885*	8908*	8964	9083*	9132*	9142*	9154*	9159	9164*	9182*	9190
		9195*	9206*	9211	9216*	9273	9303*	9312*	9314*	9324*	9329*	9344*	9365*	9370*
		9375*	9378*	9490	9501*	9505*	9508	9515*	9516*	9517	9522*	9530*	9711	9716*
		9724*	9726*	9728*	9734	9735*	9972	9989*						
R3	=%000003	72#	2659*	2660*	2663*	2693*	2763	2779	7211*	7215*	7216	7219*	7221	7257
		7260*	7263	7267*	7292*	7343	7348	7351	7387	7392	7459	7462*	7464*	7466*











UN	021254	2567*	2569*											
UNIT	015110	2316*	2508*	2656*	2703*	2709*	2723	2729	7009	7029	7034*	7296		
UNITS	015070	2315*	2654*	2659	2663	2703	7030							
UNITSL	015120	2322*	2509*	2709										
UNLOAD	015150	2360*												
UNS	= 040000	2093*												
UPE	= 020000	2034*	3556	3563	3589	3596	3654							
US1	= 000001	2021*	3191	3232	3261	3265	3276	3622	3629	3885	3889	3901		
US2	= 000002	2022*	3191	3232	3261	3265	3276	3622	3629	3885	3889	3901		
US4	= 000004	2023*	3191	3232	3261	3265	3276	3622	3629	3885	3889	3901		
UMR	= 000010	2217*												
U11	025002	3145	3171*											
VUF	= 000002	2216*												
VU30	= 010000	2170*												
VV	= 000100	2067*	2818	7337										
WAIT.T	045524	7409*	9962											
MAT	= 104424	9962*												
MC	015022	1666	2288*	4064	4128*	4170	4241	4308*	4336	4402	4448*	4474	7185	
MCE	= 040000	2035*												
MCF	= 000040	2084*												
MCRC	053502	3956	4042	4219	4881	5055	5234	5577	5748	5923	6253	6418	6602	6793
		7492	7590	7751	8644	8938*								
MCU	= 000001	2158*												
MCYL	054724	4372*	4375	4595*	4602	4705*	4712	5400*	5407	6084*	6091	8990*		
MECC1	054520	1700	4757	5452	6135	8943*								
MECC2	054522	1700	4761	5456	6139	6427	6430*	6431*	8944*					
MKEY1	054730	4375*	4598*	4708*	5403*	6087*	8992*							
MKEY2	054732	4376*	4599*	4709*	5404*	6088*	8993*							
MLE	= 004000	2090*												
WORD	052512	8624*	8628*	8636*	8638*	8640*	8642*	8644*	8685*	8689*	8719*	8729*	8742*	8759*
		9368*	9376*											
MRCHDA	046210	7580*												
MRCHDT	015164	2366*	7511											
MRCHK	015162	2365*	7610											
MRCHHD	045576	7483*												
MRDATA	053170	8535	8860*	9074										
MRFROM	015212	2382*	2825	2831*	2838	4144	4148*	4171	4225	4309	4386	4449	4537	4563
		4564	4608	4634	4718	4951	4973	5130	5145*	5156	5316	5332*	5343	5413
		5547	5666	5821	5837*	5845	6009	6025*	6033	6097	6323	6341	6499	6524
		6686	6702*	6704*	6714	6878	6894*	6895*	6903	7821	7922			
MRHEAD	055020	8988	9112*											
MRIGAT	015166	2367*	3972	4235										
MRIFOR	015170	2368*	4396	4610	4737	5432	6116	7919						
WRITE	052750	8653	8664	8793*	8883	8889	8895	8899	8902	9156	9172	9185	9197	9208
		9218												
MRL	= 004000	2072*												
MRU	= 000400	2166*												
MSECTR	054726	4374*	4597*	4707*	5402*	6086*	8975	8991*						
MSSYNC	053470	8936*												
MSU	= 000004	2160*												
MTRK	054662	8977*	8980*											
MWORD	052746	8654	8658	8661	8665	8672	8677	8681	8788*	8835*	8884	8890	8896	8900
		8903	9157	9161	9174	9186	9198	9209	9219					
X	051612	3953*	4039*	4216*	4878*	5052*	5231*	5574*	5745*	5920*	6250*	6415*	6599*	6790*

		7489*	7586*	7794*	7811	8498*	8528							
XE2	022150	2623	2657*											
X11	025104	3151	3198*											
Y	051652	8538*												
Y11	024476	3150	3153*											
ZCOOE	047666	1709	4698*	4866*	5040*	5219*	5393*	5562*	5733*	5908*	6077*	6238*	6403*	6587*
ZER =	000400	6778*	8043*	8237*										
ZWORDS	053166	2105*	8246											
\$BDADR	001122	8859*	8877*											
		242*	1655	1658	1680	1713	7319*	7326*	7330*	7343*	7348*	7351*	7373*	7381*
\$BDDAT	001126	7387*	7392*											
		244*	1648	1652	1658	1663	1675	1677	1683	1686	1690	1693	1992	2587*
		2588	2937*	2939	2966*	2968	2993*	2995	3024*	3026	3052*	3054	3083*	3085
		3110*	3190*	3191*	3193	3231*	3232*	3234	3275*	3276*	3278	3306*	3308	3368*
		3370	3420*	3423	3467*	3505*	3506*	3508	3531*	3533*	3535	3562*	3563*	3565
		3595*	3596*	3598	3628*	3629*	3631	3659*	3660*	3662	3693*	3694*	3696	3728*
		3729*	3731	3757*	3758*	3760	3792*	3793*	3795	3813*	3859*	3860*	3862	3900*
		3901*	3903	3996*	7226*	7228*	7230	7428*	7549*	7864*	7865	7874*	7875	7886*
		7887	7894*	7896	7966*	7967	8658*	8677*	9151*	9161*	9179*	9192*	9201*	9213*
		9223*												
\$BELL	001210	270*	9629	9690	9772	9792								
\$CMTAG	001100	230*	2407	2408	2415	2421	2422							
\$CM1 =	000006	256*	257*	258*	259*	260*	261*	262*						
\$CM2 =	000014	256*	257*	258*	259*	260*	261*	262*						
\$CM3 =	000006	254*	256											
\$CM4 =	000006	262*	263*	264*	265*	266*	267*	268*						
\$CNTLC	057527	9624	9690*											
\$CRLF	001215	272*	2694	2731	2738	2745	7128	9690	9746	9780	9792	9801	9820	9825
		9845												
\$DBLK	057010	9499	9533	9541*										
\$DOAGN	044044	7061	7070	7080*										
\$DTBL	057000	9502	9537*											
\$ENDAD	044034	224	226	7071	7074	7076*								
\$ENDOCT	043764	7063*												
\$ENDMG	044050	7022	7065	7081*										
\$ENULL	044065	7025	7068	7084*										
\$EOP	043730	2547	2648	7028	7053*									
\$EOPCT	043756	7060*	7064											
\$ERFLG	001103	233*	9406	9445	9447	9453*	9474	9767*	9792					
\$ERMAX	001115	239*	2423*	9447	9469*	9474								
\$ERROR	057674	2415	2548	9761*										
\$ERRPC	001116	240*	1648	1650	1652	1655	1658	1661	1663	1666	1670	1673	1675	1677
		1680	1683	1686	1690	1693	1697	1700	1703	1706	1709	1713	1990	1992
		1994	9774*	9775*	9776	9792	9807							
\$ERRTB	001220	290*	9815											
\$ERRTY	060036	9779	9800*											
\$ERTTL	001112	237*	7015	7017*	9773*	9792								
\$ESCAP	001206	269*	2422*	9468*	9787	9789	9792							
\$FILLC	001150	252*	9577	9589										
\$FILLS	001147	251*	9589											
\$GADR	001120	241*												
\$GODAT	001124	243*	1648	1650	1652	1675	1683	1686	1690	1693	1697	1992	1994	2590*
		2932*	2939	2956*	2968	2986*	2995	3013*	3026	3044*	3054	3072*	3085	3111*
		3181*	3182*	3193	3223*	3234	3261*	3278	3303*	3305	3308	3365*	3367	3370



	3412*	3416	3421*	3423	3468*	3476*	3499*	3508	3527*	3535	3556*	3558	3565	
	3587*	3598	3620*	3631	3652*	3662	3682*	3696	3722*	3731	3751*	3760	3784*	
	3795	3814*	3826*	3845*	3862	3885*	3903	3995*	7221*	7223*	7225	7230	7548*	
	7869*	7880*	7890*	7899*	7969*	8633*	8657*	8674*	8676*	8699*	9150*	9160*	9178*	
	9191*	9202*	9212*	9224*										
	7069#													
\$GET42	044006													
\$HD	= 000000	11												
\$HIOCT	057672	9734*	9745#											
\$ICNT	001104	234#	9460*	9461	9463*	9473								
\$ILLUP	060634	9968	9999#											
\$ITEMB	001114	238#	9776*	9792	9804									
\$LF	001216	273#	9681	9690	9746	9792								
\$LPADR	001106	235#	2424*	2720*	7143*	7154	9451*	9466*	9471	9473				
\$LPERR	001110	236#	2425*	7126	7153*	9451	9467*	9473	9786					
\$MXCNT	056572	9464	9473#											
\$NULL	001146	250#	9579	9589										
\$NMST=	000001	2513#	2515	2555#	2557	2579#	2596#	2711#	2713	2752#	2754	2789#	2791	2854#
		2878#	2920#	2944#	2974#	3001#	3032#	3060#	3099#	3136#	3206#	3250#	3289#	3349#
		3396#	3455#	3489#	3516#	3544#	3575#	3608#	3641#	3671#	3710#	3739#	3772#	3803#
		3835#	3874#	3910#	3927#	3929	4014#	4016	4189#	4191	4355#	4357	4492#	4494
		4669#	4671	4819#	4821	4992#	4994	5172#	5174	5364#	5366	5515#	5517	5685#
		5687	5861#	5863	6048#	6050	6193#	6195	6356#	6358	6540#	6542	6730#	6732
		6921#	6923	6958#	6960	6992#	6994							
\$OCNT	060434	9882*	9911#	9924#										
\$OMODE	060436	9877*	9881#	9886	9889*	9900*	9926#							
\$OVER	056556	9428	9444	9452	9462	9470#								
\$PASS	001100	231#	3147	7021*	7023	7057*	7058*	7066	7081	9458	9474			
\$POWER	060642	9995	10002#											
\$PMRAD	060630	9997#												
\$PMRDN	060506	2419	9968#	9992										
\$PMRMG	060624	9995#												
\$PMRUP	060554	9977	9982#											
\$QUES	001214	271#	9674	9690	9743	9746	9792							
\$RDCR	057324	9646#	9957											
\$RDEC=	***** U	9960												
\$RDLIN	057410	9666#	9958											
\$RDOCT	057534	9707#	9959											
\$RDSZ =	000011	9659#												
\$REGAO	001152	254#												
\$REGO	001154	256#												
\$REG1	001156	257#												
\$REG2	001160	258#												
\$REG3	001162	259#												
\$REG4	001164	260#												
\$REG5	001166	261#												
\$RESET	044032	7072	7075#											
\$SAVRE=	***** U	9960												
\$SAVR6	060640	9976*	9982	9983*	9984*	10001#								
\$SCOPE	056306	2413	9415#											
\$SETUP=	000017	2407#	2413	2415	2417	2419	2421	2422	2424	7055				
\$S1 =	000000	2427#												
\$STUP =	177777	2407#												
\$SVLAD	056530	9436	9465#											
\$SMR =	167700	i#	11	44	45	46	47	48	49	50	268	259	270	2421

	2422	2424	2425	2518	2563	2583	2600	2719	2761	2799	2858	2882	2924	
	2948	2978	3005	3036	3064	3103	3140	3210	3254	3293	3353	3400	3459	
	3493	3520	3548	3579	3612	3645	3675	3714	3743	3776	3807	3839	3878	
	3914	3937	4028	4206	4367	4506	4679	4831	5004	5184	5374	5527	5697	
	5873	6058	6205	6369	6553	6743	6930	6965	7002	7050	7056	7069	7081	
	9407	9408	9409	9410	9411	9427	9439	9441	9442	9445	9446	9447	9454	
	9455	9456	9467	9470	9473	9753	9754	9755	9756	9757	9770	9777	9781	
	9784	9792												
\$SMRMK=	000000	50	51	9411	9412	9443								
\$TIMES	001204	268#	2421*	2518*	2563*	2583*	2600*	2719*	2882*	2924*	2948*	2978*	3005*	3036*
		3064*	3103*	3140*	3210*	3254*	3293*	3353*	3400*	3459*	3493*	3520*	3548*	3579*
		3612*	3645*	3675*	3714*	3743*	3776*	3807*	3839*	3878*	7002*	7056*	9454*	9461
		9464*	9473											
\$TKB	001140	247#	9593	9613	9620									
\$TKCNT	057126	9593#	9608#	9627	9632#	9650	9652#							
\$TKINT	057146	2431	2602	8328	9608#	9625								
\$TKQEN=	057145	9597#	9635	9655										
\$TKQIN	057130	9594#	9609#	9610	9633#	9634#	9635	9637*						
\$TKQOU	057132	9595#	9610#	9653	9654#	9655	9657*							
\$TKQSR	057134	9596#	9609	9637	9657									
\$TKS	001136	246#	2410	3963	3964	9593	9614#							
\$TKSRV	057216	9611	9620#											
\$TMP0	001170	262#	1663	2667*	2668*	2888*	2904	3113*	3122	3188*	3229*	3269*	3470*	3480
		3790#	3816*	3829	3847*	3867	4082*	4083	4258*	4259	4419*	4420	7413*	7419
		7424	7428	7539*	7553	7655*	7701*							
\$TMP1	001172	263#	1661	1663	1677	2532*	6940*	6941*	6975*	6976*	7414*	7419	7424	7540*
		7555												
\$TMP2	001174	264#	1677	7417*	7421*	7423*	7426*	7657*	7699*					
\$TMP3	001176	265#	1663	1677	7411*	7412*	7656*	7658	7659*					
\$TMP4	001200	266#	7658*	7682*										
\$TMP5	001202	267#	7654*	7660*	7666	7783*	7862	7884						
\$TN =	000073	1#	11	2501	2513	2518#	2520	2555	2563#	2566	2579	2583#	2584	2589
		2596	2600#	2708	2711	2719#	2752	2761#	2764	2789	2799#	2800	2854	2858#
		2878	2882#	2920	2924#	2925	2940	2944	2948#	2949	2969	2974	2978#	2979
		2996	3001	3005#	3006	3027	3032	3036#	3037	3055	3060	3064#	3065	3086
		3099	3103#	3104	3136	3140#	3141	3198	3206	3210#	3211	3239	3250	3254#
		3255	3279	3289	3293#	3294	3314	3349	3353#	3354	3377	3396	3400#	3401
		3430	3455	3459#	3460	3489	3493#	3494	3509	3516	3520#	3521	3536	3544
		3548#	3549	3566	3575	3579#	3580	3599	3608	3612#	3613	3632	3641	3645#
		3646	3663	3671	3675#	3676	3697	3699	3710	3714#	3715	3732	3739	3743#
		3744	3761	3772	3776#	3777	3796	3803	3807#	3808	3835	3839#	3840	3874
		3878#	3879	3904	3910	3914#	3927	3937#	3939	3988	4010	4014	4028#	4029
		4088	4189	4206#	4207	4265	4355	4367#	4368	4425	4492	4506#	4507	4626
		4660	4669	4679#	4681	4752	4799	4819	4831#	4833	4923	4978	4992	5004#
		5006	5106	5161	5172	5184#	5186	5282	5348	5364	5374#	5376	5447	5495
		5515	5527#	5529	5619	5671	5685	5697#	5699	5797	5850	5861	5873#	5875
		5974	6038	6048	6058#	6060	6131	6178	6193	6205#	6206	6295	6346	6356
		6369#	6370	6475	6529	6540	6553#	6554	6652	6719	6730	6743#	6745	6843
		6908	6921	6930#	6931	6948	6953	6958	6965#	6966	6980	6992	7002#	
\$TPB	001144	249#	9587*	9589										
\$TPFLG	001151	253#	9564	9589										
\$TPS	001142	248#	9585	9589										
\$TRAP	060440	2417	9936#											
\$TRP =	000026	9943#	9953#	9954#	9955#	9956#	9957#	9958#	9959#	9960#	9961#	9962#	9963#	



CHECKA	38#	4628	4766	4948	5127	5312	5461	5644	5818	6005	6144	6320	6496	6682	6874
CHECKB	38#	3968	4054	4231	4392	4618	4732	4900	5083	5260	5427	5596	5775	5951	6111
	6272	6452	6630	6820	7507	7606	7779								
COMMEN	13	159#													
ENDCOM	20	159#													
ERROR	59#	2533	2571	2592	2847	2902	2941	2970	2997	3028	3056	3087	3120	3202	3242
	3280	3310	3373	3426	3478	3510	3537	3567	3600	3633	3664	3698	3733	3762	3797
	3827	3865	3905	4000	4003	4087	4159	4180	4263	4346	4423	4484	4640	4642	4759
	4763	4802	4804	4927	4935	4981	4983	5110	5119	5164	5166	5286	5295	5307	5351
	5353	5454	5458	5498	5500	5623	5631	5674	5676	5801	5810	5853	5855	5978	5987
	5999	6041	6043	6137	6141	6181	6183	6299	6307	6349	6351	6479	6488	6532	6534
	6656	6665	6677	6722	6724	6847	6856	6868	6911	6913	6946	6954	6984	7320	7327
	7331	7344	7349	7352	7374	7382	7388	7393	7429	7827	7829	7853	7855	7870	7881
	7891	7901	7971	8175	8181	8250									
ESCAPE	159#														
HCOMPR	38#														
HCOMPM	38#														
MAKECL	38#														
MSG	2513#	2854	3910												
	4191	2515	2554#	2557	2711#	2713	2752#	2754	2788#	2791	3927#	3929	4013#	4016	4188#
	5514#	4354#	4357	4492#	4494	4668#	4671	4818#	4821	4991#	4994	5171#	5174	5363#	5366
	6732	5517	5684#	5687	5860#	5863	6047#	6050	6192#	6195	6355#	6358	6539#	6542	6729#
	6920#	6923	6957#	6960	6992#	6994									
MULT	159#														
NEWST	159#	2513	2555	2579	2596	2711	2752	2789	2854	2878	2920	2944	2974	3001	3032
	3060	3099	3136	3206	3250	3289	3349	3396	3455	3489	3516	3544	3575	3608	3641
	3671	3710	3739	3772	3803	3835	3874	3910	3927	4014	4189	4355	4492	4669	4819
	4992	5172	5364	5515	5685	5861	6048	6193	6356	6540	6730	6921	6958	6992	8771
POP	159#	7190	7267	7466	7563	7704	7752	7935	7973	8200	8265	8302	8544	8701	8771
	8836	8906	9079	9226	9340	9380	9528	9735	9986						
PUSH	159#	7180	7254	7456	7530	7646	7649	7724	7916	7951	8060	8225	8276	8459	8606
	8725	8793	8864	8962	9117	9271	9361	9487	9709	9970					
RFORGC	38#	4799	4978	5161	5348	5495	5671	5850	6038	6178	6346	6529	6719	6908	
SAVE	38#	9762													
SAVTST	38#	2520	2566	2584	2761	2800	2883	2925	2949	2979	3006	3037	3065	3104	3141
	3211	3255	3294	3354	3401	3460	3494	3521	3549	3580	3613	3646	3676	3715	3744
	3777	3808	3840	3879	3939	4029	4207	4368	4507	4681	4833	5006	5186	5376	5529
	5699	5875	6060	6206	6370	6554	6745	6931	6966						
SCOPE	60#	2517	2562	2582	2599	2718	2760	2798	2857	2881	2923	2947	2977	3004	3035
	3063	3102	3139	3209	3253	3292	3352	3399	3458	3492	3519	3547	3578	3611	3644
	3674	3713	3742	3775	3806	3838	3877	3913	3936	4027	4205	4366	4505	4678	4830
	5003	5183	5373	5526	5696	5872	6057	6204	6368	6552	6742	6929	6964	7001	7054
SETTRA	9943#	9953	9954	9955	9956	9957	9958	9959	9960	9961	9962				
SETUP	159#	2407													
SKIP	38#	159#	2501	2589	2708	2764	2940	2969	2996	3027	3055	3086	3198	3239	3279
	3314	3377	3430	3509	3536	3566	3599	3632	3663	3697	3699	3732	3761	3796	3904
	3988	4010	4626	4660	4752	4923	5106	5282	5447	5619	5797	5974	6131	6295	6475
	6652	6843	6948	6953	6980										
SLASH	159#														
SMORE	38#	9416													
SPACE	159#														
STARS	30	36	159#	218	274	2004	2006	2040	2042	2513	2516	2555	2561	2579	2581
	2596	2598	2711	2717	2752	2759	2789	2797	2854	2856	2878	2880	2920	2922	2944
	2946	2974	2976	3001	3003	3032	3034	3060	3062	3099	3101	3136	3138	3206	3208
	3250	3252	3289	3291	3349	3351	3396	3398	3455	3457	3489	3491	3516	3518	3544



.STRAP	1#	9927
.STYPD	1#	9474
.STYPE	1#	9542
.STYPO	1#	9849

ADD	2844	2897	3108	4178	4344	4482	7142	7210	7262	7357	7396	7666	7673	7680	9507
	9574	9730	9815	9978	9888										
ASL	9723	9725	9727	9812	9813	9814									
ASLB	9512														
ASR	8098	8107	8115	8123											
BCC	2670	3221	8291	8730	8743	8760	9513								
BCS	3179	3687													
BEQ	2501	2589	2622	2626	2673	2675	2708	2764	2940	2969	2996	3027	3055	3086	3199
	3239	3279	3309	3371	3424	3509	3536	3566	3599	3632	3663	3697	3732	3761	3796
	3863	3904	3994	4010	4084	4260	4421	4758	4762	4926	4934	5109	5118	5285	5294
	5306	5453	5457	5622	5630	5800	5809	5977	5986	5998	6136	6140	6298	6306	6478
	6487	6655	6664	6676	6846	6855	6867	6953	7019	7028	7032	7070	7231	7275	7315
	7339	7546	7560	7522	7669	7676	7861	7863	7867	7878	7885	7888	7898	7968	8070
	8091	8158	8165	8172	8178	8245	8387	8529	8649	8655	8666	8671	8673	8692	8694
	8735	8737	8747	8752	8754	8764	8803	8813	8818	8828	8880	8888	9125	9135	9148
	9158	9176	9188	9200	9210	9221	9290	9363	9442	9444	9446	9450	9459	9651	9718
	9768	9771	9785	9788	9817	9822	9828	9839	9905						
BGE	9462														
BGT	7061	9521	9720	9912											
BHI	9448														
BHIS	8242														
BIC	2507	3191	3232	3276	3506	3533	3563	3596	3629	3660	3694	3729	3758	3793	3860
	3901	4008	4107	4110	4120	4283	4286	4296	4311	4441	4451	4932	4945	5116	5292
	5628	5641	5807	5984	6304	6317	6429	6430	6485	6662	6853	7058	7223	7228	7337
	7558	7686	7691	7696	8084	8137	8143	8152	8170	8255	8263	8332	8473	8612	8615
	8618	8976	9281	9283	9309	9311	9326	9328	9336	9621	9729	9902			
BICB	3050	3079													
BIS	2815	2818	3182	3473	3822	3853	3973	3975	4106	4109	4119	4130	4131	4133	4282
	4285	4295	4310	4312	4313	4315	4440	4450	4452	4453	4455	4944	5640	6316	6431
	7688	7693	7698	7924	7932	7960	8076	8095	8104	8112	8120	8135	8136	8138	8141
	8142	8144	8254	8262	8333	8466	8610	8611	8617	8731	8733	8750	8816	8969	9122
	9133	9280	9282	9287	9305	9306	9308	9310	9325	9327	9335	9515	9516	9907	9908
BISB	2991	3020	9904												
BIT	2534	2603	2625	2763	2779	4083	4259	4420	7274	7317	7324	7341	7346	7370	7378
	7385	7390	7419	7424	7661	7668	7675	8162	8164	8246	8386	8630	8693	8696	8736
	8753	8802	8817	9124	9134	9441	9449	9456	9770	9777	9784				
BLOS	2899	9669													
BLT	9504	9520	9582	9722	9913										
BMI	7685	7690	7695	9428	9511										
BNE	2411	2433	2528	2535	2604	2617	2662	2701	2780	3145	3150	3194	3235	3313	3376
	3429	3703	3988	3999	4012	4112	4114	4122	4288	4290	4298	4443	4531	4556	4571
	4578	4584	4591	4626	4659	4686	4729	4752	4844	4851	4923	4947	4962	5017	5024
	5106	5141	5197	5204	5282	5327	5381	5424	5447	5540	5547	5619	5643	5659	5710
	5717	5797	5832	5886	5893	5974	6020	6065	6108	6131	6217	6224	6295	6319	6334
	6380	6387	6475	6510	6565	6572	6652	6697	6756	6763	6843	6889	6942	6977	7072
	7074	7189	7217	7266	7318	7325	7342	7347	7372	7380	7386	7391	7420	7422	7425
	7427	7465	7552	7562	7700	7702	7733	7739	7810	7812	8163	8233	8235	8247	8297
	8299	8335	8470	8527	8534	8620	8622	8627	8631	8647	8663	8688	8697	8768	8770
	8832	8834	8873	8886	8892	8898	8905	8973	9073	9143	9145	9165	9171	9196	9217
	9313	9316	9330	9332	9334	9338	9371	9373	9379	9457	9509	9571	9578	9623	9628
	9636	9656	9673	9679	9778	9805	9831	9903	9985						
BPL	9495	9525	9565	9586	9782	9901									
BR	2402	2434	2439	2443	2447	2451	2455	2459	2463	2467	2471	2475	2479	2483	2487
	2491	2495	2503	2529	2537	2541	2606	2632	2636	2640	2644	2651	2677	2683	2689

# MO8

	2692	2726	2733	2740	2767	2771	2775	2782	2900	3155	3159	3163	3167	3200	3241
	3314	3377	3430	3699	3857	4002	4660	4760	5455	6138	6948	6980	7006	7012	7033
	7117	7123	7130	7134	7138	7145	7149	7322	7329	7345	7350	7376	7384	7389	7554
	7664	7671	7678	7687	7692	7697	7838	7872	7883	7893	8079	8145	8166	8176	8236
	8238	8257	8293	8319	8325	8337	8343	8349	8353	8359	8365	8369	8388	8389	8539
	8613	8635	8660	8675	8680	8682	8695	8739	8756	8807	8822	9128	9138	9163	9181
	9194	9205	9215	9307	9386	9430	9436	9439	9452	9455	9506	9523	9567	9584	9675
	9731	9744	9810	9834	9841	9879	9894	9915	9979	10000					
CLC	7214	8087	8809	8824	9129	9139									
CLR	2403	2409	2421	2422	2590	2611	2654	2656	2664	2665	2722	2934	2958	2988	3015
	3046	3074	3110	3111	3177	3219	3271	3363	3410	3467	3468	3499	3587	3620	3685
	3722	3751	3784	3813	3814	3820	3845	3851	3895	3949	3950	3951	3965	3967	3976
	4035	4037	4038	4039	4049	4053	4212	4214	4215	4226	4230	4374	4375	4376	4387
	4391	4515	4518	4519	4520	4523	4529	4554	4560	4561	4569	4576	4589	4614	4616
	4692	4695	4696	4697	4698	4708	4709	4724	4725	4739	4741	4788	4849	4860	4863
	4864	4865	4866	4878	4898	4906	4965	5022	5034	5037	5038	5039	5040	5052	5081
	5089	5149	5202	5213	5216	5217	5218	5219	5231	5259	5265	5336	5379	5387	5390
	5391	5392	5393	5403	5404	5419	5420	5434	5436	5484	5545	5556	5559	5560	5561
	5562	5574	5594	5602	5660	5715	5727	5730	5731	5732	5733	5745	5774	5780	5839
	5891	5902	5905	5906	5907	5908	5920	5949	5957	6027	6063	6071	6074	6075	6076
	6077	6087	6088	6103	6104	6118	6120	6167	6222	6232	6235	6236	6237	6238	6250
	6270	6278	6335	6385	6397	6400	6401	6402	6403	6415	6450	6458	6518	6570	6581
	6584	6585	6586	6587	6599	6629	6635	6708	6761	6772	6775	6776	6777	6778	6790
	6818	326	6897	7017	7020	7055	7056	7115	7156	7157	7161	7165	7297	7486	7487
	7489	06	7583	7584	7586	7605	7653	7654	7665	7672	7679	7736	7798	7926	7957
	8140	8228	8283	8284	8285	8294	8624	8657	8674	8685	8740	8757	8808	8823	8893
	9150	9160	9212	9374	9416	9417	9421	9425	9454	9468	9498	9501	9608	9640	9649
	9715	9716	9803	9892	9983										
CLRB	2620	9453	9527	9680	9740										
CMP	2410	2531	2672	2674	2898	2939	2968	2995	3026	3054	3085	3147	3193	3197	3234
	3238	3278	3308	3370	3423	3508	3535	3565	3598	3631	3662	3696	3702	3731	3760
	3795	3862	3903	3993	4009	4757	4761	4846	4933	5019	5108	5117	5199	5284	5293
	5304	5452	5456	5542	5629	5712	5799	5808	5888	5976	5985	5996	6135	6139	6219
	6305	6382	6477	6486	6567	6654	6663	6674	6758	6845	6854	6865	6950	6952	6982
	7031	7071	7073	7230	7313	7338	7545	7556	7559	7862	7865	7875	7884	7887	7896
	7967	8171	8177	8241	8243	8665	9175	9187	9199	9220	9437	9461	9519	9622	9627
	9635	9655	9668												
CMPB	3144	9443	9447	9577	9672	9678	9719	9721							
COM	6428	7219	7222	7224	7227	7229	8077	8096	8105	8113	8121				
DEC	2527	2616	2661	2700	2705	2964	3022	3081	3418	4011	4111	4113	4121	4287	4289
	4297	4442	4530	4555	4570	4577	4583	4590	4649	4650	4654	4655	4658	4685	4728
	4843	4850	4946	4961	5016	5023	5140	5196	5203	5326	5380	5423	5539	5546	5642
	5658	5709	5716	5831	5885	5892	6019	6064	6107	6216	6223	6318	6333	6379	6386
	6509	6564	6571	6696	6755	6762	6888	6941	6976	7027	7059	7188	7264	7265	7421
	7426	7464	7561	7699	7701	7732	7738	8234	8296	8298	8334	8469	8619	8621	8626
	8662	8687	8767	8769	8831	8833	8872	8885	8891	8897	8904	8972	9142	9144	9164
	9195	9216	9312	9315	9329	9331	9333	9337	9370	9378	9652	9811			
DECB	9581	9900	9911												
EMT	59														
HALT	167	3971	4057	4234	4395	4621	4630	4735	4769	4903	4950	5086	5129	5263	5315
	5430	5464	5599	5646	5778	5820	5954	6008	6114	6147	6275	6322	6455	6498	6633
	6685	6823	6877	7510	7609	7782	8372	9391	9566	9783	9978	9999			
INC	2615	2697	2699	2960	3017	3076	3196	3237	3414	3701	4648	4651	4652	4653	4656
	7021	7057	7543	8237	8240	8748	8765	8814	8829	9460	9505	9632	9634	9654	9773





4694	4705	4707	4710	4717	4718	4719	4721	4723	4726	4727	4730	4737	4740	4780
4781	4831	4833	4839	4840	4841	4842	4847	4859	4861	4862	4871	4875	4876	4877
4889	4890	4893	4894	4905	4931	4943	4951	4952	4955	4956	4957	4958	4959	4960
5004	5006	5012	5013	5014	5015	5020	5033	5035	5036	5045	5049	5050	5051	5063
5066	5072	5073	5076	5077	5088	5115	5130	5131	5134	5135	5136	5137	5138	5139
5145	5184	5186	5192	5193	5194	5195	5200	5212	5214	5215	5224	5228	5229	5230
5241	5244	5250	5251	5254	5255	5264	5291	5316	5317	5320	5321	5322	5323	5324
5325	5332	5374	5376	5377	5378	5386	5388	5389	5400	5402	5405	5412	5413	5414
5416	5418	5421	5422	5425	5432	5435	5476	5477	5527	5529	5535	5536	5537	5538
5543	5555	5557	5558	5567	5571	5572	5573	5585	5586	5589	5590	5601	5627	5639
5647	5648	5651	5652	5653	5654	5655	5657	5697	5699	5705	5706	5707	5708	5713
5726	5728	5729	5738	5742	5743	5744	5756	5759	5765	5766	5769	5770	5779	5806
5821	5822	5825	5826	5827	5828	5829	5830	5837	5873	5875	5881	5882	5883	5884
5889	5901	5903	5904	5913	5917	5918	5919	5931	5934	5940	5941	5944	5945	5956
5983	6009	6010	6013	6014	6015	6016	6017	6018	6025	6058	6060	6061	6062	6070
6072	6073	6084	6086	6089	6096	6097	6098	6100	6102	6105	6106	6109	6116	6119
6159	6160	6205	6206	6212	6213	6214	6215	6220	6231	6233	6234	6243	6247	6248
6249	6261	6262	6265	6266	6277	6303	6315	6323	6324	6327	6328	6329	6330	6331
6332	6369	6370	6375	6376	6377	6378	6383	6396	6398	6399	6408	6412	6413	6414
6427	6435	6441	6442	6445	6446	6457	6484	6499	6500	6503	6504	6505	6506	6507
6508	6553	6554	6560	6561	6562	6563	6568	6580	6582	6583	6592	6596	6597	6598
6610	6612	6614	6620	6621	6624	6625	6634	6661	6686	6687	6690	6691	6692	6693
6694	6695	6702	6704	6743	6745	6751	6752	6753	6754	6759	6771	6773	6774	6783
6787	6788	6789	6801	6802	6803	6809	6810	6813	6814	6825	6852	6878	6879	6882
6883	6884	6885	6886	6887	6894	6895	6931	6932	6935	6936	6937	6938	6939	6940
6966	6967	6970	6971	6972	6973	6974	6975	7002	7004	7009	7015	7023	7029	7030
7034	7062	7066	7069	7120	7126	7143	7153	7154	7181	7182	7183	7184	7185	7186
7187	7190	7191	7192	7206	7207	7208	7209	7211	7220	7221	7225	7226	7255	7256
7257	7258	7259	7260	7263	7267	7268	7269	7276	7290	7291	7292	7293	7295	7296
7310	7319	7326	7330	7336	7343	7348	7351	7367	7373	7381	7387	7392	7409	7410
7411	7413	7414	7415	7416	7417	7423	7428	7457	7458	7459	7460	7461	7462	7463
7466	7467	7468	7483	7488	7498	7499	7502	7503	7511	7531	7532	7533	7534	7535
7536	7537	7538	7539	7540	7541	7542	7544	7548	7549	7557	7564	7565	7566	7567
7568	7580	7585	7597	7598	7601	7602	7610	7647	7648	7649	7650	7651	7652	7655
7656	7657	7658	7663	7670	7677	7703	7704	7705	7706	7707	7708	7725	7726	7727
7728	7729	7730	7731	7734	7743	7746	7747	7748	7752	7753	7754	7776	7783	7784
7785	7790	7791	7793	7794	7796	7864	7868	7869	7874	7879	7880	7886	7889	7890
7894	7899	7900	7917	7918	7919	7921	7922	7923	7925	7927	7931	7935	7936	7952
7953	7955	7956	7958	7959	7966	7969	7970	7974	8061	8062	8063	8064	8065	8066
8067	8068	8075	8078	8083	8085	8094	8097	8103	8106	8111	8114	8119	8122	8153
8154	8169	8201	8202	8203	8204	8205	8206	8222	8224	8225	8226	8227	8266	8277
8278	8279	8280	8281	8282	8286	8287	8288	8289	8292	8300	8301	8302	8303	8304
8305	8306	8307	8322	8330	8331	8340	8347	8356	8362	8384	8457	8459	8460	8461
8462	8463	8464	8465	8468	8472	8474	8478	8479	8480	8481	8545	8546	8547	8548
8549	8550	8601	8602	8603	8604	8605	8606	8607	8608	8609	8614	8616	8623	8628
8632	8633	8634	8636	8638	8640	8642	8644	8650	8651	8652	8658	8659	8661	8676
8677	8678	8679	8681	8684	8689	8698	8699	8700	8702	8726	8727	8728	8732	8738
8741	8744	8745	8755	8758	8761	8762	8771	8794	8795	8796	8797	8798	8799	8800
8801	8804	8811	8819	8826	8835	8836	8837	8838	8839	8861	8862	8863	8864	8865
8866	8867	8868	8869	8870	8871	8874	8875	8877	8878	8881	8882	8884	8890	8894
8896	8900	8901	8903	8906	8907	8908	8909	8910	8960	8962	8963	8964	8965	8966
8967	8968	8971	8975	8977	8981	8982	8983	8984	8985	8986	8987	9080	9081	9082
9083	9084	9085	9112	9113	9114	9115	9116	9117	9118	9119	9120	9121	9123	9131
9132	9141	9146	9149	9151	9152	9154	9155	9157	9161	9162	9174	9177	9178	9179

	9180	9182	9183	9184	9186	9191	9192	9193	9198	9201	9202	9203	9204	9206	9207
	9209	9213	9214	9219	9222	9223	9224	9225	9227	3270	9271	9272	9273	9274	9275
	9276	9277	9278	9279	9288	9303	9304	9314	9322	9323	9324	9341	9342	9343	9344
	9345	9346	9359	9360	9361	9364	9365	9367	9368	9375	9376	9381	9390	9432	9433
	9435	9438	9451	9463	9464	9466	9467	9470	9471	9488	9489	9490	9491	9492	9493
	9494	9499	9502	9522	9528	9529	9530	9531	9532	9534	9535	9568	9569	9573	9579
	9609	9610	9611	9612	9614	9637	9646	9647	9657	9666	4667	9682	9683	9684	9685
	9707	9708	9709	9710	9711	9713	9714	9733	9734	9735	9736	9737	9763	9769	9774
	9786	9789	9802	9807	9816	9821	9826	9827	9829	9832	9836	9843	9844	9875	9883
	9884	9885	9891	9898	9916	9917	9918	9919	9920	9936	9937	9940	9968	9969	9970
	9971	9972	9973	9974	9975	9976	9977	9982	9986	9987	9988	9989	9990	9991	9992
	9993	9996													
MOV8	2423	2618	2724	2831	2936	2962	3019	3078	3367	3416	4148	4329	4468	4873	4874
	4891	4892	4953	4954	5047	5048	5074	5075	5132	5133	5226	5227	5252	5253	5318
	5319	5563	5570	5587	5588	5649	5650	5740	5741	5767	5768	5823	5824	5915	5916
	5942	5943	6011	6012	6245	6246	6263	6264	6325	6326	6410	6411	6443	6444	6501
	6502	6594	6595	6622	6623	6688	6689	6785	6786	6811	6612	6880	6881	7484	7485
	7500	7501	7581	7582	7599	7600	7744	7745	7786	7788	9469	9497	9500	9514	9517
	9526	9570	9587	9620	9633	9653	9671	9676	9717	9776	9876	9877	9880	9881	9882
	9886	9889	9890	9909	9939										
NEG	9496	9887													
NOP	3474	3475	3559	3560	3590	3591	3623	3624	3823	3824	3854	3855	7077	7078	7079
	7961	7962	7963	7964	8478	8491	8499	8512	8981	8996	9007	9021	9035	9045	9058
RESET	2601	3503	7075												
ROL	7215	9427	9724	9726	9728	9893	9895	9896	9897	9899					
ROR	2668	3178	3220	3686	7659	7660	7682	7683	8088	8089	8133	8134	8150	8151	8290
	8729	8742	8759	8806	8810	8821	8825	9127	9130	9137	9140				
RTI	7166	7277	7430	9472	9536	9575	9631	9638	9658	9686	9738	9791	9921	9998	
RTS	2572	2850	4162	4183	4349	4487	4641	4643	4803	4812	4982	4986	5165	5169	5352
	5356	5499	5508	5675	5679	5854	5858	6042	6046	6182	6191	6350	6354	6533	6537
	6723	6727	6912	6916	7193	7218	7234	7270	7298	7355	7358	7394	7397	7469	7516
	7569	7614	7709	7755	7828	7836	7854	7859	7904	7937	7975	8207	8267	8308	8551
	8703	8772	8840	8911	9086	9228	9347	9382	9588	9615	9846	9941			
SEC	8805	8820	9126	9136											
SUB	3997	7261	7311	7368	7412	7550	7777	7954	8223	8458	8656	8876	8961	9159	9190
	9211	9503	9775												
SWAB	3421														
TRAP	9943	9953	9954	9955	9956	9957	9958	9959	9960	9961	9962				
TST	2432	2500	2707	2746	2747	2895	3118	3312	3375	3428	3987	3998	4006	4566	4625
	750	4845	4922	4925	5018	5105	5198	5281	5445	5541	5618	5621	5711	5796	5887
	5473	6129	6218	6294	6297	6381	6474	6566	6651	6757	6842	7018	7216	7551	7684
	7689	7694	7809	7811	7860	8069	8090	8155	8232	8385	8526	8528	8530	8646	8648
	8654	8667	8672	8691	8734	8746	8751	8763	8812	8827	8879	8887	9072	9147	9170
	9289	9362	9372	9434	9458	9508	9518	9572	9613	9630	9650	9732	9739	9781	9787
	9838	9904	9938												
TSTB	2588	2621	7787	7789	9445	9510	9524	9564	9585	9830					
.ASCII	271	272	979	984	991	999	1003	1008	1042	1057	1072	1081	1088	1122	1132
	1141	1155	1186	1204	1213	1215	1225	1242	1256	1270	1290	1316	1333	1353	1373
	1387	1398	1418	1440	1457	1473	1495	1519	1541	1560	1582	1599	1625	1954	1962
	1976														
.ASCIZ	270	273	931	940	944	950	955	958	965	968	972	975	1019	1026	1036
	1053	1062	1096	1104	1112	1145	1167	1176	1196	1234	1249	1263	1280	1301	1312
	1324	1343	1363	1380	1393	1408	1429	1449	1465	1484	1507	1530	1551	1571	1591
	1612	1635	1783	1793	1801	1807	1812	1817	1823	1830	1834	1841	1848	1855	1858

	1862	1867	1878	1886	1895	1904	1910	1920	1929	1939	1947	1958	1969	1983	2441
	2445	2449	2453	2457	2461	2465	2469	2473	2477	2481	2485	2489	2493	2497	2505
	2539	2543	2608	2634	2638	2642	2646	2653	2679	2685	2691	2728	2735	2742	2769
	2773	2777	2784	3157	3161	3165	3169	7008	7014	7081	7119	7125	7132	7136	7140
	7147	7151	8321	8327	8339	8345	8351	8355	8361	8367	8371	9388	9690	9847	10002
.BLKB	9596	9689													
.BLKM	2315	2382	2383	8935	8936	8937	8938	8939	8940	8942	8943	8944	8945	8946	9541
.BYTE	232	233	238	239	250	251	252	253	1720	1722	1724	1727	1730	1733	1734
	1736	1739	1743	1745	1747	1750	1753	1755	1758	1761	1764	1767	1770	1773	1775
	1778	1996	1997	1999	2394	7084	9687	9688	9922	9923	9924	9925			
.ENABL	1														
.END	10006														
.ENDC	6	17	23	31	37	47	49	50	51	59	145	159	172	193	204
	215	219	227	229	254	262	268	269	270	271	274	275	2005	2007	2041
	2043	2407	2412	2413	2415	2417	2419	2421	2422	2424	2426	2441	2445	2449	2453
	2457	2461	2465	2469	2473	2477	2481	2485	2489	2493	2497	2505	2514	2515	2516
	2517	2518	2519	2539	2543	2556	2557	2561	2562	2563	2564	2580	2581	2582	2583
	2584	2597	2598	2599	2600	2601	2608	2634	2638	2642	2646	2653	2679	2685	2691
	2712	2713	2717	2718	2719	2720	2721	2728	2735	2742	2753	2754	2759	2760	2761
	2769	2773	2777	2784	2790	2791	2797	2798	2799	2855	2856	2857	2858	2879	2880
	2881	2882	2883	2921	2922	2923	2924	2925	2945	2946	2947	2948	2949	2975	2976
	2977	2978	2979	3002	3003	3004	3005	3006	3033	3034	3035	3036	3037	3061	3062
	3063	3064	3065	3100	3101	3102	3103	3104	3137	3138	3139	3140	3141	3157	3161
	3165	3169	3207	3208	3209	3210	3211	3251	3252	3253	3254	3255	3290	3291	3292
	3293	3294	3350	3351	3352	3353	3354	3397	3398	3399	3400	3401	3456	3457	3458
	3459	3460	3490	3491	3492	3493	3494	3517	3518	3519	3520	3521	3545	3546	3547
	3548	3549	3576	3577	3578	3579	3580	3609	3610	3611	3612	3613	3642	3643	3644
	3645	3646	3672	3673	3674	3675	3676	3711	3712	3713	3714	3715	3740	3741	3742
	3743	3744	3773	3774	3775	3776	3777	3804	3805	3806	3807	3808	3836	3837	3838
	3839	3840	3875	3876	3877	3878	3879	3911	3912	3913	3914	3914	3929	3935	3936
	3937	4015	4016	4026	4027	4028	4190	4191	4204	4205	4206	4356	4357	4365	4366
	4367	4493	4494	4504	4505	4506	4670	4671	4677	4678	4679	4820	4821	4829	4830
	4831	4993	4994	5002	5003	5004	5173	5174	5182	5183	5184	5365	5366	5372	5373
	5374	5516	5517	5525	5526	5527	5686	5687	5695	5696	5697	5862	5863	5871	5872
	5873	6049	6050	6056	6057	6058	6194	6195	6203	6204	6205	6357	6358	6367	6368
	6369	6541	6542	6551	6552	6553	6731	6732	6741	6742	6743	6922	6923	6928	6929
	6930	6959	6960	6963	6964	6965	6992	6993	6994	7000	7001	7002	7003	7008	7014
	7045	7048	7049	7050	7052	7055	7061	7064	7065	7069	7081	7084	7085	7091	7110
	7119	7125	7132	7136	7140	7147	7151	7171	7179	7198	7201	7240	7253	7304	7308
	7362	7366	7401	7406	7447	7455	7477	7480	7521	7529	7573	7576	7619	7645	7713
	7723	7760	7775	7908	7915	7942	7950	7981	7984	8211	8216	8271	8274	8313	8316
	8321	8327	8339	8345	8351	8355	8361	8367	8371	8392	8393	8452	8455	8555	8579
	8586	8588	8713	8715	8780	8782	8848	8850	8921	8927	8928	8954	8956	9089	9091
	9238	9240	9351	9353	9384	9388	9393	9401	9407	9412	9427	9429	9440	9443	9444
	9445	9447	9449	9456	9460	9465	9470	9473	9474	9475	9543	9590	9627	9640	9659
	9660	9667	9669	9674	9689	9690	9691	9693	9702	9746	9747	9753	9767	9774	9780
	9781	9791	9792	9793	9811	9849	9850	9928	9937	9940	9942	9952	9953	9954	9955
	9956	9957	9958	9959	9960	9961	9962	9964	9976	9986	9996	9998	10005	9954	9955
.EQUIV	59	60	62	77	78	107	108	109	110	111	112	113	114	115	116
	135	136	137	138	139	140	141	142	143	144					
.EVEN	1646	1782	1988	2441	2445	2449	2453	2457	2461	2465	2469	2473	2477	2481	2485
	2489	2493	2497	2505	2539	2543	2608	2634	2638	2642	2646	2653	2679	2685	2691
	2728	2735	2742	2769	2773	2777	2784	3157	3161	3165	3169	7008	7014	7119	7125
	7132	7136	7140	7147	7151	8321	8327	8339	8345	8351	8355	8361	8367	8371	9388

.IF	9598	9848	10004	30	36	47	48	49	50	51	57	117	145	167	193
	2	14	20	224	228	254	262	268	269	270	274	2004	2006	2040	2042
	204	215	218	2415	2417	2419	2421	2422	2424	2440	2444	2448	2452	2456	2460
	2407	2412	2413	2476	2480	2484	2488	2492	2496	2504	2513	2515	2516	2518	2519
	2464	2468	2472	2557	2561	2563	2564	2579	2581	2583	2584	2596	2598	2600	2601
	2538	2542	2555	2641	2645	2652	2678	2684	2690	2711	2713	2717	2719	2720	2727
	2607	2633	2637	2754	2759	2761	2768	2772	2776	2783	2789	2791	2797	2799	2854
	2734	2741	2752	2830	2882	2883	2920	2922	2924	2925	2944	2946	2948	2949	2974
	2856	2858	2878	3001	3003	3005	3006	3032	3034	3036	3037	3060	3062	3064	3065
	2976	2978	2979	3104	3136	3138	3140	3141	3156	3160	3164	3168	3206	3208	3210
	3099	3101	3103	3254	3255	3289	3291	3293	3294	3349	3351	3353	3354	3396	3398
	3211	3250	3252	3457	3459	3460	3489	3491	3493	3494	3516	3518	3520	3521	3544
	3400	3401	3455	3575	3577	3579	3580	3608	3610	3612	3613	3641	3643	3645	3645
	3546	3548	3549	3676	3710	3712	3714	3715	3739	3741	3743	3744	3772	3774	3776
	3671	3673	3675	3807	3808	3835	3837	3839	3840	3874	3876	3878	3879	3910	3912
	3777	3803	3805	3935	3937	4014	4016	4026	4028	4189	4191	4204	4206	4355	4357
	3914	3927	3929	4494	4504	4506	4669	4671	4677	4679	4819	4821	4829	4831	4992
	4365	4367	4492	5172	5174	5182	5184	5364	5366	5372	5374	5515	5517	5525	5527
	4994	5002	5004	5871	5861	5863	5871	5873	6048	6050	6056	6058	6193	6195	6203
	5685	5687	5695	6367	6369	6540	6542	6551	6553	6730	6732	6741	6743	6921	6923
	6205	6356	6358	6960	6963	6965	6991	6992	6994	7000	7002	7003	7007	7013	7044
	6928	6930	6958	7051	7052	7054	7060	7063	7065	7069	7080	7081	7090	7109	7118
	7048	7049	7050	7139	7146	7150	7170	7178	7197	7200	7239	7252	7303	7307	7361
	7124	7131	7135	7446	7454	7476	7479	7520	7528	7572	7575	7618	7644	7712	7722
	7365	7400	7405	7914	7941	7949	7980	7983	8210	8215	8270	8273	8312	8315	8320
	7759	7774	7907	8350	8354	8360	8366	8370	8391	8392	8451	8454	8554	8578	8585
	8326	8338	8344	8779	8781	8847	8849	8920	8926	8927	8953	8955	9088	9090	9237
	8587	8712	8714	9383	9387	9392	9400	9406	9411	9416	9427	9439	9441	9442	9443
	9239	9350	9352	9456	9458	9467	9472	9473	9474	9542	9589	9593	9622	9639	9659
	9445	9446	9447	9689	9690	9692	9698	9714	9746	9752	9762	9770	9777	9779	9780
	9667	9668	9673	9810	9826	9849	9927	9936	9940	9942	9943	9953	9954	9955	9955
	9781	9784	9791	9959	9960	9961	9962	9963	9976	9986	9994	9996	10002		
	9956	9957	9958												
.IFF	31	37	47	49	50	51	59	219	228	254	275	2005	2007	2041	2043
	2412	2514	2515	2517	2518	2519	2556	2557	2562	2563	2564	2580	2581	2582	2583
	2584	2597	2598	2599	2600	2601	2712	2713	2718	2719	2720	2753	2754	2760	2761
	2790	2791	2798	2799	2855	2856	2857	2858	2879	2880	2881	2882	2883	2921	2922
	2923	2924	2925	2945	2946	2947	2948	2949	2975	2976	2977	2978	2979	3002	3003
	3004	3005	3006	3033	3034	3035	3036	3037	3061	3062	3063	3064	3065	3100	3101
	3102	3103	3104	3137	3138	3139	3140	3141	3207	3208	3209	3210	3211	3251	3252
	3253	3254	3255	3290	3291	3292	3293	3294	3350	3351	3352	3353	3354	3397	3398
	3399	3400	3401	3456	3457	3458	3459	3460	3490	3491	3492	3493	3494	3517	3518
	3519	3520	3521	3545	3546	3547	3548	3549	3576	3577	3578	3579	3580	3609	3610
	3611	3612	3613	3642	3643	3644	3645	3646	3672	3673	3674	3675	3676	3711	3712
	3713	3714	3715	3740	3741	3742	3743	3744	3773	3774	3775	3776	3777	3804	3805
	3806	3807	3808	3836	3837	3838	3839	3840	3875	3876	3877	3878	3879	3911	3912
	3913	3914	3928	3929	3936	3937	4015	4016	4027	4028	4190	4191	4205	4206	4356
	4357	4366	4367	4493	4494	4505	4506	4670	4671	4678	4679	4820	4821	4830	4831
	4993	4994	5003	5004	5173	5174	5183	5184	5365	5366	5373	5374	5516	5517	5526
	5527	5686	5687	5696	5697	5862	5863	5872	5873	6049	6050	6057	6058	6194	6195
	6204	6205	6357	6358	6368	6369	6541	6542	6552	6553	6731	6732	6742	6743	6922
	6923	6929	6930	6959	6960	6964	6965	6992	6993	6994	7001	7002	7003	7045	7051
	7055	7061	7064	7081	7091	7110	7171	7179	7198	7201	7240	7253	7304	7308	7352
	7366	7401	7406	7447	7455	7477	7480	7521	7529	7573	7576	7619	7645	7713	7723

	7760	7775	7908	7915	7942	7950	7981	7984	8211	8216	8271	8274	8313	8316	8392
	8393	8452	8455	8555	8579	8586	8588	8713	8715	8780	8782	8848	8850	8921	8927
	8928	8954	8956	9089	9091	9238	9240	9351	9353	9384	9393	9401	9440	9443	9444
	9447	9473	9475	9543	9590	9640	9659	9660	9668	9673	9689	9693	9747	9752	9770
.IFT	9791	9792	9793	9810	9826	9850	9928	9937	9964	9996					
	2441	2445	2449	2453	2457	2461	2465	2469	2473	2477	2481	2485	2489	2493	2497
	2505	2539	2543	2608	2634	2638	2642	2646	2653	2679	2685	2691	2728	2735	2742
	2769	2773	2777	2784	3157	3161	3165	3169	7008	7014	7119	7125	7132	7136	7140
	7147	7151	8321	8327	8339	8345	8351	8355	8361	8367	8371	9388	9455	9646	9719
	9739	9746	9780												
.IFTF	2441	2445	2449	2453	2457	2461	2465	2469	2473	2477	2481	2485	2489	2493	2497
	2505	2539	2543	2608	2634	2638	2642	2646	2653	2679	2685	2691	2728	2735	2742
	2769	2773	2777	2784	3157	3161	3165	3169	7008	7014	7119	7125	7132	7136	7140
	7147	7151	8321	8327	8339	8345	8351	8355	8361	8367	8371	9388	9453	9640	9715
	9723	9745	9779												
.IIF	1	6	11	44	45	46	47	50	51	52	53	167	274	2413	2415
	2421	2422	2424	2425	2737	2744	7049	7055	7056	7067	7081	7085	9407	9408	9409
	9410	9411	9412	9454	9455	9470	9473	9474	9589	9593	9598	9682	9690	9691	9746
	9753	9754	9755	9756	9757	9792	9808	9833	9837	9952	9953	9954	9955	9956	9957
	9958	9959	9960	9961	9962										
.IRP	2407	2513	2555	2579	2596	2711	2752	2789	2854	2878	2920	2944	2974	3001	3032
	3060	3099	3136	3206	3250	3289	3349	3396	3455	3489	3516	3544	3575	3608	3641
	3671	3710	3739	3772	3803	3835	3874	3910	3927	4014	4189	4355	4492	4669	4819
	4992	5172	5364	5515	5685	5861	6048	6193	6356	6540	6730	6921	6958	6992	7181
	7190	7255	7267	7457	7466	7531	7564	7647	7649	7704	7725	7752	7917	7935	7952
	7974	8061	8201	8225	8266	8277	8302	8459	8545	8606	8702	8726	8771	8794	8836
	8864	8906	8962	9080	9117	9227	9271	9341	9361	9381	9416	9488	9528	9709	9735
	9762	9970	9986												
.LIST	1	13	24	38	50	159	167	254	256	257	258	259	260	261	262
	263	264	265	266	267	268	2407	2427	2441	2445	2449	2453	2457	2461	2465
	2469	2473	2477	2481	2485	2489	2493	2497	2505	2513	2518	2520	2539	2543	2555
	2563	2566	2579	2583	2584	2596	2600	2608	2634	2638	2642	2646	2653	2679	2685
	2691	2711	2719	2728	2735	2742	2752	2761	2769	2773	2777	2784	2789	2799	2800
	2854	2858	2878	2882	2883	2920	2924	2925	2944	2948	2949	2974	2978	2979	3001
	3005	3006	3032	3036	3037	3060	3064	3065	3099	3103	3104	3136	3140	3141	3157
	3161	3165	3169	3206	3210	3211	3250	3254	3255	3289	3293	3294	3349	3353	3354
	3396	3400	3401	3455	3459	3460	3489	3493	3494	3516	3520	3521	3544	3548	3549
	3575	3579	3580	3608	3612	3613	3641	3645	3646	3671	3675	3676	3710	3714	3715
	3739	3743	3744	3772	3776	3777	3803	3807	3808	3835	3839	3840	3874	3878	3879
	3910	3914	3927	3937	3939	4014	4028	4029	4189	4206	4207	4355	4367	4368	4492
	4506	4507	4669	4679	4681	4815	4831	4833	4992	5004	5006	5172	5184	5186	5364
	5374	5376	5515	5527	5529	5685	5697	5699	5861	5873	5875	6048	6058	6060	6193
	6205	6206	6356	6369	6370	6540	6553	6554	6730	6743	6745	6921	6930	6931	6958
	6965	6966	6992	7002	7008	7014	7119	7125	7132	7136	7140	7147	7151	8321	8327
	8339	8345	8351	8355	8361	8367	8371	9388	9411	9659	9942	9943	9952	9953	9954
	9955	9956	9957	9958	9959	9960	9961	9362	9963						
.MACRO	38	51	218	2513	2554	2711	2752	2788	3927	4013	4188	4354	4492	4668	4818
	4991	5171	5363	5514	5684	5860	6047	6192	6355	6539	6729	6920	6957	6992	9943
.MCALL	1	159													
.NLIST	1	13	24	38	50	159	167	254	256	257	258	259	260	261	262
	263	264	265	266	267	268	2407	2427	2441	2445	2449	2453	2457	2461	2465
	2469	2473	2477	2481	2485	2489	2493	2497	2505	2513	2518	2520	2539	2543	2555
	2563	2566	2579	2583	2584	2596	2600	2608	2634	2638	2642	2646	2653	2679	2685
	2691	2711	2719	2728	2735	2742	2752	2761	2769	2773	2777	2784	2789	2799	2800

	2854	2858	2878	2882	2883	2920	2924	2925	2944	2948	2949	2974	2978	2979	3001
	3005	3006	3032	3036	3037	3060	3064	3065	3099	3103	3104	3136	3140	3141	3157
	3161	3165	3169	3206	3210	3211	3250	3254	3255	3289	3293	3294	3349	3353	3354
	3396	3400	3401	3455	3459	3460	3489	3493	3494	3516	3520	3521	3544	3548	3549
	3575	3579	3580	3608	3612	3613	3641	3645	3646	3671	3675	3676	3710	3714	3715
	3739	3743	3744	3772	3776	3777	3803	3807	3808	3835	3839	3840	3874	3878	3879
	3910	3914	3927	3937	3939	4014	4028	4029	4189	4206	4207	4355	4367	4368	4492
	4506	4507	4669	4679	4681	4819	4831	4833	4992	5004	5006	5172	5184	5186	5364
	5374	5376	5515	5527	5529	5685	5697	5699	5861	5873	5875	6048	6058	6060	6193
	6205	6206	6356	6369	6370	6540	6553	6554	6730	6743	6745	6921	6930	6931	6958
	6965	6966	6992	7002	7008	7014	7119	7125	7132	7136	7140	7147	7151	8321	8327
	8339	8345	8351	8355	8361	8367	8371	9388	9411	9659	9942	9943	9952	9953	9954
	9955	9956	9957	9958	9959	9960	9961	9962	9963						
.PAGE	54	160	180	218	274	7271	8445	9474	9542	9589	9692	9746	9792	9849	9927
	9963														
.REM	2870	2906	3091	3127	3246	3283	3450	3485	3513	3540	3571	3604	3637	3668	3707
	3735	3765	3800	3831	3870										
.REPT	14	20	167	256	262										
.SBTTL	38	40	55	161	168	181	220	276	2001	2237	2397	2398	2399	2513	2555
	2579	2596	2711	2752	2789	2854	2866	2867	2868	2978	2920	2944	2974	3001	3032
	3060	3099	3136	3206	3250	3289	3349	3396	3455	3489	3516	3544	3575	3608	3641
	3671	3710	3739	3772	3803	3835	3874	3907	3910	3927	4014	4189	4355	4492	4665
	4669	4819	4992	5172	5364	5515	5685	5861	6048	6193	6356	6540	6730	6917	6921
	6958	6992	7037	7038	7039	7046	7168	7195	7236	7271	7288	7301	7399	7444	7474
	7518	7570	7616	7710	7757	7905	7939	7978	8208	8268	8310	8390	9394	9395	9396
	9402	9476	9544	9591	9694	9748	9794	9851	9929	9944	9965				
.TITLE	1														
.WORD	167	231	234	235	236	237	240	241	242	243	244	245	254	256	257
	258	259	260	261	262	263	264	265	266	267	1648	1650	1652	1655	1658
	1661	1663	1666	1670	1673	1675	1677	1680	1683	1686	1690	1693	1697	1700	1703
	1706	1709	1713	1990	1992	1994	2316	2317	2319	2321	2322	2376	2570	3941	3942
	3943	4778	5298	5474	5990	6157	6668	6859	7060	7063	7934	3456	8477	8484	8485
	8486	8487	8488	8537	8538	8541	8542	8959	8980	9076	9077	9593	9594	9595	9742
	9745	9819	9824	9926	9995	9997									

ERRORS DETECTED: 0

\*DZRPTD, DZRPTD/SOL/CRF+DZRPTD.MAC, DZRPTC.ITM, FLTINS.ITM, DZRPTC.ERR, FLTINS.ERR, DZRPTC.DEF, DZRPTD.SET, FLTINS.P11, DZRPTC.TST, DZRPT  
RUN-TIME: 81 85 15 SECONDS  
CORE USED: 23K

