

# RP11E

RP11E DISKLESS LOGIC TEST  
MD-11-DZRPW-C

EP-DZRPW-C-DL-C  
COPYRIGHT 75-77  
FICHE 1 OF 1

OCT 1977  
**digital**  
MADE IN USA

Frame 1	Frame 2	Frame 3	Frame 4	Frame 5	Frame 6	Frame 7	Frame 8	Frame 9	Frame 10
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...



B01

EOF1DZQMCSEQ

00010000

770920

PDP10 411

,DHDR1DZRPWCSEQ

00010000

770920

CO1

MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST MACY11 30(1046) 22-JUL-77 15:10 PAGE 1  
DZRPWC.P11 22-JUL-77 14:54

SEQ 0001

.REM !

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZRPW-C-D  
PRODUCT NAME: RP11E DISKLESS LOGIC TEST  
DATE CREATED: MAY 21, 1976  
MAINTAINER: DIAGNOSTIC GROUP  
AUTHOR: C. HESS

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 DOCUMENT.

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

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

COPYRIGHT (C) 1975, 1977 BY DIGITAL EQUIPMENT CORPORATION.

CONTENTS

1. ABSTRACT
2. REQUIREMENTS
  - 2.1 EQUIPMENT
  - 2.2 STORAGE
  - 2.3 PRELIMINARY PROGRAMS
3. LOADING PROCEDURE
4. STARTING PROCEDURE
  - 4.1 STARTING ADDRESSES
  - 4.2 OPERATOR ACTION
  - 4.3 UNIBUS ADDRESSES
5. OPERATING PROCEDURE
  - 5.1 SOFTWARE SWITCH REGISTER
  - 5.2 OPERATIONAL SWITCH SETTINGS
  - 5.3 RP11E CONTROLLER SETUP
  - 5.4 TEST SELECTION
  - 5.4 CHANGING RP11E UNIBUS AND VECTOR ADDRESSES
6. ERRORS
7. MISCELLANEOUS
  - 7.1 EXECUTION TIME
  - 7.2 END OF TEST
  - 7.3 STACK POINTER
  - 7.4 SUBROUTINE CALLS
8. PROGRAM DESCRIPTION
9. PROGRAM LISTING

1. ABSTRACT

THE RP11E DISKLESS TEST EXERCISES THE RP11E IN THE MAINTENANCE MODE. THE PROGRAM VERIFIES THE LOGIC CONTAINED IN THE RP11E BY UTILIZING THE THREE MAINTENANCE REGISTERS WHICH SIMULATE THE SIGNALS PASSING BETWEEN THE RP11E AND AN RPO2, RPRO2, OR RPO3 DISK DRIVE.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 COMPUTER WITH 8K OF MEMORY (WITH OR WITHOUT HARDWARE SWITCH REGISTER); CONSOLE TELETYPE; RP11E DISK CONTROLLER.

2.2 STORAGE

THIS PROGRAM WILL LOAD AND RUN IN 8K.

2.3 PRELIMINARY PROGRAMS

NONE

3. LOADING PROCEDURE

THE PROGRAM MAY BE LOADED FROM EITHER PAPER TAPE OR AN 'XXDP' DEVICE. REFER TO EITHER THE STANDARD PROCEDURES FOR LOADING 'ABS' FORMAT PAPER TAPES OR TO THE 'XXDP' SYSTEM REFERENCE DOCUMENT.

4. STARTING PROCEDURE

4.1 \*\*\* BEFOR STARTING REFER TO SECTION 5. \*\*\*  
STARTING ADDRESSES

200      NORMAL STARTING ADDRESS  
204      SELECT RP11 ADDRESS

4.1.1 START FROM LOCATION 200

WHEN THE PROGRAM IS STARTED FROM LOCATION 200, THE PROGRAM WILL PERFORM ALL OF THE TESTS IN SEQUENCE.

4.1.2 START FROM LOCATION 204

WHEN THE PROGRAM IS STARTED FROM LOCATION 204, OPERATION IS THE SAME AS THE START FROM 200, EXCEPT THAT THE PROGRAM ASKS FOR A NEW RP11E ADDRESS, VECTOR ADDRESS, AND PRIORITY. THE OPERATOR MAY ENTER NEW ADDRESS VALUES OR RETAIN THE OLD VALUES. REFER TO SECTION 4.3.

## 4.2 OPERATOR ACTION

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 3.)
2. POWER DOWN OR CYCLE DOWN DRIVE 0 (SEE SECTION 5.2).
3. LOAD ADDRESS 200 OR 204.
4. SET SWITCHES (SEE SECTION 5.1) (SOFTWARE SWITCH REGISTER IS LOC. 176)
5. PRESS START.

## 4.3 UNIBUS ADDRESSES

THE PROGRAM ASSUMES THE FOLLOWING UNIBUS ADDRESSES. THESE ADDRESSES MAY BE CHANGED AT THE INDICATED LOCATION BEFORE STARTING THE PROGRAM.

MEMORY LOCATION	CONTENTS	FUNCTION
1136	177560	TTY KEYBOARD STATUS REG
1140	177562	TTY KEYBOARD BUFFER REG
1142	177564	TTY PRINTER STATUS REGISTER
1144	177566	TTY PRINTER BUFFER REG

RP11E ADDRESSES (UNIBUS AND VECTOR) ARE CHANGED IN RESPONSE TO THE PROGRAM'S TYPED REQUESTS IF THE PROGRAM IS STARTED FROM LOCATION 204 OR IF THERE IS NO RESPONSE WHEN LOCATION 176710 IS ADDRESSED.

## 5. OPERATING PROCEDURES

IF THE DIAGNOSTIC IS RUN ON A CPU WITHOUT A SWITCH REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED WHICH ALLOWS THE USER THE SAME SWITCH OPTIONS AS THE HARDWARE SWITCH REGISTER. IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES AND IT CONTAINS ALL ONES (177777) THEN THE SOFTWARE SWITCH REGISTER (LOC. 176) IS USED.

## CONTROL:

THIS PROGRAM ALSO SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER (LOC. 176) FROM THE TTY. THIS CAN BE ACCOMPLISHED BY DOING THE FOLLOWING:

- 1) TYPE CONTROL G (<↑G>); THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.
- 2) THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW= (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)
- 3) AFTER THE ''NEW=''' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE TTY:

- A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED AND ONLY 6 NUMBERS WILL BE ALLOWED)  
IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.
- B) IF A CONTROL U <↑U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 2.

### 5.1 OPERATIONAL SWITCH SETTINGS

WITH ALL SWITCHES SET TO ZERO, THE PROGRAM WILL TYPE ALL ERRORS AND CONTINUE TESTING.

THE SWITCH SETTINGS ARE:

SW<15>=1...HALT ON ERROR  
 SW<14>=1...LOOP ON TEST  
 SW<13>=1...INHIBIT ERROR TYPEOUTS  
 SW<11>=1...INHIBIT ITERATIONS  
 SW<10>=1...RING BELL ON ERROR  
 SW<09>=1...LOOP ON ERROR  
 SW<08>=1...LOOP ON THE TEST NUMBER CONTAINED IN SWITCHES <0:6>  
 SW<07>=1...IN FLOATING 1'S & 0'S TESTS AND SILO DATA TEST, GO TO THE NEXT TEST AFTER A DATA COMPARISON ERROR.

### 5.2 CONTROLLER SETUP

BEFORE THE PROGRAM IS STARTED, THE CONTROLLER MAINTENANCE SWITCH MUST BE SET TO 'ENABLE'. DRIVE 0 MUST BE EITHER CYCLED DOWN OR BE POWERED DOWN.

AS THE CONTROLLER USES THE 'SURP03' LINE COMING FROM THE DRIVE TO DETERMINE THE MAXIMUM CYLINDER ADDRESS FOR THE DRIVE, SPECIAL SETUP PROCEDURES ARE REQUIRED TO TEST THE CONTROLLER WHEN MIXED DRIVES (RPO2 OR RPO3 DRIVES) ARE CONNECTED. IF DRIVE 0 IS AN RPO2 OR AN RPO3 AND IT IS DESIRED TO TEST THE CONTROLLER USING THE RPO3 MAXIMUM CYLINDER ADDRESS, POWER DOWN DRIVE 0 AND CONNECT A JUMPER BETWEEN H13P2 AND GROUND. GROUNDING THIS PIN FORCES THE RPO3 SIGNAL ('SURP03') HIGH.

IF DRIVE IS AN RPO3 AND THE CONTROLLER IS TO BE TESTED USING THE RPO2 MAXIMUM CYLINDER ADDRESS, POWER DOWN DRIVE 0.

IF THE CONTROLLER IS TO BE TESTED USING THE CYLINDER ADDRESS LIMIT OF DRIVE 0, THE DRIVE MUST BE CYCLED DOWN.

### 5.3 TEST SELECTION

TO EXECUTE A SPECIFIC TEST, SET SW<08> TO 1 AND SET THE TEST NUMBER (REFER TO THE PROGRAM LISTING) IN SWITCHES <0:6>, START THE PROGRAM IN THE USUAL MANNER. THE PROGRAM WILL PERFORM ALL THE TESTS UP TO THE SPECIFIED TEST FOR ONE ITERATION AND WILL LOOP ON THE TEST SPECIFIED IN THE SWITCHES. (REFERE TO SECTION 5. FOR SOFTWARE SWITCH REGISTER)

#### 5.4 CHANGING RP11E UNIBUS AND VECTOR ADDRESSES

THE OPERATOR MAY CHANGE THE DEFAULT VALUES OF THE RP11E UNIBUS AND VECTOR ADDRESSES BY STARTING THE PROGRAM FROM LOCATION 204. THE PROGRAM WILL TYPE THE FOLLOWING MESSAGE:

RPADR = 176710/

THE OPERATOR MAY EITHER ENTER A NEW RP11E UNIBUS ADDRESS VALUE OR HE MAY RETAIN THE PRELOADED VALUE BY ENTERING A CARRIAGE RETURN.

THE PROGRAM WILL THEN TYPE:

RPVEC = 254/

THE VECTOR ADDRESS MAY BE CHANGED BY ENTERING A NEW VALUE OR THE PRELOADED VALUE MAY BE RETAINED BY ENTERING A CARRIAGE RETURN.

THE PROGRAM WILL THEN REQUEST A NEW RP11E PRIORITY WITH THE FOLLOWING MESSAGE:

RPPRIO = 5/

A NEW PRIORITY MAY BE ENTERED OR THE PRESENT PRIORITY MAY BE RETAINED.

THE PROGRAM WILL THEN VERIFY THAT THE LOADED RP11E UNIBUS ADDRESS RESPONDS WHEN THE PROGRAM ADDRESSES THE LOCATION. IF THERE IS NO RESPONSE FROM THE ADDRESS, THE PROGRAM WILL TYPE:

'RP11E DIDN'T RESPOND TO ADDRESSING'

THE PROGRAM WILL LOOP BACK TO THE RP11E ADDRESS ENTRY MESSAGE.

THE PROGRAM ALWAYS CHECKS FOR RESPONSE FROM THE RP11E BEFORE STARTING THE TESTS WHETHER OR NOT A START FROM LOCATION 204 WAS INITIATED.

#### 6. ERRORS

-----  
WHEN THE PROGRAM DETECTS AN ERROR, THE ERROR ROUTINE IS CALLED AND THE SW<13> IS NOT SET, THE ERROR MESSAGE PERTAINING TO THE ERROR WILL BE TYPED. EACH ERROR TYPEOUT WILL CONTAIN THE FOLLOWING\*

- A. AN ERROR MESSAGE
- B. A DATA HEADER LINE
- C. A DATA LINE CONTAINING:
  1. THE TEST NUMBER
  2. THE PC (PROGRAM COUNTER CONTENTS) WHERE THE ERROR CALL WAS MADE.
  3. CONTENTS OF THE APPROPRIATE REGISTERS
- D. THE ERROR ROUTINE WILL HONOR A ↑G (REFERENCE OPERATING PROCEDURE SECTION 5.)



I01

MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST MACY11 30(1046) 22-JUL-77 15:10 PAGE 7  
DZRPWC.P11 22-JUL-77 14:54

SEQ 0007

7. MISCELLANEOUS  
-----

7.1 EXECUTION TIME

PASS 1 OF THE PROGRAM TAKES ABOUT 30 SECONDS. PASS 2 AND SUBSEQUENT PASSES TAKE ABOUT 3 MINUTES.

7.2 END OF TEST

ONE TEST ITERATION IS DEFINED AS 8 PASSES OF THE PROGRAM. AT END OF TEST, THE PROGRAM WILL TYPEOUT 'END OF TEST' AND CONTINUE OPERATION.

7.3 STACK POINTER

THE STACK POINTER IS INITIALLY SET TO 1100 AND EXTENDS DOWNWARD IN MEMORY.

7.4 SUBROUTINE CALLS

THE SUBROUTINE CALLS USED BY THE PROGRAM ARE:

- A. 'SCOPE' (IOT) INSTRUCTION). THIS CALL IS PLACED BETWEEN EACH TEST IN THE PROGRAM. THIS ROUTINE ESTABLISHES THE TEST ITERATION COUNT AND LOOP ON TEST AND LOOP ON ERROR ADDRESSES. ↑G IS HONORED. (REFER TO SECTION 5. FOR OPERATOR OPTIONS)
- B. 'ERROR' (EMTINSTRUCTION). THIS CALL IS USED TO REPORT ALL ERRORS. A CALL TO THIS ROUTINE IS FOLLOWED BY A NUMBER WHICH IDENTIFIES THE ERROR MESSAGE WHICH WILL BE TYPED. ↑G IS HONORED. (REFER TO SECTION 5. FOR OPERATOR OPTIONS)

THE TRAP INSTRUCTION IS USED FOR THE FOLLOWING SUBROUTINE CALLS:

TYPE - TTY TYPEOUT ROUTINE  
TYPOC - TYPE OCTAL NUMBER (WITH LEADING ZEROS)  
TYPOS - TYPE OCTAL NUMBER (NO LEADING ZEROS)  
RDCHR - READ A CHARACTER FROM THE TTY KEYBOARD  
RDLIN - READ A LINE FROM THE TTY KEYBOARD  
RDOCT - READ AN OCTAL NUMBER FROM THE TTY KEYBOARD  
SAVREG - ROUTINE TO SAVE R0 - R5  
RESREG - ROUTINE TO RESTORE R0 - R5

8. TEST DESCRIPTION  
-----

- 8.1 BEFORE TESTING IS STARTED, THE PROGRAM VERIFIES THAT THE ADDRESS SPECIFIED AS THE FIRST RP11E ADDRESS RESPONDS WHEN ADDRESSED. THE PROGRAM WILL NOT CONTINUE UNTIL IT RECEIVES A RESPONSE FROM THE RP11E.
- 8.2 TEST 0 - THIS TEST VERIFIES THAT ALL LOADABLE BITS IN REGISTERS RPER, RPCS, RPWC, RPBA, RPCA, AND RPDA CAN BE SET AND THAT A 'RESET' INSTRUCTION FROM THE PROCESSOR CLEARS THE REGISTERS.
- 8.3 TEST 1 - THIS TEST VERIFIES THAT THE SELECTED UNIT ON LINE ('SUOL') BIT IN RPDS CAN BE SET IN MAINTENANCE MODE.
- 8.4 TEST 2 - THIS TEST VERIFIES THAT THE RP11E INITIALIZE INSTRUCTION CLEARS THE REGISTERS. THE REGISTERS CHECKED ARE RPER, RPCS, RPWC, RPBA, RPCA, AND RPDA.
- 8.5 TEST 3 - THIS TEST TESTS THE LOADING OF ALL POSSIBLE BITS IN REGISTER RPER USING A FLOATING 1'S AND 0'S PATTERN.
- 8.6 TEST 4 - THIS TEST TESTS THE LOADING OF ALL POSSIBLE BITS IN REGISTER RPCS USING A FLOATING 1'S AND 0'S PATTERN.
- 8.7 TEST 5 - THIS TEST TESTS THE LOADING OF ALL POSSIBLE BITS IN REGISTER RPWC USING A FLOATING 1'S AND 0'S PATTERN.
- 8.8 TEST 6 - THIS TEST TESTS THE LOADING OF ALL POSSIBLE BITS IN REGISTER RPBA USING A FLOATING 1'S AND 0'S PATTERN.
- 8.9 TEST 7 - THIS TEST TESTS THE LOADING OF ALL POSSIBLE BITS IN REGISTER RPCA USING A FLOATING 1'S AND 0'S PATTERN.
- 8.10 TEST 10 - THIS TEST TESTS THE LOADING OF ALL POSSIBLE BITS IN REGISTER RPDA USING A FLOATING 1'S AND 0'S PATTERN.
- 8.11 TEST 11 - THIS TEST VERIFIES THE HIGH SPEED OPERATION OF THE WORD COUNT REGISTER IN A MANNER SIMILAR TO ITS OPERATION AS A COUNTER DURING A DATA TRANSFER OPERATION. THE CONTENTS OF EACH MEMORY LOCATION (UP TO 28K) ARE LOADED INTO AND READ FROM THE REGISTER AND CHECKED. THE LOAD AND READ OF THE REGISTER ARE PERFORMED AT THE MAXIMUM POSSIBLE PROGRAM RATE. FAILURES IN THIS TEST CAN INDICATE SETTLE DOWN PROBLEMS IN THE REGISTER
- 8.12 TEST 12 - THIS TEST VERIFIES THE HIGH SPEED OPERATION OF THE BUFFER ADDRESS REGISTER IN A MANNER SIMILAR TO ITS OPERATION AS A COUNTER DURING A DATA TRANSFER OPERATION. THE CONTENTS OF EACH MEMORY LOCATION (UP TO 28K) ARE LOADED INTO AND READ FROM THE REGISTER AND CHECKED. THE LOAD AND READ OF THE REGISTER ARE PERFORMED AT THE MAXIMUM POSSIBLE PROGRAM RATE. FAILURES IN THIS TEST CAN INDICATE SETTLE DOWN PROBLEMS IN THE REGISTER
- 8.13 TEST 13 - TEST THAT THE 'SURDY' BIT CAN BE SET AND CLEARED. THE BIT IS TESTED BY LOADING THE DRIVE READY BIT IN THE MAINTENANCE REGISTER

L01

MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST MACY11 30(1046) 22-JUL-77 15:10 PAGE 10  
DZRPWC.P11 22-JUL-77 14:54

SEQ 0010

8.14 TEST 14 - TEST THAT 'SELECTED UNIT SEEK INCOMPLETE' CAN BE SET IN  
MAINTENANCE MODE, THAT 'DRIVE ERROR' SETS IN RPER, AND THAT  
'ERR' AND 'HE' SET IN RPCS.

- 8.15 TEST 15 - TEST THAT 'SELECTED UNIT FILE UNSAFE' CAN BE SET AND CLEARED IN MAINTENANCE MODE.
- 8.16 TEST 16 - TEST THAT 'SELECTED UNIT WRITE PROTECT' CAN BE SET AND CLEARED IN MAINTENANCE MODE.
- 8.17 TEST 17 - TEST THAT 'SELECTED UNIT WRITE PROTECTED' CAN BE SET AND CLEARED IN MAINTENANCE MODE.
- 8.18 TEST 20 - TEST THAT EACH OF THE ATTENTION BITS CAN BE SET AND EACH IS CLEARED WHEN A '1' IS WRITTEN INTO THE BIT. VERIFY THAT A 'RESET' INSTRUCTION CLEARS THE ATTENTION BITS AND THAT WRITING 0'S INTO THE ATTENTION BITS DOES NOT CLEAR THE BITS.
- 8.19 TEST 21 - VERIFY THAT 'MPV' OCCURS WHEN A WRITE COMMAND IS ISSUED TO THE CONTROLLER AND 'SUMP' IS SET. VERIFY THAT 'ERR' AND 'HE' SET IN RPCS.
- 8.20 TEST 22 - VERIFY THAT 'FUV' OCCURS WHEN A COMMAND IS ISSUED TO THE CONTROLLER AND 'SUFU' IS SET. VERIFY THAT 'ERR' AND 'HE' BITS SET IN RPCS.
- 8.21 TEST 23 - VERIFY THAT 'NXC' DOES NOT SET WHEN A WRITE COMMAND IS ISSUED USING EACH VALID CYLINDER CODE.
- 8.22 TEST 24 - VERIFY THAT 'NXC' SETS WHEN A WRITE ORDER IS GIVEN FOR EACH INVALID CYLINDER ADDRESS. VERIFY THAT 'ERR' AND 'HE' BITS SET IN RPCS
- 8.23 TEST 25 - VERIFY THAT 'NXT' DOES NOT SET WHEN A WRITE COMMAND IS GIVEN USING EACH VALID TRACK ADDRESS.
- 8.24 TEST 26 - VERIFY THAT 'NXT' OCCURS WHEN A WRITE COMMAND IS GIVEN USING EACH INVALID TRACK ADDRESS CODE UP TO THE LIMIT OF THE REGISTER. VERIFY THAT 'ERR' AND 'HE' BITS SET IN RPCS.
- 8.25 TEST 27 - VERIFY THAT 'NXS' DOES NOT OCCUR WHEN A WRITE COMMAND IS GIVEN USING EACH VALID SECTOR ADDRESS.
- 8.26 TEST 30 - VERIFY THAT 'NXS' OCCURS WHEN A WRITE COMMAND IS GIVEN USING EACH INVALID SECTOR ADDRESS CODE UP TO THE LIMIT OF THE REGISTER. VERIFY THAT 'ERR' AND 'HE' BITS SET IN RPCS.
- 8.27 TEST 31 - VERIFY THAT A 'PROG' ERROR OCCURS IF A COMMAND IS ISSUED WHEN 'SUOL' IS NOT SET. VERIFY THAT 'ERR' AND 'HE' BITS ARE SET IN RPCS.
- 8.28 TEST 32 - VERIFY THAT A 'MODE' ERROR OCCURS IF AN ATTEMPT IS MADE TO WRITE A HEADER WHEN THE CONTROLLER IS IN 'PDP-11' MODE.
- 8.29 TEST 33 - VERIFY THAT AN INTERRUPT OCCURS WHEN AN ATTENTION BIT AND 'AIE' ARE SET. WHEN THE INTERRUPT OCCURS, VERIFY THAT 'AIE' HAS RESET.
- 8.30 TEST 34 - VERIFY THAT THE CONTROLLER DOES NOT GENERATE AN INTERRUPT WHEN 'AIE' IS SET AND NO ATTENTION BITS ARE SET.

- 8.31 TEST 35 - VERIFY THAT EACH ATTENTION BIT SET WILL GENERATE A SEPARATE INTERRUPT. ATTENTION BITS 0 AND 1 ARE SET AND 'AIE' IS SET; VERIFY THAT AN INTERRUPT OCCURS. RESET ATTENTION BIT 0 AND SET 'AIE' AGAIN; VERIFY THAT A SECOND INTERRUPT OCCURS.
- 8.32 TEST 36 - VERIFY THAT AN INTERRUPT OCCURS WHEN THE CONTROLLER IS READY AND 'IDE' IS SET. VERIFY THAT 'IDE' DOES NOT RESET WHEN AN INTERRUPT OCCURS.
- 8.33 TEST 37 - VERIFY THAT AN INTERRUPT DOES NOT OCCUR WHEN THE CONTROLLER IS READY AND 'IDE' IS NOT SET.
- 8.34 TEST 40 - VERIFY THAT A CONTROLLER INTERRUPT OCCURS WITH THE PROCESSOR AT PRIORITY 4.
- 8.35 TEST 41 - VERIFY THAT A CONTROLLER INTERRUPT DOES NOT OCCUR WHEN THE PROCESSOR IS AT PRIORITY 5.
- 8.36 TEST 42 - VERIFY THAT A CONTROLLER INTERRUPT DOES NOT OCCUR WHEN THE PROCESSOR IS AT PRIORITY 6.
- 8.37 TEST 43 - VERIFY THAT A CONTROLLER INTERRUPT DOES NOT OCCUR WHEN THE PROCESSOR IS AT PRIORITY 7.
- 8.38 TEST 44 - VERIFY THAT CONTROLLER READY RESETS WHEN 'GO' IS SET AND THAT CONTROLLER INITIALIZE RETURNS THE CONTROLLER TO THE 'READY' STATE.
- 8.39 TEST 45 - TEST THE 'SUCA' BIT PATH IN THE CONTROLLER.
- 8.40 TEST 46 - VERIFY THAT THE 'SOT' COUNTER COUNTS SECTOR PULSES PROPERLY.
- 8.41 TEST 47 - VERIFY THAT 'INDEX' CLEARS THE 'SOT' COUNTER. CHECK THAT THE 'SOT' CONTAINS SOME COUNT; IF THE 'SOT' IS CLEAR, ISSUE A SECTOR PULSE TO SET A VALUE INTO THE COUNTER. GENERATE AN INDEX PULSE AND VERIFY THAT THE 'SOT' CLEARS.
- 8.42 TEST 50 - VERIFY THAT SILO MEMORY LOGIC. CLEAR THE CONTROLLER; VERIFY THAT THE SILO'S 'INPUT READY' BIT IS SET. LOAD THE SILO AND VERIFY THAT WORD 'BUBBLES' THROUGH THE SILO AND THAT 'OUTPUT READY' SETS. READ THE SILO REGISTER AND VERIFY THAT THE DATA IS CORRECT. AFTER THE SILO REGISTER IS READ, VERIFY THAT 'OUTPUT READY' CLEARS. PERFORM THE TEST USING FLOATING 1'S AND 0'S PATTERN.
- 8.43 TEST 51 - VERIFY THAT THE SILO MEMORY CAN HOLD 64 WORDS. LOAD THE SILO WITH NUMBERS FROM 1 (8) TO 100 (8). VERIFY THAT 'INPUT READY' CLEARS AFTER 64 (10) WORDS HAVE BEEN LOADED INTO THE SILO AND THAT 'OUTPUT READY' IS SET. READ EACH WORD FROM THE SILO AND COMPARE IT. VERIFY THAT 'INPUT READY' SET WHEN THE FIRST WORD IS READ FROM THE FULL SILO. VERIFY THAT 'INPUT READY' REMAINS SET AND THAT 'OUTPUT READY' STAYS SET UNTIL THE SILO IS EMPTY.
- 8.44 TEST 52 - ISSUE A SEEK IN MAINTENANCE MODE. VERIFY THAT 'SET CYLINDER', 'RESET HEAD', 'SET HEAD', AND 'SEEK START' CONTROL SIGNALS OCCUR AT THE PROPER TIME. VERIFY THAT THE HEAD ADDRESS

AND CYLINDER ADDRESS ARE CORRECT AT THE PROPER TIME.

- 8.45 TEST 53 - ISSUE A HOME SEEK IN MAINTENANCE MODE. VERIFY THAT 'CONTROL' AND 'RESTORE' ARE SET ON THE BUS.
- 8.46 TEST 54 - ISSUE A READ IN MAINTENANCE MODE. VERIFY THAT 'CONTROL', 'SELECT HEAD' AND 'READ' ARE ON THE DRIVE BUS OUT LINES.
- 8.47 TEST 55 - ISSUE A WRITE FORMAT IN MAINTENANCE MODE. VERIFY THAT 'CONTROL', 'SELECT HEAD', 'ERASE', AND 'WRITE' ARE SET ON THE BUS OUT LINES.

9. PROGRAM LISTING

!

548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602

```

.TITLE MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST
; *COPYRIGHT (C) 1975, 1977
; *DIGITAL EQUIPMENT CORP.
; *MAYNARD, MASS. 01754
; *
; *PROGRAM BY C. HESS/F. ROEMER
; *
; *THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
; *PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
; *
.SBTTL OPERATIONAL SWITCH SETTINGS
; *
; *      SWITCH      USE
; *      -----      -----
; *      15          HALT ON ERROR
; *      14          LOOP ON TEST
; *      13          INHIBIT ERROR TYPEOUTS
; *      11          INHIBIT ITERATIONS
; *      10          BELL ON ERROR
; *      9           LOOP ON ERROR
; *      8           LOOP ON TEST IN SWR<6:0>
; *      7           EXIT FROM TEST AFTER ERROR DURING FLOATING 1'S AND 0'
; *                  TESTS AND THE SILO DATA TEST.
; *
.SBTTL TRAP CATCHER
; *
; *      =0
; *      ; *ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
; *      ; *SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
; *      ; *LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
; *
; *      =174
DISPREG: .WORD 0      ;; SOFTWARE DISPLAY REGISTER
SWREG:  .WORD 0      ;; SOFTWARE SWITCH REGISTER
; *
.SBTTL ACT11 HOOKS
; *
; *      *****
; *      ; HOOKS REQUIRED BY ACT11
; *      $SVPC=.      ; SAVE PC
; *      =46
; *      $SENDAD      ;; 1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
; *      =52
; *      .WORD 0      ;; 2)SET LOC.52 TO ZERO
; *      =$SVPC      ;; RESTORE PC
; *
.SBTTL STARTING ADDRESSES
; *
; *      =200
; *      ; *200 = NORMAL START
; *      JMP START
; *      ; *204 = SELECT RP11E ADDRESS
; *      JMP START1
; *
.SBTTL BASIC DEFINITIONS

```

```

574      000000
578      000174 000174
579 000174 000000
580 000176 000000
586      000200
587      000046 000046
588 000046 016364
589      000052 000052
590 000052 000000
591      000200
595      000200
597 000200 000137 002364
598 000204 000137 002372

```



```

603      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
604      001100  STACK= 1100
605      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
606      .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
607
608      ;*MISCELLANEOUS DEFINITIONS
609      000011  HT= 11          ;;CODE FOR HORIZONTAL TAB
610      000012  LF= 12          ;;CODE FOR LINE FEED
611      000015  CR= 15          ;;CODE FOR CARRIAGE RETURN
612      000200  CRLF= 200       ;;CODE FOR CARRIAGE RETURN-LINE FEED
613      177776  PS= 177776     ;;PROCESSOR STATUS WORD
614      .EQUIV  PS,PSW
615      177774  STKLMT= 177774  ;;STACK LIMIT REGISTER
616      177772  PIRQ= 177772   ;;PROGRAM INTERRUPT REQUEST REGISTER
617      177570  DSWR= 177570   ;;HARDWARE SWITCH REGISTER
618      177570  DDISP= 177570  ;;HARDWARE DISPLAY REGISTER
619
620      ;*GENERAL PURPOSE REGISTER DEFINITIONS
621      000000  R0= %0          ;;GENERAL REGISTER
622      000001  R1= %1          ;;GENERAL REGISTER
623      000002  R2= %2          ;;GENERAL REGISTER
624      000003  R3= %3          ;;GENERAL REGISTER
625      000004  R4= %4          ;;GENERAL REGISTER
626      000005  R5= %5          ;;GENERAL REGISTER
627      000006  R6= %6          ;;GENERAL REGISTER
628      000007  R7= %7          ;;GENERAL REGISTER
629      000006  SP= %6         ;;STACK POINTER
630      000007  PC= %7         ;;PROGRAM COUNTER
631
632      ;*PRIORITY LEVEL DEFINITIONS
633      000000  PR0= 0          ;;PRIORITY LEVEL 0
634      000040  PR1= 40         ;;PRIORITY LEVEL 1
635      000100  PR2= 100        ;;PRIORITY LEVEL 2
636      000140  PR3= 140        ;;PRIORITY LEVEL 3
637      000200  PR4= 200        ;;PRIORITY LEVEL 4
638      000240  PR5= 240        ;;PRIORITY LEVEL 5
639      000300  PR6= 300        ;;PRIORITY LEVEL 6
640      000340  PR7= 340        ;;PRIORITY LEVEL 7
641
642      ;*"SWITCH REGISTER" SWITCH DEFINITIONS
643      100000  SW15= 100000
644      040000  SW14= 40000
645      020000  SW13= 20000
646      010000  SW12= 10000
647      004000  SW11= 4000
648      002000  SW10= 2000
649      001000  SW09= 1000
650      000400  SW08= 400
651      000200  SW07= 200
652      000100  SW06= 100
653      000040  SW05= 40
654      000020  SW04= 20
655      000010  SW03= 10
656      000004  SW02= 4
657      000002  SW01= 2
658      000001  SW00= 1
    
```

## BASIC DEFINITIONS

```

659 .EQUIV SW09,SW9
660 .EQUIV SW08,SW8
661 .EQUIV SW07,SW7
662 .EQUIV SW06,SW6
663 .EQUIV SW05,SW5
664 .EQUIV SW04,SW4
665 .EQUIV SW03,SW3
666 .EQUIV SW02,SW2
667 .EQUIV SW01,SW1
668 .EQUIV SW00,SW0
669
670 :#DATA BIT DEFINITIONS (BIT00 TO BIT15)
671 100000 BIT15= 100000
672 040000 BIT14= 40000
673 020000 BIT13= 20000
674 010000 BIT12= 10000
675 004000 BIT11= 4000
676 002000 BIT10= 2000
677 001000 BIT09= 1000
678 000400 BIT08= 400
679 000200 BIT07= 200
680 000100 BIT06= 100
681 000040 BIT05= 40
682 000020 BIT04= 20
683 000010 BIT03= 10
684 000004 BIT02= 4
685 000002 BIT01= 2
686 000001 BIT00= 1
687 .EQUIV BIT09,BIT9
688 .EQUIV BIT08,BIT8
689 .EQUIV BIT07,BIT7
690 .EQUIV BIT06,BIT6
691 .EQUIV BIT05,BIT5
692 .EQUIV BIT04,BIT4
693 .EQUIV BIT03,BIT3
694 .EQUIV BIT02,BIT2
695 .EQUIV BIT01,BIT1
696 .EQUIV BIT00,BIT0
697
698 :#BASIC "CPU" TRAP VECTOR ADDRESSES
699 000004 ERRVEC= 4 ;; TIME OUT AND OTHER ERRORS
700 000010 RESVEC= 10 ;; RESERVED AND ILLEGAL INSTRUCTIONS
701 000014 TBITVEC=14 ;; "T" BIT
702 000014 TRTVEC= 14 ;; TRACE TRAP
703 000014 BPTVEC= 14 ;; BREAKPOINT TRAP (BPT)
704 000020 IOTVEC= 20 ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
705 000024 PWRVEC= 24 ;; POWER FAIL
706 000030 EMTVEC= 30 ;; EMULATOR TRAP (EMT) **ERROR**
707 000034 TRAPVEC=34 ;; "TRAP" TRAP
708 000060 TKVEC= 60 ;; TTY KEYBOARD VECTOR
709 000064 TPVEC= 64 ;; TTY PRINTER VECTOR
710 000240 PIRQVEC=240 ;; PROGRAM INTERRUPT REQUEST VECTOR
711
712 .SBTTL RP11E REGISTER INDEX EQUATES
713
714 000000 RPDS=00

```

F02

MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST MACY11 30(1046) 22-JUL-77 15:10 PAGE 17  
DZRPWC.P11 22-JUL-77 14:54 RP11E REGISTER INDEX EQUATES

SEQ 0017

715	000002	RPER=02
716	000004	RPCS=04
717	000006	RPWC=06
718	000010	RPBA=10
719	000012	RPCA=12
720	000014	RPDA=14
721	000016	RPM1=16
722	000020	RPM2=20
723	000022	RPM3=22
724	000024	SUCA=24
725	000026	SILO=26
726		
727		

```

728          .SBTTL COMMON TAGS
729
730          ;:*****
731          ;:THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
732          ;:USED IN THE PROGRAM.
733
734          001100          .=1100
735          001100          SCMTAG:          ;:START OF COMMON TAGS
736          001100          SPASS: .WORD 0          ;:CONTAINS PASS COUNT
737          001102          000          STSTNM: .BYTE 00          ;:CONTAINS THE TEST NUMBER
738          001103          000          SERFLG: .BYTE 00          ;:CONTAINS ERROR FLAG
739          001104          000000          SICNT: .WORD 00          ;:CONTAINS SUBTEST ITERATION COUNT
740          001106          000000          SLPADR: .WORD 00          ;:CONTAINS SCOPE LOOP ADDRESS
741          001110          000000          SLPERR: .WORD 00          ;:CONTAINS SCOPE RETURN FOR ERRORS
742          001112          000000          SERTTL: .WORD 00          ;:CONTAINS TOTAL ERRORS DETECTED
743          001114          000          $ITEMB: .BYTE 00          ;:CONTAINS ITEM CONTROL BYTE
744          001115          001          SERMAX: .BYTE 1          ;:CONTAINS MAX. ERRORS PER TEST
745          001116          000000          SERRPC: .WORD 00          ;:CONTAINS PC OF LAST ERROR INSTRUCTION
746          001120          000000          $GDADR: .WORD 00          ;:CONTAINS ADDRESS OF 'GOOD' DATA
747          001122          000000          $BDADR: .WORD 00          ;:CONTAINS ADDRESS OF 'BAD' DATA
748          001124          000000          $GDDAT: .WORD 00          ;:CONTAINS 'GOOD' DATA
749          001126          000000          $BDDAT: .WORD 00          ;:CONTAINS 'BAD' DATA
750          001130          000000          .WORD 00          ;:RESERVED--NOT TO BE USED
751          001132          000000          .WORD 00
752          001134          000          SAUTOB: .BYTE 00          ;:AUTOMATIC MODE INDICATOR
753          001135          000          SINTAG: .BYTE 0          ;:INTERRUPT MODE INDICATOR
754          001136          000000          .WORD 0
755          001140          177570          SWR: .WORD DSWR          ;:ADDRESS OF SWITCH REGISTER
756          001142          177570          DISPLAY: .WORD DDISP          ;:ADDRESS OF DISPLAY REGISTER
757          001144          177560          $TKS: 177560          ;:TTY KBD STATUS
758          001146          177562          $TKB: 177562          ;:TTY KBD BUFFER
759          001150          177564          $TPS: 177564          ;:TTY PRINTER STATUS REG. ADDRESS
760          001152          177566          $TPB: 177566          ;:TTY PRINTER BUFFER REG. ADDRESS
761          001154          000          $NULL: .BYTE 0          ;:CONTAINS NULL CHARACTER FOR FILLS
762          001155          002          $FILLS: .BYTE 2          ;:CONTAINS # OF FILLER CHARACTERS REQUIRED
763          001156          012          $FILLC: .BYTE 12          ;:INSERT FILL CHARS. AFTER A "LINE FEED"
764          001157          000          $STPFLG: .BYTE 0          ;:"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
765          001160          000000          $STMPD: .WORD 0          ;:USER DEFINED
766          001162          000000          $STIMES: 0          ;:MAX. NUMBER OF ITERATIONS
767          001164          000000          $SESCAPE: 0          ;:ESCAPE ON ERROR ADDRESS
768          001166          177607          000377          $SBELL: .ASCIZ <207><377><377>          ;:CODE FOR BELL
769          001172          077          $SQUES: .ASCII /?/          ;:QUESTION MARK
770          001173          015          $SCRLF: .ASCII <15>          ;:CARRIAGE RETURN
771          001174          000012          $SLF: .ASCIZ <12>          ;:LINE FEED
772          ;:*****
773          001176          000000          MAXCYL: .WORD 0          ;:MAXIMUM CYLINDER ADDRESS CHECKED
774          001200          000000          MAXPAT: .WORD 0          ;:MAXIMUM CYLINDER ADDRESS BITS CHECKED
775          001202          000000          MASK: .WORD 0          ;:STORAGE FOR PATTERN IN FLOATING 1'S & 0'S TESTS
776          001204          000000          BUSADR: .WORD 0          ;:CHANGE RP11 ADDRESS FLAG
777          001206          000000          TPL: .WORD 0          ;:STORE BITS TO BE LOADED INTO RPM3 HERE
778
779          ;RP11E ADDRESSES
780
781          001210          176710          000256          RPADR: .WORD 176710          ;RP11E BUS ADDRESS
782          001212          000254          RPVEC: .WORD 254,256          ;RP11E VECTOR ADDRESSES
783          001216          000240          RPPRIO: .WORD <5*32.>          ;RP11E PRIORITY

```

H02

784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796

001220 000000  
001222 000000  
001224 000000  
001226 000000  
001230 000000  
001232 000000  
001234 000000  
001236 000000  
001240 000000  
001242 000000

;SAVE THE RP11E REGISTERS HERE

SRPDS: .WORD 0  
SRPER: .WORD 0  
SRPCS: .WORD 0  
SRPMC: .WORD 0  
SRPBA: .WORD 0  
SRPCA: .WORD 0  
SRPDA: .WORD 0  
SRPM1: .WORD 0  
SSUCA: .WORD 0  
SSILO: .WORD 0

;DRIVE STATUS REGISTER  
;ERROR REGISTER  
;COMMAND & STATUS REGISTER  
;WORD COUNT REGISTER  
;BUFFER ADDRESS REGISTER  
;CURRENT CYLINDER ADDRESS REGISTER  
;TRACK-SECTOR ADDRESS REGISTER  
;MAINTENANCE REGISTER #1  
;SELECTED UNIT CYLINDER ADDRESS REGISTER  
;SILO REGISTER

797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852

## .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 (SERRPC).
; *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

```

## SERRTB:

## ;ERROR 1

```

      EM1      ;RP11 DIDN'T RESPOND TO ADDRESSING
      DH1
      DT1
      DF

```

## ;ERROR 2

```

      EM2      ;CAN'T LOAD REGISTER CORRECTLY
      DH2
      DT2
      DF2

```

## ;ERROR 3

```

      EM3      ;RESET DIDN'T CLEAR REGISTER
      DH2
      DT2
      DF2

```

## ;ERROR 4

```

      EM4      ;CONTROLLER CLEAR DIDN'T CLEAR REGISTER
      DH2
      DT2
      DF2

```

## ;ERROR 5

```

      EM5      ;FLOATING 1'S & 0'S TEST ERROR
      DH2
      DT2
      DF2

```

## ;ERROR 6

```

      EM6      ;REGISTER RAPID ACCESS TEST ERROR
      DH2
      DT2

```

001244

001244 022516  
001246 027757  
001250 031360  
001252 031660001254 022560  
001256 027775  
001260 031364  
001262 031670001264 022616  
001266 027775  
001270 031364  
001272 031670001274 022652  
001276 027775  
001300 031364  
001302 031670001304 022721  
001306 027775  
001310 031364  
001312 031670001314 022757  
001316 027775  
001320 031364

853	001322	031670	DF2	
854				
855				;ERROR 7
856				
857	001324	023020	EM7	;CAN'T SET 'SURDY' BIT
858	001326	030043	DH7	
859	001330	031376	DT7	
860	001332	031664	DF1	
861				
862				;ERROR 10
863				
864	001334	023046	EM10	;CAN'T CLEAR THE 'SURDY' BIT
865	001336	030043	DH7	
866	001340	031376	DT7	
867	001342	031664	DF1	
868				
869				;ERROR 11
870				
871	001344	023102	EM11	;CAN'T SET 'SUSI'
872	001346	030070	DH11	
873	001350	031404	DT11	
874	001352	031670	DF2	
875				
876				;ERROR 12
877				
878	001354	023123	EM12	; 'DSK ERR' NOT SET WITH 'SUSI'
879	001356	030070	DH11	
880	001360	031404	DT11	
881	001362	031670	DF2	
882				
883				;ERROR 13
884				
885	001364	023161	EM13	; 'ERR' OR 'HE' NOT SET WITH 'SUSI'
886	001366	030070	DH11	
887	001370	031404	DT11	
888	001372	031670	DF2	
889				
890				;ERROR 14
891				
892	001374	023223	EM14	;CAN'T SET 'SUFU'
893	001376	030070	DH11	
894	001400	031404	DT11	
895	001402	031670	DF2	
896				
897				;ERROR 15
898				
899	001404	023244	EM15	;CAN'T CLEAR 'SUFU'
900	001406	030070	DH11	
901	001410	031404	DT11	
902	001412	031670	DF2	
903				
904				;ERROR 16
905				
906	001414	023267	EM16	;CAN'T SET 'SUMP'
907	001416	030070	DH11	
908	001420	031404	DT11	

K02

909	001422	031670	DF2	
910				
911				;ERROR 17
912				
913	001424	023310	EM17	;CAN'T CLEAR 'SUMP'
914	001426	030070	DH11	
915	001430	031404	DT11	
916	001432	031670	DF2	
917				
918				;ERROR 20
919				
920	001434	023333	EM20	;CAN'T SET ATTENTION BIT
921	001436	030135	DH20	
922	001440	031416	DT20	
923	001442	031674	DF20	
924				
925				;ERROR 21
926				
927	001444	023363	EM21	;CAN'T CLEAR ATTENTION BIT
928	001446	030135	DH20	
929	001450	031416	DT20	
930	001452	031674	DF20	
931				
932				;ERROR 22
933				
934	001454	023415	EM22	;RESET DIDN'T CLEAR ATTENTION BITS
935	001456	030043	DH7	
936	001460	031376	DT7	
937	001462	031664	DF1	
938				
939				;ERROR 23
940				
941	001464	023457	EM23	;ATTENTION BITS CLEARED BY WRITING 0'S INTO RPDS
942	001466	030043	DH7	
943	001470	031376	DT7	
944	001472	031664	DF1	
945				
946				;ERROR 24
947				
948	001474	023537	EM24	;CAN'T SET 'WPV' ERROR
949	001476	030172	DH24	
950	001500	031426	DT24	
951	001502	031674	DF20	
952				
953				;ERROR 25
954				
955	001504	023565	EM25	; 'ERR' OR 'HE' NOT SET WITH 'WPV'
956	001506	030172	DH24	
957	001510	031426	DT24	
958	001512	031674	DF20	
959				
960				;ERROR 26
961				
962	001514	023626	EM26	;CAN'T SET 'FUV' ERROR
963	001516	030172	DH24	
964	001520	031426	DT24	



## ERROR POINTER TABLE

965	001522	031674	DF20	
966				
967				;ERROR 27
968				
969	001524	023654	EM27	; 'ERR' OR 'HE' NOT SET WITH 'FUV'
970	001526	030172	DH24	
971	001530	031426	DT24	
972	001532	031674	DF20	
973				
974				;ERROR 30
975				
976	001534	023715	EM30	; 'NXC' ERROR SET WITH VALID CYLINDER ADDRESS
977	001536	030227	DH30	
978	001540	031436	DT30	
979	001542	031700	DF30	
980				
981				;ERROR 31
982				
983	001544	023771	EM31	; 'NXC' DIDN(T SET WITH INVALID CYLINDER ADDRESS
984	001546	030227	DH30	
985	001550	031436	DT30	
986	001552	031700	DF30	
987				
988				;ERROR 32
989				
990	001554	024050	EM32	; 'ERR' OR 'HE' DIDN'T SET WITH 'NXC'
991	001556	030227	DH30	
992	001560	031436	DT30	
993	001562	031700	DF30	
994				
995				;ERROR 33
996				
997	001564	024114	EM33	; 'NXT' ERROR SET WITH VALID TRACK ADDRESS
998	001566	030320	DH33	
999	001570	031452	DT33	
1000	001572	031700	DF30	
1001				
1002				;ERROR 34
1003				
1004	001574	024165	EM34	; 'NXT' DIDN'T SET WITH INVALID TRACK ADDRESS
1005	001576	030320	DH33	
1006	001600	031452	DT33	
1007	001602	031700	DF30	
1008				
1009				;ERROR 35
1010				
1011	001604	024241	EM35	; 'ERR' OR 'HE' DIDN'T SET WITH 'NXT'
1012	001606	030320	DH33	
1013	001610	031452	DT33	
1014	001612	031700	DF30	
1015				
1016				;ERROR 36
1017				
1018	001614	024305	EM36	; 'NXS' ERROR SET WITH VALID SECTOR ADDRESS
1019	001616	030320	DH33	
1020	001620	031452	DT33	

## ERROR POINTER TABLE

1021	001622	031700	DF30	
1022				
1023				;ERROR 37
1024				
1025	001624	024357	EM37	; 'NXS' DIDN'T SET WITH INVALID SECTOR ADDRESS
1026	001626	030320	DH33	
1027	001630	031452	DT33	
1028	001632	031700	DF30	
1029				
1030				;ERROR 40
1031				
1032	001634	024434	EM40	; 'ERR' OR 'HE' DIDN'T SET WITH 'NXS'
1033	001636	030320	DH33	
1034	001640	031452	DT33	
1035	001642	031700	DF30	
1036				
1037				;ERROR 41
1038				
1039	001644	024500	EM41	; CAN'T SET 'PROG' ERROR
1040	001646	030172	DH24	
1041	001650	031426	DT24	
1042	001652	031674	DF20	
1043				
1044				;ERROR 42
1045				
1046	001654	024527	EM42	; 'ERR' OR 'HE' NOT SET WITH 'PROG'
1047	001656	030172	DH24	
1048	001660	031426	DT24	
1049	001662	031674	DF20	
1050				
1051				;ERROR 43
1052				
1053	001664	024571	EM43	; CAN'T SET 'MODE' ERROR
1054	001666	030172	DH24	
1055	001670	031426	DT24	
1056	001672	031674	DF20	
1057				
1058				;ERROR 44
1059				
1060	001674	024620	EM44	; 'ERR' OR 'HE' NOT SET WITH 'MODE'
1061	001676	030172	DH24	
1062	001700	031426	DT24	
1063	001702	031674	DF20	
1064				
1065				;ERROR 45
1066				
1067	001704	024662	EM45	; CAN'T CLEAR 'SUSI'
1068	001706	030070	DH11	
1069	001710	031404	DT11	
1070	001712	031670	DF2	
1071				
1072				;ERROR 46
1073				
1074	001714	024705	EM46	; NO ATTENTION INTERRUPT
1075	001716	030411	DH46	
1076	001720	031466	DT46	

1077	001722	031670	DF2	
1078				
1079				;ERROR 47
1080				
1081	001724	024734	EM47	; 'ATE' DIDN'T CLEAR WHEN INTERRUPT OCCURED
1082	001726	030411	DH46	
1083	001730	031466	DT46	
1084	001732	031670	DF2	
1085				
1086				;ERROR 50
1087				
1088	001734	025006	EM50	;NO INTERRUPT WITH ATTENTION BITS 0 & 1 SET
1089	001736	030471	DH50	
1090	001740	031500	DT50	
1091	001742	031674	DF20	
1092				
1093				;ERROR 51
1094				
1095	001744	025061	EM51	;SECOND INTERRUPT DIDN'T OCCUR WITH ATTENTION BIT 1 SET
1096	001746	030471	DH50	
1097	001750	031500	DT50	
1098	001752	031674	DF20	
1099				
1100				;ERROR 52
1101				
1102	001754	025150	EM52	;NO INTERRUPT FROM CONTROLLER 'READY'
1103	001756	030526	DH52	
1104	001760	031510	DT52	
1105	001762	031664	DF1	
1106				
1107				;ERROR 53
1108				
1109	001764	025215	EM53	; 'READY' INTERRUPT WITH 'IDE' SET
1110	001766	030526	DH52	
1111	001770	031510	DT52	
1112	001772	031664	DF1	
1113				
1114				;ERROR 54
1115				
1116	001774	025256	EM54	;NO INTERRUPT FROM RP11 WITH CPU AT PR4
1117	001776	030526	DH52	
1118	002000	031510	DT52	
1119	002002	031664	DF1	
1120				
1121				;ERROR 55
1122				
1123	002004	025325	EM55	; INTERRUPT FROM RP11 WITH CPU AT PR5
1124	002006	030526	DH52	
1125	002010	031510	DT52	
1126	002012	031664	DF1	
1127				
1128				;ERROR 56
1129				
1130	002014	025371	EM56	; INTERRUPT FROM RP11 WITH CPU AT PR6
1131	002016	030526	DH52	
1132	002020	031510	DT52	

## ERROR POINTER TABLE

1133	002022	031664	DF1	
1134				
1135				;ERROR 57
1136				
1137	002024	025435	EM57	;INTERRUPT FROM RP11 WITH CPU AT PR7
1138	002026	030526	DH52	
1139	002030	031510	DT52	
1140	002032	031664	DF1	
1141				
1142				;ERROR 60
1143				
1144	002034	025501	EM60	; 'READY' DIDN'T CLEAR AT END OF OPERATION
1145	002036	030526	DH52	
1146	002040	031510	DT52	
1147	002042	031664	DF1	
1148				
1149				;ERROR 61
1150				
1151	002044	025552	EM61	; 'SUCA' INCORRECT
1152	002046	030553	DH61	
1153	002050	031516	DT61	
1154	002052	031674	DF20	
1155				
1156				;ERROR 62
1157				
1158	002054	025573	EM62	; INDEX DIDN'T CLEAR 'SOT'
1159	002056	030621	DH62	
1160	002060	031526	DT62	
1161	002062	031664	DF1	
1162				
1163				;ERROR 63
1164				
1165	002064	025624	EM63	; 'SOT' DIDN'T COUNT CORRECTLY
1166	002066	030646	DH63	
1167	002070	031534	DT63	
1168	002072	031674	DF20	
1169				
1170				;ERROR 64
1171				
1172	002074	025661	EM64	; CONTROLLER CLEAR DIDN'T SET SILO INPUT READY
1173	002076	030713	DH64	
1174	002100	031544	DT64	
1175	002102	031674	DF20	
1176				
1177				;ERROR 65
1178				
1179	002104	025736	EM65	; SILO OUTPUT READY NOT SET AFTER WORD LOADED INTO SILO
1180	002106	030757	DH65	
1181	002110	031554	DT65	
1182	002112	031664	DF1	
1183				
1184				;ERROR 66
1185				
1186	002114	026024	EM66	; DATA READ FROM SILO IS INCORRECT
1187	002116	031004	DH66	
1188	002120	031562	DT66	

## ERROR POINTER TABLE

1189	002122	031700	DF30	
1190				
1191				;ERROR 67
1192				
1193	002124	026065	EM67	;SILO INPUT READY DIDN'T CLEAR AFTER 64 WORDS WERE
1194				;LOADED INTO THE SILO
1195	002126	030757	DH65	
1196	002130	031554	DT65	
1197	002132	031664	DF1	
1198				
1199				;ERROR 70
1200				
1201	002134	026175	EM70	;SILO OUTPUT READY DIDN'T CLEAR AFTER SILO EMPTIED
1202	002136	030757	DH65	
1203	002140	031554	DT65	
1204	002142	031664	DF1	
1205				
1206				;ERROR 71
1207				
1208	002144	026257	EM71	;BUS OUT LINES TO THE DRIVE ARE NOT CLEARED BY CONTROLLER CLEAR
1209	002146	031062	DH71	
1210	002150	031576	DT71	
1211	002152	031704	DF71	
1212				
1213				;ERROR 72
1214				
1215	002154	026357	EM72	;DRIVE BUS ERROR: 'SET CYLINDER' AND CYLINDER ADDRESS
1216				;SHOULD BE ON THE BUS
1217	002156	031147	DH72	
1218	002160	031614	DT72	
1219	002162	031700	DF30	
1220				
1221				;ERROR 73
1222				
1223	002164	026472	EM73	;DRIVE BUS ERROR: ONLY 'RESET HEAD' SHOULD BE ON THE BUS
1224	002166	031224	DH73	
1225	002170	031630	DT73	
1226	002172	031700	DF30	
1227				
1228				;ERROR 74
1229				
1230	002174	026562	EM74	;DRIVE BUS ERROR: 'SET HEAD' & HEAD ADDRESS SHOULD BE ON THE BUS
1231	002176	031303	DH74	
1232	002200	031644	DT74	
1233	002202	031700	DF30	
1234				
1235				;ERROR 75
1236				
1237	002204	026662	EM75	;DRIVE BUS ERROR: ONLY 'SEEK START' SHOULD BE ON THE BUS
1238				
1239	002206	031224	DH73	
1240	002210	031630	DT73	
1241	002212	031700	DF30	
1242				
1243				;ERROR 76
1244				

## ERROR POINTER TABLE

1245	002214	026752	EM76	;DRIVE BUS ERROR: 'RESTORE' AND 'CONTROL' SHOULD BE ON THE BUS
1246	002216	031224	DH73	
1247	002220	031630	DT73	
1248	002222	031700	DF30	
1249				
1250				;ERROR 77
1251				
1252	002224	027050	EM77	;DRIVE BUS ERROR: 'CONTROL', 'SELECT HEAD', & 'READ'
1253				;SHOULD BE ON THE BUS
1254	002226	031224	DH73	
1255	002230	031630	DT73	
1256	002232	031700	DF30	
1257				
1258				;ERROR 100
1259				
1260	002234	027164	EM100	;DRIVE BUS ERROR; 'CONTROL', 'SELECT HEAD', 'ERASE',
1261				;AND 'WRITE' SHOULD BE ON THE BUS
1262	002236	031224	DH73	
1263	002240	031630	DT73	
1264	002242	031700	DF30	
1265				
1266				;ERROR 101
1267				
1268	002244	027311	EM101	; 'READY' DIDN'T CLEAR WHEN 'GO' WAS SET
1269	002246	030526	DH52	
1270	002250	031510	DT52	
1271	002252	031664	DF1	
1272				
1273				;ERROR 102
1274				
1275	002254	027360	EM102	;CAN'T SET 'SUOL'
1276	002256	030043	DH7	
1277	002260	031376	DT7	
1278	002262	031664	DF1	
1279				
1280				;ERROR 103
1281				
1282	002264	027401	EM103	;CAN'T CLEAR 'SUOL'
1283	002266	030043	DH7	
1284	002270	031376	DT7	
1285	002272	031664	DF1	
1286				
1287				;ERROR 104
1288				
1289	002274	027424	EM104	;CAN'T SET 'SUSU'
1290	002276	030043	DH7	
1291	002300	031376	DT7	
1292	002302	031664	DF1	
1293				
1294				;ERROR 105
1295				
1296	002304	027445	EM105	;CAN'T CLEAR 'SUSU'
1297	002306	030043	DH7	
1298	002310	031376	DT7	
1299	002312	031664	DF1	
1300				

```

1301          ;ERROR 106
1302
1303 002314 027470          EM106          ;'IDE' NOT SET AFTER INTERRUPT
1304 002316 030526          DH52
1305 002320 031510          DT52
1306 002322 031664          DF1
1307
1308          ;ERROR 107
1309
1310 002324 027526          EM107          ;ATTN INTERRUPT OCCURED WITH NO ATTN BITS SET
1311 002326 030070          DH11
1312 002330 031404          DT11
1313 002332 031670          DF2
1314
1315          ;ERROR 110
1316
1317 002334 027603          EM110          ;SILO NOT FULL, 'INPUT READY' NOT SET
1318 002336 030757          DH65
1319 002340 031554          DT65
1320 002342 031664          DF1
1321
1322          ;ERROR 111
1323
1324 002344 027650          EM111          ;SILO NOT EMPTY, 'OUTPUT READY' NOT SET
1325 002346 030757          DH65
1326 002350 031554          DT65
1327 002352 031664          DF1
1328
1329          ;ERROR 112
1330
1331 002354 027717          EM112          ;'INIT' COMMAND SET 'PROG' ERROR
1332 002356 030172          DH24
1333 002360 031426          DT24
1334 002362 031674          DF20
1335
1336          .SBTTL START OF PROGRAM
1337
1338
1339 002364 005037 001204    START: CLR    BUSADR          ;CLEAR THE ADDRESS CHANGE FLAG
1340 002370 000403          BR      START2          ;GO TO SETUP
1341 002372 012737 177777 001204    START1: MOV    #-1,BUSADR    ;SET ADDRESS CHANGE FLAG
1342 002400 000005          START2: RESET          ;CLEAR THE BUS
1343          .SBTTL INITIALIZE THE COMMON TAGS
1344          ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1345 002402 012706 001100    MOV    #$CMTAG,R6          ;;FIRST LOCATION TO BE CLEARED
1346 002406 005026          CLR    (R6)+              ;;CLEAR MEMORY LOCATION
1347 002410 022706 001140    CMP    $SMR,R6 ;;DONE?
1348 002414 001374          BNE    .-6                ;;LOOP BACK IF NO
1349 002416 012706 001100    MOV    $STACK,SP          ;;SETUP THE STACK POINTER
1350          ;;INITIALIZE A FEW VECTORS
1351 002422 012737 020734 000020    MOV    $$SCOPE,@#IOTVEC    ;;IOT VECTOR FOR SCOPE ROUTINE
1352 002430 012737 000340 000022    MOV    #340,@#IOTVEC+2    ;;LEVEL 7
1353 002436 012737 016404 000030    MOV    $ERROR,@#EMTVEC    ;;EMT VECTOR FOR ERROR ROUTINE
1354 002444 012737 000340 000032    MOV    #340,@#EMTVEC+2    ;;LEVEL 7
1355 002452 012737 021310 000034    MOV    $TRAP,@#TRAPVEC    ;;TRAP VECTOR FOR TRAP CALLS
1356 002460 012737 000340 000036    MOV    #340,@#TRAPVEC+2   ;LEVEL 7

```

```

1357 002466 013737 016320 016312      MOV      SENDCT,SEOPCT      ;; SETUP END-OF-PROGRAM COUNTER
1358 002474 005037 001162              CLR      STIMES            ;; INITIALIZE NUMBER OF ITERATIONS
1359 002500 005037 001164              CLR      $ESCAPE          ;; CLEAR THE ESCAPE ON ERROR ADDRESS
1360 002504 112737 000001 001115      MOV      #1,$SERMAX        ;; ALLOW ONE ERROR PER TEST
1361 002512 012737 002512 001106      MOV      #.,$LPADR        ;; INITIALIZE THE LOOP ADDRESS FOR SCOPE
1362 002520 012737 002520 001110      MOV      #.,$LPERR        ;; SETUP THE ERROR LOOP ADDRESS
1363                                     ;; SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS
1364                                     ;; EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
1365 002526 013746 000004              MOV      @ERRVEC,-(SP)     ;; SAVE ERROR VECTOR
1366 002532 012737 002566 000004      MOV      #64,$ERRVEC      ;; SET UP ERROR VECTOR
1367 002540 012737 177570 001140      MOV      #DSWR,$SWR       ;; SETUP FOR A HARDWARE SWICH REGISTER
1368 002546 012737 177570 001142      MOV      #DDISP,$DISPLAY  ;; AND A HARDWARE DISPLAY REGISTER
1369 002554 022777 177777 176356      CMP      #-1,$SWR         ;; TRY TO REFERENCE HARDWARE SWR
1370 002562 001012              BNE      66$              ;; BRANCH IF NO TIMEOUT TRAP OCCURRED
1371                                     ;; AND THE HARDWARE SWR IS NOT = -1
1372 002564 000403              BR       65$              ;; BRANCH IF NO TIMEOUT
1373 002566 012716 002574      64$:   MOV      #65,$(SP)     ;; SET UP FOR TRAP RETURN
1374 002572 000002              RTI
1375 002574 012737 000176 001140 65$:   MOV      #SWREG,$SWR      ;; POINT TO SOFTWARE SWR
1376 002602 012737 000174 001142      MOV      #DISPREG,$DISPLAY
1377 002610 012637 000004      66$:   MOV      (SP)+,@ERRVEC   ;; RESTORE ERROR VECTOR
1378
1379 002614 005037 001100              CLR      $PASS            ;; CLEAR THE PASS COUNT
1380 002620 005227 177777              INC      #-1              ;; FIRST START ?
1381 002624 001042              BNE     1$                ;; BR IF NOT
1382 002626 103441              BCS     1$                ;; BR IF NOT
1383 002630 104401 002636              TYPE    ,68$             ;; TYPE ASCIZ STRING
1384 002634 000431              BR      67$              ;; GET OVER THE ASCIZ
1385                                     ;; 68$: .ASCIZ <15><12><12>/MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST/
1386 002720      67$:   TST      @#42             ;; UNDER MONITOR CONTROL ?
1387 002720 005737 000042              BNE     1$                ;; BR IF YES
1388 002724 001002              TYPE    ,DRVOL           ;; TYPE ONLINE MESSAGE
1389 002726 104401 022123      1$:   .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1390 002732
1391                                     .SBTTL  TST      @#42             ;; ARE WE RUNNING UNDER XXDP/ACT?
1392 002732 005737 000042              BNE     69$              ;; BRANCH IF YES
1393 002736 001006              CMP     $SWR,$SWREG      ;; SOFTWARE SWITCH REG SELECTED?
1394 002740 023727 001140 000176      BNE     70$              ;; BRANCH IF NO
1395 002746 001005              GTSWR  70$              ;; GET SOFT-SWR SETTINGS
1396 002750 104406
1397 002752 000403      69$:   BR      70$
1398 002754 112737 000001 001134 70$:   MOV      #1,$AUTOB      ;; SET AUTO-MODE INDICATOR
1399 002762
1400 002762 004737 021376              JSR     PC,GETADR        ;; CHECK THE BUS ADDRESS
1401 002766 013777 001216 176220      MOV     RPPRIO,@RPVEC+2  ;; SETUP RP11 PRIORITY
1402 002774 104401 001173      RSTART: TYPE ,SCRLF      ;; CR-LF
1403 003000 005037 000000              CLR     @#0              ;; CLEAR LOCATION 0
1404 003004 005077 176202              CLR     @RPVEC          ;; SETUP VECTOR TO LOCATION 0
1405 003010 013704 001210              MOV     RPADR,R4         ;; LOAD RP11 ADDRESS POINTER
1406 003014 012764 000000 000004      MOV     #0,RPCS(R4)     ;; SELECT DRIVE 0
1407 003022 032764 020000 000000      BIT     #BIT13,RPDS(R4) ;; DRIVE 0 AN RPD3 ?
1408 003030 001007              BNE     1$                ;; BR IF NOT
1409 003032 012737 000312 001176      MOV     #202.,MAXCYL    ;; SET MAXIMUM CYLINDER ADDRESS TO 202
1410 003040 012737 000400 001200      MOV     #256.,MAXPAT    ;; CYLINDER BITS TO BE TESTED
1411 003046 000406              BR      2$                ;; CONTINUE
1412 003050 012737 000625 001176 1$:   MOV     #405.,MAXCYL    ;; SET MAXIMUM CYLINDER ADDRESS TO 405
    
```



```

1413 003056 012737 001000 001200      MOV      #512, MAXPAT      ;CYLINDER BITS TO BE TESTED
1414 003064 012737 003064 001110 25:    MOV      #., SLPERR      ;LOAD LOOP ON ERROR ADDRESS
1415 003072 012737 003072 001106      MOV      #., SLPADR      ;LOAD LOOP ON TEST ADDRESS
1416 003100 012737 000340 177776      MOV      #PR7, PSW      ;SET PROCESSOR PRIORITY TO MAXIMUM
1417 003106 005037 001102      CLR      STSTNM          ;CLEAR TEST NUMBER
1418 003112 005037 001104      CLR      SICNT           ;CLEAR ITERATION COUNT
1419 003116 000137 003122      JMP      TSTO            ;START TEST 0
1420
1421
1422                                     .SBTTL  ### START OF TESTS ###
1423
1424                                     ;:*****
1425                                     ;*TEST 0          CONTROLLER RESET TEST
1426
1427                                     ;*TEST THAT 'RESET' CLEARS THE CONTROLLER.
1428
1429                                     ;:*****
1430
1431 TSTO:
1432 003122 000240      NOP
1433 003124 012737 003152 001106      MOV      #RESRP, SLPADR  ;LOAD LOOP ON TEST ADDRESS
1434 003132 012737 003152 001110      MOV      #RESRP, SLPERR  ;LOAD ERROR LOOP ADDRESS
1435 003140 013704 001210      MOV      RPADR, R4      ;RP11E BUS ADDRESS
1436 003144 012737 000002 001162      MOV      #2, STIMES     ;DO 2 ITERATIONS
1437 003152 012706 001100      MOV      #STACK, SP    ;SETUP THE STACK POINTER
1438 003156 012764 176656 000002      MOV      #176656, RPER(R4) ;LOAD RPER
1439 003164 012764 037576 000004      MOV      #37576, RPCS(R4) ;LOAD RPCS
1440 003172 012764 177777 000006      MOV      #-1, RPWC(R4)  ;LOAD RPWC
1441 003200 012764 177777 000010      MOV      #-1, RPBA(R4)  ;LOAD RPBA
1442 003206 012764 000777 000012      MOV      #777, RPCA(R4) ;LOAD RPCA
1443 003214 012764 017417 000014      MOV      #17417, RPDA(R4) ;LOAD RPDA
1444 003222 004737 022022      JSR      PC, SAVRP      ;SAVE THE RP11 REGISTERS
1445 003226
1446 003226 012737 000002 001122 15:    MOV      #RPER, SBADR   ;REGISTER ADDRESS INDEX
1447 003234 060437 001122      ADD      R4, SBADR      ;FORM REGISTER ADDRESS
1448 003240 012737 176656 001124      MOV      #176656, SGDDAT ;GOOD DATA
1449 003246 013737 001222 001126      MOV      $RPER, SBDDAT  ;CONTENTS TO BE CHECKED
1450 003254 053737 001126 001124      BIS      SBDDAT, SGDDAT ;SET UNUSED BITS FOR TYPEOUT
1451 003262 023737 001124 001126      CMP      SGDDAT, SBDDAT ;DID RPER LOAD PROPERLY ?
1452 003270 001401      BEQ     25             ;BR IF IT DID
1453 003272 104002      ERROR   2             ;COULD NOT LOAD RPER CORRECTLY
1454 003274
1455 003274 012737 000004 001122 25:    MOV      #RPCS, SBADR   ;REGISTER ADDRESS INDEX
1456 003302 060437 001122      ADD      R4, SBADR      ;FORM REGISTER ADDRESS
1457 003306 012737 177776 001124      MOV      #177776, SGDDAT ;GOOD DATA
1458 003314 013737 001224 001126      MOV      $RPCS, SBDDAT  ;CONTENTS TO BE CHECKED
1459 003322 053737 001126 001124      BIS      SBDDAT, SGDDAT ;SET UNUSED BITS FOR TYPEOUT
1460 003330 023737 001124 001126      CMP      SGDDAT, SBDDAT ;DID RPCS LOAD PROPERLY ?
1461 003336 001401      BEQ     35             ;BR IF IT DID
1462 003340 104002      ERROR   2             ;COULD NOT LOAD RPCS CORRECTLY
1463 003342
1464 003342 012737 000006 001122 35:    MOV      #RPWC, SBADR   ;REGISTER ADDRESS INDEX
1465 003350 060437 001122      ADD      R4, SBADR      ;FORM REGISTER ADDRESS
1466 003354 012737 177777 001124      MOV      #-1, SGDDAT    ;GOOD DATA
1467 003362 013737 001226 001126      MOV      $RPWC, SBDDAT  ;CONTENTS TO BE CHECKED
1468 003370 053737 001126 001124      BIS      SBDDAT, SGDDAT ;SET UNUSED BITS FOR TYPEOUT

```

1469	003376	023737	001124	001126	CMP	SGDDAT,SBDDAT	:DID RPWC LOAD PROPERLY ?
1470	003404	001401			BEQ	45	:BR IF IT DID
1471	003406	104002			ERROR	2	:COULD NOT LOAD RPWC CORRECTLY
1472	003410						
1473	003410	012737	000010	001122	45:	MOV	:REGISTER ADDRESS INDEX
1474	003416	060437	001122		ADD	R4,SBADR	:FORM REGISTER ADDRESS
1475	003422	012737	177777	001124	MOV	#-1,SGDDAT	:GOOD DATA
1476	003430	013737	001230	001126	MOV	SRPBA,SBDDAT	:CONTENTS TO BE CHECKED
1477	003436	053737	001126	001124	BIS	SBDDAT,SGDDAT	:SET UNUSED BITS FOR TYPEOUT
1478	003444	023737	001124	001126	CMP	SGDDAT,SBDDAT	:DID RPBA LOAD PROPERLY ?
1479	003452	001401			BEQ	55	:BR IF IT DID
1480	003454	104002			ERROR	2	:COULD NOT LOAD RPBA CORRECTLY
1481	003456						
1482	003456	012737	000012	001122	55:	MOV	:REGISTER ADDRESS INDEX
1483	003464	060437	001122		ADD	R4,SBADR	:FORM REGISTER ADDRESS
1484	003470	012737	000777	001124	MOV	#777,SGDDAT	:GOOD DATA
1485	003476	013737	001232	001126	MOV	SRPCA,SBDDAT	:CONTENTS TO BE CHECKED
1486	003504	053737	001126	001124	BIS	SBDDAT,SGDDAT	:SET UNUSED BITS FOR TYPEOUT
1487	003512	023737	001124	001126	CMP	SGDDAT,SBDDAT	:DID RPCA LOAD PROPERLY ?
1488	003520	001401			BEQ	65	:BR IF IT DID
1489	003522	104002			ERROR	2	:COULD NOT LOAD RPCA CORRECTLY
1490	003524						
1491	003524	012737	000014	001122	65:	MOV	:REGISTER ADDRESS INDEX
1492	003532	060437	001122		ADD	R4,SBADR	:FORM REGISTER ADDRESS
1493	003536	012737	017417	001124	MOV	#17417,SGDDAT	:GOOD DATA
1494	003544	013737	001234	001126	MOV	SRPDA,SBDDAT	:CONTENTS TO BE CHECKED
1495	003552	053737	001126	001124	BIS	SBDDAT,SGDDAT	:SET UNUSED BITS FOR TYPEOUT
1496	003560	023737	001124	001126	CMP	SGDDAT,SBDDAT	:DID RPDA LOAD PROPERLY ?
1497	003566	001401			BEQ	75	:BR IF IT DID
1498	003570	104002			ERROR	2	:COULD NOT LOAD RPDA CORRECTLY
1499	003572	000005			75:	RESET	:CLEAR THE RP11E
1500	003574	004737	022022		JSR	PC,SAVRP	:SAVE THE RP11E REGISTERS
1501	003600	005037	001124		CLR	SGDDAT	:EXPECTED VALUE
1502	003604	012737	000002	001122	MOV	SRPER,SBADR	:REGISTER ADDRESS INDEX VALUE
1503	003612	060437	001122		ADD	R4,SBADR	:FORM REGISTER ADDRESS
1504	003616	013737	001222	001126	MOV	SRPER,SBDDAT	:REGISTER VALUE FOR TYPEOUT
1505	003624	023737	001124	001126	CMP	SGDDAT,SBDDAT	:IS REGISTER RPER CLEAR ?
1506	003632	001401			BEQ	.+4	:BR IF IT IS
1507	003634	104003			ERROR	3	:RESET DIDN'T CLEAR INDICATED REGISTER
1508	003636	012737	000006	001122	MOV	SRPWC,SBADR	:REGISTER ADDRESS INDEX VALUE
1509	003644	060437	001122		ADD	R4,SBADR	:FORM REGISTER ADDRESS
1510	003650	013737	001226	001126	MOV	SRPWC,SBDDAT	:REGISTER VALUE FOR TYPEOUT
1511	003656	023737	001124	001126	CMP	SGDDAT,SBDDAT	:IS REGISTER RPWC CLEAR ?
1512	003664	001401			BEQ	.+4	:BR IF IT IS
1513	003666	104003			ERROR	3	:RESET DIDN'T CLEAR INDICATED REGISTER
1514	003670	012737	000010	001122	MOV	SRPBA,SBADR	:REGISTER ADDRESS INDEX VALUE
1515	003676	060437	001122		ADD	R4,SBADR	:FORM REGISTER ADDRESS
1516	003702	013737	001230	001126	MOV	SRPBA,SBDDAT	:REGISTER VALUE FOR TYPEOUT
1517	003710	023737	001124	001126	CMP	SGDDAT,SBDDAT	:IS REGISTER RPBA CLEAR ?
1518	003716	001401			BEQ	.+4	:BR IF IT IS
1519	003720	104003			ERROR	3	:RESET DIDN'T CLEAR INDICATED REGISTER
1520	003722	012737	000012	001122	MOV	SRPCA,SBADR	:REGISTER ADDRESS INDEX VALUE
1521	003730	060437	001122		ADD	R4,SBADR	:FORM REGISTER ADDRESS
1522	003734	013737	001232	001126	MOV	SRPCA,SBDDAT	:REGISTER VALUE FOR TYPEOUT
1523	003742	023737	001124	001126	CMP	SGDDAT,SBDDAT	:IS REGISTER RPCA CLEAR ?
1524	003750	001401			BEQ	.+4	:BR IF IT IS

```

1525 003752 104003          ERROR 3          ;RESET DIDN'T CLEAR INDICATED REGISTER
1526 003754 012737 000016 001122  MOV      #RPM1,$B0ADR ;REGISTER ADDRESS INDEX VALUE
1527 003762 060437 001122          ADD      R4,$B0ADR   ;FORM REGISTER ADDRESS
1528 003766 013737 001236 001126  MOV      $RPM1,$BDDAT ;REGISTER VALUE FOR TYPEOUT
1529 003774 042737 004000 001126  BIC      #4000,$BDDAT ;CLEAR THE INPUT READY BIT
1530 004002 023737 001124 001126  CMP      $GDDAT,$BDDAT ;IS RPM1 CLEAR ?
1531 004010 001401          BEQ      .+4         ;BR IF IT IS
1532 004012 104003          ERROR 3          ;RESET DIDN'T CLEAR INDICATED REGISTER
1533 004014 012737 000014 001122  MOV      #RPDA,$B0ADR ;REGISTER ADDRESS INDEX VALUE
1534 004022 060437 001122          ADD      R4,$B0ADR   ;FORM REGISTER ADDRESS
1535 004026 013737 001234 001126  MOV      $RPDA,$BDDAT ;REGISTER CONTENTS FOR TYPEOUT
1536 004034 042737 000360 001126  BIC      #360,$BDDAT  ;CLEAR THE 'SOT' BITS
1537 004042 023737 001124 001126  CMP      $GDDAT,$BDDAT ;IS REGISTER RPDA CLEAR ?
1538 004050 001401          BEQ      .+4         ;BR IF IT IS
1539 004052 104003          ERROR 3          ;RESET DIDN'T CLEAR INDICATED REGISTER
1540 004054 012737 000200 001124  MOV      #200,$GDDAT  ;EXPECTED VALUE
1541 004062 013737 001224 001126  MOV      $RPCS,$BDDAT ;REGISTER CONTENTS FOR TYPEOUT
1542 004070 023737 001124 001126  CMP      $GDDAT,$BDDAT ;CONTENTS CORRECT ?
1543 004076 001401          BEQ      8$         ;BR IF CORRECT
1544 004100 104003          ERROR 3          ;RESET DIDN'T CLEAR INDICATED REGISTER
1545 004102 000004          BS:          SCOPE ;LOOP ?

```

```

1546
1547
1548 ;:*****
1549 ;*TEST 1          TEST SETTING 'SUOL' (SELECTED UNIT ONLINE)

```

```

1550 ;:*****
1551 ;*TEST1:

```

```

1552 004104 000240          NOP
1553 004106 012737 004130 001106  MOV      #SETOL,$LPADR ;LOAD LOOP ON TEST ADDRESS
1554 004114 012737 004130 001110  MOV      #SETOL,$LPERR ;LOAD ERROR LOOP ADDRESS
1555 004122 013704 001210          MOV      RPADR,R4     ;RPIIE BUS ADDRESS
1556 004126 000005          RESET              ;CLEAR THE CONTROLLER
1557 004130 012706 001100          SETOL: MOV      #STACK,SP  ;SETUP THE STACK POINTER
1558 004134 004737 021714          JSR      PC,CLRP     ;CLEAR THE CONTROLLER
1559 004140 052764 020000 000022  BIS      #BIT13,RPM3(R4) ;SET MAINT UNIT ONLINE
1560 004146 004737 022022          JSR      PC,SAVRP    ;SAVE THE RPIIE REGISTERS
1561 004152 032737 040000 001220  BIT      #BIT14,$RPDS ;DID SELECTED UNIT ONLINE SET ?
1562 004160 001002          BNE      1$         ;BRANCH IF SET
1563 004162 104102          ERROR 102         ;SELECTED UNIT ONLINE NOT SET
1564 004164 000412          BR        2$         ;BYPASS REST OF CHECKS
1565 004166 042764 020000 000022  1$: BIC      #BIT13,RPM3(R4) ;CLEAR MAINT UNIT ONLINE
1566 004174 004737 022022          JSR      PC,SAVRP    ;SAVE THE RPIIE REGISTERS
1567 004200 032737 040000 001220  BIT      #BIT14,$RPDS ;DID SELECTED UNIT ONLINE CLEAR?
1568 004206 001401          BEQ      2$         ;
1569 004210 104103          ERROR 103         ;SELECTED UNIT ONLINE DID NOT CLEAR
1570 004212 000004          2$: SCOPE          ;LOOP ?
1571 004214 052764 020000 000022  BIS      #BIT13,RPM3(R4) ;SET MAINT DRIVE ONLINE

```

```

1572
1573
1574 ;:*****
1575 ;*TEST 2          CONTROLLER CLEAR TEST

```

```

1576
1577 ;*TEST THE ABILITY OF BIT 0 OF RPCS TO CLEAR THE CONTROLLER
1578 ;*WHEN THE COMMAND BITS ARE 0.
1579
1580 ;:*****

```

```

1581 004222          TST2:
1582 004222 000240      NOP
1583 004224 012737 004252 001106  MOV    #CLEARP,SLPADR ;LOAD LOOP ON TEST ADDRESS
1584 004232 012737 004252 001110  MOV    #CLEARP,SLPERR ;LOAD ERROR LOOP ADDRESS
1585 004240 013704 001210          MOV    RPADR,R4      ;RP11E BUS ADDRESS
1586 004244 012737 000002 001162  MOV    #2,STIMES    ;DO 2 ITERATIONS
1587 004252 000005          CLEARP: RESET ;CLEAR THE BUS
1588 004254 012706 001100          MOV    #STACK,SP    ;SETUP THE STACK POINTER
1589 004260 012764 176656 000002  MOV    #176656,RPER(R4) ;LOAD RPER
1590 004266 012764 037576 000004  MOV    #37576,RPCS(R4) ;LOAD RPCS
1591 004274 012764 177777 000006  MOV    #-1,RPWC(R4)  ;LOAD RPWC
1592 004302 012764 177777 000010  MOV    #-1,RPBA(R4) ;LOAD RPBA
1593 004310 012764 000777 000012  MOV    #777,RPCA(R4) ;LOAD RPCA
1594 004316 012764 017417 000014  MOV    #17417,RPDA(R4) ;LOAD RPDA
1595 004324 004737 022022          JSR    PC,SAVRP     ;SAVE THE RP11 REGISTERS
1596 004330
1597 004330 012737 000002 001122  1S:   MOV    #RPER,SBDADR ;REGISTER ADDRESS INDEX
1598 004336 060437 001122          ADD    R4,SBDADR   ;FORM REGISTER ADDRESS
1599 004342 012737 176656 001124  MOV    #176656,$GDDAT ;GOOD DATA
1600 004350 013737 001222 001126  MOV    $RPER,$SDDAT ;CONTENTS TO BE CHECKED
1601 004356 053737 001126 001124  BIS    $SDDAT,$GDDAT ;SET UNUSED BITS FOR TYPEOUT
1602 004364 023737 001124 001126  CMP    $GDDAT,$SDDAT ;DID RPER LOAD PROPERLY ?
1603 004372 001401          BEQ    2S          ;BR IF IT DID
1604 004374 104002          ERROR  2          ;COULD NOT LOAD RPER CORRECTLY
1605 004376
1606 004376 012737 000004 001122  2S:   MOV    #RPCS,SBDADR ;REGISTER ADDRESS INDEX
1607 004404 060437 001122          ADD    R4,SBDADR   ;FORM REGISTER ADDRESS
1608 004410 012737 177776 001124  MOV    #177776,$GDDAT ;GOOD DATA
1609 004416 013737 001224 001126  MOV    $RPCS,$SDDAT ;CONTENTS TO BE CHECKED
1610 004424 053737 001126 001124  BIS    $SDDAT,$GDDAT ;SET UNUSED BITS FOR TYPEOUT
1611 004432 023737 001124 001126  CMP    $GDDAT,$SDDAT ;DID RPCS LOAD PROPERLY ?
1612 004440 001401          BEQ    3S          ;BR IF IT DID
1613 004442 104002          ERROR  2          ;COULD NOT LOAD RPCS CORRECTLY
1614 004444
1615 004444 012737 000006 001122  3S:   MOV    #RPWC,SBDADR ;REGISTER ADDRESS INDEX
1616 004452 060437 001122          ADD    R4,SBDADR   ;FORM REGISTER ADDRESS
1617 004456 012737 177777 001124  MOV    #-1,$GDDAT   ;GOOD DATA
1618 004464 013737 001226 001126  MOV    $RPWC,$SDDAT ;CONTENTS TO BE CHECKED
1619 004472 053737 001126 001124  BIS    $SDDAT,$GDDAT ;SET UNUSED BITS FOR TYPEOUT
1620 004500 023737 001124 001126  CMP    $GDDAT,$SDDAT ;DID RPWC LOAD PROPERLY ?
1621 004506 001401          BEQ    4S          ;BR IF IT DID
1622 004510 104002          ERROR  2          ;COULD NOT LOAD RPWC CORRECTLY
1623 004512
1624 004512 012737 000010 001122  4S:   MOV    #RPBA,SBDADR ;REGISTER ADDRESS INDEX
1625 004520 060437 001122          ADD    R4,SBDADR   ;FORM REGISTER ADDRESS
1626 004524 012737 177777 001124  MOV    #-1,$GDDAT   ;GOOD DATA
1627 004532 013737 001230 001126  MOV    $RPBA,$SDDAT ;CONTENTS TO BE CHECKED
1628 004540 053737 001126 001124  BIS    $SDDAT,$GDDAT ;SET UNUSED BITS FOR TYPEOUT
1629 004546 023737 001124 001126  CMP    $GDDAT,$SDDAT ;DID RPBA LOAD PROPERLY ?
1630 004554 001401          BEQ    5S          ;BR IF IT DID
1631 004556 104002          ERROR  2          ;COULD NOT LOAD RPBA CORRECTLY
1632 004560
1633 004560 012737 000012 001122  5S:   MOV    #RPCA,SBDADR ;REGISTER ADDRESS INDEX
1634 004566 060437 001122          ADD    R4,SBDADR   ;FORM REGISTER ADDRESS
1635 004572 012737 000777 001124  MOV    #777,$GDDAT  ;GOOD DATA
1636 004600 013737 001232 001126  MOV    $RPCA,$SDDAT ;CONTENTS TO BE CHECKED

```

1637	004606	053737	001126	001124	BIS	\$BDDAT,\$GDDAT	;SET UNUSED BITS FOR TYPEOUT
1638	004614	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	;DID RPCA LOAD PROPERLY ?
1639	004622	001401			BEQ	65	;BR IF IT DID
1640	004624	104002			ERROR	2	;COULD NOT LOAD RPCA CORRECTLY
1641	004626						
1642	004626	012737	000014	001122	65:	MOV	\$RPDA,\$BDADR ;REGISTER ADDRESS INDEX
1643	004634	060437	001122		ADD	R4,\$BDADR ;FORM REGISTER ADDRESS	
1644	004640	012737	017417	001124	MOV	\$17417,\$GDDAT ;GOOD DATA	
1645	004646	013737	001234	001126	MOV	\$RPDA,\$BDDAT ;CONTENTS TO BE CHECKED	
1646	004654	053737	001126	001124	BIS	\$BDDAT,\$GDDAT ;SET UNUSED BITS FOR TYPEOUT	
1647	004662	023737	001124	001126	CMP	\$GDDAT,\$BDDAT ;DID RPDA LOAD PROPERLY ?	
1648	004670	001401			BEQ	75	;BR IF IT DID
1649	004672	104002			ERROR	2	;COULD NOT LOAD RPDA CORRECTLY
1650	004674	012737	060001	001206	75:	MOV	\$BIT14:\$BIT13:\$BIT0,\$TPL
1651	004702	112764	000001	000004	MOVB	\$1,\$RPCS(R4) ;SET THE 'GO' BIT	
1652	004710	004737	021764		JSR	PC,\$T3P ;GENERATE 3 CLOCK PULSES	
1653	004714	004737	022022		JSR	PC,\$SAVRP ;SAVE THE RPIIE REGISTERS	
1654	004720	032737	002000	001222	BIT	\$BIT10,\$RPER ;DID 'PROG' SET WITH CLEAR ?	
1655	004726	001401			BEQ	95	;BR IF NOT
1656	004730	104112			ERROR	112	
1657	004732	005037	001124		95:	CLR	\$GDDAT ;EXPECTED VALUE
1658	004736	012737	000002	001122	MOV	\$RPER,\$BDADR ;REGISTER ADDRESS INDEX VALUE	
1659	004744	060437	001122		ADD	R4,\$BDADR ;FORM REGISTER ADDRESS	
1660	004750	013737	001222	001126	MOV	\$RPER,\$BDDAT ;REGISTER VALUE FOR TYPEOUT	
1661	004756	023737	001124	001126	CMP	\$GDDAT,\$BDDAT ;IS REGISTER RPER CLEAR ?	
1662	004764	001401			BEQ	4	;BR IF IT IS
1663	004766	104004			ERROR	4	;CONTROLLER CLEAR DIDN'T CLEAR INDICATED REGISTER
1664	004770	012737	000006	001122	MOV	\$RPWC,\$BDADR ;REGISTER ADDRESS INDEX VALUE	
1665	004776	060437	001122		ADD	R4,\$BDADR ;FORM REGISTER ADDRESS	
1666	005002	013737	001226	001126	MOV	\$RPWC,\$BDDAT ;REGISTER VALUE FOR TYPEOUT	
1667	005010	023737	001124	001126	CMP	\$GDDAT,\$BDDAT ;IS REGISTER RPWC CLEAR ?	
1668	005016	001401			BEQ	4	;BR IF IT IS
1669	005020	104004			ERROR	4	;CONTROLLER CLEAR DIDN'T CLEAR INDICATED REGISTER
1670	005022	012737	000010	001122	MOV	\$RPBA,\$BDADR ;REGISTER ADDRESS INDEX VALUE	
1671	005030	060437	001122		ADD	R4,\$BDADR ;FORM REGISTER ADDRESS	
1672	005034	013737	001230	001126	MOV	\$RPBA,\$BDDAT ;REGISTER VALUE FOR TYPEOUT	
1673	005042	023737	001124	001126	CMP	\$GDDAT,\$BDDAT ;IS REGISTER RPBA CLEAR ?	
1674	005050	001401			BEQ	4	;BR IF IT IS
1675	005052	104004			ERROR	4	;CONTROLLER CLEAR DIDN'T CLEAR INDICATED REGISTER
1676	005054	012737	000012	001122	MOV	\$RPCA,\$BDADR ;REGISTER ADDRESS INDEX VALUE	
1677	005062	060437	001122		ADD	R4,\$BDADR ;FORM REGISTER ADDRESS	
1678	005066	013737	001232	001126	MOV	\$RPCA,\$BDDAT ;REGISTER VALUE FOR TYPEOUT	
1679	005074	023737	001124	001126	CMP	\$GDDAT,\$BDDAT ;IS REGISTER RPCA CLEAR ?	
1680	005102	001401			BEQ	4	;BR IF IT IS
1681	005104	104004			ERROR	4	;CONTROLLER CLEAR DIDN'T CLEAR INDICATED REGISTER
1682	005106	012737	000016	001122	MOV	\$RPM1,\$BDADR ;REGISTER ADDRESS INDEX VALUE	
1683	005114	060437	001122		ADD	R4,\$BDADR ;FORM REGISTER ADDRESS	
1684	005120	013737	001236	001126	MOV	\$RPM1,\$BDDAT ;REGISTER VALUE FOR TYPEOUT	
1685	005126	042737	004000	001126	BIC	\$4000,\$BDDAT ;CLEAR THE INPUT READY BIT	
1686	005134	023737	001124	001126	CMP	\$GDDAT,\$BDDAT ;IS RPM1 CLEAR ?	
1687	005142	001401			BEQ	4	;BR IF IT IS
1688	005144	104004			ERROR	4	;CONTROLLER CLEAR DIDN'T CLEAR INDICATED REGISTER
1689	005146	012737	000014	001122	MOV	\$RPDA,\$BDADR ;REGISTER ADDRESS INDEX VALUE	
1690	005154	060437	001122		ADD	R4,\$BDADR ;FORM REGISTER ADDRESS	
1691	005160	013737	001234	001126	MOV	\$RPDA,\$BDDAT ;REGISTER VALUE FOR TYPEOUT	
1692	005166	042737	000360	001126	BIC	\$360,\$BDDAT ;CLEAR THE 'SOT' BITS	

```

1693 005174 023737 001124 001126    CMP    $GDDAT,$BDDAT    ;IS REGISTER RPDA CLEAR ?
1694 005202 001401                    BEQ    ,+4              ;BR IF IT IS
1695 005204 104004                    ERROR  4                ;CONTROLLER CLEAR DIDN'T CLEAR INDICATED REGISTER
1696 005206 012737 000200 001124    MOV    #200,$GDDAT      ;EXPECTED VALUE
1697 005214 013737 001224 001126    MOV    $RPCS,$BDDAT     ;REGISTER CONTENTS FOR TYPEOUT
1698 005222 023737 001124 001126    CMP    $GDDAT,$BDDAT   ;CONTENTS CORRECT ?
1699 005230 001401                    BEQ    85               ;BR IF CORRECT
1700 005232 104004                    ERROR  4                ;CONTROLLER CLEAR DIDN'T CLEAR INDICATED REGISTER
1701 005234 052764 020000 000022 85:    BIS    %BIT13,RPM3(R4) ;SET MAINTENANCE DRIVE ONLINE
1702 005242 000004                    SCOPE                    ;LOOP ?

```

```

; *THE FOLLOWING 6 TESTS TEST LOADING AND READING OF ALL POSSIBLE
; *BITS IN REGISTERS RPER, RPCS, RPWC, RPBA, RPCA AND
; *RPDA USING FLOATING 1'S AND FLOATING 0'S PATTERNS.

```

```

; *****
; *TEST 3      TEST BITS IN 'RPER'

```

```

; *****
; *TEST 3:

```

```

1713 005244                    ; *****
1714 005244 000240                    ; *TEST 3:
1715 005246 012737 005274 001106    MOV    #15,$LPADR      ;LOAD LOOP ON TEST ADDRESS
1716 005254 012737 005324 001110    MOV    #ERBT,$LPERR    ;LOAD ERROR LOOP ADDRESS
1717 005262 013704 001210                    MOV    RPADR,R4        ;RP11E BUS ADDRESS
1718 005266 012737 000012 001162    MOV    #10, $TIMES     ;DO 10. ITERATIONS
1719 005274                    ; *****
1720 005274 012706 001100                    ; *TEST 3:
1721 005300 004737 021714                    MOV    #STACK,SP      ;SETUP THE STACK POINTER
1722 005304 010437 001122                    JSR    PC,CLRP        ;CLEAR THE CONTROLLER
1723 005310 062737 000002 001122    MOV    R4,$BDADR      ;RP11 ADDRESS
1724 005316 012700 000001                    ADD    #RPER,$BDADR   ;FORM ADDRESS OF REGISTER BEING CHECKED
1725 005322 005001                    MOV    #1,R0          ;BEGINNING TEST PATTERN
1726 005324 010037 001124                    CLR    R1             ;FLOATING 1'S INDICATOR
1727 005330 042737 001121 001124    ERBT: MOV    R0,$GDDAT  ;REFERENCE PATTERN
1728 005336 013764 001124 000002    BIC    #176656,$GDDAT ;CLEAR UNUSED BITS
1729 005344 004737 022022                    MOV    $GDDAT,RPER(R4); PUT PATTERN IN RPER
1730 005350 013737 001222 001126    JSR    PC,SAVRP      ;SAVE THE RP11 REGISTERS
1731 005356 013737 001222 001160    MOV    $RPER,$BDDAT  ;VALUE FOR TYPEOUT IN CASE OF ERROR
1732 005364 042737 001121 001160    MOV    $RPER,$TMP0   ;WORKING LOCATION FOR CHECKING
1733 005372 023737 001124 001160    BIC    #176656,$TMP0 ;CLEAR UNUSED BITS
1734 005400 001403                    CMP    $GDDAT,$TMP0  ;REGISTER LOADED OK ?
1735 005402 104005                    BEQ    25             ;BR IF OK
1736 005404 005064 000002                    ERROR  5              ;FAILURE IN FLOATING 1'S & 0'S TEST
1737 005410 005100                    CLR    RPER(R4)      ;CLEAR THE REGISTER
1738 005412 005101                    COM    R0            ;COMPLIMENT TEST PATTERN
1739 005414 001343                    COM    R1            ;COMPLIMENT PASS INDICATOR
1740 005416 006300                    BNE    ERBT         ;BR IF JUST FINISHED FLOATING 1'S PASS
1741 005420 103341                    ASL    R0            ;SHIFT THE TEST PATTERN
1742 005422 005064 000002                    BCC    ERBT         ;BR IF MORE TO DO
1743 005426 000004                    CLR    RPER(R4)     ;CLEAR THE REGISTER
1744                    SCOPE                    ;LOOP ?

```

```

; *****
; *TEST 4      TEST BITS IN 'RPCS'

```

```

; *****

```

```

1744
1745
1746
1747
1748

```

```

1749 005430          TST4:
1750 005430 000240      NOP
1751 005432 012737 005460 001106      MOV      #15,SLPADR      ;LOAD LOOP ON TEST ADDRESS
1752 005440 012737 005510 001110      MOV      #CSBT,SLPERR  ;LOAD ERROR LOOP ADDRESS
1753 005446 013704 001210          MOV      RPADR,R4      ;RP11E BUS ADDRESS
1754 005452 012737 000012 001162      MOV      #10.,$TIMES  ;;DO 10. ITERATIONS
1755 005460          15:
1756 005460 012706 001100      MOV      #STACK,SP    ;SETUP THE STACK POINTER
1757 005464 004737 021714          JSR      PC,CLRP      ;CLEAR THE CONTROLLER
1758 005470 010437 001122          MOV      R4,$BDADR    ;RP11 ADDRESS
1759 005474 062737 000004 001122      ADD      #RPCS,$BDADR ;FORM ADDRESS OF REGISTER BEING CHECKED
1760 005502 012700 000001          MOV      #1,R0       ;BEGINNING TEST PATTERN
1761 005506 005001          CLR      R1          ;FLOATING 1'S INDICATOR
1762 005510 010037 001124          MOV      R0,$GDDAT   ;REFERENCE PATTERN
1763 005514 042737 140201 001124      BIC      #1C37576,$GDDAT ;CLEAR UNUSED BITS
1764 005522 013764 001124 000004      MOV      $GDDAT,RPCS(R4) ;PUT PATTERN IN RPCS
1765 005530 004737 022022          JSR      PC,SAVRP    ;SAVE THE RP11 REGISTERS
1766 005534 013737 001224 001126      MOV      SRPCS,$BDAT ;VALUE FOR TIMEOUT IN CASE OF ERROR
1767 005542 013737 001224 001160      MOV      SRPCS,$TMPD ;WORKING LOCATION FOR CHECKING
1768 005550 042737 140201 001160      BIC      #1C37576,$TMPD ;CLEAR UNUSED BITS
1769 005556 023737 001124 001160      CMP      $GDDAT,$TMPD ;REGISTER LOADED OK ?
1770 005564 001403          BEQ      25          ;BR IF OK
1771 005566 104005          ERROR      5        ;FAILURE IN FLOATING 1'S & 0'S TEST
1772 005570 005064 000004          CLR      RPCS(R4)   ;CLEAR THE REGISTER
1773 005574 005100          25:      COM      R0        ;COMPLIMENT TEST PATTERN
1774 005576 005101          COM      R1        ;COMPLIMENT PASS INDICATOR
1775 005600 001343          BNE      CSBT      ;BR IF JUST FINISHED FLOATING 1'S PASS
1776 005602 006300          ASL      R0        ;SHIFT THE TEST PATTERN
1777 005604 103341          BCC      CSBT      ;BR IF MORE TO DO
1778 005606 005064 000004          CLR      RPCS(R4)   ;CLEAR THE REGISTER
1779 005612 000004          SCOPE          ;LOOP ?

```

```

;*****
;#TEST 5      TEST BITS IN 'RPWC'
;*****

```

```

1780
1781
1782
1783
1784
1785 005614          15:
1786 005614 000240      NOP
1787 005616 012737 005644 001106      MOV      #15,SLPADR  ;LOAD LOOP ON TEST ADDRESS
1788 005624 012737 005674 001110      MOV      #WCBT,SLPERR ;LOAD ERROR LOOP ADDRESS
1789 005632 013704 001210          MOV      RPADR,R4    ;RP11E BUS ADDRESS
1790 005636 012737 000012 001162      MOV      #10.,$TIMES ;;DO 10. ITERATIONS
1791 005644          15:
1792 005644 012706 001100      MOV      #STACK,SP  ;SETUP THE STACK POINTER
1793 005650 004737 021714          JSR      PC,CLRP    ;CLEAR THE CONTROLLER
1794 005654 010437 001122          MOV      R4,$BDADR  ;RP11 ADDRESS
1795 005660 062737 000006 001122      ADD      #RPWC,$BDADR ;FORM ADDRESS OF REGISTER BEING CHECKED
1796 005666 012700 000001          MOV      #1,R0     ;BEGINNING TEST PATTERN
1797 005672 005001          CLR      R1        ;FLOATING 1'S INDICATOR
1798 005674 010037 001124          MOV      R0,$GDDAT ;REFERENCE PATTERN
1799 005700 042737 000000 001124      BIC      #1C177777,$GDDAT ;CLEAR UNUSED BITS
1800 005706 013764 001124 000006      MOV      $GDDAT,RPWC(R4) ;PUT PATTERN IN RPWC
1801 005714 004737 022022          JSR      PC,SAVRP  ;SAVE THE RP11 REGISTERS
1802 005720 013737 001226 001126      MOV      SRPWC,$BDAT ;VALUE FOR TIMEOUT IN CASE OF ERROR
1803 005726 013737 001226 001160      MOV      SRPWC,$TMPD ;WORKING LOCATION FOR CHECKING
1804 005734 042737 000000 001160      BIC      #1C177777,$TMPD ;CLEAR UNUSED BITS

```

1805	005742	023737	001124	001160	CMP	SGDDAT,STMPD	:REGISTER LOADED OK ?
1806	005750	001403			BEQ	ZS	:BR IF OK
1807	005752	104005			ERROR	5	:FAILURE IN FLOATING 1'S & 0'S TEST
1808	005754	005064	000006		CLR	RPWC(R4)	:CLEAR THE REGISTER
1809	005760	005100			COM	RO	:COMPLIMENT TEST PATTERN
1810	005762	005101			COM	R1	:COMPLIMENT PASS INDICATOR
1811	005764	001343			BNE	WCBT	:BR IF JUST FINISHED FLOATING 1'S PASS
1812	005766	006300			ASL	RO	:SHIFT THE TEST PATTERN
1813	005770	103341			BCC	WCBT	:BR IF MORE TO DO
1814	005772	005064	000006		CLR	RPWC(R4)	:CLEAR THE REGISTER
1815	005776	000004			SCOPE		:LOOP ?

\*\*\*\*\*  
; \*TEST 6 TEST BITS IN 'RPBA'

\*\*\*\*\*  
; \*TEST 6 TEST BITS IN 'RPBA'

1821	006000				†ST6:			
1822	006000	000240			NOP			
1823	006002	012737	006030	001106	MOV	#15,SLPADR	:LOAD LOOP ON TEST ADDRESS	
1824	006010	012737	006060	001110	MOV	#BAPT,SLPERR	:LOAD ERROR LOOP ADDRESS	
1825	006016	013704	001210		MOV	RPADR,R4	:RP11E BUS ADDRESS	
1826	006022	012737	000012	001162	MOV	#10.,\$TIMES	;;DO 10. ITERATIONS	
1827	006030				15:			
1828	006030	012706	001100		MOV	#STACK,SP	:SETUP THE STACK POINTER	
1829	006034	004737	021714		JSR	PC,CLRP	:CLEAR THE CONTROLLER	
1830	006040	010437	001122		MOV	R4,\$BDADR	:RP11 ADDRESS	
1831	006044	062737	000010	001122	ADD	#RPBA,\$BDADR	:FORM ADDRESS OF REGISTER BEING CHECKED	
1832	006052	012700	000001		MOV	#1,RO	:BEGINNING TEST PATTERN	
1833	006056	005001			CLR	R1	:FLOATING 1'S INDICATOR	
1834	006060	010037	001124		BAPT:	MOV	RO,SGDDAT	:REFERENCE PATTERN
1835	006064	042737	000000	001124	BIC	#1C177777,SGDDAT	:CLEAR UNUSED BITS	
1836	006072	013764	001124	000010	MOV	SGDDAT,RPBA(R4)	:PUT PATTERN IN RPBA	
1837	006100	004737	022022		JSR	PC,SAVRP	:SAVE THE RP11 REGISTERS	
1838	006104	013737	001230	001126	MOV	\$RPBA,\$BDADR	:VALUE FOR TYPEOUT IN CASE OF ERROR	
1839	006112	013737	001230	001160	MOV	\$RPBA,STMPD	:WORKING LOCATION FOR CHECKING	
1840	006120	042737	000000	001160	BIC	#1C177777,STMPD	:CLEAR UNUSED BITS	
1841	006126	023737	001124	001160	CMP	SGDDAT,STMPD	:REGISTER LOADED OK ?	
1842	006134	001403			BEQ	ZS	:BR IF OK	
1843	006136	104005			ERROR	5	:FAILURE IN FLOATING 1'S & 0'S TEST	
1844	006140	005064	000010		CLR	RPBA(R4)	:CLEAR THE REGISTER	
1845	006144	005100			COM	RO	:COMPLIMENT TEST PATTERN	
1846	006146	005101			COM	R1	:COMPLIMENT PASS INDICATOR	
1847	006150	001343			BNE	BAPT	:BR IF JUST FINISHED FLOATING 1'S PASS	
1848	006152	006300			ASL	RO	:SHIFT THE TEST PATTERN	
1849	006154	103341			BCC	BAPT	:BR IF MORE TO DO	
1850	006156	005064	000010		CLR	RPBA(R4)	:CLEAR THE REGISTER	
1851	006162	000004			SCOPE		:LOOP ?	

\*\*\*\*\*  
; \*TEST 7 TEST BITS IN 'RPCA'

\*\*\*\*\*  
; \*TEST 7 TEST BITS IN 'RPCA'

1852					†ST7:		
1853					NOP		
1854					MOV	#15,SLPADR	:LOAD LOOP ON TEST ADDRESS
1855					MOV	#CABT,SLPERR	:LOAD ERROR LOOP ADDRESS
1856							
1857	006164						
1858	006164	000240					
1859	006166	012737	006214	001106			
1860	006174	012737	006244	001110			



```

1861 006202 013704 001210      MOV      RPADR,R4      ;RP11E BUS ADDRESS
1862 006206 012737 000012 001162  MOV      #10.,$TIMES ;;DO 10. ITERATIONS
1863 006214      1S:
1864 006214 012706 001100      MOV      #STACK,SP    ;SETUP THE STACK POINTER
1865 006220 004737 021714      JSR      PC,CLRP      ;CLEAR THE CONTROLLER
1866 006224 010437 001122      MOV      R4,$BDADR    ;RP11 ADDRESS
1867 006230 062737 000012 001122  ADD      #RPCA,$BDADR  ;FORM ADDRESS OF REGISTER BEING CHECKED
1868 006236 012700 000001      MOV      #1,R0        ;BEGINNING TEST PATTERN
1869 006242 005001      CLR      R1           ;FLOATING 1'S INDICATOR
1870 006244 010037 001124      MOV      R0,$GDDAT    ;REFERENCE PATTERN
1871 006250 042737 177000 001124  BIC      #1C777,$GDDAT ;CLEAR UNUSED BITS
1872 006256 013764 001124 000012  MOV      $GDDAT,RPCA(R4) ;PUT PATTERN IN RPCA
1873 006264 004737 022022      JSR      PC,SAVRP     ;SAVE THE RP11 REGISTERS
1874 006270 013737 001232 001126  MOV      $RPCA,$BDAT  ;VALUE FOR TYPEOUT IN CASE OF ERROR
1875 006276 013737 001232 001160  MOV      $RPCA,$TMPD  ;WORKING LOCATION FOR CHECKING
1876 006304 042737 177000 001160  BIC      #1C777,$TMPD ;CLEAR UNUSED BITS
1877 006312 023737 001124 001160  CMP      $GDDAT,$TMPD ;REGISTER LOADED OK ?
1878 006320 001403      BEQ     2S           ;BR IF OK
1879 006322 104005      ERROR    5          ;FAILURE IN FLOATING 1'S & 0'S TEST
1880 006324 005064 000012      CLR      RPCA(R4)    ;CLEAR THE REGISTER
1881 006330 005100      2S:  COM      R0      ;COMPLIMENT TEST PATTERN
1882 006332 005101      COM      R1          ;COMPLIMENT PASS INDICATOR
1883 006334 001343      BNE     CABT        ;BR IF JUST FINISHED FLOATING 1'S PASS
1884 006336 006300      ASL     R0          ;SHIFT THE TEST PATTERN
1885 006340 103341      BCC     CABT        ;BR IF MORE TO DO
1886 006342 005064 000012      CLR      RPCA(R4)    ;CLEAR THE REGISTER
1887 006346 000004      SCOPE   ;LOOP ?

```

\*\*\*\*\*  
;TEST 10 TEST BITS IN 'RPDA'

\*\*\*\*\*  
;TST10:

```

1893 006350      1S:
1894 006350 000240      NOP
1895 006352 012737 006400 001106  MOV      #15,$LPADR   ;LOAD LOOP ON TEST ADDRESS
1896 006360 012737 006430 001110  MOV      #DABT,$LPERR ;LOAD ERROR LOOP ADDRESS
1897 006366 013704 001210      MOV      RPADR,R4     ;RP11E BUS ADDRESS
1898 006372 012737 000012 001162  MOV      #10.,$TIMES ;;DO 10. ITERATIONS
1899 006400      1S:
1900 006400 012706 001100      MOV      #STACK,SP    ;SETUP THE STACK POINTER
1901 006404 004737 021714      JSR      PC,CLRP      ;CLEAR THE CONTROLLER
1902 006410 010437 001122      MOV      R4,$BDADR    ;RP11 ADDRESS
1903 006414 062737 000014 001122  ADD      #RPDA,$BDADR ;FORM ADDRESS OF REGISTER BEING CHECKED
1904 006422 012700 000001      MOV      #1,R0        ;BEGINNING TEST PATTERN
1905 006426 005001      CLR      R1           ;FLOATING 1'S INDICATOR
1906 006430 010037 001124      MOV      R0,$GDDAT    ;REFERENCE PATTERN
1907 006434 042737 160360 001124  BIC      #1C17417,$GDDAT ;CLEAR UNUSED BITS
1908 006442 013764 001124 000014  MOV      $GDDAT,RPDA(R4) ;PUT PATTERN IN RPDA
1909 006450 004737 022022      JSR      PC,SAVRP     ;SAVE THE RP11 REGISTERS
1910 006454 013737 001234 001126  MOV      $RPDA,$BDAT  ;VALUE FOR TYPEOUT IN CASE OF ERROR
1911 006462 013737 001234 001160  MOV      $RPDA,$TMPD  ;WORKING LOCATION FOR CHECKING
1912 006470 042737 160360 001160  BIC      #1C17417,$TMPD ;CLEAR UNUSED BITS
1913 006476 023737 001124 001160  CMP      $GDDAT,$TMPD ;REGISTER LOADED OK ?
1914 006504 001403      BEQ     2S           ;BR IF OK
1915 006506 104005      ERROR    5          ;FAILURE IN FLOATING 1'S & 0'S TEST
1916 006510 005064 000014      CLR      RPDA(R4)    ;CLEAR THE REGISTER

```

```

1917 006514 005100 25: COM R0 ;COMPLIMENT TEST PATTERN
1918 006516 005101 COM R1 ;COMPLIMENT PASS INDICATOR
1919 006520 001343 BNE DABT ;BR IF JUST FINISHED FLOATING 1'S PASS
1920 006522 006300 ASL R0 ;SHIFT THE TEST PATTERN
1921 006524 103341 BCC DABT ;BR IF MORE TO DO
1922 006526 005064 000014 CLR RPDA(R4) ;CLEAR THE REGISTER
1923 006532 000004 SCOPE ;LOOP ?

```

```

;*****
;*TEST 11 'RPWC' RAPID ACCESS TEST

```

```

1928 ;*****
1929 †ST11:
1930 006534 000240 NOP
1931 006536 012737 006564 001106 MOV #15,SLPADR ;LOAD LOOP ON TEST ADDRESS
1932 006544 012737 006622 001110 MOV #RAPWC,SLPERR ;LOAD ERROR LOOP ADDRESS
1933 006552 013704 001210 MOV RPADR,R4 ;RP11E BUS ADDRESS
1934 006556 012737 000002 001162 MOV #2,STIMES ;DO 2 ITERATIONS
1935 006564 012706 001100 15: MOV #STACK,SP ;SETUP THE STACK POINTER
1936 006570 012737 000006 001122 MOV #RPWC,$BDADR ;INDEX OF REGISTER TESTED
1937 006576 060437 001122 ADD R4,$BDADR ;ADD THE BUS ADDRESS
1938 006602 013701 001122 MOV $BDADR,R1 ;COPY REGISTER ADDRESS
1939 006606 012737 006660 000004 MOV #RAPWC1,$ERRVEC ;SETUP FOR END MEMORY TRAP
1940 006614 005000 CLR R0 ;SETUP START MEMORY LOCATION
1941 006616 004737 021714 JSR PC,CLRP ;CLEAR THE RP11E
1942 006622 011011 RAPWC: MOV (R0),(R1) ;MOVE MEMORY TO REGISTER
1943 006624 011102 MOV (R1),R2 ;READ THE REGISTER
1944 006626 020220 CMP R2,(R0)+ ;TEST FOR PROPER VALUE IN REGISTER
1945 006630 001774 BEQ RAPWC ;BR IF CORRECT
1946 006632 014037 001124 MOV -(R0),SGDDAT ;MOVE GOOD DATA TO SGDDAT FOR TYP0UT
1947 006636 010237 001126 MOV R2,$BDDAT ;MOVE REGISTER DATA TO $BDDAT FOR TYP0UT
1948 006642 104006 ERROR 6 ;REGISTER CONTENTS INCORRECT
1949 006644 032777 000200 172266 BIT #SW07,$SWR ;PRINT ANY MORE ERRORS ?
1950 006652 001002 BNE RAPWC1 ;BR IF NOT
1951 006654 005720 TST (R0)+ ;CONTINUE TEST
1952 006656 000761 BR RAPWC
1953 006660 012737 000006 000004 RAPWC1: MOV #ERRVEC+2,$ERRVEC ;RESTORE TRAP CATCHER
1954 006666 012737 000340 177776 MOV #PR7,$PSW ;RESTORE NO INTERUPTS
1955 006674 005000 CLR R0 ;CLEAR THE REGISTER
1956 006676 000004 SCOPE ;LOOP ?

```

```

;*****
;*TEST 12 'RPBA' RAPID ACCESS TEST

```

```

1957 ;*****
1958 †ST12:
1959 006700 000240 NOP
1960 006700 012737 006730 001106 MOV #15,SLPADR ;LOAD LOOP ON TEST ADDRESS
1961 006702 012737 006766 001110 MOV #RAPBA,SLPERR ;LOAD ERROR LOOP ADDRESS
1962 006704 013704 001210 MOV RPADR,R4 ;RP11E BUS ADDRESS
1963 006706 012737 000002 001162 MOV #2,STIMES ;DO 2 ITERATIONS
1964 006708 012706 001100 15: MOV #STACK,SP ;SETUP THE STACK POINTER
1965 006710 012737 000006 001122 MOV #RPBA,$BDADR ;INDEX OF REGISTER TESTED
1966 006712 060437 001122 ADD R4,$BDADR ;ADD THE BUS ADDRESS
1967 006714 013701 001122 MOV $BDADR,R1 ;COPY REGISTER ADDRESS
1968 006716 012737 007024 000004 MOV #RAPBA1,$ERRVEC ;SETUP FOR END MEMORY TRAP

```

```

1973 006760 005000          CLR      R0          ;SETUP START MEMORY LOCATION
1974 006762 004737 021714  JSR      PC,CLRP    ;CLEAR THE RP11E
1975 006766 011011          MOV      (R0),(R1)  ;MOVE MEMORY TO REGISTER
1976 006770 011102          MOV      (R1),R2   ;READ THE REGISTER
1977 006772 020220          CMP      R2,(R0)+  ;TEST FOR PROPER VALUE IN REGISTER
1978 006774 001774          BEQ      RAPBA     ;BR IF CORRECT
1979 006776 014037 001124  MOV      -(R0),SGDDAT ;MOVE GOOD DATA TO SGDDAT FOR TYP0UT
1980 007002 010237 001126  MOV      R2,$BDDAT  ;MOVE REGISTER DATA TO $BDDAT FOR TYP0UT
1981 007006 104006          ERROR   6          ;REGISTER CONTENTS INCORRECT
1982 007010 032777 000200 172122 BIT      #SW07,$SWR  ;PRINT ANY MORE ERRORS ?
1983 007016 001002          BNE     RAPBA1    ;BR IF NOT
1984 007020 005720          TST     (R0)+     ;CONTINUE TEST
1985 007022 000761          BR      RAPBA
1986 007024 012737 000006 000004  RAPBA1: MOV     #ERRVEC+2,$ERRVEC ;RESTORE TRAP CATCHER
1987 007032 012737 000340 177776  MOV     #PR7,$PSW   ;RESTORE NO INTERRUPTS
1988 007040 005000          CLR     R0        ;CLEAR THE REGISTER
1989 007042 000004          SCOPE  ;LOOP ?

```

```

1990
1991 ;:*****
1992 ;*TEST 13      TEST SETTING 'SURDY' (SELECTED UNIT READY)
1993 ;:*****

```

```

1994 ;:*****
1995 TST13:

```

```

1996 007044 000240          NOP
1997 007046 012737 007066 001106  MOV     #DSF1,$LPADR ;LOAD LOOP ON TEST ADDRESS
1998 007054 012737 007066 001110  MOV     #DSF1,$LPERR ;LOAD ERROR LOOP ADDRESS
1999 007062 013704 001210          MOV     RPADR,R4    ;RP11E BUS ADDRESS
2000 007066 012706 001100  DSF1:  MOV     #STACK,SP  ;SETUP THE STACK POINTER
2001 007072 004737 021714          JSR     PC,CLRP    ;CLEAR THE CONTROLLER
2002 007076 052764 040000 000022  BIS     #BIT14,RPM3(R4) ;SET MAINT UNIT READY
2003 007104 004737 022022          JSR     PC,SAVRP   ;SAVE THE RP11E REGISTERS
2004 007110 032737 100000 001220  BIT     #BIT15,$RPDS ;DID SELECTED UNIT READY SET?
2005 007116 001002          BNE     1$        ;BRANCH IF SET
2006 007120 104007          ERROR   7        ;SELECTED UNIT READY NOT SET
2007 007122 000412          BR      2$        ;BYPASS REST OF CHECKS
2008 007124 042764 040000 000022  1$:    BIC     #BIT14,RPM3(R4) ;CLEAR MAINT UNIT READY
2009 007132 004737 022022          JSR     PC,SAVRP   ;SAVE THE RP11E REGISTERS
2010 007136 032737 100000 001220  BIT     #BIT15,$RPDS ;DID SELECTED UNIT READY CLEAR?
2011 007144 001401          BEQ     2$        ;DID SELECTED UNIT READY CLEAR?
2012 007146 104010          ERROR   10       ;SELECTED UNIT READY DID NOT CLEAR
2013 007150 000004          SCOPE  ;LOOP ?
2014
2015

```

```

2016 ;:*****
2017 ;*TEST 14      TEST SETTING OF 'SUSI' (SELECTED UNIT SEEK INCOMPLETE)
2018 ;:*****

```

```

2019 ;:*****
2020 TST14:

```

```

2021 007152 000240          NOP
2022 007154 012737 007174 001106  MOV     #DSF4,$LPADR ;LOAD LOOP ON TEST ADDRESS
2023 007162 012737 007174 001110  MOV     #DSF4,$LPERR ;LOAD ERROR LOOP ADDRESS
2024 007170 013704 001210          MOV     RPADR,R4    ;RP11E BUS ADDRESS
2025 007174 012706 001100  DSF4:  MOV     #STACK,SP  ;SETUP THE STACK POINTER
2026 007200 004737 021714          JSR     PC,CLRP    ;CLEAR THE CONTROLLER
2027 007204 052764 002000 000022  BIS     #BIT10,RPM3(R4) ;SET MAINT. SEEK INCOMPLETE
2028 007212 004737 022022          JSR     PC,SAVRP   ;SAVE THE RP11E REGISTERS

```

E04

MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST  
DZRPWC.P11 22-JUL-77 14:54 T14

MACY11 30(1046) 22-JUL-77 15:10 PAGE 42  
TEST SETTING OF 'SUSI' (SELECTED UNIT SEEK INCOMPLETE)

SEQ 0042

```

2029 007216 032737 004000 001220 BIT #BIT11,SRPDS ;DID SEEK INCOMPLETE SET?
2030 007224 001002 BNE 1$
2031 007226 104011 ERROR 11 ;SEEK INCOMPLETE DID NOT SET
2032 007230 000426 BR 4$ ;BYPASS REST OF THE TEST
2033 007232 032737 000001 001222 1$: BIT #BIT00,SRPER ;DID DISK ERROR SET?
2034 007240 001001 BNE 2$ ;BR IF YES
2035 007242 104012 ERROR 12 ;DISK ERROR DID NOT SET AFTER SUSI
2036 007244 2$:
2037 007244 013746 001224 MOV SRPCS,-(SP) ;PUT CONTENTS OF RPCS ON THE STACK
2038 007250 005116 COM (SP) ;COMPLEMENT THE CONTENTS
2039 007252 032726 140000 BIT #BIT15!BIT14,(SP)+ ;CHECK 'ERR' AND 'HE' BITS
2040 007256 001401 BEQ 3$ ;BR IF BOTH SET
2041 007260 104013 ERROR 13 ;'ERR' OR 'HE' DIDN'T SET WITH 'SUSI'
2042 007262 042764 002000 000022 3$: BIC #BIT10,RPM3(R4) ;CLEAR 'SUSI'
2043 007270 004737 022022 JSR PC,SAVRP ;SAVE THE RP11E REGISTERS
2044 007274 032737 004000 001220 BIT #BIT11,SRPDS ;DID SEEK INCOMPLETE CLEAR?
2045 007302 001401 BEQ 4$ ;BRANCH IF CLEAR
2046 007304 104045 ERROR 4$ ;SEEK INCOMPLETE DID NOT CLEAR
2047 007306 000004 4$: SCOPE ;LOOP ?

```

\*\*\*\*\*  
;TEST 15 TEST SETTING OF 'SUSU' (SELECTED UNIT SEEK UNDERWAY)

```

2051 2052 *****
2053 007310 TST15:
2054 007310 000240 NOP
2055 007312 012737 007332 001106 MOV #DSF6,$LPADR ;LOAD LOOP ON TEST ADDRESS
2056 007320 012737 007332 001110 MOV #DSF6,$LPERR ;LOAD ERROR LOOP ADDRESS
2057 007326 013704 001210 MOV RPADR,R4 ;RP11E BUS ADDRESS
2058 007332 012706 001100 DSF6: MOV #STACK,SP ;SETUP THE STACK POINTER
2059 007336 004737 021714 JSR PC,CLRP ;CLEAR THE CONTROLLER
2060 007342 042764 040000 000022 BIC #BIT14,RPM3(R4) ;RESET MAINT READY
2061 007350 052764 020000 000022 BIS #BIT13,RPM3(R4) ;SET MAINT DRIVE ONLINE
2062 007356 004737 022022 JSR PC,SAVRP ;SAVE THE RP11E REGISTERS
2063 007362 032737 002000 001220 BIT #BIT10,SRPDS ;DID 'SUSU' SET ?
2064 007370 001002 BNE 1$
2065 007372 104104 ERROR 104 ;'SUSU' DIDN'T SET
2066 007374 000412 BR 2$ ;BYPASS REST OF THE TEST
2067 007376 052764 040000 000022 1$: BIS #BIT14,RPM3(R4) ;SET MAINT READY
2068 007404 004737 022022 JSR PC,SAVRP ;SAVE THE RP11E REGISTERS
2069 007410 032737 002000 001220 BIT #BIT10,SRPDS ;DID 'SUSU' RESET ?
2070 007416 001401 BEQ 2$ ;BR IF IT DID
2071 007420 104105 ERROR 105 ;'SUSU' DIDN'T CLEAR
2072 007422 000004 2$: SCOPE ;LOOP ?

```

\*\*\*\*\*  
;TEST 16 TEST SETTING OF 'SUFU' (SELECTED UNIT FILE UNSAFE)

```

2073 2074 *****
2075 2076
2077 2078 TST16:
2079 007424 000240 NOP
2080 007426 012737 007446 001106 MOV #DSF7,$LPADR ;LOAD LOOP ON TEST ADDRESS
2081 007434 012737 007446 001110 MOV #DSF7,$LPERR ;LOAD ERROR LOOP ADDRESS
2082 007442 013704 001210 MOV RPADR,R4 ;RP11E BUS ADDRESS
2083 007446 012706 001100 DSF7: MOV #STACK,SP ;SETUP THE STACK POINTER
2084 007452 004737 021714 JSR PC,CLRP ;CLEAR THE CONTROLLER

```

F04

MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST  
 DZRPWC.P11 22-JUL-77 14:54 T16

MACY11 30(1046) 22-JUL-77 15:10 PAGE 43  
 TEST SETTING OF 'SUFU' (SELECTED UNIT FILE UNSAFE)

SEQ 0043

2085	007456	052764	004000	000022		BIS	#BIT11,RPM3(R4)	;SET MAINT. FILE UNSAFE
2086	007464	004737	022022			JSR	PC,SAVRP	;SAVE THE RP11E REGISTERS
2087	007470	032737	001000	001220		BIT	#BIT09,SRPDS	;DID FILE UNSAFE SET?
2088	007476	001002				BNE	1\$	
2089	007500	104014				ERROR	14	;FILE UNSAFE DID NOT SET
2090	007502	000412				BR	2\$	;BYPASS REST OF THE TEST
2091	007504	042764	004000	000022	1\$:	BIC	#BIT11,RPM3(R4)	;CLEAR 'SUFU'
2092	007512	004737	022022			JSR	PC,SAVRP	;SAVE THE RP11E REGISTERS
2093	007516	032737	001000	001220		BIT	#BIT09,SRPDS	;IS FILE UNSAFE CLEAR?
2094	007524	001401				BEQ	2\$	
2095	007526	104015				ERROR	15	;FILE UNSAFE DID NOT CLEAR
2096	007530	000004			2\$:	SCOPE		;LOOP ?

\*\*\*\*\*  
 ;\*TEST 17 TEST 'SUMP' (SELECTED UNIT WRITE PROTECTED)

\*\*\*\*\*  
 †ST17:

2102	007532					NOP		
2103	007532	000240				MOV	#DSF11,SLPADR	;LOAD LOOP ON TEST ADDRESS
2104	007534	012737	007554	001106		MOV	#DSF11,SLPERR	;LOAD ERROR LOOP ADDRESS
2105	007542	012737	007554	001110		MOV	RPADR,R4	;RP11E BUS ADDRESS
2106	007550	013704	001210			MOV	#STACK,SP	;SETUP THE STACK POINTER
2107	007554	012706	001100		DSF11:	JSR	PC,CLRP	;CLEAR THE CONTROLLER
2108	007560	004737	021714			BIS	#BIT15,RPM3(R4)	;SET MAINT READ ONLY
2109	007564	052764	100000	000022		JSR	PC,SAVRP	;SAVE THE RP11E REGISTERS
2110	007572	004737	022022			BIT	#BIT08,SRPDS	;DID WRITE PROTECT SET?
2111	007576	032737	000400	001220		BNE	1\$	
2112	007604	001002				ERROR	16	;WRITE PROTECT DID NOT SET
2113	007606	104016				BR	2\$	;BYPASS REST OF THE TEST
2114	007610	000412				BIC	#BIT15,RPM3(R4)	;CLEAR MAINTENANCE READ ONLY
2115	007612	042764	100000	000022	1\$:	JSR	PC,SAVRP	;SAVE THE RP11E REGISTERS
2116	007620	004737	022022			BIT	#BIT08,SRPDS	;DID WRITE PROTECT CLEAR?
2117	007624	032737	000400	001220		BEQ	2\$	
2118	007632	001401				ERROR	17	;WRITE PROTECT DID NOT CLEAR
2119	007634	104017				SCOPE		;LOOP ?
2120	007636	000004			2\$:			

```

2121
2122 ;:*****
2123 ;*TEST 20 TEST SET AND CLEAR OF THE ATTENTION BITS
2124 ;:*****
2125 †ST20:
2126 007640 NOP
2127 007640 000240 MOV #1$, $LPADR ;LOAD LOOP ON TEST ADDRESS
2128 007642 012737 007670 001106 MOV #DSF13, $LPERR ;LOAD ERROR LOOP ADDRESS
2129 007650 012737 007702 001110 MOV RPADR, R4 ;RPIIE BUS ADDRESS
2130 007656 013704 001210 MOV #10, $TIMES ;DO 10 ITERATIONS
2131 007662 012737 000012 001162 1$: MOV #STACK, SP ;SETUP THE STACK POINTER
2132 007670 012706 001100 DSF13: MOV #1, $GDDAT ;INITIALIZE ATTENTION BIT PATTERN
2133 007674 012737 000001 001124 MOV #DSF13, $LPERR ;CHANGE LOOP ON ERROR ADDRESS
2134 007702 012737 007702 001110 JSR PC, CLR ;CLEAR THE CONTROLLER
2135 007710 004737 021714 MOV #377, RPDS(R4) ;CLEAR ATTENTION BITS
2136 007714 012764 000377 000000 MOV $GDDAT, RPM2(R4) ;SET MAINT ATTENTION BIT
2137 007722 013764 001124 000020 JSR PC, SAVR ;SAVE THE RPIIE REGISTERS
2138 007730 004737 022022 CMPB $GDDAT, $RPDS ;DID THE ATTN BIT SET IN RPDS?
2139 007734 123737 001124 001220 BEQ 2$ ;BRANCH IF OK
2140 007742 001404
2141 007744 113737 001220 001126 ERROR 20 ;ATTENTION BIT DID NOT SET
2142 007752 104020 2$: MOV #2$, $LPERR ;CHANGE LOOP ON ERROR ADDRESS
2143 007754 012737 007754 001110 MOV $GDDAT, RPDS(R4) ;CLEAR ATTENTION BIT
2144 007762 013764 001124 000000 JSR PC, SAVR ;SAVE THE RPIIE REGISTERS
2145 007770 004737 022022 TSTB $RPDS ;DID IT CLEAR?
2146 007774 105737 001220 BEQ 3$ ;BRANCH IF CLEAR
2147 010000 001401 ERROR 21 ;ATTENTION BIT DID NOT CLEAR
2148 010002 104021 3$: ASL $GDDAT ;ROTATE PATTERN
2149 010004 006337 001124 BIT #BIT08, $GDDAT ;END OF PATTERN?
2150 010010 032737 000400 001124 BEQ DSF13 ;BRANCH IF NO
2151 010016 001731 4$: MOV #4$, $LPERR ;CHANGE LOOP ON ERROR ADDRESS
2152 010020 012737 010020 001110 MOV #377, RPM2(R4) ;SET ATTENTION BITS
2153 010026 012764 000377 000020 RESET
2154 010034 000005 TSTB $RPDS ;DID RESET CLEAR ATTN BITS?
2155 010036 105737 001220 BEQ 5$
2156 010042 001401 ERROR 22 ;RESET DID NOT CLEAR ATTENTION BITS
2157 010044 104022 5$: CLR RPM2(R4)
2158 010046 005064 000020 MOV #377, RPM2(R4) ;SET ALL ATTENTION BITS
2159 010052 012764 000377 000020 CLR RPDS(R4) ;ISSUE CLEAR RPDS
2160 010060 005064 000000 JSR PC, SAVR ;SAVE THE RPIIE REGISTERS
2161 010064 004737 022022 CMPB #377, $RPDS ;DID ATTENTION BITS REMAIN SET?
2162 010070 122737 000377 001220 BEQ 6$ ;BRANCH IF YES
2163 010076 001410 CLR $GDDAT
2164 010100 005037 001124 CLR $BDDAT
2165 010104 005037 001126 MOVB $RPDS, $BDDAT ;GET BAD DATA
2166 010110 113737 001220 001126 ERROR 23 ;ATTENTION BITS CLEARED WITH A ZERO
2167 010116 104023 6$: SCOPE ;LOOP ?
2168 010120 000004
2169
2170 ;:*****
2171 ;*TEST 21 TEST THE 'WPV' BIT (WRITE PROTECT VIOLATION)
2172 ;:*****
2173 †ST21:
2174 010122 NOP
2175 010122 000240 MOV #ERF1, $LPADR ;LOAD LOOP ON TEST ADDRESS
2176 010124 012737 010144 001106

```

```

2177 010132 012737 010144 001110      MOV      #ERF1,$LPERR      ;LOAD ERROR LOOP ADDRESS
2178 010140 013704 001210      MOV      RPADR,R4         ;RP11E BUS ADDRESS
2179 010144 012706 001100      ERF1:  MOV      #STACK,SP   ;SETUP THE STACK POINTER
2180 010150 004737 021714      JSR      PC,CLRP         ;CLEAR THE CONTROLLER
2181 010154 052764 100000 000022  BIS      #BIT15,RPM3(R4)  ;SET MAINT READ ONLY
2182 010162 012764 177777 000006  MOV      #-1,RPWC(R4)    ;SET WORD COUNT
2183 010170 112764 000003 000004  MOVVB   #3,RPCS(R4)     ;ISSUE A WRITE
2184 010176 012737 160001 001206  MOV      #BIT15:BIT14:BIT13:BIT00,TPL
2185 010204 004737 021764      JSR      PC,T3P         ;GENERATE 3 CLOCK PULSES
2186 010210 004737 022022      JSR      PC,SAVRP       ;SAVE THE RP11E REGISTERS
2187 010214 032737 100000 001222  BIT      #BIT15,$RPER    ;DID WRITE VIOLATION SET?
2188 010222 001001      BNE     1$             ;
2189 010224 104024      ERROR   24             ;WRITE PROTECTION VIOLATION DID NOT SET
2190 010226      1$:
2191 010226 013746 001224      MOV      $RPCS,-(SP)     ;PUT CONTENTS OF RPCS ON THE STACK
2192 010232 005116      COM     (SP)           ;COMPLEMENT THE CONTENTS
2193 010234 032726 140000      BIT      #BIT15:BIT14,(SP)+ ;CHECK 'ERR' AND 'HE' BITS
2194 010240 001401      BEQ     2$             ;BR IF BOTH SET
2195 010242 104025      ERROR   25             ;'ERR' OR 'HE' NOT SET WITH 'WPV'
2196 010244 000004      2$:  SCOPE              ;LOOP ?
2197
2198
2199
2200
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

```

\*\*\*\*\*  
\*TEST 22 TEST THE 'FUV' BIT (FILE UNSAFE VIOLATION)

\*\*\*\*\*  
\*TST22:

```

2202 010246      NOP
2203 010246 000240      MOV      #ERF2,$LPADR    ;LOAD LOOP ON TEST ADDRESS
2204 010250 012737 010270 001106  MOV      #ERF2,$LPERR    ;LOAD ERROR LOOP ADDRESS
2205 010256 012737 010270 001110  MOV      RPADR,R4         ;RP11E BUS ADDRESS
2206 010264 013704 001210      MOV      #STACK,SP   ;SETUP THE STACK POINTER
2207 010270 012706 001100      ERF2:  MOV      #STACK,SP   ;SETUP THE STACK POINTER
2208 010274 004737 021714      JSR      PC,CLRP         ;CLEAR THE CONTROLLER
2209 010300 012764 177777 000006  MOV      #-1,RPWC(R4)    ;LOAD THE WORD COUNT
2210 010306 112764 000003 000004  MOVVB   #3,RPCS(R4)     ;ISSUE WRITE
2211 010314 012737 064001 001206  MOV      #BIT14:BIT13:BIT11:BIT00,TPL
2212 010322 004737 021764      JSR      PC,T3P         ;GENERATE 3 CLOCK PULSES
2213 010326 004737 022022      JSR      PC,SAVRP       ;SAVE THE RP11E REGISTERS
2214 010332 032737 040000 001222  BIT      #BIT14,$RPER    ;DID FILE UNSAFE VIOLATION SET?
2215 010340 001001      BNE     1$             ;
2216 010342 104026      ERROR   26             ;FILE UNSAFE VIOLATION DID NOT SET
2217 010344      1$:
2218 010344 013746 001224      MOV      $RPCS,-(SP)     ;PUT CONTENTS OF RPCS ON THE STACK
2219 010350 005116      COM     (SP)           ;COMPLEMENT THE CONTENTS
2220 010352 032726 140000      BIT      #BIT15:BIT14,(SP)+ ;CHECK 'ERR' AND 'HE' BITS
2221 010356 001401      BEQ     2$             ;BR IF BOTH SET
2222 010360 104027      ERROR   27             ;'ERR' OR 'HE' DID NOT SET WITH 'FUV'
2223 010362 000004      2$:  SCOPE              ;LOOP ?
2224
2225
2226
2227
2228
2229
2230
2231
2232

```

\*\*\*\*\*  
\*TEST 23 TEST 'NXC' BIT (NON-EXISTENT CYLINDER) WITH VALID ADDRESSES

\*\*\*\*\*  
\*TST23:

```

2229 010364      NOP
2230 010364 000240      MOV      #1,$LPADR      ;LOAD LOOP ON TEST ADDRESS
2231 010366 012737 010406 001106  MOV      #ERF3,$LPERR    ;LOAD ERROR LOOP ADDRESS
2232 010374 012737 010416 001110

```

```

2233 010402 013704 001210      MOV      RPADR,R4      ;RP11E BUS ADDRESS
2234 010406 012706 001100      1S:     MOV      #STACK,SP ;SETUP THE STACK POINTER
2235 010412 005037 001124      CLR      $GDDAT      ;START AT ADDRESS 0
2236 010416 004737 021714      ERF3:   JSR      PC,CLRP     ;CLEAR THE CONTROLLER
2237 010422 013764 001124 000012  MOV      $GDDAT,RP4(R4) ;LOAD CYLINDER ADDR
2238 010430 012737 177777 001226  MOV      #-1,$RPWC    ;LOAD WORD COUNT
2239 010436 112764 000003 000004  MOVVB   #3,$RPCS(R4)  ;ISSUE A WRITE
2240 010444 012737 060001 001206  MOV      #BIT14!BIT13!BIT00,TPL
2241 010452 004737 021764      JSR      PC,T3P      ;GENERATE 3 CLOCK PULSES
2242 010456 004737 022022      JSR      PC,$AVRP    ;SAVE THE RP11E REGISTERS
2243 010462 032737 020000 001222  BIT      #BIT13,$RPER ;DID NON-EXISTENT CYLINDER SET?
2244 010470 001401      BEQ     25           ;BR IF NOT
2245 010472 104030      ERROR   30          ;'NXC' SET ON VALID ADDRESS
2246 010474 005237 001124 2S:     INC      $GDDAT      ;UPDATE CYLINDER ADDRESS
2247 010500 023737 001176 001124  CMP     MAXCYL,$GDDAT ;IS ADDR STILL LEGAL?
2248 010506 103343      BHIS   ERF3        ;BRANCH IF YES
2249 010510 000004      SCOPE                    ;LOOP ?

```

```

;*****
;*TEST 24      TEST 'NXC' WITH INVALID ADDRESSES

```

```

;*****
†ST24:

```

```

2256 010512 000240      NOP
2257 010514 012737 010534 001106  MOV      #15,$LPADR  ;LOAD LOOP ON TEST ADDRESS
2258 010522 012737 010552 001110  MOV      #ERF4,$LPERR ;LOAD ERROR LOOP ADDRESS
2259 010530 013704 001210      MOV      RPADR,R4    ;RP11E BUS ADDRESS
2260 010534 012706 001100      1S:     MOV      #STACK,SP  ;SETUP THE STACK POINTER
2261 010540 013737 001176 001124  MOV      MAXCYL,$GDDAT ;MAXIMUM CYLINDER ADDRESS
2262 010546 005237 001124      INC      $GDDAT      ;INCREMENT BEYOND LIMIT
2263 010552 004737 021714      ERF4:   JSR      PC,CLRP     ;CLEAR THE CONTROLLER
2264 010556 013764 001124 000012  MOV      $GDDAT,RP4(R4) ;LOAD CYLINDER ADDR
2265 010564 012764 177777 000006  MOV      #-1,$RPCS(R4) ;LOAD WORD COUNT
2266 010572 112764 000003 000004  MOVVB   #3,$RPCS(R4)  ;ISSUE WRITE
2267 010600 012737 060001 001206  MOV      #BIT14!BIT13!BIT00,TPL
2268 010606 004737 021764      JSR      PC,T3P      ;GENERATE 3 CLOCK PULSES
2269 010612 004737 022022      JSR      PC,$AVRP    ;SAVE THE RP11E REGISTERS
2270 010616 032737 020000 001222  BIT      #BIT13,$RPER ;DID NON-EXISTENT CYL SET?
2271 010624 001002      BNE    25           ;'NXC' DID NOT SET
2272 010626 104031      ERROR   31          ;'NXC' DID NOT SET
2273 010630 000407      BR     3S           ;BYPASS REST OF THE TEST
2274 010632      2S:
2275 010632 013746 001224      MOV      $RPCS,-(SP)  ;PUT CONTENTS OF RPCS ON THE STACK
2276 010636 005116      COM     (SP)         ;COMPLEMENT THE CONTENTS
2277 010640 032726 140000      BIT     #BIT15!BIT14,(SP)+ ;CHECK 'ERR' AND 'HE' BITS
2278 010644 001401      BEQ     3S          ;BR IF BOTH SET
2279 010646 104032      ERROR   32          ;'ERR' OR 'HE' DIDN'T SET WITH 'NXC'
2280 010650 005237 001124 3S:     INC      $GDDAT      ;UPDATE CYLINDER ADDR
2281 010654 023737 001200 001124  CMP     MAXPAT,$GDDAT ;PATTERN EXCEEDED?
2282 010662 001333      BNE    ERF4        ;BR IF NOT
2283 010664 000004      SCOPE                    ;LOOP ?

```

```

;*****
;*TEST 25      TEST 'NXT' BIT (NON-EXISTENT TRACK) WITH VALID ADDRESSES

```

```

;*****

```



```

2289 010666          TST25:
2290 010666 000240      NOP
2291 010670 012737 010710 001106  MOV    #15,$LPADR      ;LOAD LOOP ON TEST ADDRESS
2292 010676 012737 010720 001110  MOV    #ERF5,$LPERR   ;LOAD ERROR LOOP ADDRESS
2293 010704 013704 001210          MOV    RPADR,R4       ;RP11E BUS ADDRESS
2294 010710 012706 001100 1S:    MOV    #STACK,SP     ;SETUP THE STACK POINTER
2295 010714 005037 001124          CLR    $GDDAT        ;STARTING TRACK ADDR OF 0
2296 010720 004737 021714  ERF5:  JSR    PC,CLRP       ;CLEAR THE CONTROLLER
2297 010724 113764 001124 000015  MOV    $GDDAT,RPDA+1(R4) ;LOAD TRACK ADDRESS
2298 010732 012764 177777 000006  MOV    #-1,$RPC(R4)   ;LOAD WORD COUNT
2299 010740 112764 000003 000004  MOV    #3,$PCS(R4)    ;WRITE
2300 010746 012737 060001 001206  MOV    #BIT14!BIT13!BIT00,TPL
2301 010754 004737 021764          JSR    PC,T3P        ;GENERATE 3 CLOCK PULSES
2302 010760 004737 022022          JSR    PC,SAVRP     ;SAVE THE RP11E REGISTERS
2303 010764 032737 010000 001222  BIT    #BIT12,$RPER   ;IS NXT SET?
2304 010772 001401          BEQ    25
2305 010774 104033          ERROR  33           ;'NXT' SET ON VALID ADDRESS
2306 010776 005237 001124 2S:    INC    $GDDAT        ;INCREMENT THE TRACK ADDRESS
2307 011002 022737 000023 001124  CMP    #19,$GDDAT    ;IS TRACK ADDRESS STILL VALID
2308 011010 103343          BHS   ERF5          ;BRANCH IF YES
2309 011012 000004          SCOPE ;LOOP ?
2310
2311
2312 ;*****
2313 ;*TEST 26      TEST 'NXT' BIT WITH INVALID ADDRESSES
2314 ;*****
2315 ;*****
2316 011014          TST26:
2317 011014 000240      NOP
2318 011016 012737 011036 001106  MOV    #15,$LPADR      ;LOAD LOOP ON TEST ADDRESS
2319 011024 012737 011050 001110  MOV    #ERF6,$LPERR   ;LOAD ERROR LOOP ADDRESS
2320 011032 013704 001210          MOV    RPADR,R4       ;RP11E BUS ADDRESS
2321 011036 012706 001100 1S:    MOV    #STACK,SP     ;SETUP THE STACK POINTER
2322 011042 012737 000024 001124  MOV    #20,$GDDAT    ;START WITH INVALID ADDRESS
2323 011050 004737 021714  ERF6:  JSR    PC,CLRP       ;CLEAR THE CONTROLLER
2324 011054 113764 001124 000015  MOV    $GDDAT,RPDA+1(R4) ;LOAD TRACK ADDR
2325 011062 012737 177777 001226  MOV    #-1,$RPC(R4)   ;WRITE
2326 011070 012764 000003 000004  MOV    #3,$PCS(R4)    ;WRITE
2327 011076 012737 060001 001206  MOV    #BIT14!BIT13!BIT00,TPL
2328 011104 004737 021764          JSR    PC,T3P        ;GENERATE 3 CLOCK PULSES
2329 011110 004737 022022          JSR    PC,SAVRP     ;SAVE THE RP11E REGISTERS
2330 011114 032737 010000 001222  BIT    #BIT12,$RPER   ;DID NXT SET?
2331 011122 001002          BNE   25
2332 011124 104034          ERROR  34           ;'NXT' DIDN'T SET WITH INVALID ADDRESS
2333 011126 000407          BR    35           ;BYPASS REST OF THE TEST
2334 011130          2S:
2335 011130 013746 001224          MOV    $RPCS,-(SP)   ;PUT CONTENTS OF RPCS ON THE STACK
2336 011134 005116          COM    (SP)         ;COMPLEMENT THE CONTENTS
2337 011136 032726 140000          BIT    #BIT15!BIT14,(SP)+ ;CHECK 'ERR' AND 'HE' BITS
2338 011142 001401          BEQ    35           ;BR IF BOTH SET
2339 011144 104035          ERROR  35           ;'ERR' OR 'HE' DIDN'T SET WITH 'NXT'
2340 011146 005237 001124 3S:    INC    $GDDAT        ;INCREMENT TRACK ADDR.
2341 011152 022737 000040 001124  CMP    #40,$GDDAT   ;END OF PATTERN?
2342 011160 001333          BNE   ERF6          ;BRANCH IF NOT
2343 011162 000004          SCOPE ;LOOP ?
2344

```

# K04

```

2345      ;:*****
2346      ;*TEST 27      TEST 'NXS' BIT (NON-EXISTENT SECTOR) WITH VALID ADDRESSES
2347      ;:*****
2348      ;:*****
2349      †ST27:
2350      011164      000240      NOP
2351      011164      012737      011206      001106      MOV      #15,SLPADR      ;LOAD LOOP ON TEST ADDRESS
2352      011174      012737      011216      001110      MOV      #ERF7,SLPERR      ;LOAD ERROR LOOP ADDRESS
2353      011202      013704      001210      MOV      RPADR,R4      ;RPIIE BUS ADDRESS
2354      011206      012706      001100      1$:      MOV      #STACK,SP      ;SETUP THE STACK POINTER
2355      011212      005037      001124      CLR      $GDDAT      ;STARTING SECTOR ADDRESS OF 0
2356      011216      004737      021714      ERF7:    JSR      PC,CLRP      ;CLEAR THE CONTROLLER
2357      011222      013764      001124      000014      MOV      $GDDAT,RPDA(R4) ;LOAD SECTOR ADDR
2358      011230      012764      177777      000006      MOV      #-1,RPWC(R4)      ;LOAD WORD COUNT
2359      011236      112764      000003      000004      MOV      #3,RPCS(R4)      ;WRITE
2360      011244      012737      060001      001206      MOV      #BIT14!BIT13!BIT00,TPL
2361      011252      004737      021764      JSR      PC,T3P      ;GENERATE 3 CLOCK PULSES
2362      011256      004737      022022      JSR      PC,SAVRP      ;SAVE THE RPIIE REGISTERS
2363      011262      032737      004000      001222      BIT      #BIT11,SRPER      ;DID NXS SET?
2364      011270      001401      BEQ      25
2365      011272      104036      ERROR    36      ;'NXS' SET ON VALID ADDRESS
2366      011274      005237      001124      2$:      INC      $GDDAT      ;UPDATE SECTOR ADDR
2367      011300      022737      000011      001124      CMP      #9,$GDDAT      ;IS ADDR STILL LEGAL?
2368      011306      103343      BHIS     ERF7      ;BRANCH IF YES
2369      011310      000004      SCOPE    ;LOOP ?
2370
2371      ;:*****
2372      ;*TEST 30      TEST 'NXS' BIT WITH INVALID ADDRESSES
2373      ;:*****
2374      ;:*****
2375      †ST30:
2376      011312      000240      NOP
2377      011314      012737      011334      001106      MOV      #15,SLPADR      ;LOAD LOOP ON TEST ADDRESS
2378      011322      012737      011346      001110      MOV      #ERF10,SLPERR      ;LOAD ERROR LOOP ADDRESS
2379      011330      013704      001210      MOV      RPADR,R4      ;RPIIE BUS ADDRESS
2380      011334      012706      001100      1$:      MOV      #STACK,SP      ;SETUP THE STACK POINTER
2381      011340      012737      000012      001124      MOV      #10,$GDDAT      ;START WITH MAXIMUM SECTOR ADDR + 1
2382      011346      004737      021714      ERF10:   JSR      PC,CLRP      ;CLEAR THE CONTROLLER
2383      011352      013764      001124      000014      MOV      $GDDAT,RPDA(R4) ;LOAD SECTOR ADDR
2384      011360      012764      177777      000006      MOV      #-1,RPWC(R4)      ;LOAD WORD COUNT
2385      011366      112764      000003      000004      MOV      #3,RPCS(R4)      ;WRITE
2386      011374      012737      060001      001206      MOV      #BIT14!BIT13!BIT00,TPL
2387      011402      004737      021764      JSR      PC,T3P      ;GENERATE 3 CLOCK PULSES
2388      011406      004737      022022      JSR      PC,SAVRP      ;SAVE THE RPIIE REGISTERS
2389      011412      032737      004000      001222      BIT      #BIT11,SRPER      ;DID NXS SET?
2390      011420      001002      BNE
2391      011422      104037      ERROR    37      ;'NXS' DIDN'T SET WITH INVALID ADDRESS
2392      011424      000407      BR      35
2393      011426      25:
2394      011426      013746      001224      MOV      SRPCS,-(SP)      ;PUT CONTENTS OF RPCS ON THE STACK
2395      011432      005116      COM      (SP)      ;COMPLEMENT THE CONTENTS
2396      011434      032726      140000      BIT      #BIT15!BIT14,(SP)+ ;CHECK 'ERR' AND 'HE' BITS
2397      011440      001401      BEQ      35      ;BR IF BOTH SET
2398      011442      104040      ERROR    40      ;'ERR' OR 'HE' DIDN'T SET WITH 'NXS'
2399      011444      005237      001124      3$:      INC      $GDDAT      ;UPDATE ADDRESS
2400      011450      022737      000020      001124      CMP      #20,$GDDAT      ;IS PATTERN EXHAUSTED?
    
```

```

2401 011456 001333          BNE   ERF10          ;BRANCH IF NOT
2402 011460 000004          SCOPE                ;LOOP ?
2403
2404          ;:*****
2405          ;*TEST 31      TEST SETTING OF 'PROG' BIT (PROGRAM ERROR)
2406          ;:*****
2407          ;*TEST 31:
2408 011462
2409 011462 000240          NOP
2410 011464 012737 011504 001106      MOV   #ERF11,SLPADR  ;LOAD LOOP ON TEST ADDRESS
2411 011472 012737 011504 001110      MOV   #ERF11,SLPERR  ;LOAD ERROR LOOP ADDRESS
2412 011500 013704 001210          MOV   RPADR,R4      ;RP11E BUS ADDRESS
2413 011504 012706 001100          ERF11: MOV  #STACK,SP   ;SETUP THE STACK POINTER
2414 011510 004737 021714          JSR   PC,CLRP       ;CLEAR RP11
2415 011514 042764 020000 000022      BIC   #BIT13,RPM3(R4) ;RESET MAINTENANCE 'SUOL'
2416 011522 012764 000003 000004      MOV   #3,RPCS(R4)   ;ISSUE WRITE
2417 011530 012737 060001 001206      MOV   #BIT14!BIT13!BIT00,TPL
2418 011536 004737 021764          JSR   PC,T3P        ;GENERATE 3 CLOCK PULSES
2419 011542 004737 022022          JSR   PC,SAVRP      ;SAVE THE RP11E REGISTERS
2420 011546 032737 002000 001222      BIT   #BIT10,SRPER  ;DID PROGRAM ERROR SET?
2421 011554 001001
2422 011556 104041          BNE   1$
2423 011560          ERROR 41          ;'PROG' DIDN'T SET
2424 011560 013746 001224          1$:  MOV   SRPCS,-(SP)  ;PUT CONTENTS OF RPCS ON THE STACK
2425 011564 005116          COM   (SP)          ;COMPLEMENT THE CONTENTS
2426 011566 032726 140000          BIT   #BIT15!BIT14,(SP)+ ;CHECK 'ERR' AND 'HE' BITS
2427 011572 001401          BEQ   2$
2428 011574 104042          ERROR 42          ;'ERR' OR 'HE' DIDN'T SET WITH 'PROG'
2429 011576 032764 020000 000022      2$:  BIT   #BIT13,RPM3(R4) ;SET MAINTENANCE 'SUOL'
2430 011604 000004          SCOPE                ;LOOP ?
2431
2432          ;:*****
2433          ;*TEST 32      TEST SETTING OF 'MODE' BIT (MODE ERROR)
2434          ;:*****
2435          ;*TEST 32:
2436 011606
2437 011606 000240          NOP
2438 011610 012737 011630 001106      MOV   #ERF12,SLPADR  ;LOAD LOOP ON TEST ADDRESS
2439 011616 012737 011630 001110      MOV   #ERF12,SLPERR  ;LOAD ERROR LOOP ADDRESS
2440 011624 013704 001210          MOV   RPADR,R4      ;RP11E BUS ADDRESS
2441 011630 012706 001100          ERF12: MOV  #STACK,SP   ;SETUP THE STACK POINTER
2442 011634 004737 021714          JSR   PC,CLRP       ;CLEAR THE CONTROLLER
2443 011640 012764 177777 000006      MOV   #-1,RPWC(R4)  ;LOAD WORD COUNT
2444 011646 012764 004003 000004      MOV   #4003,RPCS(R4) ;WRITE HEADER IN PDP11 MODE
2445 011654 012737 060001 001206      MOV   #BIT14!BIT13!BIT00,TPL
2446 011662 004737 021764          JSR   PC,T3P        ;GENERATE 3 CLOCK PULSES
2447 011666 004737 022022          JSR   PC,SAVRP      ;SAVE THE RP11E REGISTERS
2448 011672 032737 000400 001222      BIT   #BIT08,SRPER  ;DID MODE ERROR SET?
2449 011700 001001
2450 011702 104043          BNE   1$
2451 011704          ERROR 43          ;'MODE' ERROR DIDN'T SET
2452 011704
2453 011704 013746 001224          1$:  MOV   SRPCS,-(SP)  ;PUT CONTENTS OF RPCS ON THE STACK
2454 011710 005116          COM   (SP)          ;COMPLEMENT THE CONTENTS
2455 011712 032726 140000          BIT   #BIT15!BIT14,(SP)+ ;CHECK 'ERR' AND 'HE' BITS
2456 011716 001401          BEQ   2$

```

```

2457 011720 104044          ERROR 44          ;'ERR' OR 'HE' DIDN'T SET WITH 'MODE'
2458 011722 000004          2$: SCOPE          ;LOOP ?
2459
2460          ;:*****
2461          ;*TEST 33      TEST ATTENTION INTERRUPT
2462          ;:*****
2463          ;:*****
2464          TST33:
2465          NOP
2466 011724 000240          MOV #15,SLPADR      ;LOAD LOOP ON TEST ADDRESS
2467 011726 012737 011746 001106  MOV #CSF1,SLPERR    ;LOAD ERROR LOOP ADDRESS
2468 011734 012737 011760 001110  MOV RPADR,R4        ;RP11E BUS ADDRESS
2469 011742 013704 001210          MOV #STACK,SP       ;SETUP THE STACK POINTER
2470 011746 012706 001100          1$: MOV #1,$GDDAT      ;STARTING TEST PATTERN
2471 011752 012737 000001 001124  CSF1: JSR PC,CLRP      ;CLEAR THE CONTROLLER
2472 011760 004737 021714          MOV #PR7,RPVEC+2    ;INTERRUPT RETURN ADDRESS
2473 011772 012777 000340 167222  JSR PC,CLRP         ;CLEAR RP11
2474 012000 004737 021714          CLR #PSW            ;CLEAR PRIORITY LEVEL
2475 012004 005037 177776          BIS #BIT13,RPCS(R4) ;ENABLE ATTENTION INTERRUPT
2476 012010 052764 020000 000004  MOV $GDDAT,RPM2(R4) ;SET ATTENTION BIT AND
2477 012016 013764 001124 000020  NOP                ;WAIT FOR INTERRUPT
2478 012024 000240          NOP
2479 012026 000240          NOP
2480 012030 000240          NOP
2481 012032 012737 000340 177776  MOV #PR7,#PSW       ;LOCKOUT INTERRUPTS
2482 012040 004737 022022          JSR PC,SAVRP        ;SAVE THE RP11E REGISTERS
2483 012044 104046          ERROR 46           ;NO ATTENTION INTERRUPT
2484 012046 000407          BR 3$              ;BYPASS REST OF THE TEST
2485 012050 012706 001100          2$: MOV #STACK,SP     ;RESTORE STACK
2486 012054 032764 020000 000004  BIT #BIT13,RPCS(R4) ;DID 'AIE' CLEAR?
2487 012062 001401          BEQ 3$
2488 012064 104047          ERROR 47           ;'AIE' DIDN'T CLEAR WHEN INTERRUPT OCCURED
2489 012066 013764 001124 000000  3$: MOV $GDDAT,RPDS(R4) ;CLEAR ATTENTION BIT
2490 012074 006337 001124          ASL $GDDAT          ;SHIFT TEST PATTERN
2491 012100 032737 000400 001124  BIT #BIT08,$GDDAT   ;PATTERN EXCEEDED?
2492 012106 001724          BEQ CSF1           ;BRANCH IF NO
2493 012110 000004          SCOPE             ;LOOP ?
2494
2495          ;:*****
2496          ;*TEST 34      TEST NO ATTENTION INTERRUPT
2497          ;:*****
2498          ;:*****
2499          TST34:
2500 012112 000240          NOP
2501 012114 012737 012134 001106  MOV #15,SLPADR      ;LOAD LOOP ON TEST ADDRESS
2502 012122 012737 012146 001110  MOV #TSTNAT,SLPERR  ;LOAD ERROR LOOP ADDRESS
2503 012130 013704 001210          MOV RPADR,R4        ;RP11E BUS ADDRESS
2504 012134 012706 001100          1$: MOV #STACK,SP       ;SETUP THE STACK POINTER
2505 012140 012737 000001 001124  TSTNAT: MOV #1,$GDDAT    ;STARTING TEST PATTERN
2506 012146 004737 021714          JSR PC,CLRP         ;CLEAR THE CONTROLLER
2507 012152 012777 000340 167034  MOV #PR7,RPVEC+2    ;INTERRUPT RETURN ADDRESS
2508 012160 012777 012222 167024  MOV #15,RPVEC       ;CLEAR RP11
2509 012166 004737 021714          JSR PC,CLRP         ;SET 'AIE'
2510 012172 052764 020000 000004  BIS #BIT13,RPCS(R4) ;CLEAR PRIORITY LEVEL
2511 012200 005037 177776          CLR #PSW            ;WAIT FOR INTERRUPT
2512 012204 000240          NOP

```

```

2513 012206 000240      NOP
2514 012210 000240      NOP
2515 012212 012737 000340 177776  MOV    #PR7,2#PSW    ;LOCKOUT INTERRUPTS
2516 012220 000405      BR      25          ;OK, INTERRUPT DIDN'T OCCUR
2517 012222 012706 001100 1S:    MOV    #STACK,SP    ;RESTORE STACK
2518 012226 004737 022022      JSR    PC,SAVRP     ;SAVE THE RP11E REGISTERS
2519 012232 104107      ERROR  107         ;ATTENTION INTERRUPT OCCURED WITH NO ATTN BITS SET
2520 012234 000004      SCOPE              ;LOOP ?

```

```

;*****
;#TEST 35      TEST ATTENTION INTERRUPT WITH 2 ATTN BITS SET

```

```

;*****
†ST35:

```

```

2527 012236      NOP
2528 012236 000240      MOV    #TSTAT,$LPADR ;LOAD LOOP ON TEST ADDRESS
2529 012240 012737 012260 001106  MOV    #TSTAT,$LPERR ;LOAD ERROR LOOP ADDRESS
2530 012246 012737 012260 001110  MOV    RPADR,R4      ;RP11E BUS ADDRESS
2531 012254 013704 001210      MOV    #STACK,SP    ;SETUP THE STACK POINTER
2532 012260 012706 001100  TSTAT: MOV    #TSTAT,$LPERR ;CHANGE LOOP ON ERROR ADDRESS
2533 012264 012737 012260 001110  MOV    #PR7,2#PVEC+2
2534 012272 012777 000340 166714  MOV    #1$,2#PVEC
2535 012300 012777 012356 166704  MOV    PC,$LAP      ;CLEAR THE CONTROLLER
2536 012306 004737 021714      JSR    PC,$LAP      ;LOWER PROCESSOR PRIORITY
2537 012312 005037 177776      CLR    2#PSW        ;ENABLE ATTENTION INTERRUPT
2538 012316 052764 020000 000004  BIS    #BIT13,RPCS(R4) ;SET ATTENTION BITS
2539 012324 012764 000003 000020  MOV    #3,RPM2(R4)
2540 012332 000240      NOP
2541 012334 000240      NOP
2542 012336 000240      NOP
2543 012340 012737 000340 177776  MOV    #PR7,2#PSW    ;RAISE PROCESSOR PRIORITY
2544 012346 004737 022022      JSR    PC,SAVRP     ;SAVE THE RP11E REGISTERS
2545 012352 104050      ERROR  50          ;RP11E DID NOT INTERRUPT WITH ATTENTION
2546                                     ;BITS 0 AND 1 SET

```

```

2547 012354 000431      BR      25
2548 012356 012737 012356 001110 1S:    MOV    #1$, $LPERR  ;CHANGE LOOP ON ERROR ADDRESS
2549 012364 012706 001100      MOV    #STACK,SP    ;RESTORE STACK
2550 012370 012777 012440 166614  MOV    #2$,2#PVEC

```

```

2551 012376 005037 177776      CLR    2#PSW
2552 012402 052764 020000 000004  BIS    #BIT13,RPCS(R4) ;ENABLE ATTENTION INTERRUPT
2553 012410 012764 000001 000000  MOV    #BIT00,RPDS(R4) ;CLEAR ATTENTION BIT ZERO

```

```

2554 012416 000240      NOP
2555 012420 000240      NOP
2556 012422 000240      NOP
2557 012424 012737 000340 177776  MOV    #PR7,2#PSW    ;RAISE PROCESSOR PRIORITY
2558 012432 004737 022022      JSR    PC,SAVRP     ;SAVE THE RP11E REGISTERS
2559 012436 104051      ERROR  51          ;ATTENTION BIT 1 DID NOT INTERRUPT
2560 012440 000004      SCOPE              ;LOOP ?

```

```

;*****
;#TEST 36      TEST 'IDE' BIT (INTERRUPT ON DONE ENABLE)

```

```

;*****
†ST36:

```

```

2566 012442      NOP
2567 012442 000240      MOV    #CSF2,$LPADR ;LOAD LOOP ON TEST ADDRESS
2568 012444 012737 012464 001106

```

```

2569 012452 012737 012464 001110      MOV      #CSF2,$LPERR      ;LOAD ERROR LOOP ADDRESS
2570 012460 013704 001210      MOV      RPADR,R4        ;RP11E BUS ADDRESS
2571 012464 012706 001100      CSF2:  MOV      #STACK,SP  ;SETUP THE STACK POINTER
2572 012470 004737 021714      JSR      PC,CLRP        ;CLEAR THE CONTROLLER
2573 012474 012777 012546 166510      MOV      #1,$JRPVEC      ;RETURN VECTOR
2574 012502 012777 000340 166504      MOV      #PR7,$RPVEC+2
2575 012510 005037 177776      CLR      PSM            ;ALLOW INTERRUPTS
2576 012514 052764 000100 000004      BIS      #BIT06,RPCS(R4) ;ENABLE INTERRUPT ON READY
2577 012522 000240      NOP
2578 012524 000240      NOP
2579 012526 000240      NOP
2580 012530 012737 000340 177776      MOV      #PR7,$PSW      ;LOCKOUT INTERRUPTS
2581 012536 004737 022022      JSR      PC,SAVRP      ;SAVE THE RP11E REGISTERS
2582 012542 104052      ERROR   52            ;NO READY INTERRUPT
2583 012544 000407      BR      25            ;BYPASS 'IDE' CHECK
2584 012546 004737 022022 15:      JSR      PC,SAVRP      ;SAVE THE REGISTERS
2585 012552 032737 000100 001224      BIT      #BIT06,$RPCS   ;IS 'IDE' STILL SET ?
2586 012560 001001      BNE     25            ;BR IF IT IS
2587 012562 104106      ERROR   106          ;'IDE' NOT SET AFTER INTERRUPT
2588 012564 000004      25:      SCOPE              ;LOOP ?
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624

```

;:\*\*\*\*\*  
;:TEST 37 TEST INTERRUPT WITHOUT INTERRUPT ENABLE SET  
;:\*\*\*\*\*  
TST37:

```

2594 012566      NOP
2595 012566 000240      MOV      #CSF3,$LPADR  ;LOAD LOOP ON TEST ADDRESS
2596 012570 012737 012610 001106      MOV      #CSF3,$LPERR  ;LOAD ERROR LOOP ADDRESS
2597 012576 012737 012610 001110      MOV      RPADR,R4      ;RP11E BUS ADDRESS
2598 012604 013704 001210      CSF3:  MOV      #STACK,SP  ;SETUP THE STACK POINTER
2599 012610 012706 001100      JSR      PC,CLRP        ;CLEAR THE CONTROLLER
2600 012614 004737 021714      JSR      PC,CLRP        ;CLEAR THE CONTROLLER
2601 012620 012777 012656 166364      MOV      #1,$JRPVEC      ;INTERRUPT VECTOR
2602 012626 112764 000377 000020      MOVB    #377,RPM2(R4)   ;SET ATTENTION BITS
2603 012634 005037 177776      CLR      $PSW          ;ALLOW INTERRUPTS AND
2604 012640 000240      NOP                    ;WAIT AWHILE
2605 012642 000240      NOP
2606 012644 000240      NOP
2607 012646 012737 000340 177776      MOV      #PR7,$PSW      ;LOCKOUT INTERRUPTS
2608 012654 000403      BR      25
2609 012656 004737 022022 15:      JSR      PC,SAVRP      ;SAVE THE RP11E REGISTERS
2610 012662 104053      ERROR   53            ;READY INTERRUPT OCCURED WITHOUT 'IDE' SET
2611 012664 000004      25:      SCOPE              ;LOOP ?
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624

```

;:\*\*\*\*\*  
;:TEST 40 TEST INTERRUPT AT PRIORITY 4  
;:\*\*\*\*\*  
TST40:

```

2617 012666      NOP
2618 012666 000240      MOV      #CSF6,$LPADR  ;LOAD LOOP ON TEST ADDRESS
2619 012670 012737 012710 001106      MOV      #CSF6,$LPERR  ;LOAD ERROR LOOP ADDRESS
2620 012676 012737 012710 001110      MOV      RPADR,R4      ;RP11E BUS ADDRESS
2621 012704 013704 001210      CSF6:  MOV      #STACK,SP  ;SETUP THE STACK POINTER
2622 012710 012706 001100      JSR      PC,CLRP        ;CLEAR RP11E
2623 012714 004737 021714      JSR      PC,CLRP        ;CLEAR RP11E
2624 012720 012777 012764 166264      MOV      #1,$JRPVEC      ;SETUP INTERRUPT VECTOR

```

```

2625 012726 052764 000100 000004    BIS    #BIT06,RPCS(R4) ;ENABLE READY INTERRUPT
2626 012734 012737 000200 177776    MOV    #PR4,#PSW      ;LOWER PROCESSOR LEVEL
2627 012742 000240                    NOP
2628 012744 000240                    NOP
2629 012746 000240                    NOP
2630 012750 012737 000340 177776    MOV    #PR7,#PSW
2631 012756 004737 022022                    JSR    PC,SAVRP      ;SAVE THE RP11E REGISTERS
2632 012762 104054                    ERROR  54            ;NO READY INTERRUPT AT LEVEL 4
2633 012764 000004                    IS:    SCOPE        ;LOOP ?
2634
2635 ;:*****
2636 ;:TEST 41      TEST NO INTERRUPT AT PRIORITY 5
2637 ;:*****
2638 ;:*****
2639 ;:ST41:
2640 012766 000240                    NOP
2641 012770 012737 013010 001106    MOV    #CSF7,$LPADR  ;LOAD LOOP ON TEST ADDRESS
2642 012776 012737 013010 001110    MOV    #CSF7,$LPERR  ;LOAD ERROR LOOP ADDRESS
2643 013004 013704 001210                    MOV    RPADR,R4      ;RP11E BUS ADDRESS
2644 013010 012706 001100    CSF7:  MOV    #STACK,SP    ;SETUP THE STACK POINTER
2645 013014 004737 021714                    JSR    PC,CLRP      ;CLEAR RP11E
2646 013020 012777 013060 166164    MOV    #1$,$RPVEC   ;SETUP INTERRUPT VECTOR
2647 013026 052764 000100 000004    BIS    #BIT06,RPCS(R4) ;ENABLE INTERRUPT ON READY
2648 013034 012737 000240 177776    MOV    #PRS,#PSW    ;SET PRIORITY LEVEL TO 5
2649 013042 000240                    NOP
2650 013044 000240                    NOP
2651 013046 000240                    NOP
2652 013050 012737 000340 177776    MOV    #PR7,#PSW
2653 013056 000403                    BR     2$
2654 013060 004737 022022    IS:    JSR    PC,SAVRP ;SAVE THE RP11E REGISTERS
2655 013064 104055                    ERROR  55            ;INTERRUPT RECEIVED AT PRIORITY LEVEL 5
2656 013066 000004                    2$:    SCOPE        ;LOOP ?
2657
2658 ;:*****
2659 ;:TEST 42      TEST NO INTERRUPT AT PRIORITY 6
2660 ;:*****
2661 ;:*****
2662 ;:ST42:
2663 013070 000240                    NOP
2664 013072 012737 013112 001106    MOV    #CSF10,$LPADR ;LOAD LOOP ON TEST ADDRESS
2665 013100 012737 013112 001110    MOV    #CSF10,$LPERR ;LOAD ERROR LOOP ADDRESS
2666 013106 013704 001210                    MOV    RPADR,R4      ;RP11E BUS ADDRESS
2667 013112 012706 001100    CSF10: MOV    #STACK,SP    ;SETUP THE STACK POINTER
2668 013116 004737 021714                    JSR    PC,CLRP      ;CLEAR RP11E
2669 013122 012777 013162 166062    MOV    #1$,$RPVEC   ;TRAP VECTOR
2670 013130 052764 000100 000004    BIS    #BIT06,RPCS(R4) ;ENABLE READY INTERRUPT
2671 013136 012737 000300 177776    MOV    #PR6,#PSW    ;SET PRIORITY LEVEL TO 6
2672 013144 000240                    NOP
2673 013146 000240                    NOP
2674 013150 000240                    NOP
2675 013152 012737 000340 177776    MOV    #PR7,#PSW
2676 013160 000403                    BR     2$
2677 013162 004737 022022    IS:    JSR    PC,SAVRP ;SAVE THE RP11E REGISTERS
2678 013166 104056                    ERROR  56            ;INTERRUPT RECEIVED AT LEVEL 6
2679 013170 000004                    2$:    SCOPE        ;LOOP ?
2680

```

2681 ;:\*\*\*\*\*  
2682 ;\*TEST 43 TEST NO INTERRUPT AT PRIORITY 7  
2683 ;:\*\*\*\*\*  
2684 ;:\*\*\*\*\*

2685 013172 TST43:  
2686 013172 000240 NOP  
2687 013174 012737 013214 001106 MOV #CSF11,SLPADR ;LOAD LOOP ON TEST ADDRESS  
2688 013202 012737 013214 001110 MOV #CSF11,SLPERR ;LOAD ERROR LOOP ADDRESS  
2689 013210 013704 001210 MOV RPADR,R4 ;RP11E BUS ADDRESS  
2690 013214 012706 001100 CSF11: MOV #STACK,SP ;SETUP THE STACK POINTER  
2691 013220 004737 021714 JSR PC,CLRP ;CLEAR RP11E  
2692 013224 012777 013250 165760 MOV #1\$,JRPVEC ;SETUP VECTOR INTERRUPT  
2693 013232 052764 000100 000004 BIS #BIT06,RPCS(R4) ;ENABLE READY INTERRUPT  
2694 013240 000240 NOP  
2695 013242 000240 NOP  
2696 013244 000240 NOP  
2697 013246 000403 BR 2\$  
2698 013250 004737 022022 1\$: JSR PC,SAVRP ;SAVE THE RP11E REGISTERS  
2699 013254 104056 ERROR 56 ;INTERRUPT RECEIVED AT LEVEL 7  
2700 013256 000004 2\$: SCOPE ;LOOP ?

2701 ;:\*\*\*\*\*  
2702 ;\*TEST 44 TEST CLEAR AND SET OF 'RDY' (CONTROLLER READY)  
2703 ;:\*\*\*\*\*  
2704 ;:\*\*\*\*\*

2705 TST44:  
2706 013260  
2707 013260 000240 NOP  
2708 013262 012737 013302 001106 MOV #CSF12,SLPADR ;LOAD LOOP ON TEST ADDRESS  
2709 013270 012737 013302 001110 MOV #CSF12,SLPERR ;LOAD ERROR LOOP ADDRESS  
2710 013276 013704 001210 MOV RPADR,R4 ;RP11E BUS ADDRESS  
2711 013302 012706 001100 CSF12: MOV #STACK,SP ;SETUP THE STACK POINTER  
2712 013306 004737 021714 JSR PC,CLRP ;CLEAR RP11E  
2713 013312 112764 000003 000004 MOVB #3,RPCS(R4) ;ISSUE WRITE  
2714 013320 004737 022022 JSR PC,SAVRP ;SAVE THE RP11E REGISTERS  
2715 013324 105737 001224 TSTB \$RPCS ;IS READY SET?  
2716 013330 100001 BPL 1\$ ;BRANCH IF NO  
2717 013332 104101 ERROR 101 ;'READY' DID NOT CLEAR WITH 'GO'  
2718 013334 012737 060001 001206 1\$: MOV #BIT14!BIT13!BIT00,TPL  
2719 013342 004737 021764 JSR PC,T3P ;GENERATE 3 CLOCK PULSES  
2720 013346 004737 022022 JSR PC,SAVRP ;SAVE THE RP11E REGISTERS  
2721 013352 105737 001224 TSTB \$RPCS ;DID READY SET?  
2722 013356 100401 BMI 2\$  
2723 013360 104060 ERROR 60 ;READY DID NOT SET AT END OF OPERATION  
2724 013362 000004 2\$: SCOPE ;LOOP ?

2725 ;:\*\*\*\*\*  
2726 ;\*TEST 45 TEST 'SUCA' BIT PATH IN THE CONTROLLER  
2727 ;:\*\*\*\*\*  
2728 ;:\*\*\*\*\*

2729 TST45:  
2730 013364  
2731 013364 000240 NOP  
2732 013366 012737 013406 001106 MOV #1\$,SLPADR ;LOAD LOOP ON TEST ADDRESS  
2733 013374 012737 013416 001110 MOV #CAF1,SLPERR ;LOAD ERROR LOOP ADDRESS  
2734 013402 013704 001210 MOV RPADR,R4 ;RP11E BUS ADDRESS  
2735 013406 012706 001100 1\$: MOV #STACK,SP ;SETUP THE STACK POINTER  
2736 013412 005037 001124 CLR \$GDDAT ;START TEST PATTERN AT 0



E05

MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST  
DZRPWC.P11 22-JUL-77 14:54

MACY11 30(1046) 22-JUL-77 15:10 PAGE 55  
TEST 'SUCA' BIT PATH IN THE CONTROLLER

SEQ 0055

```

2737 013416 004737 021714          CAF1: JSR      PC,CLRP          ;CLEAR THE CONTROLLER
2738 013422 113764 001124 000021  MOVB   $GDDAT,RPM2+1(R4) ;LOAD MAINT CYLINDER ADDR
2739 013430 004737 022022          JSR      PC,SAVRP        ;SAVE THE RP11E REGISTERS
2740 013434 013737 001240 001126  MOV    $$SUCA,$BDDAT    ;GET DISK CYLINDER ADDR
2741 013442 023737 001124 001126  CMP    $GDDAT,$BDDAT    ;IS SUCA CORRECT?
2742 013450 001401          BEQ     2$              ;
2743 013452 104061          ERROR   61              ;SUCA INCORRECT
2744 013454 005237 001124          INC    $GDDAT           ;UPDATE CYLINDER ADDR.
2745 013460 032737 000400 001124 2$: BIT    #BIT08,$GDDAT    ;IS PATTERN EXCEEDED?
2746 013466 001753          BEQ     CAF1           ;BRANCH IF NO
2747 013470 000004          SCOPE                   ;LOOP ?
2748
2749 ;:*****
2750 ;*TEST 46      TEST THAT THE 'SOT' COUNTS CORRECTLY
2751 ;:*****
2752 ;:*****
2753 ;*TST46:
2754 013472 000240          NOP
2755 013474 012737 013530 001106  MOV    #15,$SLPADR      ;LOAD LOOP ON TEST ADDRESS
2756 013502 012737 013560 001110  MOV    #DAF1,$LPERR     ;LOAD ERROR LOOP ADDRESS
2757 013510 013704 001210          MOV    RPADR,R4        ;RP11E BUS ADDRESS
2758 013514 012764 020400 000022  MOV    #BIT13!BIT08,RPM3(R4) ;GENERATE A SECTOR PULSE TO CLEAR
2759 013522 012764 020000 000022  MOV    #BIT13,RPM3(R4)  ;INDEX SYNC FF (IF SET)
2760 013530 012706 001100          MOV    #STACK,SP      ;SETUP THE STACK POINTER
2761 013534 016437 000014 001124 1$: MOV    RPDA(R4),$GDDAT ;GET THE REGISTER
2762 013542 042737 177417 001124  BIC    #1C360,$GDDAT   ;LEAVE THE 'SOT' BITS
2763 013550 004737 021714          JSR    PC,CLRP        ;CLEAR THE CONTROLLER
2764 013554 012705 000024          MOV    #20,R5         ;ITERATION COUNTER
2765 013560 004737 022022          JSR    PC,SAVRP      ;SAVE THE RP11E REGISTERS
2766 013564 013737 001234 001126  DAF1: MOV    SRPDA,$BDDAT  ;GET SOT
2767 013572 013737 001234 001160  MOV    SRPDA,$TMPD    ;MOVE CONTENTS TO A WORKING LOCATION
2768 013600 042737 177417 001160  BIC    #1C360,$TMPD   ;CLEAR UNWANTED BITS
2769 013606 023737 001124 001160  CMP    $GDDAT,$TMPD   ;CONTENTS OF 'SOT' CORRECT ?
2770 013614 001401          BEQ     2$
2771 013616 104063          ERROR   63              ;CONTENTS OF SOT INCORRECT
2772 013620 005305          DEC    R5              ;DECREMENT THE ITERATION COUNTER
2773 013622 001420          BEQ     3$              ;BR WHEN FINISHED
2774 013624 012764 020400 000022  MOV    #BIT13!BIT08,RPM3(R4) ;GENERATE ONE SECTOR PULSE
2775 013632 012764 020000 000022  MOV    #BIT13,RPM3(R4)
2776 013640 062737 000020 001124  ADD    #20,$GDDAT     ;UPDATE TEST ADDR
2777 013646 022737 000360 001124  CMP    #360,$GDDAT    ;MAXIMUM VALUE ?
2778 013654 103341          BHS    DAF1           ;BRANCH IF NOT
2779 013656 005037 001124          CLR    $GDDAT         ;RESET TO ZERO
2780 013662 000736          BR     DAF1           ;CONTINUE
2781 013664 000004          SCOPE                   ;LOOP ?
2782
2783 ;:*****
2784 ;*TEST 47      TEST THAT 'INDEX' CLEARS THE 'SOT' COUNTER
2785 ;:*****
2786 ;:*****
2787 ;*TST47:
2788 013666 000240          NOP
2789 013670 012737 013710 001106  MOV    #DAF2,$SLPADR  ;LOAD LOOP ON TEST ADDRESS
2790 013676 012737 013710 001110  MOV    #DAF2,$LPERR   ;LOAD ERROR LOOP ADDRESS
2791 013704 013704 001210          MOV    RPADR,R4      ;RP11E BUS ADDRESS
2792 013710 012706 001100          DAF2: MOV    #STACK,SP   ;SETUP THE STACK POINTER

```

F05

MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST  
DZRPWC.P11 22-JUL-77 14:54 T47

MACY11 30(1046) 22-JUL-77 15:10 PAGE 56  
TEST THAT 'INDEX' CLEARS THE 'SOT' COUNTER

SEQ 0056

2793	013714	004737	021714		JSR	PC, CLRP	; CLEAR RP11E
2794	013720	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2795	013726	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2796	013734	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2797	013742	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2798	013750	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2799	013756	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2800	013764	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2801	013772	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2802	014000	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2803	014006	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2804	014014	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2805	014022	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2806	014030	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2807	014036	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2808	014044	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2809	014052	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2810	014060	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2811	014066	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2812	014074	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2813	014102	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2814	014110	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2815	014116	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2816	014124	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2817	014132	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2818	014140	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2819	014146	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2820	014154	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2821	014162	012764	020000	000022	MOV	#BIT13, RPM3(R4)	
2822	014170	004037	021774		JSR	RO, INDEXP	; GENERATE 1 INDEX PULSE
2823	014174	000001			.WORD	1	; CONSTANT FOR 1 INDEX PULSE
2824	014176	012764	020400	000022	MOV	#BIT13!BIT08, RPM3(R4)	; GENERATE A SECTOR PULSE
2825	014204	012764	020000	000022	MOV	#BIT13, RPM3(R4)	; TO CLEAR THE 'SOT'
2826	014212	004737	022022		JSR	PC, SAVRP	; STORE THE RP11E REGISTERS
2827	014216	013737	001234	001126	MOV	SRPDA, \$BDDAT	; STORE THE CONTENTS
2828	014224	013737	001234	001160	MOV	SRPDA, \$TMPO	; MOVE CONTENTS TO A WORKING LOCATION
2829	014232	042737	177417	001160	BIC	#C360, \$TMPO	; MASK OUT ALL BUT THE 'SOT' BITS
2830	014240	001401			BEQ	1\$	
2831	014242	104062			ERROR	62	; SOT DID NOT CLEAR WITH INDEX PULSE
2832	014244	000004			1\$: SCOPE		; LOOP ?
2833							
2834							
2835							
2836							
2837							
2838							
2839							
2840							
2841							
2842							
2843							
2844							
2845	014246						
2846	014246	000240			NOP		
2847	014250	012737	014270	001106	MOV	#SILOT, \$LPADR	; LOAD LOOP ON TEST ADDRESS
2848	014256	012737	014270	001110	MOV	#SILOT, \$LPERR	; LOAD ERROR LOOP ADDRESS

```

;*****
;#TEST 50      SILO TEST, PART 1
;
;#THIS TEST CHECKS THE SILO MEMORY IN MAINTENANCE
;#MODE.  IF INREADY IS SET, DATA IS OUTPUT TO REGISTER
;#SILO' WHICH IS THE SILO MEMORY.  AFTER THE DATA FILTERS
;#THRU THE MEMORY, OUTREADY GOES TRUE.  THE DATA IS READ
;#BACK AND COMPARED.

```

```

;*****
;#TST50:

```

```

2849 014264 013704 001210      MOV      RPADR,R4      ;RP11E BUS ADDRESS
2850 014270 012706 001100      MOV      #STACK,SP   ;SETUP THE STACK POINTER
2851 014274 012737 000001 001124  MOV      #1,$GDDAT    ;INITIALIZE FLOATING ONE PATTERN
2852 014302 005005                CLR      R5           ;PATTERN FLAG
2853 014304 012737 014304 001110 1S:      MOV      #1S,$LPERR   ;RESTORE THE LOOP ON ERROR ADDRESS
2854 014312 004737 021714        JSR      PC,CLR       ;CLEAR THE CONTROLLER
2855 014316 004737 022022        JSR      PC,SAVRP     ;SAVE THE RP11E REGISTERS
2856 014322 032737 004000 001236  BIT      #BIT11,$RPM1 ;IS INREADY SET?
2857 014330 001002                BNE      2S          ;BRANCH IF SET
2858 014332 104064                ERROR   64          ;SILO INREADY IS NOT SET AFTER CLEAR
2859 014334 000454                BR      9S          ;EXIT
2860 014336 012737 014336 001110 2S:      MOV      #2S,$LPERR   ;CHANGE LOOP ON ERROR ADDRESS
2861 014344 013764 001124 000026  MOV      $GDDAT,$SILO(R4) ;LOAD DATA IN SILO
2862 014352 012701 000060        MOV      #60,R1      ;WAIT FOR OUT READY TO SET
2863 014356 005301                DEC     R1
2864 014360 001376                BNE     3S
2865 014362 004737 022022        JSR     PC,SAVRP     ;SAVE THE RP11E REGISTERS
2866 014366 032737 010000 001236  BIT     #BIT12,$RPM1 ;IS OUTREADY SET?
2867 014374 001002                BNE     5S          ;BRANCH IF SET
2868 014376 104065                ERROR   65          ;SILO OUTREADY IS NOT SET
2869 014400 000432                BR      9S
2870 014402 016437 000026 001126 5S:      MOV     $SILO(R4),$SDDAT ;GET DATA BACK FROM SILO
2871 014410 023737 001124 001126  CMP     $GDDAT,$SDDAT ;IS DATA CORRECT?
2872 014416 001401                BEQ     6S
2873 014420 104066                ERROR   66
2874 014422 005705                TST     R5          ;DATA READ FROM SILO INCORRECT
2875 014424 001012                BNE     8S          ;ARE WE FLOATING A ONE?
2876 014426 006337 001124        ASL     $GDDAT       ;BRANCH IF ZERO
2877 014432 103401                BCS     7S          ;SHIFT PATTERN
2878 014434 000723                BR      1S          ;BRANCH IF PATTERN EXCEEDED
2879 014436 012705 000001        MOV     #1,R5        ;SET PATTERN FLAG
2880 014442 012737 077777 001124  MOV     #77777,$GDDAT ;FLOATING ZERO PATTERN
2881 014450 000715                BR      1S
2882 014452 006237 001124        ASR     $GDDAT       ;SHIFT FLOATING ZERO
2883 014456 052737 100000 001124  BIS     #BIT15,$GDDAT
2884 014464 103707                BCS     1S
2885 014466 000004        9S:     SCOPE        ;LOOP ?

```

```

2886
2887
2888 ;*****
2889 ;*TEST 51      SILO TEST, PART 2
2890
2891 ;*ENSURE THAT THE SILO MEMORY CAN HOLD 64 DISCREET NUMBERS
2892 ;*AT ONE TIME. AFTER LOADING THE 64 NUMBERS SIGNAL INREADY
2893 ;*SHOULD CLEAR INDICATING THE SILO IS FULL. THE SILO IS THEN
2894 ;*READ OUT EXPECTING SEQUENTIAL NUMBERS OF 1 THRU 100 OCTAL. AT
2895 ;*THIS TIME OUT READY SHOULD CLEAR.

```

```

2896
2897 ;*****
2898 ;TST51:
2899
2900 014470 000240                NOP
2901 014472 012737 014520 001106  MOV     #SILOT1,$LPADR ;LOAD LOOP ON TEST ADDRESS
2902 014500 012737 014520 001110  MOV     #SILOT1,$LPERR ;LOAD ERROR LOOP ADDRESS
2903 014506 013704 001210        MOV     RPADR,R4     ;RP11E BUS ADDRESS
2904 014512 012737 000012 001162  MOV     #10,$TIMES   ;DO 10. ITERATIONS
2905 014520 012706 001100  SILOT1: MOV     #STACK,SP   ;SETUP THE STACK POINTER

```

```

2905 014524 012737 000001 001124 MOV #1,$GDDAT ;INITIALIZE TEST PATTERN
2906 014532 012737 014532 001110 1$: MOV #1,$SLPERR ;RESTORE THE LOOP ON ERROR ADDRESS
2907 014540 004737 021714 JSR PC,CLRP ;CLEAR THE CONTROLLER
2908 014544 004737 022022 JSR PC,SAVRP ;SAVE THE RP11E REGISTERS
2909 014550 032737 004000 001236 BIT #BIT11,$RPM1 ;IS INREADY SET?
2910 014556 001001 BNE 2$ ;BRANCH IF SET
2911 014560 104064 ERROR 64 ;SILO INREADY SHOULD BE SET
2912 014562 012737 014562 001110 2$: MOV #2,$SLPERR ;CHANGE THE LOOP ON ERROR ADDRESS
2913 014570 012706 001100 MOV #STACK,SP ;SETUP THE STACK POINTER
2914 014574 013764 001124 000026 MOV $GDDAT,SILO(R4) ;LOAD PATTERN IN SILO
2915 014602 005237 001124 INC $GDDAT ;UPDATE PATTERN
2916 014606 022737 000101 001124 CMP #65.,$GDDAT ;IS THE SILO FULL?
2917 014614 001362 BNE 2$ ;BRANCH NOT FULL
2918 014616 016437 000016 001236 MOV RPM1(R4),$RPM1 ;SAVE THE REGISTER
2919 014624 032737 004000 001236 BIT #BIT11,$RPM1 ;DID INREADY CLEAR?
2920 014632 001401 BEQ 3$ ;BRANCH IF YES
2921 014634 104067 ERROR 67 ;SILO INREADY DID NOT CLEAR
2922 014636 012737 000001 001124 3$: MOV #1,$GDDAT ;RESET PATTERN
2923 014644 016437 000016 001236 4$: MOV RPM1(R4),$RPM1 ;SAVE THE REGISTER
2924 014652 032737 010000 001236 BIT #BIT12,$RPM1 ;IS OUTREADY SET?
2925 014660 001001 BNE 5$ ;BRANCH IF SET
2926 014662 104065 ERROR 65 ;SILO OUTREADY SHOULD BE SET
2927 014664 012737 015014 001164 5$: MOV #9$,$ESCAPE ;SETUP THE ESCAPE ADDRESS
2928 014672 016437 000026 001126 MOV SILO(R4),$BDDAT ;READ SILO MEMORY
2929 014700 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS DATA CORRECT?
2930 014706 001405 BEQ 6$ ;BRANCH IF EQUAL
2931 014710 104066 ERROR 66 ;INCORRECT DATA RECEIVED FROM SILO
2932 014712 032777 000200 164220 BIT #SW07,$SWR ;IS SWITCH 7 SET ?
2933 014720 001022 BNE 7$ ;BR IF IT IS - BYPASS REST OF TEST
2934 014722 012705 000400 6$: MOV #256.,R5 ;CONSTANT FOR STALL
2935 014726 005305 10$: DEC R5 ;STALL AND WAIT FOR 'INPUT READY' TO
2936 014730 001376 BNE 10$ ;SET (FOR THE FIRST TIME THROUGH THE LOOP)
2937 014732 016437 000016 001236 MOV RPM1(R4),$RPM1 ;SAVE THE MAINTENANCE REGISTER
2938 014740 005237 001124 INC $GDDAT ;UPDATE EXPECTED PATTERN
2939 014744 022737 000101 001124 CMP #65.,$GDDAT ;HAVE 64 WORDS BEEN READ
2940 014752 001413 BEQ 8$ ;BRANCH IF NO
2941 014754 032737 004000 001236 BIT #BIT11,$RPM1 ;IS INPUT READY SET ?
2942 014762 001001 BNE 7$ ;BR IF IT IS
2943 014764 104110 ERROR 110 ;SILO NOT FULL, INPUT READY SOULD BE SET
2944 014766 032737 010000 001236 7$: BIT #BIT12,$RPM1 ;SEE IF OUTPUT READY STILL SET
2945 014774 001333 BNE 5$ ;BR IF OUTPUT READY SET
2946 014776 104111 ERROR 111 ;WORDS STILL IN SILO, OUTPUT READY
2947 SHOULD BE SET
2948 BR 5$ ;CONTINUE
2949 015002 032737 010000 001236 8$: BIT #BIT12,$RPM1 ;HAS SILO OUTREADY CLEARED ?
2950 015010 001401 BEQ 9$ ;BR IF CLEAR
2951 015012 104070 ERROR 70 ;SILO SHOULD BE EMPTY BUT
2952 ;OUTREADY IS STILL SET
2953 015014 005037 001164 9$: CLR $ESCAPE ;CLEAR THE ESCAPE ADDRESS
2954 015020 000004 SCOPE ;LOOP ?

```

```

;*****
;*TEST 52 TEST 'SEEK' COMMAND BUS SIGNALS
;*****
;*TEST THE OPERATION OF A SEEK COMMAND IN MAINTENANCE MODE.

```

```

2955
2956
2957
2958
2959
2960

```

```

2961 ;*ISSUE A SEEK COMMAND AND CHECK THE SETTING OF SET CYLINDER
2962 ;*RESET HEAD, SET HEAD, SEEK START AND CAR BITS 0 THRU 7.
2963 ;*ALL THESE SIGNALS ARE FOUND ON BUS OUT CONTROL.
2964
2965 ;*****
2966 T52:
2967 015022 000240 NOP
2968 015024 012737 015044 001106 MOV #15, $LPADR ;LOAD LOOP ON TEST ADDRESS
2969 015032 012737 015052 001110 MOV #SEEK, $LPERR ;LOAD ERROR LOOP ADDRESS
2970 015040 013704 001210 MOV RPADR, R4 ;RP11E BUS ADDRESS
2971 015044 012706 001100 1$: MOV #STACK, SP ;SETUP THE STACK POINTER
2972 015050 005005 CLR R5 ;CLEAR PASS FLAG
2973 015052 004737 021714 SEEK: JSR PC, CLRP ;CLEAR THE CONTROLLER
2974 015056 004737 022022 JSR PC, SAVRP ;SAVE THE RP11 REGISTERS
2975 015062 013737 001236 001126 MOV $RPM1, $BDDAT ;GET CONTENTS OF RPM1
2976 015070 042737 174000 001126 BIC #174000, $BDDAT ;CLEAR UNWANTED BITS
2977 015076 001403 BEQ Z$ ;BRANCH IF RESULT IS ZERO
2978 015100 005037 001124 CLR $GDDAT
2979 015104 104071 ERROR 71 ;SOME BUS OUT SIGNALS TO THE
2980 ;DRIVE ARE SET AFTER RESET
2981 015106 012764 000252 000012 2$: MOV #170., RPCA(R4) ;LOAD CYCL 170 INTO RPCA
2982 015114 005705 TST R5
2983 015116 001403 BEQ 3$
2984 015120 012764 000125 000012 MOV #85, RPCA(R4) ;LOAD CYLINDER 85 INTO REGISTER
2985 015126 112764 000017 000015 3$: MOVVB #17, RPDA+1(R4) ;LOAD TRACK ADDR
2986 015134 005705 TST R5 ;IS THIS FIRST PASS
2987 015136 001403 BEQ 4$ ;BRANCH IF YES
2988 015140 112737 000020 001235 MOVVB #20, $RPDA+1 ;SET HIGH ORDER BIT OF TRACK ADDR
2989 015146 112764 000011 000004 4$: MOVVB #11, RPCS(R4) ;ISSUE SEEK COMMAND
2990 015154 012737 060001 001206 MOV #BIT14!BIT13!BIT00, TPL
2991 015162 004037 021744 JSR R0, TIMEP ;GENERATE 5 TIMING PULSES
2992 015166 000005 .WORD 5 ;CONSTANT FOR 5 TIMING PULSES
2993 015170 004737 022022 JSR PC, SAVRP ;GET THE REGISTERS
2994 015174 013737 001236 001126 MOV $RPM1, $BDDAT ;GET CONTROL LINES FROM RPM1
2995 015202 042737 174000 001126 BIC #174000, $BDDAT ;CLEAR UNWANTED BITS
2996 015210 012737 000525 001124 MOV #525, $GDDAT ;LOAD EXPECTED VALUE OF RPM1
2997 015216 005705 TST R5 ;IS THIS FIRST PASS
2998 015220 001403 BEQ 5$ ;BRANCH IF YES
2999 015222 012737 000652 001124 MOV #652, $GDDAT
3000 015230 023737 001124 001126 5$: CMP $GDDAT, $BDDAT ;WERE CONTENTS OF RPM1 CORRECT?
3001 015236 001401 BEQ 6$ ;BRANCH IF YES
3002 015240 104072 ERROR 72 ;THE CONTROL SIGNAL SET CYLINDER
3003 ;AND THE CONTENTS OF CAR SHOULD
3004 ;BE ON THE BUS OUT LINES
3005 015242 004037 021744 6$: JSR R0, TIMEP ;GENERATE 4 CLOCK PULSES
3006 015246 000004 .WORD 4 ;CONSTANT FOR 4 CLOCK PULSES
3007 015250 004737 022022 JSR PC, SAVRP ;GET THE REGISTERS
3008 015254 013737 001236 001126 MOV $RPM1, $BDDAT ;GET CONTENTS OF RPM1
3009 015262 042737 174000 001126 BIC #174000, $BDDAT ;CLEAR UNWANTED BITS
3010 015270 012737 002010 001124 MOV #BIT10!BIT03, $GDDAT ;LOAD EXPECTED VALUE OF RPM1
3011 015276 023737 001124 001126 CMP $GDDAT, $BDDAT ;ARE CONTENTS OF RPM1 CORRECT?
3012 015304 001401 BEQ 7$ ;BRANCH IF YES
3013 015306 104073 ERROR 73 ;THE CONTROL SIGNAL RESET HEAD
3014 ;SHOULD BE ON THE BUS OUT LINES
3015 015310 004037 021744 7$: JSR R0, TIMEP ;GENERATE 4 CLOCK PULSES
3016 015314 000004 .WORD 4 ;CONSTANT FOR 4 CLOCK PULSES
    
```

J05

MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST  
DZRPWC.P11 22-JUL-77 14:54 T52

MACY11 30(1046) 22-JUL-77 15:10 PAGE 60  
TEST 'SEEK' COMMAND BUS SIGNALS

SEQ 0060

```

3017 015316 004737 022022 JSR PC,SAVRP ;GET THE REGISTERS
3018 015322 013737 001236 001126 MOV SRPM1,SBDDAT ;GET CONTENTS OF RPM1
3019 015330 042737 174000 001126 BIC #174000,SBDDAT ;CLEAR UNWANTED
3020 015336 012737 001360 001124 MOV #1360,$GDDAT ;LOAD EXPECTED VALUE OF RPM1
3021 015344 005705 TST R5 ;IS THIS THE FIRST PASS ?
3022 015346 001403 BEQ B5 ;BRANCH IF YES
3023 015350 012737 001010 001124 MOV #1010,$GDDAT ;SECOND PASS VALUE
3024 015356 023737 001124 001126 8$: CMP $GDDAT,$SBDDAT ;ARE CONTENTS OF RPM1 CORRECT?
3025 015364 001401 BEQ 9$
3026 015366 104074 ERROR 74 ;THE CONTROL SIGNAL SET HEAD AND
3027 ;THE HEAD ADDRESS SHOULD BE ON
3028 ;THE BUS OUT LINES
3029 015370 004037 021744 9$: JSR R0,TIMEP ;GENERATE 4 CLOCK PULSES
3030 015374 000004 .WORD 4 ;CONSTANT FOR 4 CLOCK PULSES
3031 015376 004737 022022 JSR PC,SAVRP ;GET THE REGISTERS
3032 015402 013737 001236 001126 MOV SRPM1,SBDDAT ;GET CONTENTS OF RPM1
3033 015410 042737 174000 001126 BIC #174000,SBDDAT ;CLEAR UNWANTED BITS
3034 015416 012737 002004 001124 MOV #BIT10:BIT02,$GDDAT ;LOAD EXPECTED BITS
3035 015424 023737 001124 001126 CMP $GDDAT,$SBDDAT ;ARE CONTENTS OF RPM1 CORRECT?
3036 015432 001401 BEQ 10$ ;BRANCH IF YES
3037 015434 104075 ERROR 75 ;THE CONTROL SIGNAL SEEK START
3038 ;SHOULD BE ON THE BUS OUT LINES
3039 015436 005704 10$: TST R4 ;IS THIS FIRST PASS?
3040 015440 001002 BNE 11$ ;BRANCH IF NO
3041 015442 005205 INC R5 ;SET SECOND PASS INDICATOR
3042 015444 000602 BR SEEK ;MAKE SECOND PASS
3043 015446 000004 11$: SCOPE ;LOOP ?
3044
3045
3046 ;*****
3047 ;*TEST 53 TEST 'HOME SEEK' COMMAND BUS SIGNALS
3048
3049 ;*ISSUE A RESTORE COMMAND AND CHECK THE GENERATION OF
3050 ;*THE SIGNAL 'RESTORE' ON THE BUS OUT CONTROL LOGIC.
3051
3052 ;*****
3053 T5T53:
3054 015450 000240 NOP
3055 015452 012737 015472 001106 MOV #RESTOR,$LPADR ;LOAD LOOP ON TEST ADDRESS
3056 015460 012737 015472 001110 MOV #RESTOR,$LPERR ;LOAD ERROR LOOP ADDRESS
3057 015466 013704 001210 MOV RPADR,R4 ;RP11E BUS ADDRESS
3058 015472 012706 001100 RESTOR: MOV #STACK,SP ;SETUP THE STACK POINTER
3059 015476 004737 021714 JSR PC,CLRP ;CLEAR THE CONTROLLER
3060 015502 004737 022022 JSR PC,SAVRP ;SAVE THE REGISTERS
3061 015506 013737 001236 001126 MOV SRPM1,SBDDAT ;GET CONTENTS OF RPM1
3062 015514 042737 174000 001126 BIC #174000,SBDDAT ;CLEAR UNWANTED BITS
3063 015522 005737 001126 TST $BDDAT ;ANY SET ?
3064 015526 001403 BEQ 1$ ;BRANCH IF RESULT IS YES
3065 015530 005037 001124 CLR $GDDAT
3066 015534 104071 ERROR 71 ;SOME BUS OUT SIGNALS TO THE
3067 ;DRIVE ARE SET AFTER RESET
3068 015536 012764 000015 000004 1$: MOV #15,RPCS(R4) ;ISSUE HOME COMMAND
3069 015544 012737 060001 001206 MOV #BIT14:BIT13:BIT00,TPL
3070 015552 004037 021744 JSR R0,TIMEP ;GENERATE 4 CLOCK PULSES
3071 015556 000004 .WORD 4 ;CONSTANT FOR 4 CLOCK PULSES
3072 015560 004737 022022 JSR PC,SAVRP ;GET THE REGISTERS

```

K05

MAINDEC-11-DZRPW-C, RP11-E DISKLESS LOGIC TEST  
DZRPWC.P11 22-JUL-77 14:54

MACY11 30(1046) 22-JUL-77 15:10 PAGE 61  
TEST 'HOME SEEK' COMMAND BUS SIGNALS

SEQ 0061

```

3073 015564 013737 001236 001126      MOV      SRPM1,SBDDAT      ;GET CONTROL LINES FROM RPM1
3074 015572 042737 174000 001126      BIC      #174000,SBDDAT   ;CLEAR UNWANTED BITS
3075 015600 012737 002100 001124      MOV      #BIT10!BIT06,SGDDAT ;LOAD EXPECTED VALUE OF RPM1
3076 015606 023737 001124 001126      CMP      SGDDAT,SBDDAT   ;WERE CONTENTS OF RPM1 CORRECT?
3077 015614 001401                BEQ      25              ;BRANCH IF YES
3078 015616 104076                ERROR    76              ;ISSUED HOME COMMAND-EXPECTED
3079                                ;RESTORE AND CONTROL TO BE SET
3080                                ;ON BUS OUT LINES
3081 015620 000004                25:      SCOPE          ;LOOP ?
3082
3083
3084                                ;*****
3085                                ;*TEST 54      TEST 'READ' BUS SIGNALS
3086
3087                                ;*ISSUE A READ COMMAND WITH NO SEEK IMPLIED AND CHECK THE GENERATION
3088                                ;*OF THE SIGNAL READ IN THE BUS OUT LOGIC.
3089
3090                                ;*****
3091                                †T54:
3092 015622 000240                NOP
3093 015624 012737 015644 001106      MOV      #READT,$LPADR   ;LOAD LOOP ON TEST ADDRESS
3094 015632 012737 015644 001110      MOV      #READT,$LPERR   ;LOAD ERROR LOOP ADDRESS
3095 015640 013704 001210                MOV      RPADR,R4        ;RP11E BUS ADDRESS
3096 015644 012706 001100                READT:  MOV      #STACK,SP   ;SETUP THE STACK POINTER
3097 015650 004737 021714                JSR      PC,CLRP        ;CLEAR THE CONTROLLER
3098 015654 012764 177777 000006      MOV      #-1,RPWC(R4)    ;LOAD WORD COUNT
3099 015662 012764 000017 000004      MOV      #17,RPCS(R4)    ;ISSUE READ COMMAND
3100 015670 012737 060001 001206      MOV      #BIT14!BIT13!BIT00,TPL
3101 015676 004037 021744                JSR      RO,TIMEP       ;GENERATE 4 CLOCK PULSES
3102 015702 000004                .WORD   4                ;CONSTANT FOR 4 CLOCK PULSES
3103 015704 012737 060401 001206      MOV      #BIT14!BIT13!BIT08!BIT00,TPL
3104 015712 004037 021744                JSR      RO,TIMEP       ;GENERATE 3 CLOCK PULSES
3105 015716 000003                .WORD   3                ;CONSTANT FOR 3 CLOCK PULSES
3106 015720 004737 022022                JSR      PC,SAVRP       ;GET THE REGISTERS
3107 015724 013737 001236 001126      MOV      SRPM1,SBDDAT   ;GET CONTENTS OF RPM1
3108 015732 042737 174000 001126      BIC      #174000,SBDDAT   ;CLEAR UNWANTED BITS
3109 015740 012737 002042 001124      MOV      #BIT10!BIT05!BIT01,SGDDAT ;LOAD EXPECTED VALUE OF RPM1
3110 015746 023737 001124 001126      CMP      SGDDAT,SBDDAT   ;ARE CONTENTS OF RPM1 CORRECT
3111 015754 001401                BEQ      15              ;BRANCH IF YES
3112 015756 104077                ERROR    77              ;ISSUED READ WITH NO SEEK COMMAND
3113                                ;EXPECTED CONTROL, SELECT HEAD, AND READ
3114                                ;SIGNALS ON THE BUS OUT LINES
3115 015760 000004                15:      SCOPE          ;LOOP ?
3116
3117
3118                                ;*****
3119                                ;*TEST 55      TEST 'WRITE' BUS SIGNALS
3120
3121                                ;*ISSUE A WRITE FORMAT COMMAND AND CHECK THE GENERATION OF THE
3122                                ;*WRITE SIGNAL IN THE BUS OUT LOGIC.
3123
3124                                ;*****
3125                                †T55:
3126 015762 000240                NOP
3127 015764 012737 016004 001106      MOV      #WRITE,$LPADR   ;LOAD LOOP ON TEST ADDRESS
3128 015772 012737 016004 001110      MOV      #WRITE,$LPERR   ;LOAD ERROR LOOP ADDRESS

```

```

3129 016000 013704 001210
3130 016004 012706 001100
3131 016010 004737 021714
3132 016014 012764 177777 000006
3133 016022 012764 000001 000014
3134 016030 012764 014003 000004
3135 016036 012737 060001 001206
3136 016044 004037 021744
3137 016050 000023
3138 016052 004037 021774
3139 016056 000001
3140 016060 012764 020400 000022
3141 016066 012764 020000 000022
3142 016074 012764 020400 000022
3143 016102 004037 021744
3144 016106 000003
3145 016110 004737 022022
3146 016114 013737 001236 001126
3147 016122 042737 174000 001126
3148 016130 012737 002061 001124
3149 016136 023737 001124 001126
3150 016144 001401
3151 016146 104100
3152
3153
3154 016150 000004
3155 016152 004737 021714
3156 016156 000137 016162
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167 016162
3168 016162 104401 016170
3169 01616f 000410
3170
3171 016210
3172 016210 013746 001100
3173 016214 104403
3174 016216 005
3175 016217 000
3176 016220 104401 016226
3177 016224 000413
3178
3179 016254
3180 016254 013746 001112
3181 016260 104402
3182 016262 005037 001112
3183 016266 005037 001102
3184 016272 005037 001162

WRITE: MOV RPADR,R4 ;RP11E BUS ADDRESS
        MOV #STACK,SP ;SETUP THE STACK POINTER
        JSR PC,CLRP ;CLEAR THE CONTROLLER
        MOV #-1,RPMC(R4) ;SET UP WORD COUNT
        MOV #1,RPDA(R4) ;SELECT SECTOR 1, TRACK 0
        MOV #14003,RPCS(R4) ;ISSUE WRITE FORMAT COMMAND
        MOV #BIT14!BIT13!BIT00,TPL
        JSR RO,TIMEP ;GENERATE 19 CLOCK PULSES
        .WORD 19 ;CONSTANT FOR 19 CLOCK PULSES
        JSR RO,INDEXP ;GENERATE 1 INDEX PULSE
        .WORD 1 ;CONSTANT FOR 1 INDEX PULSE
        MOV #BIT13!BIT08,RPM3(R4) ;GENERATE SECTOR PULSE
        MOV #BIT13,RPM3(R4)
        MOV #BIT13!BIT08,RPM3(R4) ;GENERATE SECTOR PULSE
        JSR RO,TIMEP ;GENERATE 3 CLOCK PULSES
        .WORD 3 ;CONSTANT FOR 3 CLOCK PULSES
        JSR PC,SAVRP ;GET THE REGISTERS
        MOV SRPM1,$BDDAT ;GET CONTENTS OF RPM1
        BIC #174000,$BDDAT ;CLEAR UNWANTED BITS
        MOV #BIT10!BIT05!BIT04!BIT00,$GDDAT ;LOAD EXPECTED VALUE OF RPM1
        CMP $GDDAT,$BDDAT ;ARE CONTENTS OF RPM1 CORRECT
        BEQ IS ;BRANCH IF YES
        ERROR 100 ;ISSUED A WRITE FORMAT COMMAND
                    ;AND EXPECTED CONTROL, SELECT HEADS, ERASE,
                    ;AND WRITE SIGNALS ON THE BUS OUT LOGIC
IS:     SCOPE
        JSR PC,CLRP ;CLEAR THE CONTROLLER
        JMP SEOP ;GO TO THE END OF PASS ROUTINE

.SBTTL END OF PASS ROUTINE

;*****
;INCREMENT THE PASS NUMBER ($PASS)
;INDICATE END-OF-PROGRAM AFTER 8. PASSES THRU THE PROGRAM
;IF THERES A MONITOR GO TO IT
;IF THERE ISN'T JUMP TO RSTART

SEOP:   TYPE 65$ ;TYPE ASCIZ STRING
        BR 64$ ;GET OVER THE ASCIZ
;65$: .ASCIZ <15><12><12>/END OF PASS/
64$:   MOV $PASS,-(SP) ;SAVE $PASS FOR TYPEOUT
        TYPOS ;GO TYPE--OCTAL ASCII
        .BYTE 5 ;TYPE 5 DIGIT(S)
        .BYTE 0 ;SUPPRESS LEADING ZEROS
        TYPE 67$ ;TYPE ASCIZ STRING
        BR 66$ ;GET OVER THE ASCIZ
;67$: .ASCIZ / ERRORS DETECTED = /
66$:   MOV $ERTTL,-(SP) ;SAVE $ERTTL FOR TYPEOUT
        TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
        CLR $ERTTL ;CLEAR THE ERROR TOTAL
        CLR $STNM ;ZERO THE TEST NUMBER
        CLR $TIMES ;ZERO THE NUMBER OF ITERATIONS
    
```



```

3185 016276 005237 001100          INC      $PASS          ;; INCREMENT THE PASS NUMBER
3186 016302 042737 100000 001100    BIC      #100000,$PASS  ;; DON'T ALLOW A NEG. NUMBER
3187 016310 005327          DEC      (PC)+         ;; LOOP?
3188 016312 000010          SEOPCT: .WORD      B.
3189 016314 003027          BGT      $DOAGN        ;; YES
3190 016316 012737          MOV      (PC)+,2(PC)+  ;; RESTORE COUNTER
3191 016320 000010          SENDCT: .WORD      B.
3192 016322 016312          SEOPCT
3193 016324 104401 016332    TYPE     65$          ;; TYPE ASCIZ STRING
3194 016330 000407          BR       64$          ;; GET OVER THE ASCIZ
3195          ;; 65$: .ASCIZ <15><12>/END OF TEST/
3196 016350          64$:
3197 016350 104401 016400    TYPE     $ENULL       ;; TYPE NULL CHARACTER
3198 016354 013700 000042    $GET42: MOV      2#42,R0  ;; GET MONITOR ADDRESS
3199 016360 001405          BEQ      $DOAGN       ;; BRANCH IF NO MONITOR
3200 016362 000005          RESET
3201 016364 004710          SENDAD: JSR      PC,(R0) ;; CLEAR THE WORLD
3202 016366 000240          NOP
3203 016370 000240          NOP
3204 016372 000240          NOP
3205 016374          SDOAGN:
3206 016374 000137          JMP      2(PC)+       ;; RETURN
3207 016376 002774          SRTNAD: .WORD      RSTART
3208 016400          377 000  SENULL: .BYTE  -1,-1,0 ;; NULL CHARACTER STRING
3209          016404          .EVEN
3210
3211          .SBTTL  *** SUBROUTINES ***
3212
3213          .SBTTL  ERROR HANDLER ROUTINE
3214
3215          ;; *****
3216          ;; *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3217          ;; *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3218          ;; *AND GO TO TYPERR ON ERROR
3219          ;; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3220          ;; *SW15=1      HALT ON ERROR
3221          ;; *SW13=1      INHIBIT ERROR TYPEOUTS
3222          ;; *SW10=1      BELL ON ERROR
3223          ;; *SW09=1      LOOP ON ERROR
3224          ;; *CALL
3225          ;; *      ERROR  N      ;; ERROR=EMT AND N=ERROR ITEM NUMBER
3226
3227 016404          SERROR:
3228 016404 104407          CKSWR
3229 016406 105237 001103    7$:      INCB      SERFLG  ;; TEST FOR CHANGE IN SOFT-SWR
3230 016412 001775          BEQ      7$          ;; SET THE ERROR FLAG
3231 016414 013777 001102 162520    MOV      $STNM,2DISPLAY ;; DON'T LET THE FLAG GO TO ZERO
3232 016422 032777 002000 162510    BIT      #BIT10,2SWR  ;; DISPLAY TEST NUMBER AND ERROR FLAG
3233 016430 001402          BEQ      1$          ;; BELL ON ERROR?
3234 016432 104401 001166          TYPE     $BELL       ;; NO - SKIP
3235 016436 005237 001112    1$:      INC      $ERTTL     ;; RING BELL
3236 016442 011637 001116          MOV      (SP),SERRPC  ;; COUNT THE NUMBER OF ERRORS
3237 016446 162737 000002 001116    SUB      #2,SERRPC    ;; GET ADDRESS OF ERROR INSTRUCTION
3238 016454 117737 162436 001114    MOVB    2SERRPC,$ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
3239 016462 032777 020000 162450    BIT      #BIT13,2SWR  ;; SKIP TYPEOUT IF SET
3240 016470 001004          BNE     20$          ;; SKIP TYPEOUTS

```

```

3241 016472 004737 016556 JSR PC,TYPERR ;;GO TO USER ERROR ROUTINE
3242 016476 104401 001173 TYPE ,SCRLF
3243 016502 20$: TST 2SWR ;;HALT ON ERROR
3244 016502 005777 162432 BPL 3$ ;;SKIP IF CONTINUE
3245 016506 100002 HALT ;;HALT ON ERROR!
3246 016510 000000 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
3247 016512 104407 BIT #BIT09,2SWR ;;LOOP ON ERROR SWITCH SET?
3248 016514 032777 001000 162416 3$: BEQ 4$ ;;BR IF NO
3249 016522 001402 MOV $LPERR,(SP) ;;FUDGE RETURN FOR LOOPING
3250 016524 013716 001110 TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
3251 016530 005737 001164 4$: BEQ 5$ ;;BR IF NONE
3252 016534 001402 MOV $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
3253 016536 013716 001164 5$: CMP #SENDAD,2#42 ;;ACT-11 AUTO-ACCEPT?
3254 016542 022737 016364 000042 BNE 6$ ;;BRANCH IF NO
3255 016550 001001 HALT ;;YES
3256 016552 000000 6$: RTI ;;RETURN
3257 016554 000002
3258 016554
3259 016554 000002
3260
3261 016556 104413 TYPERR: SAVREG ;;SAVE R0-R5
3262 016560 005037 001160 CLR $TMPD ;;CLEAR LOCATION FOR TEST NUMBER
3263 016564 113737 001102 001160 MOV $STNM,$TMPD ;;LOAD TEST NUMBER FOR TYPEOUT
3264 016572 005000 CLR R0 ;;CLEAR R0 FOR ERROR NUMBER
3265 016574 113700 001114 MOV $ITEMB,R0 ;;ERROR NUMBER
3266 016600 005300 DEC R0 ;;FORM INDEX FOR ERROR TABLE
3267 016602 006300 ASL R0
3268 016604 006300 ASL R0
3269 016606 006300 ASL R0
3270 016610 062700 001244 1$: ADD #SERRTB,R0 ;;FORM ADDRESS
3271 016614 012037 016630 MOV (R0)+,2$ ;;GET ERROR MESSAGE (EM) POINTER
3272 016620 001404 BEQ 3$ ;;BRANCH IF THERE ISN'T ONE
3273 016622 104401 001173 TYPE ,SCRLF ;;CARRIAGE RETURN - LINE FEED
3274 016626 104401
3275 016630 000000 2$: .WORD 0 ;;"EM" POINTER GOES HERE
3276 016632 012037 016646 3$: MOV (R0)+,4$ ;;PICK UP DATA HEADER (DH) POINTER
3277 016636 001404 BEQ 5$ ;;BRANCH IF NONE
3278 016640 104401 001173 TYPE ,SCRLF ;;CARRIAGE RETURN-LINE FEED
3279 016644 104401
3280 016646 000000 4$: .WORD 0 ;;"DH" POINTER GOES HERE
3281 016650 012001 5$: MOV (R0)+,R1 ;;PICKUP DATA TABLE (DT) POINTER
3282 016652 001450 BEQ 20$ ;;BRANCH IF NONE
3283 016654 005005 CLR R5 ;;SET INDENT SWITCH
3284 016656 012000 MOV (R0)+,R0 ;;DATA FORMAT (DF) POINTER
3285 016660 012002 MOV (R0)+,R2 ;;NUMBER OF DH'S TO TYPE
3286 016662 001441 BEQ 17$ ;;BRANCH IF DH NUMBER IS 0
3287 016664 005105 COM R5 ;;NO INDENT
3288 016666 104401 001173 TYPE ,SCRLF ;;CARRIAGE RETURN-LINE FEED
3289 016672 112003 10$: MOV $R3 ;;NUMBER OF DATA WORDS TO TYPE
3290 016674 112004 MOV $R4 ;;AND HOW TO TYPE THEM
3291 016676 006004 ROR R4 ;;OCTAL OR DECIMAL?
3292 016700 103403 BCS 12$ ;;DECIMAL--BRANCH
3293 016702 013146 MOV 2(R1)+,-(SP) ;;SAVE 2(R1)+ FOR TYPEOUT
3294 016704 104402 TPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3295 016706 000402 BR 13$
3296 016710 12$:
    
```

```

3297 016710 013146          MOV      2(R1)+,-(SP)      ;;SAVE 2(R1)+ FOR TYPEOUT
3298 016712 104405          TYPDS                    ;;GO TYPE--DECIMAL ASCII WITH SIGN
3299 016714 005303          13$: DEC      R3          ;;MORE NUMBERS TO TYPE?
3300 016716 001403          BEQ      14$             ;;NO--BRANCH
3301 016720 104401 022120    TYPE     ,BLNKS2         ;;YES--TYPE SEPARATORS
3302 016724 000764          BR       11$             ;;LOOP
3303 016726 005302          14$: DEC      R2          ;;MORE DH'S?
3304 016730 003421          BLE     20$             ;;NO--BRANCH
3305 016732 104401 001173    TYPE     ,CRLF           ;;YES--START A NEW LINE
3306 016736 005105          COM      R5              ;;INDENT?
3307 016740 001002          BNE     15$             ;;NO--BRANCH
3308 016742 104401 022120    TYPE     ,BLNKS2         ;;YES--TYPE SPACES
3309 016746 012037 016754    15$: MOV      (R0)+,16$   ;;GET NEXT DH
3310 016752 104401          TYPE                    ;;AND TYPE IT
3311 016754 000000          .WORD   0               ;;DH POINTER GOES HERE
3312 016756 104401 001173    TYPE     ,CRLF           ;;CARRIAGE RETURN-LINE FEED
3313 016762 005705          TST     R5              ;;INDENT?
3314 016764 001342          BNE     10$             ;;NO--BRANCH
3315 016766 104401 022120    17$: TYPE     ,BLNKS2         ;;YES--TYPE SPACES
3316 016772 000737          BR       10$             ;;LOOP
3317 016774 104414          20$: RESREG                    ;;RESTORE R0-R5
3318 016776 000207          RTS      PC              ;;RETURN

```

.SBTTL TYPE ROUTINE

```

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

```

```

3338 017000 105737 001157    $TYPE: TSTB     $TPFLG      ;; IS THERE A TERMINAL?
3339 017004 100002          BPL     1$              ;; BR IF YES
3340 017006 000000          HALT                    ;; HALT HERE IF NO TERMINAL
3341 017010 000407          BR       3$              ;; LEAVE
3342 017012 010046          1$: MOV      R0,-(SP)     ;; SAVE R0
3343 017014 017600 000002    MOV     2(SP),R0        ;; GET ADDRESS OF ASCIZ STRING
3344 017020 112046          2$: MOVB    (R0)+,-(SP)  ;; PUSH CHARACTER TO BE TYPED ONTO STACK
3345 017022 001005          BNE     4$              ;; BR IF IT ISN'T THE TERMINATOR
3346 017024 005726          TST     (SP)+          ;; IF TERMINATOR POP IT OFF THE STACK
3347 017026 012600          60$: MOV     (SP)+,R0    ;; RESTORE R0
3348 017030 062716 000002    3$: ADD     #2,(SP)      ;; ADJUST RETURN PC
3349 017034 000002          RTI                    ;; RETURN
3350 017036 122716 000011    4$: CMPB    #HT,(SP)    ;; BRANCH IF <HT>
3351 017042 001430          BEQ     8$              ;;
3352 017044 122716 000200    CMPB    #CRLF,(SP)     ;; BRANCH IF NOT <CRLF>

```

```

3353 017050 001006      BNE      5$
3354 017052 005726      TST      (SP)+      ;;POP <CR><LF> EQUIV
3355 017054 104401      TYPE     ;;TYPE A CR AND LF
3356 017056 001173      SCRLF
3357 017060 105037 017214  CLRB     $CHARCNT   ;;CLEAR CHARACTER COUNT
3358 017064 000755      BR      2$         ;;GET NEXT CHARACTER
3359 017066 004737 017150  5$:     JSR     PC,$TYPEC  ;;GO TYPE THIS CHARACTER
3360 017072 123726 001156  6$:     CMPB   $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
3361 017076 001350      BNE     2$         ;;IF NO GO GET NEXT CHAR.
3362 017100 013746 001154  MOV     $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
3363                                ;;AND THE NULL CHAR.
3364 017104 105366 000001  7$:     DECB   1(SP)    ;;DOES A NULL NEED TO BE TYPED?
3365 017110 002770      BLT     6$         ;;BR IF NO--GO POP THE NULL OFF OF STACK
3366 017112 004737 017150  JSR     PC,$TYPEC  ;;GO TYPE A NULL
3367 017116 105337 017214  DECB   $CHARCNT   ;;DO NOT COUNT AS A COUNT
3368 017122 000770      BR      7$         ;;LOOP
    
```

;HORIZONTAL TAB PROCESSOR

```

3372 017124 112716 000040  8$:     MOVB   #' (SP)   ;;REPLACE TAB WITH SPACE
3373 017130 004737 017150  9$:     JSR     PC,$TYPEC  ;;TYPE A SPACE
3374 017134 132737 000007 017214  BITB   #',$CHARCNT ;;BRANCH IF NOT AT
3375 017142 001372      BNE     9$         ;;TAB STOP
3376 017144 005726      TST     (SP)+     ;;POP SPACE OFF STACK
3377 017146 000724      BR      2$         ;;GET NEXT CHARACTER
3378 017150 105777 161774  $TYPEC: TSTB   $STPS   ;;WAIT UNTIL PRINTER IS READY
3379 017154 100375      BPL     $TYPEC
3380 017156 116677 000002 161766  MOVB   2(SP),$STPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3381 017164 122766 000015 000002  CMPB   $CR,2(SP)  ;;IS CHARACTER A CARRIAGE RETURN?
3382 017172 001003      BNE     1$         ;;BRANCH IF NO
3383 017174 105037 017214  CLRB   $CHARCNT   ;;YES--CLEAR CHARACTER COUNT
3384 017200 000406      BR      $TYPEX
3385 017202 122766 000012 000002  1$:     CMPB   $LF,2(SP) ;;IS CHARACTER A LINE FEED?
3386 017210 001402      BEQ    $TYPEX     ;;BRANCH IF YES
3387 017212 105227      INCB   (PC)+     ;;COUNT THE CHARACTER
3388 017214 000000      $CHARCNT: .WORD 0 ;;CHARACTER COUNT STORAGE
3389 017216 000207      $TYPEX: RTS     PC
    
```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

3392
3393
3394
3395
3396
3397
3398
3399
3400
3401
3402
3403
3404
3405
3406
3407
3408
    ;;*****
    ;;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
    ;;OCTAL (ASCII) NUMBER AND TYPE IT.
    ;;$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
    ;;CALL:
    ;;      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
    ;;      TYPOS      ;;CALL FOR TYPEOUT
    ;;      .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
    ;;      .BYTE   M              ;;M=1 OR 0
    ;;                                ;;1=TYPE LEADING ZEROS
    ;;                                ;;0=SUPPRESS LEADING ZEROS
    ;;$STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
    ;;$TYPOS OR $TYPOC
    ;;CALL:
    
```

```

3409      *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
3410      *      TYPON      ;;CALL FOR TYPEOUT
3411      *
3412      *$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3413      *CALL:
3414      *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
3415      *      TYPOC      ;;CALL FOR TYPEOUT
3416
3417 017220 017646 000000      STYPOS: MOV      2(SP),-(SP)      ;;PICKUP THE MODE
3418 017224 116637 000001 017443      MOV      1(SP),SOFILL      ;;LOAD ZERO FILL SWITCH
3419 017232 112637 017445      MOV      (SP)+,SOMODE+1      ;;NUMBER OF DIGITS TO TYPE
3420 017236 062716 000002      ADD      #2,(SP)      ;;ADJUST RETURN ADDRESS
3421 017242 000406      BR      STYPOC
3422 017244 112737 000001 017443      STYPOC: MOV      #1,SOFILL      ;;SET THE ZERO FILL SWITCH
3423 017252 112737 000006 017445      MOV      #6,SOMODE+1      ;;SET FOR SIX(6) DIGITS
3424 017260 112737 000005 017442      STYPON: MOV      #5,SOCNT      ;;SET THE ITERATION COUNT
3425 017266 010346      MOV      R3,-(SP)      ;;SAVE R3
3426 017270 010446      MOV      R4,-(SP)      ;;SAVE R4
3427 017272 010546      MOV      R5,-(SP)      ;;SAVE R5
3428 017274 113704 017445      MOV      SOMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
3429      NEG      R4
3430 017302 062704 000006      ADD      #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
3431 017306 110437 017444      MOV      R4,SOMODE      ;;SAVE IT FOR USE
3432 017312 113704 017443      MOV      SOFILL,R4      ;;GET THE ZERO FILL SWITCH
3433 017316 016605 000012      MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
3434 017322 005003      CLR      R3      ;;CLEAR THE OUTPUT WORD
3435 017324 006105      1$: ROL      R5      ;;ROTATE MSB INTO "C"
3436 017326 000404      BR      3$      ;;GO DO MSB
3437 017330 006105      2$: ROL      R5      ;;FORM THIS DIGIT
3438 017332 006105      ROL      R5
3439 017334 006105      ROL      R5
3440 017336 010503      MOV      R5,R3
3441 017340 006103      3$: ROL      R3      ;;GET LSB OF THIS DIGIT
3442 017342 105337 017444      DECB      SOMODE      ;;TYPE THIS DIGIT?
3443 017346 100016      BPL      7$      ;;BR IF NO
3444 017350 042703 177770      BIC      #177770,R3      ;;GET RID OF JUNK
3445 017354 001002      BNE      4$      ;;TEST FOR 0
3446 017356 005704      TST      R4      ;;SUPPRESS THIS 0?
3447 017360 001403      BEQ      5$      ;;BR IF YES
3448 017362 005204      4$: INC      R4      ;;DON'T SUPPRESS ANYMORE 0'S
3449 017364 052703 000060      BIS      #'0,R3      ;;MAKE THIS DIGIT ASCII
3450 017370 052703 000040      5$: BIS      #' ,R3      ;;MAKE ASCII IF NOT ALREADY
3451 017374 110337 017440      MOV      R3,#$      ;;SAVE FOR TYPING
3452 017400 104401 017440      TYPE      8$      ;;GO TYPE THIS DIGIT
3453 017404 105337 017442      7$: DECB      $SOCNT      ;;COUNT BY 1
3454 017410 003347      BGT      2$      ;;BR IF MORE TO DO
3455 017412 002402      BLT      6$      ;;BR IF DONE
3456 017414 005204      INC      R4      ;;INSURE LAST DIGIT ISN'T A BLANK
3457 017416 000744      BR      2$      ;;GO DO THE LAST DIGIT
3458 017420 012605      6$: MOV      (SP)+,R5      ;;RESTORE R5
3459 017422 012604      MOV      (SP)+,R4      ;;RESTORE R4
3460 017424 012603      MOV      (SP)+,R3      ;;RESTORE R3
3461 017426 016666 000002 000004      MOV      2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
3462 017434 012616      MOV      (SP)+,(SP)
3463 017436 000002      RTI
3464 017440 000      8$: .BYTE 0      ;;RETURN
      ;;STORAGE FOR ASCII DIGIT

```

```

3465 017441 000 .BYTE 0 ;; TERMINATOR FOR TYPE ROUTINE
3466 017442 000 $OCNT: .BYTE 0 ;; OCTAL DIGIT COUNTER
3467 017443 000 $OFILL: .BYTE 0 ;; ZERO FILL SWITCH
3468 017444 000000 $OMODE: .WORD 0 ;; NUMBER OF DIGITS TO TYPE
3469
3470 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
3471
3472 ;; *****
3473 ;; *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
3474 ;; *SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
3475 ;; *NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
3476 ;; *BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
3477 ;; *REPLACED WITH SPACES.
3478 ;; *CALL:
3479 ;; * MOV NUM,-(SP) ;; PUT THE BINARY NUMBER ON THE STACK
3480 ;; * TYPDS ;; GO TO THE ROUTINE
3481
3482 $TYPDS:
3483 MOV R0,-(SP) ;; PUSH R0 ON STACK
3484 MOV R1,-(SP) ;; PUSH R1 ON STACK
3485 MOV R2,-(SP) ;; PUSH R2 ON STACK
3486 MOV R3,-(SP) ;; PUSH R3 ON STACK
3487 MOV R5,-(SP) ;; PUSH R5 ON STACK
3488 MOV #20200,-(SP) ;; SET BLANK SWITCH AND SIGN
3489 MOV 20(SP),R5 ;; GET THE INPUT NUMBER
3490 BPL 1$ ;; BR IF INPUT IS POS.
3491 NEG R5 ;; MAKE THE BINARY NUMBER POS.
3492 MOVB #'-,1(SP) ;; MAKE THE ASCII NUMBER NEG.
3493 CLR R0 ;; ZERO THE CONSTANTS INDEX
3494 MOV $DBLK,R3 ;; SETUP THE OUTPUT POINTER
3495 MOVB #' ,(R3)+ ;; SET THE FIRST CHARACTER TO A BLANK
3496 CLR R2 ;; CLEAR THE BCD NUMBER
3497 MOV $DTBL(R0),R1 ;; GET THE CONSTANT
3498 SUB R1,R5 ;; FORM THIS BCD DIGIT
3499 BLT 4$ ;; BR IF DONE
3500 INC R2 ;; INCREASE THE BCD DIGIT BY 1
3501 BR 3$
3502 ADD R1,R5 ;; ADD BACK THE CONSTANT
3503 TST R2 ;; CHECK IF BCD DIGIT=0
3504 BNE 5$ ;; FALL THROUGH IF 0
3505 TSTB (SP) ;; STILL DOING LEADING 0'S?
3506 BMI 7$ ;; BR IF YES
3507 ASLB (SP) ;; MSD?
3508 BCC 6$ ;; BR IF NO
3509 MOVB 1(SP),-1(R3) ;; YES--SET THE SIGN
3510 BIS #'0,R2 ;; MAKE THE BCD DIGIT ASCII
3511 BIS #' ,R2 ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
3512 MOVB R2,(R3)+ ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
3513 TST (R0)+ ;; JUST INCREMENTING
3514 CMP R0,#10 ;; CHECK THE TABLE INDEX
3515 BLT 2$ ;; GO DO THE NEXT DIGIT
3516 BGT 8$ ;; GO TO EXIT
3517 MOV R5,R2 ;; GET THE LSD
3518 BR 6$ ;; GO CHANGE TO ASCII
3519 TSTB (SP)+ ;; WAS THE LSD THE FIRST NON-ZERO?
3520 BPL 9$ ;; BR IF NO
    
```

F06

```

3521 017612 116663 177777 177776          MOVB    -1(SP),-2(R3)    ;;YES--SET THE SIGN FOR TYPING
3522 017620 105013          9$:     CLR      (R3)      ;;SET THE TERMINATOR
3523 017622 012605          MOV     (SP)+,R5        ;;POP STACK INTO R5
3524 017624 012603          MOV     (SP)+,R3        ;;POP STACK INTO R3
3525 017626 012602          MOV     (SP)+,R2        ;;POP STACK INTO R2
3526 017630 012601          MOV     (SP)+,R1        ;;POP STACK INTO R1
3527 017632 012600          MOV     (SP)+,R0        ;;POP STACK INTO R0
3528 017634 104401 017662          TYPE    $DBLK           ;;NOW TYPE THE NUMBER
3529 017640 016666 000002 000004          MOV     2(SP),4(SP)    ;;ADJUST THE STACK
3530 017646 012616          MOV     (SP)+,(SP)
3531 017650 000002          RTI                                     ;;RETURN TO USER
3532 017652 023420          $DTBL: 10000.
3533 017654 001750          1000.
3534 017656 000144          100.
3535 017660 000012          10.
3536 017662 000004          $DBLK: .BLKW 4
3537
3538          .SBTTL  TTY INPUT ROUTINE
3539
3540          ;;*****
3541          .ENABL  LSB
3542
3543          ;;*****
3544          #SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
3545          #ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
3546          #SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
3547          #WHEN OPERATING IN TTY FLAG MODE.
3548 017672 022737 000176 001140  $CKSWR: CMP     #SWREG,SWR    ;;IS THE SOFT-SWR SELECTED?
3549 017700 001074          BNE     15$             ;;BRANCH IF NO
3550 017702 105777 161236          TSTB   2$TKS           ;;CHAR THERE?
3551 017706 100071          BPL     15$             ;;IF NO, DON'T WAIT AROUND
3552 017710 117746 161232          MOVB   2$TKB,-(SP)     ;;SAVE THE CHAR
3553 017714 042716 177600          BIC    #1C177,(SP)    ;;STRIP-OFF THE ASCII
3554 017720 022726 000007          CMP    #7,(SP)+       ;;IS IT A CONTROL G?
3555 017724 001062          BNE     15$             ;;NO, RETURN TO USER
3556 017726 123727 001134 000001          CMPB   $AUTOB,#1      ;;ARE WE RUNNING IN AUTO-MODE?
3557 017734 001456          BEQ    15$             ;;BRANCH IF YES
3558
3559 017736 104401 020544          $GTSWR: TYPE    ,SCNTLG   ;;ECHO THE CONTROL-G (↑G)
3560 017742 104401 020551          TYPE    ,SMSWR        ;;TYPE CURRENT CONTENTS
3561 017746 013746 000176          MOV     $WREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
3562 017752 104402          TYPOC   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3563 017754 104401 020562          TYPE    ,SMNEW        ;;PROMPT FOR NEW SWR
3564 017760 005046          19$:    CLR     -(SP)    ;;CLEAR COUNTER
3565 017762 005046          CLR     -(SP)          ;;THE NEW SWR
3566 017764 105777 161154          7$:    TSTB   2$TKS     ;;CHAR THERE?
3567 017770 100375          BPL     7$             ;;IF NOT TRY AGAIN
3568
3569 017772 117746 161150          MOVB   2$TKB,-(SP)    ;;PICK UP CHAR
3570 017776 042716 177600          BIC    #1C177,(SP)    ;;MAKE IT 7-BIT ASCII
3571
3572
3573
3574 020002 021627 000025          9$:    CMP     (SP),#25    ;;IS IT A CONTROL-U?
3575 020006 001005          BNE     10$            ;;BRANCH IF NOT
3576 020010 104401 020537          TYPE    ,SCNTLU       ;;YES, ECHO CONTROL-U (↑U)
    
```

```

3577 020014 062706 000006      20$:  ADD      #6,SP      ;; IGNORE PREVIOUS INPUT
3578 020020 000757              BR          19$      ;; LET'S TRY IT AGAIN
3579
3580
3581 020022 021627 000015      10$:  CMP      (SP),#15   ;; IS IT A <CR>?
3582 020026 001022              BNE          16$      ;; BRANCH IF NO
3583 020030 005766 000004              TST      4(SP)      ;; YES, IS IT THE FIRST CHAR?
3584 020034 001403              BEQ          11$      ;; BRANCH IF YES
3585 020036 016677 000002 161074      MOV      2(SP),2$WR  ;; SAVE NEW $WR
3586 020044 062706 000006      11$:  ADD      #6,SP      ;; CLEAR UP STACK
3587 020050 104401 001173      14$:  TYPE     $CRLF     ;; ECHO <CR> AND <LF>
3588 020054 123727 001135 000001      CMPB    $INTAG,#1   ;; RE-ENABLE TTY KBD INTERRUPTS?
3589 020062 001003              BNE          15$      ;; BRANCH IF NOT
3590 020064 012777 000100 161052      MOV      #100,2$TKS ;; RE-ENABLE TTY KBD INTERRUPTS
3591 020072 000002      15$:  RTI                    ;; RETURN
3592 020074 004737 017150      16$:  JSR      PC,$TYPEC  ;; ECHO CHAR
3593 020100 021627 000060              CMP      (SP),#60   ;; CHAR < 0?
3594 020104 002420              BLT          18$      ;; BRANCH IF YES
3595 020106 021627 000067              CMP      (SP),#67   ;; CHAR > 7?
3596 020112 003015              BGT          18$      ;; BRANCH IF YES
3597 020114 042726 000060              BIC      #60,(SP)+  ;; STRIP-OFF ASCII
3598 020120 005766 000002              TST      2(SP)      ;; IS THIS THE FIRST CHAR
3599 020124 001403              BEQ          17$      ;; BRANCH IF YES
3600 020126 006316              ASL      (SP)        ;; NO, SHIFT PRESENT
3601 020130 006316              ASL      (SP)        ;; CHAR OVER TO MAKE
3602 020132 006316              ASL      (SP)        ;; ROOM FOR NEW ONE.
3603 020134 005266 000002      17$:  INC      2(SP)        ;; KEEP COUNT OF CHAR
3604 020140 056616 177776              BIS      -2(SP),(SP) ;; SET IN NEW CHAR
3605 020144 000707              BR          7$        ;; GET THE NEXT ONE
3606 020146 104401 001172      18$:  TYPE     $QUES     ;; TYPE ?<CR><LF>
3607 020152 000720              BR          20$      ;; SIMULATE CONTROL-U
3608 .DSABL  LSB

```

```

3609
3610
3611 *****
3612 *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
3613 *CALL:
3614 *      RDCHR          ;; INPUT A SINGLE CHARACTER FROM THE TTY
3615 *      RETURN HERE   ;; CHARACTER IS ON THE STACK
3616 *                  ;; WITH PARITY BIT STRIPPED OFF
3617
3618
3619 020154 011646 000004 000002 $RDCHR: MOV      (SP),-(SP)  ;; PUSH DOWN THE PC
3620 020156 016666 000004 160754 1$:  MOV      4(SP),2(SP) ;; SAVE THE PS
3621 020164 105777 160754              TSTB    2$TKS      ;; WAIT FOR
3622 020170 100375 160750 000004      BPL          1$      ;; A CHARACTER
3623 020172 117766 177600 000004      MOVB    2$TKB,4(SP) ;; READ THE TTY
3624 020200 042766 177600 000004      BIC     #177,4(SP)  ;; GET RID OF JUNK IF ANY
3625 020206 026627 000004 000023      CMP     4(SP),#23  ;; IS IT A CONTROL-S?
3626 020214 001013              BNE          3$      ;; BRANCH IF NO
3627 020216 105777 160722      2$:  TSTB    2$TKS      ;; WAIT FOR A CHARACTER
3628 020222 100375 160716 177600      BPL          2$      ;; LOOP UNTIL ITS THERE
3629 020224 117746 177600              MOVB    2$TKB,-(SP) ;; GET CHARACTER
3630 020230 042716 177600              BIC     #177,(SP)  ;; MAKE IT 7-BIT ASCII
3631 020234 022627 000021      CMP     (SP)+,#21  ;; IS IT A CONTROL-Q?
3632 020240 001366              BNE          2$      ;; IF NOT DISCARD IT

```



```

3633 020242 000750          BR      1$          ;; YES, RESUME
3634 020244 026627 000004 000140 3$:  CMP      4(SP),#140  ;; IS IT UPPER CASE?
3635 020252 002407          BLT      4$          ;; BRANCH IF YES
3636 020254 026627 000004 000175          CMP      4(SP),#175  ;; IS IT A SPECIAL CHAR?
3637 020262 003003          BGT      4$          ;; BRANCH IF YES
3638 020264 042766 000040 000004          BIC      #40,4(SP)   ;; MAKE IT UPPER CASE
3639 020272 000002          RTI          ;; GO BACK TO USER
3640                                     ;; *****
3641                                     ;; *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
3642                                     ;; *CALL:
3643                                     ;; *      RDLIN          ;; INPUT A STRING FROM THE TTY
3644                                     ;; *      RETURN HERE  ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
3645                                     ;; *                                     ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
3646
3647 020274 010346          $RDLIN: MOV      R3,-(SP)   ;; SAVE R3
3648 020276 005046          CLR      -(SP)      ;; CLEAR THE RUBOUT KEY
3649 020300 012703 020530          1$:  MOV      #TTYIN,R3  ;; GET ADDRESS
3650 020304 022703 020537          2$:  CMP      #TTYIN+7,R3  ;; BUFFER FULL?
3651 020310 101456          BLOS     4$          ;; BR IF YES
3652 020312 104410          RDCHR   ;; GO READ ONE CHARACTER FROM THE TTY
3653 020314 112613          MOVW    (SP)+,(R3)  ;; GET CHARACTER
3654 020316 122713 000177          10$: CMPB    #177,(R3)   ;; IS IT A RUBOUT
3655 020322 001022          BNE     5$          ;; BR IF NO
3656 020324 005716          TST     (SP)        ;; IS THIS THE FIRST RUBOUT?
3657 020326 001007          BNE     6$          ;; BR IF NO
3658 020330 112737 000134 020526          MOVW    #' \,9$    ;; TYPE A BACK SLASH
3659 020336 104401 020526          TYPE   9$
3660 020342 012716 177777          MOV     #-1,(SP)   ;; SET THE RUBOUT KEY
3661 020346 005303          6$:  DEC     R3        ;; BACKUP BY ONE
3662 020350 020327 020530          CMP     R3,#TTYIN  ;; STACK EMPTY?
3663 020354 103434          BLO     4$          ;; BR IF YES
3664 020356 111337 020526          MOVW    (R3),9$    ;; SETUP TO TYPEOUT THE DELETED CHAR.
3665 020362 104401 020526          TYPE   9$          ;; GO TYPE
3666 020366 000746          BR      2$          ;; GO READ ANOTHER CHAR.
3667 020370 005716          5$:  TST     (SP)        ;; RUBOUT KEY SET?
3668 020372 001406          BEQ     7$          ;; BR IF NO
3669 020374 112737 000134 020526          MOVW    #' \,9$    ;; TYPE A BACK SLASH
3670 020402 104401 020526          TYPE   9$
3671 020406 005016          CLR     (SP)        ;; CLEAR THE RUBOUT KEY
3672 020410 122713 000025          7$:  CMPB    #25,(R3)   ;; IS CHARACTER A CTRL U?
3673 020414 001003          BNE     8$          ;; BR IF NO
3674 020416 104401 020537          TYPE   ,SCNTLU     ;; TYPE A CONTROL "U"
3675 020422 000726          BR      1$          ;; GO START OVER
3676 020424 122713 000022          8$:  CMPB    #22,(R3)   ;; IS CHARACTER A "r"?
3677 020430 001011          BNE     3$          ;; BRANCH IF NO
3678 020432 105013          CLRB   (R3)        ;; CLEAR THE CHARACTER
3679 020434 104401 001173          TYPE   ,SCRLF     ;; TYPE A "CR" & "LF"
3680 020440 104401 020530          TYPE   ,TTYIN     ;; TYPE THE INPUT STRING
3681 020444 000717          BR      2$          ;; GO PICKUP ANOTHER CHARACTER
3682 020446 104401 001172          4$:  TYPE   ,QUES     ;; TYPE A '?'
3683 020452 000712          BR      1$          ;; CLEAR THE BUFFER AND LOOP
3684 020454 111337 020526          3$:  MOVW    (R3),9$   ;; ECHO THE CHARACTER
3685 020460 104401 020526          TYPE   9$
3686 020464 122723 000015          CMPB    #15,(R3)+  ;; CHECK FOR RETURN
3687 020470 001305          BNE     2$          ;; LOOP IF NOT RETURN
3688 020472 105063 177777          CLRB   -1(R3)     ;; CLEAR RETURN (THE 15)

```

```

3689 020476 104401 001174      TYPE      ,SLF      ;; TYPE A LINE FEED
3690 020502 005726      TST      (SP)+    ;; CLEAN RUBOUT KEY FROM THE STACK
3691 020504 012603      MOV      (SP)+,R3  ;; RESTORE R3
3692 020506 011646      MOV      (SP),-(SP) ;; ADJUST THE STACK AND PUT ADDRESS OF THE
3693 020510 016666 000004 000002  MOV      4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
3694 020516 012766 020530 000004  MOV      #STTYIN,4(SP)
3695 020524 000002      RTI          ;; RETURN
3696 020526      000      9$:      .BYTE      0      ;; STORAGE FOR ASCII CHAR. TO TYPE
3697 020527      000      .BYTE      0      ;; TERMINATOR
3698 020530 000007  STTYIN:  .BLKB      7      ;; RESERVE 7 BYTES FOR TTY INPUT
3699 020537      136 006525 000012  SCNTLU:  .ASCIZ  /↑U/<15><12> ;; CONTROL "U"
3700 020544 043536 005015      000  SCNTLG:  .ASCIZ  /↑G/<15><12> ;; CONTROL "G"
3701 020551      015 051412 051127  SMSWR:  .ASCIZ  <15><12>/SWR = /
3702 020556 036440 000040
3703 020562 020040 042516 020127  SMNEW:  .ASCIZ  / NEW = /
3704 020570 020075      000
3705      020574      .EVEN
3706
3707      .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
3708
3709      ;; *****
3710      ;; *THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
3711      ;; *CHANGE IT TO BINARY.
3712      ;; *THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
3713      ;; *OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
3714      ;; *FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
3715      ;; *THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
3716      ;; *CALL:
3717      ;; *      RDOCT          ;; READ AN OCTAL NUMBER
3718      ;; *      RETURN HERE  ;; LOW ORDER BITS ARE ON TOP OF THE STACK
3719      ;; *                  ;; HIGH ORDER BITS ARE IN SHIOCT
3720
3721 020574 011646 000004 000002  SRDOCT: MOV      (SP),-(SP) ;; PROVIDE SPACE FOR THE
3722 020576 016666      MOV      4(SP),2(SP) ;; INPUT NUMBER
3723 020604 010046      MOV      R0,-(SP)    ;; PUSH R0 ON STACK
3724 020606 010146      MOV      R1,-(SP)    ;; PUSH R1 ON STACK
3725 020610 010246      MOV      R2,-(SP)    ;; PUSH R2 ON STACK
3726 020612 104411      1$:      RDLIN      ;; READ AN ASCIZ LINE
3727 020614 012600      MOV      (SP)+,R0    ;; GET ADDRESS OF 1ST CHARACTER
3728 020616 010037 020722  MOV      R0,5$      ;; AND SAVE IT
3729 020622 005001      CLR      R1          ;; CLEAR DATA WORD
3730 020624 005002      CLR      R2
3731 020626 112046      2$:      MOVB      (R0)+,-(SP) ;; PICKUP THIS CHARACTER
3732 020630 001420      BEQ      3$          ;; IF ZERO GET OUT
3733 020632 122716 000060  CMPB      #'0,(SP)   ;; MAKE SURE THIS CHARACTER
3734 020636 003026      BGT      4$          ;; IS AN OCTAL DIGIT
3735 020640 122716 000067  CMPB      #'7,(SP)
3736 020644 002423      BLT      4$
3737 020646 006301      ASL      R1          ;; *2
3738 020650 006102      ROL      R2
3739 020652 006301      ASL      R1          ;; *4
3740 020654 006102      ROL      R2
3741 020656 006301      ASL      R1          ;; *8
3742 020660 006102      ROL      R2
3743 020662 042716 177770  BIC      #↑C7,(SP)   ;; STRIP THE ASCII JUNK
3744 020666 062601      ADD      (SP)+,R1   ;; ADD IN THIS DIGIT

```

```

3745 020670 000756 BR 2$ ;;LOOP
3746 020672 005726 3$: TST (SP)+ ;;CLEAN TERMINATOR FROM STACK
3747 020674 010166 000012 MOV R1,12(SP) ;;SAVE THE RESULT
3748 020700 010237 020732 MOV R2,$HIOCT
3749 020704 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
3750 020706 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
3751 020710 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
3752 020712 000002 RTI ;;RETURN
3753 020714 005726 4$: TST (SP)+ ;;CLEAN PARTIAL FROM STACK
3754 020716 105010 CLRB (R0) ;;SET A TERMINATOR
3755 020720 104401 TYPE ;;TYPE UP THRU THE BAD CHAR.
3756 020722 000000 5$: .WORD 0
3757 020724 104401 001172 TYPE $QUES ;;?" "CR" & "LF"
3758 020730 000730 BR 1$ ;;TRY AGAIN
3759 020732 000000 $HIOCT: .WORD 0 ;;HIGH ORDER BITS GO HERE
3760
3761 .SBTTL SCOPE HANDLER ROUTINE
3762
3763 *****
3764 *THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
3765 *AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
3766 *AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
3767 *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3768 *SW14=1 LOOP ON TEST
3769 *SW11=1 INHIBIT ITERATIONS
3770 *SW09=1 LOOP ON ERROR
3771 *SW08=1 LOOP ON TEST IN SWR<6:0>
3772 *CALL
3773 * SCOPE ;;SCOPE=IOT
3774
3775 $SCOPE:
3776 020734 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
3777 020736 032777 040000 160174 1$: BIT #BIT14,$SWR ;;LOOP ON PRESENT TEST?
3778 020744 001114 BNE $OVER ;;YES IF SW14=1
3779 ;;###START OF CODE FOR THE XOR TESTER####
3780 020746 000416 $XTSTR: BR 6$ ;;IF RUNNING ON THE "XOR" TESTER CHANGE
3781 ;; THIS INSTRUCTION TO A "NOP" (NOP=240)
3782 020750 013746 000004 MOV 2$ERRVEC, -(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
3783 020754 012737 020774 000004 MOV #5$,2$ERRVEC ;;SET FOR TIMEOUT
3784 020762 005737 177060 TST 2$177060 ;;TIME OUT ON XOR?
3785 020766 012637 000004 MOV (SP)+,2$ERRVEC ;;RESTORE THE ERROR VECTOR
3786 020772 000466 BR $SVLAD ;;GO TO THE NEXT TEST
3787 020774 022626 5$: CMP (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
3788 020776 012637 000004 MOV (SP)+,2$ERRVEC ;;RESTORE THE ERROR VECTOR
3789 021002 000426 BR 7$ ;;LOOP ON THE PRESENT TEST
3790 021004 6$;#####END OF CODE FOR THE XOR TESTER####
3791 021004 032777 000400 160126 BIT #BIT08,$SWR ;;LOOP ON SPEC. TEST?
3792 021012 001407 BEQ 2$ ;;BR IF NO
3793 021014 017746 160120 MOV 2$SWR, -(SP) ;;SET DESIRED TEST NUM. FROM SWR
3794 021020 042716 000200 BIC #5$SWR&K,(SP) ;;STRIP AWAY UNDESIRED BITS
3795 021024 122637 001102 CMPB (SP)+,$TSTNM ;;ON THE RIGHT TEST?
3796 021030 001462 BEQ $OVER ;;BR IF YES
3797 021032 105737 001103 2$: TSTB $ERFLG ;;HAS AN ERROR OCCURRED?
3798 021036 001421 BEQ 3$ ;;BR IF NO
3799 021040 123737 001115 001103 CMPB $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
3800 021046 101015 BHI 3$ ;;BR IF NO

```

```

3801 021050 032777 001000 160062 BIT #BIT09,JSWR ;; LOOP ON ERROR?
3802 021056 001404 BEQ 4S ;; BR IF NO
3803 021060 013737 001110 001106 7S: MOV $LPERR,$LPADR ;; SET LOOP ADDRESS TO LAST SCOPE
3804 021066 000443 BR $OVER
3805 021070 105037 001103 4S: CLRB $ERFLG ;; ZERO THE ERROR FLAG
3806 021074 005037 001162 CLR $TIMES ;; CLEAR THE NUMBER OF ITERATIONS TO MAKE
3807 021100 000415 BR 1S ;; ESCAPE TO THE NEXT TEST
3808 021102 032777 004000 160030 3S: BIT #BIT11,JSWR ;; INHIBIT ITERATIONS?
3809 021110 001011 BNE 1S ;; BR IF YES
3810 021112 005737 001100 TST $PASS ;; IF FIRST PASS OF PROGRAM
3811 021116 001406 BEQ 1S ;; INHIBIT ITERATIONS
3812 021120 005237 001104 INC $ICNT ;; INCREMENT ITERATION COUNT
3813 021124 023737 001162 001104 CMP $TIMES,$ICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
3814 021132 002021 BGE $OVER ;; BR IF MORE ITERATION REQUIRED
3815 021134 012737 000001 001104 1S: MOV #1,$ICNT ;; REINITIALIZE THE ITERATION COUNTER
3816 021142 013737 021212 001162 MOV $MXCNT,$TIMES ;; SET NUMBER OF ITERATIONS TO DO
3817 021150 105237 001102 $SVLAD: INCB $STNM ;; COUNT TEST NUMBERS
3818 021154 011637 001106 MOV (SP),$LPADR ;; SAVE SCOPE LOOP ADDRESS
3819 021160 011637 001110 MOV (SP),$LPERR ;; SAVE ERROR LOOP ADDRESS
3820 021164 005037 001164 CLR $ESCAPE ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
3821 021170 112737 000001 001115 MOVB #1,$ERMAX ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3822 021176 013777 001102 157736 $OVER: MOV $STNM,$DISPLAY ;; DISPLAY TEST NUMBER
3823 021204 013716 001106 MOV $LPADR,(SP) ;; FUDGE RETURN ADDRESS
3824 021210 000002 RTI ;; FIXES PS
3825 021212 000764 $MXCNT: 500. ;; MAX. NUMBER OF ITERATIONS

```

.SBTTL SAVE AND RESTORE R0-R5 ROUTINES

```

3826
3827
3828
3829
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

```

```

;*****
;SAVE R0-R5
;CALL:
; SAVREG
;UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
;
;TOP---(+16)
; +2---(+18)
; +4---R5
; +6---R4
; +8---R3
;+10---R2
;+12---R1
;+14---R0

```

```

$SAVREG:
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 22(SP),-(SP) ;; SAVE PS OF MAIN FLOW
MOV 22(SP),-(SP) ;; SAVE PC OF MAIN FLOW
MOV 22(SP),-(SP) ;; SAVE PS OF CALL
MOV 22(SP),-(SP) ;; SAVE PC OF CALL
RTI

```

```

3857 ;*RESTORE RO-R5
3858 ;*CALL:
3859 ;* RESREG
3860 $RESREG:
3861 MOV (SP)+,22(SP) ;;RESTORE PC OF CALL
3862 MOV (SP)+,22(SP) ;;RESTORE PS OF CALL
3863 MOV (SP)+,22(SP) ;;RESTORE PC OF MAIN FLOW
3864 MOV (SP)+,22(SP) ;;RESTORE PS OF MAIN FLOW
3865 MOV (SP)+,R5 ;;POP STACK INTO R5
3866 MOV (SP)+,R4 ;;POP STACK INTO R4
3867 MOV (SP)+,R3 ;;POP STACK INTO R3
3868 MOV (SP)+,R2 ;;POP STACK INTO R2
3869 MOV (SP)+,R1 ;;POP STACK INTO R1
3870 MOV (SP)+,R0 ;;POP STACK INTO R0
3871 RTI
    
```

.SBTTL TRAP DECODER

```

3872
3873
3874
3875 ;*****
3876 ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
3877 ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3878 ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
3879 ;*GO TO THAT ROUTINE.
    
```

```

3880
3881 $STRAP: MOV RO,-(SP) ;;SAVE RO
3882 MOV 2(SP),RO ;;GET TRAP ADDRESS
3883 TST -(RO) ;;BACKUP BY 2
3884 MOVB (RO),RO ;;GET RIGHT BYTE OF TRAP
3885 ASL RO ;;POSITION FOR INDEXING
3886 MOV $STRPAD(RO),RO ;;INDEX TO TABLE
3887 RTS RO ;;GO TO ROUTINE
    
```

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

```

3891
3892 $STRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
3893 MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
3894 RTI ;;RESTORE THE PSW
    
```

.SBTTL TRAP TABLE

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

```

3901 ; ROUTINE
3902 ;-----
3903 $STRPAD: .WORD $STRAP2
3904 $TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
3905 $TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3906 $TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
3907 $TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
3908 $TYPDS ;;CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
3909
3910 $GTSWR ;;CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING
3911
3912 $CKSWR ;;CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
    
```

3913	021364	020154		SRDCHR	::CALL=RDCHR	TRAP+10(104410)	TTY TYPEIN CHARACTER ROUTINE
3914	021366	020274		SRDLIN	::CALL=RDLIN	TRAP+11(104411)	TTY TYPEIN STRING ROUTINE
3915	021370	020574		SRDOCT	::CALL=RDOCT	TRAP+12(104412)	READ AN OCTAL NUMBER FROM TTY
3916	021372	021214		SSAVREG	::CALL=SAVREG	TRAP+13(104413)	SAVE RO-R5 ROUTINE
3917	021374	021252		SRESREG	::CALL=RESREG	TRAP+14(104414)	RESTORE RO-R5 ROUTINE

THIS ROUTINE IS USED TO ENSURE THAT THE BUS ADDRESS  
 OF THE RP11 IS SETUP TO READ THE PROPER VALUE.  
 IT WILL ALSO READ THE ADDRESS FROM THE TTY IF  
 REQUIRED.  
 NOTE: THIS ROUTINE DOES NOT PROTECT RO-R4

3925	021376	005737	001204	GETADR:	TST	BUSADR		INPUT FROM TTY REQUESTED?
3926	021402	001517			BEQ	5\$		NO--BRANCH
3927	021404	005037	001204		CLR	BUSADR		YES--CLEAR THE REQUEST FLAG
3928	021410			1\$:				
3929	021410	104401	021416		TYPE	65\$		TYPE ASCIZ STRING
3930	021414	000405			BR	64\$		GET OVER THE ASCIZ
3931				::65\$:	.ASCIZ	<15><12>/RPADR=		
3932	021430			64\$:				
3933	021430	013746	001210		MOV	RPADR,-(SP)		SAVE RPADR FOR TYPEOUT
3934								RP11 ADDRESS
3935	021434	104403			TYPOS			GO TYPE--OCTAL ASCII
3936	021436	006			.BYTE	6		TYPE 6 DIGIT(S)
3937	021437	001			.BYTE	1		TYPE LEADING ZEROS
3938	021440	104401	021446		TYPE	67\$		TYPE ASCIZ STRING
3939	021444	000401			BR	66\$		GET OVER THE ASCIZ
3940				::67\$:	.ASCIZ	a/a		
3941	021450			66\$:				
3942	021450	104412			RDOCT			GET NEW RP11 ADDRESS
3943	021452	012600			MOV	(SP)+,RO		SAVE THE ADDRESS
3944	021454	001402			BEQ	2\$		BR IF ZERO ENTRY OR A 'CR'
3945	021456	010037	001210		MOV	RO,RPADR		STORE THE ADDRESS
3946	021462			2\$:				
3947	021462	104401	021470		TYPE	69\$		TYPE ASCIZ STRING
3948	021466	000404			BR	68\$		GET OVER THE ASCIZ
3949				::69\$:	.ASCIZ	/RPVEC=		
3950	021500			68\$:				
3951	021500	013746	001212		MOV	RPVEC,-(SP)		SAVE RPVEC FOR TYPEOUT
3952								RP11 VECTOR ADDRESS
3953	021504	104403			TYPOS			GO TYPE--OCTAL ASCII
3954	021506	006			.BYTE	6		TYPE 6 DIGIT(S)
3955	021507	001			.BYTE	1		TYPE LEADING ZEROS
3956	021510	104401	021516		TYPE	71\$		TYPE ASCIZ STRING
3957	021514	000401			BR	70\$		GET OVER THE ASCIZ
3958				::71\$:	.ASCIZ	a/a		
3959	021520			70\$:				
3960	021520	104412			RDOCT			READ NEW RP11 VECTOR
3961	021522	012600			MOV	(SP)+,RO		STORE THE VECTOR ADDRESS
3962	021524	001405			BEQ	3\$		BR IF ZERO ENTRY OR 'CR'
3963	021526	010037	001212		MOV	RO,RPVEC		SAVE NEW RP11 VECTOR ADDRESS
3964	021532	005720			TST	(RO)+		ADD 2 TO RO.
3965	021534	010037	001214		MOV	RO,RPVEC+2		GET NEW PSW ADDRESS.
3966	021540			3\$:				
3967	021540	104401	021546		TYPE	73\$		TYPE ASCIZ STRING
3968	021544	000404			BR	72\$		GET OVER THE ASCIZ

```

3969
3970 021556
3971 021556 013700 001216
3972 021562 006300
3973 021564 006300
3974 021566 006300
3975 021570 000300
3976 021572 010046
3977
3978 021574 104403
3979 021576 001
3980 021577 001
3981 021600 104401 021606
3982 021604 000401
3983
3984 021610
3985 021610 104412
3986 021612 012600
3987 021614 001407
3988 021616 006300
3989 021620 006300
3990 021622 006300
3991 021624 006300
3992 021626 006300
3993 021630 010037 001216
3994 021634 013777 001216 157352 4$:
3995 021642 013701 000004 5$:
3996 021646 012737 021666 000004
3997 021654 005777 157330
3998 021660 010137 000004
3999 021664 000207
4000 021666 010137 000004 6$:
4001 021672 022626
4002 021674 104001
4003 021676 005737 000042
4004 021702 001642
4005 021704 005037 016312
4006 021710 000137 016162
4007
4008
4009
4010 021714 012737 060001 001206
4011 021722 012764 000001 000004
4012 021730 004737 021764
4013 021734 052764 020000 000022
4014 021742 000207
4015
4016
4017
4018 021744 012046
4019 021746 013764 001206 000022 1$:
4020 021754 005316
4021 021756 001373
4022 021760 005726
4023 021762 000200
4024

```

;:73\$: .ASCIZ /RPPRIO=  
 72\$: MOV RPPRIO,RO ;CONVERT PRIORITY FOR TYPEOUT  
 ASL RO  
 ASL RO  
 ASL RO  
 SWAB RO ;ALIGN FOR TYPEOUT  
 MOV RO,-(SP) ;SAVE RO FOR TYPEOUT  
 ;RP11 PRIORITY  
 ;GO TYPE--OCTAL ASCII  
 ;TYPE 1 DIGIT(S)  
 ;TYPE LEADING ZEROS  
 ;TYPE ASCIZ STRING  
 ;GET OVER THE ASCIZ  
 ;:75\$: .ASCIZ 2/2  
 74\$: RDOCT ;GET NEW PRIORITY LEVEL  
 MOV (SP)+,RO ;SAVE NEW PRIORITY  
 BEQ 4\$ ;BR IF ZERO ENTRY OR A 'CR'  
 ASL RO ;CONVERT TO PRIORITY VALUE  
 ASL RO ;CONVERT TO PRIORITY VALUE  
 ASL RO ;CONVERT TO PRIORITY VALUE  
 ASL RO ;CONVERT TO PRIORITY VALUE  
 ASL RO ;CONVERT TO PRIORITY VALUE  
 MOV RO,RPPRIO ;SAVE THE VALUE  
 MOV RPPRIO,ARPVEC+2 ;LOAD NEW PRIORITY VALUE  
 5\$: MOV ERRVEC,R1 ;SAVE THE ERROR VECTOR  
 MOV #6\$,ERRVEC ;SETUP FOR TRAP  
 TST ARPADR ;CHECK FOR RP11  
 MOV R1,ERRVEC ;RESTORE ERROR VECTOR  
 RTS PC ;RETURN  
 6\$: MOV R1,ERRVEC ;RESTORE ERROR VECTOR  
 CMP (SP)+,(SP)+ ;CLEAN OFF THE STACK  
 ERROR 1 ;REPORT THE ERROR  
 TST #42 ;IS THERE A MONITOR?  
 BEQ 1\$ ;NO--GO ASK FOR ADDRESS AGAIN  
 CLR \$EOPCT ;NO PASSES  
 JMP \$EOP ;GO TO END OF PROGRAM

;ROUTINE TO CLEAR THE RP11E IN MAINTENANCE MODE  
 CLR\$: MOV #BIT14!BIT13!BIT00,TPL  
 MOV #1,RPC5(R4) ;CLEAR THE CONTROLLER  
 JSR PC,T3P ;GENERATE 3 CLOCK PULSES  
 BIS #BIT13,RPM3(R4) ;SET 'SUOL'  
 RTS PC ;RETURN

;GENERATE CLOCK PULSES  
 TIMEP: MOV (RO)+,-(SP) ;GET NUMBER OF CLOCK PULSES  
 1\$: MOV TPL,RPM3(R4) ;SET SPECIFIED MAINTENANCE BITS  
 DEC (SP) ;DECREMENT THE COUNT  
 BNE 1\$ ;BR IF MORE CLOCK PULSES  
 TST (SP)+ ;RESTORE THE STACK POINTER  
 RTS RO ;RETURN

```

4025 ;ROUTINE TO GENERATE 3 CLOCK PULSES
4026
4027 021764 004037 021744 T3P: JSR RO,TIMEP ;GENERATE 3 CLOCK PULSES
4028 021770 000003 ;WORD 3 ;CONSTANT FOR 3 CLOCK PULSES
4029 021772 000207 RTS PC ;RETURN
4030
4031 ;GENERATE AN INDEX PULSE
4032
4033 021774 012046 INDEXP: MOV (RO)+,-(SP) ;GET THE INDEX PULSE COUNT
4034 021776 012764 030000 000022 MOV #BIT13:BIT12,RPM3(R4) ;GENERATE AN INDEX PULSE
4035 022004 012764 020000 000022 MOV #BIT13,RPM3(R4)
4036 022012 005316 DEC (SP) ;DECREMENT THE INDEX PULSE COUNTER
4037 022014 001367 BNE INDEXP ;BR IF NOT DONE
4038 022016 005726 TST (SP)+ ;RESTORE THE STACK
4039 022020 000200 RTS RO ;RETURN
4040
4041 ;ROUTINE TO SAVE THE RP11E REGISTERS FOR USE BY THE PROGRAM.
4042 ; THE PROGRAM DOES NOT USE THE 'LIVE' REGISTERS FOR ERROR
4043 ; DETERMINATION.
4044
4045 SAVRP:
4046 022022 016437 000000 001220 MOV RPDS(R4),$RPDS ;STORE RPDS
4047 022030 016437 000002 001222 MOV RPER(R4),$RPER ;STORE RPER
4048 022036 016437 000004 001224 MOV RPCS(R4),$RPCS ;STORE RPCS
4049 022044 016437 000006 001226 MOV RPWC(R4),$RPWC ;STORE RPWC
4050 022052 016437 000010 001230 MOV RPBA(R4),$RPBA ;STORE RPBA
4051 022060 016437 000012 001232 MOV RPCA(R4),$RPCA ;STORE RPCA
4052 022066 016437 000014 001234 MOV RPDA(R4),$RPDA ;STORE RPDA
4053 022074 016437 000016 001236 MOV RPM1(R4),$RPM1 ;STORE RPM1
4054 022102 016437 000024 001240 MOV SUCA(R4),$SUCA ;SAVE SUCA
4055 022110 016437 000026 001242 MOV SILO(R4),$SILO ;SAVE SILO
4056 022116 000207 RTS PC ;RETURN
4057
4058 ;MESSAGES USED BY THE PROGRAM
4059
4060 022120 020040 000 BLNKS2: .ASCIZ / /
4061
4062 022123 015 005012 047516 DRVOL: .ASCII <15><12><12>@NOTE: @
4063 022130 042524 072
4064 022133 015 005012 044124 .ASCII <15><12><12>@THE MAINTENANCE SWITCH ON THE RP11E MUST BE @
4065 022140 020105 040515 047111
4066 022146 042524 040516 041516
4067 022154 020105 053523 052111
4068 022162 044103 047440 020116
4069 022170 044124 020105 050122
4070 022176 030461 020105 052515
4071 022204 052123 041040 105
4072 022211 015 051412 052105 .ASCII <15><12><12>@SET TO 'ENABLE'. @
4073 022216 052040 020117 042447
4074 022224 040516 046102 023505
4075 022232 056
4076 022233 015 005012 051104 .ASCII <15><12><12>@DRIVE 0 MUST BE CYCLED DOWN OR TURNED OFF FOR THE @
4077 022240 053111 020105 020060
4078 022246 052515 052123 041040
4079 022254 020105 054503 046103
4080 022262 042105 042040 053517

```



4081	022270	020116	051117	052040	
4082	022276	051125	042516	020104	
4083	022304	043117	020106	047506	
4084	022312	020122	044124	105	
4085	022317	015	050012	047522	.ASCII <15><12>@PROGRAM TO FUNCTION PROPERLY. IF DRIVE 0 IS CYCLED@
4086	022324	051107	046501	052040	
4087	022332	020117	052506	041516	
4088	022340	044524	047117	050040	
4089	022346	047522	042520	046122	
4090	022354	027131	020040	043111	
4091	022362	042040	044522	042526	
4092	022370	030040	044440	020123	
4093	022376	054503	046103	042105	
4094	022404	005015	050125	020054	.ASCII <15><12>@UP, STOP THE PROGRAM, CYCLE DOWN THE DRIVE, AND@
4095	022412	052123	050117	052040	
4096	022420	042510	050040	047522	
4097	022426	051107	046501	020054	
4098	022434	054503	046103	020105	
4099	022442	047504	047127	052040	
4100	022450	042510	042040	044522	
4101	022456	042526	020054	047101	
4102	022464	104			
4103	022465	015	051012	051505	.ASCIZ <15><12>@RESTART THE PROGRAM@<15><12><12>
4104	022472	040524	052122	052040	
4105	022500	042510	050040	047522	
4106	022506	051107	046501	005015	
4107	022514	000012			
4108					
4109					.EVEN
4110					
4111					.SBTTL ERROR MESSAGES
4112					
4113	022516	050122	030461	042040	EM1: .ASCIZ @RP11 DIDN'T RESPOND TO ADDRESSING@
4114	022524	042111	023516	020124	
4115	022532	042522	050123	047117	
4116	022540	020104	047524	040440	
4117	022546	042104	042522	051523	
4118	022554	047111	000107		
4119	022560	040503	023516	020124	EM2: .ASCIZ @CAN'T LOAD REGISTER CORRECTLY@
4120	022566	047514	042101	051040	
4121	022574	043505	051511	042524	
4122	022602	020122	047503	051122	
4123	022610	041505	046124	000131	
4124	022616	042522	042523	020124	EM3: .ASCIZ @RESET DIDN'T CLEAR REGISTER@
4125	022624	044504	047104	052047	
4126	022632	041440	042514	051101	
4127	022640	051040	043505	051511	
4128	022646	042524	000122		
4129	022652	047503	052116	047522	EM4: .ASCIZ @CONTROLLER CLEAR DIDN'T CLEAR REGISTER@
4130	022660	046114	051105	041440	
4131	022666	042514	051101	042040	
4132	022674	042111	023516	020124	
4133	022702	046103	040505	020122	
4134	022710	042522	044507	052123	
4135	022716	051105	000		
4136	022721	106	047514	052101	EM5: .ASCIZ @FLOATING 1'S & 0'S TEST ERROR@

4137	022726	047111	020107	023461	
4138	022734	020123	020046	023460	
4139	022742	020123	042524	052123	
4140	022750	042440	051122	051117	
4141	022756	000			
4142	022757	122	043505	051511	EM6: .ASCIZ @REGISTER RAPID ACCESS TEST ERROR@
4143	022764	042524	020122	040522	
4144	022772	044520	020104	041501	
4145	023000	042503	051523	052040	
4146	023006	051505	020124	051105	
4147	023014	047522	000122		
4148	023020	040503	023516	020124	EM7: .ASCIZ @CAN'T SET 'SURDY' BIT@
4149	023026	042523	020124	051447	
4150	023034	051125	054504	020047	
4151	023042	044502	000124		
4152	023046	040503	023516	020124	EM10: .ASCIZ @CAN'T CLEAR THE 'SURDY' BIT@
4153	023054	046103	040505	020122	
4154	023062	044124	020105	051447	
4155	023070	051125	054504	020047	
4156	023076	044502	000124		
4157	023102	040503	023516	020124	EM11: .ASCIZ @CAN'T SET 'SUSI'@
4158	023110	042523	020124	051447	
4159	023116	051525	023511	000	
4160	023123	047	051504	020113	EM12: .ASCIZ @'DSK ERR' NOT SET WITH 'SUSI'@
4161	023130	051105	023522	047040	
4162	023136	052117	051440	052105	
4163	023144	053440	052111	020110	
4164	023152	051447	051525	023511	
4165	023160	000			
4166	023161	047	051105	023522	EM13: .ASCIZ @'ERR' OR 'HE' NOT SET WITH 'SUSI'@
4167	023166	047440	020122	044047	
4168	023174	023505	047040	052117	
4169	023202	051440	052105	053440	
4170	023210	052111	020110	051447	
4171	023216	051525	023511	000	
4172	023223	103	047101	052047	EM14: .ASCIZ @CAN'T SET 'SUFU'@
4173	023230	051440	052105	023440	
4174	023236	052523	052506	000047	
4175	023244	040503	023516	020124	EM15: .ASCIZ @CAN'T CLEAR 'SUFU'@
4176	023252	046103	040505	020122	
4177	023260	051447	043125	023525	
4178	023266	000			
4179	023267	103	047101	052047	EM16: .ASCIZ @CAN'T SET 'SUWP'@
4180	023274	051440	052105	023440	
4181	023302	052523	050127	000047	
4182	023310	040503	023516	020124	EM17: .ASCIZ @CAN'T CLEAR 'SUWP'@
4183	023316	046103	040505	020122	
4184	023324	051447	053525	023520	
4185	023332	000			
4186	023333	103	047101	052047	EM20: .ASCIZ @CAN'T SET ATTENTION BIT@
4187	023340	051440	052105	040440	
4188	023346	052124	047105	044524	
4189	023354	047117	041040	052111	
4190	023362	000			
4191	023363	103	047101	052047	EM21: .ASCIZ @CAN'T CLEAR ATTENTION BIT@
4192	023370	041440	042514	051101	

4193	023376	040440	052124	047105	
4194	023404	044524	047117	041040	
4195	023412	052111	000		
4196	023415	122	051505	052105	EM22: .ASCIZ @RESET DIDN'T CLEAR ATTENTION BITS@
4197	023422	042040	042111	023516	
4198	023430	020124	046103	040505	
4199	023436	020122	052101	042524	
4200	023444	052116	047511	020116	
4201	023452	044502	051524	000	
4202	023457	101	052124	047105	EM23: .ASCIZ @ATTENTION BITS CLEARED BY WRITING 0'S INTO RPDS@
4203	023464	044524	047117	041040	
4204	023472	052111	020123	046103	
4205	023500	040505	042522	020104	
4206	023506	054502	053440	044522	
4207	023514	044524	043516	030040	
4208	023522	051447	044440	052116	
4209	023530	020117	050122	051504	
4210	023536	000			
4211	023537	103	047101	052047	EM24: .ASCIZ @CAN'T SET 'WPV' ERROR@
4212	023544	051440	052105	023440	
4213	023552	050127	023526	042440	
4214	023560	051122	051117	000	
4215	023565	047	051105	023522	EM25: .ASCIZ @'ERR' OR 'HE' NOT SET WITH 'WPV'@
4216	023572	047440	020122	044047	
4217	023600	023505	047040	052117	
4218	023606	051440	052105	053440	
4219	023614	052111	020110	053447	
4220	023622	053120	000047		
4221	023626	040503	023516	020124	EM26: .ASCIZ @CAN'T SET 'FUV' ERROR@
4222	023634	042523	020124	043047	
4223	023642	053125	020047	051105	
4224	023650	047522	000122		
4225	023654	042447	051122	020047	EM27: .ASCIZ @'ERR' OR 'HE' NOT SET WITH 'FUV'@
4226	023662	051117	023440	042510	
4227	023670	020047	047516	020124	
4228	023676	042523	020124	044527	
4229	023704	044124	023440	052506	
4230	023712	023526	000		
4231	023715	047	054116	023503	EM30: .ASCIZ @'NXC' ERROR SET WITH VALID CYLINDER ADDRESS@
4232	023722	042440	051122	051117	
4233	023730	051440	052105	053440	
4234	023736	052111	020110	040526	
4235	023744	044514	020104	054503	
4236	023752	044514	042116	051105	
4237	023760	040440	042104	042522	
4238	023766	051523	000		
4239	023771	047	054116	023503	EM31: .ASCIZ @'NXC' DIDN'T SET WITH INVALID CYLINDER ADDRESS@
4240	023776	042040	042111	023516	
4241	024004	020124	042523	020124	
4242	024012	044527	044124	044440	
4243	024020	053116	046101	042111	
4244	024026	041440	046131	047111	
4245	024034	042504	020122	042101	
4246	024042	051104	051505	000123	
4247	024050	042447	051122	020047	EM32: .ASCIZ @'ERR' OR 'HE' DIDN'T SET WITH 'NXC'@
4248	024056	051117	023440	042510	

4249	024064	020047	044504	047104	
4250	024072	052047	051440	052105	
4251	024100	053440	052111	020110	
4252	024106	047047	041530	000047	
4253	024114	047047	052130	020047	EM33: .ASCIZ @'NXT' ERROR SET WITH VALID TRACK ADDRESS@
4254	024122	051105	047522	020122	
4255	024130	042523	020124	044527	
4256	024136	044124	053040	046101	
4257	024144	042111	052040	040522	
4258	024152	045503	040440	042104	
4259	024160	042522	051523	000	
4260	024165	047	054116	023524	EM34: .ASCIZ @'NXT' DIDN'T SET WITH INVALID TRACK ADDRESS@
4261	024172	042040	042111	023516	
4262	024200	020124	042523	020124	
4263	024206	044527	044124	044440	
4264	024214	053116	046101	042111	
4265	024222	052040	040522	045503	
4266	024230	040440	042104	042522	
4267	024236	051523	000		
4268	024241	047	051105	023522	EM35: .ASCIZ @'ERR' OR 'HE' DIDN'T SET WITH 'NXT'@
4269	024246	047440	020122	044047	
4270	024254	023505	042040	042111	
4271	024262	023516	020124	042523	
4272	024270	020124	044527	044124	
4273	024276	023440	054116	023524	
4274	024304	000			
4275	024305	047	054116	023523	EM36: .ASCIZ @'NXS' ERROR SET WITH VALID SECTOR ADDRESS@
4276	024312	042440	051122	051117	
4277	024320	051440	052105	053440	
4278	024326	052111	020110	040526	
4279	024334	044514	020104	042523	
4280	024342	052103	051117	040440	
4281	024350	042104	042522	051523	
4282	024356	000			
4283	024357	047	054116	023523	EM37: .ASCIZ @'NXS' DIDN'T SET WITH INVALID SECTOR ADDRESS@
4284	024364	042040	042111	023516	
4285	024372	020124	042523	020124	
4286	024400	044527	044124	044440	
4287	024406	053116	046101	042111	
4288	024414	051440	041505	047524	
4289	024422	020122	042101	051104	
4290	024430	051505	000123		
4291	024434	042447	051122	020047	EM40: .ASCIZ @'ERR' OR 'HE' DIDN'T SET WITH 'NXS'@
4292	024442	051117	023440	042510	
4293	024450	020047	044504	047104	
4294	024456	052047	051440	052105	
4295	024464	053440	052111	020110	
4296	024472	047047	051530	000047	
4297	024500	040503	023516	020124	EM41: .ASCIZ @CAN'T SET 'PROG' ERROR@
4298	024506	042523	020124	050047	
4299	024514	047522	023507	042440	
4300	024522	051122	051117	000	
4301	024527	047	051105	023522	EM42: .ASCIZ @'ERR' OR 'HE' NOT SET WITH 'PROG'@
4302	024534	047440	020122	044047	
4303	024542	023505	047040	052117	
4304	024550	051440	052105	053440	

4305	024556	052111	020110	050047	
4306	024564	047522	023507	000	
4307	024571	103	047101	052047	EM43: .ASCIZ @CAN'T SET 'MODE' ERROR@
4308	024576	051440	052105	023440	
4309	024604	047515	042504	020047	
4310	024612	051105	047522	000122	
4311	024620	042447	051122	020047	EM44: .ASCIZ @'ERR' OR 'HE' NOT SET WITH 'MODE'@
4312	024626	051117	023440	042510	
4313	024634	020047	047516	020124	
4314	024642	042523	020124	044527	
4315	024650	044124	023440	047515	
4316	024656	042504	000047		
4317	024662	040503	023516	020124	EM45: .ASCIZ @CAN'T CLEAR 'SUSI'@
4318	024670	046103	040505	020122	
4319	024676	051447	051525	023511	
4320	024704	000			
4321	024705	116	020117	052101	EM46: .ASCIZ @NO ATTENTION INTERRUPT@
4322	024712	042524	052116	047511	
4323	024720	020116	047111	042524	
4324	024726	051122	050125	000124	
4325	024734	040447	042511	020047	EM47: .ASCIZ @'AIE' DIDN'T CLEAR WHEN INTERRUPT OCCURED@
4326	024742	044504	047104	052047	
4327	024750	041440	042514	051101	
4328	024756	053440	042510	020116	
4329	024764	047111	042524	051122	
4330	024772	050125	020124	041517	
4331	025000	052503	042522	000104	
4332	025006	047516	044440	052116	EM50: .ASCIZ @NO INTERRUPT WITH ATTENTION BITS 0 & 1 SET@
4333	025014	051105	052522	052120	
4334	025022	053440	052111	020110	
4335	025030	052101	042524	052116	
4336	025036	047511	020116	044502	
4337	025044	051524	030040	023040	
4338	025052	030440	051440	052105	
4339	025060	000			
4340	025061	123	041505	047117	EM51: .ASCIZ @SECOND INTERRUPT DIDN'T OCCUR WITH ATTENTION BIT 1 SET@
4341	025066	020104	047111	042524	
4342	025074	051122	050125	020124	
4343	025102	044504	047104	052047	
4344	025110	047440	041503	051125	
4345	025116	053440	052111	020110	
4346	025124	052101	042524	052116	
4347	025132	047511	020116	044502	
4348	025140	020124	020061	042523	
4349	025146	000124			
4350	025150	047516	044440	052116	EM52: .ASCIZ @NO INTERRUPT FROM CONTROLLER 'READY'@
4351	025156	051105	052522	052120	
4352	025164	043040	047522	020115	
4353	025172	047503	052116	047522	
4354	025200	046114	051105	023440	
4355	025206	042522	042101	023531	
4356	025214	000			
4357	025215	047	042522	042101	EM53: .ASCIZ @'READY' INTERRUPT WITH 'IDE' SET@
4358	025222	023531	044440	052116	
4359	025230	051105	052522	052120	
4360	025236	053440	052111	020110	

H07

4361	025244	044447	042504	020047	
4362	025252	042523	000124		
4363	025256	047516	044440	052116	EM54: .ASCIZ @NO INTERRUPT FROM RP11 WITH CPU AT PR4@
4364	025264	051105	052522	052120	
4365	025272	043040	047522	020115	
4366	025300	050122	030461	053440	
4367	025306	052111	020110	050103	
4368	025314	020125	052101	050040	
4369	025322	032122	000		
4370	025325	111	052116	051105	EM55: .ASCIZ @INTERRUPT FROM RP11 WITH CPU AT PR5@
4371	025332	052522	052120	043040	
4372	025340	047522	020115	050122	
4373	025346	030461	053440	052111	
4374	025354	020110	050103	020125	
4375	025362	052101	050040	032522	
4376	025370	000			
4377	025371	111	052116	051105	EM56: .ASCIZ @INTERRUPT FROM RP11 WITH CPU AT PR6@
4378	025376	052522	052120	043040	
4379	025404	047522	020115	050122	
4380	025412	030461	053440	052111	
4381	025420	020110	050103	020125	
4382	025426	052101	050040	033122	
4383	025434	000			

4384	025435	111	052116	051105	EM57:	.ASCIZ	QINTERRUPT FROM RP11 WITH CPU AT PR7Q
4385	025442	052522	052120	043040			
4386	025450	047522	020115	050122			
4387	025456	030461	053440	052111			
4388	025464	020110	050103	020125			
4389	025472	052101	050040	033522			
4390	025500	000					
4391	025501	047	042522	042101	EM60:	.ASCIZ	Q'READY' DIDN'T CLEAR AT END OF OPERATIONQ
4392	025506	023531	042040	042111			
4393	025514	023516	020124	046103			
4394	025522	040505	020122	052101			
4395	025530	042440	042116	047440			
4396	025536	020106	050117	051105			
4397	025544	052101	047511	000116			
4398	025552	051447	041525	023501	EM61:	.ASCIZ	Q'SUCA' INCORRECTQ
4399	025560	044440	041516	051117			
4400	025566	042522	052103	000			
4401	025573	111	042116	054105	EM62:	.ASCIZ	QINDEX DIDN'T CLEAR 'SOT'Q
4402	025600	042040	042111	023516			
4403	025606	020124	046103	040505			
4404	025614	020122	051447	052117			
4405	025622	000047					
4406	025624	051447	052117	020047	EM63:	.ASCIZ	Q'SOT' DIDN'T COUNT CORRECTLYQ
4407	025632	044504	047104	052047			
4408	025640	041440	052517	052116			
4409	025646	041440	051117	042522			
4410	025654	052103	054514	000			
4411	025661	103	047117	051124	EM64:	.ASCIZ	QCONTROLLER CLEAR DIDN'T SET SILO INPUT READYQ
4412	025666	046117	042514	020122			
4413	025674	046103	040505	020122			
4414	025702	044504	047104	052047			
4415	025710	051440	052105	051440			
4416	025716	046111	020117	047111			
4417	025724	052520	020124	042522			
4418	025732	042101	000131				
4419	025736	044523	047514	047440	EM65:	.ASCIZ	QSILO OUTPUT READY NOT SET AFTER WORD LOADED INTO SILOQ
4420	025744	052125	052520	020124			
4421	025752	042522	042101	020131			
4422	025760	047516	020124	042523			
4423	025766	020124	043101	042524			
4424	025774	020122	047527	042122			
4425	026002	046040	040517	042504			
4426	026010	020104	047111	047524			
4427	026016	051440	046111	000117			
4428	026024	040504	040524	051040	EM66:	.ASCIZ	QDATA READ FROM SILO IS INCORRECTQ
4429	026032	040505	020104	051106			
4430	026040	046517	051440	046111			
4431	026046	020117	051511	044440			
4432	026054	041516	051117	042522			
4433	026062	052103	000				
4434	026065	123	046111	020117	EM67:	.ASCII	QSILO INPUT READY DIDN'T CLEAR AFTER 64 WORDS WEREQ
4435	026072	047111	052520	020124			
4436	026100	042522	042101	020131			
4437	026106	044504	047104	052047			
4438	026114	041440	042514	051101			
4439	026122	040440	052106	051105			

4440	026130	033040	020064	047527	
4441	026136	042122	020123	042527	
4442	026144	042522			
4443	026146	005015	047514	042101	.ASCIZ <15><12>@LOADED INTO THE SILO@
4444	026154	042105	044440	052116	
4445	026162	020117	044124	020105	
4446	026170	044523	047514	000	
4447	026175	123	046111	020117	EM70: .ASCIZ @SILO OUTPUT READY DIDN'T CLEAR AFTER SILO EMPTIED@
4448	026202	052517	050124	052125	
4449	026210	051040	040505	054504	
4450	026216	042040	042111	023516	
4451	026224	020124	046103	040505	
4452	026232	020122	043101	042524	
4453	026240	020122	044523	047514	
4454	026246	042440	050115	044524	
4455	026254	042105	000		
4456	026257	102	051525	047440	EM71: .ASCIZ @BUS OUT LINES TO THE DRIVE WERE NOT CLEARED BY CONTROLLER CLEAR@
4457	026264	052125	046040	047111	
4458	026272	051505	052040	020117	
4459	026300	044124	020105	051104	
4460	026306	053111	020105	042527	
4461	026314	042522	047040	052117	
4462	026322	041440	042514	051101	
4463	026330	042105	041040	020131	
4464	026336	047503	052116	047522	
4465	026344	046114	051105	041440	
4466	026352	042514	051101	000	
4467	026357	104	044522	042526	EM72: .ASCII @DRIVE BUS ERROR; 'SET CYLINDER' AND CYLINDER ADDRESS@
4468	026364	041040	051525	042440	
4469	026372	051122	051117	020073	
4470	026400	051447	052105	041440	
4471	026406	046131	047111	042504	
4472	026414	023522	040440	042116	
4473	026422	041440	046131	047111	
4474	026430	042504	020122	042101	
4475	026436	051104	051505	123	
4476	026443	015	051412	047510	.ASCIZ <15><12>@SHOULD BE ON THE BUS@
4477	026450	046125	020104	042502	
4478	026456	047440	020116	044124	
4479	026464	020105	052502	000123	
4480	026472	051104	053111	020105	EM73: .ASCIZ @DRIVE BUS ERROR: ONLY 'RESET HEAD' SHOULD BE ON THE BUS@
4481	026500	052502	020123	051105	
4482	026506	047522	035122	047440	
4483	026514	046116	020131	051047	
4484	026522	051505	052105	044040	
4485	026530	040505	023504	051440	
4486	026536	047510	046125	020104	
4487	026544	042502	047440	020116	
4488	026552	044124	020105	052502	
4489	026560	000123			
4490	026562	051104	053111	020105	EM74: .ASCIZ @DRIVE BUS ERROR: 'SET HEAD' & HEAD ADDRESS SHOULD BE ON THE BUS@
4491	026570	052502	020123	051105	
4492	026576	047522	035122	023440	
4493	026604	042523	020124	042510	
4494	026612	042101	020047	020046	
4495	026620	042510	042101	040440	



4496	026626	042104	042522	051523	
4497	026634	051440	047510	046125	
4498	026642	020104	042502	047440	
4499	026650	020116	044124	020105	
4500	026656	052502	000123		
4501	026663	051104	053111	020105	EM75: .ASCIZ @DRIVE BUS ERROR: ONLY 'SEEK START' SHOULD BE ON THE BUS@
4502	026670	052502	020123	051105	
4503	026676	047522	035122	047440	
4504	026704	046116	020131	051447	
4505	026712	042505	020113	052123	
4506	026720	051101	023524	051440	
4507	026726	047510	046125	020104	
4508	026734	042502	047440	020116	
4509	026742	044124	020105	052502	
4510	026750	000123			
4511	026752	051104	053111	020105	EM76: .ASCIZ @DRIVE BUS ERROR: 'RESTORE' AND 'CONTROL' SHOULD BE ON THE BUS@
4512	026760	052502	020123	051105	
4513	026766	047522	035122	023440	
4514	026774	042522	052123	051117	
4515	027002	023505	040440	042116	
4516	027010	023440	047503	052116	
4517	027016	047522	023514	051440	
4518	027024	047510	046125	020104	
4519	027032	042502	047440	020116	
4520	027040	044124	020105	052502	
4521	027046	000123			
4522	027050	051104	053111	020105	EM77: .ASCII @DRIVE BUS ERROR: 'CONTROL', 'SELECT HEAD', AND 'READ'@
4523	027056	052502	020123	051105	
4524	027064	047522	035122	023440	
4525	027072	047503	052116	047522	
4526	027100	023514	020054	051447	
4527	027106	046105	041505	020124	
4528	027114	042510	042101	026047	
4529	027122	040440	042116	023440	
4530	027130	042522	042101	047	
4531	027135	015	051412	047510	.ASCIZ <15><12>@SHOULD BE ON THE BUS@
4532	027142	046125	020104	042502	
4533	027150	047440	020116	044124	
4534	027156	020105	052502	000123	
4535	027164	051104	053111	020105	EM100: .ASCII @DRIVE BUS ERROR: 'CONTROL', SELECT HEAD', 'ERASE',@
4536	027172	052502	020123	051105	
4537	027200	047522	035122	023440	
4538	027206	047503	052116	047522	
4539	027214	023514	020054	042523	
4540	027222	042514	052103	044040	
4541	027230	040505	023504	020054	
4542	027236	042447	040522	042523	
4543	027244	026047			
4544	027246	005015	047101	020104	.ASCIZ <15><12>@AND 'WRITE' SHOULD BE ON THE BUS@
4545	027254	053447	044522	042524	
4546	027262	020047	044123	052517	
4547	027270	042114	041040	020105	
4548	027276	047117	052040	042510	
4549	027304	041040	051525	000	
4550	027311	047	042522	042101	EM101: .ASCIZ @'READY' DIDN'T CLEAR WHEN 'GO' WAS SET@
4551	027316	023531	042040	042111	

4552	027324	023516	020124	046103	
4553	027332	040505	020122	044127	
4554	027340	047105	023440	047507	
4555	027346	020047	040527	020123	
4556	027354	042523	000124		
4557	027360	040503	023516	020124	EM102: .ASCIZ @CAN'T SET 'SUOL'@
4558	027366	042523	020124	051447	
4559	027374	047525	023514	000	
4560	027401	103	047101	052047	EM103: .ASCIZ @CAN'T CLEAR 'SUOL'@
4561	027406	041440	042514	051101	
4562	027414	023440	052523	046117	
4563	027422	000047			
4564	027424	040503	023516	020124	EM104: .ASCIZ @CAN'T SET 'SUSU'@
4565	027432	042523	020124	051447	
4566	027440	051525	023525	000	
4567	027445	103	047101	052047	EM105: .ASCIZ @CAN'T CLEAR 'SUSU'@
4568	027452	041440	042514	051101	
4569	027460	023440	052523	052523	
4570	027466	000047			
4571	027470	044447	042504	020047	EM106: .ASCIZ @'IDE' NOT SET AFTER INTERRUPT@
4572	027476	047516	020124	042523	
4573	027504	020124	043101	042524	
4574	027512	020122	047111	042524	
4575	027520	051122	050125	000124	
4576	027526	052101	047124	044440	EM107: .ASCIZ @ATTN INTERRUPT OCCURED WITH NO ATTN BITS SET@
4577	027534	052116	051105	052522	
4578	027542	052120	047440	041503	
4579	027550	051125	042105	053440	
4580	027556	052111	020110	047516	
4581	027564	040440	052124	020116	
4582	027572	044502	051524	051440	
4583	027600	052105	000		
4584	027603	123	046111	020117	EM110: .ASCIZ @SILO NOT FULL, 'INPUT READY' NOT SET@
4585	027610	047516	020124	052506	
4586	027616	046114	020054	044447	
4587	027624	050116	052125	051040	
4588	027632	040505	054504	020047	
4589	027640	047516	020124	042523	
4590	027646	000124			
4591	027650	044523	047514	047040	EM111: .ASCIZ @SILO NOT EMPTY, 'OUTPUT READY' NOT SET@
4592	027656	052117	042440	050115	
4593	027664	054524	020054	047447	
4594	027672	052125	052520	020124	
4595	027700	042522	042101	023531	
4596	027706	047040	052117	051440	
4597	027714	052105	000		
4598	027717	047	047111	052111	EM112: .ASCIZ @'INIT' COMMAND SET 'PROG' ERROR@
4599	027724	020047	047503	046515	
4600	027732	047101	020104	042523	
4601	027740	020124	050047	047522	
4602	027746	023507	042440	051122	
4603	027754	051117	000		
4604					
4605					;ERROR HEADER (DH) MESSAGES
4606					
4607	027757	105	051122	050040	DH1: .ASCIZ @ERR PC RPADR@





4720	031134	040504	020040	020040									
4721	031142	050122	030515	000									
4722	031147	124	051505	020124	DH72:	.ASCIZ	QTEST #	ERR PC	RPDS	RPCS	RPCA	RPM1Q	
4723	031154	020043	042440	051122									
4724	031162	050040	020103	051040									
4725	031170	042120	020123	020040									
4726	031176	051040	041520	020123									
4727	031204	020040	051040	041520									
4728	031212	020101	020040	051040									
4729	031220	046520	000061										
4730	031224	042524	052123	021440	DH73:	.ASCIZ	QTEST #	ERR PC	RPDS	RPCS	RPM1	EXPTD BUSQ	
4731	031232	020040	051105	020122									
4732	031240	041520	020040	050122									
4733	031246	051504	020040	020040									
4734	031254	050122	051503	020040									
4735	031262	020040	050122	030515									
4736	031270	042440	050130	042124									
4737	031276	041040	051525	000									
4738	031303	124	051505	020124	DH74:	.ASCIZ	QTEST #	ERR PC	RPDS	RPCS	RPDA	RPM1Q	
4739	031310	020043	042440	051122									
4740	031316	050040	020103	051040									
4741	031324	042120	020123	020040									
4742	031332	051040	041520	020123									
4743	031340	020040	051040	042120									
4744	031346	020101	020040	051040									
4745	031354	046520	000061										
4746													
4747													
4748													
4749													
4750													
4751	031360	001116	001210		DT1:	.WORD		SERRPC,	RPADR				
4752	031364	001160	001116	001122	DT2:	.WORD		STMPO,	SERRPC,	SBDADR,	SGDDAT,	SBDDAT	
4753	031372	001124	001126										
4754	031376	001160	001116	001220	DT7:	.WORD		STMPO,	SERRPC,	SRPDS			
4755	031404	001160	001116	001220	DT11:	.WORD		STMPO,	SERRPC,	SRPDS,	SRPER,	SRPCS	
4756	031412	001222	001224										
4757	031416	001160	001116	001124	DT20:	.WORD		STMPO,	SERRPC,	SGDDAT,	SRPDS		
4758	031424	001220											
4759	031426	001160	001116	001222	DT24:	.WORD		STMPO,	SERRPC,	SRPER,	SRPCS		
4760	031434	001224											
4761	031436	001160	001116	001222	DT30:	.WORD		STMPO,	SERRPC,	SRPER,	SRPCS,	SRPCA,	SGDDAT
4762	031444	001224	001232	001124									
4763	031452	001160	001116	001222	DT33:	.WORD		STMPO,	SERRPC,	SRPER,	SRPCS,	SRPDA,	SGDDAT
4764	031460	001224	001234	001124									
4765	031466	001160	001116	001220	DT46:	.WORD		STMPO,	SERRPC,	SRPDS,	SRPCS,	SGDDAT	
4766	031474	001224	001124										
4767	031500	001160	001116	001220	DT50:	.WORD		STMPO,	SERRPC,	SRPDS,	SRPCS		
4768	031506	001224											
4769	031510	001160	001116	001224	DT52:	.WORD		STMPO,	SERRPC,	SRPCS			
4770	031516	001160	001116	001240	DT61:	.WORD		STMPO,	SERRPC,	SSUCA,	SGDDAT		
4771	031524	001124											
4772	031526	001160	001116	001126	DT62:	.WORD		STMPO,	SERRPC,	SBDDAT			
4773	031534	001160	001116	001126	DT63:	.WORD		STMPO,	SERRPC,	SBDDAT,	SGDDAT		
4774	031542	001124											
4775	031544	001160	001116	001236	DT64:	.WORD		STMPO,	SERRPC,	SRPM1,	SSILO		

.EVEN

;ERROR MESSAGE 'DT' ENTRIES

4776	031552	001242						
4777	031554	001160	001116	001236	DT65:	.WORD	STMPO, SERRPC, SRPM1	
4778	031562	001160	001116	001236	DT66:	.WORD	STMPO, SERRPC, SRPM1, \$SILO, \$GDDAT, \$BDDAT	
4779	031570	001242	001124	001126				
4780	031576	001160	001116	001220	DT71:	.WORD	STMPO, SERRPC, SRPDS, SRPCS, SRPCA, SRPDA, SRPM1	
4781	031604	001224	001232	001234				
4782	031612	001236						
4783	031614	001160	001116	001220	DT72:	.WORD	STMPO, SERRPC, SRPDS, SRPCS, SRPCA, SRPM1	
4784	031622	001224	001232	001236				
4785	031630	001160	001116	001220	DT73:	.WORD	STMPO, SERRPC, SRPDS, SRPCS, SRPM1, \$GDDAT	
4786	031636	001224	001236	001124				
4787	031644	001160	001116	001220	DT74:	.WORD	STMPO, SERRPC, SRPDS, SRPCS, SRPDA, SRPM1	
4788	031652	001224	001234	001236				
4789								
4790								
4791								
4792	031660	000001						
4793	031662	002			DF:	.WORD	1	
4794	031663	000				.BYTE	2	
4795						.BYTE	0	
4796	031664	000001			DF1:	.WORD	1	
4797	031666	003				.BYTE	3	
4798	031667	000				.BYTE	0	
4799								
4800	031670	000001			DF2:	.WORD	1	
4801	031672	005				.BYTE	5	
4802	031673	000				.BYTE	0	
4803								
4804	031674	000001			DF20:	.WORD	1	
4805	031676	004				.BYTE	4	
4806	031677	000				.BYTE	0	
4807								
4808	031700	000001			DF30:	.WORD	1	
4809	031702	006				.BYTE	6	
4810	031703	000				.BYTE	0	
4811								
4812	031704	000001			DF71:	.WORD	1	
4813	031706	007				.BYTE	7	
4814	031707	000				.BYTE	0	
4815								
4816		000001				.END		

DZRPWC.P11

22-JUL-77 14:54

## SYMBOL TABLE

BABT	006060	DF20	031674	DT73	031630	EM52	025150	PR6	= 000300
BIT0	= 000001	DF30	031700	DT74	031644	EM53	025215	PR7	= 000340
BIT00	= 000001	DF71	031704	EMTVEC=	000030	EM54	025256	PS	= 177776
BIT01	= 000002	DH1	027757	EM1	022516	EM55	025325	PSM	= 177776
BIT02	= 000004	DH11	030070	EM10	023046	EM56	025371	PMRVEC=	000024
BIT03	= 000010	DH2	027775	EM100	027164	EM57	025435	RAPBA	006766
BIT04	= 000020	DH20	030135	EM101	027311	EM6	022757	RAPBA1	007024
BIT05	= 000040	DH24	030172	EM102	027360	EM60	025501	RAPWC	006622
BIT06	= 000100	DH30	030227	EM103	027401	EM61	025552	RAPWC1	006660
BIT07	= 000200	DH33	030320	EM104	027424	EM62	025573	RDCHR	= 104410
BIT08	= 000400	DH46	030411	EM105	027445	EM63	025624	RDLIN	= 104411
BIT09	= 001000	DH50	030471	EM106	027470	EM64	025661	RDOCT	= 104412
BIT1	= 000002	DH52	030526	EM107	027526	EM65	025736	READT	015644
BIT10	= 002000	DH61	030553	EM11	023102	EM66	026024	RESREG=	104414
BIT11	= 004000	DH62	030621	EM110	027603	EM67	026065	RESRP	003152
BIT12	= 010000	DH63	030646	EM111	027650	EM7	023020	RESTOR	015472
BIT13	= 020000	DH64	030713	EM112	027717	EM70	026175	RESVEC=	000010
BIT14	= 040000	DH65	030757	EM12	023123	EM71	026257	RPADR	001210
BIT15	= 100000	DH66	031004	EM13	023161	EM72	026357	RPBA	= 000010
EIT2	= 000004	DH7	030043	EM14	023223	EM73	026472	RPCA	= 000012
BIT3	= 000010	DH71	031062	EM15	023244	EM74	026562	RPCS	= 000004
BIT4	= 000020	DH72	031147	EM16	023267	EM75	026662	RPDA	= 000014
BIT5	= 000040	DH73	031224	EM17	023310	EM76	026752	RPDS	= 000000
BIT6	= 000100	DH74	031303	EM2	022560	EM77	027050	RPER	= 000002
BIT7	= 000200	DISPLA	001142	EM20	023333	ERBT	005324	RPM1	= 000016
BIT8	= 000400	DISPRE	000174	EM21	023363	ERF1	010144	RPM2	= 000020
BIT9	= 001000	DRVOL	022123	EM22	023415	ERF10	011346	RPM3	= 000022
BLNKS2	022120	DSF1	007066	EM23	023457	ERF11	011504	RPPRIO	001216
BPTVEC=	000014	DSF11	007554	EM24	023537	ERF12	011630	RPVEC	001212
BUSADR	001204	DSF13	007702	EM25	023565	ERF2	010270	RPWC	= 000006
CABT	006244	DSF4	007174	EM26	023626	ERF3	010416	RSTART	002774
CAF1	013416	DSF6	007332	EM27	023654	ERF4	010552	R6	=%000006
CKSMR	= 104407	DSF7	007446	EM3	022616	ERF5	010720	R7	=%000007
CLEARP	004252	DSWR	= 177570	EM30	023715	ERF6	011050	SAVREG=	104413
CLAP	021714	DT1	031360	EM31	023771	ERF7	011216	SAVRP	022022
CR	= 000015	DT11	031404	EM32	024050	ERRVEC=	000004	SEEK	015052
CRLF	= 000200	DT2	031364	EM33	024114	GETADR	021376	SETOL	004130
CSBT	005510	DT20	031416	EM34	024165	GTSWR	= 104406	SILO	= 000026
CSF1	011760	DT24	031426	EM35	024241	HT	= 000011	SILOT	014270
CSF10	013112	DT30	031436	EM36	024305	INDEXP	021774	SILOT1	014520
CSF11	013214	DT33	031452	EM37	024357	IOTVEC=	000020	STACK	= 001100
CSF12	013302	DT46	031466	EM4	022652	LF	= 000012	START	002364
CSF2	012464	DT50	031500	EM40	024434	MASK	001202	START1	002372
CSF3	012610	DT52	031510	EM41	024500	MAXCYL	001176	START2	002400
CSF6	012710	DT61	031516	EM42	024527	MAXPAT	001200	STKLMT=	177774
CSF7	013010	DT62	031526	EM43	024571	PIRQ	= 177772	SUCA	= 000024
DABT	006430	DT63	031534	EM44	024620	PIRQVE=	000240	SWR	001140
DAF1	013560	DT64	031544	EM45	024662	PRO	= 000000	SWREG	000176
DAF2	013710	DT65	031554	EM46	024705	PR1	= 000040	SWO	= 000001
DDISP	= 177570	DT66	031562	EM47	024734	PR2	= 000100	SWO0	= 000001
DF	031660	DT7	031376	EM5	022721	PR3	= 000140	SWO1	= 000002
DF1	031664	DT71	031576	EM50	025006	PR4	= 000200	SWO2	= 000004
DF2	031670	DT72	031614	EM51	025061	PR5	= 000240	SWO3	= 000010

SW04 = 000020	TST15 007310	TST6 006000	SESCAP 001164	SRPWC 001226
SW05 = 000040	TST16 007424	TST7 006164	SFILLC 001156	SRTNAD 016376
SW06 = 000100	TST17 007532	TYPDS = 104405	SFILLS 001155	SSAVRE 021214
SW07 = 000200	TST2 004222	TYPE = 104401	SGADR 001120	SSCOPE 020734
SW08 = 000400	TST20 007640	TYPERR 016556	SGDAT 001124	SSETUP= 000127
SW09 = 001000	TST21 010122	TYPOC = 104402	SGET42 016354	SSILO 001242
SW1 = 000002	TST22 010246	TYPON = 104404	SGTSWR 017742	SSTUP = 177777
SW10 = 002000	TST23 010364	TYPOS = 104403	SHD = 000000	SSUCA 001240
SW11 = 004000	TST24 010512	T3P 021764	SHIOCT 020732	SSVLAD 021150
SW12 = 010000	TST25 010666	WCBT 005674	SICNT 001104	SSVPC = 000200
SW13 = 020000	TST26 011014	WRITE 016004	SINTAG 001135	SSMR = 167400
SW14 = 040000	TST27 011164	SAUTOB 001134	SITEMB 001114	SSWRM= 000200
SW15 = 100000	TST3 005244	SBOADR 001122	SLF 001174	STIMES 001162
SW2 = 000004	TST30 011312	SBO DAT 001126	SLPADR 001106	STKB 001146
SW3 = 000010	TST31 011462	SBELL 001166	SLPERR 001110	STKS 001144
SW4 = 000020	TST32 011606	SCHARC 017214	SMNEW 020562	STMPO 001160
SW5 = 000040	TST33 011724	SCKSWR 017672	SMSWR 020551	STN = 000056
SW6 = 000100	TST34 012112	SCHTAG 001100	SMXCNT 021212	STPB 001152
SW7 = 000200	TST35 012236	SCM3 = 000000	SNL 001154	STPFLG 001157
SW8 = 000400	TST36 012442	SCM4 = 000001	SNWTST= 000001	STPS 001150
SW9 = 001000	TST37 012566	SCNTLG 020544	SOCNT 017442	STRAP 021310
TBITVE= 000014	TST4 005430	SCNTLU 020537	SOMODE 017444	STRAP2 021332
TIMEP 021744	TST40 012666	SCRFL 001173	SOVER 021176	STRP = 000015
TKVEC = 000060	TST41 012766	SDBLK 017662	SPASS 001100	STRPAD 021344
TPL 001206	TST42 013070	SDOAGN 016374	SQUES 001172	STSTNM 001102
TPVEC = 000064	TST43 013172	SDTBL 017652	SRDCHR 020154	STTYIN 020530
TRAPVE= 000034	TST44 013260	SENDAD 016364	SRDLIN 020274	STYPDS 017446
TRTVEC= 000014	TST45 013364	SENDCT 016320	SRDOCT 020574	STYPE 017000
TSTAT 012260	TST46 013472	SENULL 016400	SRDSZ = 000007	STYPEC 017150
TSTNAT 012146	TST47 013666	SEOP 016162	SRESRE 021252	STYPEX 017216
TST0 003122	TST5 005614	SEOPCT 016312	SRPBA 001230	STYPOC 017244
TST1 004104	TST50 014246	SERFLG 001103	SRPCA 001232	STYPON 017260
TST10 006350	TST51 014470	SERMAX 001115	SRPCS 001224	STYPOS 017220
TST11 006534	TST52 015022	SERROR 016404	SRPDA 001234	SXTSTR 020746
TST12 006700	TST53 015450	SERRPC 001116	SRPDS 001220	SSGET4= 000000
TST13 007044	TST54 015622	SERRTB 001244	SRPER 001222	SOFILL 017443
TST14 007152	TST55 015762	SERTTL 001112	SRPM1 001236	. = 031710

. ABS. 031710 000

ERRORS DETECTED: 0

DZRPWC, DZRPWC/SOL/DOC=DZRPWC.P11  
 RUN-TIME: 21 14 .4 SECONDS  
 RUN-TIME RATIO: 988/36=26.7  
 CORE USED: 24K (47 PAGES)

DOCUMENT PAGES: 94