



| Row | Col 1 | Col 2 | Col 3 | Col 4 |
|-----|-------|-------|-------|-------|
| 1 | 11 | 12 | 13 | 14 |
| 2 | 15 | 16 | 17 | 18 |
| 3 | 19 | 20 | 21 | 22 |
| 4 | 23 | 24 | 25 | 26 |
| 5 | 27 | 28 | 29 | 30 |
| 6 | 31 | 32 | 33 | 34 |
| 7 | 35 | 36 | 37 | 38 |
| 8 | 39 | 40 | 41 | 42 |
| 9 | 43 | 44 | 45 | 46 |
| 10 | 47 | 48 | 49 | 50 |
| 11 | 51 | 52 | 53 | 54 |
| 12 | 55 | 56 | 57 | 58 |

B01

CO1

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 2
DZDQHC.P11 21-DEC-76 16:36

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZDQH-C-D
PRODUCT NAME: CHARACTER LENGTH AND INTERRUPT TESTS
DATE: MARCH 1977
MAINTAINER: DIAGNOSTIC GROUP

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

D01

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 3
DZDQHC.P11 21-DEC-76 16:36

1. ABSTRACT

THE FUNCTION OF THE DQ11 DIAGNOSTICS ARE TO VERIFY THAT THE OPTION OPERATES ACCORDING TO SPECIFICATIONS.

THIS TEST WILL CHECK TRANSMITTER AND RECEIVER INTERRUPTS, ENTER T AND EXIT T IF THE AB OPTION IS INSTALLED. IT ALSO CHECKS VRC AND DATA TRANSFERS. THE SECOND PART CHECKS TRANSMITTER AND RECEIVER CHARACTER LENGTHS.

CURRENTLY THERE ARE SEVEN OFF LINE DIAGNOSTICS THAT ARE TO BE RUN IN SEQUENCE TO INSURE THAT IF AN ERROR SHOULD OCCUR IT WILL BE DETECTED AT AN EARLY STAGE AND INSURING THAT DIAGNOSIS OF ERROR WILL BE IMMEDIATE TO PROBLEM
NOTE: ADDITIONAL DIAGNOSTICS MAY BE ADDED IN THE FUTURE.

THE SEVEN DIAGNOSTICS ARE:

1. DZDQA [REV] BASIS R/W TEST #1
2. DZDQB [REV] BASIC R/W TEST #2
3. DZDQC [REV] BASIC NPR AND INTERRUPT TEST
4. DZDQD [REV] RECEIVER TRANSMITTER EXERCISER TEST
5. DZDQE [REV] MISC. RX AND TX TESTS. PLUS BCC TESTS.
6. DZDQF [REV] CHARACTER DETECT TESTS.
7. DZDQH [REV] CHARACTER LENGTH AND INTERRUPT TESTS.

THERE IS ALSO AN ONLINE TEST TO BE DISCUSSED LATER.

1. DZDQ0 [REV] ONLINE TEST. (ITEP OVERLAY)

AND A PARAMETER INPUT PROGRAM IS AVAILABLE

1. DZDQG [REV] DQ11 TRIAL PROGRAM (PARAMETER INPUT)

2. REQUIREMENTS

2.1 EQUIPMENT

ANY PDP11 FAMILY CPU (WITH MINIMUM 4K MEMORY)-WITH
OR WITHOUT A HARDWARE SWITCH REGISTER (LOC. 177570)
ASR 33 (OR EQUIVALENT)
DQ11
SYNC MODEM (ONLY REQUIRED FOR ONLINE TEST)

2.2 STORAGE

PROGRAM WILL LOAD AND RUN
IN 4K OF MEMORY.
LOCATION 1400 THRU 1600 ARE ESPECIALLY TO
BE NOTED AND TO BE UNTOUCHED BY OPERATOR
AFTER DQ11 TRIAL PROGRAM HAS BEEN EXECUTED.
OR AFTER THE "AUTO SIZING" HAS BEEN DONE.

3. LOADING PROCEEDURE

3.1 METHOD

ALL PROGRAMS ARE IN ABSOLUTE FORMAT AND
ARE LOADED USING THE ABSOLUTE LOADER.

ABSOLUTE LOADER STARTING ADDRESS *500

MEMORY *
SIZE

| | |
|-----|-----|
| 4K | 17 |
| 8K | 37 |
| 12K | 57 |
| 16K | 77 |
| 20K | 117 |
| 24K | 137 |
| 28K | 157 |

3.1.1 LOAD THE ADDRESS OF ABS. LOADER (LOC.XXX500)

3.1.2 THEN START

4. STARTING PROCEEDURE

A. LOAD LOC. 200

B. SET SWR TO ZERO FOR "AUTO SIZING" OR LEAVE
LEAVE SWR BIT 7=1 TO USE EXISTING PARAMETERS SET UP
BY DQ11 TRIAL PROGRAM OR A PREVIOUSLY RUN DQ11 DIAGNOSTIC
THAT USED THE "AUTO SIZING".
***REFER TO SECTION 4.1 FOR SOFTWARE SWITCH REGISTER OPERATION
AND OPTIONS.***

NOTE: THE SOFTWARE SWITCH REGISTER IS LOCATED AT LOC.176
SOFTWARE DISPLAY REGISTER IS LOCATED AT LOC.174

C. THEN START

THE PROGRAM WILL TYPE MAINDEC NAME AND PROGRAM NAME
IF THIS WAS THE FIRST START UP OF THE PROGRAM) AND ALSO
THE FOLLOWING:

| | |
|----------------------|--------|
| "MAP OF DQ11 STATUS" | |
| 1400 | 160010 |
| 1402 | 152300 |
| 1404 | 160020 |
| 1406 | 150310 |

THE ABOVE IS ONLY AN EXAMPLE!
THIS WOULD INDICATE THE STATUS TABLE STARTING AT ADD.
1400 IN THE PROGRAM. THE STATUS TABLE MUST BE VERIFIED BY THE
USER IF AUTO SIZING IS DONE. FOR INFORMATION OF STATUS
TABLE SEE SECTION 8.4 FOR HELP.

F01

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 5
DZDQMC.P11 21-DEC-76 16:36

****IF THE SOFTWARE SWITCH REGISTER IS SELECTED THEN THE FOLLOWING
WILL BE TYPED AFTER THE PROGRAM IDENTIFIES ITSELF:
SWR=XXXXXX NEW= (REFER TO SECTION 4.1 FOR OPERATOR'S OPTION)****
NOTE: IF USING THE SOFTWARE SWITCH REGISTER WHEN A HARDWARE
SWITCH REGISTER IS AVAILABLE THE PROGRAM WILL NOT
TYPE OUT THE TITLE.

THE PROGRAM WILL TYPE "R"
AND PROCEED TO RUN THE DIAGNOSTIC

4.1 CONTROL SWITCH SETTINGS

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.

NOTE: WHEN RUNNING THIS PROGRAM WITH A
SWITCH REGISTERLESS PROCESSOR IT MAY BE NECESSARY
TO DEPRESS (<↑G>) MORE THAN ONCE BEFORE IT IS ACCEPTED
DUE TO THE NUMBER OF RESET INSTRUCTIONS THAT ARE EXECUTED.

2) THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW= (XXXXXX IS THE OCTAL CONTENTS
OF THE SOFTWARE SWITCH REGISTER.)

3) AFTER THE ''NEW=''
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.

SW 15 SET: HALT ON ERROR
SW 14 SET: LOOP ON CURRENT TEST
SW 13 SET: INHIBIT ERROR PRINT OUT

GO1

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 6
DZDQHC.P11 21-DEC-76 16:36

SW 12 SET: INHIBIT TYPE OUT/BELL ON ERROR.
SW 11 SET: INHIBIT ITERATIONS
SW 10 SET: ESCAPE TO NEXT TEST
SW 09 SET: LOOP WITH CURRENT DATA
SW 08 SET: CATCH ERROR AND LOOP ON IT
SW 07 SET: USE PREVIOUS STATUS TABLE. CLR-DO AUTO SIZE.
SW 06 SET:
SW 05 SET:
SW 04 SET:
SW 03 SET:
SW 02 SET: LOCK ON SELECTED TEST
SW 01 SET: RESTART PROGRAM AT SELECTED TEST
SW 00 SET: RESELECT DQ11'S DESIRED ACTIVE.

4.1.2 SWITCH REGISTER RESTRICTIONS

SW 00 RESELECT DQ11'S DESIRED ACTIVE.
PLEASE NOTE THAT A MESSAGE IS TYPED
OUT FOR SWITCH REGISTER BEING EQUAL TO DQ11'S
ACTIVE. THIS MEANS IF THE SYSTEM HAS
FOUR DQ11S; BITS 00,01,02,03 WILL
BE SET IN LOC "DQACTV". USING THIS
SWITCH ALTERS THAT LOCATION; THEREFORE
IF FOUR DQ11S ARE IN THE SYSTEM
DO NOT SET SWITCHS GREATER THAN
SW 03 IN THE UP POSITION. THIS WOULD BE
A FATAL ERROR. DO NOT SELECT MORE ACTIVE
DQ11S THAN HAS BEEN GIVEN INFORMATION
ABOUT IN TRIAL PROGRAM.

METHOD: A: LOAD ADDRESS 200
B: START WITH SW 00=1
C: PROGRAM WILL TYPE MESSAGE
D: CONTINUE THE BINARY NUMBER OF DQ11S DESIRED ACTIVE
EXAMPLE: 1=1 DQ11; 3=2 DQ11; 7=3 DQ11; 17=4 DQ11 37=5 DQ11 ETC.
E: NUMBER (IF VALID) WILL BE IN DATA LIGHTS (EXCLUDING 11/05, 11/04, 11/34)
F: CONTINUE WITH ANY OTHER SWITCH SETTINGS DESIRED.

SW 01 IT IS STRONGLY SUGGESTED THAT
AT LEAST ONE PASS HAS BEEN MADE
BEFORE TRYING TO SELECT A TEST
THAT IS NOT IN THE ORDER OF SEQUENCE
THE REASON BEING IS THAT THE
PROGRAM HAS TO CLEAR AREAS AND SET
UP PARAMETERS. ALSO WHEN A TEST IS
SELECTED ALWAYS START AT THE VERY
BEGINNING OF THAT TEST.

SW 09 LOOP ON CURRENT DATA:
THIS SWITCH WILL ONLY WORK IF
CALL "SCOPI" IS IN THAT TEST.
THE REASON BEING THAT MOST TESTS
DEAL WITH BLOCKS OF DIFFERENT DATA
TO BE SENT OR RECEIVED ALL AT ONCE
THUS IN BLOCK DATA; ONE PATTERN CANN'T BE SINGLED OUT.

4.1.3 SWITCH REGISTER PRIORITYS

ERROR SWITCHES

1. SW 12 DELETE PRINT OUT/BELL ON ERROR.
2. SW 13 DELETE ERROR PRINTOUT.
3. SW 15 HALT ON THE ERROR.
4. SW 08 GOTO BEGINNING OF THE TEST.
5. SW 10 GOTO NEXT TEST ON ERROR.

****HLT (ERROR) ROUTINE SUPPORTS <↑G> OPERATION****

SCOPE SWITCHES

1. SW 09 (IF ENABLED BY "SCOPI")
2. SW 14
3. SW 11

****SCOPE ROUTINE WILL SUPPORT <↑G> OPERATION****

4.2 STARTING ADDRESS

STARTING ADDRESS IS AT 000200
THERE ARE NO OTHER STARTING ADDRESSES
FOR THE DQ11 DIAGNOSTICS PREVIOUSLY MENTIONED

NOTE: IF ADDRESS 000042 IS NON-ZERO
THE PROGRAM ASSUMES IT IS UNDER
ACT11 OR DDP CONTROL AND WILL ACT ACCORDINGLY
AFTER *ALL* AVAILABLE DQ11'S ARE TESTED
THE PROGRAM WILL RETURN TO "DDP2" OR "ACT-11".

5. OPERATING PROCEDURE

WHEN PROGRAM IS INITIALLY STARTED MESSAGES AS DESCRIBED IN SECTION
FOUR WILL BE PRINTED.

AND PROGRAM WILL BEGIN RUNNING THE
DIAGNOSTIC

5.2 PROGRAM AND/OR OPERATOR ACTION

THE TYPICAL APPROACH SHOULD BE

1. HALT ON ERROR (VIA SW 15=1)
WHEN EVER AN ERROR OCCURS
2. CLEAR SW 15
3. SET SW 14: (LOOP ON THIS TEST)
4. SET SW 13: (INHIBIT ERROR PRINT OUT)

THE TEST NUMBER AND PC WILL BE TYPED OUT AND
POSSIBLY AN ERROR MESSAGE (THIS DEPENDS ON THE TEST)
TO GIVE THE OPERATOR AN IDEA AS TO THE SOURCE OF THE
PROBLEM. IF IT IS NECESSARY TO KNOW MORE INFORMATION
CONCERNING THE ERROR REPORT; LOOK IN THE LISTING
FOR THAT TEST NUMBER WHICH WAS TYPED OUT

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 8
DZDQHC.P11 21-DEC-76 16:36

AND THEN NOTE THE PC OF THE ERROR REPORT
THIS WAY THE EXACT FUNCTIONING OF THE TEST
CAN BE INTERPEDITED

6. ERRORS

AS DESCRIBED PREVIOUSLY THERE WILL ALWAYS BE
A TEST NUMBER AND PC TYPED OUT AT THE TIME OF AN
ERROR (PROVIDING SW 13=0 AND SW 12=0). IN MOST CASES ADDITIONAL
INFORMATION WILL BE SUPPLIED THE THE ERROR MESSAGE
WHICH IS TO GIVE THE OPERATOR AN INDICATION OF THE
ERROR.

6.2 ERROR RECOVERY

IF FOR SOME REASON THE DQ11 SHOULD
"HANG THE BUS" (GAIN CONTROL OF BUS SO THAT
CONSOLE MANUAL FUNCTIONS ARE INHIBITED) AN INIT
OR POWER DOWN/UP IS NECESSARY FOR OPERATOR
TO REGAIN CONTROL OF CPU.
IF THIS SHOULD HAPPEN; LOOK IN LOCATION
"TSTNO" (ADDRESS 1226) FOR THE NUMBER OF THE TEST THAT
WAS RUNNING AT THE TIME OF THE CATASTROPHIC
ERROR.
IN THIS WAY THE OPERATOR WILL HAVE AN IDEA AS TO
WHAT THE DQ11 WAS DOING AT THE TIME OF THE ERROR.

6.3 ***HALT RECOVERY WHEN USING SOFTWARE SWITCH REGISTER***

IF THE SOFTWARE SWITCH REGISTER IS TO BE CHANGED AFTER A HALT
THE THE OPERATOR IS REQUIRED TO TYPE A <↑G> BEFORE DEPRESSING CONTINUE.
THE FOLLOWING WILL BE TYPED:
SWR=XXXXXX NEW= (REFER TO SECTION 4.1 FOR OPERATOR OPTION)

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

SEE SECTION 4. (PLEASE)

7.2 OPERATING RESTRICTIONS

DQ11 TRIAL PROGRAM MUST BE RUN PRIOR TO THE
FIRST AND ONLY THE FIRST RUNNING OF ANY DQ11 DIAGNOSTIC
NOTE: IF NO PROGRAM OTHER THAN A
DQ11 DIAGNOSTIC WAS LOADED AFTER DQ11 TRIAL OR
IF CORE MEMORY HAS NOT BEEN CHANGED; OR IF THERE
IS NO DQ11 CONFIGURATION CHANGES; THE
DQ11 TRIAL PROGRAM NEED NEVER BE RUN AGAIN.
HOWEVER IF ANY OF THE ABOVE HAVE BEEN VIOLATED
THE DQ11 TRIAL PROGRAM MUST BE RUN AGAIN
BEFORE RUNNING THE DIAGNOSTICS
NOTE: AN ALTERNATIVE TO THE ABOVE IS ATTEMPTING
THE "AUTO SIZING" WHEN PROGRAM IS INITIALLY STARTED
WITH SW07=0.

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 9
 DZDQHC.P11 21-DEC-76 16:36

8. MISCELLANEOUS

8.1 EXECUTION TIME

8.2 PASS COMPLETE

WHEN THE DIAGNOSTIC HAS COMPLETED
 A PASS THE FOLLOWING IS AN EXAMPLE
 OF THE PRINT OUT TO BE EXPECTED.

END PASS DZDQH-C CSR: 160000 VEC: 300 PASSES: 000001 ERRORS: 000000

NOTE: THE NUMBERS FOR CSR AND VEC ARE
 NOT NECESSARILY THE VALUES FOR THE DEVICE
 THEY ARE ONLY FOR THIS EXAMPLE.

8.3 TST1 (MINI MONITOR)

THE VERY FIRST "TEST" (TST1)
 IS *NOT* A TEST OF THE DQ11 HARDWARE
 IT IS A MINI-MONITOR USED TO CYCLE DQ11 IN THE
 SYSTEM THROUGH THE DIAGNOSTIC.

REMEMBER: TST1 IS NOT A TEST OF DQ11 HARDWARE!!!!!!!

8.4 KEY LOCATIONS

RETURN (1214) CONTAINS THE ADDRESS WHERE PROGRAM WILL
 RETURN WHEN ITERATION COUNT IS REACHED
 OR IF LOOP ON TEST IS ASSERTED.
 NEXT (1216) CONTAINS THE ADDRESS OF THE NEXT TEST
 TO BE PERFORMED.
 TSTNO (1226) CONTAINS THE NUMBER OF THE TEST NOW
 BEING PERFORMED.
 RUN (1304) THE BIT IN "RUN" ALWAYS POINTS ONE
 PAST THE DQ11 CURRENTLY BEING TESTED.
 EXAMPLE:
 (RUN) 1304/0000000001000000
 MEANS THAT DQ11 NO.05 IS THE DQ11 NOW
 RUNNING.

DQCR00-DQCR17
 DQST00-DQST17
 (1400)-(1476)

THESE LOCATIONS CONTAIN THE INFORMATION
 NEEDED TO TEST UP TO 16 (DECIMAL) DQ11S
 SEQUENTIALLY. THEY CONTAIN THE CSR VECTOR
 AND STATUS CONCERNING THE CONFIGURATION
 OF EACH DQ11.

DQACTV (1500)

EACH BIT SET IN THIS LOCATION INDICATES
 THAT THE ASSOCIATED DQ11 WILL BE TESTED
 IN TURN.

EXAMPLE:
 (DQACTV) 1500/0000000000011111
 MEANS THAT DQ11 NO. 00,01,02,03,04
 WILL BE TESTED.
 EXAMPLE:

K01

DZDGM MACY11 27(1006) 22-DEC-76 11:40 PAGE 10
DZDQMC.P11 21-DEC-76 16:36

(DQACTV) 1500/000000000010001
MEANS THAT DQ11 NO. 00,04
WILL BE TESTED.
DQCSR (1506) CONTAINS THE RECEIVER CSR OF THE
CURRENT DQ11 UNDER TEST.
DQSTAT (1510) CONTAINS THE STATUS OF THE CURRENT
DQ11 UNDER TEST.
BIT 15 SET: TWO SYNC CHARS/ONE SYNC CHAR
BIT 14 SET: TEST JUMPER INSTALLED/NOT INSTALLED
BIT 13 SET: BB OPTION INSTALLED/NOT INSTALLED
BIT 12 SET: BA OPTION INSTALLED/NOT INSTALLED
BIT 11 SET: ACTIVE ON FIRST NON-SYNC/ACTIVE AFTER NO. OF SYNC
BIT 10 SET: AB OPTION INSTALLED/NOT INSTALLED
BIT 09 SET: ODD VRC/EVEN VRC
BIT 00-08 VECTOR "A" OF DEVICE

8.5 *** METHOD OF AUTO SIZING ***

8.5.1 FINDING THE CONTROL STATUS REGISTER.

WHEN LOOKING FOR THE CSR IT IS NECESSARY TO TAKE CARE
THAT WHEN A CSR IS FOUND THAT IT IS INDEED A DQ11. THAT
IS THE METHOD OF MY MADNESS FOR THIS ROUTINE.
AN ATTEMPT TO CLEAR THE MISC. REGISTER IS TRIED
IF A TIME-OUT TRAP OCCURES POINTERS ARE UPDATED
AND ATTEMPTED AGAIN. IF NO TIME-OUT; THE RECEIVER "ACTIVE BIT" (BIT 12)
IS SET AND A *COMPARE* FOR BOTH SYNC1 AND SYNC 2 IS DONE
AT THE MISC. REGISTER. IF THEY ARE THERE THIS IS
A DQ11. THE INFORMATION IS STORED AWAY.

8.5.2 ONE SYNC BIT OR TWO?

SINCE TOO MUCH HARDWARE MUST BE TURNED ON TO SENSE THE
PRESENTS OF ONE SYNC OR TWO. THE PROGRAM ASSUMES TWO SYNC
CHARS. NOTE: THIS ASSUMPTION MAY BE ALTERED AFTER AUTO SIZING
BY ALTERING BIT 15 IN APPRIORATE DQSTXX: LOCATION.

8.5.3 "BB" OPTION INSTALLED?

TO SENSE FOR THE "BB" OPTION THE PROGRAM SELECTS THE
CHARACTER DET. REGISTER AND THE LOADS IN ALL 1'S; IF
ANY ONE OR COMBINATION OF BITS ARE SET THE BB OPTION
IS ASSUMED TO EXIST.

8.5.4 "AB" OPTION INSTALLED?

TO SENSE FOR THE "AB" OPTION THE PROGRAM SELECTS THE
POLYNOMIAL REGISTER AND WRITES ALL 1'S INTO IT; IF ANY
ONE OR COMBINATION OF BITS ARE SET THE AB OPTION IS ASSUMED
TO EXIST.

8.5.5 "BA" OPTION INSTALLED?

TO SENSE FOR "BA" OPTION REQUEST TO SEND AND DATA TERMINAL
READY ARE SET; IF EITHER ONE OR BOTH ARE SET THE PROGRAM
ASSUMES THE BA OPTION EXISTS

8.5.6 JUMPER ON END OF CABLE?

THE PROGRAM CHECKS TO SEE IF EITHER OR BOTH CLEAR TO SEND AND CARRIER ARE SET; IF SO THE PROGRAM ASSUMES THE TEST JUMPER IS ON THE END OF THE CABLE.

8.5.7 ACTIVE ON FIRST NON-SYNC?

SINCE TOO MUCH HARDWARE MUST BE TURNED ON TO SENSE FOR WHEN THE D011 GOES ACTIVE THE PROGRAM ASSUMES "ACTIVE ON FIRST NON-SYNC". NOTE: THIS CAN BE CHANGED BY ALTERING BIT 11 IN THE APPRIORATE D0STXX: AFTER AUTO SIZING

8.5.8 SET FOR ODD OR EVEN PARITY?

AS ABOVE TOO MUCH HARDWARE IS NEED TO SENSE WHICH PARITY WAS SELECTED. SO THE PROGRAM ASSEMES ODD PARITY. NOTE: THIS CAN BE CHANGED BY ALTERING BIT 9 IN APPRIORATE D0STXX: LOCATION. AFTER AUTO SIZING

8.5.9 FINDING THE VECTOR.

THE PROGRAM SETS "PRIMARY DONE", "SECONDAY DONE", AND "INTERUPT ENABLE" AND LOOKS FOR AN INTERUPT. IF IT INTERUPTS IT IS PICKED UP AND STORED AWAY. IF NO INTERUPT OCCURES THE PROGRAM ASSUMES VECTOR =300. THIS PROBLEM WILL BE FIXED IN ONE OF THE DIAGNOSTICS AND *AUTO SIZING* SHOULD BE REDONE TO GET THE CORRECT VECTOR.

9. PROGRAM DESCRIPTION

CONTAINED WITHIN LISTING

10. LISTING

FOLLOWING

MO1

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 12
DZDQMC.P11 21-DEC-76 16:36

INTRODUCTION TO DQ11 DIAGNOSTIC

;MAINDEC-11-DZDQH-C/<377>/CHARACTER LENGTH AND INTERRUPT TESTS
;COPYRIGHT 1975, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754

;REVISED 16-DEC-76 BY R. BLACK

A)SUPPORTS SOFTWARE SWITCH REGISTER

B)SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER

BY <↑G>.

;STARTING PROCEDURE

;LOAD PROGRAM

;LOAD ADDRESS 000200

;PRESS START

;PROGRAM WILL TYPE "MAINDEC-11-DZDQH-C/<377>/CHARACTER LENGTH AND INTERRUPT TEST

;PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED

;AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE

;AND THEN RESUME TESTING

;SWITCH REGISTER OPTIONS

534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569

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

SW15=100000
SW14=40000
SW13=20000
SW12=10000
SW11=4000
SW10=2000
SW09=1000
SW08=400
SW06=100
SW05=40
SW04=20
SW03=10
SW02=4
SW01=2
SW00=1

;=1, HALT ON ERROR
;=1, LOOP ON CURRENT TEST
;=1, INHIBIT ERROR TYPEOUT
;=1, DELETE TYPEOUT/BELL ON ERROR.
;=1, INHIBIT ITERATIONS
;=1, ESCAPE TO NEXT TEST ON ERROR
;=1, LOOP WITH CURRENT DATA
;=1, LOOP ON ERROR

;LOCK ON TEST SELECT
;RESTART PROGRAM AT SELECTED TEST
;RESELECT DQ11 DESIRED ACTIVE
;NOTE: THIS MUST NOT EXCEED ORIGINAL COUNT

NO1

DZDGH MACY11 27(1006) 22-DEC-76 11:40 PAGE 13
 DZDGH.C.P11 21-DEC-76 16:36 GENERAL DEFINATIONS AND EQUIVALENCIES

```

570
571
572
573
574      000000      R0=%0          ;GENERAL REGISTER
575      000001      R1=%1          ;GENERAL REGISTER
576      000002      R2=%2          ;GENERAL REGISTER
577      000003      R3=%3          ;GENERAL REGISTER
578      000004      R4=%4          ;GENERAL REGISTER
579      000005      R5=%5          ;GENERAL REGISTER
580      000006      SP=%6         ;PROCESSOR STACK POINTER
581      000007      PC=%7         ;PROGRAM COUNTER
582
583      ;REGISTER DEFINITIONS
584
585      ;LOCATION EQUIVALENCIES
586      177570      DSWR= 177570   ;HARDWARE SWITCH REGISTER LOC.
587      177570      DLIGHTS=177570 ;HARDWARE DISPLAY REGISTER LOC.
588      177776      PS=177776     ;PROCESSOR STATUS WORD
589      001200      STACK=1200    ;START OF PROCESSOR STACK
590
591      ;INSTRUCTION DEFINITIONS
592      005746      PUSH1SP=5746   ;DECREMENT PROCESSOR STACK 1 WORD
593      005726      POP1SP=5726    ;INCREMENT PROCESSOR STACK 1 WORD
594      010046      PUSHRO=10046    ;SAVE R0 ON STACK
595      012600      POPRO=12600     ;RESTORE R0 FROM STACK
596      024646      PUSH2SP=24646  ;DECREMENT STACK TWICE
597      022626      POP2SP=22626   ;INCREMENT STACK TWICE
598      .EQUIV ENT,HLT ;BASIC DEFINITION OF ERROR CALL
599
600
601      100000      BIT15=100000
602      040000      BIT14=40000
603      020000      BIT13=20000
604      010000      BIT12=10000
605      004000      BIT11=4000
606      002000      BIT10=2000
607      001000      BIT9=1000
608      000400      BIT8=400
609      000200      BIT7=200
610      000100      BIT6=100
611      000040      BIT5=40
612      000020      BIT4=20
613      000010      BIT3=10
614      000004      BIT2=4
615      000002      BIT1=2
616      000001      BIT0=1
617
618
619      ;DB11 OPTIONAL DEFINITIONS
620
621      002000      ABBIT=2000
622      004000      ACTBIT=4000
623      010000      BABIT=10000
624      020000      BBBIT=20000
625      040000      JUMBIT=40000
  
```

626 001000
 627 100000
 628
 629
 630
 631
 632 000000
 633 000001
 634 000002
 635 000003
 636 000004
 637 000005
 638 000006
 639 000007
 640
 641 000010
 642 000011
 643 000012
 644 000013
 645 000014
 646 000015
 647 000016
 648 000017
 649
 650

ODDBIT=1000
 SYNBIT=100000

;DQ11 SECONDARY REGISTER DEFINATIONS

| | |
|-----------|-------------------------------------|
| RXBA.P=0 | ;RECEIVER BUS ADDRESS PRIMARY. |
| RXWC.P=1 | ;RECEIVER WORD COUNT PRIMARY. |
| TXBA.P=2 | ;TRANSMITTER BUS ADDRESS PRIMARY. |
| TXWC.P=3 | ;TRANSMITTER BUS ADDRESS PRIMARY. |
| RXBA.S=4 | ;RECEIVER BUS ADDRESS SECONDARY. |
| RXWC.S=5 | ;RECEIVER WORD COUNT SECONDARY. |
| TXBA.S=6 | ;TRANSMITTER BUS ADDRESS SECONDARY. |
| TXWC.S=7 | ;TRANSMITTER WORD COUNT SECONDARY. |
| CHARDT=10 | ;CHARACTER DETECT REGISTER. |
| SYNC.=11 | ;SYNC REGISTER. |
| MISC.=12 | ;MISCELLANEOUS REGISTER. |
| TX.MUX=13 | ;TRANSMITTER MUX REGISTER. |
| SEQ.=14 | ;SEQUENCE REGISTER. |
| RX.BCC=15 | ;RECEIVER BCC REGISTER. |
| TX.BCC=16 | ;TRANSMITTER BCC REGISTER. |
| POLY.=17 | ;POLYNOMIAL REGISTER. |

```

651          ;TRAPCATCAER FOR ILLEGAL INTERRUPTS
652          .=0
653 000000 000002      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
654 000002 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
655 000004 000006      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
656 000006 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
657 000010 000012      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
658 000012 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
659 000014 000016      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
660 000016 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
661 000020 000022      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
662 000022 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
663 000024 000026      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
664 000026 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
665 000030 000032      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
666 000032 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
667 000034 000036      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
668 000036 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
669 000040 000042      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
670 000042 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
671 000044 000046      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
672 000046 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
673 000050 000052      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
674 000052 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
675 000054 000056      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
676 000056 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
677 000060 000062      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
678 000062 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
679 000064 000066      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
680 000066 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
681 000070 000072      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
682 000072 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
683 000074 000076      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
684 000076 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
685 000100 000102      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
686 000102 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
687 000104 000106      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
688 000106 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
689 000110 000112      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
690 000112 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
691 000114 000116      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
692 000116 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
693 000120 000122      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
694 000122 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
695 000124 000126      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
696 000126 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
697 000130 000132      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
698 000132 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
699 000134 000136      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
700 000136 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
701 000140 000142      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
702 000142 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
703 000144 000146      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
704 000146 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
705 000150 000152      .+2      ;UNEXPECTED TRAP TO THIS LOCATION
706 000152 000000      HALT      ;EXAMINE STACK TO FIND CAUSE
    
```


| | | | | |
|-----|--------|--------|------|-----------------------------------|
| 707 | 000154 | 000156 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 708 | 000156 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 709 | 000160 | 000162 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 710 | 000162 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 711 | 000164 | 000166 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 712 | 000166 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 713 | 000170 | 000172 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 714 | 000172 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 715 | 000174 | 000176 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 716 | 000176 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 717 | 000200 | 000202 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 718 | 000202 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 719 | 000204 | 000206 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 720 | 000206 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 721 | 000210 | 000212 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 722 | 000212 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 723 | 000214 | 000216 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 724 | 000216 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 725 | 000220 | 000222 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 726 | 000222 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 727 | 000224 | 000226 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 728 | 000226 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 729 | 000230 | 000232 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 730 | 000232 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 731 | 000234 | 000236 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 732 | 000236 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 733 | 000240 | 000242 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 734 | 000242 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 735 | 000244 | 000246 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 736 | 000246 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 737 | 000250 | 000252 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 738 | 000252 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 739 | 000254 | 000256 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 740 | 000256 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 741 | 000260 | 000262 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 742 | 000262 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 743 | 000264 | 000266 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 744 | 000266 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 745 | 000270 | 000272 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 746 | 000272 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 747 | 000274 | 000276 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 748 | 000276 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 749 | 000300 | 000302 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 750 | 000302 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 751 | 000304 | 000306 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 752 | 000306 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 753 | 000310 | 000312 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 754 | 000312 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 755 | 000314 | 000316 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 756 | 000316 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 757 | 000320 | 000322 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 758 | 000322 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 759 | 000324 | 000326 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 760 | 000326 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 761 | 000330 | 000332 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 762 | 000332 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |

E02

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 17
 DZDQHC.P11 21-DEC-76 16:36 TRAPCATCHER FOR UNEXPECTED INTERRUPTS

| | | | | |
|-----|--------|--------|------|-----------------------------------|
| 763 | 000334 | 000336 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 764 | 000336 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 765 | 000340 | 000342 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 766 | 000342 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 767 | 000344 | 000346 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 768 | 000346 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 769 | 000350 | 000352 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 770 | 000352 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 771 | 000354 | 000356 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 772 | 000356 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 773 | 000360 | 000362 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 774 | 000362 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 775 | 000364 | 000366 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 776 | 000366 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 777 | 000370 | 000372 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 778 | 000372 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 779 | 000374 | 000376 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 780 | 000376 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 781 | 000400 | 000402 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 782 | 000402 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 783 | 000404 | 000406 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 784 | 000406 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 785 | 000410 | 000412 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 786 | 000412 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 787 | 000414 | 000416 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 788 | 000416 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 789 | 000420 | 000422 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 790 | 000422 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 791 | 000424 | 000426 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 792 | 000426 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 793 | 000430 | 000432 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 794 | 000432 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 795 | 000434 | 000436 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 796 | 000436 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 797 | 000440 | 000442 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 798 | 000442 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 799 | 000444 | 000446 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 800 | 000446 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 801 | 000450 | 000452 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 802 | 000452 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 803 | 000454 | 000456 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 804 | 000456 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 805 | 000460 | 000462 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 806 | 000462 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 807 | 000464 | 000466 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 808 | 000466 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 809 | 000470 | 000472 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 810 | 000472 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 811 | 000474 | 000476 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 812 | 000476 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 813 | 000500 | 000502 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 814 | 000502 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 815 | 000504 | 000506 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 816 | 000506 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 817 | 000510 | 000512 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 818 | 000512 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |

F02

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 18
 DZDQHC.P11 21-DEC-76 16:36 TRAPCATCHER FOR UNEXPECTED INTERRUPTS

| | | | | |
|-----|--------|--------|------|-----------------------------------|
| 819 | 000514 | 000516 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 820 | 000516 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 821 | 000520 | 000522 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 822 | 000522 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 823 | 000524 | 000526 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 824 | 000526 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 825 | 000530 | 000532 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 826 | 000532 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 827 | 000534 | 000536 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 828 | 000536 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 829 | 000540 | 000542 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 830 | 000542 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 831 | 000544 | 000546 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 832 | 000546 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 833 | 000550 | 000552 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 834 | 000552 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 835 | 000554 | 000556 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 836 | 000556 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 837 | 000560 | 000562 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 838 | 000562 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 839 | 000564 | 000566 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 840 | 000566 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 841 | 000570 | 000572 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 842 | 000572 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 843 | 000574 | 000576 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 844 | 000576 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 845 | 000600 | 000602 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 846 | 000602 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 847 | 000604 | 000606 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 848 | 000606 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 849 | 000610 | 000612 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 850 | 000612 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 851 | 000614 | 000616 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 852 | 000616 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 853 | 000620 | 000622 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 854 | 000622 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 855 | 000624 | 000626 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 856 | 000626 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 857 | 000630 | 000632 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 858 | 000632 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 859 | 000634 | 000636 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 860 | 000636 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 861 | 000640 | 000642 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 862 | 000642 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 863 | 000644 | 000646 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 864 | 000646 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 865 | 000650 | 000652 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 866 | 000652 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 867 | 000654 | 000656 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 868 | 000656 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 869 | 000660 | 000662 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 870 | 000662 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 871 | 000664 | 000666 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 872 | 000666 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 873 | 000670 | 000672 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 874 | 000672 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |

G02

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 19
DZDQHC.P11 21-DEC-76 16:36 TRAPCATCHER FOR UNEXPECTED INTERRUPTS

| | | | | |
|-----|--------|--------|------|-----------------------------------|
| 875 | 000674 | 000676 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 876 | 000676 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 877 | 000700 | 000702 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 878 | 000702 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 879 | 000704 | 000706 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 880 | 000706 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 881 | 000710 | 000712 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 882 | 000712 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 883 | 000714 | 000716 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 884 | 000716 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 885 | 000720 | 000722 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 886 | 000722 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 887 | 000724 | 000726 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 888 | 000726 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 889 | 000730 | 000732 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 890 | 000732 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 891 | 000734 | 000736 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 892 | 000736 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 893 | 000740 | 000742 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 894 | 000742 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 895 | 000744 | 000746 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 896 | 000746 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 897 | 000750 | 000752 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 898 | 000752 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 899 | 000754 | 000756 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 900 | 000756 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 901 | 000760 | 000762 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 902 | 000762 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 903 | 000764 | 000766 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 904 | 000766 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 905 | 000770 | 000772 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 906 | 000772 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |
| 907 | 000774 | 000776 | .+2 | :UNEXPECTED TRAP TO THIS LOCATION |
| 908 | 000776 | 000000 | HALT | :EXAMINE STACK TO FIND CAUSE |

H02

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 20
 DZDQHC.P11 21-DEC-76 16:36 ROUTINES USED FOR AUTO SIZING.

```

909                                     ;STANDARD INTERRUPT VECTORS
910
911                                     . =24
912 000024 015766                       .PFAIL                       ;POWER FAIL HANDLER
913 000026 000340                       340                          ;SERVICE AT LEVEL 7
914 000030 015436                       .HLT                          ;ERROR HANDLER
915 000032 000340                       340                          ;SERVICE AT LEVEL 7
916 000034 015404                       .TRPSRV                       ;GENERAL HANDLER DISPATCH SERVICE
917 000036 000340                       340                          ;SERVICE AT LEVEL 7
918
919 000046 014164                       . =46                       LOGICAL                       ;ACT HOOKS
920
921 000052 000000                       . =52                       .WORD 0
922                                     ; THIS ROUTINE TRIES TO FORCE THE RECEIVER TO INTERRUPT
923                                     ; TO ITS VECTOR WHERE IT WILL PICK UP THE STATUS LOCATION
924                                     ; FOR ITS NEW PC; AND PICK UP AN IOT INSTRUCTION FOR ITS
925                                     ; NEW PS. WHEN THE NEW PC IS FETCHED AN IOT INSTRUCTION IS
926                                     ; EXECUTED, TRAPPING TO LOCATION 20 WHERE A ROUTINE IS EXECUTED
927                                     ; TO TAKE THE PC FROM THE STACK AND USE IT AS THE VECTOR ADDRESS
928                                     . =56
929
930                                     VECMAP:
931 000056 010120 000004                 15:  MOV      R1,(R0)+          ;START FILLING THE VECTOR AREA
932 000060 012721 000004                 MOV      #4,(R1)+           ;WITH .+2; IOT (4)
933 000064 022021 000004                 CMP      (R0)+,(R1)+       ;UPDATE THE POINTERS
934 000066 020127 001000                 CMP      R1,#1000          ;IS ALL FLOATING VECTOR AREA DONE
935 000072 101771 000004                 BLOS    15                 ;BR IF NOT ALL DONE
936 000074 012737 000146 000020         MOV      #45,2#20          ;SET FOR IOT TRAP BY DQ11
937 000102 013737 001500 001244         MOV      DQACTV,TEMP1     ;GET THE ACTIVE DQ11 S
938 000110 006037 001244                 25:  ROR      TEMP1          ;ARE YOU ACTIVE.. DQ11
939 000114 103023 000004                 BCC     5$                 ;IF CARRY CLEAR.. NO MORE DQ11S
940 000116 005037 177776                 CLR     PS                 ;CLEAR PS
941 000122 005722 000004                 TST     (R2)+              ;PUT POINTER TO STATUS TABLE
942 000124 012772 000340 177776         MOV      #340,2-2(R2)     ;TRY AND SET PRI/SEC DONE AND IE
943 000132 105200 000004                 INCB    RO                 ;DELAY.....DELAY
944 000134 001376 000004                 BNE     -2                 ;.....DELAY
945 000136 112712 000300                 MOV     #300,(R2)         ;NO INTERRUPT ASSUME 300 FIX IN TEST C
946 000142 005722 000004                 35:  TST     (R2)+          ;UPDATE POINTERS
947 000144 000761 000004                 BR      2$                 ;GO DO IT AGAIN
948 000146 051612 000007                 45:  BIS     (SP),(R2)      ;ENTERD BY IOT TRAP BY DQ11
949 000150 042712 000007                 BIC     #7,(R2)           ;CLEAR UNWANTED BITS
950 000154 022626 000007                 CMP     (SP)+,(SP)+       ;POP IOT JUNK OFF STACK
951 000156 012716 000142                 MOV     #3$, (SP)         ;SET RETURN PC ON STACK
952 000162 000002 000007                 RTI
953 000164 000207 000007                 55:  RTS      PC           ;GO HOME
954                                     ;ALL SIZING IS DONE
955
956                                     ;****SOFTWARE SWITCH REGISTER****
957 000174 000000 000174                 . =174
958 000176 000000 000000                 DISPREG: 0                 ;SOFTWARE DISPLAY REGISTER
959                                     SWREG: 0                   ;SOFTWARE SWITCH REGISTER
960
961                                     ;PROGRAM START
962
963 000200 000137 001512                 . =200                       JMP      .START           ;GO TO START OF PROGRAM
964

```

DZDGH MACY11 27(1006) 22-DEC-76 11:40 PAGE 21
 DZDGH.C.P11 21-DEC-76 16:36 ROUTINES USED FOR AUTO SIZING.

| | | | | | | | | | | | |
|------|--------|--------|--------|--------|------|-------------|--------------|--|--|--|---|
| 965 | | 000220 | | | | | | | | | |
| 966 | 000220 | 012702 | 001400 | | =220 | CSRMAP: MOV | #1400,R2 | | | | :CLEAR ALL STATUS TABLE |
| 967 | 000224 | 005022 | | | | CLR | (R2)+ | | | | :DO CLEAR |
| 968 | 000226 | 022702 | 001512 | | | CMP | #1512,R2 | | | | :ALL TABLE DONE |
| 969 | 000232 | 001374 | | | | BNE | .-6 | | | | :BR IF MORE TO GO |
| 970 | 000234 | 005037 | 001504 | | | CLR | DQNUM | | | | :SET NUMBER OF DQ11S TO 0 |
| 971 | 000240 | 012702 | 001400 | | | MOV | #1400,R2 | | | | :SET TABLE POINTER |
| 972 | 000244 | 012701 | 160000 | | | MOV | #160000,R1 | | | | :GET FIRST FLOATING ADDRESS |
| 973 | 000250 | 012737 | 000614 | 000004 | | MOV | #55,2#4 | | | | :SET FOR TIME OUT TRAP--NO DEVICE-- |
| 974 | 000256 | 112761 | 000012 | 000005 | 15: | MOVB | #12,5(R1) | | | | :TRY AND SEL MISC REGISTER |
| 975 | 000264 | 005061 | 000006 | | | CLR | 6(R1) | | | | :TRY AND CLEAR MISC REG |
| 976 | 000270 | 012711 | 010000 | | | MOV | #10000,(R1) | | | | :TRY AND SET RX ACTIVE |
| 977 | 000274 | 022761 | 030000 | 000006 | | CMP | #30000,6(R1) | | | | :LOOK FOR SYNC 1 AND SYNC 2 |
| 978 | 000302 | 001071 | | | | BNE | 25 | | | | :THIS IS NOT A DQ11 IF I BRANCH |
| 979 | 000304 | 010122 | | | | MOV | R1,(R2)+ | | | | :NOW THIS IS A DQ11 --STORE CSR |
| 980 | 000306 | 052712 | 100000 | | | BIS | #SYNBIT,(R2) | | | | :SET FOR TWO SYNC CHARS |
| 981 | 000312 | 005011 | | | | CLR | (R1) | | | | :CLEAR DQ ACTIVE BIT |
| 982 | 000314 | 112761 | 000010 | 000005 | | MOVB | #10,5(R1) | | | | :SEL CHAR DET REGISTER |
| 983 | 000322 | 012761 | 177777 | 000006 | | MOV | #-1,6(R1) | | | | :WRITE INTO CHAR DET REG |
| 984 | 000330 | 005761 | 000006 | | | TST | 6(R1) | | | | :WAS THE REGISTER WRITTEN? |
| 985 | 000334 | 001402 | | | | BEQ | .-+6 | | | | :APPARENTLY NO BB OPTION. |
| 986 | 000336 | 052712 | 020000 | | | BIS | #BBBIT,(R2) | | | | :SET FOR BB OPTION |
| 987 | 000342 | 112761 | 000017 | 000005 | | MOVB | #17,5(R1) | | | | :SEL POLYNO. REGISTER |
| 988 | 000350 | 012761 | 177777 | 000006 | | MOV | #-1,6(R1) | | | | :WRITE POLYNO.REGISTER |
| 989 | 000356 | 005761 | 000006 | | | TST | 6(R1) | | | | :WAS REG WRITTEN?? |
| 990 | 000362 | 001402 | | | | BEQ | .-+6 | | | | :BR IF NO AB OPTION |
| 991 | 000364 | 052712 | 002000 | | | BIS | #ABBIT,(R2) | | | | :SET FOR AB OPTION |
| 992 | 000370 | 012761 | 001400 | 000002 | | MOV | #1400,2(R1) | | | | :TRY TO SET DTR. RS. |
| 993 | 000376 | 032761 | 001400 | 000002 | | BIT | #1400,2(R1) | | | | :DID ANY OF THEM SET |
| 994 | 000404 | 001402 | | | | BEQ | .-+6 | | | | :BR IF NO BA OPTION |
| 995 | 000406 | 052712 | 010000 | | | BIS | #BABIT,(R2) | | | | :SET FOR BA OPTION |
| 996 | 000412 | 032761 | 030000 | 000002 | | BIT | #30000,2(R1) | | | | :DID CS. CO. SET |
| 997 | 000420 | 001402 | | | | BEQ | .-+6 | | | | :BR IF NO JUMPER |
| 998 | 000422 | 052712 | 040000 | | | BIS | #JUMBIT,(R2) | | | | :SET FOR JUMPER |
| 999 | 000426 | 052712 | 004000 | | | BIS | #ACTBIT,(R2) | | | | :SET FOR ACTIVE ON FIRST NON-SYNC |
| 1000 | 000432 | 052712 | 001000 | | | BIS | #ODDBIT,(R2) | | | | :SET FOR ODD VRC..... |
| 1001 | 000436 | 005722 | | | | TST | (R2)+ | | | | :POP POINTER |
| 1002 | 000440 | 005011 | | | | CLR | (R1) | | | | :CLEAR RCSR |
| 1003 | 000442 | 005061 | 000002 | | | CLR | 2(R1) | | | | :CLEAR TCSR |
| 1004 | 000446 | 005061 | 000002 | | | CLR | 2(R1) | | | | :CLEAR AGAIN |
| 1005 | 000452 | 005061 | 000004 | | | CLR | 4(R1) | | | | :CLEAR ERROR REG |
| 1006 | 000456 | 005061 | 000006 | | | CLR | 6(R1) | | | | :CLEAR SEC REG |
| 1007 | 000462 | 005237 | 001504 | | | INC | DQNUM | | | | :UPDATE NUMBER OF DQ11S |
| 1008 | 000466 | 062701 | 000010 | | 25: | ADD | #10,R1 | | | | :UPDATE CSR POINTER BY 10 (8) |
| 1009 | 000472 | 022701 | 164000 | | | CMP | #164000,R1 | | | | :HAVE ALL FLOATING ADDRESSES BEEN CHECKED?? |
| 1010 | 000476 | 001267 | | | | BNE | 15 | | | | :BR IF NOT ALL DONE |
| 1011 | 000500 | 005037 | 001500 | | | CLR | DQACTV | | | | :ZERO ACTIVE DQ11S |
| 1012 | 000504 | 005737 | 001504 | | | TST | DQNUM | | | | :WERE ANY DQ11S FOUND |
| 1013 | 000510 | 001434 | | | | BEQ | 45 | | | | :HEY BUDDY. NO DQ11S FOUND IN SYSTEM |
| 1014 | 000512 | 013701 | 001504 | | | MOV | DQNUM,R1 | | | | :SAVE NUMBER OF DQ11S |
| 1015 | 000516 | 010137 | 001276 | | | MOV | R1,SAVNUM | | | | :SAVE NUMBER FOR ACT11 |
| 1016 | 000522 | 000241 | | | 35: | CLC | | | | | :CLEAR CARRY |
| 1017 | 000524 | 006137 | 001500 | | | ROL | DQACTV | | | | :***** ACTIVE ADDRESS |
| 1018 | 000530 | 005237 | 001500 | | | INC | DQACTV | | | | :SET BIT 0 |
| 1019 | 000534 | 005301 | | | | DEC | R1 | | | | :DEC NUMBER OF DQ11S |
| 1020 | 000536 | 001371 | | | | BNE | 35 | | | | :BR IF MORE TO GO |

J02

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 22
DZDQHC.P11 21-DEC-76 16:36

ROUTINES USED FOR AUTO SIZING.

```

1021 000540 012737 000006 000004      MOV      #6, R#4      ;RESET TIME OUT VECTOR
1022 000546 013737 001500 001502      MOV      DQACTV, SAVACT ;SAVE ACTIVE
1023 000554 012737 000340 000022      MOV      #340, R#22   ;SET IOT TRAP PRIO: TO 7
1024 000562 012702 001400              MOV      #1400, R2    ;SET TABLE POINTER
1025 000566 012700 000300              MOV      #300, R0     ;SET VECTOR START
1026 000572 012701 000302              MOV      #302, R1     ;SET VECTOR+2 START
1027 000576 000137 000056              JMP      VECPMAP      ;GO FIND THE VECTORS
1028 000602 104402              4S:     TYPE          ;TYPE MESSAGE
1029 000604 016327              MERR2    ;I DIDN'T FIND ANY DQ11S. DON'T USE AUTO SIZE.
1030 000606 005000              CLR      R0          ;
1031 000610 000000              HALT                    ;HOW CAN I TEST NO DQ11S
1032 000612 000776              BR      #-2           ;DON'T LET OPR HIT CONT. SW
1033 000614 012716 000466      5S:     MOV      #25, (SP) ;ENTERED BY TIME OUT TRAP
1034 000620 000002              RTI                    ;GO HOME.
1035
1036
1037
1038 001000 005377 040515 047111      .=1000  MTITLE: .ASCIZ <377><12>/MAINDEC-11-DZDQH-C/<377>/CHARACTER LENGTH AND INTERRUPT TESTS/
1039 001006 042504 026503 030461
1040 001014 042055 042132 044121
1041 001022 041455 041777 040510
1042 001030 040522 052103 051105
1043 001036 046040 047105 052107
1044 001044 020110 047101 020104
1045 001052 047111 042524 051122
1046 001060 050125 020124 042524
1047 001066 052123 177523      000
1048
1049      001200      .=1200
1050      ;INDIRECT POINTERS
1051
1052 001200 177570      SWR:      177570      ;SWITCH REGISTER POINTER
1053 001202 177570      LIGHTS:   177570     ;DISPLAY REGISTER POINTER
1054 001204 177560      TKCSR:    177560     ;TELETYPE KEYBOARD CONTROL REGISTER
1055 001206 177562      TKDBR:    177562     ;TELETYPE KEYBOARD DATA BUFFER
1056 001210 177564      TPCSR:    177564     ;TELEPRINTER CONTROL REGISTER
1057 001212 177566      TPDBR:    177566     ;TELEPRINTER DATA BUFFER
1058
1059      ;PROGRAM CONTROL PARAMETERS
1060
1061 001214 000000      RETURN:   0          ;SCOPE ADDRESS FOR LOOP ON TEST
1062 001216 000000      NEXT:     0          ;ADDRESS OF NEXT TEST TO BE EXECUTED
1063 001220 000000      LOCK:     0          ;ADDRESS FOR LOCK ON CURRENT DATA
1064 001222 000003      ICOUNT: 3          ;NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE EXECUTED
1065 001224 000000      LPCNT:    0          ;NUMBER OF ITERATIONS COMPLETED
1066 001226 000000      TSTNO:    0          ;NUMBER OF TEST IN PROGRESS
1067 001230 000000      PASCNT:   0          ;NUMBER OF PASSES COMPLETED
1068 001232 000000      ERRCNT:   0          ;TOTAL NUMBER OF ERRORS
1069 001234 000000      LSTERR:   0          ;PC OF LAST ERROR CALL
1070
1071      ;PROGRAM VARIABLES
1072
1073 001236 000000      CHAR1:    0
1074 001240 000000      CHAR2:    0
1075 001242 000000      CHAR3:    0
1076 001244 000000      TEMP1:    0          ;TEMPORARY STORAGE

```

K02

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 23
DZDQHC.P11 21-DEC-76 16:36

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

| | | | | | |
|------|--------|--------|---------|---------|---------------------------|
| 1077 | 001246 | 000000 | TEMP2: | 0 | : TEMPORARY STORAGE |
| 1078 | 001250 | 000000 | TEMP3: | 0 | : TEMPORARY STORAGE |
| 1079 | 001252 | 000000 | TEMP4: | 0 | : TEMPORARY STORAGE |
| 1080 | 001254 | 000000 | TEMPS: | 0 | : TEMPORARY STORAGE |
| 1081 | 001256 | 000000 | SAVR0: | 0 | : R0 STORAGE |
| 1082 | 001260 | 000000 | SAVR1: | 0 | : R1 STORAGE |
| 1083 | 001262 | 000000 | SAVR2: | 0 | : R2 STORAGE |
| 1084 | 001264 | 000000 | SAVR3: | 0 | : R3 STORAGE |
| 1085 | 001266 | 000000 | SAVR4: | 0 | : R4 STORAGE |
| 1086 | 001270 | 000000 | SAVR5: | 0 | : R5 STORAGE |
| 1087 | 001272 | 000000 | SAVSP: | 0 | : STACK POINTER STORAGE |
| 1088 | 001274 | 000000 | SAVPC: | 0 | : PROGRAM COUNTER STORAGE |
| 1089 | 001276 | 000000 | SAVNUM: | 0 | |
| 1090 | 001300 | 000001 | CREAM: | .BLKW 1 | |
| 1091 | 001302 | 000000 | RUNFLG: | 0 | |
| 1092 | 001304 | 000000 | RUN: | 0 | |
| 1093 | 001306 | 000000 | RUNCNT: | 0 | |

L02

DZDGM MACY11 27(1006) 22-DEC-76 11:40 PAGE 24
 DZDGM.C.P11 21-DEC-76 16:36 PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

```

1094
1095
1096
1097 001310 000
1098 001311 000
1099 001312 000
1100 001313 000
1101 000000
1102
1103
1104
1105
1106
1107
1108
1109 001314
1110 001314 104400
1111 001314 014240
1112 001316 014352
1113 001316 014352
1114 001320 014372
1115 001322 014500
1116 001322 014500
1117 001324 014616
1118 001324 014616
1119 001326 014650
1120 001326 014650
1121 001330 015064
1122 001330 015064
1123 001332 015124
1124 001332 015124
1125 001334 015156
1126 001334 015156
1127 001336 015162
1128 001336 015162
1129 001340 012576
1130 001340 012576
1131 001342 012452
1132 001342 012452
1133 001344 016064
1134 001344 016064
1135 001346 016140
1136 001346 016140
1137
1138
1139
1140
1141
1142
1143
1144 001350 000000
1145 001352 000000
1146 001354 000000
1147 001356 000000
1148 001360 000000
1149 001362 000000

;PROGRAM CONTROL FLAGS
INIFLG: .BYTE 0 ;PROGRAM INITIALIZATION FLAG
STFLG: .BYTE 0 ;TEST START FLAG
ERRFLG: .BYTE 0 ;ERROR OCCURED FLAG
LOKFLG: .BYTE 0 ;LOCK ON CURRENT TEST FLAG
SY=0

;DEFINITIONS FOR TRAP SUBROUTINE CALLS
;POINTERS TO SUBROUTINES CAN BE FOUND
;IN THE TABLE IMMEDIATLY FOLLOWING THE DEFINITIONS

;*****
;*****
TRPTAB:
SCOPE=TRAP+0 ;CALL TO SCOPE LOOP AND ITERATION HANDLER
SCOPE
SCOPI=TRAP+1 ;CALL TO LOOP ON CURRENT DATA HANDLER
SCOPI
TYPE=TRAP+2 ;CALL TO TELETYPE OUTPUT ROUTINE
TYPE
INSTR=TRAP+3 ;CALL TO ASCII STRING INPUT ROUTINE
INSTR
INSTER=TRAP+4 ;CALL TO INPUT ERROR HANDLER
INSTER
PARAM=TRAP+5 ;CALL TO NUMERICAL DATA INPUT ROUTINE
PARAM
SAVDS=TRAP+6 ;CALL TO REGISTER SAVE ROUTINE
SAVDS
RESOS=TRAP+7 ;CALL TO REGISTER RESTORE ROUTINE
RESOS
CONVRT=TRAP+10 ;CALL TO DATA OUTPUT ROUTINE
CONVRT
CNVRT=TRAP+11 ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
CNVRT
MSTCLR=TRAP+12 ;CALL TO ISSUE MASTER CLEAR
MSTCLR
MEMCLR=TRAP+13 ;CALL TO CLEAR ALL SCRATCH PAD MEMORIES
MEMCLR
CKSWR=TRAP+14 ;CALL TO ALLOW SWREG TO BE LOADED FROM TTY
CKSWR
CNTLU=TRAP+15 ;CALL TO ALLOW LOADING OF SWREG FROM TTY
.CNTLU

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

;DQ11 VECTOR AND REGISTER INDIRECT POINTERS
DQ11VEC: 0 ;POINTER TO DQ11 RECEIVER INTERRUPT VECTOR
DQ11LVL: 0 ;POINTER TO DQ11 RECEIVER INTERRUPT SERVICE PS
DQ11VEC: 0 ;POINTER TO DQ11 TRANSMITTER INTERRUPT VECTOR
DQ11LVL: 0 ;POINTER TO DQ11 TRANSMITTER INTERRUPT SERVICE PS
DQ11CSR: 0 ;POINTER TO DQ11 RECEIVER CONTROL REGISTER
DQ11CSH: 0 ;POINTER TO HIGH BYTE OF DQ11 RECEIVER CONTROL REGISTER

```

M02

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 25
 DZDQHC.P11 21-DEC-76 16:36

PROGRAM PARAMETERS, VARIABLES, AND TRAP CALLS.

| | | | | |
|------|--------|--------|-----------|---|
| 1150 | 001364 | 000000 | DQTCR: 0 | : POINTER TO DQ11 TRANSMITTER CONTROL REGISTER |
| 1151 | 001366 | 000000 | DQERR: 0 | : POINTER TO DQ11 ERROR REGISTER |
| 1152 | 001370 | 000000 | DQREG: 0 | : POINTER TO HIGH BYTE OF ERROR REGISTER |
| 1153 | 001372 | 000000 | DQSEC: 0 | : POINTER TO DQ11 SECONDARY REGISTER |
| 1154 | 001374 | 000000 | DQSECH: 0 | : POINTER TO HIGH BYTE OF DQ11 SECONDARY REGISTER |

1155
1156
1157
1158
1159

;DQ11 STATUS TABLE AND ADDRESS ASSIGNMENTS

| | | | | | |
|------|--------|--------|---------------|---|--|
| 1160 | | 001400 | . =1400 | | |
| 1161 | 001400 | 000001 | DQCR00: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 00 |
| 1162 | 001402 | 000001 | DQST00: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 00 |
| 1163 | 001404 | 000001 | DQCR01: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 01 |
| 1164 | 001406 | 000001 | DQST01: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 01 |
| 1165 | 001410 | 000001 | DQCR02: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 02 |
| 1166 | 001412 | 000001 | DQST02: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 02 |
| 1167 | 001414 | 000001 | DQCR03: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 03 |
| 1168 | 001416 | 000001 | DQST03: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 03 |
| 1169 | 001420 | 000001 | DQCR04: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 04 |
| 1170 | 001422 | 000001 | DQST04: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 04 |
| 1171 | 001424 | 000001 | DQCR05: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 05 |
| 1172 | 001426 | 000001 | DQST05: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 05 |
| 1173 | 001430 | 000001 | DQCR06: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 06 |
| 1174 | 001432 | 000001 | DQST06: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 06 |
| 1175 | 001434 | 000001 | DQCR07: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 07 |
| 1176 | 001436 | 000001 | DQST07: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 07 |
| 1177 | 001440 | 000001 | DQCR10: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 10 |
| 1178 | 001442 | 000001 | DQST10: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 10 |
| 1179 | 001444 | 000001 | DQCR11: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 11 |
| 1180 | 001446 | 000001 | DQST11: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 11 |
| 1181 | 001450 | 000001 | DQCR12: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 12 |
| 1182 | 001452 | 000001 | DQST12: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 12 |
| 1183 | 001454 | 000001 | DQCR13: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 13 |
| 1184 | 001456 | 000001 | DQST13: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 13 |
| 1185 | 001460 | 000001 | DQCR14: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 14 |
| 1186 | 001462 | 000001 | DQST14: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 14 |
| 1187 | 001464 | 000001 | DQCR15: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 15 |
| 1188 | 001466 | 000001 | DQST15: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 15 |
| 1189 | 001470 | 000001 | DQCR16: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 16 |
| 1190 | 001472 | 000001 | DQST16: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 16 |
| 1191 | 001474 | 000001 | DQCR17: .BLKW | 1 | : CONTROL STATUS REGISTER FOR DEVICE NO: 17 |
| 1192 | 001476 | 000001 | DQST17: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS FOR DEVICE NO: 17 |
| 1193 | 001500 | 000001 | DQACTV: .BLKW | 1 | : HOLD ACTIVE BITS FOR TESTING |
| 1194 | 001502 | 000001 | SAVACT: .BLKW | 1 | : SAVE NUMBER OF ACTIVE DQ11s |
| 1195 | 001504 | 000001 | DQNUM: .BLKW | 1 | : OCTAL NUMBER OF TOTAL NUMBER OF DQ11s |
| 1196 | 001506 | 000001 | DQCSR: .BLKW | 1 | : CSR OF DQ11 UNDER TEST |
| 1197 | 001510 | 000001 | DQSTAT: .BLKW | 1 | : VECTOR AND CONFIGURATION STATUS OF DQ11 UNDER TEST |

1198
1199
1200
1201
1202
1203
1204
1205

:PROGRAM INITIALIZATION
 :LOCK OUT INTERRUPTS
 :SET UP PROCESSOR STACK
 :SET UP POWER FAIL VECTOR
 :CLEAR PROGRAM CONTROL FLAGS AND COUNTS
 :TYPE TITLE MESSAGE

N02

DZDGM NACY11 27(1006) 22-DEC-76 11:40 PAGE 26
 DZDGM.P11 21-DEC-76 16:36 PROGRAM INITIALIZATION AND START UP.

| | | | | | | | | |
|------|--------|--------|--------|--------|---------|--------|-----------------|-------------------------------------|
| 1206 | 001512 | 012737 | 000340 | 177776 | .START: | MOV | #340,PS | ;LOCK OUT INTERRUPTS |
| 1207 | 001520 | 012706 | 001200 | | | MOV | #STACK,SP | ;SET UP STACK |
| 1208 | 001524 | 012737 | 015766 | 000024 | | MOV | #PFAIL,#24 | ;SET UP POWER FAIL VECTOR |
| 1209 | 001532 | 013737 | 001504 | 001276 | | MOV | DONUM,SAVNUM | |
| 1210 | 001540 | 105037 | 001311 | | | CLRB | STFLG | ;CLEAR START FLAG |
| 1211 | 001544 | 005037 | 001230 | | | CLR | PASCNT | ;CLEAR PASS COUNT |
| 1212 | 001550 | 105037 | 001312 | | | CLRB | ERRFLG | ;CLEAR ERROR FLAG |
| 1213 | 001554 | 005037 | 001302 | | | CLR | RUNFLG | |
| 1214 | 001560 | 012737 | 001400 | 001300 | | MOV | #1400,CREAM | |
| 1215 | 001566 | 005037 | 001232 | | | CLR | ERRCNT | ;CLEAR ERROR COUNT |
| 1216 | 001572 | 005037 | 001234 | | | CLR | LSTERR | ;CLEAR LAST ERROR POINTER |
| 1217 | 001576 | 012737 | 000001 | 001226 | | MOV | #1,TSTNO | ;SET UP FOR TEST 1 |
| 1218 | 001604 | 012737 | 001512 | 001214 | | MOV | #.START,RETURN | ;SET UP FOR POWER FAIL BEFORE |
| 1219 | | | | | | | | ;TESTING STARTS |
| 1220 | 001612 | 105737 | 001310 | | | TSTB | INIFLG | ;HAS INITIALIZATION BEEN PERFORMED |
| 1221 | 001616 | 001075 | | | | BNE | 125 | |
| 1222 | 001620 | 104402 | 001000 | | | TYPE | MTITLE | ;TYPE TITLE MESSAGE |
| 1223 | 001624 | 105137 | 001310 | | | COMB | INIFLG | ;IF NOT SET FLAG AND DO |
| 1224 | | | | | | | | |
| 1225 | 001630 | 012737 | 177570 | 001200 | | MOV | #DSWR,SWR | ;MOV HARDWARE SWR TO SWR |
| 1226 | 001636 | 012737 | 177570 | 001202 | | MOV | #DLIGHTS,LIGHTS | ;MOV DISPLAY LIGHTS TO LIGHTS |
| 1227 | 001644 | 013746 | 000006 | | | MOV | #286,-(SP) | ;SAVE VECTORS |
| 1228 | 001650 | 013746 | 000004 | | | MOV | #284,-(SP) | |
| 1229 | 001654 | 012737 | 001674 | 000004 | | MOV | #648,#284 | ;SET UP FOR TIMEOUT |
| 1230 | 001662 | 022777 | 177777 | 177310 | | CMP | #-1,#SWR | ;REFERENCE HARDWARE SWITCH REGISTER |
| 1231 | 001670 | 001402 | | | | BEQ | 655 | |
| 1232 | 001672 | 000407 | | | | BR | 665 | |
| 1233 | 001674 | 022626 | | | 645: | CMP | (SP)+,(SP)+ | ;ADJUST STACK |
| 1234 | 001676 | 012737 | 000176 | 001200 | 655: | MOV | #SWREG,SWR | ;POINT TO SOFTWARE SWITCH REG |
| 1235 | 001704 | 012737 | 000174 | 001202 | | MOV | #DISPREG,LIGHTS | ;POINT TO SOFT DISPLAY REG |
| 1236 | 001712 | 012637 | 000004 | | 665: | MOV | (SP)+,#284 | ;RESTORE VECTORS |
| 1237 | 001716 | 012637 | 000006 | | | MOV | (SP)+,#286 | |
| 1238 | 001722 | 005737 | 000042 | | | TST | #282 | ;UNDER MONITOR |
| 1239 | 001726 | 001005 | | | | BNE | 675 | |
| 1240 | 001730 | 022737 | 000176 | 001200 | | CMP | #SWREG,SWR | ;IS SWREG USED |
| 1241 | 001736 | 001001 | | | | BNE | 675 | |
| 1242 | 001740 | 104415 | | | | CNTLU | | |
| 1243 | 001742 | 105777 | 177232 | | 675: | TSTB | #SWR | |
| 1244 | 001746 | 100402 | | | | BMI | +6 | |
| 1245 | 001750 | 004737 | 000220 | | | JSR | PC,CSRMAP | |
| 1246 | 001754 | 104402 | 016614 | | | TYPE | XHEAD | |
| 1247 | 001760 | 012737 | 001400 | 001244 | | MOV | #1400,TEMP1 | |
| 1248 | 001766 | 017737 | 177252 | 001246 | | MOV | #TEMP1,TEMP2 | |
| 1249 | 001774 | 001406 | | | | BEQ | +.16 | |
| 1250 | 001776 | 104410 | | | | CONVRT | | |
| 1251 | 002000 | 016642 | | | | XSTAT0 | | |
| 1252 | 002002 | 062737 | 000002 | 001244 | | ADD | #2,TEMP1 | |
| 1253 | 002010 | 000766 | | | | BR | .-22 | |
| 1254 | 002012 | 032777 | 000001 | 177160 | 125: | BIT | #SM00,#SWR | |
| 1255 | 002020 | 001424 | | | | BEQ | 15 | |
| 1256 | 002022 | 104402 | | | | TYPE | | |
| 1257 | 002024 | 016636 | | | | MNEW | | |
| 1258 | 002026 | 005000 | | | | CLR | RD | |
| 1259 | 002030 | 000000 | | | | HALT | | |
| 1260 | 002032 | 104414 | | | | CKSWR | | |
| 1261 | 002034 | 027737 | 177140 | 001502 | | CMP | #SWR,SAVACT | |

B03

DZDGH MACY11 27(1006) 22-DEC-76 11:40 PAGE 27
 DZDGH.C.P11 21-DEC-76 16:36 PROGRAM INITIALIZATION AND START UP.

```

1262 002042 101404      BLOS      11$
1263 002044 104402      TYPE
1264 002046 016376      MERR3
1265 002050 000000      HALT
1266 002052 000776      BR      -2
1267 002054 017737 177120 001500 11$:  MOV      @SWR,DQACTV
1268 002062 013700 001500      MOV      DQACTV,RO
1269 002066 000000      HALT
1270 002070 104414      CKSWR
1271 002072 012700 000300 1$:  MOV      @300,RO
1272 002076 012701 000302      MOV      @302,R1
1273 002102 010120 2$:  MOV      R1,(R0)+
1274 002104 005021      CLR      (R1)+
1275 002106 022021      CMP      (RO)+,(R1)+
1276 002110 022700 001000      CMP      @1000,RO
1277 002114 001372      BNE      2$
1278
1279                      ;TEST START AND RESTART
1280
1281 002116 012737 000340 177776 .BEGIN: MOV      @340,PS          ;LOCK OUT INTERRUPTS
1282 002124 012706 001200      MOV      @STACK,SP      ;SET UP STACK
1283 002130 005737 000042      TST      @#42          ;IS PROGRAM UNDER MONITOR CONTROL
1284 002134 001040      BNE      3$
1285 002136 104414      CKSWR          ;CHECK FOR <↑G>
1286 002140 032777 000004 177032      BIT      @BIT2,@SWR    ;CHECK FOR LOCK ON TEST
1287 002146 001411      BEQ      1$
1288 002150 104402 016434      TYPE      MLOCK
1289 002154 012737 000240 014250      MOV      @NOP,TTST
1290 002162 012737 000240 014252      MOV      @NOP,TTST+2   ;SET UP TO LOCK
1291 002170 000406      BR      2$
1292 002172 013737 014346 014250 1$:  MOV      BRW,TTST
1293 002200 013737 014350 014252      MOV      BRX,TTST+2
1294 002206 032777 000002 176764 2$:  BIT      @SW01,@SWR    ;LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
1295 002214 001410      BEQ      3$          ;IF SW01=1, GET STARTING PC
1296 002216 104403      INSTR
1297 002220 016422      MTSTPC
1298 002222 104405      PARAM
1299 002224 002254      TST1
1300 002226 011622      TLAST
1301 002230 001214      @RETURN
1302 002232      001      .BYTE 1
1303 002233      001      .BYTE 1
1304 002234 000403      BR      4$
1305 002236 012737 002254 001214 3$:  MOV      @TST1,RETURN   ;START AT TEST 1
1306 002244 104402 016324 4$:  TYPE      MR          ;TYPE R
1307 002250 000177 176740      JMP      @RETURN      ;START TESTING
1308
1309                      ; TEST 1
1310 002254 012737 000001 001226 TST1: MOV      @1,TSTNO
1311 002262 012737 002644 001214      MOV      @TST2,RETURN
1312 002270 012737 002644 001216      MOV      @TST2,NEXT
1313 002276 105737 001302      TSTB     RUNFLG      ;IS THIS MY FIRST TIME HERE?
1314 002302 001010      BNE      1$          ;BR IF FLAG IS SET
1315 002304 012737 000001 001304      MOV      @BIT0,RUN    ;SET RUN POINTER.
1316 002312 012737 000020 001306      MOV      @16,RUNCNT  ;SET FOR MAX OF 16 DQ11'S PER SYSTEM
1317 002320 105137 001302      COMB     RUNFLG      ;SET RUN FLAG

```

C03

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 28
 DZDQHC.P11 21-DEC-76 16:36 PROGRAM INITIALIZATION AND START UP.

| | | | | | | | | |
|------|--------|--------|--------|--------|------|-------|-----------------|--|
| 1318 | 002324 | 033737 | 001304 | 001500 | 15: | BIT | RUN,DQACTV | ;FIND AN ACTIVE DQ11 TO TEST. |
| 1319 | 002332 | 001032 | | | | BNE | 3\$ | ;BR IF I FOUND ONE TO TEST. |
| 1320 | 002334 | 005737 | 001500 | | | TST | DQACTV | ;FIND OUT IF THERE ARE NO DQ11 ACTIVE. |
| 1321 | 002340 | 001423 | | | | BEQ | 2\$ | ;BR TO FATAL ERROR. WHY AM I HERE IF NO ACTIVE DQ11'S??? |
| 1322 | 002342 | 000257 | | | | CCC | | ;CLEAR ALL THE CONDITION CODES OF CPU |
| 1323 | 002344 | 006137 | 001304 | | | ROL | RUN | ;UPDATE RUN POINTER |
| 1324 | 002350 | 062737 | 000004 | 001300 | | ADD | #4,CREAM | ;UPDATE ADDRESS POINTER. |
| 1325 | 002356 | 005337 | 001306 | | | DEC | RUNCNT | ;DEC NUMBER OF TIMES I LOOKED AT ACTIVE. |
| 1326 | 002362 | 001360 | | | | BNE | 1\$ | ;BR AND KEEP LOOKING. |
| 1327 | 002364 | 012737 | 000020 | 001306 | | MOV | #16,RUNCNT | ;START RESTORING MY POINTERS. |
| 1328 | 002372 | 012737 | 001400 | 001300 | | MOV | #1400,CREAM | ;RESTORE ADDRESS POINTER |
| 1329 | 002400 | 012737 | 000001 | 001304 | | MOV | #1,RUN | ;RESTORE RUN POINTER. |
| 1330 | 002406 | 000746 | | | | BR | 1\$ | ;KEEP ON TESTING. |
| 1331 | 002410 | 104402 | | | 2\$: | TYPE | | ;ALLERT OPERATOR OF FATAL ERROR |
| 1332 | 002412 | 016327 | | | | MERR2 | | ;NO DQ11 ACTIVE. WHY AM I HERE??? |
| 1333 | 002414 | 000000 | | | | HALT | | ;YOU MUST RELOAD DQ11 DIAGNOSTIC!! |
| 1334 | 002416 | 000776 | | | | BR | | ;STICK HERE ON CONT. |
| 1335 | 002420 | 000257 | | | 3\$: | CCC | | ;CLEAR CPU COND. CODES |
| 1336 | 002422 | 006137 | 001304 | | | ROL | RUN | ;UPDATE RUN. ACTIVE DQ11 FOUND. |
| 1337 | 002426 | 017737 | 176646 | 001506 | | MOV | #2,CREAM,DQCSR | ;PLACE ADDRESS OF DQ11 AT DQCSR |
| 1338 | 002434 | 062737 | 000002 | 001300 | | ADD | #2,CREAM | ;UPDATE ADDRESS POINTER |
| 1339 | 002442 | 017737 | 176632 | 001510 | | MOV | #2,CREAM,DQSTAT | ;PLACE STATUS OF DQ11 AT DQSTAT |
| 1340 | 002450 | 062737 | 000002 | 001300 | | ADD | #2,CREAM | ;UPDATE ADDRESS POINTER |
| 1341 | 002456 | 013737 | 001506 | 001360 | | MOV | DQCSR,DQCSR | |
| 1342 | 002464 | 013737 | 001510 | 001350 | | MOV | DQSTAT,DQVEC | |
| 1343 | 002472 | 042737 | 177007 | 001350 | | BIC | #177007,DQVEC | |
| 1344 | 002500 | 013737 | 001350 | 001352 | | MOV | DQVEC,DQRLVL | ;GENERATE ADDRESS OF RECEIVER INTERRUPT SERVICE PS |
| 1345 | 002506 | 062737 | 000002 | 001352 | | ADD | #2,DQRLVL | |
| 1346 | 002514 | 013737 | 001352 | 001354 | | MOV | DQRLVL,DQTEC | ;GENERATE ADDRESS OF TRANSMITTER INTERRUPT VECTOR |
| 1347 | 002522 | 062737 | 000002 | 001354 | | ADD | #2,DQTEC | |
| 1348 | 002530 | 013737 | 001354 | 001356 | | MOV | DQTEC,DQTLVL | ;GENERATE ADDRESS OF TRANSMITTER INTERRUPT SERVICE PS |
| 1349 | 002536 | 062737 | 000002 | 001356 | | ADD | #2,DQTLVL | |
| 1350 | 002544 | 013737 | 001360 | 001362 | | MOV | DQCSR,DQCSH | |
| 1351 | 002552 | 005237 | 001362 | | | INC | DQCSH | ;GENERATE ADDRESS OF HIGH BYTE |
| 1352 | 002556 | 013737 | 001360 | 001364 | | MOV | DQCSR,DQTCR | ;GENERATE ADDRESS OF TRANSMITTER CONTROL REGISTER |
| 1353 | 002564 | 062737 | 000002 | 001364 | | ADD | #2,DQTCR | |
| 1354 | 002572 | 013737 | 001364 | 001366 | | MOV | DQTCR,DQERR | ;GENERATE ADDRESS OF ERROR REGISTER |
| 1355 | 002600 | 062737 | 000002 | 001366 | | ADD | #2,DQERR | |
| 1356 | 002606 | 013737 | 001366 | 001370 | | MOV | DQERR,DQREG | ;GENERATE ADDRESS OF HIGH BYTE OF ERROR REGISTER |
| 1357 | 002614 | 005237 | 001370 | | | INC | DQREG | |
| 1358 | 002620 | 013737 | 001370 | 001372 | | MOV | DQREG,DQSEC | ;GENERATE ADDRESS OF SECONDARY REGISTER |
| 1359 | 002626 | 005237 | 001372 | | | INC | DQSEC | |
| 1360 | 002632 | 013737 | 001372 | 001374 | | MOV | DQSEC,DQSECH | ;GENERATE ADDRESS OF HIGH BYTE |
| 1361 | 002640 | 005237 | 001374 | | | INC | DQSECH | |
| 1362 | | | | | | | | |
| 1363 | | | | | | | | |
| 1364 | | | | | | | | |
| 1365 | | | | | | | | |
| 1366 | | | | | | | | |
| 1367 | | | | | | | | |
| 1368 | | | | | | | | |
| 1369 | | | | | | | | |
| 1370 | | | | | | | | |
| 1371 | | | | | | | | |
| 1372 | | | | | | | | |
| 1373 | | | | | | | | |

;DQ11 HELL RAISER!!!
 ;THIS TEST WILL EXERCISE:
 ;DQ11 RECEIVER AND TRANSMITTER INTERUPTS
 ;ENTER T AND EXIT T (IF AB OPTION INSTALLED)
 ;VRC
 ;THE CABLE AND TURN AROUND (DATA ONLY)
 ;CHARACTER TRANSFERS.

```

1374 ; TEST 2
1375 ;*****
1376 002644 012737 000002 001226 1375: MOV #2,TSTNO ;ONE SYNC CHAR OR TWO?
1377 002652 012737 002732 001214 1376: MOV #2$,RETURN ;BR IF TWO
1378 002660 012737 000036 001222 1377: MOV #30,ICOUNT ;SET ONE SYNC
1379 002666 012737 003754 001216 1378: MOV #TST3,NEXT ;DBL SYNC SET TO ONE.
1380 ;ADJUST SYNC CHARACTERS. ;CONT.
1381
1382 002674 032737 100000 001510 1379: BIT #SYNBIT,DQSTAT ;ONE SYNC CHAR OR TWO?
1383 002702 001005 1380: BNE 1$ ;BR IF TWO
1384 002704 105037 012616 1381: CLRB SYNC ;SET ONE SYNC
1385 002710 005037 013422 1382: CLR XSYNC ;DBL SYNC SET TO ONE.
1386 002714 000406 1383: BR 2$ ;CONT.
1387 002716 112737 000026 012616 1384: MOV #2$,SYNC ;LOAD FOR TWO SYNC
1388 002724 012737 013026 013422 1385: MOV #13026,XSYNC ;SAME FOR DBL SYNC
1389 002732 104413 1386: MEMCLR ;CLEAR ALL REGISTERS GIVE MSTCLR
1390 002734 005037 014046 1387: CLR GDCHAR ;ZERO POINTER
1391 002740 005037 014040 1388: CLR CHAR
1392 002744 005037 177776 1389: CLR PS ;ZERO PROC. PRIO.
1393 002750 105077 176414 1390: CLRB #DQREG ;SEL THE RX BA PRI.
1394 002754 012777 013022 176410 1391: MOV #RXBUF,#DQSEC ;LOAD RX BA PRI.
1395 002762 105277 176402 1392: INCB #DQREG ;SEL RX MC PRI.
1396 002766 012777 177600 176376 1393: MOV #200,#DQSEC ;SET FOR 200 (8) CHARS
1397 002774 105277 176370 1394: INCB #DQREG ;SEL THE TX BA PRI.
1398 003000 012777 012616 176364 1395: MOV #SYNC,#DQSEC ;LOAD WITH SYNC POINTER
1399 003006 105277 176356 1396: INCB #DQREG ;SEL THE TX MC PRI.
1400 003012 012777 177576 176352 1397: MOV #202,#DQSEC ;SET FOR 2 SYNC AND 200 (8) CHARS.
1401 003020 105277 176344 1400: INCB #DQREG ;SEL THE RX BA SEC
1402 003024 012777 013630 176340 1401: MOV #RXBUF,#DQSEC ;LOAD RX BA SEC
1403 003032 105277 176332 1402: INCB #DQREG ;SEL RX MC SEC
1404 003036 012777 177600 176326 1403: MOV #200,#DQSEC ;SET FOR 200(8) CHARS
1405 003044 105277 176320 1404: INCB #DQREG ;SEL THE TX BA SEC
1406 003050 012777 013426 176314 1405: MOV #XTXBUF,#DQSEC ;LOAD IT
1407 003056 105277 176306 1406: INCB #DQREG ;SEL THE TX MC SEC
1408 003062 012777 177600 176302 1407: MOV #200,#DQSEC ;SET FOR 200 CHARS
1409 003070 112777 000011 176272 1408: MOV #11,#DQREG ;SEL THE SYNC REGISTER
1410 003076 013777 012614 176266 1409: MOV #SYNC,#DQSEC ;LOAD SYNC
1411 003104 105277 176260 1410: INCB #DQREG ;SEL THE MISC REGISTER
1412 003110 012777 104000 176254 1411: MOV #104000,#DQSEC ;SET 8 BITS PER CHAR AND VRC ENABLE.
1413 003116 032737 040000 001510 1412: BIT #JUMBIT,DQSTAT ;IS JUMPER AT END OF CABLE?
1414 003124 001003 1413: BNE .+10 ;BR IF YES
1415 003126 052777 000010 176236 1414: BIS #BIT3,#DQSEC ;NO CABLE SET TEST LOOP FOR DATA TURN AROUND
1416 003134 112777 000017 176226 1415: MOV #17,#DQREG ;SEL THE POLY REGISTER
1417 003142 012777 123456 176222 1416: MOV #123456,#DQSEC ;SET PLOYNOMIAL.
1418
1419 003150 012700 012620 1417: MOV #TXBFA,R0 ;START TO FILL TX BUFFERS
1420 003154 012703 000177 1418: MOV #177,R3 ;COUNTER
1421 003160 110320 1419: MOV R3,(R0)+ ;PRIMARY IS BINARY COUNT BACKWARDS.
1422 003162 105303 1420: DECB R3 ;DONE?
1423 003164 001375 1421: BNE 1$ ;NO
1424 003166 012700 013426 1422: MOV #XTXBUF,R0 ;SET SEC BUFFER
1425 003172 005003 1423: CLR R3 ;
1426 003174 110320 1424: MOV R3,(R0)+ ;SECONDARY IS BINARY COUNT
1427 003176 105203 1425: INCB R3 ;DONE?
1428 003200 100375 1426: BPL 2$ ;NO
1429 003202 012777 003500 176140 1427: MOV #RXISR,#DQREVC ;SET RECEIVER INTERRUPT POINTER
    
```

E03

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 30
 DZDQHC.P11 21-DEC-76 16:36 PROGRAM INITIALIZATION AND START UP.

```

1430 003210 012777 000240 176134      MOV      #240, @DQRLVL      ;SET PRIO: TO 5
1431 003216 012777 003332 176130      MOV      @TXISR, @DQTVEC   ;SET TX VECTOR
1432 003224 012777 000240 176124      MOV      #240, @DQTLVL     ;SET PRIO TO 5
1433 003232 012777 000041 176120      MOV      @BIT5+BIT0, @DQRCR ;SET RX GO AND IE
1434 003240 012777 000051 176116      MOV      @BIT5+BIT3+BIT0, @DQTCR ;SET TX GO AND IE AND ERROR IE
1435 003246 005037 001246          CLR      TEMP2            ;SET TIMER
1436 003252 012737 000113 001250      MOV      #75., TEMP3      ;SET NUMBER OF INTERUPTS WANTED
1437 003260 012737 000020 001252 4S:    MOV      #16., TEMP4      ;SET FOR 16 REGISTERS
1438 003266 142777 000017 176074      BICB    #17, @DQREG       ;SEL RX BA PRI.
1439 003274 105777 176070      3S:    TSTB    @DQREG           ;SIT HERE AND MAKE WAVES
1440 003300 005777 176066          TST     @DQSEC           ;WHILE INTERUPTS OCCUR
1441 003304 105277 176060          INCB    @DQREG           ;*****
1442 003310 005337 001252          DEC     TEMP4            ;*****
1443 003314 001367          BNE     3S              ;SAME
1444 003316 005237 001246          INC     TEMP2            ;UPDATE COUNTER
1445 003322 001356          BNE     4S              ;KEEP GOING
1446 003324 104005          HLT     5                ;RX FAILED TO CONTINUSLY INTERUPT
1447                                     ;****STRONGLY SUGGEST SW08=1 (GOTO TOP OF TEST OF ERROR)
1448 003326 000754          BR      4S              ;KEEP IT GOING.
1449 003330 104400      ENDS2: SCOPE            ;SCOPE THIS TEST.....
1450
1451
1452 003332 017737 176030 014032 TXISR: MOV      @DQERR, ERR      ;ANY ERRORS
1453 003340 100001          BPL     .+4              ;BR IF NO
1454 003342 104004          HLT     4                ;DQ11 ERROR FLAG IS SET.
1455                                     ;****STRONGLY SUGGEST SW08=1 (GOTO TOP OF TEST OF ERROR)
1456 003344 032777 000004 176012      BIT     @BIT2, @DQTCR     ;WHO SHOULD I SERVICE PRI OR SEC?
1457 003352 001425          BEQ     1S              ;BR IF SEC NEEDS SERVICE
1458 003354 112777 000002 176006      MOVB   #2, @DQREG        ;SEL TX BA PRI
1459 003362 042777 000200 175774      BIC    @BIT7, @DQTCR     ;CLEAR TX PRI DONE.
1460 003370 012777 012620 175774      MOV    @TXBFA, @DQSEC    ;LOAD THE TX BA PRI
1461 003376 105277 175766          INCB   @DQREG           ;SEL THE TX WC PRI.
1462 003402 152777 000120 175760      BISB  @BIT6+BIT4, @DQREG ;SET WRITE EN. AND ENTER T
1463 003410 012777 177600 175754      MOV    #8-200, @DQSEC    ;LOAD TX WC PRI.
1464 003416 142777 000017 175744      BICB  #17, @DQREG       ;CLEAR REG POINTER.
1465 003424 000002          RTI                    ;EXIT STAGE RIGHT
1466 003426 042777 000100 175730 1S:    BIC    @BIT6, @DQTCR     ;CLEAR TX SEC DONE
1467 003434 112777 000006 175726      MOVB  #6, @DQREG        ;SEL THE TX BA PRI.
1468 003442 012777 013426 175722      MOV    @TXBUF, @DQSEC    ;LOAD THE TX BA SEC
1469 003450 105277 175714          INCB  @DQREG           ;SEL THE TX WC SEC
1470 003454 152777 000060 175706      BISB  @BIT5+BIT4, @DQREG ;SET WRITE EN. AND EXIT T
1471 003462 012777 177600 175702      MOV    #8-200, @DQSEC    ;LOAD THE TX WC SEC
1472 003470 142777 000017 175672 2S:    BICB  #17, @DQREG       ;CLEAR REG POINTER
1473 003476 000002          RTI                    ;EXIT STAGE LEFT.
1474
1475                                     RXISR:
1476 003500          CLR     TEMP2            ;LET TIMER KNOW THAT RX INTERUPTED
1477 003504 005037 001246 175656 014032      MOV    @DQERR, ERR      ;ANY ERRORS
1478 003512 100001          BPL     .+4              ;BR IF NO
1479 003514 104004          HLT     4                ;DQ11 ERROR FLAG SET!!!!
1480                                     ;****STRONGLY SUGGEST SW08=1 (GOTO TOP OF TEST OF ERROR)
1481 003516 032777 000004 175634      BIT    @BIT2, @DQRCR     ;WHO SERVICE PRI OR SEC
1482 003524 001426          BEQ     2S              ;BR IF SEC NEEDS SERVICE
1483 003526 042777 000200 175624      BIC    @BIT7, @DQRCR     ;CLEAR RX PRI. DONE
1484 003534 105077 175630          CLRB  @DQREG           ;SEL RX BA PRI.
1485 003540 012777 013022 175624      MOV    @RXBUFF, @DQSEC  ;LOAD IT

```

F03

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 31
 DZDQHC.P11 21-DEC-76 16:36 PROGRAM INITIALIZATION AND START UP.

| | | | | | | | |
|------|--------|--------|--------|--------|-------|-------------------|---|
| 1486 | 003546 | 105277 | 175616 | | INCB | 3DQREG | SEL THE RX WC PRI. |
| 1487 | 003552 | 152777 | 000120 | 175610 | BISB | #BIT6+BIT4,3DQREG | SET WRITE EN. AND ENTER T |
| 1488 | 003560 | 012777 | 177600 | 175604 | MOV | #-200,3DQSEC | LOAD RX WC SEC |
| 1489 | 003566 | 012701 | 012620 | | MOV | #TXBFA,R1 | PREPARE TO CHECK DATA. SET TX POINTER |
| 1490 | 003572 | 012702 | 013022 | | MOV | #RXBUF,R2 | SET RX POINTER |
| 1491 | 003576 | 000137 | 003654 | | JMP | 3\$ | GO AND CHECK DATA |
| 1492 | 003602 | 042777 | 000100 | 175550 | BIC | #BIT6,3DQRCR | CLEAR RX SEC DONE |
| 1493 | 003610 | 112777 | 000004 | 175552 | MOVB | #4,3DQREG | SEL RX BA SEC |
| 1494 | 003616 | 012777 | 013630 | 175546 | MOV | #RXBUF,3DQSEC | LOAD IT |
| 1495 | 003624 | 105277 | 175540 | | INCB | 3DQREG | SEL THE RX WC SEC |
| 1496 | 003630 | 152777 | 000060 | 175532 | BISB | #BIT5+BIT4,3DQREG | SET WRITE EN. AND EXIT T |
| 1497 | 003636 | 012777 | 177600 | 175526 | MOV | #-200,3DQSEC | WRITE RX WC SEC |
| 1498 | 003644 | 012701 | 013426 | | MOV | #TXBUF,R1 | GET TX BUFFER POINTER |
| 1499 | 003650 | 012702 | 013630 | | MOV | #RXBUF,R2 | GET RX POINTER |
| 1500 | 003654 | 012700 | 000200 | | MOV | #200,R0 | GET NUMBER OF CHARS |
| 1501 | 003660 | 142711 | 000200 | | BICB | #BIT7,(R1) | CLEAR VRC |
| 1502 | 003664 | 142712 | 000200 | | BICB | #BIT7,(R2) | CLEAR VRC |
| 1503 | 003670 | 122122 | | | CMPB | (R1)+,(R2)+ | DATA OK? |
| 1504 | 003672 | 001414 | | | BEQ | 7\$ | BR IF YES |
| 1505 | 003674 | 112777 | 000012 | 175466 | MOVB | #12,3DQREG | SEL MISC REG |
| 1506 | 003702 | 052777 | 000002 | 175462 | BIS | #BIT1,3DQSEC | STOP THE DQ11 CLOCK. |
| 1507 | 003710 | 114137 | 014046 | | MOVB | -(R1),GDCHAR | STORE GOOD CHAR |
| 1508 | 003714 | 114237 | 014040 | | MOVB | -(R2),CHAR | STORE BAD CHAR. |
| 1509 | 003720 | 104003 | | | HLT | 3 | DATA COMPARE ERROR |
| 1510 | | | | | | | *****STRONGLY SUGGEST SW08=1 (GOTO TOP OF TEST OF ERROR |
| 1511 | 003722 | 122122 | | | CMPB | (R1)+,(R2)+ | POP POINTERS |
| 1512 | 003724 | 005300 | | | DEC | R0 | ALL DATA CHECKED? |
| 1513 | 003726 | 001354 | | | BNE | 4\$ | BR IF NO |
| 1514 | 003730 | 005337 | 001250 | | DEC | TEMP3 | ALL INTERRUPTS DONE? |
| 1515 | 003734 | 001003 | | | BNE | 6\$ | NO KEEP INTERRUPTING |
| 1516 | 003736 | 000005 | | | RESET | | STOP THE SHOW CLEAR THE WORLD |
| 1517 | 003740 | 012716 | 003330 | | MOV | #ENDTS2,(SP) | SET FOR END TEST RETURN |
| 1518 | 003744 | 142777 | 000017 | 175416 | BICB | #17,3DQREG | CLEAR REG POINTER |
| 1519 | 003752 | 000002 | | | RTI | | EXIT STAGE MIDDLE |
| 1520 | | | | | | | |

G03

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 32
 DZDQHC.P11 21-DEC-76 16:36 PROGRAM INITIALIZATION AND START UP.

```

1521
1522
1523          : TEST OF TRANSMITTER AND RECEIVER CHARACTER LENGTHS
1524          : THIS TEST WILL XMIT AND RECV CHARACTERS
1525          : AT 2 BITS/PER/CHAR.
1526          : DATA CHECKING WILL BE PERFORMED!
1527
1528          ; TEST 3
1529          ; *****
1530 003754 012737 000003 001226 TST3: MOV      #3,TSTNO
1531 003762 012737 004314 001216      MOV      #TST4,NEXT
1532 003770 104413          MEMCLR
1533 003772 012700 013022      MOV      #RXBUFF,R0          : CLEAR ALL THE DQ11
1534 003776 005001          CLR      R1                  : LOAD THE BUFFER POINTER
1535 004000 005020          CLR      (R0)+              : SET UP TO CLEAR THE BUFFER
1536 004002 105201          INCB   R1                  : CLEAR IT
1537 004004 100375          BPL     SS                  : DONE?
1538 004006 112777 000011 175354      MOVB   #11,DDQREG          : BRANCH IF NO
1539 004014 013737 012616 001246      MOV     SYNC,TEMP2        : SELECT THE SYNC REG
1540 004022 012737 177774 012134      MOV     #177774,MASK      : LOAD SYNC
1541 004030 143737 012134 001246      BICB   MASK,TEMP2        : LOAD THE MASK
1542 004036 000241          CLC
1543 004040 106037 001246          RORB   TEMP2              : SET UP A MASK TO GET THE
1544 004044 143737 012134 001247      BICB   MASK,TEMP2+1      : CORRECT SYNC CHARACTER
1545 004052 000241          CLC
1546 004054 106037 001247          RORB   TEMP2+1          : FOR THIS CHARACTER LENGTH
1547 004060 013737 001246 012136      MOV     TEMP2,SYNC1       : MANIPULATE DATA TO
1548 004066 013737 001246 012140      MOV     TEMP2,SYNC2       : COME UP WITH THE
1549 004074 013777 001246 175270      MOV     TEMP2,DDQSEC      : PROPER SYNC CHARACTER
1550 004102 105277 175262          INCB   DDQREG             : LOAD THE CHARACTER
1551 004106 012777 000010 175256      MOV     #BIT3,DDQSEC      : LOAD THE SYNC REGISTER
1552 004114 012700 000016          MOV     #16,R0           : SEL THE MISC REGISTER
1553 004120 000300          SWAB   R0                : SET TEST LOOP
1554 004122 050077 175244          BIS     R0,DDQSEC        : FLIP THE BYTES
1555 004126 052777 000002 175236      BIS     #BIT1,DDQSEC      : SET CHARACTER LENGTH
1556 004134 042777 000002 175230      BIC     #BIT1,DDQSEC      : TURN CLOCK OFF...
1557 004142 105077 175222          CLRB   DDQREG           : AND ON
1558 004146 012777 013022 175216      MOV     #RXBUFF,DDQSEC    : SEL RX PRIMARY ADDRESS
1559 004154 105277 175210          INCB   DDQREG           : SET ADDRESS
1560 004160 012777 177734 175204      MOV     #-36,DDQSEC       : SEL RX PRIMARY CHAR COUNT
1561 004166 105277 175176          INCB   DDQREG           : SET CHAR COUNT
1562 004172 012777 012140 175172      MOV     #SYNC2,DDQSEC     : SEL TX PRIMARY ADDRESS
1563 004200 105277 175164          INCB   DDQREG           : LOAD THE SYNC CHAR
1564 004204 012777 177732 175160      MOV     #-38,DDQSEC       : SEL TX PRI CHAR COUNT
1565 004212 005277 175142          INC     DDQRC5R          : SET CHAR COUNT
1566 004216 005277 175142          INC     DDQTC5R          : SET RX GO
1567 004222 005005          CLR     R5                : SET TX GO
1568 004224 105777 175130          TSTB   DDQRC5R          : START TIMING
1569 004230 100404          BMI     Z5                : IS DONE UP?
1570 004232 062705 000001          ADD     #1,R5            : BRANCH IF YES
1571 004236 001372          BNE     Z5                : WAIT
1572 004240 104001          HLT
1573 004242 012700 012142          MOV     #TXBUFF,R0       : BR IF MORE TO GO
1574 004246 012701 013022          MOV     #RXBUFF,R1       : ERROR--NO RX DONE
1575 004252 012702 000044          MOV     #36.,R2          : LOAD BUFFER POINTER
1576 004256          ; SET UP TO COUNT CHARACTERS
  
```

H03

DZDGM MACY11 27(1006) 22-DEC-76 11:40 PAGE 33
 DZDGM.C.P11 21-DEC-76 16:36 PROGRAM INITIALIZATION AND START UP.

```

1577 004256 112005          MOVB (R0)+,R5      ;GET A CHARACTER TO COPMARE
1578 004260 005037 001246  CLR    TEMP2      ;
1579 004264 112137 001246  MOVB (R1)+,TEMP2  ;GET REC CHARACTER
1580 004270 013704 001246  MOV    TEMP2,R4   ;MOVE TO R4
1581 004274 043705 012134  BIC   MASK,R5     ;MASK OUT UNWANTED BITS
1582 004300 020504          CMP    R5,R4      ;DO THE CHARACTERS MATCH?
1583 004302 001401          BEQ   4$          ;BR IF OK
1584 004304 104002          HLT   2           ;ERROR--DATA DOESN'T MATCH
1585 004306 005302 4$:    DEC   R2           ;ALL DONE?
1586 004310 001362          BNE   3$          ;NO--GO BACK FOR MORE
1587 004312 104400          SCOPE          ;SCOPE THIS TEST
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632

```

```

;TEST OF TRANSMITTER AND RECEIVER CHARATER LENGTHHS
;THIS TEST WILL XMIT AND RECV CHARACTERS
;AT 3 BITS/PER/CHAR.
;DATA CHECKING WILL BE PERFORMED!

```

```

; TEST 4
;*****
TST4:  MOV    #4,TSTNO
      MOV    #TSTS,NEXT
      MEMCLR
      MOV    #RXBUFF,R0
      CLR    R1
      CLR    (R0)+
      INCB  R1
      BPL   5$
      MOVB  #11,JDQREG
      MOV   SYNC,TEMP2
      MOV   #177770,MASK
      BICB  MASK,TEMP2
      CLC
      RORB  TEMP2
      BICB  MASK,TEMP2+1
      CLC
      RORB  TEMP2+1
      MOV   TEMP2,SYNC1
      MOV   TEMP2,SYNC2
      MOV   TEMP2,JDQSEC
      INCB  JDQREG
      MOV   #BIT3,JDQSEC
      MOV   #15,R0
      SWAB  R0
      BIS   R0,JDQSEC
      BIS   #BIT1,JDQSEC
      BIC   #BIT1,JDQSEC
      CLRB  JDQREG
      MOV   #RXBUFF,JDQSEC
      INCB  JDQREG
      MOV   #8-36,JDQSEC
      INCB  JDQREG
      MOV   #SYNC2,JDQSEC
      INCB  JDQREG
      MOV   #8-38,JDQSEC
      INC   JDQCSR
;CLEAR ALL THE DQ11
;LOAD THE BUFFER POINTER
;SET UP TO CLEAR THE BUFFER
;CLEAR IT
;DONE?
;BRANCH IF NO
;SELECT THE SYNC REG
;LOAD SYNC
;LOAD THE MASK
;SET UP A MASK TO GET THE
;CORRECT SYNC CHARACTER
;FOR THIS CHARACTER LENGTH
;MANIPULATE DATA TO
;COME UP WITH THE
;PROPER SYNC CHARACTER
;LOAD THE CHARACTER
;DITTO
;LOAD THE SYNC REGISTER
;SEL THE MISC REGISTER
;SET TEST LOOP
;FLIP THE BYTES
;SET CHARACTER LENGTH
;TURN CLOCK OFF...
;AND ON
;SEL RX PRIMARY ADRESS
;SET ADRESS
;SEL RX PRIMARY CHAR COUNT
;SET CHAR COUNT
;SEL TX PRIMARY ADDRESS
;LOAD THE SYNC CHAR
;SEL TX PRI CHAR COUNT
;SET CHAR COUNT
;SET RX GO

```

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 34
 DZDQHC.P11 21-DEC-76 16:36 PROGRAM INITIALIZATION AND START UP.

```

1633 004556 005277 174602          INC      @DQTCR          ;SET TX GO
1634 004562 005005                CLR      R5             ;START TIMING
1635 004564 105777 174570    15:    TSTB     @DQRCR          ;IS DONE UP?
1636 004570 100404                BMI     25             ;BRANCH IF YES
1637 004572 062705 000001          ADD     @1,R5          ;WAIT
1638 004576 001372                BNE     15             ;BR IF MORE TO GO
1639 004600 104001                HLT     1              ;ERROR--NO RX DONE
1640 004602 012700 012142    25:    MOV     @TXBUFF,R0     ;LOAD BUFFER POINTER
1641 004606 012701 013022          MOV     @RXBUFF,R1    ;LOAD RX BUFFER POINTER
1642 004612 012702 000044          MOV     @36.,R2       ;SET UP TO COUNT CHARACTERS
1643 004616                35:
1644 004616 112005                MOVB    (R0)+,R5      ;GET A CHARACTER TO COPMARE
1645 004620 005037 001246          CLR     TEMP2         ;
1646 004624 112137 001246          MOVB    (R1)+,TEMP2  ;GET REC CHARACTER
1647 004630 013704 001246          MOV     TEMP2,R4      ;MOVE TO R4
1648 004634 043705 012134          BIC     MASK,R5       ;MASK OUT UNWANTED BITS
1649 004640 020504                CMP     R5,R4         ;DO THE CHARACTERS MATCH?
1650 004642 001401                BEQ     45             ;BR IF OK
1651 004644 104002                HLT     2              ;ERROR--DATA DOESN'T MATCH
1652 004646 005302    45:    DEC     R2            ;ALL DONE?
1653 004650 001362                BNE     35            ;NO--GO BACK FOR MORE
1654 004652 104400                SCOPE                ;SCOPE THIS TEST
    
```

```

;TEST OF TRANSMITTER AND RECEIVER CHARATER LENGTHS
;THIS TEST WILL XMIT AND RECV CHARACTERS
;AT 4 BITS/PER/CHAR.
;DATA CHECKING WILL BE PERFORMED!
    
```

```

; TEST 5
;*****
1664 004654 012737 000005 001226  TSTS:  MOV     @5,TSTNO
1665 004662 012737 005214 001216      MOV     @TST6,NEXT
1666 004670 104413                MEMCLR                ;CLEAR ALL THE DQ11
1667 004672 012700 013022          MOV     @RXBUFF,R0   ;LOAD THE BUFFER POINTER
1668 004676 005001                CLR     R1            ;SET UP TO CLEAR THE BUFFER
1669 004700 005020    55:    CLR     (R0)+        ;CLEAR IT
1670 004702 105201                INCB   R1            ;DONE?
1671 004704 100375                BPL     55           ;BRANCH IF NO
1672 004706 112777 000011 174454  MOVB    @11,@DQREG   ;SELECT THE SYNC REG
1673 004714 013737 012616 001246  MOV     SYNC,TEMP2   ;LOAD SYNC
1674 004722 012737 177760 012134  MOV     @177760,MASK ;LOAD THE MASK
1675 004730 143737 012134 001246  BICB    MASK,TEMP2   ;SET UP A MASK TO GET THE
1676 004736 000241                CLC                    ;CORRECT SYNC CHARACTER
1677 004740 106037 001246                RORB   TEMP2         ;FOR THIS CHARACTER LENGTH
1678 004744 143737 012134 001247  BICB    MASK,TEMP2+1 ;MANIPULATE DATA TO
1679 004752 000241                CLC                    ;COME UP WITH THE
1680 004754 106037 001247                RORB   TEMP2+1       ;PROPER SYNC CHARACTER
1681 004760 013737 001246 012136  MOV     TEMP2,SYNC1  ;LOAD THE CHARACTER
1682 004766 013737 001246 012140  MOV     TEMP2,SYNC2  ;DITTO
1683 004774 013777 001246 174370  MOV     TEMP2,@DQSEC ;LOAD THE SYNC REGISTER
1684 005002 105277 174362                INCB   @DQREG        ;SEL THE MISC REGISTER
1685 005006 012777 000010 174356  MOV     @BIT3,@DQSEC ;SET TEST LOOP
1686 005014 012700 000014                MOV     @14,R0
1687 005020 000300                SWAB   R0            ;FLIP THE BYTES
1688 005022 050077 174344                BIS    R0,@DQSEC    ;SET CHARACTER LENGTH
    
```

J03

DZDGM MACY11 27(1006) 22-DEC-76 11:40 PAGE 35
 DZDGM.C.P11 21-DEC-76 16:36 PROGRAM INITIALIZATION AND START UP.

| | | | | | | | |
|------|--------|--------|--------|--------|----------|-----------------|------------------------------|
| 1689 | 005026 | 052777 | 000002 | 174336 | BIS | #BIT1, 200SEC | : TURN CLOCK OFF... |
| 1690 | 005034 | 042777 | 000002 | 174330 | BIC | #BIT1, 200SEC | : AND ON |
| 1691 | 005042 | 105077 | 174322 | | CLRB | 200REG | : SEL RX PRIMARY ADDRESS |
| 1692 | 005046 | 012777 | 013022 | 174316 | MOV | #RXBUFF, 200SEC | : SET ADDRESS |
| 1693 | 005054 | 105277 | 174310 | | INCB | 200REG | : SEL RX PRIMARY CHAR COUNT |
| 1694 | 005060 | 012777 | 177734 | 174304 | MOV | #-36, 200SEC | : SET CHAR COUNT |
| 1695 | 005066 | 105277 | 174276 | | INCB | 200REG | : SEL TX PRIMARY ADDRESS |
| 1696 | 005072 | 012777 | 012140 | 174272 | MOV | #SYNC2, 200SEC | : LOAD THE SYNC CHAR |
| 1697 | 005100 | 105277 | 174264 | | INCB | 200REG | : SEL TX PRI CHAR COUNT |
| 1698 | 005104 | 012777 | 177732 | 174260 | MOV | #-38, 200SEC | : SET CHAR COUNT |
| 1699 | 005112 | 005277 | 174242 | | INC | 200RCSR | : SET RX GO |
| 1700 | 005116 | 005277 | 174242 | | INC | 200TCSR | : SET TX GO |
| 1701 | 005122 | 005005 | | | CLR | R5 | : START TIMING |
| 1702 | 005124 | 105777 | 174230 | | 15: TSTB | 200RCSR | : IS DONE UP? |
| 1703 | 005130 | 100404 | | | BMI | 25 | : BRANCH IF YES |
| 1704 | 005132 | 062705 | 000001 | | ADD | #1, R5 | : WAIT |
| 1705 | 005136 | 001372 | | | BNE | 15 | : BR IF MORE TO GO |
| 1706 | 005140 | 104001 | | | HLT | 1 | : ERROR--NO RX DONE |
| 1707 | 005142 | 012700 | 012142 | | 25: MOV | #TXBUFF, R0 | : LOAD BUFFER POINTER |
| 1708 | 005146 | 012701 | 013022 | | MOV | #RXBUFF, R1 | : LOAD RX BUFFER POINTER |
| 1709 | 005152 | 012702 | 000044 | | MOV | #36, R2 | : SET UP TO COUNT CHARACTERS |
| 1710 | 005156 | | | | 35: | | |
| 1711 | 005156 | 112005 | | | MOVB | (R0)+, R5 | : GET A CHARACTER TO COMPARE |
| 1712 | 005160 | 005037 | 001246 | | CLR | TEMP2 | : : |
| 1713 | 005164 | 112137 | 001246 | | MOVB | (R1)+, TEMP2 | : GET REC CHARACTER |
| 1714 | 005170 | 013704 | 001246 | | MOV | TEMP2, R4 | : MOVE TO R4 |
| 1715 | 005174 | 043705 | 012134 | | BIC | MASK, R5 | : MASK OUT UNWANTED BITS |
| 1716 | 005200 | 020504 | | | CMP | R5, R4 | : DO THE CHARACTERS MATCH? |
| 1717 | 005202 | 001401 | | | BEQ | 45 | : BR IF OK |
| 1718 | 005204 | 104002 | | | HLT | 2 | : ERROR--DATA DOESN'T MATCH |
| 1719 | 005206 | 005302 | | | 45: DEC | R2 | : ALL DONE? |
| 1720 | 005210 | 001362 | | | BNE | 35 | : NO--GO BACK FOR MORE |
| 1721 | 005212 | 104400 | | | SCOPE | | : SCOPE THIS TEST |
| 1722 | | | | | | | |
| 1723 | | | | | | | |
| 1724 | | | | | | | |
| 1725 | | | | | | | |
| 1726 | | | | | | | |
| 1727 | | | | | | | |
| 1728 | | | | | | | |
| 1729 | | | | | | | |
| 1730 | | | | | | | |
| 1731 | 005214 | 012737 | 000006 | 001226 | | | |
| 1732 | 005222 | 012737 | 005554 | 001216 | | | |
| 1733 | 005230 | 104413 | | | | | |
| 1734 | 005232 | 012700 | 013022 | | | | |
| 1735 | 005236 | 005001 | | | | | |
| 1736 | 005240 | 005020 | | | | | |
| 1737 | 005242 | 105201 | | | | | |
| 1738 | 005244 | 100375 | | | | | |
| 1739 | 005246 | 112777 | 000011 | 174114 | | | |
| 1740 | 005254 | 013737 | 012616 | 001246 | | | |
| 1741 | 005262 | 012737 | 177740 | 012134 | | | |
| 1742 | 005270 | 143737 | 012134 | 001246 | | | |
| 1743 | 005276 | 000241 | | | | | |
| 1744 | 005300 | 106037 | 001246 | | | | |

: TEST OF TRANSMITTER AND RECEIVER CHARACTER LENGTHS
 : THIS TEST WILL XMIT AND RECV CHARACTERS
 : AT 5 BITS/PER/CHAR.
 : DATA CHECKING WILL BE PERFORMED!

: TEST 6
 : *****

| | | | | | | | |
|------|--------|--------|--------|--------|-----------|---------------|------------------------------|
| 1731 | 005214 | 012737 | 000006 | 001226 | TST6: MOV | #6, TSTNO | : CLEAR ALL THE D011 |
| 1732 | 005222 | 012737 | 005554 | 001216 | MOV | #TST7, NEXT | : LOAD THE BUFFER POINTER |
| 1733 | 005230 | 104413 | | | MEMCLR | | : SET UP TO CLEAR THE BUFFER |
| 1734 | 005232 | 012700 | 013022 | | MOV | #RXBUFF, R0 | : CLEAR IT |
| 1735 | 005236 | 005001 | | | CLR | R1 | : DONE? |
| 1736 | 005240 | 005020 | | | 55: CLR | (R0)+ | : BRANCH IF NO |
| 1737 | 005242 | 105201 | | | INCB | R1 | : SELECT THE SYNC REG |
| 1738 | 005244 | 100375 | | | BPL | 55 | : LOAD SYNC |
| 1739 | 005246 | 112777 | 000011 | 174114 | MOVB | #11, 200REG | : LOAD THE MASK |
| 1740 | 005254 | 013737 | 012616 | 001246 | MOV | SYNC, TEMP2 | : SET UP A MASK TO GET THE |
| 1741 | 005262 | 012737 | 177740 | 012134 | MOV | #177740, MASK | : CORRECT SYNC CHARACTER |
| 1742 | 005270 | 143737 | 012134 | 001246 | BICB | MASK, TEMP2 | : FOR THIS CHARACTER LENGTH |
| 1743 | 005276 | 000241 | | | CLC | | |
| 1744 | 005300 | 106037 | 001246 | | RORB | TEMP2 | |

K03

DZDQH MACY11 27(1006) 22-DEC-76 11:40 PAGE 36
 DZDQHC.P11 21-DEC-76 16:36 PROGRAM INITIALIZATION AND START UP.

| | | | | | | | |
|------|--------|--------|--------|--------|------|----------------|------------------------------|
| 1745 | 005304 | 143737 | 012134 | 001247 | BICB | MASK,TEMP2+1 | : MANIPULATE DATA TO |
| 1746 | 005312 | 000241 | | | CLC | | : COME UP WITH THE |
| 1747 | 005314 | 106037 | 001247 | | RORB | TEMP2+1 | : PROPER SYNC CHARACTER |
| 1748 | 005320 | 013737 | 001246 | 012136 | MOV | TEMP2,SYNC1 | : LOAD THE CHARACTER |
| 1749 | 005326 | 013737 | 001246 | 012140 | MOV | TEMP2,SYNC2 | : DITTO |
| 1750 | 005334 | 013777 | 001246 | 174030 | MOV | TEMP2,200SEC | : LOAD THE SYNC REGISTER |
| 1751 | 005342 | 105277 | 174022 | | INCB | 200REG | : SEL THE MISC REGISTER |
| 1752 | 005346 | 012777 | 000010 | 174016 | MOV | #BIT3,200SEC | : SET TEST LOOP |
| 1753 | 005354 | 012700 | 000013 | | MOV | #13,R0 | |
| 1754 | 005360 | 000300 | | | SWAB | R0 | : FLIP THE BYTES |
| 1755 | 005362 | 050077 | 174004 | | BIS | R0,200SEC | : SET CHARACTER LENGTH |
| 1756 | 005366 | 052777 | 000002 | 173776 | BIS | #BIT1,200SEC | : TURN CLOCK OFF... |
| 1757 | 005374 | 042777 | 000002 | 173770 | BIC | #BIT1,200SEC | : AND ON |
| 1758 | 005402 | 105077 | 173762 | | CLRB | 200REG | : SEL RX PRIMARY ADDRESS |
| 1759 | 005406 | 012777 | 013022 | 173756 | MOV | #RXBUFF,200SEC | : SET ADDRESS |
| 1760 | 005414 | 105277 | 173750 | | INCB | 200REG | : SEL RX PRIMARY CHAR COUNT |
| 1761 | 005420 | 012777 | 177734 | 173744 | MOV | #-36,200SEC | : SET CHAR COUNT |
| 1762 | 005426 | 105277 | 173736 | | INCB | 200REG | : SEL TX PRIMARY ADDRESS |
| 1763 | 005432 | 012777 | 012140 | 173732 | MOV | #SYNC2,200SEC | : LOAD THE SYNC CHAR |
| 1764 | 005440 | 105277 | 173724 | | INCB | 200REG | : SEL TX PRI CHAR COUNT |
| 1765 | 005444 | 012777 | 177732 | 173720 | MOV | #-38,200SEC | : SET CHAR COUNT |
| 1766 | 005452 | 005277 | 173702 | | INC | 200RCSR | : SET RX GO |
| 1767 | 005456 | 005277 | 173702 | | INC | 200TCSR | : SET TX GO |
| 1768 | 005462 | 005005 | | | CLR | R5 | : START TIMING |
| 1769 | 005464 | 105777 | 173670 | 15: | TSTB | 200RCSR | : IS DONE UP? |
| 1770 | 005470 | 100404 | | | BMI | 25 | : BRANCH IF YES |
| 1771 | 005472 | 062705 | 000001 | | ADD | #1,R5 | : WAIT |
| 1772 | 005476 | 001372 | | | BNE | 15 | : BR IF MORE TO GO |
| 1773 | 005500 | 104001 | | | HLT | 1 | : ERROR--NO RX DONE |
| 1774 | 005502 | 012700 | 012142 | 25: | MOV | #TXBUFF,R0 | : LOAD BUFFER POINTER |
| 1775 | 005506 | 012701 | 013022 | | MOV | #RXBUFF,R1 | : LOAD RX BUFFER POINTER |
| 1776 | 005512 | 012702 | 000044 | | MOV | #36.,R2 | : SET UP TO COUNT CHARACTERS |
| 1777 | 005516 | | | 35: | | | |

1778 P0551811112005 MD-11-DFKTH-A PDP 11/34(MEM MGR)DIAG
 DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006)KTHADECAN-77 08:51 80000000 770325

.REM 2

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DFKTH-A-D
 PRODUCT NAME: PDP 11/34 MEMORY MANAGEMENT DIAGNOSTIC
 DATE: JANUARY 31,1977
 MAINTAINER: DIAGNOSTIC PROGRAMMING
 AUTHOR: DIAGNOSTIC ENGINEERING

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22

M03

MD-11-DFKTH-A PDP 11/34 MEM MGMT DIAG MACY11 27(1006) 27-JAN-77 08:51 PAGE 2
DFKTHA.P11 19-JAN-77 16:02

MD-11-DFKTH-A PDP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

PROGRAM HISTORY

| <u>DATE</u> | <u>REVISION</u> | <u>REASON FOR REVISION</u> |
|-------------|-----------------|----------------------------|
| 31-JAN-77 | A | FIRST RELEASE |

68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103

TABLE OF CONTENTS

1.0 PROGRAM INFORMATION
 1.1 ABSTRACT
 1.2 REQUIREMENTS
 1.3 RELATED DOCUMENTS AND STANDARDS
 1.4 PRELIMINARY PROGRAMS

2.0 OPERATING INSTRUCTIONS
 2.1 LOADING PROCEDURES
 2.2 STARTING PROCEDURES
 2.3 OPERATIONAL SWITCH SETTINGS
 2.4 LOADING THE SWITCH REGISTER
 2.5 EXECUTION TIMES

3.0 ERROR INFORMATION
 3.1 ERROR REPORTING PROCEDURES
 3.2 INTERPRETING ERROR REPORTS
 3.3 SAMPLE ERROR REPORT

4.0 MISCELLANEOUS INFORMATION
 4.1 ACT/APY/XDP COMPATABILITY
 4.2 END-OF-PASS MESSAGE
 4.3 T-BIT TRAPPING
 4.4 POWER FAILURE HANDLING
 4.5 PHYSICAL BUS ADDRESS CONSTRUCTION

5.0 PROGRAM DESCRIPTION
 5.1 SUBROUTINES USED BY THIS PROGRAM
 5.2 PROGRAM LISTING
 5.3 USING THE PROGRAM TO DIAGNOSE A FAULT

104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159

1.0 PROGRAM INFORMATION

1.1 ABSTRACT

THIS PROGRAM WAS DESIGNED USING A "BOTTOM UP" APPROACH STARTING WITH THE SMALLEST SEGMENT OF MEMORY MANAGEMENT LOGIC POSSIBLE AND BUILDING TO COVER ALL OF THE LOGIC. THE DIAGNOSTIC WILL PROVIDE ENOUGH INFORMATION SUCH THAT BY DEDUCTION, THE FAILURE CAN BE ISOLATED TO A SMALL SEGMENT OF THE MEMORY MANAGEMENT LOGIC.

THE PROGRAM BEGINS BY TESTING SOME OF THE INTERNAL CPU DATA AND ADDRESS PATHS AND ADDRESS DETECTION LOGIC, THEN WORKS OUTWARD THROUGH THE MEMORY MANAGEMENT REGISTERS. AFTER THE REGISTERS ARE FOUND TO BE USEABLE, RELOCATION (CONSTRUCTION OF PHYSICAL ADDRESSES FROM A VIRTUAL ADDRESS AND THE ASSOCIATED PAR/PDR INFORMATION) IS TESTED FOLLOWED BY TESTING OF THE ABORT AND STATUS SEGMENTS OF LOGIC. FINALLY, CHECKS OF SPECIAL ABORT SEQUENCES AND TESTING OF THE MFPI/MTPI INSTRUCTIONS ARE DONE.

1.2 REQUIREMENTS

A PDP 11/34 PROCESSOR WITH A MINIMUM OF 16K OF MEMORY AND A CONSOLE TERMINAL ARE REQUIRED TO RUN THE PROGRAM UNLESS THE PROGRAM IS RUNNING UNDER APT OR ACT IN WHICH CASE THE CONSOLE TERMINAL IS NOT NECESSARY.

1.3 RELATED DOCUMENTS AND STANDARDS

1. ACT11/XXDP PROGRAMMING SPECIFICATION
2. STANDARD APT SYSTEM TO A PDP11 DIAGNOSTIC INTERFACE
3. DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS
4. PDP11 MAINDEC SYSHAC PACKAGE
5. XXDP USER'S MANUAL

1.4 PRELIMINARY PROGRAMS

BEFORE THIS MEMORY MANAGEMENT DIAGNOSTIC IS RUN, THE FOLLOWING CPU DIAGNOSTICS SHOULD BE RUN:

- | | |
|-------------|---------------------------|
| MD-11-DFKAA | PDP 11/34 BASIC CPU TESTS |
| MD-11-DFKAB | PDP 11/34 TRAPS TESTS |

ALSO, ONE OF THE MAIN MEMORY DIAGNOSTICS SHOULD BE RUN TO SCAN AT LEAST THE FIRST 16K TO SEE THAT A PROGRAM CAN BE EXECUTED.

2.0 OPERATING INSTRUCTIONS

160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215

2.1 LOADING PROCEDURES

THE PROGRAM IS SUPPLIED ON THE DIAGNOSTIC LOAD MEDIA. REFER TO THE XXDP USER'S MANUAL FOR FURTHER INFORMATION. FOR USE WITH ACT OR APT, REFER TO THEIR RESPECTIVE DOCUMENTS. THE PROGRAM CAN ALSO BE DIRECTLY LOADED USING THE ABSOLUTE LOADER AND THE BINARY PAPER TAPE.

2.2 STARTING PROCEDURES

THE PROGRAM IS STARTED BY LOADING ADDRESS 200 AND STARTING. IF THE PROCESSOR HAS THE OPTIONAL PROGRAMMER'S CONSOLE, THE SWITCH REGISTER SHOULD BE SET ACCORDING TO SECTION 2.3 BEFORE THE PROGRAM IS STARTED. IF THERE IS NO HARDWARE SWITCH REGISTER, THE PROGRAM WILL USE THE SOFTWARE SWITCH REGISTER AT LOCATION 176 (LOCATION 174 WILL BE USED AS THE SOFTWARE DISPLAY REGISTER). IN THAT CASE THE PROGRAM WILL ASK FOR THE INITIAL SWITCH REGISTER VALUE BY TYPING "SWR= XXXXXX NEW= " AFTER TYPING THE NAME OF THE PROGRAM (XXXXXX = THE OCTAL CONTENTS OF LOCATION 176). (SEE SECTION 2.4)

ALSO THE PROGRAM CAN BE MADE TO USE THE SOFTWARE SWITCH REG. EVEN IF THE HARDWARE SWITCH REG. IS PRESENT BY LOADING "177777" INTO THE HARDWARE SWITCH REG. BEFORE STARTING THE PROGRAM.

2.3 CONTROL SWITCH SETTINGS

| SWITCH | OCTAL VALUE | USE |
|--------|-------------|---|
| SW15 | 100000 | HALT ON ERROR THIS SWITCH WHEN SET WILL HALT THE PROCESSOR WHEN AN ERROR IS DETECTED AFTER THE ERROR MESSAGE HAS BEEN TYPED. PRESSING CONTINUE WILL RESUME TESTING (SEE SECTION 3.1 ABOUT LOADING THE SWITCH REG BEFORE CONTINUING). |
| SW14 | 040000 | LOOP ON TEST THIS SWITCH WHEN SET WILL CAUSE THE PROGRAM TO LOOP ON THE CURRENT SUBTEST. |
| SW13 | 020000 | INHIBIT ERROR TYPEOUTS THIS SWITCH WHEN SET WILL INHIBIT THE TYPING OF ERROR MESSAGES. |
| SW12 | 010000 | INHIBIT TRACE TRAP THIS SWITCH WHEN SET WILL INHIBIT T-BIT TRAPPING WHICH |

| | | | | |
|-----|------|--------|---------|-------------------------------------|
| 216 | | | | NORMALLY TAKES PLACE DURING |
| 217 | | | | EVERY OTHER PASS STARTING |
| 218 | | | | WITH THE THIRD PASS. |
| 219 | | | | |
| 220 | SW11 | 004000 | INHIBIT | SUBTEST ITERATIONS |
| 221 | | | | THIS SWITCH WHEN SET INHIBITS |
| 222 | | | | ITERATIONS OF EACH SUBTEST AFTER |
| 223 | | | | THE FIRST PASS. IF THIS SWITCH |
| 224 | | | | IS NOT SET, EACH SUBTEST IS RUN |
| 225 | | | | 200. TIMES. |
| 226 | | | | |
| 227 | SW10 | 002000 | BELL ON | ERROR |
| 228 | | | | THIS SWITCH WHEN SET WILL RING |
| 229 | | | | THE CONSOLE TERMINAL BELL WHEN |
| 230 | | | | AN ERROR HAS BEEN DETECTED. |
| 231 | | | | |
| 232 | SW9 | 001000 | LOOP ON | ERROR |
| 233 | | | | THIS SWITCH WHEN SET WILL |
| 234 | | | | CAUSE THE PROGRAM TO LOOP ON THE |
| 235 | | | | FIRST FAILURE WHICH IS ENCOUNTERED |
| 236 | | | | EVEN IF THE FAILURE IS INTERMITTANT |
| 237 | | | | |
| 238 | SW8 | 000400 | LOOP ON | TEST IN SWR<7:0> |
| 239 | | | | THIS SWITCH WHEN SET WILL |
| 240 | | | | CAUSE THE PROGRAM TO LOOP ON THE |
| 241 | | | | TEST WHOSE TEST NUMBER IS SET |
| 242 | | | | IN BITS 7-0 OF THE SWITCH REG. |

2.4 LOADING THE SWITCH REGISTER

THE HARDWARE SWITCH REGISTER PROVIDED WHEN THE OPTIONAL PROGRAMMER'S CONSOLE IS PRESENT IS LOADED DIRECTLY FROM THE CONSOLE KEYPAD BY DEPRESSING THE "LSR" KEY. THE VALUE OF THE HARDWARE SWITCH REG. CAN BE CHANGED ANY TIME WHETHER THE PROGRAM IS RUNNING OR NOT.

TO LOAD THE SOFTWARE SWITCH REG. WHILE THE PROGRAM IS RUNNING, A CONTROL G (↑G) SHOULD BE TYPED ON THE CONSOLE TERMINAL. (THE "SCOPE" AND "ERROR" ROUTINES CHECK TO SEE IF A ↑G HAS BEEN TYPED.) THE ORIGINAL VALUE OF THE SOFTWARE SWITCH REG. WILL BE REQUESTED AS MENTIONED IN SECTION 2.2.

IN RESPONSE TO A ↑G OR AT THE BEGINNING OF THE PROGRAM, THE PROGRAM WILL TYPE:

SWR = XXXXXX NEW =

WHERE "XXXXXX" IS THE CURRENT OCTAL CONTENTS OF LOC. 176. THE OPERATOR MAY THEN TYPE ANY ONE OF THE FOLLOWING:

- XXXXXX<CR> ONE TO SIX OCTAL DIGITS FOLLOWED BY A CARRIAGE RETURN WHICH WILL BE LOADED AS THE NEW VALUE FOR THE SWITCH REG.
- <CR> JUST A <CR>, LEAVES THE SWITCH REG. AS IT IS.
- XXX↑U A CONTROL-U (↑U) WILL CAUSE ALL OF THE

216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271

272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327

↑C DIGITS TYPED SO FAR TO BE IGNORED.
WILL CAUSE THE PROGRAM TO TYPE THE PRESENT
TEST AND PASS NUMBERS, REQUEST A NEW VALUE
FOR THE SWITCH REG. AND JUMP TO THE END-
OF-PASS ROUTINE SO THE PROGRAM WILL GO DIRECTLY
TO THE NEXT PASS WITH A NEW SW. REG. VALUE
<ILL.CHAR> ANY CHARACTER TYPED WHICH IS NOT ANY OF THE
ABOVE OR AN OCTAL DIGIT WILL CAUSE THE PROGRAM
TO TYPE A "?<CRLF>" AND REACT AS THOUGH A
↑U HAD BEEN TYPED.

NOTE: RECOGNITION OF A ↑G MAY BE HAMPERED BY
----- EXECUTION OF A COUPLE "RESET" INSTRUCTIONS
WITHIN THE PROGRAM.

2.5 EXECUTION TIMES

THE RUN TIME FOR A SINGLE PASS WITH NO ITERATIONS
OR TRACE TRAPPING IS APPROXIMATELY 5 SECONDS.

THE RUN TIME FOR A SINGLE PASS WITH ITERATIONS
AND TRACE TRAPPING ENABLED IS APPROXIMATELY 3 1/4 MINUTES.

3.0 ERROR INFORMATION

3.1 ERROR REPORTING PROCEDURES

IF AN ERROR IS DETECTED, THE PROGRAM WILL TRAP TO THE
ERROR HANDLING ROUTINE (SEERR). THE VALUE OF BITS
15, 13, 10, AND 9 IN THE SWITCH REGISTER ARE CONSIDERED
IN REPORTING AN ERROR (SEE SECTION 2.3). THE
ERROR INFORMATION WILL BE TYPED UNLESS SW13 = 1.

IF SW15 = 1, THE PROCESSOR WILL HALT AFTER THE ERROR IS
REPORTED. IF THE CONTENTS OF THE SOFTWARE SWITCH REGISTER
ARE TO BE CHANGED, A ↑G SHOULD BE TYPED BEFORE PRESSING
"CONTINUE" TO RESUME TESTING.

IF SW9 = 1 (LOOP ON ERROR), THE PROGRAM WILL GO TO THE
ADDRESS CONTAINED IN LOCATION "SLPERR". AFTER REPORTING
THE ERROR, "SLPERR" IS SET BY EACH "SCOPE" CALL AND IS
SET DIRECTLY DURING SOME SUBTESTS TO PROVIDE THE SMALLEST
LOOP FOR LOOPING ON ERROR. IF SW9 = 0, THE PROGRAM WILL
RETURN TO THE INSTRUCTION FOLLOWING THE ERROR CALL.
(SEE SECTION 5.3 FOR MORE ON "LOOP ON ERROR").

3.2 INTERPRETING ERROR REPORTS

EVERY ERROR REPORT TYPES THE NUMBER OF THE TEST IN WHICH
THE ERROR TOOK PLACE (TESTNO) AND THE LOCATION OF THE
ERROR CALL (ERRORPC). THESE TWO VALUES PINPOINT THE
PLACE IN THE CODE THAT THE ERROR OCCURRED. BY REFERRING

328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383

TO THE PROGRAM LISTING, THE OPERATOR CAN THEN READ THE COMMENTS ASSOCIATED WITH THAT PARTICULAR ERROR AND SUBTEST. A DESCRIPTION OF THE TEST FOUND IN THE PROGRAM LISTING WILL ALSO PROVIDE THE OPERATOR WITH INFORMATION ON THE LOGIC AND FUNCTIONS BEING TESTED.

EVERY ERROR REPORT ALSO TYPES AN ERROR MESSAGE GIVING A VERBAL DESCRIPTION OF THE ERROR THAT HAS BEEN DETECTED.

BY USING THE COMMENTS AND TEST DESCRIPTION FOUND IN THE PROGRAM LISTING TO DETERMINE WHAT FUNCTION OR LOGIC WAS BEING TESTED, THE OPERATOR CAN THEN REFER TO THE ENGINEERING DRAWINGS TO ISOLATE THE PROBABLE CAUSE FOR THE FAILURE.

3.3 SAMPLE ERROR REPORT

BELOW IS AN EXAMPLE OF AN ERROR WHICH COULD HAVE OCCURRED DURING EXECUTION OF THE PROGRAM:

| MEM. MGMT. REG. | BITS NOT SET CORRECTLY | | | | |
|-----------------|------------------------|------------------|------------------|---------|--------|
| REGISTR WROTE | READ | READ-(BINARY) | | | |
| ADDRESS (OCTAL) | (OCTAL) | 5432109876543210 | TESTNO | ERRORPC | |
| 177572 | 040000 | 060000 | 0110000000000000 | 000012 | 022060 |

WE SEE THAT THE ERROR OCCURRED IN TEST 12 AT LOACTION 022060. THE "REGISTR ADDRESS" TELLS US THAT WE WERE TESTING MEMORY MANAGEMENT'S STATUS REGISTER 0 (SRO). IN THE LISTING, THE TEST DESCRIPTION SAYS THAT THE ERROR BITS (BITS (15:13)) OF SRO WERE BEING SET AND CLEARED INDIVIDUALLY. THE ERROR REPORT SAYS WE TRIED TO SET BIT 14 BY WRITING "040000" TO SRO BUT WHEN WE READ IT BACK WE READ "060000". IT APPEARS THAT BIT 13 IS STUCK AT "1" OR IT IS GETTING SET WHEN BIT 14 IS SET TO "1". ERROR REPORTS BEFORE AND AFTER THIS ONE COULD TELL US WHICH IS THE CASE.

4.0 MISCELLANEOUS INFORMATION

4.1 ACT/APT/XXDP COMPATABILITY

THE PROGRAM IS FULLY ACT AND APT COMPATABLE AND IS SUPPORTED UNDER THE XXDP PACKAGE.

4.2 END-OF-PASS MESSAGE

AT THE END OF EACH PASS OF THE PROGRAM THE PASS NUMBER AND TOTAL NUMBER OF ERRORS SINCE THE LAST END-OF-PASS ARE REPORTED IN THE END-OF-PASS MESSAGE. FOR EXAMPLE:

END OF PASS #2 TOTAL ERRORS SINCE LAST REPORT 0

KERNEL (=0) SET OF PAR'S/PDR'S

5.0 PROGRAM DESCRIPTION
-----5.1 SUBROUTINES USED BY THIS PROGRAM

FOLLOWING IS A LIST OF THE SUBROUTINES AND HANDLERS USED BY THIS PROGRAM THAT ARE NOT PROVIDED BY THE "SYSMAC PACKAGE". DETAILS OF THE SUBROUTINES UNIQUE TO THIS PROGRAM MAY BE FOUND IN THE PROGRAM LISTING. REFER TO THE "SYSMAC" DOCUMENT AND PROGRAM LISTING FOR THE OTHER ROUTINES.

1. TURN OFF T-BIT AND SAVE CURRENT PSW
2. TURN ON T-BIT AND RESTORE PREVIOUS PSW
3. SET ALL WRITEABLE BITS IN ALL PAR/PDR'S
4. READ AND COMPARE KERNEL AND USER PAR/PDR'S
5. CONVERT VIRTUAL ADDRESS TO PHYSICAL ADDRESS

5.2 PROGRAM LISTING

A TABLE OF CONTENTS APPEARS AT THE BEGINNING OF THE LISTING WHICH CONTAINS THE NAMES OF EACH SECTION, SUBTEST, AND ROUTINE AND THE LINE NUMBERS CORRESPONDING TO THE START OF EACH.

FOLLOWING THIS SECTION OF DOCUMENTATION IS THE ACTUAL PROGRAM LISTING COMPLETE WITH SUBTEST DESCRIPTIONS AND "CODING COMMENTS".

5.3 USING THE PROGRAM TO DIAGNOSE A FAULT

WHEN AN ERROR OCCURS, ONE OF THE THINGS THAT'S IMPORTANT TO NOTE IS WHAT PASS THE ERROR OCCURRED ON. IF THE PASS NUMBER IS ODD AND IS THREE OR GREATER, THE ERROR MIGHT BE T-BIT SENSITIVE. TRY RUNNING THE PROGRAM AGAIN WITH BIT 12 OF THE SWITCH REG. EQUAL TO "1" TO INHIBIT T-BIT TRAPPING. IF THE PASS NUMBER IS GREATER THAN ONE, THE ERROR MAY BE ITERATION SENSITIVE. TRY RUNNING THE PROGRAM AGAIN WITH BIT 11 OF THE SWITCH REG. EQUAL TO "1" TO INHIBIT ITERATIONS. THESE HINTS SHOULD HELP YOU DETERMINE WHAT MAKES THE MACHINE FAIL AND WHEN.

IF YOU HAVE BEEN RUNNING WITH BIT 15 OF THE SWITCH REG. EQUAL TO "0", THEN YOU ARE ABLE TO LOOK AT ALL THE ERRORS THAT MAY BE RELATED TO THE FAULT YOU ARE DIAGNOSING. A FAULT IN AN EARLIER TEST MAY RESULT IN ERRORS DURING LATER TESTS WHICH MAY GIVE YOU MORE CLUES ABOUT THE NATURE OF THE FAULT. NOW USE THE METHOD OUTLINED IN SECTION 3.2 FOR EACH ERROR TO GATHER AS MUCH INFORMATION AS POSSIBLE.

440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495

496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514

NOW TO TEST YOUR IDEAS ON THE CAUSE OF THE FAILURE,
YOU MAY WANT TO SCOPE THIS ERROR CONDITION. SET BIT 09
OF THE SWITCH REG. EQUAL TO "1" TO LOOP ON THE ERROR.
FOR AN EVEN TIGHTER SCOPE LOOP THE ERROR CALL CAN BE
REPLACED WITH A BRANCH (REFER TO COMMENTS BY ERROR CALLS
IN THE PROGRAM LISTING).

OR YOU COULD LOOP ON THE TEST BY EITHER SETTING BIT 14
OF THE SWITCH REG. EQUAL TO "1" OR BY SETTING BIT 08 OF THE
SWITCH REG. EQUAL TO "1" AND THEN SETTING THE TEST NUMBER
IN BITS 07-00 OF THE SWITCH REG. YOU WILL PROBABLY WANT TO
INHIBIT ERROR TYPEOUTS BY SETTING BIT 13 OF THE SWITCH REG.
EQUAL TO "1".

515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570

001100

000011
000012
000015
000200
177776

177774
177772
177570
177570

000000
000001
000002
000003
000004
000005
000006
000007
000006
000007

000000
000040

```

.TITLE MD-11-DFKTH-A PDP 11/34 MEM MGMT DIAG
.*COPYRIGHT (C) JAN-77
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*
.*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
.*          12      INHIBIT TRACE TRAP
.*          11      INHIBIT ITERATIONS
.*          10      BELL ON ERROR
.*           9      LOOP ON ERROR
.*           8      LOOP ON TEST IN SWR<7:0>
.SBTTL BASIC DEFINITIONS
.*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE     ;;BASIC DEFINITION OF SCOPE CALL

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

.*GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0      ;;GENERAL REGISTER
R1= %1      ;;GENERAL REGISTER
R2= %2      ;;GENERAL REGISTER
R3= %3      ;;GENERAL REGISTER
R4= %4      ;;GENERAL REGISTER
R5= %5      ;;GENERAL REGISTER
R6= %6      ;;GENERAL REGISTER
R7= %7      ;;GENERAL REGISTER
SP= %6      ;;STACK POINTER
PC= %7      ;;PROGRAM COUNTER

.*PRIORITY LEVEL DEFINITIONS
PRO= 0      ;;PRIORITY LEVEL 0
PRI= 40     ;;PRIORITY LEVEL 1

```

571 000100
572 000140
573 000200
574 000240
575 000300
576 000340

PR2= 100 :: PRIORITY LEVEL 2
PR3= 140 :: PRIORITY LEVEL 3
PR4= 200 :: PRIORITY LEVEL 4
PR5= 240 :: PRIORITY LEVEL 5
PR6= 300 :: PRIORITY LEVEL 6
PR7= 340 :: PRIORITY LEVEL 7

577
578
579 100000
580 040000
581 020000
582 010000
583 004000
584 002000
585 001000
586 000400
587 000200
588 000100
589 000040
590 000020
591 000010
592 000004
593 000002
594 000001

.*"SWITCH REGISTER" SWITCH DEFINITIONS

SW15= 100000
SW14= 40000
SW13= 20000
SW12= 10000
SW11= 4000
SW10= 2000
SW09= 1000
SW08= 400
SW07= 200
SW06= 100
SW05= 40
SW04= 20
SW03= 10
SW02= 4
SW01= 2
SW00= 1
.EQUIV SW09, SW9
.EQUIV SW08, SW8
.EQUIV SW07, SW7
.EQUIV SW06, SW6
.EQUIV SW05, SW5
.EQUIV SW04, SW4
.EQUIV SW03, SW3
.EQUIV SW02, SW2
.EQUIV SW01, SW1
.EQUIV SW00, SW0

595
596
597
598
599
600
601
602
603
604
605
606
607 100000
608 040000
609 020000
610 010000
611 004000
612 002000
613 001000
614 000400
615 000200
616 000100
617 000040
618 000020
619 000010
620 000004
621 000002
622 000001

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)

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

623
624
625
626

```

627 .EQUIV BIT05,BIT5
628 .EQUIV BIT04,BIT4
629 .EQUIV BIT03,BIT3
630 .EQUIV BIT02,BIT2
631 .EQUIV BIT01,BIT1
632 .EQUIV BIT00,BIT0
633
634 ;#BASIC "CPU" TRAP VECTOR ADDRESSES
635 000004 ERRVEC= 4 ;: TIME OUT AND OTHER ERRORS
636 000010 RESVEC= 10 ;: RESERVED AND ILLEGAL INSTRUCTIONS
637 000014 TBITVEC=14 ;: "T" BIT
638 000014 TRTVEC= 14 ;: TRACE TRAP
639 000014 BPTVEC= 14 ;: BREAKPOINT TRAP (BPT)
640 000020 IOTVEC= 20 ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
641 000024 PMRVEC= 24 ;: POWER FAIL
642 000030 EMTVEC= 30 ;: EMULATOR TRAP (EMT) **ERROR**
643 000034 TRAPVEC=34 ;: "TRAP" TRAP
644 000060 TKVEC= 60 ;: TTY KEYBOARD VECTOR
645 000064 TPVEC= 64 ;: TTY PRINTER VECTOR
646 000240 PIRQVEC=240 ;: PROGRAM INTERRUPT REQUEST VECTOR
647 .SBTTL MEMORY MANAGEMENT DEFINITIONS
648
649 ;#KT11 VECTOR ADDRESS
650
651 000250 MMVEC= 250
652
653 ;#KT11 STATUS REGISTER ADDRESSES
654
655 177572 SR0= 177572
656 177574 SR1= 177574
657 177576 SR2= 177576
658 172516 SR3= 172516
659
660 ;#USER "I" PAGE DESCRIPTOR REGISTERS
661
662 177600 UIPDR0= 177600
663 177602 UIPDR1= 177602
664 177604 UIPDR2= 177604
665 177606 UIPDR3= 177606
666 177610 UIPDR4= 177610
667 177612 UIPDR5= 177612
668 177614 UIPDR6= 177614
669 177616 UIPDR7= 177616
670
671 ;#USER "I" PAGE ADDRESS REGISTERS
672
673 177640 UIPAR0= 177640
674 177642 UIPAR1= 177642
675 177644 UIPAR2= 177644
676 177646 UIPAR3= 177646
677 177650 UIPAR4= 177650
678 177652 UIPAR5= 177652
679 177654 UIPAR6= 177654
680 177656 UIPAR7= 177656
681
682 ;#KERNEL "I" PAGE DESCRIPTOR REGISTERS

```

```

683
684      172300      KIPDR0= 172300
685      172302      KIPDR1= 172302
686      172304      KIPDR2= 172304
687      172306      KIPDR3= 172306
688      172310      KIPDR4= 172310
689      172312      KIPDR5= 172312
690      172314      KIPDR6= 172314
691      172316      KIPDR7= 172316
692
693      ;#KERNEL "I" PAGE ADDRESS REGISTERS
694
695      172340      KIPAR0= 172340
696      172342      KIPAR1= 172342
697      172344      KIPAR2= 172344
698      172346      KIPAR3= 172346
699      172350      KIPAR4= 172350
700      172352      KIPAR5= 172352
701      172354      KIPAR6= 172354
702      172356      KIPAR7= 172356
703
704      .EQUIV SP,KSP
705      .EQUIV SP,USP
706      .EQUIV BIT4,TBIT
707      .EQUIV BIT6,WBIT
708      001100      KERSTK= STACK
709      000700      USESTK= STACK-200
710
711      ;#ADDITIONAL DEFINITIONS
712      ;#
713
714      .SBTTL TRAP CATCHER
715
716      000000      .=0
717
718      ;#ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
719      ;#SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
720      ;#LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
721      000174      000174      .=174
722      000174      000000      DISPRG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
723      000176      000000      SWREG:  .WORD 0          ;;SOFTWARE SWITCH REGISTER
724
725      000200      000137      020000      .SBTTL STARTING ADDRESS(ES)
726      .SBTTL JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
727      .SBTTL ACT11 HOOKS
728
729      ;#*****
730      ;#HOOKS REQUIRED BY ACT11
731      000204      000046      $SVPC=.          ;SAVE PC
732      000046      034410      .=46          SENDAD          ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
733      000052      000052      .=52
734      000052      000000      .WORD 0          ;;2)SET LOC.52 TO ZERO
735      000204      000204      .=$SVPC          ;; RESTORE PC
736      .SBTTL APT PARAMETER BLOCK
737
738      ;#*****

```

| | | |
|-----|--------|--------|
| 739 | | |
| 740 | | |
| 741 | | 000204 |
| 742 | | 000024 |
| 743 | 000024 | 000200 |
| 744 | | 000044 |
| 745 | 000044 | 000204 |
| 746 | | 000204 |
| 747 | | |
| 748 | | |
| 749 | | |
| 750 | | |
| 751 | 000204 | |
| 752 | 000204 | 000000 |
| 753 | 000206 | 001226 |
| 754 | 000210 | 000010 |
| 755 | 000212 | 000020 |
| 756 | 000214 | 000005 |
| 757 | 000216 | 000052 |

```

;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
.SX=      ;;SAVE CURRENT LOCATION
=24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
200      ;;FOR APT START UP
=44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
$APTHDR  ;;POINT TO APT HEADER BLOCK
=.SX     ;;RESET LOCATION COUNTER
;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
;INTERFACE SPEC.

```

```

$APTHD:
SHIBTS: .WORD 0      ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MADR:  .WORD $MAIL  ;; ADDRESS OF APT MAILBOX (BITS 0-15)
$TSTM:  .WORD 10     ;; RUN TIM OF LONGEST TEST
$PASTM: .WORD 20     ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD 5      ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
        .WORD $ETEND-$MAIL/2 ;; LENGTH MAILBOX-ETABLE(WORDS)

```

.SBTTL COMMON TAGS

| | | |
|-----|--------|---------------|
| 758 | | |
| 759 | | |
| 760 | | |
| 761 | | |
| 762 | | |
| 763 | | |
| 764 | | 001100 |
| 765 | 001100 | |
| 766 | 001100 | 000000 |
| 767 | 001102 | 000 |
| 768 | 001103 | 000 |
| 769 | 001104 | 000000 |
| 770 | 001106 | 000000 |
| 771 | 001110 | 000000 |
| 772 | 001112 | 000000 |
| 773 | 001114 | 000 |
| 774 | 001115 | 001 |
| 775 | 001116 | 000000 |
| 776 | 001120 | 000000 |
| 777 | 001122 | 000000 |
| 778 | 001124 | 000000 |
| 779 | 001126 | 000000 |
| 780 | 001130 | 000000 |
| 781 | 001132 | 000000 |
| 782 | 001134 | 000 |
| 783 | 001135 | 000 |
| 784 | 001136 | 000000 |
| 785 | 001140 | 177570 |
| 786 | 001142 | 177570 |
| 787 | 001144 | 177560 |
| 788 | 001146 | 177562 |
| 789 | 001150 | 177564 |
| 790 | 001152 | 177566 |
| 791 | 001154 | 000 |
| 792 | 001155 | 002 |
| 793 | 001156 | 012 |
| 794 | 001157 | 000 |
| 795 | 001160 | 000000 |
| 796 | | |
| 797 | 001162 | 000000 |
| 798 | 001164 | 000000 |
| 799 | 001166 | 000000 |
| 800 | 001170 | 000000 |
| 801 | 001172 | 000000 |
| 802 | 001174 | 000000 |
| 803 | 001176 | 000000 |
| 804 | 001200 | 000000 |
| 805 | 001202 | 000000 |
| 806 | 001204 | 000000 |
| 807 | 001206 | 000000 |
| 808 | 001210 | 000000 |
| 809 | 001212 | 000000 |
| 810 | 001214 | 000000 |
| 811 | 001216 | 177607 000377 |
| 812 | 001222 | 077 |
| 813 | 001223 | 015 |

```

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

```

SCMTAG: . =1100

```

.SBTTTL COMMON TAGS
SCMTAG: . =1100
          .WORD 0
STSTNM: .BYTE 00
SERFLG: .BYTE 0000
$ICNT:  .WORD 0000
$LPADR: .WORD 0000
$LPERR: .WORD 0000
$ERTTL: .WORD 0000
$ITEMB: .BYTE 001
$ERMAX: .BYTE 1
$ERRPC: .WORD 0000
$GDADR: .WORD 0000
$BDADR: .WORD 0000
$GDADR: .WORD 0000
$BDADR: .WORD 0000
          .WORD 0000
          .WORD 0000
SAUTOB: .BYTE 0000
SINTAG: .BYTE 0000
          .WORD 0
SWR:     .WORD DSWR
DISPLAY: .WORD DDISP
$TKS:    177560
$TKB:    177562
$TPS:    177564
$TPB:    177566
$NULL:   .BYTE 0
$FILLS:  .BYTE 2
$FILLC:  .BYTE 12
$TPFLG:  .BYTE 0
$REGAD:  .WORD 0
$REG0:   .WORD 0
$REG1:   .WORD 00
$REG2:   .WORD 00
$REG3:   .WORD 00
$REG4:   .WORD 00
$REG5:   .WORD 00
$TMP0:   .WORD 00
$TMP1:   .WORD 00
$TMP2:   .WORD 00
$TMP3:   .WORD 00
$TMP4:   .WORD 00
$TPMS:   .WORD 0
$TIMES:  0
$ESCAPE: 0
$BELL:   .ASCIZ <207><377><377>
$QUES:   .ASCII /?/
$CRLF:   .ASCII <15>

```

;; START OF COMMON TAGS

```

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

;; AUTOMATIC MODE INDICATOR
;; INTERRUPT MODE INDICATOR

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

```

814 001224 000012
815
816
817
818
819
820 001226
821 001226 000000
822 001230 000000
823 001232 000000
824 001234 000000
825 001236 000000
826 001240 000000
827 001242 000000
828 001244 000000
829 001246
830 001246 000
831 001247 000
832 001250 000000
833 001252 000000
834 001254 000000
835
836
837
838
839
840
841 001256 000
842 001257 000
843
844
845
846
847 001260 000000
848
849 001262 000
850 001263 000
851 001264 000000
852 001266 000
853 001267 000
854 001270 000000
855 001272 000
856 001273 000
857 001274 000000
858 001276 000000
859 001300 000000
860 001302 000000
861 001304 000000
862 001306 000000
863 001310 000000
864 001312 000000
865 001314 000000
866 001316 000000
867 001320 000000
868 001322 000000
869 001324 000000

```

SLF: .ASCIZ <12>          ;;LINE FEED
;*****
.SBTL  APT MAILBOX-ETABLE
;*****
.EVEN
SMAIL:          ;; APT MAILBOX
MSGTY: .WORD    AMSGTY  ;; MESSAGE TYPE CODE
SFATAL: .WORD   AFATAL  ;; FATAL ERROR NUMBER
STESTN: .WORD   ATESTN  ;; TEST NUMBER
SPASS:  .WORD   APASS   ;; PASS COUNT
SDEVCT: .WORD   ADEVCT  ;; DEVICE COUNT
SUNIT:  .WORD   AUNIT   ;; I/O UNIT NUMBER
SMSGAD: .WORD   AMSGAD  ;; MESSAGE ADDRESS
SMSGLG: .WORD   AMSGLG  ;; MESSAGE LENGTH
SETABLE:       ;; APT ENVIRONMENT TABLE
SENV:  .BYTE    AENV    ;; ENVIRONMENT BYTE
SENVN: .BYTE    AENVN   ;; ENVIRONMENT MODE BITS
SSWREG: .WORD   ASWREG  ;; APT SWITCH REGISTER
SUSWR:  .WORD   AUSWR   ;; USER SWITCHES
SCPUOP: .WORD   ACPUOP  ;; CPU TYPE, OPTIONS
;
;          BITS 15-11=CPU TYPE
;          11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
;          11/70=06, PDQ=07, Q=10
;
; BIT 10=REAL TIME CLOCK
; BIT 9=FLOATING POINT PROCESSOR
; BIT 8=MEMORY MANAGEMENT
SMAMS1: .BYTE    AMAMS1  ;; HIGH ADDRESS, M.S. BYTE
SMTYP1: .BYTE    AMTYP1  ;; MEM. TYPE, BLK#1
;
;          MEM. TYPE BYTE -- (HIGH BYTE)
;          900 NSEC CORE=001
;          300 NSEC BIPOLAR=002
;          500 NSEC MOS=003
SMADR1: .WORD    AMADR1  ;; HIGH ADDRESS, BLK#1
;
;          MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
SMAMS2: .BYTE    AMAMS2  ;; HIGH ADDRESS, M.S. BYTE
SMTYP2: .BYTE    AMTYP2  ;; MEM. TYPE, BLK#2
SMADR2: .WORD    AMADR2  ;; MEM. LAST ADDRESS, BLK#2
SMAMS3: .BYTE    AMAMS3  ;; HIGH ADDRESS, M.S. BYTE
SMTYP3: .BYTE    AMTYP3  ;; MEM. TYPE, BLK#3
SMADR3: .WORD    AMADR3  ;; MEM. LAST ADDRESS, BLK#3
SMAMS4: .BYTE    AMAMS4  ;; HIGH ADDRESS, M.S. BYTE
SMTYP4: .BYTE    AMTYP4  ;; MEM. TYPE, BLK#4
SMADR4: .WORD    AMADR4  ;; MEM. LAST ADDRESS, BLK#4
SVECT1: .WORD    AVECT1  ;; INTERRUPT VECTOR#1, BUS PRIORITY#1
SVECT2: .WORD    AVECT2  ;; INTERRUPT VECTOR#2, BUS PRIORITY#2
SBASE:  .WORD    ABASE   ;; BASE ADDRESS OF EQUIPMENT UNDER TEST
SDEVN:  .WORD    ADEVN   ;; DEVICE MAP
SCDW1:  .WORD    ACDW1   ;; CONTROLLER DESCRIPTION WORD#1
SCDW2:  .WORD    ACDW2   ;; CONTROLLER DESCRIPTION WORD#2
SDDW0:  .WORD    ADDW0   ;; DEVICE DESCRIPTOR WORD#0
SDDW1:  .WORD    ADDW1   ;; DEVICE DESCRIPTOR WORD#1
SDDW2:  .WORD    ADDW2   ;; DEVICE DESCRIPTOR WORD#2
SDDW3:  .WORD    ADDW3   ;; DEVICE DESCRIPTOR WORD#3
SDDW4:  .WORD    ADDW4   ;; DEVICE DESCRIPTOR WORD#4
SDDW5:  .WORD    ADDW5   ;; DEVICE DESCRIPTOR WORD#5

```

870 001326 000000
871 001330 000000
872 001332 000000
873 001334 000000
874 001336 000000
875 001340 000000
876 001342 000000
877 001344 000000
878 001346 000000
879 001350 000000
880
881
882 001352
883
884
885 001352 000000
886 001354 000000
887 001356 000000
888 001360 000000
889 001362 000000
890 001364 000000
891 001366 000000
892 001370 000000
893 001372 000000
894 001374 000000
895 001376 000000
896 001400 000000
897 001402 000000
898 001404 000000

SDDW6: .WORD ADDW6 ::: DEVICE DESCRIPTOR WORD#6
SDDW7: .WORD ADDW7 ::: DEVICE DESCRIPTOR WORD#7
SDDW8: .WORD ADDW8 ::: DEVICE DESCRIPTOR WORD#8
SDDW9: .WORD ADDW9 ::: DEVICE DESCRIPTOR WORD#9
SDDW10: .WORD ADDW10 ::: DEVICE DESCRIPTOR WORD#10
SDDW11: .WORD ADDW11 ::: DEVICE DESCRIPTOR WORD#11
SDDW12: .WORD ADDW12 ::: DEVICE DESCRIPTOR WORD#12
SDDW13: .WORD ADDW13 ::: DEVICE DESCRIPTOR WORD#13
SDDW14: .WORD ADDW14 ::: DEVICE DESCRIPTOR WORD#14
SDDW15: .WORD ADDW15 ::: DEVICE DESCRIPTOR WORD#15

SETEND:

TESTNO: .WORD 0 : HOLDS TEST NUMBER FOR TYPEOUTS
WASR6: .WORD 0 : USED TO STORE THE STACK POINTER AFTER A TRAP
TRAPPC: .WORD 0 : USED TO STORE THE PC OF A TRAP OR ABORT
TRAPPS: .WORD 0 : USED TO STORE THE PS OF A TRAP OR ABORT
WASSR0: .WORD 0 : USED TO STORE CONTENTS OF SR0
WASSR2: .WORD 0 : USED TO STORE CONTENTS OF SR2
TBITPS: .WORD 0 : SAVES THE PSM THAT MAY HAVE ITS T-BIT ON
ANDADR: .WORD 0 : HOLDS RESULT OF ADDRESSES BEING AND-ED
ORADR: .WORD 0 : HOLDS RESULT OF ADDRESSES BEING OR-ED
TONUM: .WORD 0 : HOLDS NUMBER OF TIME-OUTS
VIRT1: .WORD 0 : HOLDS VIRTUAL ADDRESS TO BE CONVERTED
VIRT2: .WORD 0 :
PBALO: .WORD 0 : HOLDS BITS <15:00> OF PHYSICAL ADDRESS
PBAHI: .WORD 0 : HOLDS BITS <17:16> OF PHYSICAL ADDRESS

.SBTTL ERROR POINTER TABLE

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

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

| | | | | | |
|-----|--------|--------|--|--|--|
| 899 | | | | | |
| 900 | | | | | |
| 901 | | | | | |
| 902 | | | | | |
| 903 | | | | | |
| 904 | | | | | |
| 905 | | | | | |
| 906 | | | | | |
| 907 | | | | | |
| 908 | | | | | |
| 909 | | | | | |
| 910 | | | | | |
| 911 | | | | | |
| 912 | | | | | |
| 913 | 001406 | | | | |
| 914 | | | | | |
| 915 | | | | | |
| 916 | 001406 | 041246 | | | |
| 917 | 001410 | 044655 | | | |
| 918 | 001412 | 050122 | | | |
| 919 | 001414 | 050756 | | | |
| 920 | | | | | |
| 921 | | | | | |
| 922 | 001416 | 041306 | | | |
| 923 | 001420 | 044725 | | | |
| 924 | 001422 | 050136 | | | |
| 925 | 001424 | 050763 | | | |
| 926 | | | | | |
| 927 | | | | | |
| 928 | 001426 | 041355 | | | |
| 929 | 001430 | 045015 | | | |
| 930 | 001432 | 050156 | | | |
| 931 | 001434 | 050772 | | | |
| 932 | | | | | |
| 933 | | | | | |
| 934 | 001436 | 041414 | | | |
| 935 | 001440 | 045015 | | | |
| 936 | 001442 | 050156 | | | |
| 937 | 001444 | 050772 | | | |
| 938 | | | | | |
| 939 | | | | | |
| 940 | 001446 | 041447 | | | |
| 941 | 001450 | 045015 | | | |
| 942 | 001452 | 050156 | | | |
| 943 | 001454 | 050772 | | | |
| 944 | | | | | |
| 945 | | | | | |
| 946 | 001456 | 041522 | | | |
| 947 | 001460 | 045015 | | | |
| 948 | 001462 | 050156 | | | |
| 949 | 001464 | 050772 | | | |
| 950 | | | | | |
| 951 | | | | | |
| 952 | 001466 | 041567 | | | |
| 953 | 001470 | 045055 | | | |
| 954 | 001472 | 050170 | | | |

\$ERRTB:

;*ITEM 1

EM1
DH1
DT1
DF1

;UNEXPECTED CPU TRAP TO LOC. 004
 ;OLD PC OLD PSW R6 WAS TESTNO ERRORPC
 ;TRAPPC, TRAPPS, WASR6, TESTNO, \$ERPPC, 0
 ;0,0,0,0,0

;*ITEM 2

EM2
DH2
DT2
DF2

;UNEXPECTED MEM. MGMT. TRAP TO LOC. 250
 ;OLD PC OLD PSW R6 WAS SR0 SR2 TESTNO ERRORPC
 ;TRAPPC, TRAPPS, WASR6, WASSR0, WASSR2, TESTNO, \$ERRPC,
 ;0,0,0,0,0,0,0

;*ITEM 3

EM3
DH3
DT3
DF3

;PRIORITY BITS SET WRONG IN PSW
 ;WROTE READ TESTNO ERRORPC
 ;\$REG0,\$REG1,TESTNO,\$ERRPC,0
 ;0,0,0,0

;*ITEM 4

EM4
DH3
DT3
DF3

;MODE BITS SET WRONG IN PSW
 ;WROTE READ TESTNO ERRORPC
 ;\$REG0,\$REG1,TESTNO,\$ERRPC,0
 ;0,0,0,0

;*ITEM 5

EM5
DH3
DT3
DF3

;DUAL ADDRESSING BETWEEN HI&LO BYTES OF PSW
 ;WROTE READ TESTNO ERRORPC
 ;\$REG0,\$REG1,TESTNO,\$ERRPC,0
 ;0,0,0,0

;*ITEM 6

EM6
DH3
DT3
DF3

;KERNEL R6 CHANGED BY WRITING USER R6
 ;WROTE READ TESTNO ERRORPC
 ;\$REG0,\$REG1,TESTNO,\$ERRPC,0
 ;0,0,0,0

;*ITEM 7

EM7
DH7
DT7

;A MEMORY MGMT. REG. TIMED OUT
 ;ADDRESS TESTNO ERRORPC
 ;\$REG0,TESTNO,\$ERRPC,0

| | | | | |
|------|--------|--------|-----------|--|
| 955 | 001474 | 050776 | DF7 | ;0,0,0 |
| 956 | | | | |
| 957 | | | ;*ITEM 10 | |
| 958 | 001476 | 041625 | EM10 | ;SUMMARY OF MEM. MGMT. REG. TIMEOUTS |
| 959 | 001500 | 045105 | DH10 | ;REGISTER-ADDRS NUM. OF |
| 960 | | | | ;AND-ED OR-ED TIMOUTS TESTNO ERRORPC |
| 961 | 001502 | 050200 | DT10 | ;ANDADR,ORADR,TONUM,TESTNO,\$ERRPC,0 |
| 962 | 001504 | 051001 | DF10 | ;0,0,1,0,0 |
| 963 | | | | |
| 964 | | | ;*ITEM 11 | |
| 965 | 001506 | 041671 | EM11 | ;MEM. MGMT. REG. WOULD NOT CLEAR |
| 966 | 001510 | 045205 | DH11 | ;REGISTR READ READ-(BINARY) |
| 967 | | | | ;ADDRESS (OCTAL) 5432109876543210 TESTNO ERRORPC |
| 968 | 001512 | 050214 | DT11 | ;\$REG0,\$REG1,\$REG1,TESTNO,\$ERRPC,0 |
| 969 | 001514 | 051006 | DF11 | ;0,0,2,0,0 |
| 970 | | | | |
| 971 | | | ;*ITEM 12 | |
| 972 | 001516 | 041731 | EM12 | ;MEM. MGMT. REG. BITS NOT SET CORRECTLY |
| 973 | 001520 | 045325 | DH12 | ;REGISTR WRITE READ READ |
| 974 | | | | ;ADDRESS (OCTAL) (OCTAL) (BINARY) TESTNO ERRORPC |
| 975 | 001522 | 050230 | DT12 | ;\$REG0,\$REG1,\$REG2,\$REG2,TESTNO,\$ERRPC,0 |
| 976 | 001524 | 051013 | DF12 | ;0,0,0,2,0,0 |
| 977 | | | | |
| 978 | | | ;*ITEM 13 | |
| 979 | 001526 | 042000 | EM13 | ;SRO EFFECTED BY WRITE TO PSW |
| 980 | 001530 | 045465 | DH13 | ;READ TESTNO ERRORPC |
| 981 | 001532 | 050246 | DT13 | ;\$REG0,TESTNO,\$ERRPC,0 |
| 982 | 001534 | 051021 | DF13 | ;0,0,0 |
| 983 | | | | |
| 984 | | | ;*ITEM 14 | |
| 985 | 001536 | 042035 | EM14 | ;SRI DID NOT READ ALL ZEROS |
| 986 | 001540 | 045465 | DH13 | ;READ TESTNO ERRORPC |
| 987 | 001542 | 050246 | DT13 | ;\$REG0,TESTNO,\$ERRPC,0 |
| 988 | 001544 | 051021 | DF13 | ;0,0,0 |
| 989 | | | | |
| 990 | | | ;*ITEM 15 | |
| 991 | 001546 | 042070 | EM15 | ;DUAL ADDRESSING BETWEEN BYTES OF PAR OR PDR |
| 992 | 001550 | 045325 | DH12 | ;REGISTER WRITE READ READ |
| 993 | | | | ;ADDRESS (OCTAL) (OCTAL) (BINARY) TESTNO ERRORPC |
| 994 | 001552 | 050230 | DT12 | ;\$REG0,\$REG1,\$REG2,\$REG2,TESTNO,\$ERRPC,0 |
| 995 | 001554 | 051013 | DF12 | ;0,0,0,2,0,0 |
| 996 | | | | |
| 997 | | | ;*ITEM 16 | |
| 998 | 001556 | 042144 | EM16 | ;DUAL ADDRESSING BETWEEN PAR-PDR'S |
| 999 | 001560 | 045515 | DH16 | ;PAR-PDR PAR-PDR |
| 1000 | | | | ;CLEARED EFFECTD EXPECTD RECEIVD TESTNO ERRORPC |
| 1001 | 001562 | 050256 | DT16 | ;\$REG0,\$REG1,\$REGS,\$REG2,TESTNO,\$ERRPC,0 |
| 1002 | 001564 | 051024 | DF16 | ;0,0,0,0,0,0 |
| 1003 | | | | |
| 1004 | | | ;*ITEM 17 | |
| 1005 | 001566 | 042206 | EM17 | ;PHYS. ADDR. FORMED READ WRONG IN MAINT. MODE |
| 1006 | 001570 | 045615 | DH17 | ;PHYSICAL VIRTUAL |
| 1007 | | | | ;ADDRESS ADDRESS KIPAR4 TESTNO ERRORPC |
| 1008 | 001572 | 050274 | DT17 | ;\$BAL0,VIRT1,\$REG4,TESTNO,\$ERRPC,0 |
| 1009 | 001574 | 051032 | DF17 | ;3,0,0,0,0 |
| 1010 | | | | |

| | | | | | | |
|------|--------|--------|---|----------|----------|---|
| 1011 | | | ; | *ITEM 20 | | |
| 1012 | 001576 | 042256 | | EM20 | : | PHYS. ADDR. FORMED READ WRONG IN RELOCATE MODE |
| 1013 | 001600 | 045705 | | DH20 | : | PHYSICAL PAR 4 PAR 5 |
| 1014 | | | | | : | ADDRESS VBA VBA PAR 4 PAR 5 PSW TESTNO |
| 1015 | 001602 | 050310 | | DT20 | : | PBALO, VIRT1, VIRT2, SREG4, SREG5, STMP0, TESTNO, SERRPC, 0 |
| 1016 | 001604 | 051037 | | DF20 | : | 3, 0, 0, 0, 0, 0, 0, 0 |
| 1017 | | | | | | |
| 1018 | | | | ; | *ITEM 21 | |
| 1019 | 001606 | 042330 | | EM21 | : | W-BIT DID NOT GET SET IN PDR |
| 1020 | 001610 | 046033 | | DH21 | : | PDR VIRTUAL |
| 1021 | | | | | : | TESTED ADDRESS TESTNO ERRORPC |
| 1022 | 001612 | 050332 | | DT21 | : | SREG5, SREG3, TESTNO, SERRPC, 0 |
| 1023 | 001614 | 051047 | | DF21 | : | 0, 0, 0, 0 |
| 1024 | | | | | | |
| 1025 | | | | ; | *ITEM 22 | |
| 1026 | 001616 | 042365 | | EM22 | : | W-BIT SET IN MORE THAN ONE PDR |
| 1027 | 001620 | 046113 | | DH22 | : | PDR IN PDR VIRTUAL |
| 1028 | | | | | : | ERROR TESTED ADDRESS TESTNO ERRORPC |
| 1029 | 001622 | 050344 | | DT22 | : | SREG0, SREG5, SREG3, TESTNO, SERRPC, 0 |
| 1030 | 001624 | 051053 | | DF22 | : | 0, 0, 0, 0, 0 |
| 1031 | | | | | | |
| 1032 | | | | ; | *ITEM 23 | |
| 1033 | 001626 | 042424 | | EM23 | : | W-BIT NOT CLEARED BY WRITING TO PDR |
| 1034 | 001630 | 046212 | | DH23 | : | PDR TESTNO ERRORPC |
| 1035 | 001632 | 050360 | | DT23 | : | SREG5, TESTNO, SERRPC, 0 |
| 1036 | 001634 | 051060 | | DF23 | : | 0, 0, 0 |
| 1037 | | | | | | |
| 1038 | | | | ; | *ITEM 24 | |
| 1039 | 001636 | 042470 | | EM24 | : | WRITING SRO SET W-BIT IN KIPDR7 |
| 1040 | 001640 | 046242 | | DH24 | : | PDR WAS EXPECTD TESTNO ERRORPC |
| 1041 | 001642 | 050370 | | DT24 | : | SREG2, SREG1, TESTNO, SERRPC, 0 |
| 1042 | 001644 | 051063 | | DF24 | : | 0, 0, 0, 0 |
| 1043 | | | | | | |
| 1044 | | | | ; | *ITEM 25 | |
| 1045 | 001646 | 042530 | | EM25 | : | W-BIT GOT SET DURING ODD ADDR. ABORT |
| 1046 | 001650 | 046242 | | DH24 | : | PDR WAS EXPECTD TESTNO ERRORPC |
| 1047 | 001652 | 050370 | | DT24 | : | SREG2, SREG1, TESTNO, SERRPC, 0 |
| 1048 | 001654 | 051063 | | DF24 | : | 0, 0, 0, 0 |
| 1049 | | | | | | |
| 1050 | | | | ; | *ITEM 26 | |
| 1051 | 001656 | 042575 | | EM26 | : | MEMORY MGMT. ACCESS ABORT DID NOT OCCUR |
| 1052 | 001660 | 046302 | | DH26 | : | PDR 4 PSW TESTNO ERRORPC |
| 1053 | 001662 | 050402 | | DT26 | : | SREG2, STMP0, TESTNO, SERRPC, 0 |
| 1054 | 001664 | 051063 | | DF24 | : | 0, 0, 0, 0 |
| 1055 | | | | | | |
| 1056 | | | | ; | *ITEM 27 | |
| 1057 | 001666 | 042645 | | EM27 | : | ACCESS ERROR DID NOT ABORT INSTRUCTION |
| 1058 | 001670 | 046302 | | DH26 | : | PDR 4 PSW TESTNO ERRORPC |
| 1059 | 001672 | 050402 | | DT26 | : | SREG2, STMP0, TESTNO, SERRPC, 0 |
| 1060 | 001674 | 051063 | | DF24 | : | 0, 0, 0, 0 |
| 1061 | | | | | | |
| 1062 | | | | ; | *ITEM 30 | |
| 1063 | 001676 | 042714 | | EM30 | : | SRO DID NOT REPORT ACCESS ERROR CORRECTLY |
| 1064 | 001700 | 046342 | | DH30 | : | SRO WAS EXPECTD PDR 4 PSW TESTNO ERRORPC |
| 1065 | 001702 | 050414 | | DT30 | : | WASSRO, SREG3, SREG2, STMP0, TESTNO, SERRPC, 0 |
| 1066 | 001704 | 051067 | | DF30 | : | 0, 0, 0, 0, 0, 0 |

| | | | | | |
|------|--------|--------|--|--|--|
| 1067 | | | | | |
| 1068 | | | | | |
| 1069 | 001706 | 042766 | | | |
| 1070 | 001710 | 046422 | | | |
| 1071 | 001712 | 050432 | | | |
| 1072 | 001714 | 051067 | | | |
| 1073 | | | | | |
| 1074 | | | | | |
| 1075 | 001716 | 043033 | | | |
| 1076 | 001720 | 046502 | | | |
| 1077 | 001722 | 050450 | | | |
| 1078 | 001724 | 051067 | | | |
| 1079 | | | | | |
| 1080 | | | | | |
| 1081 | 001726 | 043114 | | | |
| 1082 | 001730 | 046562 | | | |
| 1083 | 001732 | 050466 | | | |
| 1084 | 001734 | 051063 | | | |
| 1085 | | | | | |
| 1086 | | | | | |
| 1087 | 001736 | 043177 | | | |
| 1088 | 001740 | 046622 | | | |
| 1089 | 001742 | 050500 | | | |
| 1090 | 001744 | 051067 | | | |
| 1091 | | | | | |
| 1092 | 001746 | 042766 | | | |
| 1093 | 001750 | 046702 | | | |
| 1094 | 001752 | 050516 | | | |
| 1095 | 001754 | 051067 | | | |
| 1096 | | | | | |
| 1097 | | | | | |
| 1098 | 001756 | 042766 | | | |
| 1099 | 001760 | 046762 | | | |
| 1100 | 001762 | 050534 | | | |
| 1101 | 001764 | 051063 | | | |
| 1102 | | | | | |
| 1103 | | | | | |
| 1104 | 001766 | 043255 | | | |
| 1105 | 001770 | 047022 | | | |
| 1106 | | | | | |
| 1107 | 001772 | 050546 | | | |
| 1108 | 001774 | 051067 | | | |
| 1109 | | | | | |
| 1110 | | | | | |
| 1111 | 001776 | 043322 | | | |
| 1112 | 002000 | 047137 | | | |
| 1113 | 002002 | 050564 | | | |
| 1114 | 002004 | 051063 | | | |
| 1115 | | | | | |
| 1116 | | | | | |
| 1117 | 002006 | 043371 | | | |
| 1118 | 002010 | 046762 | | | |
| 1119 | 002012 | 050534 | | | |
| 1120 | 002014 | 051063 | | | |
| 1121 | | | | | |
| 1122 | | | | | |

;*ITEM 31

EM31
DH31
DT31
DF30:SR2 DID NOT LOCKUP CORRECT VIRTUAL ADDR.
:SR2 WAS EXPECTD PDR 4 PSM TESTNO ERRORPC
:WASSR2,\$REG4,\$REG2,\$TMP0,TESTNO,\$ERRPC,0
:0,0,0,0,0,0

;*ITEM 32

EM32
DH32
DT32
DF30:PAGE LGTH. ABORT OCCURRED WHEN IT SHOULDN'T HAVE
:V.B.A. KIPDR4 SRO WAS SR2 WAS TESTNO ERRORPC
:\$REG0,\$REG4,WASSRO,WASSR2,TESTNO,\$ERRPC,0
:0,0,0,0,0,0

;*ITEM 33

EM33
DH33
DT33
DF24:PAGE LGTH. ABORT DID NOT OCCUR WHEN IT SHOULD HAVE
:V.B.A. KIPDR4 TESTNO ERRORPC
:\$REG0,\$REG4,TESTNO,\$ERRPC,0
:0,0,0,0

;*ITEM 34

EM34
DH34
DT34
DF30:SRO DID NOT REPORT PAGE LGTH. ABORT CORRECTLY
:V.B.A. KIPDR4 SRO WAS EXPECTD TESTNO ERRORPC
:\$REG0,\$REG4,WASSRO,\$REG2,TESTNO,\$ERRPC,0
:0,0,0,0,0,0

;*ITEM 35

EM31
DH35
DT35
DF30:SR2 DID NOT LOCKUP CORRECT VIRUAL ADDR.
:V.B.A. KIPDR4 SR2 WAS EXPECTD TESTNO ERRORPC
:\$REG0,\$REG4,WASSR2,\$REG3,TESTNO,\$ERRPC,0
:0,0,0,0,0,0

;*ITEM 36

EM31
DH36
DT36
DF24:SR2 DID NOT LOCKUP CORRECT VIRUAL ADDR.
:SR2 WAS EXPECTD TESTNO ERRORPC
:WASSR2,\$REG1,TESTNO,\$ERRPC,0
:0,0,0,0

;*ITEM 37

EM37
DH37
DT37
DF30:SRO OR SR2 CHANGED BY A SECOND ABORT
:FIRST ABORT SECOND ABORT
:SRO WAS SR2 WAS SRO WAS SR2 WAS TESTNO ERRORPC
:\$TMP0,\$TMP2,WASSRO,WASSR2,TESTNO,\$ERRPC,0
:0,0,0,0,0,0

;*ITEM 40

EM40
DH40
DT40
DF24:SRO OR SR2 WAS NOT "RESET" BY A RESET
:SRO WAS SR2 WAS TESTNO ERRORPC
:WASSRO,WASSR2,TESTNO,\$ERRPC,0
:0,0,0,0

;*ITEM 41

EM41
DH36
DT36
DF24:SR2 NOT TRACKING CORRECTLY
:SR2 WAS EXPECTD TESTNO ERRORPC
:WASSR2,\$REG1,TESTNO,\$ERRPC,0
:0,0,0,0

;*ITEM 42

| | | | | |
|------|--------|--------|------------|--|
| 1123 | 002016 | 043424 | EM42 | : DID NOT TRAP THRU KERNEL SPACE |
| 1124 | 002020 | 047177 | DH42 | : PSW WAS R6 WAS TESTNO ERRORPC |
| 1125 | 002022 | 050576 | DT42 | : \$REG1, \$REG2, TESTNO, \$ERRPC, 0 |
| 1126 | 002024 | 051063 | DF24 | : 0, 0, 0, 0 |
| 1127 | | | | |
| 1128 | | | ; *ITEM 43 | |
| 1129 | 002026 | 043463 | EM43 | : KT ERROR SERVICED ON ODD ADDR. ERROR |
| 1130 | 002030 | 046212 | DH23 | : PDR TESTNO ERRORPC |
| 1131 | 002032 | 050360 | DT23 | : \$REG5, TESTNO, \$ERRPC, 0 |
| 1132 | 002034 | 051060 | DF23 | : 0, 0, 0 |
| 1133 | | | | |
| 1134 | | | ; *ITEM 44 | |
| 1135 | 002036 | 043530 | EM44 | : SR0 OR SR2 CHANGED BY ODD ADDR. ERROR |
| 1136 | 002040 | 047237 | DH44 | : EXPECTED RECEIVED |
| 1137 | | | | : SR0 SR2 SR0 WAS SR2 WAS TESTNO ERRORPC |
| 1138 | 002042 | 050610 | DT44 | : \$REG0, \$REG1, WASSR0, WASSR2, TESTNO, \$ERRPC, 0 |
| 1139 | 002044 | 051067 | DF30 | : 0, 0, 0, 0, 0, 0 |
| 1140 | | | | |
| 1141 | | | ; *ITEM 45 | |
| 1142 | 002046 | 043576 | EM45 | : ERROR DURING "DOUBLE ERROR" (KT & ODD ADDR.) |
| 1143 | 002050 | 047351 | DH45 | : EXPECTED: |
| 1144 | | | | : PSW PC SR0 SR2 |
| 1145 | | | | : 170017 (3\$+4) 020147 (3\$) |
| 1146 | | | | : RECEIVED |
| 1147 | | | | : PSW PC SR0 SR2 TESTNO ERRORPC |
| 1148 | 002052 | 050626 | DT45 | : \$REG1, \$REG3, WASSR0, WASSR2, TESTNO, \$ERRPC, 0 |
| 1149 | 002054 | 051067 | DF30 | : 0, 0, 0, 0, 0, 0 |
| 1150 | | | | |
| 1151 | | | ; *ITEM 46 | |
| 1152 | 002056 | 043653 | EM46 | : MFPI INSTRUCTION PUSHED WRONG DATA |
| 1153 | 002060 | 047546 | DH46 | : DATA DATA |
| 1154 | | | | : EXPECTD RECEIVD TESTNO ERRORPC |
| 1155 | 002062 | 050644 | DT46 | : \$REG0, \$REG1, TESTNO, \$ERRPC, 0 |
| 1156 | 002064 | 051075 | DF46 | : 0, 0, 0, 0 |
| 1157 | | | | |
| 1158 | | | ; *ITEM 47 | |
| 1159 | 002066 | 043716 | EM47 | : MTPI INSTRUCTION LOADED WRONG DATA |
| 1160 | 002070 | 047546 | DH46 | : DATA DATA |
| 1161 | | | | : EXPECTD RECEIVD TESTNO ERRORPC |
| 1162 | 002072 | 050644 | DT46 | : \$REG0, \$REG1, TESTNO, \$ERRPC, 0 |
| 1163 | 002074 | 051075 | DF46 | : 0, 0, 0, 0 |
| 1164 | | | | |
| 1165 | | | ; *ITEM 50 | |
| 1166 | 002076 | 043761 | EM50 | : STACK NOT PUSHED BY MFPI-MTPI |
| 1167 | 002100 | 047623 | DH50 | : TESTNO ERRORPC |
| 1168 | 002102 | 050656 | DT50 | : TESTNO, \$ERRPC, 0 |
| 1169 | 002104 | 051101 | DF50 | : 0, 0 |
| 1170 | | | | |
| 1171 | | | ; *ITEM 51 | |
| 1172 | 002106 | 044017 | EM51 | : KERNEL PAGE ACCESSED INSTEAD OF USER: MFPI-MTPI |
| 1173 | 002110 | 047643 | DH51 | : SR0 WAS SR2 WAS TESTNO ERRORPC |
| 1174 | 002112 | 050664 | DT51 | : WASSR0, WASSR2, TESTNO, \$ERRPC, 0 |
| 1175 | 002114 | 051103 | DF51 | : 0, 0, 0, 0 |
| 1176 | | | | |
| 1177 | | | ; *ITEM 52 | |
| 1178 | 002116 | 044075 | EM52 | : WRONG PDR'S REFERENCED WHILE IN RELOCATE MODE |

| | | | | |
|------|--------|--------|-----------|---|
| 1179 | 002120 | 047703 | DH52 | :PHYSICL PAR 4 |
| 1180 | | | | :ADDRESS V.B.A. PAR 4 SR0 WAS SR2 WAS PSW TESTNO |
| 1181 | 002122 | 050676 | DT52 | :PBALO,VIRT1,\$REG4,WASSR0,WASSR2,\$TMPO,TESTNO,\$ERRPC,0 |
| 1182 | 002124 | 051107 | DF52 | :3,0,0,0,0,0,0,0 |
| 1183 | | | ;*ITEM 53 | |
| 1184 | 002126 | 044153 | EM53 | :MFPD INSTRUCTION PUSHED WRONG DATA |
| 1185 | 002130 | 047546 | DH46 | :DATA DATA |
| 1186 | | | | :EXPECTD RECEIVD TESTNO ERRORPC |
| 1187 | 002132 | 050644 | DT46 | :\$REG0,\$REG1,TESTNO,\$ERRPC,0 |
| 1188 | 002134 | 051075 | DF46 | :0,0,0,0 |
| 1189 | | | | |
| 1190 | | | ;*ITEM 54 | |
| 1191 | 002136 | 044216 | EM54 | :STACK NOT PUSHED BY MFPD-MTPD |
| 1192 | 002140 | 047623 | DH50 | :TESTNO ERRORPC |
| 1193 | 002142 | 050656 | DT50 | :TESTNO,\$ERRPC,0 |
| 1194 | 002144 | 051101 | DF50 | :0,0 |
| 1195 | | | | |
| 1196 | | | ;*ITEM 55 | |
| 1197 | 002146 | 044254 | EM55 | :PAR OR PDR WAS CHANGED BY A RESET |
| 1198 | 002150 | 045205 | DH11 | :REGISTR READ READ-(BINARY) |
| 1199 | | | | :ADDRESS (OCTAL) 5432109876543210 TESTNO ERRORPC |
| 1200 | 002152 | 050214 | DT11 | :\$REG0,\$REG1,\$REG1,TESTNO,\$ERRPC,0 |
| 1201 | 002154 | 051006 | DF11 | :0,0,2,0,0 |
| 1202 | | | | |
| 1203 | | | ;*ITEM 56 | |
| 1204 | 002156 | 044312 | EM56 | :ILLEGAL MODE 01 NOT ABORTED |
| 1205 | 002160 | 047623 | DH50 | :TESTNO ERRORPC |
| 1206 | 002162 | 050656 | DT50 | :TESTNO,\$ERRPC,0 |
| 1207 | 002164 | 051101 | DF50 | :0,0 |
| 1208 | | | | |
| 1209 | | | ;*ITEM 57 | |
| 1210 | 002166 | 044346 | EM57 | :SR0 DID NOT REPORT ILLEGAL MODE 01 CORRECTLY |
| 1211 | 002170 | 050021 | DH57 | :SR0 WAS EXPECTD TESTNO ERRORPC |
| 1212 | 002172 | 050720 | DT57 | :WASSR0,\$REG1,TESTNO,\$ERRPC,0 |
| 1213 | 002174 | 051117 | DF57 | :0,0,0,0 |
| 1214 | | | | |
| 1215 | | | ;*ITEM 60 | |
| 1216 | 002176 | 044423 | EM60 | :PSW CHANGED BY AN RTI IN USER MODE |
| 1217 | 002200 | 050061 | DH60 | :PSW WAS EXPECTD TESTNO ERRORPC |
| 1218 | 002202 | 050732 | DT60 | :\$REG1,\$REG2,TESTNO,\$ERRPC,0 |
| 1219 | 002204 | 051123 | DF60 | :0,0,0,0 |
| 1220 | | | | |
| 1221 | | | ;*ITEM 61 | |
| 1222 | 002206 | 044466 | EM61 | :MAINT MODE (SR0<8>) NOT DISABLED BY A RESET |
| 1223 | 002210 | 047623 | DH50 | :TESTNO ERRORPC |
| 1224 | 002212 | 050656 | DT50 | :TESTNO,\$ERRPC,0 |
| 1225 | 002214 | 051101 | DF50 | :0,0 |
| 1226 | | | | |
| 1227 | | | ;*ITEM 62 | |
| 1228 | 002216 | 044544 | EM62 | :DATA INCORRECT AFTER A MAINT. MODE WRITE |
| 1229 | 002220 | 045015 | DH3 | :WROTE READ TESTNO ERRORPC |
| 1230 | 002222 | 050156 | DT3 | :\$REG0,\$REG1,TESTNO,\$ERRPC,0 |
| 1231 | 002224 | 050772 | DF3 | :0,0,0,0 |
| 1232 | | | | |
| 1233 | | | ;*ITEM 63 | |
| 1234 | 002226 | 044615 | EM63 | :SOURCE RELOCATED IN MAINT. MODE |

K05

MD-11-DFKTH-A POP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006) 27-JAN-77 08:51 PAGE 26
ERROR POINTER TABLE

1235 002230 044725
1236 002232 050136
1237 002234 050763
1238
1239
1240 002236 042766
1241 002240 046762
1242 002242 050744
1243 002244 051063
1244

DH2
DT2
DF2

:OLD PC OLD PSW R6 WAS SR0 SR2 TESTNO ERRORPC
:TRAPPC,TRAPPS,WASR6,WASSR0,WASSR2,TESTNO,SERRPC,0
:0,0,0,0,0,0,0

;*ITEM 64

EM31
DH36
DT64
DF24

:SR2 DIDNOT LOCKUP CORRECT VIRTUAL ADDR.
:SR2 WAS EXPECTD TESTNO ERRORPC
:WASSR2,SREG4,TESTNO,SERRPC,0
:0,0,0,0

```

1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256 002246 005227
1257 002250 177777
1258 002252 001401
1259 002254 000000
1260
1261
1262
1263
1264 002256 012637 001356
1265 002262 012637 001360
1266 002266 010637 001354
1267 002272 104001
1268 002274 012737 177777 002250
1269 002302 013746 001360
1270 002306 013746 001356
1271 002312 000006
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283 002314 005227
1284 002316 177777
1285 002320 001401
1286 002322 000000
1287
1288
1289
1290
1291 002324 012637 001356
1292 002330 012637 001360
1293 002334 010637 001354
1294 002340 013737 177572 001362
1295 002346 013737 177576 001364
1296 002354 042737 160000 177572
1297 002362 104002
1298 002364 012737 177777 002316
1299 002372 013746 001360
1300 002376 013746 001356

```

```

.SBTTL ***** TRAP HANDLING ROUTINES *****

.SBTTL CPU TRAP HANDLER ROUTINE
:*****
:*
:* THIS SUBROUTINE WILL HANDLE ALL CPU TRAPS AND ABORTS THRU
:* "ERRVEC" (LOC. 004). IF THIS SUBROUTINE IS ENTERED BY A
:* SECOND TRAP BEFORE THE FIRST HAS BEEN SERVICED, A HALT IS
:* EXECUTED.
:*
:*****
TIMERR: INC (PC)+ ;MAKE FLAG ZERO IF FIRST TIME THRU
TIMFLG: .WORD -1 ;NEGATIVE ONE FOR "HAVE ENTERED" FLAG
        BEQ 1$ ;BRANCH IF FIRST TIME IN
        HALT ;STOP! - I'VE ENTERED THIS ROUTINE
           ;A SECOND TIME BEFORE I FINISHED
           ;REPORTING THE FIRST ERROR. THE
           ;SECOND ENTRY ADDRESS SHOULD BE ON
           ;THE KERNEL STACK.
1$: MOV (KSP)+,TRAPPC ;SAVE PC+2 AT TIME OF ABORT
    MOV (KSP)+,TRAPPS ;SAVE PS AT TIME OF ABORT
    MOV KSP,WASR6 ;SAVE STACK POINTER VALUE
    ERROR 1 ;UNEXPECTED TRAP OR ABORT TO LOC. 4
    MOV 8-1,TIMFLG ;MAKE FLAG NEGATIVE ONE FOR NEXT TIME
    MOV TRAPPS,-(KSP) ;PUT PC & PS OF TRAP ON STACK
    MOV TRAPPC,-(KSP)
    RTT ;RETURN FROM INTERRUPT OR ABORT

.SBTTL MEMORY MANAGEMENT TRAP HANDLER ROUTINE
:*****
:*
:* THIS SUBROUTINE WILL HANDLE ALL UNEXPECTED MEMORY MANAGEMENT
:* TRAPS AND ABORTS THRU "MMVEC" (LOC. 250). IF THIS SUBROUTINE IS
:* ENTERED BY A SECOND TRAP BEFORE THE FIRST HAS BEEN SERVICED, A
:* HALT IS EXECUTED.
:*
:*****
MGMERR: INC (PC)+ ;MAKE FLAG ZERO IF FIRST TIME THRU
MGMFLG: .WORD -1 ;NEGATIVE ONE FOR "HAVE ENTERED" FLAG
        BEQ 1$ ;BRANCH IF FIRST TIME IN
        HALT ;STOP! - I'VE ENTERED THIS ROUTINE
           ;A SECOND TIME BEFORE I FINISHED
           ;REPORTING THE FIRST ERROR. THE
           ;SECOND ENTRY ADDRESS SHOULD BE ON
           ;THE KERNEL STACK.
1$: MOV (KSP)+,TRAPPC ;SAVE PC+2 AT TIME OF ABORT
    MOV (KSP)+,TRAPPS ;SAVE PS AT TIME OF ABORT
    MOV KSP,WASR6 ;SAVE STACK POINTER VALUE
    MOV SR0,WASSR0 ;SAVE CONTENTS OF KT STATUS REG. 0
    MOV SR2,WASSR2 ;SAVE CONTENTS OF KT STATUS REG. 2
    BIC #160000,SR0 ;CLEAR ERROR BITS IN STATUS REG 0
    ERROR 2 ;UNEXPECTED TRAP OR ABORT TO LOC. 250
    MOV 8-1,MGMFLG ;MAKE FLAG NEGATIVE ONE FOR NEXT TIME
    MOV TRAPPS,-(KSP) ;PUT PC & PS OF TRAP ON STACK
    MOV TRAPPC,-(KSP)

```


M05

MD-11-DFKTH-A PDP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006) 27-JAN-77 08:51 PAGE 28
MEMORY MANAGEMENT TRAP HANDLER ROUTINE

1301 002402 000006
1302

RTT

;RETURN FROM INTERRUPT OR ABORT.

| | | | | | | | | |
|------|--------|--------|--------|--------|--------------|---------------------|--|-------|
| 1303 | | | | | .SBTTL | | | |
| 1304 | | | | | .SBTTL | ***** | STARTING POINT OF TEST | ***** |
| 1305 | | | | | .SBTTL | ***** | STARTING ADDRESS OF 200 | ***** |
| 1306 | | 020000 | | | .=20000 | | | |
| 1307 | | | | | | | | |
| 1308 | 020000 | | | | START: | | | |
| 1309 | | | | | .SBTTL | | INITIALIZE THE COMMON TAGS | |
| 1310 | | | | | ::CLEAR | | THE COMMON TAGS (\$CMTAG) AREA | |
| 1311 | 020000 | 012706 | 001100 | | MOV | \$CMTAG,R6 | ::FIRST LOCATION TO BE CLEARED | |
| 1312 | 020004 | 005026 | | | CLR | (R6)+ | ::CLEAR MEMORY LOCATION | |
| 1313 | 020006 | 022706 | 001140 | | CMP | \$SWR,R6 | ::DONE? | |
| 1314 | 020012 | 001374 | | | BNE | .-6 | ::LOOP BACK IF NO | |
| 1315 | 020014 | 012706 | 001100 | | MOV | \$STACK,SP | ::SETUP THE STACK POINTER | |
| 1316 | | | | | ::INITIALIZE | | A FEW VECTORS | |
| 1317 | 020020 | 012737 | 034470 | 000020 | MOV | \$SCOPE,\$IOTVEC | ::IOT VECTOR FOR SCOPE ROUTINE | |
| 1318 | 020026 | 012737 | 000340 | 000022 | MOV | \$340,\$IOTVEC+2 | ::LEVEL 7 | |
| 1319 | 020034 | 012737 | 034750 | 000030 | MOV | \$ERROR,\$EMTVEC | ::EMT VECTOR FOR ERROR ROUTINE | |
| 1320 | 020042 | 012737 | 000340 | 000032 | MOV | \$340,\$EMTVEC+2 | ::LEVEL 7 | |
| 1321 | 020050 | 012737 | 040732 | 000034 | MOV | \$TRAP,\$TRAPVEC | ::TRAP VECTOR FOR TRAP CALLS | |
| 1322 | 020056 | 012737 | 000340 | 000036 | MOV | \$340,\$TRAPVEC+2 | ::LEVEL 7 | |
| 1323 | 020064 | 012737 | 041020 | 000024 | MOV | \$PWARN,\$PWAVEC | ::POWER FAILURE VECTOR | |
| 1324 | 020072 | 012737 | 000340 | 000026 | MOV | \$340,\$PWAVEC+2 | ::LEVEL 7 | |
| 1325 | 020100 | 013737 | 034236 | 034230 | MOV | \$ENDCT,\$EOPCT | ::SETUP END-OF-PROGRAM COUNTER | |
| 1326 | 020106 | 005037 | 001212 | | CLR | \$TIMES | ::INITIALIZE NUMBER OF ITERATIONS | |
| 1327 | 020112 | 005037 | 001214 | | CLR | \$ESCAPE | ::CLEAR THE ESCAPE ON ERROR ADDRESS | |
| 1328 | 020116 | 112737 | 000001 | 001115 | MOVB | \$1,\$ERRMAX | ::ALLOW ONE ERROR PER TEST | |
| 1329 | | | | | ::INITIALIZE | | THE "1-BIT" TRAP VECTOR. THEN LOAD LOCATION "\$RTRN", IN | |
| 1330 | | | | | ::THE | | "END-OF-PASS" (\$EOP) ROUTINE, WITH A "RTI" OR "RTT" | |
| 1331 | 020124 | 012737 | 034454 | 000014 | MOV | \$RTRN,\$TBITVEC | ::SET "T" BIT VECTOR TO \$RTRN | |
| 1332 | 020132 | 012737 | 000340 | 000016 | MOV | \$340,\$TBITVEC+2 | ::LEVEL 7 | |
| 1333 | 020140 | 012737 | 000002 | 034454 | MOV | \$RTI,\$RTRN | ::SET \$RTRN TO A RTI | |
| 1334 | 020146 | 012737 | 020174 | 000010 | MOV | \$65,\$RESVEC | ::TRY TO DO A RTT | |
| 1335 | 020154 | 005046 | | | CLR | -(SP) | ::DUMMY PS | |
| 1336 | 020156 | 012746 | 020164 | | MOV | \$64\$,-(SP) | ::AND PC | |
| 1337 | 020162 | 000006 | | | RTT | | ::TRY THE RTT | |
| 1338 | 020164 | 012737 | 000006 | 034454 | 64\$: MOV | \$RTT,\$RTRN | ::RTT IS LEGAL--SET \$RTRN TO A RTT | |
| 1339 | 020172 | 000402 | | | BR | 66\$ | | |
| 1340 | 020174 | 062706 | 000010 | | 65\$: ADD | \$10,SP | ::RTT ILLEGAL--CLEAN OFF THE STACK | |
| 1341 | 020200 | 012737 | 000012 | 000010 | 66\$: MOV | \$RESVEC+2,\$RESVEC | ::RESTORE TRAP CATCHER | |
| 1342 | 020206 | 005037 | 034462 | | CLR | \$TBIT | ::CLEAR "T" BIT SWITCH | |
| 1343 | 020212 | 012737 | 020212 | 001106 | MOV | \$.,\$LPADR | ::INITIALIZE THE LOOP ADDRESS FOR SCOPE | |
| 1344 | 020220 | 012737 | 020220 | 001110 | MOV | \$.,\$LPERR | ::SETUP THE ERROR LOOP ADDRESS | |
| 1345 | | | | | ::SIZE FOR A | | HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS | |
| 1346 | | | | | ::EQUAL | | TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER. | |
| 1347 | 020226 | 013746 | 000004 | | MOV | \$ERRVEC,-(SP) | ::SAVE ERROR VECTOR | |
| 1348 | 020232 | 012737 | 020266 | 000004 | MOV | \$67\$,\$ERRVEC | ::SET UP ERROR VECTOR | |
| 1349 | 020240 | 012737 | 177570 | 001140 | MOV | \$DSWR,\$SWR | ::SETUP FOR A HARDWARE SWICH REGISTER | |
| 1350 | 020246 | 012737 | 177570 | 001142 | MOV | \$DISP,\$DISPLAY | ::AND A HARDWARE DISPLAY REGISTER | |
| 1351 | 020254 | 022777 | 177777 | 160656 | CMP | \$-1,\$SWR | ::TRY TO REFERENCE HARDWARE SWR | |
| 1352 | 020262 | 001012 | | | BNE | 69\$ | ::BRANCH IF NO TIMEOUT TRAP OCCURRED | |
| 1353 | | | | | | | AND THE HARDWARE SWR IS NOT = -1 | |
| 1354 | 020264 | 000403 | | | BR | 68\$ | ::BRANCH IF NO TIMEOUT | |
| 1355 | 020266 | 012716 | 020274 | | 67\$: MOV | \$68\$,(SP) | ::SET UP FOR TRAP RETURN | |
| 1356 | 020272 | 000002 | | | RTI | | | |
| 1357 | 020274 | 012737 | 000176 | 001140 | 68\$: MOV | \$SWREG,\$SWR | ::POINT TO SOFTWARE SWR | |
| 1358 | 020302 | 012737 | 000174 | 001142 | MOV | \$DISPREG,\$DISPLAY | | |

```

1359 020310 012637 000004      69$:  MOV      (SP)+, @ERRVEC ;; RESTORE ERROR VECTOR
1360
1361 020314 005037 001234      CLR      SPASS ;; CLEAR PASS COUNT
1362 020320 132737 000200 001247  BITB    @APTSIZE, SENVM ;; TEST USER SIZE UNDER APT
1363 020326 001403      BEQ      70$ ;; YES, USE NON-APT SWITCH
1364 020330 012737 001250 001140  MOV     @SSWREG, SWR ;; NO, USE APT SWITCH REGISTER
1365 020336
1366
1367                                70$:
                                .SBTTL  TYPE PROGRAM NAME
                                ;; TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1368 020336 005227 177777      INC     @-1 ;; FIRST TIME?
1369 020342 001054      BNE     71$ ;; BRANCH IF NO
1370 020344 022737 034410 000042  CMP     @SENDAD, @42 ;; ACT-11?
1371 020352 001450      BEQ     71$ ;; BRANCH IF YES
1372 020354 104401 020422      TYPE   72$ ;; TYPE ASCIZ STRING
1373                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1374 020360 005737 000042      TST    @42 ;; ARE WE RUNNING UNDER XXDP/ACT?
1375 020364 001012      BNE     73$ ;; BRANCH IF YES
1376 020366 123727 001246 000001  CMPB   SENV, @1 ;; ARE WE RUNNING UNDER APT?
1377 020374 001406      BEQ     73$ ;; BRANCH IF YES
1378 020376 023727 001140 000176  CMP     SWR, @SWREG ;; SOFTWARE SWITCH REG SELECTED?
1379 020404 001005      BNE     74$ ;; BRANCH IF NO
1380 020406 104407      GTSWR  ;; GET SOFT-SWR SETTINGS
1381 020410 000403      BR     74$
1382 020412 112737 000001 001134  73$:  MOVB   @1, @AUTOB ;; SET AUTO-MODE INDICATOR
1383 020420 74$:
1384 020420 000425      BR     71$ ;; GET OVER THE ASCIZ
1385                                ;; 72$:
1386 020474      .ASCIZ <CRLF>@MD-11-DFKTH-A 11/34 MEMORY MGMT. DIAG. @<CRLF>
1387                                71$:
1388                                LOOP:
1389 020474 012706 001100      MOV     @STACK, KSP ;; INITIALIZE THE STACK POINTER
1390 020500 012737 002246 000004  MOV     @TIMERR, ERRVEC ;; LOAD CPU SERVICE ROUTINE INTO TRAP VECTOR
1391 020506 012737 000340 000006  MOV     @340, ERRVEC+2 ;; SET NEW PS TO PRIORITY LEVEL 7-KERNEL
1392 020514 012737 002314 000250  MOV     @MGMR, MMVEC ;; LOAD MEMORY MANAGEMENT ROUTINE INTO VECTOR
1393 020522 012737 000340 000252  MOV     @340, MMVEC+2 ;; SET NEW PS TO PRIORITY LEVEL 7-KERNEL
1394 020530 012700 177777      MOV     @-1, RO ;; PUT -1 INTO RO TO INITIALIZE FLAGS
1395 020534 010037 002250      MOV     RO, TIMFLG ;; INITIALIZE CPU ERROR FLAG
1396 020540 010037 002316      MOV     RO, MGMFLG ;; INITIALIZE MEMORY MANAGEMENT ERROR FLAG
1397 020544 012737 000340 001366  MOV     @340, TBITPS ;; INITIALIZE LOG THAT HOLDS T-BIT PSW
1398 020552 005037 177572      CLR     SRC ;; BE SURE MEM. MGMT IS OFF TO START WITH
1399

```

```

1400
1401
1402
1403
1404
1405
1406
1407
1408
1409 020556 000004
1410 020560 012737 020570 001110
1411 020566 005000
1412 020570 005001
1413 020572 106400
1414 020574 106701
1415 020576 042701 177437
1416 020602 020001
1417 020604 001401
1418 020606 104003
1419
1420
1421
1422 020610 062700 000040
1423 020614 022700 000400
1424 020620 001363
1425 020622 012737 020560 001110
1426
1427
1428
1429
1430
1431
1432
1433 020630 000004
1434 020632 012737 020642 001110
1435 020640 005000
1436 020642 005037 177776
1437 020646 050037 177776
1438 020652 013701 177776
1439 020656 042701 007777
1440 020662 020001
1441 020664 001403
1442 020666 005037 177776
1443 020672 104004
1444
1445
1446
1447 020674 062700 010000
1448 020700 001360
1449 020702 012737 020632 001110
1450 020710 005037 177776
1451
1452
1453
1454
1455

```

```

*****
*TEST 1          PSW PRIORITY BIT TEST
*
*   THIS TEST READS AND WRITES THE PROCESSOR STATUS WORD <7:5> "PRIORITY BITS"
*   TO SEE THAT SOME OF THE BASIC "DATA PATH" LOGIC IS WORKING.
*
*****
TST1:  SCOPE
1$:    MOV      #2$,SLPERR      ;SET LOOP ON ERROR POINTER TO 2$
      CLR      RO              ;INITIALIZE RO WITH PRIORITY=0 DATA
2$:    CLR      R1              ;PREPARE R1 TO ACCEPT DATA READ
      MTPS    RO              ;WRITE PRIORITY BITS IN THE PSW
      MFPS    R1              ;READ BACK THE LOW BYTE OF PSW
      BIC     #177437,R1      ;MASK OFF EVERYTHING EXCEPT PRIORITY BITS
      CMP     RO,R1           ;WAS CORRECT PRIORITY SET IN THE PSW?
      BEQ     3$              ;BRANCH IF YES
      ERROR   3               ;PRIORITY BITS SET WRONG IN PSW
                                ;FOR TIGHTER SCOPE LOOP
                                ;REPLACE ERROR CALL WITH
                                ;"BR 2$" = 000770
3$:    ADD     #40,RO          ;CHANGE DATA TO NEXT PRIORITY
      CMP     #400,RO         ;HAVE PRIORITIES 0-7 ALL BEEN CHECKED?
      BNE     2$              ;BRANCH IF NO
      MOV     #1$,SLPERR     ;RESET LOOP ON ERROR POINTER TO 1$
*****
*TEST 2          PSW MODE BIT TEST
*
*   THIS TEST READS AND WRITES THE PROCESSOR STATUS WORD <15:12> "MODE BITS"
*   TO FURTHER CHECK THE BASIC CPU DATA PATHS
*
*****
TST2:  SCOPE
1$:    MOV      #2$,SLPERR     ;SET LOOP ON ERROR POINTER TO 2$
      CLR      RO              ;INITIALIZE RO WITH MODE BITS = 0000
2$:    CLR      PSW           ;INITIALIZE PSW
      BIS     RO,PSW          ;BIT SET THE PSW MODE BITS WITH RO
      MOV     PSW,R1          ;READ BACK THE CONTENTS OF THE PSW
      BIC     #007777,R1      ;MASK OFF EVERYTHING EXCEPT THE MODE BITS
      CMP     RO,R1           ;WERE THE MODE BITS SET CORRECTLY?
      BEQ     3$              ;BRANCH IF YES
      CLR     PSW            ;CLEAR PSW FOR ERROR REPORT
      ERROR   4               ;MODE BITS SET WRONG IN PSW
                                ;FOR TIGHTER SCOPE LOOP
                                ;REPLACE ERROR CALL WITH
                                ;"BR 2$" = 000763
3$:    ADD     #10000,RO       ;CHANGE MODE BIT DATA
      BNE     2$              ;BRANCH IF STILL MORE COMBINATIONS
      MOV     #1$,SLPERR     ;RESET LOOP ON ERROR POINTER TO 1$
      CLR     PSW            ;RESET PSW BEFORE LEAVING
*****
*TEST 3          BYTE ADDRESSING TEST FOR PSW
*
*   THIS TEST WRITES THE HIGH AND LOW BYTES OF THE PROCESSOR STATUS WORD

```

D06

MD-11-DFKTH-A POP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006) 27-JAN-77 08:51 PAGE 32
T3 BYTE ADDRESSING TEST FOR PSW

```

1456
1457
1458
1459
1460 020714 000004
1461 020716 012737 020724 001110
1462 020724 005037 177776
1463 020730 012700 000360
1464 020734 110037 177777
1465 020740 013701 177776
1466 020744 042701 007437
1467 020750 000300
1468 020752 020001
1469 020754 001403
1470 020756 005037 177776
1471 020762 104005
1472
1473
1474
1475 020764 012737 020772 001110
1476 020772 005037 177776
1477 020776 012700 000340
1478 021002 110037 177776
1479 021006 013701 177776
1480 021012 042701 007437
1481 021016 020001
1482 021020 001403
1483 021022 005037 177776
1484 021026 104005
1485
1486
1487
1488 021030 012737 020716 001110
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499 021036 000004
1500 021040 005037 177776
1501 021044 012706 001100
1502 021050 012737 140000 177776
1503 021056 012706 000700
1504 021062 005037 177776
1505 021066 022706 001100
1506 021072 001404
1507 021074 012700 001100
1508 021100 010601
1509 021102 104006
1510
1511

```

```

: * AND READS THEM BACK TO BE SURE THEY CAN BE WRITTEN INDEPENDENTLY.
: * THIS CHECKS THE PSW PORTION OF THE ADDRESS DETECTION LOGIC.
: *
: *****
TST3: SCOPE
1$: MOV #2$,SLPERR ;SET LOOP ON ERROR POINTER TO 2$
2$: CLR PSW ;CLEAR THE PSW
MOV #360,R0 ;PUT THE HIGH BYTE DATA INTO R0
MOVB R0,PSW+1 ;WRITE THE HIGH BYTE OF THE PSW
MOV PSW,R1 ;READ BACK THE ENTIRE PSW
BIC #007437,R1 ;MASK OFF THE T & CC BITS
SWAB R0 ;GET DATA WRITTEN IN HIGH BYTE OF R0
CMP R0,R1 ;WAS THE PSW WRITTEN TO CORRECTLY
BEQ 3$ ;BRANCH IF YES
CLR PSW ;CLEAR PSW FOR ERROR REPORT
ERROR 5 ;LOW BYTE EFFECTED BY WRITE TO HIGH BYTE OF PSW
;FOR TIGHTER SCOPE LOOP
;REPLACE ERROR CALL WITH
;"BR 2$" = 000760
3$: MOV #4$,SLPERR ;SET LOOP ON ERROR POINTER TO 4$
4$: CLR PSW ;CLEAR THE PSW
MOV #340,R0 ;PUT THE LOW BYTE DATA INTO R0
MOVB R0,PSW ;WRITE THE LOW BYTE OF THE PSW
MOV PSW,R1 ;READ BACK THE ENTIRE PSW
BIC #007437,R1 ;MASK OFF THE T&CC BITS
CMP R0,R1 ;WAS PSW WRITTEN TO CORRECTLY
BEQ 5$ ;BRANCH IF YES
CLR PSW ;CLEAR PSW FOR ERROR REPORT
ERROR 5 ;HIGH BYTE EFFECTED BY WRITE TO LOW BYTE OF PSW
;FOR TIGHTER SCOPE LOOP
;REPLACE ERROR CALL WITH
;"BR 2$" = 000736
5$: MOV #1$,SLPERR ;RESET LOOP ON ERROR POINTER TO 1$
: *****
: *TEST 4 TEST AND SETUP OF STACK POINTERS
: *
: * THIS TEST SETS THE USER AND KERNEL STACK POINTERS FOR THE
: * REST OF THE PROGRAM AND MAKES SURE THEY ARE INDEPENDENT OF
: * EACH OTHER. KERNEL R6 IS SET TO 1100, USER R6 IS SET TO 700, THEN
: * KERNEL R6 IS READ TO BE SURE ITS STILL 1100.
: *
: *****
TST4: SCOPE
CLR PSW ;GO TO KERNEL MODE
MOV #KERSTK,KSP ;SET KERNEL STACK POINTER TO 1100
MOV #140000,PSW ;GO TO USER MODE
MOV #USESTK,USP ;SET USER STACK POINTER TO 700
CLR PSW ;BACK TO KERNEL MODE
CMP #KERSTK,KSP ;IS KERNEL R6 STILL 1100?
BEQ TST5 ;BRANCH IF KERNEL R6 IS OKAY
MOV #KERSTK,R0 ;SAVE DATA WRITTEN FOR ERROR REPORT
MOV KSP,R1 ;SAVE DATA READ AFTER USER R6 WAS WRITTEN
ERROR 6 ;KERNEL R6 CHANGED BY WRITING USER R6
;FOR TIGHTER SCOPE LOOP
;REPLACE ERROR CALL WITH

```

E06

MD-11-DFKTH-A PDP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006) 27-JAN-77 08:51 PAGE 33
T4 TEST AND SETUP OF STACK POINTERS

;000756

1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567

```
*****
*
* THE NEXT FIVE (5) TESTS WILL TRY TO ADDRESS ALL OF THE
* MEMORY MANAGEMENT REGISTERS (SR0,SR1,SR2,KERNEL & USER PAR/PDR'S).
* EVERY TIME A REGISTER TIMES OUT ITS ADDRESS WILL BE REPORTED.
* AT THE END OF EACH TEST A SUMMARY OF THE ADDRESSES THAT TIMED
* OUT DURING THAT TEST IS GIVEN. THE RESULTS OF "AND-ING" AND "OR-ING"
* THEIR ADDRESSES IS GIVEN TO SHOW WHICH ADDRESS LINES MAY BE
* STUCK AT 0 OR 1. THE PAR/PDR ADDRESS AND KT MUX'S ARE THE
* THINGS BEING CHECKED.
*
* *****
```

```
*****
* TEST 5 SR0,SR1,SR2 TIMEOUT TEST
* *****
```

```
* THIS TEST ADDRESSES THE MEMORY MANAGEMENT STATUS REGISTERS
* 0,1, AND 2. STATUS REG. 1 IS NOT USED BUT SHOULD STILL
* RESPOND TO ITS UNIBUS ADDRESS. DATA WILL BE WRITTEN OR READ
* FROM THESE REGISTERS IN LATER TESTS, THIS TEST JUST CHECK
* FOR A RESPONSE.
* *****
```

```
*****
* TESTS: SCOPE
* *****
```

021104 000004
021106 012737 021150 001110
021114 012737 021206 000004
021122 012700 177572
021126 012701 000003
021132 012737 177777 001370
021140 005037 001372
021144 005037 001374
021150 005710
021152 062700 000002
021156 077104
021160 012737 021106 001110
021166 005737 001374
021172 001401
021174 104010
021176 012737 002246 000004
021204 000414
021206 062706 000004
021212 104007
021214 010002
021216 050237 001372
021222 005102
021224 040237 001370
021230 005237 001374

```
1$: MOV #2$, $LPERR ;SET LOOP ON ERROR POINTER TO 2$
MOV #5$, 2#4 ;SET TIMEOUT VECTOR TO 5$
MOV #SR0, R0 ;LOAD R0 WITH ADDRESS OF FIRST REG.
MOV #3, R1 ;LOAD R1 WITH THE LOOP COUNT
MOV #-1, ANDADR ;INITIALIZE "AND" OF ADDRS. LOC.
CLR ORADR ;INITIALIZE "OR" OF ADDRS. LOC.
CLR TONUM ;INITIALIZE "TIMEOUTS" COUNTER
2$: TST (R0) ;TRY ADDRESSING A STATUS REGISTER
;IF IT TIMES OUT GO TO 5$
3$: ADD #2, R0 ;PUT NEXT ADDRESS IN R0
SOB R1, 2$ ;LOOP BACK TO 2$ UNTIL ALL TESTED
MOV #1$, $LPERR ;RESET LOOP ON ERROR POINTER TO 1$
TST TONUM ;DID ANY OF THE STATUS REG.S TIMEOUT?
BEQ 4$ ;BRANCH IF NO
ERROR 10 ;SUMMARY OF STATUS REG. TIMEOUTS
4$: MOV #TIMERR, 2#4 ;RESTORE NORMAL CPU TRAP ROUTINE ADDRESS
BR TST6 ;GO TO NEXT TEST

5$: ADD #4, KSP ;CLEAN UP THE STACK
ERROR 7 ;ONE OF THE STATUS REGS. TIMED OUT
;FOR TIGHTER SCOPE LOOP
;REPLACE ERROR CALL WITH
;"BR 2$" = 000756
MOV R0, R2 ;LOAD THE ADDRESS THAT TIMED OUT INTO R2
BIS R2, ORADR ;"OR" IT WITH OTHER ADDRS. THAT TIMED OUT
COM R2 ;"AND" IT WITH OTHER ADDRS. THAT TIMED OUT
BIC R2, ANDADR
INC TONUM ;INCREMENT THE TIMEOUT COUNTER
```

1568 021234 000746 BR 3\$;BRANCH BACK TO TEST THE NEXT ADDR.

1569
1570 :*****

1571 :*TEST 6 KERNEL PAR'S TIMEOUT TEST

1572 :*
1573 :* THIS TEST ADDRESSES THE EIGHT (8) KERNEL PAGE ADDRESS
1574 :* REGISTERS (KIPAR0-KIPAR7) AND CHECKS THAT SOMETHING
1575 :* RESPONDS TO THEIR ADDRESSES.
1576 :*

1577 :*****

1578 021236 000004 †ST6: SCOPE

1579
1580 021240 012737 021302 001110 1\$: MOV #2\$,SLPERR ;SET LOOP ON ERROR POINTER TO 2\$

1581 021246 012737 021340 000004 MOV #5\$,#4 ;SET TIMEOUT VECTOR TO 5\$

1582 021254 012700 172340 MOV #KIPAR0,R0 ;LOAD R0 WITH ADDRESS OF FIRST REG.

1583 021260 012701 000010 MOV #10,R1 ;LOAD R1 WITH LOOP COUNT (8)

1584 021264 012737 177777 001370 MOV #-1,ANDADR ;INITIALIZE "AND" OF ADDR. LOC

1585 021272 005037 001372 CLR ORADR ;INITIALIZE "OR" OF ADDR. LOC.

1586 021276 005037 001374 CLR TONUM ;INITIALIZE "TIMEOUTS" COUNTER

1587 021302 005710 2\$: TST (R0) ;TRY ADDRESSING A KIPAR

1588
1589 021304 062700 000002 3\$: ADD #2,R0 ;IF IT TIMES OUT, WILL GO TO 5\$

1590 021310 077104 SOB R1,2\$;PUT NEXT KIPAR ADDRESS IN R0

1591 021312 012737 021240 001110 MOV #1\$,SLPERR ;LOOP BACK TO 2\$ UNTIL ALL TESTED

1592 021320 005737 001374 TST TONUM ;RESET LOOP ON ERROR POINTER TO 1\$

1593 021324 001401 BEQ 4\$;DID ANY OF THE KIPARS TIME OUT?

1594 021326 104010 ERROR 10 ;BRANCH IF NO

1595 021330 012737 002246 000004 4\$: MOV #TIMERR,#4 ;SUMMARY OF KIPAR TIMEOUTS

1596 021336 000414 BR TST7 ;RESTORE NORMAL CPU TRAP ROUTINE ADDRESS

1597
1598 021340 062706 000004 5\$: ADD #4,KSP ;GO TO NEXT TEST

1599 021344 104007 ERROR 7 ;CLEAN UP THE STACK

1600
1601 ;ONE OF THE KIPARS TIMED OUT

1602 ;FOR TIGHTER SCOPE LOOP

1603 ;REPLACE ERROR CALL WITH

1604 021346 010002 MOV R0,R2 ;"BR 2\$" = 000756

1605 021350 050237 001372 BIS R2,ORADR ;LOAD THE ADDRESS THAT TIMED OUT INTO R2

1606 021354 005102 COM R2 ;"OR" IT WITH OTHER ADDRS. THAT TIMED OUT

1607 021356 040237 001370 BIC R2,ANDADR ;"AND" IT WITH OTHER ADDRS. THAT TIMED OUT

1608 021362 005237 001374 INC TONUM ;INCREMENT THE TIMEOUT COUNTER

1609 021366 000746 BR 3\$;BRANCH BACK TO TEST THE NEXT KIPAR

1610 :*****

1611 :*TEST 7 KERNEL PDR'S TIMEOUT TEST

1612 :*
1613 :* THIS TEST ADDRESSES THE EIGHT (8) KERNEL PAGE DESCRIPTOR
1614 :* REGISTERS (KIPDR0-KIPDR7) AND CHECKS THAT SOMETHING
1615 :* RESPONDS TO THEIR ADDRESSES.
1616 :*

1617 :*****

1618 021370 000004 †ST7: SCOPE

1619
1620 021372 012737 021434 001110 1\$: MOV #2\$,SLPERR ;SET LOOP ON ERROR POINTER TO 2\$

1621 021400 012737 021472 000004 MOV #5\$,#4 ;SET TIMEOUT VECTOR TO 5\$

1622 021406 012700 172300 MOV #KIPDR0,R0 ;LOAD R0 WITH ADDRESS OF FIRST REG.

1623 021412 012701 000010 MOV #10,R1 ;LOAD R1 WITH LOOP COUNT (8)

```

1624 021416 012737 177777 001370      MOV    #-1,ANDADR      ;INITIALIZE "AND" OF ADDR. LOC
1625 021424 005037 001372              CLR    ORADR           ;INITIALIZE "OR" OF ADDR. LOC.
1626 021430 005037 001374              CLR    TONUM          ;INITIALIZE "TIMEOUTS" COUNTER
1627 021434 005710              2$:   TST    (RO)       ;TRY ADDRESSING A KIPDR
1628                                ;IF IT TIMES OUT, WILL GO TO 5$
1629 021436 062700 000002              3$:   ADD    #2,RO      ;PUT NEXT KIPDR ADDRESS IN RO
1630 021442 077104              SOB    R1,2$          ;LOOP BACK TO 2$ UNTIL ALL TESTED
1631 021444 012737 021372 001110      MOV    #1$,SLPERR     ;RESET LOOP ON ERROR POINTER TO 1$
1632 021452 005737 001374              TST    TONUM          ;DID ANY OF THE KIPDRS TIME OUT?
1633 021456 001401              BEQ    4$             ;BRANCH IF NO
1634 021460 104010              ERROR  10             ;SUMMARY OF KIPDR TIMEOUTS
1635 021462 012737 002246 000004      4$:   MOV    #TIMERR,#4  ;RESTORE NORMAL CPU TRAP ROUTINE ADDRESS
1636 021470 000414              BR     TST10          ;GO TO NEXT TEST
1637
1638 021472 062706 000004              5$:   ADD    #4,KSP     ;CLEAN UP THE STACK
1639 021476 104007              ERROR  7              ;ONE OF THE KIPDRS TIMED OUT
1640                                ;FOR TIGHTER SCOPE LOOP
1641                                ;REPLACE ERROR CALL WITH
1642                                ;"BR 2$" = 000756
1643 021500 010002              MOV    RO,R2          ;LOAD THE ADDRESS THAT TIMED OUT INTO R2
1644 021502 050237 001372              BIS    R2,ORADR       ;"OR" IT WITH OTHER ADDRS. THAT TIMED OUT
1645 021506 005102              COM    R2             ;"AND" IT WITH OTHER ADDRS. THAT TIMED OUT
1646 021510 040237 001370              BIC    R2,ANDADR      ;
1647 021514 005237 001374              INC    TONUM          ;INCREMENT THE TIMEOUT COUNTER
1648 021520 000746              BR     3$             ;BRANCH BACK TO TEST THE NEXT KIPDR
1649
1650                                ;*****
1651                                ;TEST 10      USER PAR'S TIMEOUT TEST
1652                                ;
1653                                ;THIS TEST ADDRESSES THE EIGHT (8) USER PAGE ADDRESS
1654                                ;REGISTERS (UIPAR0-UIPAR7) AND CHECKS THAT SOMETHING
1655                                ;RESPONDS TO THEIR ADDRESSES.
1656                                ;
1657                                ;*****
1658 021522 000004      TST10: SCOPE
1659
1660 021524 012737 021566 001110      1$:   MOV    #2$,SLPERR  ;SET LOOP ON ERROR POINTER TO 2$
1661 021532 012737 021624 000004      MOV    #5$,#4         ;SET TIMEOUT VECTOR TO 5$
1662 021540 012700 177640              MOV    #UIPAR0,RO     ;LOAD RO WITH ADDRESS OF FIRST REG.
1663 021544 012701 000010              MOV    #10,R1         ;LOAD R1 WITH LOOP COUNT (8)
1664 021550 012737 177777 001370      MOV    #-1,ANDADR     ;INITIALIZE "AND" OF ADDR. LOC
1665 021556 005037 001372              CLR    ORADR          ;INITIALIZE "OR" OF ADDR. LOC.
1666 021562 005037 001374              CLR    TONUM          ;INITIALIZE "TIMEOUTS" COUNTER
1667 021566 005710              2$:   TST    (RO)       ;TRY ADDRESSING A UIPAR
1668                                ;IF IT TIMES OUT, WILL GO TO 5$
1669 021570 062700 000002              3$:   ADD    #2,RO      ;PUT NEXT UIPAR ADDRESS IN RO
1670 021574 077104              SOB    R1,2$          ;LOOP BACK TO 2$ UNTIL ALL TESTED
1671 021576 012737 021524 001110      MOV    #1$,SLPERR     ;RESET LOOP ON ERROR POINTER TO 1$
1672 021604 005737 001374              TST    TONUM          ;DID ANY OF THE UIPARS TIME OUT?
1673 021610 001401              BEQ    4$             ;BRANCH IF NO
1674 021612 104010              ERROR  10             ;SUMMARY OF UIPAR TIMEOUTS
1675 021614 012737 002246 000004      4$:   MOV    #TIMERR,#4  ;RESTORE NORMAL CPU TRAP ROUTINE ADDRESS
1676 021622 000414              BR     TST11          ;GO TO NEXT TEST
1677
1678 021624 062706 000004              5$:   ADD    #4,KSP     ;CLEAN UP THE STACK
1679 021630 104007              ERROR  7              ;ONE OF THE UIPARS TIMED OUT
    
```


H06

MD-11-DFKTH-A PDP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006) 27-JAN-77 08:51 PAGE 36
T10 USER FAR'S TIMEOUT TEST

```

1680                                     ;FOR TIGHTER SCOPE LOOP
1681                                     ;REPLACE ERROR CALL WITH
1682                                     ;"BR 2$" = 000756
1683 021632 010002                       MOV    R0,R2
1684 021634 050237 001372                BIS    R2,ORADR
1685 021640 005102                       COM    R2
1686 021642 040237 001370                BIC    R2,ANDADR
1687 021646 005237 001374                INC    TONUM
1688 021652 000746                       BR     3$
1689                                     ;INCREMENT THE TIMEOUT COUNTER
1690                                     ;BRANCH BACK TO TEST THE NEXT UIPAR

```

*TEST 11 USER PDR'S TIMEOUT TEST

* THIS TEST ADDRESSES THE EIGHT (8) USER PAGE DESCRIPTOR
* REGISTERS (UIPDR0-UIPDR7) AND CHECKS THAT SOMETHING
* RESPONDS TO THEIR ADDRESSES.

†ST11: SCOPE

```

1698 021654 000004
1699
1700 021656 012737 021720 001110 1$: MOV    #2$,SLPERR ;SET LOOP ON ERROR POINTER TO 2$
1701 021664 012737 021756 000004     MOV    #5$,J#4 ;SET TIMEOUT VECTOR TO 5$
1702 021672 012700 177600             MOV    #UIPDR0,R0 ;LOAD R0 WITH ADDRESS OF FIRST REG.
1703 021676 012701 000010             MOV    #10,R1 ;LOAD R1 WITH LOOP COUNT (8)
1704 021702 012737 177777 001370     MOV    #-1,ANDADR ;INITIALIZE "AND" OF ADDR. LOC
1705 021710 005037 001372             CLR    ORADR ;INITIALIZE "OR" OF ADDR. LOC.
1706 021714 005037 001374             CLR    TONUM ;INITIALIZE "TIMEOUTS" COUNTER
1707 021720 005710                     2$: TST    (R0) ;TRY ADDRESSING A UIPDR
1708                                     ;IF IT TIMES OUT, WILL GO TO 5$
1709 021722 062700 000002 3$: ADD    #2,R0 ;PUT NEXT UIPDR ADDRESS IN R0
1710 021726 077104                     SOB    R1,2$ ;LOOP BACK TO 2$ UNTIL ALL TESTED
1711 021730 012737 021656 001110     MOV    #1$,SLPERR ;RESET LOOP ON ERROR POINTER TO 1$
1712 021736 005737 001374             TST    TONUM ;DID ANY OF THE UIPDRS TIME OUT?
1713 021742 001401                     BEQ    4$ ;BRANCH IF NO
1714 021744 104010                     ERROR 10 ;SUMMARY OF UIPDR TIMEOUTS
1715 021746 012737 002246 000004 4$: MOV    #TIMERR,J#4 ;RESTORE NORMAL CPU TRAP ROUTINE ADDRESS
1716 021754 000414                     BR     TST12 ;GO TO NEXT TEST

```

```

1717
1718 021756 062706 000004 5$: ADD    #4,KSP ;CLEAN UP THE STACK
1719 021762 104007                     ERROR 7 ;ONE OF THE UIPDRS TIMED OUT
1720                                     ;FOR TIGHTER SCOPE LOOP
1721                                     ;REPLACE ERROR CALL WITH
1722                                     ;"BR 2$" = 000756
1723 021764 010002                       MOV    R0,R2
1724 021766 050237 001372                BIS    R2,ORADR
1725 021772 005102                       COM    R2
1726 021774 040237 001370                BIC    R2,ANDADR
1727 022000 005237 001374                INC    TONUM
1728 022004 000746                       BR     3$
1729                                     ;INCREMENT THE TIMEOUT COUNTER
1730                                     ;BRANCH BACK TO TEST THE NEXT UIPDR

```

*TEST 12 SRO(15:13) BIT TEST & SR2 TEST

* THIS TEST CHECKS BITS <15:13> OF STATUS REGISTER 0 TO SEE
* THAT EACH CAN BE SET AND CLEARED AND THAT A "RESET" WILL
* CLEAR ALL OF THEM. A TEST OF THESE THREE ERROR BITS CHECKS

1731
1732
1733
1734
1735

```

1736
1737
1738
1739
1740
1741
1742 022006 000004
1743
1744 022010 012700 177572
1745 022014 012710 160000
1746 022020 000005
1747 022022 011001
1748 022024 001404
1749 022026 104011
1750
1751
1752
1753 022030 012737 022036 001110
1754 022036 013737 177576 001364 25:
1755 022044 012701 022036
1756 022050 020137 001364
1757 022054 001401
1758 022056 104041
1759
1760
1761
1762 022060 012737 022076 001110 35:
1763 022066 012701 100000
1764 022072 012703 000003
1765 022076 005010
1766 022100 050110
1767 022102 011002
1768 022104 020102
1769 022106 001401
1770 022110 104012
1771
1772
1773
1774 022112 012704 022100
1775 022116 013737 177576 001364 65:
1776 022124 020437 001364
1777
1778 022130 001401
1779 022132 104064
1780
1781
1782
1783 022134 006001
1784 022136 077321
1785 022140 005010
1786 022142 012737 022010 001110
1787
1788
1789
1790
1791

```

```

;* PART OF SRO, THE SRO MUX AND THE KTMUX. THE REST OF THE
;* BITS IN SRO WILL BE CHECKED LATER.
;* ALSO CHECK THAT SR2 IS TRACKING WITH MEM. MGMT.
;* OFF BUT LOCKS UP WHEN ANY OF SRO ERROR BITS SET.
*****
TST12: SCOPE
15: MOV #SRO, R0 ;LOAD ADDRESS OF SRO INTO R0
MOV #160000, (R0) ;SET BITS <15:13> IN SRO (ERROR BITS)
RESET ;ISSUE AND "INIT" SIGNAL
MOV (R0), R1 ;READ SRO INTO R1 TO SEE IF CLEAR
BEQ 25 ;BRANCH IF SRO<15:13> CLEARED BY "INIT"
ERROR 11 ;SRO<15:13> NOT CLEARED BY A "RESET"
;FOR TIGHTER SCOPE LOOP
;REPLACE ERROR CALL WITH
;"BR 15" = 000770
25: MOV #25, $LPERR ;SET LOOP ON ERROR POINTER TO 25
MOV SR2, WASSR2 ;READ CONTENTS OF SR2
MOV #25, R1 ;LOAD EXPECTED CONTENTS INTO R1
CMP R1, WASSR2 ;IS SR2 TRACKING?
BEQ 35 ;BRANCH IF YES
ERROR 41 ;SR2 NOT "TRACKING" VIRTUAL ADDRESSES
;FOR TIGHTER SCOPE LOOP
;REPLACE ERROR CALL WITH
;"BR 25" = 000767
35: MOV #45, $LPERR ;SET LOOP ON ERROR POINTER TO 45
MOV #BIT15, R1 ;PUT DATA TO BE WRITTEN IN R1
MOV #3, R3 ;SETUP R3 AS A LOOP COUNTER
CLR (R0) ;CLEAR SRO
55: BIS R1, (R0) ;SET ONE OF THE ERROR BITS IN SRO
MOV (R0), R2 ;READ SRO INTO R2
CMP R1, R2 ;DID RIGHT ERROR BIT GET SET?
BEQ 65 ;BRANCH IF YES
ERROR 12 ;BITS WERE SET WRONG IN SRO
;FOR TIGHTER SCOPE LOOP
;REPLACE ERROR CALL WITH
;"BR 45" = 000772
65: MOV #55, R4 ;LOAD EXPECTED CONTENTS OF SR2 IN R4
MOV SR2, WASSR2 ;READ SR2
CMP R4, WASSR2 ;DID SR2 LOCK UP WHEN ERROR
;BIT SET IN SR1?
BEQ 75 ;BRANCH IF YES
ERROR 64 ;SR2 DID NOT LOCK UP
;FOR TIGHTER SCOPE LOOP
;REPLACE ERROR CALL WITH
;"BR 45" = 000761
75: ROR R1 ;CHANGE DATA TO CHECK NEXT ERROR BIT
SOB R3, 45 ;LOOP BACK UNTIL <15:13> ALL TESTED
CLR (R0) ;CLEAR SRO BEFORE LEAVING
MOV #15, $LPERR ;RESET LOOP ON ERROR POINTER TO 15
*****
;TEST 13 SRO & PSW DUAL ADDRESSING TEST
;*
;* THIS TEST CHECKS MORE OF THE ADDRESS DETECTION LOGIC BY

```

JOB

MD-11-DFKTH-A PDP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006) 27-JAN-77 08:51 PAGE 38
T13 SRO & PSW DUAL ADDRESSING TEST

```

1792
1793
1794
1795
1796
1797
1798 022150 000004
1799
1800 022152 005037 177776
1801 022156 005037 177572
1802 022162 106427 000340
1803 022166 013700 177572
1804 022172 001401
1805 022174 104013
1806
1807
1808
1809 022176 005037 177572
1810 022202 005037 177776
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820 022206 000004
1821 022210 012700 177777
1822 022214 013700 177574
1823 022220 001401
1824 022222 104014
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839 022224 000004
1840
1841 022226 012700 172340
1842 022232 012703 000010
1843 022236 012737 022244 001110
1844 022244 005010
1845 022246 011001
1846 022250 001401
1847 022252 104011
    
```

```

:*      VERIFYING THAT STATUS REGISTER 0 IS NOT EFFECTED BY WRITING
:*      TO THE PSW AND THAT THE LOW BYTE OF STATUS REGISTER 0
:*      IS NOT EFFECTED BY WRITING TO ITS HIGH BYTE. THIS IS TO
:*      SEE IF ADJACENT OUTPUTS ARE SHORTED ON THE ADDRESS DET. LOGIC.
:*
:*****
TST13: SCOPE
1$:      CLR      PSW          ;CLEAR THE PSW
        CLR      SRO          ;CLEAR STATUS REGISTER 0
        MTPS     #340        ;SET PRIORITY 7 IN LOW BYTE OF PSW
        MOV      SRO,R0       ;READ STATUS REGISTER 0
        BEQ      2$,          ;BRANCH IF IT WAS STILL 0
        ERROR    13           ;SRO EFFECTED BY A WRITE TO THE PSW
                                ;FOR TIGHTER SCOPE LOOP
                                ;REPLACE ERROR CALL WITH
                                ;"BR 1$" = 000767
2$:      CLR      SRO          ;BE SURE SRO IS 0 BEFORE LEAVING
        CLR      PSW          ;BE SURE PSW IS 0 BEFORE LEAVING
:*****
TEST 14      TEST THAT SRI READS ALL ZEROS
:*
:*      THIS TESTS CHECKS THAT EVEN THOUGH STATUS REGISTER 1
:*      IS NON-EXISTENT, ITS ADDRESS SHOULD RESPOND WITH ALL ZEROS,
:*      THEREBY CHECK ANOTHER PORTION OF THE ADDRESS DETECTION LOGIC.
:*
:*****
TST14: SCOPE
        MOV      #-1,R0       ;FILL R0 WITH ALL ONES
        MOV      SRI,R0       ;READ SRI INTO R0
        BEQ      TST15        ;BRANCH IF SRI READS ALL ZEROS
        ERROR    14           ;SRI DID NOT READ ALL ZEROS
                                ;FOR TIGHTER SCOPE LOOP
                                ;REPLACE ERROR CALL WITH
                                ;000772
:*****
TEST 15      BIT TEST OF KERNEL & USER PAR'S
:*
:*      THE FOLLOWING TEST CHECKS THE BITS <11:00> OF BOTH THE KERNEL
:*      AND USER PAGE ADDRESS REGISTERS. A "0" IS ROTATED THRU
:*      THE REGISTERS FROM LEFT TO RIGHT. THIS CHECKS THE OPERATION
:*      OF THE PAR/PDR ADDRESS MUX, THE KT MUX, AND THE PAR
:*      OUTPUT DATA LINES.
:*
:*****
TST15: SCOPE
1$:      MOV      #KIPAR0,R0   ;LOAD ADDRESS OF FIRST PAR IN R0
2$:      MOV      #10,R3       ;SETUP R3 TO COUNT 8 PAR'S
        MOV      #3$,SLPERR    ;SET LOOP ON ERROR POINTER TO 3$
3$:      CLR      (R0)         ;CLEAR THE PAR
        MOV      (R0),R1       ;READ THE PAR INTO R1
        BEQ      4$,          ;BRANCH IF PAR CLEARED OK
        ERROR    11           ;PAR WOULD NOT CLEAR
    
```



```

1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886 022346 000004
1887
1888 022350 012700 172300
1889 022354 012703 000010
1890 022360 012737 022366 001110
1891 022366 005010
1892 022370 011001
1893 022372 001401
1894 022374 104011
1895
1896
1897
1898 022376 012704 077777
1899 022402 012737 022410 001110
1900 022410 005010
1901 022412 010401
1902 022414 042701 100361
1903 022420 050110
1904 022422 011002
1905 022424 020102
1906 022426 001401
1907 022430 104012
1908
1909
1910
1911 022432 000261
1912 022434 006004
1913 022436 103764
1914 022440 062700 000002
1915 022444 077330
1916 022446 022700 177620
1917 022452 103003
1918 022454 012700 177600
1919 022460 000735
1920 022462 012737 022350 001110
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930

```

```

*****
:TEST 16 BIT TEST OF KERNEL & USER PDR'S
:
: THE FOLLOWING TEST CHECKS THE BITS <14:8> AND <3:1> OF BOTH THE
: KERNEL AND USER PAGE DESCRIPTOR REGISTERS. A "0" IS ROTATED
: THRU THE REGISTERS FROM LEFT TO RIGHT. SOME TEST PATTERNS WILL
: BE LOADED MORE THAN ONCE DUE TO THE UNUSED BITS IN THE PDR'S.
: THE PAR/PDR ADDRESS MUX, KTMUX, AND PDR OUTPUT DATA LINES
: ARE BEING CHECKED.
*****
TST16: SCOPE
1S: MOV #KIPDR0,R0 ;LOAD ADDRESS OF FIRST PDR IN R0
2S: MOV #10,R3 ;SETUP R3 TO COUNT 8 PDR'S
3S: MOV #3S,SLPERR ;SET LOOP ON ERROR POINTER TO 3S
: CLR (R0) ;CLEAR THE PDR
: MOV (R0),R1 ;READ THE PDR INTO R1
: BEQ 4S ;BRANCH IF PDR CLEARED OK
: ERROR 11 ;PDR WOULD NOT CLEAR
: ;FOR TIGHTER SCOPE LOOP
: ;REPLACE ERROR CALL WITH
: ;"BR 3S" = 000774
4S: MOV #077777,R4 ;LOAD "WALKING 0" TEST PATTERN IN R4
5S: MOV #5S,SLPERR ;SET LOOP ON ERROR POINTER TO 5S
: CLR (R0) ;CLEAR THE PDR BEFORE LOADING DATA
: MOV R4,R1 ;LOAD DATA INTO R1
: BIC #100361,R1 ;MASK UNUSED BITS OUT OF THE DATA
: BIS R1,(R0) ;BIT SET THE TEST PATTERN INTO THE PDR
: MOV (R0),R2 ;READ THE PDR INTO R2
: CMP R1,R2 ;DOES DATA WRITTEN=DATA READ?
: BEQ 6S ;BRANCH IF YES
: ERROR 12 ;PDR BITS DID NOT SET CORRECTLY
: ;FOR TIGHTER SCOPE LOOP
: ;REPLACE ERROR CALL WITH
: ;"BR 5S" = 000767
6S: SEC ;SET THE C-BIT FOR THE ROTATE INST.
: ROR R4 ;ROTATE THE TEST PATTERN IN R4
: BCS 5S ;BRANCH BACK IF MORE BITS TO TEST
: ADD #2,R0 ;GET NEXT PDR ADDRESS IN R0
: SOB R3,3S ;BRANCH BACK UNTIL ALL PDR'S TESTED
: CMP #UIPDR7+2,R0 ;HAVE USER PDR'S BEEN TESTED?
: BHIS 7S ;BRANCH IF YES
: MOV #UIPDR0,R0 ;LOAD FIRST USER PDR ADDR. IN R0
: BR 2S ;BRANCH BACK TO TEST USER PDR'S
7S: MOV #1S,SLPERR ;RESET LOOP ON ERROR POINTER TO 1S
: ;LEAVE TEST WITH ALL WRITEABLE BITS IN
: ;ALL PDR'S = 1

```

```

*****
:TEST 17 TEST FOR DUAL BYTE ADDRESSING OF KERNEL & USER PAR'S
:
: THE FOLLOWING TEST WRITES TO BOTH BYTES OF THE KERNEL & USER
: PAR'S SEPERATELY TO SEE THAT WRITING TO ONE DOES NOT EFFECT
: THE OTHER. THIS FURTHER VERIFIES THE OPERATION OF THE PAR/PDR
: ADDR. MUX AND THE ADDR. DETECTION LOGIC.

```

M06

MD-11-DFKTH-A PDP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006) 27-JAN-77 08:51 PAGE 41
T17 TEST FOR DUAL BYTE ADDRESSING OF KERNEL & USER PAR'S

1931
1932
1933 022470 000004

!*
:*****
†ST17: SCOPE

N06

MD-11-DFKTH-A PDP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006) 27-JAN-77 08:51 PAGE 42
T17 TEST FOR DUAL BYTE ADDRESSING OF KERNEL & USER PAR'S

| | | | | | | | | | |
|------|--------|--------|--------|--------|------|-----|-------------|--|------------------------------------|
| 1934 | | | | | | | | | |
| 1935 | 022472 | 012700 | 172340 | | 1\$: | MOV | #KIPAR0,R0 | | ;LOAD ADDRESS OF FIRST PAR INTO R0 |
| 1936 | 022476 | 012737 | 022510 | 001110 | 2\$: | MOV | #3\$,SLPERR | | ;SET LOOP ON ERROR POINTER TO 3\$ |
| 1937 | 022504 | 012703 | 000010 | | | MOV | #10,R3 | | ;LOAD LOOP COUNTER TO DO 8 PAR'S |
| 1938 | 022510 | 012701 | 177777 | | 3\$: | MOV | #-1,R1 | | ;LOAD TEST PATTERN INTO R1 |
| 1939 | 022514 | 005010 | | | | CLR | (R0) | | ;CLEAR THE PAR |

1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016

022620 000004

022622 012700 172300
022626 012737 022640 001110
022634 012703 000010
022640 012701 177777
022644 005010
022646 110110
022650 011002
022652 042701 177761
022656 020102
022660 001401
022662 104015

022664 012737 022672 001110
022672 005010
022674 012701 177777
022700 110160 000001
022704 011002
022706 042701 100377
022712 020102
022714 001401
022716 104015

022720 062700 000002
022724 077333
022726 022700 177620
022732 103003
022734 012700 177600
022740 000732
022742 012737 022622 001110

```
*****  
:TEST 20 TEST FOR DUAL BYTE ADDRESSING OF KERNEL & USER PDR'S  
:  
:* THE FOLLOWING TEST WRITES TO BOTH BYTES OF THE KERNEL & USER  
:  
:* PDR'S SEPERATELY TO SEE THAT WRITING TO ONE DOES NOT EFFECT  
:* THE OTHER. THIS FURTHER VERIFIES THE OPERATION OF THE PAR/PDR  
:* ADDR. MUX AND THE ADDR. DETECTION LOGIC.  
:  
:*****  
TST20: SCOPE  
  
1$: MOV #KIPDR,RO ;LOAD ADDRESS OF FIRST PDR INTO RO  
2$: MOV #3$,SLPERR ;SET LOOP ON ERROR POINTER TO 3$  
MOV #10,R3 ;LOAD LOOP COUNTER TO DO 8 PDR'S  
3$: MOV #-1,R1 ;LOAD TEST PATTERN INTO R1  
CLR (RO) ;CLEAR THE PDR  
MOV R1,(RO) ;WRITE 1'S TO THE LOW BYTE OF THE PDR  
MOV (RO),R2 ;READ THE ENTIRE PDR INTO R2  
BIC #177761,R1 ;MASK HIGH BYTE & UNUSED BITS OUT OF DATA  
CMP R1,R2 ;WAS ONLY THE LOW BYTE WRITTEN TO?  
BEQ 4$ ;BRANCH IF YES  
ERROR 15 ;HIGH BYTE EFFECTED BY WRITING LOW BYTE IN PDR  
FOR TIGHTER SCOPE LOOP  
REPLACE ERROR CALL WITH  
"BR 3$" = 000766  
4$: MOV #5$,SLPERR ;SET LOOP ON ERROR POINTER TO 5$  
5$: CLR (RO) ;CLEAR THE PDR  
MOV #-1,R1 ;LOAD TEST PATTERN INTO R1  
MOV R1,1(RO) ;WRITE 1'S TO THE HIGH BYTE OF THE PDR  
MOV (RO),R2 ;READ THE ENTIRE PDR INTO R2  
BIC #100377,R1 ;MASK LOW BYTE & UNUSED BITS OUT OF DATA  
CMP R1,R2 ;WAS ONLY THE HIGH BYTE WRITTEN TO?  
BEQ 6$ ;BRANCH IF YES  
ERROR 15 ;LOW BYTE EFFECTED BY WRITING HIGH BYTE IN PDR  
FOR TIGHTER SCOPE LOOP  
REPLACE ERROR CALL WITH  
"BR 5$" = 000765  
6$: ADD #2,RO ;PUT ADDRESS OF NEXT PDR IN RO  
SOB R3,3$ ;BRANCH BACK UNTIL 8 PDR'S TESTED  
CMP #UIPDR7+2,RO ;HAVE USER PDR'S BEEN TESTED?  
BHS 7$ ;BRANCH IF YES  
MOV #UIPDR,RO ;LOAD ADDRESS OF FIRST USER PDR IN RO  
BR 2$ ;BRANCH BACK TO TEST USER PDR'S  
7$: MOV #1$,SLPERR ;RESET LOOP ON ERROR POINTER TO 1$
```

2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069

```
*****
*TEST 21 PAR-PDR DUAL ADDRESSING TEST
*
* THE FOLLOWING TEST SETS ALL OF THE WRITEABLE BITS TO 1
* IN THE SIXTEEN (16) PAR'S AND PDR'S USING THE "SETREG"
* SUBROUTINE AND THEN CLEARS JUST ONE OF THEM. THE "CMPREG"
* SUBROUTINE IS USED TO READ ALL OF THE PAR'S AND PDR'S TO SEE
* THAT ONLY ONE REGISTER WAS CLEARED IN RESPONSE TO THAT ONE
* PAR OR PDR ADDRESS. THE "CMPREG" SUBROUTINE REPORTS THE
* ADDRESS OF ANY REGISTER WHOSE BITS DID NOT REMAIN SET WHEN
* ANOTHER REGISTER WAS CLEARED.
```

```
*
* THE PAR AND PDR CHIPS, PAR/PDR ADDR. MUX, AND ADDR. DETECTION
* LOGIC ARE CHECKED.
```

```
*****
*TEST21: SCOPE
```

| | | | | | | | | | |
|--------|--------|---------------|--|--|--|--|--|--|--|
| 022750 | 000004 | | | | | | | | |
| 18: | MOV | #2\$, \$LPERR | | | | | | | SET LOOP ON ERROR POINTER 2\$ |
| | MOV | #10, R3 | | | | | | | LOAD LOOP COUNTER WITH AN 8 |
| | MOV | #KIPDR0, RO | | | | | | | LOAD ADDRESS OF FIRST KERNEL PDR AND RO |
| 25: | MOV | #KERSTK, KSP | | | | | | | SETUP STACK POINTER |
| | JSR | PC, SETREG | | | | | | | SET ALL BITS IN ALL PAR'S IN PDR'S |
| | CLR | (RO) | | | | | | | CLEAR ONE OF THE KERNEL PDR'S |
| | JSR | PC, CMPREG | | | | | | | SEE IF OTHER PAR/PDR'S WERE EFFECTED |
| | ADD | #2, RO | | | | | | | FORM ADDRESS OF NEXT KERNEL PDR TO CLEAR |
| | SOB | R3, 2\$ | | | | | | | LOOP TO 2\$ UNTIL ALL KERNEL PDR'S CHECKED |
| 001110 | MOV | #3\$, \$LPERR | | | | | | | SET LOOP ON ERROR POINTER TO 3\$ |
| | MOV | #10, R3 | | | | | | | LOAD LOOP COUNTER WITH AN 8 |
| | MOV | #UIPDR0, RO | | | | | | | LOAD ADDRESS OF FIRST KERNEL PAR IN RO |
| 35: | MOV | #KERSTK, KSP | | | | | | | SETUP STACK POINTER |
| | JSR | PC, SETREG | | | | | | | SET ALL BITS IN ALL PAR'S AND PDR'S |
| | CLR | (RO) | | | | | | | CLEAR ONE OF THE KERNEL PAR'S |
| | JSR | PC, CMPREG | | | | | | | SEE IF OTHER PAR/PDR'S WERE EFFECTED |
| | ADD | #2, RO | | | | | | | FORM ADDRESS OF NEXT KERNEL PAR TO CLEAR |
| | SOB | R3, 3\$ | | | | | | | LOOP TO 3\$ UNTIL ALL KERNEL PAR'S CHECKED |
| 001110 | MOV | #4\$, \$LPERR | | | | | | | SET LOOP ON ERROR POINTER TO 4\$ |
| | MOV | #10, R3 | | | | | | | LOAD LOOP COUNTER WITH AN 8 |
| | MOV | #UIPDR0, RO | | | | | | | LOAD ADDRESS OF FIRST USER PDR IN RO |
| 45: | MOV | #KERSTK, KSP | | | | | | | SETUP STACK POINTER |
| | JSR | PC, SETREG | | | | | | | SET ALL BITS IN ALL PAR'S AND PDR'S |
| | CLR | (RO) | | | | | | | CLEAR ONE OF THE USER PDR'S |
| | JSR | PC, CMPREG | | | | | | | SEE IF OTHER PAR/PDR'S WERE EFFECTED |
| | ADD | #2, RO | | | | | | | FORM ADDRESS OF NEXT USER PDR TO CLEAR |
| | SOB | R3, 4\$ | | | | | | | LOOP TO 4\$ UNTIL ALL USER PDR'S CHECKED |
| 001110 | MOV | #5\$, \$LPERR | | | | | | | SET LOOP ON ERROR POINTER TO 5\$ |
| | MOV | #10, R3 | | | | | | | LOAD LOOP COUNTER WITH AN 8 |
| | MOV | #UIPAR0, RO | | | | | | | LOAD ADDRESS OF FIRST USER PAR IN RO |
| 55: | MOV | #KERSTK, KSP | | | | | | | SETUP STACK POINTER |
| | JSR | PC, SETREG | | | | | | | SET ALL BITS IN ALL PAR'S AND PDR'S |
| | CLR | (RO) | | | | | | | CLEAR ONE OF THE USER PAR'S |
| | JSR | PC, CMPREG | | | | | | | SEE IF OTHER PAR/PDR'S WERE EFFECTED |

2070 023154 062700 000002
2071 023160 077312
2072 023162 012737 022752 001110

ADD #2,RO ;FORM ADDRESS OF NEXT USER PAR TO CLEAR
SOB R3,5\$;LOOP TO 5\$ UNTIL ALL USER PAR'S CHECKED
MOV #1\$,SLPERR ;SET LOOP ON ERROR POINTER TO 1\$

2073
2074
2075
2076
2077
2078
2079
2080
2081
2082

*TEST 22 TEST THAT PAR-PDR'S NOT AFFECTED BY RESET

* THIS TEST CHECKS TO SEE THAT THE KERNEL OR USER PAR/PDR'S ARE
* NOT AFFECTED BY THE EXECUTION OF A "RESET" INSTRUCTION. THE
* "SETREG" SUBROUTINE IS USED TO SET ALL WRITEABLE BITS TO A "1" IN
* THE PAR/PDR'S. THEN THEY ARE READ TO SEE THAT THEY REMAINED
* UNCHANGED

2083
2084 023170 000004

TST22: SCOPE

2085
2086
2087 023172 004737 035520
2088 023176 000005
2089 023200 012700 172300
2090 023204 012704 000010
2091 023210 011001
2092 023212 022701 077416
2093 023216 001401
2094 023220 104055
2095
2096
2097

1\$: JSR PC,SETREG ;SET ALL BITS IN ALL PAR'S AND PDR'S
RESET ;ISSUE AN "INIT" BY EXECUTING A RESET
MOV #KIPDR,RO ;LOAD ADDRESS OF FIRST KERNEL PDR IN RO
MOV #10,R4 ;LOAD LOOP COUNTER WITH AN 8
2\$: MOV (RO),R1 ;READ A KERNEL PDR INTO R1
CMP #77416,R1 ;ARE ALL THE BITS STILL SET?
BEQ 3\$;BRANCH IF YES
ERROR 55 ;KERNEL PDR AFFECTED BY A RESET
FOR TIGHTER SCOPE LOOP
REPLACE ERROR CALL WITH
"BR 2\$" = 000773

2098 023222 062700 000002
2099 023226 077410
2100 023230 012700 172340
2101 023234 012704 000010
2102 023240 011001
2103 023242 022701 007777
2104 023246 001401
2105 023250 104055
2106
2107
2108

3\$: ADD #2,RO ;FORM ADDRESS OF NEXT KERNEL PDR
SOB R4,2\$;LOOP TO 2\$ UNTIL ALL KERNEL PDR'S CHECKED
MOV #KIPAR,RO ;LOAD ADDRESS OF FIRST KERNEL PAR IN RO
MOV #10,R4 ;LOAD LOOP COUNTER WITH AN 8
4\$: MOV (RO),R1 ;READ A KERNEL PAR INTO R1
CMP #7777,R1 ;ARE ALL THE BITS STILL SET?
BEQ 5\$;BRANCH IF YES
ERROR 55 ;KERNEL PAR AFFECTED BY A RESET
FOR TIGHTER SCOPE LOOP
REPLACE ERROR CALL WITH
"BR 4\$" = 000773

2109 023252 062700 000002
2110 023256 077410
2111 023260 012700 177600
2112 023264 012704 000010
2113 023270 011001
2114 023272 022701 077416
2115 023276 001401
2116 023300 104055
2117
2118
2119

5\$: ADD #2,RO ;FORM ADDRESS OF NEXT KERNEL PAR
SOB R4,4\$;LOOP TO 4\$ UNTIL ALL KERNEL PAR'S CHECKED
MOV #UIPDR,RO ;LOAD ADDRESS OF FIRST USER PDR IN RO
MOV #10,R4 ;LOAD LOOP COUNTER WITH AN 8
6\$: MOV (RO),R1 ;READ A USER PDR INTO R1
CMP #77416,R1 ;ARE ALL THE BITS STILL SET?
BEQ 7\$;BRANCH IF YES
ERROR 55 ;USER PDR AFFECTED BY A RESET
FOR TIGHTER SCOPE LOOP
REPLACE ERROR CALL WITH
"BR 6\$" = 000773

2120 023302 062700 000002
2121 023306 077410
2122
2123 023310 012700 177640
2124 023314 012704 000010
2125 023320 011001

7\$: ADD #2,RO ;FORM ADDRESS OF NEXT USER PDR
SOB R4,6\$;LOOP TO 6\$ UNTIL ALL USER PDR'S CHECKED
MOV #UIPAR,RO ;LOAD ADDRESS OF FIRST USER PAR IN RO
MOV #10,R4 ;LOAD LOOP COUNTER WITH AN 8
8\$: MOV (RO),R1 ;READ A USER PAR INTO R1

F07

MD-11-DFKTH-A POP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006) 27-JAN-77 08:51 PAGE 47
T22 TEST THAT PAR-PDR'S NOT AFFECTED BY RESET

| | | | | | | | |
|------|--------|--------|--------|--------|--------|----------------|--|
| 2126 | 023322 | 022701 | 007777 | | CMP | #7777,R1 | ;ARE ALL THE BITS STILL SET? |
| 2127 | 023326 | 001401 | | | BEQ | 9\$ | ;BRANCH IF YES |
| 2128 | 023330 | 104055 | | | ERROR | 55 | ;USER PAR AFFECTED BY A RESET |
| 2129 | | | | | | | ;FOR TIGHTER SCOPE LOOP |
| 2130 | | | | | | | ;REPLACE ERROR CALL WITH |
| 2131 | | | | | | | ; "BR 8\$" = 000773 |
| 2132 | 023332 | 062700 | 000002 | 9\$: | ADD | #2,R0 | ;FORM ADDRESS OF NEXT USER PAR |
| 2133 | 023336 | 077410 | | | SOB | R4,8\$ | ;LOOP TO 8\$ UNTIL ALL USER PAR'S CHECKED |
| 2134 | | | | | | | |
| 2135 | | | | | | | |
| 2136 | | | | | | | |
| 2137 | | | | | | | |
| 2138 | | | | | | | |
| 2139 | | | | | | | |
| 2140 | | | | | | | |
| 2141 | | | | | | | |
| 2142 | | | | | | | |
| 2143 | | | | | | | |
| 2144 | | | | | | | |
| 2145 | | | | | | | |
| 2146 | | | | | | | |
| 2147 | | | | | | | |
| 2148 | | | | | | | |
| 2149 | | | | | | | |
| 2150 | | | | | | | |
| 2151 | | | | | | | |
| 2152 | | | | | | | |
| 2153 | | | | | | | |
| 2154 | | | | | | | |
| 2155 | | | | | | | |
| 2156 | | | | | | | |
| 2157 | | | | | | | |
| 2158 | | | | | | | |
| 2159 | 023340 | 000004 | | | TST23: | SCOPE | |
| 2160 | 023342 | 004737 | 035432 | 1\$: | JSR | PC,TOFF | ;TURN T-BIT TRAPPING OFF FOR THIS TEST |
| 2161 | 023346 | 012700 | 172300 | | MOV | #KIPDR0,R0 | ;LOAD ADDRESS OF FIRST KERNEL PDR INTO R0 |
| 2162 | 023352 | 012704 | 000010 | | MOV | #10,R4 | ;LOAD LOOP COUNTER WITH AN 8 |
| 2163 | 023356 | 005020 | | 2\$: | CLR | (R0)+ | ;CLEAR PDR - MAPPING PAGE NON-RES, 0 BLKS. |
| 2164 | 023360 | 077402 | | | SOB | R4,2\$ | ;LOOP TO 2\$ UNTIL ALL KERNEL PDR'S CLEARED |
| 2165 | 023362 | 012737 | 000252 | 000250 | MOV | #MMVEC+2,MMVEC | ;LOAD TRAP CATCHER INTO MEM MGMT. VECTOR |
| 2166 | 023370 | 005037 | 000252 | | CLR | MMVEC+2 | |
| 2167 | | | | | | | |
| 2168 | | | | | | | |
| 2169 | 023374 | 012737 | 001006 | 172300 | MOV | #1006,KIPDR0 | ;A HALT WILL OCCUR IF RESET FAILS |
| 2170 | 023402 | 012737 | 023410 | 001110 | MOV | #3\$,SLPERR | ;TO DISABLE DEST.-ONLY RELOCATION |
| 2171 | 023410 | 012737 | 000400 | 177572 | MOV | #BIT8,SRO | ;MAP KERNEL PG 0 R/W, 3 BLOCKS LONG. |
| 2172 | 023416 | 000005 | | | MOV | #BIT8,SRO | ;SET LOOP ON ERROR POINTER TO 3\$ |
| 2173 | 023420 | 032737 | 000400 | 177572 | 3\$: | RESET | ;TURN ON DEST-ONLY-RELOCATION |
| 2174 | 023426 | 001403 | | | BIT | #BIT8,SRO | ;SHOULD CLEAR MAINT. BIT - WILL ABORT IF RELOCATED |
| 2175 | 023430 | 005037 | 177572 | | BEQ | 4\$ | ;WAS MAINT. BIT (BIT 8) OF SRO CLEARED? |
| 2176 | 023434 | 104061 | | | CLR | SRO | ;BRANCH IF YES |
| 2177 | | | | | ERROR | 61 | ;CLEAR SRO SO ERROR CAN BE REPORTED |
| 2178 | | | | | | | ;MAINT. MODE NOT DISABLED BY A RESET |
| 2179 | | | | | | | ;FOR A TIGHTER SCOPE LOOP |
| 2180 | 023436 | 012706 | 001100 | 4\$: | MOV | #KERSTK,KSP | ;REPLACE ERROR CALL WITH |
| 2181 | 023442 | 005037 | 177572 | | CLR | SRO | ; "BR 3\$" = 000765 |
| | | | | | | | ;RESTORE STACK POINTER |
| | | | | | | | ;BE SURE SRO IS CLEAR |

```

*****
:TEST 23      INSTRUCTION FETCH NOT RELOCATED IN MAINT. MODE
:
:THIS TEST CHECKS TO SEE THAT WHEN MEMORY MANAGEMENT IS IN
:MAINTENANCE MODE (DESTINATION-ONLY-RELOCATION), AN INSTRUCTION
:FETCH IS NOT RELOCATED AND A RESET CLEARS THE MAINTENANCE BIT
:(BIT 08) IN SRO.  IF THE "FETCH" IS RELOCATED, A PG.LENGTH ABORT
:SHOULD OCCUR, CAUSING A HALT SINCE TRAP CATCHER IS PLACED IN VECTOR 250
:
:NOTE:  A HALT MAY OCCUR IF MAINT. MODE NOT DISABLED BY RESET
:
*****

```

G07

MD-11-DFKTH-A PDP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006) 27-JAN-77 08:51 PAGE 48
T23 INSTRUCTION FETCH NOT RELOCATED IN MAINT. MODE

2182 023446 012737 002314 000250
2183 023454 012737 000340 000252
2184 023462 012737 023342 001110
2185 023470 004737 035466

MOV #MGMERR,MMVEC ;RESTORE MEM. MGMT. TRAP VECTOR
MOV #340,MMVEC+2 ;RESTORE MEM. MGMT VECTOR+2
MOV #1\$,SLPERR ;RESET LOOP ON ERROR POINTER TO 1\$
JSR PC,TON ;TURN T-BIT TRAPPING BACK ON

2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
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
2233
2234
2235
2236
2237

*TEST 24 TEST THAT SOURCE NOT RELOCATED IN MAINTENANCE MODE
*
* THIS TEST CHECKS TO SEE THAT WHEN MEMORY MANAGEMENT IS IN
* MAINTENANCE MODE, THE SOURCE IS NOT RELOCATED. ONLY THE
* DESTINATION IS RELOCATED. KERNEL PAR'S 3 & 4 ARE MAPPED TO
* PHYSICAL ADDRESS 60000-77776. PDR4 IS SET TO ALLOW FULL READ/WRITE
* BUT PDR3 IS CLEAR ALLOWING TO ACCESS. VIRTUAL ADDRESSES REFERENCING
* PAR/PDR'S 4 & 3 ARE USED IN AS DESTINATION AND SOURCE RESPECTIVELY.
* IF THE SOURCE IS RELOCATED IN MAINTENANCE A MEM. MGMT. TRAP WILL
* OCCUR AND THE ERROR WILL BE REPORTED. KERNEL PG. 7 IS MAPPED R/W.

*TEST24: SCOPE
1\$: JSR PC,TOFF ;TURN OFF T-BIT TRAPPING FOR THIS TEST
MOV #2\$,SLPERR ;SET LOOP ON ERROR POINTER TO 2\$
MOV #600,KIPAR3 ;MAP KERNAL PAGE 3 TO 12-16K
MOV #600,KIPAR4 ;MAP KERNAL PAGE 4 TO 12-16K
MOV #7600,KIPAR7 ;MAP KERNAL PAGE 7 TO THE I/O PAGE
CLR KIPDR3 ;MAP KERNAL PAGE 3 "NO ACCESS"
MOV #77406,KIPDR4 ;MAP KERNAL PAGE 4 R/W, 128 BLKS
MOV #77406,KIPDR7 ;MAP KERNAL PAGE 7 R/W, 128 BLKS
MOV #4\$,MMVEC ;SET M.M. VECTOR TO 4\$ IN CASE OF ABORT
MOV #377,2#60000 ;LOAD ALL 1'S IN LOW BYTE OF TEST LOC.
MOV #-1,R0 ;LOAD EXPECTED CONTENTS OF TEST LOC. IN R0
2\$: BIS #BIT8,SRO ;TURN ON DEST.-ONLY-RELOCATION
MOVB 2#60000,2#100001 ;LOAD HI BYTE OF TEST LOC.-ABORT IF SOURCE RELOCATED
RESET ;TURN OFF DEST.-ONLY-RELOCATION
MOV 2#60000,R1 ;READ CONTENTS OF TEST LOC.
CMP R0,R1 ;WAS TEST LOCATION LOADED PROPERLY?
BEQ 3\$;BRANCH IF YES
ERROR 62 ;TEST LOC. NOT LOADED - INST. WAS ABORTED
;WHEN SOURCE WAS RELOCATED
;FOR TIGHTER SCOPE LOOP REPLACE
;ERROR CALL WITH
;"BR 2\$" = 000764
3\$: MOV #MGMERR,MMVEC ;RESTORE MEM. MGMT. VECTOR
MOV #77406,KIPDR3 ;MAP KERNAL PAGE 3 R/W, 128 BLKS
MOV #1\$,SLPERR ;RESET LOOP ON ERROR POINTER TO 1\$

023474 000004
023476 004737 035432
023502 012737 023572 001110
023510 012737 000600 172346
023516 012737 000600 172350
023524 012737 007600 172356
023532 005037 172306
023536 012737 077406 172310
023544 012737 077406 172316
023552 012737 023652 000250
023560 012737 000377 060000
023566 012700 177777
023572 052737 000400 177572
023600 113737 060000 100001
023606 000005
023610 013701 060000
023614 020001
023616 001401
023620 104062

H07

MD-11-DFKTH-A PDP 11/34 MEM MGMT DIAG
DFKTHA.P11 19-JAN-77 16:02

MACY11 27(1006) 27-JAN-77 08:51 PAGE 49
T24 TEST THAT SOURCE NOT RELOCATED IN MAINTENANCE MODE

| | | | | | | | | |
|------|--------|--------|--------|--------|---|--------|------------------------------|--|
| 2238 | 023644 | 004737 | 035466 | | JSR | PC,TON | :TURN T-BIT TRAPPING BACK ON | |
| 2239 | 023650 | 000430 | | | BR | TST25 | :SKIP TO NEXT TEST | |
| 2240 | | | | | ;* IF THE PROGRAM TRIES TO RELOCATE THE SOURCE, IT SHOULD TRAP TO 4\$ | | | |
| 2241 | | | | | | | | |
| 2242 | 023652 | 042737 | 000400 | 177572 | 4\$: | BIC | #BIT0,SRO | :TURN OFF DEST.-ONLY-RELOCATION THRU PAR/PDR 7 |
| 2243 | 023660 | 013737 | 177572 | 001362 | | MOV | SRO,WASSRO | :SAVE REST OF SRO CONTENTS |
| 2244 | 023666 | 013737 | 177576 | 001364 | | MOV | SR2,WASSR2 | :SAVE CONTENTS OF SR2 |
| 2245 | 023674 | 010637 | 001354 | | | MOV | SP,WASR6 | :SAVE VALUE OF THE STACK POINTER |
| 2246 | 023700 | 012637 | 001356 | | | MOV | (SP)+,TRAPPC | :SAVE PC OF TRAP |
| 2247 | 023704 | 012637 | 001360 | | | MOV | (SP)+,TRAPPS | :SAVE PSW OF TRAP |
| 2248 | 023710 | 042737 | 160000 | 177572 | | BIC | #160000,SRO | :CLEAR ERROR BITS IN SR1 |
| 2249 | 023716 | 104063 | | | | ERROR | 63 | :SOURCE APPARENTLY RELOCATED IN MAINT. MODE |
| 2250 | | | | | | | | :FOR TIGHTER SCOPE LOOP |
| 2251 | | | | | | | | :REPLACE ERROR CALL WITH A |
| 2252 | | | | | | | | : "NOP" = 000240 |
| 2253 | 023720 | 013746 | 001360 | | | MOV | TRAPPS,-(SP) | :PUT PSW OF TRAP BACK ON THE STACK |
| 2254 | 023724 | 013746 | 001356 | | | MOV | TRAPPC,-(SP) | :PUT PC OF TRAP BACK ON THE STACK |
| 2255 | 023730 | 000002 | | | | RTI | | :RETURN TO TEST |
| 2256 | | | | | | | | |
| 2257 | | | | | | | | |
| 2258 | | | | | | | | |
| 2259 | | | | | | | | |
| 2260 | | | | | | | | |
| 2261 | | | | | | | | |
| 2262 | | | | | | | | |
| 2263 | | | | | | | | |
| 2264 | | | | | | | | |
| 2265 | | | | | | | | |
| 2266 | | | | | | | | |
| 2267 | | | | | | | | |
| 2268 | | | | | | | | |
| 2269 | | | | | | | | |
| 2270 | | | | | | | | |
| 2271 | | | | | | | | |
| 2272 | | | | | | | | |
| 2273 | | | | | | | | |
| 2274 | | | | | | | | |
| 2275 | | | | | | | | |
| 2276 | | | | | | | | |
| 2277 | | | | | | | | |
| 2278 | | | | | | | | |
| 2279 | | | | | | | | |
| 2280 | | | | | | | | |
| 2281 | | | | | | | | |
| 2282 | | | | | | | | |
| 2283 | | | | | | | | |
| 2284 | | | | | | | | |
| 2285 | | | | | | | | |
| 2286 | 023732 | 000004 | | | | | | |
| 2287 | | | | | | | | |
| 2288 | 023734 | 012700 | 172340 | | 1\$: | MOV | #KIPAR0,R0 | :LOAD ADDRESS OF FIRST KERNEL PAR IN R0 |
| 2289 | 023740 | 005001 | | | | CLR | R1 | :CLEAR R1 |
| 2290 | 023742 | 012702 | 000007 | | | MOV | #7,R2 | :LOAD LOOP COUNTER WITH A 7 |
| 2291 | 023746 | 010120 | | | 2\$: | MOV | R1,(R0)+ | :MAP KERNEL PAR'S TO PAGES 0-6 (4K EACH) |
| 2292 | 023750 | 062701 | 000200 | | | ADD | #200,R1 | |
| 2293 | 023754 | 077204 | | | | SOB | R2,2\$ | :LOOP UNTIL KIPAR0 - KIPAR6 ARE LOADED |

```

*****
*TEST 25          RELOCATION & ADDER TEST (NO CARRIES)
*
* THE FOLLOWING TEST SETS UP THE KERNEL PAR'S AND PDR'S
* FOR THE REST OF THE PROGRAM. IT THEN USES DIFFERENT
* VIRTUAL ADDRESSES AND DIFFERENT VALUES FOR KERNEL PAR 4
* TO PUT DIFFERENT PATTERNS AT THE INPUTS OF THE THREE
* MEMORY MANAGEMENT ADDER CHIPS. THE VALUES ARE SUCH
* THAT NO CARRIES ARE GENERATED OUT OF ANY OF THE ADDER CHIPS.
*
* THE METHOD USED TO SEE THAT THE RIGHT PHYSICAL BUS ADDRESS
* IS FORMED BY THE ADDERS IS TO WRITE A PATTERN TO VIRTUAL
* LOCATION WITH MEMORY MGMT. IN THE MAINTENANCE MODE, AND
* THEN READ THAT LOCATION USING THE PHYSICAL ADDRESS THAT SHOULD
* HAVE BEEN FORMED TO SEE IF THE TEST PATTERN GOT THEIR.
*****
TST25:  SCOPE

```

| | | | | | | | | |
|------|--------|--------|--------|--------|-----|-------|-------------|--|
| 2294 | 023756 | 012710 | 007600 | | | MOV | #7600,(R0) | :MAP KIPAR7 TO THE I/O PAGE |
| 2295 | 023762 | 012700 | 172300 | | | MOV | #KIPDR0,R0 | :LOAD ADDRESS OF FIRST KERNEL PDR IN R0 |
| 2296 | 023766 | 012701 | 077406 | | | MOV | #77406,R1 | :LOAD PDR DATA INTO R1 |
| 2297 | 023772 | 012702 | 000010 | | | MOV | #10,R2 | :LOAD LOOP COUNTER WITH AN 8 |
| 2298 | 023776 | 010120 | | | 3S: | MOV | R1,(R0)+ | :MAP ALL 8 PAGES 128 BLOCKS, UPWARD |
| 2299 | 024000 | 077202 | | | | SOB | R2,3S | :EXPANDABLE, READ/WRITE |
| 2300 | | | | | | | | |
| 2301 | 024002 | 012737 | 024002 | 001110 | 4S: | MOV | #4S,\$LPERR | :SET LOOP ON ERROR POINTER TO 4S |
| 2302 | 024010 | 012700 | 067776 | | | MOV | #67776,R0 | :LOAD PHYSICAL ADDR. PBA INTO R0 |
| 2303 | 024014 | 012701 | 107776 | | | MOV | #107776,R1 | :LOAD VIRTUAL ADDR. VBA INTO R1 |
| 2304 | 024020 | 012702 | 125250 | | | MOV | #125250,R2 | :LOAD TEST PATTERN INTO R2 |
| 2305 | 024024 | 012704 | 000600 | | | MOV | #600,R4 | :LOAD R4 WITH PAR VALUE |
| 2306 | 024030 | 010437 | 172350 | | | MOV | R4,KIPAR4 | :LOAD KERNEL PAR 4 BITS <11:00> |
| 2307 | 024034 | 011037 | 001176 | | | MOV | (R0),\$TMPD | :SAVE CONTENTS AT TEST LOCATION |
| 2308 | 024040 | 052737 | 000400 | 177572 | | BIS | #BIT0,SRO | :TURN ON "DESTINATION-ONLY-RELOCATION" |
| 2309 | 024046 | 010211 | | | | MOV | R2,(R1) | :LOAD 125250 USING ADDER CHIPS (PAR4 + VIRT ADDR.) |
| 2310 | 024050 | 011003 | | | | MOV | (R0),R3 | :READ 125250 BACK WITHOUT USING MEM. MGMT. |
| 2311 | 024052 | 000005 | | | | RESET | | :TURN OFF MEMORY MGMT. MAINT. MODE |
| 2312 | 024054 | 013710 | 001176 | | | MOV | \$TMPD,(R0) | :RESTORE ORIGINAL CONTENTS TO TEST LOC. |
| 2313 | 024060 | 020203 | | | | CMP | R2,R3 | :WAS SAME PATTERN READ BACK THAT WAS |
| 2314 | | | | | | | | :WRITTEN USING "DEST-ONLY-RELOC."? |
| 2315 | 024062 | 001405 | | | | BEQ | 5S | :BRANCH IF YES |
| 2316 | 024064 | 010137 | 001376 | | | MOV | R1,VIRT1 | :SAVE VIRTUAL ADDR. TO FORM PHYS. ADDR |
| 2317 | 024070 | 004737 | 036004 | | | JSR | PC,FORMPA | :GO FORM PHYSICAL ADDRESS FOR TYPING |
| 2318 | 024074 | 104017 | | | | ERROR | 17 | :TEST LOCATION DID NOT HAVE PATTERN |
| 2319 | | | | | | | | :THAT SHOULD HAVE BEEN WRITTEN TO IT. |
| 2320 | | | | | | | | :APPARENTLY PHYSICAL ADDR. WAS |
| 2321 | | | | | | | | :FORMED WRONG BY ADDERS USING |
| 2322 | | | | | | | | :THE VIRTUAL ADDR. AND KIPAR4 |
| 2323 | | | | | | | | :FOR TIGHTER SCOPE LOOP |
| 2324 | | | | | | | | :REPLACE ERROR CALL WITH |
| 2325 | | | | | | | | : "BR 4S" = 000742 |
| 2326 | 024076 | | | | 5S: | | | |
| 2327 | 024076 | 012737 | 024076 | 001110 | 6S: | MOV | #6S,\$LPERR | :SET LOOP ON ERROR POINTER TO 6S |
| 2328 | 024104 | 012700 | 067776 | | | MOV | #67776,R0 | :LOAD PHYSICAL ADDR. PBA INTO R0 |
| 2329 | 024110 | 012701 | 102576 | | | MOV | #102576,R1 | :LOAD VIRTUAL ADDR. VBA INTO R1 |
| 2330 | 024114 | 012702 | 125251 | | | MOV | #125251,R2 | :LOAD TEST PATTERN INTO R2 |
| 2331 | 024120 | 012704 | 000652 | | | MOV | #652,R4 | :LOAD R4 WITH PAR VALUE |
| 2332 | 024124 | 010437 | 172350 | | | MOV | R4,KIPAR4 | :LOAD KERNEL PAR 4 BITS <11:00> |
| 2333 | 024130 | 011037 | 001176 | | | MOV | (R0),\$TMPD | :SAVE CONTENTS AT TEST LOCATION |