IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZMMK-A-D

PRODUCT NAME: UP-DOWN ADDRESS TEST

DATE CREATED: JUNE 7, 1972

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: JIM LACEY

1. ABSTRACT

THIS TEST DEMONSTRATES THAT THE SELECTED MEMORY TEST AREA IS CAPABLE OF BASIC READ AND WRITE OPERATIONS WHEN ADDRESS PROPAGATION IS BOTH UPWARD AND DOWNWARD THROUGH MEMORY.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11

2.2 STORAGE

2.2.1 PROGRAM STORAGE - THE ROUTINE OCCUPIES MEMORY FROM 100 TO 560

3. LOADING PROCEDURE

3.1 METHOD

PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED.

1. ABSOLUTE LOADER MUST BE IN MEMORY.
2. PLACE BINARY TAPE IN READER.
3. LOAD ADDRESS 17500.
4. PRESS "START" (PROGRAM WILL LOAD).

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTING

N/A

4.2 STARTING ADDRESS

200

4.3 PROGRAM AND/OR OPERATOR ACTION

LOAD PROGRAM INTO MEMORY.
SET SWITCH REGISTER TO STARTING ADDRESS.
LOAD ADDRESS.
PRESS START.
THE PROGRAM WILL TEST SELECTED MEMORY AREA AND LOOP.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

5.1.1 SW12 = 1 OR UP...HALT AT END OF TEST

5.2 SUBROUTINE ABSTRACTS

N/A

5.3 PROGRAM AND/OR OPERATOR ACTION

5.3.1 AFTER THE PROGRAM IS LOADED THE TEST LIMIT BOUNDARIES MAY BE MODIFIED AS FOLLOWS:

A. LOAD LOCATION 102 (TAGGED AS LOLMT) WITH THE LOW TEST LIMIT BOUNDARY. (SEE 7.2 FOR OPERATIONAL RESTRICTION)

B. LOAD LOCATION 104 (TAGGED AS HILMT) WITH THE HIGH LIMIT BOUNDARY ADDRESS OF THE TEST.

C. LOAD LOCATION 106 (TAGGED AS LMTKEY) WITH ALL ONES.

D. LOAD ADDRESS AND START AT 200. IF AN ERROR OCCURS THE PROGRAM WILL MOVE THE ERROR ADDRESS TO REGISTER 0 AND THE ERROR DATA TO REGISTER 1 AND THEN HALT. AT THIS TIME THE ERROR ADDRESS MAY BE DETERMINED BY EXAMINING 177700 (R0) AND THE ERROR DATA MAY BE DETERMINED BY EXAMINING 177701 (R1).

6. ERRORS

6.1 ERROR RECOVERY

DEPRESS CONTINUE TO RESTART SECTION

7. RESTRICTIONS

7.1 STARTING RESTRICTION

NONE

7.2 OPERATIONAL RESTRICTION

DO NOT EXTEND THE LOWER LIMIT OF THIS TEST BELOW 452 AS THIS WILL CAUSE THE PROGRAM TO DESTROYED ITSELF.

8. MISCELLANEOUS

8.1 EXECUTION TIME

8.1.1 THE PROGRAM WILL RING THE TELETYPE BELL AFTER FOUR(4) PASSES THROUGH THE PROGRAM WHICH IS APPROXIMATELY ONCE PER MINUTE WITH 4K OF MEMORY AND TEST LIMITS SET UNDER PROGRAM CONTROL.

9. PROGRAM DESCRIPTION

THE UP-DOWN ADDRESS TEST VERIFIES THAT THE MEMORY AREA TESTED IS CAPABLE OF BASIC READ-WRITE OPERATIONS WHEN ADDRESSING IS BOTH UPWARD AND DOWNWARD THROUGH MEMORY. THE PROGRAM WRITES THE ADDRESS OF EACH LOCATION INTO ITSELF, UPWARD THROUGH MEMORY, THEN READ-CHECKS DOWNWARD THROUGH THE TEST AREA. AFTER SUCCESSFULLY WRITING UPWARD AND READ-CHECKING DOWNWARD, THE PROGRAM WILL WRITE DOWNWARD AND READ-CHECK UPWARD. IF, AT ANY TIME, AN ERROR OCCURS, THE PROGRAM WILL HALT WITH THE ERROR ADDRESS IN R0 AND THE ERROR DATA IN R1. A CONTINUE FROM THE ERROR HALT WILL ALLOW THE PROGRAM TO RESUME TESTING.

10. LISTING

```
                                .NLIST  SEQ
                                .TITLE  UP-DOWN ADDRESS TEST
                                ;COPYRIGHT 1972 DIGITAL EQUIPMENT CORP., MAYNARD, MA.  01754
                                ;AUTHOR: JIM LACEY

            000000              R0=%0
            000001              R1=%1
            000002              R2=%2
            000003              R3=%3
            000006              SP=%6
            000007              PC=%7
            010000              SWR12=10000

                                ;TRAP CATCHER IN UNUSED LOCATIONS OF 0-176

            000004                      .=4
000004      000500                      TLG
            000100                      .=100
000100      000002              CKNUM:  2
000102      000452              LOLMT:  STACK
000104      017476              HILMT:  17476
000106      000000              LMTKEY: 0
000110      000000              PASSCT: 0
000112      000000              LPCNTR: 0
000114      000000              RDCNTR: 0
000116      177564              TPS:    177564
000120      177566              TPB:    177566
000122      177570              SWREG:  177570
            000200                      .=200
000200      012706  000452      START:  MOV     #STACK,SP       ;SETUP STACK POINTER
000204      005767  177676              TST     LMTKEY          ;IF LMTKEY IS ZERO GO FIND OUT HOW MUCH
000210      001002                      BNE     BEGIN           ;CORE THERE IS AND SETUP LOLMT AND
000212      004767  000234              JSR     PC,SIZER        ;HILMT, OTHERWISE USE PRESENT LIMITS
000216      016767  177666  177666  BEGIN:  MOV     PASSCT,LPCNTR   ;SETUP PASS COUNTER
                                ;STARTING AT LOLMT AND GOING UP THRU MEMORY TO HILMT, WRITE INTO EACH
                                ;LOCATION ITS OWN ADDRESS.
000224      016700  177652      LOOP:   MOV     LOLMT,R0        ;FIRST ADDRESS TO BE TESTED
000230      010010              WRTUP:  MOV     R0,(R0)         ;WRITE THIS ADRESS AT THIS ADDRESS
000232      005720                      TST     (R0)+           ;UPDATE THE ADDRESS
000234      020067  177644              CMP     R0,HILMT        ;LAST ADDRESS?
000240      101773                      BLOS    WRTUP           ;BR IF NO
                                ;STARTING AT HILMT AND GOING DOWN THRU MEMORY TO LOLMT, VERIFIY THAT
                                ;EVERY LOCATION CONTAINS ITS OWN ADDRESS.
000242      016767  177632  177644      MOV     CKNUM,RDCNTR    ;SETUP READ COUNTER
000250      016700  177630      SRTDN:  MOV     HILMT,R0        ;SETUP FOR CHECKING
000254      011001              READDN: MOV     (R0),R1         ;READ
000256      020001                      CMP     R0,R1           ;CHECK
000260      001402                      BEQ     DWN.OK          ;BR IF OK
000262      000000                      HALT                    ;FAILED R0=ADDRESS, R1=DATA
000264      010010                      MOV     R0,(R0)         ;RESTORE PROPER DATA
000266      005740              DWN.OK: TST     -(R0)           ;UPDATE THE ADDRESS
000270      020067  177606              CMP     R0,LOLMT        ;LAST ADDRESS COMPLETED?
000274      103367                      BHIS    READDN          ;BR IF NO
000276      005367  177612              DEC     RDCNTR          ;DO WE WANT TO READ FROM HILMT TO LOLMT AGAIN?
000302      003362                      BGT     SRTDN           ;BR IF YES
```

```
                                        ;STARTING AT HILMT AND GOING DOWN THRU MEMORY TO LOLMT, WRITE INTO
                                        ;EACH LOCATION ITS OWN ADDRESS
000304  016700  177574                        MOV     HILMT,R0        ;FIRST ADDRESS
000310  010010                        WRTDN:  MOV     R0,(R0)         ;WRITE THE ADDRESS AT THE ADDRESS
000312  005740                                TST     -(R0)           ;UPDATE THE ADDRESS
000314  020067  177562                        CMP     R0,LOLMT        ;LAST ADDRESS COMPLETED?
000320  103373                                BHIS    WRTDN           ;BR IF NO
                                        ;STARTING AT LOLMT AND GOING UP THRU MEMORY TO HILMT, VERIFIY THAT
                                        ;EVERY LOCATION CONTAINS ITS OWN ADDRESS.
000322  016767  177552  177564                MOV     CKNUM,RDCNTR    ;SETUP TO READ EACH LOCATION
000330  016700  177546                SRTUP:  MOV     LOLMT,R0        ;SETUP FOR CHECKING
000334  011001                        READUP: MOV     (R0),R1         ;READ
000336  020001                                CMP     R0,R1           ;CHECK IF DATA IS PROPER
000340  001402                                BEQ     UP.OK           ;BR IF OK
000342  000000                                HALT                    ;FAILED R0=ADR, R1=DATA
000344  010010                                MOV     R0,(R0)         ;RESTORE PROPER DATA
000346  005720                        UP.OK:  TST     (R0)+           ;UPDATE ADDRESS
000350  020067  177530                        CMP     R0,HILMT        ;LAST ADDRESS?
000354  101767                                BLOS    READUP          ;BR IF NO
000356  005367  177532                        DEC     RDCNTR          ;DO WE WANT TO READ FROM LOLMT TO HILMT AGAIN?
000362  003362                                BGT     SRTUP           ;BR IF YES
                                        ;CHECK FOR END OF TEST
000364  005367  177522                        DEC     LPCNTR
000370  003315                                BGT     LOOP            ;BR IF MORE LOOPS ARE REQUIRED
                                        ;END OF TEST SO RING BELL AND DETERMINE IF ANOTHER PASS IS TO BE MADE.
000372  105777  177520                        TSTB    @TPS
000376  100375                                BPL     .-4             ;WAIT ON FLAG
000400  012777  000207  177512                MOV     #207,@TPB       ;RING BELL
000406  032777  010000  177506                BIT     #SWR12,@SWREG   ;HALT SWITCH SET?
000414  001401                                BEQ     CONT            ;BR IF NO
000416  000000                                HALT
000420  013702  000042                CONT:   MOV     @#42,R2         ;LOADED BY A MONITOR?
000424  001404                                BEQ     GOAGN           ;BR IF NO
000426  004712                        ENDADR: JSR     PC,(R2)         ;YUP--RETURN TO MONITOR
000430  000240                                NOP
000432  000240                                NOP
000434  000240                                NOP
000436  000167  177554                GOAGN:  JMP     BEGIN           ;MAKE ANOTHER PASS

        000452                                  .=.+10
000452                                  STACK:
                                        ;THIS IS ONCE ONLY CODE AND IS USED TO FIND OUT HOW MUCH CORE THERE IS,
000452  052767  177777  177426          SIZER:  BIS     #-1,LMTKEY      ;SET KEY SO WE WON'T TRY TO DO THIS AGAIN
000460  012767  000452  177414                  MOV     #STACK,LOLMT    ;SET THE LOLMT
000466  012703  000564                          MOV     #LAST,R3        ;SETUP STARTING ADDRESS TO TEST
000472  005723                          SEE:    TST     (R3)+           ;DO A DATI
000474  000240                                  NOP
000476  000775                                  BR      SEE             ;DIDN'T TRAP SO GO TRY NEXT ADDRESS
000500  022626                          TLG:    CMP     (SP)+,(SP)+     ;ADJUST THE STACK
000502  162703  000004                          SUB     #4,R3           ;R3=LAST MEMORY ADDRESS
000506  005737  000042                          TST     @#42            ;LOADED BY A MONITOR?
000512  001407                                  BEQ     S1              ;BR IF NO
000514  023727  000042  000426                  CMP     @#42,#ENDADR    ;YES--WAS IT DDT1 OR ACT11?
000522  001413                                  BEQ     S2              ;BR IF ACT11
```

```
000524  162703  002734                  SUB     #1500.,R3       ;GIVE OOPS THE LAST 1500. MEMORY LOCATIONS
000530  000410                          BR      S2
000532  162703  000300          S1:     SUB     #300,R3         ;PROTECT THE LOADERS
000536  012767  000004  177344          MOV     #4,PASSCT       ;SETUP FOR 1 MINUTE BEFORE RINGING BELL
000544  012767  000132  177326          MOV     #90.,CKNUM
000552  010367  177326          S2:     MOV     R3,HILMT
000556  012737  000006  000004          MOV     #6,@#4          ;SETUP MEMORY TIME OUT VECTOR
000564  000207                  LAST:   RTS     PC

        000001                          .END
```

```
BEGIN   000216      CKNUM   000100      CONT    000420      DWN.OK  000266
ENDADR  000426      GOAGN   000436      HILMT   000104      LAST    000564
LMTKEY  000106      LOLMT   000102      LOOP    000224      LPCNTR  000112
PASSCT  000110      PC    =%000007      RDCNTR  000114      READDN  000254
READUP  000334      R0    =%000000      R1    =%000001      R2    =%000002
R3    =%000003      SEE     000472      SIZER   000452      SP    =%000006
SRTDN   000250      SRTUP   000330      STACK   000452      START   000200
SWREG   000122      SWR12 = 010000      TLG     000500      TPB     000120
TPS     000116      UP.OK   000346      WRTDN   000310      WRTUP   000230
$1      000532      $2      000552      .     = 000566
```

ERRORS DETECTED: 0

```
*DZMMKA,DZMMKA=DZMMKA/SOL
RUN-TIME:  1  1  0 SECONDS
CORE USED:  3K
```