Osborne/McGraw-Hill

Z80® ASSEMBLY LANGUAGE SUBROUTINES

Lance A. Leventhal
Winthrop Saville

Tests a RAM area specified by a base address and a length in bytes. Writes the values 0, FF_{16}, AA_{16} (10101010_2), and 55_{16} (01010101_2) into each byte and checks whether they can be read back correctly. Places 1 in each bit position of each byte and checks whether it can be read back correctly with all other bits cleared. Clears the Carry flag if all tests run properly. If it finds an error, it exits immediately, setting the Carry flag and returning the test value and the address at which the error occurred.

Procedure: The program performs the single value checks (with 0, FF_{16}, AA_{16}, and 55_{16}) by first filling the memory area and then comparing each byte with the specified value. Filling the entire area first should provide enough delay between writing and reading to detect a failure to retain data (perhaps caused by improperly designed refresh circuitry). The program then performs the walking bit test, starting with bit 7; here it writes the data into memory and attempts to read it back immediately for a comparison.

Registers Used: AF, BC, DE, HL

Execution Time: Approximately 633 cycles per byte tested plus 663 cycles overhead

Program Size: 82 bytes

Data Memory Required: None

Special Cases:

1. An area size of 0000_{16} causes an immediate exit with no memory tested. The Carry flag is cleared to indicate no errors.

2. Since the routine changes all bytes in the tested area, using it to test an area that includes itself will have unpredictable results.

Note that Case 1 means this routine cannot be asked to test the entire memory. Such a request would be meaningless anyway since it would require the routine to test itself.

3. Testing a ROM causes a return with an error indication after the first occasion on which the test value differs from the memory's contents.

Entry Conditions

Base address of test area in HL
Size of test area in bytes in DE

Exit Conditions

If an error is found:
  Carry = 1
  Address containing error in HL
  Test value in A
If no error is found:
  Carry = 0
  All bytes in test area contain 0

Example

1. Data: Base address = 0380_{16}
   Length (size) of area = 0200_{16}

   Result: Area tested is the 0200_{16} bytes starting at address 0380_{16}, that is, addresses 0380_{16} through $057F_{16}$. The order of the tests is

1. Write and read 0
2. Write and read FF_{16}
3. Write and read AA_{16} (10101010_2)
4. Write and read 55_{16} (01010101_2)
5. Walking bit test, starting with 1 in bit 7. That is, start with 10000000_2 (80_{16}) and move the 1 one position right for each subsequent test of a byte.

```
;
;
;
;
;       Title           RAM test
;       Name:           RAMTST
;
;
;
;       Purpose:        Test a RAM (read/write memory) area
;                         1) Write all 0 and test
;                         2) Write all FF hex and test
;                         3) Write all AA hex and test
;                         4) Write all 55 hex and test
;                         5) Shift a single 1 through each bit,
;                            while clearing all other bits
;
;                         If the program finds an error, it exits
;                         immediately with the Carry flag set and
;                         indicates where the error occurred and
;                         what value it used in the test.
;
;       Entry:          Register pair HL = Base address of test area
;                       Register pair DE = Size of area in bytes
;
;       Exit:           If there are no errors then
;                         Carry flag = 0
;                         test area contains 0 in all bytes
;                       else
;                         Carry flag = 1
;                         Register pair HL = Address of error
;                         Register A = Expected value
;
;       Registers used: AF,BC,DE.HL
;
;       Time:           Approximately 633 cycles per byte plus
;                       663 cycles overhead
;
;       Size:           Program 82 bytes
;
;
```

```
RAMTST:
        ;EXIT WITH NO ERRORS IF AREA SIZE IS 0
        LD      A,D             ;TEST AREA SIZE
        OR      E
        RET     Z               ;EXIT WITH NO ERRORS IF SIZE IS ZERO
        LD      B,D             ;BC = AREA SIZE
        LD      C,E

        ;FILL MEMORY WITH 0 AND TEST
        SUB     A
        CALL    FILCMP
        RET     C               ;EXIT IF ERROR FOUND

        ;FILL MEMORY WITH FF HEX (ALL 1'S) AND TEST
        LD      A,0FFH
        CALL    FILCMP
        RET     C               ;EXIT IF ERROR FOUND

        ;FILL MEMORY WITH AA HEX (ALTERNATING 1'S AND 0'S) AND TEST
        LD      A,0AAH
        CALL    FILCMP
        RET     C               ;EXIT IF ERROR FOUND

        ;FILL MEMORY WITH 55 HEX (ALTERNATING 0'S AND 1'S) AND TEST
        LD      A,55H
        CALL    FILCMP
        RET     C               ;EXIT IF ERROR FOUND

        ;PERFORM WALKING BIT TEST.  PLACE A 1 IN BIT 7 AND
        ; SEE IF IT CAN BE READ BACK.  THEN MOVE THE 1 TO
        ; BITS 6, 5, 4, 3, 2, 1, AND 0 AND SEE IF IT CAN
        ; BE READ BACK
WLKLP:
        LD      A,10000000B     ;MAKE BIT 7 1, ALL OTHER BITS 0
WLKLP1:
        LD      (HL),A          ;STORE TEST PATTERN IN MEMORY
        CP      (HL)            ;TRY TO READ IT BACK
        SCF                     ;SET CARRY IN CASE OF ERROR
        RET     NZ              ;RETURN IF ERROR
        RRCA                    ;ROTATE PATTERN TO MOVE 1 RIGHT
        CP      10000000B
        JR      NZ,WLKLP1       ;CONTINUE UNTIL 1 IS BACK IN BIT 7
        LD      (HL),0          ;CLEAR BYTE JUST CHECKED
        INC     HL
        DEC     BC              ;DECREMENT AND TEST 16-BIT COUNTER
        LD      A,B
        OR      C
        JR      NZ,WLKLP        ;CONTINUE UNTIL MEMORY TESTED
        RET                     ;NO ERRORS (NOTE OR C CLEARS CARRY)


        ;*************************************
        ;ROUTINE: FILCMP
        ;PURPOSE: FILL MEMORY WITH A VALUE AND TEST
        ;         THAT IT CAN BE READ BACK
```

```
        ;ENTRY: A = TEST VALUE
        ;       HL = BASE ADDRESS
        ;       BC = SIZE OF AREA IN BYTES
        ;EXIT:  IF NO ERRORS THEN
        ;         CARRY FLAG IS 0
        ;       ELSE
        ;         CARRY FLAG IS 1
        ;         HL = ADDRESS OF ERROR
        ;         DE = BASE ADDRESS
        ;         BC = SIZE OF AREA IN BYTES
        ;         A = TEST VALUE
        ;REGISTERS USED: AF,BC,DE,HL
        ;************************************
FILCMP:
        PUSH    HL              ;SAVE BASE ADDRESS
        PUSH    BC              ;SAVE SIZE OF AREA
        LD      E,A             ;SAVE TEST VALUE
        LD      (HL),A          ;STORE TEST VALUE IN FIRST BYTE
        DEC     BC              ;REMAINING AREA = SIZE - 1
        LD      A,B             ;CHECK IF ANYTHING IN REMAINING AREA
        OR      C
        LD      A,E             ;RESTORE TEST VALUE
        JR      Z,COMPARE       ;BRANCH IF AREA WAS ONLY 1 BYTE

        ;FILL REST OF AREA USING BLOCK MOVE
        ; EACH ITERATION MOVES TEST VALUE TO NEXT HIGHER ADDRESS
        LD      D,H             ;DESTINATION IS ALWAYS SOURCE + 1
        LD      E,L
        INC     DE
        LDIR                    ;FILL MEMORY

        ;NOW THAT MEMORY HAS BEEN FILLED, TEST TO SEE IF
        ; EACH BYTE CAN BE READ BACK CORRECTLY
COMPARE:
        POP     BC              ;RESTORE SIZE OF AREA
        POP     HL              ;RESTORE BASE ADDRESS
        PUSH    HL              ;SAVE BASE ADDRESS
        PUSH    BC              ;SAVE SIZE OF VALUE

        ;COMPARE MEMORY AND TEST VALUE
CMPLP:
        CPI
        JR      NZ,CMPER        ;JUMP IF NOT EQUAL
        JP      PE,CMPLP        ;CONTINUE THROUGH ENTIRE AREA
                                ;NOTE CPI CLEARS P/V FLAG IF IT
                                ;DECREMENTS BC TO 0

        ;NO ERRORS FOUND, SO CLEAR CARRY
        POP     BC              ;BC = SIZE OF AREA
        POP     HL              ;HL = BASE ADDRESS
        OR      A               ;CLEAR CARRY, INDICATING NO ERRORS
        RET

        ;ERROR EXIT, SET CARRY
        ;HL = ADDRESS OF ERROR
        ;A = TEST VALUE
```

```
CMPER:
        POP     BC              ;DE = SIZE OF AREA
        POP     DE              ;BC = BASE ADDRESS
        SCF                     ;SET CARRY, INDICATING AN ERROR
        RET


;                                                                       ;
;                                                                       ;
;       SAMPLE EXECUTION                                                ;
;                                                                       ;
;                                                                       ;

SC9G:
        ;TEST RAM FROM 2000 HEX THROUGH 300F HEX
        ;  SIZE OF AREA = 1010 HEX BYTES
        LD      HL,2000H        ;HL = BASE ADDRESS
        LD      DE,1010H        ;DE = NUMBER OF BYTES
        CALL    RAMTST          ;TEST MEMORY
                                ;CARRY FLAG SHOULD BE 0

        JR      SC9G            ;LOOP FOR MORE TESTING

        END
```