ECED3403 – Assignment 3

Grace Yu

B00902046

July 8th, 2024

1. Testing

1. LD and ST can successfully load and store data into memory

Purpose: Checks for successful implementation of LD and ST instructions.

Configuration: The data string “Grace Yu” is stored into data memory beginning at 0x0100. Using LD and ST, this is copied into data memory beginning at 0x0110. Both increment and decrement are used in these instructions, as well as word and byte manipulation.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: LDST4.asm
Time of assembly: Mon 8 Jul 2024 12:56:32
  1                     ;
  2                     ; Example of index addressing #2
  3                     ; Load and store using bytes and indexing
  4                     ; This is an example of a post-increment
  5                     ;
  6                     ; Copy three bytes from S1 to S2
  7                     ;
  8                             data
  9                             org     #100
 10     0100    7247    S1      ascii   "Grace Yu"
 10     0102    6361
 10     0104    2065
 10     0106    7559
 11                             org     #110
 12     0110    5858    S2      ascii   "XXX"
 12     0112    0058
 13                    ;
 14                            code
 15                            org     #200
 16                    LDST4
 17                    ;
 18                    ; Get address of V1 into R0
 19                    ;
 20     0200    6000            movl    S1,R0           ; R0 = #??00
 21     0202    7808            movh    S1,R0           ; R0 = #0100
 22                    ;
 23                    ; Get address of V1 into R1
 24                    ;
 25     0204    6081            movl    S2,R1           ; R1 = #??10
 26     0206    7809            movh    S2,R1           ; R1 = #0110
 27                    ;
 28                    ; Repeat (without loop):
 29                    ;
 30                    ; mem[R1] <- mem[R0]
 31                    ;
 32                    ; R0, R1 contain effective addresses
 33                    ; Need a temporary (use R2)
 34                    ; R2 <- mem[R0]
 35                    ; mem[R1] <- R2
 36                    ;
 37                    ; Do this 3x:
 38                    ; 110 <- 100
 39                    ; 111 <- 101
 40                    ; 112 <- 102
 41                    ;
 42     0208    5882            LD      R0+,R2          ; R1 <- mem[R0]
 43     020A    5C91            ST      R2,R1+          ; mem[R1] <- R1
 44                    ;
 45     020C    5882            LD      R0+,R2          ; R1 <- mem[R0]
 46     020E    5C91            ST      R2,R1+          ; mem[R1] <- R1
 47                    ;
 48     0210    58C2            LD.B    R0+,R2          ; R1 <- mem[R0]
 49     0212    5CD1            ST.b    R2,R1+          ; mem[R1] <- R1
 50                    ;
 51     0214    58C2            LD.B    R0+,R2          ; R1 <- mem[R0]
 52     0216    5CD1            ST.b    R2,R1+          ; mem[R1] <- R1
 53                    ;
 54     0218    4088            add     $1,R0
 55     021A    4089            add     $1,R1
 56                    ;
 57     021C    5942            LD.b    R0-,R2          ; R1 <- mem[R0]
 58     021E    5D51            ST.b    R2,R1-          ; mem[R1] <- R1
 59                    ;
 60     0220    5942            LD.b    R0-,R2          ; R1 <- mem[R0]
 61     0222    5D51            ST.b    R2,R1-          ; mem[R1] <- R1
 62                    ;
 63                    ; Put breakpoint on next address
 64                    ;
 65                    BrkPtHere
 66                    ;
 67                            end LDST4

Successful completion of assembly - 1P

** Symbol table **

Constants (Equates)
Name                                    Type    Value   Decimal

Labels (Code)
Name                                    Type    Value   Decimal
BrkPtHere                               REL     0224    548     PRI
LDST4                                   REL     0200    512     PRI

Labels (Data)
Name                                    Type    Value   Decimal
S2                                      REL     0110    272     PRI
S1                                      REL     0100    256     PRI

Registers
Name                                    Type    Value   Decimal
R7                                      REG     0007    7       PRI
R6                                      REG     0006    6       PRI
R5                                      REG     0005    5       PRI
R4                                      REG     0004    4       PRI
R3                                      REG     0003    3       PRI
R2                                      REG     0002    2       PRI
R1                                      REG     0001    1       PRI
R0                                      REG     0000    0       PRI
```

Expected Results:	Actual Results:
“Grace Yu” should appear beginning in the original 0x0100 location, as well as beginning in the new 0x0110 location.	Option: m d 100 130 Display instruction or data memory? I – instruction memory D – data memory Enter lower and upper bound 0100: 47 72 61 63 65 20 59 75 00 00 00 00 00 00 00 00 Grace Yu........ 0110: 47 72 61 63 65 20 59 75 00 00 00 00 00 00 00 00 Grace Yu........ Option:

Pass/Fail: PASS

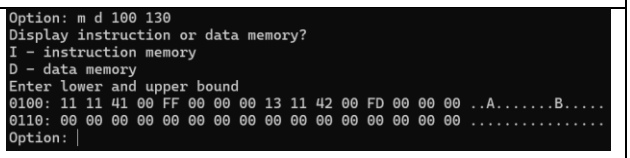
2. LDR and STR can successfully load and store data into memory

Purpose: Checks for successful implementation of LDR and STR instructions.

Configuration: The following .xme file is loaded into the emulator. It uses both LDR and STR commands, with negative and positive offsets, as well as word and byte manipulation.

```
X-Makina Assembler - Version XM-23P Single Pass+ Assembler - Release 24.04.17
Input file name: struct.asm
Time of assembly: Mon 8 Jul 2024 14:52:44
  1                     ;
  2                     ; Structure example
  3                     ; ECED 3403
  4                     ; 25 Jun 2024
  5                     ;
  6                     ; struct stex
  7                     ; {
  8                     ;       unsigned short V0;
  9                     ;       char V1;
 10                     ;       short V2;
 11                     ; };
 12                     ;
 13                     ; struct stex S1, S2;
 14                     ;
 15                     ; Define offsets
 16                     ;
 17                     V0      equ     $0
 18                     V1      equ     $2
 19                     V2      equ     $4
 20                     V3      equ     $-1
 21                     ;
 22                     ; Data - reserve space for structures
 23                     ;
 24                             DATA
 25                             org     #100
 26     0100    0000    S1      bss     $6      ; 6 bytes
 27     0106    0000            bss     $2      ; Fill
 28     0108    0000    S2      bss     $6      ; 6 bytes
 29                     ;
 30                     ; Code - initialize S1
 31                     ;
 32                             CODE
 33                             org     #1000
 34                     ;
 35     1000    6800    StrtEx  movlz   S1,R0
 36     1002    7808            movh    S1,R0
 37                     ;
 38                     ; Initialize S1
 39                     ;
 40     1004    7889            movh    #1100,R1
 41     1006    6089            movl    #11,R1
 42     1008    C008            str     R1,R0,V0
 43                     ;
 44     100A    6A09            movlz   'A',R1
 45     100C    C148            str.b   R1,R0,V1
 46                     ;
 47     100E    6FF9            movlz   #FF,R1
 48     1010    C208            str     R1,R0,V2
 49                     ;
 50                     ; Initialize S2 from S1
 51                     ;
 52     1012    4C02            mov     R0,R2
 53     1014    40A2            add     #8,R2   ; Addr of S2
 54                     ;
 55                     ; s2.v0 <- s1.v0 + 2
 56                     ;
 57     1016    8001            ldr     R0,V0,R1
 58     1018    4091            add     #2,R1
 59     101A    C00A            str     R1,R2,V0
 60                     ;
 61                     ; S2.v1 <- S1.v1 + 1
 62                     ;
 63     101C    8141            ldr.b   R0,V1,R1
 64     101E    4089            add     #1,R1
 65     1020    C14A            str.b   R1,R2,V1
 66                     ;
 67                     ; S2.v2 <- S1.v2 - 2
 68                     ;
 69     1022    409A            add     $4,R2
 70     1024    408A            add     $1,R2
 71     1026    8201            ldr     R0,V2,R1
 72     1028    4291            sub     #2,R1
 73     102A    FF8A            str     R1,R2,V3
 74                     ;
 75                     Done
 76                             end     StrtEx

Successful completion of assembly - 1P

** Symbol table **

Constants (Equates)
Name                                    Type    Value   Decimal
V3                                      CON     FFFF    -1      PRI
V2                                      CON     0004    4       PRI
V1                                      CON     0002    2       PRI
V0                                      CON     0000    0       PRI

Labels (Code)
Name                                    Type    Value   Decimal
Done                                    REL     102C    4140    PRI
StrtEx                                  REL     1000    4096    PRI

Labels (Data)
Name                                    Type    Value   Decimal
S2                                      REL     0108    264     PRI
S1                                      REL     0100    256     PRI

Registers
Name                                    Type    Value   Decimal
R7                                      REG     0007    7       PRI
R6                                      REG     0006    6       PRI
R5                                      REG     0005    5       PRI
R4                                      REG     0004    4       PRI
R3                                      REG     0003    3       PRI
R2                                      REG     0002    2       PRI
R1                                      REG     0001    1       PRI
R0                                      REG     0000    0       PRI
```

Expected Results:	Actual Results:

• Data 0x1111
  ○ Loaded relative from address 0x0100 by 0
  ○ Increased by 2 to 0x1113
  ○ Stored relative from address 0108 by 0
• Data 0x0041
  ○ Loaded relative from address 0x0100 by 2
  ○ Increased by 1 to 0x0042
  ○ Stored relative from address 0x0108 by 2
• Data 0x00FF
  ○ Loaded relative from address 0x0100 by 4
  ○ Decreased by 2 to 0x00FD
  ○ Stored relative from address 0x010D by -1
	

Pass/Fail: PASS