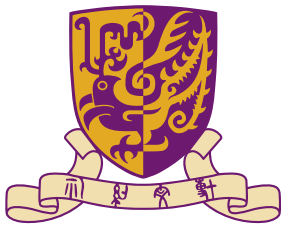
CENG3420
Lab 2-3: LC-3b Simulator to `toupper2.cod`

Bei Yu

Department of Computer Science and Engineering
The Chinese University of Hong Kong

byu@cse.cuhk.edu.hk

Mar. 09, 2017



香 港 中 文 大 學
The Chinese University of Hong Kong

Overview

Asssembler Example: `toupper2`

Task 4 Assignment

Task 4 Golden Results

Overview

Asssembler Example: `toupper2`

Task 4 Assignment

Task 4 Golden Results

toupper2.asm

```
        .ORIG x3000
        LEA R3, LOWERA      ; load address of Lower ``q'' in to R3
        LDW R3, R3, #0      ; load ``q'' into R3
                            ; get read start address
        LEA R0, RSTART      ; load address of instruction rstart
        LDW R0, R0, #0      ; load x4000
        STB R3, R0, #0      ; write ``q'' to mem[x4000]
                            ; get write start address
        LEA R1, WSTART      ; load address of instruction wstart
        LDW R1, R1, #0      ; load x4002
                            ; converstion start
LOOP    LDB R2, R0, #0      ; load character
        BRz EXIT            ; if character is null, exit
        ADD R2, R2, #-16
        ADD R2, R2, #-16    ; r2 - 32
WRITE   STB R2, R1, #0      ; write r2 to address r1
        ADD R0, R0, #1      ; move to next read address
        ADD R1, R1, #1      ; move to next write address
        BR  LOOP

EXIT    STB R2, R1, #0      ; store null-terminated character
        HALT

RSTART .FILL x4000
WSTART .FILL x4002
LOWERA .FILL x0071          ;This is ``q'' in ASCii, hexadecimal
       .END
```

toupper2.asm & toupper2.cod

```
          .ORIG x3000
          LEA R3, LOWERA
          LDW R3, R3, #0

          LEA R0, RSTART
          LDW R0, R0, #0
          STB R3, R0, #0

          LEA R1, WSTART
          LDW R1, R1, #0

LOOP      LDB R2, R0, #0
          BRz EXIT
          ADD R2, R2, #-16
          ADD R2, R2, #-16
WRITE     STB R2, R1, #0
          ADD R0, R0, #1
          ADD R1, R1, #1
          BR  LOOP

EXIT      STB R2, R1, #0
          HALT

RSTART .FILL x4000
WSTART .FILL x4002
LOWERA .FILL x0071
       .END
```

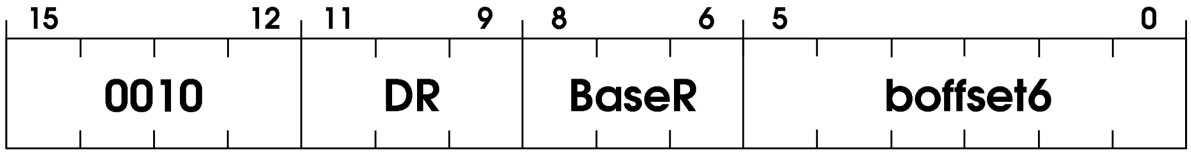
```
0x3000
0xE612
0x66C0

0xE00E
0x6000
0x3600

0xE20C
0x6240

0x2400
0x0406
0x14B0
0x14B0
0x3440
0x1021
0x1261
0x0FF8

0x3440
0xF025

0x4000
0x4002
0x0071
```

toupper2.asm & toupper2.cod

```
          .ORIG x3000
          LEA R3, LOWERA
          LDW R3, R3, #0

          LEA R0, RSTART
          LDW R0, R0, #0
          STB R3, R0, #0

          LEA R1, WSTART
          LDW R1, R1, #0

LOOP      LDB R2, R0, #0
          BRz EXIT
          ADD R2, R2, #-16
          ADD R2, R2, #-16
WRITE     STB R2, R1, #0
          ADD R0, R0, #1
          ADD R1, R1, #1
          BR  LOOP

EXIT      STB R2, R1, #0
          HALT

RSTART .FILL x4000
WSTART .FILL x4002
LOWERA .FILL x0071
       .END
```

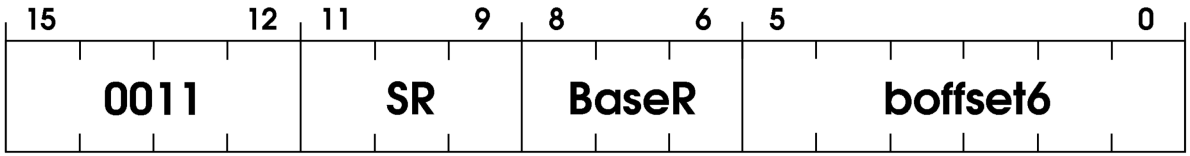
```
0x3000
0xE612
0x66C0

0xE00E
0x6000
0x3600

0xE20C
0x6240

0x2400
0x0406
0x14B0
0x14B0
0x3440
0x1021
0x1261
0x0FF8

0x3440
0xF025

0x4000
0x4002
0x0071
```

LDB:



1. `DR = SEXT(mem[BaseR + SEXT(boffset6)])`
2. `setcc()`

STB:



- `mem[BaseR + SEXT(boffset6)] = SR[7:0]`

Overview

Asssembler Example: `toupper2`

Task 4 Assignment

Task 4 Golden Results

Task 4: Parse `LDB`, `STB` instructions

- Finish the following part

```
490
491             case 2: /* ldb */
492                 /* Lab2-3 assignment: */
493                 break;
494
495             case 3: /* STB */
496                 /* Lab2-3 assignment: */
497                 break;
498
```

- Then the simulator can work on `toupper2.cod`

Overview

Asssembler Example: `toupper2`

Task 4 Assignment

Task 4 Golden Results

Golden Result of Task 4: bench/toupper2.cod

1. run 7

Instructions:

```
process_instruction()| curInstr = 0xe612
process_instruction()| curInstr = 0x66c0
process_instruction()| curInstr = 0xe00e
process_instruction()| curInstr = 0x6000
process_instruction()| curInstr = 0x3600
process_instruction()| curInstr = 0xe20c
process_instruction()| curInstr = 0x6240
```

Registers:

```
Instruction Count : 7
PC                : 0x300e
CCs: N = 0  Z = 0  P = 1
Registers:
0: 0x4000
1: 0x4002
2: 0x0000
3: 0x0071
4: 0x0000
5: 0x0000
6: 0x0000
7: 0x0000
```

Memory Information:

```
Memory content [0x4000..0x4002] :
------------------------------------
  0x4000 (16384) : 0x0071
  0x4002 (16386) : 0x0000
```

Golden Result of Task 4: bench/toupper2.cod

2.Go on run 4

Instructions:

```
process_instruction()| curInstr = 0x2400
process_instruction()| curInstr = 0x0406
process_instruction()| curInstr = 0x14b0
process_instruction()| curInstr = 0x14b0
```

Memory Information:

```
Memory content [0x4000..0x4002] :
------------------------------------
  0x4000 (16384) : 0x0071
  0x4002 (16386) : 0x0000
```

Registers:

```
Instruction Count : 11
PC                : 0x3016
CCs: N = 0  Z = 0  P = 1
Registers:
0: 0x4000
1: 0x4002
2: 0x0051
3: 0x0071
4: 0x0000
5: 0x0000
6: 0x0000
7: 0x0000
```

Golden Result of Task 4: bench/toupper2.cod

3.Go on run 1

Instructions:

```
process_instruction()| curInstr = 0x3440
```

Memory Information:

```
Memory content [0x4000..0x4002] :
--------------------------------------
  0x4000 (16384) : 0x0071
  0x4002 (16386) : 0x0051
```

Registers:

```
Instruction Count : 12
PC                : 0x3018
CCs: N = 0  Z = 0  P = 1
Registers:
0: 0x4000
1: 0x4002
2: 0x0051
3: 0x0071
4: 0x0000
5: 0x0000
6: 0x0000
7: 0x0000
```

Golden Result of Task 4: bench/toupper2.cod

4.Go on run 3

Instructions:

```
process_instruction()| curInstr = 0x1021
process_instruction()| curInstr = 0x1261
process_instruction()| curInstr = 0x0ff8
```

Memory Information:

```
Memory content [0x4000..0x4002] :
------------------------------------
  0x4000 (16384) : 0x0071
  0x4002 (16386) : 0x0051
```

Registers:

```
Instruction Count : 15
PC                : 0x300e
CCs: N = 0  Z = 0  P = 1
Registers:
0: 0x4001
1: 0x4003
2: 0x0051
3: 0x0071
4: 0x0000
5: 0x0000
6: 0x0000
7: 0x0000
```

Golden Result of Task 4: bench/toupper2.cod

5.Go on run 5

Instructions:

```
process_instruction()| curInstr = 0x2400
process_instruction()| curInstr = 0x0406
process_instruction()| curInstr = 0x3440
process_instruction()| curInstr = 0xf025
Simulator halted
```

Registers:

```
Instruction Count : 19
PC                : 0x0000
CCs: N = 0  Z = 1  P = 0
Registers:
0: 0x4001
1: 0x4003
2: 0x0000
3: 0x0071
4: 0x0000
5: 0x0000
6: 0x0000
7: 0x3022
```

Memory Information:

```
Memory content [0x4000..0x4002] :
-------------------------------------
  0x4000 (16384) : 0x0071
  0x4002 (16386) : 0x0051
```