CENG3420
Lab 2-2: LC-3b Simulator

Wei Li

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

wli@cse.cuhk.edu.hk

Spring 2020

香港中文大學
The Chinese University of Hong Kong

Overview

Basis

LC-3b Example: Count From 10 To 1

Tasks

Overview

Basis

LC-3b Example: Count From 10 To 1

Tasks

The Slides are self-contained? NO!

Do please refer to the following documents:

- LC-3b-ISA.pdf
- LC-3b-assembly.pdf

Notations

`DR`

- Destination register

`LSHF(A,b)`

- Shift A to the left by b bits
- If A = 1111 1111 1111 1111, b = 5
- Then `LSHF(A,b)` = 1111 1111 1110 0000

`MEM[addr]`

- Word starting at the given memory address

`setcc()`

- Set condition codes `N, Z, P` based on `DR` value

`SEXT(A)`

- Sign-extend A to 16 bits
- If A = 11 0000, `SEXT(A)` = 1111 1111 1111 0000

Overview

Basis

LC-3b Example: Count From 10 To 1

Tasks

LC-3b Example 2: Count from 10 to 1

count10.asm:

```
      .ORIG x3000
      LEA R0, TEN
      LDW R1, R0, #0
START ADD R1, R1, #-1
      BRZ DONE
      BR START

DONE  TRAP x25
TEN   .FILL x000A
      .END
```

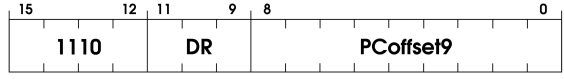
count10.cod:

```
    0x3000
    0xE005
    0x6200
    0x127F
    0x0401
    0x0FFD

    0xF025
    0x000A
```

LEA: Load Effective Address

```
      .ORIG x3000          0x3000
      LEA R0, TEN          0xE005
      LDW R1, R0, #0       0x6200
START ADD R1, R1, #-1      0x127F
      BRZ DONE             0x0401
      BR START             0x0FFD

DONE  TRAP x25             0xF025
TEN   .FILL x000A          0x000A
      .END
```

- 0xE005 → 1110 000 000000101



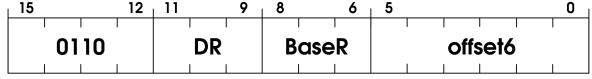
1. DR = PC + 2+ LSHF(SEXT(PCoffset9),1);
2. setcc();

LDW: Load Word

```
      .ORIG x3000
      LEA R0, TEN
      LDW R1, R0, #0
START ADD R1, R1, #-1
      BRZ DONE
      BR START

DONE  TRAP x25
TEN   .FILL x000A
      .END
```

```
0x3000
0xE005
0x6200
0x127F
0x0401
0x0FFD

0xF025
0x000A
```

- 0x6200 → 0110 001 000 000000

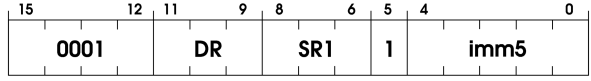


1. DR = MEM[BaseR + LSHF(SEXT(offset6), 1)];
2. setcc();

ADD: Addition

```
      .ORIG x3000          0x3000
      LEA R0, TEN          0xE005
      LDW R1, R0, #0       0x6200
START ADD R1, R1, #-1      0x127F
      BRZ DONE             0x0401
      BR START             0x0FFD

DONE  TRAP x25             0xF025
TEN   .FILL x000A          0x000A
      .END
```

- 0x127F → 0001 001 001 1 11111



1. DR = SR1 + SEXT(imm5);
2. setcc();

Sample: Codes of Addition

```
461     {
462         case 1: /* add, and */
463         case 5:
464             DR = partVal(curInstr, 11, 9);
465             SR1 = partVal(curInstr, 8, 6);
466             if (partVal(curInstr, 5, 5)) /* imm5 */
467             {
468                 imm5 = partVal(curInstr, 4, 0);
469                 value = SEXT(imm5, 5);
470             }
471             else
472             {
473                 SR2 = partVal(curInstr, 2, 0);
474                 value = CURRENT_LATCHES.REGS[SR2];
475             }
476             if (opCode == 1)
477             {
478                 NEXT_LATCHES.REGS[DR] = Low16bits(CURRENT_LATCHES.REGS[SR1] + value);
479             }
480             else if (opCode == 5)
481             {
482                 NEXT_LATCHES.REGS[DR] = Low16bits(CURRENT_LATCHES.REGS[SR1] & value);
483             }
484             setCC(NEXT_LATCHES.REGS[DR]);
485             break;
```

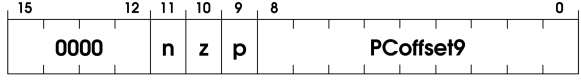
1. `DR = SR1 + SEXT(imm5);`
2. `setcc();`

BR: Conditional Branch

```
      .ORIG x3000
      LEA R0, TEN
      LDW R1, R0, #0
START ADD R1, R1, #-1
      BRZ DONE
      BR START

DONE  TRAP x25
TEN   .FILL x000A
      .END
```

```
0x3000
0xE005
0x6200
0x127F
0x0401
0x0FFD

0xF025
0x000A
```

- 0x0401 → 0000 010 000000001



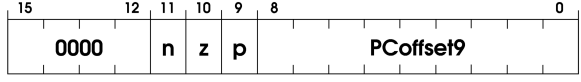
1. if (CURRENT_LATCHES.Z) then:
2. PC = PC + 2 + LSHF(SEXT(PCoffset9), 1);

BR: Conditional Branch

```
      .ORIG x3000
      LEA R0, TEN
      LDW R1, R0, #0
START ADD R1, R1, #-1
      BRZ DONE
      BR START

DONE  TRAP x25
TEN   .FILL x000A
      .END
```

```
0x3000
0xE005
0x6200
0x127F
0x0401
0x0FFD

0xF025
0x000A
```

- 0x0FFD → 0000 111 111111101



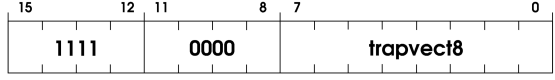
1. `if (CURRENT_LATCHS.N || CURRENT_LATCHES.Z || CURRENT_LATCHES.P)` then:
2. `PC = PC + 2 + LSHF(SEXT(PCoffset9), 1);`

TRAP x25: Halt

```
      .ORIG x3000
      LEA R0, TEN
      LDW R1, R0, #0
START ADD R1, R1, #-1
      BRZ DONE
      BR START

DONE  TRAP x25
TEN   .FILL x000A
      .END
```

```
0x3000
0xE005
0x6200
0x127F
0x0401
0x0FFD

0xF025
0x000A
```

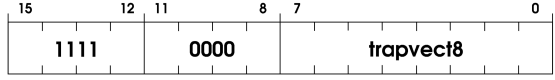
- 0xF025 → 1111 0000 00100101



1. R7 = PC + 2;
2. PC = MEM[LSHF(ZEXT(trapvect8), 1)];

Sample: Codes of TRAP x25

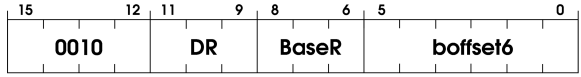
```
523         case 15: /* TRAP */
524             trapvect8 = partVal(curInstr, 7, 0);
525             NEXT_LATCHES.REGS[7] = Low16bits(CURRENT_LATCHES.PC + 2);
526             NEXT_LATCHES.PC = memWord(trapvect8<<1);
527             break;
528
```



1. `R7 = PC + 2;`
2. `PC = MEM[LSHF(ZEXT(trapvect8), 1)];`

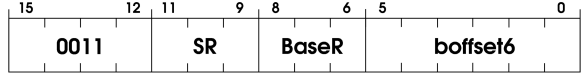
Sample: Codes of LDB and STB

LDB:(Load Byte)



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

STB:(Store Byte)



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

Special treatment of LDB and STB!

The LDB/STB structure is different based on your SID

SID with even ending: LDB structure is different!

1. **Orig:** DR: 9-11, BaseR: 6-8;
2. **In CENG3420:** DR: 6-8, BaseR: 9-11;

SID with odd ending: STB structure is different!

1. **Orig:** SR: 9-11, BaseR: 6-8;
2. **In CENG3420:** SR: 6-8, BaseR: 9-11;

Overview

Basis

LC-3b Example: Count From 10 To 1

Tasks

Task 2:

- Implement `SEXT()` & `setCC()` functions
- Parse `LDW, BR` instructions
- Parse `LDB, STB` instructions

Notes

- You can refer to the implementation of `LEA`
- **Note on the different `LDB/STB` structure for students with different SID**

Example: toupper2.cod & run 10 & rdump

```
LC-3b-SIM> run 10

Simulating for 10 cycles...

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
process_instruction()| curInstr = 0x2080
process_instruction()| curInstr = 0x0406
process_instruction()| curInstr = 0x14b0
LC-3b-SIM> rdump


Current register/bus values :
-------------------------------------
Instruction Count : 10
PC                : 0x3014
CCs: N = 0  Z = 0  P = 1
Registers:
0: 0x4000
1: 0x4002
2: 0x0061
3: 0x0071
4: 0x0000
5: 0x0000
6: 0x0000
7: 0x0000
```

```
LC-3b-SIM> run 10

Simulating for 10 cycles...

process_instruction()| curInstr = 0xe612
process_instruction()| curInstr = 0x66c0
process_instruction()| curInstr = 0xe00e
process_instruction()| curInstr = 0x6000
process_instruction()| curInstr = 0x30c0
process_instruction()| curInstr = 0xe20c
process_instruction()| curInstr = 0x6240
process_instruction()| curInstr = 0x2400
process_instruction()| curInstr = 0x0406
process_instruction()| curInstr = 0x14b0
LC-3b-SIM> rdump


Current register/bus values :
-------------------------------------
Instruction Count : 10
PC                : 0x3014
CCs: N = 0  Z = 0  P = 1
Registers:
0: 0x4000
1: 0x4002
2: 0x0061
3: 0x0071
4: 0x0000
5: 0x0000
6: 0x0000
7: 0x0000
```

even case (left) & odd case (right)