Computer Architecture – LAB 1

김원표
kwp@hallym.ac.kr

Lab 0 – review

- Print string

```
.data              # data part
str: .asciiz "Hello Computer Architecture !"

.text              # source code part
.globl main

main:              # every MIPS program will start from 'main'
    li $v0, 4      # system code 4: print string
    la $a0, str    # $a0 will point out 'str' for printing a string
    syscall        # print

    li $v0, 10     # system code 10: exit program
    syscall        # exit
```

= System.out.print(str);

Lab 1

- **Architecture ?**
  - The programmer's view of computer.
  - Defined by instructions (operation and operand), locations (register, memory).
  - operation과 operand는 what? 을 의미, register, memory는 where? 를 의미.
  - 즉, 프로그래머 입장에서 instruction은 operations + operand

- **MIPS 설계 원칙**
  - Simplicity favors regularity.
    - 간단하기 위해선 규칙적인 것이 중요
  - Make the common case fast.
    - 자주 발생되는건 빠르게 처리되게
  - Smaller is faster.
  - Good designs demands good compromise.

Lab 1

- Instruction Format

op	rs	rt	rd	shamt	funct
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

- opcode : the operation code
  - Every R-type operation's opcode is zero
- rs, rt : the source registers
- rd : the destination register
- shamt : the amount to shift
  - Among every R-type instructions, only shift operations use this field.
- function : the specific function

Lab 1

- R-type Instructions : ADD & SUB
- ADD
  - add $s0, $s1, $s2 # $s0 <- $s1 + $s2

0	17 ($s1)	18 ($s2)	16 ($s0)	0	32
op	rs	rt	rd	shmat	funct

- SUB
  - sub $t0, $t3, $t5 # $t0 <- $t3 - $t5

0	11 ($t0)	13 ($t3)	8 ($t5)	0	34
op	rs	rt	rd	shmat	funct

Lab 1

- R-type Instructions : Logical Operations 1
- AND
  - and $s3, $s1, $s2

$s3 <- $s1 AND $s2

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

Source Registers

$s1	1111	1111	1111	1111	0000	0000	0000	0000
$s2	0100	0110	1010	0001	1111	0000	1011	0111

Destination Registers

$s3	0100	0110	1010	0001	0000	0000	0000	0000

- OR
  - or $s4, $s1, $s2

$s4 <- $s1 OR $s2

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

Source Registers

$s1	1111	1111	1111	1111	0000	0000	0000	0000
$s2	0100	0110	1010	0001	1111	0000	1011	0111

Destination Registers

$s4	1111	1111	1111	1111	1111	0000	1011	0111

Lab 1

- R-type Instructions : Logical Operations 2
- XOR
  - xor $s5, $s1, $s2

$s5 <- $s1 XOR $s2

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

Source Registers

$s1	1111	1111	1111	1111	0000	0000	0000	0000
$s2	0100	0110	1010	0001	1111	0000	1011	0111

Destination Registers

$s5	1011	1001	0101	1110	1111	0000	1011	0111

- NOR
  - nor $s6, $s1, $s2

$s6 <- $s1 NOR $s2

A	B	Output
0	0	1
0	1	0
1	0	0
1	1	0

Source Registers

$s1	1111	1111	1111	1111	0000	0000	0000	0000
$s2	0100	0110	1010	0001	1111	0000	1011	0111

Destination Registers

$s6	0000	0000	0000	0000	0000	1111	0100	1000

Lab 1

- **실습 1 : AND and SUB**

```
1
2   .data
3   str: .asciiz "자기학번 / 이름"
4   newline: .asciiz "\n"
5
6   .text
7   .globl main
8
9   main:
10      li $v0, 4
11      la $a0, str
12      syscall
13      li $v0, 4
14      la $a0, newline
15      syscall
16
17      la $s1, 30
18      la $s2, 40
19
20      add $s3, $s1, $s2
21      sub $s4, $s1, $s2
22
23      li $v0, 1
24      move $a0, $s3
25      syscall
```

```
26
27      li $v0, 4
28      la $a0, newline
29      syscall
30
31      li $v0, 1
32      move $a0, $s4
33      syscall
34
35      li $v0, 4
36      la $a0, newline
37      syscall
38
39      li $v0, 10
40      syscall
```

- **result**

```
R17 [s1] = 1e
R18 [s2] = 28
R19 [s3] = 46
R20 [s4] = fffffff6
```

$s1 : 0x1e = 30
$s2 : 0x28 = 40

Console

```
자기학번 / 이름
70
-10
```

Lab 1

- 실습 2 : Logical

```
2   .data
3   str: .asciiz "자기학번 / 이름"
4   newline: .asciiz "\n"
5
6   .text
7   .globl main
8
9   main:
10      li $v0, 4
11      la $a0, str
12      syscall
13      li $v0, 4
14      la $a0, newline
15      syscall
16
17      la $s1, 0x1234
18      la $s2, 0xABCD
19
20      and $s3, $s1, $s2
21      or  $s4, $s1, $s2
22      xor $s5, $s1, $s2
23      nor $s6, $s1, $s2
24
25      li $v0, 10
26      syscall
```

- **result**

```
R17 [s1] = 1234
R18 [s2] = abcd
R19 [s3] = 204
R20 [s4] = bbfd
R21 [s5] = b9f9
R22 [s6] = ffff4402
```

Lab 1

- I-type Instructions : ADDI
  - addi $s1, $s0, 5 # $s1 <- $s0 + 5

8	16 ($s0)	17 ($s1)	5
op	rs	rt	imm

- I-type Instructions : Logical Operations
- ANDI
  - andi $s2, $s1, 0xFA34 # $s2 <- $s1 AND 0xFA34
- ORI
  - ori $s3, $s1, 0xFA34 # $s3 <- $s1 OR 0xFA34
- XORI
  - xori $s4, $s1, 0xFA34 # $s4 <- $s1 XOR 0xFA34

Source Registers

$s1	0000	0000	0000	0000	0000	0000	1111	1111
imm	0000	0000	0000	0000	1111	1010	0011	0100

Destination Register

$s4	0000	0000	0000	0000	1111	1010	1100	1011

Lab 1

- 실습 3 : I-type Instruction – ADDI

```
2   .data
3   str: .asciiz "자기학번 / 이름"
4   newline: .asciiz "\n"
5
6   .text
7   .globl main
8
9   main:
10      li $v0, 4
11      la $a0, str
12      syscall
13      li $v0, 4
14      la $a0, newline
15      syscall
16
17      la $s0, 30
18
19      addi $s1, $s0, 5
20      li $v0, 1
21      move $a0, $s1
22      syscall
23
24      li $v0, 4
25      la $a0, newline
26      syscall
```

```
28      addi $s2, $s0, -50
29      li $v0, 1
30      move $a0, $s2
31      syscall
32
33      li $v0, 10
34      syscall
```

- result

```
R16 [s0] = 1e
R17 [s1] = 23
R18 [s2] = fffffec
```

Console

```
자기학번 / 이름
35
-20
```

Lab 1

- 실습 4 : I-type Instruction – Logical

```
2   .data
3   str: .asciiz "자기학번 / 이름"
4   newline: .asciiz "\n"
5
6   .text
7   .globl main
8
9   main:
10      li $v0, 4
11      la $a0, str
12      syscall
13      li $v0, 4
14      la $a0, newline
15      syscall
16
17      la $s1, 0xABCD
18
19      andi $s2, $s1, 0x123
20      ori $s3, $s1, 0x123
21      xori $s4, $s1, 0x123
22
23      li $v0, 10
24      syscall
```

- **result**

```
R17 [s1] = abcd
R18 [s2] = 101
R19 [s3] = abef
R20 [s4] = aaee
```

- **Source Registers**

$s1	0xABCD	0000	0000	0000	0000	1010	1011	1100	1101
imm	0x0123	0000	0000	0000	0000	0000	0001	0010	0011

- **Destination Registers**

$s2	0x0101	0000	0000	0000	0000	0000	0001	0000	0001
$s3	0xABEF	0000	0000	0000	0000	1010	1011	1110	1111
$s4	0xAAEE	0000	0000	0000	0000	1010	1010	1110	1110

Lab 1

- **과제 : 아래의 문제를 해결**
- 다음과 같은 2진수 32비트의 값이 있다. 빈 칸에 알맞은 **16진수** 값을 넣으시오

$s1	0x	0000	0000	0010	0001	1011	1000	1101	1111
$s2	0x	0000	1110	0000	0101	0000	0011	0110	0111

- 위의 **$s1, $s2를 각각 사용**하여 $s3은 두 값의 **NOR** 값을, $s4은 두 값의 **XOR** 값을 넣으시오

$s3	0x								
$s4	0x								

- **과제**
  - 실습한 내용(실습 1, 2, 3, 4)의 화면캡쳐 본
  - **실습 코드와 주석 (코드 내의 적색 네모 칸)**
  - **과제 1**
  - 워드 문서로 합하여 제출

- **파일명 ex) ca_01_학번_이름.docx**
  - **스마트 캠퍼스 과제란 제출 – 파일명 엄수**

- **제출기한**
  - 9월 28일 23:59까지

- **수업시간 내 완료시 조교의 확인을 받고 퇴실 가능, 미확인시 결석처리**