Computer Architecture – LAB 2

김원표

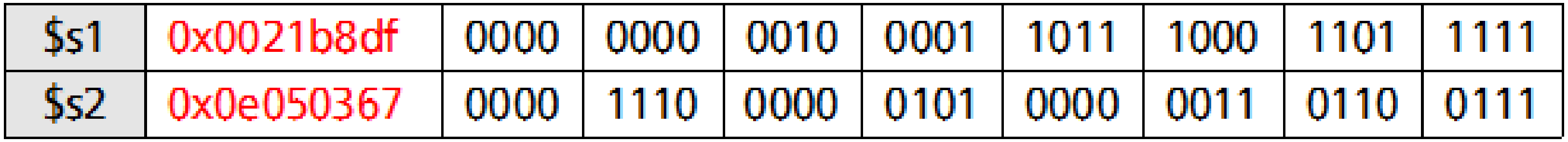
kwp@hallym.ac.kr

Index

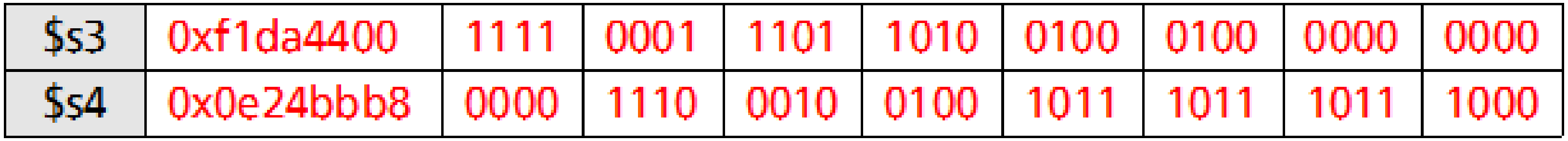
- Review
- Little Bit Different Kinds of Instructions
- Shift Instructions
- Some Ways for Console I/O

Lab 1 – Review

- **Review**



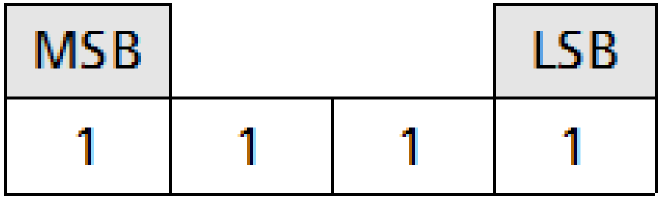
$s1	0x0021b8df	0000	0000	0010	0001	1011	1000	1101	1111
$s2	0x0e050367	0000	1110	0000	0101	0000	0011	0110	0111



$s3	0xf1da4400	1111	0001	1101	1010	0100	0100	0000	0000
$s4	0x0e24bbb8	0000	1110	0010	0100	1011	1011	1011	1000

Lab 2

- **MSB(Most Significant Bit) and LSB(Least Significant Bit)**



MSB			LSB
1	1	1	1

- **가장 왼쪽 자리는 MSB, 오른쪽 자리는 LSB를 나타낸다.**

- **e.g**
  - 만약 누군가가 15K (15: binary 1111)을 다른 사람에게 전달할 때,
  - MSB가 손상되었다면, 다른 사람은 7K (0111)를 받게될 수 있다.
  - LSB가 손상되었다면, 14K(1110)를 받게될 것이다.

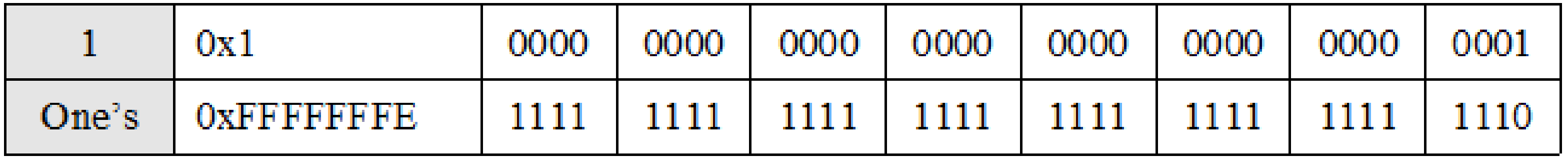
Lab 2

Signed and Unsigned Numbers

- 2진 숫자는 signed 이거나 unsigned 이다.
- 일반적으로, signed 숫자는 2의 보수 표현을 통해 표시된다.

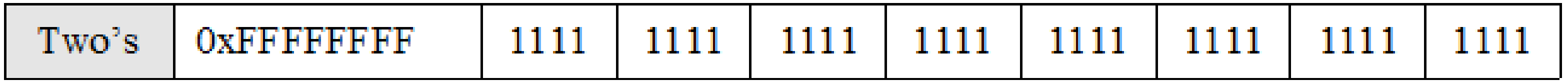
How can we describe negative numbers?

- The number 1 and the one's complement.



1	0x1	0000	0000	0000	0000	0000	0000	0000	0001
One's	0xFFFFFFFE	1111	1111	1111	1111	1111	1111	1111	1110

- The two's complement, further it is same with the number -1.



Two's	0xFFFFFFFF	1111	1111	1111	1111	1111	1111	1111	1111

Lab 2

- ## ADDU and ADDIU
  - addu \$s3, \$s1, \$s2 # \$s3 <- \$s1 + \$s2
  - It does not have a possibility of overflow(just ignore).

- ## What is the difference with ADD
  - add \$s4, \$s1, \$s2 # \$s4 <- \$s1 + \$s2
  - It has a possibility of overflow

- ## ADDIU (I-type Instruction)
  - addiu \$s5, \$s1, 5 # \$s5 <- \$s1 + 5
  - It uses a 16-bit immediate value (constant) as a source.
  - It does not have a possibility of overflow (just ignore).

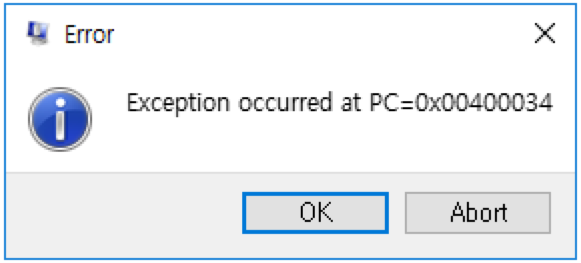
- **In hexadecimal number system, one digit is 4-bit, two digits are 8-bit (1-byte).**

Lab 2

ADDU and ADDIU

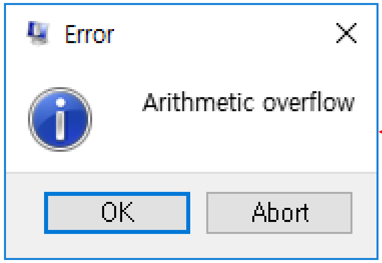
```
.text
.globl main

main:
    la $s1, 0x80000001
    la $s2, 0x80000001

    add  $s3, $s1, $s2      # $s3 <- $s1 + $s2 (overflow)
    addu $s4, $s1, $s2      # $s4 <- $s1 + $s2 (done)

    # I-type uses a 16-bit constant value. (e.g 0x1234)
    la $s5, 0xefffffff
    addiu $s6, $s5, 0x0fff  # $s6 <- $s5 + 0x0fff

    li $v0, 10
    syscall
```

Result



Error

Exception occurred at PC=0x00400034



Error

Arithmetic overflow

과제 1. 다음과 같은 오류가 발생하는 이유를 설명하시오

Lab 2

- **SUBU (U: Unchecked)**
  - subu $s3, $s1, $s2 # $s3 <- $s1 - $s2
  - It does not have a possibility of overflow (just ignore).

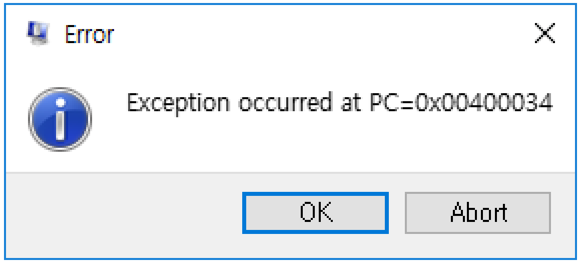
- **What is the difference with SUB**
  - sub $s4, $s1, $s2 # $s4 <- $s1 - $s2
  - It has a possibility of overflow.

Lab 2

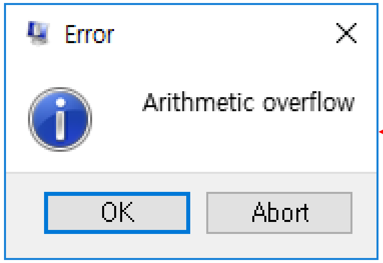
SUBU

```
.text
.globl main

main:
    la $s1, 0x90000001
    la $s2, 0x70000001

    sub  $s3, $s1, $s2      # $s3 <- $s1 - $s2 (overflow)
    subu $s4, $s1, $s2      # $s4 <- $s1 - $s2

    li $v0, 10
    syscall
```

Result



Exception occurred at PC=0x00400034



Arithmetic overflow

과제 2. 다음과 같은 오류가 발생하는 이유를 설명하시오

Lab 2

- ## Logical NOT
  - Source Registers

$s1	0x1234	0000	0000	0000	0000	0001	0010	0011	0100
$0	0x0000	0000	0000	0000	0000	0000	0000	0000	0000

  - Destination Register

$s2	0xffffedcb	1111	1111	1111	1111	1110	1101	1100	1011

- ## NOT $s1

$s1	0x1234	0000	0000	0000	0000	0001	0010	0011	0100
NOT	0xffffedcb	1111	1111	1111	1111	1110	1101	1100	1011

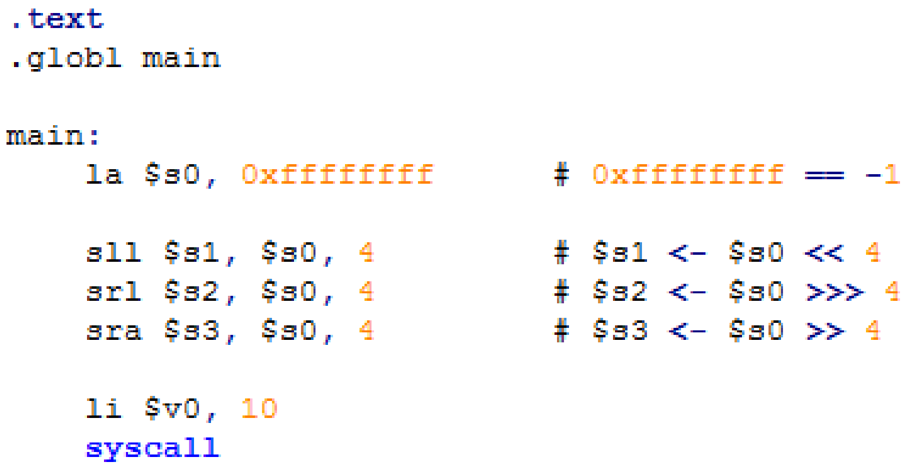
HEX	0xffffedcb	f	f	f	f	e	d	c	b

Lab 2

- **Shift Instructions**
- **SLL (Shift Left Logical)**
  - sll $s1, $s0, 4 # $s1 == $\$s0 * 2^4$ $s1 <- $s0 << 4
  - 왼쪽 시프트는 항상 LSB를 0으로 채운다.
  - Shifting a value left by N is equivalent to multiplying it by 2^N
- **SRL (Shift Right Logical)**
  - srl $s2, $s0, 4 # $s2 <- $s0 >>> 4
  - 오른쪽 시프트는 항상 MSB를 0으로 채운다.
- **SRA (Shift Right Arithmetic)**
  - sra $s3, $s0, 4 # $s3 == $\$s0 * 2^{-4}$ $s3 <- $s0 >> 4
  - In case of SRA, the sign bit shifts into the most significant bits.
  - Arithmetically shifting a value right by N is equivalent to dividing it by 2^N

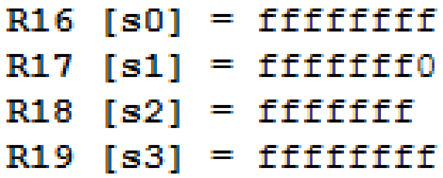
Lab 2

- SLL, SRL and SRA



```
.text
.globl main

main:
    la $s0, 0xffffffff        # 0xffffffff == -1

    sll $s1, $s0, 4           # $s1 <- $s0 << 4
    srl $s2, $s0, 4           # $s2 <- $s0 >>> 4
    sra $s3, $s0, 4           # $s3 <- $s0 >> 4

    li $v0, 10
    syscall
```

- Result



```
R16 [s0] = ffffffff
R17 [s1] = fffffff0
R18 [s2] = fffffff
R19 [s3] = ffffffff
```

Lab 2

- ## SLL, SRL and SRA

Assembly Code
sll $s1, $s0, 4
srl $s2, $s0, 4
sra $s3, $s0, 4

op	rs	rt	rd	shamt	funct
0	0	16	17	4	0
0	0	16	18	4	2
0	0	16	19	4	3

	Source values							
$s0	1111	1111	1111	1111	1111	1111	1111	1111
shamt							0	0100

Assembly Code
sll $s1, $s0, 4
srl $s2, $s0, 4
sra $s3, $s0, 4

Results : Destination Registers							
1111	1111	1111	1111	1111	1111	1111	0000
0000	1111	1111	1111	1111	1111	1111	1111
1111	1111	1111	1111	1111	1111	1111	1111

Lab 2

- Shift Instructions
- SLLV (Shift Left Logical Variable)
  - sllv $s3, $s1, $s2   # $s3 == $\$s1 \times 2^{\$s2}$ $s3 <- $s1 << $s2
  - Left shifts always fill the least significant bits with 0's.
  - Shifting a value left by N is equivalent to multiplying it by 2^N.

- SRLV (Shift Right Logical Variable)
  - srlv $s4, $s1, $s2   # $s4 <- $s1 >>> $s2
  - In case of SRLV, 0's shift into the most significant bits.

- SRAV (Shift Right Arithmetic Variable)
  - srav $s5, $s1, $2   # $s5 == $\$s1 \times 2^{-\$s2}$ $s5 <- $s1 >> $s2
  - In case of SRAV, the sign bit shifts into the most significant bits.
  - Arithmetically shifting a value right by N is equivalent to dividing it by 2^N.

Lab 2

SLLV, SRLV and SRAV

Assembly Code
sllv $s3, $s1, $s2
srlv $s4, $s1, $s2
srav $s5, $s1, $s2

op	rs	rt	rd	shamt	funct
0	18	17	19	0	4
0	18	17	20	0	6
0	18	17	21	0	7

	Source values							
$s1	1111	1111	1111	1111	1111	1111	1111	1111
$s2							0	0100

Assembly Code
sllv $s3, $s1, $s2
srlv $s4, $s1, $s2
srav $s5, $s1, $s2

Results : Destination Registers							
1111	1111	1111	1111	1111	1111	1111	0000
0000	1111	1111	1111	1111	1111	1111	1111
1111	1111	1111	1111	1111	1111	1111	1111

Reference

- **MIPS 명령어 및 기본지식**
  - https://en.wikipedia.org/wiki/MIPS_instruction_set

- **과제**
  - 실습한 내용의 화면캡쳐 본
  - **실습 코드, 과제 1, 2**
  - 워드 문서로 합하여 제출

- **파일명 ex) ca_02_학번_이름.docx**
  - **스마트 캠퍼스 과제란 제출 – 파일명 엄수**

- **제출기한**
  - 10월 5일 23:59까지

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