King Fahd University of Petroleum and Minerals
College of Computer Sciences and Engineering
Computer Engineering Department
COE 301: Computer Architecture

LAB 03:
Integer Arithmetic

Saleh AlSaleh

Agenda

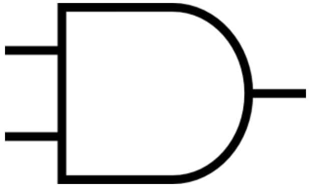
- Overflow
- Logical Bitwise Instructions
- Shift Instructions
- Pseudo Instructions
- Live Examples
- Tasks

Overflow

- Maximum positive integer number represented in 4-bit: $(+7)_{10} = (0111)_2$
- Minimum negative integer number represented in 4-bit: $(-8)_{10} = (1000)_2$
- Maximum positive integer number represented in 32-bit: $(0x7FFFFFFF)_{16}$
- Minimum negative integer number represented in 32-bit: $(0x80000000)_{16}$
- add/sub causes/raises arithmetic exception in the case of overflow and result is not written.
- addu/subu ignores overflow and writes result to destination register

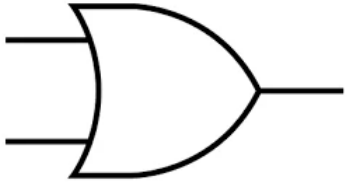
Logical Bitwise Instructions

- AND



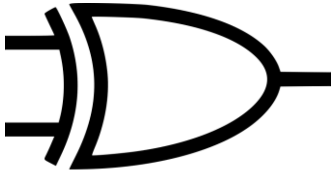
A	0	1	0	1
B	1	1	0	0
A & B	0	1	0	0

- OR



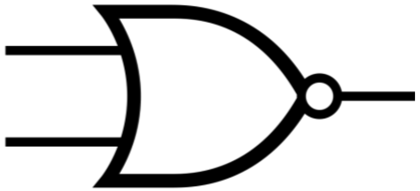
A	0	1	0	1
B	1	1	0	0
A | B	1	1	0	1

- XOR



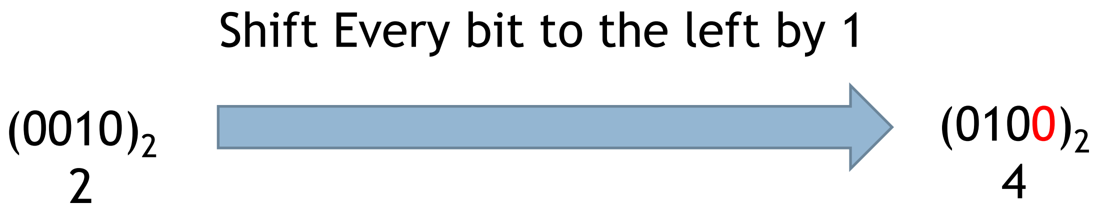
A	0	1	0	1
B	1	1	0	0
A xor B	1	0	0	1

- NOR



A	0	1	0	1
B	1	1	0	0
A nor B	0	0	1	0

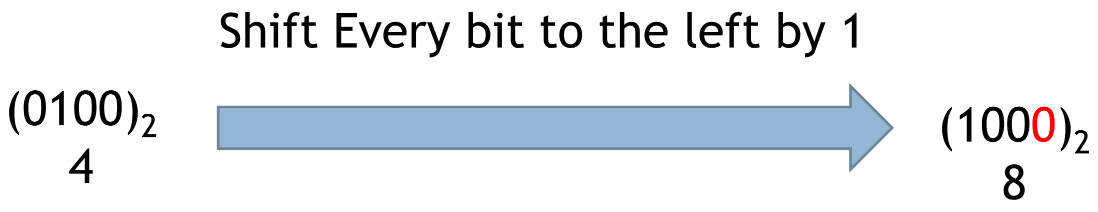
Shift Instructions (Left Shift)



Shift Every bit to the left by 1

$(0010)_2$
2

$(0100)_2$
4



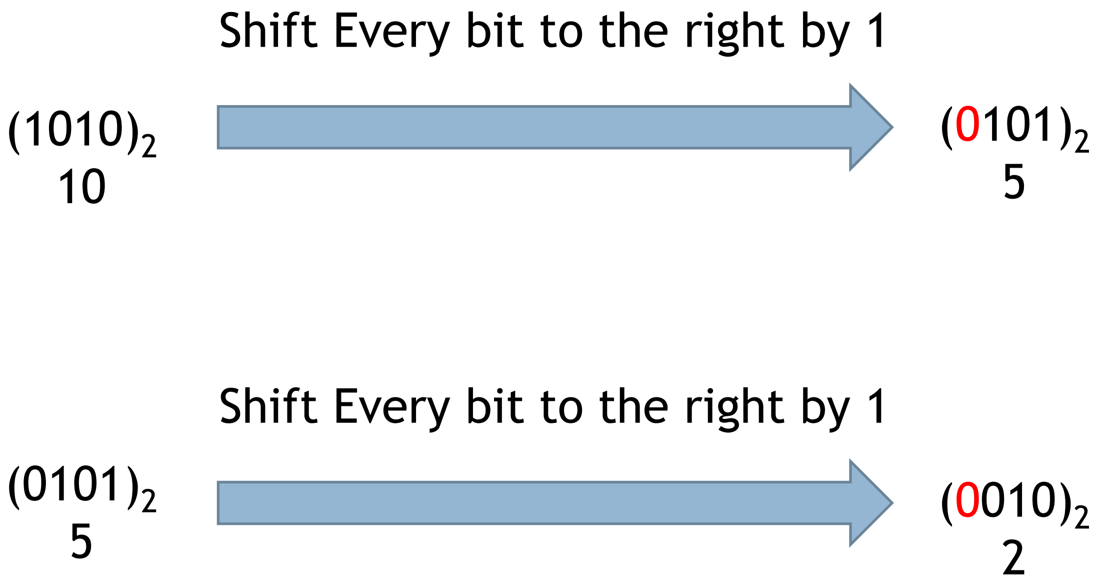
Shift Every bit to the left by 1

$(0100)_2$
4

$(1000)_2$
8

This is called Shift Left Logical (sll)
Every single shift left logical is equivalent to multiplying by 2
MIPS instruction: sll $dst, $src, shift_amount

Shift Instructions (Logical Right Shift)



Shift Every bit to the right by 1

$(1010)_2$
10

$(0101)_2$
5

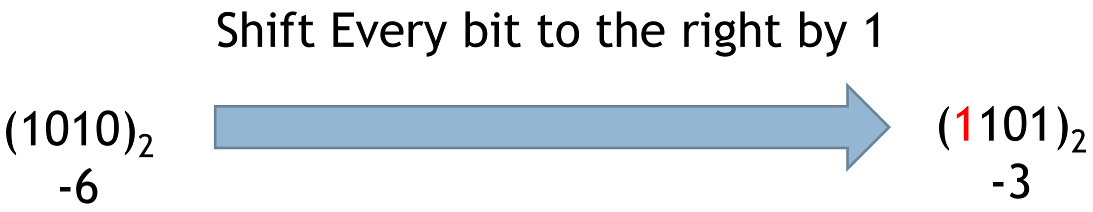
Shift Every bit to the right by 1

$(0101)_2$
5

$(0010)_2$
2

This is called Shift Right Logical (srl)
Every single shift right logical is equivalent to dividing by 2 (with floor)
MIPS instruction: srl $dst, $src, shift_amount

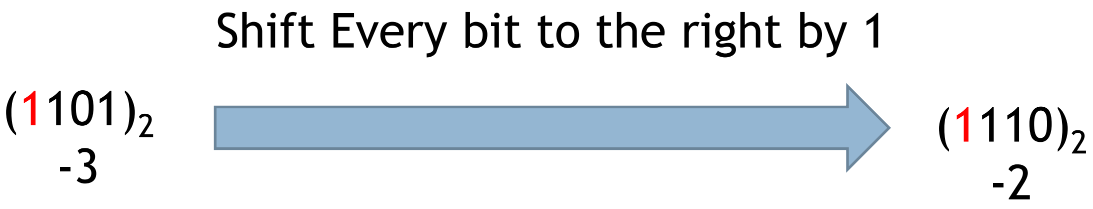
Shift Instructions (Arithmetic Right Shift)



Shift Every bit to the right by 1

$(1010)_2$
-6

$(1101)_2$
-3



Shift Every bit to the right by 1

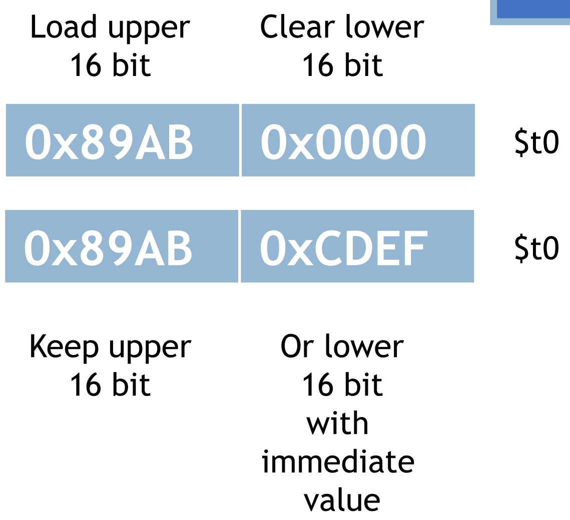
$(1101)_2$
-3

$(1110)_2$
-2

This is called Shift Right Arithmetic (sra)
Every single shift right arithmetic is equivalent to dividing by 2 (with floor) for **signed** numbers
MIPS instruction: sra $dst, $src, shift_amount

Pseudo Instructions

- Maps to one or more basic simple assembly instruction(s)
- Eases the programmer's tasks in writing applications.
- Common pseudo instructions: li, la, abs
  - li $t0, 0xABCD => addi $t0, $0, 0xABCD
  - li $t0, 0x89AB_CDEF => lui $t0, 0x89AB
    ori $t0, $t0, 0xCDEF



Load upper 16 bit

Clear lower 16 bit

0x89AB 0x0000 $t0

0x89AB 0xCDEF $t0

Keep upper 16 bit

Or lower 16 bit with immediate value

Live Examples