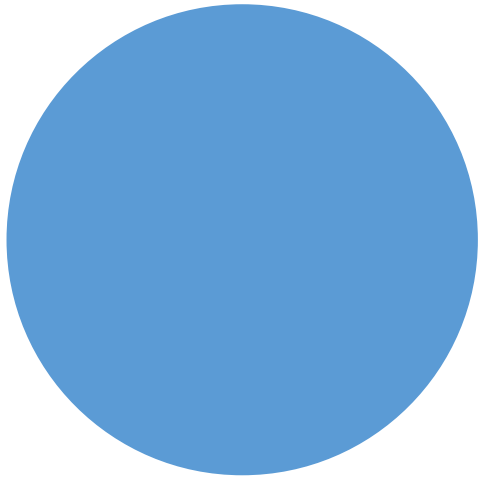
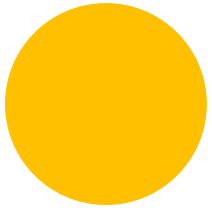



# CSARCH Lecture Series: Integer operation and overflow


Sensei RL Uy

College of Computer Studies

De La Salle University

Manila, Philippines

# Copyright Notice

This lecture contains copyrighted materials and is use solely for instructional purposes only, and not for redistribution.

Do not edit, alter, transform, republish or distribute the contents without obtaining express written permission from the author.

# Overview

Reflect on the following questions:

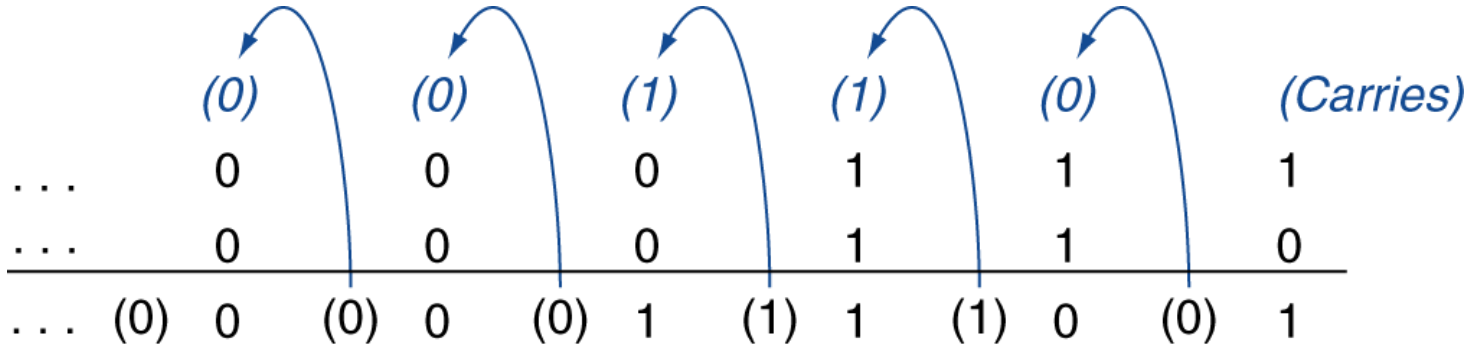
- How do you perform addition and subtraction for binary integer ?
- How do we know if the result is correct (i.e., not overflow)?

$$10111111_2 + 11000001_2 = ?$$

# Overview

- This sub-module discusses how to perform addition and subtraction for binary integer and determine if the result of the operation is an overflow.
- The objectives are as follows:
  - ✓Describe the process of performing integer addition and subtraction
  - ✓Explain if overflow occurs after performing addition and subtraction

# Addition of Integer




```
   111   (carry)
   1111
+) 1111
------
1 1110
(carry out)
```

2= $10_2$ → 0 carry 1

3= $11_2$ → 1 carry 1

- In order to add multiple-bit numbers, we used a method analogous to that used for manual computation with decimal numbers ➔ Add bit pairs starting from the LSb of the bit vectors, propagating carries toward the MSb

# Overflow and Addition of Unsigned Integer

```
   111   (carry)
   1111
+) 1111
------
1  1110
(carry out)
```

- 4-bit binary addition of unsigned integer:

  15 + 15 = ? [$1111_2 + 1111_2 = ?$]

  Is the result an overflow?

- 15 + 15 = 30 [$1111_2 + 1111_2 = 1110_2$]
- The result is $1110_2$ or 14 but the correct answer is 30
- The result is therefore an overflow.
- There is an overflow because the result needs an additional bit to store the result (i.e., there is a carry-out)
- For unsigned integer, overflow occurs if there is a carry out from the MSb

# Overflow and Addition of Unsigned Integer

```
   111   (carry)
    0111
+)  0111
------
    1110
```

- 4-bit binary addition of unsigned integer:

  $7 + 7 = ?$ $[0111_2 + 0111_2 = ?]$

  Is the result an overflow?

- $7 + 7 = 14$ $[0111_2 + 0111_2 = 1110_2]$
- The result is $1110_2$ or 14 and the correct answer is 14
- The result is not an overflow (i.e., there is no carry out)

# Overflow and Addition of Unsigned Integer

- Unsigned integer: overflow occurs if there is a carry-out

# Overflow and subtraction of unsigned integer

Subtraction of two *n*-digit unsigned numbers *M*-*N* in base *r* can be done as follows:

- Add the minuend *M* to the r's complement of the subtrahend *N*
- If *M* >=*N*, the sum will produce and end carry which can be discarded. What is left is the result of *M*-*N*
- If *M*<*N*, the sum does not produce an end carry and the result is the *r*'s complement of (*M*-*N*). To "view" the answer in a familiar format, take the *r*'s complement of the sum and place a negative sign in front. (Note: since its unsigned, negative should not exist!).

# Overflow and subtraction of unsigned integer

Using 10's complement, perform 72532-3250

```
M  =   72532
N* = + 96750
       -----
      169282
(ignore carry)= 69282
```

*10's complement; 5 digits (i.e., 03250) since minuend is 5 digits

Using 10's complement, perform 3250 - 72532

```
M  =   03250
N = +  27468
       -----
       30718
(Overflow)
```

# Overflow and subtraction of unsigned integer

Using 2's complement, perform $1010100_2 - 1000011_2$

```
 M   =    1010100
 N* = + 0111101
        -------
       10010001
(ignore carry)= 0010001
```

*2's complement

Using 2's complement, perform $1000011_2 - 1010100_2$

```
M   =    1000011
N   = + 0101100
        -------
        1101111
no carry
(overflow)
```

# Try

Using 10’s complement, perform

(1)15-14

(2)14-15

and determine if the result is an overflow

Using 2’s complement, perform

(3)$1111_2$-$1110_2$

(4)$1110_2$-$1111_2$

and determine if the result is an overflow

# Try

Using 10's complement, perform

(1)15-14

(2)14-15

(1)

M = 15

N = + 86

--

101

(ignore carry)= 01

→ not overflow

(2)

M = 14

N = + 85

--

99

(no carry)

→Overflow

# Try

Using 2's complement, perform

(3) $1111_2$-$1110_2$

(4) $1110_2$-$1111_2$

(3)

M = 1111

N = + 0010

---

10001

(ignore carry)= 0001

→ not overflow

(4)

M = 1110

N = + 0001

---

1111

(no carry)

→ overflow

# Overflow and Addition of Signed Integer

```
  111   (carry)
  0111
+)0111
------
  1110
```

- 4-bit binary addition of signed integer:

  $7 + 7 = ?\ [0111_2 + 0111_2 = ?]$

  Is the result an overflow?

- $7 + 7 = 14\ [0111_2 + 0111_2 = 1110_2]$
- The result is $1110_2$ or -2 but the correct answer should be 14
- The result is therefore an overflow.
- There is an overflow because adding two positive integers and the result is a negative integer

# Overflow and Addition of Signed Integer

- 4-bit binary addition of signed integer:

  $3 + 4 = ?$ $[0011_2 + 0100_2 = ?]$

  Is the result an overflow?

```
  0011
+)0100
------
  0111
```

- $3 + 4 = 7$ $[0011_2 + 0100_2 = 0111_2]$
- The result is $0111_2$ or 7 and the correct answer is 7
- The result is not an overflow (adding 2 positive integers and the result is a positive integer)

# Overflow and Addition of Signed Integer

- 4-bit binary addition of signed integer:

  -7 + -6 = ? [$1001_2 + 1010_2 = ?$]

  Is the result an overflow?

```
   1001
+) 1010
------
1  0011
(carry out)
```

- -7 + -6 = -13 [$1001_2 + 1010_2 = 0011_2$]
- The result is $0011_2$ or 3 but the correct answer should be -13
- The result is therefore an overflow.
- There is an overflow because adding two negative numbers results in a positive result

# Overflow and Addition of Signed Integer

```
   1   (carry)
   1101
+) 1100
------
   1001
```

- 4-bit binary addition of signed integer:

  -3 + -4 = ? [$1101_2 + 1100_2$ = ?]

  Is the result an overflow?

- -3 + -4 = -7 [$1101_2 + 1100_2 = 1001_2$]
- The result is $1001_2$ or -7 and the correct answer is -7
- The result is not an overflow (adding 2 negative integers and the result is a negative integer)

# Overflow and Addition of Signed Integer

```
   111   (carry)
   1101
+) 0111
------
1  0100
(carry out)
```

- 4-bit binary addition of signed integer:

  -3 + 7 = ? [$1101_2 + 0111_2 = ?$]

  Is the result an overflow?

- -3 + 7 = +4 [$1101_2 + 0111_2 = 0100_2$]
- The result is $0100_2$ or +4 and the correct answer is +4
- The result is not an overflow (adding a positive with a negative integer is similar to performing subtraction)
- Subtraction operation never cause an overflow result. Why?

# Overflow and Addition of Signed Integer

- Adding positive integer with negative integer and vice-versa→ no overflow
- Adding two positive integers
  - Overflow if the result is negative
- Adding two negative integers
  - Overflow if the result is positive

# Try

$10111111_2 + 11000001_2 = ?$

| Is it an overflow (8-bit integer)? | Yes/No |
|---|---|
| View the result as unsigned integer | |
| View the result as signed integer | |

# Try

$10111111_2 + 11000001_2 = ?$

| Is it an overflow (8-bit integer)? | Yes/No |
|---|---|
| View the result as unsigned integer | Yes |
| View the result as signed integer | No |

```
  10111111
+)11000001
----------
1 10000000
```

```
Unsigned:
  191
+)193
-------
  384
```

```
Signed:
  -65
+)-63
-------
 -128
```

# Subtraction of Signed Integer

- Subtraction can be viewed as adding the 2’s complement of the second operand
  - A – B = A+ (-B)
  - 2’s complement is  used to change the sign of the second operand

# Subtraction of Signed Integer

```
  1       (carry)
  0110
+)1101
------
1 0011
(carry out)
```

- 4-bit binary subtraction using 2's complement addition

  $6 - (+3) = 6 + (-3) = [0110_2 + 1101_2 = ?]$

  Is the result an overflow?

- $6 - (+3) = 6 + (-3) = [0110_2 + 1101_2 = 0011]$
- The result is $0011_2$ or +3 and the correct answer is 3
- The result is not an overflow since "adding" positive with negative integer will never cause an overflow

# To recall …

- What have we learned:
  - ✓Describe the process of performing integer addition and subtraction
  - ✓Explain if overflow occurs after performing addition and subtraction