These slides are being provided with permission from the copyright for in-class (CS2208B) use only. The slides must not be reproduced or provided to anyone outside of the class.

All download copies of the slides and/or lecture recordings are for personal use only. Students must destroy these copies within 30 days after receipt of final course evaluations.

Tutorial 02: Signed Numbers

Computer Science Department

CS2208: Introduction to Computer Organization and Architecture

Winter 2020-2021

Instructor: Mahmoud R. El-Sakka

Office: MC-419

Email: elsakka@csd.uwo.ca

Phone: 519-661-2111 x86996



Signed Numbers

- Computer designers have adopted various techniques to represent negative numbers, including
 - sign and magnitude,
 - biased representation, and
 - o two's complement.



■ Example 1: Convert –743₈ to binary using sign and magnitude method

```
743<sub>8</sub>

→ 1111 100 011<sub>2</sub>

→ 111100011<sub>2</sub>
```

```
0 = 000

1 = 001

2 = 010

3 = 011

4 = 100

5 = 101

6 = 110

7 = 111
```

```
-743_{8}
```

→ 1111100011₂



■ Example 2: Convert –AB.BA₁₆ to binary using sign and magnitude method unsigned

AB.BA₁₆

- **→**1010 1011.1011 1010₂
- → 10101011.1011101₂

-AB.BA₁₆

→110101011.1011101₂

```
0 = 0000
```

$$1 = 0001$$

$$2 = 0010$$

$$3 = 0011$$

$$4 = 0100$$

$$5 = 0101$$

$$6 = 0110$$

$$7 = 0111$$

$$8 = 1000$$

$$9 = 1001$$

$$A = 1010$$

$$B = 1011$$

$$C = 1100$$

$$D = 1101$$

$$E = 1110$$

$$F = 1111$$

value



■ Example 3: Convert –0.0A₁₆ to binary using sign and magnitude method unsigned

 $0.0A_{16}$

- $\rightarrow 0000.0000 \ 1010_2$
- \rightarrow 0.0000101₂

 $-0.0A_{16}$

→ 10.0000101₂

0 = 00001 = 0001

2 = 0010

3 = 0011

4 = 0100

5 = 0101

6 = 0110

7 = 0111

8 = 1000

9 = 1001

A = 1010

B = 1011

C = 1100

D = 1101

E = 1110

F = 1111

value



Biased Representation

■ Example 4: Encode −14₁₀ using excess-32 representation method (a.k.a. biased representation)

To encode a number using *excess-32* method, you need to add 32 to that number.

$$\Box$$
 -14₁₀ + 32₁₀ = 18₁₀

 \square 18₁₀ is the *excess-32* representation of -14_{10}

To decode an *excess-32* value to its original value, you need to subtract 32.

$$\square 18_{10} - 32_{10} = -14_{10}$$



Biased Representation

■ <u>Example 5</u>: Encode 14₁₀ using <u>excess-127</u> representation method (a.k.a. <u>biased representation</u>)

To encode a number using *excess-127* method, you need to add 127 to that number.

$$\Box 14_{10} + 127_{10} = 141_{10}$$

 \square 141₁₀ is the *excess-127* representation of 14₁₀

To decode an *excess-127* value to its original value, you need to subtract 127.

$$\square 141_{10} - 127_{10} = 14_{10}$$



- □ In binary arithmetic, the *two's complement* of a number is formed by
 - Subtracting the number from 2^n .

The *two*'s complement of 01100101_2 is $100000000_2 - 01100101_2 = 10011011_2$

In binary system, the sign is encoded as: MSD = 0 → positive MSD = 1→ negative

Flipping (inverting) all the bits of the number and adding 1.

The *two*'s complement of 01100101_2 is $10011010_2 + 1_2 = 10011011_2$.

Just for the sake of completeness, in radix R systems, the sign is encoded as: MSD < R/2 → positive MSD ≥ R/2→ negative,

- o Processing all the bits of the number from the <u>least significant bit</u> (LSB) towards the <u>most significant bit</u> (MSB)
 - > copying all the zeros until the first 1 is reached,
 - > copying that 1,
 - flipping (inverting) all the remaining bits.

The *two*'s complement of 01100100_2 is 10011100_2 . The *two*'s complement of 01100101_2 is 10011011_2 .



Example 6: Convert –AB.BA₁₆ to binary using 2's complement method

AB.BA₁₆

- \rightarrow 1010 1011.1011 1010₂
- → 10101011.1011101₂
- +AB.BA₁₆
 - \rightarrow 010101011.1011101₂
- -AB.BA₁₆
 - **→**101010100.0100011₂

unsigned value

- 0 = 0000
- 1 = 0001
- 2 = 0010
- 3 = 0011
- 4 = 0100
- 5 = 0101
- 6 = 0110
- 7 = 0111
- 8 = 1000
- 9 = 1001
- A = 1010
- B = 1011
- C = 1100
- D = 1101
- E = 1110
- F = 1111

0 = 0000

9 = 1001

A = 1010

B = 1011

C = 1100

D = 1101

E = 1110

F = 1111



2's Complement

■ Example 7: Convert –0.0A₁₆ to binary using 2's complement method

 $0.0A_{16}$

- $\rightarrow 0000.0000 \ 1010_2$
- \rightarrow 0.0000101₂

 $+0.0A_{16}$

- \rightarrow 00.0000101₂
- $-0.0A_{16}$
 - **→** 11.1111011₂

unsigned value 1 = 0001 2 = 0010 3 = 0011 4 = 0100 5 = 0101 6 = 0110 7 = 01118 = 1000



Signed Numbers

Binary pattern	Unsigned	Signed-and-magnitude	2's complement	Excess-8
0000	0	+0	+0	-8
0001	1	+1	+1	– 7
0010	2	+2	+2	- 6
0011	3	+3	+3	- 5
0100	4 5	+4	+4	-4
0101		+5	+5	-3
0110	6	+6	+6	-2
0111	7	+7	+7	– 1
1000	8	-0	-8	+ 0
1001	9	-1	- 7	+1
1010	10	-2	<u> </u>	+2
1011	11	-3	- 5	+3
1100	12	$-\frac{4}{2}$	-4	+4
1101	13	-5	- 3	+5
1110	14	$-\frac{6}{2}$	-2	+6
1111	15	- 7	-1	+7

For a given *n* bit binary pattern

What is the number of zeros for various values of n?

What is the range for various values of n?

$$\frac{2}{n-1} = 1) \rightarrow 2^{n-1} = 2^{n-1}$$



Unsigned

■ Example 8: Convert 11011.11011₂ to decimal, assuming that it is an unsigned number.

```
11011_{2} \rightarrow 27_{10}
0.11011_{2} \rightarrow 0.84375_{10}
11011.11011_{2} \rightarrow 27.84375_{10}
```

Another method:

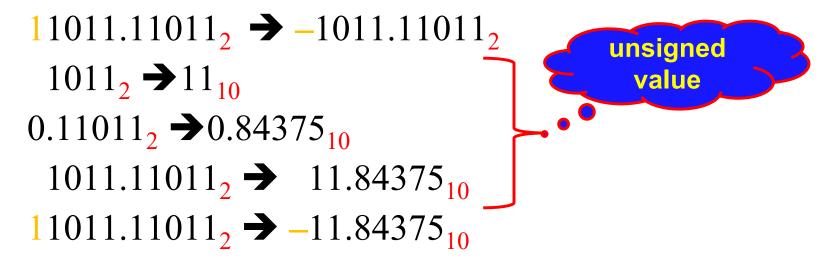
$$11011.11011_{2} = 11011111011_{2} / 100000_{2}$$

$$= 891_{10} / 32_{10}$$

$$= 27.84375_{10}$$



■ Example 9: Convert 11011.11011₂ to decimal, <u>assuming</u> that it is encoded using sign and magnitude method.



Another method:

11011.11011₂
$$\rightarrow$$
 -1011.11011₂
1011.11011₂ = 101111011₂ / 100000₂
= 379₁₀ / 32₁₀ = 11.84375₁₀
11011.11011₂ \rightarrow -11.84375₁₀



■ Example 10: Convert 11011.11011₂ to decimal, assuming that it is encoded using 2's complement method.

```
11011.11011<sub>2</sub> \rightarrow negative number

11011.11011<sub>2</sub> \rightarrow -00100.00101<sub>2</sub> unsigned

00100<sub>2</sub> \rightarrow 4<sub>10</sub>

0.00101<sub>2</sub> \rightarrow 0.15625<sub>10</sub>

00100.00101<sub>2</sub> \rightarrow 4.15625<sub>10</sub>

11011.11011<sub>2</sub> \rightarrow -4.15625<sub>10</sub>
```

Another method:

```
11011.11011<sub>2</sub> \rightarrow negative number

11011.11011<sub>2</sub> \rightarrow -00100.00101<sub>2</sub>

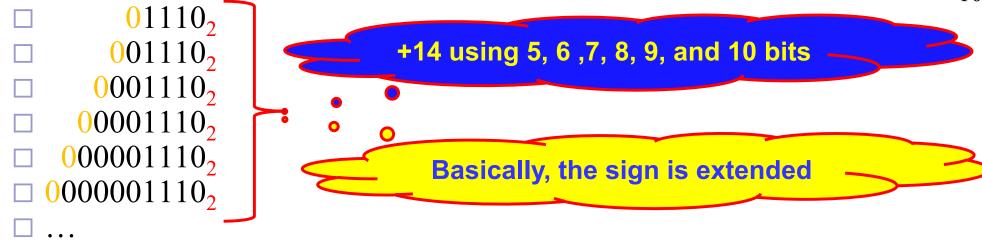
00100.00101<sub>2</sub> = 0010000101<sub>2</sub> / 100000<sub>2</sub>

= 133<sub>10</sub> / 32<sub>10</sub> = 4.15625<sub>10</sub>

11011.11011<sub>2</sub> \rightarrow -4.15625<sub>10</sub>
```



■ The following numbers represent the same value, which is $+14_{10}$



■ By Converting these numbers into the 2's complement, you get



■ Example 11: Convert 11011₂ to decimal, <u>assuming</u> that it is encoded using 2's complement method.

- 11011_2 → negative number
- $\blacksquare 11011_2 \rightarrow -00101_2$
- \bullet 00101₂ \rightarrow 5₁₀
- $\blacksquare 11011_2 \rightarrow -5_{10}$



■ Example 12: Convert 1111011₂ to decimal, assuming that it is encoded using 2's complement method.

- 1111011₂ → negative number
- \blacksquare 1111011₂ \rightarrow -0000101₂
- \bullet 0000101₂ \rightarrow 5₁₀
- \blacksquare 1111011₂ \rightarrow -5₁₀



■ Example 13: Convert 1111111011₂ to decimal, assuming that it is encoded using 2's complement method.

- 1111111011_2 → negative number
- \blacksquare 111111011₂ \rightarrow -000000101₂
- \bullet 000000101₂ \rightarrow 5₁₀
- \blacksquare 1111111011₂ \rightarrow -5₁₀