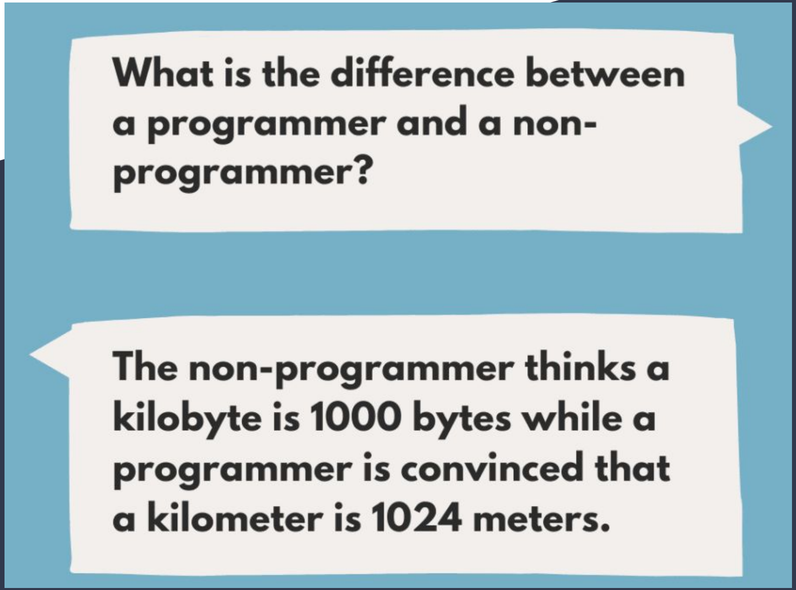
CS 2340 - Computer Architecture

4 Signed Number Representations

Dr. Alice Wang



Review: Bin/Dec/Hex

Hex	Dec	Bin
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111

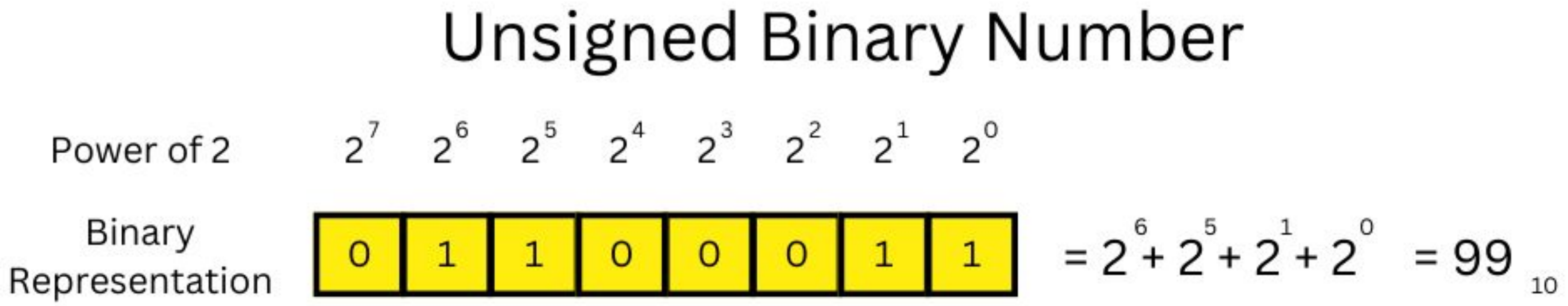
Hex	Dec	Bin
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Review: Bin/Dec/Hex conversion rules

Given a number: $a_{N-1}, a_{N-2}, \dots, a_1, a_0$

From	To		Example	
Decimal	Binary	Divide by 2, Remainder is binary	9	9/2 = 4R**1**, 4/2 = 2R**0**, 2/2 = 1R**0,** 1/2 = 0R**1** → 0b1001
Decimal	Hexadecimal	Divide by 16, Remainder is binary	93	93/16 = 5R**13** “D”, 5/16 = 0R**5** → 0x5D
Binary	Decimal	$\Sigma\, a_i * 2^i$ from i = 0 to N-1	0b0110	$0*2^3 + 1*2^2 + 1*2^1 + 0*2^0 = 6$
Hexadecimal	Decimal	$\Sigma\, a_i * 16^i$ from i = 0 to N-1	0xB8	$11*16^1 + 8*16^0 = 184$
Binary	Hexadecimal	Convert every 4 bits to hex	0b0011_1110	0b0011 → 0x**3**, 0b1110 → **E** 0x3E
Hexadecimal	Binary	Convert every hex digit to bits	0x59	5 → 0101, 9 → 1001: 0b0101_1001

Review: Unsigned Binary Numbers



$$\Sigma\ a_i * r^i \text{ from } i = 0 \text{ to } N-1$$

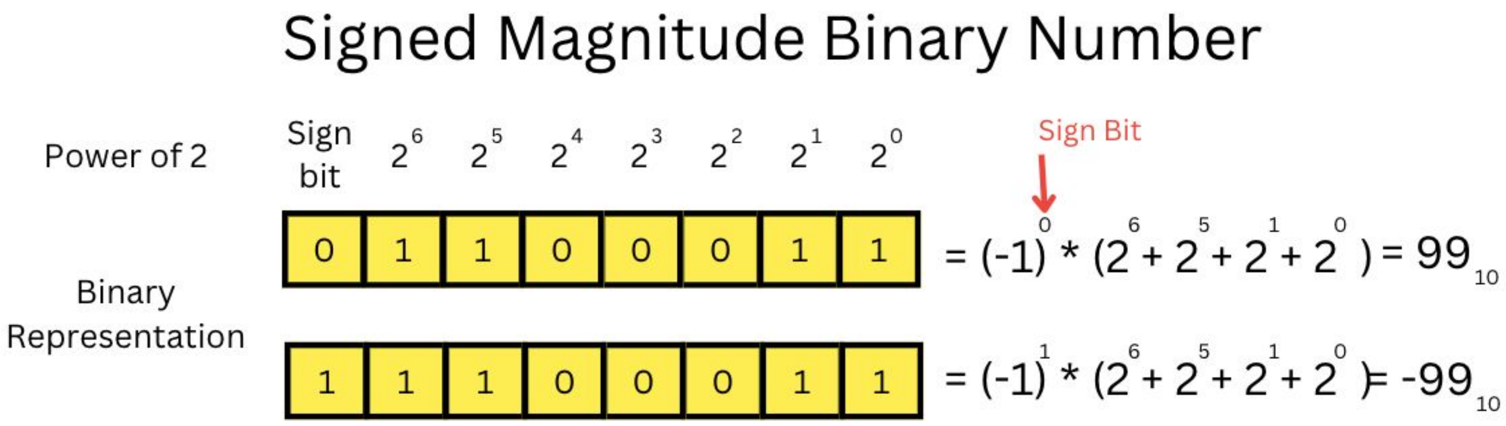
$$r = \text{radix } 2 \text{ (binary)}$$

- Multiply every digit in the binary number with 2 raised to the power based on its position to get the decimal number
- Assumes all numbers are positive (unsigned)

Signed Binary Numbers

- How do we represent negative numbers?
- 2 methods
  - Signed Magnitude
  - Two’s Complement

Signed Magnitude



- Most Significant Bit (MSB) is the sign bit
- Multiply the magnitude by -1 raised to the sign bit
  - Note: $(-1)^0$ is equal to 1

Signed Magnitude example - My Turn

Example: What is the decimal value of Signed Magnitude 0b10011?

1	0 0 1 1
Sign	2^3 2^2 2^1 2^0

=

Signed Magnitude example – Your Turn

Example: What is the decimal value of Signed Magnitude 0b11101?

1	1 1 0 1
Sign	2^3 2^2 2^1 2^0

=

Signed Magnitude Format

$$A = (-1)^{a_{n-1}} * \sum_{i=0}^{n-2} a_i 2^i$$

- The MSB indicates the sign (1 = negative, 0 = positive)
- Range of an *N*-bit signed magnitude number: **$[-(2^{N-1}-1), 2^{N-1}-1]$**

Example: N=4

- Most positive 4-bit number: **0111 = 7**
- Most negative 4-bit number: **1111 = -7**
- Range of a 4-bit signed magnitude number: **[-7, 7]**

Signed Magnitude arithmetic

Two problems with Signed Magnitude representation:

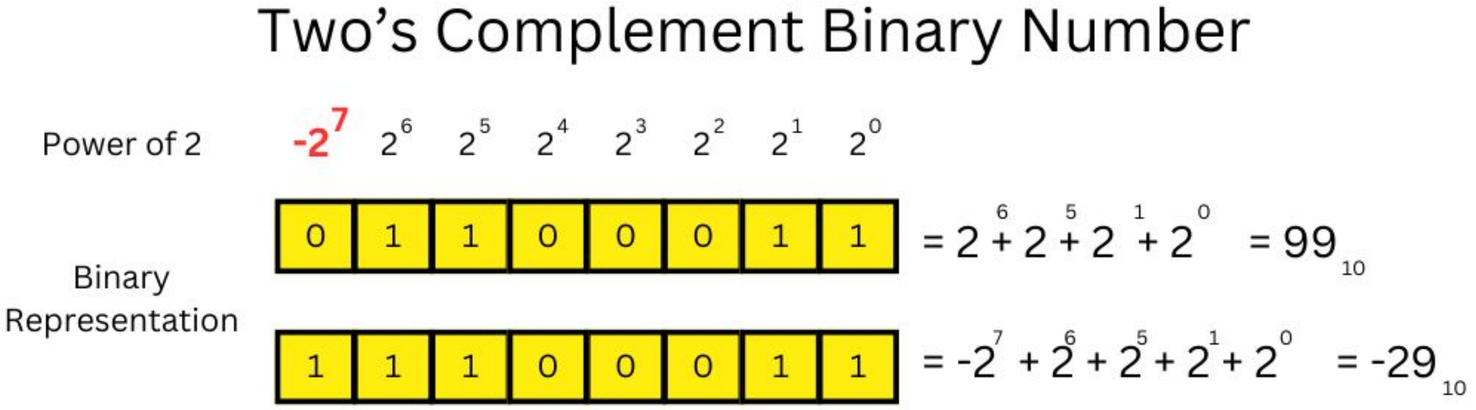
1. Addition doesn't work, for example -6 + 6:

-6		1110
<u>+6</u>		<u>+ 0110</u>
0	≠	10100 **(not zero!)**

2. Two representations of 0 (± 0):

1000	is zero
0000	is also zero

Two’s Complement number representation



- Second way to represent Positive and Negative numbers
- The Power of 2 of the MSB is <u>negative</u>
- Multiply every binary number with its base raised to the power based on its position to get the decimal number

Two's Complement example - My Turn

Example: What is the decimal value of Two's Complement 0b10101?

1 0 1 0 1

$-2^4\ 2^3\ 2^2\ 2^1\ 2^0$

=

Two’s Complement example – Your Turn

Example: What is the decimal value of Two’s Complement 0b11000?

$$1\ 1\ 0\ 0\ 0$$

$$-2^4\ 2^3\ 2^2\ 2^1\ 2^0$$

$=$

Two’s Complement facts

- Doesn’t have the same issues as Signed/Magnitude
  - Addition works!
  - One “Zero”, not two
- Two’s complement is a <u>number representation</u> or <u>format</u>
  - Both positive and negative numbers are Two’s Complement numbers
- Two’s complement is also a <u>procedure</u> or a <u>method</u>
  - “Take the Two’s complement of a number” means that
  - Pos (+) → Neg (-)
  - Neg (-) → Pos (+)

Two's Complement Format

$$A = a_{n-1}\left(-2^{n-1}\right) + \sum_{i=0}^{n-2} a_i 2^i$$

- The MSB indicates the sign (1 = negative, 0 = positive)
- Range of an *N*-bit two's comp number: **$[-2^{N-1}, 2^{N-1}-1]$**

Example : N = 4

- Most positive 4-bit number: **0111 = 7**
- Most negative 4-bit number: **1000 = -8**
- Range of a 4-bit number: **[-8, 7]**

Two's Complement Procedure – Step-by-step

- This procedure converts from a positive 2s complement number to negative 2s complement number
  - Step 1: Starting with the equivalent positive number.
  - Step 2: Inverting (or flipping) all bits – changing every 0 to 1, and every 1 to 0;
  - Step 3: Adding 1 to the entire inverted number, **ignoring any overflow**. Accounting for overflow will produce the wrong value for the result.
- It is the <u>same procedure</u> to go from negative 2s complement to positive 2s complement number

Example: Two's Complement method – My turn

Example: What is -7 in Two's Complement Binary (Hex)?

Step 1: Start with pos # **0b____________ (0x__________)**

Step 2: Flip the bits **0b____________ (0x__________)**

Step 3: +1 to LSB **0b____________ (0x__________)**

Example: Two's Complement method – Your turn

Example: What is -12 in Two's Complement Binary (Hex)?

Step 1: Start with pos # **0b____________ (0x__________)**

Step 2: Flip the bits **0b____________ (0x__________)**

Step 3: +1 to LSB **0b____________ (0x__________)**

Important concept: Sign Extension

- Sign bit copied to MSB's
- Number value is same

- **Example 1:**
  - 4-bit representation of 3 = 0011
  - 8-bit sign-extended value: 00000011 is still 3
- **Example 2:**
  - 4-bit representation of -7 = 1001
  - 8-bit sign-extended value: 11111001 is still -7

Decimal to bin/hex example

Express decimal 14 in all formats below (My Turn):

	Unsigned	Sign/Magnitude	Two’s Complement
Binary			
Hexadecimal			

Express decimal -14 in all formats below (Your Turn):

	Unsigned	Sign/Magnitude	Two’s Complement
Binary			
Hexadecimal			

Two's complement is best for arithmetic

- Has the advantage that the fundamental arithmetic operations of **addition**, **subtraction**, and **multiplication** are identical to those for unsigned binary numbers
- The system is simpler to implement, especially for higher-precision arithmetic
- This why computers use the unsigned and two's complement number formats and not signed magnitude

Two's complement subtraction – *My turn*

- Example: Perform 20-7 in binary.
- Hint: Use 2's complement format on the second number, then perform binary addition

`20 =`

`7  =`

`-7 =`

Add them...

Two's complement subtraction – Your turn

- Example: Perform 7-20 in binary.
- Hint: Use 2's complement format on the second number, then perform binary addition

```
7    =

20   =

-20  =
```

Add them...

Summary: Number System Comparison

Number System	Range
Unsigned	$[0, 2^N\text{-}1]$
Sign/Magnitude	$[-(2^{N-1}\text{-}1), 2^{N-1}\text{-}1]$
Two's Complement	$[-2^{N-1}, 2^{N-1}\text{-}1]$

For example, 4-bit representation:

	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Unsigned									0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Two's Complement	1000	1001	1010	1011	1100	1101	1110	1111	0000	0001	0010	0011	0100	0101	0110	0111								
Sign/Magnitude		1111	1110	1101	1100	1011	1010	1001	0000 1000	0001	0010	0011	0100	0101	0110	0111								

Next lecture

Arithmetic and Logical Operations