Integers I

CS 105: Computer Systems
Lecture 02

Melissa O'Neill

January 26, 2026

Lecture 02 Learning Goals

- **Understand the difference between the encodings of unsigned and signed integers**
- **Compute the minimum and maximum values for unsigned and signed `ints` for a given # of bits (word size)**
- **Explain unsigned and signed (two's complement) encoding of `ints`**
- **Reason about the impact of casting between signed and unsigned `ints`**

Nonnegative (unsigned) Integer Addition

(teaser)

- **Example: adding two integers, word size = 4 bits**

```
  1011
+ 0011
------
```

- **Important: must constrain result to 4 bits!**
  - More on the potential *overflow* later…

Representing Negative Integers

- **We know with *w* bits, we can create 2^w distinct bit vectors**
  - Encoding integers that can only be nonnegative → yields range $0 – 2^w-1$
  - What might be some good properties for how we encode integers using *w* bits that can have *both* negative and nonnegative values?

Negative Integers – Attempt

- **Idea: use one bit to indicate the *sign* of the number**
  - Other bits are the *magnitude* of the number

- **Example: 4 bits, sign bit=1 for negative, 0 for nonnegative**

6 = 0 1 1 0

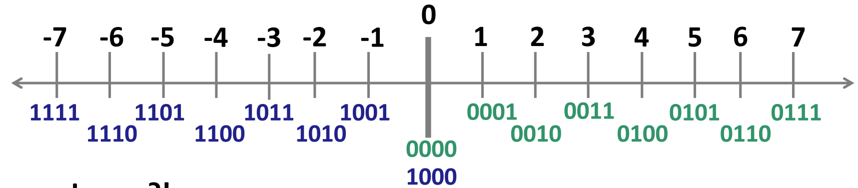
-3 = 1 0 1 1

Sign bit Magnitude bits

Can tell if a number is negative or not just using sign bit!

Negative Integers – Attempt (contd)

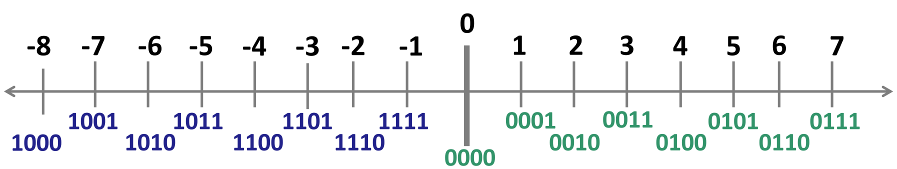
- **Example: 4 bits, sign bit=1 for negative, 0 for nonnegative**
  - Numeric range: -7 to 7



- **Issues?!**

Two's complement encoding

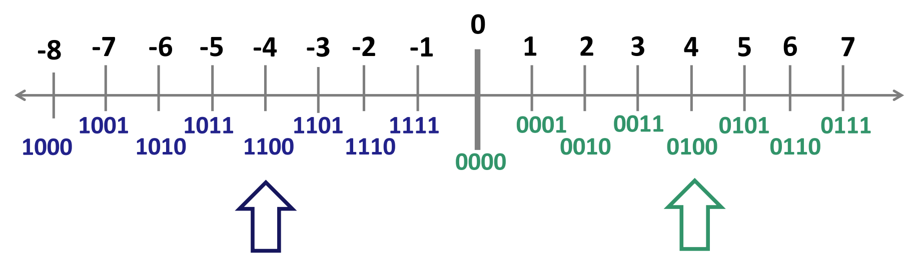
- **Two's complement**
  - Used by systems to represent signed integers
  - Most significant bit still represents sign bit



- **Idea: have bit patterns for 0 and 1, what makes sense for -1?**
  - ______ + 0001 = 0000
- **Etc.,**
  - ______ + 0010 = 0000

Two's complement: negating numbers

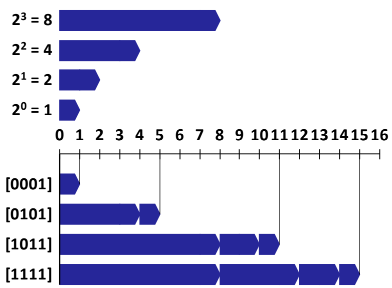
- **Subtract: Just subtraction from zero with wrap-around**
  - `-x == 0 - x`
- **Alternatively:**



Encoding Integers: Unsigned

$$B2U(X) = \sum_{i=0}^{w-1} x_i \cdot 2^i$$

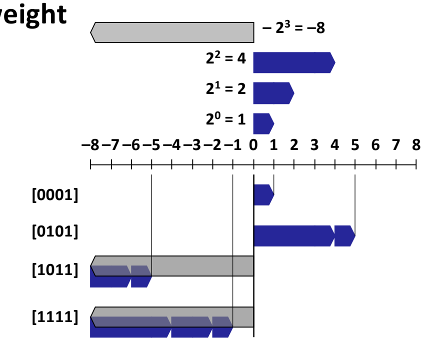
- **Each bit contributes its own "weight" to the sum**
  - **E.g., with w=4**



Encoding Integers: Signed

$$B2T(X) = -x_{w-1} \cdot 2^{w-1} + \sum_{i=0}^{w-2} x_i \cdot 2^i$$

- **Sign Bit**
  - For 2's complement, most significant bit indicates sign
  - 0 for nonnegative, 1 for negative
- **MSB has *negative* weight**
  - E.g., w=4



Exercise:

Compute the decimal value when the binary sequence is interpreted as an integer as unsigned vs. signed (using 3 bits)

Binary	Decimal value *unsigned*	Decimal value *signed*
000		
001		
010		
011		
100		
101		
110		
111		

Numeric Ranges for word size *w*

- **Unsigned Values**
  - ***UMax*** = $2^w - 1$
  - ***UMin*** = 0
- **Two's Complement Values**
  - ***TMax*** = $2^{w-1} - 1$
  - ***TMin*** = -2^{w-1}
- **Observations**
  - | *TMin* | = *TMax* + 1
    - Asymmetric range
  - *UMax* = 2 * *Tmax* + 1
- Examples for varying *w*

	W			
	8	**16**	**32**	**64**
UMax	255	65,535	4,294,967,295	18,446,744,073,709,551,615
TMax	127	32,767	2,147,483,647	9,223,372,036,854,775,807
TMin	-128	-32,768	-2,147,483,648	-9,223,372,036,854,775,808

Exercise

1. What is the two's complement encoding for the following decimal numbers? Your answer should be in binary using 8 bits.
   a. 42_{10}
   b. -105_{10}
2. What is the decimal equivalent for the following two's complement integers? Your answer should be in decimal.
   a. $1001\ 0101_2$
   b. $0101\ 0010_2$
3. Which of the following decimal numbers *cannot* be encoded in two's complement if the number of bits *w* is limited to 8?
   a. 250
   b. 128
   c. -128

Mapping Signed ↔ Unsigned

Example: w=4 bits

X	Signed: B2T(X)		Unsigned: B2U(X)	
0000	0		0	*UMin*
0001	1		1	
0010	2		2	
0011	3	=	3	
0100	4		4	
0101	5		5	
0110	6		6	
0111	7	*TMax*	7	
1000	-8	*TMin*	8	
1001	-7		9	
1010	-6	+/- 16	10	
1011	-5		11	
1100	-4		12	
1101	-3		13	
1110	-2		14	
1111	-1		15	*UMax*

Relation between Signed & Unsigned (Intuition)

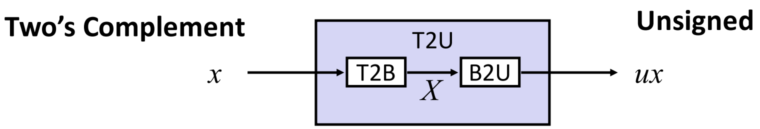
- Intuition: suppose w=4 bits, MSB is a 1

Unsigned	Signed (2's Compl)
1 0 1 0	1 0 1 0
weight: 2^{w-1} (positive); Positive weight	weight: -2^{w-1} (negative); Positive weight

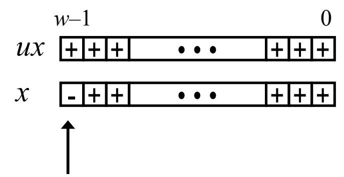
Difference = $2(2^{w-1}) = 2^w$

For w=4, difference = 16

Relation between Signed & Unsigned



Maintain Same Bit Pattern



Large negative weight
becomes
Large positive weight