

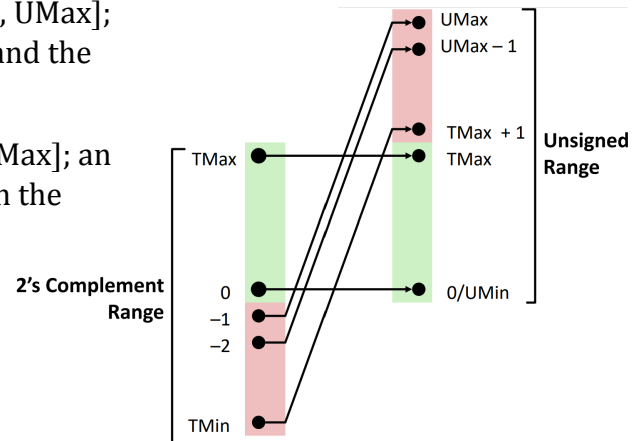
# CSE 351 Section 3 – Integers and Floating Point

Welcome back to section, we're happy that you're here ☺

## Integers and Arithmetic Overflow

Arithmetic overflow occurs when the result of a calculation can't be represented in the current encoding scheme (*i.e.*, it lies outside of the representable range of values), resulting in an incorrect value.

- **Unsigned overflow:** the result lies outside of  $[U_{\min}, U_{\max}]$ ; an indicator of this is when you add two numbers and the result is smaller than either number.
- **Signed overflow:** the result lies outside of  $[T_{\min}, T_{\max}]$ ; an indicator of this is when you add two numbers with the same sign and the result has the opposite sign.



### Exercises:

- 1) [Spring 2016 Midterm 1C] Assuming these are all signed two's complement 6-bit integers, compute the result of each of the following additions. For each, indicate if it resulted in overflow.

$$\begin{array}{r} 001001 \\ + 110110 \\ \hline \end{array}$$

$$\begin{array}{r} 110001 \\ + 111011 \\ \hline \end{array}$$

$$\begin{array}{r} 011001 \\ + 001100 \\ \hline \end{array}$$

$$\begin{array}{r} 101111 \\ + 011111 \\ \hline \end{array}$$

- 2) [Autumn 2019 Midterm 1C] Find the largest 8-bit unsigned numeral (answer in hex) such that  $c + 0x80$  causes NEITHER signed nor unsigned overflow in 8 bits.

# IEEE 754 Floating Point Standard

## Goals

- ★ Represent a large range of values (both very small and very large numbers),
- ★ Include a high amount of precision, and
- ★ Allow for real arithmetic results (*e.g.*,  $\infty$  and NaN).

## Encoding

The value of a real number can be represented in normalized scientific binary notation as:

$$(-1)^{\text{sign}} \times \text{Mantissa}_2 \times 2^{\text{Exponent}} = (-1)^S \times 1.M_2 \times 2^{E-\text{bias}}$$

The binary representation for floating point encodes these three components into separate fields:

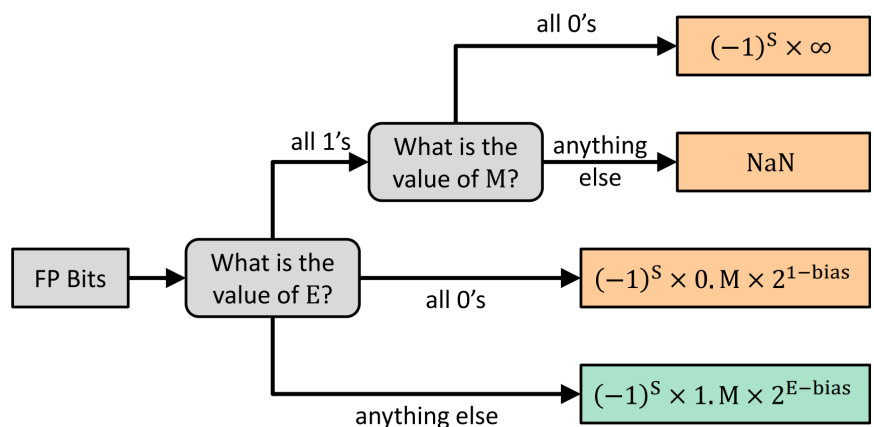
	S	E	M
float:	1	8 bits	23 bits
double:	1	11 bits	52 bits

- S: the sign of the number (0 for positive, 1 for negative)
- E: the exponent in biased notation (unsigned with a bias of  $2^{w-1}-1$ )
- M: the mantissa (also called the significand or fraction) *without* the implicit leading 1

## Special Cases and Interpretations

The interpretation depends on the values in the exponent and mantissa fields:

E	M	Meaning
0b0...0	anything	denormalized number (denorm)
anything else	anything	normalized number
0b1...1	zero	infinity ( $\infty$ )
0b1...1	nonzero	not-a-number (NaN)



## Mathematical Properties

- Not associative:  $(2 + 2^{50}) - 2^{50} \neq 2 + (2^{50} - 2^{50})$
- Not distributive:  $100 \times (0.1 + 0.2) \neq 100 \times 0.1 + 100 \times 0.2$
- Not cumulative:  $2^{25} + 1 + 1 + 1 + 1 \neq 2^{25} + 4$

### Exercises:

- 3) Let's say that we want to represent the number 3145728.125 (broken down as  $2^{21} + 2^{20} + 2^{-3}$ )
- a) Convert this number to into single precision floating point representation:

[illegible]

- b) Which limitation of floating point representation does this result highlight?

- 4) [Summer 2018 Midterm 1E-G] We are working with a new floating point datatype (`fl_o`) that follows the same conventions as IEEE 754 except using 8 bits split into the following fields:

Sign (1)	Exponent (3)	Mantissa (4)
----------	--------------	--------------

- What is the encoding of the most negative real number that we can represent ( $\infty$  is not a real number) in this floating point scheme (binary)?
- If we have `signed char x = 0b10101000 = -88`, what will occur if we cast `float f = (float) x` (i.e., try to represent the value stored in `x` as a `float`)?

## Rounding

## Underflow

## Overflow

None of these

- 5) Based on the floating point representation, explain why each of the three mathematical properties shown on the previous page occurs.

- a) Not associative:

- b) Not distributive:

- c) Not cumulative:

- 6) If we have `float x, y;`, give two *different* reasons why `(x+2*y)-y == x+y` might evaluate to false.