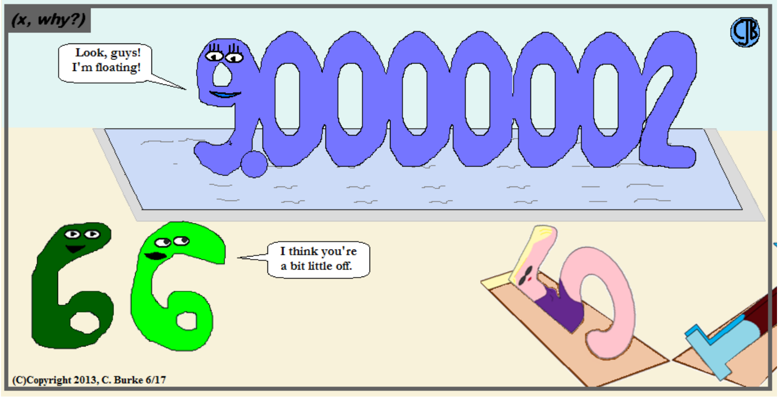
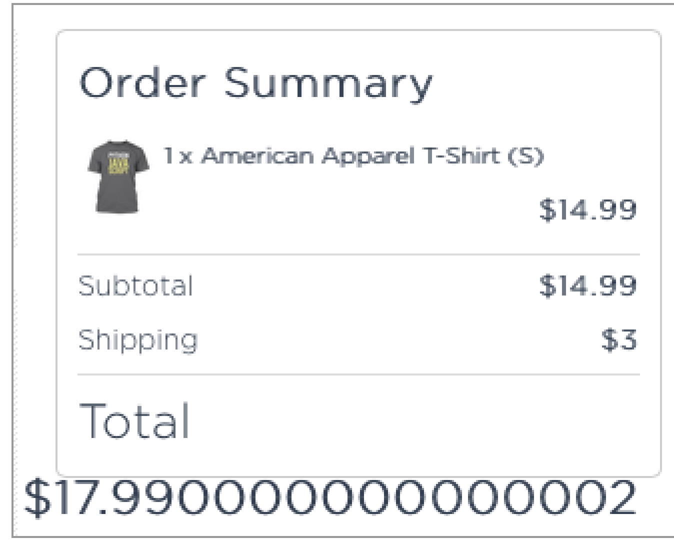


16. Floating Point Numbers

Chapter 10, section 10.4



FLOATING POINT NUMBERS

Floating Point

- Floating point is the formulaic representation that approximates a real number so support a trade-off between ***range*** and ***precision***

- Floating point representation is based on scientific notation
  - $0.55_{(10)}$ is equivalent to $5.5_{(10)} \text{ X } 10^{-1}$

- Use base 2 instead of base 10
  - $101.1_{(2)}$ is equivalent to $1.011_{(2)} \text{ X } 2^{2}$

https://en.wikipedia.org/wiki/Floating_point

Mapping Floating Point to Words

- How do we map a number in binary scientific notation onto a word?
- Use the following:
  - Sign: 1 for negative, 0 for positive
  - Exponent: the base (2) is raised to the exponent
  - Significand (mantissa): the digits in the number
- Notice that the base is implicit and not stored

Exponent

- Let's assume that the number of bits used to represent the exponent is 4 bits
- What is the range of values for the exponent?
- What is the problem with this?
- What is the solution? Biased Exponent!
  - Biased Exponent = True Exponent + Bias
  - Bias = $2^{k-1} - 1$, where k is the number of bits used to represent the exponent

Significand and Normalized Numbers

- Floating point numbers can be expressed in many ways:
  - 0.110×2^5
  - 110×2^2
  - 0.0110×2^6

- To simplify operations on floating point numbers, they must be normalized

- A normal number is one in which the most significant digit of the significand is nonzero
- Since the most significant digit in binary is always 1, we do not store it in floating point representation.

Floating Point Numbers

- General form for floating point numbers:

$$\pm 1.bbb \ldots b \times 2^{\pm E}$$

- Where:
  - b is a binary digit (0 or 1)
  - E is the exponent

Examples

- Fill in the following table:

Decimal	Binary	Normalized Floating Point	Sign (1-bit)	Biased Exponent (4-bits)	Significand (3 bits)
5.5	101.1	1.011×2^2			
-96					
			0	0101	100

Sign (1-bit)	Biased Exponent (4-bits)	Significand (3-bits)

REPRESENTABLE VALUES

Example Word

Sign (1-bit)	Biased Exponent (4-bits)	Significand (3-bits)

- What is the range for the biased exponent?

- What is the range for the significand?

Representable Numbers

Sign (1-bit)	Biased Exponent (4-bits)	Significand (3-bits)

- What is the smallest possible positive number that can be represented with example format

- What is the largest positive number that can be represented with the example format?

Representable Numbers

- Floating-point representations can't possibly represent all of the numbers in its range as there are only $2^8 = 256$ distinct values

- Example: Represent $51_{(10)}$ in this scheme

C++ Example

- Write a program that assigns 3.99 to a float variable
- Debug the program and look at what value is actually stored in the variable
- Is it what you expected?

Trade-offs

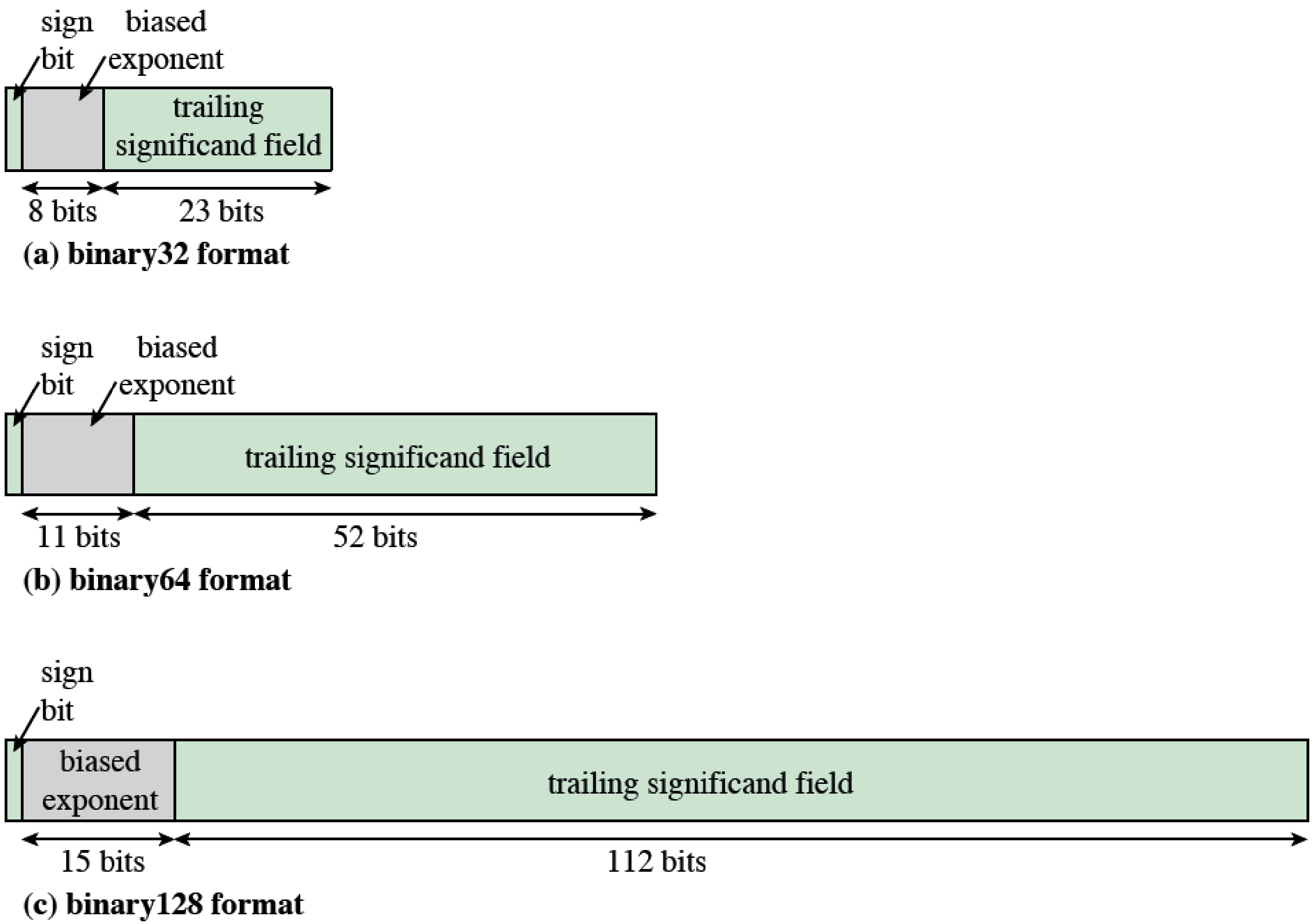
- Precision:
  - More significand bits == more precision
- Range:
  - More exponent bits == wider range of numbers to represent

IEEE STANDARD 754-2008

IEEE Standard 754 Floating-point Format

- IEEE 754 adopted in 1985 and revised in 2008
- There are 32-bit (single precision), 64-bit (double precision), and 128-bit (quadruple precision) representations
- Similar to the 8-bit format we've been using so far

IEEE 754-2008 Formats



IEEE 754-2008 Bias

- What is the value of the bias for each format:
  - Binary32 format
  - Binary64 format
  - Binary128 format

Examples

- What decimal value does $C03E0000_{(16)}$ represent in the IEEE 754 Binary32 format.

- What decimal value does $C03E000000000000_{(16)}$ represent in the IEEE 754 Binary64 format.

IEEE 754-2008

- It is important to note that not all bit patterns in the IEEE format are interpreted in the usual way. Here are some exceptions. Notice that an significand is called a fraction in IEEE 754 language.
  1. An exponent of 0 with a fraction of 0 represents +0 or -0 depending on the sign bit.
  2. An exponent of all 1's with a fraction of 0 represents positive or negative infinity.
  3. An exponent of all 1's with a nonzero fraction represents a NaN (not a number) and is used to represent various exceptions.
  4. Exponents in the range of 1-254 with normalized fractions implies the resulting exponent value will be in the range of -126 to +127. Since the number is normalized, we do not need to represent the 1. This bit is implied and called the hidden 1. It is actually a way of adding one more bit of precision to the fraction.

Examples

- What is the representation of each of the following in IEEE 754 Binary32 representation. Express your result in HEX.
  1. -1.0
  2. 1/32
  3. -14.5