CS271 - Computer Architecture and Assembly

D. Kevin McGrath

February 5, 2014
1 Floating-Point
Floating-point

- Representation for non-integral numbers
  - Useful for very large or very small numbers
- Like scientific notation
  - $-2.34 \times 10^{56}$
  - $+0.002 \times 10^{-4}$
  - $+987.02 \times 10^9$
- In binary
  - $\pm 1.xxxxxx_2 \times 2^{yyyy}$
- Types `float` and `double` in C
Floating-point standard

- Defined by IEEE Std. 754-1985
- Developed in response to divergence of representations
  - Portability issues for scientific code
- Almost universal in adoption
- Two representations
  - Single-precision (32-bit)
  - Double-precision (64-bit)
IEEE Floating-point format

- Single: 8 bits
- Single: 23 bits
- Double: 11 bits
- Double: 52 bits

<table>
<thead>
<tr>
<th>S</th>
<th>Exponent</th>
<th>Fraction</th>
</tr>
</thead>
</table>

\[ x = (-1)^S \times (1 + \text{Fraction}) \times 2^{\text{Exponent} - \text{Bias}} \]

- S: sign bit (0 ⇒ non-negative, 1 ⇒ negative)
- Normalize significand: \(1.0 \leq |\text{significand}| < 2.0\)
  - Always has a leading pre-binary-point 1 bit, so no need to represent it explicitly
  - Significand is fraction with the “1.” restored
- Exponent: excess representation: actual exponent + Bias
  - Ensures exponent is unsigned
  - Single-precision bias: 127
  - Double-precision bias: 1023
Single-precision range

- Exponents 00000000 and 11111111 reserved
- Smallest value:
  - Exponent: 00000001 ⇒ $1 - 127 = -126$
  - Fraction: 000...00 ⇒ significand = 1.0
  - $\pm 1.0 \times 2^{-126} \approx \pm 1.2 \times 10^{-38}$
- Largest value:
  - Exponent: 11111110 ⇒ $254 - 127 = +127$
  - Fraction: 111...11 ⇒ significand ≈ 2.0
  - $\pm 2.0 \times 2^{+127} \approx \pm 3.4 \times 10^{+38}$
Double-precision range

- Exponents 000...00 and 111...11 reserved
- Smallest value:
  - Exponent: 00000000001 ⇒ 1 − 1023 = −1022
  - Fraction: 000...00 ⇒ significand = 1.0
  - ±1.0 × 2−1022 ≈ ±2.2 × 10−308
- Largest value:
  - Exponent: 11111111110 ⇒ 2046 − 1023 = +1023
  - Fraction: 111...11 ⇒ significand ≈ 2.0
  - ±2.0 × 2+1023 ≈ ±1.8 × 10+308
Floating-point precision

- Relative precision
  - All fraction bits are significant
  - Single-precision: \( \approx 2^{-23} \)
    - Equivalent to \( 23 \times \log_{10} 2 \approx 23 \times 0.3 \approx 6 \) decimal digits
  - Double-precision: \( \approx 2^{-52} \)
    - Equivalent to \( 52 \times \log_{10} 2 \approx 52 \times 0.3 \approx 16 \) decimal digits
Floating-point example

• Represent \(-0.75\)
  • \(-0.75 = (-1)^1 \times 1.1_2 \times 2^{-1}\)
  • \(S = 1\)
  • Fraction = 1000...00_2
  • Exponent:
    • Single: \(-1 + 127 = 126 = 01111110_2\)
    • Double: \(-1 + 1023 = 1022 = 01111111110_2\)
  • Single: \textbf{1011111101000...00}
  • Double: \textbf{1011111111101000...00}
Floating-point example

- What number is represented by the single-precision float
  \[ 1100000010100 \ldots 00 \]
Floating-point example

- What number is represented by the single-precision float
  \[1100000010100\ldots00\]

-5.0
Denormal numbers

- Exponent = 000...00 ⇒ hidden bit is 0
  \[ x = (-1)^S \times (0 + \text{Fraction}) \times 2^{-\text{Bias}} \]

- Smaller than normal numbers
  - Allows for gradual underflow, with diminishing precision

- Denormal with fraction = 000...0
  \[ x = (-1)^S \times (0 + 0) \times 2^{-\text{Bias}} = \pm 0 \]

Two representations of 0!
Infinities and NaNs

- **Exponent = 111...1, fraction = 000...0**
  - $\pm\infty$
  - Can be used in subsequent calculations, avoiding need for overflow check
- **Exponent = 111...1, fraction \neq 000...0**
  - Not a number (NaN)
  - Indicates illegal or undefined result
  - Can be used in subsequent calculations
  - Propogate through all subsequent operations – result will be NaN
Accurate arithmetic

- IEEE 754 specifies additional rounding control
  - Extra bits of precision
  - Choice of rounding modes
  - Allows programmer to fine-tune numerical behavior
- Not all FPUs implement all options
  - Most programming languages and FP libraries just use defaults
- Trade-off between hardware complexity, performance, and market requirements
IEEE 754 features

- 4 rounding modes available
  - Round to nearest value – favors even numbers if equidistant
  - Round towards 0
  - Round towards $+\infty$
  - Round towards $-\infty$

- Four comparisons defined:
  - equal to
  - less than
  - greater than
  - unordered – used with NaNs

- Five exceptions
  - Invalid operation
  - Division by zero
  - Overflow
  - Underflow
  - Inexact