Floating Point Introduction

Topics

- Chapter 2.4
- IEEE Floating Point Standard

Standard Decimal Scientific Notation

- **Real numbers expressed as $x*10^y$**
  - e.g. $4.782*10^{27}$, and $-1.396*10^{-17}$, or 7.088e-6, or 3.14E10
- **Expansion:**
  - $4.782*10^{27} = 4*10^{27} + 7*10^{26} + 8*10^{25} + 2*10^{24}$

$$\text{decimal} = d_m d_{m-1} \ldots d_1 d_0 . d_{-1} d_{-2} \ldots d_{-n}$$

$$= \sum_{i=-n}^{m} d_i * 10^i$$

- **Not all numbers can be expressed exactly in base 10**
  - e.g. 1/3 = 0.33333…, so it must be approximated
- **Our goal is to represent real numbers using binary**
  - We follow the approach of decimal scientific notation except using base 2

IEEE Floating Point

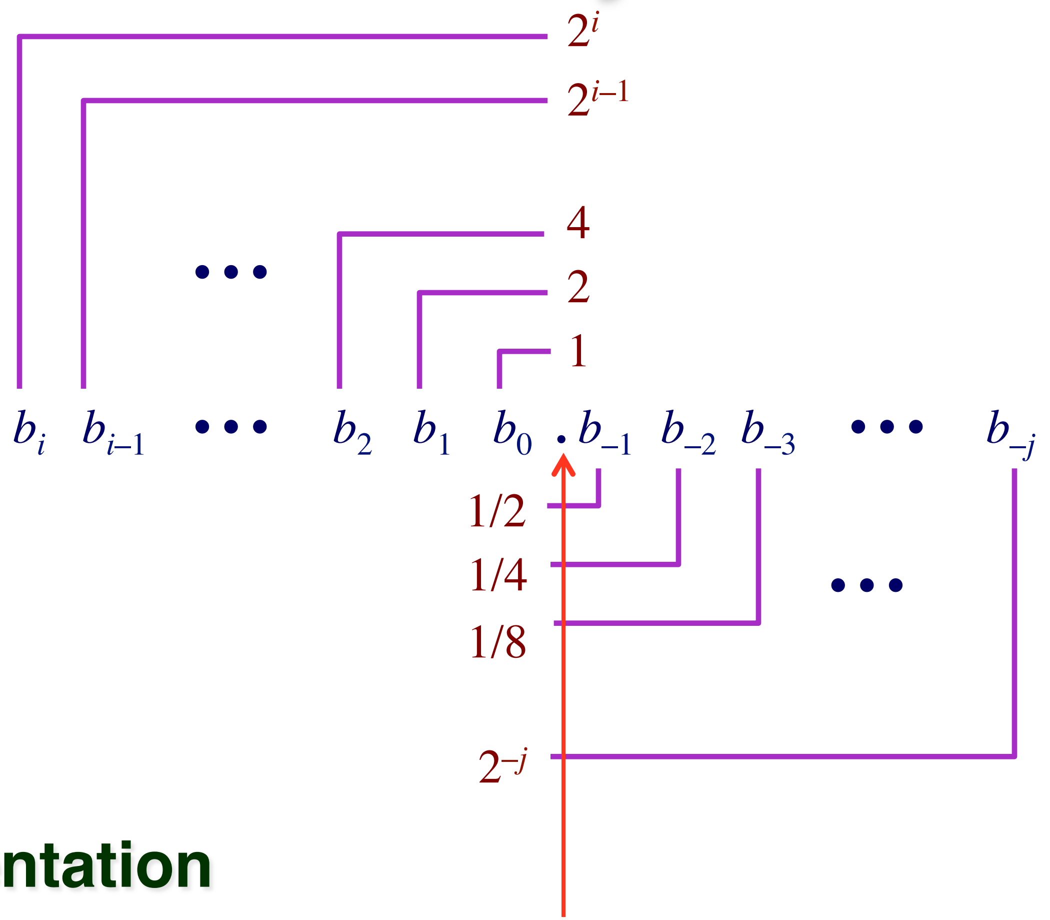
IEEE Standard 754

- Established in 1985 as uniform standard for floating point arithmetic
  - Before that, many idiosyncratic formats
- Supported by all major CPUs

Driven by Numerical Concerns

- Nice standards for rounding, overflow, underflow
- Hard to make go fast
  - Numerical analysts predominated over hardware types in defining standard

Fractional Binary Numbers



Representation

- Bits to right of "binary point" represent fractional powers of 2
- Represents rational number:

$$\sum_{k=-j}^{i} b_k \cdot 2^k$$

Fractional Binary Number Examples

Value / Representation

Value	Representation	
5 3/4	101.11_2	$= 2^2 + 2^0 + 2^{-1} + 2^{-2}$
2 7/8	10.111_2	$= 2^1 + 2^{-1} + 2^{-2} + 2^{-3}$
63/64	0.111111_2	$= 2^{-1} + 2^{-2} + 2^{-3} + 2^{-4} + 2^{-5} + 2^{-6}$

Observations

- Divide by 2 by shifting right
- Multiply by 2 by shifting left
- Numbers of form $0.111111\ldots_2$ just below 1.0
  - $1/2 + 1/4 + 1/8 + \ldots + 1/2^i + \ldots \rightarrow 1.0$
  - Use notation $1.0 - \varepsilon$

Representable Numbers

Limitation

- This style of binary point notation is not very good at representing larger numbers

e.g. 1010000000000000000000000000000000000000000000000…
(followed by a 100 zeros)

- Can only exactly represent numbers of the form $x/2^k$
- Other numbers have repeating bit representations

Value	Representation
1/3	$\texttt{0.0101010101[01]}\ldots_2$
1/5	$\texttt{0.001100110011[0011]}\ldots_2$
1/10	$\texttt{0.0001100110011[0011]}\ldots_2$

Floating Point Representation

Numerical Form

- Real number = $(-1^s) * M * 2^E$
  - Sign bit *s* determines whether number is negative or positive
  - Significand *M* normally a fractional value in range [1.0,2.0).
  - Exponent *E* weights value by power of two

Encoding

`s`	`exp`	`frac`

- MSB is sign bit
- `exp` field encodes *E*
- `frac` field encodes *M*

Floating Point Precisions

s	exp	frac

Encoding

- MSB is sign bit
- `exp` field encodes *E*
- `frac` field encodes *M*

$$\text{Real number} = (-1^{s}) * M * 2^{E}$$

Sizes

- Single precision: 8 `exp` bits, 23 `frac` bits
  - 32 bits total. Can represent from 2^{127} (1.7e38) down to 2^{-126}
- Double precision: 11 `exp` bits, 52 `frac` bits
  - 64 bits total
- Extended precision: 15 `exp` bits, 63 `frac` bits
  - Only found in Intel-compatible machines
  - Stored in 80 bits - 1 bit wasted
- Quad Precision (IEEE 754r - revised) - 15 `exp`, 112 `frac`
  - 128 bits total
- Half Precision (IEEE 754r) - 5 `exp`, 10 `frac`
  - 16 bits total