CSC 411

Computer Organization (Spring 2024)
Lecture 7: Floating Point

Prof. Marco Alvarez, University of Rhode Island

Fractional binary numbers

- Bits to the right of binary point
  - fractional powers of 2
  - don't worry about negatives for now

2^i	2^{i-1}		2	1	$1/2$	$1/4$		2^{-j+1}	2^{-j}
b_i	b_{i-1}	…	b_1	b_0	b_{-1}	b_{-2}	…	b_{-j+1}	b_{-j}

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

`11.010 =`

`1101.101 =`

Practice

- Convert fractional binary numbers to decimal

```
 1.10 =
11.001 =
100.01 =
11.111 =
```

Observations

- Not all decimal fractions have exact binary equivalents
  - can only represent numbers of the form $\frac{x}{2^k}$
    - e.g., 1/5 and 1/10
- Limited precision due to finite number of bits
  - can't easily represent very small or very large values
- 0.111111… represents a number just below 1.0

$$\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots \frac{1}{2^i} + \ldots = 1.0 - \varepsilon$$

IEEE Floating Point

- IEEE standard 754
  - defines a common format for representing real numbers in computers
    - developed in response to divergence of representations
  - supported by major CPUs (almost universally adopted)
  - provides different precision levels (single, double, extended) for various needs
  - standardizes sign, exponent, and fraction components

sign	exponent	fraction

Floating point representation

- Numerical form

$$(-1)^s M 2^E$$

  - **sign bit s**: 0 for positive, 1 for negative
  - **exponent E**: magnitude of the number (power of 2)
    - encoded in exp
  - **significand M**: captures the fractional part, scaled by the exponent, normally in range $[1.0, 2.0)$
    - encoded in frac

s	exp	frac

Precision options

- Single-precision (32 bits)
  - good balance of performance and range (7 decimal digits)

s	exp	frac
1	8 bits	23 bits

- Double-precision (64 bits)
  - higher precision (15-17 decimal digits) for demanding calculations

s	exp	frac
1	11 bits	52 bits

- Others
  - half precision, quad precision

Normalized and denormalized numbers

- Normalized
  - maximizes precision
  - **exp** != 000…000 and **exp** != 111…111
- Denormalized
  - used for very small numbers, reducing precision
  - **exp** == 000…000
- Special
  - **exp** == 111…111

Normalized values

$(-1)^s M 2^E$

- Exponent **E**
  - E = exp - bias
  - bias is $2^{k-1} - 1$, where k is the number of bits in exp
    - single precision, bias = 127
    - double precision, bias = 1023
- Significand **M**
  - assume xx…xx are the bits in frac
  - M = 1.xx…xx
    - always with an implied leading 1
    - minimum?
    - maximum?

Encoding example

$(-1)^s M 2^E$

- Assume a float F = 2024.0
  - fractional binary: 11111101000
  - make it form 1.xxx…: 1.1111101000 x 2^{10}
    - M = 1.1111101000
    - frac = 11111010000000000000000
  - calculate exp using E = exp - bias
    - exp = E + bias = 10 + 127 = 137 = 10001001

0	1	0	0	0	1	0	0	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0x44FD0000

Decoding example

$(-1)^s M 2^E$

- Assume a float F = 0x43CDA000
  - write the binary
  - extract s, exp, frac
  - calculate M and E
  - write final number