Aritmética computacional

5. Representação de ponto flutuante

Ponto flutuante

- Representação de números não inteiros
  - Incluindo muito pequenos e muito grandes
- Parte da notação científica
  - -2.34×10^{56} ← normalizado
  - $+0.002 \times 10^{-4}$ ← não normalizado
  - $+987.02 \times 10^{9}$ ← não normalizado
- Em binário:
  - $\pm 1.xxxxxxx_2 \times 2^{yyyy}$
- Tipos `float` e `double` em C

Padrão de ponto flutuante

- Definido pela norma IEEE 754-1985
- Desenvolvido para padronizar as representações
  - Problemas de portabilidade para código científico
- Atualmente, é adotado mundialmente
- Duas representações
  - Precisão simples (32-bit)
  - Precisão dupla (64-bit)

Formato de ponto flutuante

simples: 8 bits duplo: 11 bits		simples: 23 bits duplo: 52 bits
S	Exponent	Fraction

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

- S: bit de sinal (0 $\Rightarrow$ não negativo, 1 $\Rightarrow$ negativo)
- Significando normalizado: $1.0 \le |\text{significando}| < 2.0$
  - Fração: o que aparece à direita do ponto
  - Significando: o número completo, incluindo o 1.
  - Sempre possui o bit 1 à esquerda do ponto binário, portanto este não precisa ser representado explicitamente (bit escondido)
- Expoente: representação por excesso: expoente verdadeiro + Bias
  - Garante que o expoente é sempre sem sinal
  - Precisão simples: Bias = 127; precisão dupla: Bias = 1023

Capacidade da precisão simples

- Expoentes 00000000 e 11111111 são reservados para uma representação especial
- Menor valor representável
  - Exponent: 00000001
    $\Rightarrow$ expoente verdadeiro = 1 – 127 = –126
  - Fração: 000…00 $\Rightarrow$ significando = 1.0
  - $\pm 1.0 \times 2^{-126} \approx \pm 1.2 \times 10^{-38}$
- Maior valor
  - expoente: 11111110
    $\Rightarrow$ expoente verdadeiro = 254 – 127 = +127
  - Fração: 111…11 $\Rightarrow$ significando ≈ 2.0
  - $\pm 2.0 \times 2^{+127} \approx \pm 3.4 \times 10^{+38}$

Capacidade da precisão dupla

- Expoentes 0000…00 e 1111…11 são reservados para uma representação especial
- Menor valor
  - Expoente: 00000000001
    $\Rightarrow$ expoente verdadeiro = 1 – 1023 = –1022
  - Fração: 000…00 $\Rightarrow$ significando = 1.0
  - $\pm 1.0 \times 2^{-1022} \approx \pm 2.2 \times 10^{-308}$
- Maior valor
  - Expoente: 11111111110
    $\Rightarrow$ expoente verdadeiro = 2046 – 1023 = +1023
  - Fração: 111…11 $\Rightarrow$ significando $\approx$ 2.0
  - $\pm 2.0 \times 2^{+1023} \approx \pm 1.8 \times 10^{+308}$

Precisão de um ponto flutuante

- Precisão relativa
  - Todos os bits da fração são significantes
  - Precisão simples: aprox. 2^{-23}
    - Equivalente a $23 \times \log_{10}2 \approx 23 \times 0.3 \approx 6$ casas decimais de precisão
  - Precisão dupla: aprox. 2^{-52}
    - Equivalente a $52 \times \log_{10}2 \approx 52 \times 0.3 \approx 16$ casas decimais de precisão

Exemplo de ponto flutuante

- Representar –0.75
  - $-0.75 = (-1)^1 \times 1.1_2 \times 2^{-1}$
  - S = 1
  - Fração = $1000\ldots00_2$
  - Expoente = –1 + Bias
    - Simples: $-1 + 127 = 126 = 01111110_2$
    - Dupla: $-1 + 1023 = 1022 = 01111111110_2$
- Simples: 1011111101000…00
- Dupla: 1011111111101000…00

Exemplo de ponto flutuante

- Que número é representado pelo ponto flutuante de precisão simples:

  11000000101000…00

  - S = 1
  - Fração = $01000\ldots00_2$
  - Expoente = $10000001_2 = 129$
- $x = (-1)^1 \times (1 + 01_2) \times 2^{(129 - 127)}$

  $= (-1) \times 1.25 \times 2^2$

  $= -5.0$

Infinito e NaNs

- Expoente = 111...1, Fração = 000...0
  - ±Infinito
  - Pode ser usado em cálculos para evitar a checagem de *overflow*
- Expoente = 111...1, Fração ≠ 000...0
  - Not-a-Number (NaN)
  - Indica uma operação ilegal ou um resultado não definido
    - e.g., 0.0 / 0.0

Representações IEEE 754

O padrão IEEE 754 especifica as seguintes representações especiais.

Precisão simples		Precisão dupla		Representação
Expoente	Fração	Expoente	Fração	
0	0	0	0	0
0	não zero	0	não zero	$\pm$ número desnormalizado
1-254	qualquer coisa	1-2046	qualquer coisa	$\pm$ número de ponto flutuante
255	0	2047	0	$\pm$ infinito
255	não zero	2047	não zero	NaN (Not a Number)

Adição em ponto flutuante

- Exemplo:
  - $9.999 \times 10^{1} + 1.610 \times 10^{-1}$
- 1. Alinhar o ponto decimal (igualar os expoentes)
  - Dar shifts à direita no número com menor expoente: $9.999 \times 10^{1} + 0.016 \times 10^{1}$
- 2. Somar os significandos
  - $9.999 \times 10^{1} + 0.016 \times 10^{1} = 10.015 \times 10^{1}$
- 3. Normalizar o resultado
  - 1.0015×10^{2}
- 4. Arredondar e renormalizar, se necessário
  - 1.002×10^{2}

Adição em ponto flutuante

- Exemplo 2:
  - $1.000_2 \times 2^{-1} + -1.110_2 \times 2^{-2}$ (0.5 + –0.4375)
- 1. Alinhar o ponto decimal
  - Dar shifts à direita no número com menor expoente
  - $1.000_2 \times 2^{-1} + -0.111_2 \times 2^{-1}$
- 2. Somar os significandos
  - $1.000_2 \times 2^{-1} + -0.111_2 \times 2^{-}1 = 0.001_2 \times 2^{-1}$
- 3. Normalizar o resultado
  - $1.000_2 \times 2^{-4}$, sem underflow/overflow
- 4. Arredondar e renormalizar, se necessário
  - $1.000_2 \times 2^{-4}$ (sem mudanças) = 0.0625

Multiplicação em ponto flutuante

- Exemplo
  - $1.110 \times 10^{10} \times 9.200 \times 10^{-5}$
- 1. Soma os expoentes
  - Novo expoente = 10 + –5 = 5
- 2. Multiplica os significandos
  - $1.110 \times 9.200 = 10.212 \Rightarrow 10.212 \times 10^{5}$
- 3. Normaliza o resultado e checa por overflow/underflow
  - 1.0212×10^{6}
- 4. Arredonda e renormaliza, se necessário
  - 1.021×10^{6}
- 5. Determina o sinal do resultado a partir dos sinais dos operandos
  - $+1.021 \times 10^{6}$

Multiplicação em ponto flutuante

- Exemplo
  - $1.000_2 \times 2^{-1} \times -1.110_2 \times 2^{-2}$ (0.5×-0.4375)
- 1. Soma os expoentes
  - Sem o bias: $-1 + -2 = -3$
  - Com o bias: $(-1 + 127) + (-2 + 127) = -3 + 254 - 127 = -3 + 127$
- 2. Multiplica os significandos
  - $1.000_2 \times 1.110_2 = 1.1102 \Rightarrow 1.110_2 \times 2^{-3}$
- 3. Normaliza o resultado e checa por over/underflow
  - $1.110_2 \times 2^{-3}$ sem overflom
- 4. Arredonda e renormaliza
  - $1.110_2 \times 2^{-3}$ sem arredondamento
- 5. Determina o sinal: +ve × –ve ⇒ –ve
  - $-1.110_2 \times 2^{-3} = -0.21875$

Instruções em MIPS

- Há um hardware próprio para ponto flutuante: o coprocessador 1
  - Processador adjunto que expande a ISA
- Registradores separados
  - 32 registradores de 32 bits: \$f0, \$f1, … \$f31
  - Precisão dupla aos pares: \$f0/\$f1, \$f2/\$f3, …
- As instruções que vimos até agora não funcionam no coprocessador 1
  - O coprocessador 1 possui suas instruções próprias
- Instruções de acesso à memória
  - `load word` e `load double`
  - `lwc1`, `ldc1`, `swc1`, `sdc1`
    - e.g., `ldc1 $f8, 32($sp)`

Instruções em MIPS

- Aritmética de precisão simples
  - `add.s`, `sub.s`, `mul.s`, div.s
    - e.g., `add.s $f0, $f1, $f6`
- Aritmética de precisão dupla
  - `add.d`, `sub.d`, `mul.d`, `div.d`
    - e.g., `mul.d $f4, $f4, $f6`
- Comparação
  - `c.`*xx*`.s`, `c.`*xx*`.d` (*xx* is eq, `lt`, `le`, …)
  - Recebe dois operações, e define o bit de condição
    - e.g. `c.lt.s $f3, $f4`
- Faz o desvio baseado no bit de condição
  - `bc1t`, `bc1f`
    - e.g., `bc1t TargetLabel`

Exemplo: °F to °C

- Código C:

```
float f2c (float fahr) {
  return ((5.0/9.0)*(fahr - 32.0));
}
```