Capítulo 2

João Lourenço
Joao.Loourenco@di.fct.unl.pt
Faculdade de Ciências e Tecnlogia
Universidade Nova de Lisboa
2006-2007

Representação de dados em sistemas computacionais

2

Adaptado dos transparentes das autoras do livro "*The Essentials of Computer Organization and Architecture*"

Números inteiros

- Os computadores foram feitos para manusear dados
- Que números se podem guardar em N bits?
  - Inteiros sem sinal:
    0 a $2^N - 1$
  - Inteiros com sinal (complemento p/ 2)
    $-2^{(N-1)}$ a $2^{(N-1)} - 1$

2

Representação de números inteiros (com sinal)

- Sinal = bit de maior peso
  - 0 = positivo, 1 = negativo
- Complemento para um
  - Para negar, fazer complemento binário
- Complemento para dois
  - Para negar, fazer complemento binário e adicionar 1
  - Pode ser usado directamente em operações matemáticas

3

Outros números

- E os outros números?
  - Números muito grandes: [seg./século]
    $3,155,760,000_{10}$ ($3.15576_{10} \times 10^9$)
  - Números muito pequenos: [diâmetro atómico]
    0.00000001_{10} ($1.0_{10} \times 10^{-8}$)
  - Números racionais: [padrões de repetição]
    $2/3_{10}$ ($0.666666\ldots_{10}$)
  - Números irracionais: [sem padrões de repetição]
    $2^{1/2}$ ($1.414213562373\ldots$)
  - Números transcendentais: [e, π]
    $e = 2.718\ldots$ $\pi = 3.1415926\ldots$
- Representáveis em notação científica (aprox.)

4

Notação científica

mantissa **expoente**

$$\mathbf{6{,}02_{10} \times 10^{23}}$$

ponto decimal **base**

- Formato normalizado: um e um só dígito (≠0) à esquerda do ponto
- Alternativas para representar 1/1,000,000,000
  - Não normalizado: 0.1×10^{-8}, 10.0×10^{-10}
  - Normalizado: 1.0×10^{-9}

5

Representação de números em vírgula flutuante

- Usados para representar números reais e grandes inteiros
- Usa-se a notação científica

$$n = m * 2^{e}$$

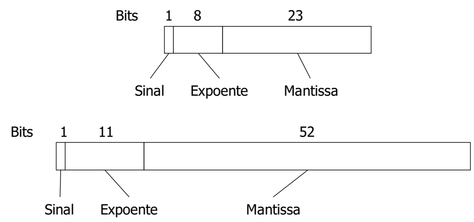
6

Norma IEEE 754

- Precisão simples (32 bits)
  - 1 bit sinal
  - 8 bits para expoente (excesso 127)
  - 23 bits para mantissa
- Dupla precisão (64 bits)
  - 1 bit sinal
  - 11 bits para expoente (excesso 1023)
  - 52 bits para mantissa

7

Formato Norma IEEE 754



Formato IEEE para vírgula flutuante.
(a) Precisão simples (b) Precisão dupla

8

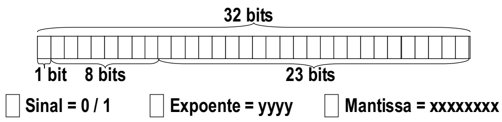
Formato normalizado para vírgula flutuante

- Um número binário normalizado começa necessariamente com um bit a 1
- Logo, pode-se assumir que existe sempre um bit a 1 seguido do ponto decimal
- Logo, apenas a parte fraccionária necessita de ser guardada
- Os expoentes mínimos e máximos não são usados nos números normalizados, estando reservados para significados especiais

9

Representação de números em vírgula flutuante [1]

- $+1.xxxxxxxx_2 * 2^{yyyy_2}$
- Múltiplos tam. da palavra (e.g., 32 bits)



- Representa números tão pequenos como 2 x 10^{-38} e tão grandes como 2 x 10^{38}

10

Representação de números em vírgula flutuante [2]

- E se a operação resultar num número demasiado grande?
  (> 2.0×10^{38})
- *Overflow*
  - Expoente positivo é maior que o número representável nos 8 bits reservados para o efeito

11

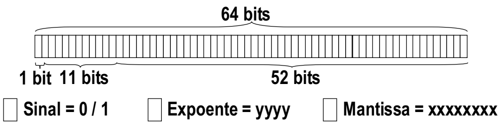
Representação de números em vírgula flutuante [3]

- E se a operação resultar num número muito pequeno?
  (> 0 e < 2.0×10^{-38})
- *Underflow*
  - Expoente negativo é menor que o número representável nos 8 bits reservados para o efeito

12

Representação de números em vírgula flutuante [4]

- Como evitar *overflows* e *underflows*?
- Utiliza-se dupla precisão
  - 64 bits para representar o número



13

Representação de números em vírgula flutuante [5]

	Precisão Simples	Dupla Precisão
Declaração em C	float	double
Tamanho	4 bytes	8 bytes
Maior número	$\approx\pm 2.0 \times 10^{38}$	$\approx\pm 2.0 \times 10^{308}$
Menor número	$\approx\pm 2.0 \times 10^{-38}$	$\approx\pm 2.0 \times 10^{-308}$

14

Norma IEEE 754 para números em vírgula flutuante [1]

- Mesmos princípios para precisão simples e dupla
- Bit de sinal: 1 = negativo
  0 = positivo
- Expoente: por excesso de 127 ou 1023
- Mantissa
  - Para englobar mais dígitos, o 1 inicial está implícito
    - 1+23 bits em precisão simples
    - 1+52 bits em dupla precisão
  - Corresponde sempre a um valor < 1

15

Norma IEEE 754 para números em vírgula flutuante [2]

- O zero não tem 1 inicial!
  - Reserva-se o valor particular de expoente zero para representar o número 0
- Representação foi pensada para que os cálculos fossem eficientes mesmo sem FPU (*Floating Point Unit*)

16

Norma IEEE 754 para números em vírgula flutuante [3]

- Como comprar dois números VF sem FPU?
  - Particiona-se o número em 3 partes:
    sinal | expoente | mantissa
    e comparam-se por esta ordem
- Para ser rápido, deveria funcionar com comparação inteira (com sinal)

17

Norma IEEE 754 para números em vírgula flutuante [4]

- Surgiu assim a convenção...
  - Bit de maior peso é o sinal
    (negativo < positivo)
  - Bits seguintes são o expoente
    (> expoente $\Rightarrow$ > número)
  - Bits de menor peso contêm a mantissa
    (mesmo expoente, > mantissa $\Rightarrow$ > número)

18

Norma IEEE 754 para números em vírgula flutuante [5]

- Expoente negativo! Como representar?
- Complemento para 2?
  1.0 x 2^{-1} vs. 1.0 x2^{+1} (1/2 vs. 2)

½	0	1111 1111	000 0000 0000 0000 0000 0000
2	0	0000 0001	000 0000 0000 0000 0000 0000

- Nesta notação, a comparação inteira deste números resulta em ½ > 2

19

Norma IEEE 754 para números em vírgula flutuante [6]

- Usar notação alternativa onde:
  - 0000 0001 é o menor negativo
  - 1111 1111 é o maior positivo

½	0	0111 1110	000 0000 0000 0000 0000 0000
2	0	1000 0000	000 0000 0000 0000 0000 0000

- Nesta notação, a comparação inteira deste números resulta em ½ < 2

20

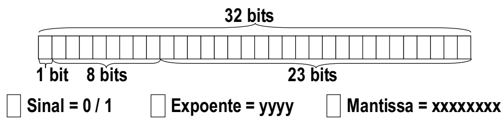
Norma IEEE 754 para números em vírgula flutuante [7]

- Chamada "Notação por excesso de N"
  - IEEE 754 usa um excesso de 127 para precisão simples
    - Subtrai-se 127 ao campo expoente, para determinar o valor efectivo deste
  - O valor de 1023 é usado como excesso para precisão dupla

21

Norma IEEE 754 para números em vírgula flutuante [8]

- Resumo (precisão simples)
  - $(-1)^S$ x (1 + mantissa) x 2 $^{(\text{expoente - 127})}$



- Precisão dupla
  - Idêntico mas com excesso de 1023
    - $(-1)^S$ x (1 + mantissa) x $2^{(\text{expoente - 1023})}$

22

O "Pai" da norma IEEE 754

Norma IEEE 754 para Aritmética Binária em Vírgula Flutuante

1989
ACM Turing
Award Winner!

Prof. Kahan

www.cs.berkeley.edu/~wkahan/
…/ieee754status/754story.html

23

Compreendendo a mantissa [1]

- Método 1: usando fracções
  - Em decimal:
    - 0.340_{10} ➔ $340_{10}/1000_{10}$
      ➔ $34_{10}/100_{10}$
  - Em binário:
    - 0.110_2 ➔ $110_2/1000_2 = 6_{10}/8_{10}$
      ➔ $11_2/100_2 = 3_{10}/4_{10}$

24

Compreendendo a mantissa [2]

- Método 2: calculando o valor
  - Conversão da notação científica
    - Em decimal:
      - $1.6732 = (1 \times 10^{0}) + (6 \times 10^{-1}) + (7 \times 10^{-2}) + (3 \times 10^{-3}) + (2 \times 10^{-4})$
    - Em binário:
      - $1.1001 = (1 \times 2^{0}) + (1 \times 2^{-1}) + (0 \times 2^{-2}) + (0 \times 2^{-3}) + (1 \times 2^{-4})$
  - Método bom para calcular rapidamente o valor da mantissa

Exemplo: números em v.f. Binário ➔ Decimal [1]

$$\underbrace{0}_{\text{sinal}}\,\underbrace{0\,1\,1\,0\,1\,0\,0\,0}_{\text{expoente}}\,\underbrace{1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,0\,1\,0}_{\text{mantissa}}$$

- Sinal: 0 ➔ Positivo
- Expoente: $0110\ 1000_2$ ➔ 104_{10}
  - Ajuste do excesso: 104 – 127 = -23
- Mantissa:
  - $1 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} + 0 \times 2^{-4} + 1 \times 2^{-5} + \ldots$
    $= 1 + 2^{-1} + 2^{-3} + 2^{-5} + 2^{-7} + 2^{-9} + 2^{-14} + 2^{-15} + 2^{-17} + 2^{-22}$
    $= 1.0 + 0.666115$
- Representa: $1.666115_{10} \times 2^{-23} \approx 1.986 \times 10^{-7}$
  (aprox. 2/10,000,000)

Exemplo: números em v.f. Binário ➔ Decimal [2]

- Caso simples
  - Número é representável por uma fracção cujo denominador é uma potência de dois
- Representação de -0.75
  - $-0.75 = -3/4 = -11_2/100_2 = -0.11_2$
  - Normalizado: $-1.1_2 \times 2^{-1}$
  - $(-1)^S \times (1 + \text{mantissa}) \times 2^{(\text{expoente}-127)}$
  - $(-1)^1 \times (1 + .100\ 0000\ \ldots\ 0000) \times 2^{(126-127)}$

$$\underbrace{1}_{\text{sinal}}\,\underbrace{0\,1\,1\,1\,1\,1\,1\,0}_{\text{expoente}}\,\underbrace{1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0}_{\text{mantissa}}$$

Exemplo: números em v.f. Binário ➔ Decimal [3]

- Caso menos simples
  - Número não é representável por uma fracção cujo denominador é uma potência de dois
- Então o número exacto não pode ser representado
- Usa-se uma aproximação do número
  - É aqui que o número de bits da mantissa se torna importante
- Uma vez obtida a mantissa, é fácil normalizar o número e calcular o expoente
- Como obter a mantissa de um número?

Exemplo: números em v.f. Binário ➔ Decimal [4]

- Todos os números racionais têm um padrão de repetição, quando escritos em decimal
- A lei acima também é válida para números escritos em binário
- Procedimento:
  1. Escrever o número em binário com o padrão de repetição
  2. Cortar o bits em excesso (depende da precisão)
  3. Determinar o sinal, expoente e mantissa

Exemplo: números em v.f. Binário ➔ Decimal [5]

- Como representar 1/3?
- 1/3
  - $= 0.33333\ldots_{10}$
  - $= 0.25 + 0.0625 + 0.015625 + 0.00390625 + 0.0009765625 + \ldots$
  - $= 1/4 + 1/16 + 1/64 + 1/256 + 1/1024 + \ldots$
  - $= 2^{-2} + 2^{-4} + 2^{-6} + 2^{-8} + 2^{-10} + \ldots$
  - $= 0.0101010101\ldots_2 \times 2^{0}$
  - $= 1.0101010101\ldots_2 \times 2^{-2}$ (formato normalizado)

Exemplo: números em v.f. Binário ➔ Decimal [6]

- Sinal = 0
- Expoente = -2 + 127 = 125_{10} =
  = 01111101
- Mantissa = 0101010101…

0	01111101	01010101010101010101010
sinal	expoente	mantissa

Representação de +/- Infinito

- Em FP, a divisão por zero deveria resultar em +/- Infinito, não *overflow*
- Porquê?
  - É possível fazer mais computações com Infinito, e.g., (x/0 > y) pode ser uma comparação válida
- IEEE 754 representa +/- Infinito
  - Maior expoente positivo reservado para representar Infinito
  - Mantissa toda a zeros

Representação de Zero

- Como representar 0?
  - Expoente a zeros
  - Mantissa também a zeros
  - E o sinal?
    - `+0: 0 00000000 00000000000000000000000`
    - `-0: 1 00000000 00000000000000000000000`
- Porquê dois zeros?
  - Ajuda na comparação de alguns limites
  - -0 ≠ +0 (zero negativo ≠ zero positivo)

Números em vírgula flutuante [1]

Item	Single precision	Double precision
Bits in sign	1	1
Bits in exponent	8	11
Bits in fraction	23	52
Bits, total	32	64
Exponent system	Excess 127	Excess 1023
Exponent range	−126 to +127	−1022 to +1023
Smallest normalized number	2^{-126}	2^{-1022}
Largest normalized number	approx. 2^{128}	approx. 2^{1024}
Decimal range	approx. 10^{-38} to 10^{38}	approx. 10^{-308} to 10^{308}
Smallest denormalized number	approx. 10^{-45}	approx. 10^{-324}

Figure B-5. Characteristics of IEEE floating-point numbers.

Números em vírgula flutuante [2]

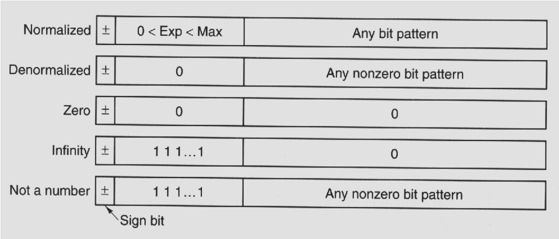


Figure B-6. IEEE numerical types.

Exercício

- Qual é o equivalente decimal de
  1100 0000 1111 0000 0000 0000 0000 0000

A:	-3.5	D:	-7.5	G:	-7.75
B:	-3.75	E:	+3.75	H:	-15.0
C:	-7.0	F:	-7 x 2^{129}	I:	-129.75

Exercício: Solução

$\underbrace{1}_{\text{sinal}}\underbrace{100\ 0000\ 1}_{\text{expoente}}\underbrace{111\ 0000\ \ 0000\ 0000\ 0000\ 0000}_{\text{mantissa}}$

- $(-1)^S \times (1 + \text{mantissa}) \times 2^{(\text{expoente - excesso})}$
- $(-1)^1 \times (1 + .111) \times 2^{(129 - 127)}$
- $-1 \times 1.111 \times 2^2$
- -1×111.1
- -111.1_2
- -7.5_{10} (opção D:)

37

Tópicos a recordar

- Números em VF aproximam os números que queremos usar
- Norma IEEE 754 reúne o maior consenso sobre como representar tais números
  - Todos os computadores vendidos desde ~1997 usam esta convenção
- Os dados não têm um tipo associado
- Os bits só têm significado dado o seu contexto

38