Lógica Digital

Sistemas de Numeração

João Paulo de Campos Carvalho
Revisão: Eliseu C. Miguel

Universidade Federal de Alfenas

July 9, 2021

Tópicos

Tópicos

Tópicos

Bases Numéricas

Definições informais sobre Bases Numéricas.

Podemos considerar, a fim de simplificação, que base numérica é um conjunto de símbolos (ou algarismos) com o qual podemos representar uma certa quantidade ou número.

http://www.dainf.ct.utfpr.edu.br/~robson/prof/aulas/common/bases

De acordo com o sítio Matematics LibreText, *A number base is the number of digits or combination of digits that a system of counting uses to represent numbers. A base can be any whole number greater than 0.*

https://math.libretexts.org/Courses/Mount_Royal_University/MATH_2150%3A_Higher_Arithmetic/7%3A_Number_systems/7.2%3A_Number_Bases

Bases Numéricas: Representação de números

Seja o número V representado na base numérica B. A representação de V pode ser definida como:

- $V_{(B)} = v_n \ v_{n-1} \ \ldots \ v_i \ \ldots \ v_1 \ v_0$
  - Que se lê como:
    A representação de V na base B é a sequência de símbolos v_i tal que $n \geq i \geq 0$
  - E os valores de v_i são definidos no intervalo:
    $B > v_i \geq 0$

Exemplos:

- seja $V = 10011011_{(2)}$, então $B = 2$ e $v_i \in \{0, 1\}$
- seja $V = 1032_{(4)}$, então $B = 4$ e $v_i \in \{0, 1, 2, 3\}$
- seja $V = 10011011_{(8)}$, então $B = 8$ e $v_i \in \{0, 1, 2, 3, 4, 5, 6, 7\}$

Bases Numéricas: Conversão da base B para a base decimal

Seja o número $V_{(B)} = v_n \ v_{n-1} \ ... \ v_i \ ... \ v_1 \ v_0$

A conversão de $V_{(B)}$ para $Z_{(10)}$ é definida como:

$Z_{(10)} = v_n \ x \ B^n + v_{n-1} \ x \ B^{n-1} + ... + v_i \ x \ B^i + ... + v_1 \ x \ B^1 + v_0 \ x \ B^0$
ou seja:

$$Z_{(10)} = \sum_{i=0}^{n} v_i x B^i$$

Destarte, a representação numérica $Z_{(10)}$ está associada à mesma *quantidade de elementos* associada à representação numérica $V_{(B)}$.

Como exemplo, $11010_{(2)} = (1x2^4 + 1x2^3 + 0x2^2 + 1x2^1 + 0x2^0)_{(10)} = 26_{(10)}$

Veja que se $B = 10$, a regra é consistente:
$1431_{(10)} = (1x10^3 + 4x10^2 + 3x10^1 + 1x10^0)_{(10)} = 1431_{(10)}$

Bases Numéricas: Conversão da base 10 para a base B

Definições:

1. $V_{(10)}$ um número na base decimal e B a base alvo
2. q_i o i-ésimo quociente de divisão e r_i o i-ésimo resto de divisão
3. *div* a função divisão inteira e *mod* o resto da divisão inteira

A conversão de $V_{(10)}$ para $Z_{(B)}$ é definida pelos passos:

1. $q_0 = V_{(10)}$
2. enquanto $(q_i \neq 0)$ faça
   1. $q_{i+1} = q_i \ div \ B$
   2. $r_i = q_i \ mod \ B$
3. Então, $Z_{(B)} = r_n \ r_{n-1} \ r_{n-2} \ \ldots \ r_1 \ r_0$

Bases Numéricas: Exemplo de conversão base 10 $\rightarrow B$

Seja $V = 23_{(10)}$. Então, $q_0 = 23$

Conversão de $23_{(10)}$ para $Z_{(2)}$:

Etapa	q_i	operação	Quociente	Resto
1	23	div 2 $\rightarrow$	$q_1 = 11$	$r_0 = 1$
2	11	div 2 $\rightarrow$	$q_2 = 5$	$r_1 = 1$
3	5	div 2 $\rightarrow$	$q_3 = 2$	$r_2 = 1$
4	2	div 2 $\rightarrow$	$q_4 = 1$	$r_3 = 0$
5	1	div 2 $\rightarrow$	$q_5 = 0$	$r_4 = 1 \Uparrow$

- Assim, o resultado da conversão de $V_{(10)}$ para $B = 2$ é $Z = 10111_{(2)}$

Qual é a representação de $V = 23_{(10)}$ para $Z_{(3)}$?

Bases Numéricas: Representação em diferentes bases

Diferentes bases para a representação $V_{(B)} = v_n \ v_{n-1} \ \dots \ v_i \ \dots \ v_1 \ v_0$

- Base 2 (binária), $B = 2$ e $v_i \in \{0, 1\}$
- Base 3, $B = 3$ e $v_i \in \{0, 1, 2\}$
- Base 4, $B = 4$ e $v_i \in \{0, 1, 2, 3\}$
- ...
- Base 8, $B = 8$ e $v_i \in \{0, 1, 2, 3, 4, 5, 6, 7\}$
- ...
- Base 10, $B = 10$ e $v_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- ...

Representação de números em diferentes bases

Hexadecimal (Base 16)

- Base 16, $B = 16$ e $v_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$
- Como os símbolos numéricos estão no intervalo [0..9], precisamos de símbolos atômicos para representar todos os de $B = 16$. Assim, temos:
  - $A_{(16)} = 10_{(10)}$
  - $B_{(16)} = 11_{(10)}$
  - $C_{(16)} = 12_{(10)}$
  - $D_{(16)} = 13_{(10)}$
  - $E_{(16)} = 14_{(10)}$
  - $F_{(16)} = 15_{(10)}$

Exemplo:

$V_{(16)} = A35F_{16} = (10x16^3 + 3x16^2 + 5x16^1 + 15x16^0)_{(10)}$

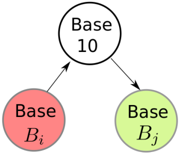
Comparação entre bases

Decimal	*Binário*	*Hexadecimal*	*Decimal*	*Binário*	*Hexadecimal*
0	0000	0	8	1000	8
1	0001	1	9	1001	9
2	0010	2	10	1010	*A*
3	0011	3	11	1011	*B*
4	0100	4	12	1100	*C*
5	0101	5	13	1101	*D*
6	0110	6	14	1110	*E*
7	0111	7	15	1111	*F*

Conversão entre diferentes bases

Como fazemos para transformar de uma base B_i para a base B_j?

Como aprendemos a transformar as representações numéricas de qualquer base B para a base decimal 10 e da base decimal para qualquer base B, podemos usar a base decimal como ponte entre as bases;



Pesquise formas para se utilizar a base binária $B = 2$ como ponte entre as bases $B = 2^x \ / \ x \in \mathbb{N} \ e \ x > 1$

Representação binária em sistemas computacionais

Cada símbolo de uma cadeia que representa um número binário em um sistema computacional é armazenado em uma unidade de memória chamada *bit*. Desta forma, consideramos que os números representados em computadores são limitados pelo tamanho reservado à sua representação. Veja que um número binário de 16 *bits* deve ser representado ocupando todos e não mais que os 16 *bits* a ele reservados.

- Exemplos de números e quantos bits eles possuem:
    - $10_{(2)} \rightarrow 2$ bits
    - $1010_{(2)} \rightarrow 4$ bits
    - $10010010_{(2)} \rightarrow 8$ bits

Considerando o primeiro exemplo da lista, $10_{(2)} = 2_{(10)}$, ele pode ser representado mantendo-se seu valor absoluto em 16 *bits* como $0000\ 0000\ 0000\ 0010_{(2)}$, cujos espaços apenas facilitam a visualização.

Representação de números fracionários na base binária

Os números fracionários são representados na base binária apenas pela aparição da vírgula que separa a parte inteira da parte fracionária. O valor representado por um número $X_{(2)}$ em decimal segue a mesma técnica ensinada. Contudo os dígitos da parte fracionária são multiplicados pela base binária pelas potências negativas, como se segue:

Exemplo do número $4,75_{(10)}$:

- $4,75_{(10)} = 4_{(10)} + 0,75_{(10)}$
  - $4_{(10)} = 100_{(2)} = 1x2^2 + 0x2^1 + 0x2^0$
  - $0,75_{(10)} = 0,11_{(2)} = 1x2^{-1} + 1x2^{-2} = \frac{1}{2} + \frac{1}{4} = 0,5 + 0,25$
- $\boxed{4,75_{(10)} = 100,11_{(2)}}$

Conversão de números fracionários decimais para a base binária

Sem apresentar o formalismo anterior, a conversão de números fracionários, ao contrário de se agrupar os restos das divisões (no caso dos números inteiros), a conversão é feita pelas sucessivas multiplicações pela base. Todos os valores inteiros da multiplicação são agrupados para formar a representação. veja o exemplo:

Exemplo do número $0,6875_{10}$

$$\begin{array}{ccccccc} 2 & \text{x} & ,6875 & = & 1,3750 & 1 & \Downarrow \\ 2 & \text{x} & ,3750 & = & 0,7500 & 0 & \\ 2 & \text{x} & ,7500 & = & 1,5000 & 1 & \\ 2 & \text{x} & ,5000 & = & 1,0000 & 1 & \end{array}$$

Destarte, $0,6875_{(10)}$ é representado como $0,1011_{(2)}$

Exercício: Formalize a regra de conversão entre números fracionários na base decimal para a base binária.

Representação de números fracionários na base binária

Assim como números fracionários na base decimal podem não oferecer uma representação finita, o caso de dízimas periódicas e não periódicas, o mesmo ocorre com números fracionários na base binária. Vejamos, como exemplo, o número com representação exata na base decimal $0,7_{(10)}$ sendo convertido para a base binária.

Exemplo do número $0,7_{(10)}$

2	x	,7	=	1,4	1 ⇓
2	x	,4	=	0,8	0
2	x	,8	=	1,6	1
2	x	,6	=	1,2	1
2	x	,2	=	0,4	0
2	x	,4	=	0,8	0 (repete)
2	x	,8	=	1,6	1 (repete)
					...(repete)

Representação de números fracionários na base binária

O exemplo da conversão do número decimal $0,7_{(10)}$ para a base binária exige uma representação (infinta) com uma dízima periódica.

Vejamos as seguintes considerações:

- $0,7_{(10)} = 0{,}1\,0110\,\underline{0110}\ldots_{(2)}$
- $0110_{(2)}$ é a fração geratriz do número
- O que acontece quando a representação é feita no computador?
- Quais são os problemas que podem ocorrer na computação a partir destas representações?
- O que é feito para minimizar os efeitos negativos destas representações não exatas?

Representação de números inteiros com sinal

Até o momento, estudamos a representação de números em valor absoluto em diferentes bases. Agora, estudaremos quatro estratégias para representar números com sinal em sistemas computacionais, que são:

- Sinal de magnitude
- Complemento de Um
- Complemento de Dois
- Excesso de $2^{n-1} - x$ (em particular, o *polarizado*)

Para tal, faz-se necessário definir uma quantidade de *bits* para representar os números. Com isso, definem-se os limites superior e inferior das representações, além de algumas particularidades. Em nossas apresentações, consideramos 8 *bits* para exemplificar os sistemas

Como exemplo, o $V = 1010_{(2)}$ é representado em 8 bits como $00001010_{(2)}$

Representação de números inteiros com sinal

A determinação imposta pelos sistemas computacionais sobre a quantidade de *bits* para representar os números implica em:

- Definição da *quantidade* de números que podem ser representados (2^n).
    - Por exemplo, com 8 *bits* temos 256 representações diferentes.
    - Para 8 *bits*, os valores com *representação binária* (representações binárias que correspondem à mesma quantidade quando convertidas em suas representações em decimal) estão no intervalo fechado $[0_{(10)} .. 255_{(10)}]$
- As técnicas de representação de números negativos usam as representações dos números para *associar* valores não necessariamente correspondentes ao valor exato da representação binária.
- Em geral, os sistemas dividem metade das representações numéricas para os números positivos e outra metade para os não positivos.
- Ao se definir a posição do zero, identificar os números positivos e os negativos. (Isso altera o valor do número em relação à sua representação binária)
- A quantidade de números S representada continua a mesma (ou $S - 1$). Porém, os limites inferior e superior das representações é alterado.

Representação de números inteiros com sinal

A determinação imposta pelos sistemas computacionais sobre a quantidade de *bits* para representar os números implica em:

- Definição da *quantidade* de números que podem ser representados (2^n).
  - Por exemplo, com 8 *bits* temos 256 representações diferentes.
  - Para 8 *bits*, os valores com *representação binária* (representações binárias que correspondem à mesma quantidade quando convertidas em suas representações em decimal) estão no intervalo fechado $[0_{(10)} .. 255_{(10)}]$
- As técnicas de representação de números negativos usam as representações dos números para *associar* valores não necessariamente correspondentes ao valor exato da representação binária.
- Em geral, os sistemas dividem metade das representações numéricas para os números positivos e outra metade para os não positivos.
- Ao se definir a posição do zero, identificar os números positivos e os negativos. (Isso altera o valor do número em relação à sua representação binária)
- A quantidade de números S representada continua a mesma (ou $S - 1$). Porém, os limites inferior e superior das representações é alterado.

Representação de números inteiros com sinal

A determinação imposta pelos sistemas computacionais sobre a quantidade de *bits* para representar os números implica em:

- Definição da *quantidade* de números que podem ser representados (2^n).
  - Por exemplo, com 8 *bits* temos 256 representações diferentes.
  - Para 8 *bits*, os valores com *representação binária* (representações binárias que correspondem à mesma quantidade quando convertidas em suas representações em decimal) estão no intervalo fechado $[0_{(10)} .. 255_{(10)}]$
- As técnicas de representação de números negativos usam as representações dos números para *associar* valores não necessariamente correspondentes ao valor exato da representação binária.
- Em geral, os sistemas dividem metade das representações numéricas para os números positivos e outra metade para os não positivos.
- Ao se definir a posição do zero, identificar os números positivos e os negativos. (Isso altera o valor do número em relação à sua representação binária)
- A quantidade de números S representada continua a mesma (ou $S - 1$). Porém, os limites inferior e superior das representações é alterado.

Representação de números inteiros com sinal

A determinação imposta pelos sistemas computacionais sobre a quantidade de *bits* para representar os números implica em:

- Definição da *quantidade* de números que podem ser representados (2^n).
  - Por exemplo, com 8 *bits* temos 256 representações diferentes.
  - Para 8 *bits*, os valores com *representação binária* (representações binárias que correspondem à mesma quantidade quando convertidas em suas representações em decimal) estão no intervalo fechado $[0_{(10)} .. 255_{(10)}]$
- As técnicas de representação de números negativos usam as representações dos números para *associar* valores não necessariamente correspondentes ao valor exato da representação binária.
- Em geral, os sistemas dividem metade das representações numéricas para os números positivos e outra metade para os não positivos.
- Ao se definir a posição do zero, identificar os números positivos e os negativos. (Isso altera o valor do número em relação à sua representação binária)
- A quantidade de números S representada continua a mesma (ou $S - 1$). Porém, os limites inferior e superior das representações é alterado.

Representação de números inteiros com sinal

A determinação imposta pelos sistemas computacionais sobre a quantidade de *bits* para representar os números implica em:

- Definição da *quantidade* de números que podem ser representados (2^n).
  - Por exemplo, com 8 *bits* temos 256 representações diferentes.
  - Para 8 *bits*, os valores com *representação binária* (representações binárias que correspondem à mesma quantidade quando convertidas em suas representações em decimal) estão no intervalo fechado $[0_{(10)} .. 255_{(10)}]$
- As técnicas de representação de números negativos usam as representações dos números para *associar* valores não necessariamente correspondentes ao valor exato da representação binária.
- Em geral, os sistemas dividem metade das representações numéricas para os números positivos e outra metade para os não positivos.
- Ao se definir a posição do zero, identificar os números positivos e os negativos. (Isso altera o valor do número em relação à sua representação binária)
- A quantidade de números S representada continua a mesma (ou $S-1$). Porém, os limites inferior e superior das representações é alterado.

Representação de números inteiros com sinal

A determinação imposta pelos sistemas computacionais sobre a quantidade de *bits* para representar os números implica em:

- Definição da *quantidade* de números que podem ser representados (2^n).
    - Por exemplo, com 8 *bits* temos 256 representações diferentes.
    - Para 8 *bits*, os valores com *representação binária* (representações binárias que correspondem à mesma quantidade quando convertidas em suas representações em decimal) estão no intervalo fechado $[0_{(10)} .. 255_{(10)}]$
- As técnicas de representação de números negativos usam as representações dos números para *associar* valores não necessariamente correspondentes ao valor exato da representação binária.
- Em geral, os sistemas dividem metade das representações numéricas para os números positivos e outra metade para os não positivos.
- Ao se definir a posição do zero, identificar os números positivos e os negativos. (Isso altera o valor do número em relação à sua representação binária)
- A quantidade de números S representada continua a mesma (ou $S - 1$). Porém, os limites inferior e superior das representações é alterado.

Sinal de Magnitude

O sistema Sinal de Magnitude é o mais simples. Nele, o *bit* mais significativo (o mais a esquerda) é utilizado para representar o sinal do número.

Sinais: 0 para números positivos e 1 para negativos:

$\boxed{0}\boxed{0000011} \equiv +3_{(10)}$

$\boxed{1}\boxed{0000011} \equiv -3_{(10)}$

Valores (em decimal) das representações:

Tanto os números negativos como os positivos representam, em módulo, o valor da representação binária presente nos $n-1$ *bits* menos significativos.

$\boxed{0}\boxed{0000101} \equiv + \boxed{0000101_{(2)}} \equiv +5_{(10)}$

$\boxed{1}\boxed{0100100} \equiv - \boxed{0100100_{(2)}} \equiv -36_{(10)}$

Sinal de Magnitude

Uma observação é que neste sistema há duas representações para o 0, sendo uma positiva e outra negativa. Contudo, em valor absoluto, não há distinção na matemática entre +0 e -0

Zero positivo e zero negativo:

$\boxed{00000000} \equiv +0_{(10)}$

$\boxed{10000000} \equiv -0_{(10)}$

Quais são os problemas enfrentados na implementação de sistemas de numeração que permitem mais de uma representação para um número específico?

Complemento de Um (C1)

No sistema de representação Complemento de Um, os números positivos serão representados em sua representação binária. Já a representação dos negativos é obtida pela regra de troca de sinal a partir dos positivos. Mais uma vez, o bit mais significativo representará o sinal.

Representação de números positivos:

$\boxed{0}\boxed{0000101} \equiv + \boxed{0000101_{(2)}} \equiv +5_{(10)}$

Troca de sinal

Para o número x, a representação de $-(x)$ é feita da seguinte forma: Troca-se em x todas as ocorrências de *zero* por *um* e todas as ocorrências de *um* por *zero*. Ex:

Seja $A = \boxed{0}\boxed{0000101}$. Então $(-A) = \boxed{1}\boxed{1111010}$

Seja $B = \boxed{1}\boxed{0000111}$. Então $(-B) = \boxed{0}\boxed{1111000}$

Complemento de Um

O complemento de um apresenta a mesma desvantagem do sistema sinal de magnitude.

Zero positivo e zero negativo:

$\boxed{00000000} \equiv +0_{(10)}$

$\boxed{11111111} \equiv -0_{(10)}$

Observe que a representação em complemento de um para números negativos é diferente da *representação binária* presente nos (n-1) bits menos significativos.

$\boxed{10000010}$ não representa o valor $-2_{(10)}$

$\boxed{10000010} \equiv -?_{(10)}$

Complemento de Dois (C2)

No sistema de representação complemento de dois, os números positivos serão representados em sua representação binária. Já a representação dos negativos é obtida pela regra de troca de sinal a partir dos positivos. Mais uma vez, o bit mais significativo representará o sinal.

Representação de números positivos:

$\boxed{0}\boxed{0000101} \equiv + \boxed{0000101_{(2)}} \equiv +5_{(10)}$

Troca de sinal de um número X:

1º Obter o complemento de um de X;

2º Somar o valor $1_{(10)}$ ao resultado obtido no passo [1º].

Exemplo: seja $A = 00000101$, um valor em Complemento de Dois, então $-A$ é:

1111010 $-A$ em C1
+00000001

$\boxed{11111011}$ $-A$ em C2

Considerações sobre os três sistemas estudados

Comparando os sistemas: Sinal de magnitude, complemento de um e complemento de dois

- Todos representam números positivos da mesma forma. (são equivalentes à representação binária)
- Apenas complemento de dois tem uma única representação para o $0_{(10)}$
- Cada um deles tem regra própria para a troca de sinal de um número dentro do sistema

Para complementar os estudos, veja outros materiais didáticos e estude as operações algébricas para os diferentes sistemas.
Sugestões:
`http://www.ic.uff.br/~boeres/slides_FAC/FAC-complemento-um-e-dois.pdf`

Excesso de $2^{n-1} - x$

O sistema Excesso de $2^{n-1} - x$ representa os números com sinal de uma forma muito simples. A quantidade de *bits* n e o valor inteiro x definem o deslocamento do zero na linha dos números naturais (representação binária). Em geral, busca-se o zero no meio do intervalo, para aproximar a quantidade de números negativos e positivos.

Sistema polarizado é o Excesso de $2^{n-1} - x$ quando $x = 1$

Neste sistema, o zero será o maior número em sua representação binária em que o *bit* mais significativo é zero. Veja o exemplo para $n = 8$.

- Representação do *zero do sistema*:
  $(2^{8-1} - 1)_{(10)} = 127_{(10)} \rightarrow \boxed{01111111}$ polarização do zero
- Neste caso, todas as representações de números positivos são maiores que zero e começam com 1 e os não positivos com 0.

Sistema Polarizado: Exemplos para $n = 8$

Exemplo da representação do número $+1_{(10)}$:

1. $2^{8-1} - 1 = 127_{(10)} \rightarrow$ *zero do sistema*
2. $zero + 1_{(10)} = 127_{(10)} + 1_{(10)} = 128_{(10)}$
3. $128_{(10)} \equiv \boxed{10000000} \rightarrow$ representação de $+1_{(10)}$.

Exemplo da representação do número $-1_{(10)}$:

1. $2^{8-1} - 1 = 127_{(10)} \rightarrow$ *zero do sistema*
2. $zero - 1_{(10)} = 127_{(10)} + (-1_{(10)}) = 126_{(10)}$
3. $126_{(10)} \equiv \boxed{01111110} \rightarrow$ representação de $-1_{(10)}$.

Para o exemplo, qual é o maior valor e o menor valor que podem ser representados?

Representação de números inteiros com sinal

Como a representação de números em sistemas computacionais é limitada pela quantidade de *bits*, a tentativa de representar números que excedem o limite de representação ocasiona o evento que chamamos de *overflow*. Um exemplo de *overflow* pode surgir da soma de dois números representados corretamente, mas que geram um resultado que excede o intervalo de representação.

Exemplo de *overflow* em complemento de dois:

$$01111111 + 00000001 = 10000000$$

Neste caso precisaríamos de 9 *bits* para representar o resultado, pois o resultado é negativo para a representação em 8 *bits*

Representações dos quatro sistemas (por Stallings - 8ª Ed.)

Tabela 9.2 Representação alternativa para inteiros de 4 bits

Representação decimal	Representação sinal-magnitude	Representação em complemento de dois	Representação polarizada
+8	–	–	1111
+7	0111	0111	1110
+6	0110	0110	1101
+5	0101	0101	1100
+4	0100	0100	1011
+3	0011	0011	1010
+2	0010	0010	1001
+1	0001	0001	1000
+0	0000	0000	0111
–0	1000	–	–
–1	1001	1111	0110
–2	1010	1110	0101
–3	1011	1101	0100
–4	1100	1100	0011
–5	1101	1011	0010
–6	1110	1010	0001
–7	1111	1001	0000
–8	–	1000	–

Conteúdo complementar a esta apresentação

Consulte o material bibliográfico para estudar as operações, que envolvem:

- Operações (soma, subtração, multiplicação e divisão) entre números binários
- Operações entre números em sinal de magnitude
- Operações entre números em complemento de um
- Operações entre números em complemento de dois
- Operações entre números em excesso de $2^{n-1} - x$

Ponto flutuante

O ponto flutuante é uma estrutura para representar números binários que possuem parte fracionária. Representa um subconjunto dos números reais.

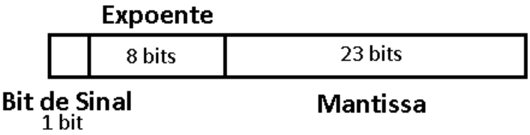
Em um intervalo fechado, vários números não podem ser representados. Você pode citar duas explicações para isso?

Há vários formatos de números em ponto flutuante. Em particular, daremos mais atenção à representação IEEE 754 de 32*bits*.

A representação é feita em três partes na sequência de *bits*:

1. *Bit* de Sinal do número $\pm$: $0 \rightarrow$ *positivo* e $1 \rightarrow$ *negativo*
2. Expoente do número E: sistema polarizado
3. Mantissa M: representa a parte fracionária do número

Ou seja, $\pm M$ x 2^E. Observe que a parte inteira do número é implícita.



Ponto Flutuante IEEE 754: Representação

Há três etapas para se representar um número em ponto flutuante IEEE 754:

- Converter o número para sua representação binária (já visto);
- *Normalizar* a representação binária;
- Preencher os campos de *bits* da representação em ponto flutuante.

Veja as igualdades dos números binários abaixo:

$101,1 = 0,01011x2^4 = 1011,0x2^{-1} = 10,11x2^1 = 1,011x2^2$

Neste caso, a última representação está normalizada no padrão IEEE 754.

$$\boxed{1, bbbbx2^E}$$

A normalização facilita as operações entre os números e possibilita tornar implícita a parte inteira do número.

Para nosso exemplo 1,011 $x2^2$, a mantissa é 011 (que deve ser representada em 23 *bits*, o expoente é $2_{(10)}$, que deve ser polarizado e representado em 8 *bits* e o sinal o bit 0.

O 1 antes da vírgula não será representado para economizar um bit no número.

Assim, a mantissa terá efetivamente 24*bits*

Ponto Flutuante IEEE 754: Representação

Há três etapas para se representar um número em ponto flutuante IEEE 754:

- Converter o número para sua representação binária (já visto);
- *Normalizar* a representação binária;
- Preencher os campos de *bits* da representação em ponto flutuante.

Veja as igualdades dos números binários abaixo:

$101,1 = 0,01011x2^4 = 1011,0x2^{-1} = 10,11x2^1 = 1,011x2^2$

Neste caso, a última representação está normalizada no padrão IEEE 754.

$$\boxed{1,bbbbx2^E}$$

A normalização facilita as operações entre os números e possibilita tornar implícita a parte inteira do número.

Para nosso exemplo 1,011 $x2^2$, a mantissa é 011 (que deve ser representada em 23 *bits*, o expoente é $2_{(10)}$, que deve ser polarizado e representado em 8 *bits* e o sinal o bit 0.

O 1 antes da vírgula não será representado para economizar um bit no número.

Assim, a mantissa terá efetivamente 24*bits*

Representação de números no sistema IEEE 754

Exemplo do número $K = 9,5_{(10)}$:

1º- Número positivo: primeiro bit tem valor 0.

2º- $9,5_{(10)} \equiv 1001,1_{(2)} \equiv 1,0011_{(2)}x2^3 \rightarrow$ Expoente $= 3$

3º- $3_{(10)} \equiv 10000010_{(2)}$ no Sistema Polarizado

4º- A mantissa será o valor do passo 2 após a vírgula: $0011_{(2)}$

5º- Por fim, juntando os valores encontrados e adicionando zeros ao resto dos bits da mantissa teremos:

$K =$ | 0 | 10000010 | 00110000000000000000000 |

sinal$= 1bit$ expoente $= 8bits$ mantissa $= 23bits$

Como é a representação de $Z = -9,5_{(10)}$?

Como é a representação de $Z = -17,3_{(10)}$?

Representação de números no sistema IEEE 754

Exemplo do número $K = 9,5_{(10)}$:

1º- Número positivo: primeiro bit tem valor 0.

2º- $9,5_{(10)} \equiv 1001,1_{(2)} \equiv 1,0011_{(2)} x 2^3 \rightarrow$ Expoente $= 3$

3º- $3_{(10)} \equiv 10000010_{(2)}$ no Sistema Polarizado

4º- A mantissa será o valor do passo 2 após a vírgula: $0011_{(2)}$

5º- Por fim, juntando os valores encontrados e adicionando zeros ao resto dos bits da mantissa teremos:

$$K = \boxed{0}\boxed{10000010}\boxed{00110000000000000000000}$$

sinal $= 1 bit$ expoente $= 8 bits$ mantissa $= 23 bits$

Como é a representação de $Z = -9,5_{(10)}$?

Como é a representação de $Z = -17,3_{(10)}$?

Representação de números no sistema IEEE 754

Exemplo do número $K = 9,5_{(10)}$:

1º- Número positivo: primeiro bit tem valor 0.

2º- $9,5_{(10)} \equiv 1001,1_{(2)} \equiv 1,0011_{(2)} x 2^3 \rightarrow$ Expoente $= 3$

3º- $3_{(10)} \equiv 10000010_{(2)}$ no Sistema Polarizado

4º- A mantissa será o valor do passo 2 após a vírgula: $0011_{(2)}$

5º- Por fim, juntando os valores encontrados e adicionando zeros ao resto dos bits da mantissa teremos:

$$K = \boxed{0}\boxed{10000010}\boxed{00110000000000000000000}$$

sinal$= 1bit$ expoente $= 8bits$ mantissa $= 23bits$

Como é a representação de $Z = -9,5_{(10)}$?
Como é a representação de $Z = -17,3_{(10)}$?

Representação de números no sistema IEEE 754

Exemplo do número $K = 9,5_{(10)}$:

1º- Número positivo: primeiro bit tem valor 0.

2º- $9,5_{(10)} \equiv 1001,1_{(2)} \equiv 1,0011_{(2)} x 2^3 \rightarrow$ Expoente $= 3$

3º- $3_{(10)} \equiv 10000010_{(2)}$ no Sistema Polarizado

4º- A mantissa será o valor do passo 2 após a vírgula: $0011_{(2)}$

5º- Por fim, juntando os valores encontrados e adicionando zeros ao resto dos bits da mantissa teremos:

$$K = \boxed{0}\boxed{10000010}\boxed{00110000000000000000000}$$

sinal$= 1bit$ expoente $= 8bits$ mantissa $= 23bits$

Como é a representação de $Z = -9,5_{(10)}$?

Como é a representação de $Z = -17,3_{(10)}$?

Representação de números no sistema IEEE 754

Exemplo do número $K = 9,5_{(10)}$:

1º- Número positivo: primeiro bit tem valor 0.

2º- $9,5_{(10)} \equiv 1001,1_{(2)} \equiv 1,0011_{(2)}x2^3 \rightarrow$ Expoente $= 3$

3º- $3_{(10)} \equiv 10000010_{(2)}$ no Sistema Polarizado

4º- A mantissa será o valor do passo 2 após a vírgula: $0011_{(2)}$

5º- Por fim, juntando os valores encontrados e adicionando zeros ao resto dos bits da mantissa teremos:

$$K = \boxed{0}\boxed{10000010}\boxed{00110000000000000000000}$$

sinal$= 1bit$ expoente $= 8bits$ mantissa $= 23bits$

Como é a representação de $Z = -9,5_{(10)}$?

Como é a representação de $Z = -17,3_{(10)}$?

Representação de números no sistema IEEE 754

Exemplo do número $K = 9,5_{(10)}$:

1º- Número positivo: primeiro bit tem valor 0.

2º- $9,5_{(10)} \equiv 1001,1_{(2)} \equiv 1,0011_{(2)} x 2^3 \rightarrow$ Expoente $= 3$

3º- $3_{(10)} \equiv 10000010_{(2)}$ no Sistema Polarizado

4º- A mantissa será o valor do passo 2 após a vírgula: $0011_{(2)}$

5º- Por fim, juntando os valores encontrados e adicionando zeros ao resto dos bits da mantissa teremos:

$$K = \boxed{0}\boxed{10000010}\boxed{00110000000000000000000}$$

sinal$= 1bit$ expoente $= 8bits$ mantissa $= 23bits$

Como é a representação de $Z = -9,5_{(10)}$?

Como é a representação de $Z = -17,3_{(10)}$?

Representação de números no sistema IEEE 754

Exemplo do número $K = 9,5_{(10)}$:

1º- Número positivo: primeiro bit tem valor 0.

2º- $9,5_{(10)} \equiv 1001,1_{(2)} \equiv 1,0011_{(2)}x2^3 \rightarrow$ Expoente $= 3$

3º- $3_{(10)} \equiv 10000010_{(2)}$ no Sistema Polarizado

4º- A mantissa será o valor do passo 2 após a vírgula: $0011_{(2)}$

5º- Por fim, juntando os valores encontrados e adicionando zeros ao resto dos bits da mantissa teremos:

$$K = \boxed{0}\boxed{10000010}\boxed{00110000000000000000000}$$

sinal$= 1bit$ expoente $= 8bits$ mantissa $= 23bits$

Como é a representação de $Z = -9,5_{(10)}$?

Como é a representação de $Z = -17,3_{(10)}$?

Representação de números no sistema IEEE 754

Exemplo do número $K = 9,5_{(10)}$:

1º- Número positivo: primeiro bit tem valor 0.

2º- $9,5_{(10)} \equiv 1001,1_{(2)} \equiv 1,0011_{(2)} x 2^3 \rightarrow$ Expoente $= 3$

3º- $3_{(10)} \equiv 10000010_{(2)}$ no Sistema Polarizado

4º- A mantissa será o valor do passo 2 após a vírgula: $0011_{(2)}$

5º- Por fim, juntando os valores encontrados e adicionando zeros ao resto dos bits da mantissa teremos:

$$K = \boxed{0}\boxed{10000010}\boxed{00110000000000000000000}$$

sinal$= 1bit$ expoente $= 8bits$ mantissa $= 23bits$

Como é a representação de $Z = -9,5_{(10)}$?
Como é a representação de $Z = -17,3_{(10)}$?

Representação de números no sistema IEEE 754

Arredondamento na representação

- O arredondamento acontece quando a representação da mantissa é maior que 23 *bits*
- Neste caso, somamos o valor da posição 24 com a mantissa. Veja o exemplo:
- $X = 0.72_{(10)}$ $\quad 0,\overline{10111000010100011110}1011100001\ldots_{(2)}$
- normalizando: $1,0111\ 0000\ 1010\ 0011\ 1101\ 011\ 100\ldots \cdot 2^{-1}$
- arredondando a mantissa:

$$\begin{array}{r} 0111\ 0000\ 1010\ 0011\ 1101\ 011 \\ +\qquad\qquad\qquad\qquad\qquad\qquad 1 \\ \hline \end{array}$$

$$X = \boxed{0}\boxed{01111110}\boxed{0111\ 0000\ 1010\ 0011\ 1101\ 100}$$

https://www.h-schmidt.net/FloatConverter/IEEE754.html

Representação de números no sistema IEEE 754

Arredondamento na representação

- O arredondamento acontece quando a representação da mantissa é maior que 23 *bits*
- Neste caso, somamos o valor da posição 24 com a mantissa. Veja o exemplo:
- $X = 0.72_{(10)}$ $0,\overline{10111000010100011110}1011100001\ldots_{(2)}$
- normalizando: $1,0111\ 0000\ 1010\ 0011\ 1101\ 011\ 100\ldots \cdot 2^{-1}$
- arredondando a mantissa:

$$
\begin{array}{r}
0111\ 0000\ 1010\ 0011\ 1101\ 011 \\
+ \quad \underline{\hspace{6cm} 1}
\end{array}
$$

$$X = \boxed{0}\boxed{01111110}\boxed{0111\ 0000\ 1010\ 0011\ 1101\ 100}$$

https://www.h-schmidt.net/FloatConverter/IEEE754.html

Representação de números no sistema IEEE 754

Arredondamento na representação

- O arredondamento acontece quando a representação da mantissa é maior que 23 *bits*
- Neste caso, somamos o valor da posição 24 com a mantissa. Veja o exemplo:
- $X = 0.72_{(10)}$ $\quad 0,\overline{10111000010100011110101110000}1\ldots_{(2)}$
- normalizando: $1,0111\ 0000\ 1010\ 0011\ 1101\ 011\ 100\ldots \cdot 2^{-1}$
- arredondando a mantissa:

$$\begin{array}{r} 0111\ 0000\ 1010\ 0011\ 1101\ 011 \\ +\quad 1 \\ \hline \end{array}$$

$$X = \boxed{0}\boxed{01111110}\boxed{0111\ 0000\ 1010\ 0011\ 1101\ 100}$$

https://www.h-schmidt.net/FloatConverter/IEEE754.html

Representação de números no sistema IEEE 754

Arredondamento na representação

- O arredondamento acontece quando a representação da mantissa é maior que 23 *bits*
- Neste caso, somamos o valor da posição 24 com a mantissa. Veja o exemplo:
- $X = 0.72_{(10)}$ $0,\overline{10111000010100011110}1011100001\ldots_{(2)}$
- normalizando: $1,0111\ 0000\ 1010\ 0011\ 1101\ 011\ 100\ldots \cdot 2^{-1}$
- arredondando a mantissa:

$$\begin{array}{r} 0111\ 0000\ 1010\ 0011\ 1101\ 011 \\ + \quad 1 \\ \hline \end{array}$$

$X = \boxed{0}\boxed{01111110}\boxed{0111\ 0000\ 1010\ 0011\ 1101\ 100}$

https://www.h-schmidt.net/FloatConverter/IEEE754.html

Representação de números no sistema IEEE 754

Arredondamento na representação

- O arredondamento acontece quando a representação da mantissa é maior que 23 *bits*
- Neste caso, somamos o valor da posição 24 com a mantissa. Veja o exemplo:
- $X = 0.72_{(10)}$ $\quad 0,\overline{10111000010100011110101}1100001\ldots_{(2)}$
- normalizando: $1,0111\ 0000\ 1010\ 0011\ 1101\ 011\ 100\ldots \cdot 2^{-1}$
- arredondando a mantissa:

$$\begin{array}{rr} & 0111\ 0000\ 1010\ 0011\ 1101\ 011 \\ + & 1 \\ \hline \end{array}$$

$$X = \boxed{0}\boxed{01111110}\boxed{0111\ 0000\ 1010\ 0011\ 1101\ 100}$$

https://www.h-schmidt.net/FloatConverter/IEEE754.html

Representação de números no sistema IEEE 754

Arredondamento na representação

- O arredondamento acontece quando a representação da mantissa é maior que 23 *bits*
- Neste caso, somamos o valor da posição 24 com a mantissa. Veja o exemplo:
- $X = 0.72_{(10)}$ $\quad 0,\overline{10111000010100011110101110}0001\ldots_{(2)}$
- normalizando: $1,0111\ 0000\ 1010\ 0011\ 1101\ 011\ 100\ldots \cdot 2^{-1}$
- arredondando a mantissa:

$$
\begin{array}{r}
0111\ 0000\ 1010\ 0011\ 1101\ 011 \\
+ \quad \underline{\qquad\qquad\qquad\qquad\qquad\qquad 1}
\end{array}
$$

$X = \boxed{0}\boxed{01111110}\boxed{0111\ 0000\ 1010\ 0011\ 1101\ 100}$

https://www.h-schmidt.net/FloatConverter/IEEE754.html

Representação de números no sistema IEEE 754

Arredondamento na representação

- O arredondamento acontece quando a representação da mantissa é maior que 23 *bits*
- Neste caso, somamos o valor da posição 24 com a mantissa. Veja o exemplo:
- $X = 0.72_{(10)}$ $\quad 0,\overline{10111000010100011110101}1100001\ldots_{(2)}$
- normalizando: $1{,}0111\ 0000\ 1010\ 0011\ 1101\ 011\ 100\ldots \cdot 2^{-1}$
- arredondando a mantissa:

$$\begin{array}{r} 0111\ 0000\ 1010\ 0011\ 1101\ 011 \\ +\quad 1 \\ \hline \end{array}$$

$$X = \boxed{0}\boxed{01111110}\boxed{0111\ 0000\ 1010\ 0011\ 1101\ 100}$$

https://www.h-schmidt.net/FloatConverter/IEEE754.html

Representação de números no sistema IEEE 754

Arredondamento na representação

- O arredondamento acontece quando a representação da mantissa é maior que 23 *bits*
- Neste caso, somamos o valor da posição 24 com a mantissa. Veja o exemplo:
- $X = 0.72_{(10)}$ $\quad 0,\overline{10111000010100011110101}100001\ldots_{(2)}$
- normalizando: $1,0111\ 0000\ 1010\ 0011\ 1101\ 011\ 100\ldots \cdot 2^{-1}$
- arredondando a mantissa:

$$\begin{array}{rr} & 0111\ 0000\ 1010\ 0011\ 1101\ 011 \\ + & 1 \\ \hline \end{array}$$

$$X = \boxed{0}\boxed{01111110}\boxed{0111\ 0000\ 1010\ 0011\ 1101\ 100}$$

https://www.h-schmidt.net/FloatConverter/IEEE754.html

Representação de números no sistema IEEE 754

Arredondamento na representação

- O arredondamento acontece quando a representação da mantissa é maior que 23 *bits*
- Neste caso, somamos o valor da posição 24 com a mantissa. Veja o exemplo:
- $X = 0.72_{(10)} \quad 0,\overline{10111000010100011110}1011\color{red}{1}\color{black}00001\ldots_{(2)}$
- normalizando: $1,\color{blue}{0111\ 0000\ 1010\ 0011\ 1101\ 011}\ \color{red}{1}\color{black}00\ldots\cdot 2^{-1}$
- arredondando a mantissa:

$$
\begin{array}{lr}
 & \color{blue}{0111\ 0000\ 1010\ 0011\ 1101\ 011} \\
+ & \color{red}{1} \\
\hline
\end{array}
$$

$$X = \boxed{0}\boxed{01111110}\boxed{0111\ 0000\ 1010\ 0011\ 1101\ \color{red}{100}}$$

https://www.h-schmidt.net/FloatConverter/IEEE754.html

Representação de números no sistema IEEE 754

Arredondamento na representação

- O arredondamento acontece quando a representação da mantissa é maior que 23 *bits*
- Neste caso, somamos o valor da posição 24 com a mantissa. Veja o exemplo:
- $X = 0.72_{(10)} \quad 0{,}\overline{101110000101000111101}011\textcolor{red}{1}00001\ldots_{(2)}$
- normalizando: $1{,}\textcolor{blue}{0111\ 0000\ 1010\ 0011\ 1101\ 011}\ \textcolor{red}{1}00\ldots \cdot 2^{-1}$
- arredondando a mantissa:

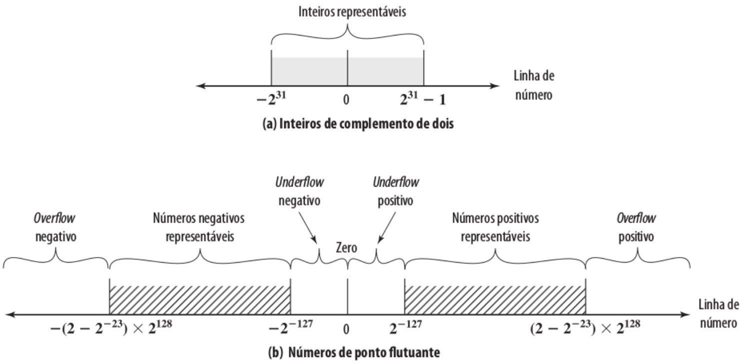
$$
\begin{array}{r}
\textcolor{blue}{0111\ 0000\ 1010\ 0011\ 1101\ 011} \\
+ \quad \underline{\hspace{12em}\textcolor{red}{1}}
\end{array}
$$

$$X = \begin{array}{|c|c|c|} \hline 0 & 01111110 & 0111\ 0000\ 1010\ 0011\ 1101\ \textcolor{red}{100} \\ \hline \end{array}$$

https://www.h-schmidt.net/FloatConverter/IEEE754.html

Comparativo de espaços representáveis

Figura 9.19 Números expressos em formatos típicos de 32 bits



Formatos do padrão IEEE 754

Tabela 9.3 Parâmetros de formato IEEE 754

Parâmetro	Formato			
	Isolado	Estendido isolado	Duplo	Estendido duplo
Tamanho da palavra (bits)	32	≥ 43	64	≥ 79
Tamanho do expoente (bits)	8	≥ 11	11	≥ 15
Polarização do expoente	127	Não especificado	1023	Não especificado
Expoente máximo	127	≥ 1023	1023	≥ 16383
Exponente mínimo	-126	≤ -1022	-1022	≤ -16382
Intervalo numérico (base 10)	$10^{-38}, 10^{+38}$	Não especificado	$10^{-308}, 10^{+308}$	Não especificado
Tamanho do significando (bits)*	23	≥ 31	52	≥ 63
Número de expoentes	254	Não especificado	2046	Não especificado
Número de frações	2^{23}	Não especificado	2^{52}	Não especificado
Número de valores	$1{,}98 \times 2^{31}$	Não especificado	$1{,}99 \times 2^{63}$	Não especificado

* Não incluso o bit implícito.

Particularidades dos números IEEE 754

Tabela 9.4 Interpretação dos números de ponto flutuante IEEE 754

	Precisão simples (32 bits)				Precisão dupla (64 bits)			
	Sinal	Expoente polarizado	Fração	Valor	Sinal	Expoente viesado	Fração	Valor
Zero positivo	0	0	0	0	0	0	0	0
Zero negativo	1	0	0	-0	1	0	0	-0
Mais infinito	0	255 (todos 1s)	0	∞	0	2047 (todos 1s)	0	∞
Menos infinito	1	255 (todos 1s)	0	$-\infty$	1	2047 (todos 1s)	0	$-\infty$
NaN silencioso	0 ou 1	255 (todos 1s)	$\neq 0$	NaN	0 ou 1	2047 (todos 1s)	$\neq 0$	NaN
Nan sinalização	0 ou 1	255 (todos 1s)	$\neq 0$	NaN	0 ou 1	2047 (todos 1s)	$\neq 0$	NaN
Diferente de zero normalizado positivo	0	$0 < e < 255$	f	$2^{e-127}(1.f)$	0	$0 < e < 2047$	f	$2^{e-1023}(1.f)$
Diferente de zero normalizado negativo	1	$0 < e < 255$	f	$-2^{e-127}(1.f)$	1	$0 < e < 2047$	f	$-2^{e-1023}(1.f)$
Desnormalizado positivo	0	0	$f \neq 0$	$2^{e-126}(0.f)$	0	0	$f \neq 0$	$2^{e-1022}(1.f)$
Desnormalizado negativo	1	0	$f \neq 0$	$-2^{e-126}(0.f)$	1	0	$f \neq 0$	$-2^{e-1022}(1.f)$

Sobre as apresentações didáticas

Atenção: este material não é completo para seus estudos. Ao contrário, é apenas um guia para lhe informar quais são os tópicos importantes a serem estudados.

O estudo completo e necessário dá-se por materiais específicos sobre cada tópico, como livros e conteúdo na Internet.

Selecione materiais de fontes de seu interesse considerando a bibliografia citada no plano de ensino da disciplina para estudar os tópicos aqui abordados.

Consulte o professor, em caso de dúvidas

Agradecimentos especiais

Agradeço ao monitor da disciplina João Paulo de Campos Carvalho por ter elaborado esta apresentação.

Agradeço a toda a comunidade LaTeX.
Em especial a *Till Tantau* pelo *Beamer*.
https://www.tcs.uni-luebeck.de/mitarbeiter/tantau/
Desta forma, tornou-se possível a escrita deste material didático.