

Sistemas de Numeração

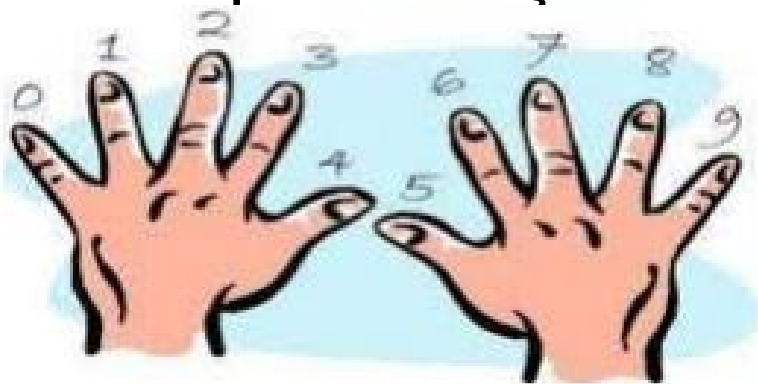
Carlos Eduardo Tanajura da Silva

Eng. Da Computação – Area1

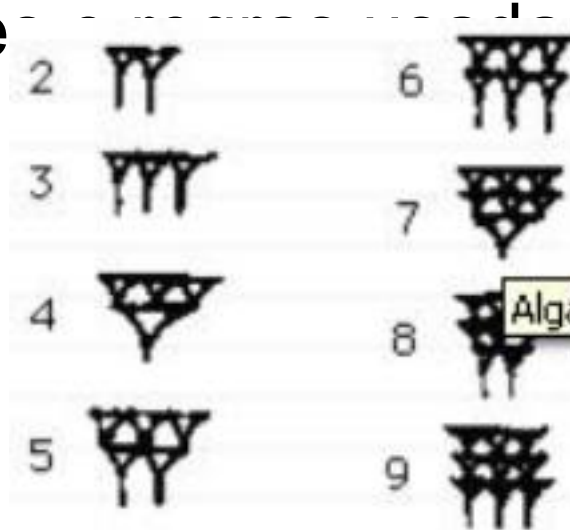
Msc Mecatrônica - UFBA

Introdução

- Um número é visto como um conceito abstrato para representação de quantidade, sendo por tanto algo relevante para a computação;
- Um sistema de numeração é o conjunto de símbolos que representam quantidade para s



Sistema de numeração Decimal. Sistema de numeração da Babilônia
Criado aproximadamente 4 mil anos.



1	.		11	≡ o .
2	..	o :	12	≡ o :
3	...	o :	13	≡ o :
4	o :	14	≡ o :
5	—	o	15	≡ o
6	÷ o .		16	≡ o .
7	÷ o :		17	≡ o :
8	÷ o :		18	≡ o :
9	÷ o :		19	≡ o :
10	= o			

Sistema de numeração Maia.

Introdução

- Principais sistemas numéricos:
 - Decimal:
 - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
 - Binário:
 - 0, 1
 - Octal:
 - 0, 1, 2, 3, 4, 5, 6, 7,
 - Hexadecimal:
 - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
 - **OBS** - neste sistemas temos algumas letras de A até F, que equivalem em decimal: 10, 11, 12, 13, 14, 15

Introdução

- Representação Decimal
 - São 10 símbolos usados de forma repetida para contagem: **0, 1, 2, 3, 4, 5, 6, 7, 8, 9**
 - Composição de um número: **Unidade, Dezena, Centena, ...**
 - Base: **10**
- Exemplo: **283_{10}**

$$283_{10} = 2x10^2 + 8x10^1 + 3x10^0$$

Introdução

- Representação Binário
 - São 2 símbolos usados de forma repetida para contagem: **0, 1**
 - Base: **2**
- Exemplo: **101_2**

$$101_2 = 1x2^2 + 0x2^1 + 1x2^0$$

$$101_2 = 1x4 + 0x2 + 1x1$$

$$101_2 = 4 + 0 + 1$$

$$101_2 = 5_{10}$$

Introdução

- Representação Octal
 - São 8 símbolos usados de forma repetida para contagem: **0, 1, 2, 3, 4, 5, 6, 7**
 - Base: **8**
- Exemplo: **548₈**

$$548_8 = 5 \times 8^2 + 4 \times 8^1 + 8 \times 8^0$$

$$548_8 = 5 \times 64 + 4 \times 8 + 8 \times 1$$

$$548_8 = 320 + 32 + 8$$

$$548_8 = 360_{10}$$

Conversão Base X para Base 10

- Equação para conversão:

$$num_d = a_n x^n + a_{n-1} x^{n-1} + \dots + a_0 x^0$$

- Exemplos, Converter para a base 10:
 - 1011_2
 - $4A3B_{16}$
 - 7271_8

Conversão Base X para Base 10

$$num_d = a_n x^n + a_{n-1} x^{n-1} + \dots + a_0 x^0$$

- **Binário – Decimal:** 1011_2

$$1011_2 = 1x2^3 + 0x2^2 + 1x2^1 + 1x2^0$$

$$1011_2 = 1x8 + 0x4 + 1x2 + 1x1$$

$$1011_2 = 8 + 0 + 2 + 1 = 11_{10}$$

- **Octal – Decimal:** 7271_8

$$7271_8 = 7x8^3 + 2x8^2 + 7x8^1 + 1x8^0$$

$$7271_8 = 7x512 + 2x64 + 7x8 + 1x1$$

$$7271_8 = 3.584 + 128 + 56 + 1 = 3.769_{10}$$

- **Hexadecimal – Decimal:** $4A3B_{16}$

$$4A3B_{16} = 4x16^3 + Ax16^2 + 3x16^1 + Bx16^0$$

$$4A3B_{16} = 4x4096 + 10x256 + 3x16 + 11x1$$

$$4A3B_{16} = 16.384 + 2.560 + 48 + 11 = 19.003_{10}$$

Aritmética Binária

- Soma:

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 1 = 0 \text{ ("vai" 1 para o dígito com ordem superior)}$$

- Propriedades:

- Operações entre zeros, mantém-se a operação de forma neutra.
- A ordem das parcelas não altera o resultado da soma.

Aritmética Binária

- Soma:

Exemplo: $101 + 011 = 1000$

$$\begin{array}{r} 111 \\ 101 \\ +011 \\ \hline 1000 \end{array}$$

Aritmética Binária

- Subtração:

$$0 - 0 = 0$$

$$1 - 0 = 1$$

$$1 - 1 = 0$$

$$0 - 1 = 1 \text{ (pega 1 "empréstado" do dígito superior)}$$

- Propriedades:

- Operações entre zeros, mantém-se a operação de forma neutra.

Aritmética Binária

- Soma:

Exemplo: $101 - 011 = 010$

$$\begin{array}{r} \textcolor{red}{1} \\ 101 \\ -011 \\ \hline 010 \end{array}$$

Aritmética Binária

- Multiplicação:

$$0 \times 0 = 0$$

$$1 \times 0 = 0$$

$$1 \times 1 = 1$$

Propriedades:


- Operações entre zeros, mantém-se a operação de forma neutra.
- A ordem dos fatores não altera o produto.

Aritmética Binária

- Soma:

Exemplo: $101 \times 011 = 01111$

$$\begin{array}{r} 101 \\ \times 011 \\ \hline 101 \\ 101 \\ 000 \\ \hline \end{array}$$

Produto  01111

Aritmética Binária

- Divisão:

$$1 : 1 = 1$$

$$0 : 1 = 0$$

$$1 : 0 = \nexists$$

Propriedades:


- Não existe divisão por zero;

Aritmética Binária

- Divisão:

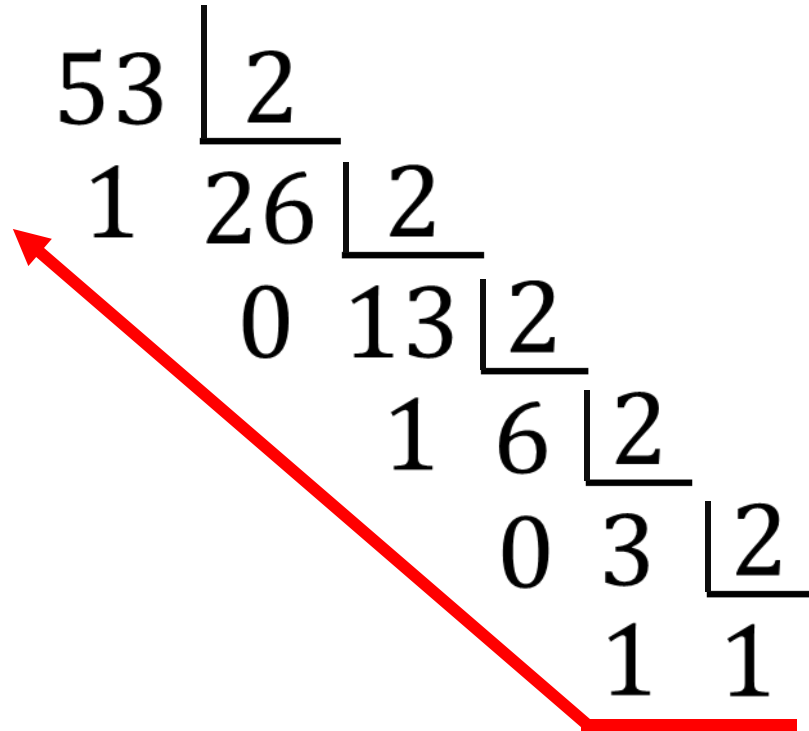
Exemplo: $101010 : 110 = 1000$

$$\begin{array}{r} 101010 \quad | \quad 110 \\ \underline{110} \\ 01001 \\ \underline{110} \\ 00110 \\ \underline{110} \\ 000 \end{array}$$

Produto 

Conversão Base 10 para Base 2

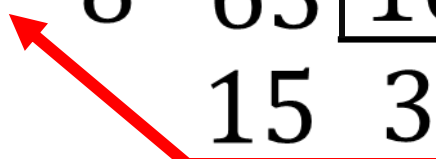
- Exemplo: $53_{10} = X_2$



$$53_{10} = 110101_2$$


Conversão Base 10 para Base 16

- Exemplos: $1016_{10} = X_{16}$

$$\begin{array}{r|l} 1016 & 16 \\ \hline 8 & 63 \\ 15 & 3 \end{array}$$
A red arrow originates from the bottom row of the division table (15 and 3) and points diagonally upwards and to the left towards the top row (8 and 63), indicating the sequence of remainders to be read from bottom to top.

$$1016_{10} = 3F8_{16}$$

- $53_{10} = X_{16}$

$$\begin{array}{r|l} 53 & 16 \\ \hline 5 & 3 \end{array}$$
A red arrow originates from the bottom row of the division table (5 and 3) and points diagonally upwards and to the left towards the top row, indicating the sequence of remainders to be read from bottom to top.

$$53_{10} = 35_{16}$$

Exercício

- Converta para:

$$25_{10} = X_2$$

$$156_{10} = B_{16}$$

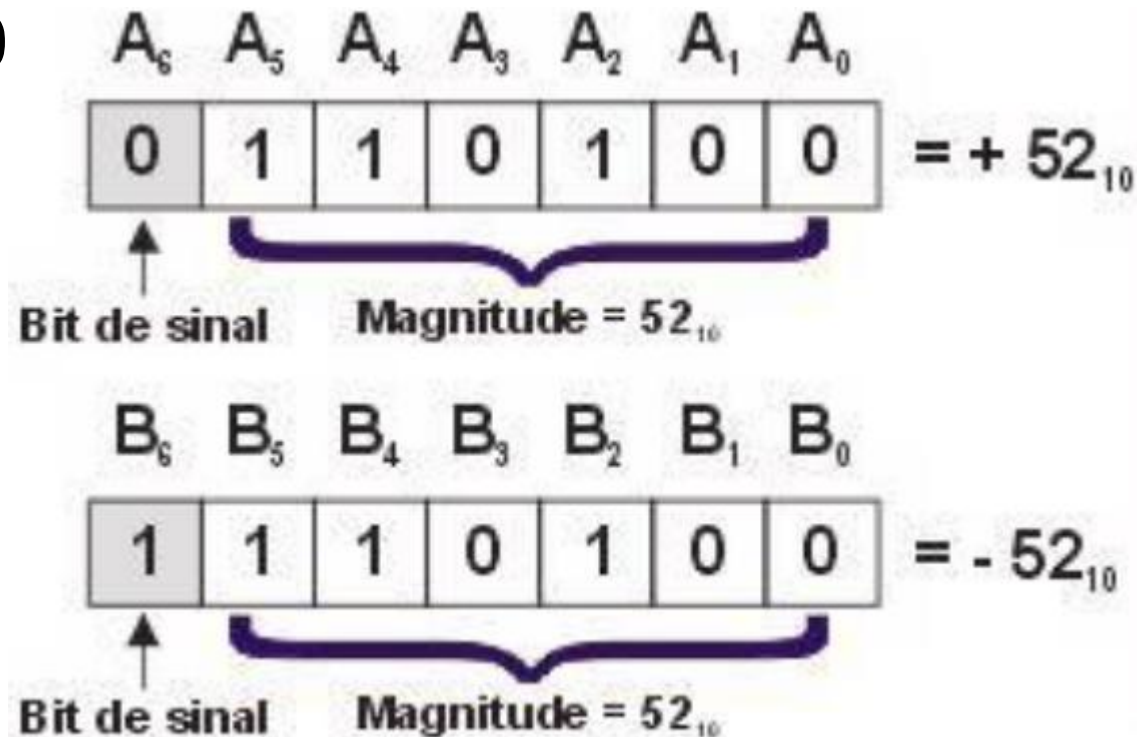
- Respostas:

$$25_{10} = 11001_2$$

$$156_{10} = 9C_{16}$$

Representação com Sinal

- Uso de sinal para representação: “+” para positivo e “–” para negativo;
- Em binário não é possível fazer uso de símbolos;
- Uso de zero “0” para positivo e “1” para negativo;



Representação de Soma

- Sinal diferente (+/-):
 - Idêntica o número de maior magnitude;
 - Subtrai o maior pelo menor;
 - Atribuir o sinal do número com maior magnitude;
- Sinais iguais (+/+ ou -/-):
 - Soma os dois números;
 - Atribuir o sinal dos operandos;
 - Observar o estouro da magnitude, tanto para mais quanto para menos;

Projeto em Sistemas

- Lógica em sistema é vista como complexa para as diversas condições que são submetidas (necessários vários testes);
- A nível de hardware operações aritméticas são execuções complicadas em sua execução;
- Operações:
 - Multiplicação realizadas por várias somas;
 - Exemplo: $4 \times 3 = 4 + 4 + 4$;
 - A divisão pode ser realizada por várias subtrações;
 - Exemplo: $9 : 3 = 9 - 3 - 3 - 3$;

Representação de Subtração

- É realizada através do método: **Complemento a Base**.
 - É nada mais que a diferença entre dois números.
- Objetivo é determinar o complemento do número em relação a sua base e depois realizar a soma dos números;
- Grande aplicabilidade na computação;
- Como os computadores realiza operações em Base 2, o complemento será em (**C2**).
- Números com bits mais a esquerda são utilizados para determinar negativo (1), positivo (0).
- Para isso é necessário saber a quantidade de bits que o número deve ter.
- Processo:
 - Os números negativos terão seus bits invertidos;
 - Soma um ao resultado final;

Complemento

- Exemplo: Complemento de 297_{10}

Número máximo para essa centena: 999_{10}

$$\text{Complemento: } \begin{array}{r} 999 \\ - 297 \\ \hline 702 \end{array}$$

Complemento

- Exemplo: Complemento de 0011_2

Número máximo para essa centena: 1111_2

$$\begin{array}{r} \text{Complemento: } \quad 1111 \\ \quad - 0011 \\ \hline \quad 1100 \end{array}$$

Complemento

- Exemplo: Complemento de $3A7E_{16}$

Número máximo para essa centena: $FFFF_{16}$


$$\text{Complemento: } \begin{array}{r} FFFF \\ - 3A7E \\ \hline C581 \end{array}$$

Quantidade de Representações

- Quantos números conseguimos representar:
 - Um bit:
 $2^1 = 2$ números (0,1)
 - Dois bits:
 $2^2 = 4$ *números* (00, 01, 10, 11)
 - Três bits:
 $2^3 = 8$ *números* (000, 001, 010, 011, 100, 101, 110, 111)

Representação de Complemento

- Tabela de complemento para 4 bits



Decimal Positivo	Binário sem sinal	Decimal Negativo	Binário (Complemento)
0	0000	0	1111
1	0001	-1	1110
2	0010	-2	1101
3	0011	-3	1100
4	0100	-4	1011
5	0101	-5	1010
6	0110	-6	1001
7	0111	-7	1000

OBS: Neste caso o zero tem duas representações

Representação de Complemento

Como solução para o complemento onde o zero possui duas representações, basta realizarmos a soma de um “1”.


Exemplo:

Complemento do número: 0011_2

$$\begin{array}{r} 1111 \\ - 0011 \\ \hline 1100 \\ + 0001 \\ \hline 1101 \end{array}$$

Representação de Complemento

Por fim, a representação de 4 bits fica:



Decimal Positivo	Binário sem sinal	Decimal Negativo	Binário (Complemento)
0	0000	-1	1111
1	0001	-2	1110
2	0010	-3	1101
3	0011	-4	1100
4	0100	-5	1011
5	0101	-6	1010
6	0110	-7	1001
7	0111	-8	1000

OBS: Resolvido o problema das duas representações e ganho de um n

Representação de Complemento

- Complementos para 4 bits:

$$\begin{array}{r} 0101 \quad 5 \\ 0110 \quad 6 \\ + \\ \hline 1011 \quad 11 \end{array}$$

Carry sobre o bit de sinal
-> **estouro = overflow**

$$\begin{array}{r} 0101 \quad 5 \\ 0010 \quad 2 \\ + \\ \hline 0111 \quad 7 \end{array}$$

Não houve *Carry* = **não overflow**

Representação de Complemento

- Complementos para 4 bits:

$$\begin{array}{r} 0101 \\ + 1010 \\ \hline 1111 \end{array}$$

$$\begin{array}{r} 5 \\ - 6 \\ \hline -1 \end{array}$$

Não houve *Carry* = **não overflow**

$$\begin{array}{r} 0110 \\ + 1011 \\ \hline 0001 \end{array}$$

$$\begin{array}{r} 6 \\ - 5 \\ \hline 1 \end{array}$$

Carry sobre o “bit de sinal” e após ele
= **não overflow**

$$\begin{array}{r} 1011 \\ + 1010 \\ \hline 0101 \end{array}$$

$$\begin{array}{r} -5 \\ - 6 \\ \hline -11 \end{array}$$

Carry somente após o “bit de sinal” = **overflow**