

Sistemas de Computação

Sistemas de numeração

Sistemas de numeração

- Permitem a escrita de números utilizando **dígitos** ou **símbolos** de **forma consistente**
- Define o contexto que permite interpretar o **valor** de uma dada combinação de **símbolos**
 - O número 11 pode ser interpretado como '3' em binário, 'onze' em decimal, '9' em octal, etc.

Base

- Número de dígitos (únicos) utilizados por um sistema de numeração para representar **números**
 - Base 10 utiliza 10 dígitos (0 ... 9) (decimal)
 - Base 2 utiliza 2 dígitos (0, 1) (binário)
 - Base 8 utiliza 8 dígitos (0 ... 7) (octal)
- Um **número**, numa dada base, só pode ser representado pelos **dígitos/símbolos** dessa **base**
 - 3_2 ou 9_8 não existem porque 3 e 9 não pertencem às respectivas bases

Base (cont.)

- O **valor** de um **número** é **sempre** calculado da mesma forma
 - $d_1 \times b^{n-1} + d_2 \times b^{n-2} + \dots + d_n \times b^0$
 - $d_1 \dots d_n$ são a sequência de símbolos do número
 - b a base em que o número está expresso
 - Ex: 237_8 tem o valor de $2 \times 8^2 + 3 \times 8^1 + 7 \times 8^0$
- Um mesmo **valor** pode ter representações diferentes conforme a base utilizada
 - Mas o **seu valor** é **sempre o mesmo**

Mudança de base

- $98_{10} = \text{????}_8 \rightarrow 98_{10} = d_1 \times 8^2 + d_2 \times 8^1 + d_3 \times 8^0$

$$\begin{array}{r} 98 \overline{) 8} \\ d_3 \rightarrow 2 \quad 12 \overline{) 8} \\ \quad \quad \quad \nearrow 4 \quad 1 \\ \quad \quad \quad \quad \uparrow \\ \quad \quad \quad \quad d_1 \end{array}$$

- $98_{10} = 142_8$

Mudança de base (Cont.)

- Divisão do número, **em base decimal**, pela nova base

$$\begin{array}{r} 98 \overline{) 8} \\ 2 \quad 12 \end{array}$$

- Divisão sucessiva dos quocientes pela nova base, enquanto este for divisível

$$\begin{array}{r} 98 \overline{) 8} \\ 2 \quad 12 \overline{) 8} \\ 4 \quad 1 \end{array}$$

- O número na nova base é a concatenação do último quociente com os restos anteriores

142

Mudança de base (Cont.)

- Mais exemplos:

– $98_{10} \rightarrow \text{?????}_2$

$$\begin{array}{r} 98 \overline{) 2} \\ 0 \quad 49 \overline{) 2} \\ 1 \quad 24 \overline{) 2} \\ 0 \quad 12 \overline{) 2} \\ 0 \quad 6 \overline{) 2} \\ 0 \quad 3 \overline{) 2} \\ 1 \quad 1 \end{array}$$

– $98_{10} \rightarrow 1100010_2$

– $1100010_2 = 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^1 = 98_{10}$

Mudança de base (Cont.)

- Alternativamente podemos utilizar potências
 - Tal como em decimal

- $1280_{10} = 1 \times 10^3 + 2 \times 10^2 + 8 \times 10^1 + 0 \times 10^0$

- Ex: $620_{10} \rightarrow \text{????}_{16}$

$$\begin{array}{cccc} 4096 & 256 & 16 & 1 \\ ? \times 16^3 & + ? \times 16^2 & + ? \times 16^1 & + ? \times 16^0 \end{array}$$

Mudança de base (Cont.)

- Ex: $620_{10} \rightarrow \text{????}_{16}$

$$\overset{4096}{?} \times 16^3 + \overset{256}{?} \times 16^2 + \overset{16}{?} \times 16^1 + \overset{1}{?} \times 16^0$$

$$620 < \textcolor{green}{1} \times \textcolor{red}{16}^3 \rightarrow \textcolor{green}{0} \times 16^3 + \dots$$

$$620 < \textcolor{green}{3} \times \textcolor{red}{16}^2 \rightarrow 0 \times 16^3 + \textcolor{green}{2} \times 16^2 + \dots$$

$$620 - (2 \times 256) = 108 < \textcolor{green}{7} \times \textcolor{red}{16}^1 \rightarrow 0 \times 16^3 + 2 \times 16^2 + \textcolor{green}{6} \times 16^1 + \dots$$

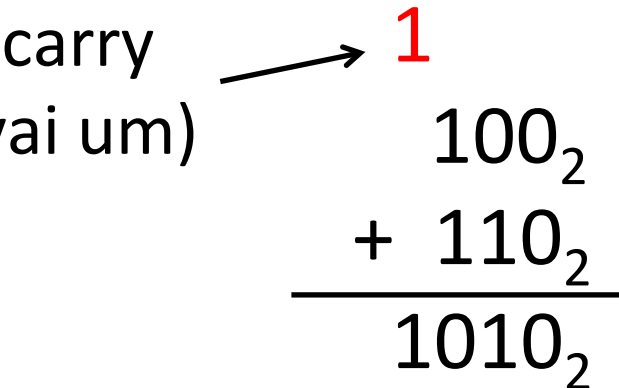
$$108 - (6 \times 16) = 12 < \textcolor{green}{13} \times \textcolor{red}{16}^0 \rightarrow 0 \times 16^3 + 2 \times 16^2 + 6 \times 16^1 + \textcolor{green}{12} \times 16^0$$

- $620_{10} = \textcolor{green}{26C}_{16}$

Aritmética

- As operações aritméticas são sempre iguais independentemente da base usada
 - Algoritmo e resultado
- Apenas a representação do número varia com a base
- Ex:
 - $2_{10} + 2_{10} = 4_{10}$
 - $10_2 + 10_2 = 100_2 \rightarrow 4_{10}$

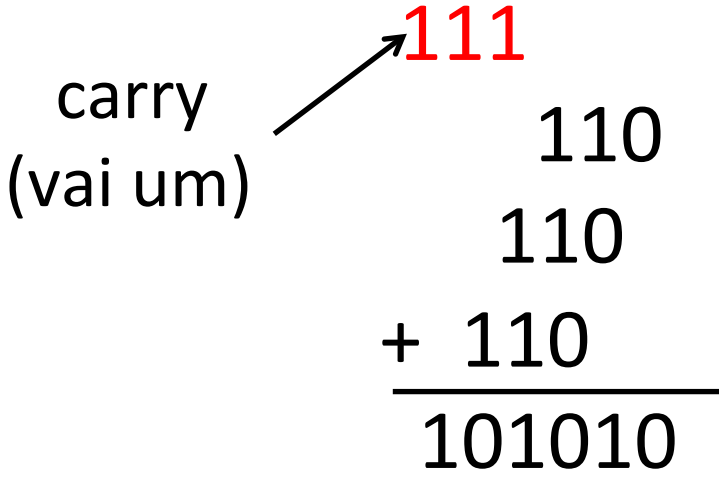
Aritmética

- $100_2 + 110_2 = \text{????}_2$ (vai um) 

$$\begin{array}{r} 100_2 \\ + 110_2 \\ \hline 1010_2 \end{array}$$
- $110_2 \times 111_2 = \text{?????}_2$

$$\begin{array}{r} 110_2 \\ \times 111_2 \\ \hline 110 \\ 110 \\ + 110 \\ \hline \end{array}$$

carry
(vai um)



$$\begin{array}{r} 110 \\ 110 \\ + 110 \\ \hline 101010 \end{array}$$

Números fraccionários

- Compostos por
 - parte inteira . parte fraccionária
- $0.75 = 0 \times 10^0 + 7 \times 10^{-1} + 5 \times 10^{-2}$
- Em decimal
... 10^4 10^3 10^2 10^1 10^0 . 10^{-1} 10^{-2} 10^{-3} ...

Números fraccionários(Cont.)

- Em binário

$$\dots 2^4 2^3 2^2 2^1 2^0 \cdot 2^{-1} 2^{-2} 2^{-3} 2^{-4} \dots$$

- Ex:

$$0.5_{10} = 1/2 = 2^{-1}$$

$$0.5_{10} \rightarrow 0.1_2$$

- $0.1_2 \rightarrow 1 \times 2^{-1} = 1/2 = 0.5_{10}$

Conversão números fraccionários

- $0.625_{10} = \text{????}_2$

$$\begin{array}{r} 0.625 \\ \times 2 \\ \hline 1.25 \end{array}$$
- $0.625_{10} = 101_2$

$$\begin{array}{r} \times 2 \\ \hline 0.5 \end{array}$$
- $$\begin{array}{r} \times 2 \\ \hline 1.0 \end{array}$$
- $0.8125_{10} = \text{????}_2$

$$0.8125 \times 2 = 1.625$$
$$0.625 \times 2 = 1.25$$
$$0.25 \times 2 = 0.5$$
$$0.5 \times 2 = 1.0$$
- $0.8125_{10} = 1101_2$

Conversão números fraccionários

- Atenção, em binário também podem existir dízimas infinitas

- $0.35_{10} = \text{????}_2$

$$0.35 \times 2 = 0.7$$

$$0.7 \times 2 = 1.4$$

$$0.4 \times 2 = 0.8$$

$$0.8 \times 2 = 1.6$$

$$0.6 \times 2 = 1.2$$

$$0.2 \times 2 = 0.4$$

$$0.4 \times 2 = 0.8$$

- $0.35_{10} = 0.01(0110)_2$

← Repetição

Números negativos em Binário

- Na base decimal os números negativos são representados com um '-' antes
 - Em binário o '-' não existe
- Mas podemos adicionar um bit extra para o sinal
 - Bit mais significativo (à esquerda) → bit do sinal
 - '0' → '+'; '1' → '-'

Números negativos em Binário

- Binário com sinal
 - Bit mais significativo reservado para o sinal
 - Restantes bits codificam o valor (como positivo)

Decimal	Binário	Binário c/ sinal
3	11	011
2	10	010
1	01	001
0	000	000
-1	-	101
-2	-	110
-3	-	111
-4	-	-

Números negativos em Binário

Decimal	Binário	Binário c/ sinal
3	11	011
2	10	010
1	01	001
0	000	000 / 100
-1	-	101
-2	-	110
-3	-	111
-4	-	-

- Utilizando esta codificação, se virem com atenção o número 100_2 também é '0'
 - Neste caso é (-0)

Resolver o problema do (-0)

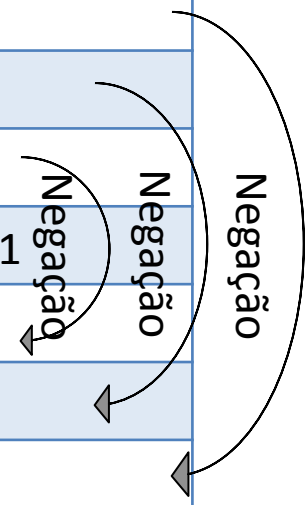
- Utiliza-se uma codificação em 'Complemento para 2'
 - Complemento para 2 \Rightarrow Complemento para 1 + 1_2

Complemento para 1

- Codificação binária com sinal
 - Bit mais significativo representa o sinal
 - '0' \rightarrow '+'; '1' \rightarrow '-'
- Números positivos codificados em binário
- Números negativos codificados como a **negação do positivo**
 - Inversão dos bits
 - Os 1's passam a 0's e vice-versa

Complemento para 1

Decimal	Binário	Binário c/ sinal	Complemento para 1
3	11	011	011
2	10	010	010
1	01	001	001
0	000	000 / 100	000 / 111
-1	-	101	110
-2	-	110	101
-3	-	111	100
-4	-	-	-



- Como dá para ver não resolve o problema dos “2 zeros”

Complemento para 2

- Codificação binária com sinal
 - Bit mais significativo representa o sinal
 - '0' \rightarrow '+'; '1' \rightarrow '-'
- Números positivos codificados em binário
- Números negativos codificados como a **negação do positivo + 1_2**

Complemento para 2

Decimal	Binário	Binário c/ sinal	Comp. para 1	Comp. para 2
3	11	011	011	011
2	10	010	010	010
1	01	001	001	001
0	000	000 / 100	000 / 111	000
-1	-	101	110 $\xrightarrow{+1_2}$	111
-2	-	110	101 $\xrightarrow{+1_2}$	110
-3	-	111	100 $\xrightarrow{+1_2}$	101
-4	-	-	-	100

- Como dá para ver resolve o problema dos “2 zeros” e ganhamos um número
 - Porquê?

Binary Coded Decimal (BCD)

- Sistema de codificação binário em que cada **dígito decimal** é codificado por **4 bits**
- Ex: $324_{10} \rightarrow 0011\ 0010\ 0100_{\text{BCD}}$
 - Atenção: $324_{10} \rightarrow 10100100_2$