

# Sistema Octal

---

<b>Decimal</b>	<b>Binário</b>	<b>Octal</b>
0	000	0
1	001	1
2	010	2
3	011	3
4	100	4
5	101	5
6	110	6
7	111	7

# Conversão Octal-Decimal

---

- ▶ Para converter um número octal em um número decimal, basta aplicar a fórmula genérica já conhecida :

- ▶  $N_{10} = S_y * 8^{n-1} + S_{y-1} * 8^{n-2} \dots S_1 * 8^1 + S_0 * 8^0$

**Exemplo 1:**  $X_{10} = 346_8$

Solução:  $3 * 8^2 + 4 * 8^1 + 6 * 8^0$

Solução:  $3 * 64 + 4 * 8 + 6 * 1 =$

$192 + 32 + 6 = 230_{10}$

Portanto:  $(346)_8 = 230_{10}$

**Exemplo 2:**  $X_{10} = 477_8$

Solução:  $4 * 8^2 + 7 * 8^1 + 7 * 8^0$

Solução:  $4 * 64 + 7 * 8 + 7 * 1 =$

$256 + 56 + 7 = 319_{10}$

Portanto:  $(477)_8 = 319_{10}$

# Conversão Decimal-octal

- ▶ O processo é idêntico à conversão Decimal - Binário, dividindo-se o número decimal pela base 8 até que o resultado seja zero.

$X_8=(90)_{10}$	90	8		
1º resto	→ 2	11	8	
2º resto	→	3	1	8
3º resto	→		1	0
$(90)_{10}=(132)_8$				

$X_8=(128)_{10}$	128	8		
1º resto	→ 0	16	8	
2º resto	→	0	2	8
3º resto	→		2	0
$(128)_{10}=(200)_8$				

# Conversão Binário-Octal

---

- ▶ A conversão Binário - Octal é feita transformando-se grupos de três dígitos binários, no sentido da direita para a esquerda, diretamente em números octais.

1	110	010
1	6	2

10	001
2	1

# Conversão Octal-binário

---

- ▶ A conversão de números Octais em Binários é feita transformando os símbolos Octais diretamente em números binários de 3 dígitos.

$$X_2 = (123)_8$$

1	2	3
---	---	---

001010011

$$(123)_8 = (001010011)_2 = (1010011)_2$$

# Conversão Octal-Hexadecimal

---

- ▶ Como fazer?

# Conversão Hexadecimal-Octal

---

## ► Como fazer?

N.º Decimal 10	N.º Binário 2	N.º Hexadecimal 16	N.º Octal 8
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17

## Converter Decimais Fracionários em Binário

---

- ▶ Multiplica por 2 a parte fracionária e retira o número depois da vírgula.
- ▶ Para a operação apenas quando encontrar o valor 1,00. Nesse caso, retira-se o 1 e finaliza-se a operação.
- ▶ Como representar 8,7?



# Converter Decimais Fracionários em Binário

$$\begin{array}{r} 8,7 \longrightarrow \begin{array}{r} 8 \overline{) 2} \\ 0 \quad 4 \overline{) 2} \\ 0 \quad 2 \overline{) 2} \\ 0 \quad 1 \end{array} \\ \hline 1000,??? \end{array}$$

# Converter Decimais Fracionários em Binário

$$\begin{array}{r} 8,7 \longrightarrow 8 \overline{) 2} \\ \underline{0} \phantom{0} 4 \overline{) 2} \\ \underline{0} \phantom{0} 2 \overline{) 2} \\ \underline{0} \phantom{0} 1 \end{array}$$

---

1 0 0 0 , ? ? ?

---

$$2 \times 0,7 = 1,4 \longrightarrow 1$$

$$2 \times 0,4 = 0,8 \longrightarrow 0$$

$$2 \times 0,8 = 1,6 \longrightarrow 1$$

$$2 \times 0,6 = 1,2 \longrightarrow 1$$

---

1 0 0 0 , 1 0 1 1

---

# Converter Binário Fracionários em Decimal

$$\begin{array}{r} 1 \ 0 \ 0 \ 0 , 1 \ 0 \ 1 \ 1 \\ \text{3} \ \text{2} \ \text{1} \ \text{0} \ \text{-1} \ \text{-2} \ \text{-3} \ \text{-4} \\ \hline 1 \times 2^3 = 8 \\ \hline 1 \times 2^{-1} = \frac{1}{2} = 0,5 \\ 1 \times 2^{-3} = \frac{1}{8} = 0,125 \\ 1 \times 2^{-4} = \frac{1}{16} = 0,0625 \end{array}$$

## Converter Binário Fracionários em Decimal

$$\begin{array}{r} + \quad 0,5 \\ \quad 0,125 \\ \quad 0,0625 \\ \hline \end{array}$$

0,6875

$$\begin{array}{r} + \quad 0,6875 \\ \quad 8,0 \\ \hline \end{array}$$

8,6875




Disciplina:

# Programação Computacional

Prof. Fernando Rodrigues

e-m@il: fernandorodrigues@sobral.ufc.br

## Aula 04\_B: Sistemas de numeração:

- ❖ Operações aritméticas básicas no sistema binário:
  - ❖ Adição
  - ❖ Subtração
  - ❖ Multiplicação
  - ❖ Divisão
- 

## Operações Aritméticas Básicas no Sistema Binário

# Soma (Adição) de Números Binários

---

## ► Regras

- $0+0=0$ .
- $0+1=1$ .
- $1+0=1$ .
- $1+1=10$ , ou seja, 0 e “vai 1” para somar com dígito à esquerda (de ordem superior).
- $1 + 1 + 1 = 1$  (e “vai 1” para o dígito de ordem superior)

```
  11010
+ 10011
-----
```

# Soma (Adição) de Números Binários

---

## ► Regras

- $0+0=0$ .
- $0+1=1$ .
- $1+0=1$ .
- $1+1=10$ , ou seja, 0 e “vai 1” para somar com dígito à esquerda (de ordem superior).
- $1 + 1 + 1 = 1$  (e “vai 1” para o dígito de ordem superior)

$$\begin{array}{r} \phantom{1} \\ 11010 \\ + 10011 \\ \hline 101101 \end{array}$$

$$\begin{array}{r} 1 \quad 1 \quad 1 \\ \downarrow \downarrow \downarrow \\ \begin{array}{r} 1 \quad 0 \quad 1_2 \\ + 0 \quad 1 \quad 1_2 \\ \hline 1 \quad 0 \quad 0 \quad 0_2 \end{array} \end{array}$$



# Subtração de Números Binários

---

## ► Regras

- $0 - 0 = 0$ .
- $0 - 1 = 1$  e “vai 1” (“pede emprestado 1”) para ser subtraído do dígito à esquerda (de ordem superior).
- $1 - 0 = 1$ .
- $1 - 1 = 0$ .

```
  11010
-  10011
-----
```

# Subtração de Números Binários

---

## ► Regras

- $0 - 0 = 0$ .
- $0 - 1 = 1$  e “vai 1” (“pede emprestado 1”) para ser subtraído do dígito à esquerda (de ordem superior).
- $1 - 0 = 1$ .
- $1 - 1 = 0$ .

```
  11010
-  10011
-----
   00111
```

# Multiplicação de Números Binários

---

- ▶  $0 * 0 = 0.$
- ▶  $0 * 1 = 0.$
- ▶  $1 * 0 = 0.$
- ▶  $1 * 1 = 1.$
- ▶ A multiplicação em binário é similar à multiplicação em decimal.

```
  11
x 101
-----
```

```
 1101
x 111
-----
```

# Multiplicação de Números Binários

---

- ▶  $0 * 0 = 0$ .
  - ▶  $0 * 1 = 0$ .
  - ▶  $1 * 0 = 0$ .
  - ▶  $1 * 1 = 1$ .
- 
- ▶ A multiplicação em binário é similar à multiplicação em decimal.

```
  11
x 101
-----
  11
 00+
 11+
 ----
 1111
```

```
 1101
x 111
-----
 1101
1101+
1101+
-----
1011011
```

# Divisão Binária

---

- ▶ Mesmo método que o decimal: deslocamentos e subtrações.

$$\begin{array}{r} 10100 \mid 101 \\ \underline{\phantom{10100}101} \\ 000 \end{array}$$

$$\begin{array}{r} 101010 \mid 110 \\ \underline{110} \\ 1001 \\ \underline{110} \\ 0110 \\ \underline{110} \\ 000 \end{array}$$

## Multiplicação e Divisão Binárias – Obs.

---

- ▶ Na maioria dos circuitos lógicos, não existem as operações de multiplicação e divisão binárias: o processador trabalha com somas (para a multiplicação) e subtrações (para a divisão) sucessivas.
- ▶ Por exemplo: para fazer a operação  $2 \times 5$ , o processador vai somar cinco vezes o número dois.
- ▶ Da mesma forma, para realizar a operação  $10 / 2$ , o processador subtrai o valor dois (do número dez) até que o resultado seja zero.




Disciplina:

# Programação Computacional

Prof. Fernando Rodrigues

e-mail: fernandorodrigues@sobral.ufc.br

## Aula 04\_C: Sistemas de numeração:

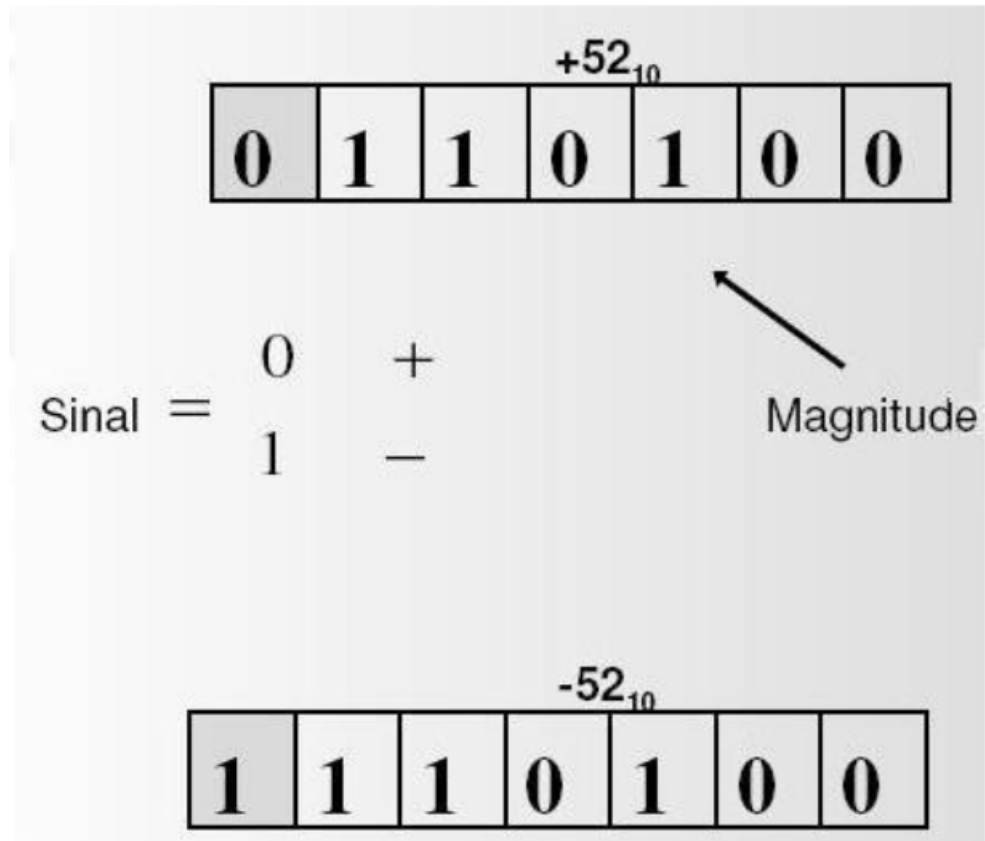
- ❖ Representação de Números Binários com Sinal:
  - ❖ Sistema Sinal-Magnitude (ou Representação Direta);
  - ❖ Sistema de Representação Binário em Complemento de 2 (dois).
- 

# Representação de Números Binários com Sinal



# Sistema sinal-magnitude

---



# Sistema sinal-magnitude

---

- ▶ Algoritmo de soma (números com sinal):
  - ▶ Sinais diferentes
    - ▶ Encontra número com maior magnitude
    - ▶ Subtrai menor do maior
    - ▶ Atribui ao resultado o sinal do número de maior magnitude
  - ▶ Sinais iguais
    - ▶ Soma e atribui sinal dos operandos
    - ▶ Atenção deve ser dada ao estouro de magnitude
    - ▶ Algoritmo de soma (números com sinal)

# Questões de projeto de circuitos lógicos

---

- ▶ Algoritmo do sistema sinal-magnitude: lógica complexa por conta das diversas condições (requer vários testes) e leva a aritmética complicada em termos de hardware.
- ▶ Também a multiplicação em computadores é feita por um artifício: para multiplicar um número  $A$  por  $n$ , basta somar  $A$  com  $A$ ,  $n$  vezes. Por exemplo,  $4 \times 3 = 4 + 4 + 4$ .
- ▶ E a divisão também pode ser feita por subtrações sucessivas.

# Complemento a Base

---

- ▶ Nos computadores, a subtração em binário é feita por um artifício: o "Método do Complemento a Base".
- ▶ Consiste em encontrar o complemento do número em relação a base e depois somar os números.
- ▶ Os computadores funcionam sempre na base 2, portanto o complemento a base será complemento a dois.

# Representação de números em complemento

---

- ▶ Complemento é a diferença entre o maior algarismo possível na base e cada algarismo do número.
- ▶ Através da representação em complemento, a subtração entre dois números pode ser substituída pela sua soma em complemento.
- ▶ A representação de números positivos em complemento é idêntica à representação em sinal e magnitude.

# Sistema de Representação Binário em Complemento de 2

# Complemento de 2: Conceito

---

- ▶ A representação de Complemento de 2 é usada para representar números negativos (bit de sinal (ou MSB) = 1).
- ▶ O complemento de dois de um número de N bits é definido como o complemento em relação a  $2^N$ .
- ▶ Para calcular o complemento de dois de um número, basta subtrair este número de  $2^N$ , que em binário é representado por 1 (um) seguido de N zeros.
- ▶ Por exemplo: 0110 (6 em binário com N=4 bits)
- ▶ Pegamos  $2^N$ , que é 10000\_ e subtraímos  
      0110    
    1010 e então fazemos o MSB ser 1:  
    11010 (-6 em complemento de 2)

# Complemento de 2: sinal e magnitude

---

- ▶ Definimos números positivos como aqueles que possuem o MSB igual a 0.
- ▶ Números negativos são definidos da seguinte forma: inverte todos os bits do número positivo e soma 1 ao resultado (Conversão mais usada).
  - Ex:  $6_{10}$  na base 2 =  $00110_2 \Rightarrow -6$  ??
  - Complemento de 1 =  $11001$  (Inverte todos os bits)
  - Complemento de 2 =  $\underline{\quad}1 (+)$  (Soma 1 ao resultado)
  - $11010$
- ▶ Existe outra maneira de calcular o complemento de dois:
  - ▶ Dado o número binário 00110.
  - ▶ Começando da direita para esquerda você vai repetindo o número (para a esquerda) até encontrar o número 1, depois que encontrá-lo repita-o e passe a inverter o restante. Então temos: **11010**, ou seja, **bits repetidos** e **bits invertidos**.



## Complemento de 2: Observações

---

- ▶ Por que usar complemento de 2?
  - Operações de subtração implementadas como soma binária com números negativos:
    - Sistemas computacionais mais simples, que apresentam somente circuito somador binário, sem a necessidade de um circuito subtrator.
  - Muita atenção para o tamanho da representação!!!
  - O complemento do complemento é sempre igual ao número original. P. ex.: O complemento de 0110 é igual a 1010. Por sua vez, o complemento de 1010 é 0110. Ou seja, volta ao valor original.

# Complemento de 2 – Exemplo prático

---

- ▶ Ex:
  - ▣  $10_{10}$  na base 2 =  $1010_2$
  - ▣  $6_{10}$  na base 2 =  $0110_2$
- ▶ Em Complemento de 2:
  - ▣  $+10 = 01010_2$
  - ▣  $-6 = 11010_2$
- ▶ Faça  $(+10) + (-6)$
- ▶ Se resultado vai 1, resultado positivo;
- ▶ Se resultado não vai 1, resultado negativo: para converter para número decimal, precisa tirar complemento antes.

# Complemento de 2 – Notação

Decimal	Sem sinal	Sinal-e-magnitude	Complemento para um	Complemento de dois
+16	–	–	–	–
+15	1111	–	–	–
+14	1110	–	–	–
+13	1101	–	–	–
+12	1100	–	–	–
+11	1011	–	–	–
+10	1010	–	–	–
+9	1001	–	–	–
+8	1000	–	–	–
+7	0111	0111	0111	0111
+6	0110	0110	0110	0110
+5	0101	0101	0101	0101
+4	0100	0100	0100	0100
+3	0011	0011	0011	0011
+2	0010	0010	0010	0010
+1	0001	0001	0001	0001
+0	–	0000	0000	–
0	0000	–	–	0000
–0	–	1000	1111	–
–1	–	1001	1110	1111
–2	–	1010	1101	1110
–3	–	1011	1100	1101
–4	–	1100	1011	1100

# Complemento de 2

- ▶  $5 - 3 = 2$
- ▶  $3 - 5 = -2$

Decimal	Sem sinal	Sinal-e-magnitude	Complemento para um	Complemento de dois
+16	-	-	-	-
+15	1111	-	-	-
+14	1110	-	-	-
+13	1101	-	-	-
+12	1100	-	-	-
+11	1011	-	-	-
+10	1010	-	-	-
+9	1001	-	-	-
+8	1000	-	-	-
+7	0111	0111	0111	0111
+6	0110	0110	0110	0110
+5	0101	0101	0101	0101
+4	0100	0100	0100	0100
+3	0011	0011	0011	0011
+2	0010	0010	0010	0010
+1	0001	0001	0001	0001
+0	-	0000	0000	-
0	0000	-	-	0000
-0	-	1000	1111	-
-1	-	1001	1110	1111
-2	-	1010	1101	1110
-3	-	1011	1100	1101
-4	-	1100	1011	1100

Decimal	Binário s/ sinal	Binário (Compl. 2)
-8	-	1000
-7	-	1001
-6	-	1010
-5	-	1011
-4	-	1100
-3	-	1101
-2	-	1110
-1	-	1111
0	000	0000
1	001	0001
2	010	0010
3	011	0011
4	100	0100
5	101	0101
6	110	0110
7	111	0111

# Conclusões

---

- ▶ Qualquer operação aritmética pode ser realizada em computadores apenas através de somas (diretas ou em complemento)!
- ▶ Em circuitos lógicos, será visto como essas propriedades serão úteis para os engenheiros que projetam os computadores.

# Referências

---

- ▶ [https://www.inf.ufes.br/~zegonc/material/Introducao\\_a\\_Computacao/Aritmetica\\_binaria\\_Complemento.pdf](https://www.inf.ufes.br/~zegonc/material/Introducao_a_Computacao/Aritmetica_binaria_Complemento.pdf)



Fim