

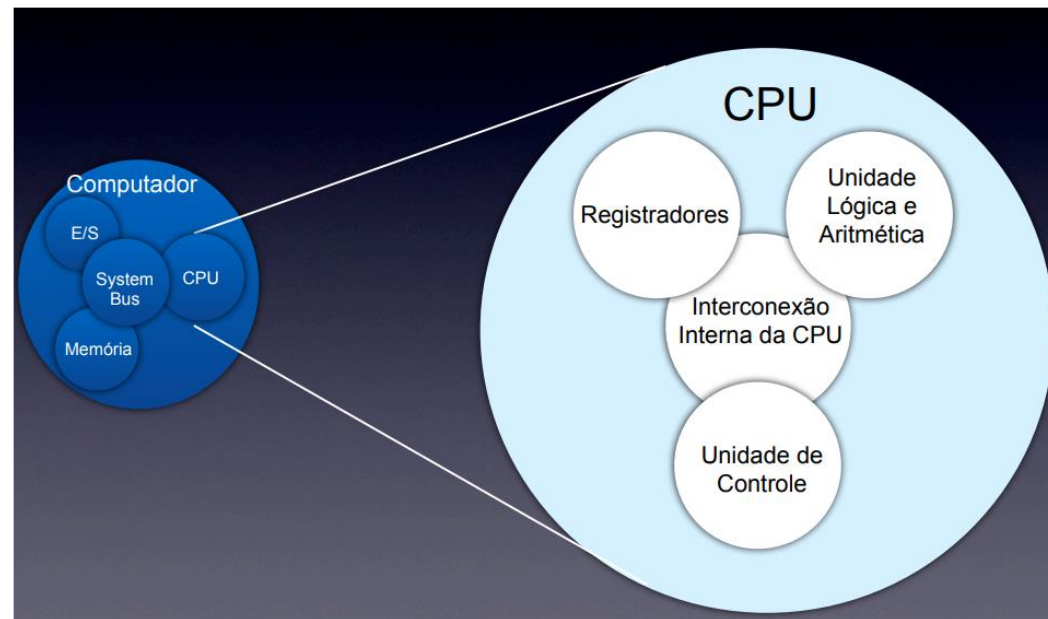
Arquitetura e organização de computadores

Uma visão geral

SIAC 202 - Arquitetura de Computadores
Prof.: Félix do Rêgo Barros
felixregobarros@gmail.com

Baseado em W. Stallings 8ª edição – Arquitetura e Organização de Computadores

- Sistemas Hierárquico
 - Um computador pode ser visto como um sistema formado por um conjunto estruturado de componentes, e sua função pode ser compreendida em termos das funções desses componentes.
 - Cada componentes, por sua vez, pode ser descrito em termos de sua estrutura e função internas.



Objetivo da disciplina

- Estudo de um sistema de computação sob dois pontos de vista:
 - **arquitetura** - se refere aos atributos do sistema visíveis a um programador de linguagem de máquina;
 - **organização** - as unidades operacionais e sua interconexão que realizam a arquitetura.
- Vamos estudar a estrutura e a função de um computador
 - **estrutura** - a forma em que os componentes estão interconectados e
 - **função** - a operação de cada componente individualmente. Cada componente pode, por sua vez, de forma hierárquica, ser decomposto em subcomponentes, descrevendo a sua estrutura e função.

Arquitetura e organização

- Definição
 - **Arquitetura de computador:** refere-se aos atributos de um sistema visíveis a um programador, com um impacto direto na execução de um programa. Exemplos de atributos arquiteturais: conjunto de instruções (instruction set), número de bits usados para representar vários tipos de dados, mecanismos de entrada e saída, e técnicas de endereçamento de memória.
 - **Organização de computador:** refere-se às unidades operacionais e sua interconexão que realizam as especificações arquiteturais. Exemplos de atributos organizacionais: detalhes de hardware transparentes ao programador, tais como sinais de controle, interface entre o computador e os periféricos, tecnologia de memória usada, etc.

Estrutura e função

- Um computador possui milhões de componentes eletrônicos.
- Um sistema hierárquico é um conjunto de subsistemas inter-relacionados, cada um destes, por sua vez, hierárquico em estrutura até alcançarmos algum nível mais baixo de subsistemas elementar.
- Como vamos descrever um computador?
- Usamos o enfoque hierárquico. O projetista se preocupa com a descrição um nível por vez, descrevendo os componentes e sua interconexão. Os níveis são descritos de forma (top-down) (Weinberg, 1975), descrevendo-se os componentes de um nível, depois os de seus subníveis, e assim por diante.

Estrutura e função

Em cada nível o projetista se preocupa com a estrutura e a função.

- **Estrutura:** a maneira em que os componentes são inter-relacionados.
- **Função:** a operação de cada componente individual como parte da estrutura.
 - Exemplos de funções básicas de computador pode desempenhar:
 - armazenado de dados
 - transferências de dados
 - processamento de dados
 - controle. .

Estrutura e função

Processamento de dados: Os dados podem assumir muitas formas e o intervalo de requisitos de processamento é amplo. Porém, veremos que existem apenas alguns métodos fundamentais ou tipos de processamento de dados.

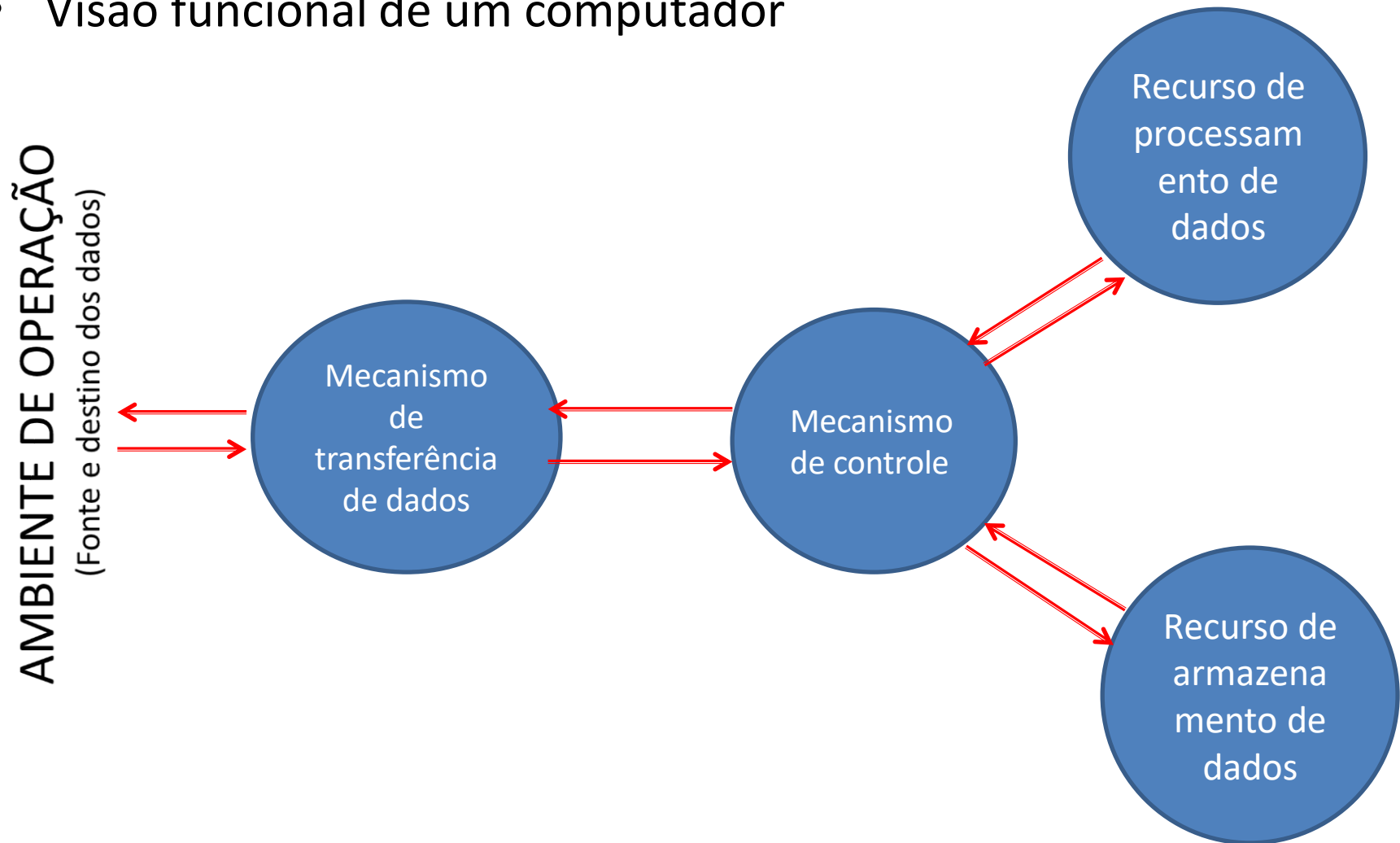
Armazenamento: o computador precisa armazenar temporariamente pelo menos as partes dos dados que estão sendo trabalhadas em determinado momento. Assim, existe pelo menos uma função de armazenamento de dados a curto prazo. Igualmente importante, o computador realiza uma função de armazenamento de dados a longo prazo. Os arquivos de dados são armazenados no computador para subsequente recuperação e atualização.

Movimentação de dados: fundamental a movimentação de dados entre ele e o mundo exterior. O ambiente operacional do computador consiste em dispositivos que servem como suas origens ou destinos de dados. Quando os dados são recebidos ou entregues a um dispositivo conectado diretamente ao computador, o **processo é conhecido como entrada/saída (E/S)**, e o dispositivo é referenciado como um **periférico**. Quando os dados são movimentados por distâncias maiores, de ou para um dispositivo remoto, o processo é conhecido como **comunicações de dados**.

Controle: Finalmente, é preciso haver controle dessas três funções, e esse controle é exercido por quem fornece instruções ao computador. Dentro do computador, uma unidade de controle gerencia os recursos do computador e coordena o desempenho de suas partes funcionais em resposta a essas instruções. Nesse nível geral de discussão, o número de operações que podem ser realizadas abaixo.

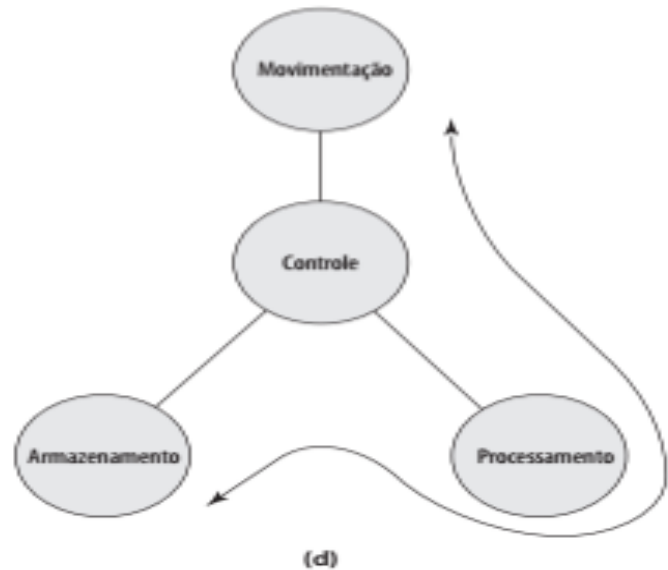
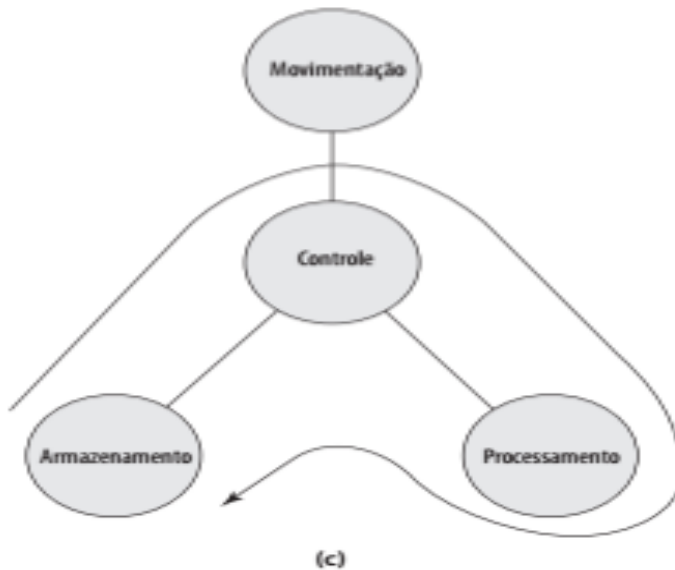
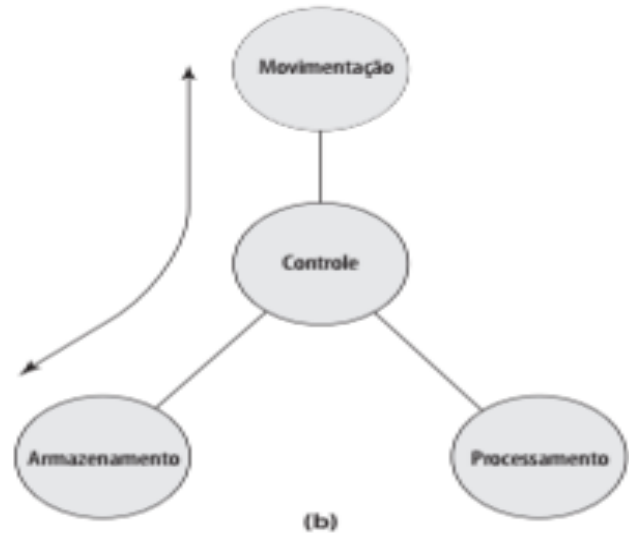
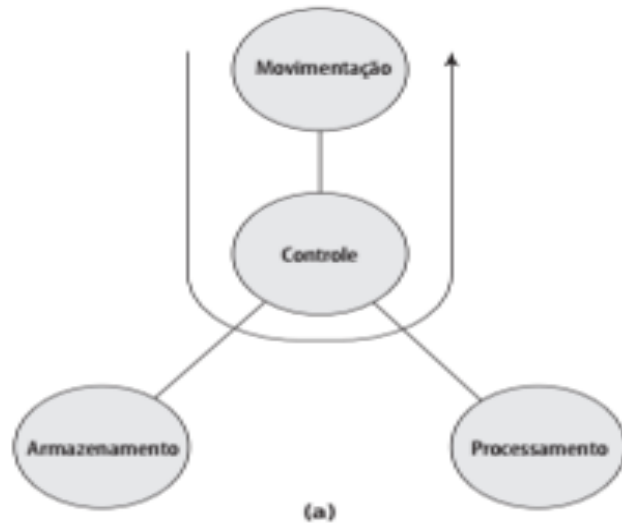
Estrutura e função

- Visão funcional de um computador



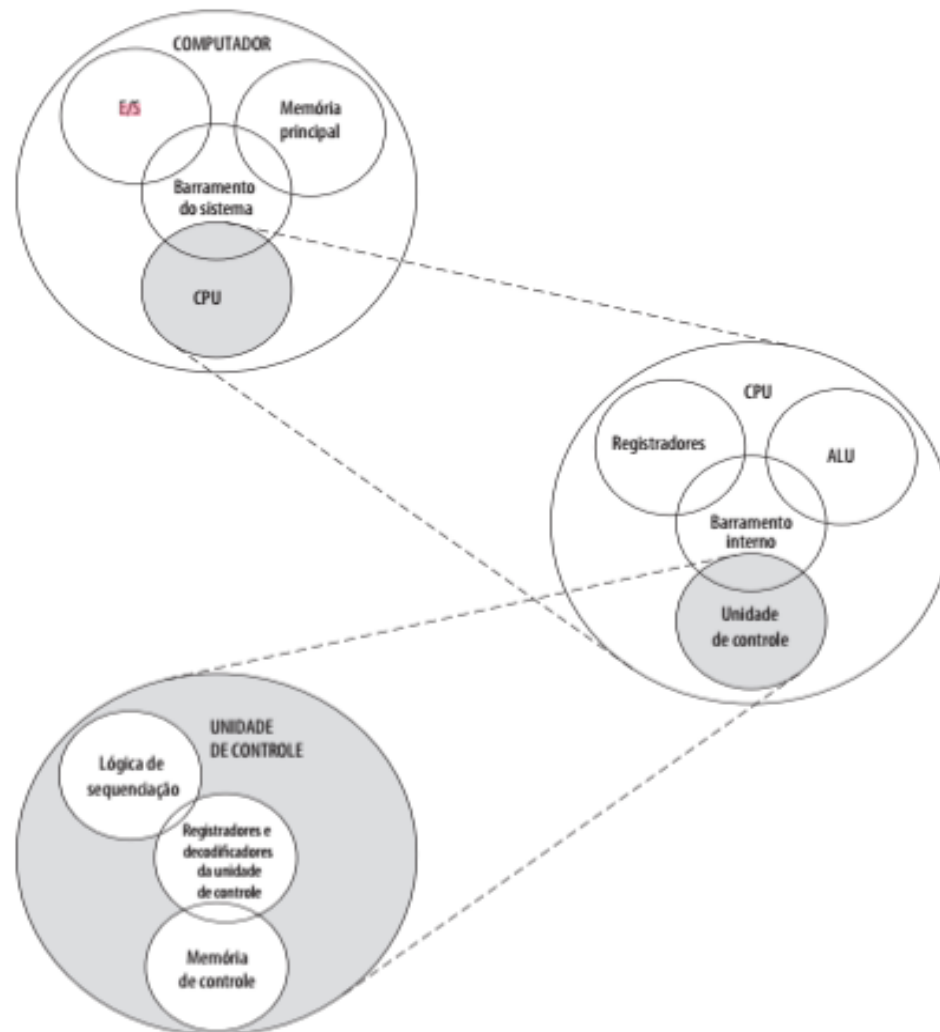
Estrutura e função

- Operações possíveis do computador



Estrutura e função

- Estrutura de alto nível



Estrutura e função



São quatro componentes estruturais principais:

- **Unidade central de processamento (CPU):** Controla a operação do computador e realiza suas funções de processamento de dados; normalmente é chamado apenas de processador.
- **Memória principal:** armazena dados.
- **E/S:** move dados entre o computador e seu ambiente externo.
- **Interconexão do sistema:** algum mecanismo que oferece comunicação entre CPU, memória principal e E/S.

Unidade de controle: controla a operação da CPU e, portanto, do computador.

Unidade aritmética e lógica (ALU, do inglês arithmetic and logic unit): realiza as funções de processamento de dados do computador.

Registradores: oferece armazenamento interno à CPU.

Interconexão da cpu: algum mecanismo que oferece comunicação entre unidade de controle, ALU e registradores.

- A evolução dos computadores tem sido caracterizada:
 - Aumento na velocidade do processador;
 - Diminuição no tamanho do componente;
 - Aumento no tamanho da memória;
 - Aumento na capacidade e velocidade da E/S.

Evolução e desempenho do Computador

- A evolução dos computadores tem sido caracterizada:
 - Aumento na velocidade do processador;
 - Diminuição no tamanho do componente;
 - Aumento no tamanho da memória;
 - Aumento na capacidade e velocidade da E/S.

Evolução e desempenho do Computador

Um fator responsável pelo grande aumento na velocidade do processador é o encolhimento no tamanho dos componentes do microprocessador; isso reduz a distância entre os componentes e, portanto, aumenta a velocidade. Contudo, os verdadeiros ganhos na velocidade nos anos recentes têm vindo da organização do processador, incluindo o uso intenso das **técnicas de pipeline** e **execução paralela** e do uso de **técnicas de execução especulativas** (tentativa de execução de instruções futuras que poderiam ser necessárias). Todas essas técnicas são projetadas para manter o processador ocupado pelo máximo de tempo possível.

Uma questão crítica no projeto de sistema de computador é equilibrar o desempenho dos diversos elementos de modo que os ganhos no desempenho em uma área não sejam prejudicados por um atraso em outras áreas. Em particular, a velocidade do processador aumentou mais rapidamente do que o tempo de acesso da memória. Diversas técnicas são usadas para compensar essa divergência, incluindo **caches**, **caminhos de dados mais largos da memória ao processador**, e **chips de memória mais inteligentes**.

Um breve histórico dos computadores

- Pré História: **Ábaco**
- Primeira Geração: **Válvula**
- Segunda Geração: **Transístores**
- Terceira Geração: **Circuitos Integrados**
- Quarta Geração: **Microprocessador**
- Quinta Geração: **Inteligência Artificial**

[http://producao.virtual.ufpb.br/books/camyl
e/introducao-a-computacao-
livro/livro/livro.chunked/ch01s02.html](http://producao.virtual.ufpb.br/books/camyl
e/introducao-a-computacao-
livro/livro/livro.chunked/ch01s02.html)

- [https://prezi.com/t7wjeku-9r2a/a-6-geracao-
de-computadores/](https://prezi.com/t7wjeku-9r2a/a-6-geracao-de-computadores/)

Aula 2 – Sistemas de Numeração

- **Objetivo**

Compreender o sistema de numeração utilizado pelos sistemas computacionais

Arquitetura e organização de computadores
Uma visão geral

Sistemas de Numeração

Aula 2

2.1 Bases e sistemas de numeração

Desde o início de sua existência, o homem sentiu a necessidade de contar objetos, fazer divisões, diminuir, somar, entre outras operações aritméticas de que hoje se tem conhecimento. Diversas formas de contagem e representação de valores foram propostas. Podemos dizer que a forma mais utilizada para a representação numérica é a **notação posicional**.

Segundo Monteiro (2007), na notação posicional, os algarismos componentes de um número assumem valores diferentes, dependendo de sua posição relativa nele. O valor total do número é a soma dos valores relativos de cada algarismo. Dessa forma, dependendo do sistema de numeração adotado, é dito que a quantidade de algarismos que o compõem é denominada base. Assim, a partir do conceito de notação posicional, tornou-se possível a conversão entre diferentes bases.

2.1.1 Notação posicional

A notação posicional é uma consequência da utilização dos numerais **hindu-arábicos**. Os números romanos, por exemplo, não utilizam a notação posicional. Desejando efetuar uma operação de soma ou subtração, basta colocar um número acima do outro e efetuar a operação desejada entre os numerais, obedecendo a sua ordem. A civilização ocidental adotou um sistema de numeração que possui dez algarismos (0, 1, 2, 3, 4, 5, 6, 7, 8 e 9), denominado de sistema decimal.

A quantidade de algarismos de um dado sistema é chamada de *base*; portanto, no sistema decimal a base é 10. O sistema binário possui apenas dois algarismos (0 e 1), sendo que sua base é 2.

Aula 2 – Sistemas de Numeração

2.1.1 Notação posicional

Exemplos:

$$4325_{10} = 5 \times 10^0 + 2 \times 10^1 + 3 \times 10^2 + 4 \times 10^3$$

$$1011_2 = 1 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 = 1 + 2 + 0 + 8 = 11_{10}$$

$$3621_8 = 1 \times 8^0 + 2 \times 8^1 + 6 \times 8^2 + 3 \times 8^3 = 1937_{10}$$

$$1A7B_{16} = 11 \times 16^0 + 7 \times 16^1 + 10 \times 16^2 + 1 \times 16^3 = 6779_{10}$$

Generalizando, num sistema de numeração posicional qualquer, um número

N é expresso da seguinte forma:

$$N = d_{n-1} \times b^{n-1} + d_{n-2} \times b^{n-2} + \dots + d_1 \times b^1 + d_0 \times b^0$$

Observações importantes:

- O número de algarismos diferentes em uma base é igual à própria base.
- Em uma base “**b**” e utilizando “**n**” ordens temos **bn** números diferentes

Aula 2 – Sistemas de Numeração

2.1.2 Conversão de bases

Equivalência entre as bases decimal, binária, octal hexadecimal.

Binário	Octal	Decimal	Hexadecimal
0	0	0	0
1	1	1	1
10	2	2	2
11	3	3	3
100	4	4	4
101	5	5	5
110	6	6	6
111	7	7	7
1000	10	8	8
1001	11	9	9
1010	12	10	A
1011	13	11	B
1100	14	12	C
1101	15	13	D
1110	16	14	E
1111	17	15	F
10000	20	16	10
10001	21	17	11
10010	22	18	12
10011	23	19	13
10100	24	20	14
10101	25	21	15

Fonte: Adaptado de Monteiro (2007)

Aula 2 – Sistemas de Numeração

2.1.2.1 Base binária para base octal ou hexadecimal

Observe que os dígitos octais e hexadecimais correspondem à combinações de 3 (para octais) e 4 (para hexadecimais) bits (ou seja, da representação binária – disponível na tabela de equivalências apresentada anteriormente), permitindo a fácil conversão entre estes sistemas.

$$\begin{array}{ccccccc} \underbrace{111}_{7} & \underbrace{010}_{2} & \underbrace{100}_{4} & \underbrace{101}_{5} & = 7245_8 \end{array}$$

$$\begin{array}{ccccccc} \underbrace{1111}_{F} & \underbrace{1010}_{A} & \underbrace{1101}_{D} & = \text{FAD}_{16} \end{array}$$

Aula 2 – Sistemas de Numeração

2.1.2.2 Base octal ou hexadecimal para base binária

conversão inversa de octal ou hexadecimal para binário deve ser feita a partir da representação binária de cada algarismo do número, seja octal ou hexadecimal.

$$7245_8 = \underbrace{111}_7 \underbrace{010}_2 \underbrace{100}_4 \underbrace{101}_5$$

$$\text{FAD}_{16} = \underbrace{1111}_F \underbrace{1010}_A \underbrace{1101}_D$$

Aula 2 – Sistemas de Numeração

2.1.2.3 Base octal para base hexadecimal (e vice-versa)

A representação binária de um número octal é idêntica à representação binária de um número hexadecimal, a conversão de um número octal para hexadecimal consiste simplesmente em agrupar os *bits* não mais de três em três (octal), mas sim de quatro em quatro *bits* (hexadecimal), e vice-versa.

$$(3174)_8 = \underbrace{011}_3 \underbrace{001}_1 \underbrace{111}_7 \underbrace{100}_4$$

$$(011001111100)_2 = \underbrace{0110}_6 \underbrace{0111}_7 \underbrace{1100}_c$$

$$(67C)_{16}$$

Aula 2 – Sistemas de Numeração

2.1.2.4 Base B (qualquer) para base decimal

Atenção, nos exemplos de casos citados a seguir, sempre utilizamos a definição de Notação Posicional:

$$101101_2 = ?_{10}$$

$$b = 2, n = 6$$

$$\text{Portanto: } 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 32 + 8 + 4 + 1 = 45_{10}$$

$$\text{Logo: } 101101_2 = 45_{10}$$

$$27_8 = ?_{10}$$

$$b = 8, n = 2$$

$$\text{Portanto: } 2 \times 8^1 + 7 \times 8^0 = 23_{10}$$

$$\text{Logo: } 27_8 = 23_{10}$$

$$2A5_{16} = ?_{10}$$

$$b = 16, n = 3$$

$$\text{Portanto: } 2 \times 16^2 + 10 \times 16^1 + 5 \times 16^0 = 512 + 160 + 5 = 677_{10}$$

$$\text{Logo: } 2A5_{16} = 677_{10}$$

Aula 2 – Sistemas de Numeração

2.1.2.5 Base decimal para base B (qualquer)

Consiste no processo inverso, ou seja, efetuamos divisões sucessivas do número decimal pela base desejada, até que o quociente seja menor que a referida base. Utilizamos os restos e o último quociente (a começar dele) para formação do número desejado, conforme abaixo

$$\begin{array}{r} 25 \quad | \quad 2 \\ - 24 \quad | \quad 12 \quad 2 \\ \hline \text{resto } 1 \quad - 12 \quad | \quad 6 \quad 2 \\ \hline \text{resto } 0 \quad - 6 \quad | \quad 3 \quad 2 \\ \hline \text{resto } 0 \quad - 2 \quad | \quad 1 \\ \hline \text{resto } 1 \end{array}$$

1 1 0 0 1₂

$$\begin{array}{r} 3964 \quad | \quad 8 \\ - 3960 \quad | \quad 495 \quad 8 \\ \hline \text{resto } 4 \quad - 488 \quad | \quad 61 \quad 8 \\ \hline \text{resto } 7 \quad - 56 \quad | \quad 7 \\ \hline \text{resto } 5 \end{array}$$

7574₈

$$\begin{array}{r} 2754 \quad | \quad 16 \\ - 2752 \quad | \quad 172 \quad 16 \\ \hline \text{resto } 2 \quad - 160 \quad | \quad 10 \quad 16 \\ \hline \text{resto } 12 \end{array}$$

AC2₁₆

2.2 Aritmética não-decimal

Procedimentos para realização das quatro operações aritméticas:

- Adição
- Subtração
- Multiplicação
- divisão

2.2.1 Aritmética Binária

2.2.1.1 Adição Binária

A operação de soma de dois números em base 2 é efetuada de modo semelhante à soma decimal, levando-se em conta, apenas, que só há dois algarismos disponíveis (0 e 1). Assim, podemos criar uma tabela com todas as possibilidades:

0	+	0	=	0
0	+	1	=	1
1	+	0	=	1
1	+	1	=	0

“VAI 1”

2.2.1 Aritmética Binária

2.2.1.1 Soma Binária

0	+	0	=	0
0	+	1	=	1
1	+	0	=	1
1	+	1	=	0

“VAI 1”

Efetua a soma 45_{10} e 47_{10} :

Decimal	Binário
45	1 1 1 1 1
+47	1 0 1 1 0 1
92	+1 0 1 1 1 1
	1 0 1 1 1 0 0

Efetue a soma $37_{10} + 87_{10}$

Decimal

37

87

124

Binário

1 1 1

0 1 0 0 1 0 1

1 0 1 0 1 1 1

1 1 1 1 1 0 0

2.2.1 Aritmética Binária

2.2.1.2 Subtração Binária

A subtração em base 2, na forma convencional, usada também no sistema decimal (minuendo – subtraendo = diferença), é relativamente mais complicada por dispomos apenas dos algarismos 0 e 1 e, dessa forma, 0 menos 1 necessita de “empréstimo” de um valor igual à base (no caso é 2), obtido do primeiro algarismo diferente de zero, existente à esquerda. Se estivéssemos operando na base decimal, o “empréstimo” seria de valor igual a 10.

a) Efetuar a subtração $101101 - 100111$:

$$\begin{array}{r} 2 \\ 002 \\ 10\cancel{1}\cancel{1}01 \\ - 100111 \\ \hline 000110 \end{array}$$

A partir da direita para a esquerda, vamos executar a operação algarismo por algarismo (6 algarismo)

1) $1 - 1 = 0$

2) $0 - 1$ não é possível. Então, retira-se 1 da ordem à esquerda (3ª ordem a partir da direita), que fica com $1 - 1 = 0$, e passa-se para a ordem à direita, o valor equivalente, que é 2, visto que 1 unidade de ordem à esquerda vale uma base de unidades (no caso: base= 2) da ordem à direita. $2 - 1 = 1$

3) Agora tem-se $0 - 1$ e, portanto, repete-se o procedimento do item anterior. $2 - 1 = 1$

4) $0 - 0 = 0$

5) $0 - 0 = 0$

6) $1 - 1 = 0$

Resultado: $(000110)_2$ ou simplesmente $(110)_2$.

2.2.1 Aritmética Binária

2.2.1.3 Multiplicação Binária

0	x	0	=	0
0	x	1	=	0
1	x	0	=	0
1	x	1	=	1

a) Efetuar a multiplicação 110×101 :

$$\begin{array}{r} 110 \\ 101 \\ \hline 110 \\ 000 \\ 110 \\ \hline 11110 \end{array}$$

Resultado: $(11110)_2$

2.2.1 Aritmética Binária

2.2.1.4 Divisão Binária

a) Efetuar a divisão $(101010)_2$ por $(101)_2$

$$\begin{array}{r} 101010 \overline{) 110} \\ - 110 \\ \hline 1001 \\ - 110 \\ \hline 0110 \\ - 110 \\ \hline 000 \end{array}$$

2.2.1 Aritmética Binária

2.2.1.4 Divisão Binária

a) Efetuar a divisão $(37)_{10}$ por $(4)_{10}$

Decimal

$$\begin{array}{r|l} 37 & 4 \\ - 36 & \\ \hline 1 & 9 \end{array}$$

Binário

$$\begin{array}{r|l} 100101 & 100 \\ - 100 & \\ \hline 000101 & 1001 \\ - 100 & \\ \hline 001 & \end{array}$$

2.2.1 Aritmética Octal

2.2.1.5 Soma Octal

a) Efetuar a soma $(3657)_8$ por $(1741)_8$

$$\begin{array}{r} 111 \\ 3657 \\ +1741 \\ \hline +5620 \end{array}$$

Da direita para esquerda, temos para cada um dos 4 algarismos

- 1) Como não há algarismo 8 na base 8, emprega-se o conceito posicional, isto é, 8 unidades de um ordem valem 1 unidade da ordem imediatamente à esquerda. Então: fica $0 = 8-8$ e vai 1 para esquerda.
- 2) $1 + 5 + 4 = 10$. Então utilizando o mesmo conceito $10-8 = 2$ e vai 1.
- 3) $1 + 6 + 7 = 14$. Então $14-8 = 6$ e vai 1.
- 4) $1 + 3 + 1 = 5$. Não há vai 1 porque não se excedeu 7.

2.2.1 Aritmética Octal

2.2.1.6 Soma Octal

a) Efetuar a soma $(443)_8$ por $(653)_8$

$$\begin{array}{r} 11 \\ 443 \\ + 653 \\ \hline 1316 \end{array}$$

$(1316)_8$

2.2.1 Aritmética Octal

2.2.1.7 Subtração Octal

a) Efetuar a subtração $(7312)_8 - (3465)_8$

$$\begin{array}{r} 88 \\ 6208 \\ 7312 \\ - 3465 \\ \hline 3625 \end{array}$$

$(3625)_8$

Da direita para a esquerda, temos para cada um dos 4 algarismos:

1) $2 - 5$ não é possível. Então, retira-se 1 unidade da ordem à esquerda, a qual vale uma base de unidades (no caso base = 8) da direita, somando-se ao valor 2.

$$8 + 2 = 10 - 5 = 5$$

2) $1 - 1 = 0 - 6$ não é possível. Então, retira-se 1 unidade da esquerda (que fica com $3 - 1 = 2$ unidades), passando 8 para a direita, o que fica $8 + 0 = 8$.

$$8 - 6 = 2$$

3) $3 - 1 = 2 - 4$ não é possível. Então, retira-se 1 da esquerda ($7 - 1 = 6$), passando 8 unidades para direita

$$8 + 2 = 10 - 4 = 6$$

4) $7 - 1 = 6 - 3 = 3$

2.2.1 Aritmética Hexadecimal

2.2.1.7 Soma Hexadecimal

a) Efetuar a soma $(3A943B)_{16} + (23B7D5)_{16}$

$$\begin{array}{r} 1 1 1 \\ 3 A 9 4 3 B \\ + 2 3 B 7 D 5 \\ \hline 5 E 4 C 1 0 \end{array}$$

2.2.1 Aritmética Hexadecimal

2.2.1.7 Soma Hexadecimal

a) Efetuar a soma $(3A943B)_{16} + (23B7D5)_{16}$

$$\begin{array}{r} 1 1 1 \\ 3 A 9 4 3 B \\ + 2 3 B 7 D 5 \\ \hline 5 E 4 C 1 0 \end{array}$$

2.2.1 Aritmética Hexadecimal

2.2.1.7 Subtração Hexadecimal

a) Efetuar a soma $(4C7BE8)_{16} - (1E927A)_{16}$

$$\begin{array}{r} 4\ C\ 7\ B\ E\ 8 \\ -\ 1\ E\ 9\ 2\ 7\ A \\ \hline 2\ D\ E\ 9\ 6\ E \end{array}$$

1. $8 - A$ não é possível. Retira-se, então, 1 unidade da ordem à esquerda ($E - 1 = D$), passando 16 unidades (valor igual ao da base) para a direita, as quais são somadas ao valor existente, 8
 $16 + 8 = 24 - A = 24 - 10 = 14_{10}$ equivale ao algarismo E_{16}
2. $D - 7 = 13 - 7 = 6$
3. $B - 2 = 11 - 2 = 9$
4. $7 - 9$ não é possível. Retira-se 1 unidade da ordem à esquerda ($C - 1 = B$), passando 16 unidades para a direita, as quais são somadas ao valor existente, 7.
 $16 + 7 = 23 - 9 = 14_{10}$, equivalente ao algarismo E_{16}
5. $C - E$ não é possível. Retira-se 1 unidade da ordem à esquerda ($4 - 1 = 3$), passando 16 unidades para a direita, as quais são somadas ao valor existente, $B_{16} = 11_{10}$. $16_{10} + 11_{10} = 27 - 14 = 13_{10}$ equivalente ao algarismo D_{16}
6. $3 - 1 = 2$

2.2.1 Aritmética Binária

2.2.1 Sistema de numeração com virgula

16	8	4	2	1	0,5	0,25	0,125	0,0625
2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}

Transformação de binário para decimal **$(101,1001)_2$**

parte inteira

$$101 = (2^0 \times 1) + (2^1 \times 0) + (2^2 \times 1) = 1 + 0 + 4 = 5$$

parte decimal

$$1001 = (2^{-1} \times 1) + (2^{-2} \times 0) + (2^{-3} \times 0) + (2^{-4} \times 1) = 0,5 + 0 + 0 + 0,0625 = 0,5625$$

$(5,5625)_{10}$

2.1.3 Conversão de Números Binários Fracionários em Decimais

- O método de conversão é obtido observando-se a regra básica de formação de um número fracionário no sistema decimal. Para exemplificar, tem-se o número $(10,5)_{10}$

$$(10,5)_{10} = 1 \times 10^1 + 0 \times 10^0 + 5 \times 10^{-1}$$

- Desta forma, para converter o número binário fracionário 101,101 para o sistema decimal, adota-se o mesmo procedimento.

$$(101,101) = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$$

$$(101,101) = 1 \times 4 + 0 \times 2 + 1 \times 1 + 1 \times \frac{1}{2} + 0 \times \frac{1}{4} + 1 \times \frac{1}{8}$$

$$(101,101) = 4 + 0 + 1 + \frac{1}{2} + 0 + \frac{1}{8}$$

$$(101,101) = 4 + 1 + 0,5 + 0,125 = (5,625)_{10}$$

Aula 2 – Sistemas de Numeração

2.1.3.1 Conversão de Números Decimais Fracionários em Binários

- O processo consiste em separar o número decimal na parte inteira e na fracionária.
- O método das divisões sucessivas é aplicado a parte inteira, conforme estudado anteriormente.
- Para a parte fracionária aplica-se o método das multiplicações sucessivas até que se atinja zero.
- Para exemplificar, será convertido o número decimal 8,375 em binário

$$8 + 0,375$$

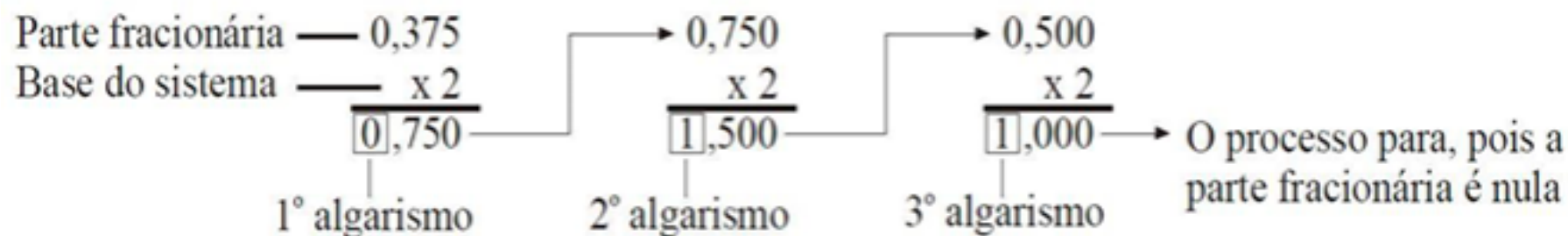
- Parte Inteira:

$$\begin{array}{r} 8 \overline{) 2} \\ \underline{0} \\ 4 \overline{) 2} \\ \underline{0} \\ 2 \overline{) 2} \\ \underline{0} \\ 0 \end{array} \begin{array}{l} \text{LSB} \text{---} \textcircled{0} \\ \phantom{\text{LSB}} \textcircled{0} \\ \phantom{\text{LSB}} \textcircled{0} \end{array} \text{--- MSB} \quad 8_{10} = 1000_2$$

Aula 2 – Sistemas de Numeração

2.1.3.1 Conversão de Números Decimais Fracionários em Binários

- Parte Fracionária:



Pode-se observar que é utilizado somente a parte fracionária dos números em todas as multiplicações.

Os algarismos inteiros, resultantes das multiplicações, irão compor o número binário.

Estes números são tomados na ordem da multiplicação. Assim:

$$0,375_{10} = 0,011_2$$

Para completar a conversão basta efetuar a composição da parte inteira com a fracionária:

$$8,375_{10} = 1000,011_2$$

Aula 2 – Sistemas de Numeração

2.1.3.1 Conversão de Números Decimais Fracionários em Binários

- Observação Importante: existem casos em que o método das multiplicações sucessivas encontra novamente os números já multiplicados e o processo entra em um “loop” infinito.
- Isto equivale a uma dízima periódica. Como exemplo, tem-se:

$$(0,8)_{10} = (0, 1100\ 1100\ 1100)_2$$

Aula 2 – Sistemas de Numeração

2.1.3.1 Conversão de Números Decimais Fracionários em Octal Fracionários

2.2.1 Aritmética Binária

2.2.1 Notação dos Números Binários Positivos e Negativos

A representação de números binários positivos e negativos pode ser feita utilizando-se os sinais “+” ou “-” respectivamente. Na prática, porém, em hardware dos sistemas digitais que processam operações aritméticas, microcomputadores por exemplo, estes sinais não podem ser utilizados, pois tudo deve ser codificado em **0** e **1**. Uma forma de representar em alguns casos utilizada, é a de acrescentar ao número um ***bit de sinal*** colocando à esquerda, na posição de algarismo mais significativo. Se o número for positivo, o bit de sinal será **0**, se o número for negativo este será **1**. este processo de representação é denominado **SINAL-MÓDULO**.

2.2.1 Aritmética Binária

2.2.1 Notação dos Números Binários Positivos e Negativos

Para exemplificar o exposto, vamos representar os números decimais $+(35)_{10}$ e -73_{10} em binário utilizando a notação sinal-módulo:

$$(35)_{10} = (10011)_2$$

$$+ (10011)_2 = (0100011)_2$$

bit de sinal (0 → indica número positivo

$$(73)_{10} = (11001001)_2$$

bit de sinal (1 → indica número negativo

2.2.1 Aritmética Binária

2.2.1 Notação dos Números Binários Positivos e Negativos

Uma outra forma para representar número binários negativos bastante utilizada nos sistemas já citados é a notação do **complemento de 2**. mas para obtê-la, devemos primeiramente converter o número na notação do **complemento de 1**.

A obtenção do complemento de 1 de um número binário se dá pela troca de cada bit do número pelo seu inverso ou complemento. Para demonstrar esse procedimento, vamos obter o complemento de 1 do número $(10011011)_2$. Assim sendo, temos:

Número binário: 1 0 0 1 0 1 1

Complemento de 1: 0 1 1 0 1 0 0

+ 1

Complemento de 2: 0 1 1 0 1 0 1

Arquitetura e organização de computadores
Uma visão geral

Conceitos da Lógica Digital

Aula 3

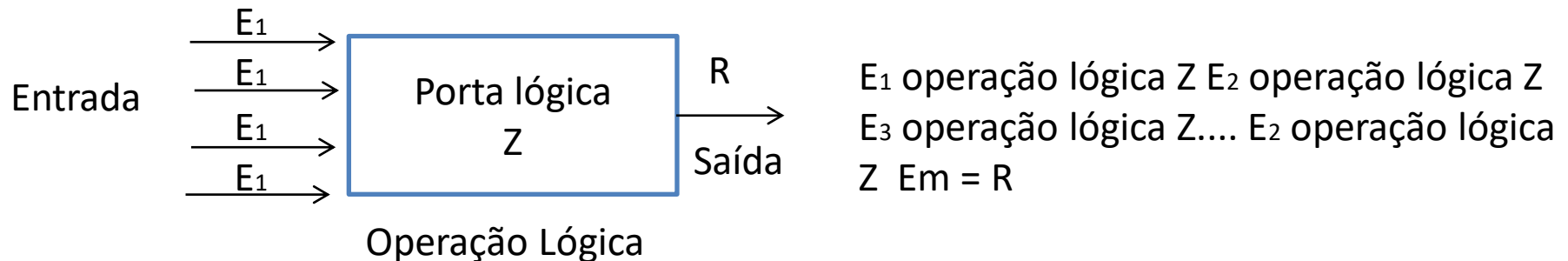
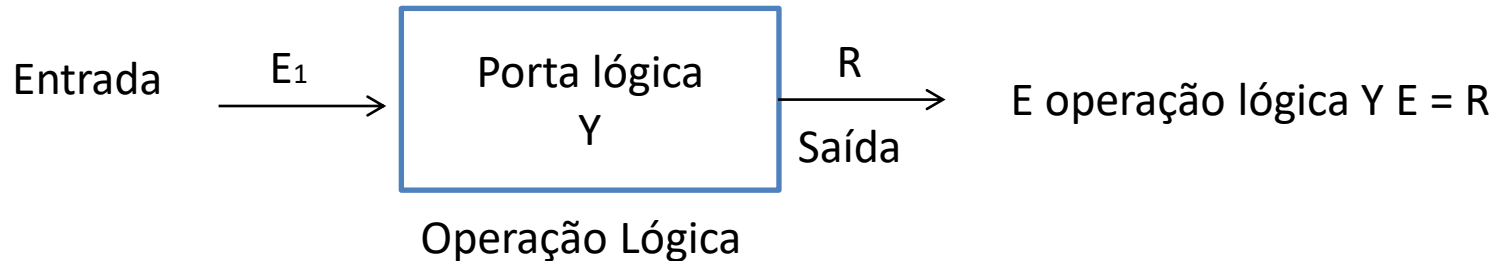
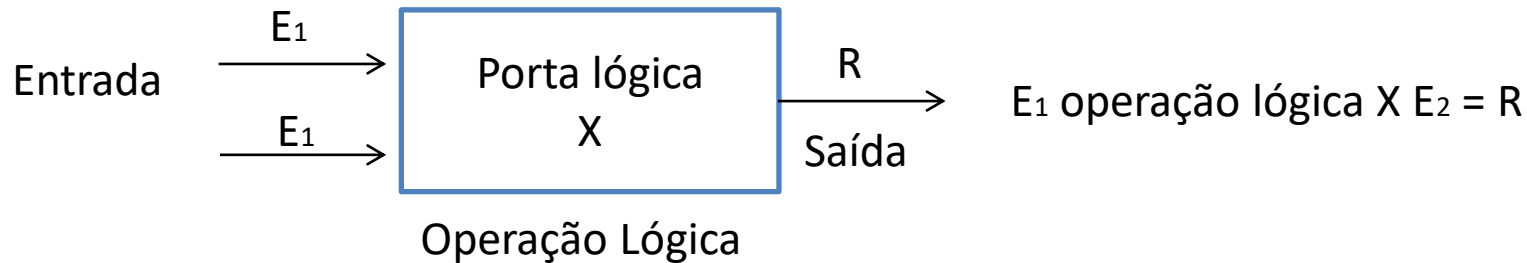
3. 1 Operações lógicas

- As operações lógicas são estudadas pela álgebra de Boole (George Boole)
- A álgebra de Boole trabalha com apenas duas grandezas: **falso** ou **verdadeiro**.
- As duas grandezas são representadas por **0** (falso) e **1** (verdadeiro).
- Nos circuitos lógicos do computador, os sinais binários são representados por níveis de tensão.

3. 2 Portas lógicas

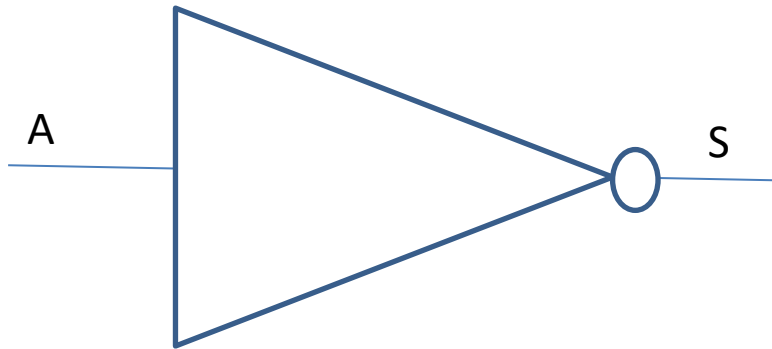
- As portas lógicas são os elementos mais básicos e elementares de um sistema de computação.
- Elas são responsáveis por realizar as operações lógicas sobre os bits.
- Os valores de entrada e saída são números binários.
- Cada porta lógica realiza uma tarefa trivial.

3.3 Portas e operações lógicas



3.3.1 Portas e operações lógicas NOT (inversor)

- **NOT:** inverte a entrada.

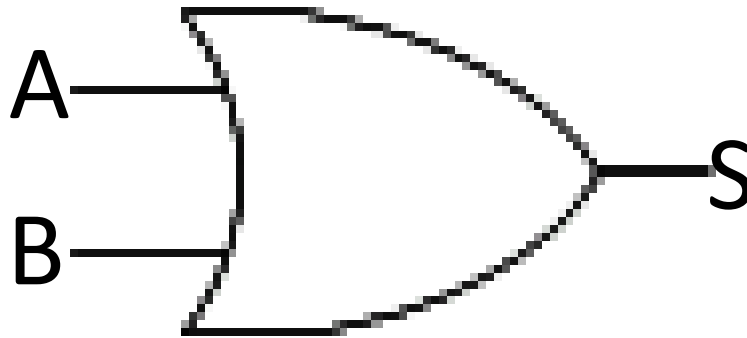


Entrada	Saída
A	$S = \bar{A}$
0	1
1	0

Expressão: $S = \bar{A}$

3. 3. 1 Operação lógica ou Porta OR (OU)

- **OR**: retorna 1 se uma das entradas é 1.

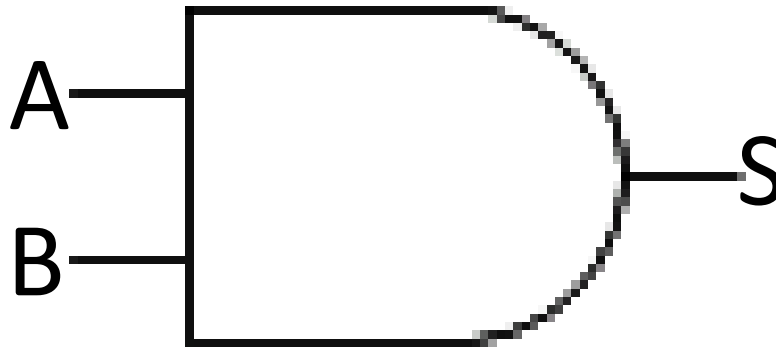


Entrada		Saída
A	B	$S = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

Expressão: $S = A + B$

3. 3. 2 Operação lógica ou Porta AND (E)

- **AND**: retorna 1 se ambas as entradas são 1.

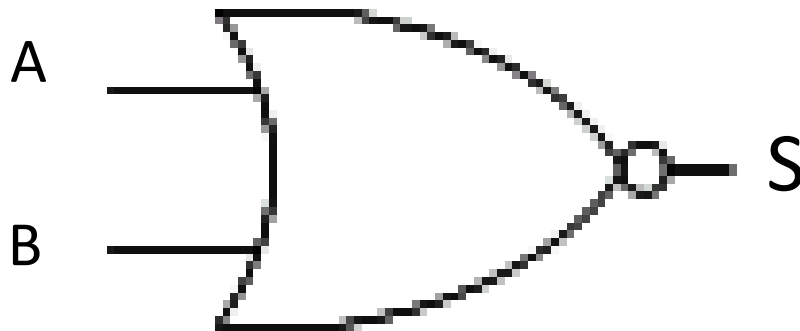


Entrada		Saída
A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

Expressão: $S = AB$

Aula 3 – Portas lógicas

- **NOR:** é uma porta OR e uma porta NOT combinadas. O resultado é exatamente o inverso da porta OR.

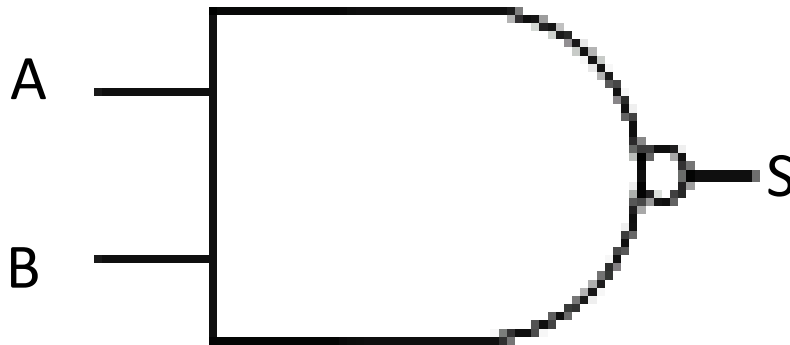


Expressão: $x = \overline{A + B}$

Entrada		Saída
A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

Portas lógicas

- **NAND:** é uma porta AND e uma porta NOT combinadas. O resultado é exatamente o inverso da porta AND.

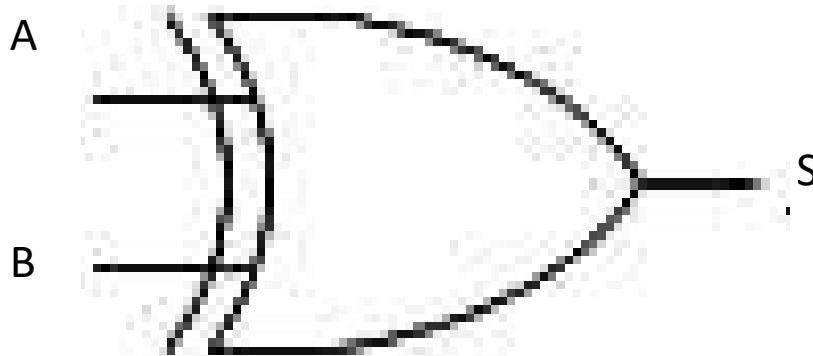


ENTRADA		SAÍDA
A	B	S
0	0	1
1	0	1
0	1	1
1	1	0

Expressão: $x = \overline{AB}$

Portas lógicas

- **XOR**: retorna 1 somente se uma das entradas é 1.

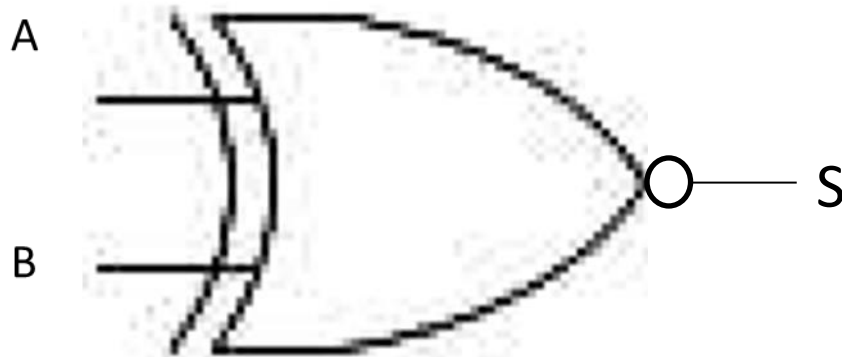


ENTRADA		SAÍDA
A	B	S
0	0	0
1	0	1
0	1	1
1	1	0

Expressão: $x = A \oplus B$

Portas lógicas

- **NXOR**: é uma porta XOR e uma porta NOT combinadas. O resultado é exatamente o inverso da porta XOR.



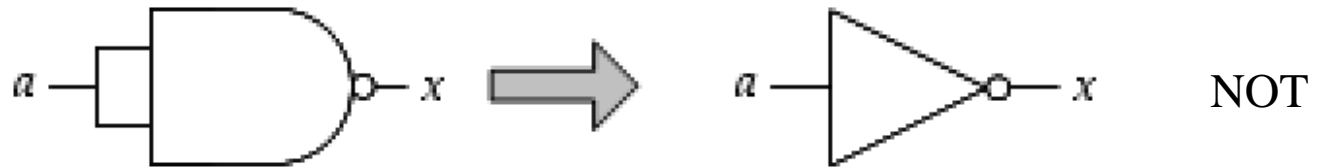
ENTRADA		SAÍDA
A	B	S
0	0	1
1	0	0
0	1	0
1	1	1

Expressão: $S = A \otimes B$

Portas lógicas

- Combinações de portas NAND podem ser usadas para simular todas as outras.
- Por este motivo, a porta NAND é considerada uma **porta universal**.
- Isso significa que qualquer circuito pode ser expresso pela combinação de portas NAND.

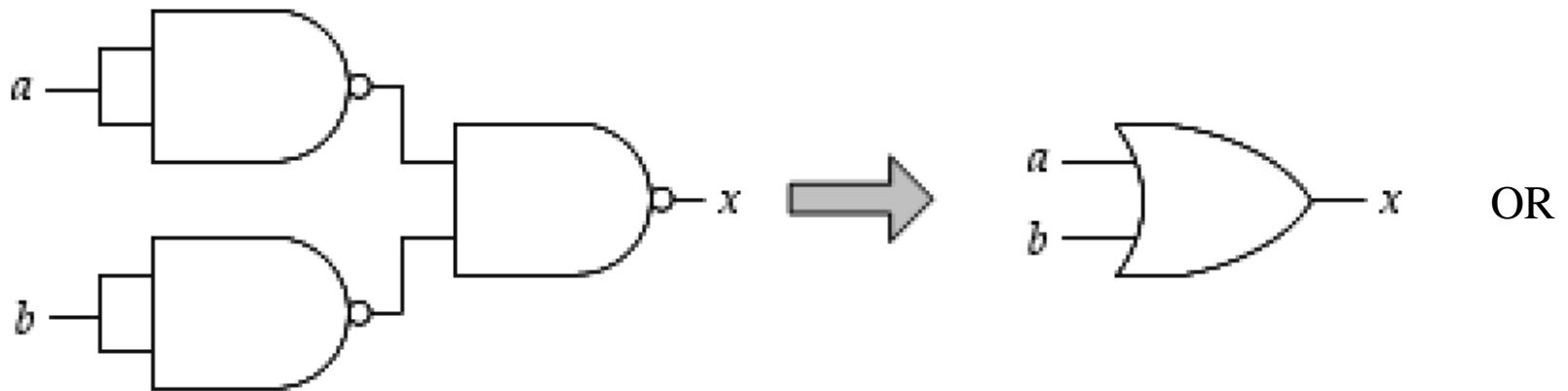
Portas lógicas



NOT



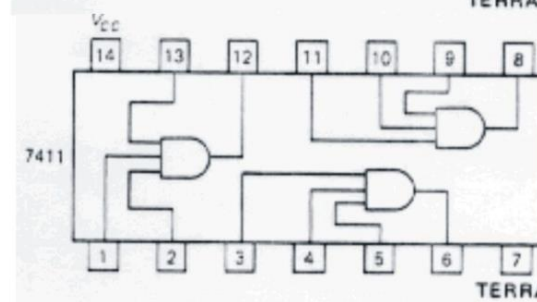
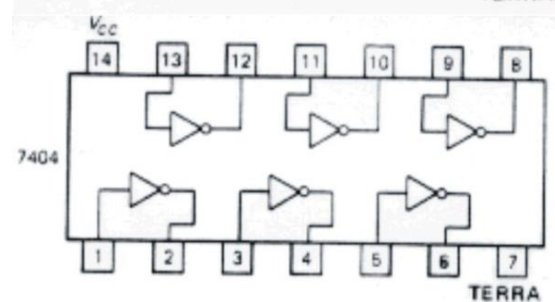
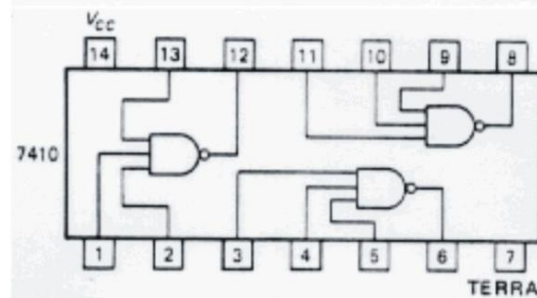
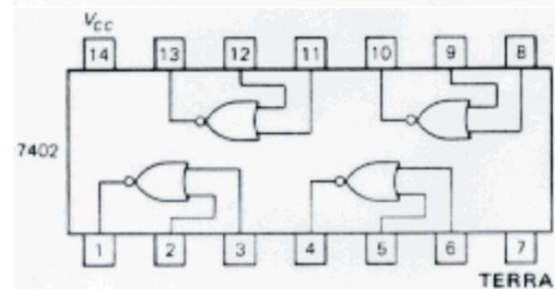
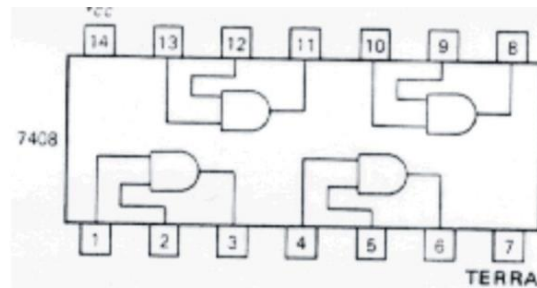
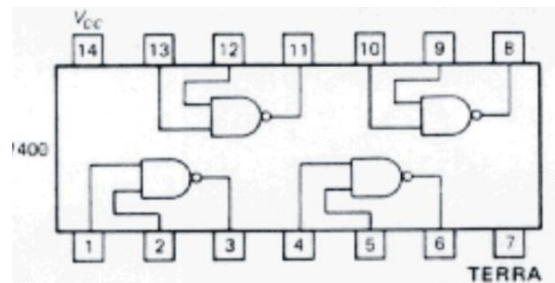
AND



OR

Circuitos

- As portas lógicas são encontradas no mercado encapsuladas em chips de silício.



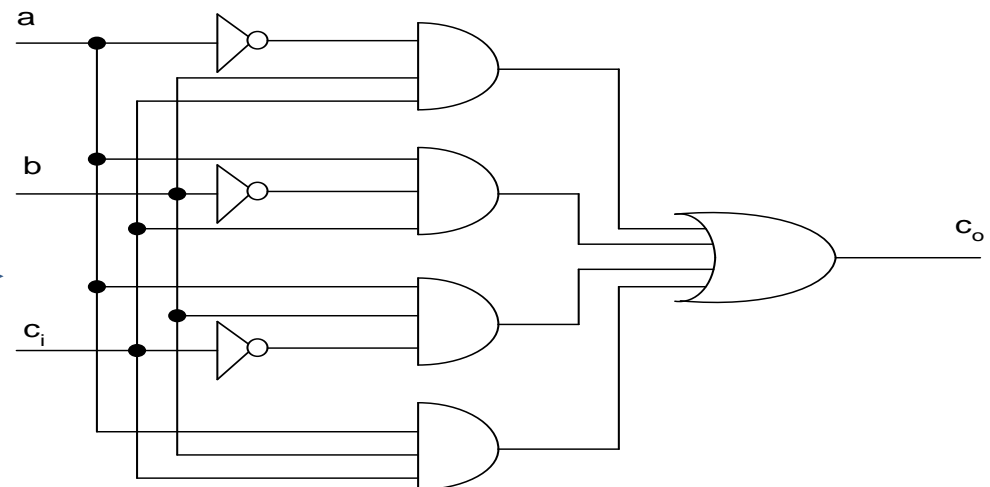
Circuitos

- É um conjunto de portas lógicas interligadas para resolver um problema maior.
- Para facilitar o desenvolvimento, em primeiro lugar, deve-se montar uma expressão booleana e, em seguida, partir para a implementação do circuito propriamente dito.

Circuitos

Como converter uma tabela verdade em um circuito lógico?

a	b	c _i	c _o
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



Soma de MinTermos

- Para cada saída, fazer uma soma de produtos, ou seja, a função de chaveamento é uma soma (OR) de produtos (AND) de variáveis e variáveis complementadas.
- Deve-se considerar apenas as saídas “1” e ignorar as saídas “0”.
- Após encontrar a função de chaveamento, desenhar o circuito.

a	b	s
0	0	0
0	1	1
1	0	0
1	1	1

$$s = a' b + a b$$

Soma de MinTermos

- Vocês fazem:
 - Dado a seguinte tabela verdade, encontrar a **função de chaveamento** e em seguida construir o **circuito lógico**.

x_2	x_1	x_0	z
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$$z = x_2' x_1' x_0 + x_2' x_1 x_0' + x_2 x_1' x_0' + x_2 x_1 x_0$$

Circuito meio-somador

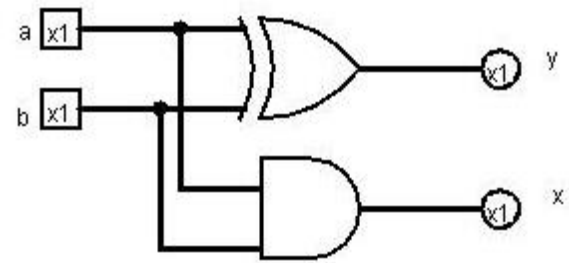
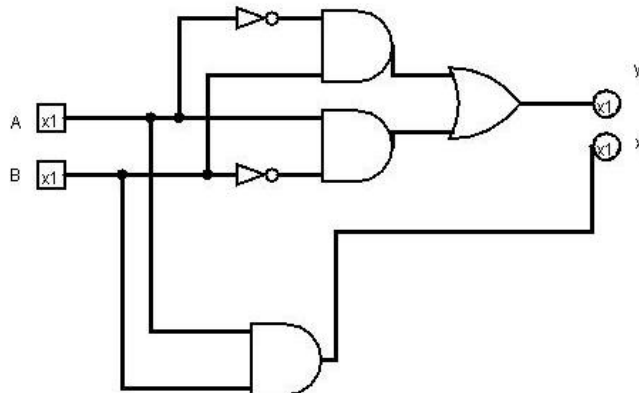
- Tabela Verdade:

A	B	X	Y
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$$X = A \cdot B$$

$$Y = A' \cdot B + A \cdot B'$$

- Circuito:



Versão simplificada