**Revisão Histórica da Evolução dos Computadores**

**História do Computador**



A palavra computador vem do verbo computar que significa calcular, a criação de computadores começa na idade antiga, naquela época a relação de contar já encafifava os homens, por isso, uma das primeiras máquinas de computar foi o ábaco, instrumento mecânico de origem chinesa criado no século V a.C.

**Primeira Geração (1951-1959)**



Nesta geração os computadores funcionavam por meio de circuitos e válvulas eletrônicas. Seu uso era restrito, além de serem imensos e consumirem muita energia.



Um exemplo é o Eletronic Numerical Integrator and Computer, conhecido como ENIAC, que possuía 19.000 válvulas e consumia cerca de 200 Kwatts.

**Segunda Geração (1959-1965)**

Os computadores da segunda geração, ainda eram grandes e, funcionavam por meio de transistores, os quais substituíram as válvulas que eram maiores e mais lentas. Nesse período já começam o seu uso no comércio.



**Terceira Geração (1965-1975)**

Nesta geração os computadores utilizavam circuitos integrados. Esses substituíram os transistores e já apresentavam uma dimensão menor e maior capacidade de processamento. Foi nesse período que os chips foram criados e a utilização de computadores pessoais se iniciou.

**Quarta Geração (1975-até os dias atuais)**

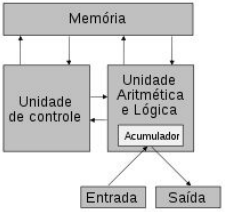


Com o desenvolvimento da tecnologia da informação, os computadores diminuem de tamanho, aumentam a velocidade e capacidade de processamento de dados.

O que marca esta geração é a utilização de microprocessadores com gasto cada vez menor de energia.

**John von Neumann**

A Arquitetura de von Neumann é uma arquitetura de computador que se caracteriza pela possibilidade de uma máquina digital armazenar seus programas no mesmo espaço de memória que os dados, podendo assim manipular tais programas.



**Conceitos: Arquitetura e Organização**

**Arquitetura de Computadores:**

Refere-se aos aspectos funcionais do Sistema Computacional que são “visíveis” ao programador.

Exemplo:

* Conjunto de Instruções (Tipos de Instruções)
* Tamanho dos Dados (Número de Bits)

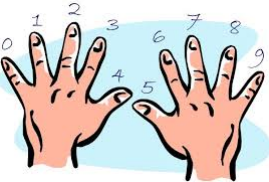
**Organização de Computadores:**

Refere-se aos aspectos estruturais do Sistema Computacional que não são “visíveis” ao programador.

Exemplo:

* Sinais de Controle, Frequência de Clock
* Multiplicação implementada por adições ou hardware específico

**Bases de Numeração**

**Notação Posicional - Base Decimal**

A cultura ocidental adotou um sistema de numeração que possui dez diferentes algarismos — 0, 1, 2 , 3, 4, 5, 6, 7, 8, 9 e, por essa razão, foi chamado de sistema decimal.

Vamos exemplificar o conceito de sistema posicional. Seja o número 1303, representado na base 10, escrito da seguinte forma: 130310

Neste exemplo, o número é composto de quatro algarismos: 1, 3, 0 e 3 e cada algarismo possui um valor correspondente à sua posição no número.

Assim, o primeiro 3 (algarismo mais à direita) representa 3 unidades. Neste caso, o valor absoluto do algarismo (que é 3) é igual ao seu valor relativo (que também é 3), por se tratar da 1a posição (posição mais à direita, que é a ordem das unidades).

Considerando-se o aspecto três vezes a potência 0 da base 10 ou 3 X 100 = 3

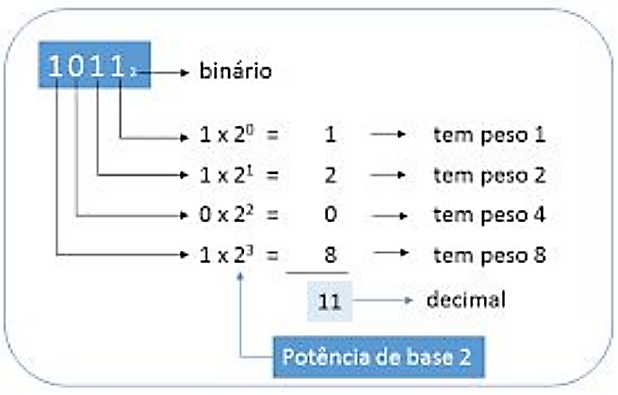
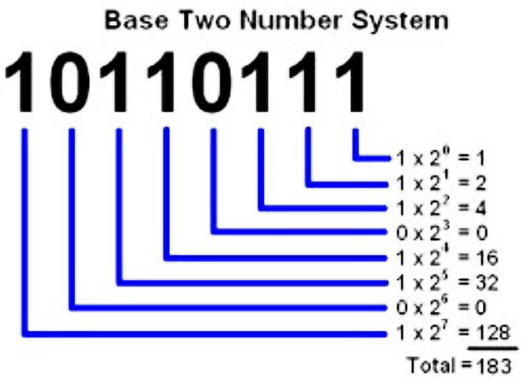
Enquanto o segundo 3 vale três vezes a potência 2 da base 10 ou 3 X 102 = 300

O valor total do número seria então:

1000 + 300 + 0 + 3 = 130310

1 X 103 + 3 X 102 + O X 101 + 3 X 100 = 130310

**Outras Bases de Numeração**

Entre as bases diferentes da 10, consideremos apenas as bases 2 e potências de 2, visto que todo computador digital representa internamente as informações em algarismos binários, ou seja, trabalha em base 2.

No entanto, nas bases diferentes de 10, o valor relativo do algarismo (valor dependente de sua posição no número) é normalmente calculado usando-se valores resultantes de operações aritméticas em sua base.

Exemplo:

Seja o número na base 2: (1011)2)

Se aplicássemos a expressão anterior teríamos:

1X23 + 0X22 + 1X21 + 1X20 = 8 + 0 + 2 + 1 = (11)10

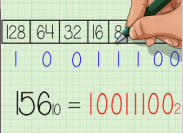
O número máximo de algarismos diferentes de uma base é igual ao valor da base.

Exemplo:

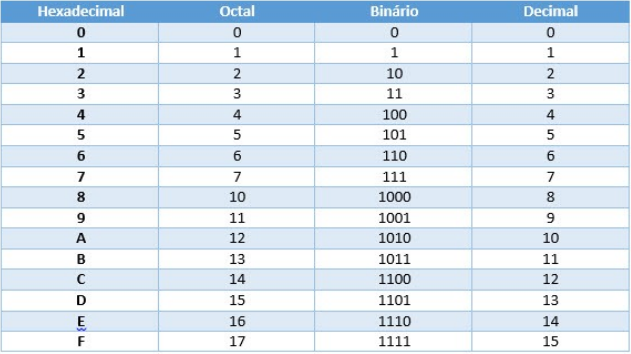
na base 10 temos dez dígitos: de 0 a 9;

na base 2 temos apenas dois dígitos: 0 e 1;

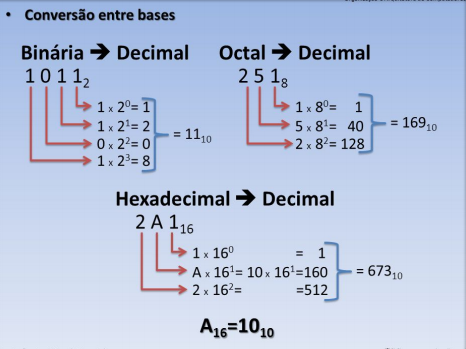
na base 5 temos cinco dígitos: de 0 a 4.

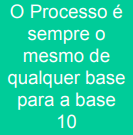
Exemplo:

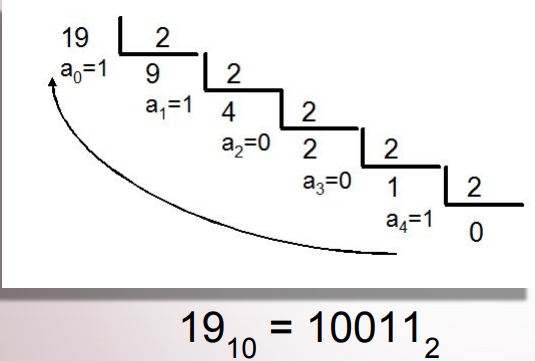
**Tabela de Conversão de Bases**



**Conversão de uma Base Binaria para a Base 10**





**Conversão da Base 10 para a Base 2**

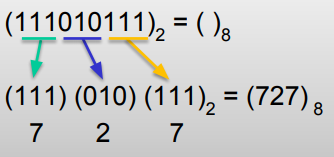
**Conversão Entre as Bases 2, 8, 16 e Aritmética Binária**

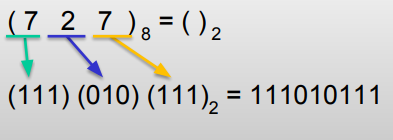
**Conversão entre Bases Potência de 2**

**Entre as Bases 2 e 8**

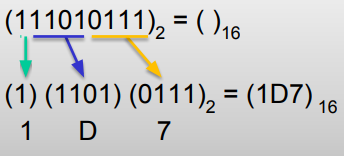
Como 8 = 23, um número binário (base 2) pode ser facilmente convertido para o seu valor equivalente na base 8 (octal). Se o número binário for inteiro, basta dividi-lo, da direita para a esquerda, em grupos de 3 bits.

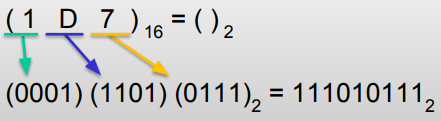
Então, para cada grupo, acha-se o algarismo octal equivalente, conforme o exemplo.





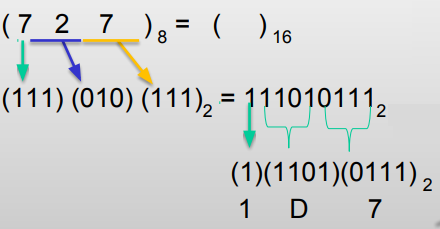
**Entre Bases 2 e 16**

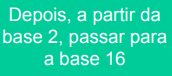




**Entre Bases 8 e 16**

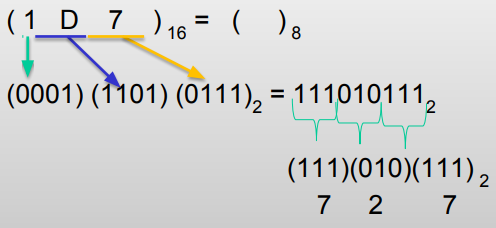


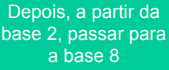














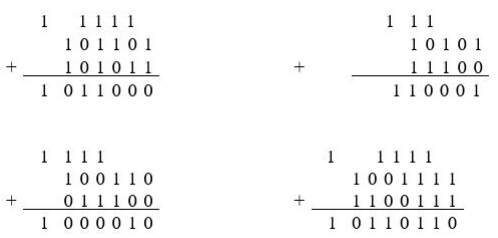
**Soma e Subtração na Base 2**

**Soma 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:

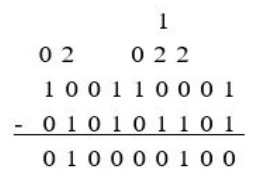
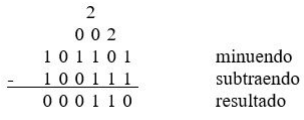
0 + 0 = 0 0 + 1 = 1

1 + 0 = 1 1 + 1 = 0, com “vai 1”



**Subtração Binária**

Na subtração quando houver 0 menos 1, então haverá a necessidade de “empréstimo” de um valor igual 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.



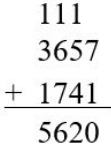
**Aritmética nas Bases 8 e 16**

**Aritmética Octal (Em Base 8)**

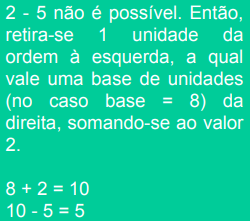
Soma: Consiste em processo semelhante ao da aritmética binária, com exceção do fato de que, neste caso, tem-se oito algarismos disponíveis. Ocorrerá “vai 1” quando a soma de 2 algarismos for igual ou ultrapassar o valor da base, isto é, 8.

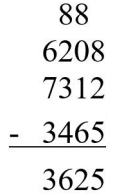


**Soma:**

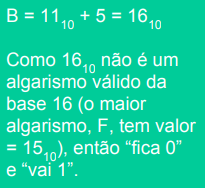


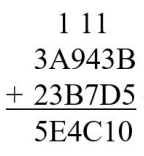
**Subtração:**





**Aritmética Hexadecimal (em Base 16)**





**Fundamentos de Lógica**

**Funções Lógicas**

Variáveis têm apenas 2 estados: 0 ou 1, F ou V

Também chamadas de Funções Booleanas devido a George Boole

**Funções:**

- AND

- OR

- NOT

- BUFFER

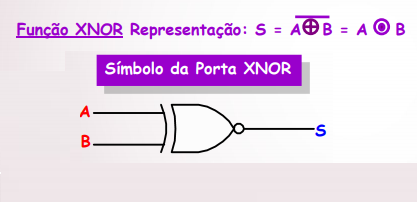
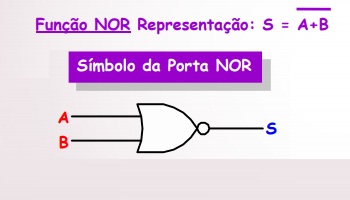
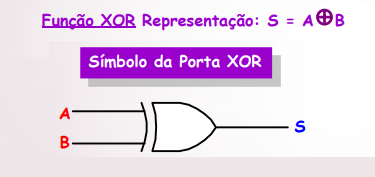
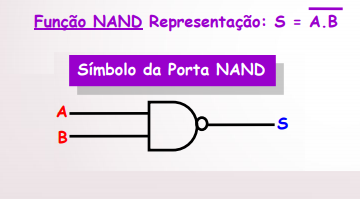
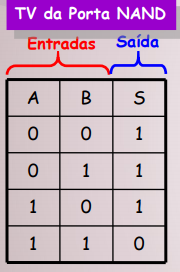
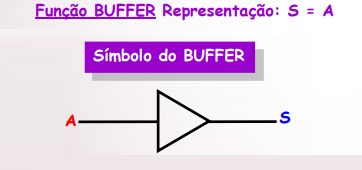
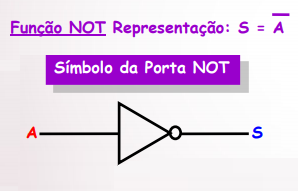
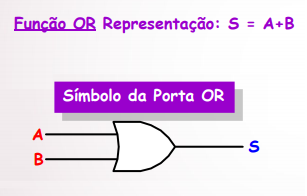
- NAND

- NOR

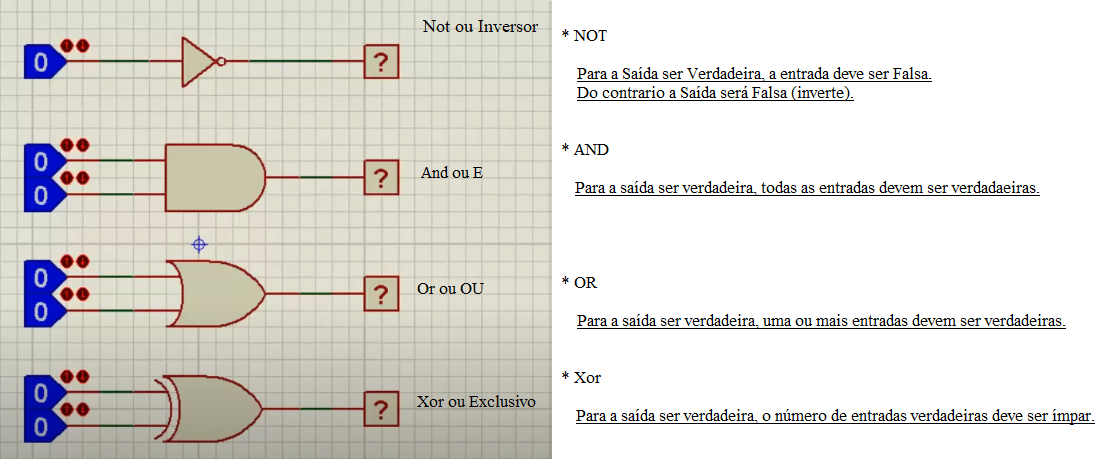
- XOR

- XNOR

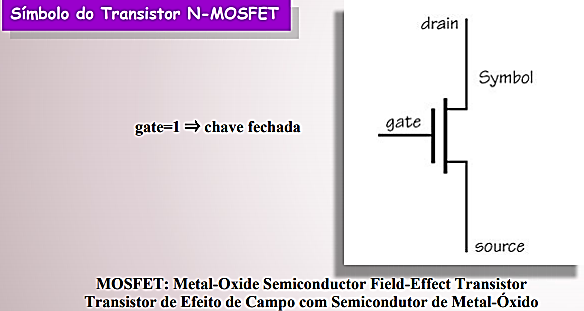
Tabela Verdade: Mapa onde se colocam todas as possíveis situações de entradas e saídas de um circuito lógico.

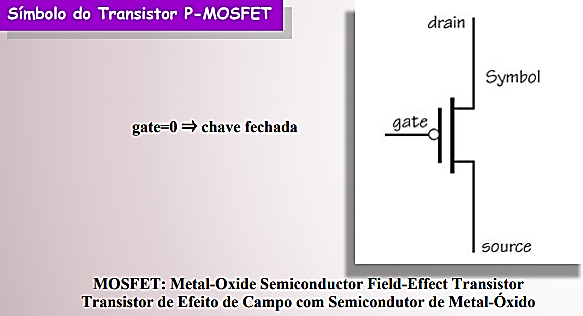


**Macete Da Tabela Verdade**

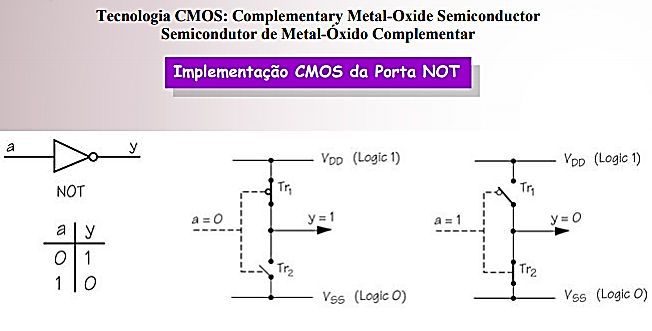


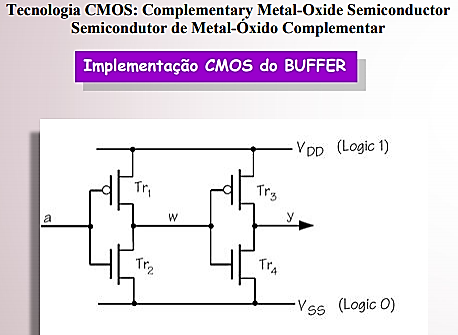
**Implementação das Portas Lógica**

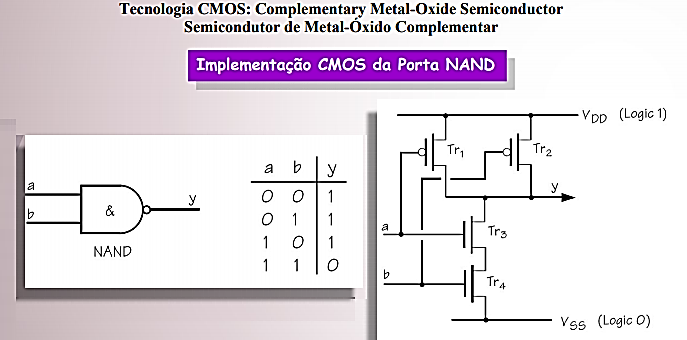








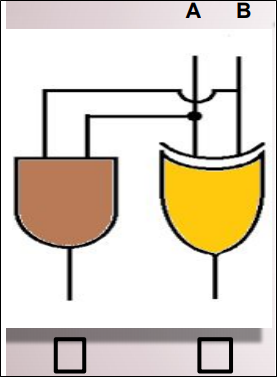


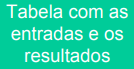


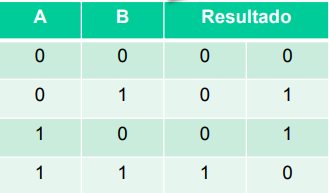


**Como o Computador Realiza Soma**

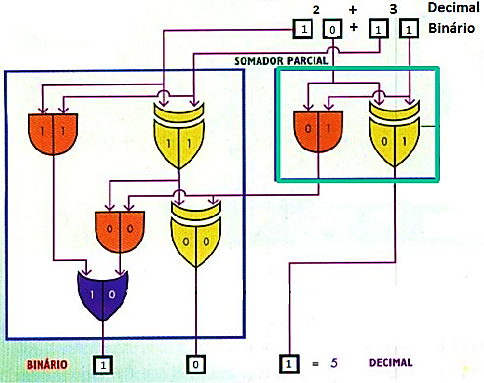
Com diferentes combinações de portas lógicas, um computador realiza os cálculos que são a base de todas as suas operações. Isto é conseguido com projetos chamados somadores parciais e somadores completos.

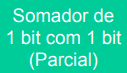
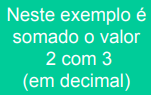












**Frequência**

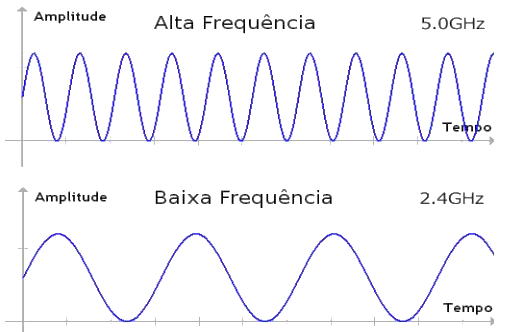
Elementos combinacionais

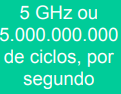
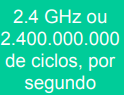
Exemplo: ULA

Elementos de estado Exemplo: Registradores e Memória

Metodologia de sincronização sincroniza o elemento de estado para a permissão de leitura e de escrita

Sinal de Clock usado para determinar quando se pode escrever em um elemento de estado. A leitura pode ser a qualquer momento





**Processo de tradução de um programa em Código executável**

Tradução de um programa para linguagem de máquina

O Programador gera comandos em linguagem de alto nível, para que a mesma possa ser executada, é necessário converte-la para a linguagem de máquina ().

Esta tarefa de conversão é feita por um programa especial de computador, ou seja, um programa que recebe as instruções em linguagem de alto nível e dá como saída outro programa constituído de instruções binárias.

Ao programa original, em linguagem de alto nível, dá-se o nome de Programa Fonte e ao resultado, em linguagem de máquina, de Programa Obcódigos bináriosjet

Existem três métodos básicos de abordagem na tradução de linguagem de alto nível para linguagem de

máquina:

Interpretador

Compilador

Tradutor

Interpretador:

O programa recebe uma instrução do programa fonte, converte-a em linguagem de máquina e então ordena ao computador que execute esta instrução.

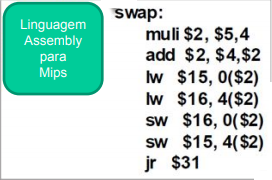
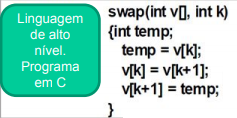
Depois repete o processo para a segunda instrução, e assim sucessivamente.

Compilador:

Um compilador é um programa de computador que, a partir de um código fonte escrito em uma determinada linguagem, cria um programa semanticamente equivalente, porém escrito em outra linguagem.

Tradutor:

Neste método, o tradutor é um programa que transforma/traduz um código assembly, por exemplo, em um código executável.



**Linguagem de Máquina**

Instrução é uma palavra da linguagem de máquina.

Instruction Set do MIPS ( usado pela NEC, Nintendo, Silicon, Graphics e Sony.

O MIPS trabalha com 3 operandos:



