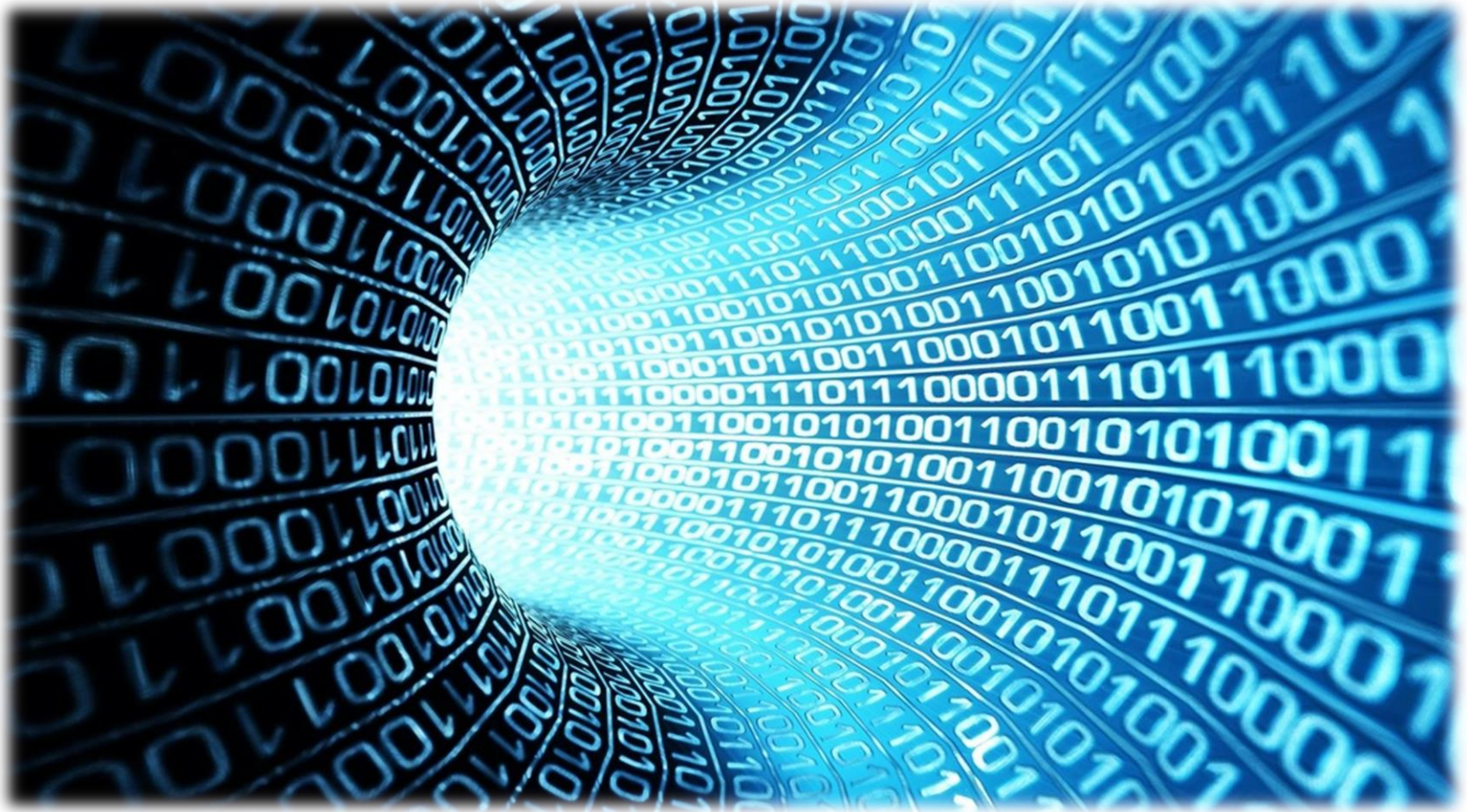


Organização e Arquitetura de Computadores



Fonte da imagem: <https://cutt.ly/D4jVvQY>

OBJETIVOS

- Compreender os conceitos do que seria a aritmética computacional:
 - Representação de números;
 - Conversões entre bases.
- Como trabalhar com a aritmética não decimal ou aritmética binária;
- Aritmética Octal, Hexadecimal, Divisão e Multiplicação.
- Representação Numérica:
 - Binário mais significativo e menos significativo;
 - Forma dos complementos de 1 e de 2 de um número binário;
 - Conhecer os números fracionários na arquitetura de computadores;
 - Ser capaz de realizar a representação numérica computacional;
- Ponto Fixo e Ponto Flutuante.

- A estrutura da **Linguagem Humana** permite a junção de elementos necessários à sua comunicação, como o “ABC” e os números decimais de 0 à 9.
- O **menor elemento** da linguagem humana é o “**caractere**”, no computador é o “**dígito binário**”, **bit**, 0 e 1.
- Para os sistemas compreenderem os **caracteres** da linguagem humana, a linguagem computacional precisa ser traduzida por **compiladores** da leitura de alto nível para a leitura de baixo nível em conjunto de *bits*, o qual chamamos de **código de representação de caracteres de base 2**.

→ Cada **conjunto de bits de base 2** nos sistemas computacionais representaram um determinado conjunto de **caracteres**, **símbolos** e **expoente**, assim:

- **4 bits** por caractere deverão ser codificados **16 símbolos** diferentes ou **2^4** ;
- **5 bits** por caractere deverão ser codificados **32 símbolos** diferentes ou **2^5** ;
- **6 bits** por caractere deverão ser codificados **64 símbolos** diferentes ou **2^6** ;
- **7 bits** por caractere deverão ser codificados **128 símbolos** diferentes ou **2^7** ;
- **8 bits** por caractere deverão ser codificados **256 símbolos** diferentes ou **2^8** ;
- **9 bits** por caractere deverão ser codificados **512 símbolos** diferentes ou **2^9** , etc.

O **Byte**, assim como a representação de caracteres procuram definir grupos ordenado de **8 bits** não importando seu tamanho ao **armazenamento e transferência** dos dados.

IMPORTANTE: O termo **caractere** é muito empregado para fins comerciais (propaganda, apresentações a pessoas não familiarizadas com a computação), enquanto que o termo **Byte** é empregado na linguagem técnica dos profissionais da área de tecnologia.

- Os valores de medição nesses sistemas são as **potências de “base 2”** medidos em *Bytes* no momento de representar a **nomencatura** pela letra “B” em **maiúscula**:
- 1 **Byte** = 8 *bits*;
 - 1 kilobyte (**KB** ou KBytes) = 1024 **Bytes** ou 2^{10} ;
 - 1 megabyte (**MB** ou MBytes) = 1024 **Kilobytes** ou 2^{20} ;
 - 1 gigabyte (**GB** ou GBytes) = 1024 **Megabytes** ou 2^{30} ...
- Na transmissão de dados **entre** os dispositivos usa-se medições relacionadas **aos bits** e **não aos Bytes**, representado pela letra “b” em **minúsculo**:
- 1 kilobit (**Kb** ou Kbit) = 1024 **bits**;
 - 1 megabit (**Mb** ou Mbit) = 1024 **kilobits**;
 - 1 gigabit (**Gb** ou Gbit) = 1024 **megabits**;
 - 1 terabit (**Tb** ou Tbit) = 1024 **gigabits** ...

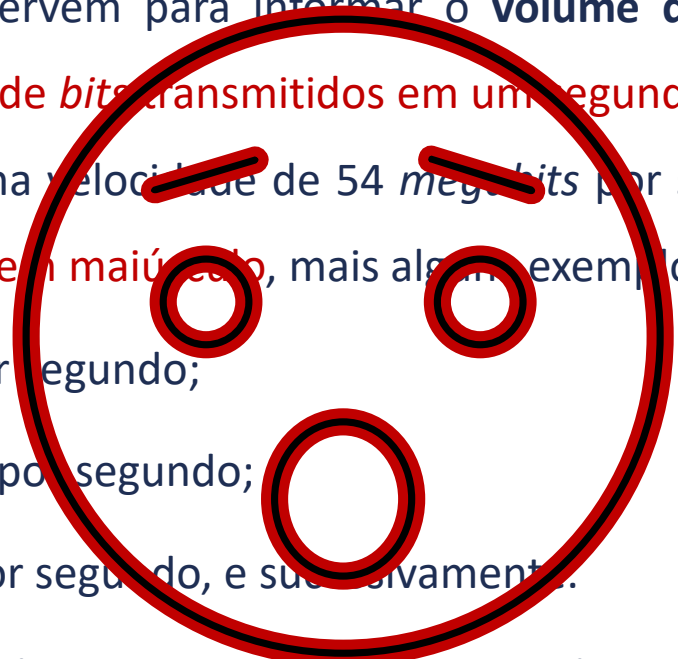
Mas qual seria o motivo?

→ As medições em *bits* servem para informar o **volume de dados em transmissões**, indicando a **quantidade de bits transmitidos em um segundo**, assim se um dispositivo é capaz de trabalhar a uma velocidade de 54 *megabits* por segundo, sua representação será **54 Mb/s**, com “M” em maiúsculo, mais alguns exemplos:

- 1 Kb/s = 1 kilobit por segundo;
- 1 Mb/s = 1 megabit por segundo;
- 1 Gb/s = 1 gigabit por segundo, e sucessivamente.

→ O sistema computacional interpreta 1 *kilobit* = 1.024 *bits*, 1 *megabit* = 1.048.576 *bits*, 1 *gigabit* = 1.073.741.824 *bits*...

→ Já, os fabricantes de discos rígidos, comercialmente, o 1 *kilobit* = 1.000 *bits*, 1 *megabit* = 1.000.000 *bits*, 1 *gigabit* = 1.000.000.000 *bits* para facilitar a nomenclatura do produto...



Devido ao impasse nessas terminologias e abreviações, a *International Electrotechnical Commission (IEC)* estipulou que as medições em **1024 Bytes** e seus múltiplos (potência de 2) fossem usados para a leitura binária.



PALAVRA

- Na linguagem humana a **palavra** é definida pelo número de caracteres para armazenar as variáveis e construir conjuntos de palavras convencional, **por exemplo**:
 - A palavra “**computador**” possui 10 caracteres;
 - A frase “**Mauro Aparecido**” possui 15 caracteres, contando o espaço.
- Nos sistemas aritmético eletrônicos (binário) o **conceito de palavra** é usado como um valor fixo e constante ao **dado que será processado**, assim:
 - ☺ No armazenamento das unidades digitais considera-se como medida válida um tamanho **mínimo igual a um 1 Byte**, necessário para formular uma “**Palavra**”.
 - ☺ Já análise do tamanho da palavra é analisado pela CPU, que processa os valores representados por uma quantidade “**X**” de **bits igual à uma palavra**, a partir de 8 **bits** para **poder indicar a capacidade de processamento do sistema**.

Comparações:

Linguagem humana **X** dos computadores.

Computadores	Humanos
Bit	Caractere
Byte e Caractere	Palavra
Palavra	Frases
Registro	Textos
Arquivo	Livros
Banco de Dados	Grupo de Livros

Bases e Sistemas de Numeração

A quantidade de dígitos numéricos em um sistema é chamado de “**base**”, onde:

- ☹ No sistema **decimal (humano)** a base é **10** ou b_{10} (0, 1, 2, 3, 4, 5, 6, 7, 8, 9);
- ☹ No sistema binário a **base é 2** ou b_2 (0 e 1);
- ☹ Temos mais alguns sistemas dentro da leitura computacional, mas estudaremos mais adiante:
 - ☞ **Octal** => base 8 ou b_8
 - ☞ **Hexadecimal** => base 16 ou b_{16} .

Por definição: O sistema de numeração são formados por conjuntos de símbolos, que ao serem utilizados representam as quantidades e as regras que definem suas formas das representações.

EXEMPLOS DE CONVERSÃO PARA A BASE 10

Base 10 (decimal):

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

Base 2 (binária):

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

Base 8 (octal):

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

Base 16 (hexadecimal):

$$\text{☺ } 231A_{16} = A * 16^0 + 1 * 16^1 + 3 * 16^2 + 2 * 16^3 = 8986_{10}$$

Em um **Sistema Posicional**, humano ou computacional, cada número está representado por “**N**”, maiúsculo, ou “**n**”, minúsculo, onde esses valores que dependem de sua **posição**.

Para encontrar esse valor usa-se a fórmula “ **$N = n * b^{\wedge}$** ”.

Onde:

- A simbologia “ **\wedge** ” é o expoente da base “**b**” representado por **b^{\wedge}** ;
- O “**n**” minúsculo indica a posição e também o símbolo do numeral de base 10 (unidade, dezena, centena, milhar, ...);

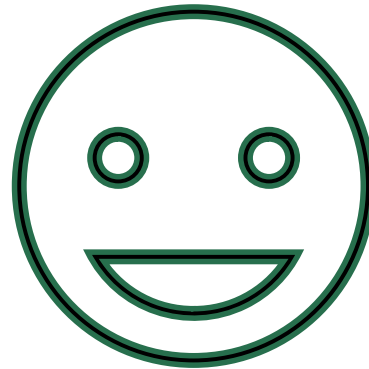
Exemplo:

- Uma base decimal contendo **três dígitos**, por exemplo $(678)_{10}$, teremos três expoente **a partir do expoente “0”** representada pela simbologia **b^{\wedge}** (b^0 , b^1 e b^2).
- Para a resultante “**N**”, maiúsculo, usa-se a seguinte **fórmula**:

$$N = ((n_1 * b^{\wedge}) + (n_2 * b^{\wedge}) + (n_3 * b^{\wedge}) + (n_4 * b^{\wedge}) + (n_5 * b^{\wedge}) + \dots)$$



**CALMA, a aplicação da fórmula apresentada
é simples na prática.**



Por exemplo \Rightarrow Calcular o decimal **2459**₁₀, onde a leitura deverá ser feita da **direita para a esquerda**:

- \rightarrow O dígito **9** representa a unidade, seu **expoente** é “0” e sua **posição** é “1”;
- \rightarrow O dígito **5** representa a dezena, seu **expoente** é “1” e sua **posição** é “2”;
- \rightarrow O dígito **4** representa a centena, seu **expoente** é “2” e sua **posição** é “3”;
- \rightarrow O dígito **2** representa o milhar, seu **expoente** é “3” e sua **posição** é “4”.

b^n (exponente da base “b”)	10^3	10^2	10^1	10^0
Posição do Número	4	3	2	1
Símbolo do Número	2	4	5	9

Fonte: Adaptada pelo autor (2023)

Mas como montar a fórmula do valor apresentado.



A **equação** do valor “2459” usando a fórmula “ **$N = n * b^{\wedge}$** ” é fácil:

$$2459 = 2 * 10^3 + 4 * 10^2 + 5 * 10^1 + 9 * 10^0$$

- Usando a fórmula $N = n * b^{\wedge}$ para o cálculo.
- O sentido de cálculo deverá ser **da esquerda para direita**.
- Vamos usar para o cálculo o número **$(56992)_{10}$** .
- **Cálculo detalhado:**

$$N = 5 * 10^4 + 6 * 10^3 + 9 * 10^2 + 9 * 10^1 + 2 * 10^0 =$$

$$N = 5 * 10000 + 6 * 1000 + 9 * 100 + 9 * 10 + 2 * 1 =$$

$$N = 50000 + 6000 + 900 + 90 + 2 =$$

$$N = (56992)_{10}$$

Para uma melhor fixação do **Sistema Posicional**, vamos detalhar o número 56992_{10} , e verificar algumas de suas propriedades:

- ☺ A **base 10 ou b_{10}** , indica que se trata de um decimal;
- ☺ O **primeiro** algarismo, 2, no número 5699**2**, simboliza a **unidade**;
- ☺ O **segundo** algarismo, 9, no número 569**9**2, simboliza as **dezenas**;
- ☺ O **terceiro** algarismo, 9, no número 56**9**92, simboliza as **centenas**;
- ☺ O **quarto** algarismo, 6, no número 5**6**992, que simboliza o **milhar**.
- ☺ O **quinto** algarismo, 5, no número **5**6992, que simboliza o **milhão**.

Então:

- Um **Sistema Posicional** usa um mesmo símbolo, podendo assumir valores diferentes dependendo de sua posição.
- Para saber-se o **valor** de qualquer **número** usando o Sistema Posicional, é necessário conhecer o **valor da posição** de cada símbolo.

Como exemplo vamos calcular o **expoente** no decimal $(345)_{10}$:

- ☺ A **posição** número é feita da direita para a esquerda com início em um.
- ☺ O **símbolo** indica o **valor real do número** também com início da direita para esquerda.
- ☺ A base “**b**” determina se o cálculo será de base 10, 2, 8 ou 16.

Calculo do **expoente**:

- O valor da unidade da **posição n** no **símbolo 5** é **0** de base 10 ou $10^0 = \text{unidade} = 5$;
- A dezena da **posição n** no **símbolo 4** é **1** de base 10 ou $10^1 = \text{dezena} = 40$;
- A centena da **posição n** no **símbolo 3** é **2** de base 10 ou $10^2 = \text{centena} = 300$.

TABELA PARA CONVERSÃO – $(345)_{10}$			
Expoente da base (b_{10})	2	1	0
Posições	3	2	1
Símbolos	3	4	5
Valor Posicional = $N = n * b^{\wedge}$	300	40	5

Fonte: Adaptada pelo autor (2023)

Uso **detalhado e prático** da fórmula ao preenchimento da tabela apresentada e calcular o valor do numeral decimal **$(5613)_{10}$** :

Sentido de Leitura / Base	Da direita para esquerda / Base 10 $\rightarrow (5613)_{10}$			
Expoente da base $\rightarrow b^n$	3	2	1	0
Posição do número	4	3	2	1
Símbolo do número $\rightarrow "n"$	5	6	1	3
Valor Posicional $\rightarrow N = n * b^n$	$N = 5 * (10^3)$ $N = 5 * 1000$ $N = 5000$	$N = 6 * (10^2)$ $N = 6 * 100$ $N = 600$	$N = 1 * (10^1)$ $N = 1 * 10$ $N = 10$	$N = 3 * (10^0)$ $N = 3 * 1$ $N = 3$

Fonte: Adaptada pelo autor (2023)

- ☺ Com o uso de tabela os cálculos deveram ser individuais.
- ☺ Somando os valores encontrados temos: **$5000 + 600 + 10 + 3 = (5613)_{10}$** .
- ☺ É provável **que nunca será necessário efetuar todos esses cálculos** para concluir que o número decimal $(5613)_{10}$ é igual a ele mesmo.

Treino de Conversão 1 para o decimal **(44678)₁₀**:

Sentido de Leitura / Base	Da direita para esquerda / Base 10 → (44678)₁₀				
Expoente da base → b^n					
Posição do número					
Símbolo do número → n					
Valor Posicional → $N = n * b^n$					

Fonte: Adaptada pelo autor (2023)

Treino de Conversão 2: Encontrar a base 10 do binário $(11010)_2$:

Sentido de Leitura / Base	Da direita para esquerda / Base 2 $\rightarrow (11010)_2$				
Expoente da base $\rightarrow b^n$					
Posição do número					
Símbolo do número $\rightarrow n$					
Valor Posicional $\rightarrow N = n * b^n$					

Fonte: Adaptada pelo autor (2023)

Treino de Conversão 3: Encontrar a base 10 do octal $(47271)_8$:

Sentido de Leitura / Base	Da direita para esquerda / Base 8 $\rightarrow (47271)_8$				
Expoente da base $\rightarrow b^n$					
Posição do número					
Símbolo do número $\rightarrow n$					
Valor Posicional $\rightarrow N = n * b^n$					

Fonte: Adaptada pelo autor (2023)

Treino de Conversão 4: Encontrar a base 10 do hexadecimal $(D54A)_{16}$:

Sentido de Leitura / Base	Da direita para esquerda / Base 16 $\rightarrow (D54A)_{16}$				
Expoente da base $\rightarrow b^n$					
Posição do número					
Símbolo do número $\rightarrow n$					
Valor Posicional $\rightarrow N = n * b^n$					

Fonte: Adaptada pelo autor (2023)

Agora que conhecemos algumas das notações binárias mais usadas, precisamos calcular sem o uso da fórmula e sem o uso de uma calculadora binária?



Antes, vamos facilitar a leitura e estudar a equivalência entre bases decimal, binária, octal e hexadecimal.

Organização e Arquitetura de Computadores

Sistema Binário - Bases e Sistemas de Numeração – Tabela de Bases de 0 à 31

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" ⇒ Parte 1				TABELA DE CONVERSÃO DE BASES Do valor "16" ao valor "31" ⇒ Parte 2			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16	Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0	16	10000	20	10
1	001	1	1	17	10001	21	11
2	010	2	2	18	10010	22	12
3	011	3	3	19	10011	23	13
4	100	4	4	20	10100	24	14
5	101	5	5	21	10101	25	15
6	110	6	6	22	10110	27	16
7	111	7	7	23	10111	27	17
8	1000	10	8	24	11000	30	18
9	1001	11	9	25	11001	31	19
10	1010	12	A	26	11010	32	1A
11	1011	13	B	27	11011	33	1B
12	1100	14	C	28	11100	34	1C
13	1101	15	D	29	11101	35	1D
14	1110	16	E	30	11110	36	1E
15	1111	17	F	31	11111	37	1F

Os sistemas **octais** e **hexadecimais** são formados pelas combinações de suas subdivisões, onde:

- **Octais** – Subdivisões de 3 em 3 *bits*;
- **Hexadecimais** – Subdivisões de 4 em 4 *bits*.

O cálculo das bases 8 e 16 a representação binária possui duas formas, pelo uso de uma tabela de conversões entre bases, **por exemplo**:

$$\underbrace{101}_5 \underbrace{111}_7 \underbrace{011}_3 \underbrace{101}_5_2 = 5735_8$$

$$\underbrace{1011}_B \underbrace{1101}_D \underbrace{1101}_D_2 = BDD_{16}$$

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" ⇒ Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

☺ **Entendendo:** Calcular a base 8 e 16 do binário

$(111000101011)_2$ usando a tabela de conversões:

→ $(111|000|101|011)_2$ para **Base 8 = $(7053)_8$**

→ $(1110|0010|1011)_2$ para **Base 16 = $(E2B)_{16}$**

→ $(00|1100|1111|0000)_2$ = base 16 = **$(CF0)_{16}$**

IMPORTANTE ⇒ **Todo o zero à esquerda do binário poderá ser desconsiderado no resultado FINAL.**

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" ⇒ Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Treino 1 \Rightarrow Calcular a base 8 e base 16 do binário

$(111010011110)_2$ usando a tabela de conversões:

$\rightarrow (111|010|011|110)_2$ para **Base 8** = $(XX)_8$

$\rightarrow (1110|1001|1110)_2$ para **Base 16** = $(XX)_{16}$

IMPORTANTE \Rightarrow Todo o zero a esquerda do binário poderá ser desconsiderado no resultado FINAL.

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" \Rightarrow Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Treino 2 \Rightarrow Calcular a base 2 para do valor $(2307)_8$

usando a tabela de conversões:



Dica: iniciar a leitura da direita para esquerda.

$\rightarrow (2307)_8$ para **Base 2** = $(XX)_2$

IMPORTANTE \Rightarrow **Todo o zero à esquerda**
do binário poderá ser desconsiderado no
resultado FINAL.

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" \Rightarrow Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Treino 3 \Rightarrow Calcular a base 2 e 16 dos valores $(35763)_8$ e $(38793)_8$, usando a tabela de conversões:

$= (35763)_8$ para **Base 2** = $(XX)_2$

$= (XX)_2 = (35763)_8$

$= (XX)_2 =$ para **Base 16** = $(XX)_{16}$

\Rightarrow Calcular a base 2 e 16 do **octal** (38793):

$= (38793)_8$ para **Base 2** = $(XX)_2$

$= (XX)_2 = (38793)_8$

$= (XX)_2 =$ para **Base 16** = $(XX)_{16}$

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" \Rightarrow Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Treino 4 \Rightarrow Calcular a base 2 e 16 dos valores $(567)_8$ e $(675012)_8$, usando a tabela de conversões:

– $(567)_8$ para $(XX)_2$

= $(XX)_2$ para $(XX)_{16}$

– $(675012)_8$ para $(XX)_2$

= $(XX)_2$ para $(XX)_{16}$

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" \Rightarrow Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Treino 5 \Rightarrow Encontrar o valor de base 8 do binário

$(011110011001111)_2$, usando a tabela de conversões:

$(011110011001111)_2$ para $= (XX)_8$

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" \Rightarrow Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Cálculos binários, octais e hexadecimais **sem o uso da Tabela.**

1. Passar o binário $(101101)_2$ para a base 10;
2. Iniciar a leitura do valor da **esquerda para a direita ou da direita para esquerda** se achar mais fácil;
3. Usar a tabela de conversões do professor para encontrar o expoente ($^{\wedge}$, **p - 1**), o símbolo (n) e a posição (p).

4. Iniciando o cálculo:

$$\Rightarrow N = n * b^{\wedge}$$

$$\Rightarrow N = 1 * b^5 + 0 * b^4 + 1 * b^3 + 1 * b^2 + 0 * b^1 + 1 * b^0 =$$

$$\Rightarrow N = 1 * 2^5 + 0 * 2^4 + 1 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0 =$$

$$\Rightarrow N = 32 + 0 + 8 + 4 + 0 + 1 =$$

$$\Rightarrow N = (101101)_2 = (45)_{10}$$

Sentido de Leitura / Base	Da direita para esquerda / Base 10 $\rightarrow (5613)_{10}$			
Expoente da base $\rightarrow b^n$	3	2	1	0
Posição do número	4	3	2	1
Símbolo do número $\rightarrow "n"$	5	6	1	3
Valor Posicional $\rightarrow N = n * b^{\wedge}$	N= 5 * (10 ³) N= 5 * 1000 N= 5000	N= 6 * (10 ²) N= 6 * 100 N= 600	N= 1 * (10 ¹) N= 1 * 10 N= 10	N= 3 * (10 ⁰) N= 3 * 1 N= 3

Fonte: Adaptada pelo autor (2023)

Treino 6 \Rightarrow Passar o valor $(10011101)_2$ para a base 10:

1. $(10011101)_2 = (XX)_{10}$
2. O resultado em conversões sempre será de base 10.
3. A fórmula para encontrar o expoente é p (posição) $- 1$.
4. $n = 8$ posições, o expoente sempre terá seu início em 0 ou b^0 , e para este treino terminará no expoente 7 ou b^7 .
 - $N = n * b^{\wedge}$
 - $N =$
 - $N =$
 - $N = (XX)_{10}$
 - $(10011101)_2 = (XX)_{10}$

Treino 7 \Rightarrow Passar o valor $(1101111101)_2$ para a base 10:

1. $(\overbrace{1101111101}^{\rightarrow})_2 = (XX)_{10}$
2. O resultado em conversões sempre será de base 10.
3. A fórmula para encontrar o expoente é p (posição) $- 1$.
4. $n = 9$ posições, o expoente sempre terá seu início em 0 ou b^0 , e para este treino terminará no expoente 9 ou b^9 .
 - $N = n * b^{\wedge}$
 - $N =$
 - $N =$
 - $N = XX$
 - $(1101111101)_2 = (XX)_{10}$

Cálculos binário, octal e hexa com o uso da Tabela.

→ Passar o valor $(27)_8$ para a base 10:

– $(27)_8 = (?)_{10}$

– **base = 8 e n = 2 posições**, portanto:

– $2 * 8^1 + 7 * 8^0 =$

– $16 + 7 =$

– $(23)_{10}$

– $(27)_8 = (23)_{10}$

– $(27)_8 = (23)_{10} = (010111)_2$

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" ⇒ Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Treino 8 \Rightarrow Passar o valor $(357)_8$ para a base 10, 2 e 16

usando a tabela:

Passando b_8 para b_{10} :

😊 **As bases 2, 8, 10 e 16 terão $n = 3$ posições.**

😊 Os expoentes estão entre 0 e 2.

😊 $(357)_8 = (XX)_{10}$

😊 $XX = (XX)_{10}$

😊 $(XX)_{10} = (XX)_2 = XX = (XX)_2$

😊 $(XX)_2 = (XX)_{16} =$

😊 $(357)_8 = (XX)_{10} = (XX)_2 = (XX)_{16}$

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" \Rightarrow Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Exemplos de cálculos binário, octal e hexa.

→ Passar o valor $(2A5)_{16}$ para a base 10:

– $(2A5)_{16} = (XX)_{10}$

– base = 16 e n = 3 posições, portanto:

– $2 * 16^2 + A=10 * 16^1 + 5 * 16^0 =$

– $512 + 160 + 5 =$

– $(677)_{10}$

– $(2A5)_{16} = (677)_{10}$

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" ⇒ Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Treino 9 \Rightarrow Passar o valor $(C3D4)_{16}$ para a base 10:

- $(C3D4)_{16} = (XX)_{10}$
- **base = 16** e **n = 4**, portanto:
- **N=**
- **N =**
- **N = $(XX)_{10}$**
- **$N = (C3D4)_{16} = (XX)_{10}$**

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" \Rightarrow Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Cálculos binário, octal e hexa usando **DIVISÕES**.

→ A ideia é em passar a **base decimal** para a **base binária** através de sucessivas divisões:


- $(25)_{10} = (XX)_2$
- $25 : 2 = 12$ (resto **1**);
- $12 : 2 = 6$ (resto **0**);
- $6 : 2 = 3$ (resto **0**);
- $3 : 2 = \mathbf{1}$ (resto **1**).
- **Resultado = $(11001)_2$**

Entendo o cálculo:

- Em 25: 2 o resultado é 12 com resto **1**.
- Em 12: 2 o resultado é 6 com resto **0**.
- Em 6: 2 o resultado é 3 com resto **0**.
- **Em 3: 2 o resultado = 1 com resto = 1.**
- **Resultante é igual a $(25)_{10} = (11001)_2$.**

Logo, o valor do binário será o **resultado** da última divisão
mais os **restos** de todas divisões **de baixo para cima**.

Treino 10 \Rightarrow Passar a **base 10** para **base 2** usando sucessivas divisões:

- $(340)_{10} = (XX)_2$
 - $340 : 2 = 170$ (resto **X**);
 - $170 : 2 = 85$ (resto **X**);
 - $85 : 2 = 42$ (resto **X**);
 - $42 : 2 = 21$ (resto **X**);
 - $21 : 2 = 10$ (resto **X**);
 - $10 : 2 = 5$ (resto **X**);
 - $5 : 2 = 2$ (resto **X**);
 - $2 : 2 = \mathbf{X}$ (resto **X**)
 - **Resultado = $(XX)_2$**
- 

O valor do binário será o **resultado** da última divisão mais os **restos** de todas divisões **de baixo para cima**.

Cálculos binário, octal e hexa.

→ Passar uma **base 10** para a **base 8** usando sucessivas divisões:

– $(3964)_{10} = (XX)_8$

– $3964 : 8 = 495$ (resto 4)

– $495 : 8 = 61$ (resto 7)

– $61 : 8 = 7$ (resto 5)

– **Resultado = $(XX)_8$**

O valor do binário será o **resultado** da última divisão mais
os **restos** de todas divisões **de baixo para cima**.

Treino 11 \Rightarrow Passar a **base 10** para **base 8** usando sucessivas divisões:

- $(2459)_{10} = (XX)_8$
- $2459 : 8 = 307$ (resto **X**)
- $307 : 8 = 38$ (resto **X**)
- $38 : 8 = \mathbf{X}$ (resto **X**)
- **Resultado** = $(XX)_8$

O valor do binário será o **resultado** da última divisão mais os **restos** de todas divisões **de baixo para cima**.

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" \Rightarrow Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Exemplos de cálculos binário, octal e hexa.

→ Passar uma **base 10** para **base 16** usando sucessivas divisões:

- $(2574)_{10} = (XX)_{16}$
- $2574 : 16 = 160$ (resto **14**)
- $160 : 16 = 10$ (resto **0**)
- **Resultado = $(A0E)_{16}$**

O valor do binário será o **resultado** da última divisão mais os **restos** de todas divisões **de baixo para cima**.

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" ⇒ Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Treino 12 \Rightarrow Passar a **base 10** para **base 16** por sucessivas divisões:

- $(35679)_{10} = (XX)_{16}$
- $35679 : 16 = 2229$ (resto **X**)
- $2229 : 16 = 139$ (resto **X**)
- $139 : 16 = \mathbf{X}$ (resto **X**)
- **Resultado** = $(XX)_{16}$

O valor do binário será o **resultado** da última divisão mais os **restos** das demais divisões, onde:
10 = A, 12 = C, 2 = 2 (valores da tabelados).

TABELA DE CONVERSÃO DE BASES Do valor "0" ao valor "15" \Rightarrow Parte 1			
Decimal Base 10	Binário Base 2	Octal (3 bits) Base 8	Hex. (4 bits) Base 16
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Resumo dos Conceitos

- O menor **elemento** disponível de uma linguagem humana é o caractere (abcde[...]) e os valores de 0 a 9) .
- A **menor unidade** de informação **armazenável** é o **bit** binário com apenas dois valores 0 e 1.
- A **menor unidade** de informação na **linguagem** humana é o “**caractere**”.
- O computador usa a leitura **bit a bit** codificadas em **grupos ordenados de bits**.
- Qualquer caractere armazenado em um sistema de computação deverá ser convertido em um **conjunto de bits**, que recebe o nome de “**código de representação de caracteres**”.
- Cada sistema **necessita definir a quantidade de bits necessários a sua codificação**.
- A **organização** de cada **conjunto de bits** irão gerar a representação de um determinado **caractere**, por exemplo:



- ☺ A representação de caracteres de **cindo bits por caractere ou 2^5 serão codificados 32 símbolos diferentes**, seis bits ou 2^6 serão codificados 64 símbolos diferentes, e assim por diante com 2^7 , 2^8 , 2^9 , 2^{10} , 2^{11} [...].
- O **byte** define também o **menor tamanho** de uma “palavra” no computador, e considerado o **elemento de referência** para a **construção e funcionamento** dos dispositivos de **armazenamento** e dos **processos de transferência de dados** entre periféricos, a UCP e a MP.
 - Os principais códigos de representação de *caracteres* utilizam **grupos de 08 bits por caractere** para que esses *caracteres* se tornem semelhantes as **palavras**.
 - O termo **caractere** costuma ser mais empregado para **fins comerciais**, e o **byte** para a **linguagem técnica**.
 - Em um sistema de computação, os números podem ser **representados com conjuntos de 32 bits ou 4 bytes** de dados para cada número.

- Os computadores são máquinas binárias e suas indicações numéricas referem-se a potência de 2;
- A potência de 10 se refere ao sistema métrico dos humanos;
- O “K” representa 1.024 unidades ou 2^{10} ;
- O “M” ou mega, representa 1.048.576 (1024K) unidades = $1024 * 1024$ ou $2^{10} * 2^{10} = 2^{20}$;
- O “G” ou giga, representa 1.073.741.824 (1.048.576K) unidades = $2^{10} * 2^{10} * 2^{10} = 2^{30}$, por exemplo:
 - ☺ 512 *Kbytes* corresponde a um valor de $512 * 1024 = 524.288$ bytes;
 - ☺ 32 *Mbytes*, trinta e dois mega caracteres, corresponde a $32 * 1024 * 1024 = 33.554.432$ caracteres;
 - ☺ 2 *Gbytes* corresponde a $2 * 1024^3 = 2.147.483.648$ bytes.

- O conceito de “**palavra**” seria um conjunto de **bits** que estão relacionados ao armazenamento e a transferências de informações entre a **UCP** e a **MP** com relação ao processamento de dados pelo sistema de processamento e controle.
- A forma mais empregada a representação numérica é chamada de **notação decimal**, onde os algarismos de um número assumem valores diferentes, que dependem de sua posição relativa no número, por exemplo, no número “**2378907**” temos 7 posições: “**2**”, “**3**”, “**7**”, “**8**”, “**9**”, “**0**” e “**7**”.
- A quantidade de algarismos disponíveis em um dado sistema de numeração é chamada de “**base**”, que serve para contarmos grandezas maiores, indicando a noção de agrupamento dessas bases.
- Sistema de dez algarismos, chamamos de **base 10** (0...9), dois algarismos de **base 2** (0 e 1), oito algarismos de **base 8** (0...7), dezesseis algarismos de **base 16** (0...9 e A...F).

Bibliografia Básica

TANENBAUM, A. S. Organização estruturada de computadores. 6. ed. São Paulo: Pearson Prentice Hall, 2013 (e-book).

MONTEIRO, M. A. Introdução à organização de computadores. 4. ed. Rio de Janeiro: LTC, 2002.

STALLINGS, W. Arquitetura e organização de computadores: projeto para o desempenho. 5. ed. São Paulo: Prentice-Hall, 2002.

Bibliografia Complementar

CORRÊA, A. G. D. [org.]. Organização e arquitetura de computadores. São Paulo: Pearson Education do Brasil, 2016 (e-book).

DELGADO, J.; RIBEIRO, C. Arquitetura de computadores. 5. ed. Rio de Janeiro: LTC, 2017 (e-book).

PAIXÃO, R. R. Arquitetura de computadores - PCs. São Paulo: Érica, 2014 (e-book).

WEBER, R. F. Fundamentos de arquitetura de computadores. 4. ed. Porto Alegre: Bookman, 2012 (e-book).

WIDMER, N. S.; MOSS, G. L.; TOCCI, R. J. Sistemas digitais: princípios e aplicações. 12. ed. São Paulo: Pearson Education do Brasil, 2018 (e-book).

Conteúdo elaborado por:

Prof. Ms. Celso Candido
celsoc@unicid.edu.br

OneDrive: https://cutt.ly/Alunos_Unicid_Aulas

Fim da Apresentação