Arquitetura e Organização de Computadores

Professor: Lucas Cambuim

Aula: Conversão de Bases e Aritmética Computacional

Objetivos

- Entender conceitos básicos de sistemas de numeração como base, valor posicional e valor de símbolo.
- Entender como trabalhar com números representados nos sistemas de numeração binário, octal e hexadecimal.
- Abreviar números binários como números octais ou hexadecimais.
- Converter números octais e hexadecimais em números binários.
- Converter nos dois sentidos entre números decimais e seus equivalentes binários, octais e hexadecimais.
- Entender a aritmética binária e como os números binários negativos são representados utilizando a notação de complemento de dois.
- Entender os números fracionários

Roteiro

- O Sistema de Numeração
  - Introdução
- O Sistema de Numeração Binário
  - Conversão do Sistema Decimal para o Sistema Binário
- O Sistema de Numeração Octal
  - Conversão do Sistema Decimal para o Sistema Octal
  - Conversão do Sistema Octal para o Sistema Binário
  - Conversão do Sistema Binário para o Sistema Octal
- O Sistema de Numeração Hexadecimal
  - Conversão do Sistema Decimal para o Sistema Hexadecimal
  - Conversão do Sistema Hexadecimal para o Sistema Binário
  - Conversão do Sistema Binário para o Sistema Hexadecimal
- Números Fracionários
  - Conversão de Números Binários Fracionários em Decimais
  - Conversão de Números Decimais Fracionários em Binários
- Operações Aritméticas no Sistema Binário
  - Adição no Sistema Binário
  - Subtração no Sistema Binário
  - Multiplicação no Sistema Binário
  - Divisão no Sistema Binário
- Representação e operação de números com sinal
  - Sinal e magnitude
  - Complemento a 2
- Álgebra de boole

Sistemas de Numeração

- Método para representar números
  - Necessidade do homem contar objetos
  - Realizar operações aritméticas
  - Soma (+) , Subtração (-) , Divisão (/) , Multiplicação (*)
- O sistema decimal é o mais importante dos sistemas numéricos.
  - Deriva dos nossos antepassados utilizarem os 10 dedos para contar
  - Ele está fundamentado em certas regras que são a base de formação para qualquer outro sistema.
- Além do sistema decimal, que apresenta 10 algarismos distintos de 0 a 9, existe o binário, o octal e o hexadecimal.
  - O sistema binário e o hexadecimal são muito importantes nas áreas de técnicas digitais e informática.

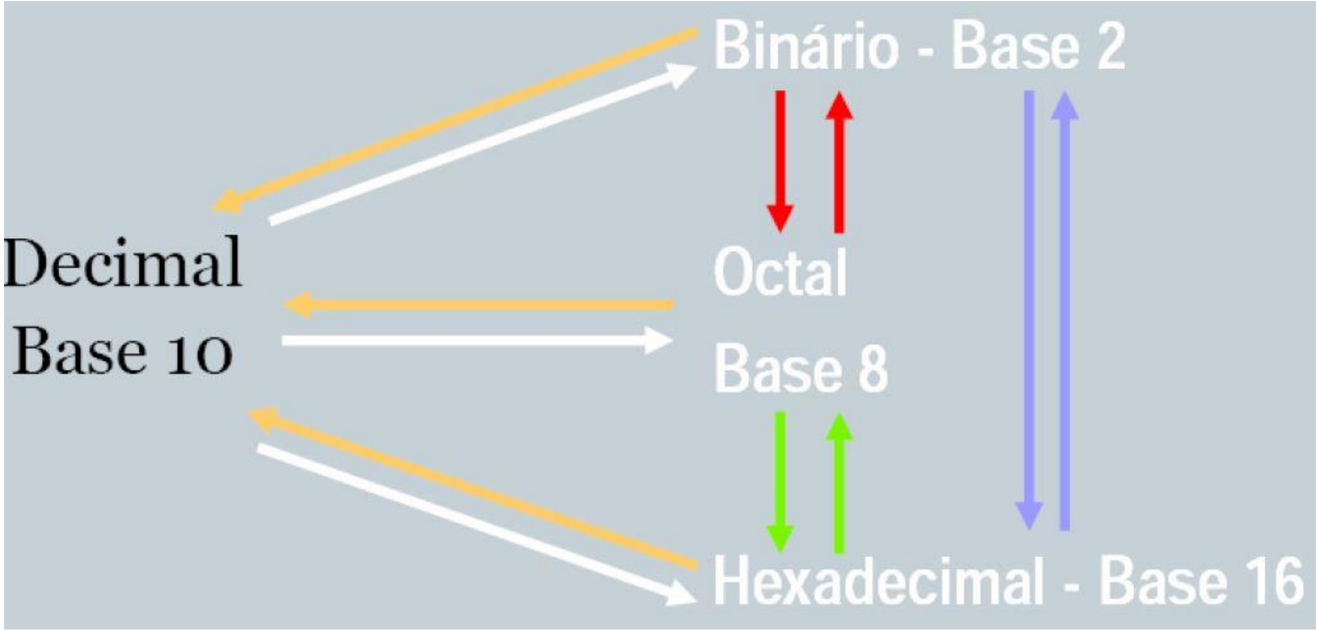
Sistemas de Numeração

- O sistema binário, por sua vez, apresenta somente 2 algarismos (0 e 1), com os quais é possível representar qualquer quantidade, até mesmo números fracionários.
- No sistema octal existem 8 algarismos que vão de 0 a 7.
- Para representar o sistema hexadecimal são utilizados 10 algarismos e as 6 primeiras letras do alfabeto e, desta forma, tem-se:
  - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.
- Base: É a quantidade de algarismos disponíveis
  - Sistema decimal – Base 10
  - Sistema binário – Base 2
  - Sistema octal – Base 8
  - Sistema hexadecimal – Base 16

BASE	ALGARISMOS
BASE 10 (DECIMAL)	0 - 9
BASE 2 (BINÁRIO)	0 E 1
BASE 8 (OCTAL)	0 - 7
BASE 16 (HEXADECIMAL)	0-9, A-F

Sistemas de Numeração

- Observando a formação dos infinitos números do sistema decimal é possível aprender as regras de formação dos demais sistemas numéricos.



Sistemas de Numeração Decimal

- Para conceber a formação do sistema decimal basta observar o hodômetro (marcador de quilômetro) de um automóvel.

- Quando a "rodinha" das unidades comuta de 9 para 0, um pino nessa rodinha força a rodinha das dezenas a avançar de 1. Assim ocorre sucessivamente formando todos os algarismos.

Sistemas de Numeração

- O mesmo se observa nos demais sistemas.
- No binário, por exemplo, quando a rodinha da unidade alcança 1 e posteriormente comuta para zero, a rodinha da dezena avança para 1.
- Pode-se notar que a quantidade de dígitos necessário para representar um número qualquer, no sistema binário, é muito maior quando comparado ao sistema decimal.

Sistemas de Numeração decimal

- O número decimal 975_{10} pode ser representado da seguinte forma:

$$975_{10} = 900 + 70 + 5 = 9 \times 10^2 + 7 \times 10^1 + 5 \times 10^0$$

- Neste exemplo, nota-se que o algarismo menos significativo (5) multiplica a unidade (1 ou 10^0), o segundo algarismo (7) multiplica a dezena (10 ou 10^1) e o mais significativo (9) multiplica a centena (100 ou 10^2).
- A soma dos resultados irá representar o número.

Sistemas de Numeração

- Pode-se afirmar que, de maneira geral:

- A regra básica de formação de um número consiste no somatório de cada algarismo correspondente multiplicado pela base (no exemplo o número 10 ou 2 ou 8 ou 16) elevada por um índice conforme o ***posicionamento*** do algarismo no número.
  - Por isso chamado de sistema posicional
  - O valor absoluto: o valor propriamente dito
  - O valor relativo: o valor multiplicado por 10 elevado a sua posição no número.

Sistemas de Numeração

- Assim, um sistema de numeração genérico pode ser expresso da seguinte forma:

$$N = d_n \times B^n + \ldots + d_3 \times B^3 + d_2 \times B^2 + d_1 \times B^1 + d_0 \times B^0 \qquad (3.2)$$

- Onde:

N é a representação do número na base B;

d_n é o dígito ou algarismo na posição n;

B é a base do sistema utilizado

n é o peso posicional do dígito ou algarismo.

Sistemas de Numeração Decimal

Dígitos Decimais:

0
1
2
3
4
5
6
7
8
9

Base =10

Potências de base 10

$10^0 = 1$

$10^1 = 10$

$10^2 = 100$

$10^3 = 1000$

$10^4 = 10\ 000$

Sistemas de Numeração

- Exemplo: na base 10, podemos representar um número:
  - N= 3748
- Onde:
  - n = 4 (quatro dígitos inteiros)
  - Utilizando a fórmula indicada na Equação 3.2
  - $N = 3 * 10^3 + 7 * 10^2 + 4 * 10^1 + 8 * 10^0 = 3000 + 700 + 40 + 8 = 3748_{10}$

Sistemas de Numeração Binário

Potências de base 2

$2^0 =$	1
$2^1 =$	2
$2^2 =$	4
$2^3 =$	8
$2^4 =$	16
$2^5 =$	32
$2^6 =$	64
$2^7 =$	128
$2^8 =$	256
$2^9 =$	512
$2^{10} =$	1024

Dígitos Binários:

0
1

Base =2

Sistema de Numeração Binário

- O sistema binário utiliza dois dígitos, ou seja, possui base 2. De acordo com a definição de um sistema de numeração genérico, o número binário 1101 pode ser representado da seguinte forma:

  $1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

  $1101_2 = 8 + 4 + 0 + 1 = 13_{10}$

- Nota-se que o número 1101 na base 2 é equivalente ao número 13 na base 10, ou seja, $1101_2 = 13_{10}$.
- Esta regra possibilita a conversão do sistema binário em decimal.

Sistema de Numeração Binário

- Números com base 2, foram criados para representar os sinais que o computador entende, ligado e desligado.

- O sistema binário é a base para a álgebra booleana, o que permite representar por circuitos eletrônicos digitais (portas lógicas) os números, os caracteres e realizar operações lógicas e aritméticas.

- A eletrônica digital e a computação estão baseadas no sistema binário e na lógica de boole, que permite representar por circuitos eletrônicos digitais, os números, as letras e realizar operações lógicas e aritméticas.

Sistema de Numeração Binário

- A vantagem do sistema binário reside no fato de que, possuindo apenas dois dígitos, estes são facilmente representados por dois níveis de tensão, uma chave aberta e uma chave fechada ou, um relé ativado e um relé desativado, ou, um transistor saturado e um transistor cortado; o que torna simples a implementação de sistemas digitais mecânicos, eletromecânicos ou eletrônicos.
- Em computação, chama-se um dígito binário (0 ou 1) de *bit*, que vem do inglês *Binary Digit*. Um agrupamento de 8 bits corresponde a um byte (Binary Term). Um grupamento de 4 bits é chamado de nibble.

Exemplos

- Seja o número na base 2 $(1011)_2$
  - Aplicando a Eq. 3.2, como ficaria?:
    - $1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0 = 8 + 0 + 2 + 1 = (11)_{10}$

- $(1043)_5 =$
  - Aplicando a Eq. 3.2, como ficaria?
    - $1 * 5^3 + 0 * 5^2 + 4 * 5^1 + 3 * 5^0 = 125 + 0 + 20 + 3 = (148)_{10}$

Sistemas de Numeração Hexadecimal

Dígitos Hexadecimal:

0 1
2 3
4 5
6 7
8 9
16

A = 10
B = 11
C = 12
D = 13
E = 14
F = 15

Base =16

Potências de base 16

$16^0 = 1$

$16^1 = 16$

$16^2 = 256$

$16^3 = 4096$

$16^4 = 65\ 536$

- E largamente utilizado na área dos microprocessadores e também no mapeamento de memórias em sistemas digitais.
- Trata-se de um sistema numérico muito importante, aplicado em projetos de software e hardware.
- Foi criado para facilitar a representação e manuseio de bytes (conjunto de 8 bits)

Sistemas de Numeração Octal

Dígitos Hexadecimal:

0
1
2
3
4
5
6
7

Base =8

Potências de base 16

$8^0 = 1$

$8^1 = 8$

$8^2 = 64$

$8^3 = 512$

$8^4 = 4096$

Este sistema é pouco utilizado no campo da Eletrônica Digital, tratando-se apenas de um sistema numérico intermediário dos sistemas binário e hexadecimal.

Sistemas de Numeração

- Exemplo: na base 16 o número:
  - N= $1A7B_{16}$
- O seu valor na base 10 será obtido usando-se a Equação 3.2
- Onde:
  - n = 4 (quatro dígitos inteiros)
  - B = 16
  - $N = 1 * 16^3 + 10 * 16^2 + 7 * 16^1 + 11 * 16^0 = 4096 + 2560 + 112 + 11 = 6779_{10}$

Sistemas de Numeração

- Observamos que na Eq 3.2 foram usados os valores 10 (para o algarismo A) e 11 (para o algarismo B), Por isso obtemos o valor do número na base 10.
- Em outras palavras, utilizamos valores e regras de aritmética na base 10 e por isso, o resultado encontrado é um valor na decimal.

Tabela 3.1

Base 2	Base 8	Base 10	Base 16
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

Sistemas de Numeração

- Podemos observar que os dígitos octais e hexadecimais correspondem a combinações de 3 (octais) e 4 (hexadecimais) bits (algarismos binários)
  - Isso é devido a essas bases serem todos de tamanho de potência de 2
- Isso permite converter rapidamente de uma base para a outra ou vice e versa.

Tabela 3.1

Base 2	Base 8	Base 10	Base 16
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

Conversão de Bases

- Conversão entre bases potência de 2
  - De base 2 para a base 8, onde 8 = 2^3
    - Basta dividi-lo, da **direita para a esquerda** em grupos de 3 bits.
    - Para cada grupo acha-se o algarismo octal equivalente.
    - Exemplo1: $(111010111)_2 = (\quad)_8$

$$(111)\ (010)\ (111)_2$$
$$7 \qquad 2 \qquad 7 \quad = (727)_8$$

Exemplo2: $(1010011111)_2 = (\quad)_8$

$$(001)(010)\ (011)\ (111)_2$$
$$1 \qquad 2 \qquad 3 \qquad 7 = (1237)_8$$

Tabela 3.1

Base 2	Base 8	Base 10	Base 16
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

Conversão de Bases

- Conversão entre bases potência de 2
  - De base 8 para base 2
    - Substitui-se cada algarismo octal pelo seus 3 bits correspondentes

    - Exemplo1: $(327)_8 = (\quad)_2$

      $(011)\ (010)\ (111)_2 = (011010111)_2$

      3 2 7

    Exemplo2: $(673)_8 = (\quad)_2$

      $(110)\ (111)\ (011)_2 = (110111011)_2$

      6 7 3

Tabela 3.1

Base 2	Base 8	Base 10	Base 16
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

Conversão de Bases

- Conversão entre bases potência de 2
  - De base 2 para a base 16, onde 8 = 2^4
    - Basta dividi-lo, da **direita para a esquerda** em grupos de **4 bits.**
    - Para cada grupo acha-se o algarismo hexadecimal equivalente.

Tabela 3.1

Base 2	Base 8	Base 10	Base 16
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

- Exemplo1: $(1011011011)_2 = (\quad)_{16}$

$$(0010)\ (1101)\ (1011)_2 = (2DB)_{16}$$

$$2 \qquad D \qquad B$$

Exemplo2: $(10011100101101)_2 = (\quad)_{16}$

$$(0010)(0111)\ (0010)\ (1101)_2 = (272D)_{16}$$

$$2 \qquad 7 \qquad 2 \qquad D$$

Conversão de Bases

- Conversão entre bases potência de 2
  - De base 16 para base 2
    - Substitui-se cada algarismo hexadecimal pelo seus **4 bits** correspondentes
    - Exemplo1: $(306)_{16} = (\quad)_2$

      $(0011)\ (0000)\ (0110)_2 = (011010111)_2$

      3 0 6
    - Exemplo2: $(F50)_{16} = (\quad)_2$

      $(1111)\ (0101)\ (0000)_2 = (110111011)_2$

      F 5 0

Tabela 3.1

Base 2	Base 8	Base 10	Base 16
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

Conversão de Bases

- Conversão entre bases potência de 2
  - De base 8 para base 16
    - Primeiro converte para a base 2 e depois para a base 16
  - De 16 para a base 8
    - Primeiro converte para a base 2 e depois para a base 8

Conversão de Bases

- Conversão entre bases potência de 2
  - Exemplo1: $(3174)_8 = (\quad)_{16}$

    1º Passo (p/ base 2):

    $(011)\,(001)\,(111)\,(100)_2 = (011001111100)_2$

    2º Passo (p/ base 16):

    $(0110)\,(0111)\,(1100) = (67C)_{16}$

  - Exemplo2: $(254)_8 = (\quad)_{16}$

    1º Passo: $(010)\,(101)\,(100)_2 = (010101100)_2$

    2º Passo: $(1010)\,(1100)_2 = (AC_{16})$

Conversão de Bases

- Conversão entre bases potência de 2
  - Exemplo3: $(2E7A)_{16} = (\quad)_8$

    1º Passo (p/ base 2):

    $$(0010)\ (1110)\ (0111)\ (1010)_2 = (0010111001111010)_2$$

    2º Passo (p/ base 8):

    $$(010)\ (111)\ (001)(111)\ (010)_2 = (27172)_8$$

  - Exemplo4: $(3C7)_{16} = (\quad)_8$

    1º Passo: $(0011)\ (1100)\ (0111)_2 = (1111000111)_2$

    2º Passo: $(001)\ (111)\ (000)\ (111)_2 = (1707)_8$

Conversão do Sistema de base B para o Sistema Decimal

- Empregamos a Eq 3.2 do slide 11

$$N = d_n \times B^n + \ldots + d_3 \times B^3 + d_2 \times B^2 + d_1 \times B^1 + d_0 \times B^0 \qquad (3.2)$$

- Exemplo: $(101101)_2 = (\quad)_{10}$

Substituindo na Eq 3.2 as letras pelos valores do exemplo, teremos:

$b = 2$ (a base origem do número a ser convertido)

$n = 6$ (6 algarismos)

$n - 1 = 5$ (Expoente do 1º produto mais à esquerda)

$d_{\{n-1\}} = 1$ (Algarismo mais à esquerda)

Os demais produtos seguem a sequência da Eq. 3.2, resultando em:

$1 * 2^5 + 0 * 2^4 + 1 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0 =$

$= 32 + 0 + 8 + 4 + 0 + 1 = (45)_{10}$

Conversão do Sistema de base B para o Sistema Decimal

- Exercícios:

- $(27)_8 = (\quad)_{10}$
- $(2A5)_{16} = (\quad)_{10}$
- $(6734)_8 = (\quad)_{10}$
- $(27)_8 = (\quad)_{10}$

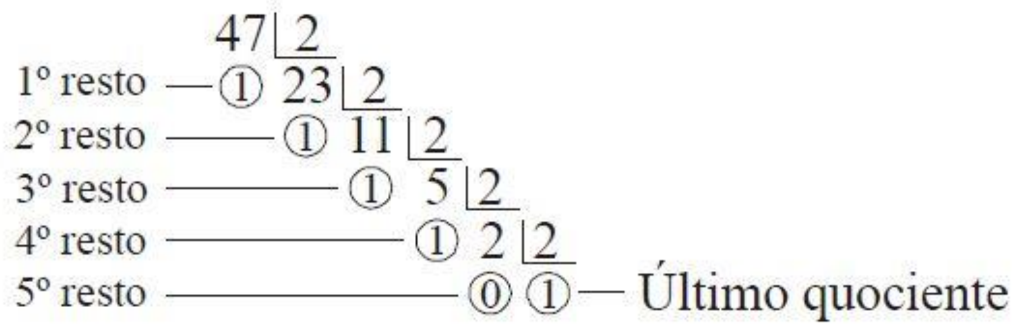
Conversão do Sistema Decimal para o Sistema de Base B

- Para se converter um número decimal em binário, aplica-se o método das divisões sucessivas.

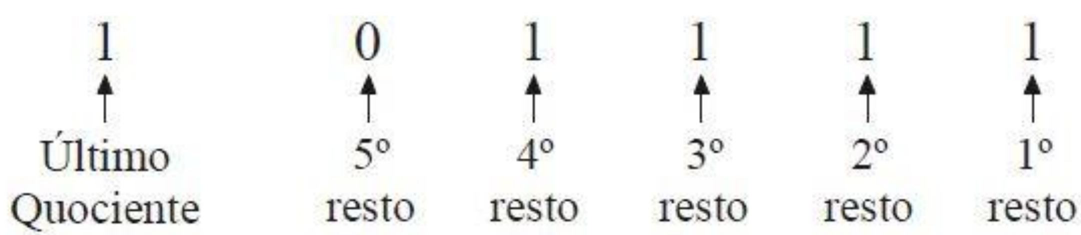
- Este método consiste em efetuar sucessivas divisões pela base a ser convertida até que:
  - Abordagem 1: o quociente seja igual a 0 ou
    - O número transformado será composto por todos os restos na ordem inversa às divisões.
  - **Abordagem 2: o último quociente possível (adotado)**
    - ou seja, quando o quociente for menor que o divisor termine a divisão.
    - O número transformado será composto por este último quociente (algarismo mais significativo) e, todos os restos na ordem inversa às divisões.

Conversão do Sistema Decimal para o Sistema Binário

- Neste caso, será efetuado sucessivas divisões pelo algarismo 2, base do sistema binário.



- O último quociente será o algarismo mais significativo e ficará colocado à esquerda. Os outros algarismos seguem-se na ordem até o 1º resto:



- Como mostra o exemplo, $47_{10} = 101111_2$.

Conversão do Sistema Decimal para o Sistema Binário

- Exemplos
  - $(45)_{10} = (\quad)_2$
    - 45/2 = 22 :: resto_0 = 1 (algarismo menos significativo)
    - 22/2 = 11 :: resto_1 = 0
    - 11/2 = 5 :: resto_2 = 1
    - 5/2 = 2 :: resto_3 = 1
    - 2/2 = 1 :: resto_4 = 0
    - Como 1 < 2 então acabaram as divisões.
    - Assim temos: $(101101)_8$
  - $(97)_{10} = (\quad)_2$
    - Resposta: $(1100001)_2$

Conversão do Sistema Decimal para o Sistema Binário

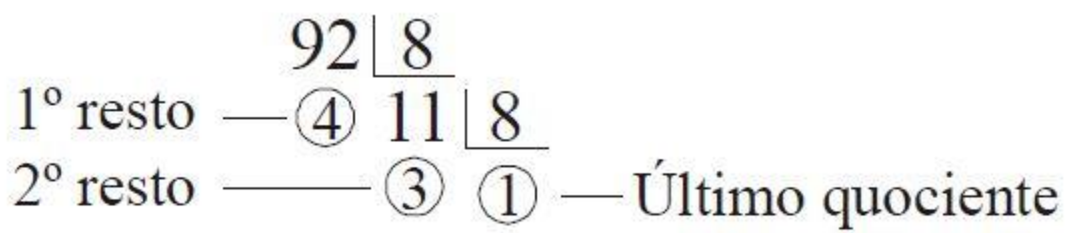
- Como mostra o exemplo, $47_{10} = 101111_2$.

- Na prática, o bit menos significativo de um número binário recebe a notação de LSB (“Least Significant Bit) e o mais significativo de MSB (“Most Significant Bit”).

Conversão do Sistema Decimal para o Sistema Octal

- Neste caso, será efetuado sucessivas divisões pelo algarismo 8, base do sistema octal.

- Para exemplificar, será realizada a conversão do número 92_{10} para o sistema octal:



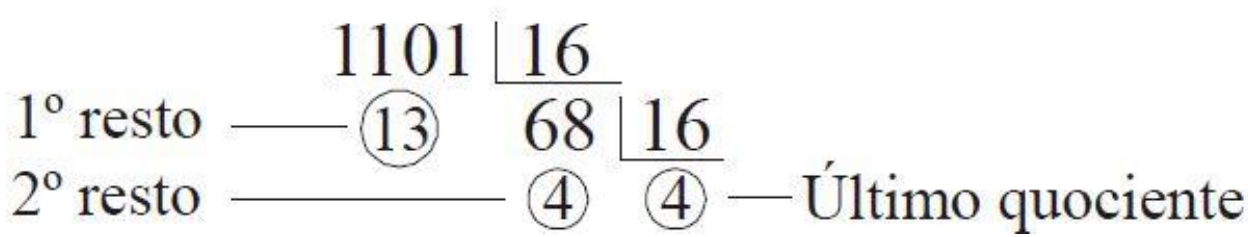
- Assim, seguindo a mesma regra de formação, $92_{10} = 134_8$.

Conversão do Sistema Decimal para o Sistema Octal

- Exemplos
  - $(3964)_{10} = (\quad)_8$
    - 3964/8 = 495 :: resto_0 = 4 (algarismo menos significativo)
    - 495/8 = 61 :: resto_1 = 7
    - 61/8 = 7 :: resto_2 = 5
    - Como 7 < 8 então acabou as divisões. Assim temos: $(7574)_8$
  - $(483)_{10} = (\quad)_8$
    - 483/8 = 60 :: resto_0 = 3
    - 60/8 = 7 :: resto_1 = 4
    - Como 7 < 8 então acabou as divisões. Assim temos: $(743)_8$
    - Para verificar:
      - $(743)_8 = 7 * 8^2 + 4 * 8^1 + 3 * 8^0 = (483)_{10}$

Conversão do Sistema Decimal para o Sistema Hexadecimal

- Novamente a conversão se faz através de divisões sucessivas pela base do sistema a ser convertido, que no caso é igual a 16. Para exemplificar, o número 1101 na base 10 será convertido para o sistema hexadecimal.



- Sendo $13_{10} = D_{16}$, tem-se que $1101_{10} = 44D_{16}$.

Conversão do Sistema Decimal para o Sistema Binário

- Exemplos
  - $(2754)_{10} = (\quad)_{16}$
    - 2754/16 = 172 :: resto_0 = 2 (algarismo menos significativo)
    - 172/16 = 10 :: resto_1 = 12
    - Como 10 < 16 então acabaram as divisões.
    - Assim temos: $(AC2)_{16}$
  - $(490)_{10} = (\quad)_{16}$
    - Resposta: $(1EA)_{16}$

Operações Aritméticas não-decimal: Base 2

- Nas áreas de Eletrônica Digital e dos Microprocessadores, o estudo das operações aritméticas no sistema binário é muito importante, pois estas serão utilizadas em circuitos aritméticos, que serão estudados posteriormente.
- Por enquanto considere:
  - números inteiros
  - sem limite de tamanho e
  - positivos (sem sinal)

Adição no Sistema Binário

A adição no sistema binário é efetuada de maneira idêntica ao sistema decimal, levando-se em conta que só há dois algarismos disponíveis (0 e 1). Desta forma, tem-se:

$$\begin{array}{r} 0 \\ +0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 0 \\ +1 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +0 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +1 \\ \hline 10 \end{array}$$

Observa-se, entretanto, a existência de uma pequena regra: 1+1=0 e transporta 1 (vai um) para a próxima coluna.

Adição no Sistema Binário

- Para exemplificar serão realizadas as seguintes adições:

$$\begin{array}{r} 1 \\ 11 \\ +10 \\ \hline 101 \end{array} \quad \text{Transporte} \qquad\qquad \begin{array}{r} 11 \\ 110 \\ +111 \\ \hline 1101 \end{array} \quad \text{Transporte}$$

- Nota-se, então que a adição é realizada coluna a coluna, considerando sempre o transporte proveniente da coluna anterior.
- Para verificar a soma basta converter os números para o sistema decimal.

$$11_2+10_2 = 101_2 \quad \text{equivalente a} \quad 3_{10}+2_{10} = 5_{10}$$

$$110_2+111_2 = 1101_2 \quad \text{equivalente a} \quad 6_{10}+7_{10} = 13_{10}$$

Adição no Sistema Binário

- Outros exemplos
  - Efetuar a soma 45_{10} e 47_{10}

```
     1
    45
+   47
---------
    92
```

```
  101111
   101101
+  101111
-----------------
  1011100
```

  - Efetuar a soma 27_{10} e 25_{10}

$= 110100_2$

Subtração no Sistema Binário

- O método de subtração é análogo a uma subtração no sistema decimal. Assim, tem-se:

$$\begin{array}{r} 0 \\ -\,0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 0 \\ -\,1 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ -\,0 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ -\,1 \\ \hline 0 \end{array}$$

- Para o caso 0-1, o resultado será igual a 1, porém haverá um transporte para a coluna seguinte que deve ser acumulado no subtraendo e, obviamente, subtraído do minuendo. Para exemplificar, tem-se:

$$\begin{array}{r} 111 \\ -\,100 \\ \hline 011 \end{array} \qquad\qquad \begin{array}{r} 1011 \\ {}^{1}\;\;\; \\ -\;\;101 \\ \hline 0110 \end{array} \quad \text{Transporte}$$

Subtração no Sistema Binário

- Outro exemplo
  - Efetuar a subtração 101101 – 100111

$$
\begin{array}{rl}
 & {}^{2}{}^{2} \\
 & 10\cancel{1}\cancel{1}01 \\
-- & 100111 \\
\hline
 & 000110 \quad \text{ou } 110_2
\end{array}
$$

  - Efetuar a subtração 100110001 - 10101101

$= 010000100_2$

Multiplicação no Sistema Binário

- Ocorre exatamente como uma multiplicação no sistema decimal. Assim sendo, tem-se:

$$0 \times 0 = 0$$
$$0 \times 1 = 0$$
$$1 \times 0 = 0$$
$$1 \times 1 = 1$$

- Enquanto que na multiplicação decimal temos uma tabela com 100 operações, do tipo:
  - 1 x 2 = 2
  - 2 x 7 = 14
  - 5 x 6 = 30
  - Etc.

Multiplicação no Sistema Binário

- Para exemplificar, efetua-se a multiplicação entre os números 11010_2 e 101_2.
- O procedimento consiste em multiplicar cada algarismo do multiplicador pelos algarismos do multiplicando.
- Isto resulta em produtos parciais, tantos quanto forem os algarismos do multiplicador
- Cada produto parcial é colocado de modo a se posicionar uma casa para a esquerda do produto anterior
- Em seguida, os três produtos são somados produzindo o resultado desejado.

```
    11010      Multiplicando
    x 101      Multiplicador
  -------
    11010      Produtos parciais
   00000+
  11010++
 --------
 10000010
```

Multiplicação no Sistema Binário

- Mais exemplos:
  - Efetuar a multiplicação 6 x 5

    $= 11110_2$

  - Efetuar a multiplicação 21 x 13

    $= 100010001_2$

  - Efetuar a multiplicação 18 x 4

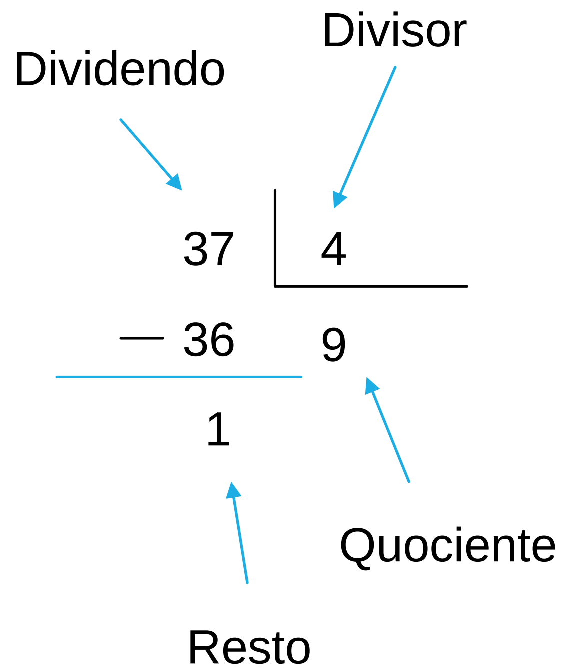
    $= 1001000_2$

Divisão no Sistema Binário

- Semelhante a divisão com números decimais
  - Deslocamentos e adições

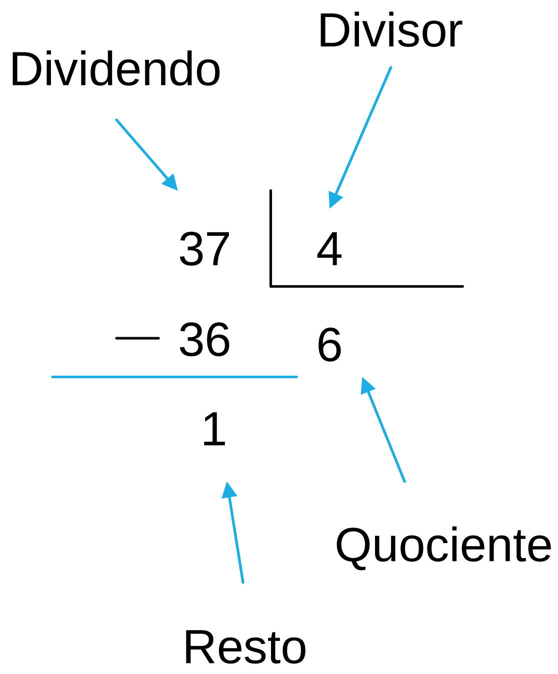
- O procedimento compreende a manipulação de quatro elementos:

- Dividendo – o valor a ser dividido
- Divisor – Valor que deve estar contido n vezes no dividendo e que, então, se deseja saber qual o valor de n
- Quociente – Quantidade de vezes que o divisor se repete no dividendo (o valor de n)
- Resto – Caso a divisão não seja exata, isto é, o divisor vezes n não seja igual ao dividendo, a diferença é chamada de resto



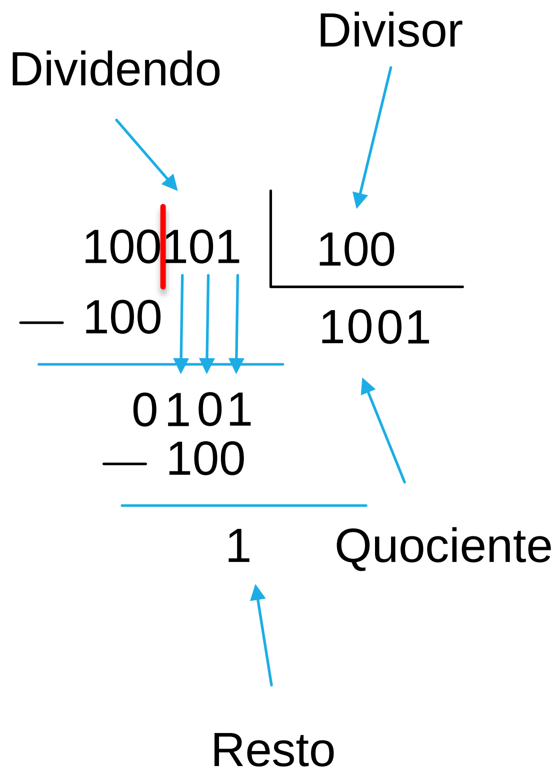
Divisão no Sistema Binário

- Procedimento decimal
  - a) verificasse quantas vezes o divisor cabe no dividendo por tentativa
  - b) busca o maior valor do quociente cuja a sua multiplicação com o divisor não seja maior que o dividendo
  - c) subtrai-se de 35 o valor resultante
  - d) O resto da divisão deve ser um valor igual, no máximo, ao divisor menos 1



Divisão no Sistema Binário

- Procedimento binário
  1) Verifica-se que valor é suficientemente maior que o divisor, de modo que o primeiro algarismo do quociente seja 1
     a) No exemplo utilizado, o valor 100 três primeiros algarismos da esquerda para a direita) é igual ao divisor
  2) Acrescenta-se ao resto algarismos do dividendo (um a um da esquerda para a direita) quantos forem necessários para que o valor obtido seja igual ou maior que o divisor
     1) A Cada algarismo selecionado e não suficiente acrescenta-se um zero ao quociente.



Divisão no Sistema Binário

- Exemplo:
  - Efetuar a divisão 101010_2 por 110_2
  - Resposta:

```
  101010 | 110
— 110    |------
  ------   111
   1001
—  110
  ------
   00110
—    110
  ------
       0
```

Exercícios

1) Converter os seguintes valores decimais em valores binários equivalentes (conversão de base 10 para base 2)
   a) 329
   b) 284
   c) 473
   d) 581
   e) 135
2) Converter os seguintes valores binários em valores decimais equivalentes (conversão de base 2 para base 10)
   a) 11011101010
   b) 11001101101
   c) 11101100010
   d) 101100011000
   e) 111001101001

Exercícios

3) Converter os seguintes valores decimais em valores hexadecimais equivalentes (conversão de base 10 para base 16)
   a) 447
   b) 544
   c) 223
   d) 622
   e) 297
4) Converter os seguintes valores hexadecimais em valores decimais equivalentes (conversão da base 16 para base 10)
   a) 3A2
   b) 33B
   c) 621
   d) 1ED4
   e) 7EF

Exercícios

5) Efetuar as seguintes somas:

a) $1100111101_2 + 101110110_2$

b) $110011110_2 + 11011111_2$

6) Efetuar a seguintes operações de subtração:

a) $11001000010_2 - 1111111111_2$

b) $10001101000_2 - 101101101_2$

7) Efetuar as seguintes conversões de base

a) $3651_{16} = (\quad)_2$

b) $26DF8_{16} = (\quad)_2$

c) $FFAB_{16} = (\quad)_2$

d) $10010_{16} = (\quad)_2$

Exercícios

8) Efetue as seguintes operações aritméticas:

a) $(101)_2 \text{ x } (111)_2 = (\quad)_2$

b) $(11101)_2 \text{ x } (1010)_2 = (\quad)_2$

c) $(11001110)_2 \ / \ (1101)_2 = (\quad)_2$

d) $(10010011)_2 \ / \ (11101)_2 = (\quad)_2$

9) Se um número binário é deslocado uma ordem para a esquerda, isto é, cada um de seus bits move-se uma posição para a esquerda e um zero é inserido na posição mais à direita, obtém-se um novo número. Qual é a relação matemática existente entre os dois números. E se for deslocado para a direita, qual é a relação?

Notação de números Binários Positivos e Negativos

- Em aplicações práticas, os números binários devem ser representados com sinal. Uma maneira de fazer isto é adicionar um bit de sinal ao número.
- Este bit é adicionado mais a esquerda do número, por convenção se for 0, o número em questão é positivo, caso seja 1, o número é negativo.
- Este processo é denominado sinal-magnitude.

Notação de números Binários Positivos e Negativos

- Vamos ver alguns exemplos:
  - Representar em binários sinal-magnitude os números 23_{10} , -15_{10} , 11_{10} e -9_{10} usando palavras de 8 bits.

Notação de números Binários Positivos e Negativos

- Vamos ver alguns exemplos:
  - Representar em binários sinal-magnitude os números 23_{10} , -15_{10} , 11_{10} e -9_{10} usando palavras de 8 bits.

    $23_{10} = 10111_2$ usando 8 bits temos: 00010111_2

Notação de números Binários Positivos e Negativos

- Vamos ver alguns exemplos:
  - Representar em binários sinal-magnitude os números 23_{10} , -15_{10} , 11_{10} e -9_{10} usando palavras de 8 bits.

    $23_{10} = 10111_2$ usando 8 bits temos: 00010111_2

    $15_{10} = 1111_2$ usando 8 bits temos: 00001111_2 como o sinal é negativo vem $-15_{10} = 10001111_2$.

Notação de números Binários Positivos e Negativos

- Vamos ver alguns exemplos:
  - Representar em binários sinal-magnitude os números 23_{10} , -15_{10} , 11_{10} e -9_{10} usando palavras de 8 bits.

    $23_{10} = 10111_2$ usando 8 bits temos: 00010111_2

    $15_{10} = 1111_2$ usando 8 bits temos: 00001111_2 como o sinal é negativo vem $-15_{10} = 10001111_2$.

    $11_{10} = 1011_2$ usando 8 bits temos: 00001011_2

Notação de números Binários Positivos e Negativos

- Vamos ver alguns exemplos:
  - Representar em binários sinal-magnitude os números 23_{10} , -15_{10} , 11_{10} e -9_{10} usando palavras de 8 bits.

$23_{10} = 10111_2$ usando 8 bits temos: 00010111_2

$15_{10} = 1111_2$ usando 8 bits temos: 00001111_2 como o sinal é negativo vem $-15_{10} = 10001111_2$.

$11_{10} = 1011_2$ usando 8 bits temos: 00001011_2

$9_{10} = 1001_2$ usando 8 bits temos: 00001001_2 , como o sinal é negativo vem $-9_{10} = 10001001_2$

Aritmética em Sinal Magnitude

- Soma
  - Se os sinais forem iguais soma e conserva o sinal da parcela de maior magnitude
  - Exemplo1:

```
     0 010
+    0 101
-------------
```

Aritmética em Sinal Magnitude

- Soma
  - Se os sinais forem iguais soma e conserva o sinal da parcela de maior magnitude
  - Exemplo1:

0 010	+2
+ 0 101	+5
-------------	--------------
0 111	+7

Aritmética em Sinal Magnitude

- Soma
  - Se os sinais forem iguais soma e conserva o sinal da parcela de maior magnitude
  - Exemplo2:

```
     1 111
+    0 011
-------------
```

Aritmética em Sinal Magnitude

- Soma
  - Se os sinais forem iguais soma e conserva o sinal da parcela de maior magnitude
  - Exemplo2:

	1 111	-7
+	0 011	+2
	-------------	--------------
	1 100	-5

Aritmética em Sinal Magnitude

- Soma
  - Se os sinais forem diferentes subtrai e conserva o sinal da parcela de maior magnitude
  - Exemplo1:

```
     0 111
+    1 011
-------------
```

Aritmética em Sinal Magnitude

- Soma
  - Se os sinais forem diferentes subtrai e conserva o sinal da parcela de maior magnitude
  - Exemplo1:

```
      0 111              +7
 +    1 011              -3
 -------------      -------------
      0 100              +4
```

Aritmética em Sinal Magnitude

- Soma
  - Se os sinais forem diferentes subtrai e conserva o sinal da parcela de maior magnitude
  - Exemplo2:

```
      1 111
+     0 011
-------------
```

Aritmética em Sinal Magnitude

- Soma
  - Se os sinais forem diferentes subtrai e conserva o sinal da parcela de maior magnitude
  - Exemplo2:

1 111	-7
+ 0 011	+2
-------------	--------------
1 100	-5

Aritmética em Sinal Magnitude

- Subtração
  - Sejam dois número binário A e B
  - A-B corresponde a A+(-B)

Aritmética em Sinal Magnitude

- Quantidade de números com sinal que podem ser representados com um número N de bits de representação:

$$-2^{\{N-1\}} + 1 \leq \ X \leq \ 2^{\{N-1\}} - 1$$

  — Assim, para N = 8, -127 ≤ X ≤ 127

Aritmética em Sinal Magnitude

- Problema da Aritmética em Sinal Magnitude:
  - Duas representações para o zero

Notação de números Binários Positivos e Negativos

- Outra forma de representação de números negativos bastante utilizada é o **complemento de 2**.
- Para obtermos o complemento de 2 de um número binário, precisamos inicialmente converter o número em seu complemento de 1.
- O complemento de 1 de um número binário obtém-se trocando cada bit pelo seu complemento (0→1 e 1 →0).
- A seguir, soma-se 1 ao complemento de 1, obtendo assim o complemento de 2.

Notação de números Binários Positivos e Negativos

- Vamos exemplificar obtendo os complementos de 2 dos números binários abaixo:

binário	compl. de 1	compl. de 2
10001001		

Notação de números Binários Positivos e Negativos

- Vamos exemplificar obtendo os complementos de 2 dos números binários abaixo:

binário	compl. de 1	compl. de 2
10001001	01110110	01110111

Notação de números Binários Positivos e Negativos

- Vamos exemplificar obtendo os complementos de 2 dos números binários abaixo:

binário	compl. de 1	compl. de 2
10001001	01110110	01110111
00111100		

Notação de números Binários Positivos e Negativos

- Vamos exemplificar obtendo os complementos de 2 dos números binários abaixo:

binário	compl. de 1	compl. de 2
10001001	01110110	01110111
00111100	11000011	11000100

Notação de números Binários Positivos e Negativos

- Vamos exemplificar obtendo os complementos de 2 dos números binários abaixo:

binário	compl. de 1	compl. de 2
10001001	01110110	01110111
00111100	11000011	11000100
10011111		

Notação de números Binários Positivos e Negativos

- Vamos exemplificar obtendo os complementos de 2 dos números binários abaixo:

binário	compl. de 1	compl. de 2
10001001	01110110	01110111
00111100	11000011	11000100
10011111	01100000	01100001

Notação de números Binários Positivos e Negativos

- Vamos exemplificar obtendo os complementos de 2 dos números binários abaixo:

binário	compl. de 1	compl. de 2
10001001	01110110	01110111
00111100	11000011	11000100
10011111	01100000	01100001
11000101		

Notação de números Binários Positivos e Negativos

- Vamos exemplificar obtendo os complementos de 2 dos números binários abaixo:

binário	compl. de 1	compl. de 2
10001001	01110110	01110111
00111100	11000011	11000100
10011111	01100000	01100001
11000101	00111010	00111011

Notação de números Binários Positivos e Negativos

- Vamos exemplificar obtendo os complementos de 2 dos números binários abaixo:

binário	compl. de 1	compl. de 2
10001001	01110110	01110111
00111100	11000011	11000100
10011111	01100000	01100001
11000101	00111010	00111011
01101011		

Notação de números Binários Positivos e Negativos

- Vamos exemplificar obtendo os complementos de 2 dos números binários abaixo:

binário	compl. de 1	compl. de 2
10001001	01110110	01110111
00111100	11000011	11000100
10011111	01100000	01100001
11000101	00111010	00111011
01101011	10010100	10010101

Notação de números Binários Positivos e Negativos

- Devemos observar que devido ao seu emprego em hardware os números binários são representados sempre com um número fixo de bits.
- A conversão inversa, ou seja, de um número em representação complemento de 2 para a notação binária original é feita obtendo-se novamente o seu complemento de 2.

Notação de números Binários Positivos e Negativos

- Valor em decimal de um número com sinal

- $01010110_2 = +86_{10}$
  - $2^6 + 2^4 + 2^2 + 2^1 = 64 + 16 + 4 + 2 = 86_{10}$
- $10101010_2 = -86_{10}$
  - $-2^7 + 2^5 + 2^3 + 2^1 = -128 + 32 + 8 + 2 = -86_{10}$

Notação de números Binários Positivos e Negativos

- Utilização do complemento de 2 em operações aritméticas.
- Podemos utilizar a notação complemento de 2 para efetuar operações de soma (e subtração).
- Para efetuar operações envolvendo números negativos usamos seu complemento de 2

Notação de números Binários Positivos e Negativos

- Por exemplo:
  - Efetuar 11010111_2-00100101_2

  obtemos o complemento de 2 de 00100101 temos 11011011

Notação de números Binários Positivos e Negativos

a seguir efetuamos a soma 11010111 + 11011011

$$
\begin{array}{r}
11010111 \\
+\underline{11011011} \\
110110010
\end{array}
$$

Notação de números Binários Positivos e Negativos

- Outro exemplo:
  - Efetuar 001101_2-010101_2 $(13\text{-}21)_{10}$ usando notação de complemento de 2

Notação de números Binários Positivos e Negativos

- Outro exemplo:
  - Efetuar 001101_2-010101_2 $(13\text{-}21)_{10}$ usando notação de complemento de 2

O complemento de 2 de 010101 é 101011 (confere?), agora temos

Notação de números Binários Positivos e Negativos

```
  001101
 +101011
  111000
```

O resultado foi 56 ?? O que deu errado?

- Nada! Como o subtraendo é o maior, o resultado é um número negativo e portanto já está representado em complemento de 2.
- Para obtermos o módulo do resultado, basta obter novamente o complemento de 2, assim
- 11000 → 1000, ou seja, trata-se de –8.

Complemento de 2

1000	−8
1001	−7
1010	−6
1011	−5
1100	−4
1101	−3
1110	−2
1111	−1
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

Comparação das representações

Decimal	Sinal e magnitude	Complemento a 2
−16	—	10000
−15	11111	10001
−14	11110	10010
−13	11101	10011
−12	11100	10100
−11	11011	10101
−10	11010	10110
−4	10100	11100
−3	10011	11101
−2	10010	11110
−1	10001	11111
−0	10000	—
+0	00000	00000
+1	00001	00001
+2	00010	00010
+3	00011	00011
+4	00100	00100
+10	01010	01010
+11	01011	01011
+12	01100	01100
+13	01101	01101
+14	01110	01110
+15	01111	01111

Comparação das representações

Tipo de Representação	Dupla representação para o zero	Custo	Velocidade
Sinal e magnitude	SIM (desvantagem)	Alto (componentes separados para soma e subtração)	Baixa (algoritmo de verificação de sinais, soma e subtração)
Complemento a 2	Não (vantagem)	Baixo (um componente único para soma e subtração)	Alta (algoritmo simples e igual para soma e subtração)

Overflow

- Ocorre sempre que o resultado de uma operação não pode ser representado no hardware disponível

Operação	Operando A	Operando B	Resultado
A+B	>= 0	>=0	<0
A+B	<0	<0	>=0
A-B	>=0	<0	<0
A-B	<0	>=0	>=0

- Se um número for negativo, e o outro positivo, não ocorrerá overflow.

Overflow

- Outra forma de verificar a ocorrência de overflow
  - Some os dois números e observe se ocorre carry (vai 1) sobre o bit de sinal e se ocorreu carry após o bit de sinal.
  - Se ocorreu um e somente um dos dois carrys houve estouro (resultado errado), caso contrário a soma está correta.

$(40_{10}) + (-50_{10}) = -10_{10}$

$40_{10} = 00101000_2$

$50_{10} = 00110010_2 \implies -50_{10} = 11001110_2$

$$\begin{array}{r} 00101000_2 \\ +\underline{11001110_2} \\ 11110110 \end{array}$$

$11110110 = -2^7 + 2^6 + 2^5 + 2^4 + 2^2 + 2^1 = -10$ (correto)

Overflow

- Soma (carry sobre bit de sinal)

$(5_{10}) + (6_{10}) = 11_{10}$

$5_{10} = 0101_2$

$6_{10} = 0110_2$

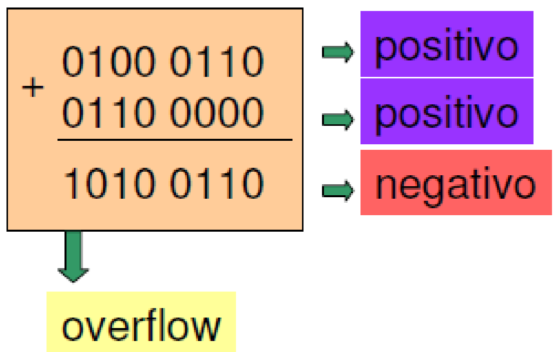
$$
\begin{array}{r}
1 \\
0101_2 \\
\underline{+0110_2} \\
1011
\end{array}
$$

1011 => carry sobre bit de sinal (estouro = overflow)

$-2^3 + 2^1 + 2^0 = -5$ (resultado errado)

Overflow

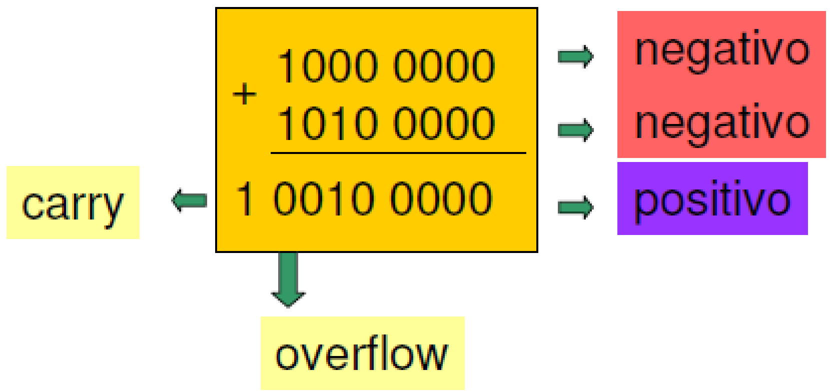
- Exemplos de overflow



- Isto significa que o resultado está correto se o bit de sinal for ignorado

Overflow

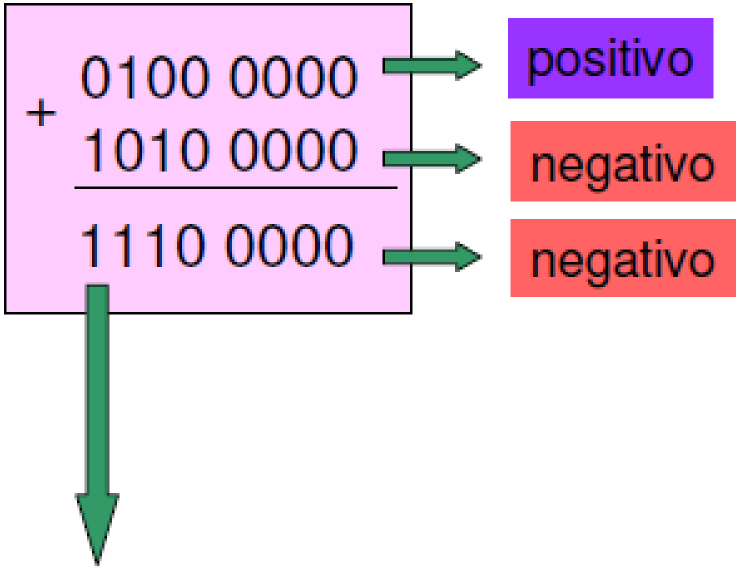
- Exemplos de overflow



- Isto significa que o resultado é negativo e está em complemento a 2

Overflow

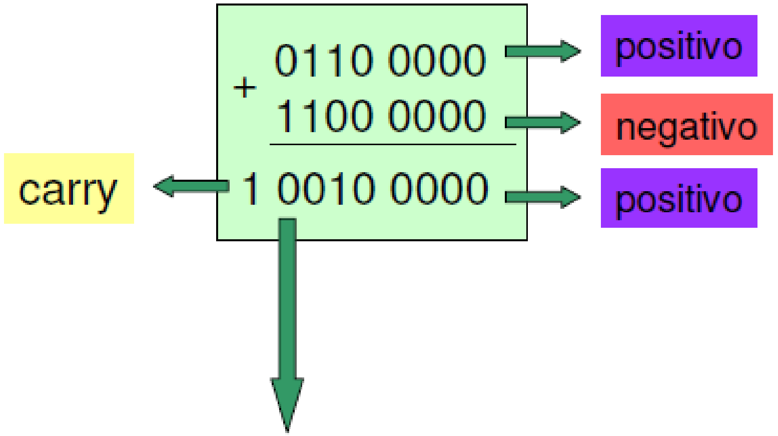
- Exemplos de overflow



- Não ocorre overflow, o resultado é negativo e está em complemento a 2

Overflow

- Exemplos de overflow



- Não ocorre overflow, o carry é ignorado e o resultado é positivo

Complemento de 2

- Exercícios

- Efetue as operações binárias

  a) 10001+1111 b) 1110+1001011 c) 1011+ 11100

  d) 110101+1011001+1111110 e) 1100+1001011+11101

  f) 10101-1110 g) 100000-11100 h) 1011001-11011

  i) 11001x101 j) 11110x110 k) 11110x111

- Represente os números em notação sinal-módulo 8bits

  a) 97 b) -121 c) 79 d) -101

- Represente os números do exercício anterior em complemento de 2.
- Efetue as operações utilizando complemento de 2.

  a) 111100-11101011 b) 101101-100111 c) 75_8-30_8

Números Fracionários

- Discutiram-se, até o momento, as diversas formas de conversão de números **inteiros, pertencentes a um dado sistema, em outro.**

- **Neste tópico, serão mostrados os procedimentos para converter números fracionários.**

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 $10{,}5_{10}$.

$$10{,}5_{10} = 1 \text{ x } 10^{1} + 0 \text{ x } 10^{0} + 5 \text{ x } 10^{-1}$$

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

$$101{,}101_{2} = 1 \text{ x } 2^{2} + 0 \text{ x } 2^{1} + 1 \text{ x } 2^{0} + 1 \text{ x } 2^{-1} + 0 \text{ x } 2^{-2} + 1 \text{ x } 2^{-3}$$

$$101{,}101_{2} = 1\text{x}4 + 0\text{x}2 + 1\text{x}1 + 1\text{x}\frac{1}{2} + 0\text{x}\frac{1}{4} + 1\text{x}\frac{1}{8}$$

$$101{,}101_{2} = 4 + 1 + 0{,}5 + 0{,}125 = 5{,}625_{10}$$

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.

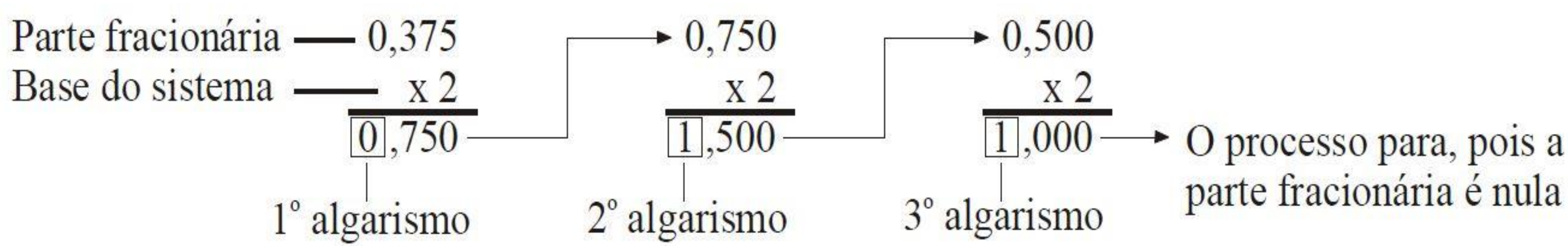
$$8{,}375 = 8 + 0{,}375$$

- Parte inteira:

$$\begin{array}{llllll}
 & 8 & \underline{|2} & & & \\
\text{LSB} — & 0 & 4 & \underline{|2} & & \\
 & & 0 & 2 & \underline{|2} & \\
 & & & 0 & 1 & — \text{MSB}
\end{array}
\qquad 8_{10} = 1000_2$$

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 interia com a fracionária:

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

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:

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

Sistema de Numeração Binário

- **Bits e Bytes**
  - A menor unidade de informação usada pelo computador é o *bit*. Este tem atribuições lógicas **0** ou **1**.
  - Cada um destes estados pode, internamente, ser representado por meios eletro-magnéticos (negativo/positivo, ligado/desligado, etc).
  - É por isso que é mais fácil para armazenar dados em formato binário. Assim, todos os dados do computador são representados de forma binária.
  - Mesmo os números são comumente representados na base 2, em vez da base 10, e suas operações são feitas na base 2.

Sistema de Numeração Binário

- Um conjunto de 8 bits é chamado de *byte* e pode ter até $2^8 = 256$ configurações diferentes.

- As seguintes denominações são comumente usadas na área de informática

nome	*memória*
bit	$\{0,1\}$
byte	8 bits
kilobyte (kbyte)	2^{10} bytes (pouco mais de mil bytes ($2^{10} = 1024$))
megabyte	2^{20} bytes (pouco mais de um milhão de bytes)
gigabyte	2^{30} bytes (pouco mais de um bilhão de bytes)

O código binário e o correspondente valor decimal de alguns caracteres no padrão ASCII:

O principal padrão usado para Representar caracteres *(’a’,’b’,’c’,...,’A’,’B’,’C’,...,’!’,’@’,’#’,’$’,...) é o padrão ASCII (American Standard Code for Information Interchange), usado na maioria dos computadores.*

Cada um destes caracteres é representado por um byte.

Caracter	Representação em ASCII	Valor na base decimal
⋮	⋮	⋮
	00100000	32
!	00100001	33
”	00100010	34
#	00100011	35
$	00100100	36
⋮	⋮	⋮
0	00110000	48
1	00110001	49
2	00110010	50
3	00110011	51
⋮	⋮	⋮
A	01000001	65
B	01000010	66
C	01000011	67
D	01000100	68
⋮	⋮	⋮
a	01100001	97
b	01100010	98
c	01100011	99
d	01100100	100
⋮	⋮	⋮

Tabela ASCII

- Observe que:

1. As codificações para letras em maiúsculas e minúsculas são diferentes.

2. A codificação de ’B’ é a codificação de ’A’ somado de 1; a codificação de ’C’ é a codificação de ’B’ somado de 1; assim por diante.

Esta codificação permite poder comparar facilmente se um caráter vem antes do outro ou não.

Algebra Booleana

- Em 1854 o matemático inglês George Boole apresentou um sistema matemático de análise lógica conhecido como **álgebra de Boole.**

- Somente em 1938, um engenheiro americano utilizou as teorias da álgebra de Boole para a solução de problemas de circuitos de telefonia com relés, tendo publicado um artigo que praticamente introduziu na área tecnológica o campo da eletrônica digital.

Álgebra Booleana

- Os sistemas digitais são formados por circuitos lógicos denominados de portas lógicas que, utilizados de forma conveniente, podem implementar todas as expressões geradas pela álgebra de Boole.

- Em muitas aplicações é necessário processar bits isolados dentro de uma palavra -> operações lógicas

- Existem três portas básicas (**E, OU e NÃO) que podem ser conectadas de várias maneiras, formando sistemas que vão de simples relógios digitais aos computadores de grande porte.**

Função E ou AND

- Para compreender a função **E da álgebra Booleana, deve-se analisar o circuito da Fig. 2.1, para o qual se adota as seguintes convenções:**
- **chave aberta=0, chave fechada=1,**
- **lâmpada apagada=0 e lâmpada acesa=1.**

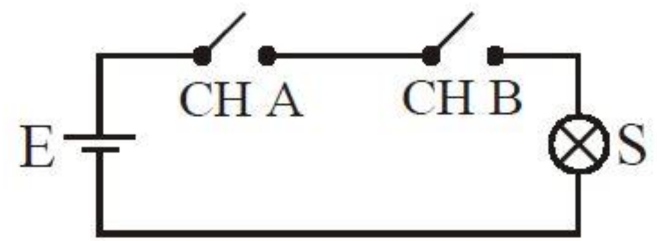


*Figura 2.1 – Circuito representativo da função **E**.*

A análise da Fig. 2.1 revela que a lâmpada somente acenderá se ambas as chaves estiverem fechadas e, seguindo a convenção, tem-se: CH A=1, CH B=1, resulta em S=1.

Função E ou AND

- Pode-se, desta forma, escrever todas as possíveis combinações de operação das chaves na chamada **Tabela da Verdade, que é definida como um mapa onde se depositam todas as possíveis situações com seus respectivos resultados. O número de combinações possíveis é igual a 2^N, onde N é o número de variáveis de entrada.**

*Tabela da verdade da função **E**.*

A	B	S
0	0	0
0	1	0
1	0	0
1	1	1

Função E ou AND

- A porta lógica E é um circuito que executa a função E da álgebra de Boole, sendo representada, na prática, através do símbolo visto na Fig. 2.2.

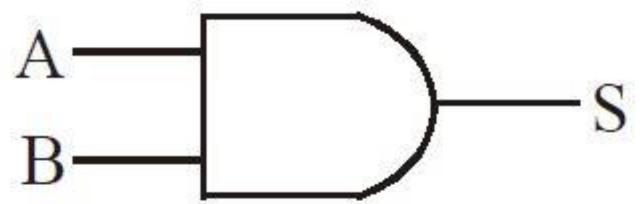


*Figura 2.2 – Porta lógica **E**.*

- “A saída da porta E será 1, somente se todas as entradas forem 1”.

Função OU ou OR

- A função OU é aquela que assume valor 1 quando uma ou mais variáveis de entrada forem iguais a 1 e assume 0 se, e somente se, todas as variáveis de entrada forem iguais a zero. Sua representação algébrica para duas variáveis de entrada é S=A+B, onde se lê: S=A ou B.

Função OU ou OR

- O circuito abaixo mostra que a lâmpada acende quando qualquer uma das chaves estiver fechada e permanece apagada se ambas estiverem abertas, ou seja, CH A=0, CH B=0, resulta em S=0.

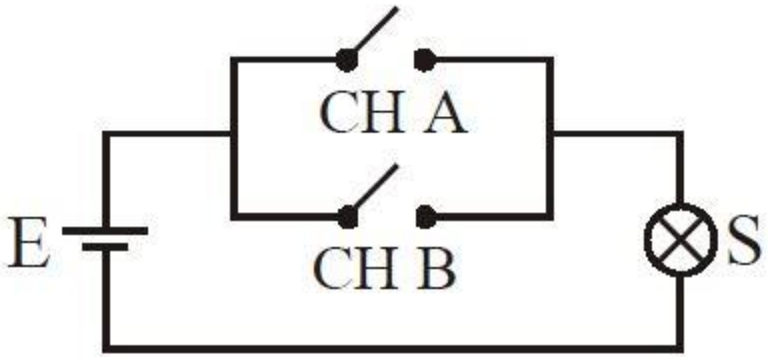
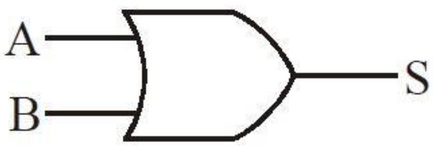


Figura 2.3 – Circuito que representa a função ***OU****.*

Função OU ou OR

- A Fig. 2.4 ilustra a porta lógica que executa a função **OU da álgebra de Boole, juntamente com a sua tabela da verdade.**



*Porta lógica **OU***

A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

*Tabela da verdade da função **OU***

*Figura 2.4 – Porta lógica e tabela da verdade da função **OU**.*

- **“A saída de uma porta OU será 1 se uma ou mais entradas forem 1”.**

Função NÃO ou NOT

- A função **NÃO é aquela que inverte ou complementa o estado da variável de entrada, ou seja, se a variável estiver em 0, a saída vai para 1, e se estiver em 1 a saída vai para 0.**

- **É representada algebricamente da seguinte forma:, onde se lê: A barra ou NÃO A.**

2.4 Função NÃO ou NOT

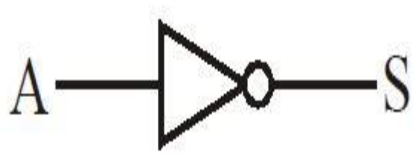
- A análise do circuito da Fig. 2.5 ajuda a compreender melhor a função **NÃO da álgebra Booleana. Será utilizada a mesma convenção dos casos anteriores.**

*Figura 2.5 – Circuito representativo da função **NÃO**.*

- Observando o circuito da Fig. 2.5, pode-se concluir que a lâmpada estará acesa somente se a chave estiver aberta
- (CH A=0, S=1), quando a chave fecha, a corrente desvia por ela e a Lâmpada apaga (CH A=1, S=0).

2.4 Função NÃO ou NOT

- O inversor é o bloco lógico que executa a função **NÃO. Sua representação simbólica é vista na Figura juntamente com sua tabela da verdade.**



Porta lógica ***NÃO*** *ou* ***inversora***

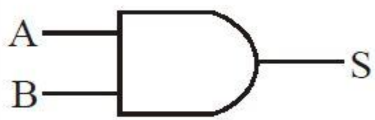
A	S
0	1
1	0

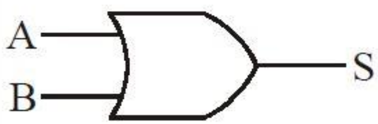
Tabela da verdade da função ***NÃO***

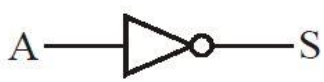
- **"A saída de uma porta NÃO assume o nível lógico 1**
- **somente quando sua entrada é 0 e vice-versa".**

Blocos Lógicos Basicos

<table>
<tr><th colspan="5">BLOCOS LÓGICOS BÁSICOS</th></tr>
<tr><th>PORTA</th><th>Símbolo Usual</th><th>Tabela da Verdade</th><th>Função Lógica</th><th>Expressão</th></tr>
<tr><td>E

AND</td><td>
</td><td><table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td><td>Função E: Assume 1 quando todas as variáveis forem 1 e 0 nos outros casos.</td><td>S=A.B</td></tr>
<tr><td>OU

OR</td><td>
</td><td><table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td><td>Função E: Assume 0 quando todas as variáveis forem 0 e 1 nos outros casos.</td><td>S=A+B</td></tr>
<tr><td>NÃO

NOT</td><td>
</td><td><table><tr><th>A</th><th>S</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table></td><td>Função NÃO: Inverte a variável aplicada à sua entrada.</td><td>$S=\overline{A}$</td></tr>
</table>

Exercícios para serem feitos do livro base

1) Converter os seguintes valores decimais em valores binários equivalentes (conversão de base 10 para base 2):

a) 329
b) 284
c) 473
d) 69
e) 135
f) 215
g) 581
h) 197

2) Converter os seguintes valores binários em valores decimais equivalentes (conversão de base 2 para base 10):

a) 11011101010
b) 11001101101
c) 10000001111
d) 11101100010
e) 111001101001
f) 111111000011
g) 101100011000
h) 100000000110

4) Converter os seguintes valores decimais em valores binários equivalentes (conversão de base 10 para base 2):

a) 417
b) 113
c) 819
d) 77
e) 251
f) 769
g) 180
h) 27

Exercícios para serem feitos do livro base

8) Converter os seguintes valores decimais em valores hexadecimais equivalentes (conversão de base 10 para base 16):

a) 447
b) 544
c) 223
d) 71
e) 622
f) 97
g) 121
h) 297

9) Converter os seguintes valores hexadecimais em valores decimais equivalentes (conversão de base 16 para base 10):

a) 3A2
b) 33B
c) 621
d) 99
e) 1ED4
f) 7EF
g) 22C
h) 110A

Exercícios para serem feitos do livro base

14) Efetuar as seguintes somas:

a) $31752_8 + 6735_8 =$

b) $37742_8 + 26573_8 =$

c) $2A5BEF_{16} + 9C829_{16} =$

d) $356_7 + 442_7 =$

e) $1100111101_2 + 101110110_2 =$

f) $211312_4 + 121313_4 =$

g) $3645_8 + 2764_8 =$

h) $110011110_2 + 11011111_2 =$

15) Efetuar as seguintes operações de subtração:

a) $64B2E_{16} - 27EBA_{16} =$

b) $2351_8 - 1763_8 =$

c) $543_6 - 455_6 =$

d) $43321_5 - 2344_5 =$

e) $11001000010_2 - 1111111111_2 =$

f) $10001101000_2 - 101101101_2 =$

g) $43DAB_{16} - 3EFFA_{16} =$

h) $100010_2 - 11101_2 =$

Exercícios para serem feitos do livro base

19) Quantos números inteiros positivos podem ser representados em uma base B, cada um com *n* algarismos significativos?

20) A partir do valor binário 110011, escreva os cinco números que se seguem em seqüência.

21) A partir do valor binário 101101, escreva seis números, saltando de três em três números, de forma crescente.

22) A partir do valor octal 1365, escreva os oito números que se seguem em seqüência.

23) A partir do valor octal 3745, escreva os oito números pares seguintes.

24) A partir do valor hexadecimal 2BEF9, escreva os 12 números que se seguem em seqüência.

25) A partir do valor hexadecimal 3A57, escreva os 10 números subseqüentes, saltando de quatro em quatro valores (por exemplo, o primeiro subseqüente é 3A5B).

26) A maioria das pessoas só pode contar até 10 utilizando seus dedos. Entretanto, quem trabalha com computador pode fazer melhor. Se você imaginar cada um dos seus dedos como um dígito binário, convencionando que o dedo estendido significa o algarismo 1 e o recolhido significa 0, até quanto você poderá contar usando as duas mãos?

Exercícios para serem feitos do livro base

42) Efetue as seguintes operações aritméticas:

a) $(101)_2 \times (111)_2 = (\quad)_2$

b) $(11101)_2 \times (1010)_2 = (\quad)_2$

c) $(11001110)_2 / (1101)_2 = (\quad)_2$

d) $(111110001)_2 \times (10011)_2 = (\quad)_2$

e) $(100100011)_2 / (11101)_2 = (\quad)_2$

f) $(1101101)_2 / (100)_2 = (\quad)_2$

g) $(111000001) \times (101001)_2 = (\quad)_2$