Arquitetura e Organização de Computadores

Aula 04: Continuação: Conversão de Bases e Aritmética Computacional

Adição no Sistema Binário

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

```
   1                 101111
   45                 101101
+  47              +  101111
-----              ----------
   92                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:

$$\begin{array}{r} 111 \\ -\,100 \\ \hline 011 \end{array}$$

$$\begin{array}{r} 1011 \\ {\scriptstyle 1}\blacktriangleleft \\ -\,101 \\ \hline 0110 \end{array}$$ Transporte

Subtração no Sistema Binário

- Outro exemplo
  - Efetuar a subtração 101101 – 100111

```
          2 2
      101101
  --  100111
-----------------------------
      0 0 0 1 1 0        ou 110₂
```

  - 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

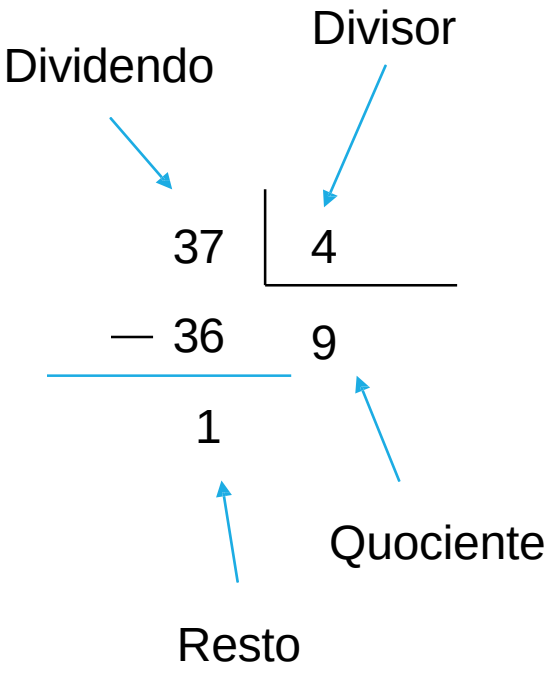
    = 1001000_2

Divisão no Sistema Binário

- Semelhante a divisão com números decimais
  - Deslocamentos e adições

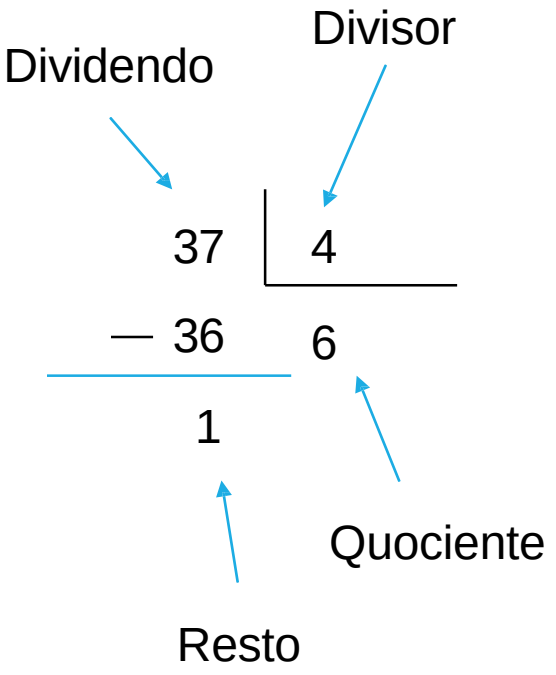
- O procedimento compreende a manipulação de quatro elementos:

- 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



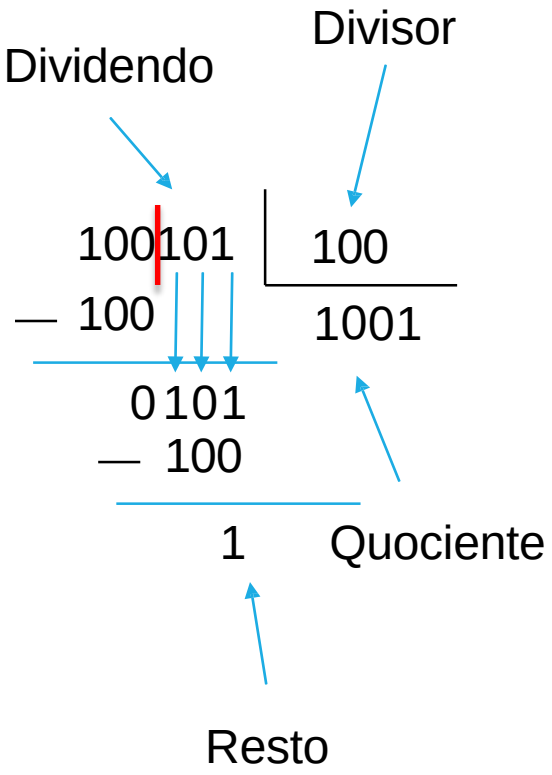
Divisão no Sistema Binário

- 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) Efetuar a seguintes operações de subtração:

a) $11001000010_2 - 1111111111_2$

b) $10001101000_2 - 101101101_2$

2) 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) $(10010011)_2 / (11101)_2 = (\quad)_2$

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.

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

$40_{10} = 00101000_2$

$50_{10} = 00110010_2$ ==> $- 50_{10} = 11001110_2$

00101000_2

$+11001110_2$

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

Overflow

- Soma (carry sobre bit de sinal)

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

$5_{10} = 0101_2$

$6_{10} = 0110_2$

1

0101_2

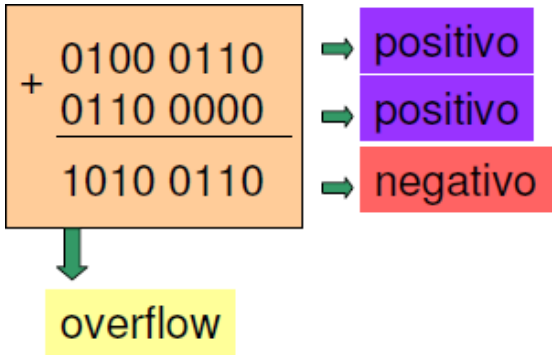
$+0110_2$

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

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

Overflow

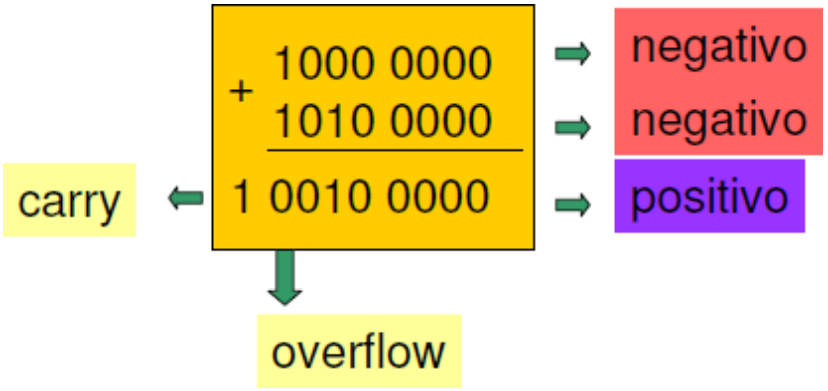
- Exemplos de overflow



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

Overflow

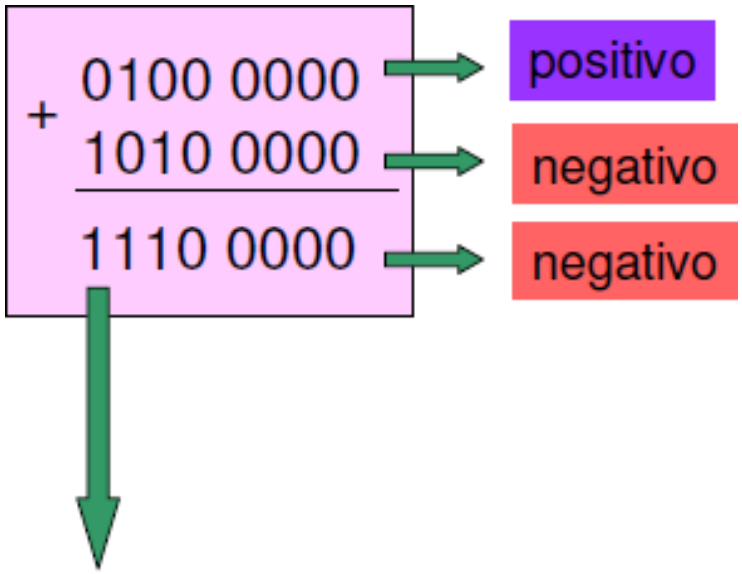
- Exemplos de overflow



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

Overflow

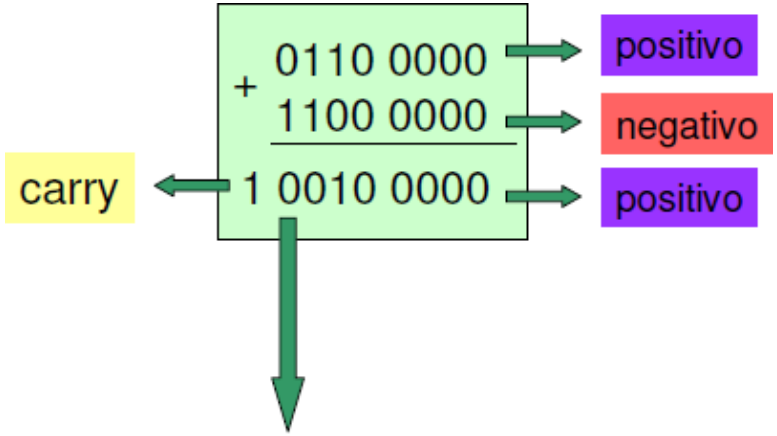
- Exemplos de overflow



- 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 $10,5_{10}$.

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

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

$$101,101_2 = 1 \times 2^{2} + 0 \times 2^{1} + 1 \times 2^{0} + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$$

$$101,101_2 = 1x4 + 0x2 + 1x1 + 1x\frac{1}{2} + 0x\frac{1}{4} + 1x\frac{1}{8}$$

$$101,101_2 = 4 + 1 + 0,5 + 0,125 = 5,625_{10}$$

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.

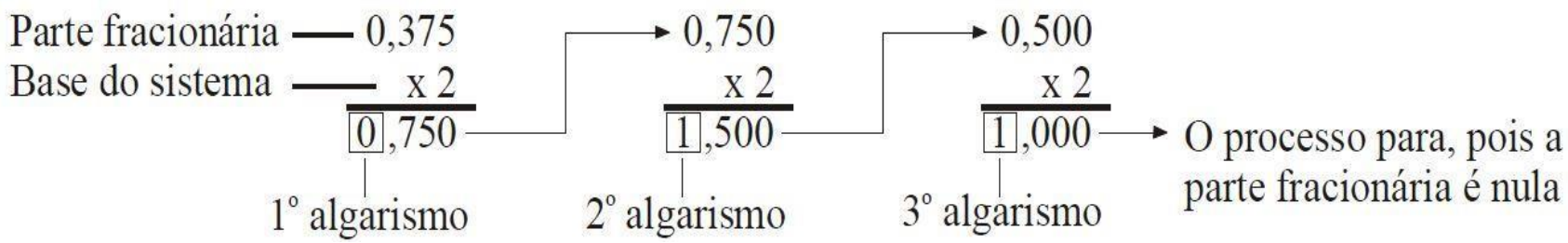
$$8{,}375 = 8 + 0{,}375$$

- Parte inteira:

$$\begin{array}{rl} & 8 \mid 2 \\ \text{LSB} - \textcircled{0} & 4 \mid 2 \\ & \textcircled{0}\ 2 \mid 2 \\ & \quad \textcircled{0}\ \textcircled{1} - \text{MSB} \end{array} \qquad 8_{10} = 1000_2$$

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.

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

42) Efetue as seguintes operações aritméticas:

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

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

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

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

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

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

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