Organização e Arquitetura de Computadores

Educação a Distância
Cruzeiro do Sul Educacional
Campus Virtual

Material teórico

Sistemas de Numeração

Responsável pelo Conteúdo:
Prof. Ms. Vagner Silva

Revisão Textual:
Profª. Esp. Vera Lidia de Sa Cicaroni

UNIDADE Sistemas de Numeração

Os sistemas de numeração são utilizados frequentemente por nós, seres humanos, e pelas máquinas. Os seres humanos têm mais facilidade em manipular o sistema de numeração decimal, porém as máquinas manipulam melhor o sistema de numeração binário. Há ainda os sistemas de numeração octal e hexadecimal.

Nesta unidade você vai aprender a realizar conversões entre bases de sistemas de numeração e conhecer como são feitos os cálculos aritméticos com o sistema de numeração binário. Leia atentamente o conteúdo disponibilizado, pois seu entendimento irá facilitar o desenvolvimento das atividades.

Atenção

Para um bom aproveitamento do curso, leia o material teórico atentamente antes de realizar as atividades. É importante também respeitar os prazos estabelecidos no cronograma.

Contextualização

Abaixo você irá encontrar parte de uma entrevista realizada com Renato Degiovani. Esta entrevista está disponível em: http://meiobit.com/17470/entrevista-renato-degiovani/ Acessada em 01.05.2012.

Cenário: 1983. No exterior surgiam o Apple Lisa, o Apple IIe, o Atari 1200XL. A Lotus lançava o 1-2-3, a IBM anunciava o PC-XT e a Microsoft mostrava o DOS 2.0. Por aqui, reserva de mercado. As novidades eram o CP300, da Prológica; o TK85 da Microdigital e falava-se que a Gradiente fabricaria um micro compatível com o poderoso padrão MSX.

A revista de informática mais influente do mercado era a "Micro Sistemas", que trazia, na sua edição de julho, um marco na história do desenvolvimento de jogos no Brasil: a listagem (é, naquela época se digitavam os programas que vinham nas revistas) do "Aeroporto 83". Sucesso imediato, a edição esgotou rapidamente, deixando os leitores ávidos por mais e mais jogos. O responsável pelo feito: Renato Degiovani.

MB: O senhor já disse que seu primeiro micro foi um NE-Z80. Portanto, o senhor teve que aprender linguagem de máquina: assembly. Como foi a experiência, para alguém que não veio de uma formação técnica na área? E qual linguagem utiliza hoje?

RD: Aprender linguagem de máquina foi uma das minhas maiores conquistas, porque, naquela época (inicio dos anos 80), não existiam livros sobre o assunto, não existiam manuais e não tinha nem a quem perguntar, ou tirar uma dúvida. Porém eu já tinha assimilado (via basic) a principal regra da programação: o computador vai executar o que for programado, em sequência e instrução por instrução.

Eu olhava para os códigos em LM e ficava me perguntando onde estava o sentido daquilo. As poucas referências à programação eram do tipo "com linguagem de máquina você literalmente conversa com o processador". Ora, como tirar de frases como essa o sentido da programação e dos códigos? Para mim era angustiante não compreender uma simples listagem. Só quando caiu a ficha e entendi que LD A,10 significava "load register A with 10" e que isso significava algo parecido com "LET A = 10" do basic, é que tudo passou a fazer sentido. Dai em diante não teve mais nenhum mistério.

Programei em linguagem de máquina (escrevendo os códigos hexa diretamente), depois em assembler (porque a compilação era feita manualmente e não pelo computador) e finalmente em assembly (usando o computador para compilar o código) até 96/97. Fazia tudo em asm. Daí em diante passei a usar Delphi e nunca mais consegui "gostar" de outra linguagem de programação.

Atualmente, ainda se utiliza a linguagem de máquina em várias situações, no entanto há emuladores, por exemplo, em linguagem C que convertem o programa em zeros e uns.

Sistemas de numeração

Em sua evolução, após deixar de ser nômade, o homem sentiu necessidade de contabilizar. Devido a essa necessidade, foi desenvolvido o sistema de numeração que utilizamos atualmente na representação de quantidades. Existem vários sistemas numéricos, dentre os quais se destacam o sistema decimal, o qual já estamos acostumados a manipular, o binário, o octal e o hexadecimal.

Os sistemas binário, octal e hexadecimal são importantes na área de técnicas digitais e computação, pois são usados na representação dos sinais digitais, na programação em linguagem de baixo nível, na representação de endereçamento de memória, nos dispositivos de entrada e saída e no processamento dos microprocessadores.

O sistema binário de numeração é um sistema que possui apenas dois símbolos: o símbolo 1 e o símbolo zero. Podemos representar qualquer número decimal em binário usando apenas esses dois símbolos. Para representar o número zero, em binário, usamos o algarismo 0 e para representar o número 1, em binário, usamos o algarismo 1. Pois bem, se temos apenas esses dois algarismos, como poderíamos representar o algarismo 2 em binário? Você já parou para pensar nisso? Não possuímos o símbolo 2 nesse sistema de numeração; como faríamos, então, já que em binário só temos dois símbolos (0 e 1) ?

Não é tão complicado quanto parece. No sistema decimal, nós não possuímos o algarismo dez e nós representamos a quantidade de uma dezena utilizando-nos do algarismo 1 seguido do algarismo zero, ou seja, passamos a repetir os algarismos que já existem nesse sistema de numeração.

No sistema binário, com um bit, conseguimos representar as duas combinações (o 0 ou o 1) e, para representar outros valores maiores que 0 e 1, usamos a mesma regra feita para o sistema decimal. Portanto, para representar outros valores, temos que começar a agrupar outros bits.

A unidade central de processamento, mais especificamente a CPU, trabalha apenas com zeros e uns, ou seja, trabalha com sistema binário para processar as informações armazenadas na memória. Essa forma de trabalhar aumenta a capacidade de processamento e, consequentemente, aumenta a velocidade do processamento das informações. As linguagens de baixo nível, linguagem de máquina e assembler são linguagens em que as representações das instruções são feitas por números binários e hexadecimais.

O número hexadecimal tem as seguintes características:

- são muito compactos;
- são de fácil conversão para o sistema binário, ou seja, é fácil a conversão de hexadecimal para binário e de binário para hexadecimal.

Em computação, o sistema de numeração hexadecimal é usado em várias situações, como, por exemplo, quando calculamos a posição da cabeça leitora e os setores de um disco rígido ou, então, quando usamos programas de editor do disco rígido para analisar algumas características diferentes ou algum problema ocorrido. Para esses casos necessitaremos de um bom conhecimento do sistema hexadecimal para ajudar na análise.

O sistema de numeração hexadecimal usa a base 16 e inclui somente os símbolos de 0 a 9 e as letras A, B, C, D, E, e F. Usamos a letra H depois do número para denotar que ele está no sistema hexadecimal.

Método genérico de transformação de números.

A conversão entre bases é possível, usando-se o método de decomposição dos números, portanto, para converter de uma base X para uma base Y, separamos cada número em unidade, dezena, centena etc. e, então, multiplicamos pela base em que o número se encontra elevada à sua representatividade (unidade, dezena, centena etc.), lembrando que o expoente para a unidade será 0, para dezena será 1 e assim por diante.

Exemplo:

Decompor o decimal 6434 .

6434 : 10 = 643 , fica **4** 10^0
643 : 10 = 64 , fica **3** 10^1
64 : 10 = 6 , fica **4** 10^2
6 : 10 = 0 , fica **6** 10^3

Perceba que, a cada vez que vamos dividindo pela base, neste caso a base 10, o resto é multiplicado pela base elevada à sua representatividade. Vamos analisar a primeira divisão: (6434 : 10). Tivemos como resultado o número 643 e como resto ficou o número 4. O resto da divisão (4) representamos da seguinte forma: $4x10^{0.}$ Sabemos que qualquer número elevado a 0 tem como valor o número 1, então, como resultado, temos 4x1 = 4. E assim foi feito com as outras divisões até decompormos o número 6434. Como resultado da decomposição temos $6x10^3 + 4x10^2 + 3x10^1 + 4x10^0$.

Para número real usamos a multiplicação para realizar a decomposição. Vejamos um exemplo abaixo.

Decompor o número 0,8125 (decimal)

0,8125	* 10 =	**8**, 125	10^{-1}
0,125	* 10 =	**1**, 25	10^{-2}
0,25	* 10 =	**2**, 5	10^{-3}
0,5	* 10 =	**5**, 0	10^{-4}
0,0			

Conforme pode ser visto acima, na decomposição de números reais, devemos multiplicar pela base. Neste caso estamos na base 10. O valor inteiro multiplicamos pela base, neste caso 10, elevada à sua representatividade negativa. Na primeira parte da decomposição, fizemos 0,8125 * 10; como resultado, tivemos 8,125. A parte inteira (8) multiplicamos por 10^{-1}. Há uma propriedade na matemática que descreve o seguinte: toda e qualquer potência que tenha expoente negativo é equivalente a uma fração na qual o numerador é a unidade positiva e o denominador é a mesma potência, porém apresentando o expoente positivo. Sendo assim, podemos representar $8x10^{-1}$ como 8x1/10, ou seja, invertemos o 10^{-1} e tornamos o expoente positivo. Como resultado desse cálculo, temos novamente o 0,8 (8/10). Esse procedimento é realizado com os outros números reais, conforme pode ser observado no exemplo acima.

Conversão de decimal para binário

Para converter uma base decimal para qualquer outra base, seja ela binária, octal ou hexadecimal, basta dividir, sucessivamente, o número decimal a ser convertido, pela base para a qual se quer converter, guardando o resto da divisão. O resultado dessa operação é novamente dividido pela base e devemos guardar o resto. Este processo irá se repetir até que o resultado seja menor que a base. Exemplo.

Ao converter 29 (decimal) para binário, após o cálculo, devemos chegar ao resultado $\mathbf{11101_2}$.

Como queremos representar o número 29 na base binária, então devemos dividi-lo por 2.

$$
\begin{array}{rrrr}
29 : 2 = & 14 \text{ , resto } \mathbf{1} & 2^0 \\
14 : 2 = & 7 \text{ , resto } \mathbf{0} & 2^1 \\
7 : 2 = & 3 \text{ , resto } \mathbf{1} & 2^2 \\
3 : 2 = & 1 \text{ , resto } \mathbf{1} & 2^3 \\
1 : 2 = & 0 \text{ , resto } \mathbf{1} & 2^4
\end{array}
$$

Conforme descrito acima, dividimos o número 29 por dois, guardamos o resto da divisão (1). O resultado (14) dividimos novamente por dois e guardamos o resto. Esse procedimento deve ser repetido até que o resultado seja menor que dois. O resultado final é composto por todos aqueles restos da divisão que guardamos durante o processo. A leitura é feita de baixo para cima, portanto temos como resultado $\mathbf{11101_2}$.

Conversão de números fracionários decimais em binário

Para converter números fracionários do sistema decimal em números fracionários do sistema binário, devemos multiplicar o número fracionário decimal, aquele que está depois da vírgula, por dois, pois é para essa base que estamos querendo converter. Este procedimento é o mesmo que utilizamos para decompor um número real na base dez; a única diferença é que estamos multiplicando pela base dois em vez de pela base dez.

Exemplo

Ao converter 0,8125 (decimal) para binário, o resultado, após o cálculo, deverá ser $\mathbf{0,1101_2}$. Veja, abaixo, como é feito o cálculo.

$$
\begin{array}{llll}
0{,}8125 & * 2 = & \mathbf{1}, 625 & 2^{-1} \\
0{,}625 & * 2 = & \mathbf{1}, 25 & 2^{-2} \\
0{,}25 & * 2 = & \mathbf{0}, 5 & 2^{-3} \\
0{,}5 & * 2 = & \mathbf{1}, 0 & 2^{-4} \\
 & & & \\
0{,}0 & & &
\end{array}
$$

No exemplo acima, o valor 0,8125 foi multiplicado por dois, resultando o valor 1,625. Guardamos o valor inteiro, neste caso o valor 1, e multiplicamos novamente o número

fracionário (0,625) por dois. Novamente guardamos o valor inteiro e multiplicamos o número fracionário por dois. Repetimos esse procedimento até que o valor decimal seja zero. Há casos, aqueles que caracterizam dízima, em que o valor do lado direito da vírgula nunca dará zero; neste caso temos que, em algum momento, parar a multiplicação.

Veja abaixo outro exemplo

Converter 0,3 (decimal) para binário; o resultado deverá ser 0,01001 1001 1001… $_{2,}$

0,3 * 2 =	**0**, 6	2^{-1}
0,6 * 2 =	**1**, 2	2^{-2}

0,2 * 2 =	**0**, 4	2^{-3}
0,4 * 2 =	**0**, 8	2^{-4}
0,8 * 2 =	**1**, 6	2^{-5}
0,6 * 2 =	**1**, 2	2^{-6}

…		
…		

Se continuarmos com essa multiplicação, veremos que não haverá um fim, pois a sequência 1001 começa a se repetir, o que caracteriza dízima, ou seja, essa mesma sequência de números sempre irá aparecer.

Conversão de decimal para hexadecimal

Usamos a mesma regra da conversão de decimal para binário, no entanto temos que usar a base adequada, ou seja, em vez de dividir o decimal por dois, iremos dividi-lo por 16, pois esta é a base para a qual queremos converter o número em questão. Não podemos esquecer que a base hexadecimal tem 16 símbolos: 0,1,3,4,5,6,7,8,9,A,B,C,D,E e F.

Para exemplificar, vamos converter o número 2540,34 (decimal) para hexadecimal e o resultado deverá ser, aproximadamente **9EC, 570A3$_{16}$. A conversão será feita em duas partes, conforme você poderá notar abaixo.**

a) Primeiro, resolveremos o lado esquerdo da vírgula, lado em que temos o número inteiro. Portanto devemos dividi-lo por 16.

$$
\begin{array}{rrrrr}
2540 : 16 = & 158 \text{ , resto } 12 & \Rightarrow \mathbf{C} & 16^0 \\
158 : 16 = & 9 \text{ , resto } 14 & \Rightarrow \mathbf{E} & 16^1 \\
9 : 16 = & 0 \text{ , resto } 9 & \Rightarrow \mathbf{9} & 16^2
\end{array}
$$

Na base 16, quando o resto da divisão for maior que 10, temos que representá-lo por sua respectiva letra, portanto, para o cálculo efetuado acima, para o lado esquerdo da vírgula teremos => **9EC** $_{16}$

b) Em segundo lugar, vamos resolver o lado direito da vírgula, no qual temos o número fracionário. Portanto devemos multiplicá-lo por 16. Veja, abaixo, como fica o cálculo.

$$
\begin{array}{rrrrr}
0{,}34 * 16 = & 5{,}44 & 5 \Rightarrow & \mathbf{5} & 16^{-1} \\
0{,}44 * 16 = & 7{,}04 & 7 \Rightarrow & \mathbf{7} & 16^{-2} \\
0{,}04 * 16 = & 0{,}64 & 0 \Rightarrow & \mathbf{0} & 16^{-3} \\
0{,}64 * 16 = & 10{,}24 & 10 \Rightarrow & \mathbf{A} & 16^{-4} \\
0{,}24 * 16 = & 3{,}84 & 3 \Rightarrow & \mathbf{3} & 16^{-5} \\
\ldots & & & & \\
\ldots & & & &
\end{array}
$$

Continuando a multiplicação teremos => **0, 570A3...** $_{16}$

Agora temos que agrupar os dois resultados, o que calculamos para o lado esquerdo da vírgula e o que calculamos para o lado direito da vírgula, e temos como resultado final **~9EC,570A3$_{16}$**

Convertendo de binário para decimal

Para fazer conversão do sistema binário para o sistema decimal, devemos proceder da seguinte forma: multiplicamos o primeiro número binário, da direita para a esquerda, por dois elevado a zero; o segundo número, da direita para a esquerda, multiplicamos por dois elevado a um; e assim sucessivamente até que todos os dígitos binários sejam multiplicados por dois e seu respectivo expoente. Os resultados dessas multiplicações devem ser somados para obtermos o número decimal. Veja abaixo um exemplo.

Converter 110101 (binário) para número decimal:

$$
\begin{aligned}
110101_{\,2} &= 1*2^5 + 1*2^4 + 0*2^3 + 1*2^2 + 0*2^1 + 1*2^0 \\
&= 1*2^5 + 1*2^4 + 1*2^2 + 1*2^0 \\
&= 1*32 + 1*16 + 1*4 + 1*1 \\
&= \mathbf{53}_{\,10}
\end{aligned}
$$

Para melhor visualizar o que foi feito, vamos analisar a tabela abaixo:

1	1	0	1	0	1
2^5	2^4	2^3	2^2	2^1	2^0

Como pode ser visto, cada dígito binário, representado na primeira linha da tabela, será multiplicado, da direita para a esquerda, pela base elevada a 0, depois a 1 e assim sucessivamente. A soma dessas multiplicações será o número em decimal.

Vamos desenvolver uma conversão considerando um binário negativo e fracionário. Vamos converter -11,101 (binário) para decimal.

$$
\begin{aligned}
-11{,}101_{\,2} &= -\,(1*2^1 + 1*2^0 + 1*2^{-1} + 0*2^{-2} + 1*2^{-3}) \\
&= -\,(1*2^1 + 1*2^0 + 1*2^{-1} + 1*2^{-3}) \\
&= -\,(1*2 + 1*1 + 1*0{,}5 + 1*0{,}125) \\
&= \mathbf{-\,3\,,\,625}
\end{aligned}
$$

Para a parte à direita da vírgula, usamos o mesmo método do exemplo anterior a este. A diferença está no cálculo do lado direito da vírgula. Perceba que multiplicamos o dígito binário pela base elevada a um valor negativo, começando por 2^{-1}, depois 2^{-2}, e assim sucessivamente. Lembre-se de que usamos a propriedade do expoente negativo; sendo assim, teremos ½ = 0.5 – ¼ = 0,25 e 1/8 = 0,125.

Conversão de hexadecimal para decimal

Para converter o sistema de numeração hexadecimal para o sistema de numeração decimal, usamos a mesma regra de conversão do sistema de numeração binário para o sistema de numeração decimal, no entanto temos que trocar a base.

Como exemplo, vamos converter o número hexadecimal 9EC,570A3 para decimal

$9EC,\ 570A3_{16} = 9*16^2 + 14*16^1 + 12*16^0 + 5*16^{-1} + 7*16^{-2} + 0*16^{-3} + 10*16^{-4} + 3*16^{-5}$

$= 2304 + 224 + 12 + 0{,}3125 + 0{,}02734375 + 0{,}0001525878... + 0{,}00000286102...$

$= \mathbf{2540\ ,\ 339999198913574218 75}$

Novamente vamos recorrer à tabela para facilitar a explanação.

9	E	C,	5	7	0	A	3
16^2	16^1	16^0	16^{-1}	16^{-2}	16^{-3}	16^{-4}	16^{-5}

Conforme pode ser observado na tabela acima, devemos multiplicar a parte esquerda da vírgula por 16^0, depois 16^1 e 16^2. A parte que está à direita da vírgula, multiplicamos por 16^{-1}, depois 16^{-2} e assim sucessivamente. Para efetuar a multiplicação, não podemos esquecer de converter as letras por números, conforme segue A=10, B=11, C=12, D=13, E=14, F=15.

Número binário e sua correspondência em hexadecimal e octal

Conforme a tabela abaixo, podemos perceber que precisamos de quatro dígitos binários para representar todos os símbolos hexadecimais.

Binario	**Hex**	**Binário**	**Octal**
0000	0	000	0
0001	1	001	1
0010	2	010	2
0011	3	011	3
0100	4	100	4
0101	5	101	5
0110	6	110	6

0111	7	111	7
1000	8		
1001	9		
1010	10		
1011	11		
1100	12		
1101	13		
1110	14		
1111	15		

Para fazer uma conversão rápida entre bases hexadecimais para binário, podemos separar cada número hexadecimal e representá-lo em binário, depois basta agregar estes números binários para a representação final. Veja o exemplo abaixo.

$-6F,A_{16}$ =

6 F A

$-0110\ 1111\ ,\ 1010_{2}$ = **$-1101111\ ,\ 101_{2}$**

O contrário também é verdadeiro; para representar um número hexadecimal em binário, devemos agrupar sempre em quatro dígitos, da direita para a esquerda, do lado esquerdo da vírgula, e da esquerda para a direita, do lado direito da vírgula, e dar seu equivalente em hexadecimal. Exemplo:

Converter $11{,}000011001_{2}$ para hexadecimal:

$11{,}000011001_{2} = 11\ ,\ 0000\ 1100\ 1_{2}$

$= 0011\ ,\ 0000\ 1100\ 1000_{2}$

$= 3\ ,\ 0\ C\ 8_{16}$

$= \mathbf{3\ ,\ 0C8}_{16}$

Separando a parte esquerda da vírgula, sempre considerando da direita para a esquerda, temos 11 e, como temos que agrupar sempre em quatro dígitos, completamos o 11 com dois zeros inseridos à esquerda, portanto teremos 0011. No lado direito da vírgula temos 000011001 e devemos proceder da mesma forma, ou seja, agrupamos em quatro dígitos a partir da esquerda para a direita; portanto teremos 0000 – 1100 e sobrará um dígito 1, o qual deverá ser agrupado com três zeros à direita, e, então, teremos 1000. Tendo os agrupamentos já feitos, então devemos considerar cada grupo binário e convertê-lo para hexadecimal.

Vamos converter outro valor - 101011101- para hexadecimal. Agora não temos mais o lado fracionário. O agrupamento fica da seguinte forma.

0001 0101 1101
1 5 D

Conforme a regra descrita, agrupamos da direita para a esquerda, os dígitos binários e completamos com zero o dígito 1, que ficou sozinho. Agora é só converter cada grupo para o hexadecimal. Portanto teremos o seguinte resultado: $15D_{16.}$

Conforme se pode perceber, podemos trabalhar com qualquer base, bastando, para isso, usar os procedimentos descritos acima e alterar a base para realizar o cálculo.

Aritmética binária

Os processadores realizam vários cálculos aritméticos para resolverem os trabalhos a eles entregues. Os cálculos aritméticos realizados na base binária são idênticos àqueles usados na base dez, no entanto não podemos esquecer que, nessa base, temos apenas dois símbolos.

Adição no sistema binário.

A adição no sistema binário segue a seguinte regra:

0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0 *e* vai 1

Observe que no sistema decimal, 1 + 1 tem como resultado o valor 2. Em binário, representamos o valor dessa soma como 10_2 , ou seja, o bit com valor 1 + o bit com valor 1, conforme demonstrado acima, resulta 0 e vai 1. Sendo assim, temos 10_2.

Veja abaixo a adição de $11_2 + 10_2$

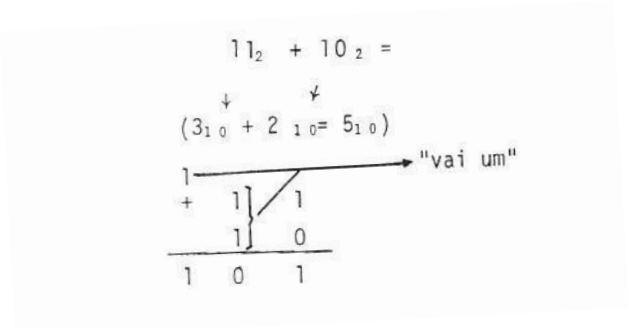


Figura 1: Adição de números binários: Idoeta, Capuano, 1985

Conforme se pode observar na figura acima, a soma é feita da mesma forma como se faz uma adição no sistema decimal, ou seja, primeiro somamos os primeiros termos da direita, neste caso 1 + 0. Pela regra apresentada, o valor dessa adição tem como resultado o valor 1.

Depois, somamos o valor 1 + 1 e, pela regra apresentada, o resultado dará zero (0) e vai 1. Este 1 que foi é somado com o próximo dígito, que, neste caso, não há, portanto o resultado desta operação será o próprio 1 que foi. O resultado final desta operação, como apresentado na operação, é 101_2. Vamos ver outro exemplo mais elaborado.

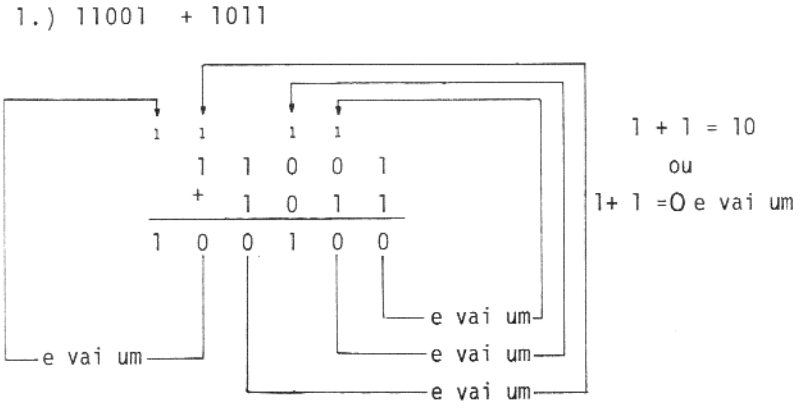


Figura 2: Adição de números binários: Idoeta, Capuano, 1985

No exemplo acima, temos vários casos em que a soma dos termos tem como resultado o zero e vai um. Analisando a figura, é possível acompanhar cada "vai um"; basta aplicar as regras estabelecidas para o caso da adição para cada termo.

Subtração de números binários

A subtração de números binários é semelhante à feita para o sistema de numeração decimal.

$0 - 0 = 0$

$1 - 1 = 0$

$1 - 0 = 1$

$0 - 1 = 1$ e "empresta 1"

A figura abaixo ilustra a subtração de números binários. Neste caso, temos que tomar bastante cuidado para não nos perdermos nos resultados. Cada termo deve ser subtraído e depois é preciso verificar se foi emprestado o valor 1 para ser considerado no resultado. Por exemplo, os primeiros termos da direita (0 – 1) foram subtraídos, tendo como resultado o valor 1, e foi emprestado 1 para o próximo termo. Perceba que este valor emprestado foi inserido ao lado do próximo termo para ser considerado na subtração. No segundo termo da direita para a esquerda foi efetuada a subtração (0 – 1). Pela regra temos como resultado o valor 1 e é emprestado 1. O resultado (1) foi subtraído do valor que foi emprestado; sendo assim temos 1 – 1 = 0. A subtração segue até que o último termo da direita para esquerda seja subtraído.

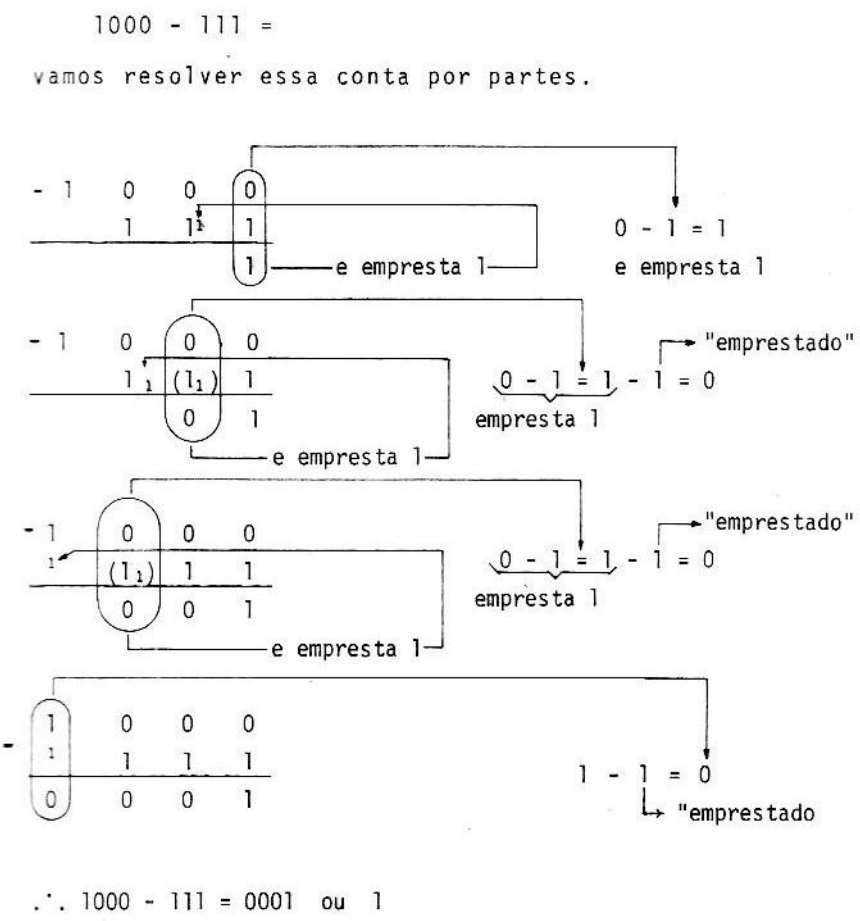


Figura 3: Subtração de números binários: Idoeta, Capuano, 1985

Multiplicação de números binários

A multiplicação é idêntica à realizada no sistema decimal.

0 x 0 = 0

0 x 1 = 0

1 x 0 = 0

1 x 1 = 1

Vamos analisar o exemplo abaixo.

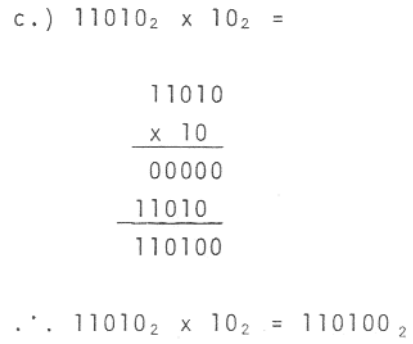


Figura 4: Multiplicação de números binários: Idoeta, Capuano, 1985

Como se pode perceber, a multiplicação é idêntica à feita no sistema decimal, no entanto deve ser aplicada a regra da adição ao somar o resultado da multiplicação pelos termos.

Material Complementar

STALLINGS, W. **Arquitetura e Organização de Computadores: Projeto Para o Desempenho.** 5. ed. São Paulo: Prentice Hall, 2004. (Biblioteca Digital)

TANENBAUM, A. S. **Organização Estruturada de Computadores.** 5 ed. São Paulo. Pearson Prentice Hall, 2007. (Biblioteca Digital)

Referências

IDOETA I, V.; CAPUANO, F. G. **Elementos de Eletrônica Digital**. 8. ed. São Paulo: Erica Editora LTDA. 1985.

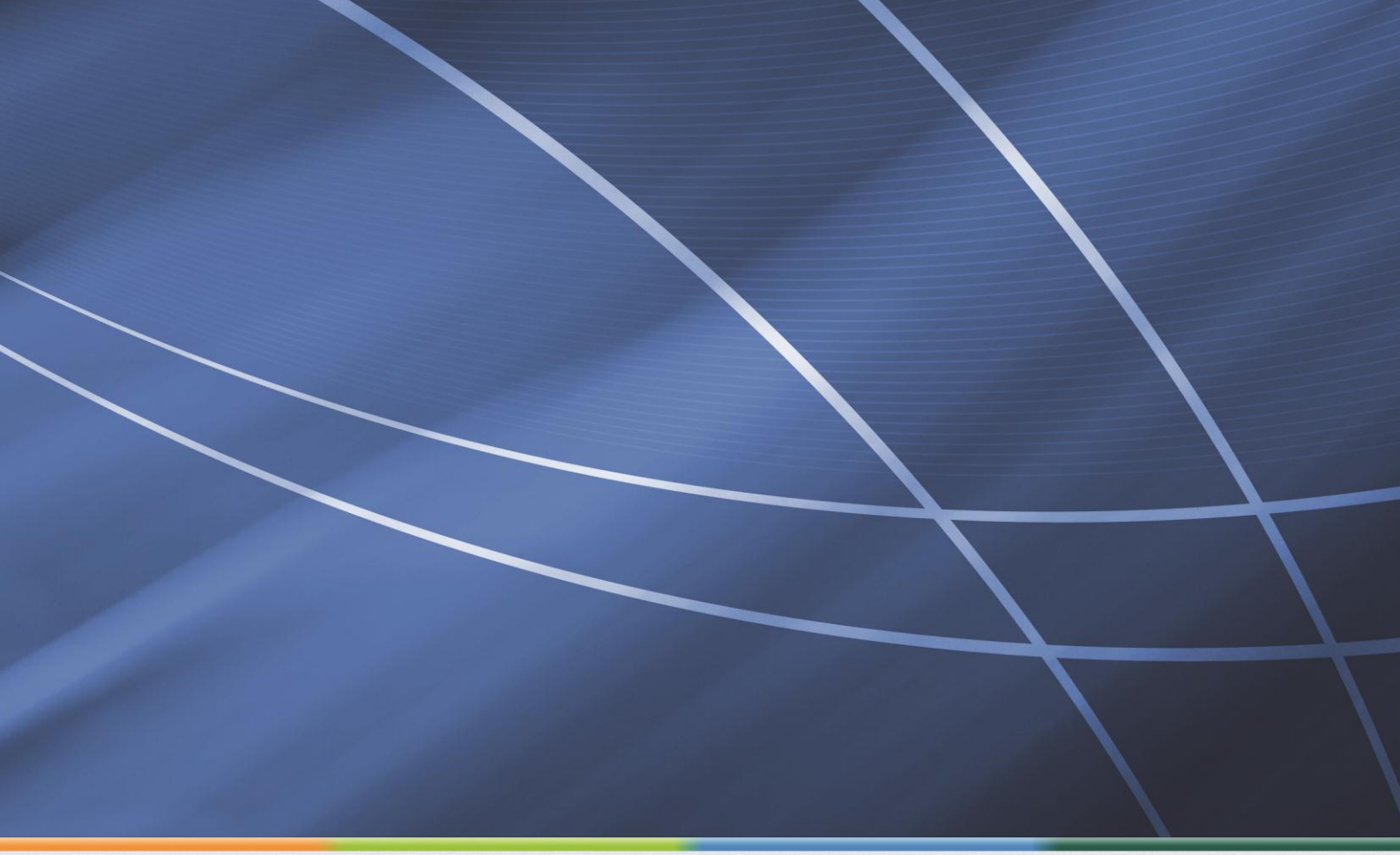
STALLINGS, W. **Arquitetura e Organização de Computadores:** Projeto Para o Desempenho. 5. ed. Sao Paulo: Prentice Hall, 2004.

REED, D. **A Balanced Introduction to Computer Science and Programming**. Creighton University, Prentice Hall. Tradução do capítulo 14. Disponível em http://professores.faccat.br/assis/davereed/14-DentroDoComputador.html acessado em 20.03.2012.

Anotações

Educação a Distância
Cruzeiro do Sul Educacional
Campus Virtual

www.cruzeirodosulvirtual.com.br
Campus Liberdade
Rua Galvão Bueno, 868
CEP 01506-000
São Paulo SP Brasil
Tel: (55 11) 3385-3000



Universidade Cruzeiro do Sul

Módulo Centro Universitário

UDF Centro Universitário

UNICID Universidade Cidade de S. Paulo