Introdução à Computação (I.C)

Módulo 03

Prof. Daniel Caixeta

Conteúdo programático

7 Conversão numérica: Como o computador pensa e executa.

7.1. Sistema posicional.
7.2. As bases [...].
7.3. Conversão entre bases 2, 8 e 16.
7.4. Conversão de base b para base 10.
7.5. Conversão de base 10 para base b.
7.6. Números binários negativos.
7.7. Aritmética binária.

8 A lógica binária: Circuitos lógicos e operadores.

8.1. Sistemas dicotômicos e a Álgebra de Boole.
8.2. Interruptores.
8.3. A lógica binária.
8.4. A soma em um computador.

Referências

7. CONVERSÃO NUMÉRICA.

Como o computador pensa e executa.

7.1. SISTEMA POSICIONAL

- O método de numeração de quantidades que adotamos, utiliza um sistema de numeração posicional.
- Significa que a posição ocupada por cada algarismo em um número altera seu valor de uma potência decimal (base 10) para cada casa à esquerda. Vejamos o exemplo abaixo:

$$125_{10} = \underbrace{1 \times 10^2}_{\substack{100 \\ \text{Centena}}} + \underbrace{2 \times 10^1}_{\substack{20 \\ \text{Dezena}}} + \underbrace{5 \times 10^0}_{\substack{5 \\ \text{Unidade}}}$$

7.2. AS BASES [...]

- A base de um sistema é a quantidade de algarismos disponíveis em sua representação.
- A base 10 (decimal) é hoje a mais utilizada, mas não é a única. Por exemplo, temos:
  - ✓ A dúzia (base 12).
  - ✓ O minuto = 60 segundos (base 60).
  - ✓ Etc ...
- Em computadores usamos outras bases como a binária (base 2), octal (base 8) e hexadecimal (base 16).

- Portanto, temos:

  - ✓ Base decimais (b_{10}) ➡ 0 1 2 3 4 5 6 7 8 9
  - ✓ Base binária (b_2) ➡ 0 1
  - ✓ Base octal (b_8) ➡ 0 1 2 3 4 5 6 7
  - ✓ Base hexadecimal (b_{16}) ➡ 0 1 2 3 4 5 6 7 8 9 A B C D E F

(10 algarismos + 6 símbolos)

$$N_b = a_0 \times b^n + a_1 \times b^{n-1} + \ldots + a_n \times b^0$$

- Exemplo: Converta o número 35 nas bases abaixo:

Decimal

$35 = \underbrace{3}\times10^1 + \underbrace{5}\times10^0 = 30 + 5 = 35_{10}$

Binário

$35 = \underbrace{1}\times2^5 + \underbrace{0}\times2^4 + \underbrace{0}\times2^3 + \underbrace{0}\times2^2 + \underbrace{1}\times2^1 + \underbrace{1}\times2^0 = 32 + 0 + 0 + 0 + 2 + 1 = 100011_2$

Octal

$35 = \underbrace{4}\times8^1 + \underbrace{3}\times8^0 = 32 + 3 = 43_8$

Hexadecimal

$35 = \underbrace{2}\times16^1 + \underbrace{3}\times16^0 = 32 + 3 = 23_{16}$

PRÁTICAS [...]

1. Converta os números abaixo para as suas respectivas bases usando o método de sistema posicional.

a) $24_{10} \rightarrow ?_2 =$

b) $121_{10} \rightarrow ?_2 =$

c) $24_{10} \rightarrow ?_8 =$

d) $121_{10} \rightarrow ?_{16} =$

7.3. CONVERSÃO ENTRE BASES 2, 8 E 16.

- As conversões mais simples são as que envolvem bases que são potências entre si. (FARIAS, 2013).
- Leva-se em consideração que $2^3 = 8$ e $2^4 = 16$.

- Exemplifiquemos a conversão 2^3, que funciona da seguinte forma:
    - 1º. Separa-se os algarismos de um número binário em grupos de três (começando sempre da direita para a esquerda).
    - 2º. Converta cada grupo de três algarismos por seu equivalente em octal.
- Vejamos:

cont.[...]

$$\underbrace{10}\underbrace{101}\underbrace{001}_2 = \color{red}{0}\color{black}{10\ .\ 101\ .\ 001_2}$$

- Olhando a tabela de conversão direta temos:

$$010_2 = 2_8 \quad 101_2 = 5_8 \quad 001_2 = 1_8 \qquad 251_8$$

$$10101001_2 = 251_8$$

Tabela 1. Conversão direta de binário para octal e vice-versa.

Binário	Octal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

cont.[...]

- Agora a conversão entre as bases 2 e 16.
- Como $2^4 = 16$, seguimos o mesmo processo anterior, bastando agora separarmos em grupos com <u>quatro</u> algarismos e converter cada grupo seguindo a Tabela 2. Por exemplo:

$$11010101101_2 = \underbrace{0110}\,.\,\underbrace{1010}\,.\,\underbrace{1101}_2$$

$$\left.\begin{array}{l} 0110_2 = 6_{16} \\ 1010_2 = A_{16} \\ 1101_2 = D_{16} \end{array}\right\} \quad 11010101101_2 = 6AD_{16}$$

Tabela 2. Conversão direta de binário para hexadecimal e vice-versa.

Bin.	Hexa.	Bin.	Hexa.
0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

cont.[...]

- Analisaremos agora a conversão inversa. Por exemplo:

$$A81_{16} = \underbrace{A}\,.\,\underbrace{8}\,.\,\underbrace{1}_{16}$$

- De acordo com a tabela ao lado, temos:

$$\left.\begin{array}{l} A_{16} = 1010_2 \\ 8_{16} = 1000_2 \\ 1_{16} = 0001_2 \end{array}\right\} A81_{16} = 1010.1000.0001_2$$

Tabela 3. Conversão direta de hexadecimal para binário e vice-versa.

Bin.	Hexa.	Bin.	Hexa.
0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

7.4. CONVERSÃO DE BASE b PARA BASE 10

- Lembremos da expressão geral descrita na página 6.

$$N_b = a_0 \times b^n + a_1 \times b^{n-1} + \ldots + a_n \times b^0$$

- A melhor forma de fazer a conversão é usando essa expressão. Como exemplo usemos o valor 101101_2.

$$101101_2 = 1\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0 = 45_{10}$$

- Outros exemplos:

$$125_5 = 1\times5^3 + 0\times5^2 + 0\times5^1 + 0\times5^0 = 100_{10}$$

$$485_9 = 4\times9^2 + 8\times9^1 + 5\times9^0 = 324 + 72 + 5 = 401_{10}$$

7.5. CONVERSÃO DE BASE 10 PARA BASE b

- Já a conversão de números da base 10 para uma base qualquer, emprega-se algoritmos que serão de ordem inversa das anteriores.
- O número decimal será dividido sucessivas vezes pela base 2, o resto de cada divisão ocupará sucessivamente as posições de ordem 0 e 1, e assim por diante, até que o resto da última divisão resulte em quociente 0.
- Esse último quociente irá ocupar a posição mais alta ordem.

- Exemplo: Converta o numero 19_{10} para a base 2.

$$
\begin{array}{cccccc}
19 & \multicolumn{1}{|l}{2} & & & & \\ \cline{2-2}
1 & 9 & \multicolumn{1}{|l}{2} & & & \\ \cline{3-3}
 & 1 & 4 & \multicolumn{1}{|l}{2} & & \\ \cline{4-4}
 & & 0 & 2 & \multicolumn{1}{|l}{2} & \\ \cline{5-5}
 & & & 0 & 1 &
\end{array}
$$

$19_{10} = 10011_2$

$a_4 = 1 \quad a_3 = 0 \quad a_2 = 0 \quad a_1 = 1 \quad a_0 = 1$

- Usando a conversão anterior como prova real, temos:

$$10011_2 = (1 \times 2^4) + (0 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = 19_{10}$$

- Conversão do número 512_{10} para a base 16:

$$
\begin{array}{rr|l}
512 & 16 & \\
\hline
0 & 32 & 16 \\
\cline{3-3}
 & 0 & 2
\end{array}
\qquad\qquad 512_{10} = 200_{16}
$$

$a_2 = 2 \quad a_1 = 0 \quad a_0 = 0$

- Na prova real, temos:

$$200_{16} = (2 \times 16^2) + (0 \times 16^1) + (0 \times 16^0) = 512_{10}$$

7.6. NÚMEROS BINÁRIOS NEGATIVOS

- Segundo Farias (2013), os computadores operam com números positivos (+) e negativos (-), sendo necessário encontrar uma representação para números com sinal negativo.
- Existem uma grande variedade de opções. Apresentemos aqui as duas formas mais usuais:

1. Complemento de 1.
2. Complemento de 2.

1. Complemento de 1

- Na representação em complemento de 1 invertem-se todos os *bits* de um número para representar o seu complementar.
- Converte-se um valor positivo por um negativo, e vice-versa.
- Quando o *bit* mais à esquerda é 0, esse valor é positivo; se for 1, então é negativo. Por exemplo:

$$100_{10} = 01100100_2 \text{ (com 8 } \textit{bits}\text{)}$$

$$-100_{10} = 10011011_2 \text{ (}\textit{bits}\text{ invertidos)}$$

O problema desta representação é que existem 2 padrões de *bits* para o 0, havendo assim desperdício de representação:

$$0_{10} = 00000000_2 = 11111111_2$$

2. Complemento de 2

- Para determinar o negativo de um número na forma de complemento de 2, basta inverter todos os *bits* e somar 1 unidade.

1. Representação binária
$101_{10} = 01100101_2$ (com 8 *bits*)

2. Invertendo todos os *bits*
$01100101_2 \rightarrow 10011010_2$

3. Somando 1 unidade
$10011010_2 + 1$

$$\begin{array}{r} 10011010_2 \\ +\ 1 \\ \hline \end{array}$$

$$\boxed{10011011_2 \quad = \quad -101_{10}}$$

Aritmética binária
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0 (sobe 1)

EXERCÍCIO

1. Determine o número binário negativo de 120_{10} em 8 *bits* usando a representação de complemento 1.

Solução:

1. Converte dec. para bin. $120_{10} = 01111000_2$

2. Inverte os bits binários $10000111_2 = -120_{10}$

2. Qual o número representado por 11100100_2 (com 8 *bits*)? Este número é negativo ou positivo?

Solução:

1. Verifique se o *bit* mais a esquerda é 0 (para positivo) ou 1 para negativo).
Obs.: Já vimos esta informação na página 19.

11100100_2 Negativo

2. Inverte todos os bits e soma 1 unidade

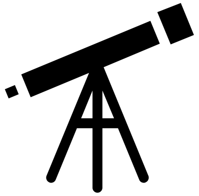
$00011011_2 + 1$

$$\begin{array}{r} {\scriptstyle 11} \\ 00011011_2 \\ +1 \\ \hline 00011100_2 \end{array} = 28_{10}$$

$$\boxed{11100100_2 = -28_{10}}$$

Aritmética binária

$0 + 0 = 0$
$0 + 1 = 1$
$1 + 0 = 1$
$1 + 1 = 0$ (sobe 1)



Aqui ... Dica boa!

7.7. ARITMÉTICA BINÁRIA

- Como o computador manipula dados (números) através de uma representação binária, veremos a partir de agora como a aritmética do sistema binário, a mesma usada pela ULA (Unidade Lógica Aritmética) dos processadores, processa esses dados.

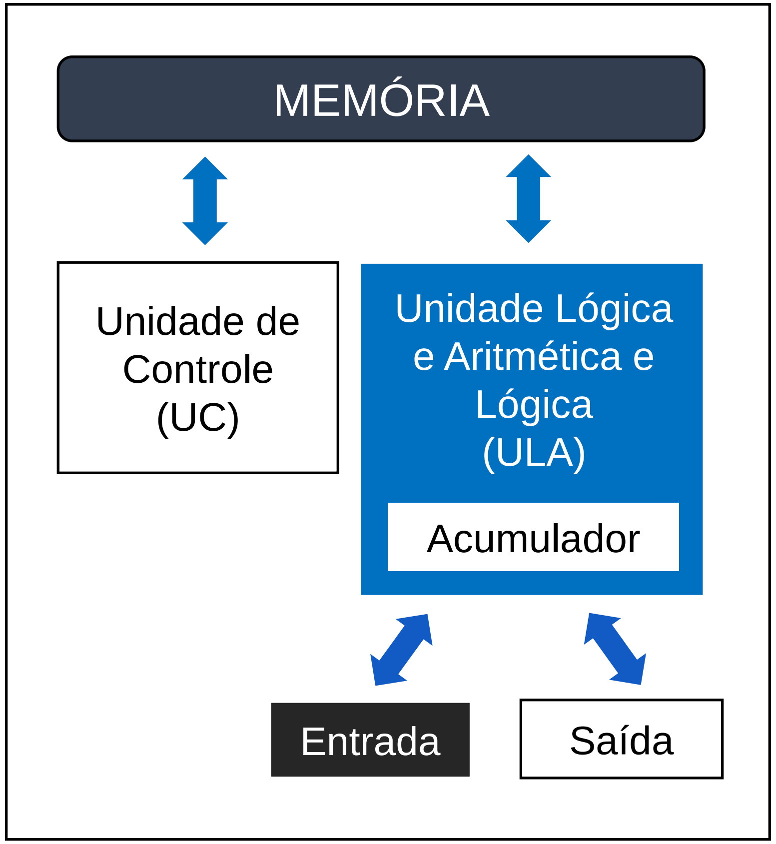


Figura 1. Arquitetura proposta por John von Neumann (1946).

Soma e Subtração Binária

Tabela 4. Tabuada de soma binária.

Operação	Valor	Obs.
0 + 0 =	0	
0 + 1 =	1	
1 + 0 =	1	
1 + 1 =	0	“vai um”(*)
1 + 1 + 1 =	1	“vai um” (*)

Tabela 5. Tabuada de subtração binária.

Operação	Valor	Obs.
0 - 0 =	0	
0 - 1 =	1	“vem um”(**)
1 - 0 =	1	
1 - 1 =	0	

(*) – Vai um para a ordem superior ou seja, sobe 1, como na aritmética tradicional.

(**) – Vem um do próximo

cont.[...]

- Exemplo 01: Efetue $011100_2 + 011010_2$

Soma-se as posições da direita para esquerda, tal como uma soma decimal.

$$
\begin{array}{rl}
 & 1\,1 \quad \leftarrow \text{“vai um”} \\
 & 0\,1\,1\,1\,0\,0_2 \\
+ & 0\,1\,1\,0\,1\,0_2 \\
\hline
 & 1\,1\,0\,1\,1\,0_2
\end{array}
$$

- Exemplo 02: Efetue $11100_2 - 01010_2$

Como não é possível tirar 1 de 0, o artifício é “pedir emprestado” 1 da casa de ordem superior, ou seja, na realidade o que se faz é subtrair 1_2 de 10_2 e encontramos 1_2 como resultado, devendo então subtrair 1 do dígito de ordem superior. Este modo é o mesmo da subtração em decimal.

Subtrai-se a direita para a esquerda.

$$
\begin{array}{rl}
 & 0\,2 \quad \leftarrow \text{“vem um”} \\
 & 1\,1\,\cancel{1}\,0\,0_2 \\
- & 0\,1\,0\,1\,0_2 \\
\hline
 & 1\,0\,0\,1\,0_2
\end{array}
$$

Acesse aqui a calculadora binária

APRENDA MAIS

- Dica bacana [...].

Vídeo sobre soma e subtração binária.

https://www.youtube.com/watch?v=NeQBC9Z5FHk

A subtração nos computadores

- Na eletrônica digital a construção de circuitos simples custa menos e operam mais rápido do que circuitos mais complexos.
- Portanto, os números utilizados na aritmética em Complemento de 2 (pg. 19), permitem a implementação e uso de circuitos mais simples, baratos e rápidos.
- Uma característica é que tanto os números com sinal quanto os números sem sinal podem ser somados pelo mesmo circuito.
- Por exemplo, suponha que você deseja somar os números sem sinal 132_{10} e 14_{10}.

cont.[...]

$$
\begin{array}{r}
1\,0\,0\,0\,0\,1\,0\,0_2 \\
+ \quad 0\,0\,0\,0\,1\,1\,1\,0_2 \\
\hline
1\,0\,0\,1\,0\,0\,1\,0_2
\end{array}
$$

Entrada: A

Entrada: B

Saída: C

$$
\begin{array}{r}
+\ 132_{10} \\
+\ \ 14_{10} \\
\hline
+\ 146_{10}
\end{array}
$$

?

- O microprocessador tem um circuito na ULA que pode somar números binários sem sinal, que quando aparece o padrão A em uma entrada e B na outra entrada, resulta C na saída.

- Surge a pergunta: como a ULA sabe que os padrões de *bits* nas entradas representam número sem sinal e não em complemento de dois?
- A resposta é: NÃO SABE! A ULA sempre soma como se as entradas fossem números binários sem sinal. Sempre produzirá o resultado correto, mesmo se as entradas forem números em complemento de dois. É um <u>circuito/instrução corretiva</u>.

- Isto comprova um ponto importante. O somador na ULA sempre soma padrões de *bits* como se eles fossem números binários sem sinal.
- É de nossa interpretação decidir se números com ou sem sinal estão sendo tratados, ou melhor ainda, o bom do complemento de dois é que os padrões de *bits* podem ser interpretados de qualquer maneira.
- Isto nos permite trabalhar com números com e sem sinal sem requerer diferentes circuitos para cada padrão. (FARIAS, 2013).

- A aritmética de complemento de 2 também simplifica a ULA em outro ponto.
- Todo microprocessador precisa da instrução de subtração. Assim, a ULA deve ser capacitada a subtrair um número de outro. Entretanto, se isto necessitar de um circuito de subtração separado, a complexidade e o custo da ULA seriam aumentados.
- Felizmente, a aritmética de complemento de 2 permite realizar operações de subtração usando um circuito somador. Ou seja, a CPU usa o mesmo circuito tanto para soma como para subtração.

Multiplicação e Divisão Binária

Tabela 6. Tabuada multiplicação binária.

Operação	Valor
0 x 0 =	0
0 x 1 =	0
1 x 0 =	0
1 x 1 =	1

O processo é idêntico à multiplicação entre números decimais.

- Exemplo: Efetue $101_2 \times 110_2$

$$\begin{array}{rr} & 1\,0\,1_2 \\ \times & 1\,1\,0_2 \\ \hline & 0\,0\,0 \\ + & 1\,0\,1 \\ & 1\,0\,1 \\ \hline & 1\,1\,1\,1\,0_2 \end{array} \qquad \begin{array}{rr} & 5_{10} \\ \times & 6_{10} \\ \hline & 30_{10} \end{array} = 1\,1\,1\,1\,0_2$$

8. A LÓGICA BINÁRIA.

Circuitos lógicos e operadores.

8.1. SISTEMAS DICOTÔMICOS E A ÁLGEBRA DE BOOLE

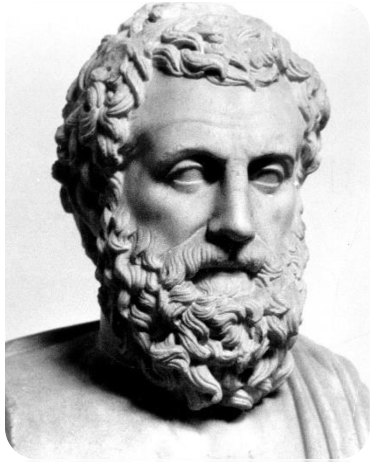
- Segundo Daghlian (2008), o mundo em que vivemos apresenta situações dualísticas em sua grande maioria, ou seja, com dois estados que mutuamente se excluem.

Tabela 7. Situações dualísticas.

Valor 1	Valor 2
1	0
Sim	Não
Dia	Noite
Preto	Branco
Ligado	Desligado

- Existem situações como morno, tépido, diferentes tons de cores que não apresentam estritamente como dicotômicas, i.e., com dois estados excludentes bem definidos.

- Segundo Daghlian (2008) a Lógica começou a se desenvolver no século IV a.C., com Aristóteles.
- Neste período os filósofos gregos passaram a usar em suas discussões sentenças lógicas enunciadas nas formas afirmativas e negativas, resultando assim em grande simplificação da realidade no dia a dia [...]. E quase 2.000 anos depois, por volta de 1666, Leibniz usou em vários trabalhos o que chamou de *Calculus ratiotinator*. Essas ideias nunca foram teorizadas, porém seus escritos trazem a ideia do que seria a Lógica Matemática.



Aristóteles
(384 – 322 a.C.)

Gottfried W. Leibniz
(1646 – 1716)

- Em 1854, George Boole introduz o formalismo que até hoje usamos para o tratamento sistemático da lógica, chamada de Álgebra Booleana, que pode ser definida como um conjunto de operadores e de axiomas, que são assumidos verdadeiros sem a necessidade de prova. (Güntzel & Nascimento, 2001).
- Em 1938, C. E. Shannon aplicou esta álgebra para mostrar que as propriedades de circuitos elétricos de chaveamento podem ser representadas com dois valores. (*ibidem*).
- Para Güntzel & Nascimento (2001), diferentemente da álgebra ordinária dos números reais, onde as variáveis podem assumir valores no intervalo (-∞ ; +∞), as variáveis booleanas só podem assumir um número finito de valores.

0 ou 1

George Boole
(1815 – 1864)

- Em particular, na álgebra booleana de dois valores, cada variável pode assumir um dentre dois valores possíveis. Por exemplo:

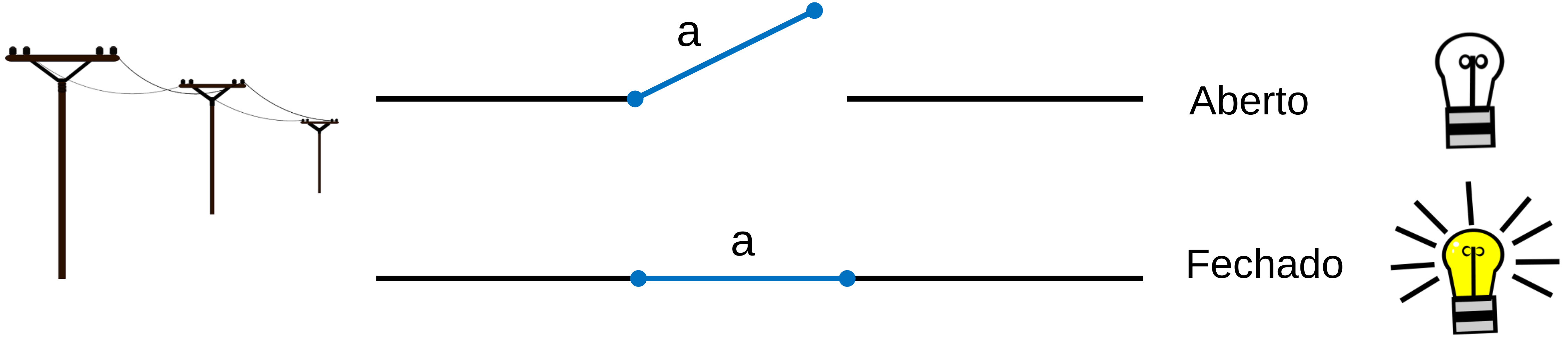
Tabela 8. Exemplo de operadores.

Operador	Valores
V ou F	Verdadeiro ou Falso
C ou E	Certo ou Errado
0 ou 1	-

- Portanto são sistemas dicotômicos. Computacionalmente dizendo 0 e 1, a qual é também utilizada na eletrônica digital de circuitos.

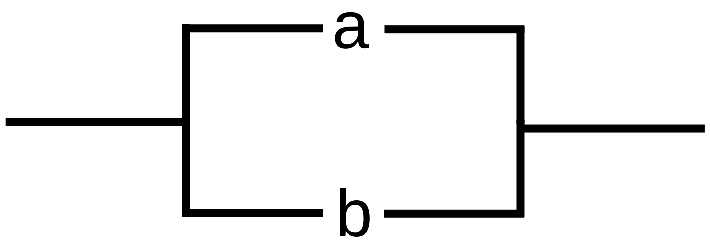
8.2. INTERRUPTORES

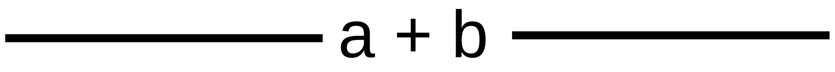
- Chamamos interruptor ao dispositivo ligado a um ponto de um circuito elétrico, o que pode assumir um dos dois estados, *e.g.*, Fechado (1) e Aberto (0). (DAGHLIAN, 2008).
- Quando fechado, o interruptor permite que a corrente passe através do ponto, enquanto aberto nenhuma corrente passa.



- Por conveniência, representaremos os interruptores da seguinte forma (DAGHLIAN, 2008):

- Sejam a e b dois interruptores ligados em paralelo, só passará corrente se pelo menos um dos interruptores estiver fechado. Então denotaremos a ligação de dois interruptores em paralelo por a + b. (*ibidem*). Então:





- Mas se dois interruptores estão ligados em série, só passará corrente se ambos estiverem fechados, i.e., a = b = 1. Portanto detonaremos a ligação de dois interruptores a e b por a . b ou simplesmente ab. (DAGHLIAN, 2008):

—— a —— b ——   —— a . b ——

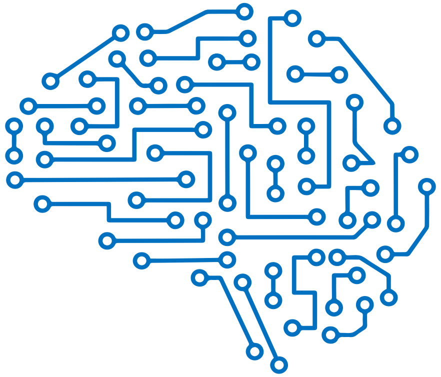
- Então temos:

Paralelo	Série
0 + 0 = 0	0 . 0 = 0
0 + 1 = 1	0 . 1 = 0
1 + 0 = 1	1 . 0 = 0
1 + 1 = 1	1 . 1 = 1

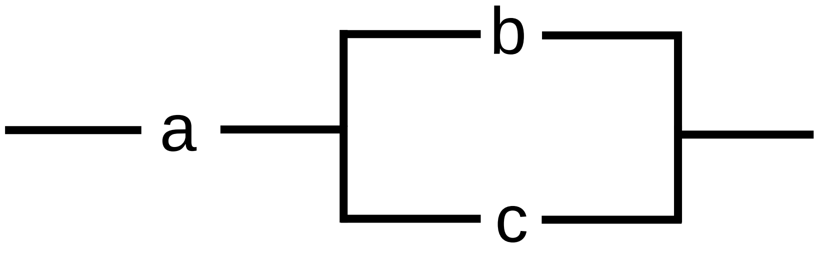
Tabela 9. Possíveis ligações em interruptores.

cont.[...]

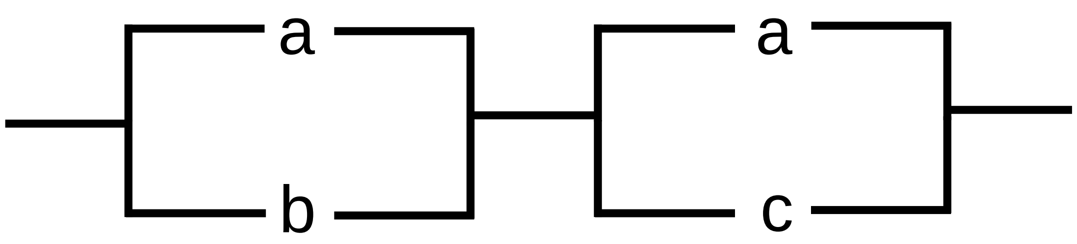
- Observações importantes segundo Daghlian (2008):
  - Conhecendo o estado de um interruptor a, podemos compreender que qualquer outro interruptor tenha o mesmo estado de a, i.e., aberto quando a está aberto e fechado quando a está fechado.
  - Chamamos de complemento quando um interruptor aberto está fechado e vice-versa. Isso se chama de inversão ou negação.
  - Por exemplo: $a \neq a' \longrightarrow 1 \neq 0$
  - É possível criar inúmeras operações/expressões lineares na lógica digital compreendendo suas ligações nos circuitos apresentados.



- Apresentamos abaixo algumas equações baseadas em circuitos lógicos.



e



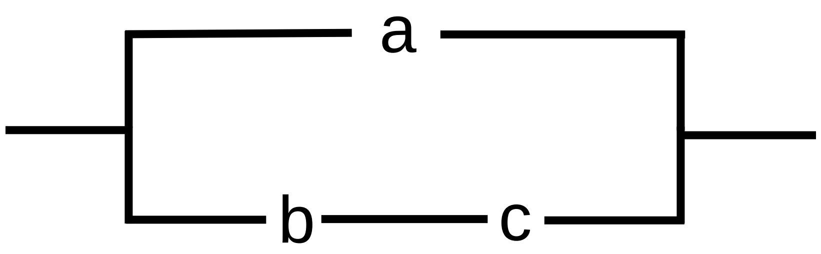
a . (b + c)

(a + b) . (a + c)

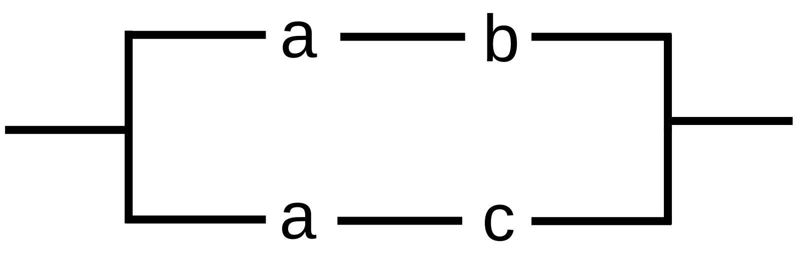
a . (b + c) = (a + b) . (a + c)

(Circuitos iguais)

Outro exemplo:



e

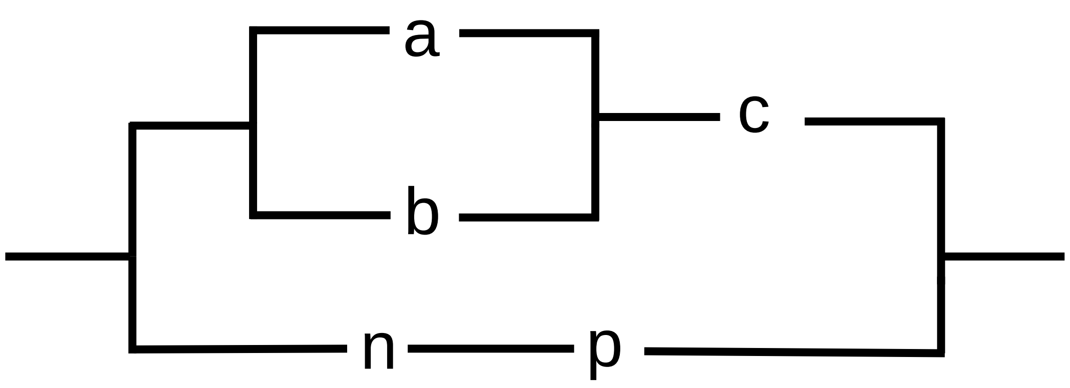


$a + (b \cdot c)$ $(a \cdot b) + (a \cdot c)$

$$a + (b \cdot c) = (a \cdot b) + (a \cdot c)$$

(Circuitos iguais)

- Exercício 1. Determine a ligação do seguinte circuito:

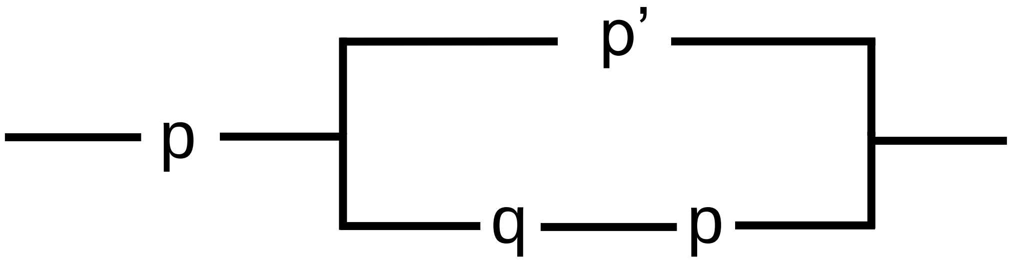


Solução: $(a + b) . c + (n . p)$

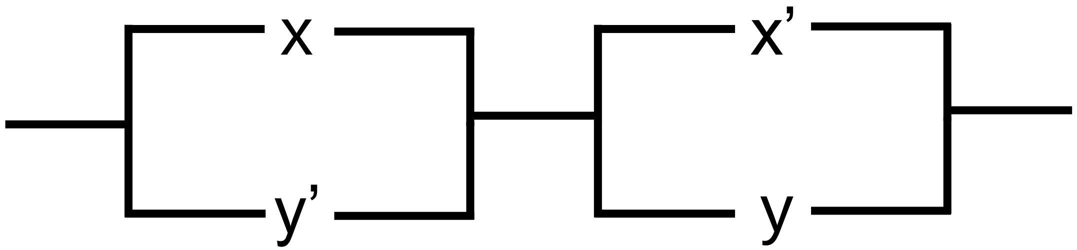
- Exercício 2. Desenhe os circuitos cujas as ligações são:

Solução <u>a</u>

a) p . (p’ + q . p)
b) (x + y’) . (x’ + y)



Solução <u>b</u>



8.3. A LÓGICA BINÁRIA

- George Boole publicou a Álgebra booleana em 1854 como sendo um sistema completo que permitia a construção de modelos matemáticos para o processamento computacional.
- O interessante na lógica booleana é que a partir de três operadores básicos (NOT, AND e OR), podemos construir circuitos lógicos capazes de realizar diversas operações em um computador.
- Como o número de valores que cada variável pode assumir é finito (e pequeno), o número de estados que uma função booleana pode assumir também será finito, o que significa que podemos descrevê-las completamente utilizando tabelas.

- Essa tabela recebe o nome de Tabela Verdade, e nela são listadas todas as combinações de valores que as variáveis de entrada podem assumir e os correspondentes valores da função (saídas).

1. Operador NOT

O operador unário NOT, negação binária, resulta no complemento do operando, ou seja, será um *bit* 1 se o operando for 0, e será 0 caso contrário, conforme apresentado Tabela 10, onde A é o *bit* de entrada e S é a resposta, ou *bit* de saída.

Tabela 10. Tabela verdade operador NOT.

A	S ou A’
0	1
1	0

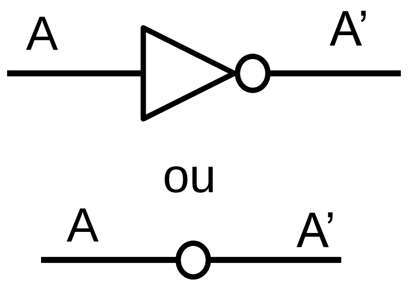


Figura 2. Representação gráfica do operador lógico NOT, com seus valores de entrada e saída.

cont.[...]

2. Operador AND

O operador binário AND, ou conjunção binária devolve um *bit* 1 sempre que ambos operandos sejam 1, conforme podemos confirmar na Tabela 11, onde A e B são *bits* de entrada, e S é o *bit*-resposta, ou *bit* de saída.

Tabela 11. Tabela verdade operador AND.

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

(A . B)

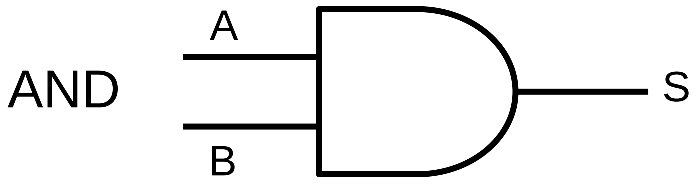


Figura 3. Representação gráfica do operador lógico AND, com seus valores de entrada e saída.

cont.[...]

3. Operador OR

O operador binário OR, ou <u>disjunção binária</u> devolve um *bit* 1 sempre que pelo menos um dos operandos seja 1, conforme podemos confirmar na Tabela 12, onde A e B são os *bits* de entrada, e S é o *bit*-resposta, ou *bit* de saída.

Tabela 12. Tabela verdade operador OR.

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

$$(A + B)$$

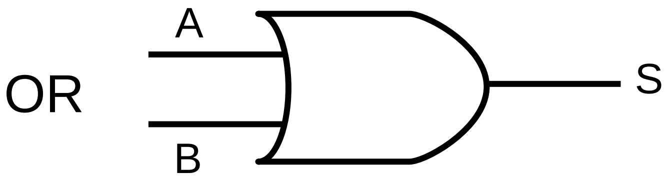


Figura 4. Representação gráfica do operador lógico OR, com seus valores de entrada e saída.

8.4. A SOMA EM UM COMPUTADOR

- Neste módulo, aprendemos sobre os sistemas de numeração, dando ênfase ao sistema binário, sendo este o sistema adotado pelos computadores.
- Aprendemos como funciona a aritmética binária (soma, subtração, multiplicação, etc.), representação negativa dos números, entre outros.
- Mas como um computador soma?
- Primeiro, precisamos abordar as portas lógicas, elas são a base para as outras operações.

- A construção de uma porta lógica, utiliza conhecimentos de circuitos eletrônicos formados por diodos, resistências, resistores, capacitores entre outros que são abordados em cursos avançados da Eletrônica Analógica/Digital, entretanto, seu entendimento foge ao escopo da nossa disciplina.
- O importante é sabermos que existem portas lógicas que seguem a lógica binária já apresentada e que estas portas podem ser combinadas, formando os circuitos digitais.
- A Figura 5 apresenta um circuito digital somador de 2 *bits*.

cont.[...]

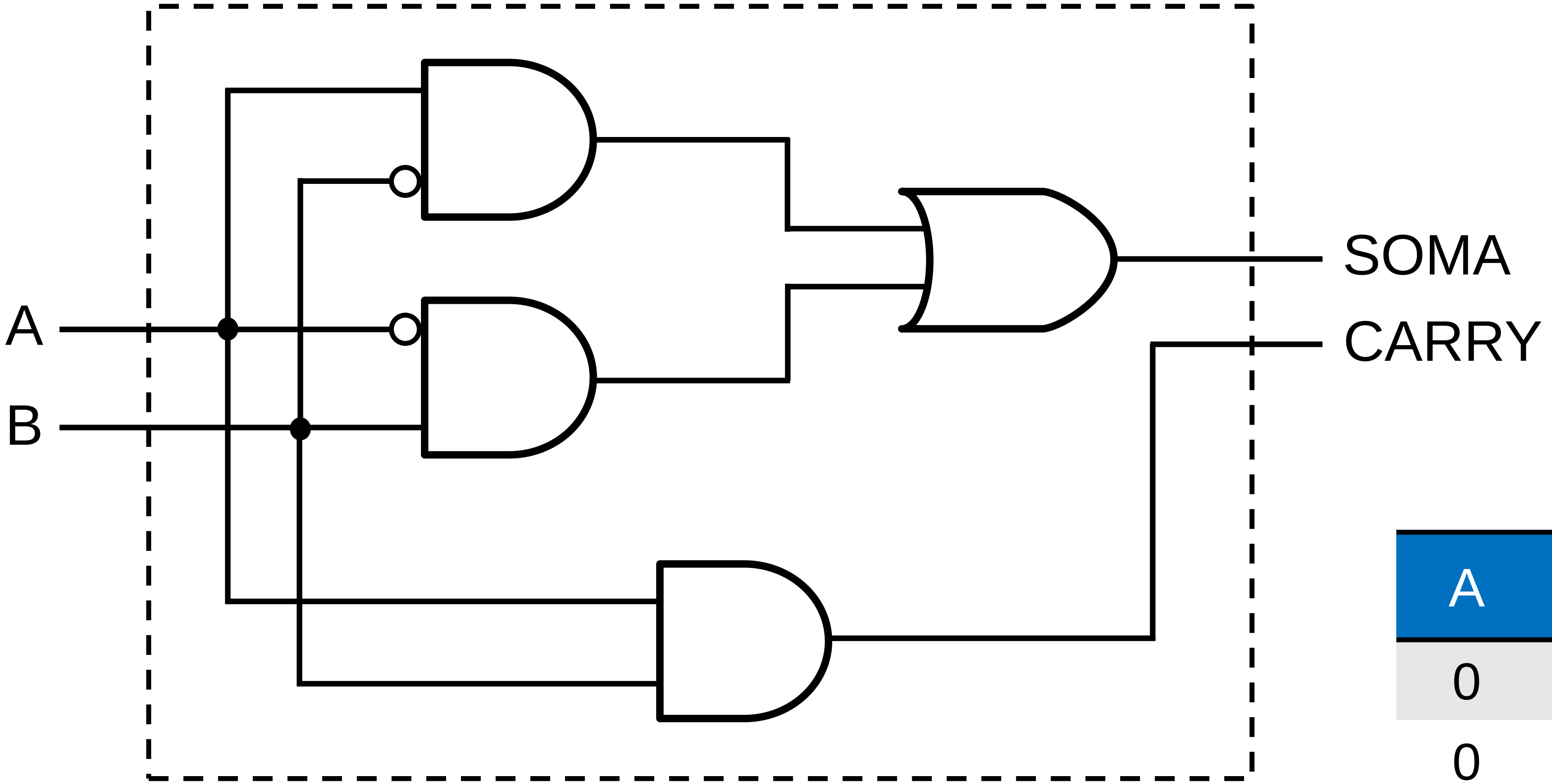


Figura 5. Circuito digital somador de dois *bits* formado pelas portas lógicas básicas (AND, OR, NOT).

A	B	SOMA	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Tabela 13. Tabela de valores da operação de Soma de 2 *bits*.

- Para entendermos o passo a passo do circuito digital proposto, torna-se necessário criarmos uma tabela verdade para que em caso de dúvidas sobre os valores, revisarmos a operação binária (SOMA) dos dois *bits*, e constatar se a saída corresponde ao esperado, considerando que a saída CARRY é a operação “vai um” da aritmética binária, neste caso a soma.
- E uma dica bem legal. Vejam o vídeo abaixo:

Circuito digital somador de 2 *bits*:

https://www.youtube.com/watch?v=E5yDNF2clQw

REFERÊNCIAS

DAGHLIAN, Jacob. Lógica e Álgebra de Boole. 4ª ed. 12ª reimpr. São Paulo : Atlas, 2008.

FARIAS, Gilberto. Introdução à computação. UFPB, 2013.

GÜNTZEL, José Luís; NASCIMENTO, Francisco de Assis. Introdução aos Sistemas Digitais (v.2001/1). Disponível in: https://www.inf.ufsc.br/~j.guntzel/isd/isd.html.

TANENBAUM, Andrew S. Sistemas Operacionais Modernos. 3ª ed. Pearson, 2010.

TANENBAUM, Andrew S. AUSTIN, Todd. Organização Estruturada de Computadores. 6ª ed. Pearson, 2013.

MONTEIRO, Mário A. Introdução à Organização dos Computadores. 5ª ed. LTC. 2014.

TANENBAUM, Andrew S. VAN STEEN, Maarten. Sistemas Distribuídos. Princípios e Paradigmas. 2ª ed. Pearson, 2007.

RIBEIRO, Carlos; DELGADO, José. Arquitetura de Computadores. 2ª ed. Rio de Janeiro: LTC, 2009.

Obrigado!

Disciplina: Introdução à Computação (I.C)