***Sistemas de Informação***
***Disciplina: Arquitetura e Organização de Computadores - 1º Período***
***Professor: José Maurício S. Pinheiro***

## *AULA 2: Bases Numéricas*

As informações binárias são representadas em computadores digitais por meio de sinais elétricos. Estes sinais podem ser representados pelo nível de tensão, que vai especificar um estado entre dois possíveis. Por exemplo, se um sinal digital apresenta um nível de tensão de +5v, considera-se que o mesmo representa o valor digital 1. Da mesma forma, se o sinal digital apresenta um nível de tensão de -5v, então este representa o valor digital 0 (Fig. 1).

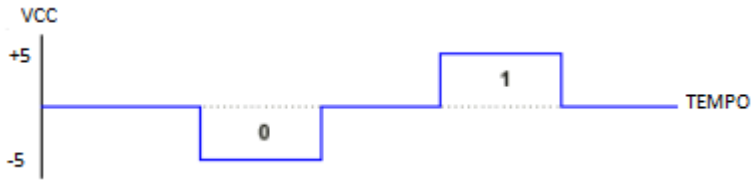



**Figura 1 - Representação de sinais digitais**

## 1. Unidades de Informação

A menor unidade de informação armazenável em um computador é o algarismo binário ou dígito binário, conhecido como bit (*binary digit*). O bit assume dois valores: 0 e 1. O menor grupo ordenado de bits representando uma informação útil e inteligível para o ser humano é o caractere. Cada sistema poderá definir como cada conjunto de bits irá representar um determinado caractere, quantos bits e como se organizam.
Ao conjunto de 4 bits denominamos *nibble*. Já o conjunto de 8 bits foi definido pela primeira vez pela IBM e é chamado de *Byte*. A palavra é o conjunto de bits que representa uma informação útil e está relacionada com o armazenamento e a transferência de informações entre a Memória Principal e a CPU. Portanto, palavra é um conjunto de Bytes. Um computador com uma palavra de 32 bits tem 4 Bytes/palavra.
As operações de armazenamento e recuperação de dados na memória são feitas Byte a Byte ou palavra a palavra, sendo comum mencionar o tamanho de uma memória em termos de Bytes. Para representar valores maiores, utilizamos o sistema métrico de grandeza, com algumas adaptações para os computadores, conforme mostra a Tabela 1.

**Tabela 1 - Grandezas de medida**

| Sufixo | Quantidade |
|---|---|
| Kilo (K) | 1 Kilobytes ou 1 KB = 1024 = $2^{10}$ |
| Mega (M) | 1 Megabytes ou 1 MB = 1.048.576 = $2^{20}$ |
| Giga (G) | 1 Gigabytes ou 1 GB = 1.073.741.824 = $2^{30}$ |
| Tera (T) | 1 Terabytes ou 1 TB = 1.099.511.627.776 = $2^{40}$ |
| Peta (P) | 1 Petabytes ou 1 PB = 1.125.899.906.843.624 = $2^{50}$ |
| Exa (E) | 1 Exabytes ou 1 EB = 1.152.921.504.607.870.976 = $2^{60}$ |

## 2. Sistemas Numéricos

São sistemas de notação usados para representar quantidades abstratas denominadas números. Um sistema numérico é definido pela base que utiliza. A base de um sistema é o número de símbolos diferentes, ou algarismos, necessários para representar um número qualquer.
O sistema decimal, utilizado de forma universal, utiliza dez símbolos diferentes (ou dígitos) para representar um número. É, portanto, um sistema numérico na base 10. Já os computadores utilizam a base 2 (sistema binário) e os programadores, por facilidade, usam em geral uma base que seja uma potência de 2, tal como $2^4$ (base 16 ou sistema hexadecimal) ou eventualmente ainda $2^3$ (base 8 ou sistema octal).
Se na base 10, dispomos de 10 algarismos para a representação do número: 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9. Na base 2, seriam apenas 2 algarismos: 0 e 1. Na base 16, seriam 16: os 10 algarismos aos quais estamos acostumados, mais os símbolos A, B, C, D, E, F, representando respectivamente 10, 11, 12, 13, 14 e 15 unidades. Generalizando, temos que uma base b qualquer disporá de b algarismos, variando entre 0 e (b-1).

### 2.1.1 Sistema Decimal

Os símbolos ou dígitos utilizados são os algarismos 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9. Os elementos são agrupados de dez em dez e, por essa razão, os números podem ser expressos por intermédio de potência de dez e recebem o nome de sistema de numeração decimal. Ex. $486 = 400 + 80 + 6 = 4 \times 100 + 8 \times 10 + 6 \times 1 = 4 \times 10^2 + 8 \times 10^1 + 6 \times 10^0$, ou seja, $486 = 4 \times 10^2 + 8 \times 10^1 + 6 \times 10^0$.

### 2.1.2 Sistema Binário

Como o próprio nome já indica tem base 2, pois utiliza apenas dois símbolos ou algarismos: 0 e 1. Assim, a cada posição de cada algarismo corresponde uma potência de 2. Os dígitos binários dependendo do posicionamento o algarismo ou bit terá um peso; o da extrema esquerda será o bit mais significativo (*most significant bit* – MSB) e o da extrema direita o bit menos significativo (*least significant bit* – LSB). Ex. $10111_{(2)} = 1x2^4 + 0x2^3 + 1x2^2 + 1x2^1 + 1x2^0 = 16 + 0 + 4 + 2 + 1 = 23$, ou seja, $10111_{(2)} = 23_{(10)}$ ou $10111_{(2)} = 23$

#### 2.1.2.1 Conversão de Decimal em Binário

Na conversão decimal-binário, podem ser utilizados dois métodos: o primeiro que é mais geral, dito das divisões sucessivas, consiste em dividir sucessivamente o número por 2 até obtermos o cociente 0 (zero). O resto dessa divisão colocado na ordem inversa corresponde ao número binário correspondente ao decimal dado (Fig. 2).

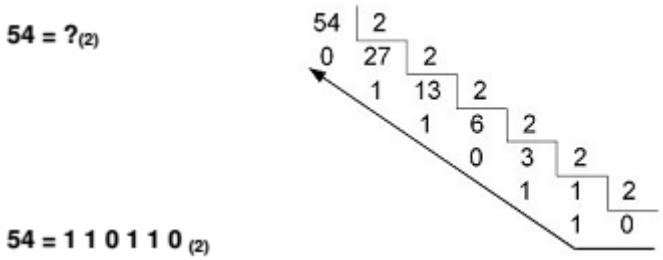



**Figura 2 - Conversão decimal para binário – método das divisões sucessivas**

O segundo método de conversão consiste em, começando como número decimal a ser convertido, extrair a maior potência de 2 (menor ou igual) possível. Repetindo este processo para o resto dessa subtração até que o resto seja zero. Concluindo, marque com o dígito 1 os expoentes utilizados e com o dígito zero os expoentes não utilizados (Fig. 3).

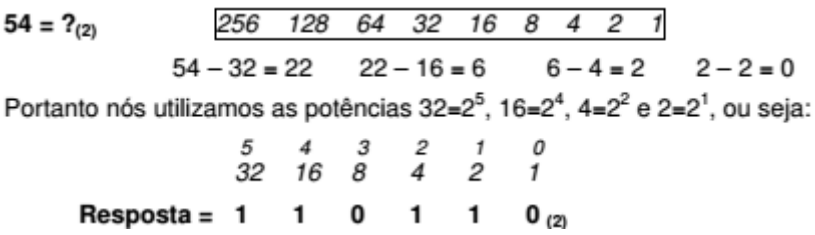



**Figura 3 - Conversão decimal para binário – método de conversão**

### 2.1.3 Sistema Octal

O sistema octal ou base 8 é composto por oito símbolos ou dígitos: 0, 1, 2, 3, 4, 5, 6, e 7. Os números binários, são mais apropriados para as máquinas ou computadores, mas para seres humanos são muito trabalhosos. Se considerarmos três dígitos binários, o maior que pode ser expresso por esses três dígitos é $111_{(2)}$ ou em decimal 7. Como o 7 é também o algarismo mais significativo do sistema octal, conclui-se que com a combinação de três dígitos binários pode-se ter o algarismo octal correspondente; daí também se pode dizer que os números octais têm um terço do comprimento de um número binário e fornecem a mesma informação.

#### 2.1.3.1 Conversão de Binário em Octal

É feita pela combinação de três dígitos binários, podendo assim ter todos os algarismos octais (Fig. 4):

$11011011_{(2)} = 11\ \ 011\ \ 011 = 3\ 3\ 3_{(8)} \rightarrow 11011011_{(2)} = 333_{(8)}$

$1011101_{(2)} = 1\ \ 011\ \ 101 = 1\ 3\ 5_{(8)} \rightarrow 1011101_{(2)} = 135_{(8)}$

**Figura 4 - Conversão de binário para octal**

#### 2.1.3.2 Conversão de Octal em Binário

A conversão de uma base em outra é bastante simples, uma vez que se trata da operação inversa, ou seja, basta converter individualmente cada dígito octal em três binários (Fig. 5).

$137_{(8)} = ?_{(2)}$

O número 1 equivale a $001_{(2)}$, o número 3 igual a $011_{(2)}$ e o número 7 vale $111_{(2)}$.

Portanto:

$137_{(8)} = 001011111_{(2)}$, ou seja, $137_{(8)} = 1011111_{(2)}$.

**Figura 5 - Conversão de octal para binário**

#### 2.1.3.3 Conversão de Octal em Decimal

Esta conversão passa pela conversão em binário e posteriormente em decimal, conforme a Figura 6:

$17_{(8)} = ?$

$17_{(8)} \rightarrow 001\ 111_{(2)} \rightarrow 1x2^3 + 1x2^2 + 1x2^1 + 1x2^0 \rightarrow 8 + 4 + 2 + 1 = 15.$

**Figura 6 - Conversão de octal em decimal**

#### 2.1.3.4 Conversão de Decimal em Octal

Neste caso, devemos passar pelo sistema binário e depois para octal (Fig.7).

$22 = ?_{(8)}$

$22 \rightarrow 10110_{(2)} \rightarrow 10\ 110_{(2)} \rightarrow 26_{(8)}$, ou seja, $22 = 26_{(8)}$.

**Figura 7 - Conversão de decimal em octal**

### 2.1.4 Sistema Hexadecimal

O sistema hexadecimal foi criado também para minimizar a representação de um número binário nos sistemas computacionais. Analogamente, se considerarmos quatro dígitos binários, o maior número que pode ser expresso por esses quatro dígitos é 1111 ou em decimal 15, da mesma forma que 15 é o algarismo mais significativo do sistema hexadecimal, portanto com a combinação de 4 bits ou dígitos binários pode-se ter o algarismo hexadecimal correspondente. Assim, com esse agrupamento de 4 bits ou dígitos, podem-se definir 16 símbolos, de 0 até 15. Contudo, como não existem símbolos dentro do sistema arábico que possam representar os números decimais entre 10 e 15 sem repetir os símbolos anteriores, foram usadas as letras A, B, C, D, E, F.

Portanto, o sistema hexadecimal é formado por 16 símbolos alfanuméricos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

#### 2.1.4.1 Conversão de Hexadecimal em Binário

Basta converter cada dígito hexadecimal em seu similar binário, ou seja, cada dígito em hexa equivale a um grupo de 4 bits (Fig. 8).

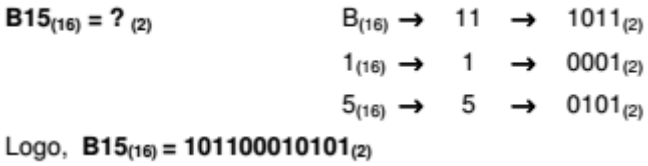



**Figura 8 - Conversão hexadecimal em binário**

#### 2.1.4.2 Conversão de Binário em Hexadecimal

De maneira análoga, basta realizar o processo inverso de hexadecimal para binário (Fig. 9).

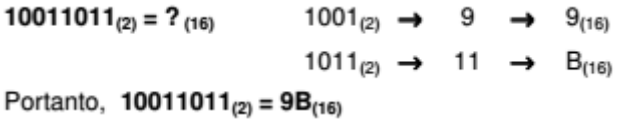



**Figura 9 - Conversão binário em hexadecimal**

Na Tabela 2 temos a correspondência de conversão entre os sistemas.

**Tabela 2 - Conversão entre os diferentes sistemas numéricos**

| TABELA DE CONVERSÃ0 | | | |
|---|---|---|---|
| Decimal | Binário | Octal | Hexadecimal |
| 0 | 0000 | 0 | 0 |
| 1 | 0001 | 1 | 1 |
| 2 | 0010 | 2 | 2 |
| 3 | 0011 | 3 | 3 |
| 4 | 0100 | 4 | 4 |
| 5 | 0101 | 5 | 5 |
| 6 | 0110 | 6 | 6 |
| 7 | 0111 | 7 | 7 |
| 8 | 1000 | | 8 |
| 9 | 1001 | | 9 |
| 10 | 1010 | | A |
| 11 | 1011 | | B |
| 12 | 1100 | | C |
| 13 | 1101 | | D |
| 14 | 1110 | | E |
| 15 | 1111 | | F |

## 3. Sistema Digital

É mais simples para o ser humano trabalhar com valores numéricos na base decimal, mas um sistema computacional trabalha de maneira diferente. Um sistema digital é um sistema matemático que define informações como valores numéricos. Dessa forma, é possível definir operações digitais como cálculos matemáticos.
Em analogia ao sistema decimal, onde cada dígito possui 10 valores possíveis, um sistema digital é um sistema binário, onde cada dígito possui apenas 2 valores possíveis. Esses dois valores são definidos como "níveis lógicos" e adota-se o valor de 0 (zero) ou 1 (um) apenas. Transportando esse sistema para um sistema computacional é necessário apresentar esses dois valores como sinais elétricos. Para tanto, podemos entendê-los como:

- Ligado ou desligado;
- Nível alto ou nível baixo;
- Alimentado ou em zero;
- VCC ou Terra.

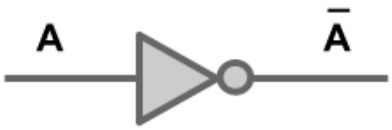
Ao impormos nas entradas níveis lógicos determinados (nível 1 ou nível 0 para cada uma das entradas), vamos obter na saída o resultado da combinação dessas entradas. É por esta razão que se chamam circuitos de decisão ou combinatórios, uma vez que para cada combinação de valores 1 e 0 nas entradas se obtém à saída um nível lógico que é determinado pela estrutura do circuito.
Existem três circuitos lógicos básicos, chamados genericamente portas ou “gates”: a porta “e” (and), a porta “ou” (or) e o “inversor” ou “negação” (inverter ou not). As operações observáveis para esses níveis lógicos são definidas como operações lógicas. Todas as possíveis operações lógicas são baseadas em apenas três operações primárias:

- Inversão;
- Soma lógica;
- Produto lógico.

Portanto, a manipulação de informação binária em um computador é feita por meio de circuitos lógicos, chamados portas lógicas. As portas lógicas básicas são:

### 3.1 Porta “Não” (Inversor, NOT)

A porta NOT inverte o sinal de entrada (executa a NEGAÇÃO do sinal de entrada), ou seja, se o sinal de entrada for 0 ela produz uma saída 1, se a entrada for 1 ela produz uma saída 0 (Fig. 10).

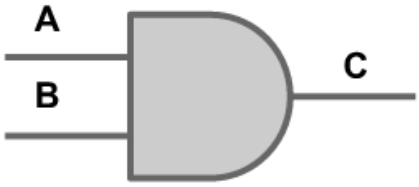



| A | $\bar{A}$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

**Figura 10 - Porta NOT**

## 3.2 Porta E (AND)

A porta AND combina dois ou mais sinais de entrada de forma equivalente a um circuito em série, para produzir um único sinal de saída, ou seja, ela produz uma saída 1, se todos os sinais de entrada forem 1. Caso qualquer um dos sinais de entrada seja 0, a porta AND produzirá um sinal de saída igual a zero (Fig. 8).




| A | B | C |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

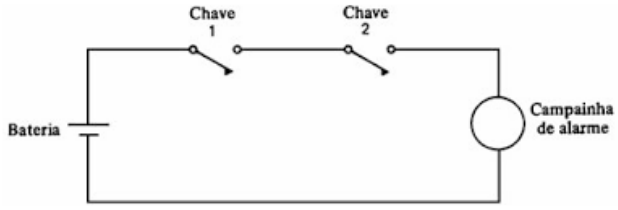



**Figura 11 - Porta AND**

## 3.3 Porta OU (OR)

A porta OR combina dois ou mais sinais de entrada de forma equivalente a um circuito em paralelo, para produzir um único sinal de saída, ou seja, ela produz uma saída 1, se qualquer um dos sinais de entrada for igual a 1. A porta OR produzirá um sinal de saída igual a zero apenas se todos os sinais de entrada forem 0 (Fig.9).

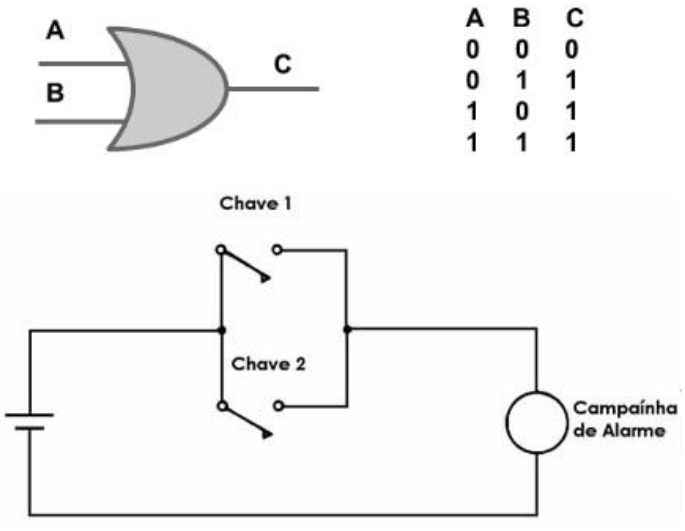



**Figura 12 - Porta OR**

### 3.4 Porta "Não-E" (NAND)

A porta NAND equivale a uma porta AND seguida por uma porta NOT, isto é, ela produz uma saída que é o inverso da saída produzida pela porta AND (Fig. 10).

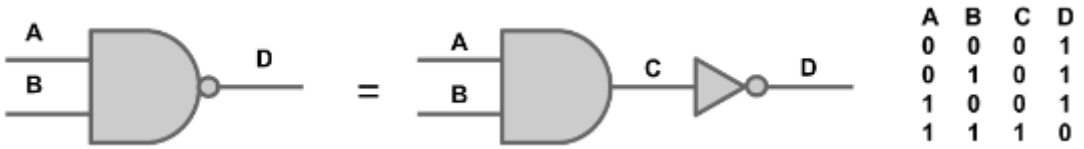
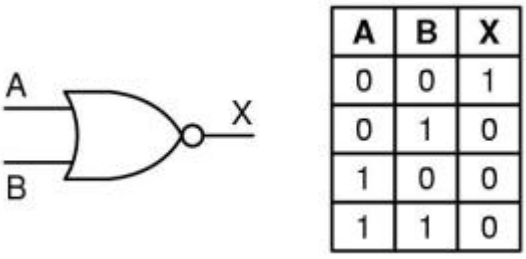



**Figura 13 - Porta NAND**

### 3.5 Porta "Não-OU" (NOR)

A porta NOR equivale a uma porta OR seguida por uma porta NOT, isto é, ela produz uma saída que é o inverso da saída produzida pela porta OR (Fig. 11).




| A | B | X |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

**Figura 14 - Porta NOR**

### 3.6 Porta OU Exclusivo (XOR)

A porta XOR compara os bits; ela produz saída 0 quando todos os bits de entrada são iguais e saída 1 quando pelo menos um dos bits de entrada é diferente dos demais (Fig. 12).

| A | B | XOR |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

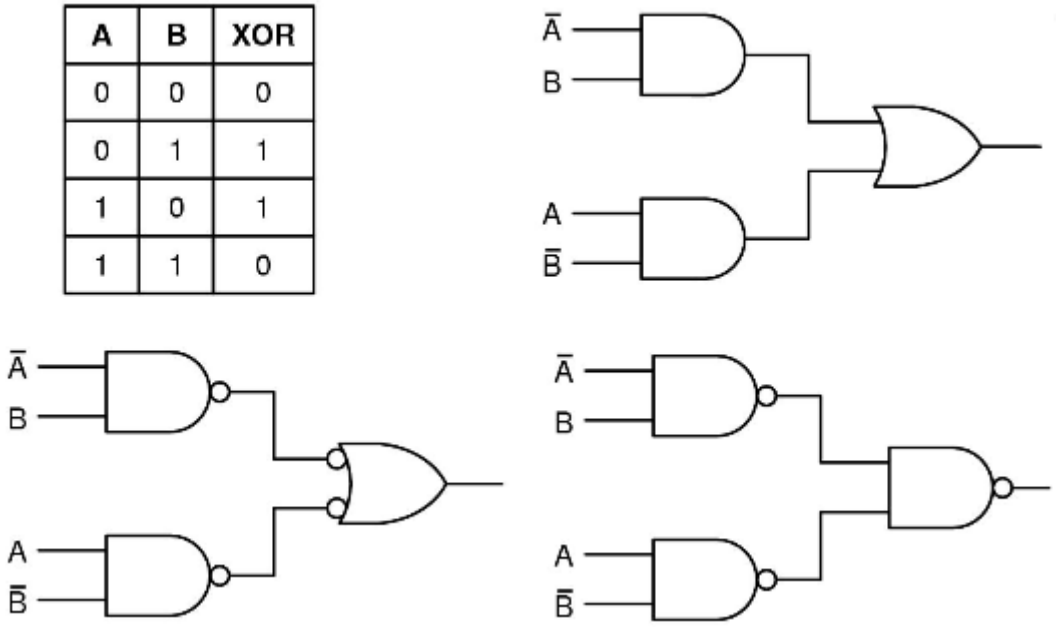
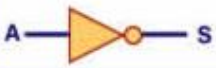
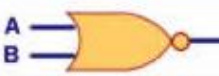



**Figura 15 - Portas XOR**

Na Tabela 3, temos um resumo das portas apresentadas e sua representação algébrica.

**Tabela 3 - Resumo das portas lógicas apresentadas**

| NOME | Símbolo Gráfico | Símbolo Algébrico |
|---|---|---|
| NOT |  | $S = \overline{A}$ ou $S = A'$ |
| AND |  | $S = A \cdot B$ ou $S = AB$ |
| OR |  | $S = A + B$ |
| NAND |  | $S = \overline{(A\,B)}$ |
| NOR |  | $S = \overline{(A + B)}$ |
| XOR |  | $S = A \oplus B$ |

## 4. Álgebra Booleana

Para descrever os circuitos que podem ser construídos pela combinação de portas lógicas, um novo tipo de álgebra é necessário, onde as variáveis e funções podem ter apenas valores 0 e 1. Tal álgebra é denominada álgebra booleana.
George Boole nasceu na Inglaterra em 1815 e, em 1847, publicou um trabalho científico sob o título "*The Mathematical Analysis of Logic*", onde introduz os conceitos de lógica simbólica demonstrando que a lógica podia ser representada por equações algébricas. Este trabalho se tornou fundamental para a construção e programação dos modernos computadores eletrônicos.
Na Álgebra de Boole existem apenas três operadores: E, OU e NÃO (AND, OR, NOT). Estas três funções são as únicas operações necessárias para efetuar comparações ou as quatro operações aritméticas base.
Em 1937, Claude Shannon estabeleceu a relação entre a Álgebra de Boole e os circuitos eletrônicos transferindo os dois estados lógicos (SIM e NÃO) para os diferentes estados no circuito.
Uma função booleana tem uma ou mais variáveis de entrada e fornece somente um resultado na saída, o qual depende apenas dos valores destas variáveis. Como uma função de n variáveis possui apenas 2n conjuntos possíveis de valores de entrada, a função pode ser descrita completamente através de uma tabela de 2n linhas, cada linha mostrando o valor da função para uma combinação diferente dos valores de entrada. Tal tabela é denominada tabela verdade.

### 4.1 Operação OU (Adição Lógica)

Uma definição para a operação OU, que também é denominada adição lógica, é: "A operação OU resulta 1 se pelo menos uma das variáveis de entrada vale 1". Como uma variável Booleana ou vale 1 ou vale 0, e como o resultado de uma operação qualquer pode ser encarado como (ou atribuído a) uma variável Booleana, basta que definamos quando a operação vale 1. Automaticamente, a operação resultará 0 nos demais casos. Assim, pode-se dizer que a operação OU resulta 0 somente quando todas as variáveis de entrada valem 0.
Um símbolo possível para representar a operação OU é "+", tal como o símbolo da adição algébrica. Porém, como estamos trabalhando com variáveis Booleanas, sabemos que não se trata da adição algébrica, mas sim da adição lógica. Outro símbolo também encontrado na bibliografia é "V".
Listando as possibilidades de combinações entre dois valores Booleanos e os respectivos resultados para a operação OU, tem-se:

$$\begin{aligned} 0+0 &= 0 \\ 0+1 &= 1 \\ 1+0 &= 1 \\ 1+1 &= 1 \end{aligned}$$

Note que a operação OU só pode ser definida se houver, pelo menos, duas variáveis envolvidas. Ou seja, não é possível realizar a operação sobre somente uma variável. Devido a isso, o operador "+" (OU) é dito binário.
Nas equações, não se costuma escrever todas as possibilidades de valores. Apenas adotamos uma letra (ou uma letra com um índice) para designar uma variável Booleana. Com isso, já se sabe que aquela variável pode assumir ou o valor 0 ou o valor 1.
Então, supondo que queiramos demonstrar o comportamento da equação A+B (lê-se A ou B), poderíamos fazê-lo utilizando uma tabela verdade, como segue:

| A | B | A+B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

Da mesma forma, podemos mostrar o comportamento da equação A+B+C (lê-se A ou B ou C) por meio de uma tabela verdade. Como na equação há somente o símbolo "+", trata-se da operação OU sobre três variáveis. Logo, pode-se aplicar diretamente a definição da operação OU: o resultado será 1 se pelo menos uma das variáveis de entrada valer 1.

| A | B | C | A+B+C |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

É importante notar que, devido ao fato de haver somente um operador na equação, pode-se também avaliar a equação decompondo-a em pares. Por exemplo, pode-se primeiramente achar o resultado de A+B, para depois operar os valores resultantes com os respectivos valores de C. Esta propriedade é conhecida como associativa. Também a ordem em que são avaliadas as variáveis A, B e C é irrelevante (propriedade comutativa).
Estas propriedades são ilustradas pela tabela verdade a seguir. Nela, os parêntesis indicam subexpressões já avaliadas em coluna imediatamente à esquerda. Note que os valores das colunas referentes às expressões A+B+C, (A+B)+C e (B+C)+A são os mesmos (na mesma ordem).

| A B C | A+B+C | A+B | (A+B)+C | B+C | (B+C)+A |
|---|---|---|---|---|---|
| 0 0 0 | 0 | 0 | 0 | 0 | 0 |
| 0 0 1 | 1 | 0 | 1 | 1 | 1 |
| 0 1 0 | 1 | 1 | 1 | 1 | 1 |
| 0 1 1 | 1 | 1 | 1 | 1 | 1 |
| 1 0 0 | 1 | 1 | 1 | 0 | 1 |
| 1 0 1 | 1 | 1 | 1 | 1 | 1 |
| 1 1 0 | 1 | 1 | 1 | 1 | 1 |
| 1 1 1 | 1 | 1 | 1 | 1 | 1 |

### 4.2 Operação E (Multiplicação Lógica)

A operação E, ou multiplicação lógica, pode ser definida da seguinte forma: “A operação E resulta 0 se pelo menos uma das variáveis de entrada vale 0”. Pela definição dada, pode-se deduzir que o resultado da operação E será 1 se, e somente se, todas as entradas valerem 1. O símbolo usualmente utilizado na operação E é “·”, porém outra notação possível é “∧”.
Podemos, também, listar as possibilidades de combinações entre dois valores Booleanos e os respectivos resultados, para a operação E:

$$\begin{aligned} 0 \cdot 0 &= 0 \\ 0 \cdot 1 &= 0 \\ 1 \cdot 0 &= 0 \\ 1 \cdot 1 &= 1 \end{aligned}$$

Assim como a operação OU, a operação E só pode ser definida entre, pelo menos duas variáveis. Ou seja, o operador “·” (E) também é binário. Para mostrar o comportamento da equação A · B (lê-se A e B), escreve-se uma tabela verdade, como segue:

| A | B | A·B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

De forma semelhante, pode-se determinar o resultado da equação A·B·C (lê-se A e B e C) utilizando diretamente a definição da operação E: o resultado será 0 se pelo menos uma das variáveis de entrada for 0.

| A B C | A·B·C |
|---|---|
| 0 0 0 | 0 |
| 0 0 1 | 0 |
| 0 1 0 | 0 |
| 0 1 1 | 0 |
| 1 0 0 | 0 |
| 1 0 1 | 0 |
| 1 1 0 | 0 |
| 1 1 1 | 1 |

Também para a operação E valem as propriedades associativa e comutativa. Então, a equação A.B.C pode ainda ser avaliada tomando-se as variáveis aos pares, em qualquer ordem. Veja a tabela verdade a seguir e compare os resultados.

| A B C | A·B·C | A·B | (A·B)·C | B·C | A·(B·C) |
|---|---|---|---|---|---|
| 0 0 0 | 0 | 0 | 0 | 0 | 0 |
| 0 0 1 | 0 | 0 | 0 | 0 | 0 |
| 0 1 0 | 0 | 0 | 0 | 0 | 0 |
| 0 1 1 | 0 | 0 | 0 | 1 | 0 |
| 1 0 0 | 0 | 0 | 0 | 0 | 0 |
| 1 0 1 | 0 | 0 | 0 | 0 | 1 |
| 1 1 0 | 0 | 1 | 0 | 0 | 0 |
| 1 1 1 | 1 | 1 | 1 | 1 | 1 |

### 4.3 Complementação (ou Negação, ou Inversão)

A operação complementação dispensa uma definição. É a operação cujo resultado é simplesmente o valor complementar ao que a variável apresenta. Também devido ao fato de uma variável Booleana poder assumir um entre somente dois valores, o valor complementar será 1 se a variável vale 0 e será 0 se a variável vale 1.
Os símbolos utilizados para representar a operação complementação sobre uma variável Booleana A são Ā, ~A e A' (lê-se A negado). O resultado da operação complementação pode ser listado:

$$\overline{0} = 1$$
$$\overline{1} = 0$$

Diferentemente das operações OU e E, a complementação só é definida sobre uma variável, ou sobre o resultado de uma expressão. Ou seja, o operador complementação é dito unário. E a tabela verdade para A é:

| A | $\overline{A}$ |
|---|---|
| 0 | 1 |
| 1 | 0 |