Pontifícia Universidade Católica do Rio Grande do Sul
Faculdade de Engenharia
Disciplina de Lógica Computacional Aplicada

Prof. Dr. Fabian Vargas

Índice

2. PORTAS LÓGICAS E EXPRESSÕES BOOLEANAS BÁSICAS

1. SISTEMAS NUMÉRICOS

1.1. Caracterização dos Sistemas Numéricos

Todos nós, quando ouvimos pronunciar a palavra números, automaticamente a associamos ao sistema decimal com o qual estamos acostumados a operar. Este sistema está fundamentado em certas regras que são base para qualquer outro. Vamos, portanto, estudar estas regras e aplicá-las aos sistemas de numeração binária, octal e hexadecimal. Estes sistemas são utilizados em computadores digitais, circuitos lógicos em geral e no processamento de informações dos mais variados tipos.

1.2. Sistemas Numéricos em uma Base B Qualquer

1.2.1. Sistema de Numeração Decimal

Entre os sistemas numéricos existentes, o sistema decimal é o mais utilizado. 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.

Dígitos: 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9

Base 10

Organização posicional:

$486 = 400 + 80 + 6 = 4 \times 100 + 8 \times 10 + 6 \times 1 = 4 \times 10^2 + 8 \times 10^1 + 6 \times 10^0$, ou seja:

$486 = 4 \times 10^2 + 8 \times 10^1 + 6 \times 10^0$

Note que aquele algarismo situado na extrema esquerda do número está sendo multiplicado pela potência de dez maior, ou seja, é o dígito mais significativo (*most significant digit* – **MSD**).

Analogamente, o que está situado na extrema direita será multiplicado pela menor potência, ou seja, é o dígito menos significativo (*least significant digit* – **LSD**).

1.2.2. Sistema de Numeração Binário

Os atuais computadores processam suas operações em um sistema diferente do decimal, o sistema binário.

O sistema binário corresponde a qualquer conjunto dual, como por exemplo: não e sim; falso e verdadeiro; desligado e ligado; negativo e positivo falso e verdadeiro, etc. Nos circuitos lógicos, 0 e 1 representam respectivamente níveis de tensão baixa e alto ou estados de saturação e corte de transistores. Daí, uma outra designação comum: L e H (*Low* e *High levels* do inglês: baixo e alto níveis de tensão).

Na seqüência binária, cada digito é chamado de BIT (*Binary Digit*).

Dígitos: 0 e 1

Base 2

Organização posicional:

$10101 = 1x2^4 + 0x2^3 + 1x2^2 + 0x2^1 + 1x2^0$

Números são expressos como somas de potências de 2 (a base do sistema binário)

MSB (bit mais significativo): Bit mais a esquerda.

LSB (bit menos significativo): Bit mais a direita.

Agrupamento de dados:

- ⇨ 4 bits: *NIBBLE*.
- ⇨ 8 bits: *BYTE.*

1.2.3. Sistema Octal

É utilizado por ser um sistema que tem relação direta com o sistema binário.

Para representação de um número no sistema octal, considera-se três dígitos binários. Assim, o maior dígito que pode ser representado neste sistema é 111 ou em decimal 7.

Dígitos: 0, 1, 2, 3, 4, 5, 6 e 7

Base 8

Organização posicional:

$10_8 = 1x8^1 + 0x8^0 = 8+0 = 8_{10}$

Números são expressos como somas de potências de 8 (a base do sistema octal)

MSB (bit mais significativo): Bit mais a esquerda.

LSB (bit menos significativo): Bit mais a direita.

1.2.4. Sistema Hexadecimal

O sistema hexadecimal (hexa) foi criado com o mesmo propósito do sistema octal, para minimizar a representação de um número binário que é o utilizado em processamento. Tanto os números em hexa como em octal são os meios de manipulação do homem, porém existirão sempre conversores internos à máquina que os converta em binário, com o qual a máquina trabalha.

Analogamente, se considerarmos quatro dígitos ou bits binários, o maior número que se 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 grupamento de 4 bits ou dígitos, podem-se definir 16 símbolos, 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 usados os símbolos A, B, C, D, E e F, portanto o sistema hexadecimal será formato por 16 símbolos alfanuméricos.

Dígitos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E e F.

Base: 16

Organização posicional:

$10_{16} = 1x16^1 + 0x16^0 = 16+0 = 16_{10}$

MSB (bit mais significativo): Bit mais a esquerda.

LSB (bit menos significativo): Bit mais a direita.

1.3. Conversão de Números Inteiros

1.3.1. Conversão de Binário em Decimal

Exemplo: $\mathbf{10111_2} = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 16 + 0 + 4 + 2 + 1 = 23$

Exemplo: $\mathbf{10010110_2} = 1 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 128 + 0 + 0 + 16 + 0 + 4 + 2 + 0 = 150$

1.3.2. Conversão de Decimal em Binário

Na conversão decimal-binário pode ser utilizado o método dito das divisões sucessivas, consiste m dividir sucessivamente o número por 2 até obtermos o cociente 0 (zero). O resto dessa divisão colocado na ordem inversa corresponde ao número binário, resultado da conversão de decimal em binário de um certo número de dados.

Exemplo: $54_{10} \Rightarrow 54_2$

```
54 |_2_
 0  27 |_2_
     1  13 |_2_
         1   6 |_2_
             0   3 |_2_
                 1   1 |_2_
                     1   0
```

54_{10} = **110110_2**

1.3.3. Conversão de Octal em Binário

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

Exemplo: $137_8 = ?_2$

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 **$137_8 = 1011111_2$**

1.3.4. Conversão de Binário em Octal

É feita pela combinação de três dígitos binários, como vimos, podendo assim ter todos os algarismos octais:

Exemplo: **11011011_2** = **11 011 011** = **3 3 3 $_8$** $\Rightarrow$ **$11011011_2 = 333_8$**

Exemplo: **1011101_2** = **1 011 101 = 1 3 5 $_8$** $\Rightarrow$ **$1011101_2 = 135_8$**

1.3.5. Conversão de Octal em Decimal

Esta conversão se passa primeiramente de octal para binário e posteriormente para decimal, ou seja:

Exemplo: 17_8 = $001\ 111_2$ $\longrightarrow$ **$1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 8 + 4 + 2 + 1 = 15_{10}$**

1.3.6. Conversão Decimal para Octal

Esta conversão se passa primeiramente de decimal para binário e posteriormente para octal, ou seja:

Exemplo: $22_{10} = 10110_2$ $\longrightarrow$ **$10\ 110 = 26_8$**

1.3.7. Conversão Hexa para Binário

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

Exemplo: $B15_{16} = ?_2$
B $\longrightarrow$ 1011_2
1 $\longrightarrow$ 0001_2
5 $\longrightarrow$ 0101_2
Logo, **$B15_{16} = 1011.0001.0101_2$**

1.3.8. Conversão Binária para Hexa

De maneira análoga, basta realizar o processo inverso de hexa para binário.
Exemplo: $10011011_2 = ?_{16}$
1001_2 $\longrightarrow$ 9_{16}
1011_2 $\longrightarrow$ B_{16}
Portanto, **$10011011_2 = 9B_{16}$**

1.4. Conversão de Números Fracionários

1.4.1. Conversão de Números Binários Fracionários em Decimal

A conversão segue o mesmo processo binário para decimal já visto, utilizando a mesma expressão, inclusive os dígitos após a vírgula em que as potências ficam com o expoente negativo.

Exemplo: $110{,}11_2 = 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} = 4 + 2 + 0 + 1 \times 1/2^1 + 1 \times 1/2^2 = 4 + 2 + 0 + 0{,}5 + 0{,}25 = 6{,}75$

Portanto: $110{,}11_2 = 6{,}75$

1.4.2. Conversão de Número Decimal Fracionário em Binário

Neste tipo de conversão, o processo é dividido em duas etapas: conversão da parte inteira (já estudada) e conversão da parte fracionária.

Exemplo: 6,6
6 = parte inteira
0,6 = parte fracionária

A parte inteira do número é convertida conforme o processo já demonstrado anteriormente, e obtemos o número **110_2**. A parte fracionária, **0,6**, é convertida da seguinte maneira:

Multiplica-se a parte fracionária (multiplicando) pela base "b" (multiplicador), neste caso o 2, e separa-se a parte inteira do produto. O resultado obtido da subtração da parte inteira do produto passa a ser o próximo multiplicando. Faz-se sucessivamente esta operação até que consiga uma precisão satisfatória. Lê-se os algarismos separados de cima para baixo.

Exemplo:

$4,6_{10} = ?_2$

0,6 x 2=1,2

menos a parte inteira (**1**) = 0,2

x 2 = 0,4

menos a parte inteira (**0**) = 0,4

x 2 = 0,8

menos a parte inteira (**0**) = 0,8

x 2 = 1,6

menos a parte inteira (**1**) = 0,6

x 2 = 1,2

menos a parte inteira (**1**) = 0,2

e assim por diante.

Lendo de cima para baixo teremos **10011**, então **$0,6_{10}=10011_2$**. Se fizermos uma conferência, descobriremos que $0,10011_2$ é igual a:

$$1x2^{-1} + 0x2^{-2} + 0x2^{-3} + 1x2^{-4} + 1x2^{-5} = ½ + 1/16 + 1/32 = 19/32 = 0,59375 \approx 0,6.$$

Portanto, como podemos perceber, **teremos sempre diferenças de precisão entre bases**.

2. PORTAS LÓGICAS E EXPRESSÕES BOOLEANAS BÁSICAS

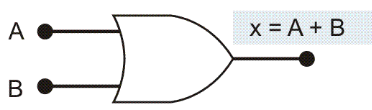
2.1. PORTAS LÓGICAS

2.1.1.PORTA OR

OR

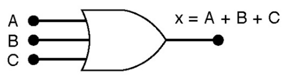
A	B	x = A + B
0	0	0
0	1	1
1	0	1
1	1	1

(a)

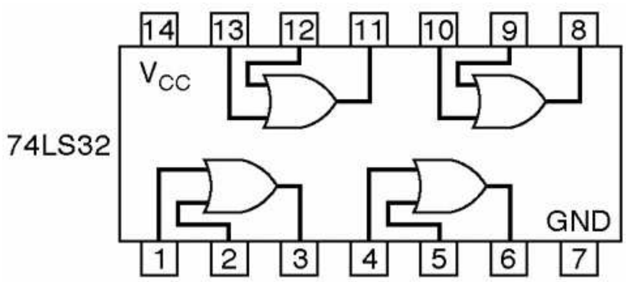


Porta OR

(b)



A	B	C	x = 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



Aplicações com portas OR:

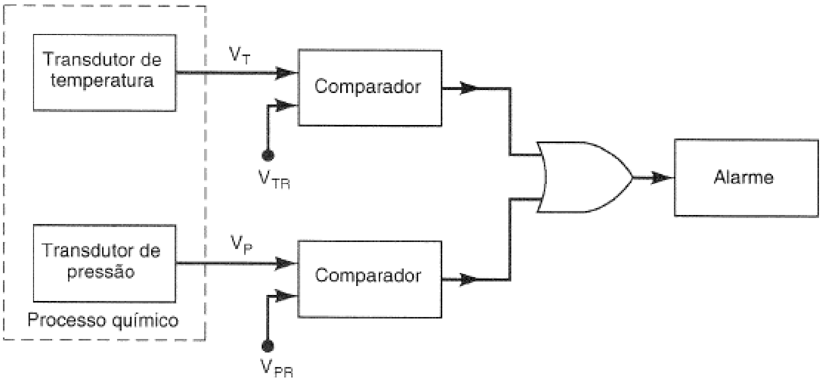
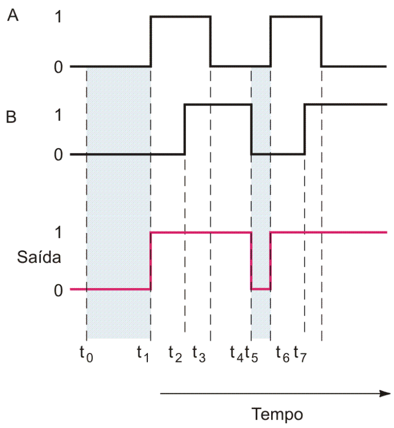
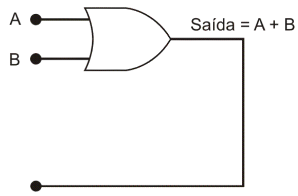
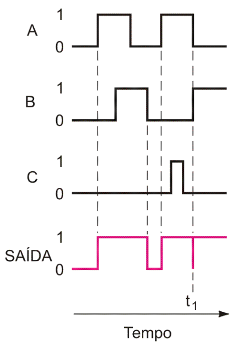


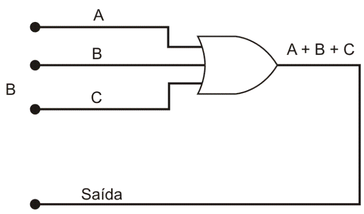
Fig. 3-4 Exemplo de utilização da porta OR em um sistema de alarme.

Dadas as entradas que variam no tempo, determine a saída da porta OR:





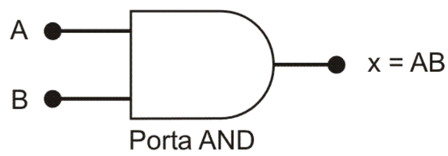




2.1.2. PORTA AND

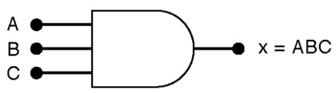
AND

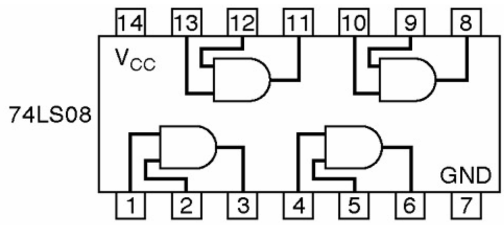
A	B	x = A · B
0	0	0
0	1	0
1	0	0
1	1	1



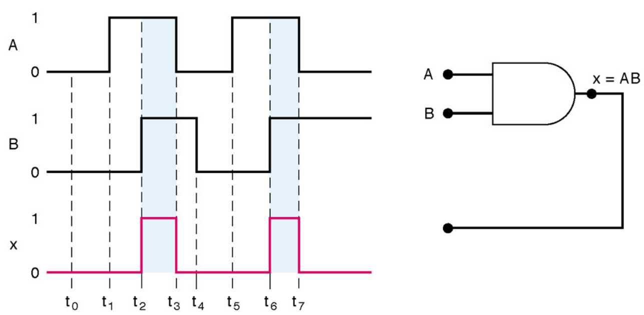
(a) (b)

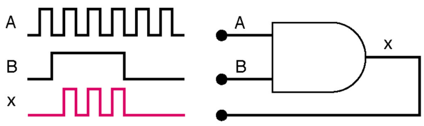
A	B	C	x = ABC
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1





Dadas as entradas que variam no tempo, determine a saída da porta AND:



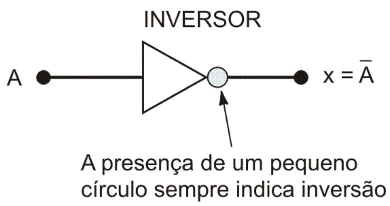


2.1.3. PORTA NOT

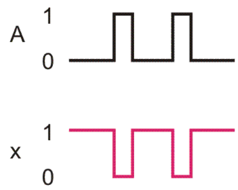
INVERSOR

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

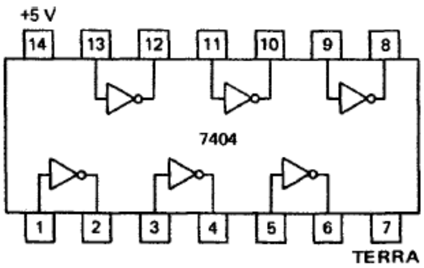
(a)



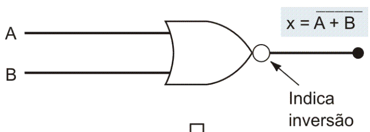
(b)



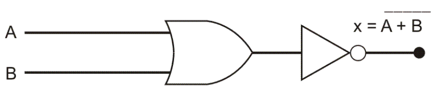
(c)



2.1.4. PORTA NOR



(a)

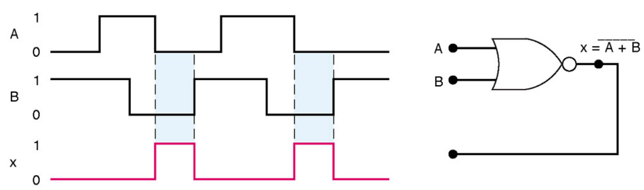


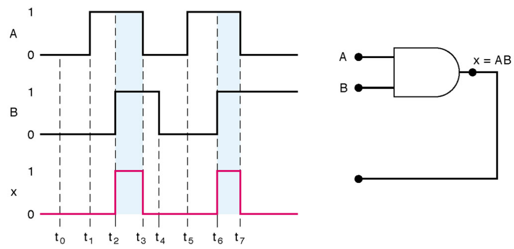
(b)

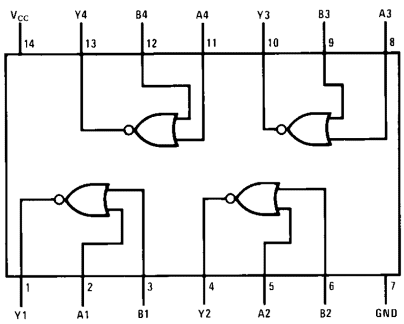
A	B	OR: $A + B$	NOR: $\overline{A + B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

(c)

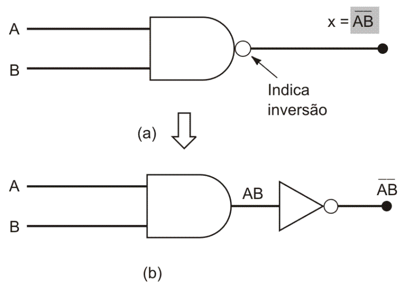
Dadas as entradas que variam no tempo, determine a saída da porta OR:





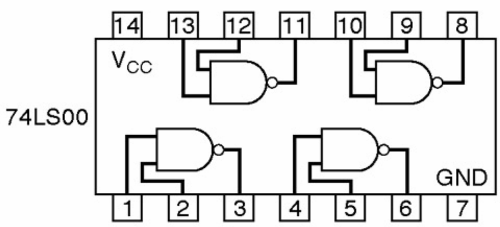


2.1.5. PORTA NAND

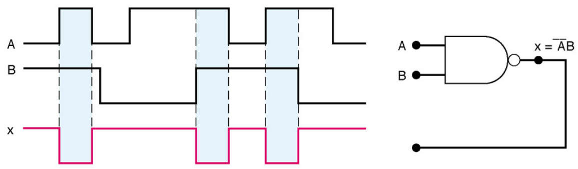


	AND	NAND
A B	AB	$\overline{AB}$
0 0	0	1
0 1	0	1
1 0	0	1
1 1	1	0

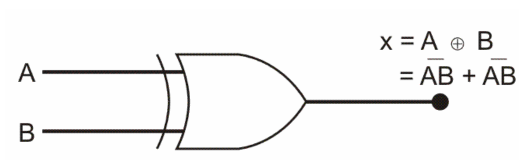
(c)



Determine a saída da porta NAND onde é dado as entradas que variam no tempo:

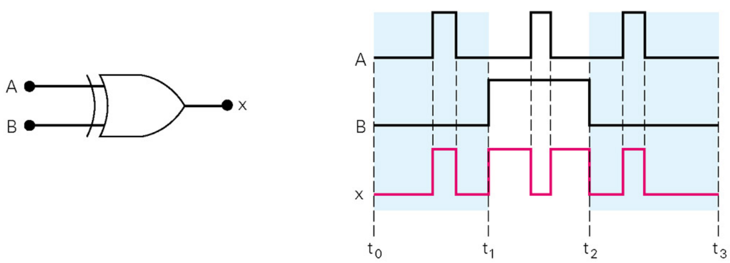


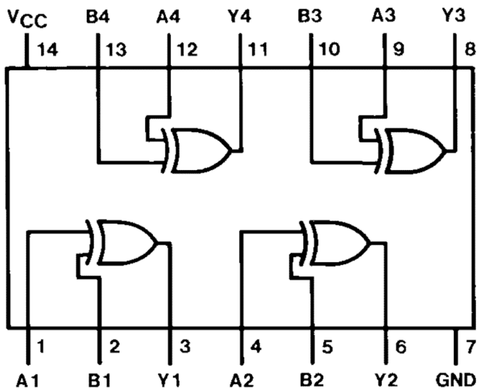
2.1.6. PORTA XOR



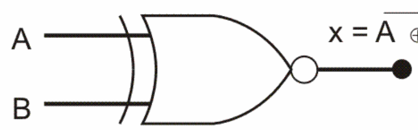
A	B	x
0	0	0
0	1	1
1	0	1
1	1	0

Determine a saída da porta XOR onde é dado as entradas que variam no tempo:



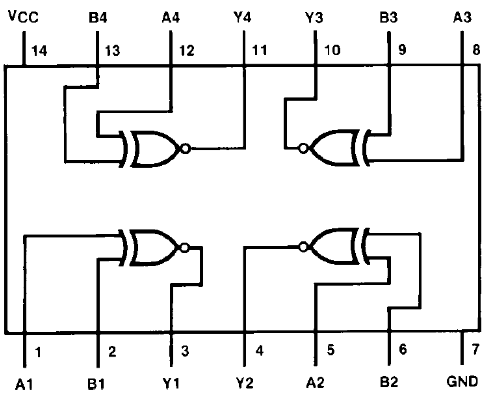


2.1.7. PORTA XNOR



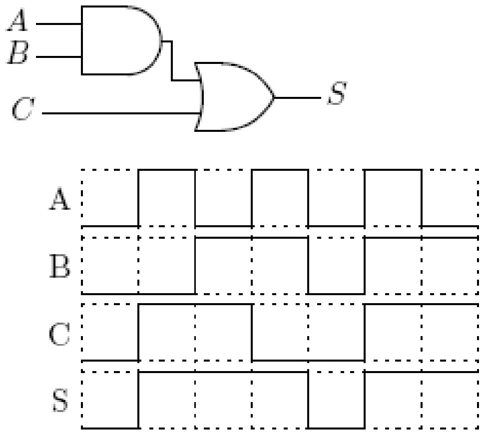
$$x = \overline{A \oplus B} = AB + \overline{A}\,\overline{B}$$

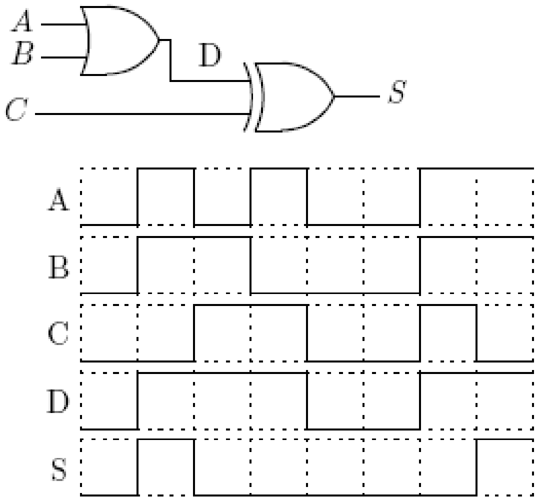
A	B	x
0	0	1
0	1	0
1	0	0
1	1	1

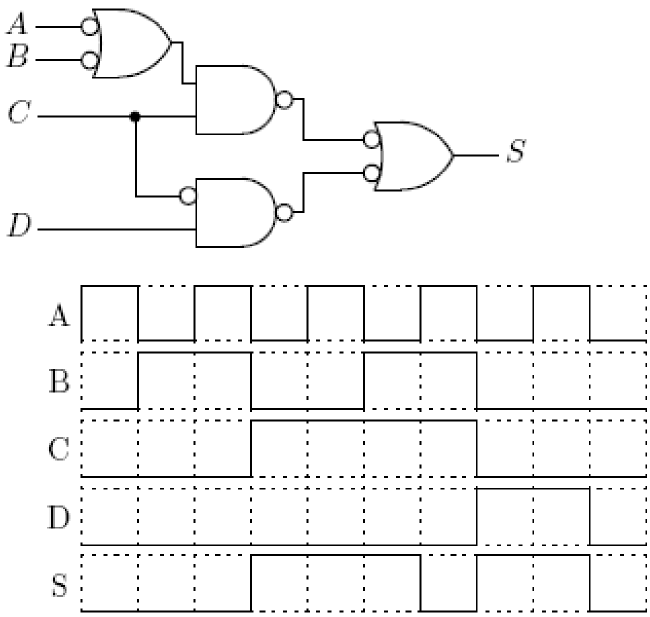


2.2. DIAGRAMAS DE TEMPO

Determine o diagrama de tempo dos seguintes circuitos lógicos:



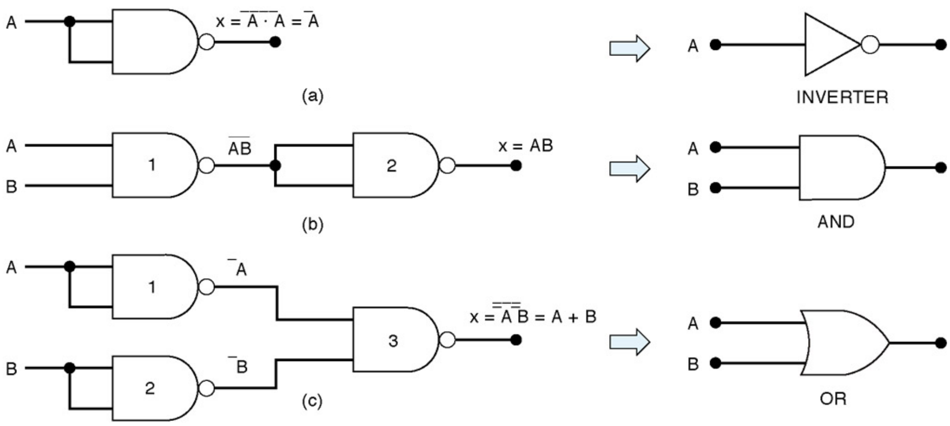




2.3. UNIVERSALIDADE DAS PORTAS NAND E NOR

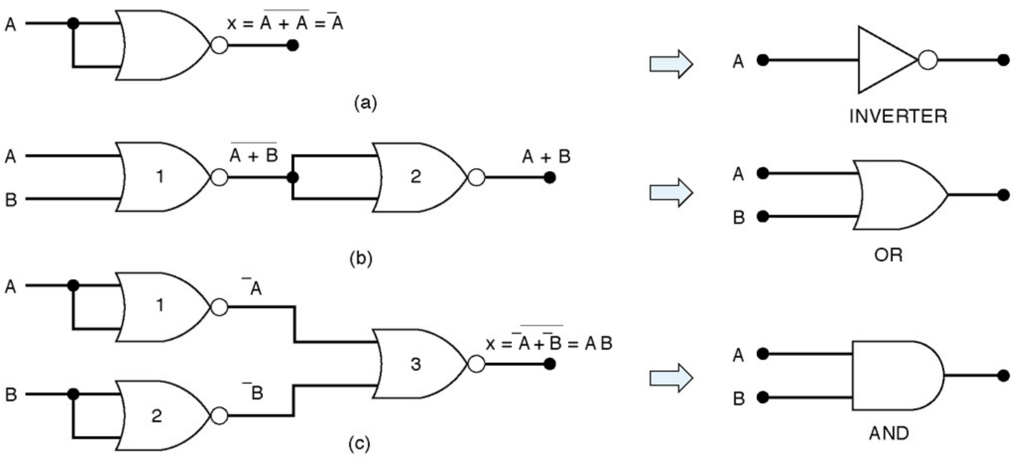
2.3.1. Universalidade das Portas NAND

As portas NAND podem ser usadas para implementar qualquer função booleana.



2.3.2. Universalidade das Portas NOR

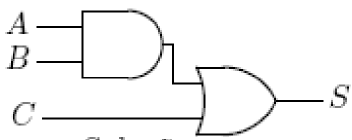
As portas NOR podem ser usadas para implementar qualquer função booleana.



2.4. EXPRESSÕES BOOLEANAS

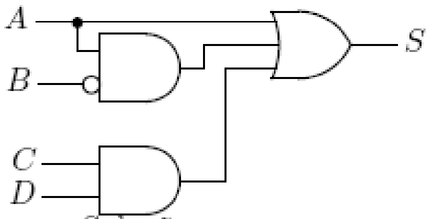
2.4.1. Equação Booleana a partir do Diagrama de Portas Lógicas

Exemplo 1:



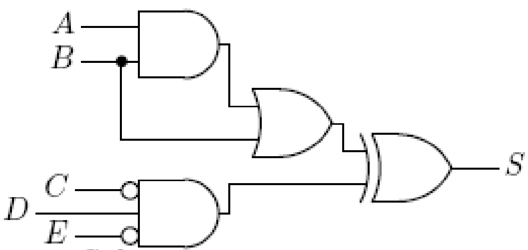
Solução:
$M = A\,B$
$S = M + C$
$\boxed{S = A\,B + C}$

Exemplo 2:



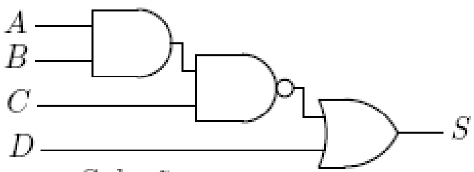
Solução:
$M = A\,\overline{B}$
$N = C\,D$
$S = A + M + N$
$\boxed{S = A + A\,\overline{B} + C\,D}$

Exemplo 3:



Solução:
$M = A\,B$
$N = B + M$
$P = \overline{C}\,D\,\overline{E}$
$S = N \oplus P$
$\boxed{S = (B + A\,B) \oplus (\overline{C}\,D\,\overline{E})}$

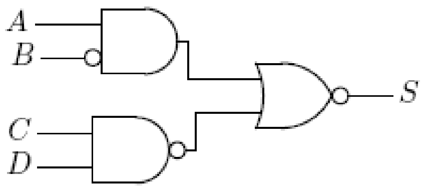
Exemplo 4:



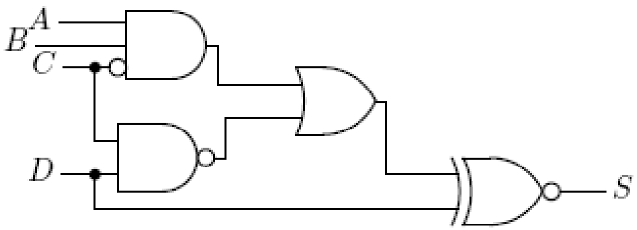
Solução:
$M = A\,B$
$N = \overline{M\,C}$
$S = N + D$
$\boxed{S = \overline{A\,B\,C} + D}$

2.5. Obtenção do Circuito a partir das Equações Booleanas

Exemplo 1: $f(A,B,C,D) = \overline{A\,\overline{B} + \overline{C\,D}}$
Solução:

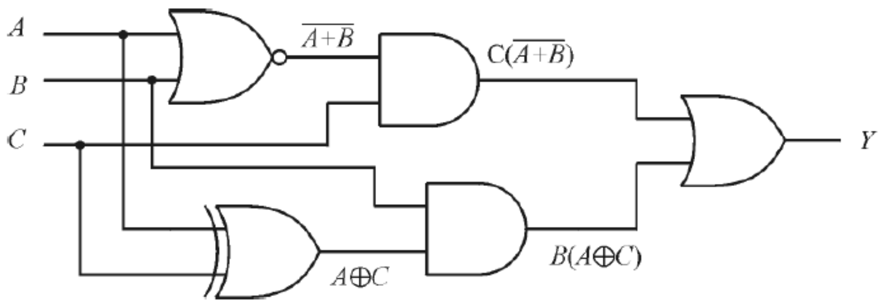


Exemplo 2: $f(A,B,C,D) = \overline{(A\,B\,\overline{C} + \overline{C\,D}) \oplus D}$
Solução:

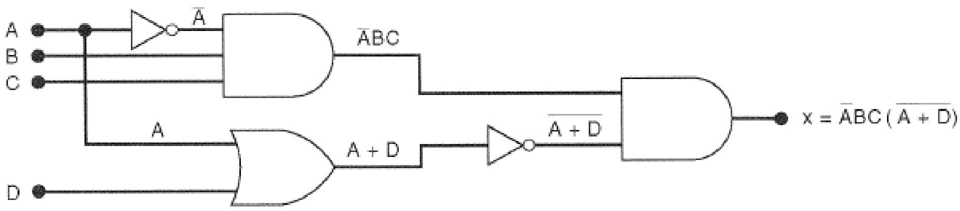


Exemplo 3:

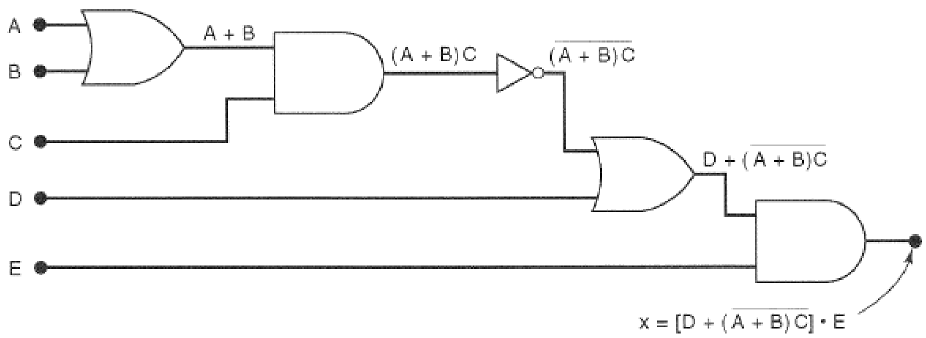
$$Y = \overline{(A+B)}C + B(A \oplus C)$$



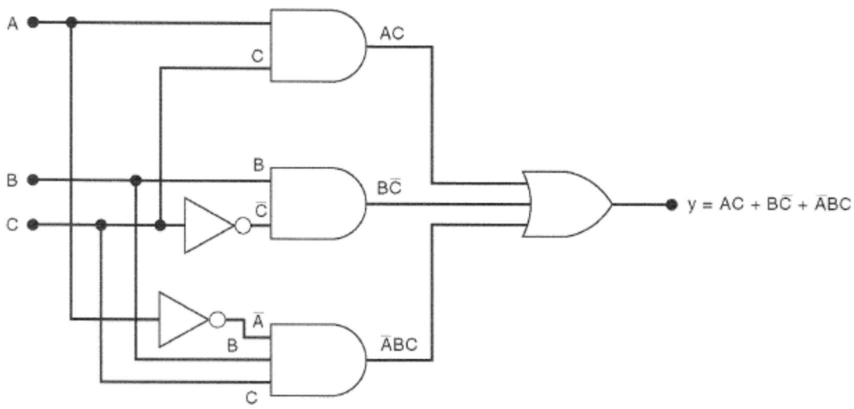
Exemplo 4:



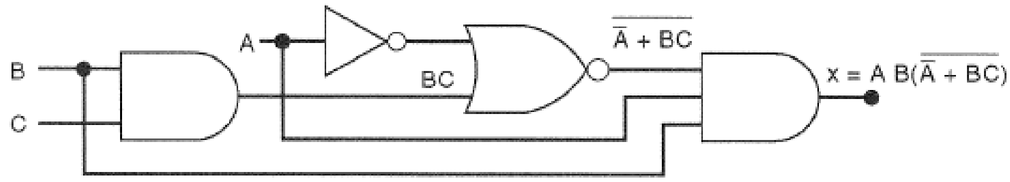
Exemplo 5:



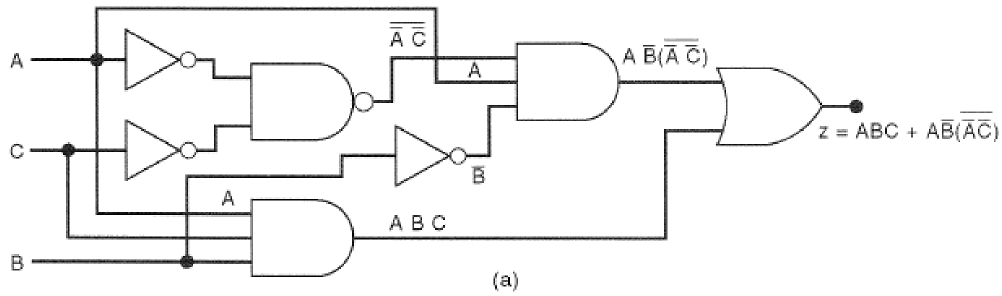
Exemplo 6:



Exemplo 7:

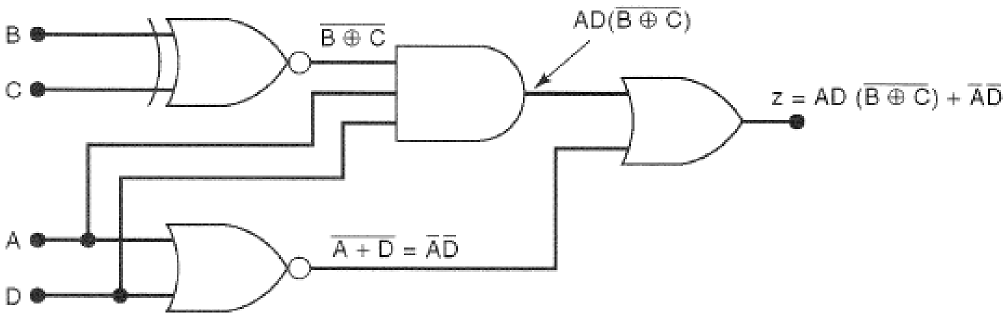


Exemplo 8:



(a)

Exemplo 9:



2.6. Obtenção das Expressões Booleanas a partir da Tabela Verdade

Exemplo 1:

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

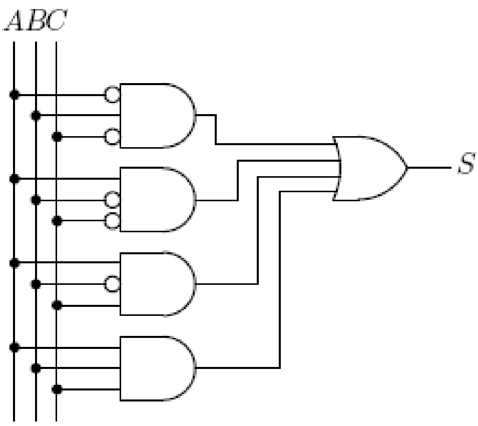
Solução:

$$Y = \overline{A}\overline{B}C + \overline{A}BC + AB\overline{C}$$

Exemplo 2:

A	B	C	S	
0	0	0	0	
0	0	1	0	
0	1	0	1	$\overline{A}\,B\,\overline{C}$
0	1	1	0	
1	0	0	1	$A\,\overline{B}\,\overline{C}$
1	0	1	1	$A\,\overline{B}\,C$
1	1	0	0	
1	1	1	1	$A\,B\,C$

$$S = \overline{A}\,B\,\overline{C} + A\,\overline{B}\,\overline{C} + A\,\overline{B}\,C + A\,B\,C$$



2.7. Obtenção da Tabela Verdade a partir de uma Descrição

Exemplo1:

Determinar a tabela verdade que implementa o problema abaixo.

Em uma sala há três pessoas (A,B,C) que podem votar Sim (1) ou Não (0), sobre um determinado assunto. Implemente a lógica que seja capaz de identificar as seguintes situações.

X - Indicar que a maioria votou Sim;
Y - Indicar que a maioria votou Não;
W - Indicar que houve unanimidade de Sim;
Z - Indicar que houve unanimidade de Não.

Solução:

A	B	C	X	Y	W	Z
0	0	0	0	0	0	1
0	0	1	0	1	0	0
0	1	0	0	1	0	0
0	1	1	1	0	0	0
1	0	0	0	1	0	0
1	0	1	1	0	0	0
1	1	0	1	0	0	0
1	1	1	0	0	1	0

$$X = \overline{A}\,B\,C + A\,\overline{B}\,C + A\,B\,\overline{C}$$
$$Y = \overline{A}\,\overline{B}\,C + \overline{A}\,B\,\overline{C} + A\,\overline{B}\,\overline{C}$$
$$W = A\,B\,C$$
$$Z = \overline{A}\,\overline{B}\,\overline{C}$$

Exemplo2:

Em uma fábrica que produz uma determinada luminária (que possui quatro lâmpadas denominadas de: A, B, C, D) o teste final de produção tem por objetivo liberar as luminárias para os clientes (**X = 1**) ou devolver para o setor de recuperação da produção (**Y = 1**) em função da detecção de pelo menos uma lâmpada com defeito.

Notação: **lâmpada sem defeito: 1**; **lâmpada com defeito: 0**.

Implemente a **função lógica** que seja capaz de identificar as seguintes situações (Implemente também a **Tabela Verdade** e o **circuito lógico**).

Solução:

X = A.B.C.D

Y = /A.B.C.D + A./B.C.D + A.B./C.D + A.B.C./D +
+ /A./B.C.D + /A.B./C.D + /A.B.C./D + A./B./C.D + A./B.C./D + A.B./C./D +
+ /A./B./C.D + /A.B./C./D + A./B./C./D +
+ /A./B./C./D

OBS: implemente também a Tabela Verdade e o circuito lógico...