Descrevendo Circuitos Lógicos

CPCX – UFMS
Prof. Renato F. dos Santos

3.1 Constantes e variáveis booleanas

- Uma variável booleana é uma quantidade que pode ser, em diferentes momentos, igual a 0 ou 1
- São muitas vezes usadas para representar o nível de tensão presente em uma conexão ou em terminais de entrada/saída de um circuito
  - 0 = tensão entre 0V e 0,8V
  - 1 = tensão entre 2v e 5V
- Variáveis booleanas 0 e 1 representam níveis de tensão, o qual é denominado nível lógico
- Em lógica digital, vários outros termos são usados como sinônimos para os níveis lógicos 0 e 1

3.1 Constantes e variáveis booleanas (Continuação)

Lógico 0	Lógico 1
Falso	Verdadeiro
Desligado	Ligado
Baixo	Alto
Não	Sim
Aberto	Fechado

Tabela 3.1 Termos sinônimos

3.1 Constantes e variáveis booleanas (Continuação)

- **A álgebra booleana é um modo de expressar a relação entre as entradas e saídas de um circuito lógico**
- **As entradas são consideradas variáveis lógicas, que determinam os níveis das saídas**
- **Ao longo de nossos estudos usaremos letras como símbolos para representar as variáveis lógicas. Por exemplo:**
  - **A letra *A* pode ser usada para representar a entrada ou a saída de um determinado circuito digital**
  - **Teremos $A = 0$ ou $A = 1$**

3.1 Constantes e variáveis booleanas (Continuação)

- **A álgebra booleana é mais fácil de ser manipulada se comparada com a álgebra convencional**
- **Na álgebra booleana não existem frações, decimais, números negativos, raízes quadradas, raízes cúbicas, logaritmos, números imaginários, e assim por diante**
- **Tem de fato, apenas três operações básicas:**
  - ***OR* (*OU*), *AND* (*E*) e *NOT* (*NÃO*)**

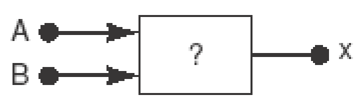
3.2 Tabela-verdade

- **É uma técnica usada para descrever como a saída de um circuito lógico depende dos níveis lógicos presentes nas entradas do circuito**
- **Essa tabela representa todas as combinações possíveis para as entradas, resultando na saída *x***
- **Observe que há nas tabelas-verdade:**
  - **Quatro linhas para duas entradas**
  - **Oito linhas para três entradas**
  - **Dezesseis linhas para quatro entradas**
- **O número combinações de entradas é igual a 2^N para uma tabela-verdade de *N* entradas**

FIGURA 3.1
Exemplos de tabelas-verdade para circuitos de: (a) duas entradas, (b) três entradas e (c) quatro entradas.

Entradas / Saída

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



(a)

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

(b)

A	B	C	D	x
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

(c)

3.3 Operação OR ('OU') e a porta OR

- **A operação booleana para a operação OR é**
  - $x = A + B$
- **Nessa expressão o sinal '+' não representa a adição convencional; ele representa a operação OR**
- **A operação lógica OR produz 1 + 1 = 1, não 1 + 1 = 2**
- **O mesmo é válido para três entradas usando o operador OR.**
  - **Então teremos** $x = A + B + C$
  - **Se considerarmos todas as três entradas em nível 1, teremos**
  - $x = 1 + 1 + 1 = 1$

3.3 Operação OR (‘OU’) e a porta OR (Continuação)

- **A expressão $x = A + B$ é lida como**
  - **‘*x* é igual a *A* OR *B*’**
- **Da mesma maneira a expressão $x = A + B + C$ é lida como**
  - **‘*x* é igual a *A* OR *B* OR *C*’**
- **Poderíamos dizer que *x* é *verdadeiro* (1) quando *A* é *verdadeiro* (1) OU *B* é *verdadeiro* (1) OU *C* é *verdadeiro (1)***

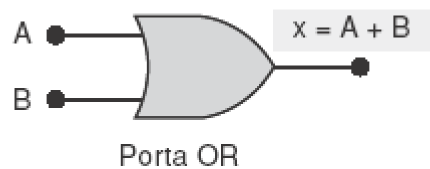
Porta OR

- **Em circuitos digitais, uma porta OR é um circuito que tem duas ou mais entradas**
- **A porta OR opera de modo que sua saída será ALTA se a entrada *A* ou *B* ou ambas forem nível lógico 1**
- **A saída será nível BAIXO apenas se todas as entradas forem nível 0**
- **A idéia é a mesma para quando houver mais de duas entradas**

OR

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

(a)



(b)

FIGURA 3.2
(a) Tabela-verdade que define a operação OR; (b) Símbolo de uma porta OR de duas entradas.

Resumo da operação OR

– **Os pontos importantes a serem lembrados em relação à operação OR e às portas OR são:**

1. **A operação OR gera um resultado (saída) 1 sempre que *quaisquer* das entradas for 1. Caso contrário a saída é 0.**
2. **Uma porta OR é um circuito lógico que realiza uma operação OR sobre as entradas do circuito**
3. **A expressão $x = A + B$ é lida assim: '*x* é igual a *A* OR *B*'**

3.4 Operação AND ('E') e a porta AND

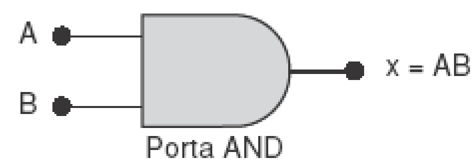
- A expressão booleana para a operação AND é
  - $x = A \cdot B$
- Nessa expressão o sinal (.) representa a operação booleana AND e não a operação de multiplicação
- Entretanto, a operação AND sobre variáveis booleanas equivale à multiplicação convencional
- A expressão $x = A \cdot B$ é lida com
  - '*x* igual a *A* AND *B*'
- *x* será 1 somente quando *A* e *B* forem, ambas, nível 1
- O sinal '.' normalmente é omitido, e a expressão torna-se simplesmente $x = AB$

FIGURA 3.7
(a) Tabela-verdade para a operação AND; (b) símbolo da porta AND.

AND

A	B	x = A · B
0	0	0
0	1	0
1	0	0
1	1	1

(a)



(b)

Porta AND

- **A saída da porta AND é igual ao produto lógico AND das entradas, que é $x = AB$**
- **É um circuito que opera de modo que sua saída seja nível ALTO somente quando todas as entradas forem nível ALTO.**
- **Para todos os outros casos, a saída é nível BAIXO**

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

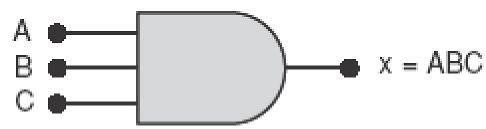


FIGURA 3.8
Tabela-verdade e símbolo para uma porta AND de três entradas.

Resumo das operações AND

1. **A operação AND é realizada da mesma maneira que a multiplicação convencional de 1s e 0s**
2. **Uma porta AND é um circuito lógico que realiza uma operação AND sobre as entradas do circuito**
3. **A saída de uma porta AND será 1 somente quando todas as entradas forem 1; para todos os outros casos, a saída será 0**
4. **A expressão *x* = *AB* é lida como '*x* é igual a *A* AND *B*'**

3.5 Operação NOT ('NÃO') ou INVERSOR

- **A operação NOT, também denominada INVERSOR, é diferente das operações OR e AND**
- **Pode ser realizada sobre uma única entrada**
- **Se a variável A for submetida à operação de inversão, o resultado *x* pode ser expresso como**
  - $x = \overline{A}$
- **onde a barra sobre o nome da variável representa a operação de inversão**
- **Essa expressão é lida como**
  - **'*x* é igual a *A* negado', ou**
  - **'*x* é igual ao inverso de *A*', ou**
  - ***x* é igual ao complemento de *A*'**

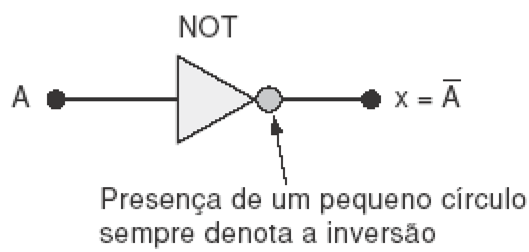
3.5 Operação NOT ('NÃO') ou INVERSOR (Continuação)

- **A tabela-verdade da figura 3.11(a) esclarece isso para os dois casos: $A = 0$ e $A = 1$. Isto é,**
  - **$0 = \overline{1}$ porque 0 é 1 NEGADO**
  - **$1 = \overline{0}$ porque 1 é 0 NEGADO**
- **A operação NOT também é conhecida como *inversão* ou *complemento***
- **Além da barra, podemos utilizar como indicador de inversão o símbolo apóstrofo ('). Isto é,**
  - **$A' = \overline{A}$**

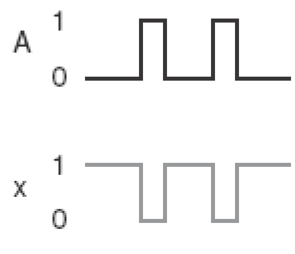
NOT

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

(a)



(b)



(c)

FIGURA 3.11
(a) Tabela-verdade; (b) Símbolo para o INVERSOR (circuito NOT); (c) Exemplos de formas de ondas.

Circuito NOT (INVERSOR)

- **Esse circuito sempre tem apenas uma entrada**
- **Seu nível lógico de saída é sempre o oposto ao nível lógico de entrada**
- **Se a entrada for = 0, a saída = 1 e vice-versa**

Resumo da operações booleanas

– **As regras para as operações OR, AND e NOT podem ser resumidas como segue:**

OR	***AND***	***NOT***
0 + 0 = 0	0 . 0 = 0	0 = 1
0 + 1 = 1	0 . 1 = 0	1 = 0
1 + 0 = 1	1 . 0 = 0	
1 + 1 = 1	1 . 1 = 1	