Circuitos Digitais

Aula 4

Operações lógicas e funções básicas

- Funções binárias
  - $f(X) = f(x_1, x_2, \ldots\ldots, x_r)$, $x_i \; \varepsilon \; \{0, 1\}$

    f(X) é uma correspondência única que associa um valor que pode ser 0 ou 1 para cada um dos 2^r valores possíveis que o vetor $[x_1, \ldots\ldots, x_r]$ pode tomar.
- Representação de uma função escalar
  - tabela verdade

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

Quantas combinações diferentes podem definidas com r = 3 ?
$2^3 = 8$

- **Quando r é muito grande este tipo de representação torna-se impraticável**

Operações lógicas

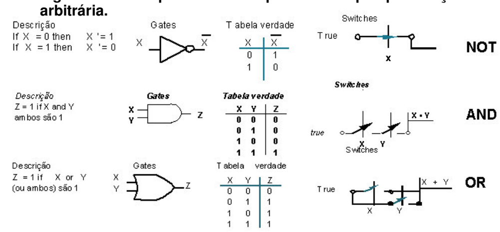
- Operadores lógicos são funções escalares usados para representarem o comportamento de um circuito lógico.
- Tipos de operadores
  - Operadores Unários
    - **Têm uma única variável**

x	$f_1(x) = 0$	$f_2(x) = x$	$f_3(x) = \bar{x}$	$f_4(x) = 1$
0	0	0	1	1
1	0	1	0	1

- **As funções $f_1(x) = 0$ e $f_4(x) = 1$ tomam valores constantes independentes de x**
- **A função $f_2(x) = x$ é chamada função identidade**
- **A função $f_3(x) = \bar{x}$ é chamada complementar, negação ou inversa. Esta função pode ser ainda representada por x' e $\neg x$**

Operações lógicas

- **Operações binárias**
  - **Operações binárias são funções representadas por duas variáveis representadas na forma $f(x_1, x_2) = x_1$ <operador> x_2**
  - **Seis destas funções são muito usadas e tem nomes especiais. Junto com a função NOT formam o conjunto de operações lógicas básicas que será usado para definir qualquer função arbitrária.**



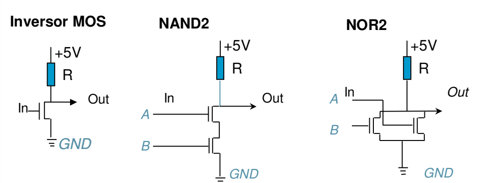
X	$\bar{X}$
0	1
1	0

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

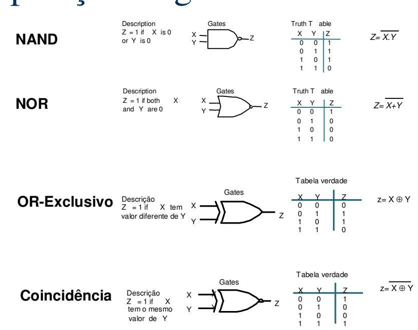
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

Operações lógicas

Tecnologia MOS (Metal Óxido Silício)



Operações lógicas



NAND: $Z = \overline{X.Y}$

X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0

NOR: $Z = \overline{X+Y}$

X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	0

OR-Exclusivo: $z = X \oplus Y$

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

Coincidência: $z = \overline{X \oplus Y}$

X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	1

Funções compostas

- Se $f(x_1, x_2, ..., x_r)$, $g(x_1, x_2, ..., x_r)$ e $h(x_1, x_2, ..., x_r)$ são três funções, então h $[f(x_1, x_2, ..., x_r), g(x_1, x_2, ..., x_r)]$ também é uma função.
  - Exemplo:

$$f(x_1, x_2) = x_1 \oplus x_2$$
$$g(x_2, x_3) = \overline{x_2 + x_3}$$
$$h(f, g) = f + g$$

$\downarrow$ = **nor**
$\uparrow$ = **nand**
$\wedge$ = **and**
$\vee$ = **or**

então podemos definir

$$h(x_1, x_2, x_3) = x_1 \oplus x_2 + (\overline{x_2 + x_3})$$

- Avaliação de funções compostas

$f(x_1, x_2, x_3) = (\overline{x_1 \cdot x_2}) \oplus (\overline{x_2 + x_3})$ *com* $y_1 = \overline{x_1 \cdot x_2}$ e $y_2 = \overline{x_2 + x_3}$

Logo $f(x_1, x_2, x_3) = y_1 \oplus y_2$

Com $[x_1, x_2, x_3] = [1\ 1\ 0]$, $y_1 = \overline{1 \cdot 1} = 0$ e $y_2 = \overline{1 + 0} = 0$, *temos*

portanto $f(x_1, x_2, x_3) = 0 \oplus 0 = 0$

Funções compostas

- Quando usamos operadores NAND, NOR ou OU-Exclusivo é melhor usar parênteses para separar operações lógicas.
- Quando se usa operadores AND, OR e NOT os parênteses podem ser eliminados desde que se use as seguintes regras de precedência:
  - Expressões sem parêntese
    - **Aplicar todas as instâncias da operação NOT às variáveis da esquerda para a direita.**
    - **Aplicar as instâncias da operação AND e em seguida as da operação OR sempre da esquerda para a direita**
  - **Expressões com parênteses**
    - **Neste caso é feito uma avaliação dentro dos parênteses e o valor resultante de cada parêntese entra como uma variável nas futuras avaliações.**
  - **O operador NOT aplicado sobre uma expressão tem o efeito de colocar parênteses na expressão.**

Uso de tabela verdade para avaliação de funções compostas

- $f(x_1, x_2, x_3) = (x_1 \wedge \overline{x}_2 \oplus x_3) \odot (x_1 \vee x_3)$

$x_1\ x_2\ x_3$	$x_1\ \overline{x}_2$	$x_1\ \overline{x}_2 \oplus x_3$	$x_1 \vee x_3$	$f(x_1, x_2, x_3)$
0 0 0	0	0	0	1
0 0 1	0	1	1	1
0 1 0	0	0	0	1
0 1 1	0	1	1	1
1 0 0	1	1	1	1
1 0 1	1	0	1	0
1 1 0	0	0	1	0
1 1 1	0	1	1	1

Equivalência lógica de funções

- Duas funções $f(x_1, x_2, ..., x_r)$ e $g(x_1, x_2, ..., x_r)$ definidas para os mesmos argumentos são equivalentes logicamente se e somente se $f(x_1, x_2, ..., x_r) = g(x_1, x_2, ..., x_r)$ para todos as possíveis combinações da r-tupla $x_1, x_2, ..., x_r$
  - Exemplo: $f(x_1, x_2, x_3) = (x_1 \oplus x_2)(x_2 \vee x_3)$
    $g(x_1, x_2, x_3) = \overline{x}_1\, x_2\, \overline{x}_3 \vee \overline{x}_1 x_2 x_3 \vee x_1 \overline{x}_2 x_3$

$x_1\ x_2\ x_3$	$x_2 \oplus x_3$	$x_2 \vee x_3$	$f(x_1, x_2, x_3)$	$\overline{x}_1 x_2 \overline{x}_3$	$\overline{x}_1 x_2 x_3$	$x_1 \overline{x}_2 x_3$	$g(x_1, x_2, x_3)$
0 0 0	0	0	0	0	0	0	0
0 0 1	0	1	0	0	0	0	0
0 1 0	1	1	1	1	0	0	1
0 1 1	1	1	1	0	1	0	1
1 0 0	1	0	0	0	0	0	0
1 0 1	1	1	1	0	0	1	1
1 1 0	0	1	0	0	0	0	0
1 1 1	0	1	0	0	0	0	0

Forma canônica de funções escalares

- Nós podemos mostrar que qualquer tabela verdade pode ser representada por uma função escalar que envolve apenas operações de AND, OR e NOT.
- Suponha uma r-tupla $[x_1, x_2, ..., x_r]$
  - **Termo de produto**- é uma função que é definida como o AND de um conjunto de termos, onde cada termo é uma variável x_j ou o seus complemento $\overline{x}_i$. Nenhuma variável pode aparecer mais que uma vez em um termo produto.

x_1, x_2, x_3	$m_0 = \overline{x1}\ \overline{x_2},\ \overline{x_3}$	$m_1 = \overline{x_1}\ \overline{x_2},\ x_3$	$m_2 = \overline{x_1}\ x_2, \overline{x_3}$	....	$m_6 = x_1\ x_2, \overline{x_3}$	$m_7 = x_1\ x_2,\ x_3$
0 0 0	1	0	0		0	0
0 0 1	0	1	0		0	0
0 1 0	0	0	1		0	0
0 1 1	0	0	0		0	0
1 0 0	0	0	0		0	0
1 0 1	0	0	0		0	0
1 1 0	0	0	0		1	0
1 1 1	0	0	0		0	1

Forma canônica de funções escalares

- **Minitermo**
- Chamamos de minitermo (ou minitermo –1) um termo produto no qual todas as variáveis aparecem apenas uma vez. A razão que estes termos são chamados minitermos é que eles tem um valor ‘1’ para apenas uma das 2^r possíveis combinações da r-tupla $[x_1, x_2, ..., x_r]$ e zero para os outros valores.
- **Observe que:**
  - $\mathbf{m_4 = x_1 \overline{x}_2 \overline{x}_3}$ **porque** $\mathbf{4_{10} = 100_2}$
  - $\mathbf{m_5 = x_1 \overline{x}_2 x_3}$ **porque** $\mathbf{5_{10} = 101_2}$
  - $\mathbf{m_7 = x_1 x_2 x_3}$ **porque** $\mathbf{7_{10} = 111_2}$

  ..

  - Qualquer função booleana pode ser expressa como uma soma destes minitermos

Forma canônica de funções escalares

Termo de soma -é uma função que é definida como o OR de um conjunto de termos, onde cada termo é uma variável x_j ou o seus complemento $\overline{x_i}$. Nenhuma variável pode aparecer mais que uma vez no termo soma.

x_1 x_2, x_3	$M_0= x_1 +x_2 +x_3$	$M_1= x_1 +x_2 +\overline{x_3}$	$M_2= x_1 +\overline{x_2} +x_3$	....	$M_7=\overline{x_1} +\overline{x_2} +\overline{x_3}$
0 0 0	0	1	1	.	1
0 0 1	1	0	1	.	1
0 1 0	1	1	0	.	1
0 1 1	1	1	1	.	1
1 0 0	1	1	1	.	1
1 0 1	1	1	1	.	1
1 1 0	1	1	1	.	1
1 1 1	1	1	1	.	0

Forma canônica de funções escalares

Maxtermo (ou Maxtermo – 0) é um termo soma em que todas as variáveis aparecem uma e apenas uma única vez. Chamamos maxtermos porque eles têm um valor '0' para apenas uma das 2^r possíveis combinações da r-tupla [x_1, x_2,......, x_r] e '1' para todos os outros valores.

Observe que:

- **$M_4 = \overline{x}_1+x_2+ x_3$ porque $4_{10} = 100_2$**
- **$M_5 = \overline{x}_1+x_2+ \overline{x}_3$ porque $5_{10} = 101_2$**
- **$M_7 = \overline{x}_1+\overline{x}_2+ \overline{x}_3$ porque $7_{10} = 111_2$**

  ....
- Qualquer função booleana pode ser expressa como uma soma destes Maxtermos

Representação canônica de expressão lógica

- Após o conceito de minitermos e maxitermos podemos usar estas funções como um conjunto de blocos básicos que representam qualquer função descrita por uma tabela verdade.

  – **tabela verdade**

termo	x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

$f(x_1, x_2, x_3)= m_1+m_3+m_4+ m_7$

$f(x_1, x_2, x_3)= M_0.M_2.M_5. M_6$

- **Forma canônica soma de produtos ou forma normal disjuntiva**

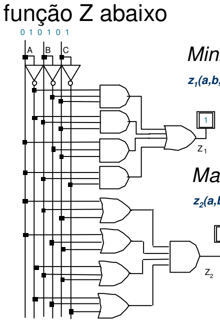
  $f(x_1, x_2, x_3)= m_1+m_3+m_4+ m_7$ ou Σ (m_1,m_3,m_4, m_7)
- **Forma canônica produto de somas ou forma normal conjuntiva**

  $f(x_1, x_2, x_3)= M_0.M_2.M_5. M_6$ ou Π (M_0,M_2,M_5, M_6)
- **Qualquer função de r variáveis pode ser representada em termos de NAND (NOT, AND) e NOR (OR, NOT)**
- **Para converter de uma forma canônica para outra troque o símbolo Σ por Π e liste os números que forma excluídos da foram original.**

Exemplos de circuitos lógicos

- Implementar a função Z abaixo

a	b	c	z
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0



Minitermos

$z_1(a,b,c)= m_1+m_3+m_5+ m_6$

Maxtermos

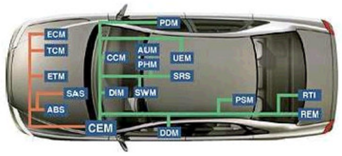
$z_2(a,b,c)= M_0.M_2.M_4.M_7$

Exemplo:

Desenvolver um circuito de alarme de um automóvel com as seguintes características funcionais:

O alarme/advertência deve ser acionado quando a ignição estiver acionada (carro ligado) e pelos menos uma das porta estiver aberta.

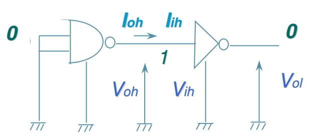
Obs: considere que o carro possui apenas duas portas.



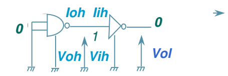
Parâmetros elétricos básicos em Circuitos Digitais

Terminologia usada em circuitos integrados

Embora existam diferentes tipos de famílias lógicas, a nomenclatura usada para identificar certos parâmetros elétricos e operacionais são padronizados:

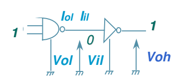


Terminologia



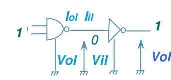
- **I_{oh}** - High Level Output Current - Corrente que flui na saída de uma porta lógica em nível lógico 1(alto) sob condições normais de carga.
- **I_{ih}** - High Level Input Current - Corrente de entrada de uma porta lógica quando um nível lógico 1(alto) é aplicado à entrada da porta.
- **V_{oh}**(min) - High Level Output Voltage - Nível de tensão de saída de uma porta lógica no estado lógico 1(alto).
- **V_{ih}**(min) - High Level Input Voltage - Nível de tensão de entrada necessário para se assumir nível lógico 1(alto) na entrada de um circuito lógico.

Terminologia



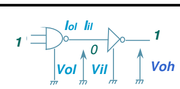
- **I_{ol}** - Low Level Output Current - Corrente que flui na saída de uma porta lógica em nível lógico 0(baixo) sob condições normais de carga.
- **I_{il}** - Low Level Input Current - Corrente de entrada de uma porta lógica quando um nível lógico 0(baixo) é aplicado à entrada da porta.
- **V_{ol}**(máx) - Low Level Output Voltage - Nível de tensão de saída de uma porta lógica no estado lógico 0(baixo).
- **V_{il}**(máx) - Low Level Input Voltage - Nível de tensão de entrada necessário para se assumir nível lógico 0(baixo) na entrada de um circuito lógico.

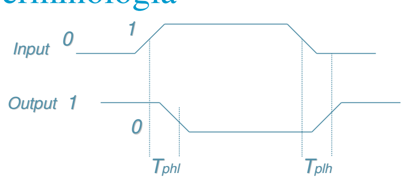
Terminologia



- **I_{cc}** - (Supply Current) - Corrente necessária para alimentação do circuito integrado.
- **V_{cc}** (Vdd) - (Supply Voltage) - tensão de alimentação do circuito integrado.

Terminologia

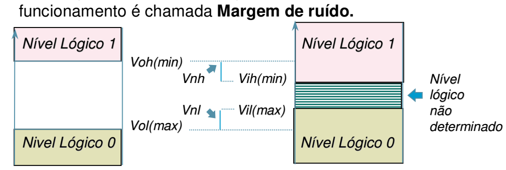




- **T_{plh}** - Tempo entre o sinal de entrada e o sinal de saída com a saída do circuito lógico indo de 0(baixo) para 1(alto).
- **T_{phl}** - Tempo entre o sinal de entrada e o sinal de saída com a saída do circuito lógico indo de 1(baixo) para 0(baixo).

Imunidade à ruídos

- Descargas elétricas e campos magnéticos podem induzir tensões nos fios que conectam circuitos lógicos. Estas tensões podem algumas vezes alterar o nível de tensão de entrada de um circuito lógico, modificando o nível lógico original.
- Todo circuito lógico deve surportar uma certa variação de tensão na entrada e ser imune a um acerta faixa de ruído. Esta faixa a qual o circuito deve suportar sem alteração de funcionamento é chamada **Margem de ruído.**



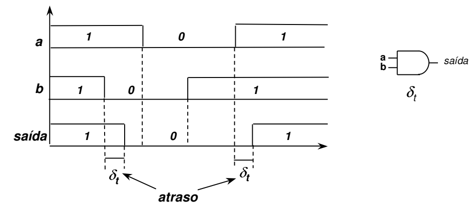
Vnh = margem de ruído do nível lógico 1(alto) = Voh(min)-Vih(min)
Vnl = margem de ruído do nível lógico 0(baixo) = Vil(máx)-Vol(máx)

Características elétricas de circuitos digitais

- Quando conectamos circuitos digitais, o sinal de um circuito pode ser o sinal de entrada de vários circuitos, quantos?
  - Para que a informação seja transferida através da rede é necessário que haja energia para acionar os circuitos.
  - Por simplicidade de projeto, o carregamento da entrada (energia absorvida pela entrada) e a quantidade de energia fornecida pela saída é dada em unidades de carga.
  - Unidade de carga equivale a energia absorvida por uma entrada de um circuito digital.
  - A quantidade máxima de entradas que um circuito pode alimentar e o número máximo de entrada de um circuito digital são definidas por dois parâmetros.
    - **FIN-IN - representa o número máximo de variáveis de entrada (ou unidades de carga) independentes de um determinado circuito.**
    - **FAN-OUT - representa o número máximo de unidades de carga que podem ser acionadas adequadamente por uma saída de um circuito digital.**

Comportamento dinâmico de circuitos digitais

- Diagrama de tempo



• *A velocidade de operação da rede é determinada pelo tempo total que a porta lógica leva para alcançar um estado permanente*

Características elétricas de circuitos digitais