



# Circuitos Digitais II - 6882

Paulo Roberto de Oliveira

Universidade Estadual de Maringá  
Departamento de Informática

Bacharelado em Ciência da Computação

# Aula de Hoje

- **Revisão da aula anterior**
- **Comandos Condicionais**
  - **Comando *WHEN ELSE***
  - **Comando *IF THEN ELSE***
  - **Comando *CASE WHEN***
  - **Comando *WITH SELECT WHEN***

# Revisão

- Operadores

# VHDL - Operadores

## Operadores de atribuição

- Utilizados para a associação de valores às variáveis, sinais e constantes.

OPERADOR	SIGNIFICADO	EXEMPLO
<code>&lt;=</code>	Atribuição de sinal	<code>Sig &lt;= '0';</code>
<code>:=</code>	Atribuição de variável	<code>Var := '0';</code>
<code>:=</code>	Inicialização de constantes, sinais e variáveis	<code>Signal sig : BIT := '0';</code>
<code>=&gt;</code>	Atribuição de valores únicos em vetores	<code>Vet &lt;= (0 =&gt; '1');</code>
<code>=&gt;</code>	Atribuição de vários valores em vetores junto com a palavra reservada OTHERS	<code>Vet &lt;= (0 =&gt; '1', OTHERS =&gt; '0');</code>

# VHDL - Operadores

## Operadores aritméticos

- Utilizados para a realização de operações matemáticas e empregados em objetos dos tipos INTEGER e REAL.

OPERADOR	SIGNIFICADO	EXEMPLO
+	Soma	<code>i := i + 1;</code>
-	Subtração	<code>j := j - 1;</code>
*	Multiplicação	<code>s := a * b;</code>
/	Divisão	<code>b := a / c;</code>
mod	Módulo	<code>s0 &lt;= s1 MOD s2;</code>
rem	Resto	<code>s0 &lt;= s1 REM s2;</code>
Abs	Valor absoluto	<code>i0 := ABS s1;</code>
**	Exponenciação	<code>a := b ** c;</code>

# VHDL - Operadores

## Operadores de sinais

- Utilizados como função identidade e negação de um valor numérico, podendo alterar o sinal de determinado valor.

OPERADOR	SIGNIFICADO	EXEMPLO
+	Identidade	Var1 := + var2;
-	Negação	Var1 := - var1;

# VHDL - Operadores

## Operador de concatenação

- Operador & concatena dois vetores, 'vet\_a' e 'vet\_b', de um mesmo tipo em um terceiro vetor, 'vet\_c'. Neste caso, 'vet\_c' deve possuir tamanho igual à soma dos tamanhos de 'vet\_a' e 'vet\_b'. Concatenação entre elementos também pode ser realizada, produzindo como resultado um vetor do mesmo tipo dos elementos.

- Exemplo:

```
SIGNAL    a : BIT := '0';
```

```
SIGNAL vet0 : BIT_VECTOR (2 DOWNT0 0) := "001";
```

```
SIGNAL vet1 : BIT_VECTOR (3 DOWNT0 0);
```

```
SIGNAL vet2 : BIT_VECTOR (4 DOWNT0 0);
```

```
vet1 <= a & vet0;           --"0001"
```

```
vet2 <= a & vet0 & vet0(0); --"00011"
```

# VHDL - Operadores

## Operadores de deslocamento

- Utilizados para a realização de operações de deslocamento aritmético, deslocamento lógico e rotação em vetores com elementos dos tipos BIT, STD\_LOGIC, STD\_ULOGIC ou BOOLEAN.

OPERADOR	SIGNIFICADO
sll	Shift left - deslocamento lógico para a esquerda
srl	Shift right - deslocamento lógico para a direita
sla	Shift left arithmetic - deslocamento aritmético para a esquerda
sra	Shift right arithmetic - deslocamento aritmético para a direita
rol	Rotate left - rotação para a esquerda
ror	Rotate right - rotação para a direita

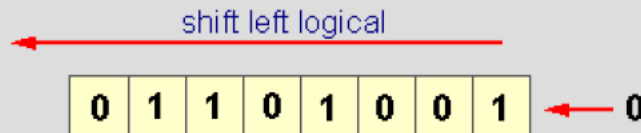


# VHDL - Operadores

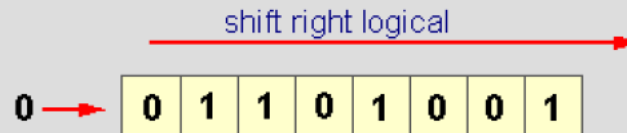
## Operadores de deslocamento

- Exemplo:

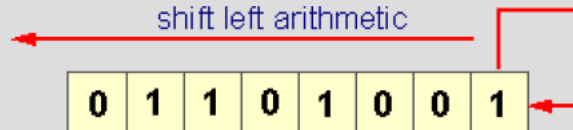
sll →



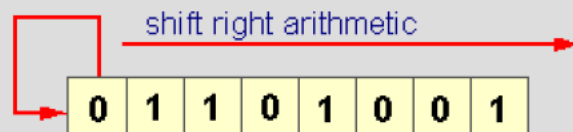
srl →



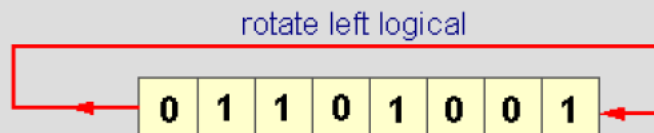
sla →



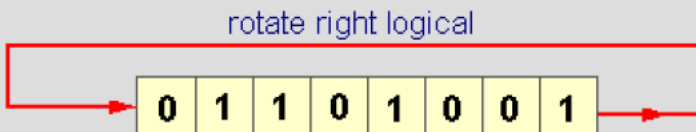
sra →



rol →



ror →



# VHDL - Operadores

## Operadores relacionais

- Utilizados para a comparação de dois elementos de um mesmo tipo e o resultado de uma operação relacional é sempre do tipo booleano.

OPERADOR	SIGNIFICADO
=	Equivalência
/=	Desigualdade
>	Maior que
<	Menor que
>=	Maior ou igual a
<=	Menor ou igual a

# VHDL - Operadores

## Operadores lógicos

OPERADOR	EXPRESSÃO	EXEMPLO
NOT	$\bar{a}$	NOT a
AND	$a \bullet b$	a AND b
OR	$a + b$	a OR b
XOR	$a \oplus b$	a XOR b
NAND	$\overline{a \bullet b}$	a NAND b
NOR	$\overline{a + b}$	a NOR b
XNOR	$\overline{a \oplus b}$	a XNOR b

# VHDL - Operadores

## Precedência de operadores

PRECEDÊNCIA	OPERADORES	FUNÇÃO
<p>Menor</p> <p>↓</p> <p>Maior</p>	AND, OR, NAND, NOR, XOR, XNOR	Operadores lógicos
	=, /=, <, >, =<, >=	Operadores relacionais
	SLL, SRL, SLA, SRA, ROL, ROR	Operadores de deslocamento
	+, -, &	Soma, subtração e concatenação
	+, -	Sinais positivo e negativo
	*, /, MOD, REM	Multiplicação, divisão, módulo e resto
	**, ABS, NOT	Exponenciação, valor absoluto e negação lógica

# VHDL – Operadores

## Precedência de operadores

- Nota 1: Os operadores de uma mesma classe possuem o mesmo nível de precedência.
- Nota 2: Uma forma de explicitar que determinadas operações sejam executadas antes que outras é pelo uso de parênteses.
- Nota 3: A função NOT possui precedência sobre as demais e é executada primeiro, caso a prioridade não seja forçada com o uso de parênteses.
- Nota 4: Quando uma expressão contém duas operações que estão em um mesmo nível de precedência, a operação mais à esquerda será analisada primeiro.
- Nota 5: Devido à rigidez da linguagem, VHDL, com relação a tipos, normalmente os operandos de uma operação são do mesmo tipo.

# Aula de Hoje

- **Comandos Condicionais**
  - **Comando *WHEN ELSE***
  - **Comando *IF THEN ELSE***
  - **Comando *CASE WHEN***
  - **Comando *WITH SELECT WHEN***

# VHDL - Comandos Condicionais

## Comandos Condicionais

- Comandos Condicionais permitem alterar o fluxo de execução do código
- Em VHDL há 4 comandos condicionais:
  - WHEN ELSE
  - IF THEN ELSE
  - CASE WHEN
  - WITH SELECT WHEN

# VHDL - Comandos Condicionais

## WHEN ELSE

- É um comando concorrente
- Não pode ser usado dentro de procedimentos, funções e processos
- Transfere o valor de uma expressão para um sinal destino caso uma determinada condição seja satisfeita

## Sintaxe

```
sinal_destino <= expressao_1 WHEN condicao_1 ELSE  
                    expressao_2 WHEN condicao_2 ELSE  
                    expressao_3 WHEN condicao_3 ELSE  
                    expressao_4;
```

- Nota: O comando condicional WHEN ELSE é útil para expressar funções lógicas em forma de tabela verdade



# VHDL - Comandos Condicionais

## WHEN ELSE

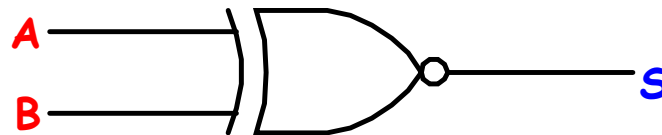
- Exemplo:
- Implemente a função XNOR com o comando condicional WHEN ELSE

### TV da Porta XNOR

Entradas		Saída
A	B	S
0	0	1
0	1	0
1	0	0
1	1	1

Função XNOR Representação:  $S = \overline{A \oplus B} = A \odot B$

### Símbolo da Porta XNOR



### Condições

A variável de saída S será igual a 1 quando as variáveis de entrada A e B forem iguais, senão a variável de saída S será igual a 0.

# VHDL - Comandos Condicionais

## WHEN ELSE

- Exemplo:
- Implemente a função XNOR com o comando condicional WHEN ELSE

TV da Porta XNOR

Entradas		Saída
A	B	S
0	0	1
0	1	0
1	0	0
1	1	1

```
LIBRARY ieee;                -- Funcao xnor usando comando when else
USE ieee.std_logic_1164.all;

ENTITY xnor_cc_when IS
    PORT (a, b : IN BIT;
          s : OUT BIT);

END xnor_cc_when;

ARCHITECTURE condicional OF xnor_cc_when IS
BEGIN
    s <= '1' WHEN (a=b) ELSE '0';

END condicional;
```

# VHDL - Comandos Condicionais

## WHEN ELSE

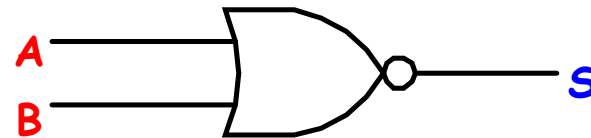
- Exercício 01:
- Implemente a função NOR com o comando condicional WHEN ELSE

### TV da Porta NOR

Entradas		Saída
A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

Função NOR Representação:  $S = \overline{A+B}$

### Símbolo da Porta NOR



### Condições

A variável de saída S será igual a 1 quando as variáveis de entrada A e B forem iguais a 0, senão a variável de saída S será igual a 0.

# VHDL - Comandos Condicionais

## Solução

- Exercício 01:
- Implemente a função NOR com o comando condicional WHEN ELSE

TV da Porta NOR

Entradas		Saída
A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

```
LIBRARY ieee;                                -- Funcao nor usando comando when else
USE ieee.std_logic_1164.all;
ENTITY nor_cc_when IS
    PORT (a, b : IN BIT;
          s : OUT BIT);
END nor_cc_when;
ARCHITECTURE condicional OF nor_cc_when IS
BEGIN
    s <= '1' WHEN a='0' AND b='0' ELSE '0';
END condicional;
```

# VHDL - Comandos Condicionais

## WHEN ELSE

- Exercício 02: Implemente um comparador de duas palavras com 2 bits usando o comando condicional WHEN ELSE.
- Considere as palavras - **a** (**a(1)**, **a(0)**) e **b** (**b(1)**, **b(0)**) - como sendo as variáveis de entrada.
- Condição: A variável de saída será igual a 1 quando as variáveis de entrada **a** e **b** forem iguais, senão a variável de saída será igual a 0.



# VHDL - Comandos Condicionais

## Solução

```
LIBRARY ieee;                                -- Comparador de 2 bits usando when else
USE ieee.std_logic_1164.all;

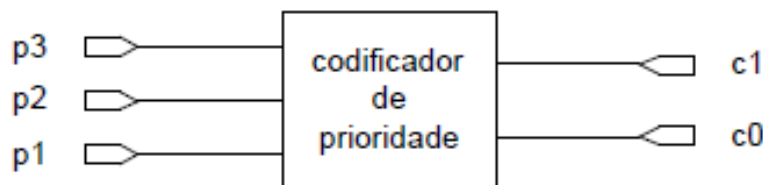
ENTITY compara_cc_when IS
    PORT (a, b : IN BIT_VECTOR(1 DOWNTO 0);
          igual : OUT BIT);
END compara_cc_when;

ARCHITECTURE condicional OF compara_cc_when IS
BEGIN
    igual <= '1' WHEN (a=b) ELSE '0';
END condicional;
```

# VHDL - Comandos Condicionais

## WHEN ELSE

- Exercício 03: Implemente um codificador de prioridade com 3 variáveis de entrada **p(3)**, **p(2)** e **p(1)**, sendo que p(3) tem prioridade em relação a p(2) e p(2) tem prioridade em relação a p(1). A saída é constituída por 2 bits (**c(1)** e **c(0)**), indicando a entrada prioritária quando os 2 bits são iguais a 1 e a entrada de menor prioridade com todos os bits iguais a zero.



p3	p2	p1	c1	c0
1	-	-	1	1
0	1	-	1	0
0	0	1	0	1
0	0	0	0	0

- => não importa

# VHDL - Comandos Condicionais

## Solução

```
LIBRARY ieee;                -- Codificador de prioridade usando comando when else
USE ieee.std_logic_1164.all;

ENTITY cod_pri_cc_when IS
    PORT (p : IN BIT_VECTOR(3 DOWNT0 1);
          c : OUT BIT_VECTOR(1 DOWNT0 0));
END cod_pri_cc_when;

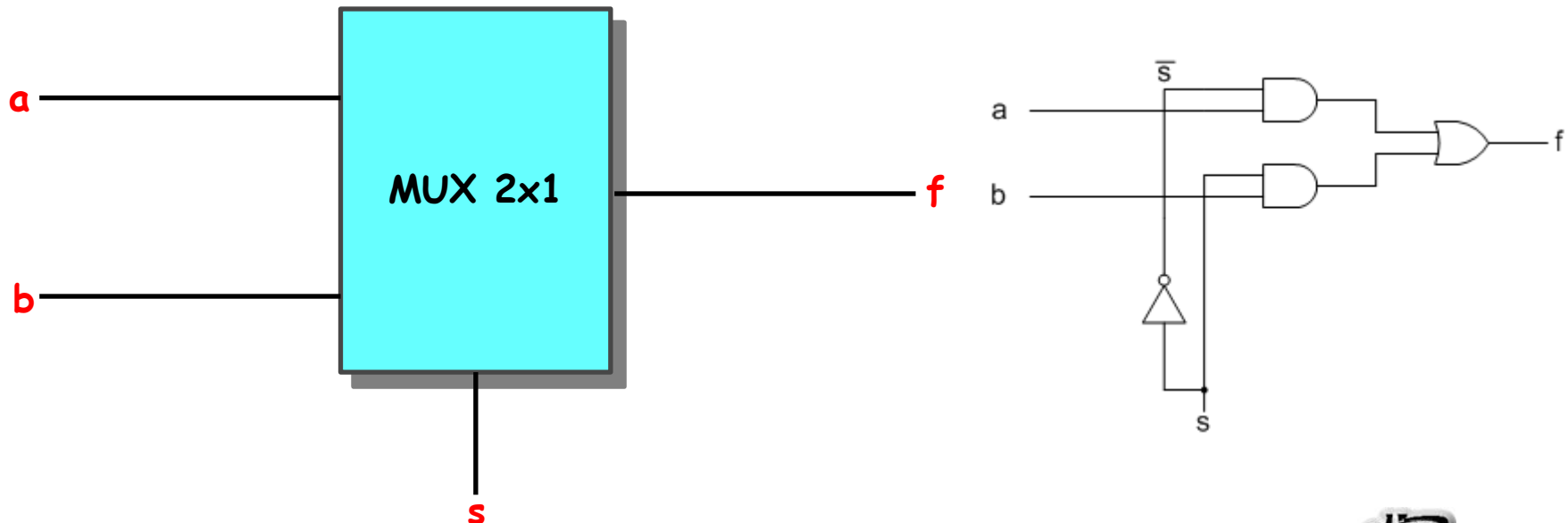
ARCHITECTURE condicional OF cod_pri_cc_when IS
BEGIN
    c <= "11" WHEN p(3)='1' ELSE
        "10" WHEN p(2)='1' ELSE
        "01" WHEN p(1)='1' ELSE
        "00";
END condicional;
```



# VHDL - Comandos Condicionais

## WHEN ELSE

- Exercício 04: Implemente um multiplexador de duas entradas de dados (**a** e **b**), uma entrada de seleção (**s**) e uma única saída de dados (**f**). O seu funcionamento basear-se-á na escolha da entrada de seleção que determinará qual das entradas de dados aparecerá na saída de dados.



# VHDL - Comandos Condicionais

## Solução

```
LIBRARY ieee;           -- Multiplexador 2 x 1 usando comando when else
USE ieee.std_logic_1164.all;

ENTITY mux_cc_when IS
    PORT (a, b : IN BIT;
          s  : IN BIT;
          f  : OUT BIT);

END mux_cc_when;

ARCHITECTURE condicional OF mux_cc_when IS

BEGIN
    f <= a WHEN s = '0' ELSE b;

END condicional;
```

# VHDL – Comandos Condicionais

## WHEN ELSE

- Exercício 05: Implemente um codificador decimal usando os comandos condicionais WHEN ELSE.
- Considere os nomes das entradas como sendo ch0 até ch9 (para representar chaves de entradas).
- Considere que cada chave de entrada é ativa em nível 0.
- A tabela verdade do codificador decimal é dada a seguir:

# VHDL - Comandos Condicionais

## WHEN ELSE

Tabela Verdade do Codificador Decimal

BCD	saida3	saida2	saida1	saida0
ch0	0	0	0	0
ch1	0	0	0	1
ch2	0	0	1	0
ch3	0	0	1	1
ch4	0	1	0	0
ch5	0	1	0	1
ch6	0	1	1	0
ch7	0	1	1	1
ch8	1	0	0	0
ch9	1	0	0	1

# VHDL – Comandos Condicionais

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY cod_dec_cc_when IS
    PORT (ch0, ch1, ch2, ch3, ch4, ch5, ch6, ch7, ch8, ch9 : IN BIT;
          saida : OUT BIT_VECTOR (3 DOWNT0 0));
END cod_dec_cc_when;
ARCHITECTURE condicional OF cod_dec_cc_when IS
BEGIN
    saida <= "0000" WHEN ch0 = '0' ELSE
              "0001" WHEN ch1 = '0' ELSE
              "0010" WHEN ch2 = '0' ELSE
              "0011" WHEN ch3 = '0' ELSE
              "0100" WHEN ch4 = '0' ELSE
              "0101" WHEN ch5 = '0' ELSE
              "0110" WHEN ch6 = '0' ELSE
              "0111" WHEN ch7 = '0' ELSE
              "1000" WHEN ch8 = '0' ELSE
              "1001";
END condicional;
```

# VHDL - Comandos Condicionais

## IF THEN ELSE

- É um comando sequencial
- Utilizado na descrição comportamental de componentes → Utilizado em procedimentos, funções e processos.
- Transfere o valor de uma expressão para um sinal destino caso uma determinada condição seja satisfeita

• Sintaxe →

```
IF condicao_1 THEN
    comando_sequencial;
ELSIF condicao_2 THEN                -- Clausula ELSIF opcional
    comando_sequencial;
ELSIF condicao_3 THEN
    comando_sequencial;
ELSE                                -- Clausula ELSE opcional
    comando_sequencial;
END IF;
```

# VHDL - Comandos Condicionais

## IF THEN ELSE

- Aninhar vários níveis de construções IF THEN ELSE.

- Sintaxe →

```
IF condicao_1 THEN
    IF condicao_2 THEN
        comando_sequencial;
    ELSE
        comando_sequencial;
    END IF;
ELSE
    IF condicao_3 THEN
        comando_sequencial;
    ELSE
        comando_sequencial;
    END IF;
END IF;
```

# VHDL - Comandos Condicionais

## IF THEN ELSE

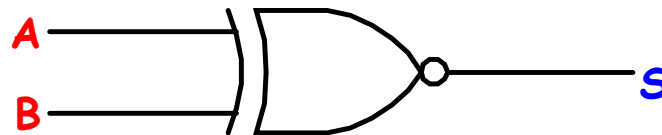
- Exemplo:
- Implemente a função XNOR com o comando condicional IF THEN ELSE

### TV da Porta XNOR

Entradas		Saída
A	B	S
0	0	1
0	1	0
1	0	0
1	1	1

Função XNOR Representação:  $S = \overline{A \oplus B} = A \odot B$

### Símbolo da Porta XNOR



### Condições

Se as variáveis de entrada A e B forem iguais então a variável de saída S será igual a 1, senão a variável de saída S será igual a 0.



# VHDL - Comandos Condicionais

## Solução

```
LIBRARY ieee;                                -- Funcao xnor usando comando if then else
USE ieee.std_logic_1164.all;

ENTITY xnor_cc_if IS
    PORT (a, b : IN BIT;
          s : OUT BIT);

END xnor_cc_if;

ARCHITECTURE condicional OF xnor_cc_if IS
BEGIN
    PROCESS (a, b)
    BEGIN
        IF (a=b) THEN s <= '1';
        ELSE
            s <= '0';
        END IF;
    END PROCESS;
END condicional;
```

# VHDL - Comandos Condicionais

## IF THEN ELSE

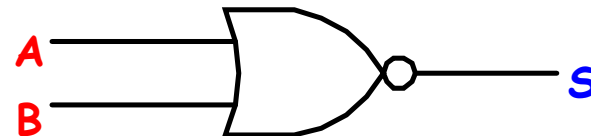
- Exercício 01:
- Implemente a função NOR com o comando condicional IF THEN ELSE

### TV da Porta NOR

Entradas		Saída
A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

Função NOR Representação:  $S = \overline{A+B}$

### Símbolo da Porta NOR



### Condições

Se as variáveis de entrada A e B forem iguais a 0 então a variável de saída S será igual a 1, senão a variável de saída S será igual a 0.

# VHDL - Comandos Condicionais

## Solução

```
LIBRARY ieee;                                -- Funcao nor usando comando if then else
USE ieee.std_logic_1164.all;

ENTITY nor_cc_if IS
    PORT (a, b : IN BIT;
          s : OUT BIT);

END nor_cc_if;

ARCHITECTURE condicional OF nor_cc_if IS
BEGIN
    PROCESS (a, b)
    BEGIN
        IF (a='0') AND (b='0') THEN s <= '1';
        ELSE
            s <= '0';
        END IF;
    END PROCESS;
END condicional;
```

# VHDL - Comandos Condicionais

## IF THEN ELSE

- Exercício 02: Implemente um comparador de duas palavras com 2 bits usando o comando condicional IF THEN ELSE.
- Considere as palavras - **a** (**a(1)**, **a(0)**) e **b** (**b(1)**, **b(0)**) - como sendo as variáveis de entrada.
- Condição: A variável de saída será igual a 1 quando as variáveis de entrada **a** e **b** forem iguais, senão a variável de saída será igual a 0.



# VHDL - Comandos Condicionais

## Solução

```
LIBRARY ieee;                                -- Comparador de 2 bits usando comando if then else
USE ieee.std_logic_1164.all;

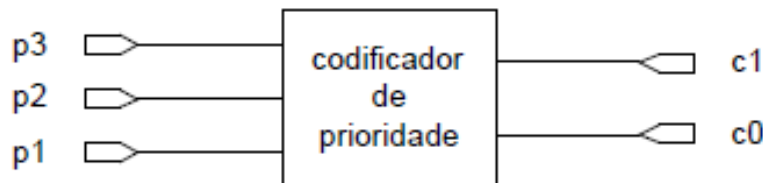
ENTITY compara_cc_if IS
    PORT (a, b : IN BIT_VECTOR(1 DOWNT0 0);
          igual : OUT BIT);
END compara_cc_if;

ARCHITECTURE condicional OF compara_cc_if IS
BEGIN
    PROCESS (a, b)
    BEGIN
        IF a = b THEN igual <= '1';
        ELSE
            igual <= '0';
        END IF;
    END PROCESS;
END condicional;
```

# VHDL - Comandos Condicionais

## IF THEN ELSE

- Exercício 03: Implemente um codificador de prioridade com 3 variáveis de entrada **p(3)**, **p(2)** e **p(1)**, sendo que p(3) tem prioridade em relação a p(2) e p(2) tem prioridade em relação a p(1). A saída é constituída por 2 bits (**c(1)** e **c(0)**), indicando a entrada prioritária quando os 2 bits são iguais a 1 e a entrada de menor prioridade com todos os bits iguais a zero.



p3	p2	p1	c1	c0
1	-	-	1	1
0	1	-	1	0
0	0	1	0	1
0	0	0	0	0

- => não importa

# VHDL - Comandos Condicionais

## Solução

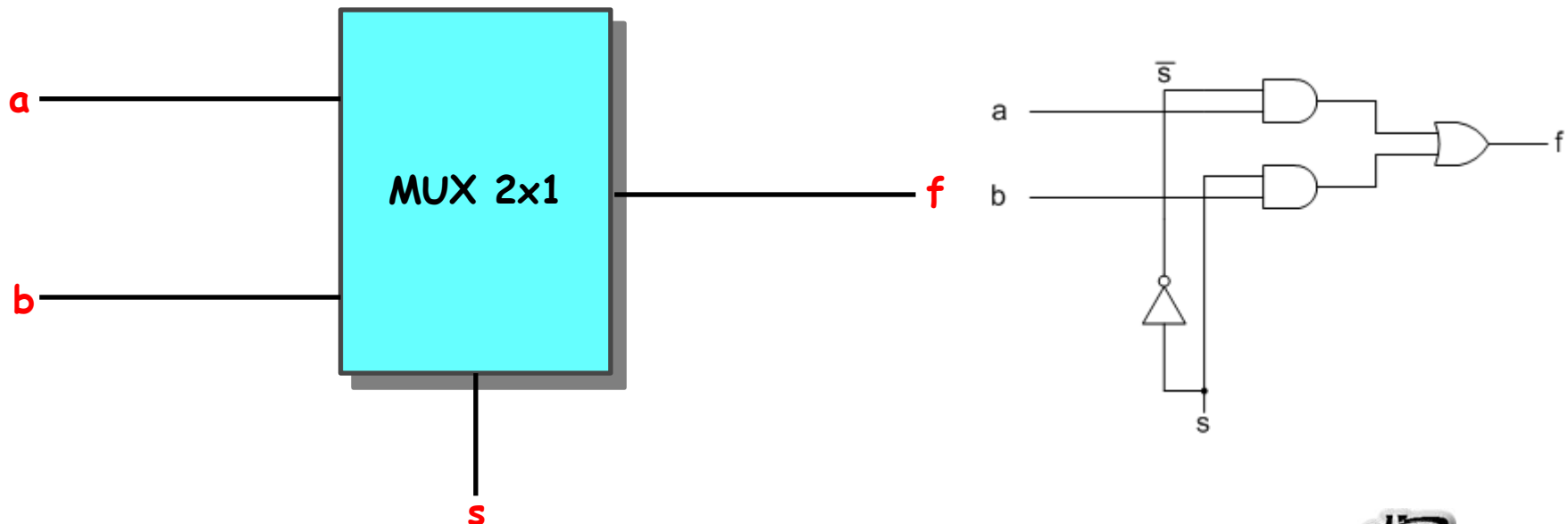
-- Codificador de prioridade usando comando if then else

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY cod_pri_cc IS
    PORT (p : IN BIT_VECTOR(3 DOWNT0 1);
          c : OUT BIT_VECTOR(1 DOWNT0 0));
END cod_pri_cc;
ARCHITECTURE condicional OF cod_pri_cc IS
BEGIN
    PROCESS (p)
    BEGIN
        IF p(3) = '1' THEN c <= "11";
        ELSIF p(2) = '1' THEN c <= "10";
        ELSIF p(1) = '1' THEN c <= "01";
        ELSE c <= "00";
        END IF;
    END PROCESS;
END condicional;
```

# VHDL - Comandos Condicionais

## IF THEN ELSE

- Exercício 04: Implemente um multiplexador de duas entradas de dados (**a** e **b**), uma entrada de seleção (**s**) e uma única saída de dados (**f**). O seu funcionamento basear-se-á na escolha da entrada de seleção que determinará qual das entradas de dados aparecerá na saída de dados.





# VHDL - Comandos Condicionais

## Solução

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY mux_cc_if IS
    PORT (a, b : IN BIT;
          s : IN BIT;
          f : OUT BIT);
END mux_cc_if;

ARCHITECTURE condicional OF mux_cc_if IS
BEGIN
    PROCESS (a, b, s)
    BEGIN
        IF s='0' THEN f<=a;
        ELSE f<=b;
        END IF;
    END PROCESS;
END condicional;
```

-- Multiplexador 2 x 1 usando comando if then else

# VHDL - Comandos Condicionais

## CASE WHEN

- É um comando sequencial com uso dentro de procedimentos, funções e processos.
- Permite a definição de várias condições em um componente.
- Neste comando, as comparações sempre são feitas em torno de um único objeto ou expressão, e será o valor desse objeto ou determinada condição que indicará quais comandos serão executados.

- Sintaxe:

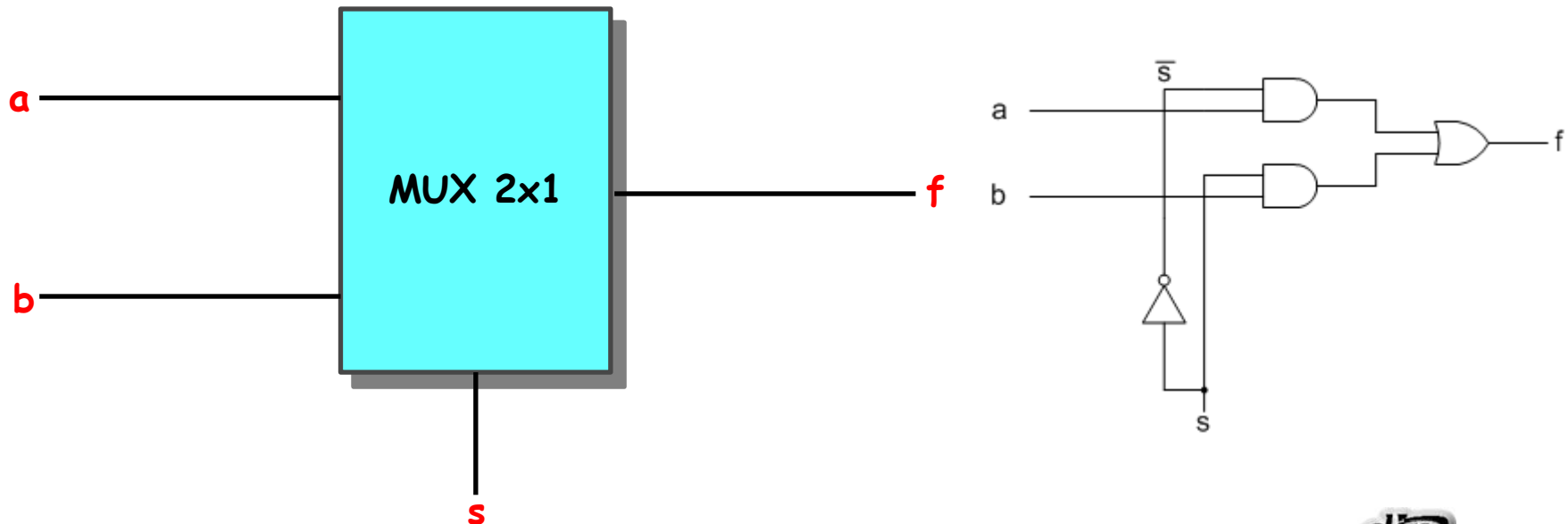
```
CASE expressao_de_escolha IS                                -- expressao_de_escolha =  
    WHEN condicao_1                                          => comando_a;                -- condicao_1  
    WHEN condicao_2                                          => comando_b; comando_c; -- condicao_2  
    WHEN condicao_3 | condicao_4                              => comando_d;                -- condicao_3 ou condicao_4  
    WHEN condicao_5 TO condicao_9                            => comando_d;                -- condicao_5 ate condicao_9  
    WHEN OTHERS                                             => comando_e; comando_f; -- condicoes restantes  
END CASE;
```

- NOTA: O delimitador | equivale a uma operação OU entre as condições de escolha. As palavras reservadas TO e DOWNTO servem para delimitar uma faixa de condições. A palavra reservada OTHERS na última condição serve para agrupar as condições não-relacionadas na lista.

# VHDL - Comandos Condicionais

## CASE WHEN

- Exercício: Implemente um multiplexador de duas entradas de dados (**a** e **b**), uma entrada de seleção (**s**) e uma única saída de dados (**f**). O seu funcionamento basear-se-á na escolha da entrada de seleção que determinará qual das entradas de dados aparecerá na saída de dados.



# VHDL - Comandos Condicionais

## Solução 01

```
LIBRARY ieee;           -- Multiplexador 2 x 1 usando comando case when
USE ieee.std_logic_1164.all;

ENTITY mux_cc_case IS
    PORT (a, b : IN BIT;
          s : IN BIT;
          f : OUT BIT);

END mux_cc_case;

ARCHITECTURE condicional OF mux_cc_case IS
BEGIN

    PROCESS (a, b, s)
    BEGIN

        CASE s IS

            WHEN '0' => f <= a;
            WHEN '1' => f <= b;

        END CASE;

    END PROCESS;

END condicional;
```

# VHDL - Comandos Condicionais

## Solução 02

```
LIBRARY ieee;           -- Multiplexador 2 x 1 usando comando case when
USE ieee.std_logic_1164.all;
ENTITY mux_cc_case IS
    PORT (a, b : IN BIT;
          s : IN BIT;
          f : OUT BIT);
END mux_cc_case;
ARCHITECTURE condicional OF mux_cc_case IS
BEGIN
    PROCESS (a, b, s)
    BEGIN
        CASE s IS
            WHEN '0' => f <= a;
            WHEN OTHERS => f <= b;
        END CASE;
    END PROCESS;
END condicional;
```

# VHDL – Comandos Condicionais

## Solução 03

```
LIBRARY ieee;           -- Multiplexador 2 x 1 usando comando case when
USE ieee.std_logic_1164.all;

ENTITY mux_cc_case IS
    PORT (a, b, s : IN STD_LOGIC;
          f : OUT STD_LOGIC);

END mux_cc_case;

ARCHITECTURE condicional OF mux_cc_case IS
BEGIN

    PROCESS (a, b, s)
    BEGIN

        CASE s IS
            WHEN '0' => f <= a;
            WHEN '1' => f <= b;
            WHEN OTHERS => f <= 'X';

        END CASE;

    END PROCESS;

END condicional;
```

# VHDL – Comandos Condicionais

## WITH SELECT WHEN

- É um comando concorrente.
- Transfere um valor a um sinal de destino segundo uma relação de opções.
- Todas as condições de seleção devem ser consideradas e elas devem ser mutuamente exclusivas.
- A lista de opções nesta construção não contém uma prioridade.

### Sintaxe:

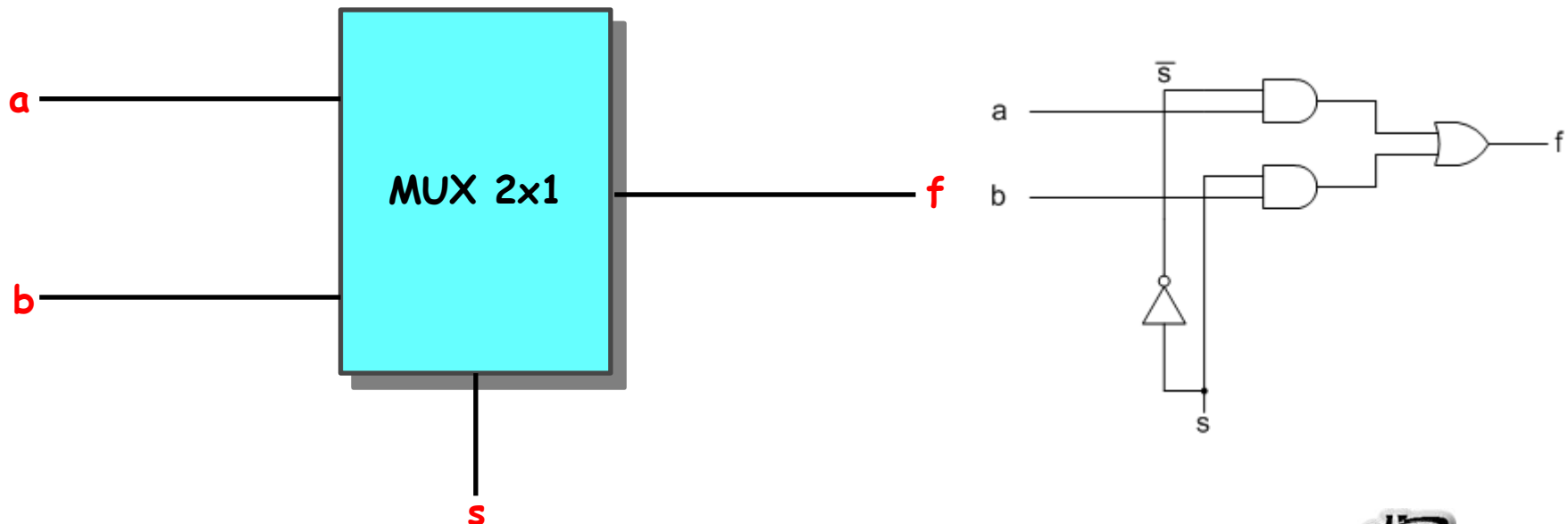
```
WITH expressao_de_escolha SELECT                -- expressao_de_escolha =  
    sinal_destino <= expressao_a WHEN condicao_1,  -- condicao_1  
        expressao_b WHEN condicao_2,              -- condicao_2  
        expressao_c WHEN condicao_3 | condicao_4,  -- condicao_3 ou condicao_4  
        expressao_d WHEN condicao_5 TO condicao_9, -- condicao_5 ate condicao_9  
        expressao_e WHEN OTHERS;                 -- condicoes restantes
```

- NOTA: O delimitador | equivale a uma operação OU entre as condições de escolha. As palavras reservadas TO e DOWNTO servem para delimitar uma faixa de condições. A palavra reservada OTHERS na última condição serve para agrupar as condições não-relacionadas na lista.

# VHDL - Comandos Condicionais

## WITH SELECT WHEN

- Exercício: Implemente um multiplexador de duas entradas de dados (**a** e **b**), uma entrada de seleção (**s**) e uma única saída de dados (**f**). O seu funcionamento basear-se-á na escolha da entrada de seleção que determinará qual das entradas de dados aparecerá na saída de dados.





# VHDL - Comandos Condicionais

## Solução 01

```
LIBRARY ieee; -- Multiplexador 2 x 1 usando comando with select when
USE ieee.std_logic_1164.all;
ENTITY mux_cc_case IS
    PORT (a, b : IN BIT;
          s : IN BIT;
          f : OUT BIT);
END mux_cc_case;
ARCHITECTURE condicional OF mux_cc_case IS
BEGIN
    WITH s SELECT
        f <= a WHEN '0',
            b WHEN '1';
END condicional;
```

# VHDL - Comandos Condicionais

## Solução 02

```
LIBRARY ieee; -- Multiplexador 2 x 1 usando comando with select when
USE ieee.std_logic_1164.all;
ENTITY mux_cc_case IS
    PORT (a, b : IN BIT;
          s : IN BIT;
          f : OUT BIT);
END mux_cc_case;
ARCHITECTURE condicional OF mux_cc_case IS
BEGIN
    WITH s SELECT
        f <= a WHEN '0',
            b WHEN OTHERS;
END condicional;
```

# VHDL – Comandos Condicionais

## Solução 03

```
LIBRARY ieee; -- Multiplexador 2 x 1 usando comando with select when
USE ieee.std_logic_1164.all;

ENTITY mux_cc_case IS
    PORT (a, b, s : IN STD_LOGIC;
          f : OUT STD_LOGIC);
END mux_cc_case;

ARCHITECTURE condicional OF mux_cc_case IS
BEGIN
    WITH s SELECT
        f <= a WHEN '0',
            b WHEN '1',
            'X' WHEN OTHERS;
END condicional;
```

# Resumo da Aula de Hoje

## Tópicos mais importantes:

- **Comandos Condicionais**
  - **Comando WHEN ELSE**
  - **Comando IF THEN ELSE**
  - **Comando CASE WHEN**
  - **Comando WITH SELECT WHEN**

# Próxima da Aula

- **Comandos de Repetição**
  - **Comando *FOR LOOP***
  - **Comando *WHILE LOOP***
  - **Comando *NEXT e EXIT***