



Universidade Federal de Santa Catarina
Centro Tecnológico
Departamento de Informática e Estatística
Curso de Graduação em Ciências da Computação



Sistemas Digitais

INE 5406

Aula 5-P

Processos em VHDL. Comandos de atribuição em VHDL: if-then-else e case. Descrição em VHDL de circuitos combinacionais, de latches e de flip-flops com processos

Prof. José Luís Güntzel
guntzel@inf.ufsc.br

www.inf.ufsc.br/~guntzel/ine5406/ine5406.html

Introdução à Linguagem VHDL

► Comandos de Atribuição em VHDL

VHDL provê os seguintes comandos de atribuição:

- **Simple**
- **Sinal selecionado** (*selected signal assignment*)
- **Sinal condicional** (*conditional signal assignment*)
- **Geração** (*for generate statement*)
- **If-then-else** (*if-then-else statement*)
- **Case** (*case statement*)

Usados
somente dentro
de processos

Introdução à Linguagem VHDL

▶ **Processos**

- São uma forma de gerar uma **avaliação seqüencial (não concorrente)** de atribuições.
- O processo é **concorrente em relação aos outros elementos da arquitetura.**
- A **ordem das atribuições passa a ser relevante.**
- **É usado dentro da arquitetura.**
- **Precisa de uma lista de sinais de sensibilização. Quanto um sinal de sensibilização muda de valor, o processo “acorda” e se torna ativo.**
- **As atribuições que aparecem dentro do processo não são visíveis de fora do processo até que todas as atribuições do processo tenham sido avaliadas.**

Introdução à Linguagem VHDL

▶ Comandos de Atribuição e Processos

Comandos de atribuição VHDL usados somente em processos:

- **If-then-else** (*if-then-else statement*)
- **Case** (*case statement*)

Introdução à Linguagem VHDL

▶ Usando Processo

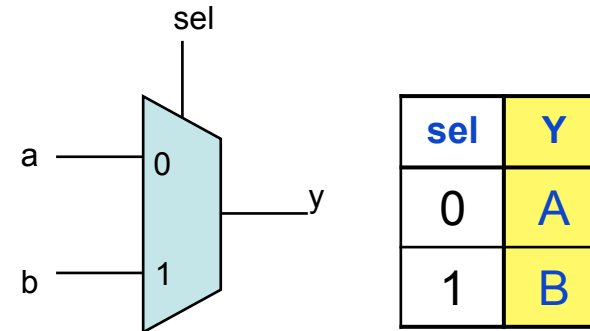
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY mux2para1 IS
PORT ( sel, a, b : IN STD_LOGIC;
      y : OUT STD_LOGIC);
END mux2para1;

ARCHITECTURE comportamento OF mux2para1 IS
BEGIN
  PROCESS (a, b, sel)  -- lista de sensibilização
  BEGIN
    IF sel = '0' THEN
      y <= a;
    ELSE
      y <= b;
    END IF;
  END PROCESS;
END comportamento;
```

Sistemas Digitais - semestre 2011/1

Multiplexador 2:1



Introdução à Linguagem VHDL

▶ Usando Processo (2ª versão)

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

```
ENTITY mux2para1 IS  
PORT ( sel, a, b : IN STD_LOGIC;  
      y : OUT STD_LOGIC);  
END mux2para1;
```

```
ARCHITECTURE comportamento OF mux2para1 IS  
BEGIN
```

```
    PROCESS (a, b, sel)  -- lista de sensibilização
```

```
    BEGIN
```

```
        y <= a;
```

```
        IF sel = '1' THEN
```

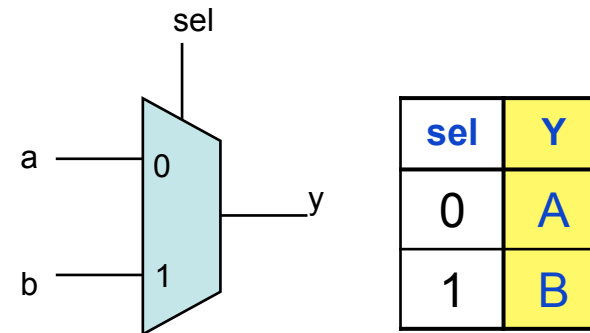
```
            y <= b;
```

```
        END IF;
```

```
    END PROCESS;
```

```
END comportamento;
```

Multiplexador 2:1



Introdução à Linguagem VHDL

▶ Usando Processo

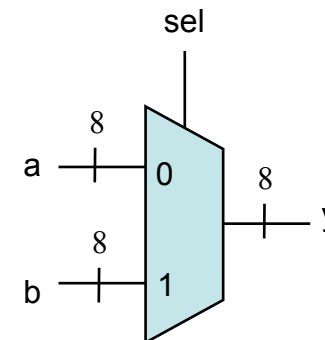
```
LIBRARY ieee;  
USE ieee.std_logic_1164.all;
```

```
ENTITY mux2para1 IS  
PORT ( sel: IN STD_LOGIC;  
      a, b : IN STD_LOGIC_VECTOR (7 DOWNTO 0);  
      y : OUT STD_LOGIC_VECTOR (7 DOWNTO 0) );  
END mux2para1 ;
```

```
ARCHITECTURE comportamento OF mux2para1 IS  
BEGIN  
  PROCESS (a, b, sel)  -- lista de sensibilização  
  BEGIN  
    IF sel = '0' THEN  
      y <= a;  
    ELSE  
      y <= b;  
    END IF;  
  END PROCESS;  
END comportamento;
```

Multiplexador 2:1 no Nível RT

sel	y
0	A
1	B



Introdução à Linguagem VHDL

▶ Usando Processo e Atribuição com Sinal Condicional

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

ENTITY prioridade IS
PORT ( w : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
      y : OUT STD_LOGIC_VECTOR(1 DOWNTO 0);
      z : OUT STD_LOGIC);
END prioridade;

ARCHITECTURE comportamento OF prioridade IS
BEGIN
  PROCESS (w)
  BEGIN
    IF w(3) = '1' THEN
      y <= "11";
    ELSIF w(2) = '1' THEN
      y <= "10";
    ELSIF w(1) = '1' THEN
      y <= "01";
    ELSE
      y <= "00";
    END IF;
  END PROCESS;
  z <= '0' WHEN w = "0000" ELSE '1';
END comportamento;
```

Codificador de prioridade 4:2

w3	w2	w1	w0	y1	y0	z
0	0	0	0	X	X	0
0	0	0	1	0	0	1
0	0	1	X	0	1	1
0	1	X	X	1	0	1
1	X	X	X	1	1	1

Introdução à Linguagem VHDL

▶ Usando Processo

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

ENTITY prioridade IS
PORT ( w : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
      y : OUT STD_LOGIC_VECTOR(1 DOWNTO 0);
      z : OUT STD_LOGIC);
END prioridade;

ARCHITECTURE comportamento OF prioridade IS
BEGIN
  PROCESS (w)
  BEGIN
    y <= "00";
    IF w(1) = '1' THEN y <= "01"; END IF;
    IF w(2) = '1' THEN y <= "10"; END IF;
    IF w(3) = '1' THEN y <= "11"; END IF;

    z <= '1';
    IF w = "0000" THEN z <= '0'; END IF;
  END PROCESS;
END comportamento;
```

(Uma versão alternativa)

Codificador de prioridade 4:2

w3	w2	w1	w0	y1	y0	z
0	0	0	0	X	X	0
0	0	0	1	0	0	1
0	0	1	X	0	1	1
0	1	X	X	1	0	1
1	X	X	X	1	1	1

Introdução à Linguagem VHDL

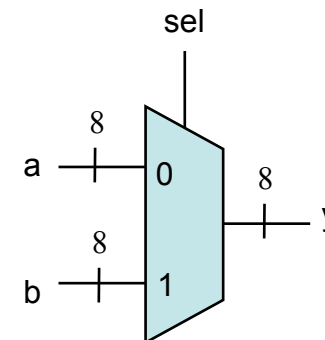
▶ Usando Processo e Case

```
LIBRARY ieee;  
USE ieee.std_logic_1164.all;  
  
ENTITY mux2para1 IS  
PORT ( sel: IN STD_LOGIC;  
      a, b : IN STD_LOGIC_VECTOR (7 DOWNTO 0);  
      y : OUT STD_LOGIC_VECTOR (7 DOWNTO 0) );  
END mux2para1 ;
```

```
ARCHITECTURE comportamento OF mux2para1 IS  
BEGIN  
  PROCESS (a, b, sel) -- lista de sensibilização  
  BEGIN  
    CASE sel IS  
      WHEN '0' => y <= a;  
      WHEN OTHERS => y <= b;  
    END CASE;  
  END PROCESS;  
END comportamento;
```

Multiplexador 2:1

sel	y
0	A
1	B



Introdução à Linguagem VHDL

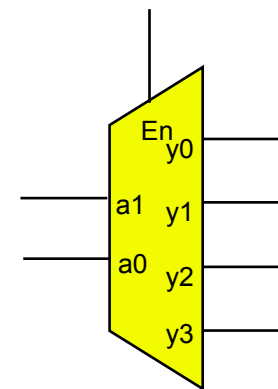
▶ Usando Processo

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

ENTITY dec2para4 IS
PORT ( a : IN STD_LOGIC_VECTOR (1 DOWNTO 0);
      En : IN STD_LOGIC;
      y : OUT STD_LOGIC_VECTOR (0 TO 3) );
END dec2para4;

ARCHITECTURE comportamento OF dec2para4 IS
BEGIN
  PROCESS (a, En)
  BEGIN
    IF En = '1' THEN
      CASE a IS
        WHEN "00" => y <= "1000";
        WHEN "01" => y <= "0100";
        WHEN "10" => y <= "0010";
        WHEN OTHERS => y <= "0001";
      END CASE;
    ELSE
      y <= "0000";
    END IF;
  END PROCESS;
END comportamento;
```

Decodificador 2:4



En	a1	a0	y0	y1	y2	y3
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

Prof. José Luís Güntzel

Introdução à Linguagem VHDL

▶ Usando Processo

Descrição de uma ULA (equivalente ao TTL74381)

Funcionalidade:

Operação	Controle ('sel')	Saída ('F')
Clear	0 0 0	0 0 0 0
B - A	0 0 1	B - A
A - B	0 1 0	A - B
ADD	0 1 1	A + B
XOR	1 0 0	A XOR B
OR	1 0 1	A OR B
AND	1 1 0	A AND B
Preset	1 1 1	1 1 1 1

Introdução à Linguagem VHDL

▶ Usando Processo

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

ENTITY ula74381 IS
PORT ( s : IN STD_LOGIC_VECTOR (2 DOWNTO 0);
      A, B : IN STD_LOGIC_VECTOR (3 DOWNTO 0);
      F : OUT STD_LOGIC_VECTOR (3 DOWNTO 0));
END ula74381;

ARCHITECTURE comportamento OF ula74381 IS
BEGIN
  PROCESS (s, A, B)
  BEGIN
    CASE s IS
      WHEN "000" => F <= "0000";
      WHEN "001" => F <= B - A;
      WHEN "010" => F <= A - B;
      WHEN "011" => F <= A + B;
      WHEN "100" => F <= A XOR B;
      WHEN "101" => F <= A OR B;
      WHEN "110" => F <= A AND B;
      WHEN OTHERS => F <= "1111";
    END CASE;
  END PROCESS;
END comportamento;
```

Introdução à Linguagem VHDL

▶ Comparador Usando Processo

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

ENTITY compara IS
PORT ( a, b : IN STD_LOGIC;
      AeqB : OUT STD_LOGIC);
END compara;

ARCHITECTURE comportamento OF compara IS
BEGIN
  PROCESS (a,b)
  BEGIN
    IF a = b THEN
      AeqB <= '1';
    END IF;
  END PROCESS;
END comportamento;
```

Comparador de 1 bit

Onde está o erro?

Introdução à Linguagem VHDL

▶ Comparador Usando Processo

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

ENTITY compara IS
PORT ( a, b : IN STD_LOGIC;
      AeqB : OUT STD_LOGIC);
END compara;

ARCHITECTURE comportamento OF compara IS
BEGIN
  PROCESS (a,b)
  BEGIN
    AeqB <= '0';
    IF a = b THEN
      AeqB <= '1';
    END IF;
  END PROCESS;
END comportamento;
```

Comparador de 1 bit

Versão correta

Introdução à Linguagem VHDL

▶ Comparador Usando Processo

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

Comparador para números de 8 bit

```
ENTITY compara IS  
PORT (a, b : IN STD_LOGIC_VECTOR(7 DOWNTO 0);  
      AeqB : OUT STD_LOGIC);
```

```
END compara;
```

```
ARCHITECTURE comportamento OF compara IS
```

```
BEGIN
```

```
    PROCESS (a,b)
```

```
    BEGIN
```

```
        AeqB <= '0';
```

```
        IF a = b THEN
```

```
            AeqB <= '1';
```

```
        END IF;
```

```
    END PROCESS;
```

```
END comportamento;
```

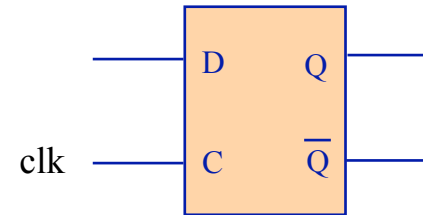

Introdução à Linguagem VHDL

▶ Latch D (ativado por nível lógico alto)

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

ENTITY latchD IS
PORT ( D, clk : IN STD_LOGIC;
      Q :      OUT STD_LOGIC);
END latchD;

ARCHITECTURE comportamento OF latchD IS
BEGIN
  PROCESS (D, clk)
  BEGIN
    IF clk = '1' THEN
      Q <= D;
    END IF;
  END PROCESS;
END comportamento;
```



C	D	Q_{t+1}
0	X	Q_t
1	0	0
1	1	1

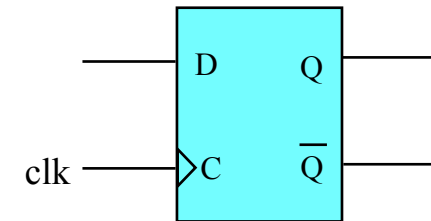
Introdução à Linguagem VHDL

▶ Flip-flop D (disparado pela borda ascendente)

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

```
ENTITY ffD IS  
PORT ( D, clk : IN STD_LOGIC;  
      Q : OUT STD_LOGIC);  
END ffD;
```

```
ARCHITECTURE comportamento OF ffD IS  
BEGIN  
  PROCESS (clk)  
  BEGIN  
    IF clk'EVENT AND clk = '1' THEN  
      Q <= D;  
    END IF;  
  END PROCESS;  
END comportamento;
```



C	D	Q_{t+1}
$\neq \uparrow$	X	Q_t
\uparrow	0	0
\uparrow	1	1

Atributo: refere-se a troca de nível de clk