



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



# **Sistemas Digitais**

**INE 5406**

## **Aula 2-P**

**Introdução à Linguagem VHDL: descrição de somadores,  
síntese e simulação.**

**Prof. José Luís Güntzel**  
**[guntzel@inf.ufsc.br](mailto:guntzel@inf.ufsc.br)**

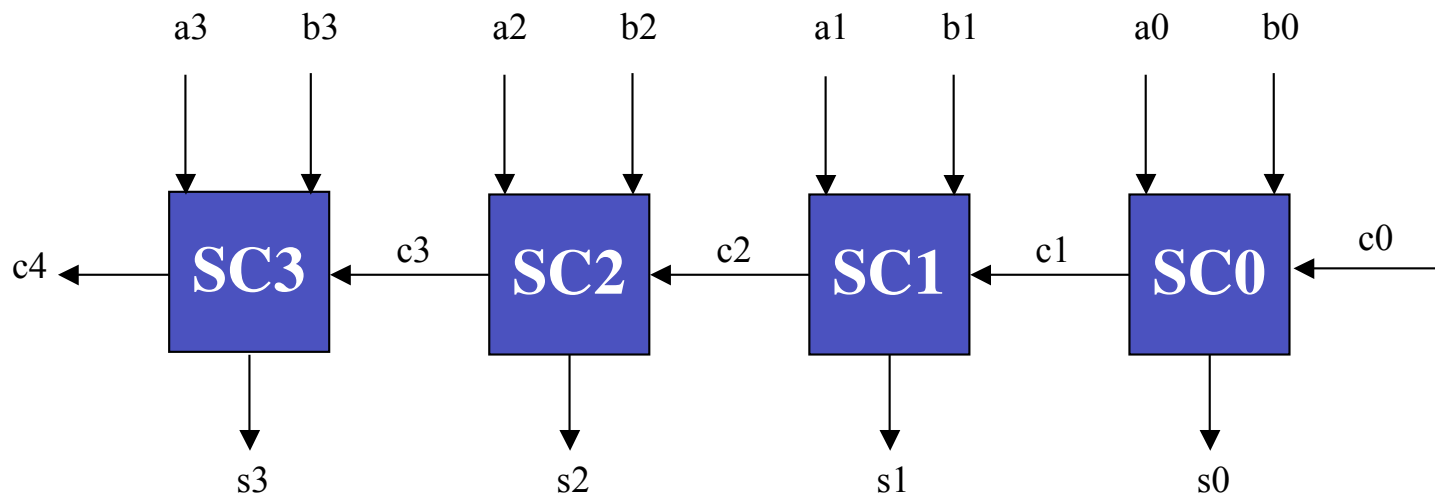
**[www.inf.ufsc.br/~guntzel/ine5406/ine5406.html](http://www.inf.ufsc.br/~guntzel/ine5406/ine5406.html)**

# Noções de Linguagem VHDL

## ► Descrição de um Operador Aritmético

Somador de 4 bits:

- Com hierarquia
- Usando 4 somadores completos de 1 bit

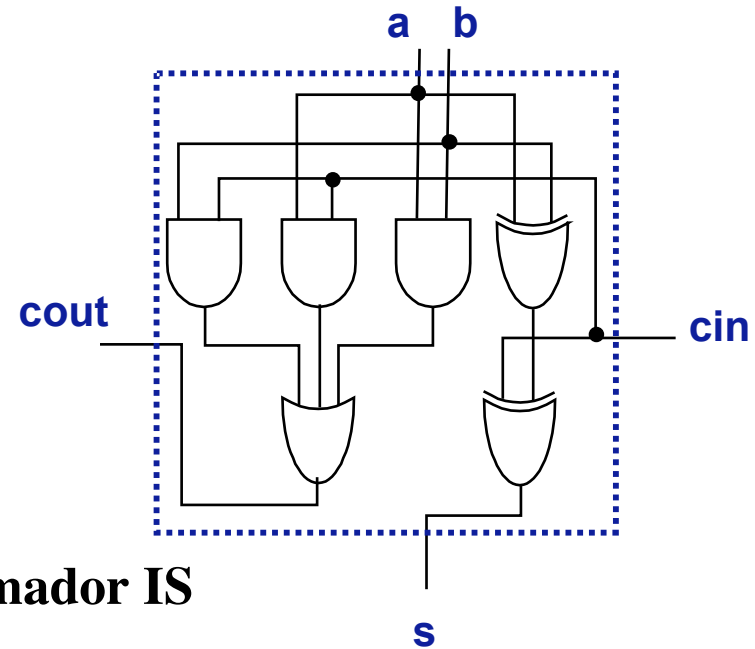


# Noções de Linguagem VHDL

## ► Somador Completo de 1 bit

```
LIBRARY ieee;  
USE ieee.std_logic_1164.all;  
  
ENTITY somador IS  
    PORT (cin, a, b : IN STD_LOGIC;  
          s, cout : OUT STD_LOGIC);  
END somador;
```

```
ARCHITECTURE comportamento OF somador IS  
BEGIN  
    s <= a XOR b XOR cin;  
    cout <= (a AND b) OR (a AND cin) OR (b AND cin);  
END comportamento;
```



# Noções de Linguagem VHDL

## Somador de 4 bits (Nível Mais Alto da Hierarquia )

```
LIBRARY ieee;
```

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

```
ENTITY somador4bits IS
```

```
    PORT (c0, a3, a2, a1, a0, b3, b2, b1, b0 : IN STD_LOGIC;  
          s3, s2, s1, s0, c4 : OUT STD_LOGIC);
```

```
END somador4bits;
```

```
ARCHITECTURE estrutura OF somador4bits IS
```

```
    SIGNAL c1, c2, c3: STD_LOGIC;
```

```
    COMPONENT somador
```

```
        PORT (cin, a, b : IN STD_LOGIC;  
              s, cout : OUT STD_LOGIC);
```

```
END COMPONENT;
```

```
BEGIN
```

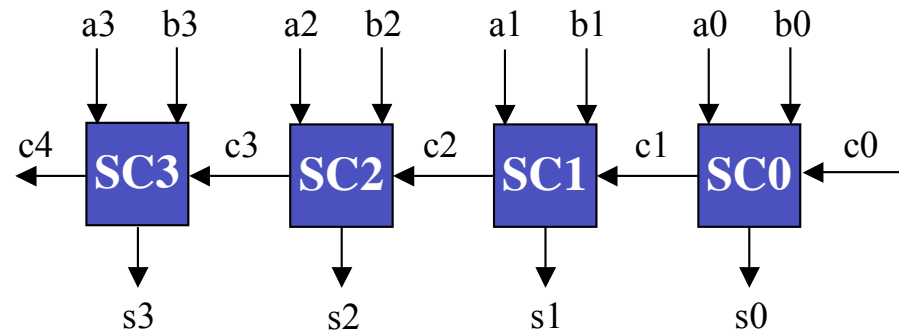
```
    SC0: somador PORT MAP (c0, a0, b0, s0, c1);
```

```
    SC1: somador PORT MAP (c1, a1, b1, s1, c2);
```

```
    SC2: somador PORT MAP (c2, a2, b2, s2, c3);
```

```
    SC3: somador PORT MAP (c3, a3, b3, s3, c4);
```

```
END estrutura;
```



# Noções de Linguagem VHDL

## Somador de 4 bits

**BEGIN**

**SC0: somador PORT MAP (c0, a0, b0, s0, c1);**

**SC1: somador**

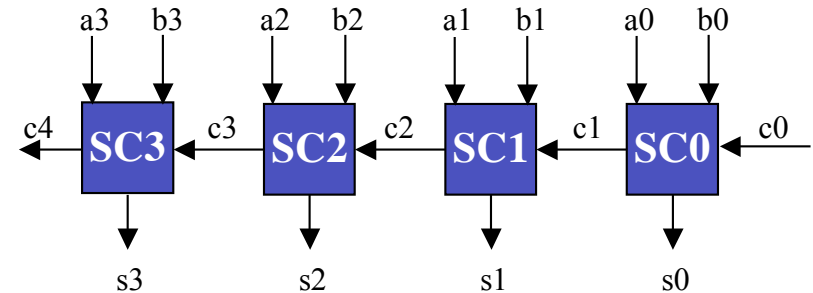
**PORT MAP (cin => c1,  
a => a1,  
b => b1,  
s => s1,  
cout => c2);**

**SC2: somador**

**PORT MAP (s => s2,  
b => b2,  
a => a2,  
cin => c2,  
cout => c3);**

**SC3: somador PORT MAP (c3, a3, b3, s3, c4);**

**END comportamento;**



**Quando o mapeamento dos pinos é explícito, não é necessário seguir a ordem da declaração**

# Noções de Linguagem VHDL

## Somador de 4 bits sem hierarquia

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

```
ENTITY somador4bits IS
```

```
    PORT (cin, a3, a2, a1, a0, b3, b2, b1, b0 : IN STD_LOGIC;  
          s3, s2, s1, s0, cout : OUT STD_LOGIC);
```

```
END somador4bits;
```

```
ARCHITECTURE comportamento OF somador4bits IS
```

```
    SIGNAL c1, c2, c3: STD_LOGIC;
```

```
BEGIN
```

```
    s0 <= a0 XOR b0 XOR c0;
```

```
    c1 <= (a0 AND b0) OR (a0 AND c0) OR (b0 AND c0);
```

```
    s1 <= a1 XOR b1 XOR c1;
```

```
    c2 <= (a1 AND b1) OR (a1 AND c1) OR (b1 AND c1);
```

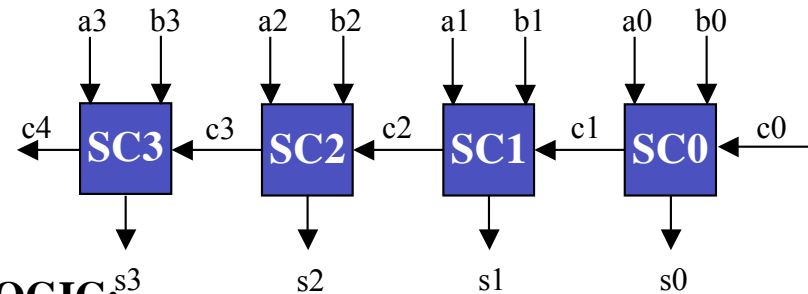
```
    s2 <= a2 XOR b2 XOR c2;
```

```
    c3 <= (a2 AND b2) OR (a2 AND c2) OR (b2 AND c2);
```

```
    s3 <= a3 XOR b3 XOR c3;
```

```
    c4 <= (a3 AND b3) OR (a3 AND c3) OR (b3 AND c3);
```

```
END comportamento;
```



# Noções de Linguagem VHDL

## ► Representação de Números em VHDL

São sinais em múltiplos fios

**Exemplo:**

(...)

```
SIGNAL teste : STD_LOGIC_VECTOR ( 1 TO 3);
```

```
SIGNAL outro : STD_LOGIC_VECTOR ( 3 DOWNTO 0);
```

(...)

```
teste <= "011";    OU teste(1) <= '0'; teste(2) <= '1'; teste(3) <= '1';
```

```
outro <= "1101";  OU outro(3) <= '1'; outro(2) <= '1'; outro(1) <= '0'; outro(0) <= '1';
```

```
OU outro(3) <= '1'; outro(2 DOWNTO 0) <= "101";
```

# Noções de Linguagem VHDL

## Somador de 4 bits com hierarquia

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

```
ENTITY somador4bits IS
```

```
  PORT (c0 : IN STD_LOGIC;
```

```
        a, b : IN STD_LOGIC_VECTOR (3 DOWNTO 0);
```

```
        s : OUT STD_LOGIC_VECTOR (3 DOWNTO 0);
```

```
        c4 : OUT STD_LOGIC);
```

```
END somador4bits;
```

```
ARCHITECTURE comportamento OF somador4bits IS
```

```
  SIGNAL c : STD_LOGIC_VECTOR (3 DOWNTO 1);
```

```
  COMPONENT somador
```

```
    PORT (cin, a, b : IN STD_LOGIC;
```

```
          s, cout : OUT STD_LOGIC);
```

```
END COMPONENT;
```

```
BEGIN
```

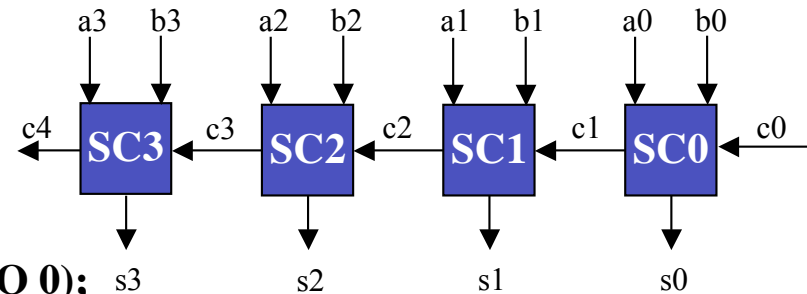
```
  SC0: somador PORT MAP (c0, a(0), b(0), s(0), c(1));
```

```
  SC1: somador PORT MAP (c(1), a(1), b(1), s(1), c(2));
```

```
  SC2: somador PORT MAP (c(2), a(2), b(2), s(2), c(3));
```

```
  SC3: somador PORT MAP (c(3), a(3), b(3), s(3), c4);
```

```
END comportamento;
```





# Noções de Linguagem VHDL

## Somador de 4 bit sem hierarquia

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

```
ENTITY somador4bits IS
```

```
  PORT (c0 : IN STD_LOGIC;
```

```
        a, b : IN STD_LOGIC_VECTOR (3 DOWNTO 0);
```

```
        s : OUT STD_LOGIC_VECTOR (3 DOWNTO 0);
```

```
        c4 : OUT STD_LOGIC);
```

```
END somador4bits;
```

```
ARCHITECTURE comportamento OF somador4bits IS
```

```
  SIGNAL c : STD_LOGIC_VECTOR (3 DOWNTO 1);
```

```
BEGIN
```

```
  s(0) <= a(0) XOR b(0) XOR c0;
```

```
  c(1) <= (a(0) AND b(0)) OR (a(0) AND cin) OR (b(0) AND c0);
```

```
  s(1) <= a(1) XOR b(1) XOR c(1);
```

```
  c(2) <= (a(1) AND b(1)) OR (a(1) AND c(1)) OR (b(1) AND c(1));
```

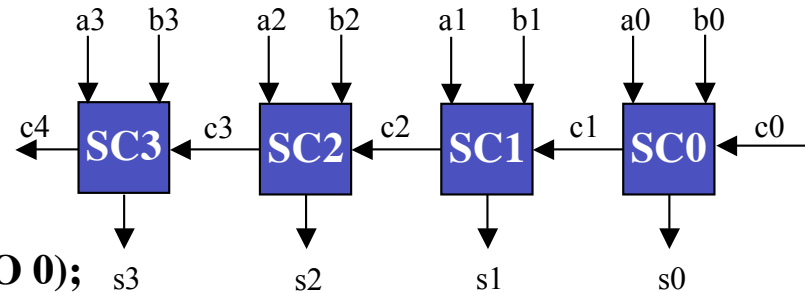
```
  s(2) <= a(2) XOR b(2) XOR c(2);
```

```
  c(3) <= (a(2) AND b(2)) OR (a(2) AND c(2)) OR (b(2) AND c(2));
```

```
  s(3) <= a(3) XOR b(3) XOR c(3);
```

```
  c4 <= (a(3) AND b(3)) OR (a(3) AND c(3)) OR (b(3) AND c(3));
```

```
END comportamento;
```



# Noções de Linguagem VHDL

## ▶ Atribuição de Operações Aritméticas

Se:

```
SIGNAL A, B, S : STD_LOGIC_VECTOR ( 15 DOWNTO 0);
```

Então:

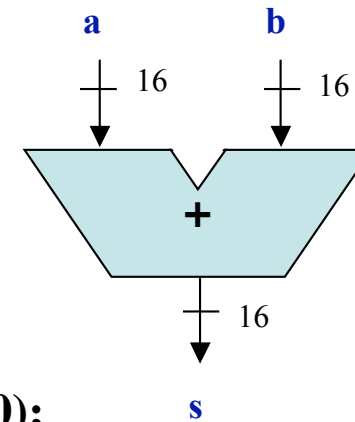
```
S <= A + B;
```

representa um somador de 16 bits...

# Noções de Linguagem VHDL

## Somador de 16 bits sem hierarquia

```
LIBRARY ieee;  
USE ieee.std_logic_1164.all;  
USE ieee.std_logic_signed.all;  
  
ENTITY somador16bits IS  
PORT ( a, b : IN STD_LOGIC_VECTOR (15 DOWNTO 0);  
      s : OUT STD_LOGIC_VECTOR (15 DOWNTO 0));  
END somador16bits;  
ARCHITECTURE comportamento OF somador16bits IS  
  
BEGIN  
    s <= a + b;  
END comportamento;
```



**A maior parte dos dispositivos FPGAs possuem estruturas otimizadas que facilitam a implementação da adição**