Circuitos Digitais II - 6882

André Barbosa Verona
Nardênio Almeida Martins

Universidade Estadual de Maringá
Departamento de Informática

Bacharelado em Ciência da Computação

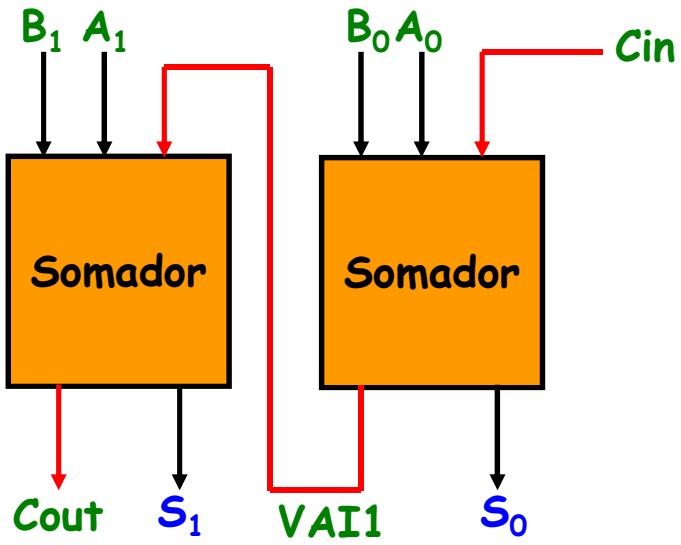
Aula de Hoje

Projeto e Simulação de um circuito Somador Completo de 2 bits usando funções e procedimentos

VHDL – Funções e Procedimentos

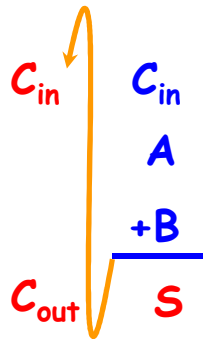
- **Exercício:**
- **Implemente um código para o circuito somador de 2 bits usando Funções e Procedimentos.**
- **Use os seguintes nomes para as entradas e saídas:**
  - **A,B: para os dados;**
  - **Cin para a entrada do vem-1;**
  - **VAI1 para o vai-1 interno;**
  - **Cout para a saída do vai-1 final;**
  - **S para a saída da soma.**
  - **Veja figura a seguir**

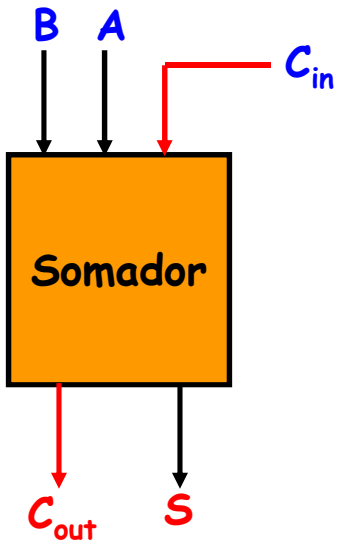
Somador Completo de 2 bits



Somador Completo de 1 Bit

Entradas			Saídas	
A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

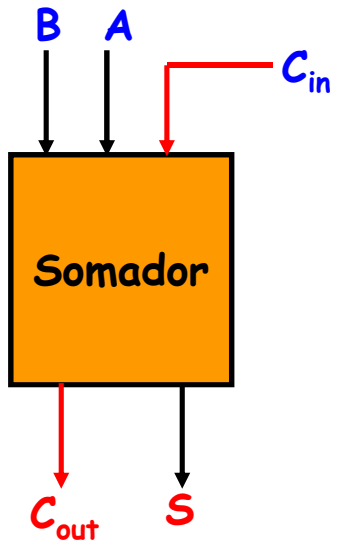




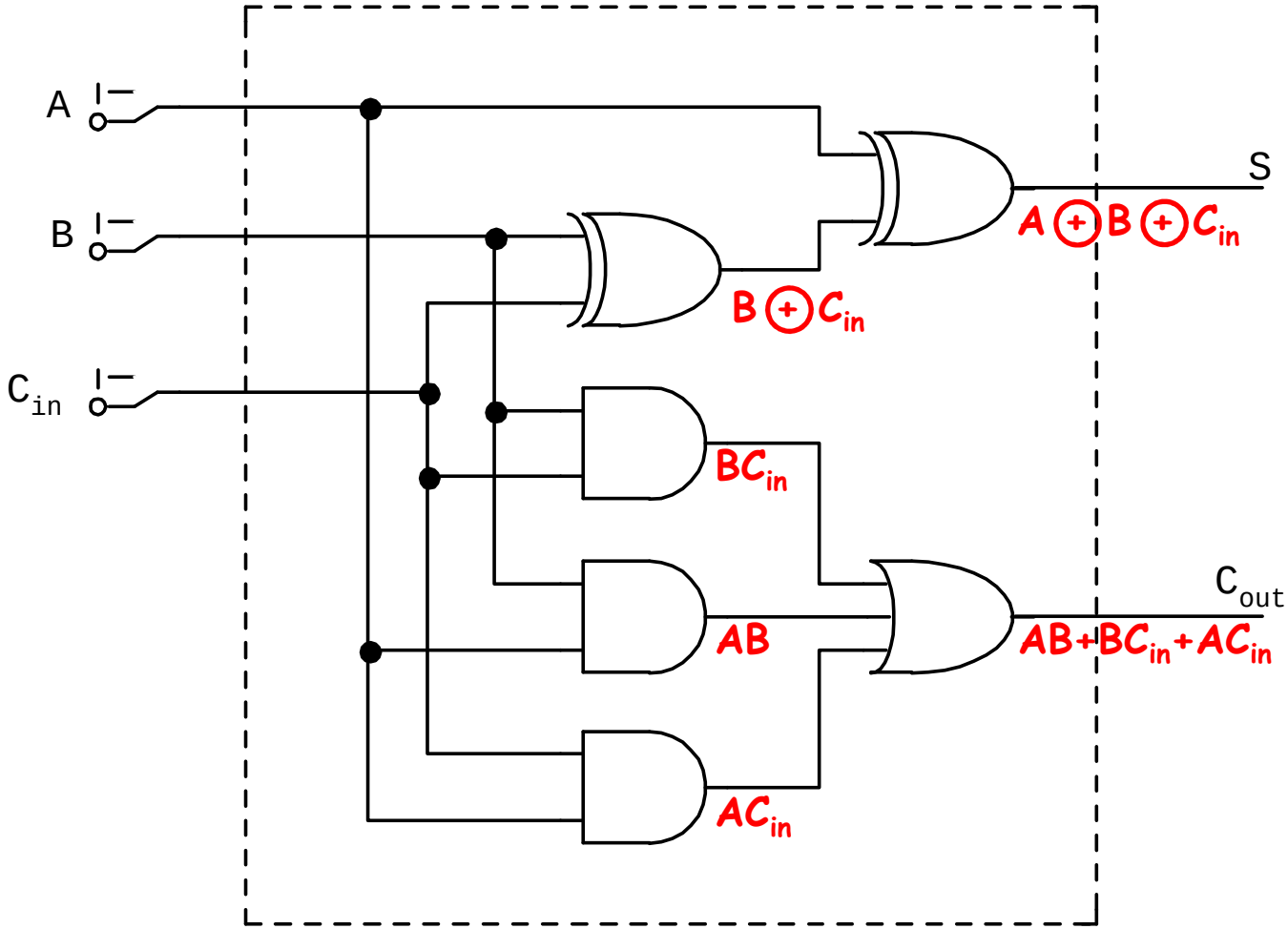
Somador Completo de 1 bit

$S = A \oplus B \oplus C_{in}$

$C_{out} = AB + BC_{in} + AC_{in}$



Circuito Somador Completo de 1 bitt



VHDL – Funções e Procedimentos

Criar as seguintes pastas dentro do diretório "work":

1. som_2_bits
2. som_2_bits_p
3. som_2_bits_e
4. som_2_bits_pp

Uso de Pacotes em VHDL

Criar os pacotes para os seguintes projetos:

som_2_bits_p
som_2_bits_pp

OBS:

- Salvar os códigos dos pacotes nas respectivas pastas dos projetos.

- Os nomes dos pacotes são:
  - som_2_bits_package.vhd --funcao
  - som_2_bits_package_p.vhd --procedimento

Uso de Pacotes em VHDL

Criar os projetos para os seguintes circuitos:

1. **som_2_bits**
2. **som_2_bits_p**
3. **som_2_bits_e**
4. **som_2_bits_pp**

OBS:

- **Na janela de adição de arquivos, adicione o seguinte arquivo ao projeto:**
  - **Para o som_2_bits_p:**
    - som_2_bits_package.vhd --funcao
  - **Para o som_2_bits_pp:**
    - som_2_bis_package_p.vhd --procedimento

VHDL – Funções e Procedimentos

Solução 01: Inserção no Corpo da Arquitetura da Entidade

```
LIBRARY ieee;
USE ieee.Std_Logic_1164.all;

ENTITY som_2_bits IS

  GENERIC (n: INTEGER  := 2);                          -- numero de bits

  PORT     (A, B : IN  BIT_VECTOR (n-1 DOWNTO 0);      -- entradas do somador
            Cin : IN  BIT;                             -- vem 1
            S : OUT BIT_VECTOR (n-1 DOWNTO 0);         -- soma
            Cout : OUT BIT);                           -- vai 1

END som_2_bits;
```

VHDL – Funções e Procedimentos

Solução 01: Inserção no Corpo da Arquitetura da Entidade

```
ARCHITECTURE function_add OF som_2_bits IS

  CONSTANT m : INTEGER := 2;

  FUNCTION Soma(X : BIT_VECTOR; Y : BIT_VECTOR; VEM1 : BIT) RETURN BIT_VECTOR IS

          VARIABLE VAI1 : BIT_VECTOR(m DOWNTO 0);
          VARIABLE RES : BIT_VECTOR(m-1 DOWNTO 0);

          BEGIN

              VAI1(0) := VEM1;
              loopSoma : FOR i IN 0 TO m-1 LOOP
                    RES(i) := X(i) XOR Y(i) XOR VAI1(i);
                    VAI1(i+1) := (X(i) AND Y(i)) OR (Y(i) AND VAI1(i)) OR (X(i) AND VAI1(i));
              END LOOP loopSoma;
              RETURN RES;                          -- Resultado de Soma

          END Soma;
```

VHDL – Funções e Procedimentos

Solução 01: Inserção no Corpo da Arquitetura da Entidade

```
FUNCTION Couti(X : BIT_VECTOR; Y : BIT_VECTOR; VEM1 : BIT) RETURN BIT IS

        VARIABLE VAI1 : BIT_VECTOR(m DOWNTO 0);

        BEGIN

            VAI1(0) := VEM1;
            loopVai1 : FOR i IN 0 TO m-1 LOOP
                  VAI1(i+1) := (X(i) AND Y(i)) OR (Y(i) AND VAI1(i)) OR (X(i) AND VAI1(i));
            END LOOP loopVai1;
            RETURN VAI1(m);                    -- Resultado do Vai 1

        END Couti;

BEGIN
                                               --regiao de codigo sequencial
        PROCESS (A, B, Cin)
                BEGIN
                        S <= Soma(A, B, Cin);          --Soma
                        Cout <= Couti(A, B, Cin);      -- Vai 1
                END PROCESS;

END function_add;
```

VHDL – Funções e Procedimentos

Solução 02: Inserção no Pacote

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

PACKAGE som_2_bits_package IS

   FUNCTION Soma(X : BIT_VECTOR; Y : BIT_VECTOR; VEM1 : BIT) RETURN BIT_VECTOR;

   FUNCTION Couti(X : BIT_VECTOR; Y : BIT_VECTOR; VEM1 : BIT) RETURN BIT;

END som_2_bits_package;
```

VHDL – Funções e Procedimentos

Solução 02: Inserção no Pacote

```
PACKAGE BODY som_2_bits_package IS

  CONSTANT m : INTEGER := 2;

  FUNCTION Soma(X : BIT_VECTOR; Y : BIT_VECTOR; VEM1 : BIT) RETURN BIT_VECTOR IS

          VARIABLE VAI1 : BIT_VECTOR(m DOWNTO 0);
          VARIABLE RES : BIT_VECTOR(m-1 DOWNTO 0);

          BEGIN

              VAI1(0) := VEM1;
              loopSoma: FOR i IN 0 TO m-1 LOOP
                    RES(i) := X(i) XOR Y(i) XOR VAI1(i);
                    VAI1(i+1) := (X(i) AND Y(i)) OR (Y(i) AND VAI1(i)) OR (X(i) AND VAI1(i));
              END LOOP loopSoma;
              RETURN RES;                          -- Resultado de Soma

          END Soma;
```

VHDL – Funções e Procedimentos

Solução 02: Inserção no Pacote

```
FUNCTION Couti(X : BIT_VECTOR; Y : BIT_VECTOR; VEM1 : BIT) RETURN BIT IS

        VARIABLE VAI1 : BIT_VECTOR(m DOWNTO 0);

        BEGIN

            VAI1(0) := VEM1;
            loopVai1: FOR i IN 0 TO m-1 LOOP
                  VAI1(i+1) := (X(i) AND Y(i)) OR (Y(i) AND VAI1(i)) OR (X(i) AND VAI1(i));
            END LOOP loopVai1;
            RETURN VAI1(m);                    -- Resultado do Vai 1

        END Couti;

END som_2_bits_package;
```

VHDL – Funções e Procedimentos

Solução 02: Inserção no Pacote

```
LIBRARY ieee;
USE ieee.Std_Logic_1164.all;

-- LIBRARY work;
USE work.som_2_bits_package.all;

ENTITY som_2_bits_p IS

  GENERIC (n      : INTEGER  := 2);                    -- numero de bits

  PORT     (A, B  : IN  BIT_VECTOR (n-1 DOWNTO 0);     -- entradas do somador
            Cin   : IN  BIT;                           -- vem 1
            S     : OUT BIT_VECTOR (n-1 DOWNTO 0);     -- soma
            Cout  : OUT BIT);                          -- vai 1

END som_2_bits_p;
```

VHDL – Funções e Procedimentos

Solução 02: Inserção no Pacote

```
ARCHITECTURE function_add OF som_2_bits_p IS

BEGIN                                          --regiao de codigo sequencial

        PROCESS (A, B, Cin)

                BEGIN

                        S <= Soma (A, B, Cin);          --Soma
                        Cout <= Couti(A, B, Cin);       -- Vai 1

                END PROCESS;

END function_add;
```

VHDL – Funções e Procedimentos

Solução 03: Inserção no Corpo da Declaração da Entidade

```
LIBRARY ieee;
USE ieee.Std_Logic_1164.all;

ENTITY som_2_bits_e IS

  GENERIC (n      : INTEGER  := 2);                    -- numero de bits

  PORT     (A, B  : IN  BIT_VECTOR (n-1 DOWNTO 0);     -- entradas do somador
            Cin   : IN  BIT;                           -- vem 1
            S     : OUT BIT_VECTOR (n-1 DOWNTO 0);     -- soma
            Cout  : OUT BIT);                          -- vai 1
```

VHDL – Funções e Procedimentos

Solução 03: Inserção no Corpo da Declaração da Entidade

```
CONSTANT m : INTEGER := 2;

PROCEDURE Soma(X:IN BIT_VECTOR(m-1 DOWNTO 0);Y:IN BIT_VECTOR(m-1 DOWNTO 0);
                VEM1:IN BIT; SIGNAL RES:OUT BIT_VECTOR(m-1 DOWNTO 0);
                SIGNAL VAI1:OUT BIT) IS

        VARIABLE VAI1i : BIT_VECTOR(m DOWNTO 0);

        BEGIN

            VAI1i(0) := VEM1;
            loopSoma: FOR i IN 0 TO m-1 LOOP
                  RES(i) <= X(i) XOR Y(i) XOR VAI1i(i);
                  VAI1i(i+1) := (X(i) AND Y(i)) OR (Y(i) AND VAI1i(i)) OR (X(i) AND VAI1i(i));
            END LOOP loopSoma;
             VAI1 <= VAI1i(m);

        END Soma;

END som_2_bits_e;
```

VHDL – Funções e Procedimentos

Solução 03: Inserção no Corpo da Declaração da Entidade

```
ARCHITECTURE procedure_add OF som_2_bits_e IS

BEGIN                                          --regiao de codigo concorrente

                Soma(A, B, Cin, S, Cout);      -- Soma e Vai 1

END procedure_add;
```

VHDL – Funções e Procedimentos

Solução 04: Inserção no Pacote

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

PACKAGE som_2_bits_package_p IS

        PROCEDURE Soma(X : IN BIT_VECTOR; Y : IN BIT_VECTOR;
                       VEM1 : IN BIT; SIGNAL RES : OUT BIT_VECTOR;
                       SIGNAL VAI1 : OUT BIT);

END som_2_bits_package_p;
```

VHDL – Funções e Procedimentos

Solução 04: Inserção no Pacote

```
PACKAGE BODY som_2_bits_package_p IS

  CONSTANT m : INTEGER := 2;

  PROCEDURE Soma(X:IN BIT_VECTOR(m-1 DOWNTO 0); Y:IN BIT_VECTOR(m-1 DOWNTO 0);
                 VEM1:IN BIT; SIGNAL RES:OUT BIT_VECTOR(m-1 DOWNTO 0);
                 SIGNAL VAI1:OUT BIT) IS

        VARIABLE VAI1i : BIT_VECTOR(m DOWNTO 0);

        BEGIN

            VAI1i(0) := VEM1;
            loopSoma: FOR i IN 0 TO m-1 LOOP
                  RES(i) <= X(i) XOR Y(i) XOR VAI1i(i);
                  VAI1i(i+1) := (X(i) AND Y(i)) OR (Y(i) AND VAI1i(i)) OR (X(i) AND VAI1i(i));
            END LOOP loopSoma;
            VAI1 <= VAI1i(m);

        END Soma;

END som_2_bits_package_p;
```

VHDL – Funções e Procedimentos

Solução 04: Inserção no Pacote

```
LIBRARY ieee;
USE ieee.Std_Logic_1164.all;

-- LIBRARY work;
USE work.som_2_bits_package_p.all;

ENTITY som_2_bits_pp IS

  GENERIC (n      : INTEGER  := 2);                          -- numero de bits

  PORT     (A, B  : IN  BIT_VECTOR (n-1 DOWNTO 0);           -- entradas do somador
            Cin   : IN  BIT;                                  -- vem 1
            S     : OUT BIT_VECTOR (n-1 DOWNTO 0);           -- soma
            Cout  : OUT BIT);                                 -- vai 1

END som_2_bits_pp;

ARCHITECTURE procedure_add OF som_2_bits_pp IS

BEGIN                                                         --regiao de codigo concorrente

                  Soma(A, B, Cin, S, Cout);                   -- Soma e Vai 1

END procedure_add;
```