

Universidade Tecnológica Federal do Paraná – Toledo
Engenharia da Computação – COENC

Lógica Reconfigurável

Código Concorrente
WHEN-ELSE

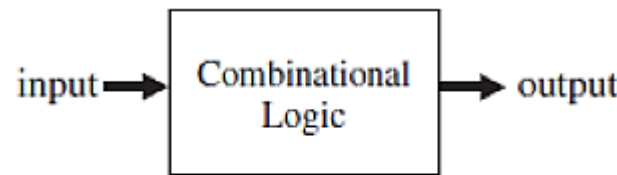
Tiago Piovesan Vendruscolo



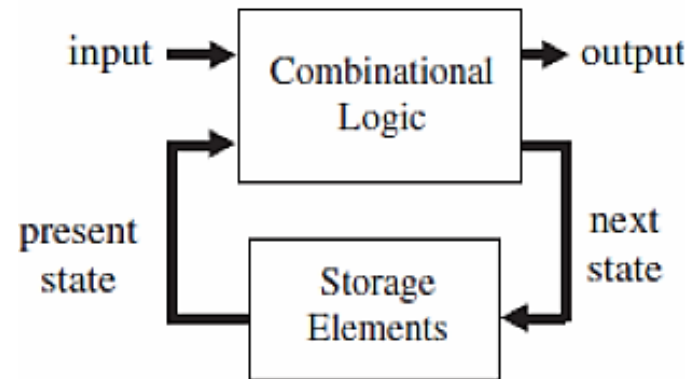
Esta licença permite que outros remixem, adaptem e criem a partir do trabalho para fins não comerciais, desde que atribuam o devido crédito aos autores originais. [4.0 international](https://creativecommons.org/licenses/by-nc-nd/4.0/)

Código concorrente x código sequencial

- ❑ Lógica Combinacional: As saídas dependem apenas das entradas atuais;
- ❑ Lógica Sequencial: As saídas dependem das entradas anteriores.
 - *Elementos de memória são necessários;*



Lógica Combinacional



Lógica Sequencial

Código concorrente x código sequencial

- Códigos em VHDL funcionam por padrão de forma concorrente.
- Em códigos concorrentes usa-se:
 - Operadores;
 - WHEN (WHEN/ELSE ou WITH/SELECT/WHEN);
 - GENERATE;
 - BLOCK;
- Em códigos sequenciais usa-se:
 - IF-THEN-ELSE;
 - CASE-WHEN;
 - FOR-LOOP;
 - WHILE-LOOP;
 - WAIT (a partir do VHDL 2008).

Código concorrente x código sequencial

- ❑ Código concorrente (paralelo):
 - A ordem das declarações não importam;
 - Também chamado de dataflow code;
 - Código puramente concorrente não pode ser usado para implementar circuitos síncronos;
 - As declarações são feitas fora de PROCESS, FUNCTION e PROCEDURE;
- ❑ Código Sequencial: Todas as declarações são feitas dentro de PROCESS, FUNCTION e PROCEDURE;

Comando WHEN ELSE

- O comando WHEN-ELSE permite a transferência condicional de um sinal, de acordo com a ordem das opções apresentadas.
- A construção WHEN-ELSE define uma prioridade nas ordens das opções. É análogo a uma cadeia de seletores com prioridades, onde a primeira tem prioridade máxima e a última, prioridade mínima;

- *Exemplo com 1 bit de seleção (variável SEL):*

```
X <= A when SEL = '1' else B;  
X <= "00" when SEL = '1' else "11";
```

NOTA 1:

```
Signal Dados : bit_vector (1 downto 0);  
Dados = "10"; -- Para 2 ou mais bits, usa-se aspas.  
Dados(1) = '1';  
Dados(0) = '0'; -- Para 1 bit usa-se apóstrofo
```

Comando WHEN ELSE

NOTA 2: O Comando sempre deve usar a finalização ELSE

```
architecture COND of WRONG is
begin
    Z <= A when X > 5;  -- illegal
end COND;
```

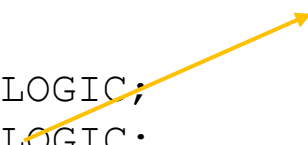
Apresentará erro de
compilação

Comando WHEN ELSE

- Exemplo com 2 bits de seleção (variável SEL1 e SEL2):

```
Sel : in  STD_LOGIC_VECTOR (1 downto 0);
```

```
Entity mux_4_1_when_else is  
Port(i0, i1, i2, i3 : IN STD_LOGIC;  
      sel1, sel2 : IN STD_LOGIC;  
      s0 : OUT STD_LOGIC);  
end mux_4_1_when_else;
```



```
Architecture teste of mux_4_1_when_else is  
begin  
s0 <= i0 WHEN sel1 = '0' AND sel2 = '0' ELSE  
      i1 WHEN sel1 = '0' AND sel2 = '1' ELSE  
      i2 WHEN sel1 = '1' AND sel2 = '0' ELSE  
      i3;  
end teste;
```

Exercício 1

- Mux 2:1 de 2 bits: Faça um software que coloque na saída X o vetor de 2 bits da entrada A ou B, dependendo do bit na entrada SEL.
- Utilizar como entrada: DIP Switch[0] e DIP Switch [1] para A e DIP Switch[2] e DIP Switch [3] para B e KEY [0] para SEL, e como saída: LED[0] e LED[1].

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity mux2to1 is
    Port ( SEL : in  STD_LOGIC;
          A   : in  STD_LOGIC_VECTOR (1 downto 0);
          B   : in  STD_LOGIC_VECTOR (1 downto 0);
          X   : out STD_LOGIC_VECTOR (1 downto 0));
end mux2to1;

architecture funcao of mux2to1 is
begin
    X <= A when (SEL = '1') else B;
end funcao;
```


Exercício 1

- Com a DE1-SoC e pin_assignments:

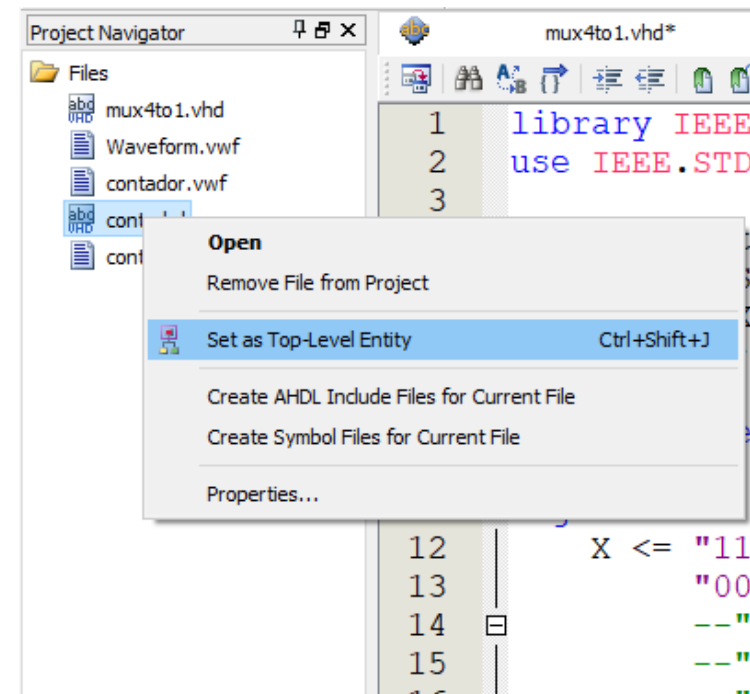
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

Entity mux2to1 is
Port(SW : IN STD_LOGIC_VECTOR (0 to 3);
      KEY : IN STD_LOGIC_VECTOR (0 to 0);
      LEDR : OUT STD_LOGIC_VECTOR (0 to 1));
end mux2to1;

Architecture aula of mux2to1 is
begin
LEDR <= SW(0 to 1) WHEN KEY(0)= '0' ELSE
        SW(2 to 3);
end aula;
```

Como utilizar mais de um código no Quartus

- Não é necessário criar um novo projeto quando se deseja testar e simular um novo código. Basta seguir os seguintes passos:
 - *Crie um novo arquivo .vhd, e nomeie ele de acordo com o nome que será utilizado na entidade e na arquitetura.*
 - *Configure esse novo arquivo como top-level.*
 - *No entanto, TODOS os .vhd serão compilados, e se algum deles tiver algum erro, dará erro de compilação.*
 - *Se utilizar os mesmos PORTs, será necessário limpar o pin planner.*



Comando WHEN ELSE

- Também pode-se representar:

```
"00110000" when SEL = "01" else
```

Por:

```
"00110000" when SEL(0) = '0' AND SEL(1)='1' else
```

Ou ainda:

```
"00110000" when SEL(1 downto 0) = "01" else
```

- Caso SEL tivesse 4 bits (3 downto 0), e deseja-se testar os bits 3, 1 e 0, tem-se:

```
"00110000" when SEL(3) = '1' AND SEL(1 downto 0) = "01" else
```

Exercício 2

- Mux 4:1 de 8 bits: Faça um software que coloque na saída X um dos vetores de 8 bits “11000000”, “00110000”, “00001100” e “00000011”, dependendo dos bits na entrada SEL (00, 01, 10, 11).
- Utilizar como entrada: DIP Switch[0] e DIP Switch [1] para SEL e como saída: LED[0-7].

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity mux4to1 is
    Port ( SEL : in  STD_LOGIC_VECTOR (1 downto 0);
          X   : out STD_LOGIC_VECTOR (7 downto 0));
end mux4to1;

architecture funcao of mux4to1 is
begin
    X <= "11000000" when SEL = "00" else
         "00110000" when SEL = "01" else
         "00001100" when SEL = "10" else
         "00000011" when SEL = "11";
end funcao;
```

Exercício 3

- Faça e simule um programa com um contador de 0 a 15 que retorne:
 - “00” quando a entrada contador for menor que 7;
 - “11” quando a entrada contador for maior que 10;
 - “ZZ” quando a entrada contador for outros valores;

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity cont is
    Port ( contador : in  INTEGER RANGE 0 to 15;
          X      : out STD_LOGIC_VECTOR (1 downto 0));
end cont;

architecture funcao of cont is
begin
    X <= "00" when contador < 7 else
         "11" when contador > 10 else
         "ZZ";
end funcao;
```

Count Value

Radix: Unsigned Decimal

Start value: 0

Increment by: 1

Count type

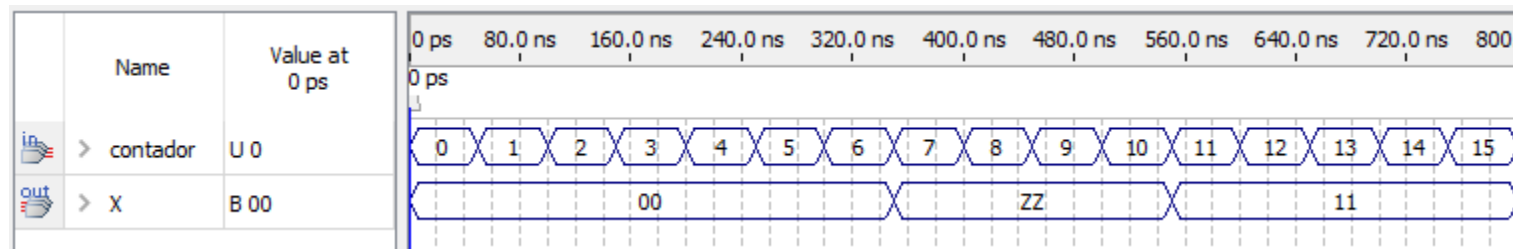
☒ Binary

☐ Gray code

Transitions occur

Count every: 50.0 ns

OK Cancel



Exercício 4

- Fazer um somador $a + b$ que também faça a comparação $a > b$.
 - *a e b variam de 0 a 7.*
 - *Saída “comp” = ‘1’ quando $a > b$.*

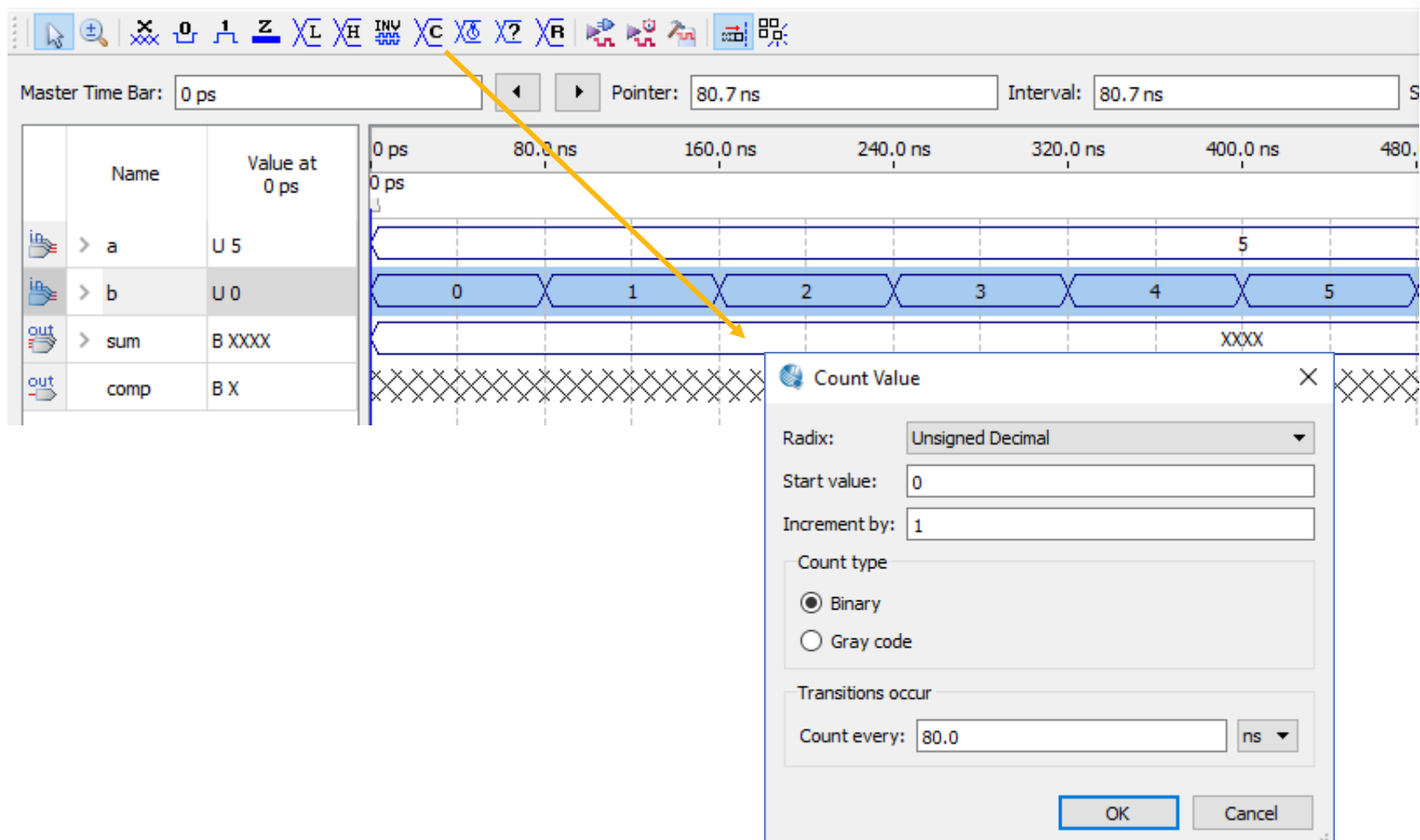
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity comp_add is
    Port ( a, b: IN INTEGER RANGE 0 to 7;
          comp: OUT STD_LOGIC;
          sum: OUT INTEGER RANGE 0 TO 15);
end comp_add;

architecture funcao of comp_add is
begin
    comp <= '1' WHEN a > b ELSE '0';
    sum <= a + b;
end funcao;
```

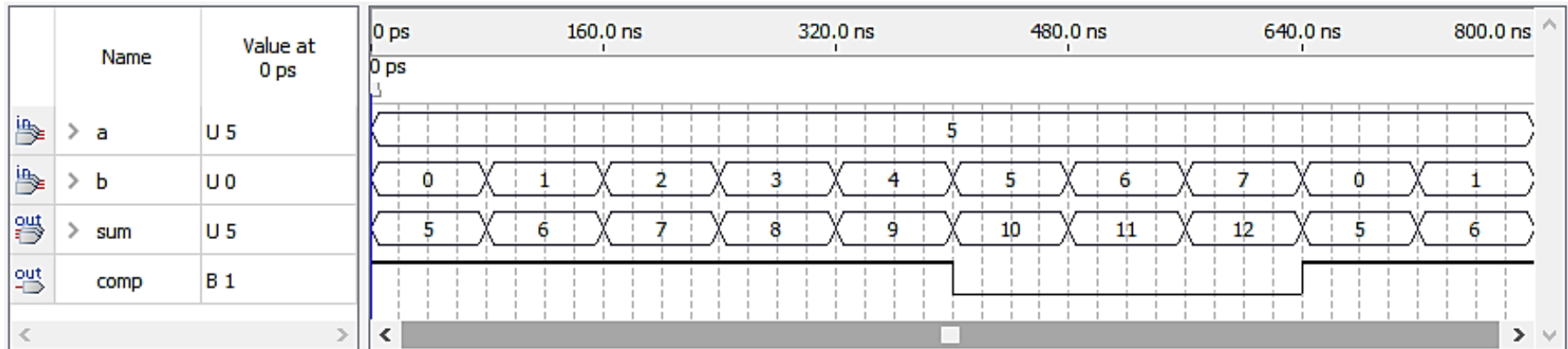
CIRCUITO SOMADOR COM COMPARAÇÃO

- Parâmetros simulação: End time: 800ns Grid: 40ns.

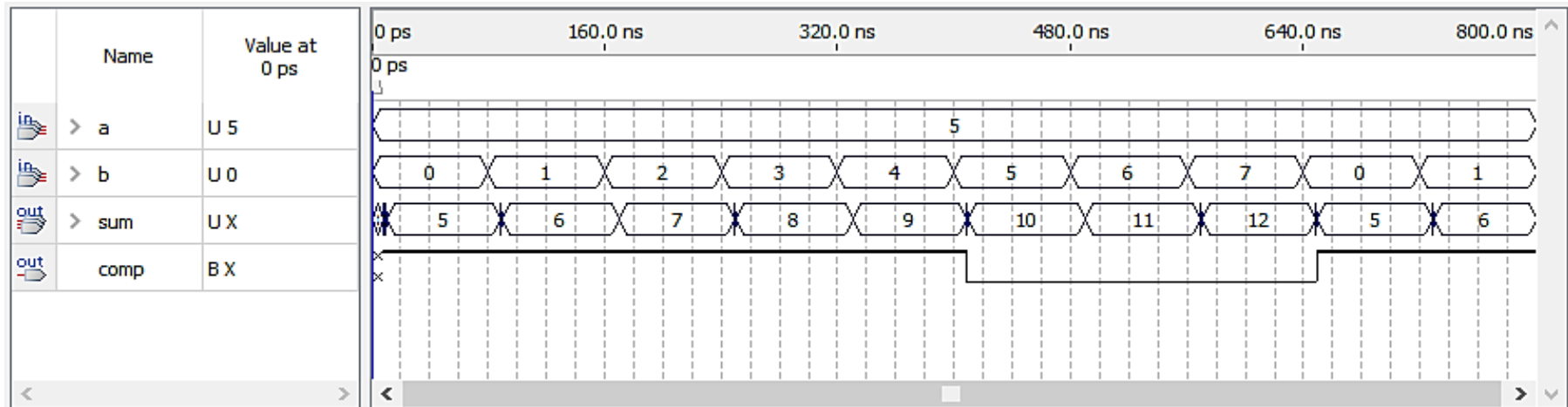


CIRCUITO SOMADOR COM COMPARAÇÃO

- Simulação funcional



- Simulação no tempo



- Códigos concorrentes – WITH/SELECT/WHEN.

- PEDRONI, Volnei A. Eletrônica Digital Moderna e VHDL. 1. ed. Campus. 2010, 648 p. ISBN 8535234659