

PCS2304

Projeto Lógico Digital

Aula 10 –

Projeto de Circuitos com VHDL

Edson T. Midorikawa

Recapitulação

- Circuitos biestáveis (*flip-flops e latches*)
 - Parâmetros de tempo
 - Aplicações de circuitos biestáveis

Objetivos

- Projeto de circuitos com VHDL
 - Introdução à linguagem VHDL
 - Exemplos de circuitos com VHDL

Aula 10 - Projeto de Circuitos com VHDL

3

Introdução à VHDL

- baseado no material do prof. Rodolfo J. de Azevedo (IC-Unicamp):
 - *VHDL - introdução*
 - *VHDL Coding Style*
- Material disponível em
<http://www.ic.unicamp.br/~rodolfo/Cursos/mo801/1s2006/>

Aula 10 - Projeto de Circuitos com VHDL

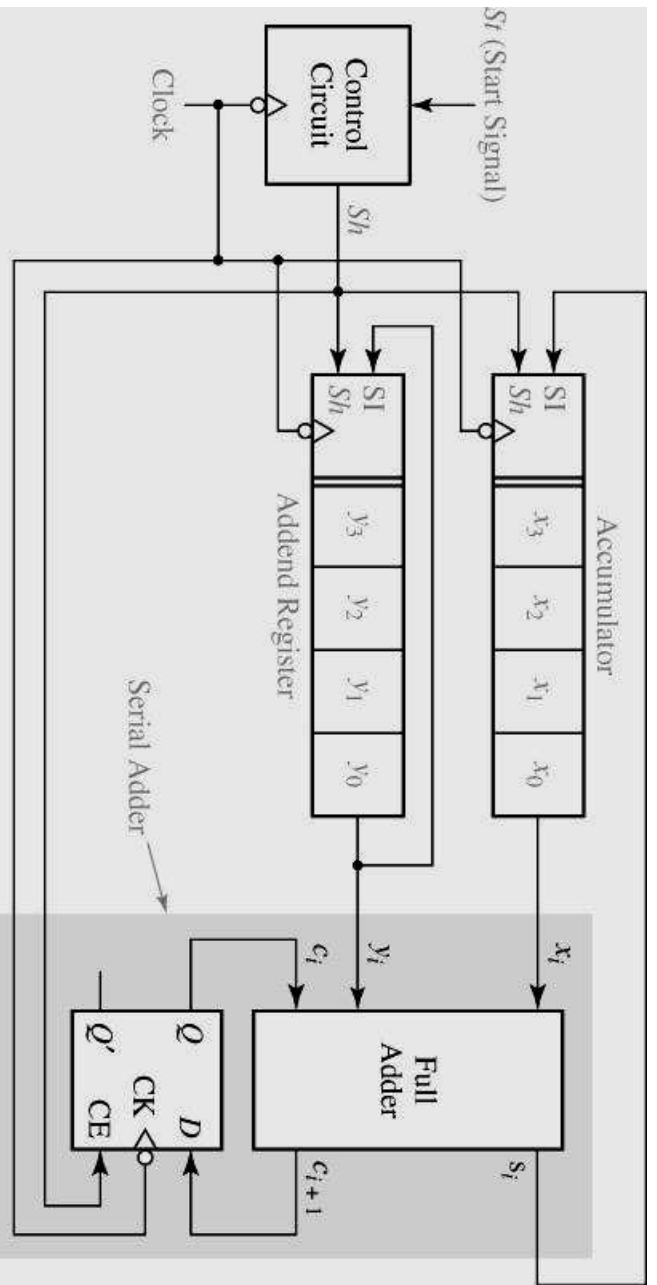
4

Exemplos de circuitos com VHDL

- Baseado no material do livro “*Fundamentals of Logic Design*”, 5th edition, Charles H. Roth Jr.

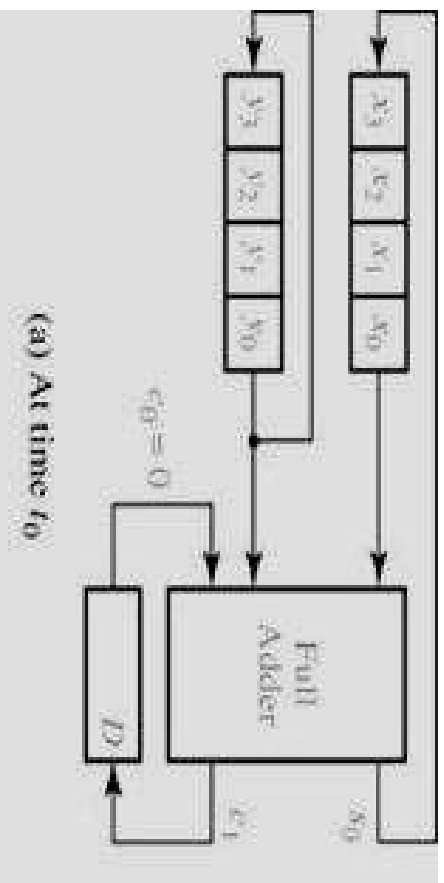


Somador Serial

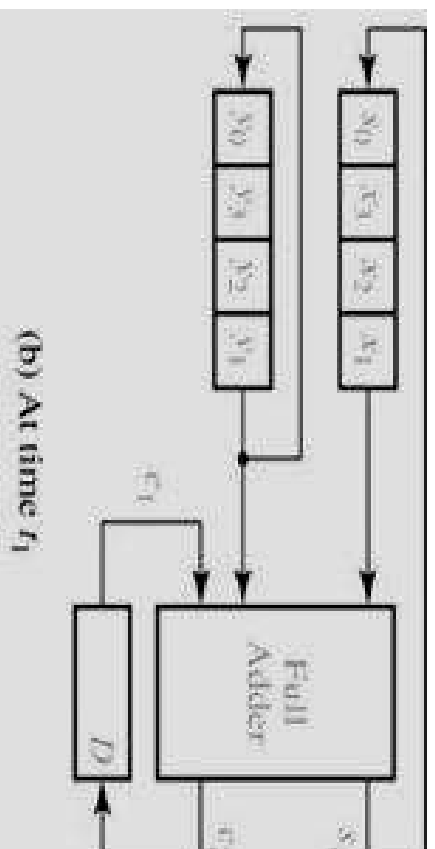


Somador Serial (2)

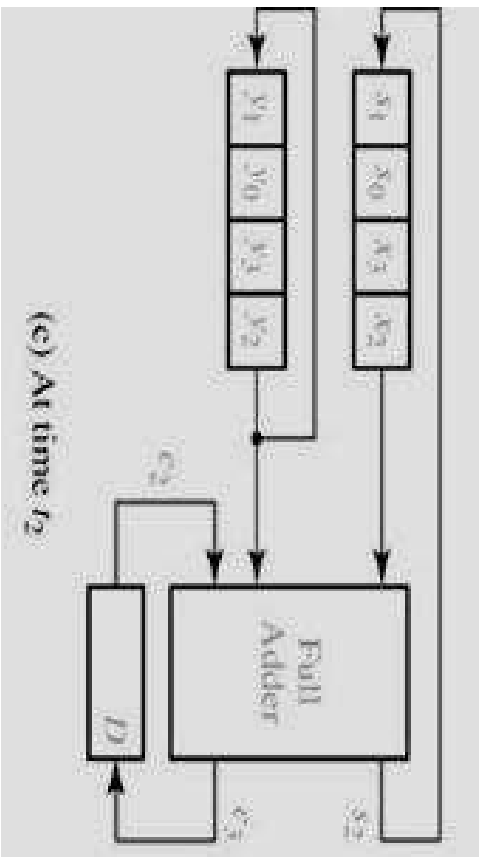
- Exemplo: $x_3x_2x_1x_0 + y_3y_2y_1y_0$



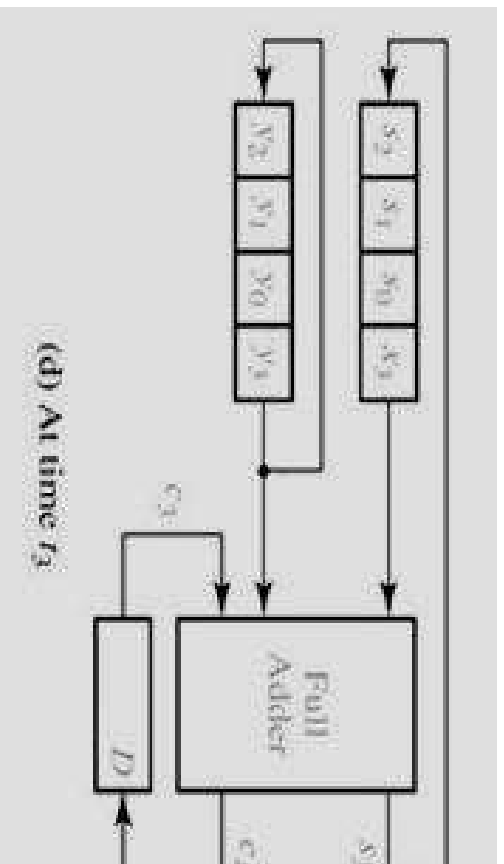
Somador Serial (3)



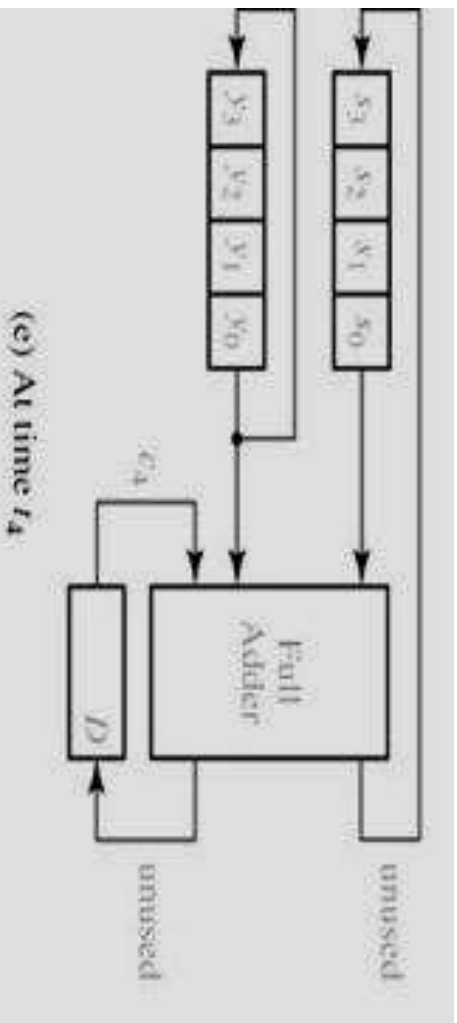
Somador Serial (4)



Somador Serial (5)

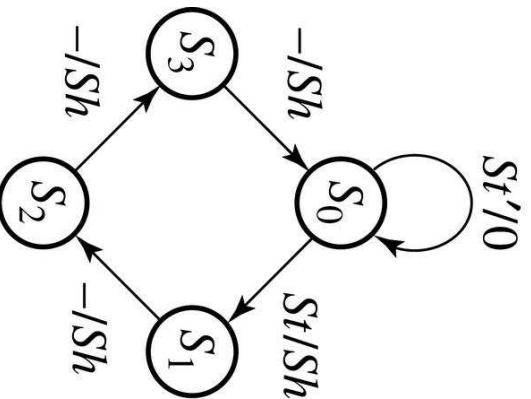


Somador Serial (6)



Somador Serial (7)

- Controle do somador serial

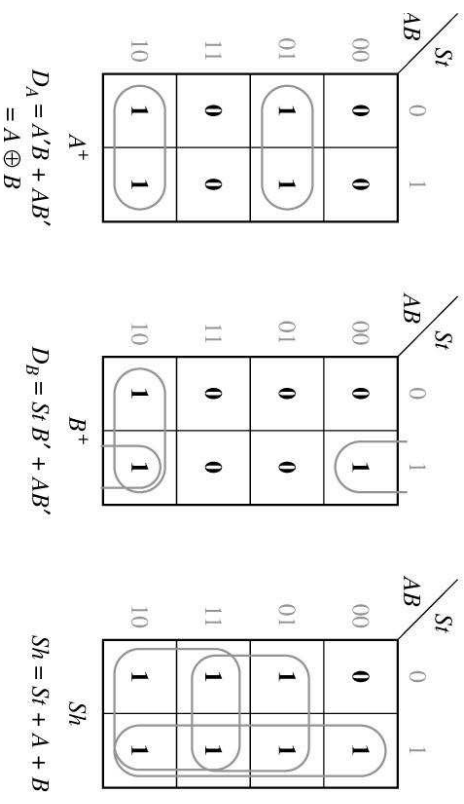


	Next State	Sh		
	St = 0	1	0	1
S_0	S_0	S_1	0	1
S_1	S_1	S_2	1	1
S_2	S_2	S_3	1	1
S_3	S_3	S_0	1	1

Somador Serial (8)

- Equações do circuito de controle

	AB	A+B+	
		0	1
S ₀	00	00	01
S ₁	01	10	10
S ₂	10	11	11
S ₃	11	00	00



Aula 10 - Projeto de Circuitos com VHDL

13

```

1  library IEEE;
2  use IEEE.STD_LOGIC_1164.all;
3
4  entity serial is
5      Port (St: in std_logic;
6           Clk: in std_logic;
7           Xout: out std_logic_vector(3 downto 0));
8  end serial;
9
10 architecture Behavioral of serial is
11     signal X, Y: std_logic_vector(3 downto 0);
12     signal Sh: std_logic;
13     signal Ci, Ciplus: std_logic;
14     signal Sumi: std_logic;
15     signal State, NextState: integer range 0 to 3;
16
17     -- full adder
18     Sumi <= X(0) xor Y(0) xor Ci;
19     Ciplus <= (Ci and X(0)) or (Ci and Y(0)) or (X(0) and Y(0));
20     Xout <= X;
21
22     process (State, St)
23     begin
24         case State is
25             when 0 =>
26                 if St = '1' then Sh <= '1'; NextState <= 1;

```

Aula 10 - Projeto de Circuitos com VHDL

14

```

23     else Sh <= '0', NextState <= 0; end if;
24     when 1 => Sh <= '1', NextState <= 2;
25     when 2 => Sh <= '1', NextState <= 3;
26     when 3 => Sh <= '1', NextState <= 0;
27     end case;
28     end process;
29
30     process (clk)
31     begin
32         if clk'event and clk = '0' then
33             State <= NextState;
34             if Sh = '1' then
35                 X <= Sumi & X(3 downto 1);
36                 Y <= Y (0) & Y(3 downto 1);
37                 Ci <= Ciplus; end if;
38             end if;
39         end process;
40     end Behavioral;
41
42     -- update state register
43     -- shift Sumi into X register
44     -- rotate right Y register
45     -- store next carry

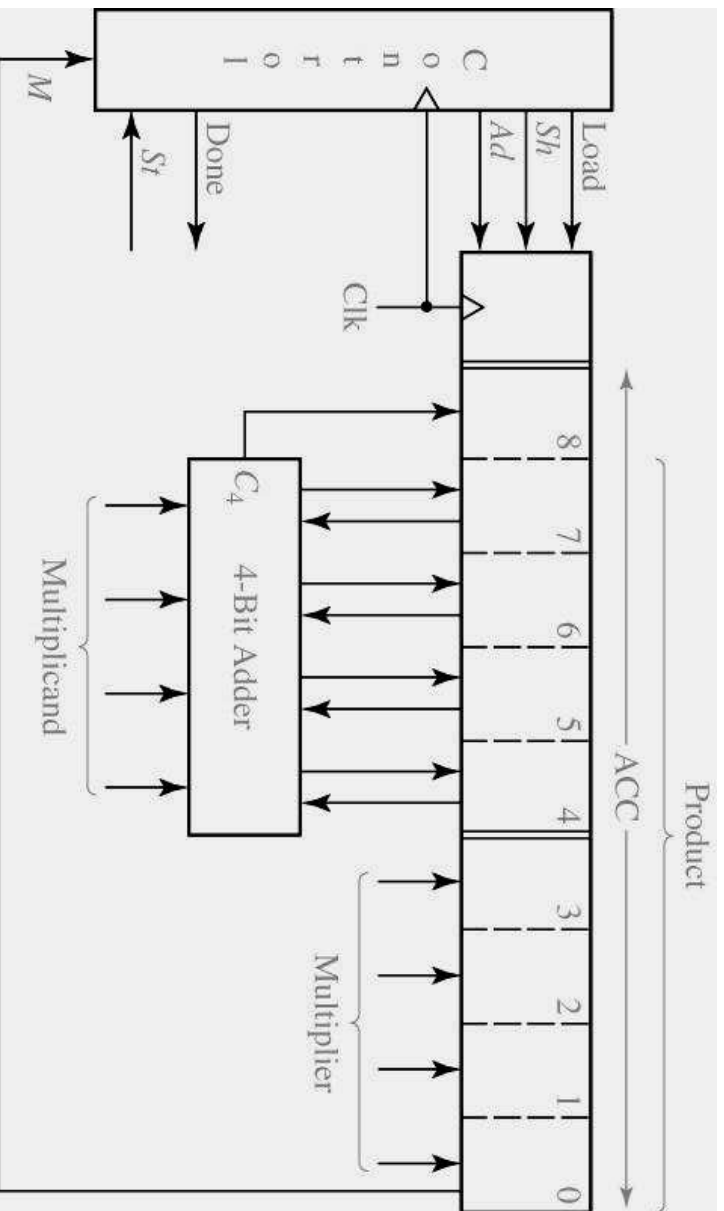
```

(2) Serial somador

Multiplicador Binário

conteúdo inicial do registrador produto	0 0 0 0 0 1 0 1 1	← M	(11=1011)
(soma multiplicando porque M=1)	1 1 0 1		(13=1101)
depois da soma	0 1 1 0 1 1 0 1 1		
depois do deslocamento	0 0 1 1 0 1 1 0 1	← M	
(soma multiplicando porque M=1)	1 1 0 1		
depois da soma	1 0 0 1 1 1 1 0 1		
depois do deslocamento	0 1 0 0 1 1 1 1 0	← M	
(pula soma porque M=0)			
depois do deslocamento	0 0 1 0 0 1 1 1 1	← M	
(soma multiplicando porque M=1)	1 1 0 1		
depois da soma	1 0 0 0 1 1 1 1 1		
depois do deslocamento (resposta final)	0 1 0 0 0 1 1 1 1		(143=10001111)

Multiplicador Binário



Aula 10 - Projeto de Circuitos com VHDL

17

Multiplicador Binário

```
1 library IEEE;
2 use IEEE.STD_LOGIC_1164. all;
3 use IEEE.STD_LOGIC_ARITH. all;
4 use IEEE.STD_LOGIC_UNSIGNED. all;
5 entity mult8X8 is
6     Port (CLK, St: in std_logic;
7           Mplier, Mcand: in std_logic_vector(7 downto 0));
8     Done: out std_logic;
9     Product: out std_logic_vector(15 downto 0));
10 end mult8X8;
11 architecture Behavioral of mult8X8 is
12     signal State, NextState: Integer range 0 to 3;
13     signal count: std_logic_vector(2 downto 0) := "000"; -- 3-bit counter
14     signal A: std_logic_vector(8 downto 0); -- accumulator
15     signal B: std_logic_vector(7 downto 0);
16     alias M: std_logic is B(0); -- M is bit 0 of B
17     signal addout: std_logic_vector(8 downto 0);
18     signal K, Load, Ad, Sh: std_logic;
```

Aula 10 - Projeto de Circuitos com VHDL

18

```

19 begin
20     Product <= A(7 downto 0) & B;           -- 16-bit product is in A and B
21     addout <= '0' & A(7 downto 0) + Mcand; -- adder output is 9 bits including carry
22     K <= '1' when count = 7 else '0';
23     process (St, State, K, M)
24     begin
25         Load <= '0'; Sh <= '0'; Ad <= '0'; Done <= '0';
                -- control signals are '0' by default
26         case State is
27             when 0 =>
28                 if St = '1' then Load <= '1'; NextState <= 1;
29                 else NextState <= 0; end if;
30             when 1 =>
31                 if M = '1' then Ad <= '1'; NextState <= 2;
32                 else if K = '0' then Sh <= '1'; NextState <= 1;
33                 else Sh <= '1'; NextState <= 3; end if;

```

Aula 10 - Projeto de Circuitos com VHDL

19

```

34     end if;
35     when 2 =>
36         if K = '0' then Sh <= '1'; NextState <= 1;
37         else Sh <= '1'; NextState <= 3; end if;
38     when 3 =>
39         Done <= '1'; NextState <= 0;
40     end case;
41     end process;
42     process (clk)
43     begin
44         if clk'event and clk = '1' then           -- update registers on rising edge of clk
45             if load = '1' then
46                 A <= "000000000"; Count <= "000"; -- clear A and counter
47                 B <= Mplier;
48             end if;
49             if Ad = '1' then A <= addout; end if;
50             if Sh = '1' then
51                 A <= '0' & A(8 downto 1); B <= A(0) & B(7 downto 1);
                    -- right shift A and B
52                 count <= count + 1;
                    -- increment counter
53                 -- uses "+" operator from ieee_std_logic_unsigned package
54             end if;
55             State <= NextState;
56         end if;
57     end process;
58     end Behavioral;

```

Aula 10 - Projeto de Circuitos com VHDL

20

Exercício para próxima aula

- Estudar a descrição do multiplicador binário
- Pesquisar uma descrição VHDL de um divisor binário
 - trazer para a aula para discussão

Aula 10 - Projeto de Circuitos com VHDL

21

Próxima Aula

- Metodologia de Projetos
 - Circuito = FD + UC
 - VHDL (continuação)
- Exercícios

Aula 10 - Projeto de Circuitos com VHDL

22

Extra

Somador completo

- $S_i = A_i \oplus B_i \oplus C_i$
- $C_{i+1} = A_i B_i + A_i C_i + B_i C_i$

ou

$$C_{i+1} = A_i B_i + (A_i \oplus B_i) C_i$$

