Universidade Federal de Santa Catarina

Centro Tecnológico – CTC

Departamento de Engenharia Elétrica

“EEL7020 – Sistemas Digitais”

Prof. Eduardo Augusto Bezerra

Eduardo.Bezerra@eel.ufsc.br

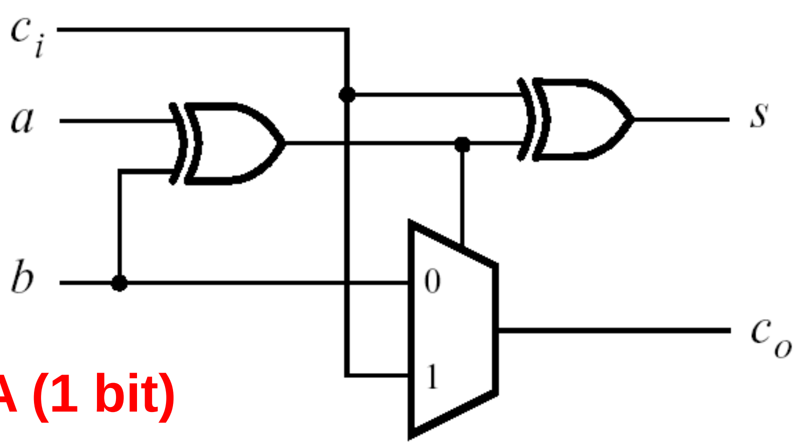
Florianópolis, março de 2010.

Sistemas Digitais

Projeto de somador de 8 bits
"construindo uma calculadora"

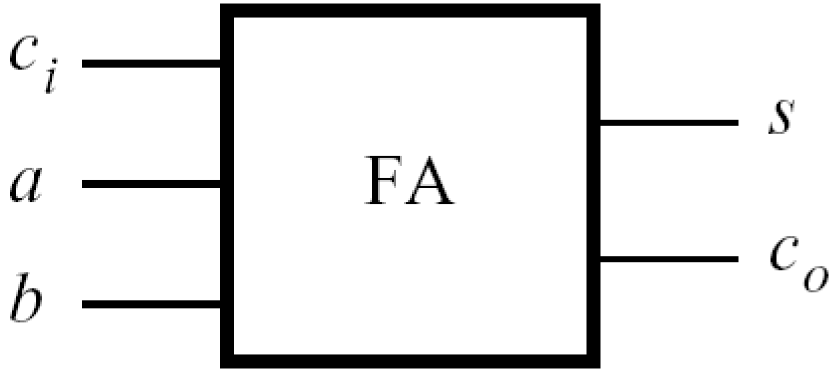
Arquivo: lab2_VHDL.pdf
parte III

Projeto de somador de 4 bits



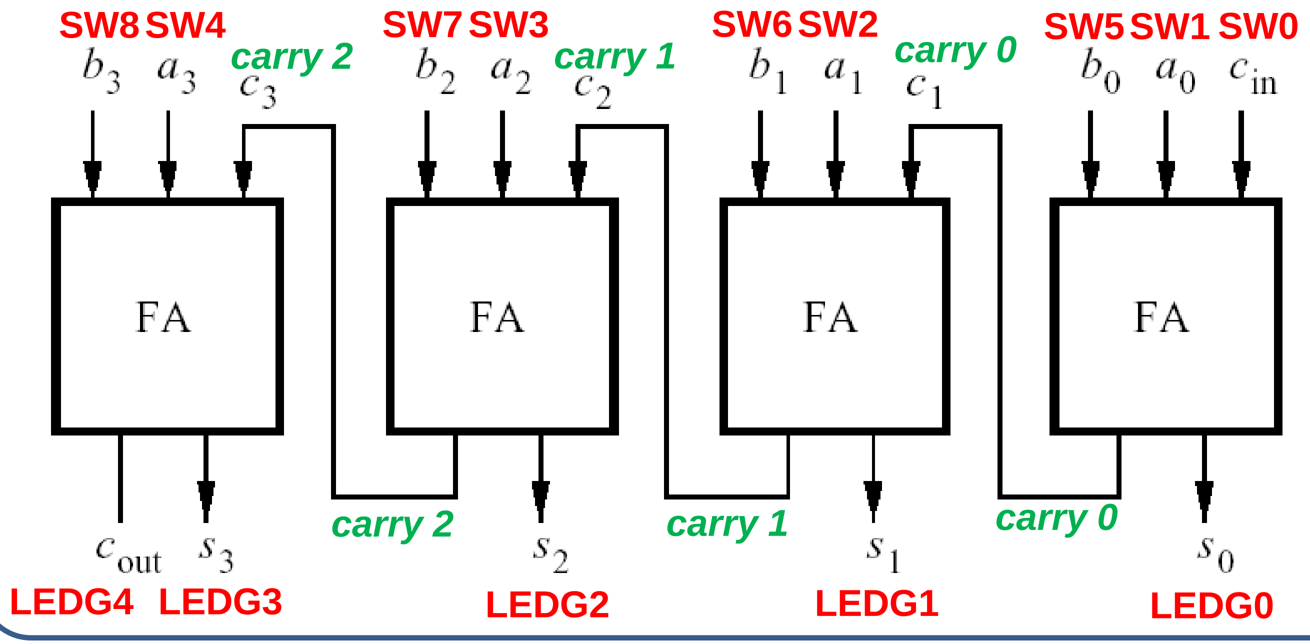
FA (1 bit)

FA (1 bit)



b	a	c_i	c_o	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Ripple-carry adder de 4 bits, construído com 4 FAs



Tarefas

A partir da Solução II apresentada anteriormente, implementar um somador de 8 bits com as seguintes entradas e saídas:

Entrada Cin e A	*Entrada B*	*Saída Soma*	*Saída Flags*
SW(0) = Cin			
SW(1) = A0	SW(9) = B0	LEDG(0) = S0	LEDR(0) = carry out
SW(2) = A1	SW(10) = B1	LEDG(1) = S1	LEDR(1) = overflow
SW(3) = A2	SW(11) = B2	LEDG(2) = S2	LEDR(2) = negativo
SW(4) = A3	SW(12) = B3	LEDG(3) = S3	LEDR(3) = zero
SW(5) = A4	SW(13) = B4	LEDG(4) = S4	
SW(6) = A5	SW(14) = B5	LEDG(5) = S5	
SW(7) = A6	SW(15) = B6	LEDG(6) = S6	
SW(8) = A7	SW(16) = B7	LEDG(7) = S7	

$$\begin{array}{cccccccccccl}
 & C_8 & C_7 & C_6 & C_5 & C_4 & C_3 & C_2 & C_1 & & \longleftarrow & \text{“vai-um” } (\textit{carry}) \\
 & & A_7 & A_6 & A_5 & A_4 & A_3 & A_2 & A_1 & A_0 & \longleftarrow & 1^\circ \text{ operando} \\
+ & & B_7 & B_6 & B_5 & B_4 & B_3 & B_2 & B_1 & B_0 & \longleftarrow & 2^\circ \text{ operando} \\
\hline
 & & S_7 & S_6 & S_5 & S_4 & S_3 & S_2 & S_1 & S_0 & \longleftarrow & \text{soma}
\end{array}$$

```
library  ieee;
use  ieee.std_logic_1164.all;
-- somador completo (FA)
entity FA is
   port (a, b, c: in std_logic;
          soma, carry: out std_logic);
   end FA;
architecture FA_beh of FA is
begin
   soma <= (a xor b) xor c;
   carry <= b when ((a xor b) = '0')
                  else c;
end FA_beh;
```

SW(1) = A0	SW(9) = B0
SW(2) = A1	SW(10) = B1
SW(3) = A2	SW(11) = B2
SW(4) = A3	SW(12) = B3
SW(5) = A4	SW(13) = B4
SW(6) = A5	SW(14) = B5
SW(7) = A6	SW(15) = B6
SW(8) = A7	SW(16) = B7

SW(0) = Cin

LEDG(0) = S0
LEDG(1) = S1
LEDG(2) = S2
LEDG(3) = S3
LEDG(4) = S4
LEDG(5) = S5
LEDG(6) = S6
LEDG(7) = S7

LEDR(0) = carry out
LEDR(1) = overflow
LEDR(2) = negativo
LEDR(3) = zero

```
library  ieee;
use  ieee.std_logic_1164.all;
entity RCA is
   port (SW : IN STD_LOGIC_VECTOR(17 downto 0);
          LEDG : OUT STD_LOGIC_VECTOR(7 downto 0) ;
          LEDR : OUT STD_LOGIC_VECTOR(17 downto 0)
        );
end RCA;
architecture RCA_stru of RCA is
   signal c: std_logic_vector (7 downto 0);
   signal s: std_logic_vector (7 downto 0);
component FA
   port (a, b, c: in std_logic;
          soma, carry: out std_logic);
   end component;
begin
FA0:  FA port map (sw(1), sw(9), sw(0), s(0), c(0));
FA1:  FA port map (sw(2), sw(10), c(0), s(1), c(1));
FA2:  FA port map (sw(3), sw(11), c(1), s(2), c(2));
FA3:  FA port map (sw(4), sw(12), c(2), s(3), c(3));
FA4:  FA port map (sw(5), sw(13), c(3), s(4), c(4));
FA5:  FA port map (sw(6), sw(14), c(4), s(5), c(5));
FA6:  FA port map (sw(7), sw(15), c(5), s(6), c(6));
FA7:  FA port map (sw(8), sw(16), c(6), s(7), c(7));
LEDG <= s;
LEDR(0) <= c(7);                       -- carry out
LEDR(1) <= c(6) xor c(7);              -- overflow
LEDR(2) <= s(7);                       -- negativo
LEDR(3) <= ‘1’ when (s = “00000000”) else ‘0’;     -- zero
end RCA_stru;
```