Instituto Politécnico Nacional

Escuela Superior de Computo

Práctica 4.

Sumador/Restador de 4 bits con acarreo en cascada.

Nombre: Flores Castro Luis Antonio.

Arquitectura de Computadoras.

Profesora: Vega García Nayeli.

Código VHDL del Sumador / Restador.

```
1. library IEEE;
2. use IEEE.STD_LOGIC_1164.ALL;
3.
4. entity SumadorRes is
5.     Port ( a,b : in STD_LOGIC_VECTOR (3 downto 0);
6.            sel : in STD_LOGIC;
7.            sal : out STD_LOGIC_VECTOR (3 downto 0);
8.            cot : out STD_LOGIC);
9. end SumadorRes;
10.
11.     architecture Behavioral of SumadorRes is
12.
13.     signal EB:std_logic_vector(3 downto 0);
14.     signal c:std_logic_vector(4 downto 0);
15.
16.     begin
17.
18.     c(0)<=sel;
19.
20.         bucle : for i in 0 to 3 generate
21.             EB(i)<=b(i) xor c(0);
22.             sal(i)<=a(i) xor EB(i) xor c(i);
23.             c(i+1)<=(a(i) and EB(i)) or (a(i) and c(i)) or (
   EB(i) and c(i));
24.         end generate;
25.         cot<=c(4);
26.
27.     end Behavioral;
```

Código VHDL Test-Bench del Simulador / Restador.

```
1. library IEEE;
2. use IEEE.STD_LOGIC_1164.ALL;
3.
4. entity SumadorTB is
5. --  Port ( );
6. end SumadorTB;
7.
8. architecture Behavioral of SumadorTB is
9.
10.     component SumadorRes is
11.         Port ( a,b : in STD_LOGIC_VECTOR (3 downto 0);
12.                sel : in STD_LOGIC;
13.                sal : out STD_LOGIC_VECTOR (3 downto 0);
14.                cot : out STD_LOGIC);
15.     end component;
16.
17.
18.                signal a,b : STD_LOGIC_VECTOR (3 downto 0);
19.                signal sel : STD_LOGIC;
20.                signal sal : STD_LOGIC_VECTOR (3 downto 0);
21.                signal cot : STD_LOGIC;
22.
23.     begin
24.     mapeo: SumadorRes
25.         Port map( a=>a,
26.                   b=>b,
27.                sel=>sel,
28.                sal=>sal,
29.                cot=>cot);
30.
31.     process
32.     begin
33.         a<="0110";
34.         b<="0111";
35.         sel<='0';
36.         wait for 20 ns;
37.         a<="0110";
38.         b<="1001";
39.         sel<='0';
40.         wait for 20 ns;
41.         a<="0100";
42.         b<="1001";
43.         sel<='0';
44.         wait for 20 ns;
45.         a<="1111";
46.         b<="0001";
47.         sel<='1';
48.         wait for 20 ns;
```

```
49.         a<="0011";
50.         b<="1010";
51.         sel<='0';
52.         wait for 20 ns;
53.         a<="1100";
54.         b<="0101";
55.         sel<='1';
56.         wait for 20 ns;
57.         a<="1110";
58.         b<="1000";
59.         sel<='1';
60.         wait for 20 ns;
61.         a<="1010";
62.         b<="0110";
63.         sel<='1';
64.         wait for 20 ns;
65.         a<="1001";
66.         b<="0100";
67.         sel<='1';
68.         wait;
69.     end process;
70.
71.     end Behavioral;
```

Diagrama RTL.

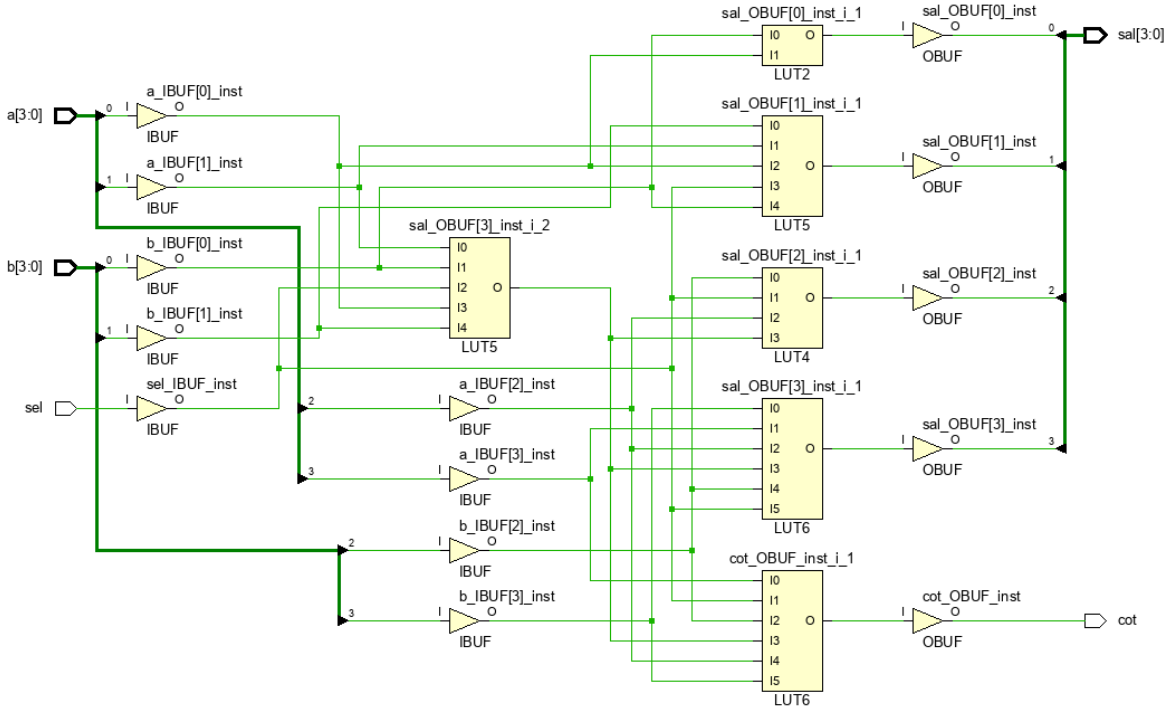

Figura 1. Circuito esquemático del Sumador / Restador.

Diagrama RTL.

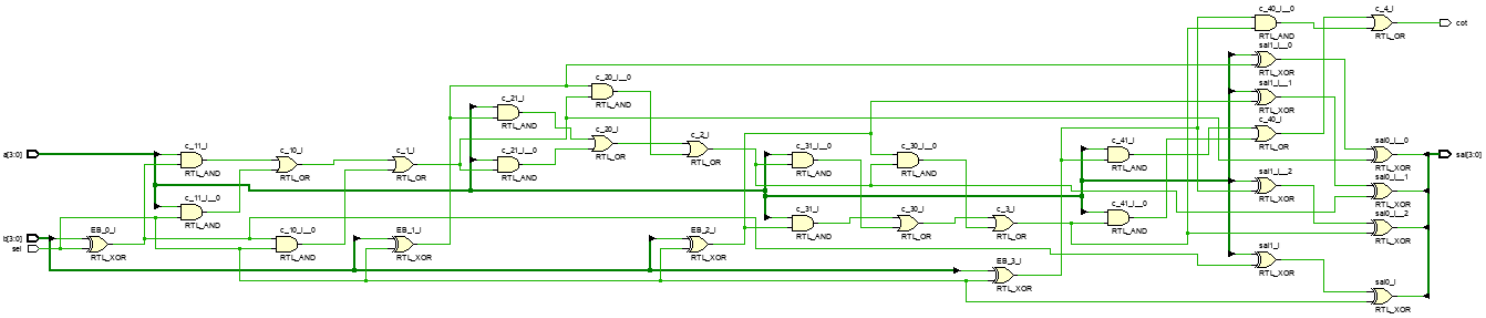

Figura 1.1. Circuito lógico del Sumador / Restador.

Tabla de Resultados.

Operación	A	B	S	Cout
Suma	6	7	0	d
Suma	6	9	0	f
Suma	4	9	0	d
Resta	15	1	1	e
Suma	3	10	0	d
Resta	12	5	1	7
Resta	14	8	1	6
Resta	10	6	1	4
Resta	9	4	1	5

Tabla 1. Resultados de la simulación del Sumador / Restador.

Diagrama de Onda del Sumador / Restador.

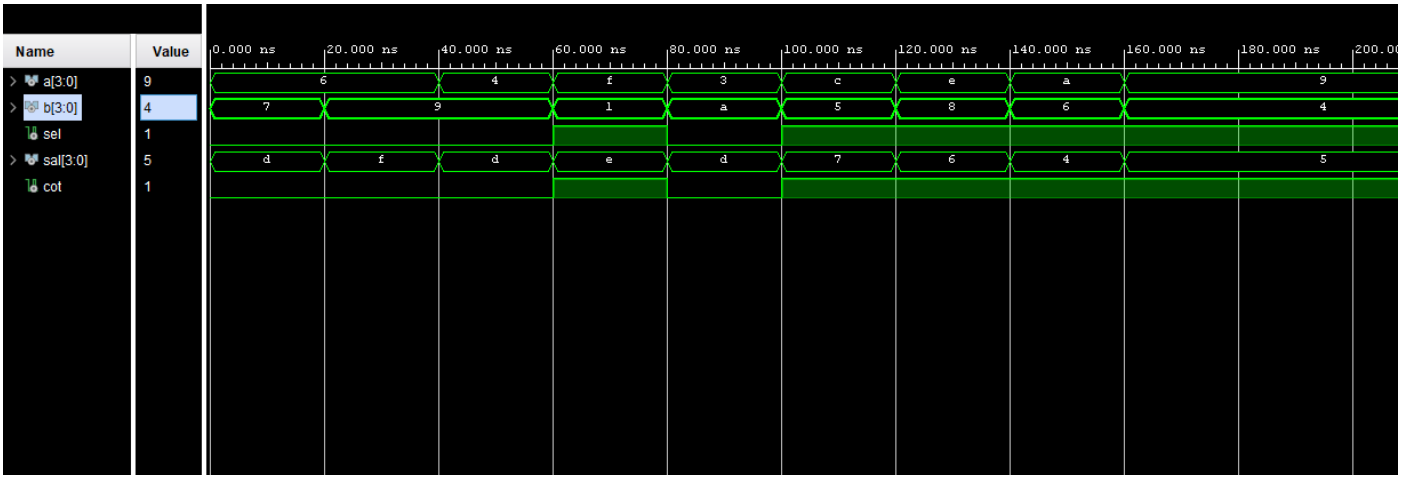

Figura 2. Simulación del Test-Bench del Sumador / Restador.