Reporte de practica 3

González Pardo Adrian

Febrero 2020

1. Código VHDL

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

entity practica is
    Port ( a,b : in STD_LOGIC_VECTOR (7 downto 0);
           cin : in STD_LOGIC;
           cout : out STD_LOGIC;
           s : out STD_LOGIC_VECTOR (7 downto 0));
end practica;

architecture Behavioral of practica is
begin
    process(a,b,cin)
    variable P,g:std_logic_vector(7 downto 0);
    variable c:std_logic_vector(8 downto 0);
    variable auxc,auxa,auxb,auxd:std_logic;
    begin
        c(0):=cin;
        for i in 0 to 7 loop
            P(i):= a(i) xor b(i);
            g(i):= a(i) and b(i);
            s(i) <= P(i)xor c(i);

            auxc:='1';

            for j in 0 to i loop
                auxc:= auxc and P(j);
            end loop;
            auxa:= auxc and C(0);

            auxd:='0';
            for k in 0 to i-1 loop
                auxb:= '1';
                for m in k+1 to i loop
                    auxb:= auxb and p(m);
                end loop;
                auxd:= auxd or (auxb and g(k));
```

```
            end loop;
            c(i+1):= g(i) or auxa or auxd;
        end loop;
        cout<=c(8);
    end process;
end Behavioral;
```

Código fuente de sumador con acarreo anticipado

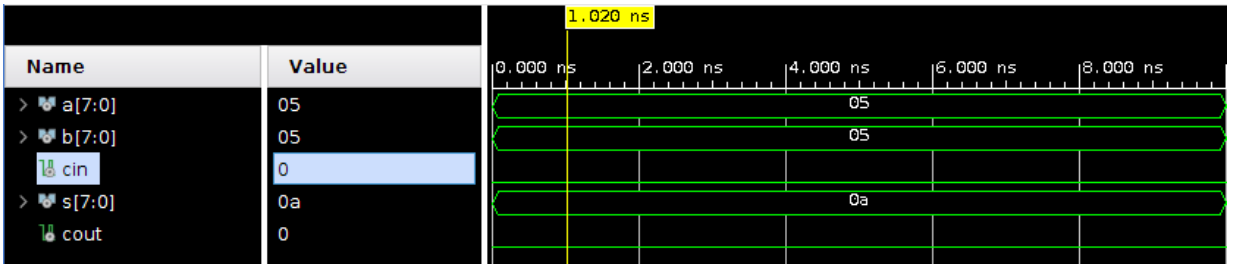
2. Test-Bench VHDL Código

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

entity simPrac2 is
--  Port ( );
end simPrac2;

architecture Behavioral of simPrac2 is
component practica is
    Port(a,b : in STD_LOGIC_VECTOR (7 downto 0);
           cin : in STD_LOGIC;
           cout : out STD_LOGIC;
           s : out STD_LOGIC_VECTOR (7 downto 0));
end component;
signal a:STD_LOGIC_VECTOR (7 downto 0) := x"00";
signal b:STD_LOGIC_VECTOR (7 downto 0) := x"00";
signal cin: STD_LOGIC :='0';
signal s: STD_LOGIC_VECTOR (7 downto 0);
signal cout: STD_LOGIC;
begin
sumaAnt: practica
    Port map (
        a=>a,
        b=>b,
        cin=>cin,
        s=>s,
        cout=>cout
    );

p2: process
begin
 cin<='0';
 a<=x"05";
 b<=x"05";
 wait for 10 ns;
 a<=x"0C";
 b<=x"07";
 wait for 10 ns;
 a<=x"09";
 b<=x"05";
 wait for 10 ns;
```

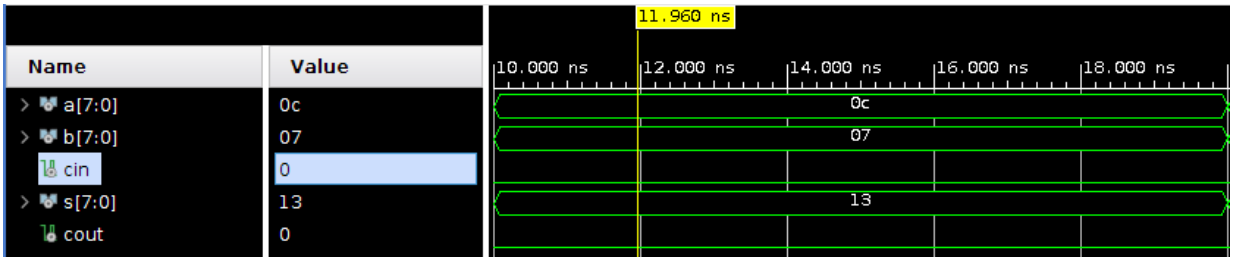
```
 a<=x"0E";
 b<=x"09";
 wait for 10 ns;
 a<=x"04";
 b<=x"02";
 wait for 10 ns;
 a<=x"07";
 b<=x"07";
 wait for 10 ns;
 a<=x"0F";
 b<=x"05";
 wait for 10 ns;
 a<=x"0B";
 b<=x"08";
 wait for 10 ns;
 a<=x"01";
 b<=x"04";
 wait;
end process;

end Behavioral;
```

Código de simulación

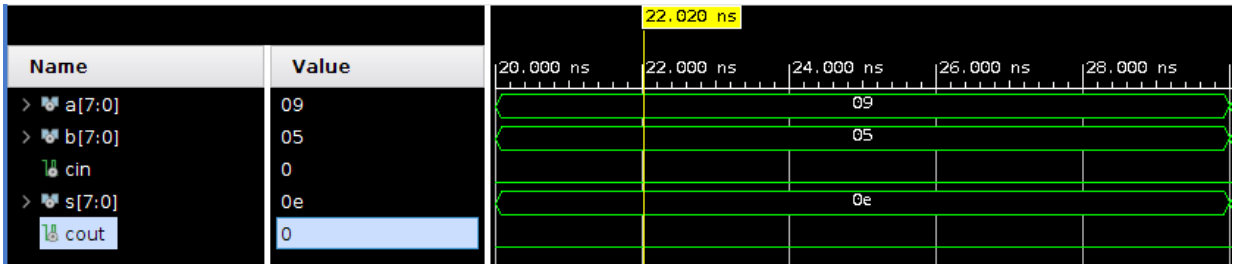
Anexo de fotos de la simulación a impulsos



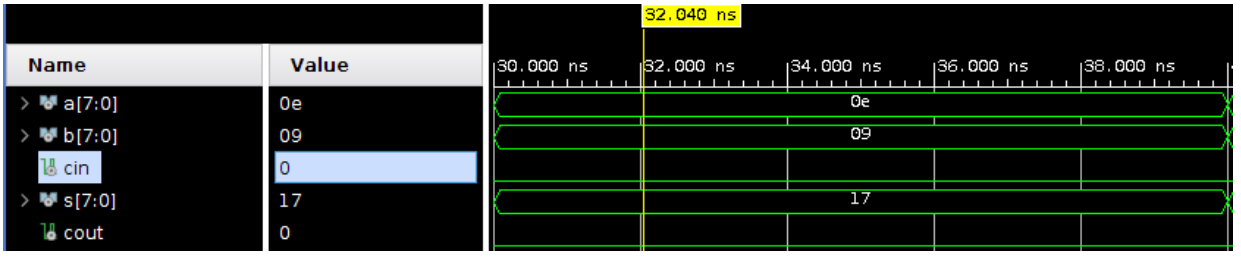
Primer parte con valores hexadecimales equivales a valores decimales: $a = 5_{10}$ & $b = 5_{10}$ *con salida* $s = A_{16} = 10_{10}$



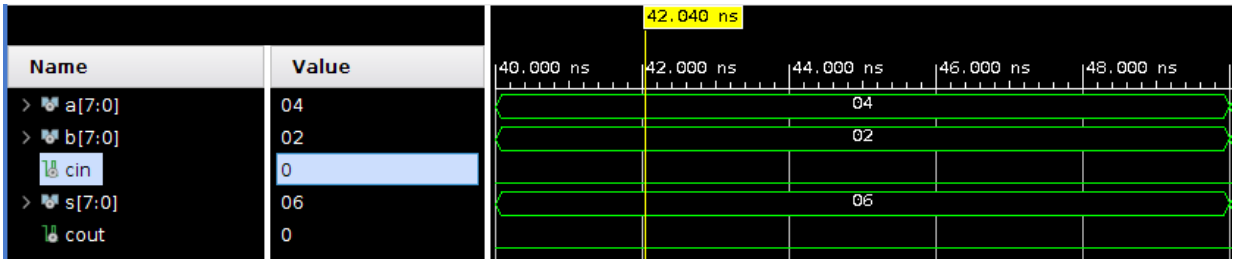
Segunda parte con valores hexadecimales equivales a valores decimales: $a = C_{16} = 12_{10}$ & $b = 7_{10}$ *con salida* $s = 13_{16} = 19_{10}$



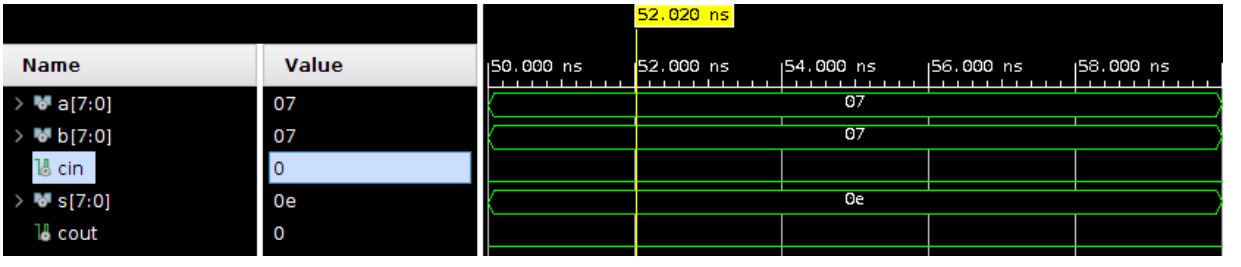
Tercer parte con valores hexadecimales equivales a valores decimales: $a = 9_{10}$ *&* $b = 5_{10}$ *con salida* $s = E_{16} = 14_{10}$



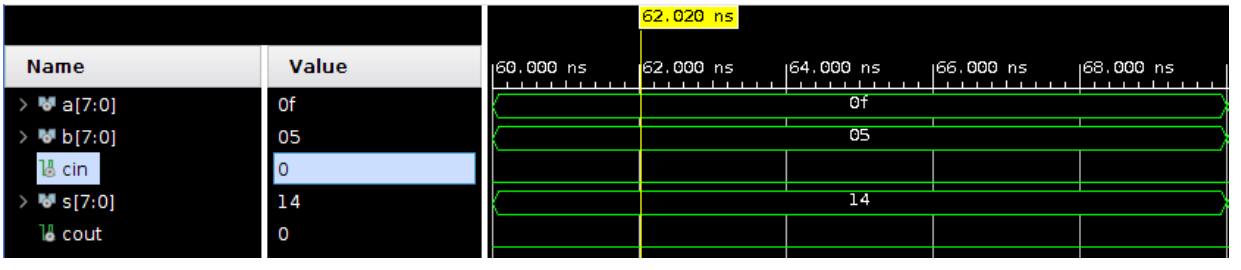
Cuarta parte con valores hexadecimales equivales a valores decimales: $a = E_{16} = 14_{10}$ *&* $b = 9_{10}$ *con salida* $s = 17_{16} = 23_{10}$



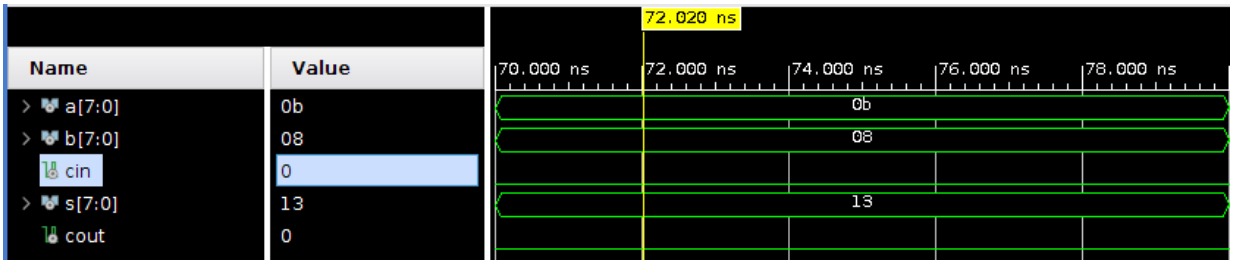
Quinta parte con valores hexadecimales equivales a valores decimales: $a = 4_{10}$ *&* $b = 2_{10}$ *con salida* $s = 6_{10}$



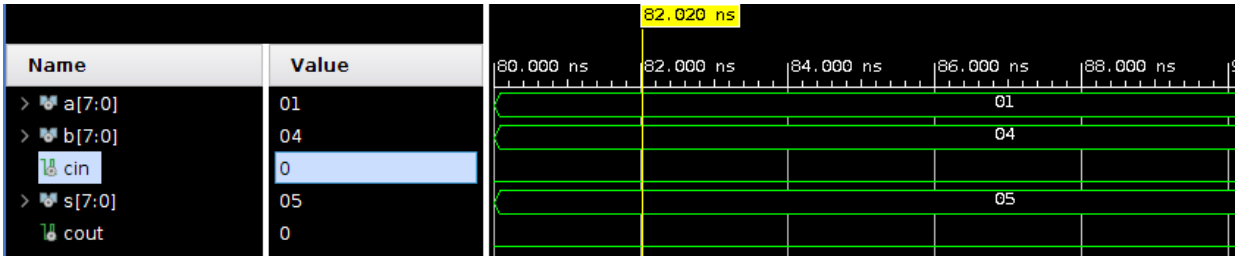
Sexta parte con valores hexadecimales equivales a valores decimales: $a = 7_{10}$ *&* $b = 7_{10}$ *con salida* $s = E_{16} = 14_{10}$



Septima parte con valores hexadecimales equivales a valores decimales: $a = F_{16} = 15_{10}$ & $b = 5_{10}$ *con salida* $s = 14_{16} = 20_{10}$



Octava parte con valores hexadecimales equivales a valores decimales: $a = B_{16} = 11_{10}$ & $b = 8_{10}$ *con salida* $s = 13_{16} = 19_{10}$



Novena parte con valores hexadecimales equivales a valores decimales: $a = 1_{10}$ & $b = 4_{10}$ *con salida* $s = 5_{10}$

3. Tabla de resultados

Operación	A	B	S	Cout
Suma	5	5	10	0
Suma	12	7	19	0
Suma	9	5	14	0
Suma	14	9	23	0
Suma	4	2	6	0
Suma	7	7	14	0
Suma	15	5	20	0
Suma	11	8	19	0
Suma	1	4	5	0

4. Diagrama RTL

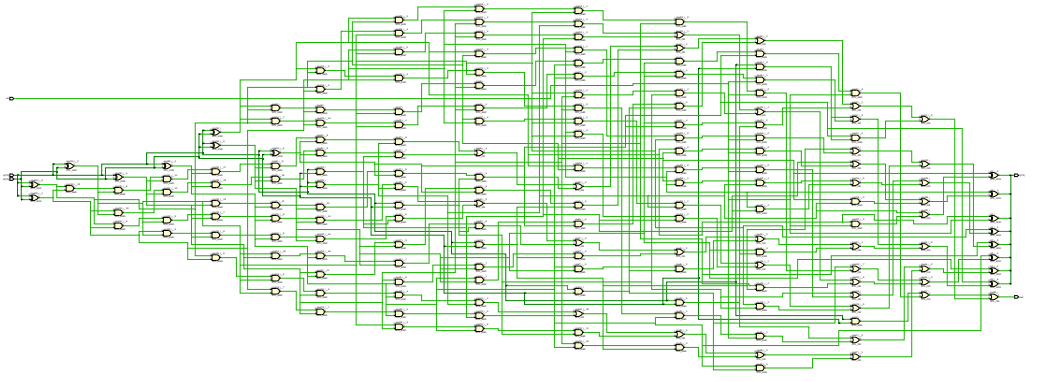


Diagrama RTL del archivo VHDL del sumador con acarreo anticipado de 8 bits

5. Diagrama de Síntesis

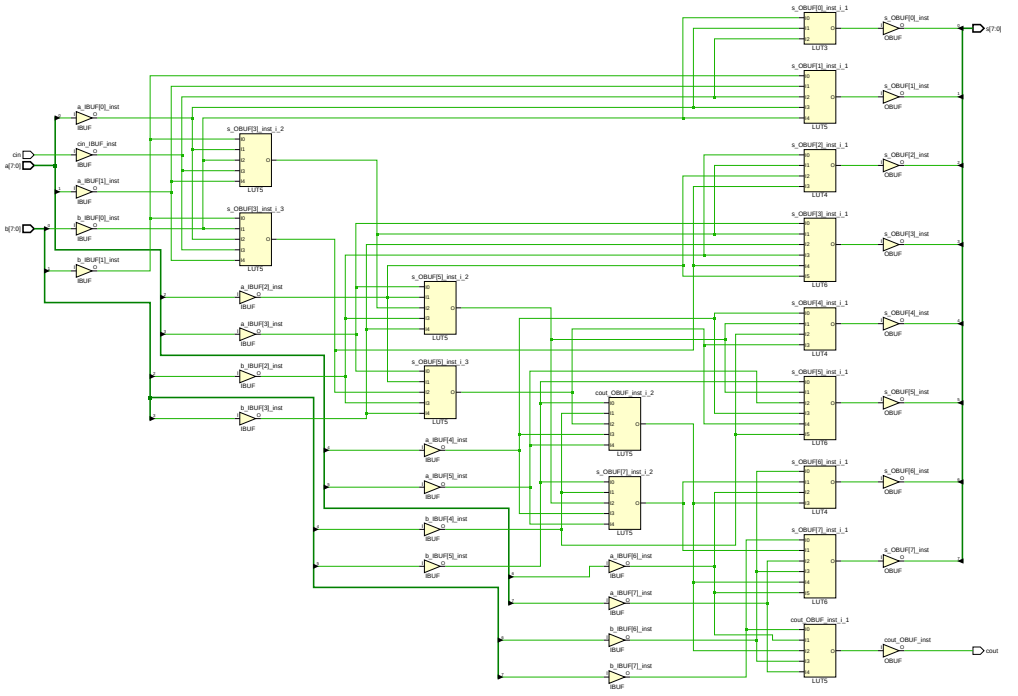


Diagrama de Síntesis del archivo VHDL del sumador con acarreo anticipado de 8 bits