CIRCUITO CONTADOR DE BITS '1'

Primero desarrollamos cada bloque independientemente

NOMBRE DE LA ENTIDAD	ENTRDAS	SALIDAS
Full adder	2 de bits, acarreo	1 bit, acarreo
Sumador2	2 de 2 bits, acarreo	2 bits, acarreo
Sumador3	2 de 3 bits, acarreo	3 bits, acarreo
Sumador4	2 de 4 bits, acarreo	4 bits, acarreo
Incrementador5	Una de 4 bits, acarreo, un bit extra	6 bits

Ahora declaro cada uno de estos en archivos diferentes:

1º problema1.vhd

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

entity fulladder is
    Port ( Ci : in std_logic;
           A : in std_logic;
           B : in std_logic;
           Co : out std_logic;
           s : out std_logic);
end fulladder;

architecture estructural of fulladder is
--la arquitectura en si, sera una ACS incondicional.
begin
s<=A xor B xor Ci;
Co<=(A and B)or (A and Ci) or (B and Ci);

end estructural;
```

2º sumador2.vhd

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

entity sumador2 is
    Port ( a : in std_logic_vector(1 downto 0);
           b : in std_logic_vector(1 downto 0);
           cin : in std_logic;
           s : out std_logic_vector(1 downto 0);
           cout : out std_logic);
end sumador2;

architecture estructural of sumador2 is

component fulladder is
    Port ( Ci : in std_logic;
           A : in std_logic;
           B : in std_logic;
           Co : out std_logic;
           s : out std_logic);
end component;
signal acarreo: std_logic;
begin
U0: fulladder port map (cin,a(0),b(0),acarreo,s(0));
U1: fulladder port map (acarreo,a(1),b(1),cout,s(1));
end estructural;
```

3º sumador3.vhd

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

entity sumador3 is
    Port ( a : in std_logic_vector(2 downto 0);
           b : in std_logic_vector(2 downto 0);
           cin : in std_logic;
           s : out std_logic_vector(2 downto 0);
           cout : out std_logic);
end sumador3;

architecture estructural of sumador3 is

component fulladder is
    Port ( Ci : in std_logic;
           A : in std_logic;
           B : in std_logic;
           Co : out std_logic;
           s : out std_logic);
end component;
signal acarreo1: std_logic;
signal acarreo2: std_logic;
begin
U0: fulladder port map (cin,a(0),b(0),acarreo1,s(0));
U1: fulladder port map (acarreo1,a(1),b(1),acarreo2,s(1));
U2: fulladder port map (acarreo2,a(2),b(2),cout,s(2));
end estructural;
```

4º sumador4.vhd

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

entity sumador4 is
    Port ( a : in std_logic_vector(3 downto 0);
           b : in std_logic_vector(3 downto 0);
           cin : in std_logic;
           s : out std_logic_vector(3 downto 0);
           cout : out std_logic);
end sumador4;

architecture estructural of sumador4 is

component fulladder is
    Port ( Ci : in std_logic;
           A : in std_logic;
           B : in std_logic;
           Co : out std_logic;
           s : out std_logic);
end component;
signal acarreo1: std_logic;
signal acarreo2: std_logic;
signal acarreo3: std_logic;
begin
U0: fulladder port map (cin,a(0),b(0),acarreo1,s(0));
U1: fulladder port map (acarreo1,a(1),b(1),acarreo2,s(1));
U2: fulladder port map (acarreo2,a(2),b(2),acarreo3,s(2));
U3: fulladder port map (acarreo3,a(3),b(3),cout,s(3));
end estructural;
```

5º incrementador5.vhd

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

entity incrementador5 is
    Port ( entrada : in std_logic_vector(4 downto 0);
           cin : in std_logic;
           salida : out std_logic_vector(5 downto 0));
end incrementador5;

architecture algoritmico of incrementador5 is
begin
process(cin,entrada)
variable temporal : std_logic_vector (5 downto 0);
```

```
begin
     temporal:='0'&(entrada);
     if cin='1' then
          temporal:=temporal+1;
     end if;
     salida<=temporal;
end process;
end algoritmico;
```

6º finalmente cuentabits.vhd

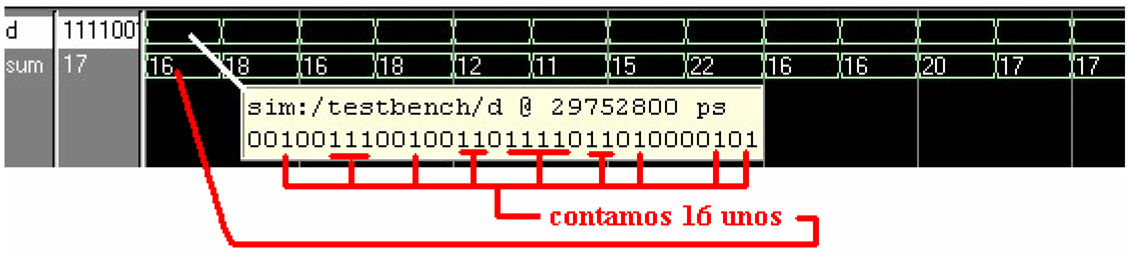
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity cuentabits is
    Port ( D : in std_logic_vector(31 downto 0);
           SUM : out std_logic_vector(5 downto 0));
end cuentabits;

architecture estructural of cuentabits is
--incluimos los componentes de las entidades que
--se van a usar en este circuito
--PRIMER NIVEL
component fulladder is
    Port ( Ci : in std_logic;
           A : in std_logic;
           B : in std_logic;
           Co : out std_logic;
           s : out std_logic);
end component;
--SEGUNDO NIVEL
component sumador2 is
    Port ( a : in std_logic_vector(1 downto 0);
           b : in std_logic_vector(1 downto 0);
           cin : in std_logic;
           s : out std_logic_vector(1 downto 0);
           cout : out std_logic);
end component;
--TERCER NIVEL
component sumador3 is
    Port ( a : in std_logic_vector(2 downto 0);
           b : in std_logic_vector(2 downto 0);
           cin : in std_logic;
           s : out std_logic_vector(2 downto 0);
           cout : out std_logic);
end component;
--CUARTO NIVEL
component sumador4 is
    Port ( a : in std_logic_vector(3 downto 0);
           b : in std_logic_vector(3 downto 0);
           cin : in std_logic;
           s : out std_logic_vector(3 downto 0);
           cout : out std_logic);
end component;
--QUINTO NIVEL
component incrementador5 is
    Port ( entrada : in std_logic_vector(4 downto 0);
           cin : in std_logic;
           salida : out std_logic_vector(5 downto 0));
end component;
--ZONA DE DECLARACION DE SENALES
--PRIMERA CAPA
signal aca1,sal1: std_logic_vector(7 downto 0);
--segunda capa
signal conca1,conca2,conca3,conca5: std_logic_vector (1
downto 0);
signal conca6,conca7,conca4,conca0: std_logic_vector (1
downto 0);
signal aca2: std_logic_vector(3 downto 0);
--tercera capa
signal concb1,concb2,concb3,concb4: std_logic_vector(2 downto
0);
signal sal20,sal21,sal22,sal23: std_logic_vector(1 downto 0);
--cuarta capa
signal aca31,aca32: std_logic;
signal sal31,sal32: std_logic_vector(2 downto 0);
signal concc1,concc2: std_logic_vector (3 downto 0);
--quinta capa
signal concd: std_logic_vector (4 downto 0);
signal aca4: std_logic;
signal sal4: std_logic_vector(3 downto 0);
begin
--PRIMERA CAPA UNIDA A LAS ENTRADAS DEL CIRCUITO
V1: fulladder port map (d(0),d(1),d(2),aca1(0),sal1(0));
V2: fulladder port map (d(3),d(4),d(5),aca1(1),sal1(1));
V3: fulladder port map (d(6),d(7),d(8),aca1(2),sal1(2));
V4: fulladder port map (d(9),d(10),d(11),aca1(3),sal1(3));
V5: fulladder port map (d(12),d(13),d(14),aca1(4),sal1(4));
V6: fulladder port map (d(15),d(16),d(17),aca1(5),sal1(5));
V7: fulladder port map (d(18),d(19),d(20),aca1(6),sal1(6));
V8: fulladder port map (d(21),d(22),d(23),aca1(7),sal1(7));
--SEGUNDA CAPA UNIDA
conca0<=aca1(0)&sal1(0);
conca1<=aca1(1)&sal1(1);
conca2<=aca1(2)&sal1(2);
conca3<=aca1(3)&sal1(3);
conca4<=aca1(4)&sal1(4);
conca5<=aca1(5)&sal1(5);
conca6<=aca1(6)&sal1(6);
conca7<=aca1(7)&sal1(7);
```

```
W1: sumador2 port map (conca0,conca1,d(24),sal20,aca2(0));
W2: sumador2 port map (conca2,conca3,d(25),sal21,aca2(1));
W3: sumador2 port map (conca4,conca5,d(26),sal22,aca2(2));
W4: sumador2 port map (conca6,conca7,d(27),sal23,aca2(3));

--TERCERA CAPA UNIDA
concb1<=aca2(0)&sal20;
concb2<=aca2(1)&sal21;
concb3<=aca2(2)&sal22;
concb4<=aca2(3)&sal23;
X1: sumador3 port map (concb1,concb2,d(28),sal31,aca31);
X2: sumador3 port map (concb3,concb4,d(29),sal32,aca32);
--CUARTA CAPA UNIDA
concc1<=aca31&sal31;
concc2<=aca32&sal32;
Y1: sumador4 port map (concc1,concc2,d(30),sal4,aca4);
--QUINTA CAPA - CAPA FINAL
concd<=aca4&sal4;
Z1: incrementador5 port map (concd,d(31),SUM);
end estructural;
```



```
d    111100
sum  17     16  18  16  18  12  11  15  22  16  16  20  17  17
sim:/testbench/d @ 29752800 ps
00100111001001101111011010000101
```