

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  use IEEE.NUMERIC_STD.ALL;
4
5  entity velocimetro is
6      Port ( clk           : in STD_LOGIC;
7            reset         : in STD_LOGIC;
8            sensor        : in STD_LOGIC;
9            VueltasOut    : out std_logic_vector(5 downto 0));
10 end velocimetro;
11
12 architecture Behavioral of velocimetro is
13
14     constant ContalsegMAX : integer := 50*(10**6) - 1;
15     signal contalseg : integer range 0 to ContalsegMAX;
16     signal UnSeg : std_logic;
17
18     signal Vueltas: unsigned(5 downto 0);
19
20 begin
21
22     -- Divisor de Frecuencia a 1 segundo
23     DivFreq: process(clk,reset)
24     begin
25         if reset = '1' then
26             contalseg <= 0;
27         elsif clk'event and clk = '1' then
28             if contalseg < ContalsegMAX then
29                 contalseg <= contalseg + 1;
30             else
31                 contalseg <= 0;
32             end if;
33         end if;
34     end process;
35     UnSeg <= '1' when contalseg = ContalsegMAX else '0';
36
37     -- Proceso en el que se cuentan las vueltas y se actualiza la salida (1 vez
38     -- por segundo)
39     ContaVueltas: process(clk,reset)
40     begin
41         if reset = '1' then
42             Vueltas <= "000000";
43             VueltasOut <= "000000";
44         elsif clk'event and clk = '1' then
45             if Unseg = '1' then
46                 VueltasOut <= std_logic_vector(Vueltas);
47                 Vueltas <= "000000";
48             elsif sensor = '1' then
49                 Vueltas <= Vueltas + 1;
50             end if;
51         end if;
52     end process;
53 end Behavioral;
54
```

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  use IEEE.NUMERIC_STD.ALL;
4
5  entity TopVelocimetro is
6      Port ( clk : in STD_LOGIC;
7            reset : in STD_LOGIC;
8            sensor : in STD_LOGIC;
9            leds : out STD_LOGIC_VECTOR(7 DOWNTO 0)
10           );
11 end TopVelocimetro;
12
13 architecture Behavioral of TopVelocimetro is
14
15     component velocimetro is
16         Port ( clk : in STD_LOGIC;
17               reset : in STD_LOGIC;
18               sensor : in STD_LOGIC;
19               VueltasOut : out std_logic_vector(5 downto 0));
20     end component;
21
22     signal VueltasOut : std_logic_vector(5 downto 0);
23     signal uV : unsigned(5 downto 0);
24
25 begin
26
27     -- Instancia del componente velocímetro
28     instVelo: velocimetro
29         port map(
30             clk => clk,
31             reset => reset,
32             sensor => sensor,
33             VueltasOut => VueltasOut
34         );
35
36     --Transformación a unsigned para facilitar la codificación
37     uV <= unsigned(VueltasOut);
38
39     leds <= "00000001" when uV < 8 else
40            "00000010" when uV < 16 else
41            "00000100" when uV < 24 else
42            "00001000" when uV < 32 else
43            "00010000" when uV < 40 else
44            "00100000" when uV < 48 else
45            "01000000" when uV < 56 else
46            "10000000";
47 end Behavioral;
48
```

```

1  -----
2  -- Electrónica Digital --
3  -- PEC diciembre 2017 --
4  -- Cuestión 3          --
5  -----
6
7  library ieee;
8  use ieee.std_logic_1164.all;
9  use ieee.numeric_std.all;
10
11 entity c3 is
12     port (
13         clk    : in  std_logic;
14         reset  : in  std_logic;
15         seq    : out std_logic
16     );
17 end c3;
18
19 -----
20 -- La solución comienza a partir de aquí --
21 -----
22
23 -- a) Base: contador binario síncrono
24 architecture cnt_binario of c3 is
25     signal cnt : unsigned(2 downto 0);
26 begin
27
28     process(clk,reset)
29     begin
30         if reset = '1' then
31             cnt <= (others => '0');
32         elsif clk'event and clk = '1' then
33             cnt <= cnt + 1;
34         end if;
35     end process;
36
37     with cnt select
38         seq <= '1' when "000",
39              '1' when "001",
40              '0' when "010",
41              '1' when "011",
42              '1' when "100",
43              '0' when "101",
44              '0' when "110",
45              '1' when "111",
46              '-' when others;
47
48 end cnt_binario;
49
50 -- b) Base: contador Johnson
51 architecture cnt_johnson of c3 is
52     signal shiftreg : std_logic_vector(3 downto 0);
53 begin
54
55     process(clk,reset)
56     begin
57         if reset = '1' then
58             shiftreg <= (others => '0');
59         elsif clk'event and clk = '1' then
60             shiftreg <= shiftreg(2 downto 0) & not shiftreg(3);
61         end if;
62     end process;
63
64     with shiftreg select
65         seq <= '1' when "0000",
66              '1' when "0001",
67              '0' when "0011",
68              '1' when "0111",
69              '1' when "1111",
70              '0' when "1110",
71              '0' when "1100",
72              '1' when "1000",
73              '-' when others;

```

```

74
75 end cnt_johnson;
76
77 -- c) Base: contador en anillo
78 architecture cnt_anillo of c3 is
79     signal shiftreg : std_logic_vector(7 downto 0);
80 begin
81
82     process(clk,reset)
83     begin
84         if reset = '1' then
85             shiftreg <= (0 => '1', others => '0');
86         elsif clk'event and clk = '1' then
87             shiftreg <= shiftreg(6 downto 0) & shiftreg(7);
88         end if;
89     end process;
90
91     with shiftreg select
92     seq <= '1' when "00000001",
93           '1' when "00000010",
94           '0' when "00000100",
95           '1' when "00001000",
96           '1' when "00010000",
97           '0' when "00100000",
98           '0' when "01000000",
99           '1' when "10000000",
100           '-' when others;
101
102 end cnt_anillo;
103

```