Contador ascendente y descendente en VHDL

El objetivo es diseñar y simular el siguiente bloque contador en VHDL.

Debe ser capaz de contar 4 bits ascendente y descendentemente con un control de sentido.

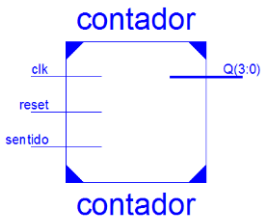

Fig. 1

1. **Creamos un nuevo proyecto en el ISE Project Navigator**[1]

File -> New Project

Se abrirá la siguiente interfaz.

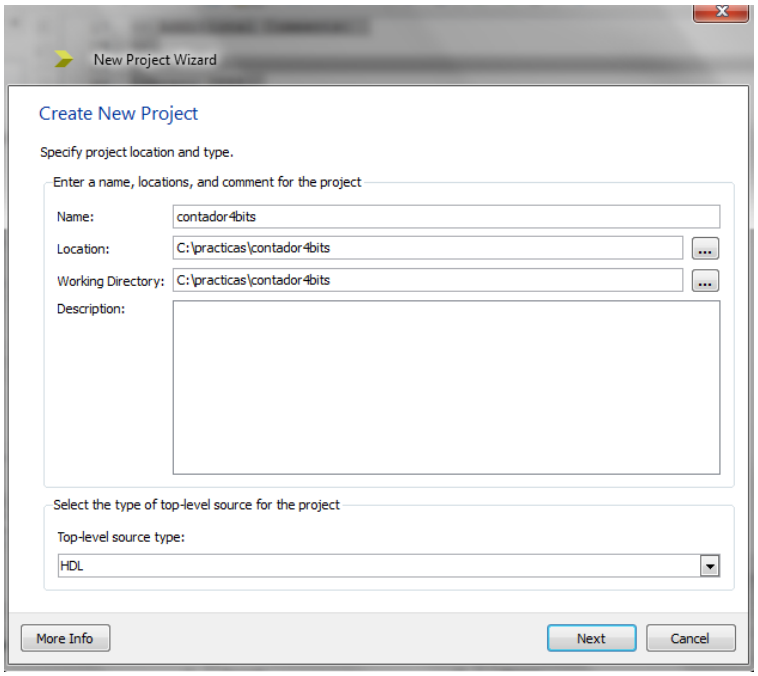

Fig. 2 Nuevo Proyecto

Ponemos un nombre que no contenga espacios por ejemplo “contador4bits”.

[1] Para el presente tutorial se ha utilizado la versión 14.2 de Xilinx ISE Design Suite. Con licencia ISE Web Pack.

Elegimos la locación y el directorio de trabajo (Dejar las ubicaciones por defecto)

Presionamos “next”.

2. **En la siguiente ventana debemos especificar la tarjeta en la que vamos a desarrollar el proyecto.**

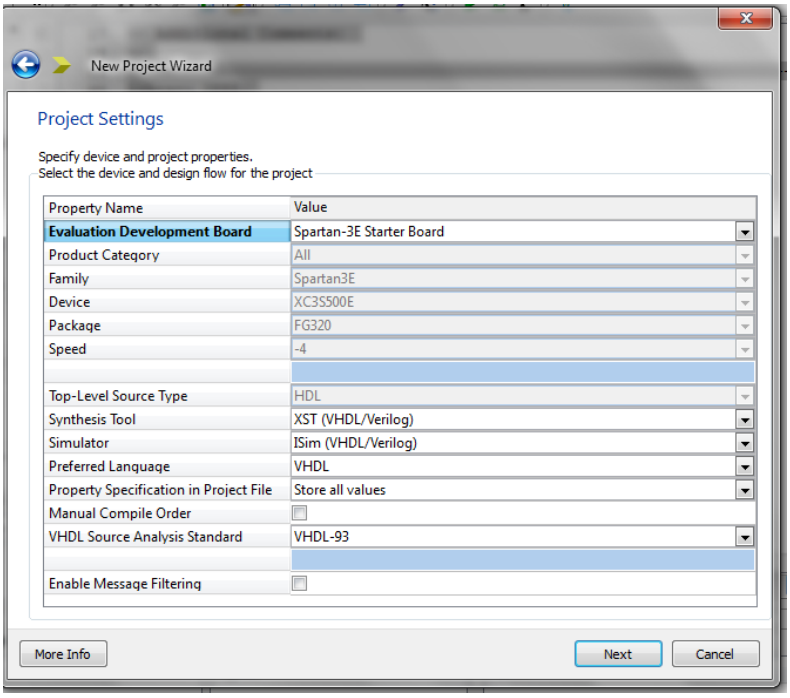

Fig. 3 Especificaciones de tecnología

En nuestro caso contamos con la tarjeta Spartan 3E Starter Board que contiene las siguiente FPGA:

Family: Spartan3E

Device: XC3S500E

Package: FG320

Speed: -4

*Los datos de arriba se pueden obtener directamente del chip FPGA incluido en la tarjeta.

Top-Level Source Type: HDL

Synthesis Tool: XST (VHDL/Verilog)

Simulator: ISim (VHDL/Verilog)

Preferred lenguaje: VHDL

Presionamos “next”.

3. **Aparecerá la siguiente ventana que simplemente detalla el proyecto creado.**

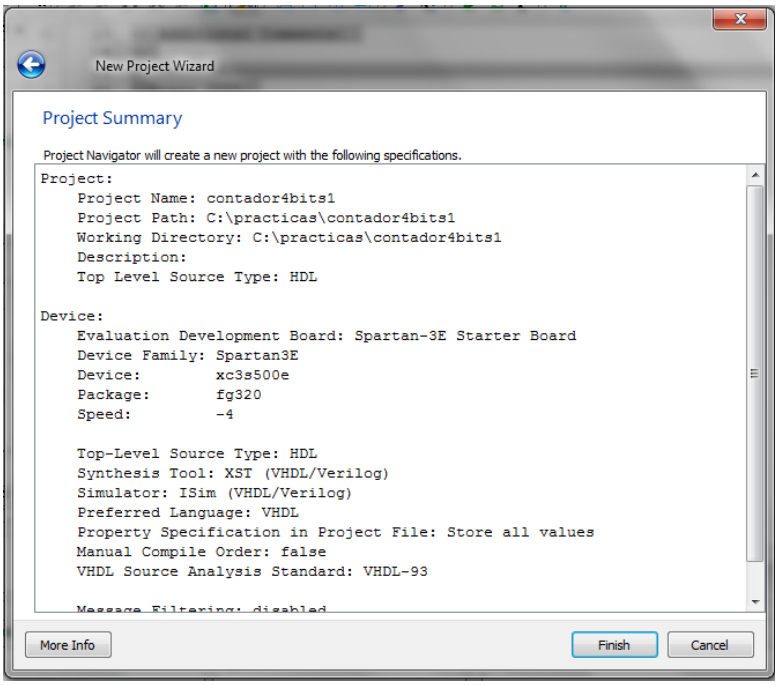

Fig. 4

Presionamos “Finish”.

4. **En el tab “design” de la izquierda hacemos doble clic sobre el ícono de nuestro proyecto y creamos un nuevo recurso (New Source), como en la fig 5.**

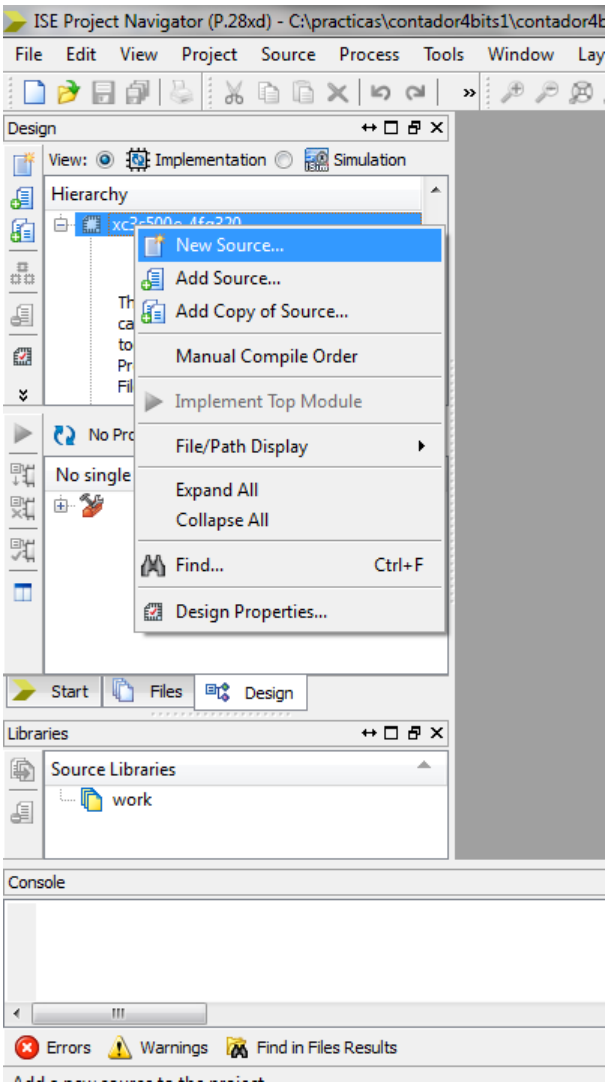

Fig. 5

5. **En la ventana emergente seleccionamos el tipo de recurso a crear, como queremos diseñar el contador en vhdl, escogemos un nuevo recurso de tipo "VHDL module".**

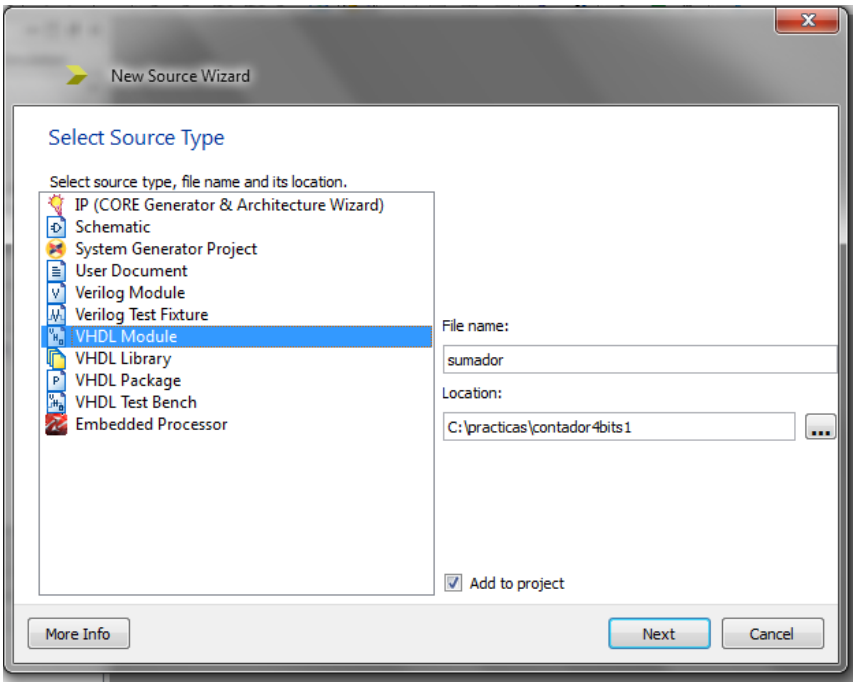

Fig. 6

En “File name” le damos un nombre al módulo VHDL a crear, igualmente no debe contener espacios por ejemplo “sumador”.

Clic en “next”.

6. **En la siguiente ventana podemos ingresar las entradas y salidas que tendrá la entidad. (consulte la Fig. 1).**

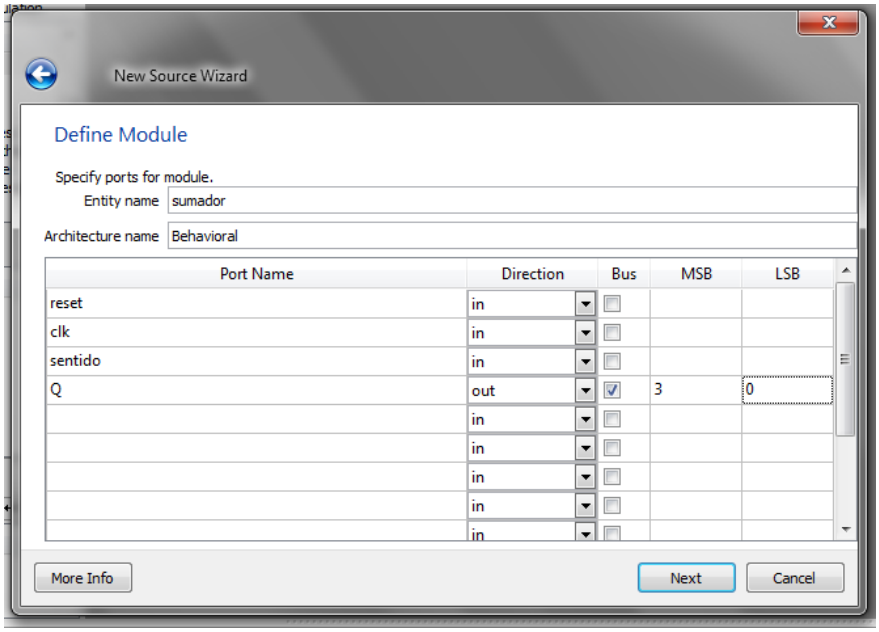

Fig. 7

Declaramos como entradas en la columna Port Name:

Reset -> in

clk -> in

sentido -> in

Q -> out -> bus -> MSB 3 LSB 0

En salida “Q”, seleccionamos la opción de “Bus” debido a que es la salida del contador de 4 bits, por ende debe tener de 0 a 3 bits.

A continuación damos click en “next”, y luego en “finish” en la ventana resumen.

7. **Ahora está preparado el archivo Vhdl para su edición.**

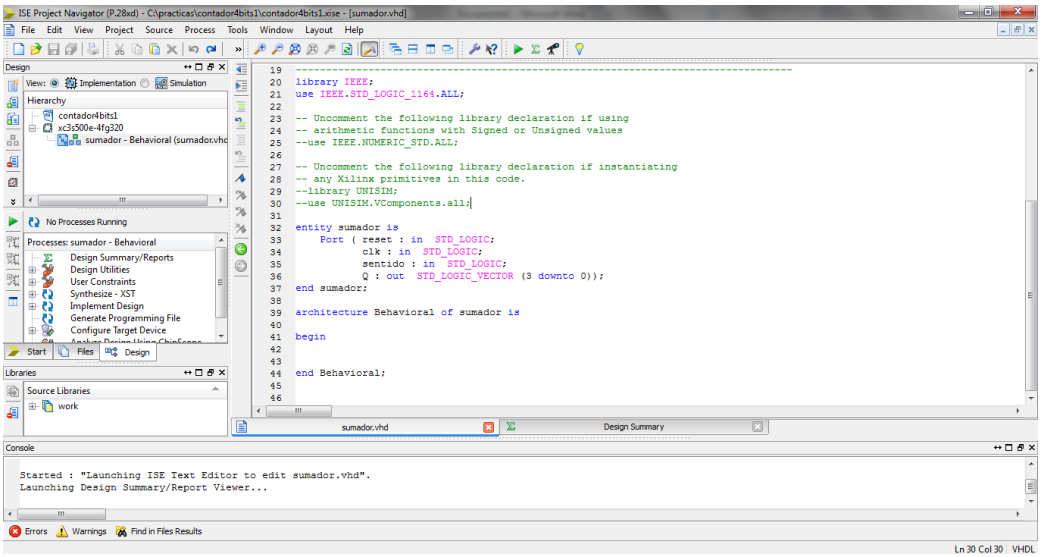

Fig. 8

En el bloque de código de las librerías debemos incluir las siguiente sentencia “use IEEE.std_logic_unsigned.all;” esto para poder trabajar con sumas y restas sin signo.

Quedaría el código superior así:

---

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_unsigned.all;

8. La declaración de la entidad debe quedar de la siguiente manera:

---

entity contador is
Port (reset : in STD_LOGIC;

```
        clk : in  STD_LOGIC;
        sentido : in STD_LOGIC := '0';
         Q : out  STD_LOGIC_VECTOR (3 downto 0));
end contador;
```

*recordar que las líneas de código anteriores pueden generarse automáticamente siguiendo el proceso del paso 6, o pueden escribirse directamente en el código obviando el paso 6. Sin embargo estas líneas pueden modificarse en cualquier momento.

9. **Luego describimos la arquitectura de la entidad, esta arquitectura será de tipo comportamental o "behavioral".**

```
architecture Behavioral of contador is

signal counter : STD_LOGIC_VECTOR (3 downto 0) := "0000";
begin
        process(reset,clk)

begin

        if reset='1' then
                counter <= "0000";
        elsif clk'event and clk = '1' then
                if sentido = '1' then
                        counter <= counter + 1;
                 else
                        counter <= counter - 1;
                end if;
        Q <= counter;
        end if;
end process;
end Behavioral;
```

Arquitectura comportamental.

En cada pulso de reloj se incrementa una unidad a la señal contador

10. El código complete quedaría así:

```
----------------------------------------------------------------------------------

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_unsigned.all;


entity contador is
   Port ( reset : in  STD_LOGIC;
          clk : in  STD_LOGIC;
         sentido : in STD_LOGIC := '0';
```

```
        Q : out  STD_LOGIC_VECTOR (3 downto 0));
end contador;

architecture Behavioral of contador is

signal counter : STD_LOGIC_VECTOR (3 downto 0) := "0000";
begin
            process(reset,clk)

begin

                        if reset='1' then
                                    counter <= "0000";
                        elsif clk'event and clk = '1' then
                                    if sentido = '1' then
                                                counter <= counter + 1;
                          else
                                                counter <= counter - 1;
                                    end if;
                        Q <= counter;
                        end if;

end process;

end Behavioral;
```

11. SIMULACIÓN

Una vez terminado nuestro código podemos realizar la simulación para constatar el correcto funcionamiento.

Creamos un nuevo recurso como en el paso 4, con excepción de que esta vez necesitamos un recurso de tipo “VHDL test bench”.

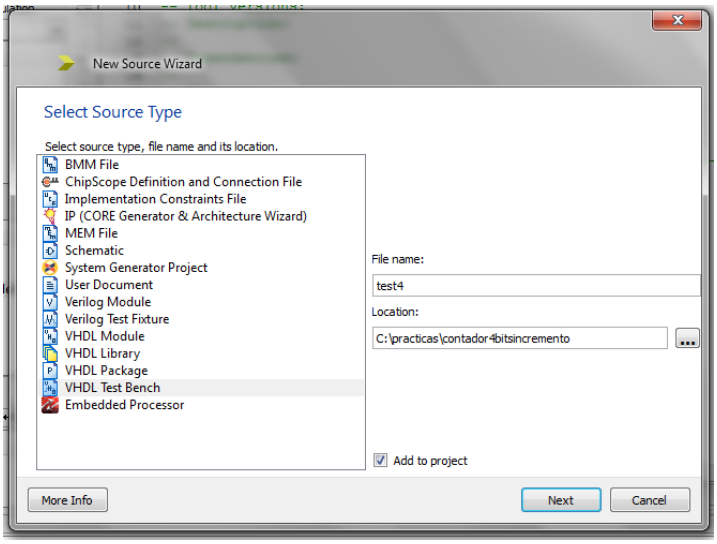

Fig. 9

Le asignamos un nombre al nuevo recurso, click en “next”.
En la ventana siguiente se nos pide que asociemos un recurso a este “test bench”, seleccionamos el recurso correcto, en este caso solo existe un recurso.

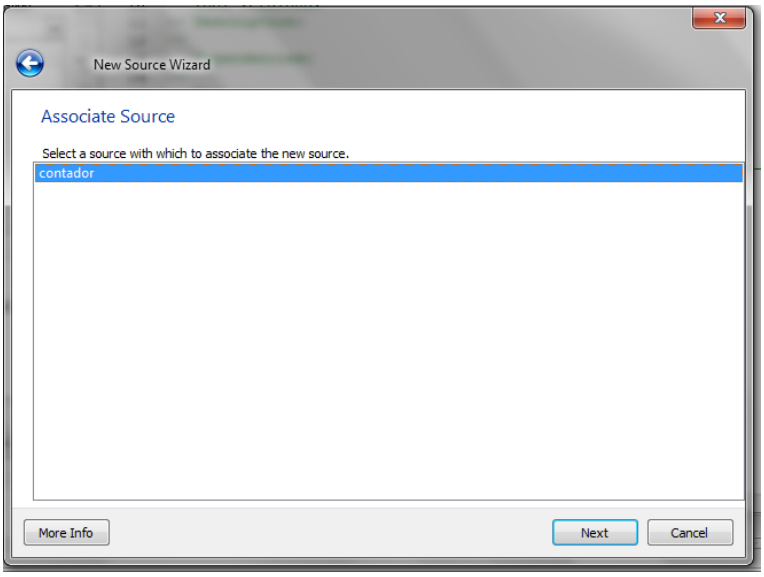

Fig. 10

A continuación clic en “next” y luego en “finish”.

Ahora tenemos un archivo de tipo VHDL pero con las señales, puertos y temporizaciones necesarias para una simulación.

*Seleccionando la interfaz “simulation” en la barra de la izquierda podemos visualizar el archivo de simulación.

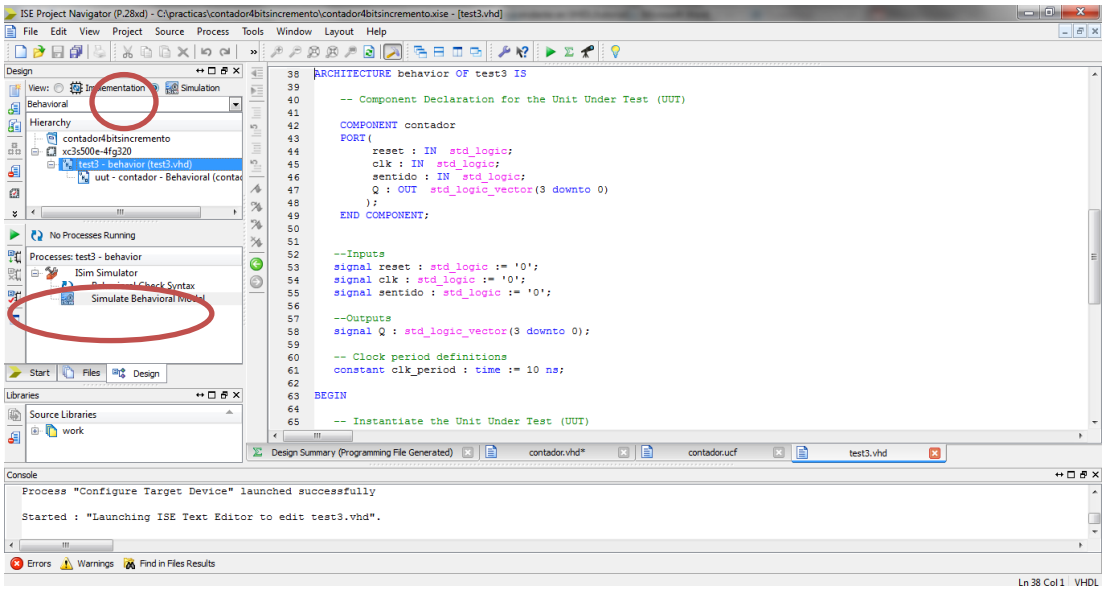

Fig. 11

Seleccionamos el archivo “VHDL test bench” recién creado, chequeamos la sintaxis en la barra de la izquierda y luego procedemos a correr la simulación.

12. Al terminar de correr el proceso “Simulate Behavioral Model”, se abrirá otro programa llamado “Isim”.

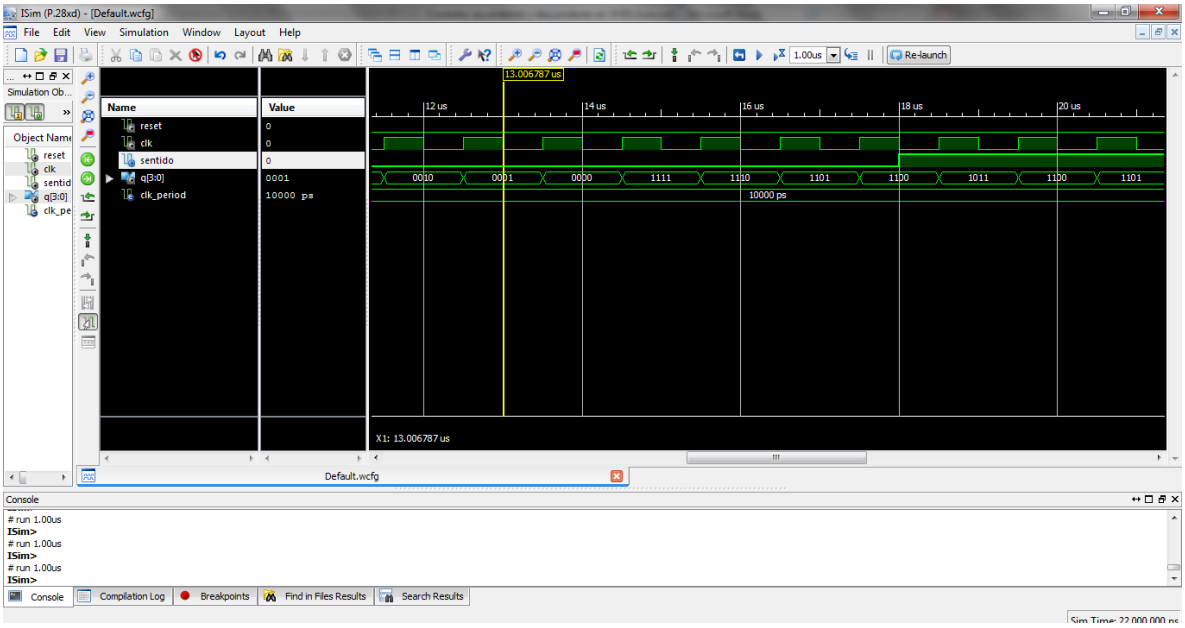

Fig. 12

En el mismo podemos interactuar con las señales de entrada y observar la simulación y comportamiento de nuestro circuito.

Para modificar las señales de entradas se procede de la siguiente manera:

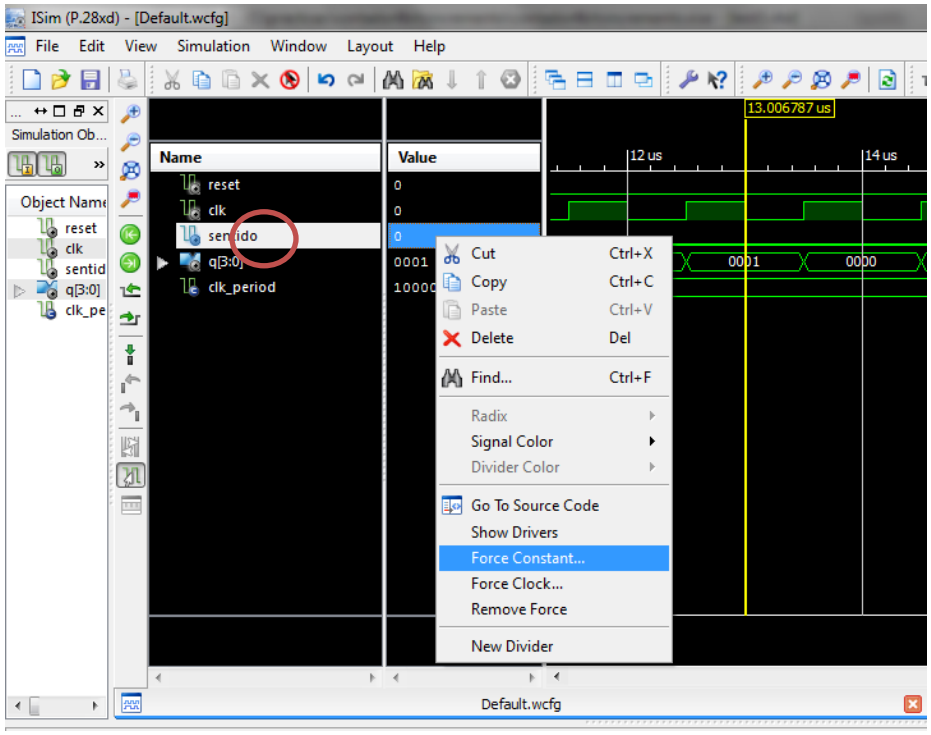

Fig. 13

Damos click derecho en cualquier señal de entrada, por ejemplo en “sentido”. Y seleccionamos “force constant”.

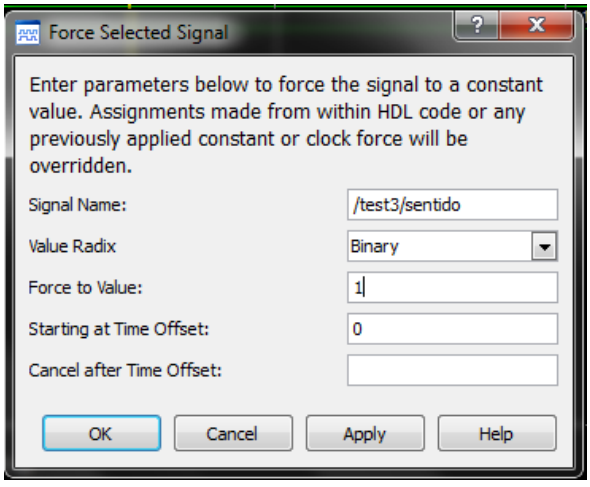

Fig. 14

En la ventana emergente damos un valor de ‘0’ o ‘1’, en “Force to value”.

*Según nuestro ejemplo si la señal “sentido” está en ‘0’ el contador descenderá, caso contrario ‘1’, el contador ascenderá.

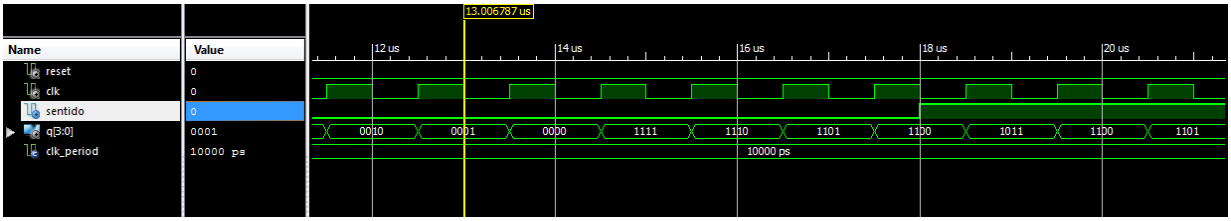

Fig. 15

En la figura 15 se puede observar el funcionamiento del contador y el cambio de la señal sentido a los 18us.

Para correr la simulación y observar las señales se debe dar clic en el ícono que se observa en Fig 16 e interactuar con los íconos de zoom out y zoom in.

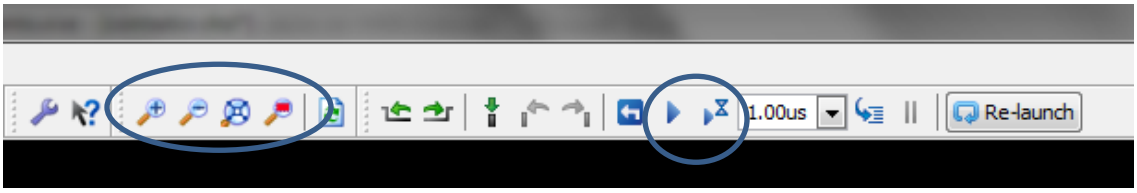

Fig. 16

Creado por:
Tuesman Daniel Castillo Ing.
Escuela de Electrónica y Telecomunicaciones UTPL
Universidad Técnica Particular de Loja UTPL