

Práctica N° 1. Introducción al entorno ISE® de Xilinx

1. Datos de la Práctica

Carrera	INGENIERÍA ELECTRÓNICA		
Semestre		Grupo	
Tipo de Práctica	<input type="checkbox"/> Laboratorio <input type="checkbox"/> Simulación	Fecha	
Asignatura	Electrónica Digital I		
Unidad Temática			
N° Alumnos por práctica	2	N° Alumnos por reporte	2
Nombre del Profesor			
Nombre(s) de Alumno(s)	1. 2.		
Tiempo estimado		Vo. Bo. Profesor	
Comentarios			

2. Objetivo

Familiarizar al usuario en el uso del ambiente de software integrado Xilinx ISE®v14.7 Project Navigator y el kit de desarrollo Basys2 mediante la creación de un programa de lógica combinatorial y su respectiva simulación.

3. Medios a utilizar

Por cada práctica y por cada puesto de laboratorio, los materiales a utilizar son:

Cantidad	Descripción
1	Computadora
1	Tarjeta de desarrollo Basys2 Digilent
1	Software Xilinx ISE® Webpack v14.7

4. Introducción

Los circuitos digitales presentan dos estados, un estado se aproxima a cero volts (low) y el otro estado suministra el voltaje (high) y se pueden procesar en números binarios, en este caso 1 o 0.

Los circuitos digitales pueden estar conectados entre sí por medio de compuertas digitales: OR, NOR, AND, exclusivamente OR, exclusivamente NOR y NAND. En esta práctica de laboratorio se implementan las compuertas: AND, OR y XOR. El programa consiste en describir tres salidas c, d y e como la combinación lógica de las entradas a y b. Así, $c = a + b$, $d = a * b$ y $e = a \oplus b$.

5. Actividades previas

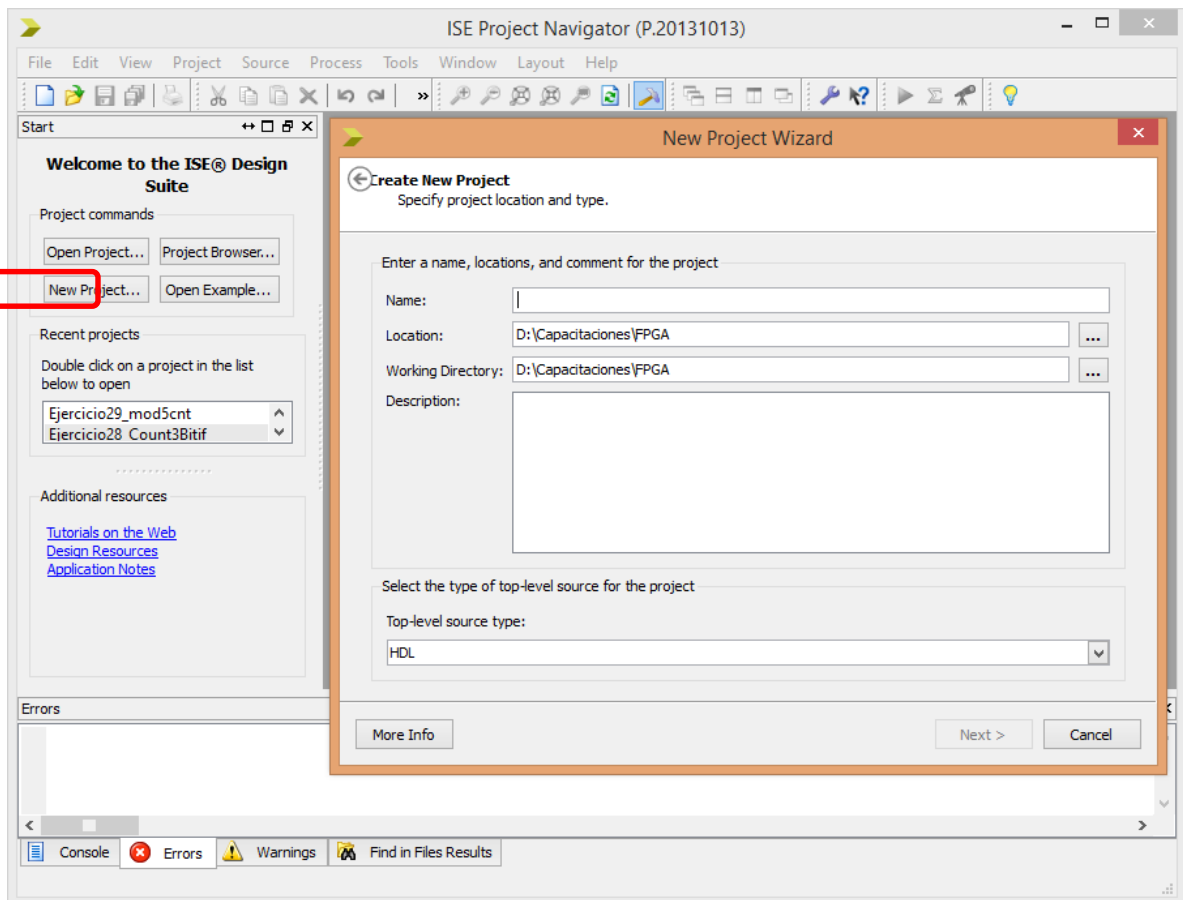
Completar la tabla de las compuertas AND, OR y XOR con las ecuaciones descritas anteriormente.

Entradas		Salidas		
a	b	c	d	e
0	0			
0	1			
1	0			
1	1			

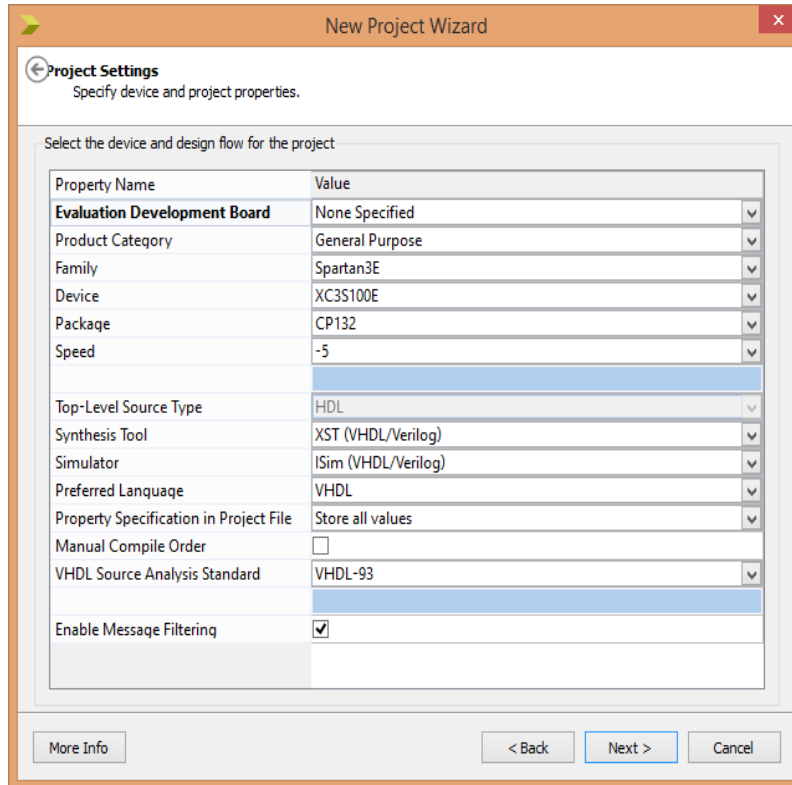
6. Desarrollo de la práctica

Creación de un proyecto en Project Navigator

1. Abrir Xilinx-ISE®.
2. Cerrar cualquier programa que esté abierto (File → Close Project).
3. Abrir un nuevo proyecto (File → New Project). Escribir el nombre del proyecto y presionar en la pestaña Next.



4. Asegurar que las propiedades del proyecto son iguales a la de la figura y presionar la pestaña Next.



New Project Wizard

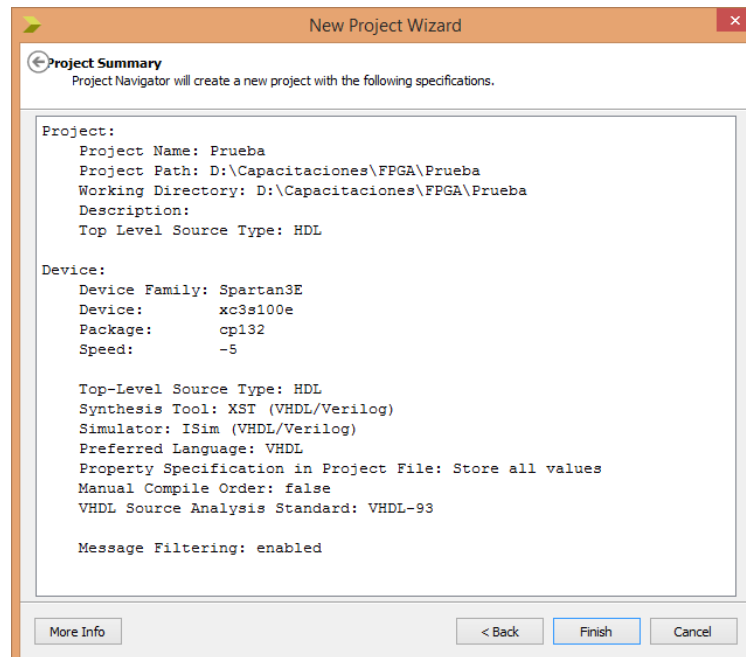
Project Settings
Specify device and project properties.

Select the device and design flow for the project

Property Name	Value
Evaluation Development Board	None Specified
Product Category	General Purpose
Family	Spartan3E
Device	XC3S100E
Package	CP132
Speed	-5
Top-Level Source Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	VHDL
Property Specification in Project File	Store all values
Manual Compile Order	<input type="checkbox"/>
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	<input checked="" type="checkbox"/>

More Info < Back Next > Cancel

5. En la siguiente ventana, clic en la pestaña Finish.



New Project Wizard

Project Summary
Project Navigator will create a new project with the following specifications.

Project:

- Project Name: Prueba
- Project Path: D:\Capacitaciones\FPGA\Prueba
- Working Directory: D:\Capacitaciones\FPGA\Prueba
- Description:
- Top Level Source Type: HDL

Device:

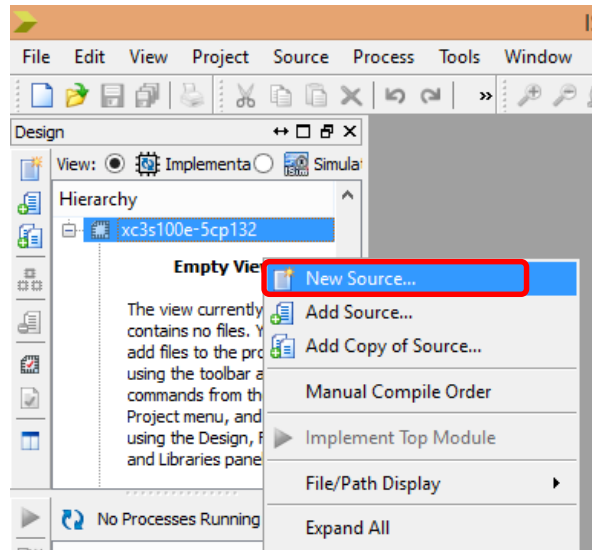
- Device Family: Spartan3E
- Device: xc3s100e
- Package: cp132
- Speed: -5

Top-Level Source Type: HDL
Synthesis Tool: XST (VHDL/Verilog)
Simulator: ISim (VHDL/Verilog)
Preferred Language: VHDL
Property Specification in Project File: Store all values
Manual Compile Order: false
VHDL Source Analysis Standard: VHDL-93

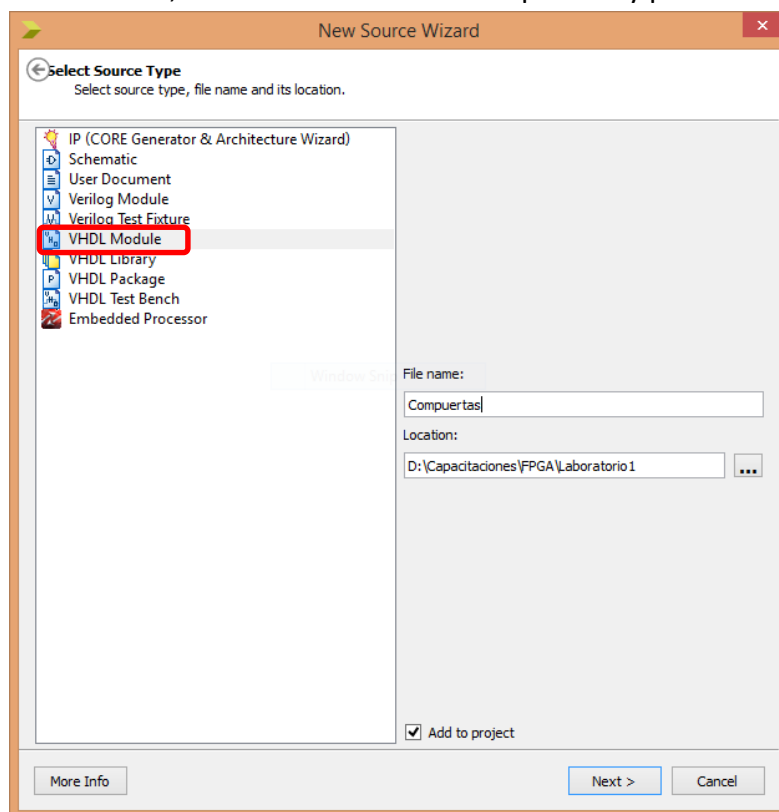
Message Filtering: enabled

More Info < Back Finish Cancel

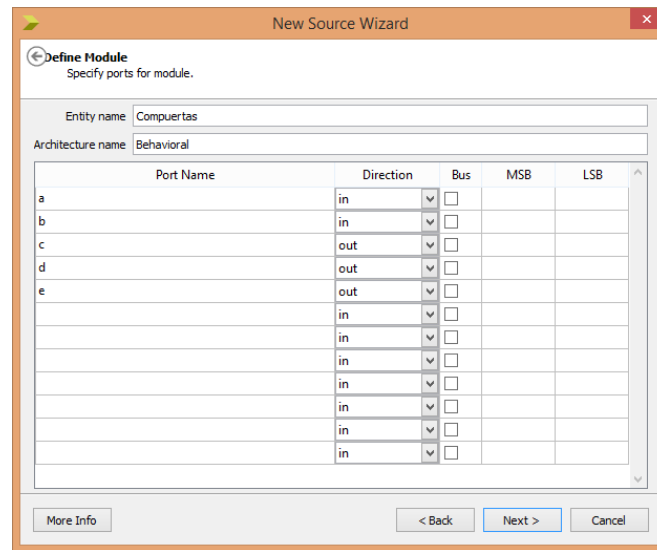
6. Crear una nueva fuente. En la ventana Sources clic derecho en la carpeta y seleccionar New Source... y presionar Next.



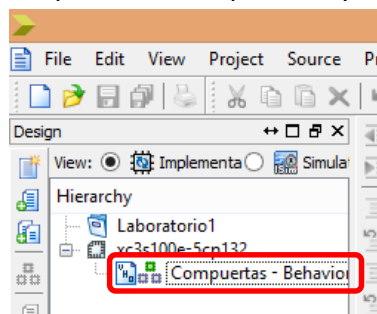
7. Elegir la opción VHDL Module, en File name escribir Compuertas y presionar Next.



8. En la siguiente ventana definimos las entradas y salidas del módulo. Las variables a y b están definidas como entradas (input) y las variables c, d y e como salida (output).



9. Clic en todas las demás opciones: Next → Finish.
10. En la ventana Sources verificar que aparezca el dispositivo y el módulo principal.




11. En la ventana de edición aparece las partes del código VHDL, en el cual contienen las librerías, la entidad y la arquitectura.

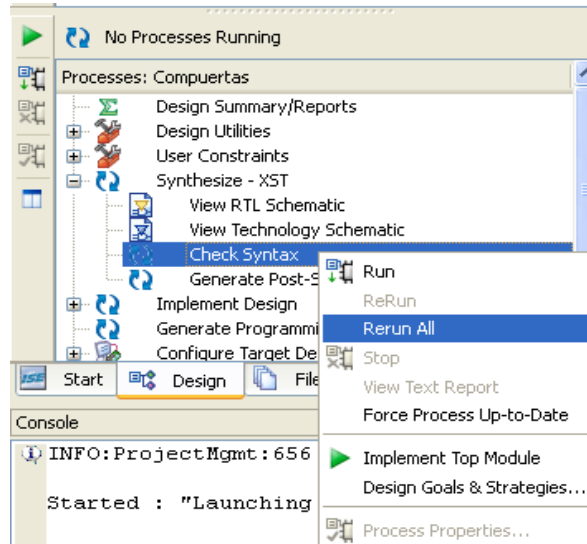
```

20 library IEEE;
21 use IEEE.STD_LOGIC_1164.ALL; } Librerías
22
23 entity Compuertas is
24     Port ( a : in  STD_LOGIC;
25           b : in  STD_LOGIC;
26           c : out STD_LOGIC;
27           d : out STD_LOGIC;
28           e : out STD_LOGIC); } Entidad
29 end Compuertas;
30
31 architecture Behavioral of Compuertas is
32
33 begin
34     c <= a or b;
35     d <= a and b;
36     e <= a xor b;
37
38 end Behavioral; } Arquitectura

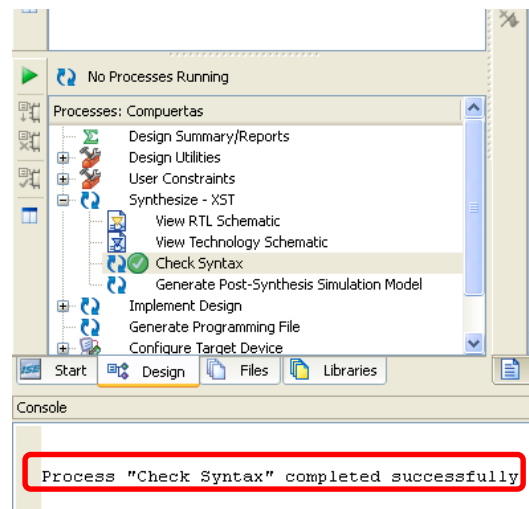
```

12. Guardar los cambios en el código, realizados en el paso anterior.
13. Seleccionar el módulo principal  **Compuertas - Behavioral (Compuertas.vhd)** en la ventana Sources.

14. En Processes, elegir Synthesize-XST en el más desplegable, clic derecho en la opción Check Syntax y elegir Rerun All.



15. En la ventana Console, verificar que aparezca el siguiente mensaje.

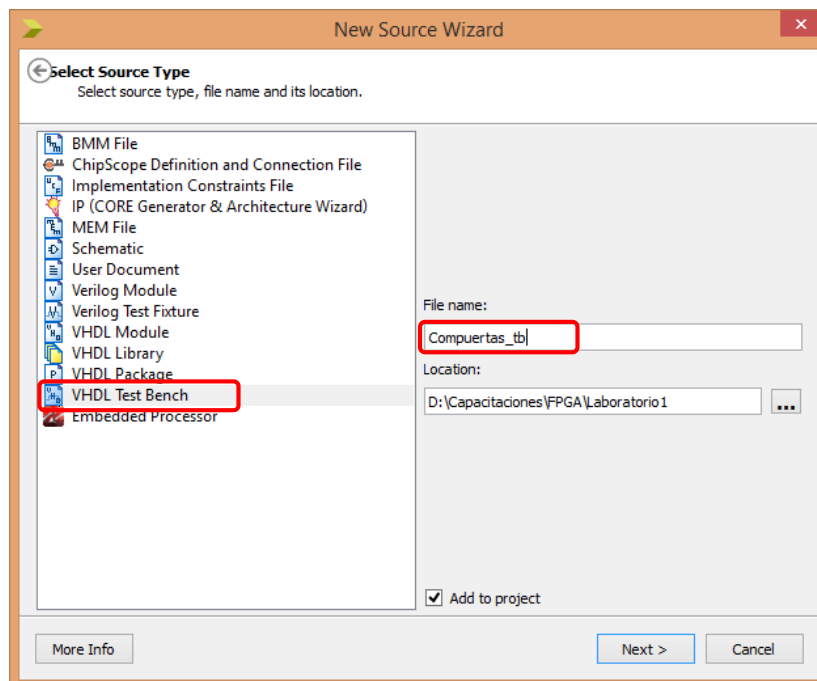


Nota: El proceso verifica que no haya errores de sintaxis, de haberlo hay que corregir.

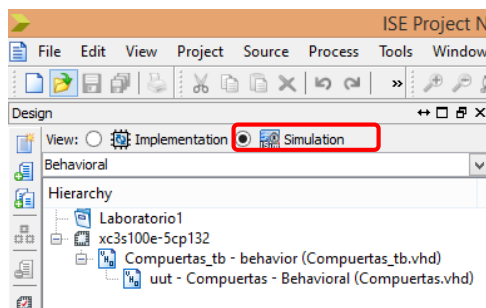
Simulación de un circuito digital

Durante la simulación en HDL, el software verifica la funcionalidad y la temporización del circuito como parte del proceso de diseño. Primero hay que tener una descripción del circuito a realizar a través del lenguaje VHDL. La simulación es un proceso iterativo, es decir que puede ser un proceso repetitivo hasta que la funcionalidad del tiempo y del diseño sea satisfactorio.

1. Una vez verificado la sintaxis, el siguiente paso sería la simulación, en el cual crearemos un nuevo archivo VHDL llamado "VHDL Test Bench". Para poder hacerlo se agrega una nueva fuente "New Source" y escogemos el tipo de archivo el cual llamaremos `Compuertas_tb`.



2. En Design, escoger en modo Simulation.



3. Aparece en la ventana principal lo siguiente

```

61         e => e
62     );
63
64     -- Clock process definitions
65     <clock>_process :process
66     begin
67         <clock> <= '0';
68         wait for <clock>_period/2;
69         <clock> <= '1';
70         wait for <clock>_period/2;
71     end process;
72
73
74     -- Stimulus process
75     stim_proc: process
76     begin
77         -- hold reset state for 100 ns.
78         wait for 100 ns;
79
80         wait for <clock>_period*10;
81
82         -- insert stimulus here
83
84         wait;
85     end process;
86
87 END;
88

```

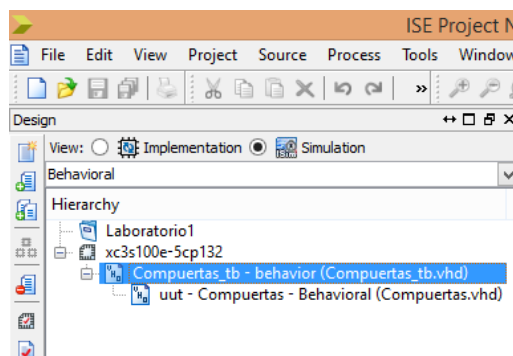
4. Comentar las líneas referentes a circuitos de reloj con las líneas --.
5. Añadir las siguientes líneas después del comentario **--Insert stimulus here** y antes de la sentencia **end**.

```

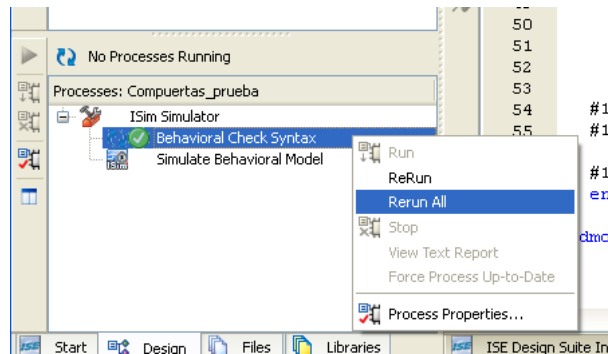
a <= '0';
b <= '1';
wait for 100 ns;
a <= '1';
b <= '0';
wait for 100 ns;
a <= '1';
b <= '1';
wait for 100 ns;
a <= '0';
b <= '0';
wait for 100 ns;

```

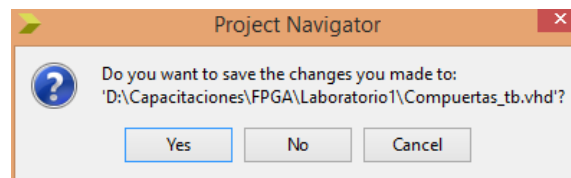
6. Seleccionar Compuertas_tb.vhd en la ventana Sources.




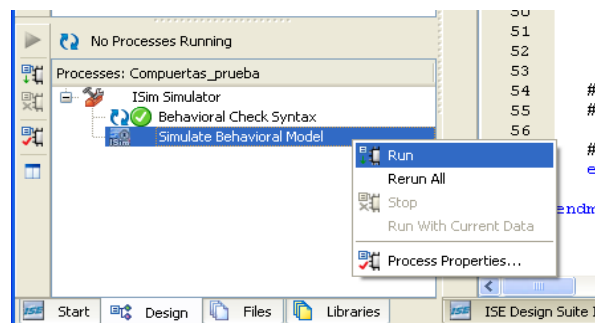
7. Clic derecho en Behavioral Check Syntax → Processes → Rerun All.



Aparece la siguiente ventana, clic en Yes.

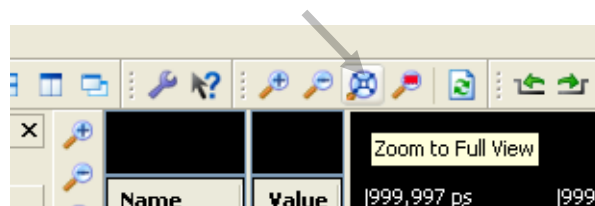


8. Verificar que aparezca un  en Behavioral Check Syntax, elegir la opción Simulate Behavioral Model y clic derecho en Run.

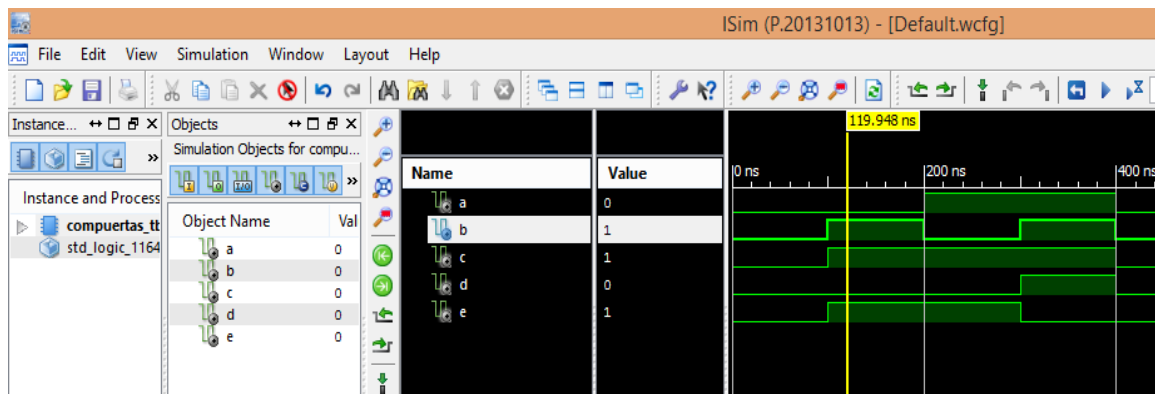


9. Aparece la ventana ISim Simulator donde se muestra la forma de onda de los estímulos agregados en el paso 6.

Utilizar la herramienta *Zoom to Full View* para ver la vista completa de la simulación, que se encuentra a la derecha en la barra de herramientas del panel de simulación.



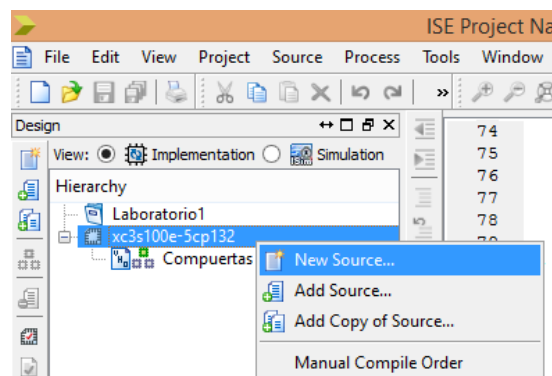
En la ventana **Wave Window** donde se visualiza la configuración de las formas de onda, que contienen la lista de las señales y su forma de onda¹.



10. Mover la línea amarilla sobre las formas de onda (de color verde) y ver en el cuadro Value que los valores de las señales cambian. Comprobar que las formas de onda de la figura anterior corresponden a las tablas de verdad de las compuertas hechas en el código.

Asignación de los pines en la tarjeta Basys2

1. En el modo Implementation, dar clic derecho en Compuertas.vhd y elegir New Source...

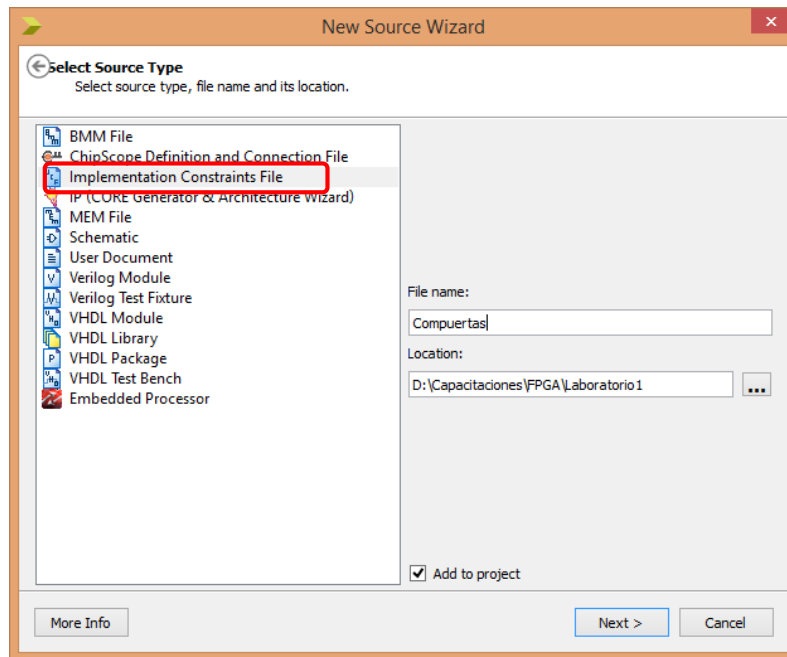


2. Elegir la opción Implementation Constraints File, en File Name escribir Compuertas, clic en Next → Finish.

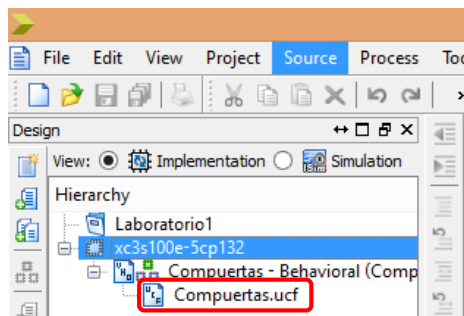
¹ Más información? Visita

http://www.xilinx.com/support/documentation/sw_manuals/xilinx12_2/plugin_ism.pdf

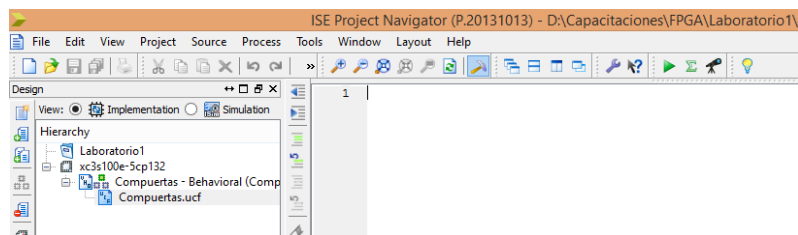
Elaborado por: Ing. Carlos Ortega



3. En la ventana Sources, verificar que la nueva fuente que acabamos de crear aparece debajo del módulo principal.



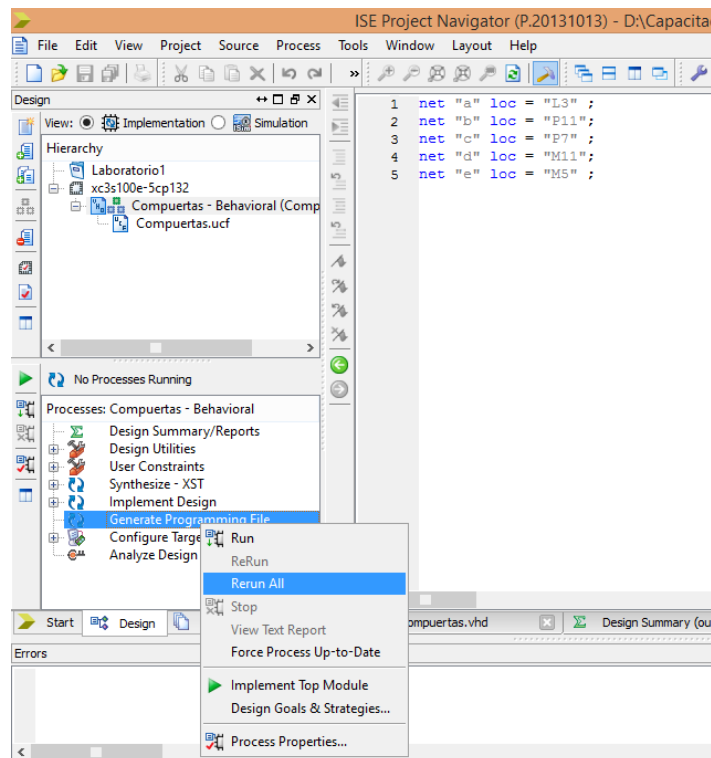
4. Dar doble click en el archivo Compuertas.ucf y aparecerá un editor de texto en blanco en la ventana principal




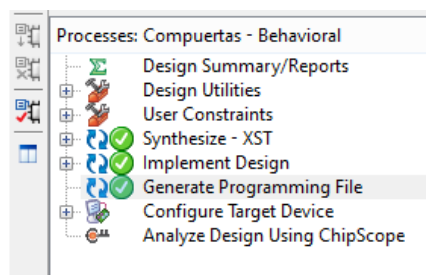
5. Escribir lo siguiente en el archivo .ucf

```
net "a" loc = "L3" ;
net "b" loc = "P11";
net "c" loc = "P7" ;
net "d" loc = "M11";
net "e" loc = "M5" ;
```

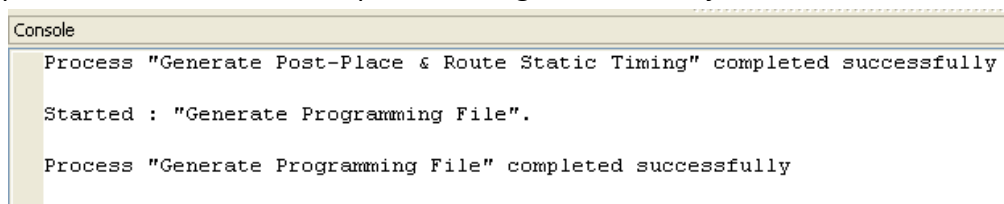
6. Seleccionar en la ventana Sources el módulo principal Compuertas.vhd, en la ventana Processes elegir Generate Programming File y doble clic en la opción Rerun All.



7. Verificar que aparezca el símbolo  en los procesos que seleccionamos en el paso anterior de la ventana Processes.

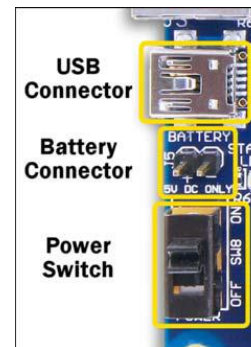



8. Verificar que en la ventana Console aparezca el siguiente mensaje:

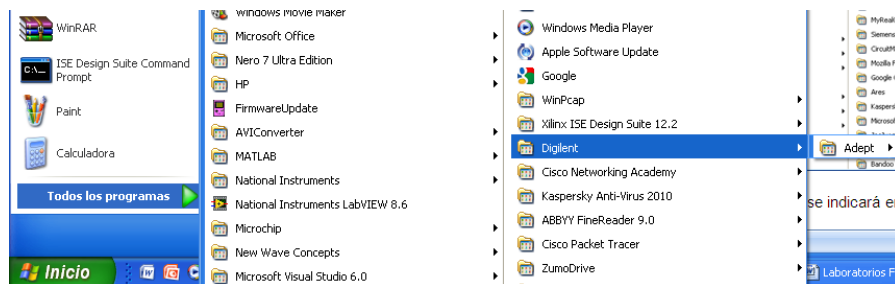


Programación del chip de FPGA

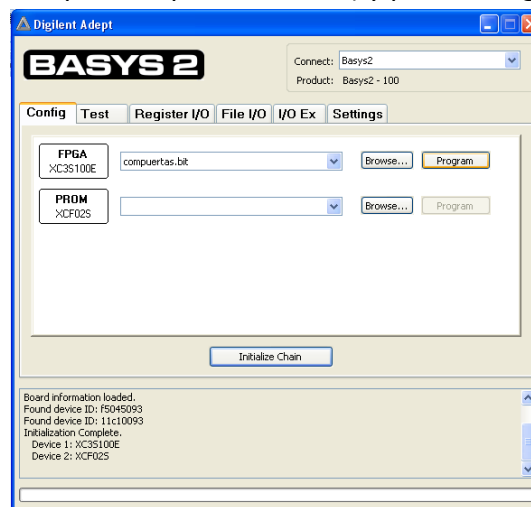
1. Conectar el kit de entrenamiento Basys2 mediante el cable USB a la computadora, elegir las opciones para que la computadora lo instale automáticamente.



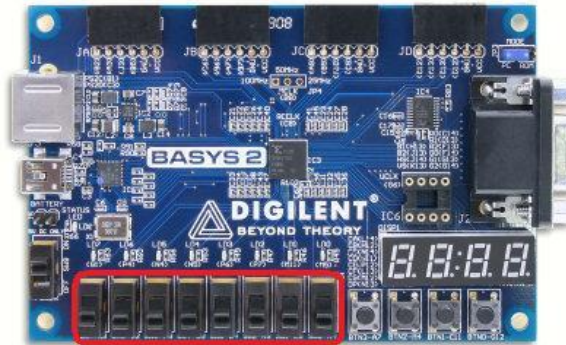
2. Buscar en el Inicio → todos los programas → Digilent →  Adept



3. Si el kit está conectado correctamente se indicará en la siguiente ventana.
4. En la pestaña Browse buscar el archivo Compuertas.bit en el directorio que guardamos a la hora de crear el proyecto (ver en la parte superior de ISE®) y pulsar Program.



5. Cambiar de posición los switches SW1 y SW0 (las variables de entrada a y b) y comprobar que los LED's actúan correctamente (LD2 como OR, LD1 como AND, y LD0 como XOR).
6. Estas líneas significan la ubicación de las entradas y las salidas en la tarjeta Basys2. Las entradas son los switches...



7. ... y las salidas las visualizaremos con los leds.



7. Actividades propuestas

Práctica

Llene la tabla de verdad según las ecuaciones dadas y realice la descripción de las combinaciones lógicas en un programa de VHDL. Verifique que la sintaxis del programa es correcta utilizando los pasos descritos anteriormente incluyendo su simulación.

Entradas			Salidas	
x	y	z	$a = \overline{(x + y)} * z$	$b = x \oplus y \oplus z$
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Actividades de aprendizaje

1. Introduzca una nueva variable (x), de tal manera que las salidas sean una and de tres entradas, una or de tres entradas, ¿qué sucede con los valores de salidas?
2. En una compuerta and de 2 entradas; si en una de sus entradas recibe un 0 y en la otra un 1, ¿Cuál es su salida?
3. Si una compuerta Nand recibe las mismas señales de entrada de la pregunta anterior, ¿Cuál es su salida?
4. Explique por qué es útil simular un sistema o un circuito. ¿Considera importante simular un circuito electrónico descrito con programación?
5. Cuando hacemos la simulación de un circuito, ¿es necesario especificar la magnitud del retardo entre las señales de entrada y las señales de salida? Justifique su respuesta.
6. ¿Qué sucede si cambia los tiempos de temporización? Cambiar #100 a #50 en el programa que hizo en la práctica.

8. Reporte de Laboratorio

El reporte se entregará en un archivo comprimido y deberá incluir:

- Tabla de verdad de las variables.
- Descripción de las ecuaciones con compuertas lógicas.
- Descripción de las ecuaciones en VHDL.
- Verificación del programa en ISE. Utilice las capturas de pantalla (tecla: PRTSC) para explicar el procedimiento.
- Resultados de la simulación.