

# Ciclo de desarrollo con Vivado

## Introducción

En esta práctica se utiliza el IDE Vivado para crear un sistema básico a partir de una descripción HDL para la placa Arty Z7. Se simulará, sintetizará e implementará el sistema con las opciones por defecto; y finalmente se generará el archivo de configuración y se configurará la FPGA para verificar el funcionamiento del sistema.

## Objetivos

- Crear un proyecto en Vivado utilizando un código fuente HDL para un dispositivo ZYNQ en la placa Arty Z7-10
- Usar un archivo de restricciones predefinido(Xilinx Design Constraint (XDC) file) para restringir las ubicaciones de los pines de I/O
- Simular el sistema usando el simulador del IDE Vivado
- Sintetizar e Implementar el sistema
- Generar el archivo de configuración (bitstream)
- Configurar el dispositivo ZYNQ usando el bitstream y verificar la funcionalidad del sistema

## Procedimiento

### Descripción del sistema

El sistema consiste en algunas entradas que realizan ciertas funciones lógicas y sus resultados se muestran en LEDs (**Figura 1**).

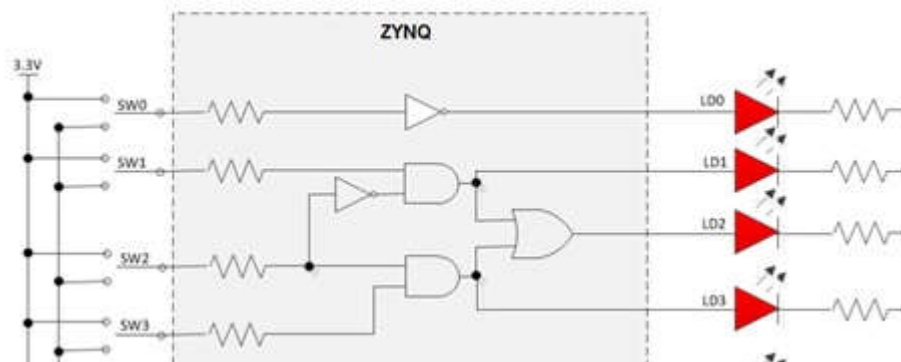
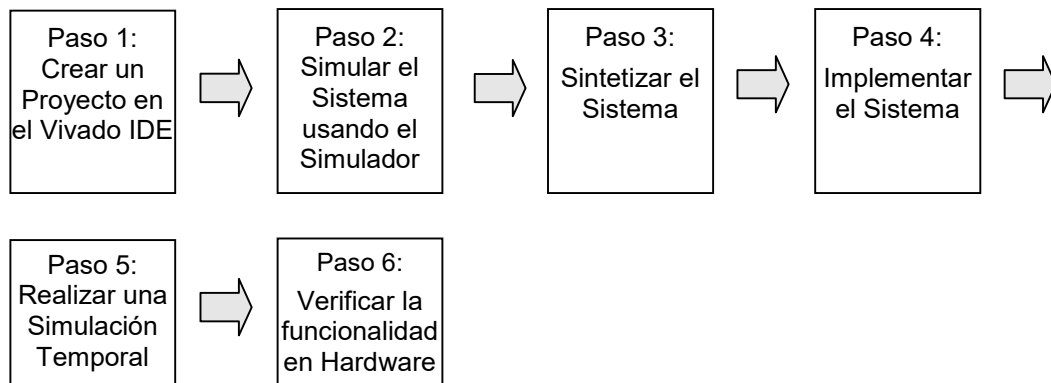


Figura 1. Sistema a Implementar

## Hoja de Ruta de la Práctica



## Crear un Proyecto en Vivado

### Paso 1

**1-1. Iniciar el IDE Vivado y crear un proyecto para la placa Arty Z7-10 usando el HDL Verilog. Usar los archivos lab1.vhd y lab1\_ArtyZ7\_10.xdc.**

**1-1-1.** Abrir el IDE Vivado: **Start > All Programs >Xilinx Design Tools>Vivado2018.1>Vivado2018.1**

**1-1-2.** Descomprimir los archivos de descripción de la placa Arty Z7-10. Escribir en la consola TCL:

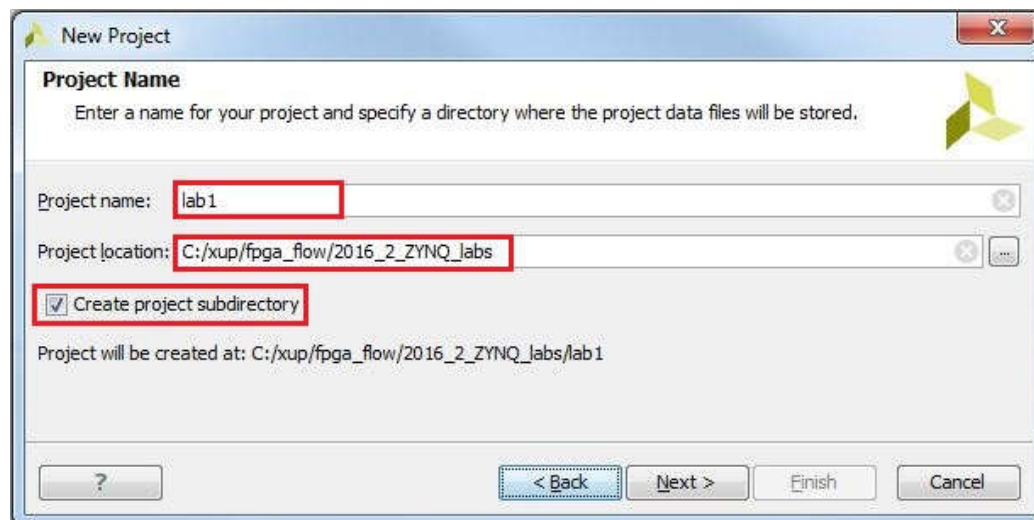
```
set_param board.repoPaths [list "C:/Xilinx/Vivado/DigilentBoards/board_files"]
```

(o la ubicación donde se descomprimieron los archivos de descripción)

**1-1-3.** Presionar **Create Project** para iniciar el asistente. Aparecerá el dialogo *Create A New Vivado Project* dialog box. Presionar **Next**.

**1-1-4.** Presionar el botón Browse del campo *Project location* del formulario **New Project**, navegar a la ubicación de la practica (p.ej. **c:\Facultad\SistemasDigitales\Practicas\**), y presionar **Select**.

**1-1-5.** Escribir **Practica01** en el campo *Project name*. Verificar que el recuadro *Create Project Subdirectory* este seleccionado. Presionar **Next**.



**Figura 2. Formulario para el nombre del proyecto y su ubicación**

**1-1-6.** Seleccionar la opción **RTL Project** en el formulario *Project Type* y presionar **Next**.

**1-1-7.** Seleccionar la pestaña **Boards**

**1-1-7.1.** En el cuadro *Vendor* seleccionar: Digilentinc.com

**1-1-7.2.** Seleccionar la placa Arty Z7-10, presionar **Next**, y después **Finish**

- 1-1-8.** En el menú **Flow Navigator**, en la sección *Project Manager*, seleccionar *Add Sources*. En el cuadro de dialogo seleccionar *Add or create design sources*, presionar **Next**. En el cuadro de dialogo, presionar el botón **Add Files...** Navegar hasta el archivo *lab1.vhd* file, presionar **OK**.

Verificar que este seleccionado **Copy sources into project**. Presionar **Finish**.

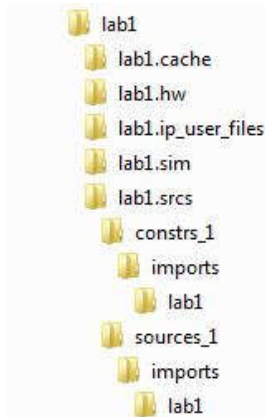
- 1-1-9.** En el menú **Flow Navigator**, en la sección *Project Manager*, seleccionar *Add Sources*. En el cuadro de dialogo seleccionar *Add or create constraints*, presionar **Next**. En el cuadro de dialogo, presionar el botón **Add Files...** Navegar hasta el archivo *lab1\_ArtyZ7\_10.xdc*, presionar **OK**.

Verificar que este seleccionado **Copy sources into project**. Presionar **Finish**

- 1-1-10.** Este archivo de restricciones asigna los pines de I/O de la FPGA a los botones y leds de la placa. Esta información se puede obtener de los diagramas esquemáticos de la placa o del manual.

- 1-1-11.** Estructura del proyecto:

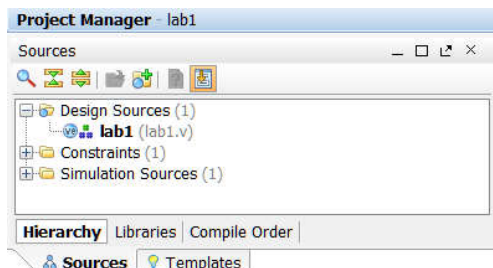
Usando el Explorador de Archivos, analizar el directorio del proyecto. Allí se crearon los directorios *.cache* y *.srcs*, al igual que el archivo *.xpr*. El directorio *.cache* es la ubicación física de la base de datos del proyecto. A su vez, se crean dos directorios, *constrs\_1* and *sources\_1* dentro del directorio *.srcs*; dentro de ellos están los archivos *.xdc* (restricciones) y *.vhd* (código fuente HDL).



**Figura 3. Estructura de archivos del proyecto**

## **1-2. Abrir el código fuente lab1.vhd y analizar su contenido.**

- 1-2-1.** En el panel *Sources* seleccionar y presionar dos veces el archivo **lab1.vhd** para abrir el archivo en modo texto.



**Figura 4. Abriendo el archivo de código fuente**

**1-2-2.** La línea 5 define el comienzo de la entidad módulo (con la palabra reservada **entity**) y la línea 9 define el final de la entidad (con la palabra reservada **end**).

**1-2-3.** Las líneas 6 y 7 definen los puertos de entrada y de salida, mientras que las líneas 11 a 17 definen la lógica (a través de la arquitectura).

### 1-3. Abrir el archivo .xdc y analizar el contenido.

**1-3-1.** En la ventana *Sources*, abrir la carpeta *Constraints* y presionar dos veces el archivo **lab1\_ArtyZ7\_10.xdc** para abrir el archivo en modo texto.



Figure 5. Abriendo el archivo de restricciones

**1-3-2.** Las líneas 5 a 8 definen las ubicaciones de los pines conectados a los botones de entrada [3:0] y las líneas 13 a 16 definen las ubicaciones de los pines conectados a los LEDs [3:0].

### 1-4. Realizar un análisis RTL del código fuente.

**1-4-1.** En el *Flow Navigator*, en la sección *RTL Analysis* expandir *Open Elaborated Design* y presionar en **Schematic**. Dar **OK** en el cuadro de diálogo.

El sistema es elaborado, y se genera una vista lógica.

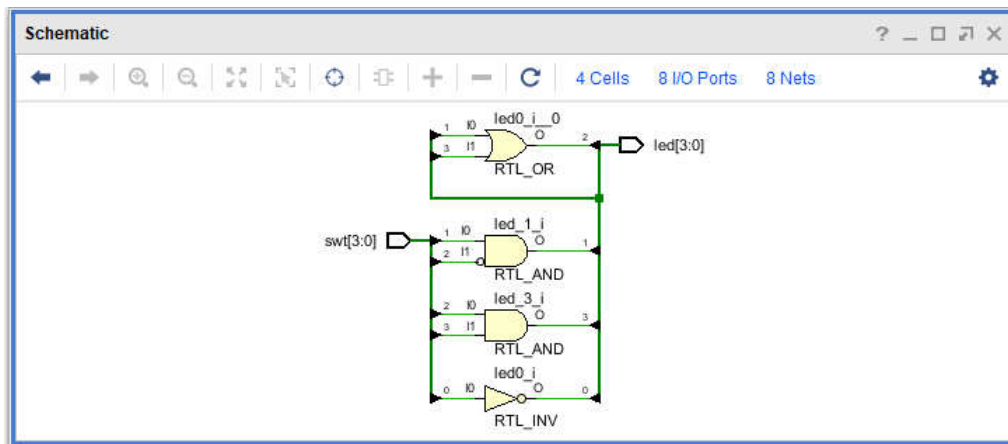


Figura 6. Una vista lógica del sistema

## Simular el sistema usando el simulador de Vivado

## Paso 2

### 2-1. Agregado del archivo de pruebas lab1\_tb.v .

- 2-1-1. En el menú **Flow Navigator**, en la sección *Project Manager*, seleccionar *Add Sources*. En el cuadro de dialogo seleccionar *Add or create Simulation Sources*, presionar **Next**. En el cuadro de dialogo, presionar el botón **Add Files...** Navegar hasta el archivo *lab1\_tb.vhd*, presionar **OK**.

Verificar que este seleccionado *Copy sources into project*. Presionar **Finish**.

- 2-1-2. Seleccionar la pestaña *Sources* y expandir *Simulation Sources* .

El archivo **lab1\_tb.v** esta agregado en el grupo *Simulation Sources* group, y **lab1.vhd** está ubicado en la jerarquía como instancia dut (device under test).

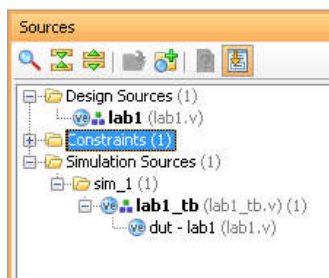


Figura 7. Jerarquía de archivos en la simulación

- 2-1-3. Usando el explorador de Archivos verificar que se creó el directorio **sim\_1** al mismo nivel que **constr\_1** y **sources\_1** dentro de **lab1.srscs** y que hay una copia de **lab1\_tb.vhd** en **lab1.srscs->sim\_1 ->imports ->lab1**.

- 2-1-4. Presionar dos veces en **lab1\_tb** en el panel *Sources* para ver su contenido

```

1 `timescale 1ns / 1ps
2 ///////////////////////////////////////////////////////////////////
3 // Module Name: lab1_tb
4 ///////////////////////////////////////////////////////////////////
5 module lab1_tb(
6
7 );
8
9 reg [7:0] switches;
10 wire [7:0] leds;
11 reg [7:0] e_led;
12
13 integer i;
14
15 lab1 dut(.led(leds),.swt(switches));
16
17 function [7:0] expected_led;
18 input [7:0] swt;
19 begin
20     expected_led[0] = ~swt[0];
21     expected_led[1] = swt[1] & ~swt[2];
22     expected_led[3] = swt[2] & swt[3];
23     expected_led[2] = expected_led[1] | expected_led[3];
24     expected_led[7:4] = swt[7:4];
25 end
26 endfunction
27
28 initial
29 begin
30     for (i=0; i < 255; i=i+2)
31     begin
32         #50 switches=i;
33         #10 e_led = expected_led(switches);
34         if (leds == e_led)
35             $display("LED output matched at", $time);
36         else
37             $display("LED output mis-matched at ", $time, ": expected: %b, actual: %b", e_led, leds);
38         end
39     end
40 endmodule
41
42

```

Figura 8. Archivo de pruebas (Verilog)

El archivo de pruebas define el paso y la resolución de simulación en la línea 1. La definición del módulo de prueba comienza en la línea 5. En la línea 15 se instancia el DUT (device/module under test). Las líneas 17 a 26 definen una función que genera los valores esperados de salida de acuerdo a la entrada. Las líneas 28 a 39 generan las señales de entrada y comparan la salida esperada con lo que genera el DUT. La función \$display imprime los resultados de las comparaciones durante la simulación.

## 2-2. Simular el sistema por 200 ns usando el simulador de Vivado.

2-2-1. En el menú **Flow Navigator**, en la sección *Project Manager*, seleccionar *Settings*.

Aparecerá el formulario **Project Settings**; seleccionar **Simulation** para ver sus propiedades.

2-2-2. Seleccionar la pestaña **Simulation**, y fijar la propiedad **Simulation Run Time** en 200 ns. Presionar **OK**.

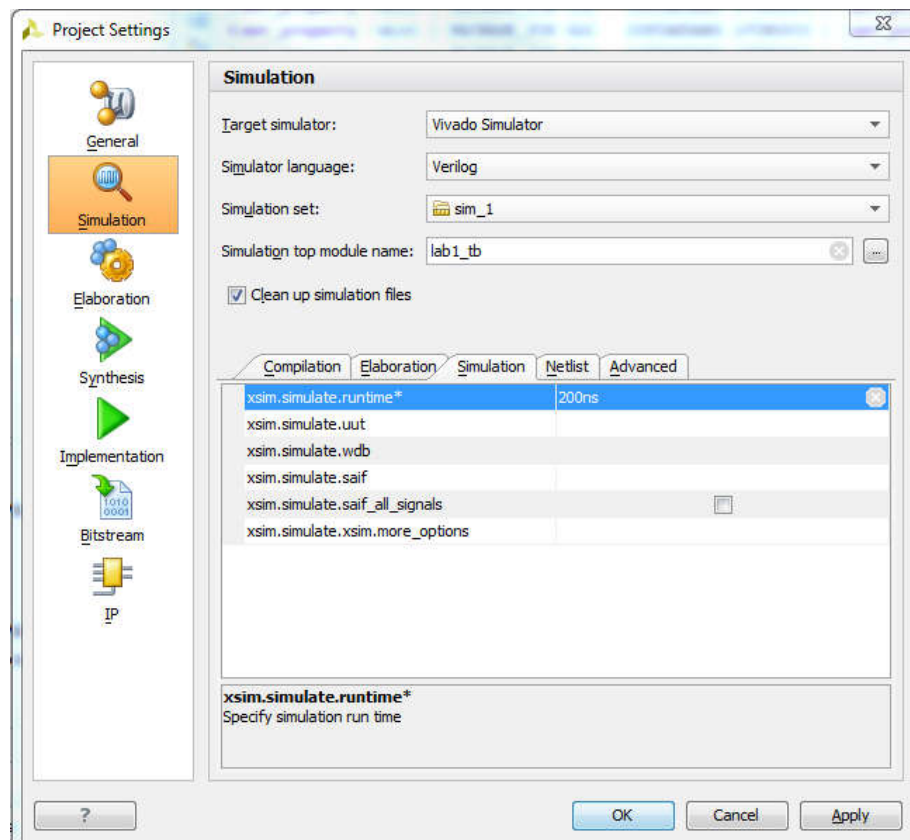
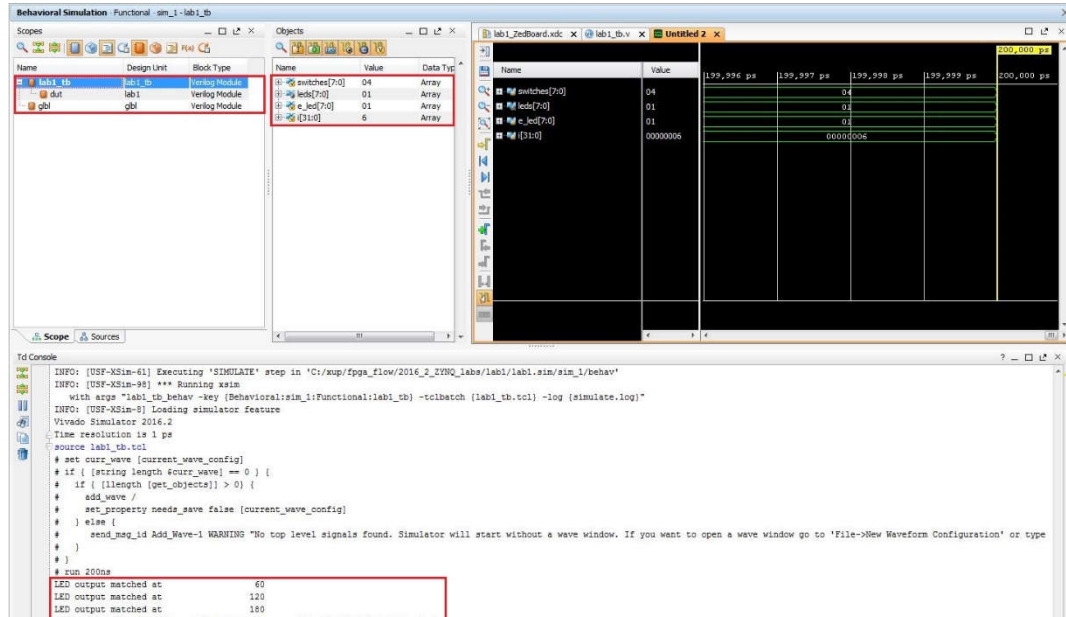


Figura 9. Estableciendo el tiempo de simulación

2-2-3. En el menú **Flow Navigator**, en la sección *Simulation* seleccionar **Run Simulation -> Run Behavioral Simulation**.

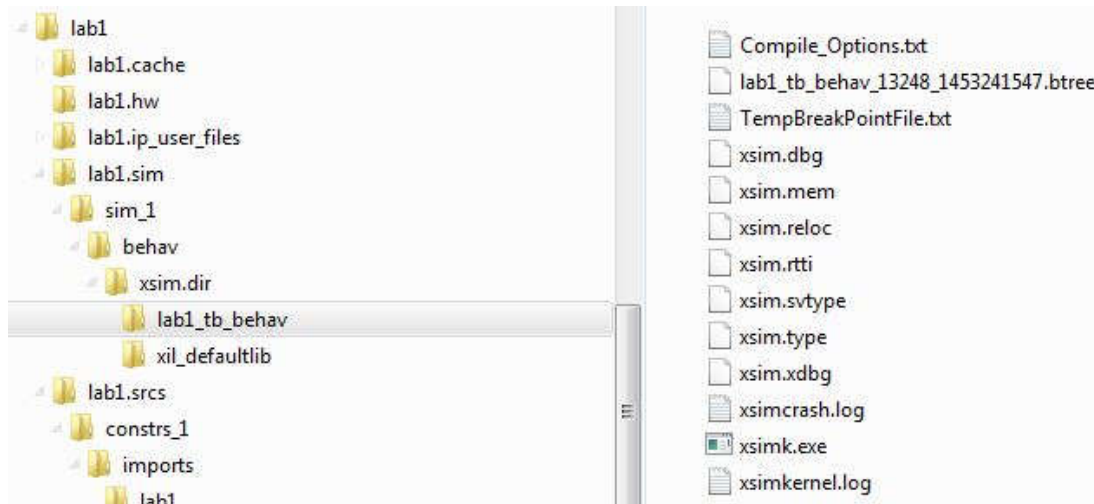
El archivo de pruebas y los fuentes se compilan y se ejecuta el simulador de Vivado. Se ve una salida similar a esta:



**Figura 10. Salida del simulador**

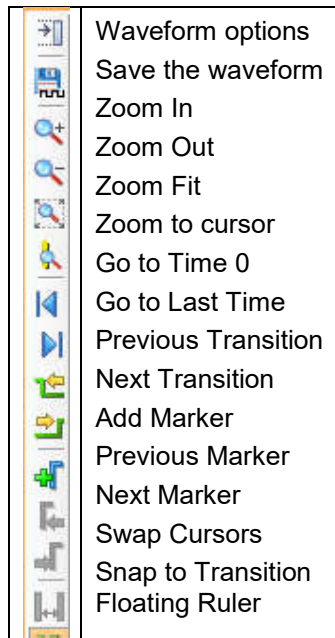
En la salida del simulador hay 4 vistas principales: (i) *Scope*, donde se muestra la jerarquía de la simulación, (ii) *Objects*, donde se muestran las señales del top-level, (iii) la ventana de formas de onda, y (iv) la *Tcl Console*, donde se ven los mensajes de la simulación. Notar que como el archivo de pruebas tiene autochequeo, los resultados se muestran a medida que se ejecuta la simulación.

En la estructura de archivos, el directorio **lab1.sim** se crea debajo del directorio **lab1**, junto con otros subdirectorios.



**Figura 11. Estructura de directorios después de la simulación funcional**

En la ventana de formas de onda se ven distintos botones que se utilizan de acuerdo a la siguiente tabla:

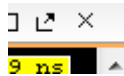


**Tabla 1: Botones para navegar en la ventana de formas de onda**

**2-2-4.** Presionar el botón *Zoom Fit* () para ver la forma de onda entera.

Verificar los cambios en la salida a medida que cambia la entrada.

También se puede poner la ventana de simulación flotante presionando el botón *Float* en la parte superior derecha. Esto permite una ventana más ancha para ver las formas de onda. Para volver a poner la ventana en la interface gráfica, presionar el botón *Dock Window*.



**Figure 12. botón Float**



**Figure13. botón Dock Window**

## **2-3. Cambiar el formato de las señales.**

Seleccionar **i[31:0]** en la ventana de formas de onda, presionar el botón derecho, seleccionar *Radix*, y después seleccionar *Unsigned Decimal* para ver el índice del ciclo for in formato *integer*. De la misma manera cambiar el formato de **switches[3:0]** a *Hexadecimal*. Cambiar los formatos de **leds[3:0]** y **e\_led[3:0]** a *binary* para ver cada bit de salida.

## **2-4. Agregar señales para monitorear señales de más bajo nivel y continuar la simulación por 500 ns.**

**2-4-1.** En la ventana *Scopes* expandir la instancia **lab1\_tb** y seleccionar la instancia **dut**.

Las señales **swt[3:0]** y **led[3:0]** aparecerán en la ventana *Objects*.



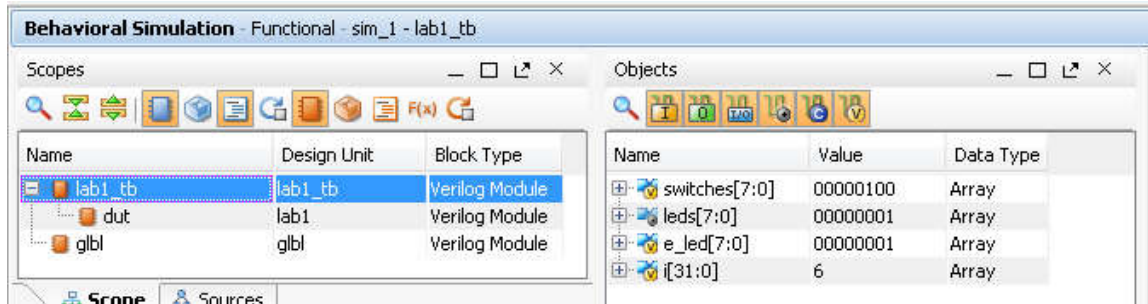


Figure 14. Selección de señales de más bajo nivel

**2-4-2.** Seleccionar **swt[3:0]** y **led[3:0]** y arrastrarlas a la ventana de formas de onda para monitorear esas señales de más bajo nivel.

**2-4-3.** En la barra superior del simulador, escribir 500 en la ventana *simulation run time*, presionar en el botón desplegable de las unidades y seleccionar *ns* ( 500 ns ) dado que queremos continuar la simulación por 500 ns (700 ns en total), y presionar el botón de avance ( ).

La simulación continuara por otros 500 ns.

**2-4-4.** Presionar en el botón *Zoom Fit* y verificar la salida (poner la ventana de formas de onda flotante para aumentar su tamaño).

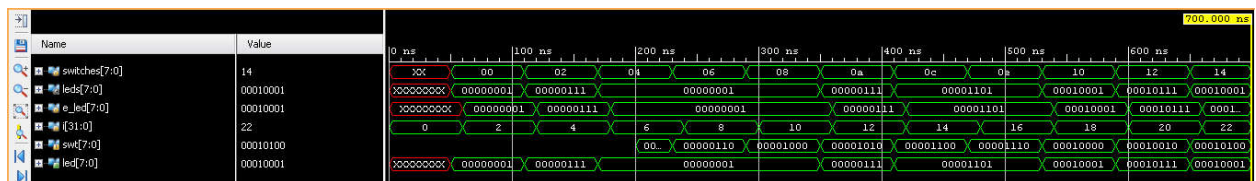


Figura 15. Simulación ejecutada por otros 500 ns

En la ventana *Tcl Console* se observa como la salida se muestra a medida que el archivo de pruebas usa la función `$display`.

```
INFO: [USF-XSim-96] XSim completed. Design snapshot 'lab1_tb_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 200ns
launch_simulation: Time (s): cpu = 00:00:02 ; elapsed = 00:00:08 . Memory (MB): peak = 1159.863 ;
run 500 ns
LED output matched at          240
LED output matched at          300
LED output matched at          360
LED output matched at          420
LED output matched at          480
LED output matched at          540
LED output matched at          600
LED output matched at          660
```

Figura 16. Ventana Tcl Console luego de ejecutar la simulación por otros 500 ns

**2-4-5.** Cerrar el simulador mediante **File ->Close Simulation**.

**2-4-6.** Presionar **OK** y luego **Discard** para cerrar la simulación sin guardar los resultados.

## Sintetizar el sistema

## Paso 3


### 3-1. Sintetizar el sistema con la herramienta de síntesis y analizar el resumen de proyecto (Project Summary).

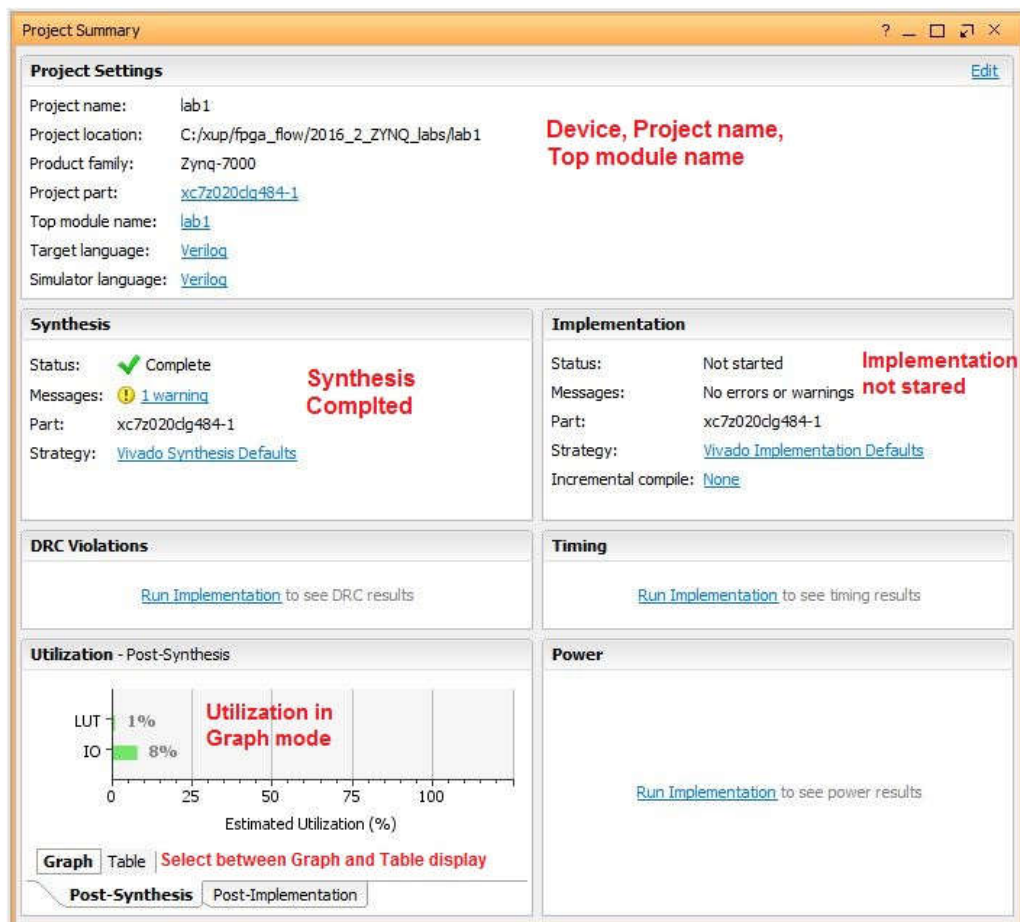
#### 3-1-1. En el menú **Flow Navigator**, en la sección *Synthesis*, seleccionar **Run Synthesis**. Presionar **OK**.

El proceso de síntesis se realizara en el archivo lab1.v (y sus sub-archivos dentro de la jerarquía, si existen). Cuando se completa el proceso de síntesis, aparece el cuadro de dialogo *Synthesis Completed* dialog con tres opciones.

#### 3-1-2. Seleccionar la opción *Open Synthesized Design* y presionar **OK**, ya que queremos ver la salida del proceso de síntesis antes de pasar a la etapa de implementación.

#### 3-1-3. Seleccionar la pestaña **Project Summary** y ver las distintas secciones.

Si no se ve la pestaña *Project Summary*, seleccionar **Layout -> Default Layout**, o presionar el botón de **Project Summary** (  ). Maximizar y poner flotante la pestaña para ver toda la información

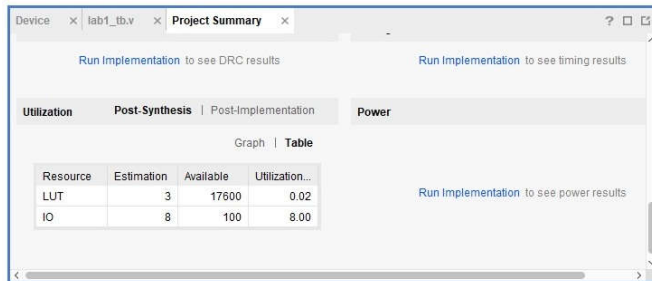


**Figura 17. Pestaña Project Summary**

Presionando en algunos enlaces se puede ver que información proveen, y algunos permiten cambiar las opciones de síntesis.

**3-1-4.** En la sección **Utilization**, donde se encuentra el gráfico, presionar en el botón **Table**.

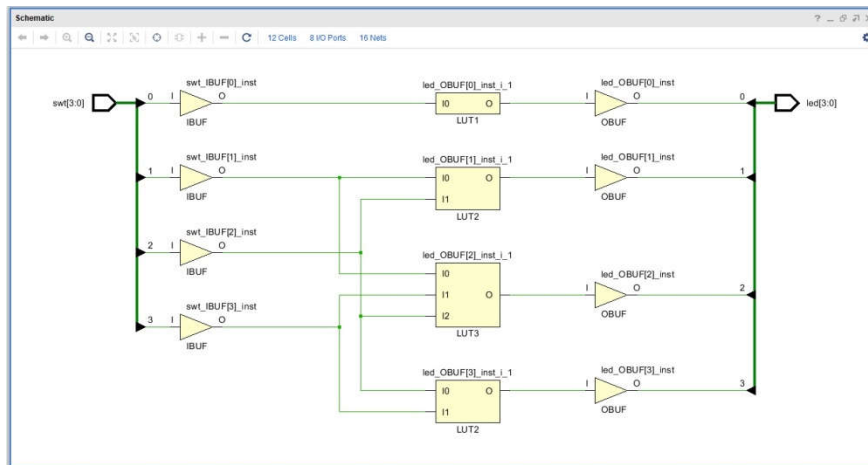
La estimación es que se utilizarán tres LUTs y 8 IOs (4 de entrada y 4 de salida).



Resource	Estimation	Available	Utilization...
LUT	3	17500	0.02
IO	8	100	8.00

**Figura 18.** Estimación de recursos para la FPGA de la placa Arty Z7

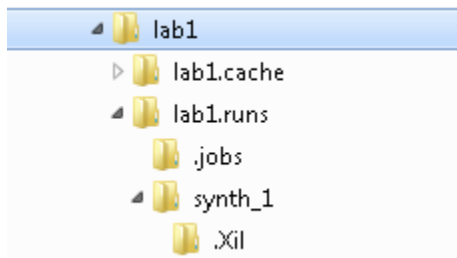
**3-1-5.** En el *Flow Navigator*, bajo *Synthesis* (expandir *Synthesized Design* si es necesario), presionar en **Schematic** para ver el sistema sintetizado en una vista esquemática.



**Figura 19.** Diagrama esquemático del sistema sintetizado

Notar que los IBUFs y OBUFs se instancian en forma automática (se agregan) al sistema, dado que las entradas y salidas son acondicionadas (buffered). Las compuertas lógicas se implementan con LUTs (las compuertas de 1 entrada se muestra como LUT1, las de 2 entradas se muestran como LUT2, y las compuertas de 3 entradas como LUT3). Las 4 compuertas del análisis RTL se mapean a 4 LUTs en la salida del proceso de síntesis.

Mediante el explorador de Archivos, verificar que el directorio **lab1.runs** fue creado dentro del directorio **lab1**. Dentro del directorio **runs**, se creó el directorio **synth\_1**, el cual tiene distintos archivos relacionados con el proceso de síntesis.



**Figura 20.** Estructura de directorios después del proceso de síntesis

## Implementación del sistema

## Paso 4

### 4-1. Implementar el sistema con las opciones por defecto y analizar la salida del resumen de proyecto (Project Summary output).

#### 4-1-1. En el menú **Flow Navigator**, en la sección *Implementation*, seleccionar **Run Implementation**. Presionar **OK**.

Se ejecutara el proceso de implementación en el sistema sintetizado. Cuando se complete el proceso, aparecerá el cuadro de dialogo *Implementation Completed* con tres opciones.

#### 4-1-2. Seleccionar **Open implemented design** y presionar **OK**, ya que queremos ver el sistema implementado en la pestaña de vista de dispositivo.

#### 4-1-3. Si aparece un cuadro de dialog, presionar **Yes**, para cerrar el sistema sintetizado.

Se abrirá la vista del sistema implementado.

#### 4-1-4. En el panel *Netlist*, seleccionar una de las redes (p.ej. led\_OBUF[2]). Ver que se muestra la red in la región de reloj X1Y1 en la pestaña de vista de dispositivo (puede ser necesario hacer zoom para verla).

#### 4-1-5. Si no está seleccionado, presionar el botón *Routing Resources* para ver el ruteo.

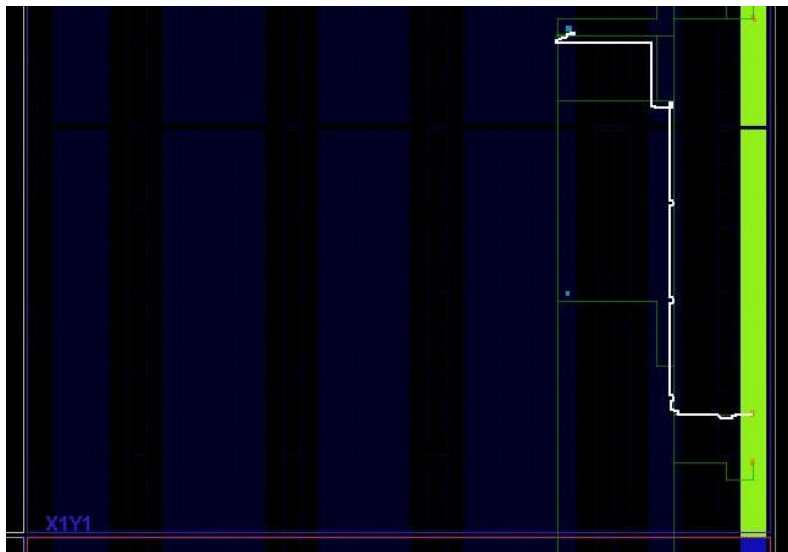


Figura 21. Vista del ruteo realizado

#### 4-1-6. Cerrar la vista del sistema implementado mediante **File -> Close Implemented Design**, y seleccionar la pestaña **Project Summary** (puede ser necesario cambiar a la distribución por defecto) y ver los resultados.

Presionar el botón **Post-Implementation** en la parte *Utilization*. Luego presionar el botón **Table**

Se observa que la utilización de recursos es tres LUTs y 8 IOs. También en la parte de *Timing* se ver que no hay definidas restricciones de temporización, ya que el circuito es puramente combinacional.

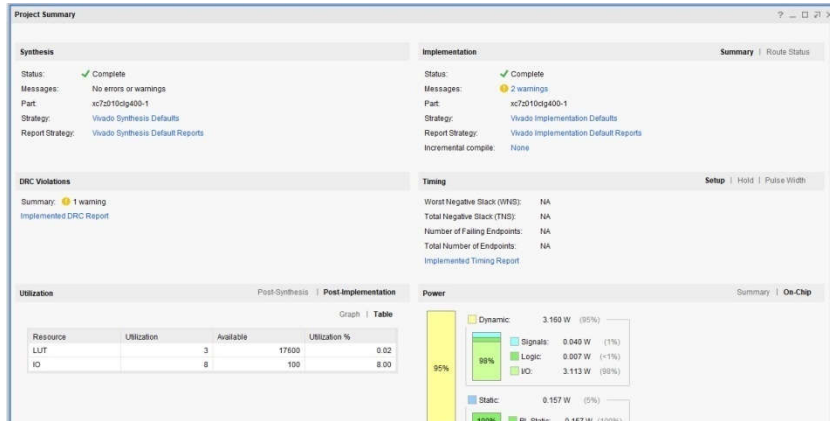


Figura 22. Resumen de Implementación

Usando el explorador de Archivos, verificar que el directorio **impl\_1** se creó en el mismo nivel que **synth\_1** dentro del directorio **lab1.runs**. El directorio **impl\_1** contiene distintos archivos de implementación, incluidos los reportes.

- 4-1-7.** En el IDE Vivado, seleccionar la pestaña **Reports** en el panel de abajo (si no está visible, presionar *Window* en la barra de menú, y seleccionar **Reports**), en la parte de *Implementation*, dentro de **impl\_1**, buscar la sección *Place Design* y presionar dos veces en **impl\_1\_place\_report\_utilization\_0**. Se abrirá el reporte mostrando la utilización de recursos. Dado que el circuito es puramente combinacional, no hay uso de registros.

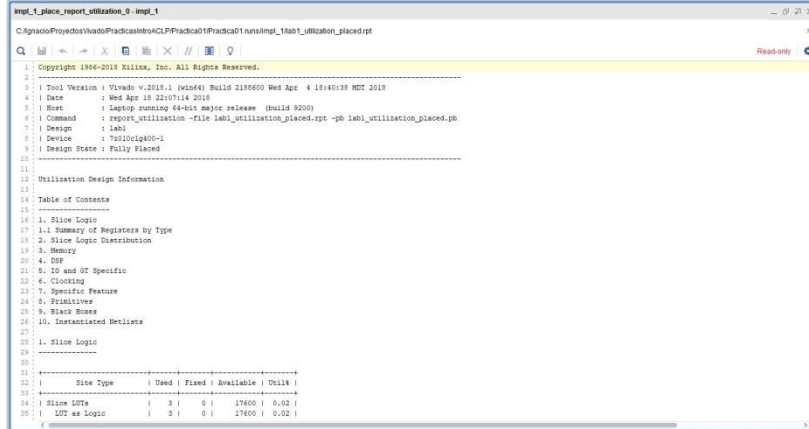


Figure 23. Reporte de utilización de recursos

## Simulación Temporal

## Paso 5

### 5-1. Run a timing simulation.

- 5-1-1.** En el menú Flow Navigator, en la sección Simulation, seleccionar Run Simulation -> Run Post-Implementation Timing Simulation.


Se iniciara el simulador de Vivado usando el sistema implementado y **lab1\_tb** como modulo principal.

Usando el Explorador de Archivos, verificar que se creó el directorio **timing->xsim** dentro de **lab1.sim-> sim\_1-> impl**. Este directorio contiene los archivos necesarios para la simulación temporal.

- 5-1-2.** Presionar el botón **Zoom Fit** para ver las formas de onda en el rango 0 a 200 ns.

- 5-1-3.** Presionar el botón derecho en la marca de 50 ns (hacerlo por debajo de las formas de onda, donde la entrada se fija en 0000b) y seleccionar **Markers-> Add Marker**.

- 5-1-4.** De la misma manera, agregar una marca alrededor de los 58ns, donde cambia la salida (**leds**).

- 5-1-5.** También se puede agregar un marcador mediante el botón Add Marker (  ). Presionar el botón **Add Marker** y presionar el botón izquierdo alrededor de los 60 ns, donde cambia **e\_led**.

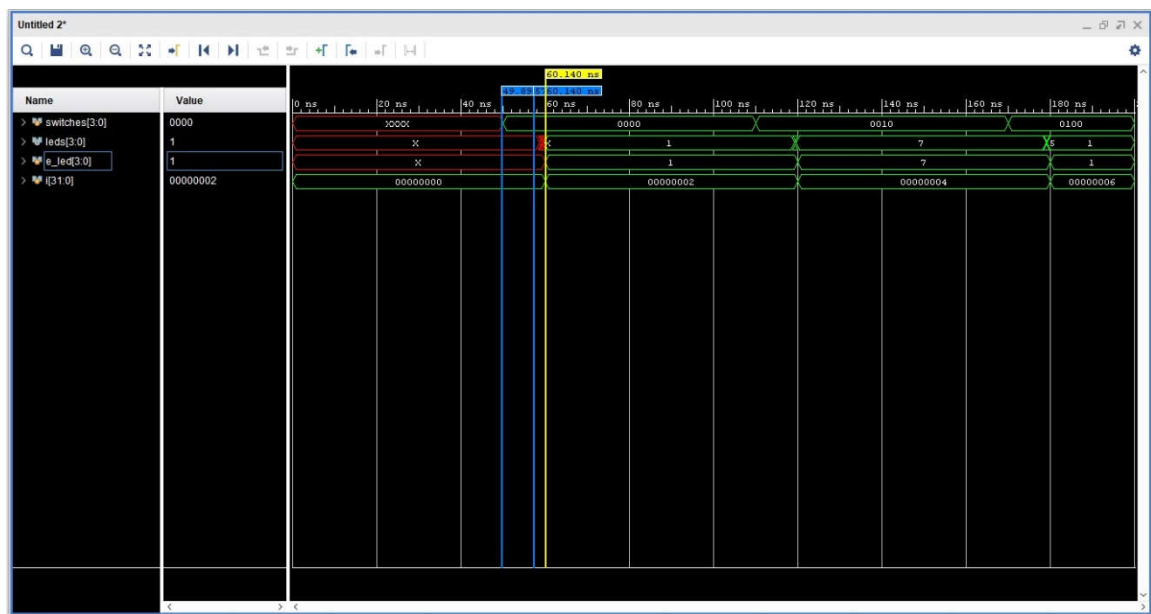


Figura 24. Agregado de marcadores

Notar que la salida leds[3:0] comienza a cambiar su estado a los 58ns, y la salida del sistema e\_leds[3:0] cambia a los 59.7ns, lo que permite evaluar tiempos de propagación

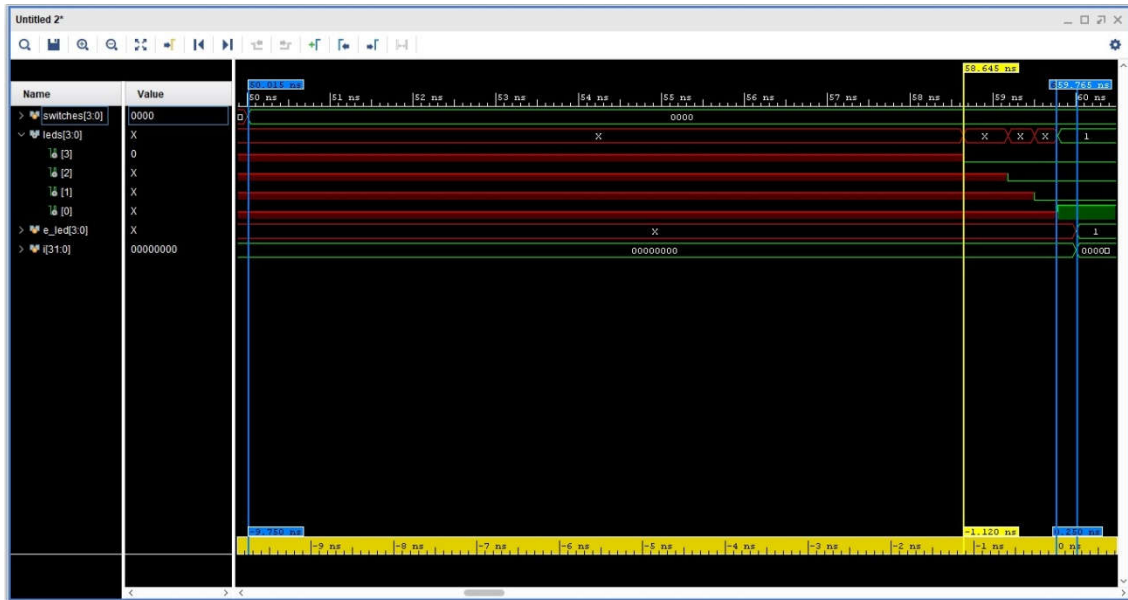


Figura 25. Agregado de marcadores (detalle)

- 5-1-6. Cerrar el simulador mediante el menú **Flow Navigator**, en la sección *Simulation*, presionar el botón derecho y seleccionar **Close Simulation** sin guardar cambios.

## Generar el archivo de configuración (BitStream)

### Paso 6

#### 6-1. Conectar la placa y encenderla, generar el archivo de configuracion, abrir una sesion de hardware y configurar la FPGA.

##### 6-1-1. Verificar que el cable Micro-USB esté conectado al conector PROG/UART al lado del conector Ethernet.

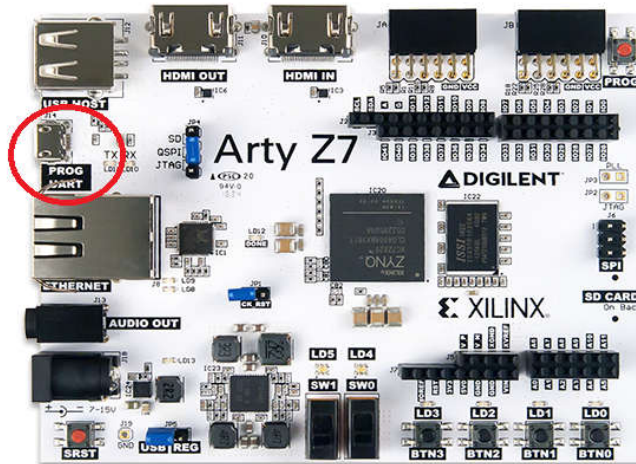


Figura 26. Conector de programación de la placa

##### 6-1-2. En el menú **Flow Navigator**, en la sección *Program and Debug*, seleccionar **Generate Bitstream**. Presionar **OK**.

Se ejecutara el proceso de generación del archivo de configuración en el sistema implementado. Cuando el proceso se complete, se abrirá el cuadro de dialogo *Bitstream Generation Completed* con 3 opciones (este proceso generará el archivo **lab1.bit** en el directorio **impl\_1** dentro de **lab1.runs**).

##### 6-1-3. Seleccionar la opción *Open Hardware Manager* y presionar **OK**.

Se abrirá la ventana Hardware Manager indicando el estado “unconnected”.

##### 6-1-4. Presionar el link **Open target**.

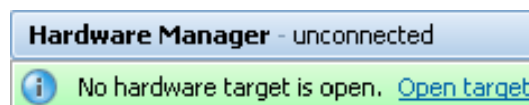
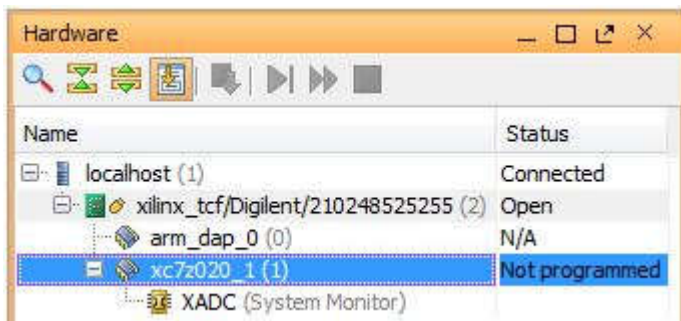


Figura 27. Hardware Manager

##### 6-1-5. Del menú desplegable, seleccionar **Auto Connect**.

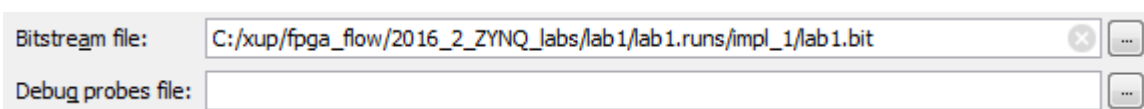
El estado de la sesión de hardware cambiara de Unconnected al nombre del servidor (localhost) y el dispositivo quedara seleccionado. Asimismo el estado será “Not programmed”.





**Figura 28. Sesión de hardware**

Seleccionar el dispositivo y verificar que el archivo lab1.bit este seleccionado como archivo de programación en la pestaña *General* del panel **Hardware Device Properties**.



**Figura 29. Archivo de configuración**

**6-1-6.** Presionar en el link *Program device*.

Otra opción es presionar el botón derecho en el dispositivo y seleccionar *Program Device...*



**Figura 29. Link para programar la FPGA**

**6-1-7.** Presionar el botón **Program** para configurar la FPGA.

El led DONE se encenderá al finalizar la configuración; los otros leds cambiarán de acuerdo a la posición de los interruptores.

**6-1-8.** Verificar la funcionalidad del sistema cambiando los interruptores y observando los leds de salida de acuerdo al diagrama lógico.

**6-1-9.** Para finalizar, desconectar la placa.

**6-1-10.** Cerrar la sesión de hardware mediante **File ->Close Hardware Manager**.

**6-1-11.** Presionar **OK** para finalizar la sesión.

**6-1-12.** Cerrar el entorno **Vivado** mediante **File -> Exit** y presionar **OK**.