

# **Circuitos lógicos programables**

## **Práctica 4**

### **Restricciones**

## Introducción

En esta práctica se utilizará el sistema basado en UART de prácticas anteriores. Se iniciará el proyecto con la planificación de los pines de I/O; esta restricción se exportará al código RTL y se agregarán las restricciones de temporización. Luego se realizará un análisis de temporización y se implementará el proyecto en el hardware.

## Objetivos

Después de completar esta práctica será capaz de:

- Crear un proyecto del tipo planificación de I/O
- Realizar el ingreso de la ubicación de pines y su estándar mediante la vista de dispositivo (Device view), la pestaña de pines del encapsulado (Package Pins tab), y mediante comandos Tcl
- Crear retardos de Periodo, de Entrada y de Salida
- Realizar un análisis de temporización

## Procedimiento

Esta práctica está separada en pasos que consisten en sentencias generales que proveen información sobre las instrucciones detalladas que le siguen. Siga estas instrucciones detalladas para avanzar dentro de esta práctica.

Esta práctica está compuesta por 5 pasos principales: crear un proyecto de planificación de I/O, asignar pines y código fuente, sintetizar y agregar restricciones temporales, implementar y realizar un análisis temporal y, verificar el sistema en el hardware.

## Descripción del Diseño

El sistema consiste en un receptor serie asíncrono (UART) que recibe caracteres y muestra la representación binaria de la parte baja del carácter en 4 Leds. Cuando se presiona un botón, se representa la parte alta del carácter. En la Figura 1 se ve el diagrama en bloques del sistema.

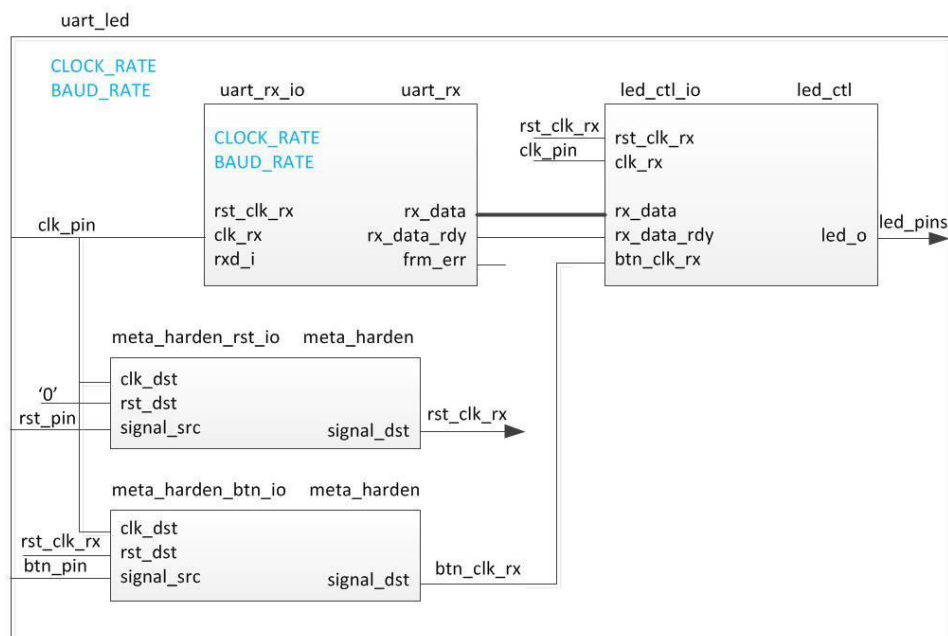
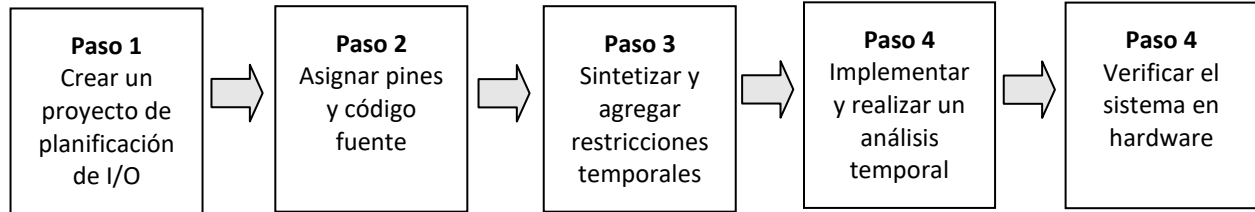


Figura 1. Sistema a implementar

## Flujo General para esta práctica



## Crear un proyecto de planificación de I/O

### Paso 1

**1-1. Iniciar la herramienta Vivado y crear un proyecto con la placa Arty Z7. Agregar los archivos HDL y crear los archivos de restricciones uart\_led\_pins\_ArtyZ7.xdc y uart\_led\_timing\_ArtyZ7.xdc.**

**1-1-1.** Iniciar la aplicación Vivado.

**1-1-2.** Presionar **Create Project** para iniciar el asistente. Aparecerá el cuadro de diálogo **Create A New Vivado Project**. Presionar **Next**.

**1-1-3.** Presionar el botón de navegación del campo **Project location** del formulario **New Project**, elegir la ubicación del proyecto y presionar **Select**.

**1-1-4.** Ingresar el nombre del proyecto en el campo **Project name**. Verificar que la opción **Create Project Subdirectory** esté seleccionada. Presionar **Next**.

**1-1-5.** Seleccionar la opción **I/O Planning Project** en el formulario **Project Type**, y presionar **Next**.

**1-1-6.** Seleccionar **Do not import I/O ports at this time**, y presionar **Next**.

**1-1-7.** En el formulario **Default Part**, usar la opción **Boards**. Seleccionar la placa **Arty Z7-10**. Presionar **Next**.

Se deberá ver una imagen como la siguiente:

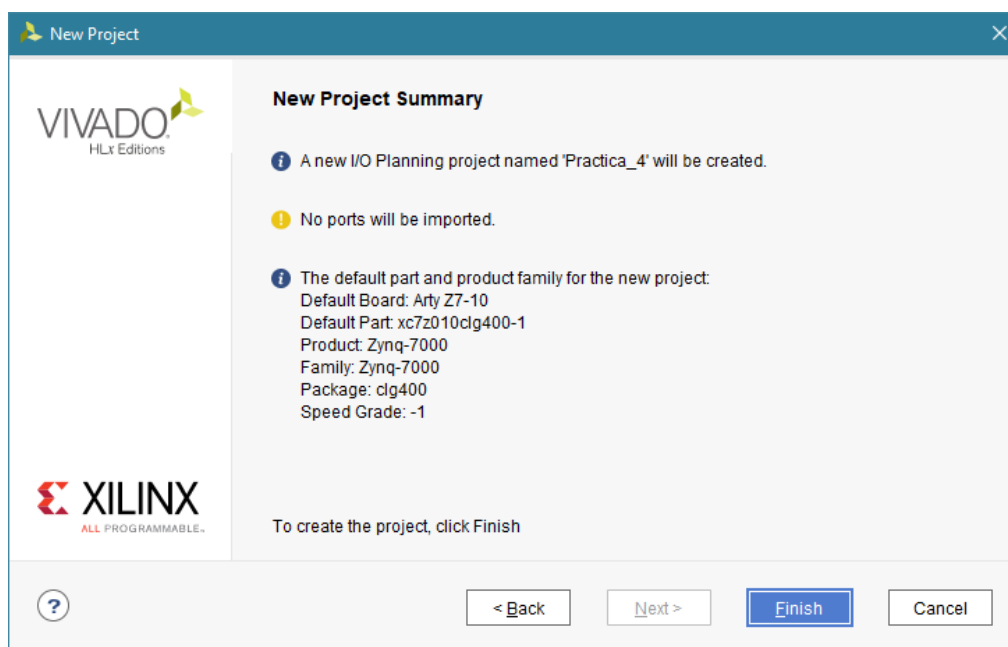


Figura 2. Resumen de la configuración del proyecto

### 1-1-8. Presionar **Finish** para crear el proyecto.

Se creará el proyecto y se abrirán las ventanas de vista de dispositivo (device view) y la pestaña de pines del encapsulado (package pins tab).

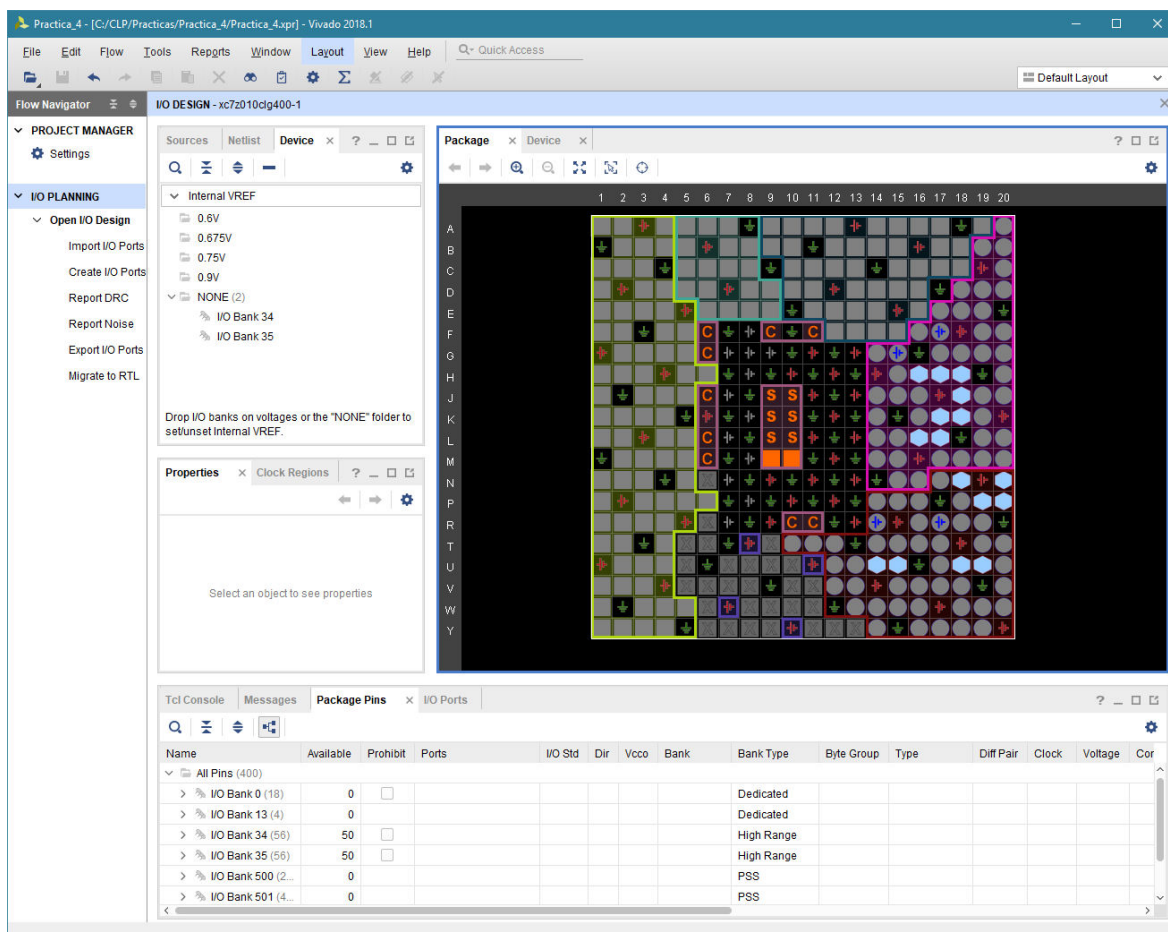


Figura 3. Ventana de vista de dispositivo y pestaña de pines del encapsulado

## Asignar pines y código fuente

## Paso 2

### 2-1. Crear los puertos de entrada rxd\_pin, clk\_pin, btn\_pin y rst\_pin

- 2-1-1. En **Flow Manager**, en la **sección I/O Planning**, expandir **Open I/O Design** y presionar **Create I/O Ports**.

Se abrirá el formulario **Create Ports**.

- 2-1-2. En el campo **Name**, poner clk\_pin; en el campo **Direction** seleccionar Input y como **I/O Standard** seleccionar LVCMOS33. Presionar **OK**.

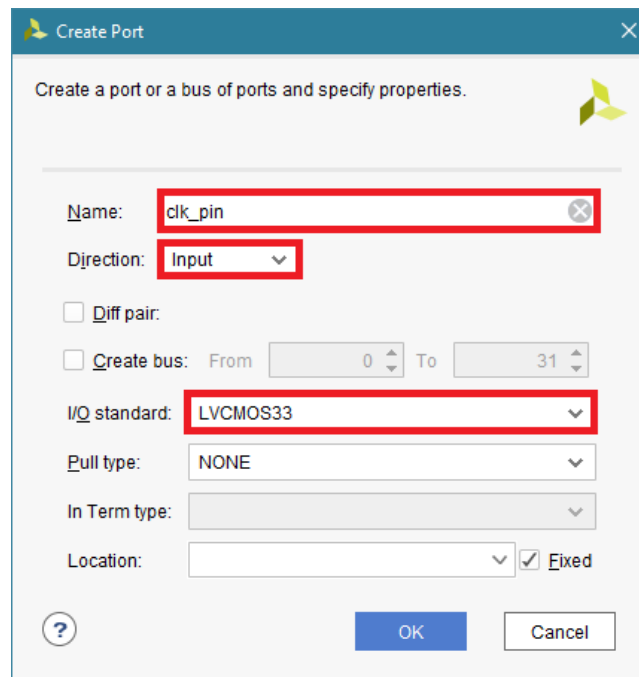


Figura 4. Formulario Create Port

- 2-1-3. De la misma manera, crear los puertos de entrada rxd\_pin, btn\_pin y rst\_pin.

Asignar los puertos rxd\_pin, clk\_pin, btn\_pin y rst\_pin a los pads Y18, H16, D19, y L19 utilizando la vista **Package** y la pestaña de pines del encapsulado (**Package Pins**).

Deslizar el puntero del ratón sobre el pad H16 en la ventana de vista del dispositivo

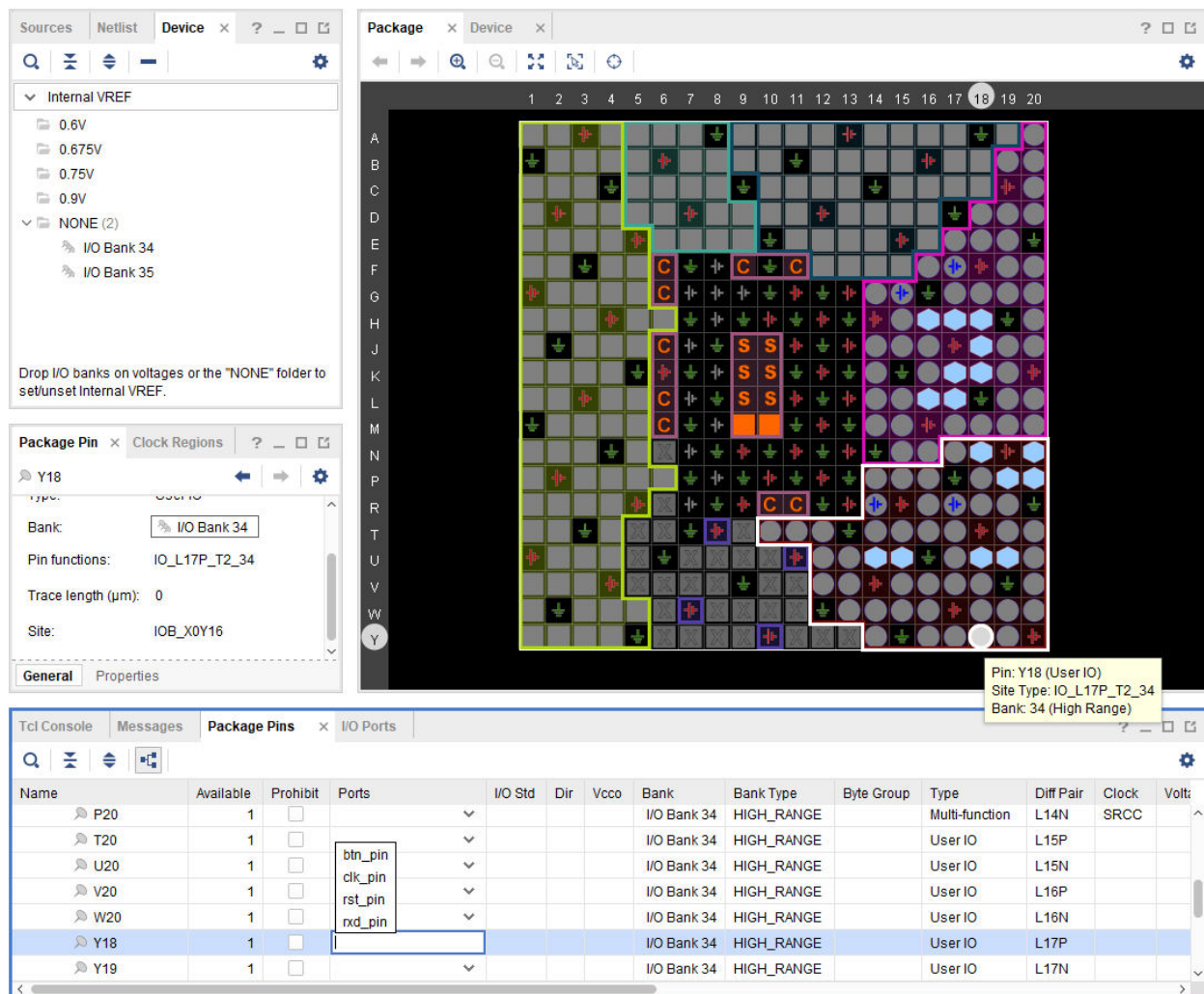


Figura 5. Pin Y18 en la vista Package

**2-1-4.** Presionar el ratón sobre el pad.

El pad quedará seleccionado y se mostrará en la pestaña de pines del encapsulado.

**2-1-5.** En la pestaña **Package Pins**, presionar en la columna **Ports** del pad H16, y seleccionar clk\_pin del cuadro desplegable.

**2-1-6.** De la misma manera, agregar el puerto btn\_pin al pad D19 y el puerto rxn\_pin al pad Y18

**2-1-7.** Para el botón de reset, Seleccionar Edit ► Find o Ctrl-F para abrir el cuadro de búsqueda. En el cuadro desplegable **Find** seleccionar **Package Pins**, En el campo de búsqueda escribir \*L19, y presionar **OK**.

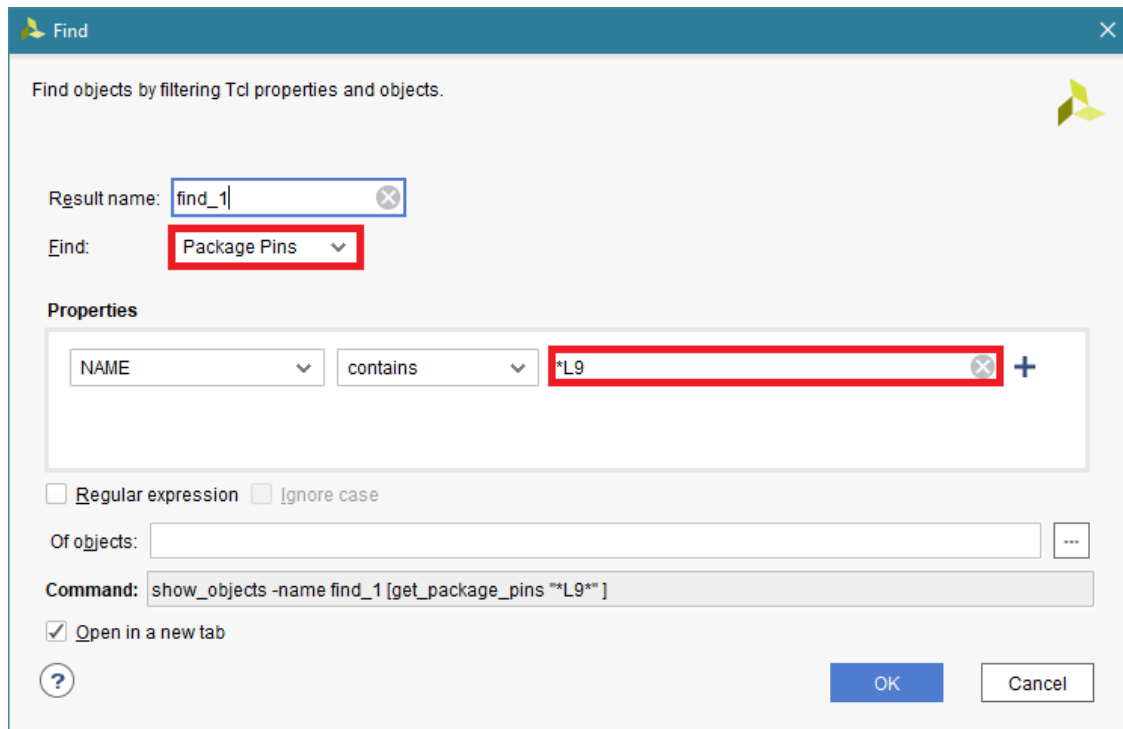


Figura 6. Cuadro de búsqueda

Se abrirá una nueva pestaña **Find Results** con el pad buscado.

**2-1-8.** Asignar el puerto rst\_pin al pad de la misma manera que los otros puertos.

**2-2. Asignar los pines de salida led\_pins[0] a led\_pins[3] a los pads R14, P14, N16 y M14. Crearlos como un vector y asignarlos mediante el comando Tcl set\_property como tipo LVCMOS33**

**2-2-1.** En la pestaña **I/O Ports**, presionar el botón **Create** (símbolo "+") y seleccionar **Create I/O Port ...** en el menú desplegable

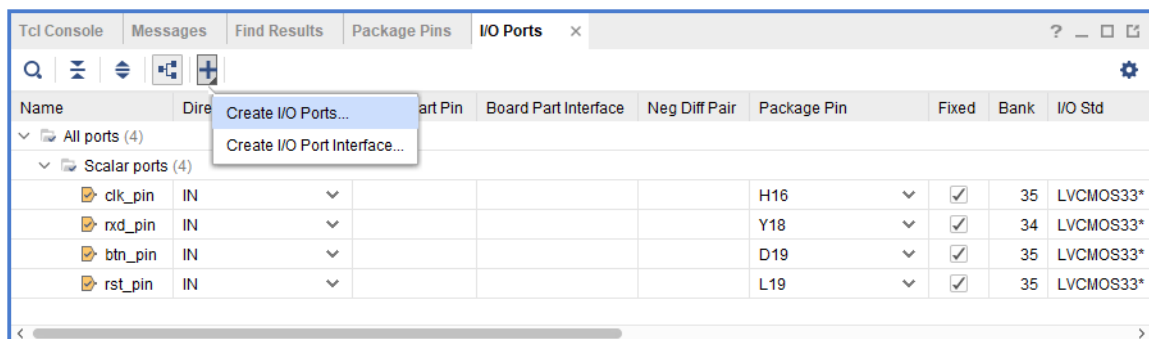


Figura 7. Crear puertos con el botón Create ("+")

Aparecerá el formulario **Cled\_pinscreate Port**



- 2-2-2.** En el campo **Name**, escribir `led_pins` y en el cuadro desplegable **Direction** seleccionar **Output**. Seleccionar el recuadro **Create bus**, establecer el **msb** en 3 y el **lsb** en 0, y seleccionar como **I/O standard** **LVC MOS33**. Presionar **OK**.

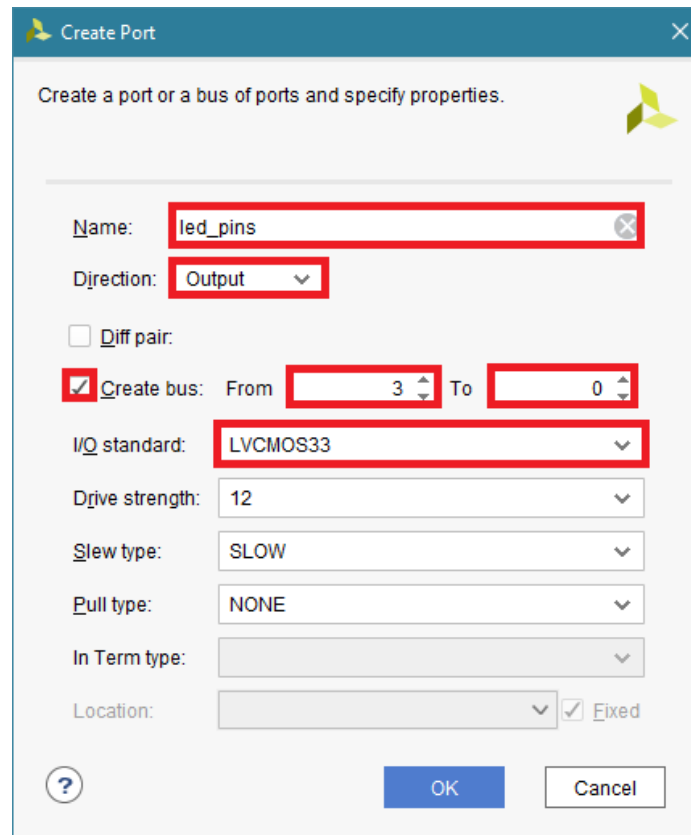


Figure 8. Creating I/O ports for the `led_pins` output

En la pestaña de puertos de I/O aparecerán los 4 pines creados. Ya están establecidos la dirección y el estándar, quedando solamente por asignar el pad.

- 2-2-3.** La asignación de pads se realizará mediante la consola Tcl utilizando los siguientes comandos:

```
set_property -dict { PACKAGE_PIN R14 } [get_ports { led_pins[0] }]
set_property -dict { PACKAGE_PIN P14 } [get_ports { led_pins[1] }]
set_property -dict { PACKAGE_PIN N16 } [get_ports { led_pins[2] }]
set_property -dict { PACKAGE_PIN M14 } [get_ports { led_pins[3] }]
```

Seleccionar **File ► Constraints ► Save**.

Aparecerá el cuadro de diálogo **Save Constraints**.

- 2-2-4.** En el campo **File name** escribir `uart_led_ArtyZ7` y presionar **OK**.

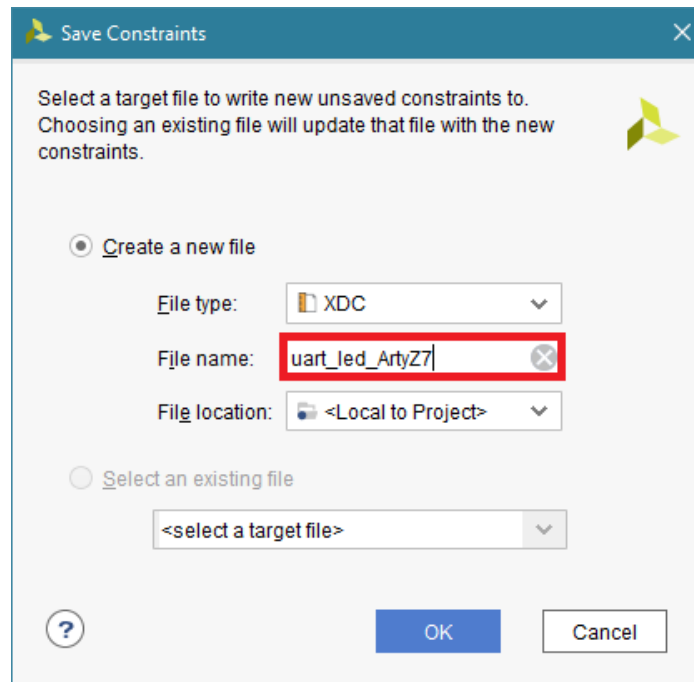


Figura 9. Cuadro de diálogo Save Constraints

Se creará el archivo `uart_led_ArtyZ7.xdc` y se agregará en la pestaña **Sources**.

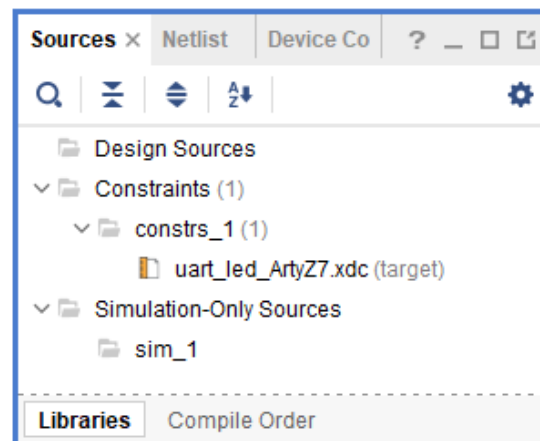


Figura 10. El archivo `uart_led_ArtyZ7.xdc` agregado a Sources

- 2-2-5. En el panel **Flow Navigator**, en la sección **I/O Planning**, expandir **Open I/O Design**, presionar **Report DRC** y presionar **OK**. Ignorar el warning relativo al uso del bloque PS7 (Procesador).
- 2-2-6. Presionar en **Report Noise** y presionar **OK**. Se realizará un análisis de ruido en los pines de salida y se mostrarán los resultados.
- 2-2-7. Presionar en **Migrate to RTL**.  
Aparecerá el formulario **Migrate to RTL** con el campo **Top RTL file** mostrando el archivo `io_1.v`
- 2-2-8. Cambiar el campo **Netlist format** de Verilog a VHDL y `io_1.vhd` a `uart_top.vhd`, y presionar **OK**.

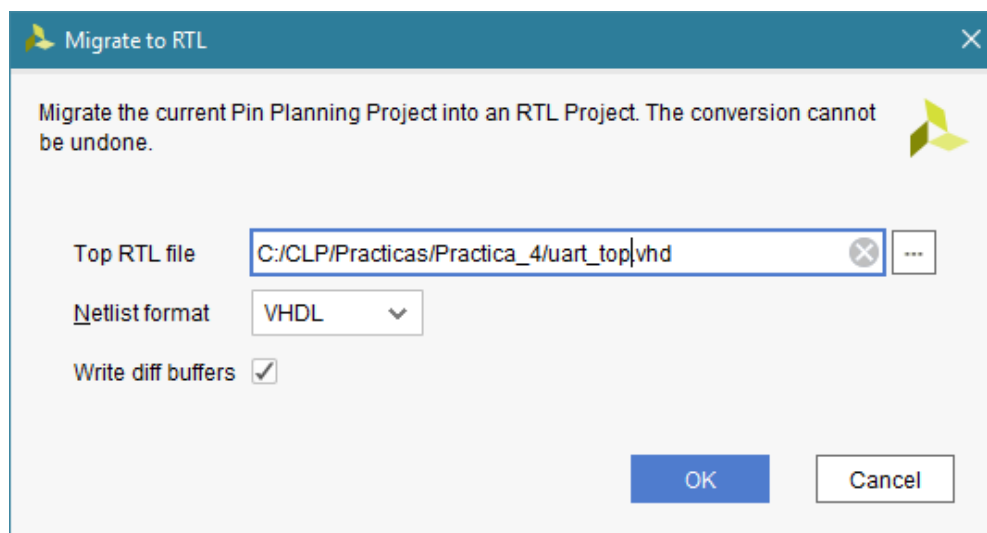


Figura 11. Asignación del nombre de archivo

**2-2-9.** Seleccionar la pestaña **Hierarchy** dentro del panel **Sources** y verificar que el archivo `uart_top.vhd` se agregó al proyecto con una entidad **ios**. Presionar dos veces y ver el listado de puertos.

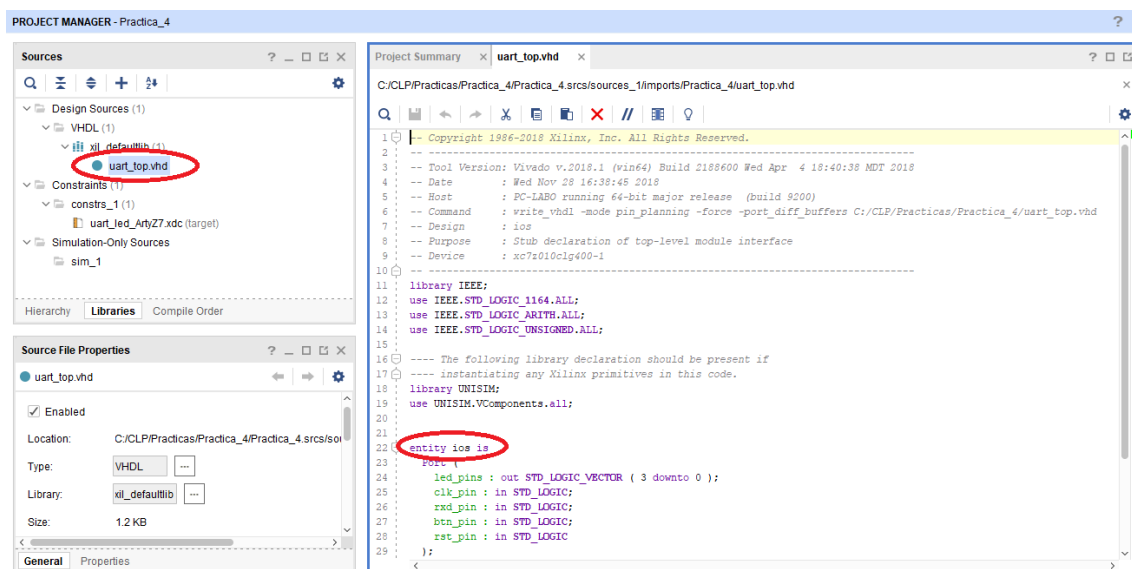


Figura 12. El modulo top y la jerarquía de módulos después de migrar a RTL

**2-3. Agregar los archivos de código fuente al proyecto. Modificar el archivo `uart_top.vhd`.**

**2-3-1.** En el panel **Flow Navigator**, presionar **Add Sources**. Presionar **Next** (se deja seleccionada la opción por defecto **Add or create design sources**).

**2-3-2.** En el cuadro de diálogo **Add Sources**, presionar el botón **Add Files**.

**2-3-3.** Navegar a la ubicación de los archivos .vhd (led\_ctl.vhd, meta\_harden.vhd, uart\_baud\_gen.vhd, uart\_led.vhd, uart\_rx.vhd y uart\_rx\_ctl.vhd), seleccionarlos y presionar **OK**.

**2-3-4.** Presionar **Finish**.

**2-3-5.** Abrir el archivo uart\_top.txt con un editor de texto. Copiar la declaración del componente uart\_led y pegarla en la parte declarativa del componente ios (archivo uart\_top.vhd, dentro del proyecto). Luego copiar la instanciación del componente uart\_led y pegarla en la parte descriptiva (luego del begin de la arquitectura) del componente ios. Guardar el archivo uart\_top.vhd.

```

31 end ios;
32
33 architecture STRUCTURE of ios is
34
35     component uart_led is
36     generic(
37         BAUD_RATE: integer := 115200;
38         CLOCK_RATE: integer := 50E6
39     );
40     port(
41         -- Write side inputs
42         clk_pin:    in std_logic;           -- Clock input (from pin)
43         rst_pin:    in std_logic;           -- Active HIGH reset (from pin)
44         btn_pin:    in std_logic;           -- Button to swap high and low bits
45         rxd_pin:    in std_logic;           -- RS232 RXD pin - directly from pin
46         led_pins:   out std_logic_vector(3 downto 0) -- 8 LED outputs
47     );
48 end component;
49
50 begin
51
52     U0: uart_led
53     generic map(
54         BAUD_RATE => 115200,
55         CLOCK_RATE => 125E6
56     )
57     port map(
58         clk_pin => clk_pin, -- Clock input (from pin)
59         rst_pin => rst_pin, -- Active HIGH reset (from pin)
60         btn_pin => btn_pin, -- Button to swap high and low bits
61         rxd_pin => rxd_pin, -- RS232 RXD pin - directly from pin
62         led_pins => led_pins -- 8 LED outputs
63     );
64
65 end STRUCTURE;

```

Figura 13. Estructura del código del componente ios

---

**Sintetizar y agregar las restricciones de temporización****Paso 3**

---

- 3-1. Sintetizar el sistema. Usar el asistente de restricciones (Constraints Wizard) para especificar una frecuencia de reloj y retardos de entrada y salida.**
- 3-1-1.** En el menú **Flow Navigator**, presionar **Run Synthesis**. Presionar **OK**.  
Si aparece el cuadro de diálogo **Save Project**, presionar el botón **Save**.  
Cuando termine el proceso de síntesis, aparecerá un cuadro de diálogo con tres opciones.
- 3-1-2.** Seleccionar **Open Synthesized Design** y presionar **OK**.
- 3-1-3.** En el menú **Flow Navigator**, dentro de **Open Synthesized Design**, presionar en **Constraints Wizard**. Esto abrirá el Asistente de Restricciones.
- 3-1-4.** Leer la pantalla inicial **Identify and Recommend Missing Timing Constraints** para entender el funcionamiento del asistente y presionar **Next**.
- 3-1-5.** Especificar la frecuencia del objeto "clk\_pin" en 125 MHz, en el cuadro **Frequency** (MHz) de la tabla **Recommended Constraints**. Presionar **Enter** y verificar que los cuadros **Period**, **Rise At** y **Fall At** se completan automáticamente. En la pestaña **Tcl Command Preview** se puede ver el comando Tcl que creará el reloj. Presionar **Next**.

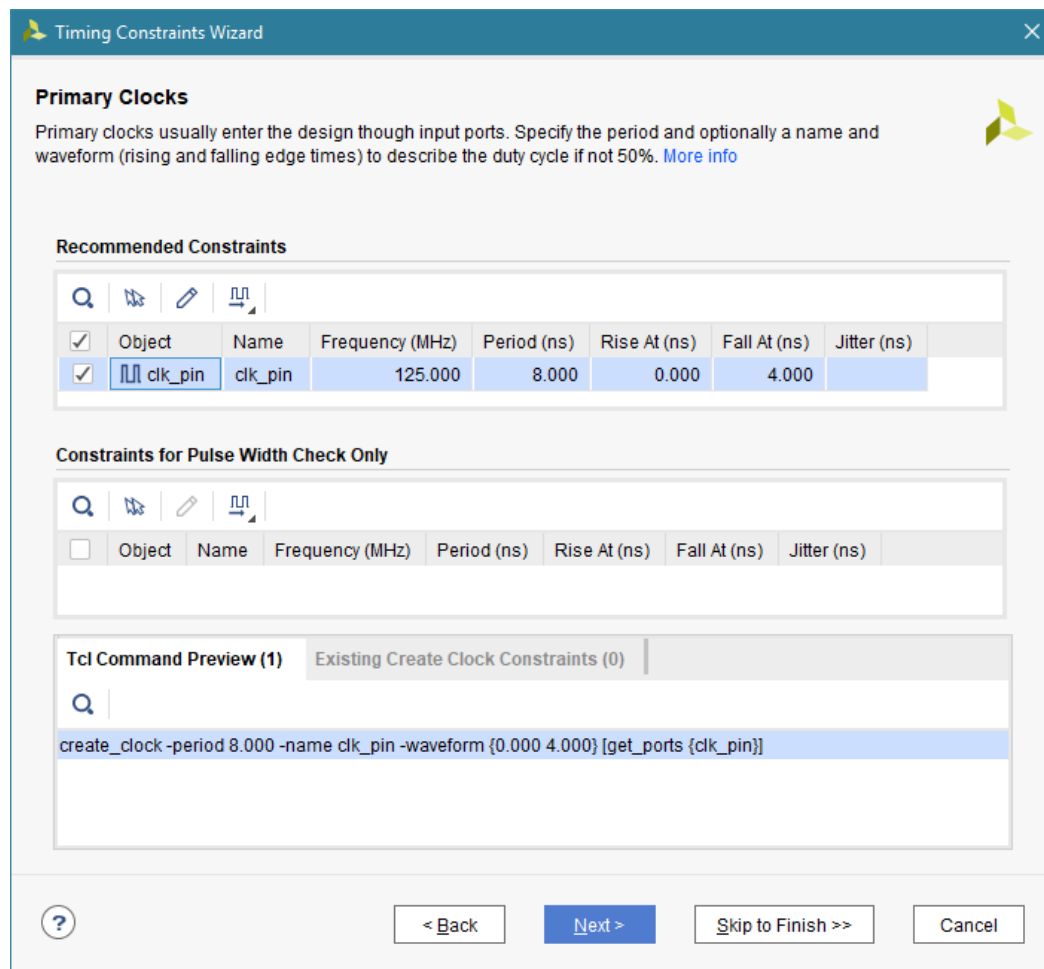


Figura 14. Asistente de Restricciones: Reloj clk\_pin

- 3-1-6.** El asistente no detecta **Generated Clocks** faltantes, presionar **Next**.
- 3-1-7.** El asistente no detecta **Forwarded Clocks** faltantes, presionar **Next**.
- 3-1-8.** El asistente no detecta **External Feedback Delays** faltantes, presionar **Next**.
- 3-1-9.** El asistente identifica que es necesario definir retardos de entrada para los pines btn\_pin, rxd\_pin y rst\_pin. Los mismos se definen de la siguiente manera:
- (1) Seleccionar las tres filas.
  - (2) Ingresar un valor de tco\_min de -0.5 ns y 0 ns para los otros campos. Presionar **Apply** (en la pestaña **Tcl Command Preview**, hay 6 comandos Tcl para aplicar las restricciones).
  - (3) Presionar **Next**.

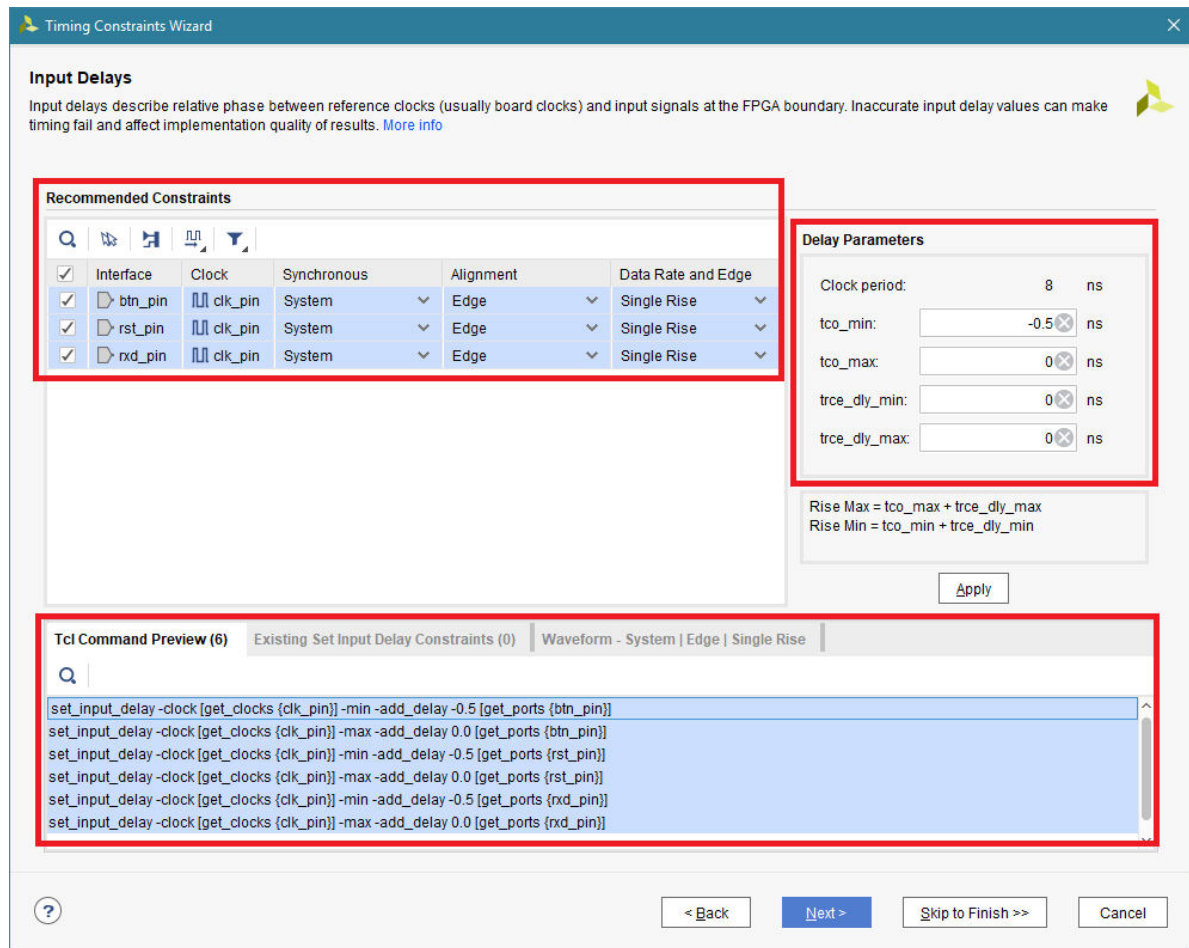


Figura 15. Especificar los retardos de entrada para las señales rxd\_pin, btn\_pin y rst\_pin

**3-1-10.** Para los retardos de salida, establecer tsu y thd as 0 ns y para trce\_dly\_max y trce\_dly\_min establecerlo en -2 ns. Presionar **Apply** y **Next**.

**3-1-11.** El asistente no detecta **Combinatorial Delays** faltantes, presionar **Next**.

**3-1-12.** Presionar **Skip to Finish** para llegar a la página de resumen de restricciones (Constraints Summary).

**3-1-13.** Seleccionar **View Timing Constraints** en el panel **On Finish** y presionar **Finish** para finalizar el asistente. Esta opción abrirá el editor de restricciones (Timing Constraints Editor) para mostrar las restricciones generadas.

**3-1-14.** El asistente generó una restricción para clk\_pin de 8 ns de periodo (o 125 MHz). En la ventana **All Constraints** se pueden observar las 9 restricciones creadas.

No hace falta presionar el botón **Apply** porque las restricciones ya fueron aplicadas por el asistente de restricciones

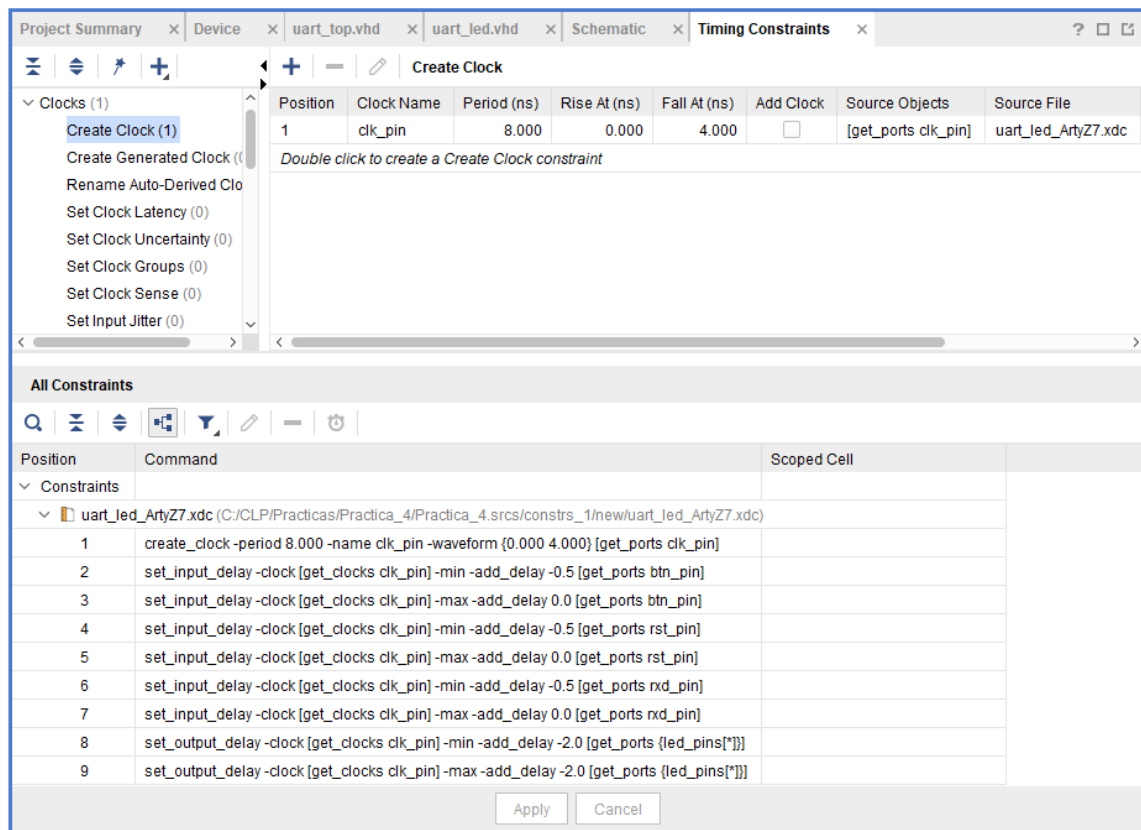


Figura 16. Restricciones agregadas por el Asistente de Restricciones

**3-1-15.** Abrir el archivo `uart_led_ArtyZ7.xdc` (si estaba abierto, presionar el enlace **Reload**) y verificar las restricciones agregadas al final del archivo.

**3-2. Generar un reporte de temporización estimado mostrando los tiempos de establecimiento y retención de los caminos de señal del sistema.**

**3-2-1.** En el menú **Flow Navigator**, seleccionar **Open Synthesized Design** ► **Report Timing Summary**.

**3-2-2.** En la pestaña de **Options**, verificar que esté seleccionada la opción **min\_max** en **Path delay type**



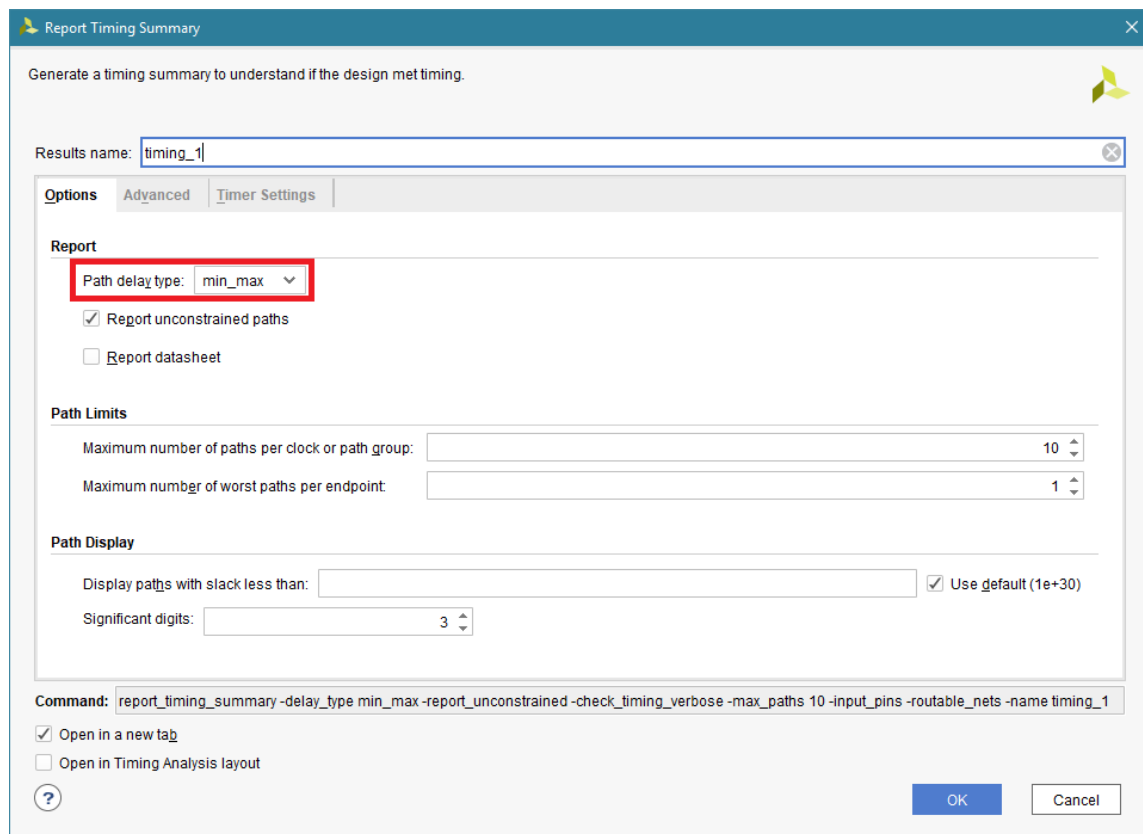


Figura 17. Opciones del análisis temporal

### 3-2-3. Presionar **OK** para ejecutar el análisis.

Los resultados del análisis aparecen en la pestaña **Timing** en la parte inferior de la pantalla.

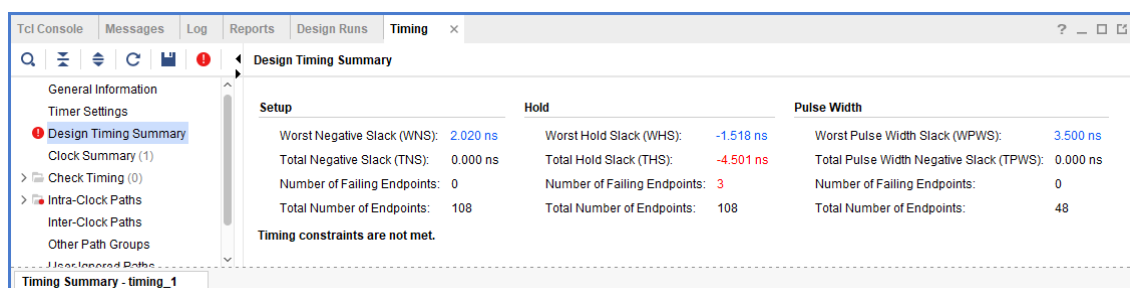


Figura 18. Reporte de temporización

El reporte **Design Timing Summary** da información sobre los peores tiempos de establecimiento y retención, y el número de caminos de señal que no cumplen su temporización. En este caso hay 3 caminos que no cumplen su tiempo de retención.

### 3-2-4. Expandir *Intra-Clock Paths* ► *clk\_pin* ► *Hold* para ver los caminos que no cumplen la temporización

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay
Path 11	-1.518	1	1	1	btn_pin	U0/meta_harden_btn_i0/signal_meta_reg/D	2.166	1.407	0.760
Path 12	-1.497	1	1	1	rxn_pin	U0/uart_rx_i0/meta_reg/D	2.187	1.428	0.759
Path 13	-1.486	1	1	1	rst_pin	U0/meta_harden_btn_i0/signal_meta_reg/D	2.198	1.438	0.760
Path 14	0.134	1	2	1	U0/led_ctl_i0/_data_reg[4]/C	U0/led_ctl_i0/_eg_reg[0]/D	0.370	0.239	0.131
Path 15	0.134	1	2	1	U0/led_ctl_i0/_data_reg[6]/C	U0/led_ctl_i0/_eg_reg[2]/D	0.370	0.239	0.131
Path 16	0.137	1	2	1	U0/led_ctl_i0/_data_reg[5]/C	U0/led_ctl_i0/_eg_reg[1]/D	0.373	0.242	0.131
Path 17	0.137	1	2	1	U0/led_ctl_i0/_data_reg[7]/C	U0/led_ctl_i0/_eg_reg[3]/D	0.373	0.242	0.131

Figura 19. Lista de caminos que no cumplen la temporización

Los tres caminos de señal que no cumplen la temporización son el de *rxn\_pin*, *btn\_pin* y *rst\_pin*.

3-2-5. Presionar dos veces sobre el camino 11 para ver los detalles.

Summary

Name	Path 11
Slack (Hold)	-1.518ns
Source	btn_pin (input port clocked by clk_pin {rise@0.000ns fall@4.000ns period=8.000ns})
Destination	U0/meta_harden_btn_i0/signal_meta_reg/D (rising edge-triggered cell FDRE clocked by clk_pin {rise@0.000ns fall@4.000ns period=8.000ns})
Path Group	clk_pin
Path Type	Hold (Min at Slow Process Corner)
Requirement	0.000ns (clk_pin rise@0.000ns - clk_pin rise@0.000ns)
Data Path Delay	2.166ns (logic 1.407ns (64.930%) route 0.760ns (35.070%))
Logic Levels	1 (IBUF=1)
Input Delay	-0.500ns
Clock Path Skew	2.942ns
Clock Uncertainty	0.035ns

Data Path

Delay Type	Incr (ns)	Path ...	Locati...	Netlist Resource(s)
(clock clk_...rise edge)	(r) 0.000	0.000		
input delay	-0.500	-0.500		
	(r) 0.000	-0.500	Sit...19	btn_pin
net (fo=0)	0.000	-0.500		btn_pin
			Sit...19	btn_pin_IBUF_inst/I
IBUF (Prop_ibuf I O)	(r) 1.407	0.907	Sit...19	btn_pin_IBUF_inst/O
net (fo=1, unplaced)	0.760	1.666		U0/meta_harden_btn_i0/btn_pin
FDRE				U0/meta_harden_btn_i0/signal_meta_reg/D
Arrival Time		1.666		

Destination Clock Path

Delay Type	Incr (ns)	Path ...	Locati...	Netlist Resource(s)
(clock clk_...rise edge)	(r) 0.000	0.000		
	(r) 0.000	0.000	Sit...16	clk_pin
net (fo=0)	0.000	0.000		clk_pin
			Sit...16	clk_pin_IBUF_inst/I

Figura 20. Detalle de temporización del camino de señal 11

**3-2-6.** Seleccionar el camino 11, presionar el botón derecho y seleccionar **Schematic**.

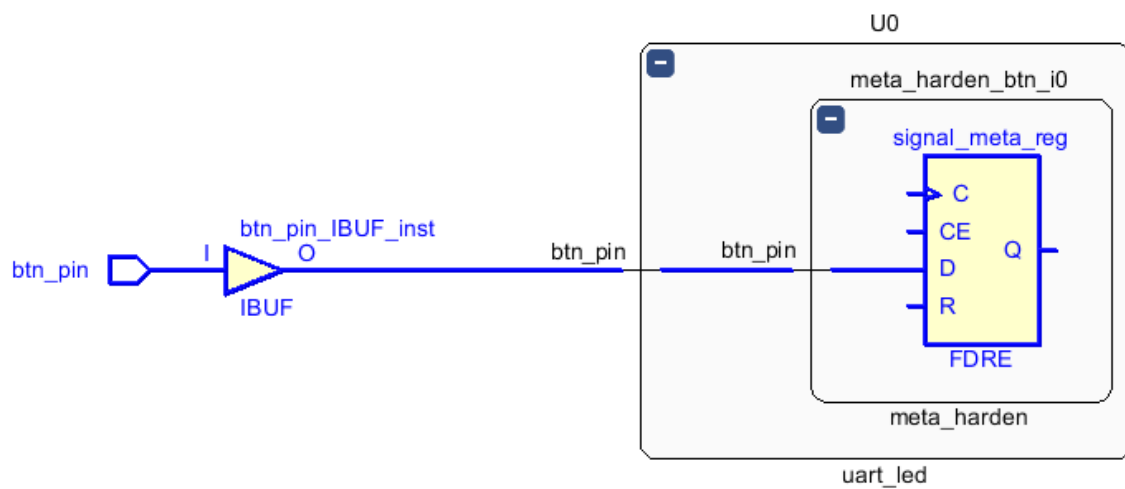


Figura 21. Diagrama esquemático del camino 11

## Implementar y realizar un análisis

## Paso 4

### 4-1. Implementar el sistema.

4-1-1. Presionar **Run Implementation** en el menú **Flow Navigator**.

4-1-2. Presionar **Yes** para ejecutar primero el proceso de Síntesis.

Cuando finalice el proceso de implementación, aparecerá un cuadro de diálogo con tres opciones.

4-1-3. Seleccionar la opción **Open Implemented Design**. Presionar **Yes** si aparece un cuadro de diálogo preguntando si se cierra el sistema sintetizado.

4-1-4. Presionar **OK** en el cuadro de diálogo **Critical Messages**.

### 4-2. Generar un reporte Timming Summary.

4-2-1. En el menú **Flow Navigator**, en Implementation ► Open Implemented Design, presionar **Report Timing Summary**.

4-2-2. Presionar **OK** para generar el reporte con las opciones por defecto.

Los resultados del análisis aparecen en la pestaña **Timing** en la parte inferior de la pantalla. Luego de la implementación los tiempos que no se cumplen son los de establecimiento.

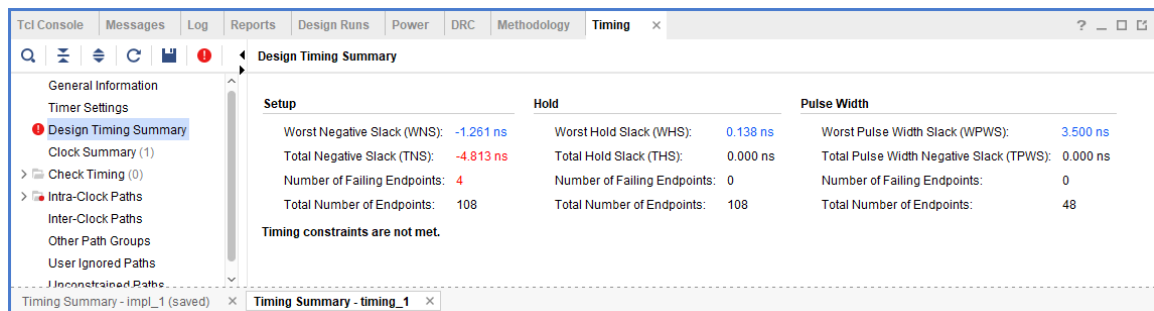


Figura 22. Tiempos de establecimiento incumplidos

4-2-3. Presionar en el enlace **WNS** para ver los caminos que no cumplen su temporización. Los mismos resultan ser los caminos de salida led\_pins.

4-2-4. Presionar dos veces en el primer camino que no cumple su temporización y ver los detalles del mismo.

Es posible reducir el retardo de salida colocando los registros asociados en bloques IOB.

4-2-5. Para aplicar la restricción que los registros asociados a led\_pins estén en bloques IOB se ejecuta el siguiente comando Tcl:

```
set_property IOB TRUE [get_ports led_pins[*]]
```

4-2-6. Seleccionar File ► Constraints ► Save. Presionar **OK** en el cuadro de diálogo.

4-2-7. Presionar **Run Implementation**.

4-2-8. Presionar **Yes** para reiniciar los procesos de síntesis e implementación.

4-2-9. Seleccionar **Open Implemented Design**. Presionar **OK**. La pestaña **Timing** debería haberse actualizado. Verificar que ahora no hay caminos que no cumplan con la temporización.

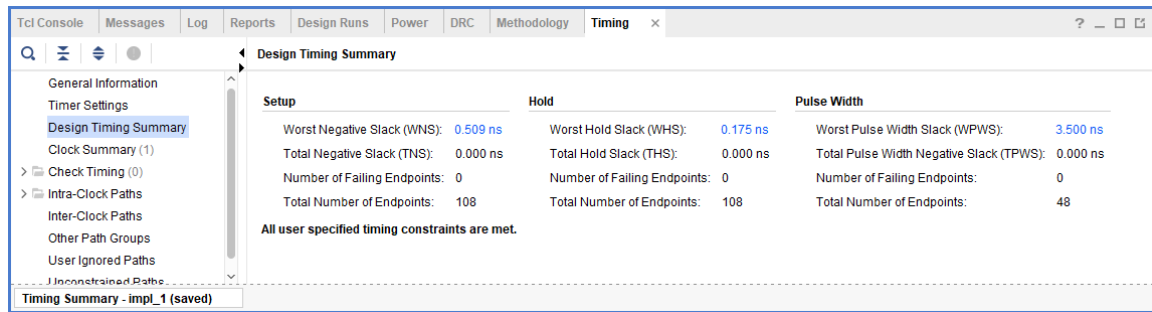


Figura 23. Tiempos de establecimiento cumplidos

4-2-10. Guardar el archivo uart\_led\_ArtyZ7.xdc si el mismo no está guardado.

## Verificar el sistema en hardware

## Paso 5

### 5-1. Generar el archivo de configuración.

**5-1-1.** En el menú **Flow Navigator**, en **Program and Debug**, presionar **Generate Bitstream**. Presionar **Yes** y a continuación **OK** para realizar los procesos de síntesis e implementación si es necesario.

**5-1-2.** Presionar **Cancel** cuando aparezca el cuadro de diálogo luego de que se generó el archivo de configuración

### 5-2. Conectar la placa, conectar el adaptador usb-serie. Abrir una sesión de hardware y configurar la FPGA.

**5-2-1.** Conectar el cable micro-usb al conector prog/uart de la placa Arty Z7. Este conector se encuentra al costado del conector ethernet. Conectar el otro extremo a un puerto usb de la computadora.

**5-2-2.** Conectar el adaptador usb-serie a la computadora. Verificar el puerto COM al que se mapea.

**5-2-3.** Conectar los 3 cables de Tx; Rx y GND entre el conector USB-serie y el puerto JA de la placa ArtyZ7: TxD en el pin 1; RxD en el pin 2 y GND en el pin 5

Primero conectar la placa ArtyZ7 a la computadora, luego conectar el adaptador usb-serie.

**5-2-4.** En el panel **Flow Navigator**, seleccionar **Program and Debug**, presionar **Open Hardware Manager**.

Aparecerá la interfaz del administrador de hardware (Hardware Manager) indicando el estado desconectado (**unconnected**).

**5-2-5.** Presionar en el enlace **Open target**, y luego en la opción **Auto Connect** del menú.

**5-2-6.** El estado de la sesión de hardware pasa de desconectado a tener el nombre del servidor (en este caso local host porque la placa está conectada en forma local). También se indica que el estado del dispositivo es **not programmed**.

**5-2-7.** Seleccionar el dispositivo (xc7z010\_1) en la ventana de hardware. Presionar el botón derecho y seleccionar Program Device....

**5-2-8.** En el cuadro de diálogo, presionar el botón **Program**

Se transferirá el archivo de configuración y el led verde DONE se prenderá cuando la FPGA esté configurada

### 5-3. Iniciar un programa de emulación de terminal (por ejemplo TeraTerm), configurarlo con el puerto serie del adaptador usb-serie y con los parámetros de comunicación 115200 N-8-1.

**5-3-1.** Iniciar el programa de comunicaciones (por ejemplo TeraTerm).

- 5-3-2.** Seleccionar el puerto serie (COM) del adaptador usb-serie.
- 5-3-3.** Configurar el puerto serie a 115200, sin paridad, 8 bits de datos, 1 bit de parada.
- 5-3-4.** En el programa de comunicaciones, escribir algunos caracteres. Su valor hexadecimal se verá en los 4 leds (se verá el nibble bajo del ASCII equivalente del carácter).
- 5-3-5.** Presionar y mantener presionado el botón BTN0 y ver como ahora los leds muestran el nibble alto del ASCII equivalente del caracter escrito en el programa de comunicaciones.
- 5-3-6.** Seleccionar File ► Close Hardware Manager. Presionar **OK**.
- 5-3-7.** Cerrar el programa de comunicaciones. Desconectar el adaptador usb-serie de la computadora. Desconectar la placa ArtyZ7 de la computadora. Hacerlo en ese orden, sino se puede dañar la placa ArtyZ7.