

Diseño Digital Avanzado

Unidad 1 - Herramientas

Dr. Ariel L. Pola

apola@fundacionfulgor.org.ar

September 9, 2021

Tabla de Contenidos

1. Proyecto Leds
2. Herramienta Vivado
3. Instancia de VIO eILA
4. Tunel y Asignación de FPGA
5. Programación y Ejecución

Proyecto Leds

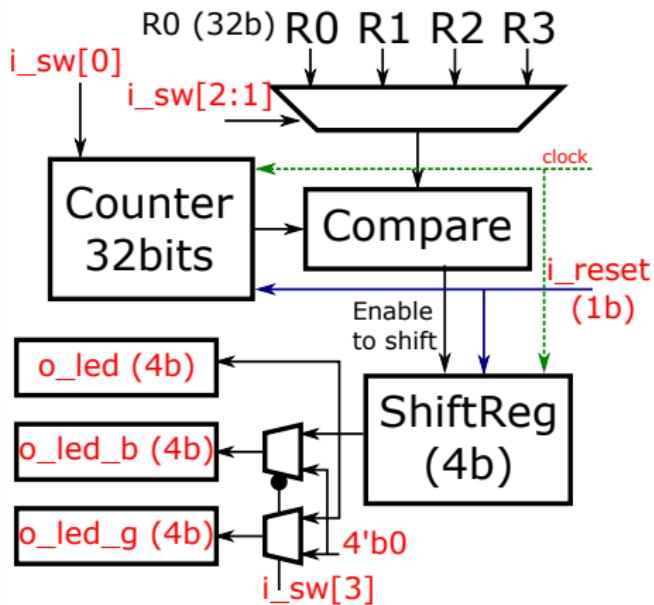


Proyecto LEDS

Implementación en FPGA - Leds

Descripción

- Los nombres en ROJO son puertos.
- *ck_rst* es el reset del sistema, el cual pone a cero el contador e inicializa el shift register (SR).
- *i_sw[0]* controla el enable (1) del contador. En estado (0) todo se detiene sin alterar el estado actual del contador y del SR.
- El SR se desplaza únicamente cuando el contador llegó a algún límite R0-R3.
- La elección del límite se puede realizar en cualquier momento del funcionamiento.
- *i_sw[3]* elige el color de los leds RGB.



Proyecto LEDS

Test Bench

Descripción

- El testbench (banco de pruebas) es un módulo que no tiene puertos declarados.
- Genera los estímulos de reloj y señales de control para modelar un escenario de prueba.
- Las variables utilizadas para estimular los puertos de entrada son declaradas como tipo **reg**.
- Las variables utilizadas para conectar los puertos de salida son declaradas como tipo **wire**.
- Lectura y cambio de estado de variables internas a los módulos
 - Definiendo las instancias se puede leer las variables internas
Ejemplo, assign tb_count = tb_shiftleds.u_shiftleds.counter;
 - Utilizando **force** se cambia el valor de una variable en una instancia. Se debe definir dentro de un bloque de procesamiento **initial**.
Ejemplo, force tb_shiftleds.u_shiftleds.o_led = 4'b0001;

Proyecto LEDS

Test Bench

Ejemplo - Generando Estímulos

```
1  `define N_LEDS 4
2  `define NB_SW 4
3
4  `timescale 1ns/100ps
5
6  module tb_shiftleds();
7
8    parameter N_LEDS = 'N_LEDS;
9
10   wire [N_LEDS - 1 : 0] o_led;
11   reg [NB_SW - 1 : 0] i_sw;
12   reg                  ck_rst;
13   reg                  CLK100MHZ;
14
15   initial begin
16     i_sw      = 4'b0000;
17     CLK100MHZ = 1'b0;
18     ck_rst   = 1'b0;
19     #100 ck_rst = 1'b1;
20     #100 i_sw  = 4'b0001;
21     #1000000 i_sw = 4'b0011;
22     #1000000 i_sw = 4'b1011;
23     #1000000 $finish;
24   end
25
26   shifteds
27   #(.N_LEDS (N_LEDS),
28     .NB_SW (NB_SW)
29     )
30
31   u_shiftleds
32   (.o_led (o_led),
33     .i_sw (i_sw),
34     .ck_rst (ck_rst),
35     .CLK100MHZ (CLK100MHZ)
36   );
37
38
39   endmodule // tb_shiftleds
```

Proyecto LEDS

Test Bench

Ejemplo - Estímulos desde Archivos

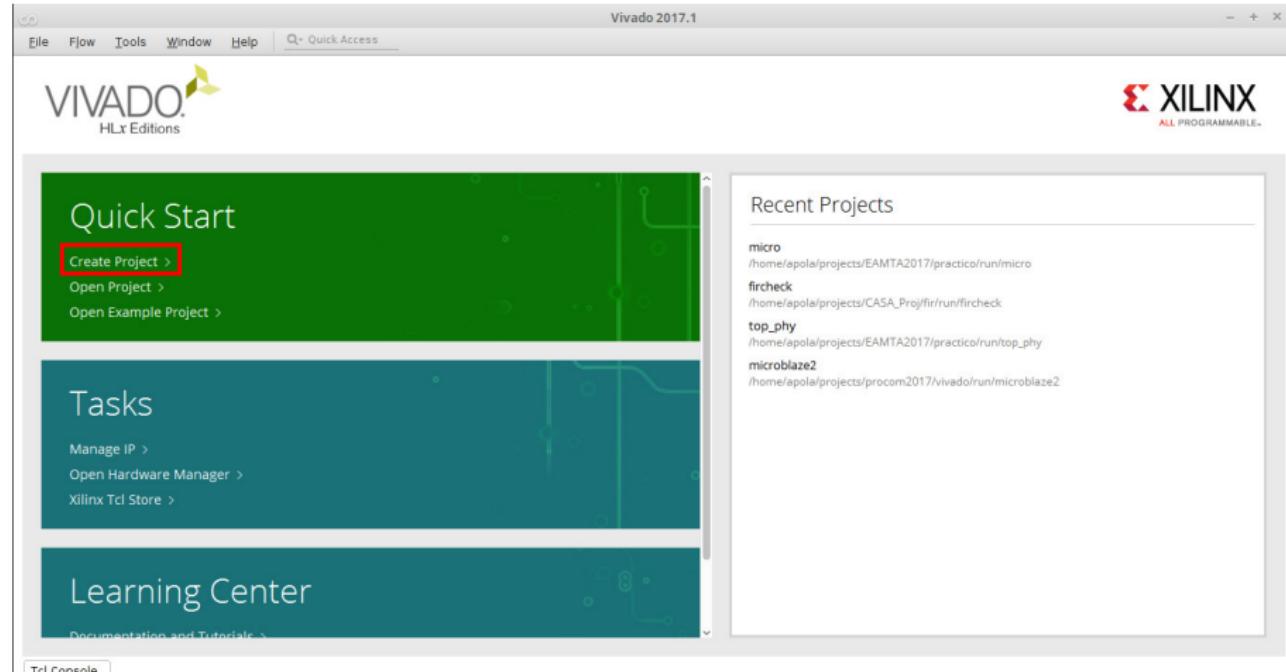
```
1  `define N_LEDS 4
2  `define NB_SW 4
3
4  `timescale 1ns/100ps
5
6  module tb_shiftleds_file();
7
8      parameter N_LEDS    = 'N_LEDS    ;
9      parameter NB_SW     = 'NB_SW     ;
10
11     wire [N_LEDS - 1 : 0] o_led    ;
12     reg  [NB_SW  - 1 : 0] i_sw     ;
13     reg  [NB_SW  - 1 : 0] sw_tmp   ;
14     reg ck_rst, CLK100MHZ, reset_tmp;
15
16     integer fid_reset,fid_sw;
17     integer code_error,code_error1;
18     integer          ptr_sw;
19
20     initial begin
21         fid_reset = $fopen("./vectors/reset.out"
22             , "r");
23         if(fid_reset==0) $stop;
24         fid_sw = $fopen("./vectors/switch.out","r"
25             );
26         if(fid_sw==0) $stop;
27         CLK100MHZ = 1'b0 ;
28     end
29
30     always #5 CLK100MHZ = ~CLK100MHZ;
31
32     always@(posedge CLK100MHZ) begin
33         code_error <=
34             $fscanf(fid_reset,"%d",reset_tmp);
35         if(code_error!=1) $stop;
36
37         for(ptr_sw=0;ptr_sw<NB_SW;
38             ptr_sw = ptr_sw+1) begin
39             code_error1 <=
40                 $fscanf(fid_sw,"%d",sw_tmp[(ptr_sw+1)
41                     -1 -: 1]);
42             if(code_error1!=1) $stop;
43         end
44
45         ck_rst <= reset_tmp;
46         i_sw   <= sw_tmp;
47         $display("%d",ck_rst);
48     end
49
50     shiftleds
51         u_shiftleds
52             (.o_led    (o_led)    ,
53              .i_sw     (i_sw)    ,
54              .ck_rst   (ck_rst)  ,
55              .CLK100MHZ(CLK100MHZ));
56
57     endmodule // tb_shiftleds
```

Herramienta Vivado



Herramienta Vivado

Vivado



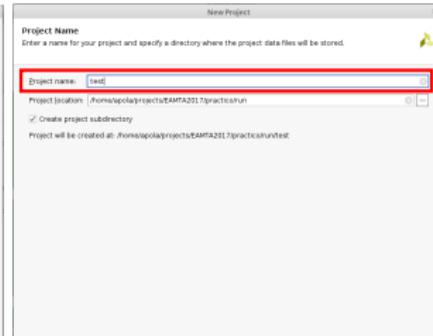
Crea un nuevo proyecto

Herramienta Vivado

Vivado



The screenshot shows the 'Create a New Vivado Project' wizard. In the first step, 'Project Name', the user has entered 'test'. The 'Project location' is set to '/home/espola/projects/EARTFA2017/practicals/untest'. A red box highlights the 'Project name' input field.



The screenshot shows the 'New Project' wizard. In the 'Default Part' step, the 'Part' tab is selected. A red box highlights the 'Part' tab button. The table lists various Xilinx parts, with 'K200' highlighted by a red box.

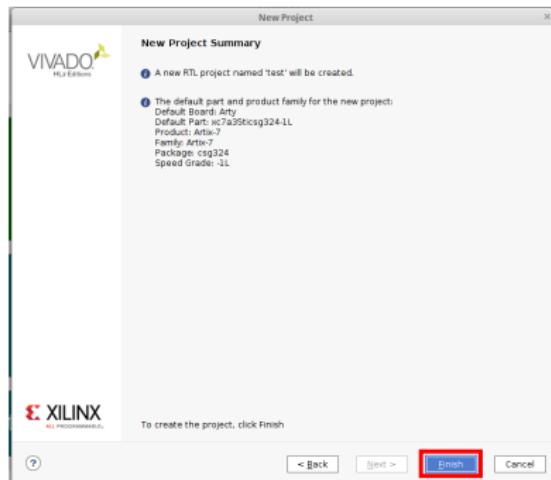
Device Name	Vendor	Board Part	Part	IOB Per Cell	Cells
K200	digilentinc.com	C.0	@xc7a100tsg324-1	24	1.1
Cmod A7-128	digilentinc.com	B.0	@xc7a100tsg324-0	236	1.1
Cmod A7-256	digilentinc.com	B.0	@xc7a100tsg324-1	236	1.1
Noisy4	digilentinc.com	B.1	@xc7a100tsg324-1	324	1.1
Noisy4 DDR	digilentinc.com	C.1	@xc7a100tsg324-1	324	1.1

Configura el kit de trabajo

Diseño Digital Avanzado

Herramienta Vivado

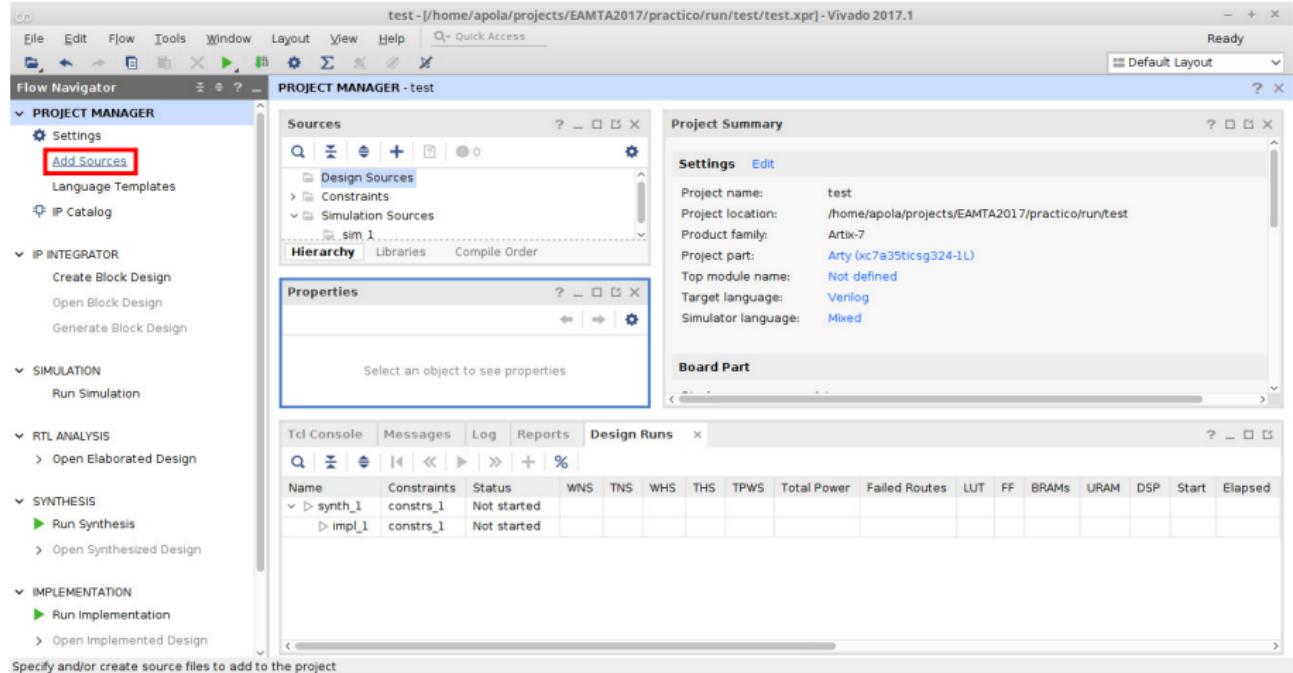
Vivado



Configura el kit de trabajo

Herramienta Vivado

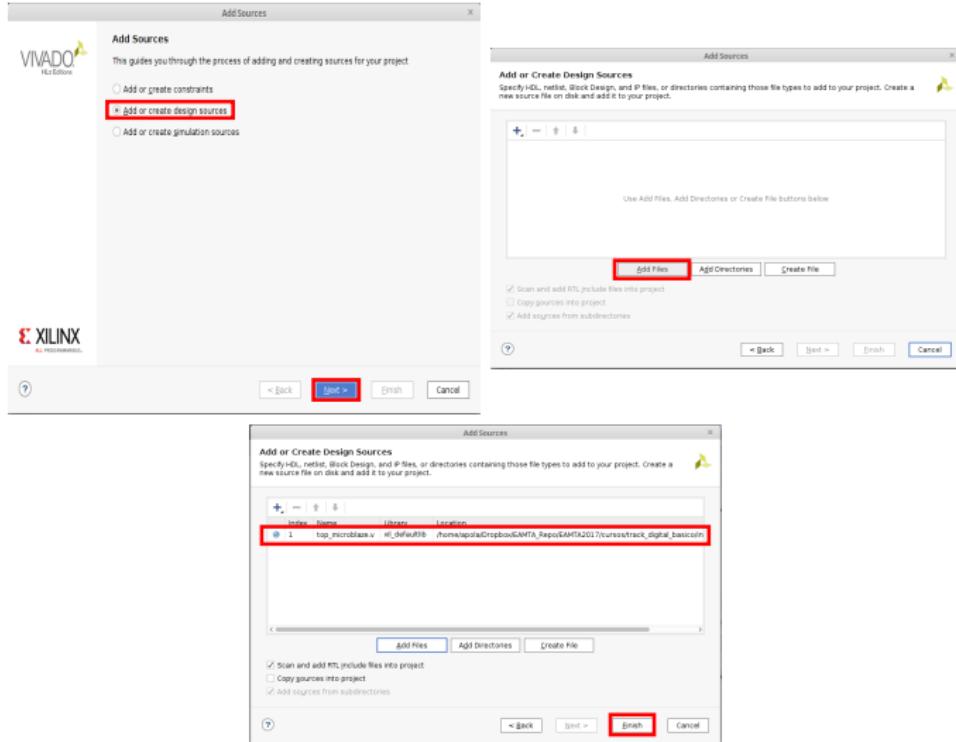
Vivado



Agrega nuevas fuentes a *Design Sources*

Herramienta Vivado

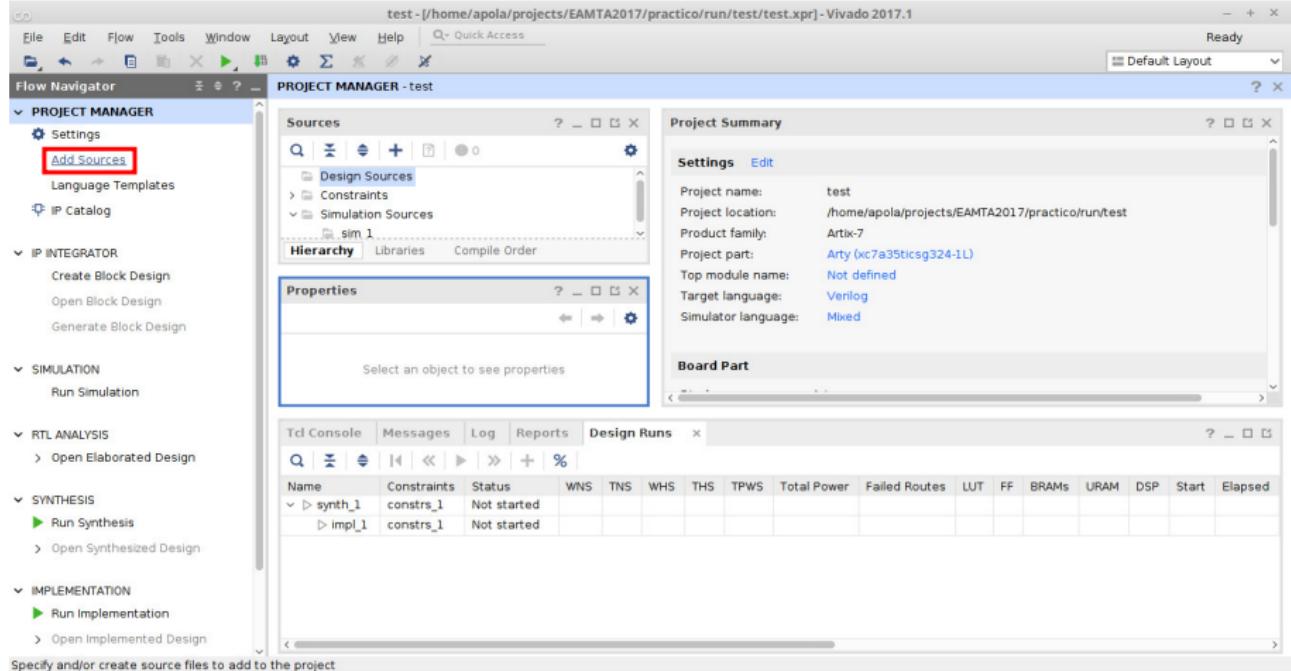
Vivado



Seleccionar todos los archivos verilog relacionados al diseño

Herramienta Vivado

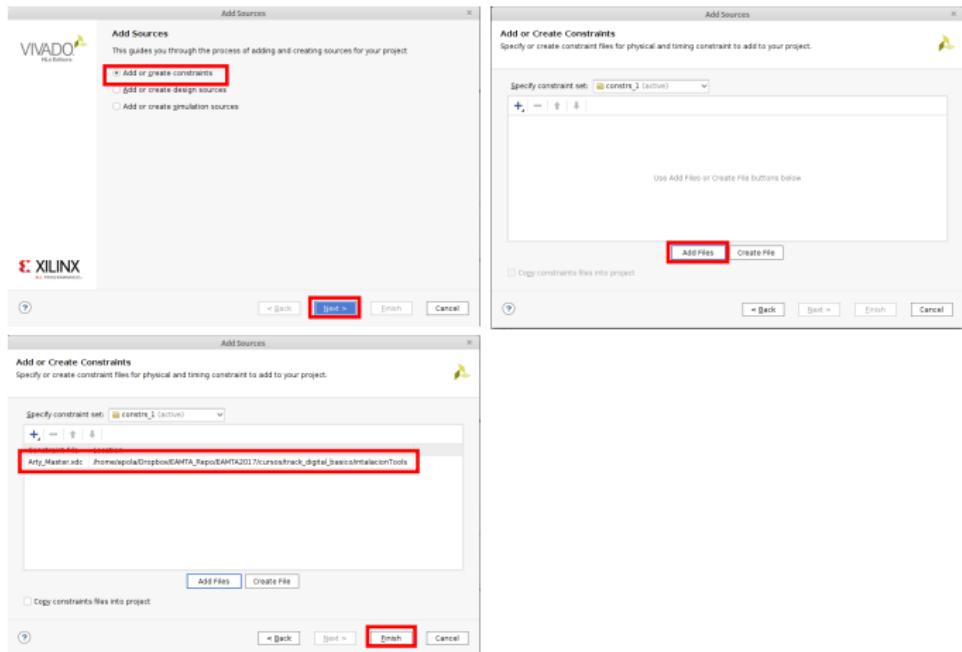
Vivado



Agrega nuevas fuentes a *Constraints*

Herramienta Vivado

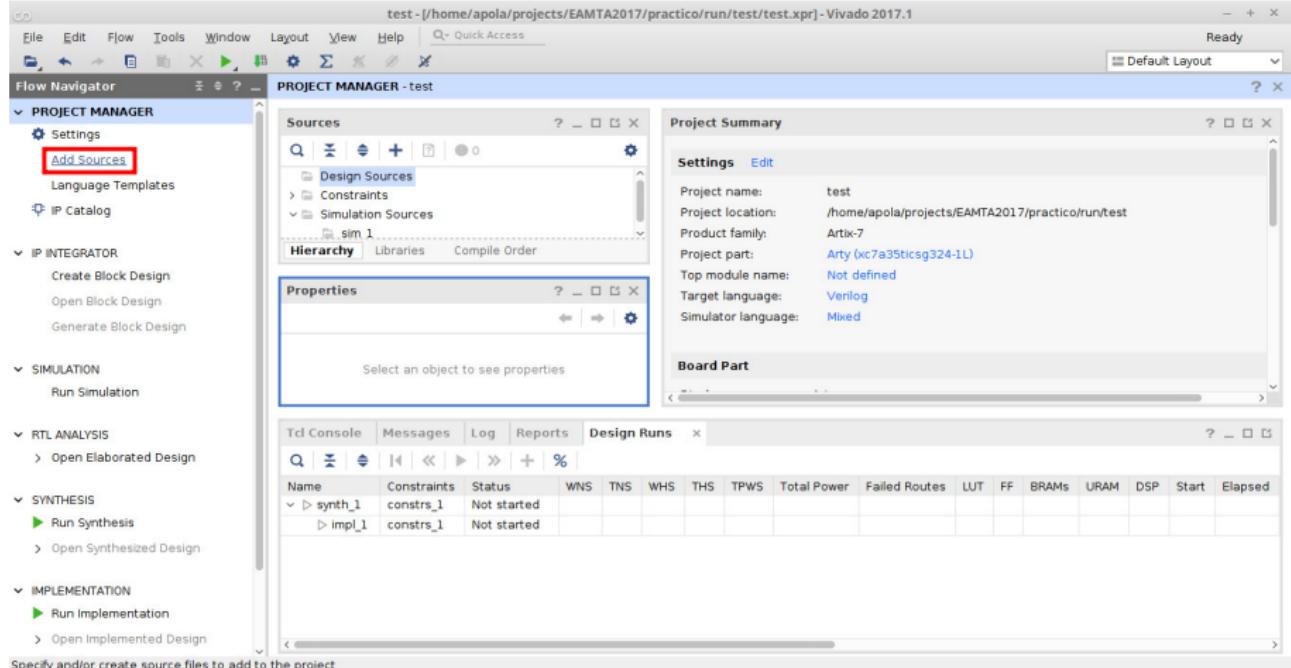
Vivado



Selecciona el archivo xdc

Herramienta Vivado

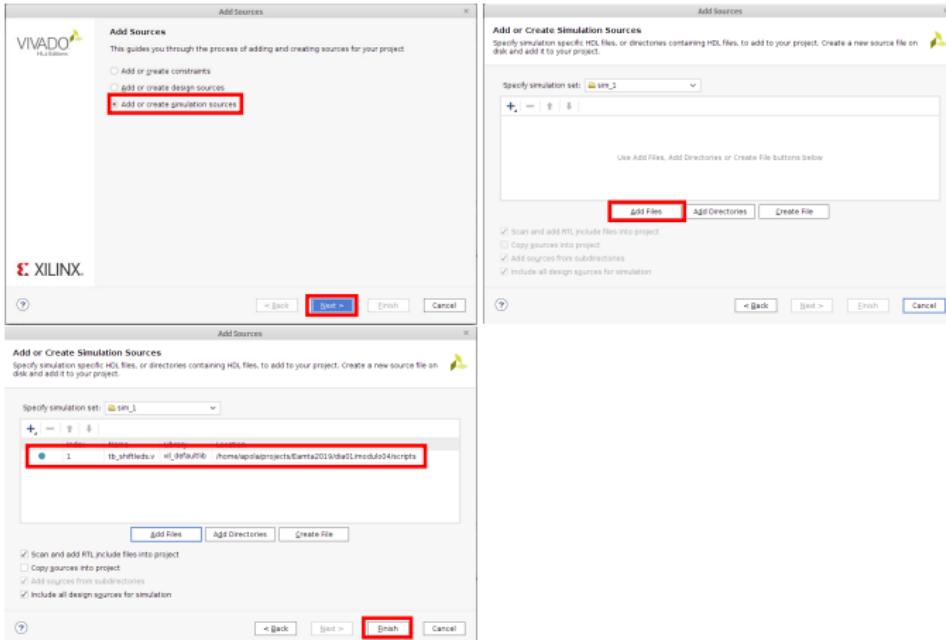
Vivado



Agrega nuevas fuentes a *Simulation Sources*

Herramienta Vivado

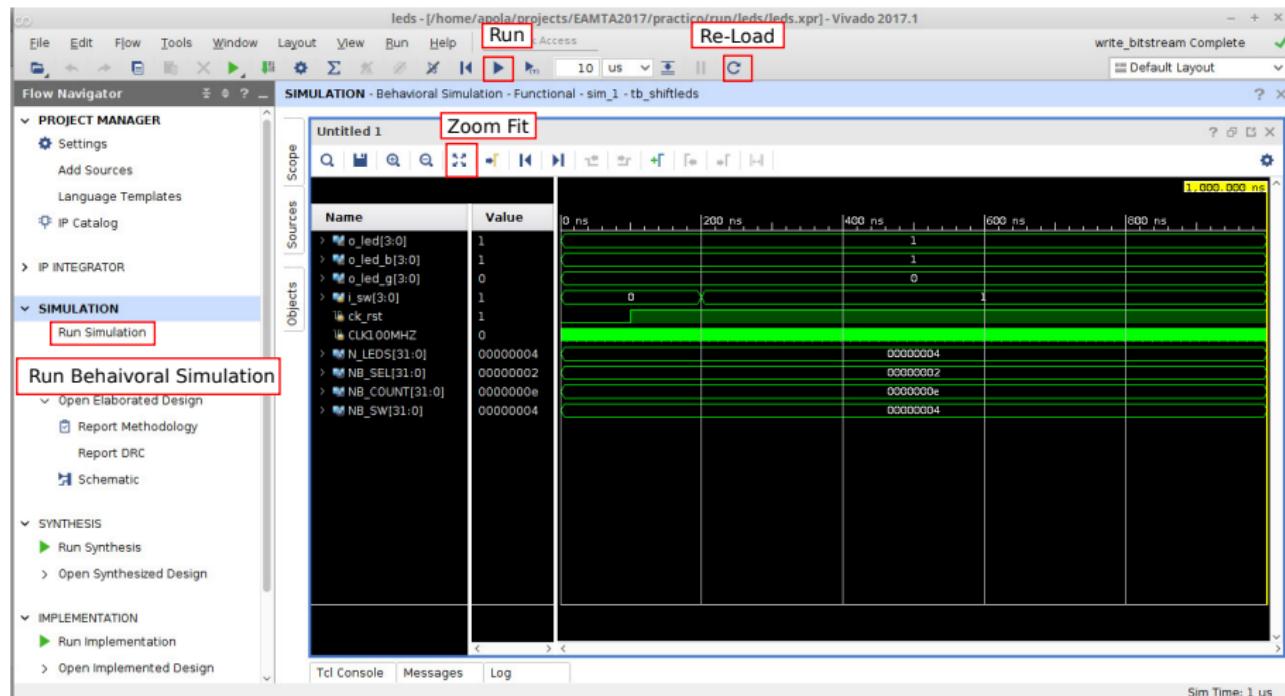
Vivado



Selecciona el archivo verilog para simulación

Herramienta Vivado

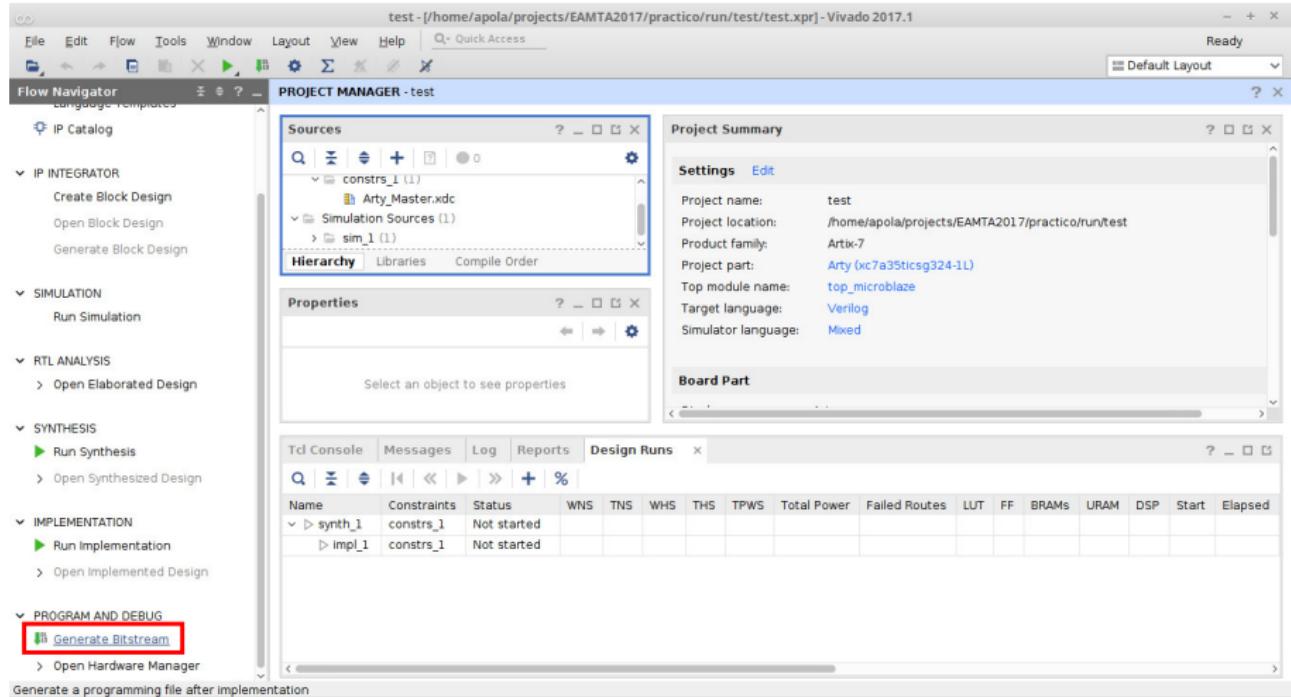
Run Simulation - Run Behavioral Simulation



Simulando el comportamiento del diseño

Herramienta Vivado

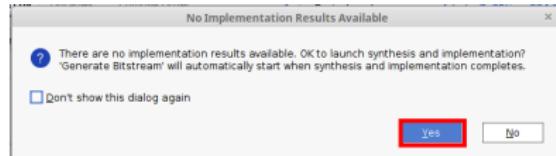
Vivado



Generar el *bitstream*

Herramienta Vivado

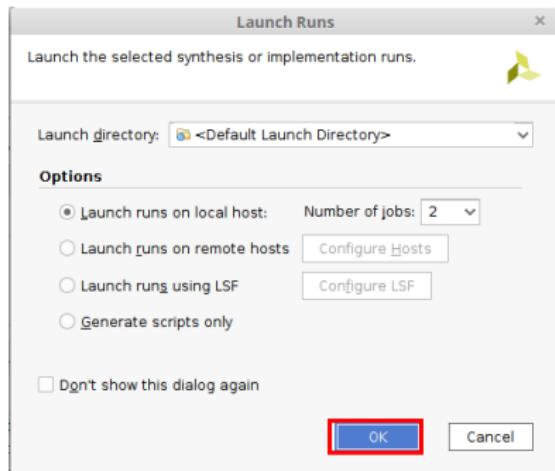
Vivado



Generar el *bitstream*

Herramienta Vivado

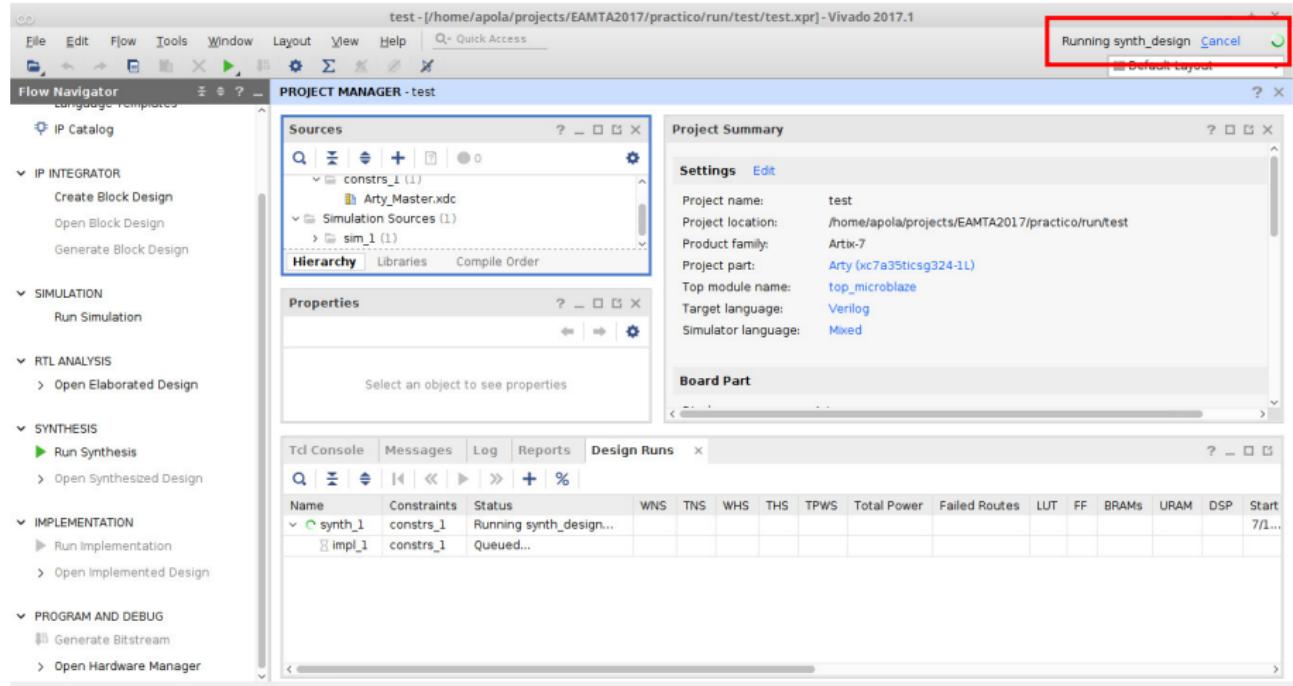
Vivado



Generar el *bitstream*

Herramienta Vivado

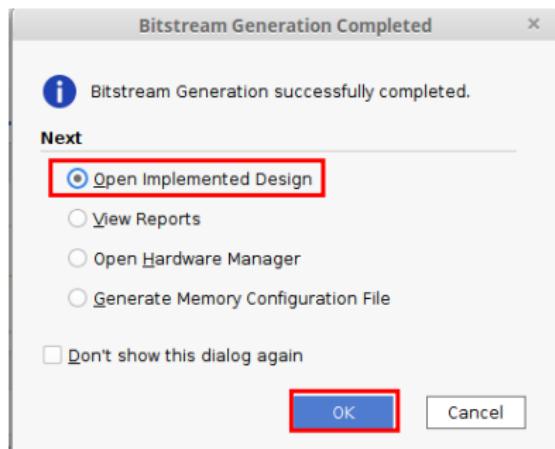
Vivado



Ejecutando tareas de implementación

Herramienta Vivado

Vivado



Abriendo el modelo implementado

Herramienta Vivado

Vivado

test - [/home/apola/projects/EAMTA2017/practico/run/test/test.xpr] - Vivado 2017.1

File Edit Flow Tools Window Layout View Help Q Quick Access

Flow Navigator Language Templates ? ? ?

IMPLEMENTED DESIGN - xc7a35tcssg324-1L (active)

Sources Netlist ? - □ □

IP Catalog

IP INTEGRATOR

- Create Block Design
- Open Block Design
- Generate Block Design

SIMULATION

- Run Simulation

RTL ANALYSIS

- Open Elaborated Design

SYNTHESIS

- Run Synthesis
- Open Synthesized Design

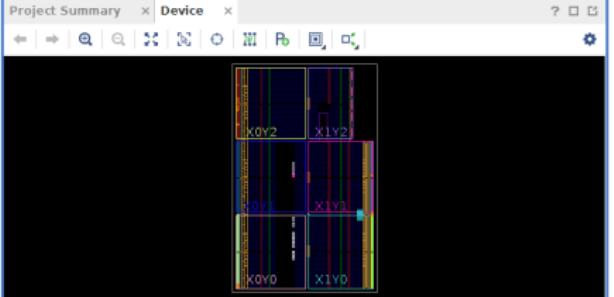
IMPLEMENTATION

- Run Implementation
- Open Implemented Design**
- Constraints Wizard
- Edit Timing Constraints
- Report Timing Summary
- Report Clock Networks

Properties ? - □ □

Select an object to see properties

Project Summary Device



Tcl Console Messages Log Reports Design Runs Power DRC Methodology **Timing** ? - □ □

Design Timing Summary

Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 7.592 ns	Worst Hold Slack (WHS): 0.130 ns	Worst Pulse Width Slack (WPWS):
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack:
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints:
Total Number of Endpoints: 6	Total Number of Endpoints: 6	Total Number of Endpoints:

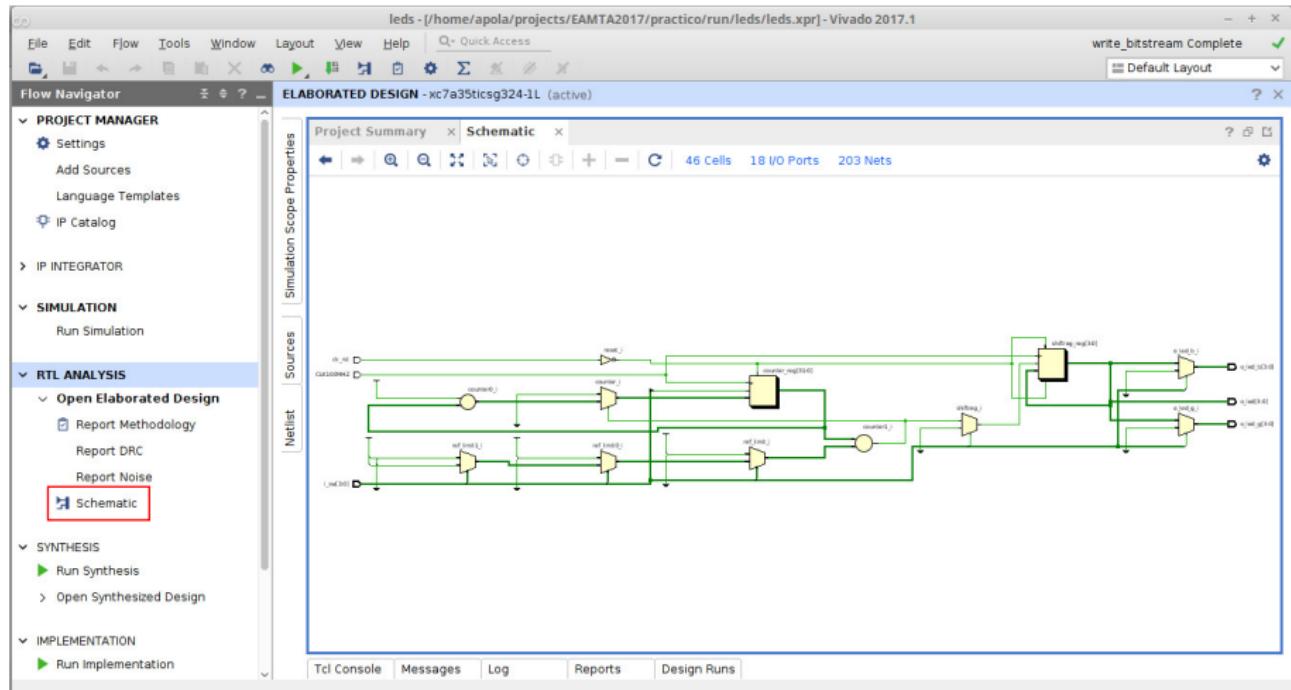
All user specified timing constraints are met.

Timing Summary - impl_1 (saved)

Diseño implementado

Herramienta Vivado

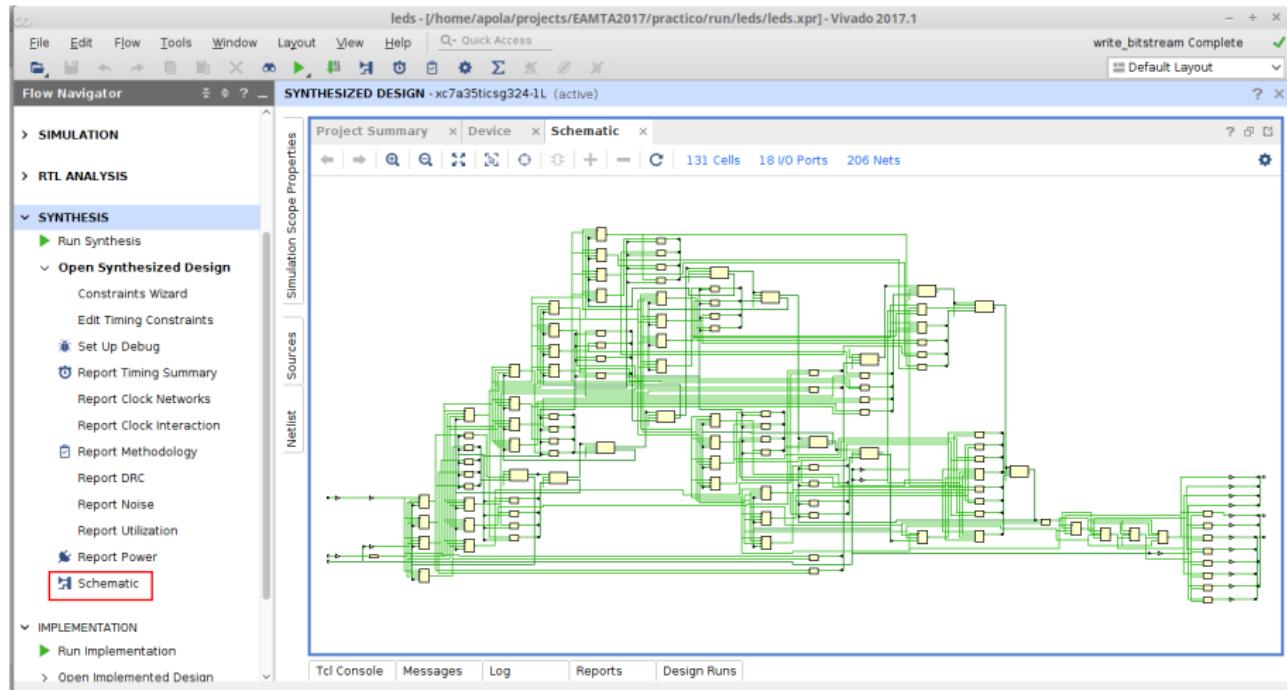
RTL Schematic



Esquemático RTL

Herramienta Vivado

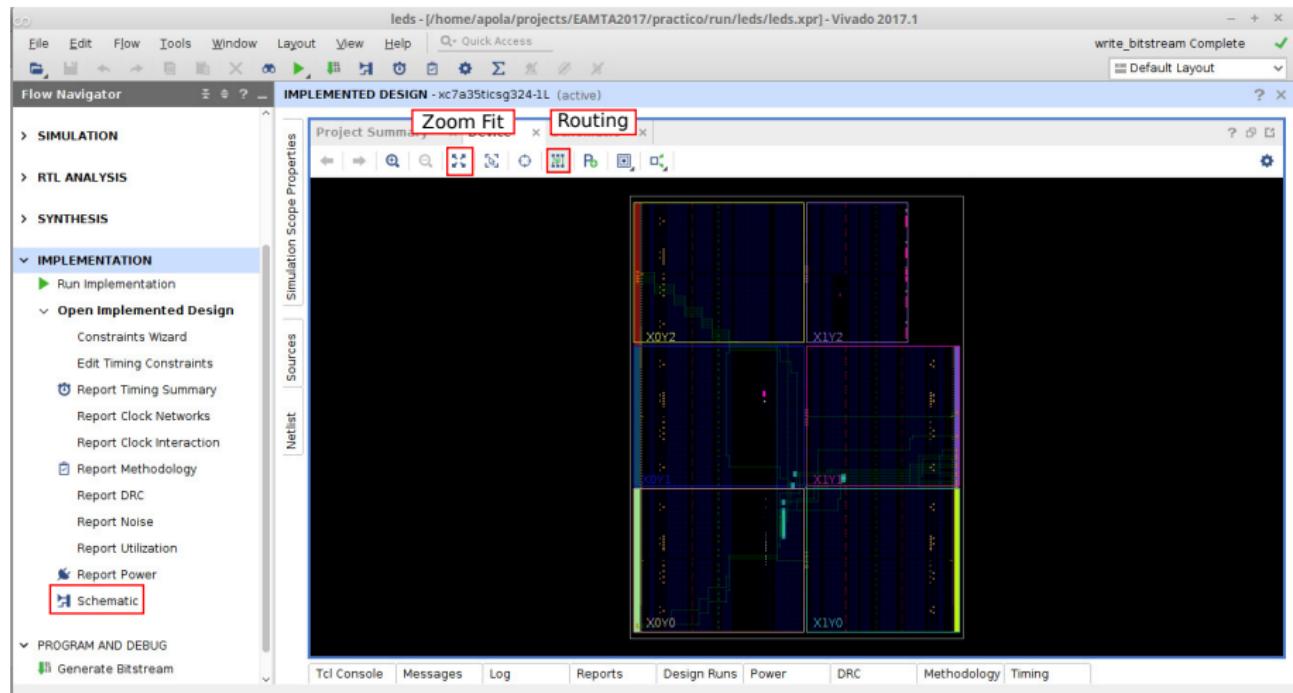
Synthesis - Schematic



Esquemático Implementación

Herramienta Vivado

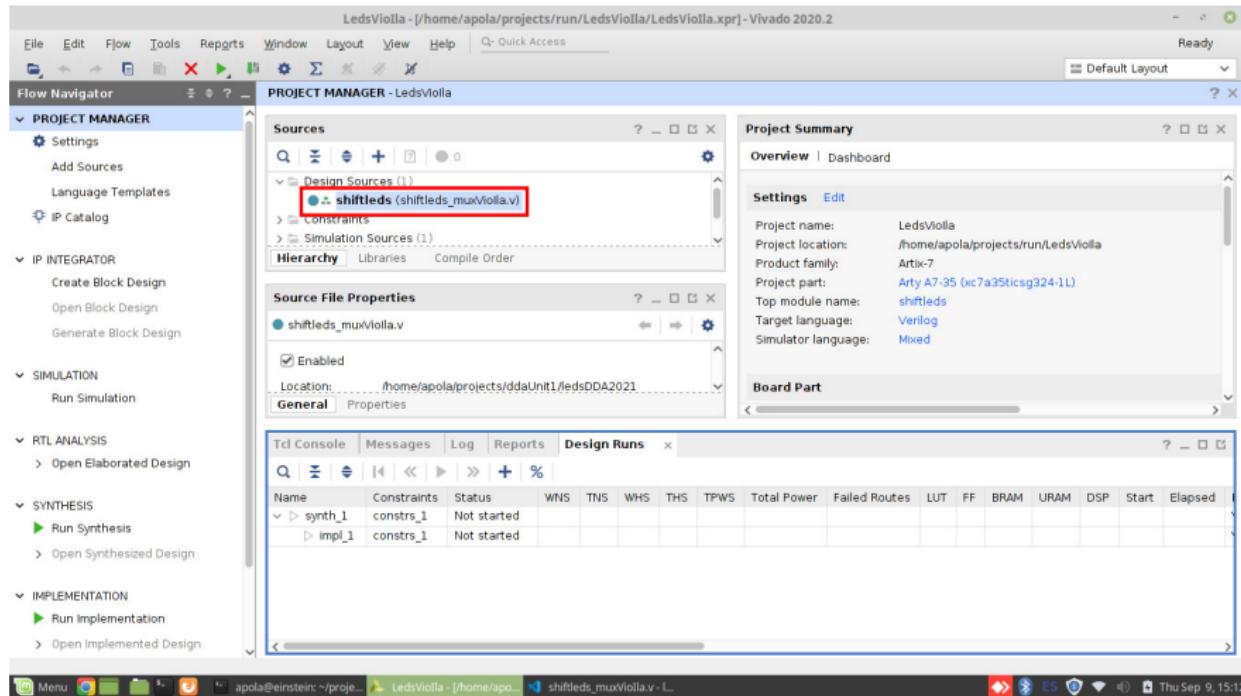
Implementation - Schematic



Implementación

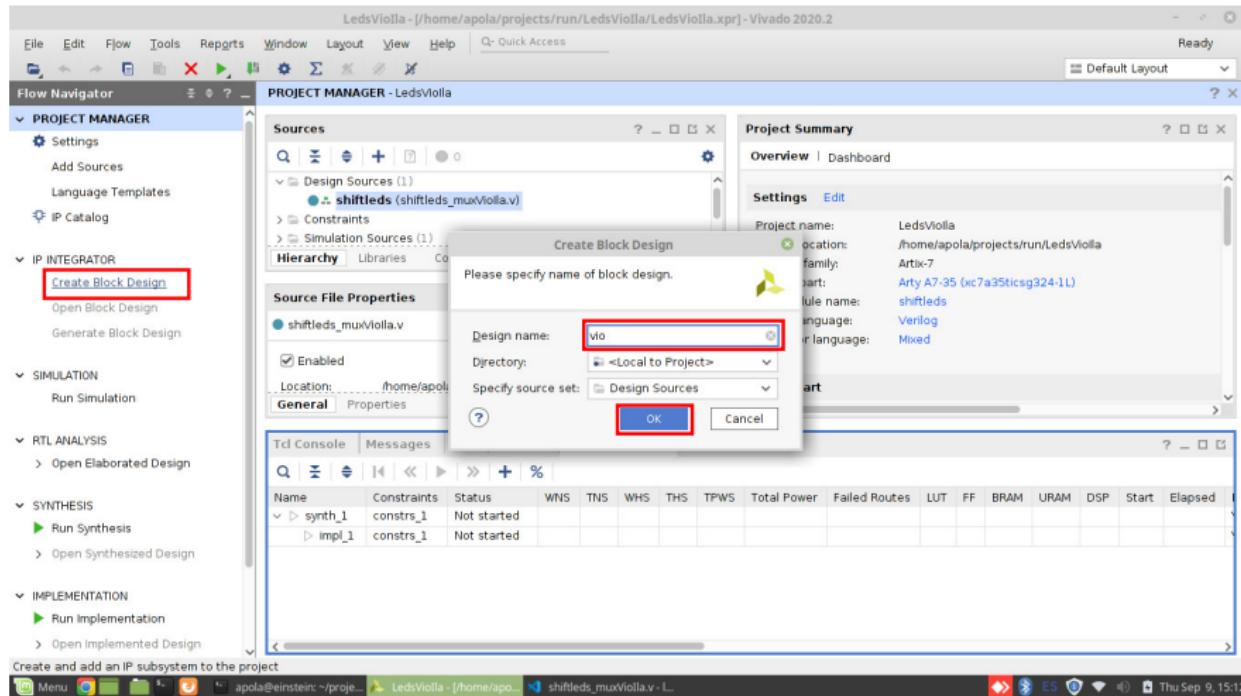
Instancia de VIO e ILA

Instancia de VIO e ILA



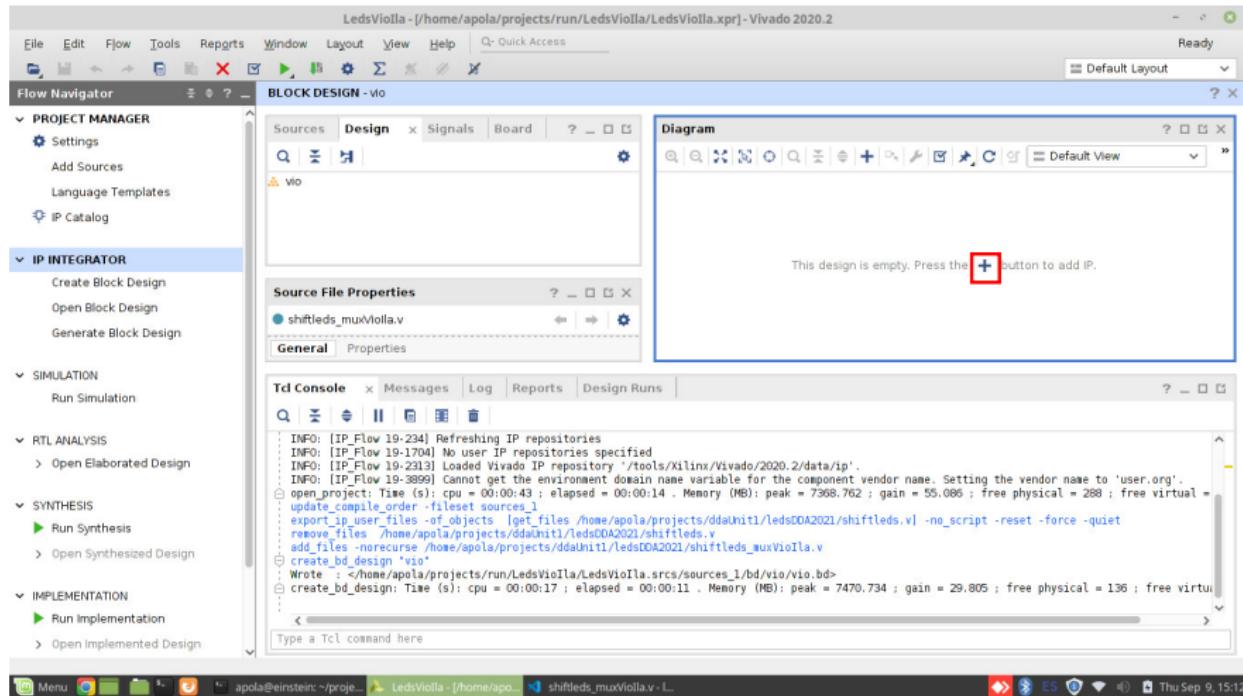
Incluir los archivos **shiftleds_muxViolla.v** y **xdc**.

Instancia de VIO e ILA



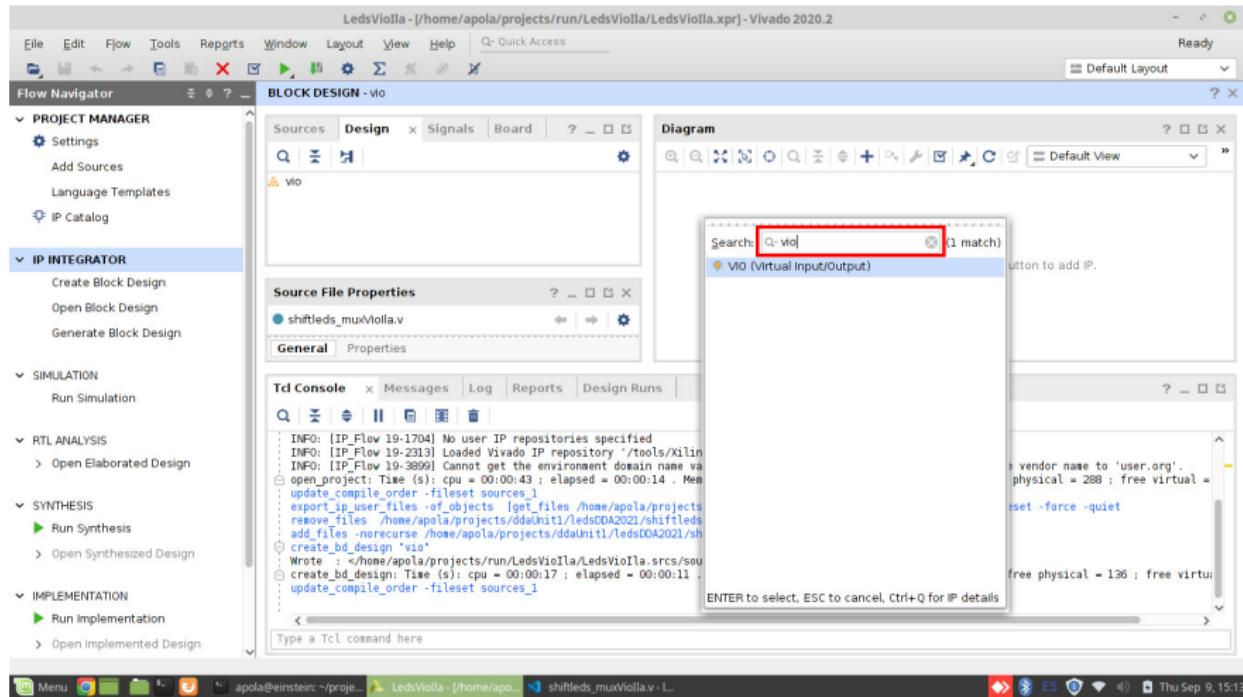
Crear el bloque VIO.

Instancia de VIO e ILA



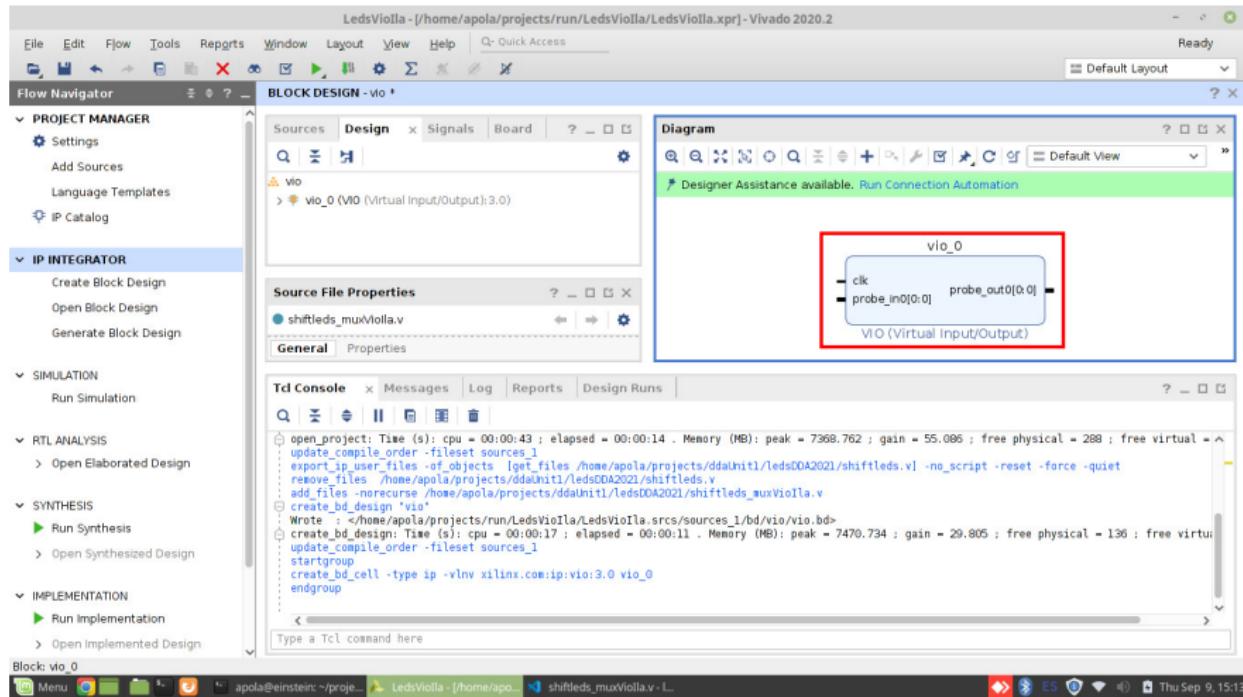
Buscar el IP VIO.

Instancia de VIO e ILA



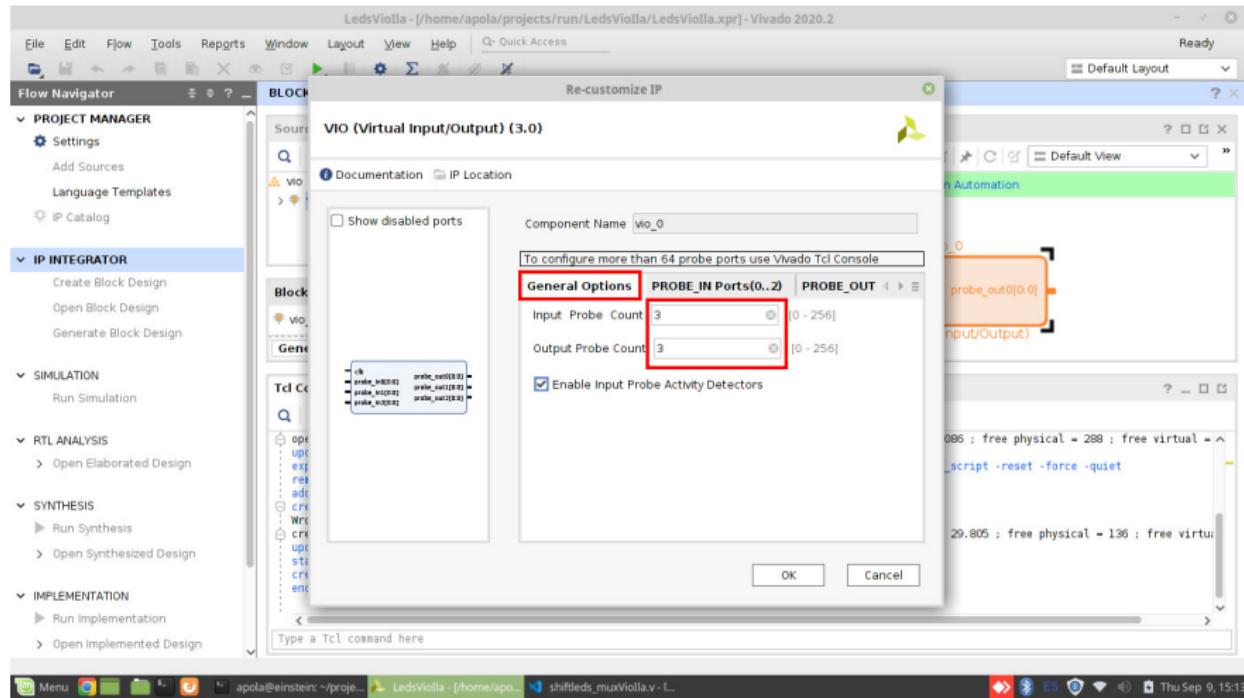
Buscar el IP VIO.

Instancia de VIO e ILA



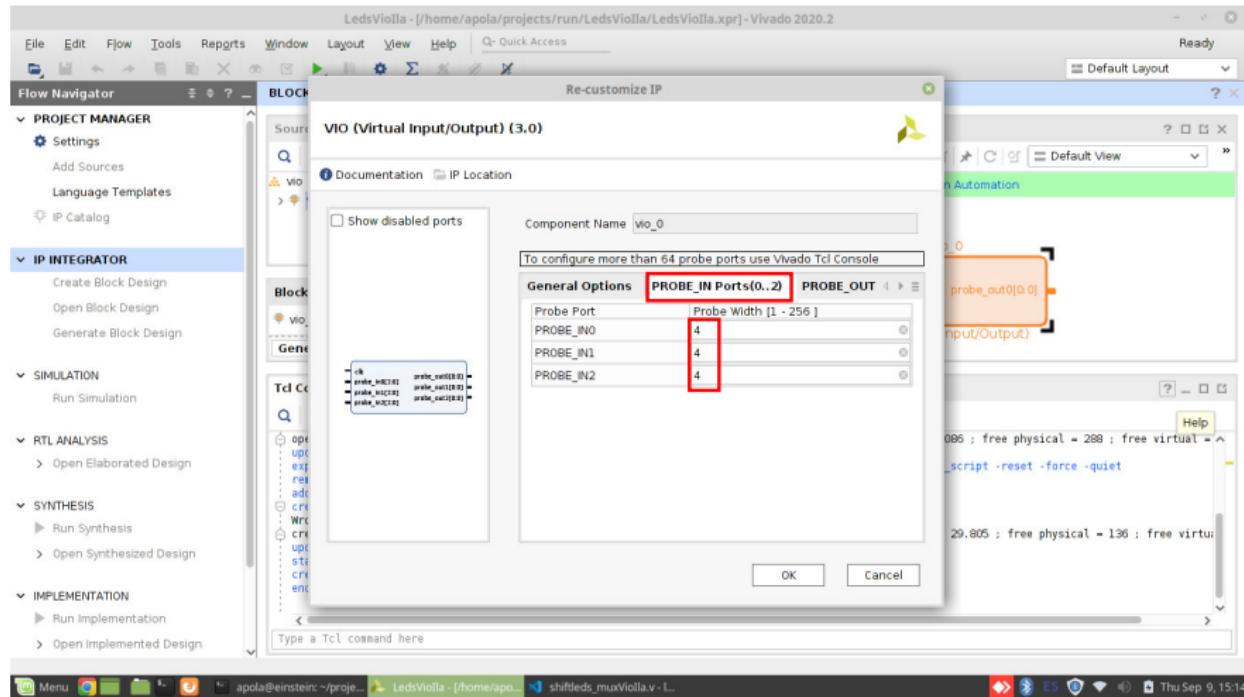
Doble click sobre el IP para entrar a la configuración.

Instancia de VIO e ILA



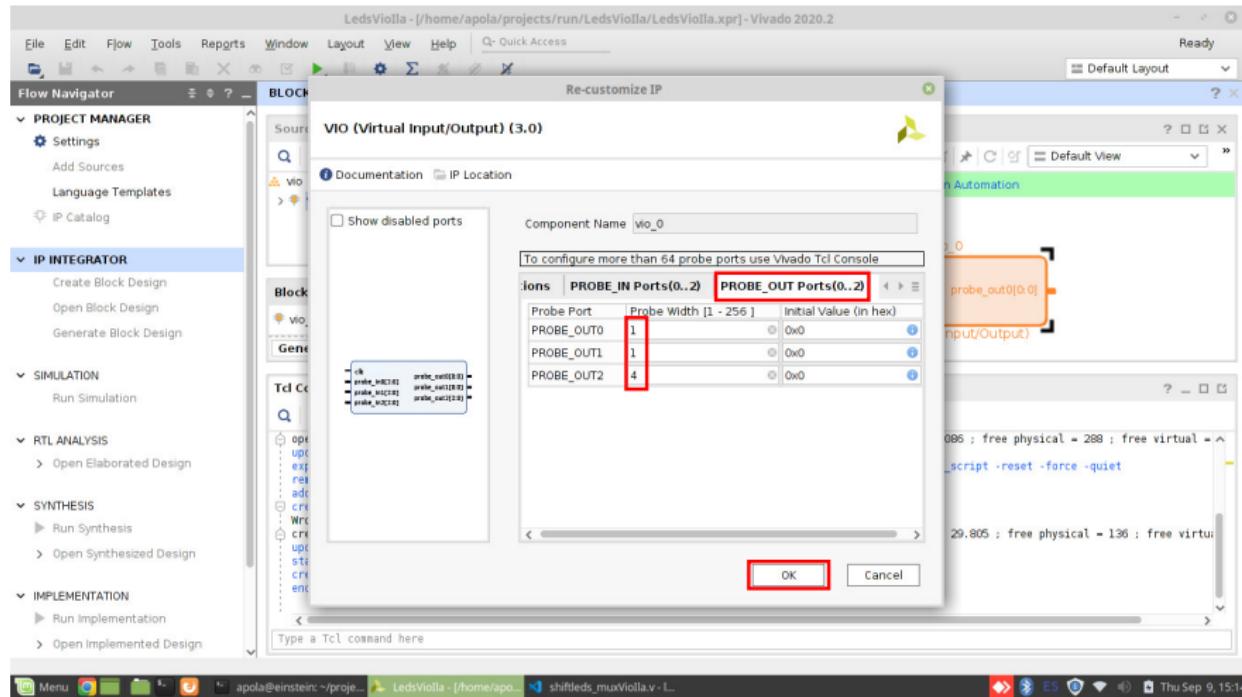
Asignar el número de puertos.

Instancia de VIO e ILA



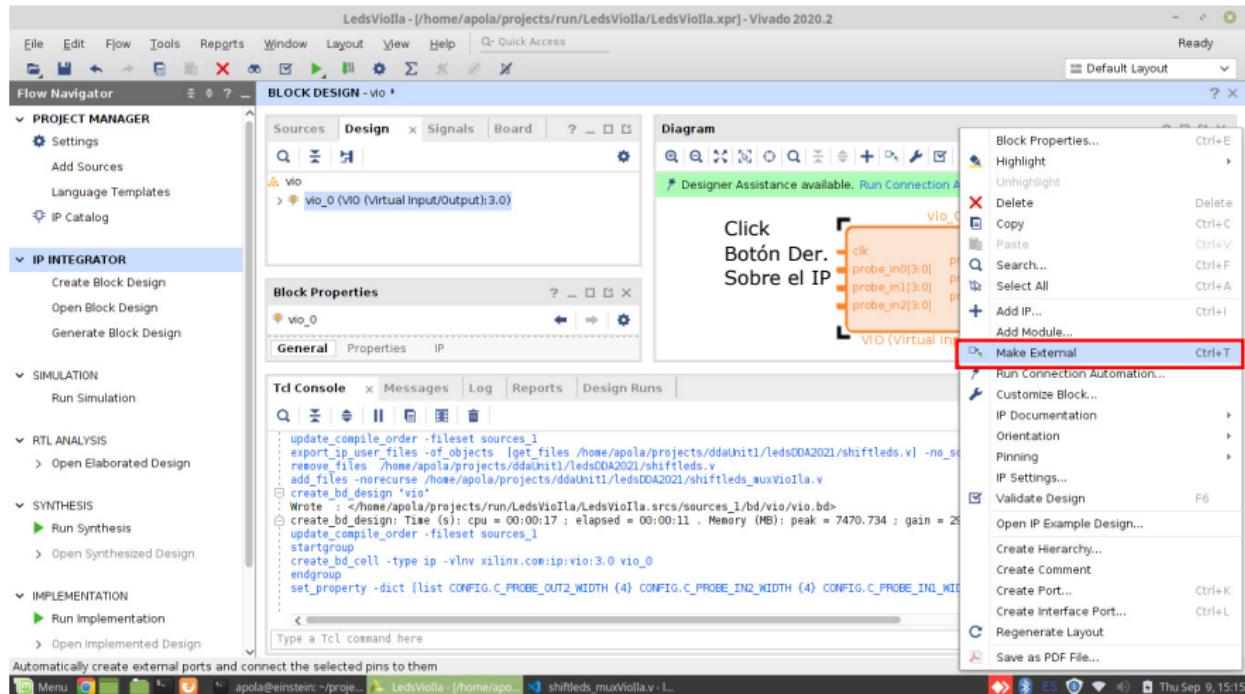
Asignar el número de bits por cada puerto.

Instancia de VIO eILA



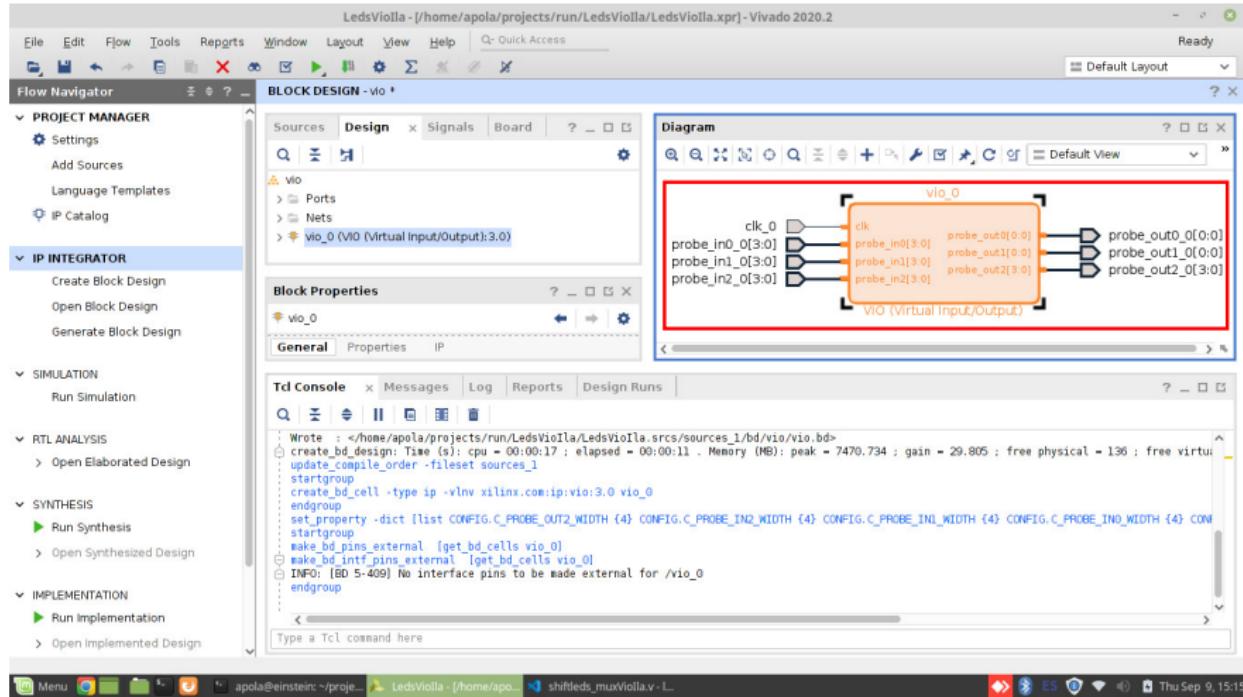
Asignar el número de bits por cada puerto.

Instancia de VIO eILA



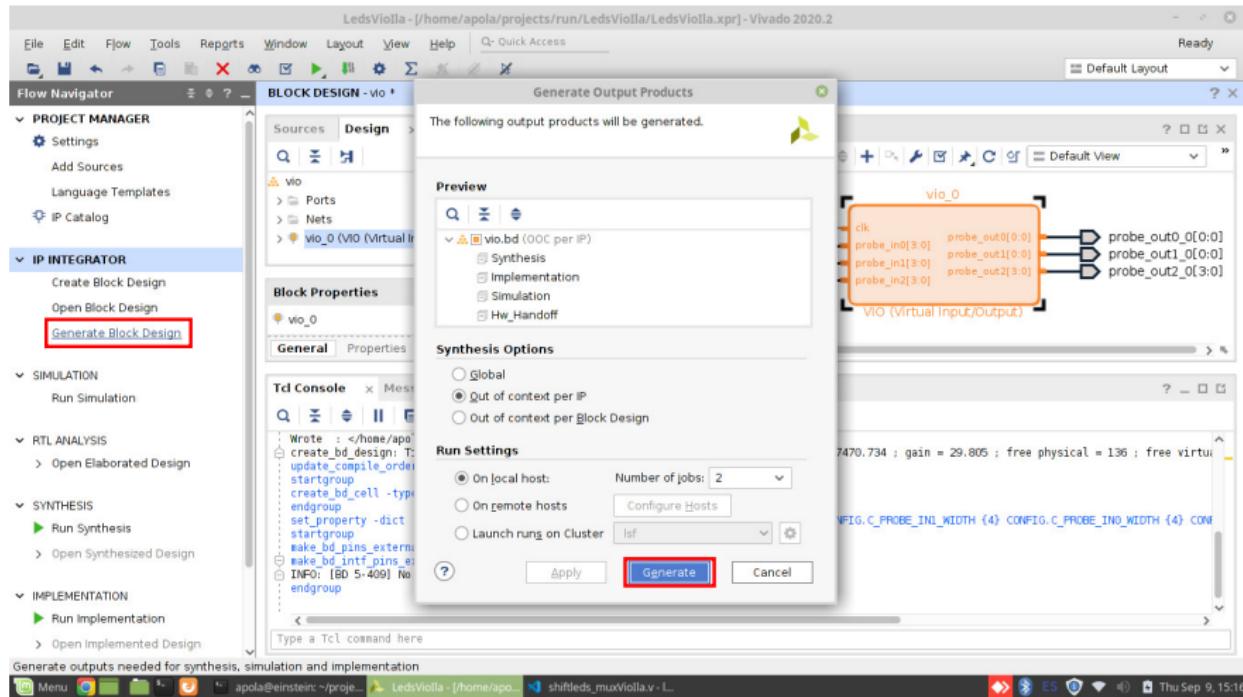
Conectar los puertos.

Instancia de VIO e ILA



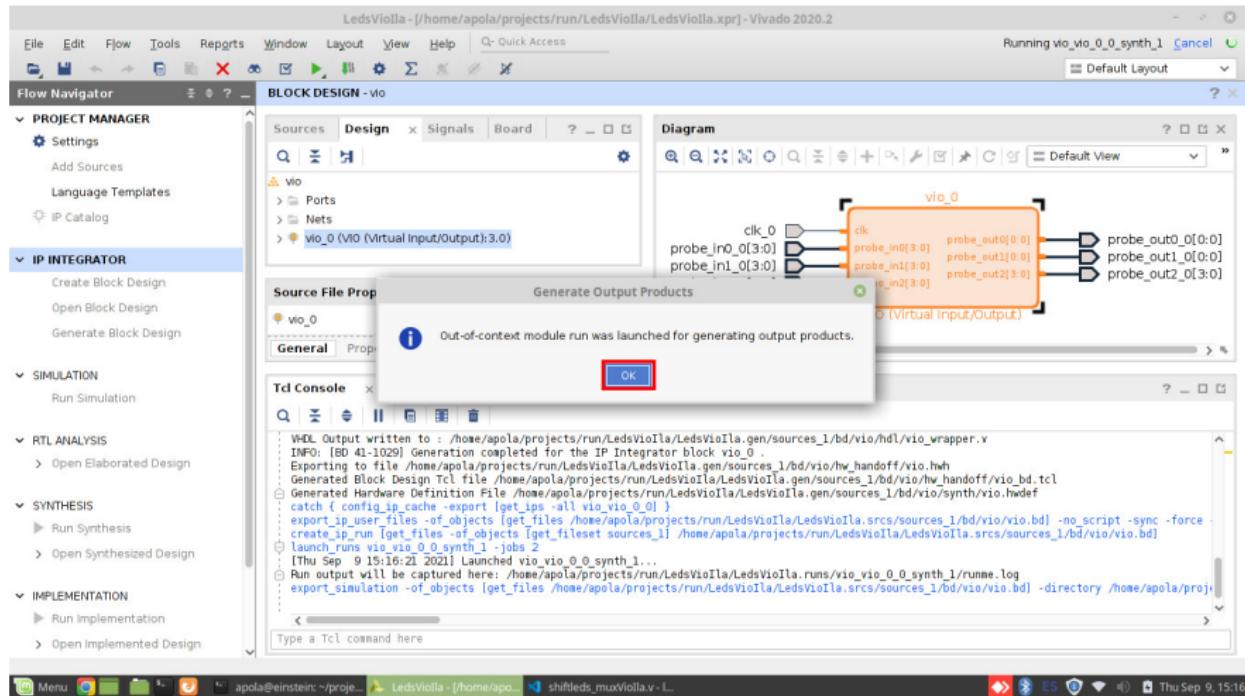
IP conectado.

Instancia de VIO e ILA



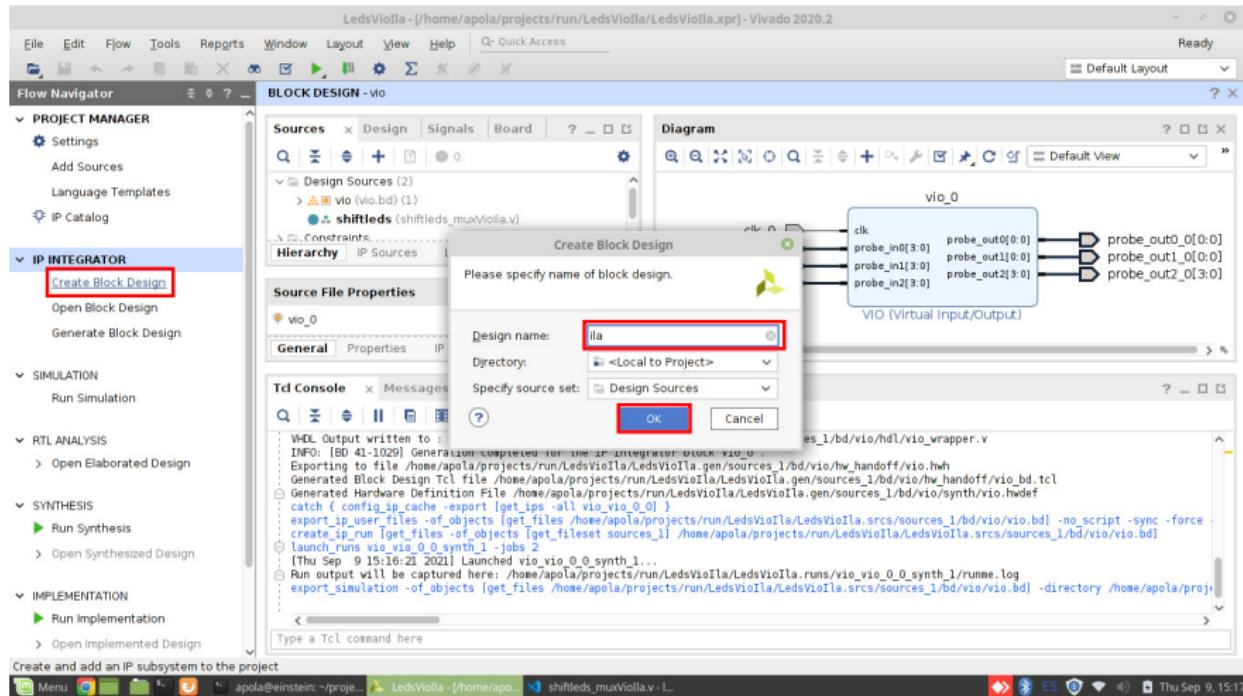
Generar el IP y compilarlo.

Instancia de VIO e ILA



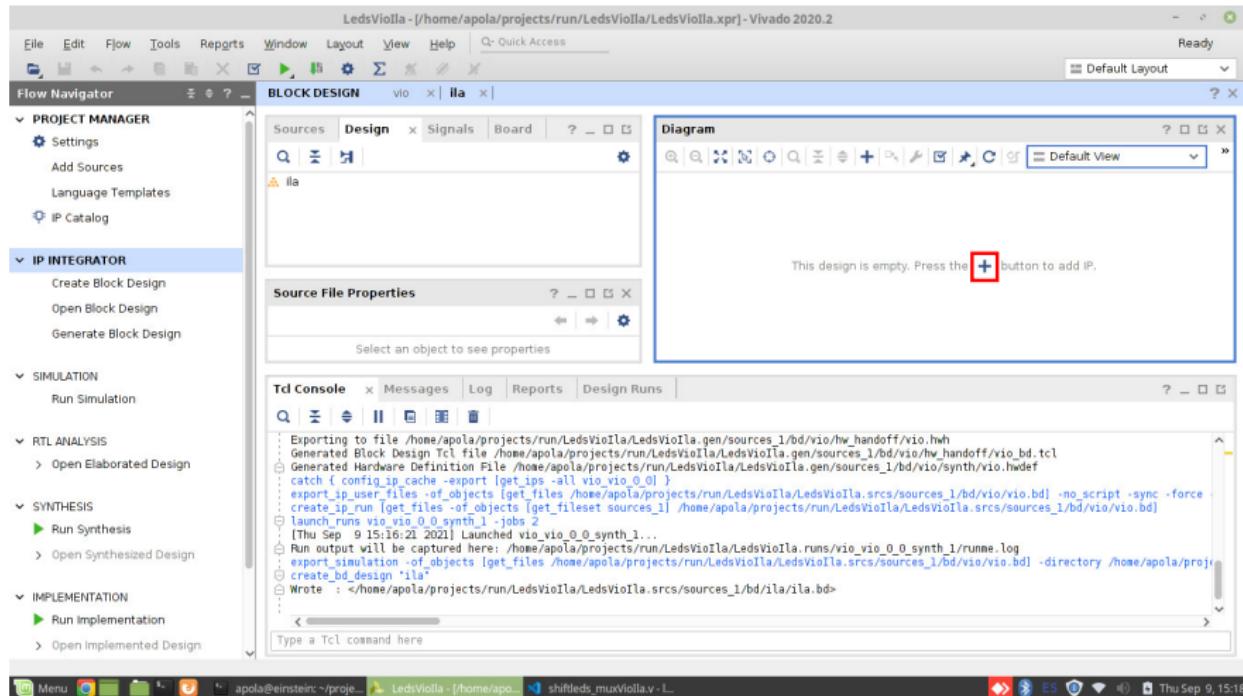
Generar el IP y compilarlo.

Instancia de VIO e ILA



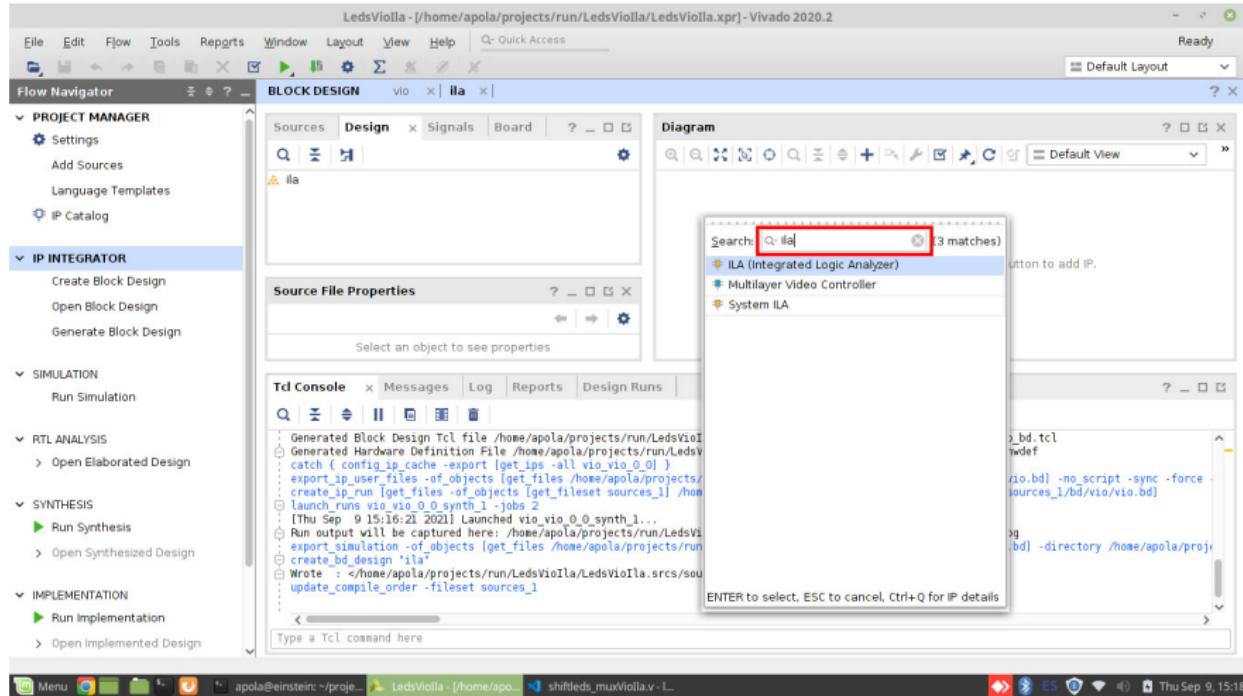
Crear el bloque ILA.

Instancia de VIO e ILA



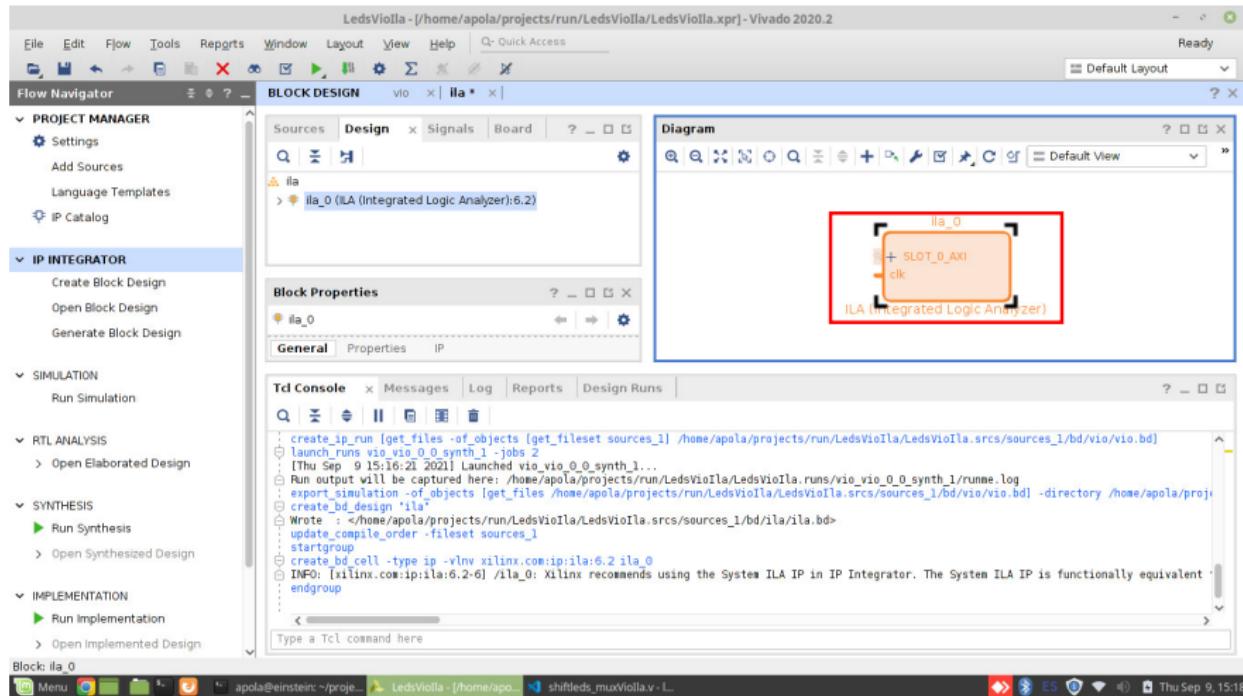
Buscar el IP ILA.

Instancia de VIO e ILA



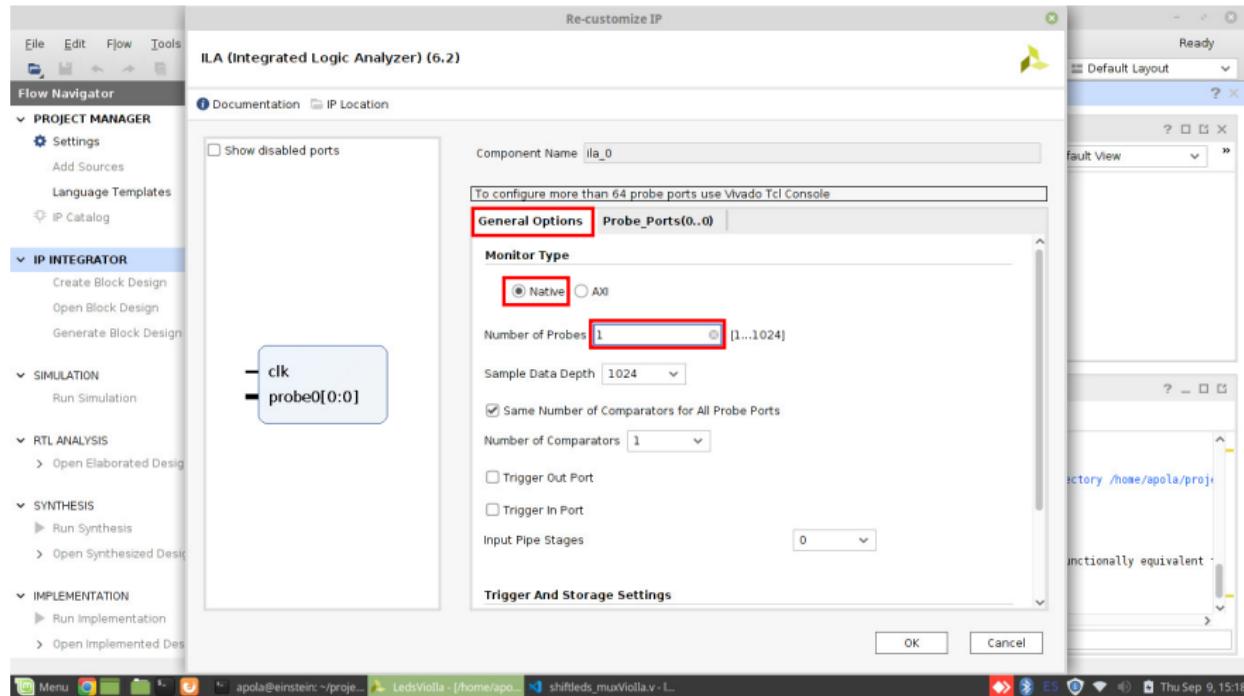
Buscar el IP ILA.

Instancia de VIO e ILA



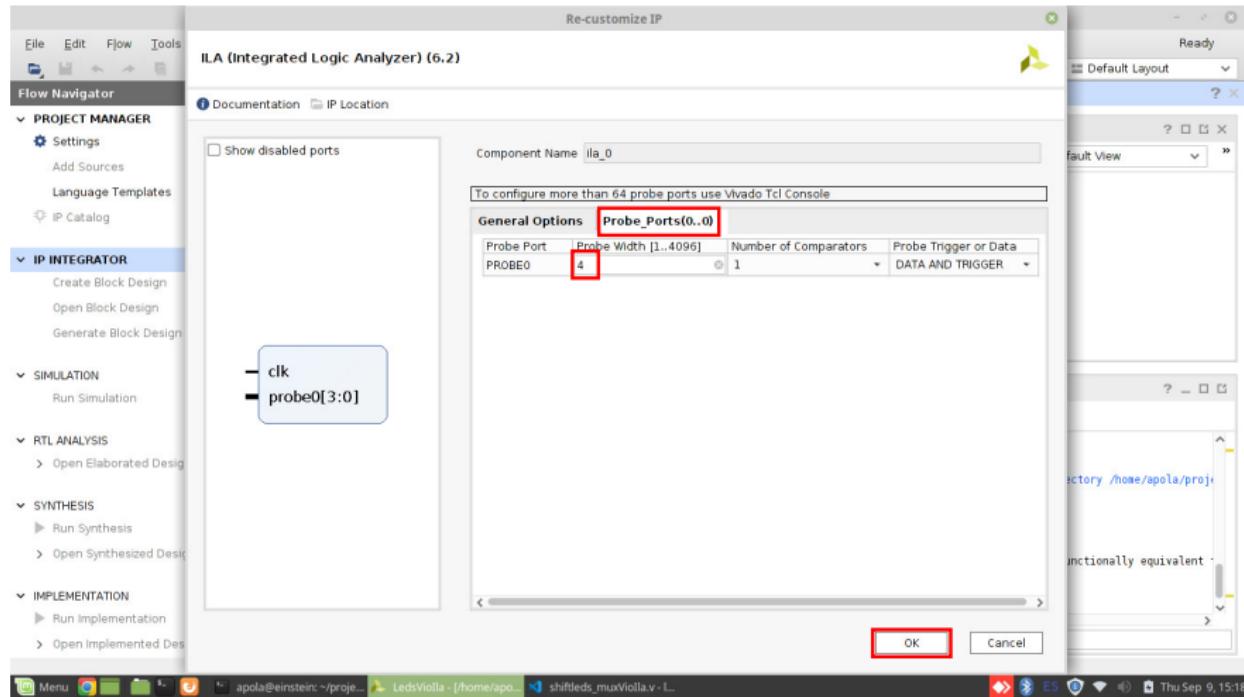
Doble click sobre el IP para entrar a la configuración.

Instancia de VIO eILA



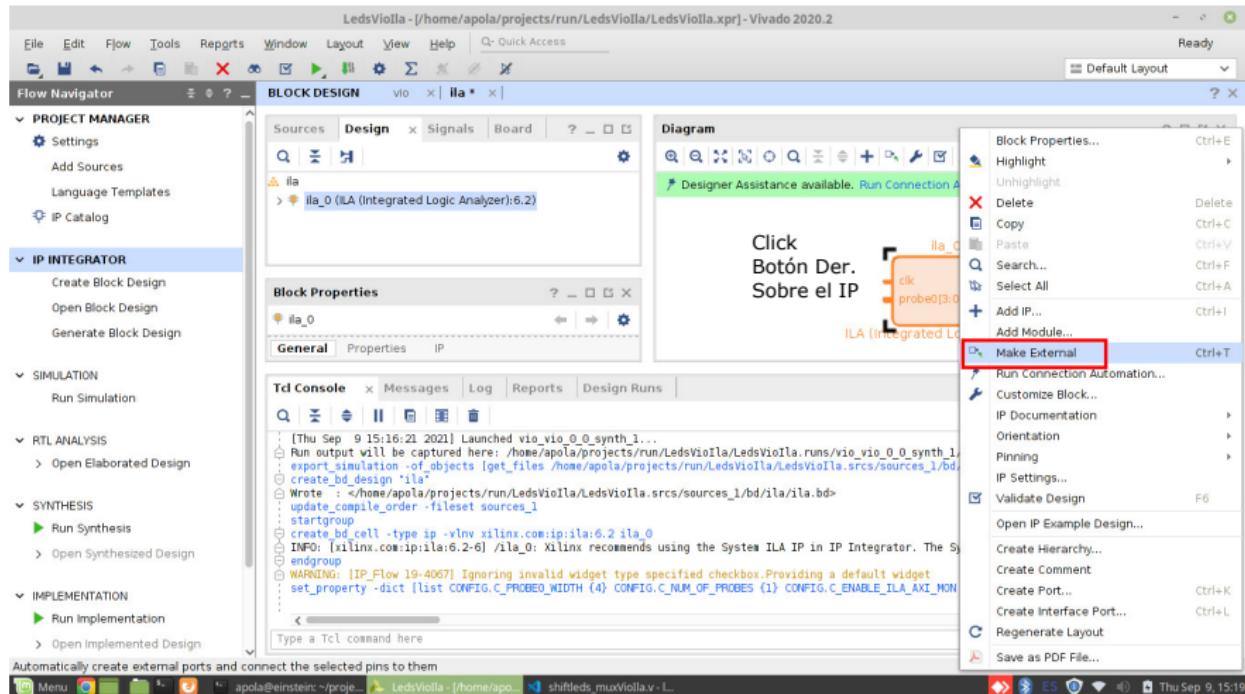
Asignar el tipo de puerto y el número de puntas de prueba.

Instancia de VIO eILA



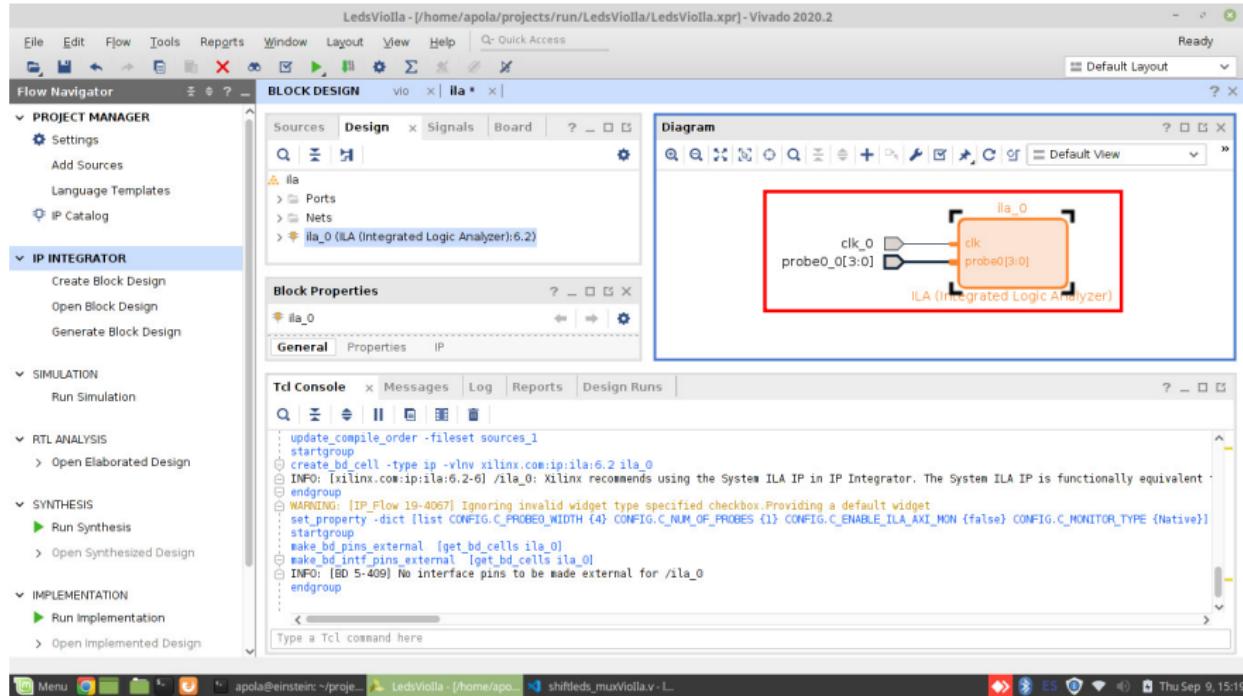
Asignar el número de bits.

Instancia de VIO eILA



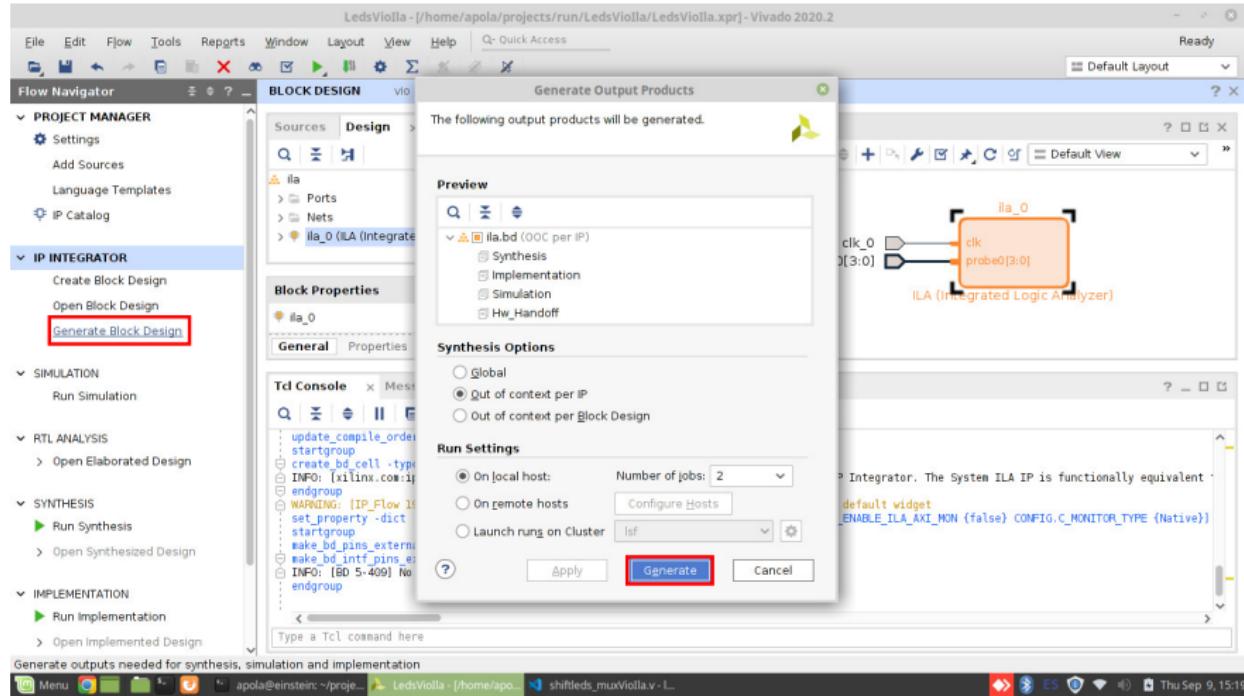
Conectar los puertos.

Instancia de VIO e ILA



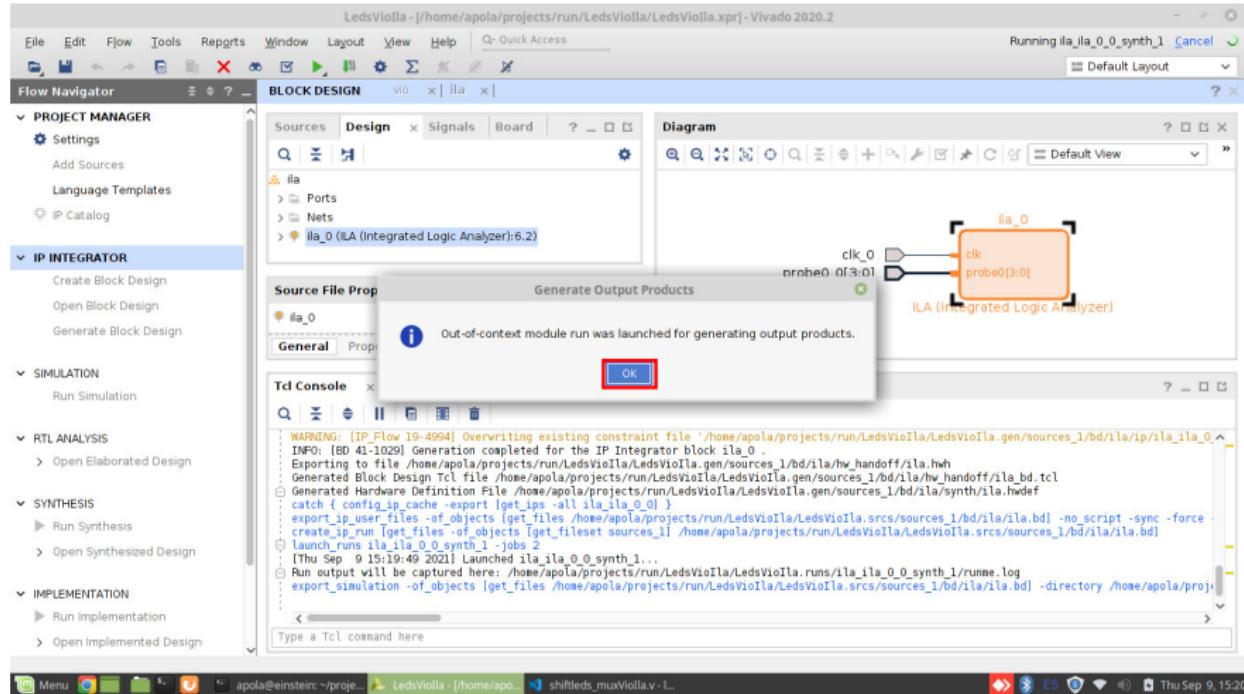
IP conectado.

Instancia de VIO e ILA



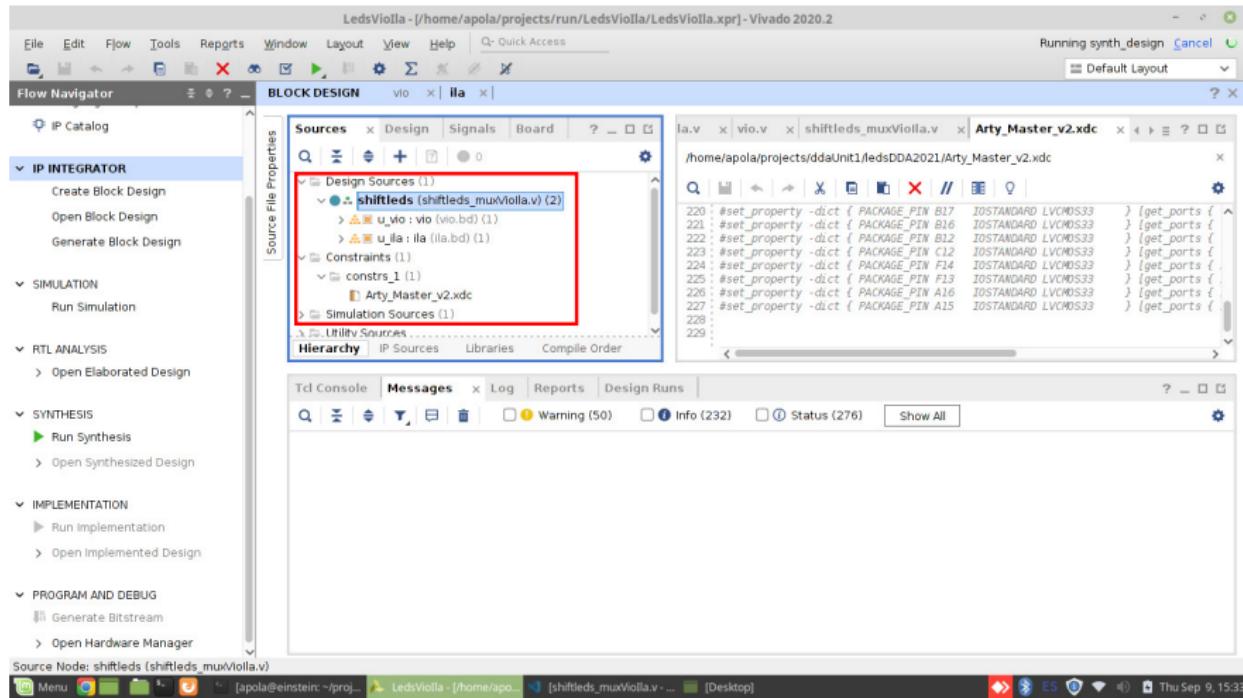
Generar el IP y compilarlo.

Instancia de VIO eILA



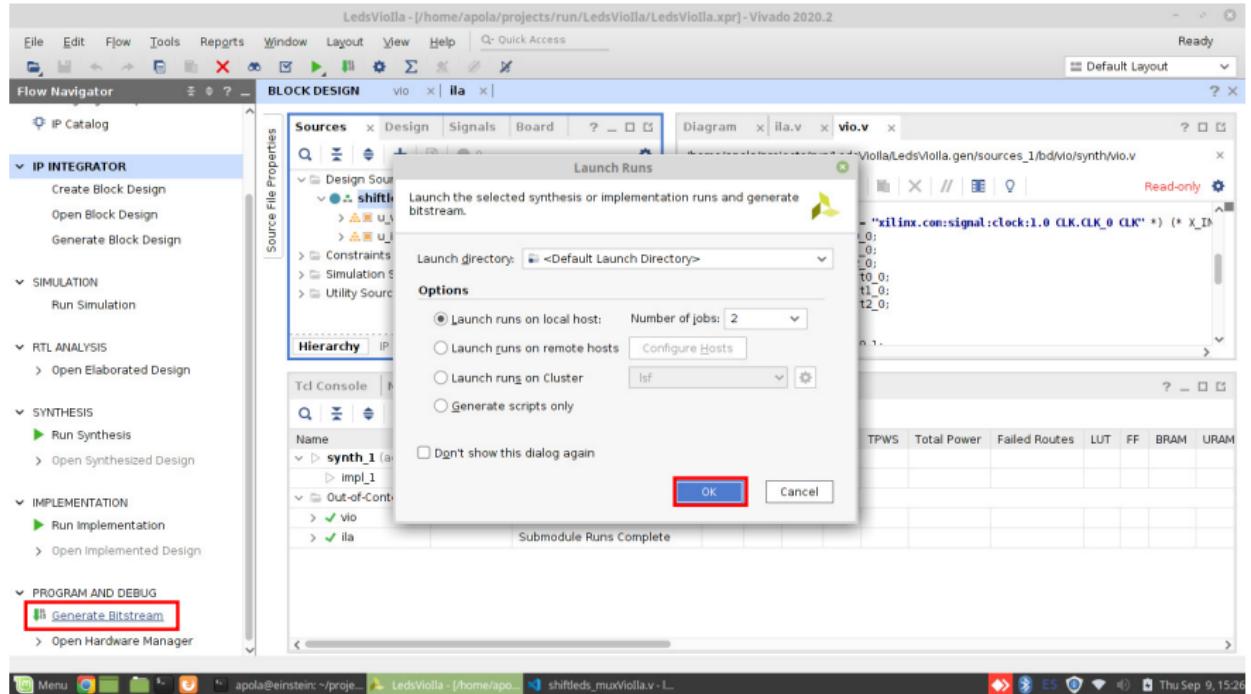
Generar el IP y compilarlo.

Instancia de VIO e ILA



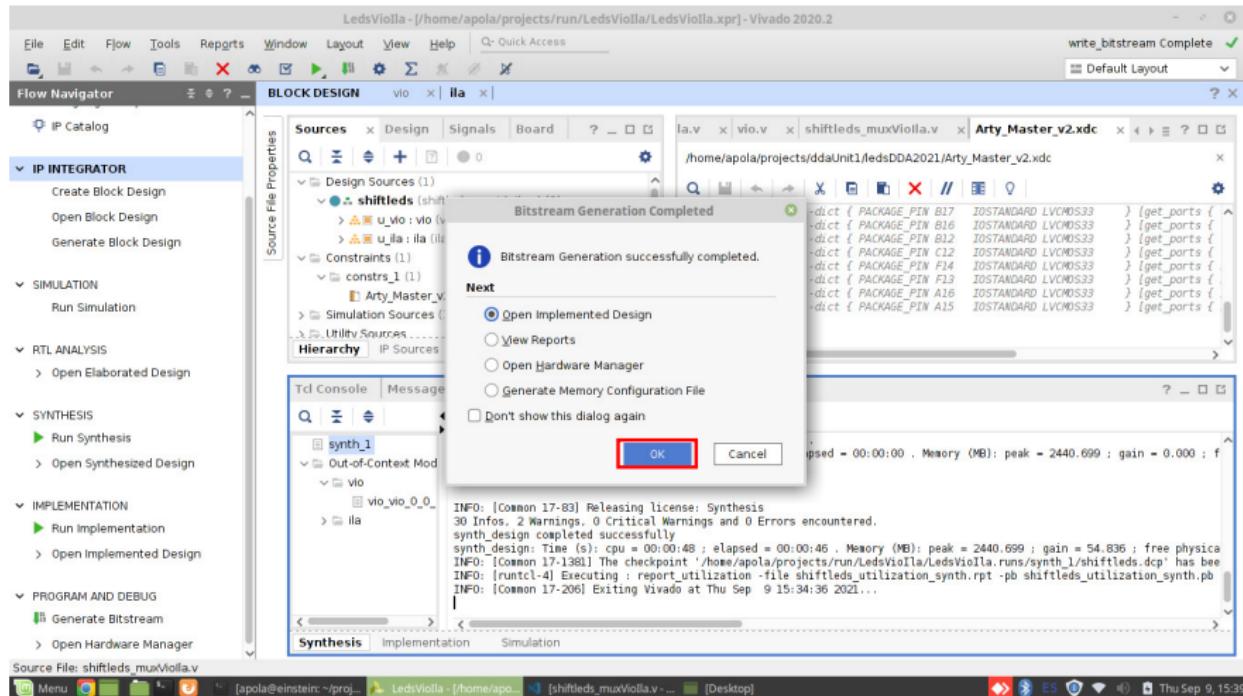
Jerarquía de archivos.

Instancia de VIO e ILA



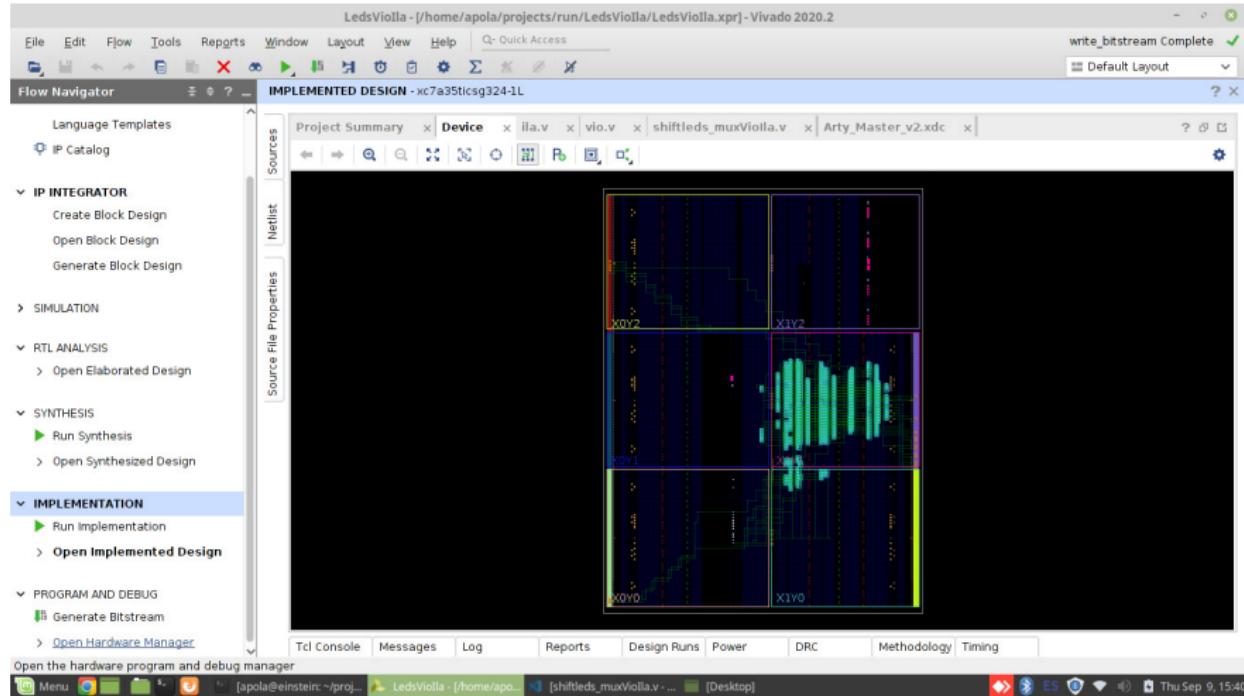
Crear el bitstream.

Instancia de VIO e ILA



Revisar el diseño.

Instancia de VIO e ILA



Revisar el diseño.

Tunel y Asignación de FPGA



Tunel y Asignación de FPGA

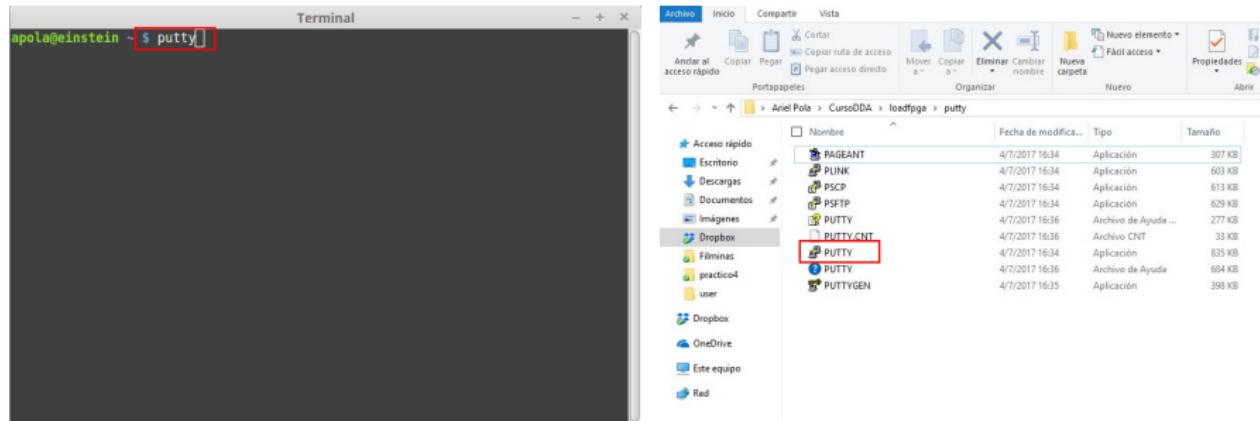
Acceso Remoto a Servidor

- Instalar la herramienta Putty. En el caso de Windows se tendrán los ejecutables descargados desde

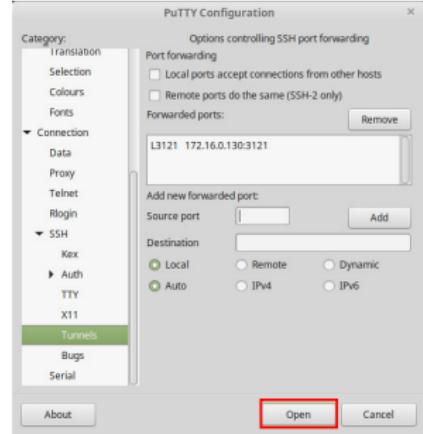
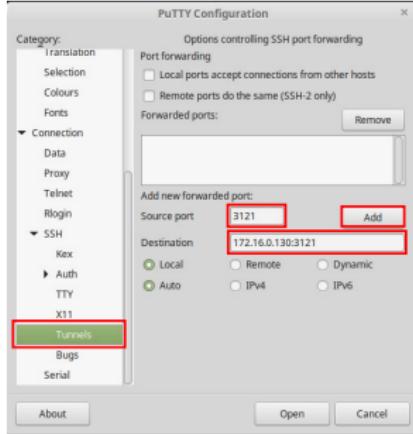
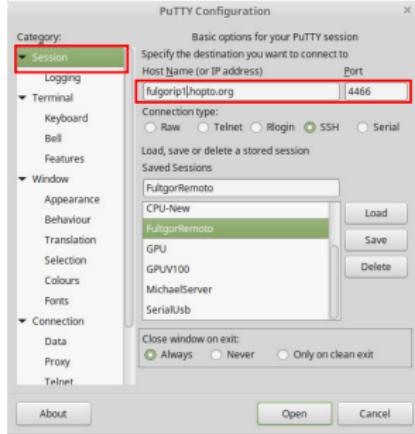
<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>.

- Ejecutar

- **Linux:** En un terminal ejecutar el programa Putty.
- **Windows:** Hacer doble click sobre el ícono de Putty.



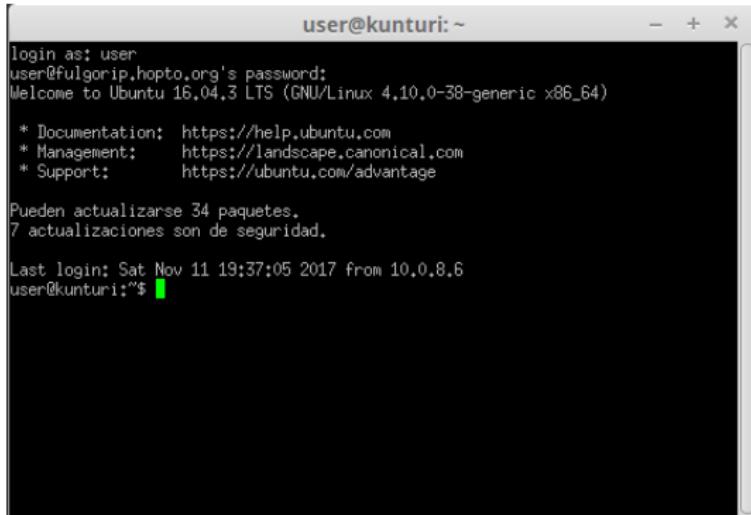
Tunel y Asignación de FPGA



- (a) En la categoría **Session**, setear el host: *fulgorip.hopto.org* o *fulgorip1.hopto.org* y el puerto: *4466*.
- (b) En la categoría **Connection** –> **SSH** –> **Tunnels** incluir el *forwarded* del puerto y agregarlo a la lista.
- (c) Abrir el tunel.

Tunel y Asignación de FPGA

- Para el acceso se solicita usuario (**user**) y contraseña (**Cai1keaw**).
- Este acceso es para todos por igual.



```
user@kunturi:~  
login as: user  
user@fulgorip.hopto.org's password:  
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-38-generic x86_64)  
  
 * Documentation: https://help.ubuntu.com  
 * Management: https://landscape.canonical.com  
 * Support: https://ubuntu.com/advantage  
  
Pueden actualizarse 34 paquetes.  
7 actualizaciones son de seguridad.  
  
Last login: Sat Nov 11 19:37:05 2017 from 10.0.8.6  
user@kunturi:$
```

Tunel y Asignación de FPGA

- Para generar el entorno de trabajo debemos ejecutar la siguiente linea de comando dentro de la carpeta **work_dda** agregando el usuario personal con la inicial del nombre y el apellido completo (Ej. apola).

- 1 *cd work_dda*
- 2 *python ..//Escritorio/scripts/make_user.py*

- El script copia unos archivos y arma el árbol de carpetas.

```
user@kunturi:~/work_dda
login as: user
user@fulgorip.hopto.org's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-38-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

Pueden actualizarse 34 paquetes,
7 actualizaciones son de seguridad.

Last login: Tue Nov 14 13:53:16 2017 from 181.95.86.185
user@kunturi:~$ cd work_dda/
user@kunturi:~/work_dda$ python ..//Escritorio/scripts/make_user.py
Write user (Ex. apola): apola
Copy> client.py
Copy> protocolo_uart_dda.py
Copy> Dec2bin.py
Copy> DSPtools.py
End Process
user@kunturi:~/work_dda$
```

Tunel y Asignación de FPGA

- Acceder a la carpeta <user>/scripts/ y ejecutar **client.py** para solicitar un puerto USB y el ID de la placa FPGA disponible.

1 *cd apola/scripts*

2 *python client.py*

- El script retorna el USB: **USB3** y el ID de la FPGA: **210319A2CECCA**.
- En caso de no haber disponible ninguna placa, el script retorna que los puertos estan ocupados.
- Esta ventana y el script no se tienen que cerrar hasta no terminar de usar la FPGA, ya que se libera el puerto y dará acceso a otro usuario.
- En las siguientes figuras se observa el caso de asignación correcta y puertos ocupados.

Nota: Copia de archivos o carpetas

- Usamos el comando **pscp** para copiar archivos entre sesiones.

- Remoto a local

```
pscp -P 4466 user@fulgorip1.hopto.org://home/user/work_dda/<user>/<file> ./
```

- Local a remoto

```
pscp -P 4466 ./<file> user@fulgorip1.hopto.org://home/user/work_dda/<user>/
```

Tunel y Asignación de FPGA

```
user@kunturi:~/work_dda/apola/scripts
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage

Pueden actualizarse 34 paquetes,
7 actualizaciones son de seguridad.

Last login: Tue Nov 14 14:03:43 2017 from 181.95.86.185
user@kunturi:~$ cd work_dda/
user@kunturi:~/work_dda$ cd apola/scripts/
user@kunturi:~/work_dda/apola/scripts$ python client.py

Write-> Exit <-to close the session
Ip: 127.0.0.1
Port: 5005
Checking Free Port
USB Free: USB3-210319A2CECCA
Write Text to check Session or Exit to close: [ ]
```

```
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-38-generic x86_64)

* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage

Pueden actualizarse 34 paquetes,
7 actualizaciones son de seguridad.

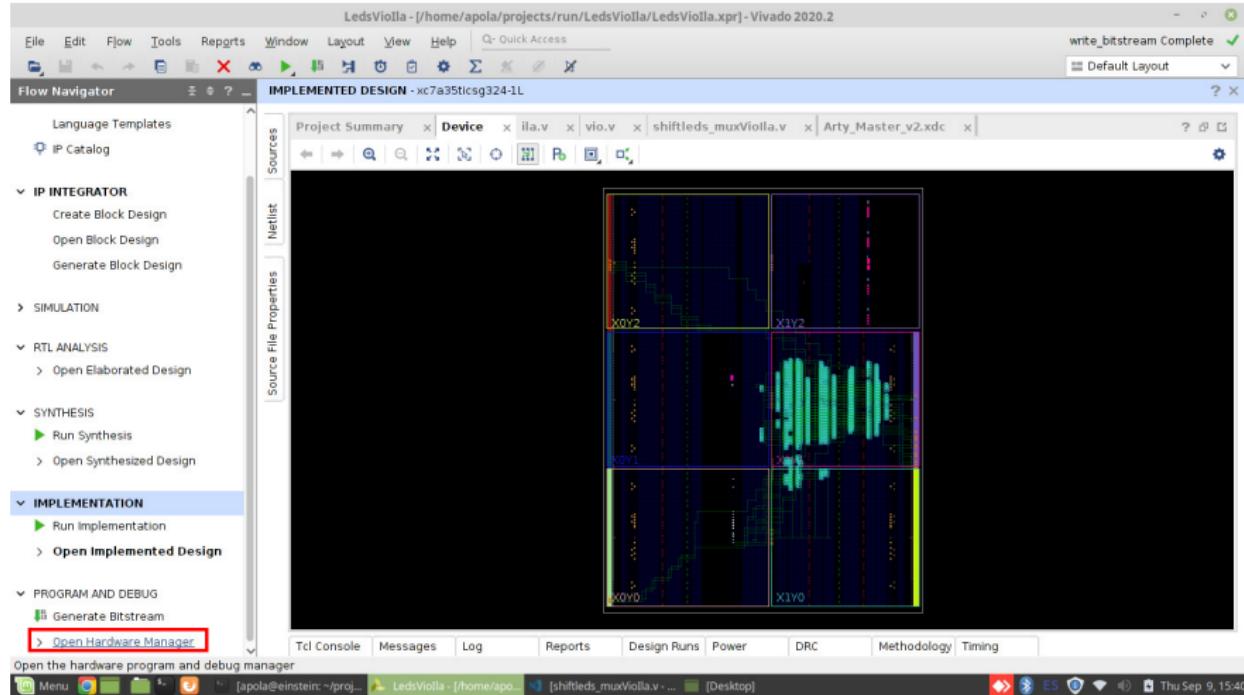
Last login: Tue Nov 14 14:23:29 2017 from 181.95.86.185
user@kunturi:~$ cd work_dda/apola/scripts/
user@kunturi:~/work_dda/apola/scripts$ python client.py

Write-> Exit <-to close the session
Ip: 127.0.0.1
Port: 5005
Checking Free Port
Ports Busy
user@kunturi:~/work_dda/apola/scripts$ [ ]
```

Las figuras muestran la asignación correcta de puerto (izq.) y los puertos ocupados (der.).

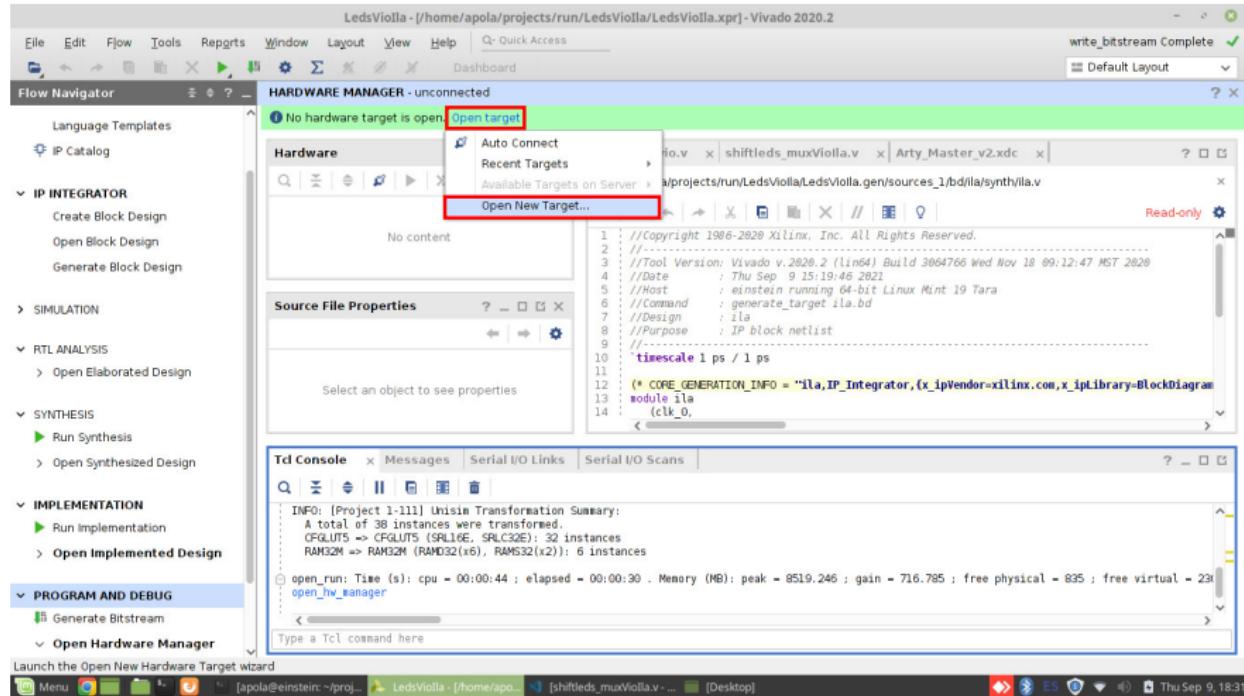
Programación y Ejecución

Programación y Ejecución



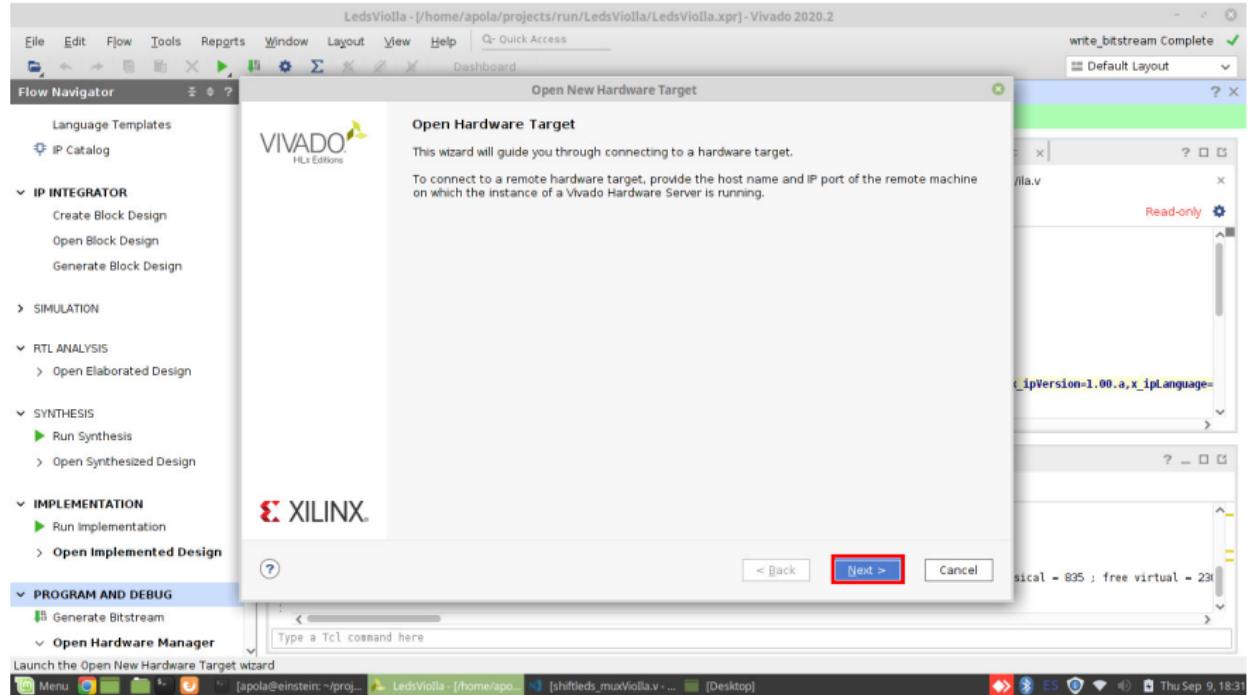
Open Hardware Manager.

Programación y Ejecución



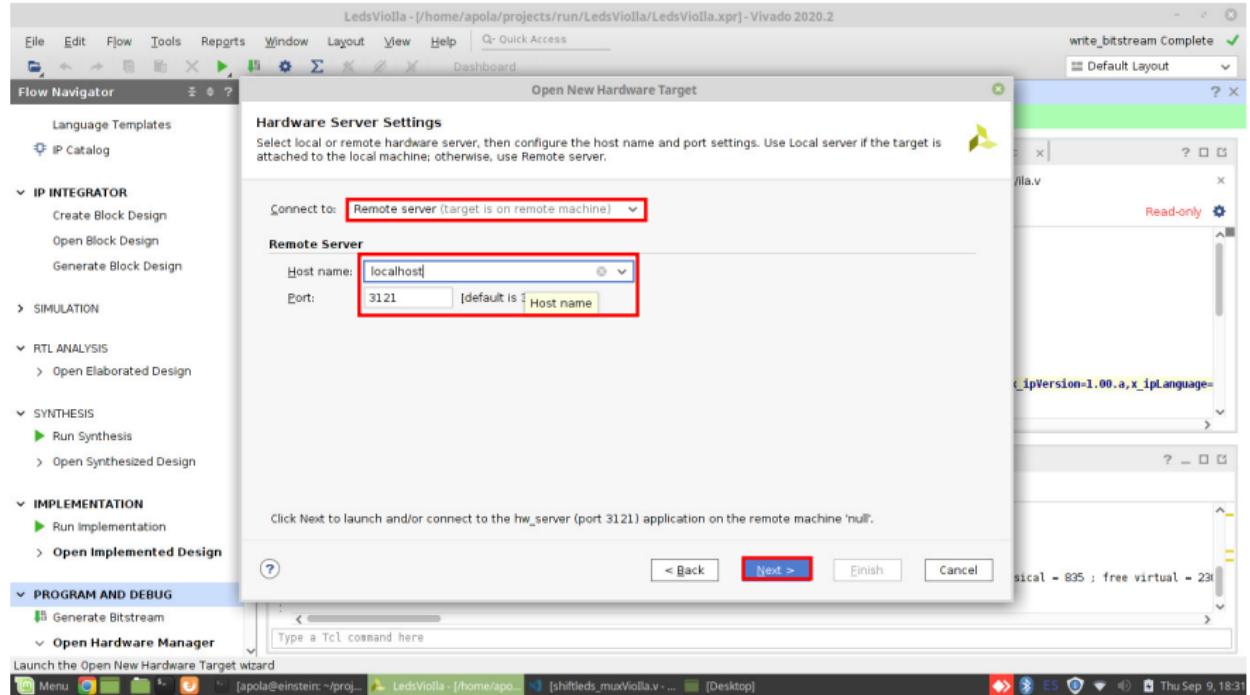
Abrir el dispositivo.

Programación y Ejecución



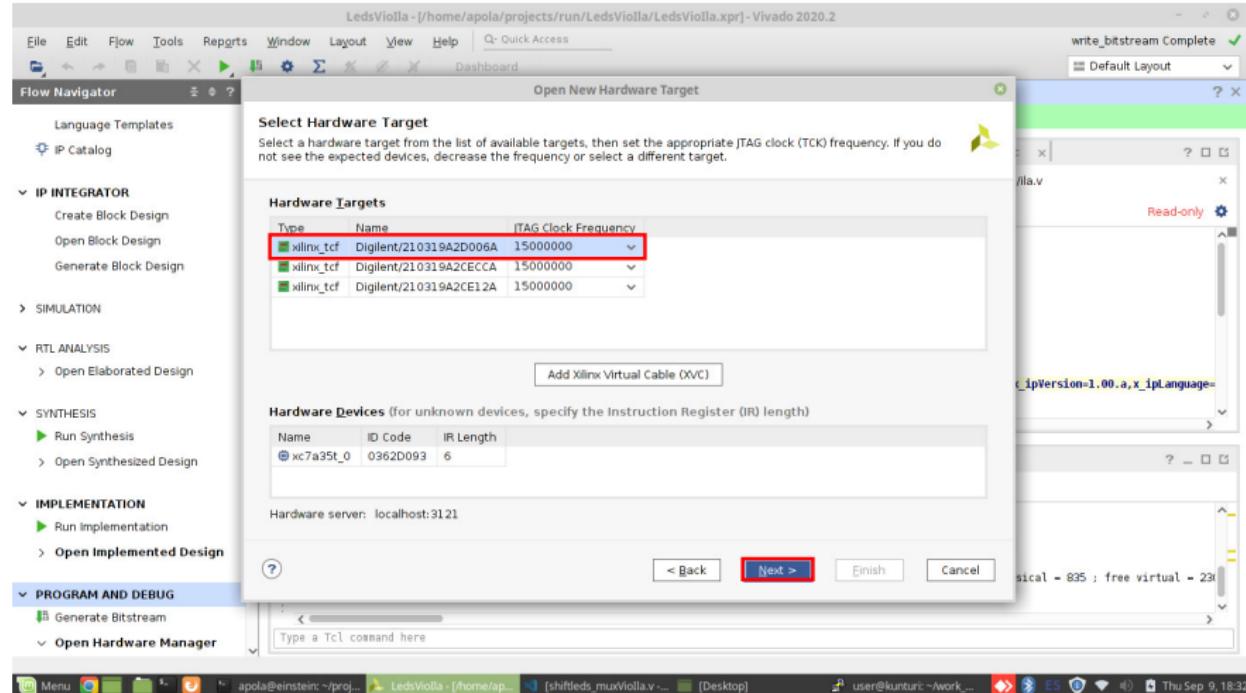
Abrir el dispositivo.

Programación y Ejecución



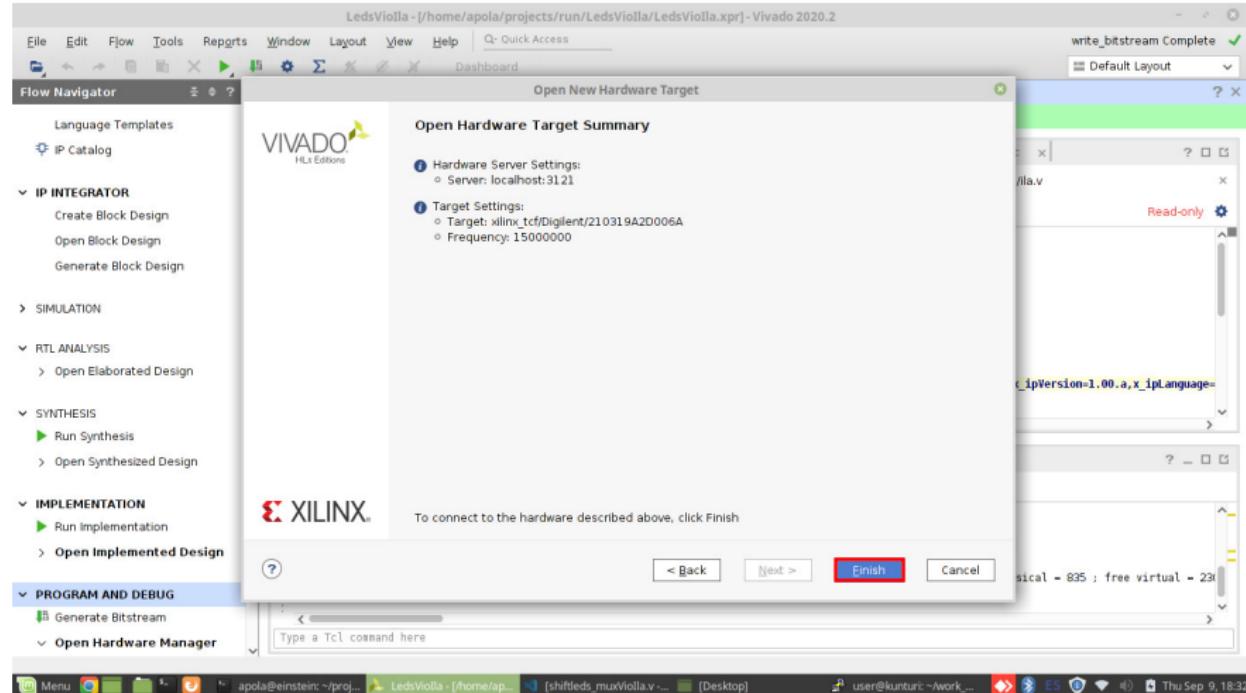
Seleccionar el servidor remoto y asignar el nombre del Host (localhost).

Programación y Ejecución



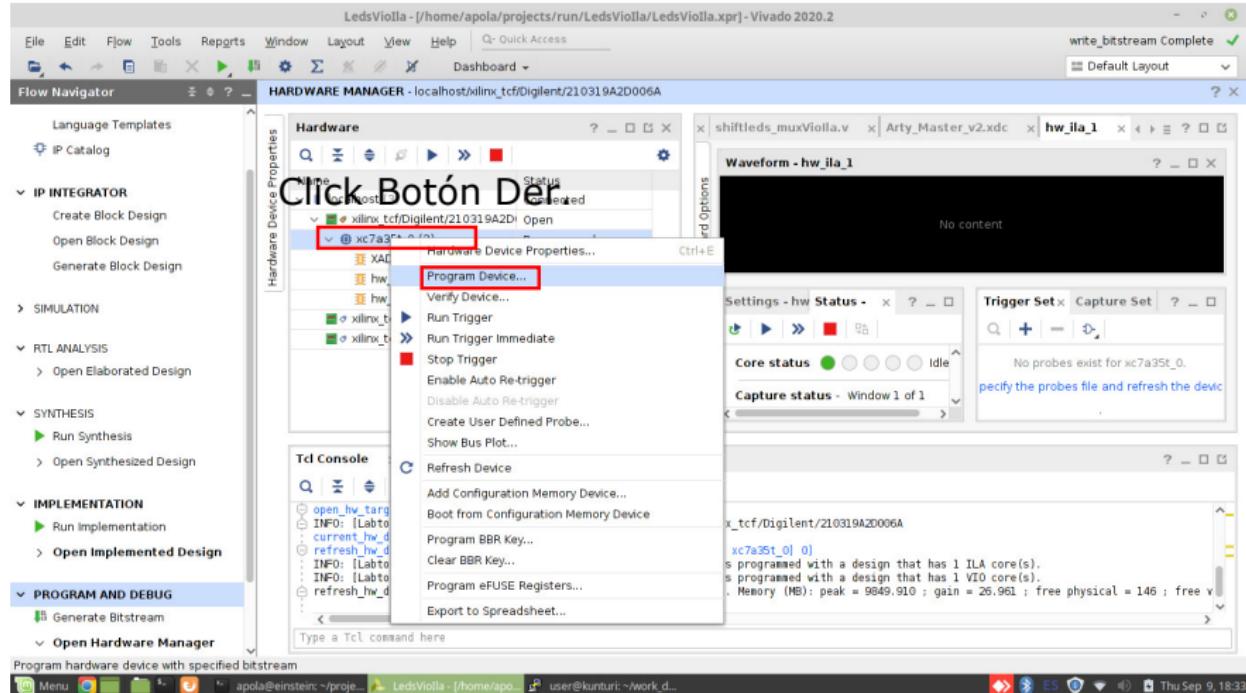
Seleccionar el ID previamente asignado por la conexión remota.

Programación y Ejecución



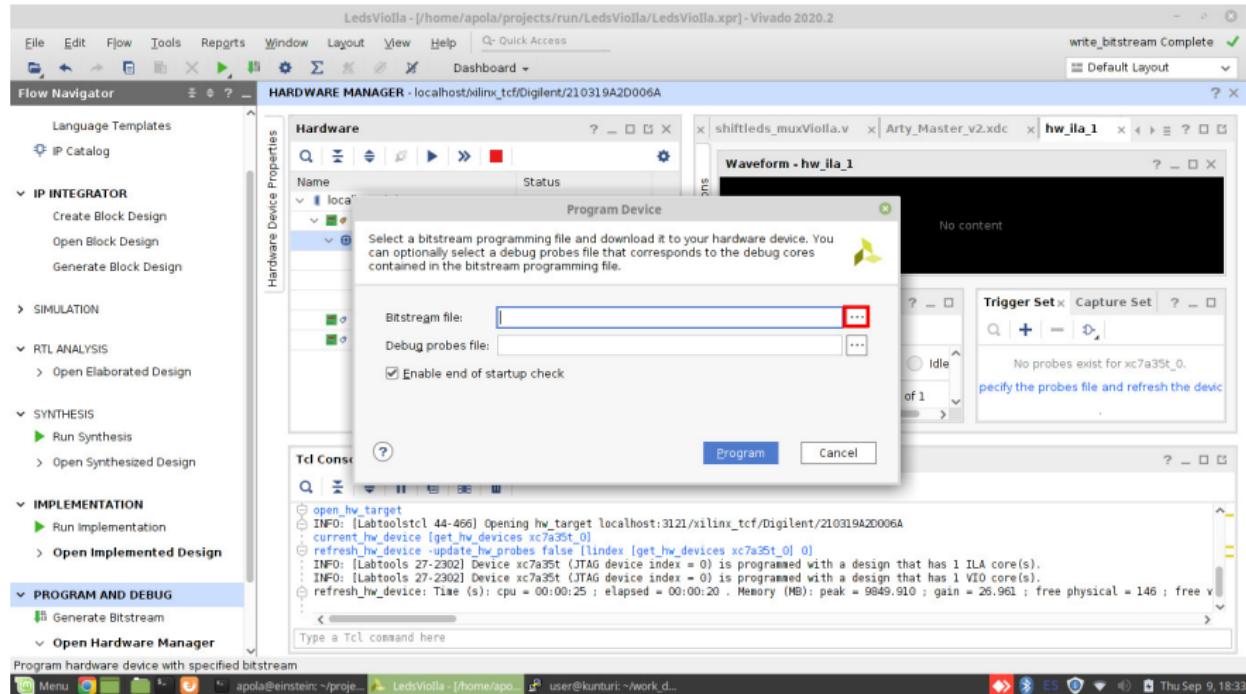
Finalizar configuración.

Programación y Ejecución



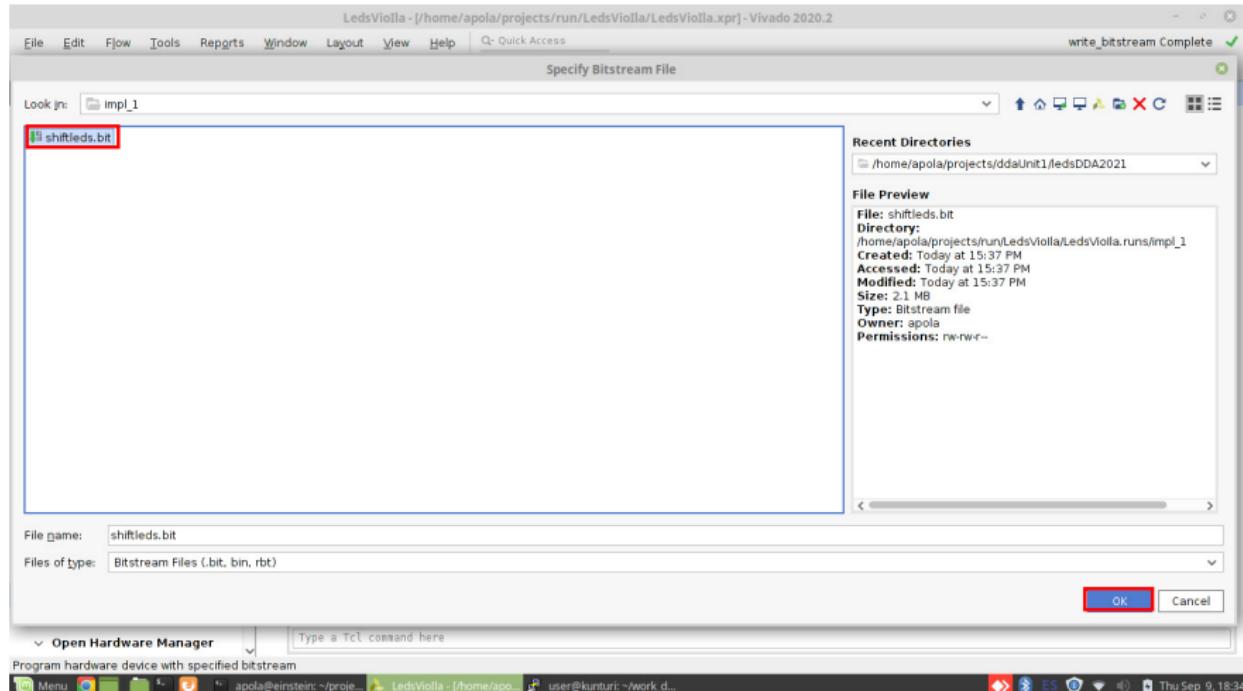
Programar la FPGA.

Programación y Ejecución



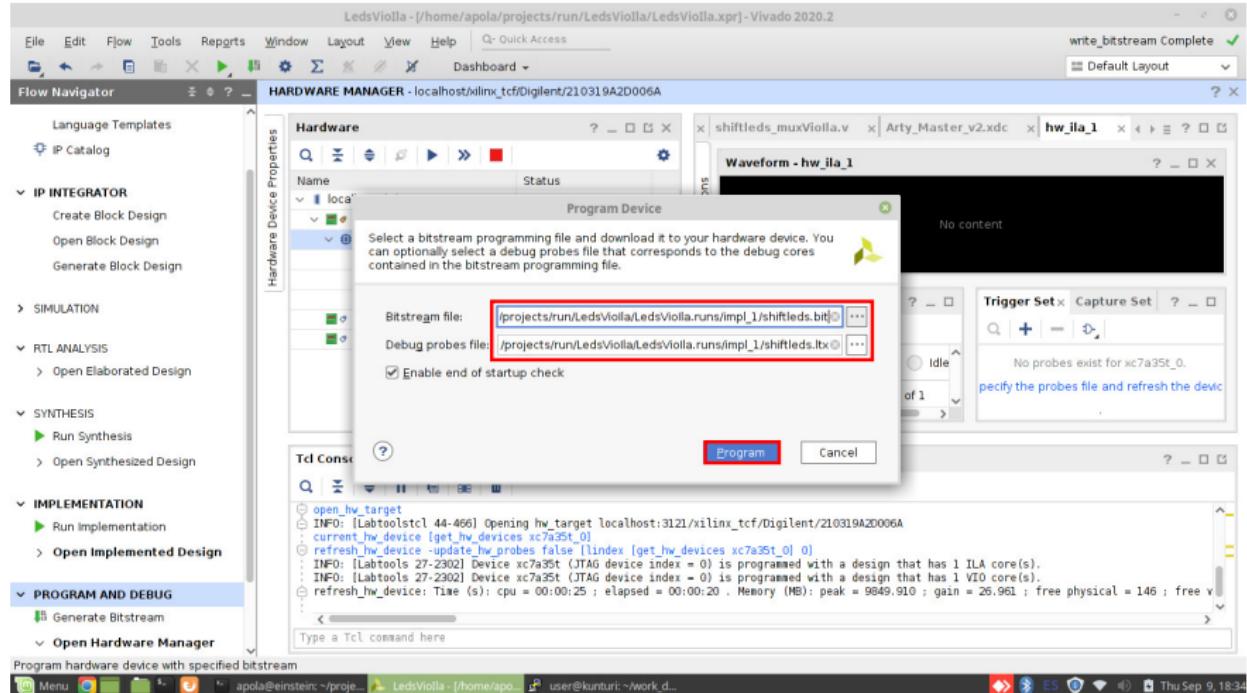
Buscar el archivo binario.

Programación y Ejecución



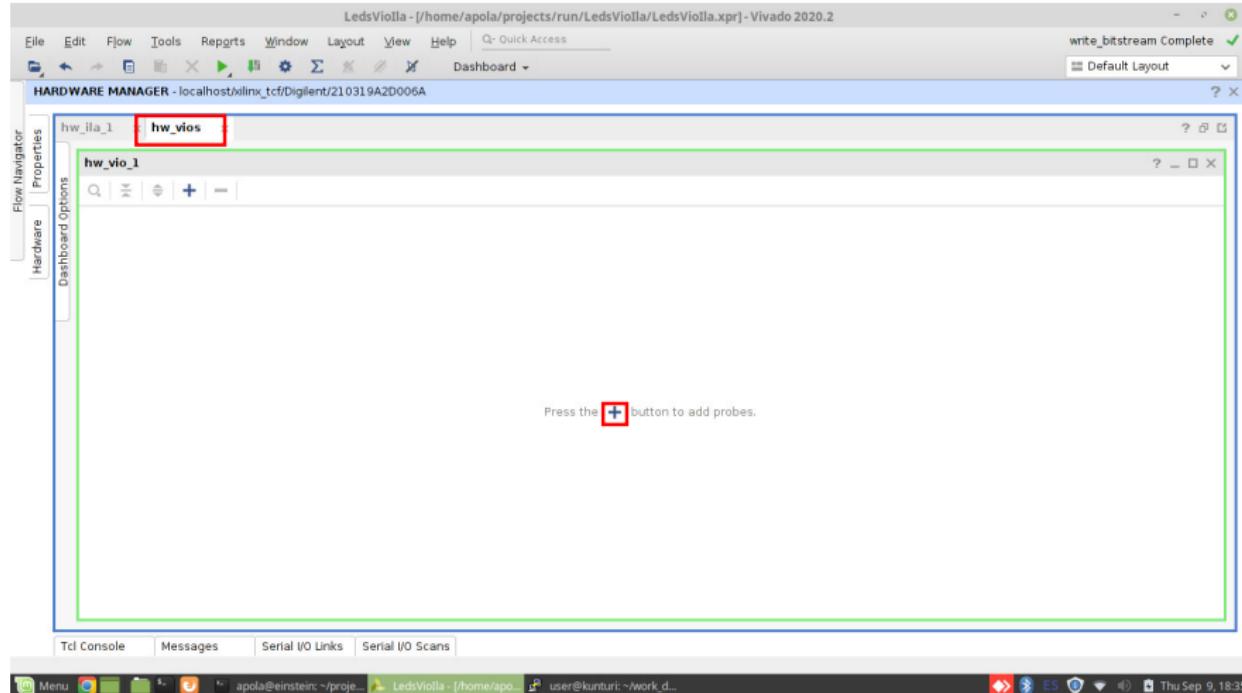
Buscar el archivo binario.

Programación y Ejecución



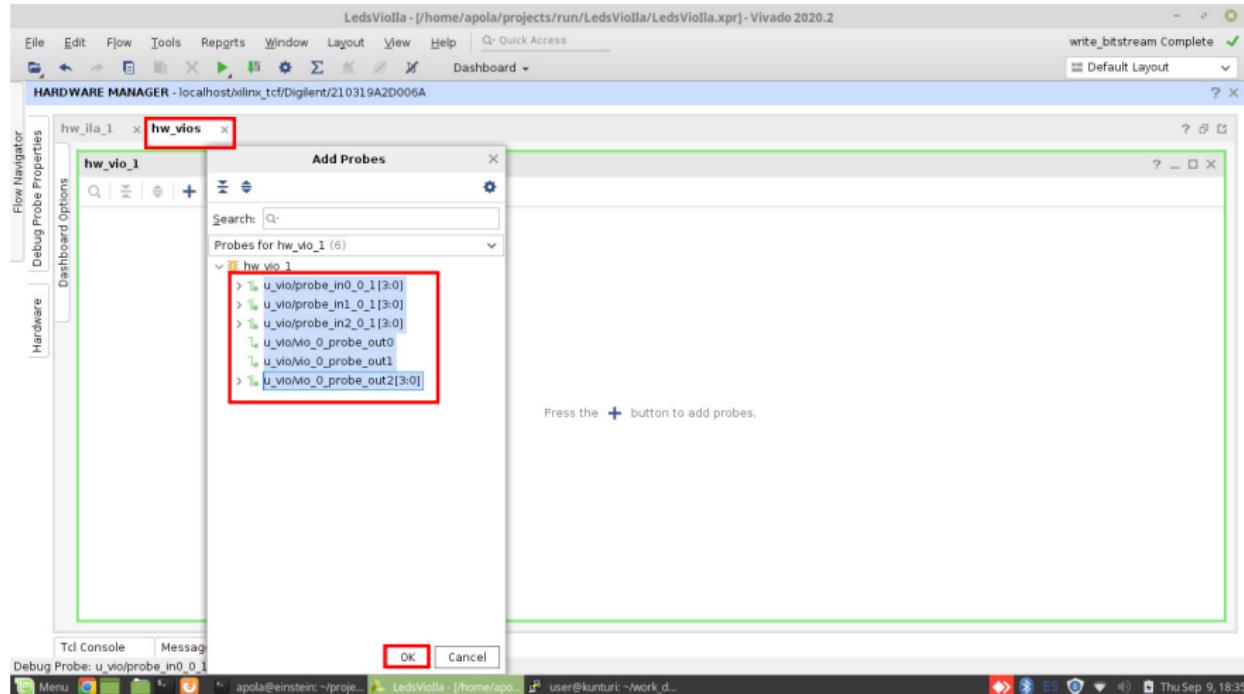
Verificar que se asigne el binario y el archivo de puntas de prueba.

Programación y Ejecución



Seleccionar los puertos de control.

Programación y Ejecución



Seleccionar los puertos de control.

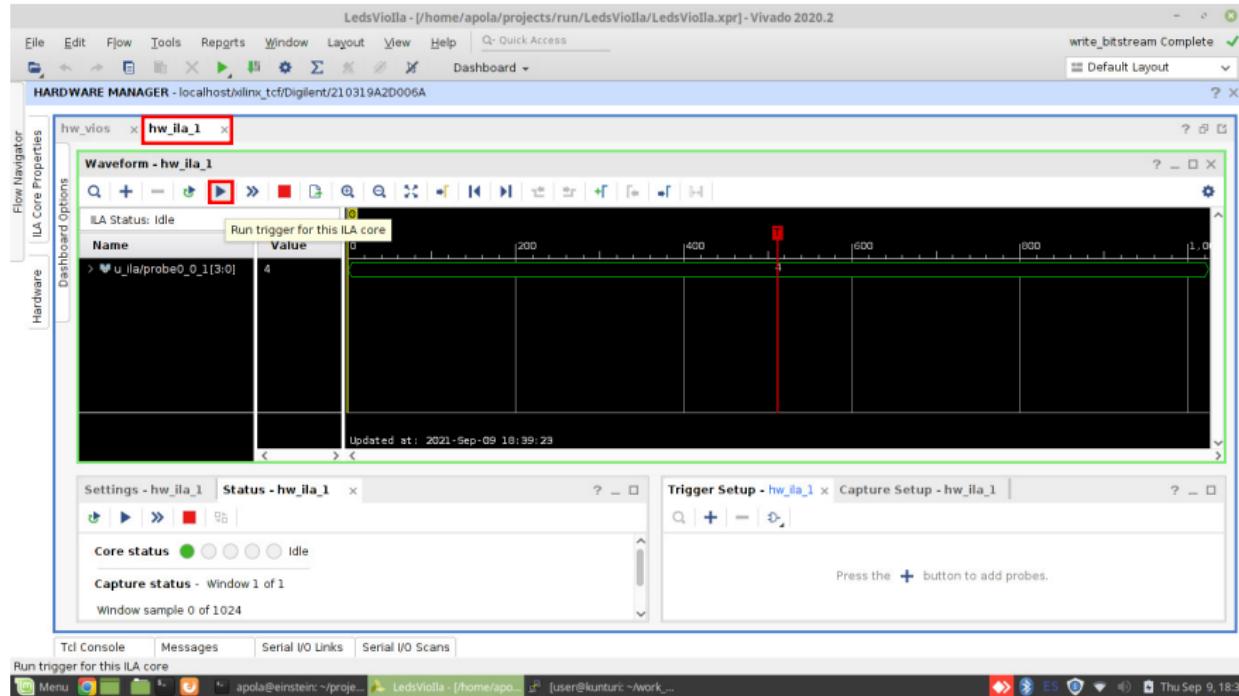
Programación y Ejecución

The screenshot shows the Vivado 2020.2 Hardware Manager interface. The title bar indicates the project is "LedsViolla - [/home/apola/projects/run/LedsViolla/LedsViolla.xpr] - Vivado 2020.2". The menu bar includes File, Edit, Flow, Tools, Reports, Window, Layout, View, Help, and Quick Access. The right side of the window shows a status message: "write_bitstream Complete" with a green checkmark. The main area is titled "HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210319A2D006A". On the left, there's a "Flow Navigator" with tabs for "Hardware", "Debug Probe Options", and "Dashboard". The central pane is titled "hw_vios" and contains a table of I/O ports. The table has columns: Name, Value, Activity, Direction, and VIO. The "hw_vio_1" row is expanded, showing detailed settings for each port. At the bottom, there are tabs for "Tcl Console", "Messages", "Serial I/O Links", and "Serial I/O Scans".

Name	Value	Activity	Direction	VIO
> u_vio/probe_in0_0_1[3:0]	[H] 8		Input	hw_vio_1
> u_vio/probe_in1_0_1[3:0]	[H] 8		Input	hw_vio_1
> u_vio/probe_in2_0_1[3:0]	[H] 0		Input	hw_vio_1
u_vio/vio_0_probe_out0	[B] 1		Output	hw_vio_1
u_vio/vio_0_probe_out1	[B] 1		Output	hw_vio_1
u_vio/vio_0_probe_out2[3:0]	[H] 1		Output	hw_vio_1
u_vio/vio_0_probe_out2[0]	1		Output	hw_vio_1
u_vio/vio_0_probe_out2[1]	0		Output	hw_vio_1
u_vio/vio_0_probe_out2[2]	0		Output	hw_vio_1
u_vio/vio_0_probe_out2[3]	0		Output	hw_vio_1

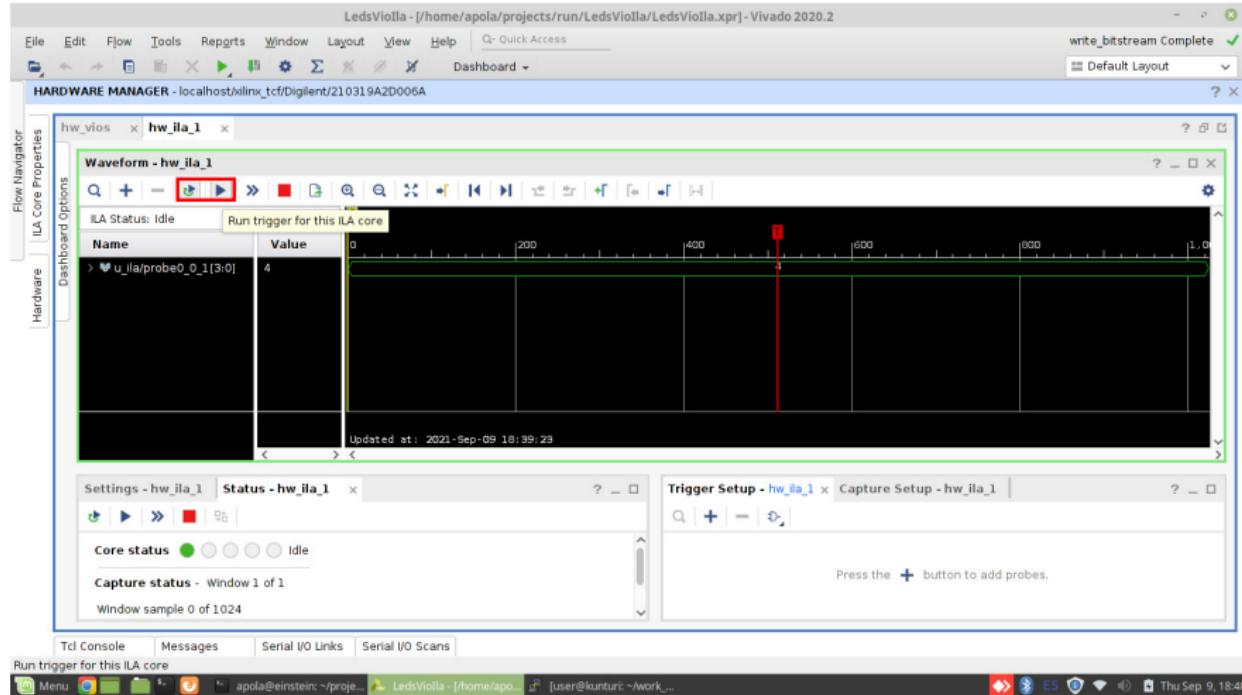
Puertos de control.

Programación y Ejecución



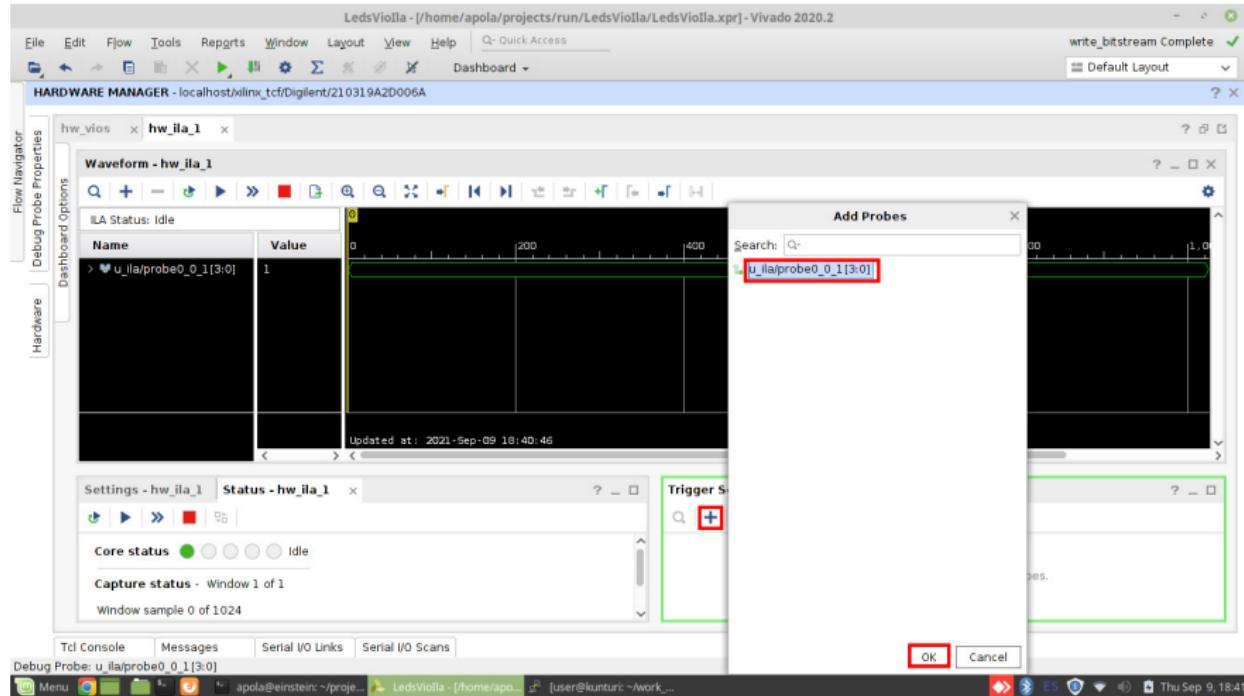
Ejecutar una vez el trigger.

Programación y Ejecución



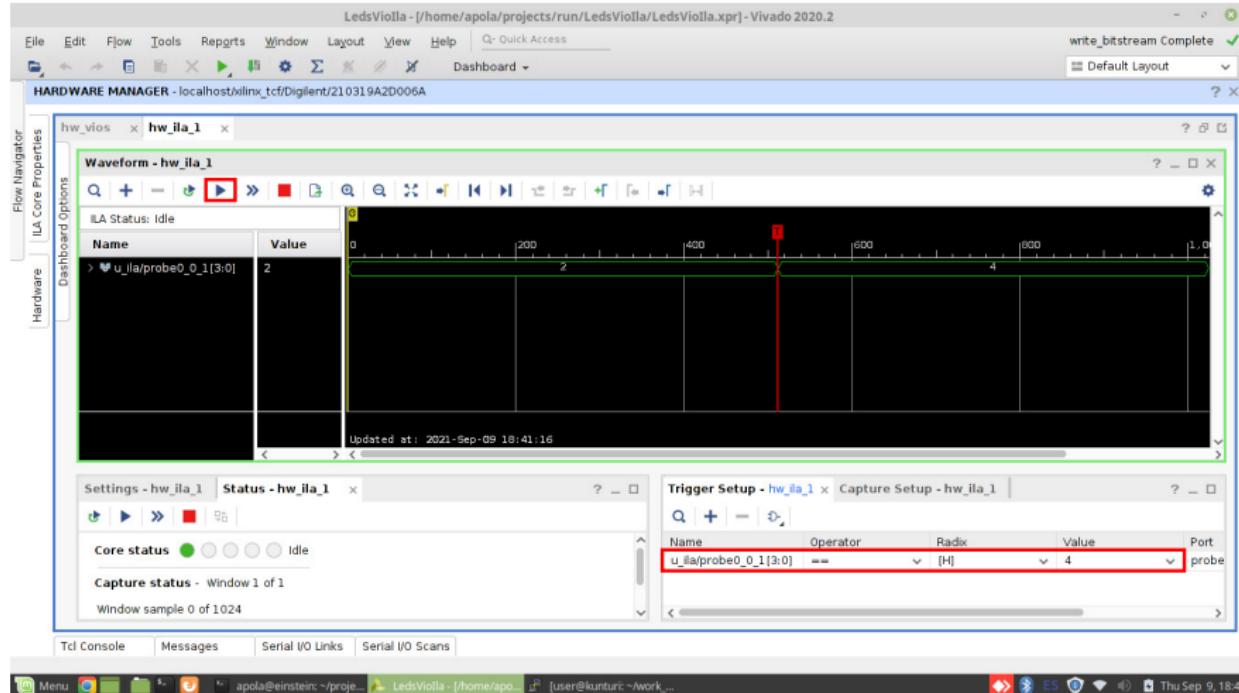
Seleccionar trigger continuo y luego ejecutar el trigger.

Programación y Ejecución



Agregar una condición de trigger y seleccionar el puerto.

Programación y Ejecución



Agregar una condición de comparación y ejecutar el trigger.