



PROGRAMA UNIVERSITARIO DE IMAGINATION

Práctica 1 RVfpga

Creación de un proyecto de Vivado

1. INTRODUCCIÓN

Para poder trabajar con el Sistema RVfpga así como modificarlo, el usuario necesitará construir un proyecto que incluya todos los archivos Verilog, SystemVerilog, de cabecera, de configuración y de texto que definen el sistema. En esta práctica se mostrará cómo crear un proyecto de Vivado que cargue RVfpgaNexys en la placa FPGA Nexys A7 de Digilent, versión -100T. (Recuerde que si tiene una placa Nexys4 DDR, también puede usarla.) Siguiendo estos mismos pasos, podrá modificar el Sistema RVfpga y resintetizarlo.

IMPORTANTE: Antes de comenzar con las prácticas de RVfpga, el usuario debe de haber completado la Guía de Inicio RVfpga proporcionada por el Programa Universitario de Imagination (<https://university.imgtec.com/>).

Así, si no lo ha hecho previamente, debe instalar los programas Vivado de Xilinx, PlatformIO y Verilator conforme a las instrucciones que se detallan en esa guía. Además, asegúrese de haber copiado en su máquina la carpeta **RVfpga** que ha descargado del Programa Universitario de Imagination. Se hará referencia a la ruta absoluta del directorio donde copie la carpeta RVfpga como [RVfpgaPath]. La carpeta [RVfpgaPath]/RVfpga/src contiene las fuentes Verilog y SystemVerilog para el Sistema RVfpga, el SoC RISC-V que se utilizará y modificará a lo largo de estas prácticas. La carpeta [RVfpgaPath]/RVfpga/Labs contiene los recursos que se usarán durante las prácticas 1 a 10.

2. Creación de un Proyecto Vivado para RVfpga

Se utilizará la Suite de Diseño Vivado de Xilinx¹ para construir RVfpgaNexys usando el RTL, los archivos Verilog que definen el sistema. Siga los pasos que se detallan a continuación para construir RVfpgaNexys y cargarlo en una placa FPGA Nexys A7.

Paso 1. Abrir Vivado

Paso 2. Crear un nuevo proyecto RTL

Paso 3. Añadir los archivos fuente RTL y los archivos de restricciones

Paso 4. Seleccionar Nexys A7 como placa objetivo

Paso 5. Configuración del proyecto: Establecer *rvfpganexys* como *Top Module*, *common_defines.vh* como *Global include*, añadir el fichero *boot_main.mem* al proyecto e incluir los directorios de Pulp Platform.

Paso 6. Generar el *Bitstream*

Paso 1. Abrir Vivado

Si no ha instalado Vivado en su máquina tal y como se describe en la Guía de inicio de RVfpga, por favor hágalo ahora. Asegúrese de instalar también los archivos correspondientes a la placa.

Ahora, ejecute Vivado (en **GNU/Linux**, abra un terminal y escriba: vivado; en **Windows**, abra Vivado desde el menú Inicio). Se abrirá la pantalla de bienvenida de Vivado. Haga clic en Crear Proyecto (*Create Project*) (ver Figura 1).

¹ En este material se utiliza Vivado 2019.2. Aunque la mayoría de las cosas también deberían funcionar en las distribuciones más recientes de Vivado, se recomienda el uso de esta versión.

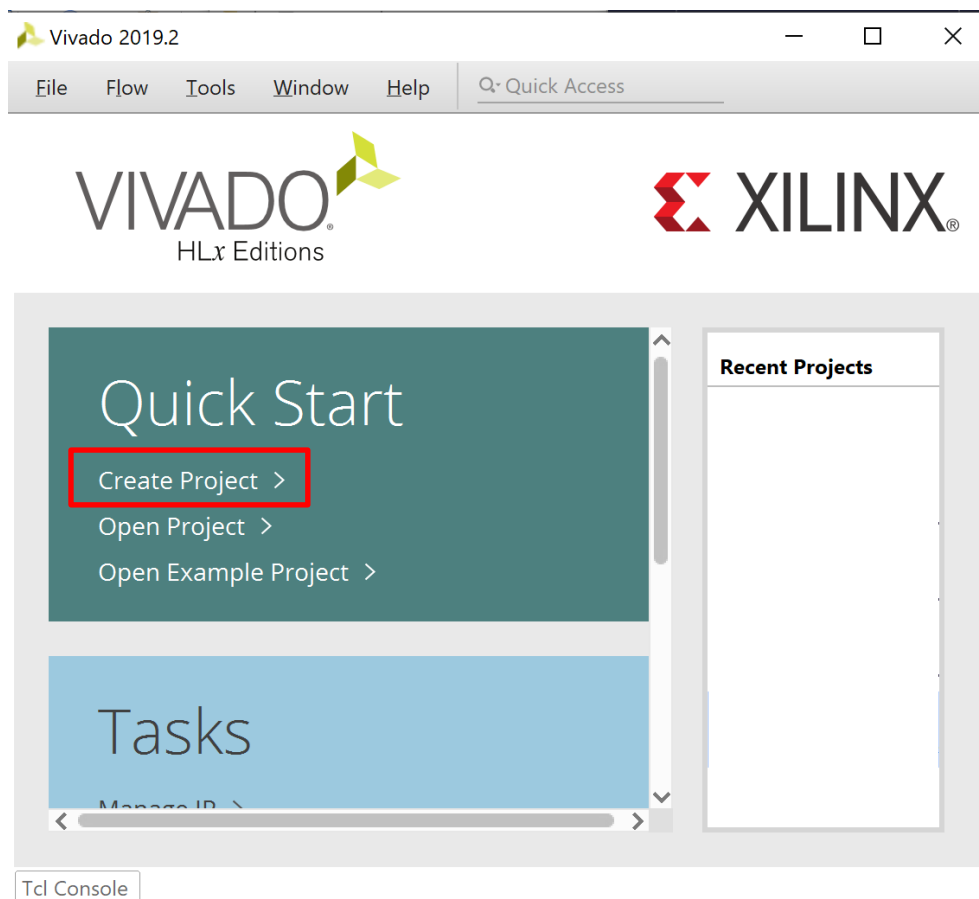


Figura 1. Pantalla de bienvenida de Vivado: Crear proyecto

Paso 2. Crear un nuevo proyecto RTL

A continuación se abrirá el Asistente para Crear un Nuevo Proyecto Vivado (ver Figura 2). Haga clic en Siguiente (*Next*).

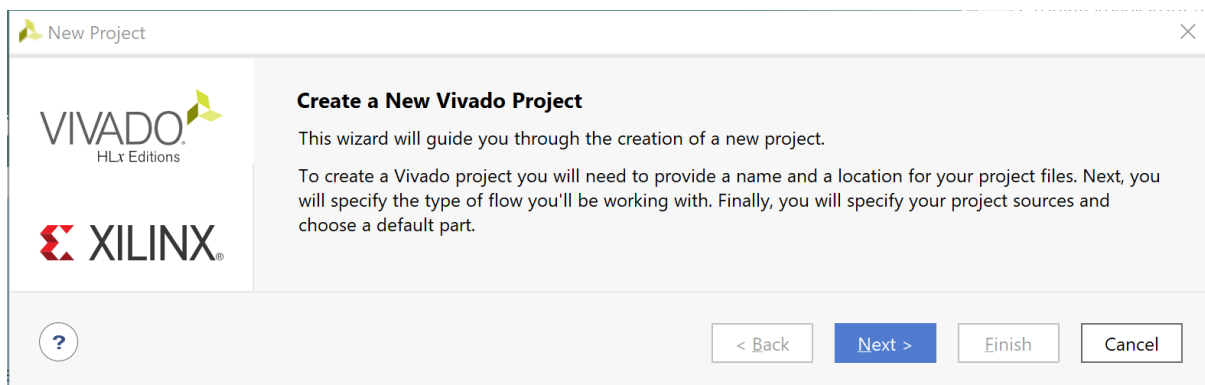


Figura 2. Asistente para Crear un Nuevo Proyecto Vivado

Ponga como nombre del Proyecto (*Project name*) Proyecto1 y guárdelo (*Project location*) en la carpeta `[RVfpgaPath]/RVfpga/Labs/Lab1`. Marque la opción *Create project subdirectory*. Después haga clic en Siguiente (*Next*) (ver Figura 3).

Project Name

Enter a name for your project and specify a directory where the project data files will be stored.



Project name:

Project location:

☒ Create project subdirectory

Project will be created at: /home/dchaver/VM_SharedFolder/RVfpga_1-1/RVfpga/Labs/Lab1/project_1




Figura 3. Nombre del proyecto

Seleccione el tipo de proyecto como Proyecto RTL (*RTL Project*), y haga clic en Siguiente (*Next*) (ver Figura 4).

New Project

Project Type
Specify the type of project to create.

☒ **_RTL Project**
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.

☐ Do not specify sources at this time

☐ **_Post-synthesis Project**




Figura 4. Proyecto RTL

Paso 3. Añadir los archivos fuente RTL y los archivos de restricciones

En la ventana Agregar fuentes (*Add Sources*), haga clic en Agregar directorios (*Add Directories*) y seleccione *[RVfpgaPath]/RVfpga/src* (ver Figura 5). Asegúrese que las dos opciones siguientes estén seleccionadas (como se muestra en la Figura 5):

- Escanear y añadir archivos de inclusión RTL en el proyecto (*Scan and add RTL include files into project*)
- Añadir fuentes de subdirectorios (*Add sources from subdirectories*)

Luego haga clic en Siguiente (*Next*).

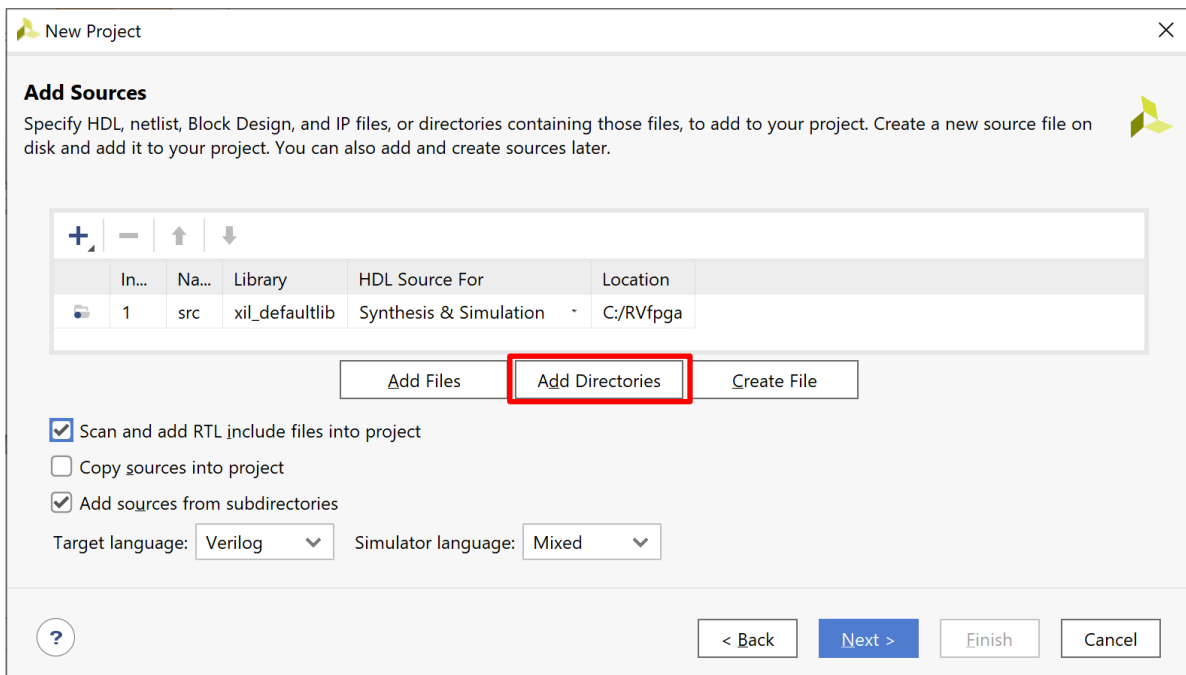


Figura 5. Añadir fuentes

Ahora se añadirán las restricciones para el sistema. Estos archivos asignan los nombres de las señales a los pines de la placa. Por ejemplo, los LEDs de la placa FPGA Nexys A7 están conectados a los pines de la FPGA en la placa a través de vías en la PCB. Vivado debe conocer esto para poder mapear el nombre correcto de la señal en el RTL al pin correcto de la FPGA. Por ejemplo, la siguiente línea en el archivo `[RVfpgaPath]/RVfpga/src/rvfpganexys.xdc`, un archivo de restricciones de diseño de Xilinx, indica que el pin H17 de la FPGA se mapea al LED menos significativo (`o_led[0]`) y que utiliza la señalización LVCMOS 3.3V:

```
set_property -dict { PACKAGE_PIN H17 IOSTANDARD LVCMOS33 } get_ports { o_led[0] }
```

Tenga en cuenta que el nombre de la señal `o_led` es el nombre utilizado en el código Verilog del Sistema RVfpga para controlar los LEDs de la placa Nexys A7.

En la ventana Agregar restricciones (*Add Constraints*), haga clic en Agregar archivos (*Add Files*) y seleccione los dos siguientes archivos (ver Figura 6):

```
[RVfpgaPath]/RVfpga/src/rvfpganexys.xdc  
[RVfpgaPath]/RVfpga/src/LiteDRAM/liteDRAM.xdc
```

Luego haga clic en Siguiente (*Next*).

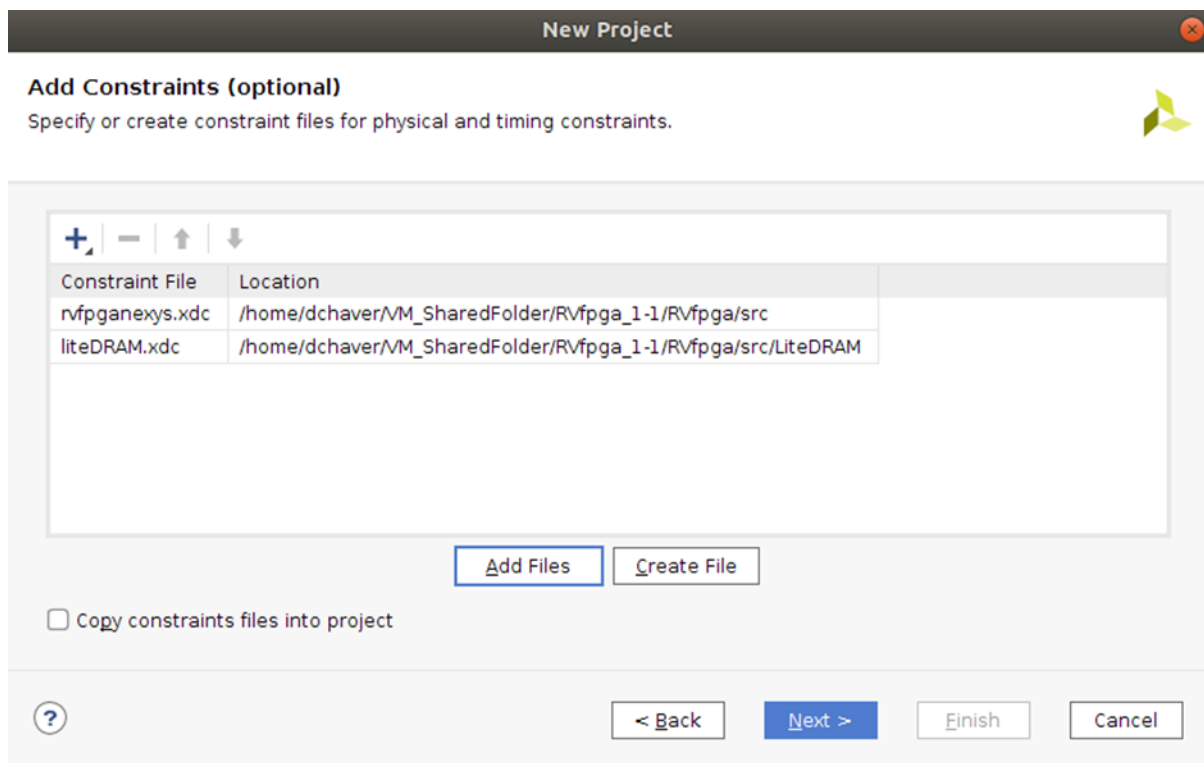


Figura 6. Agregar restricciones

Paso 4. Selecciona Nexys A7 como placa objetivo

En la ventana Componente por defecto (*Default Part*), haga clic en Placas (*Boards*) y seleccione Nexys A7-100T (Figura 7). Puede utilizar el cuadro de búsqueda para filtrar los resultados. También puede comprobar que el nombre de la FPGA objetivo real aparece en la columna Componente (*Part*): xc7a100tcs9324-1. Esto indica que es una FPGA Xilinx Artix-7 con 100k puertas equivalentes, con un paquete CSG (*chip scale grid*) y 324 pines.

Haga clic en Siguiente (*Next*).

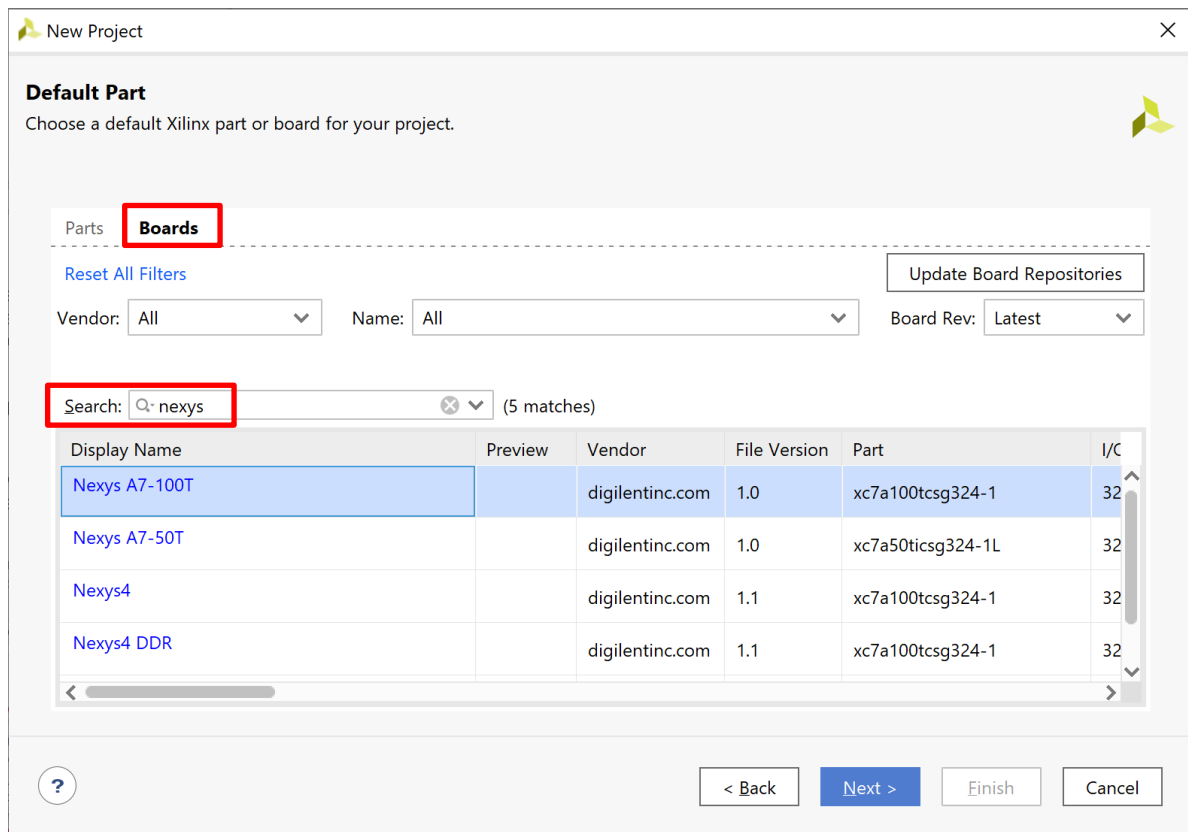


Figura 7. Selección de la placa objetivo: Nexys A7-100T

En la ventana de resumen de un nuevo proyecto (*New Project Summary*), haga clic en Finalizar (*Finish*), véase la Figura 8.

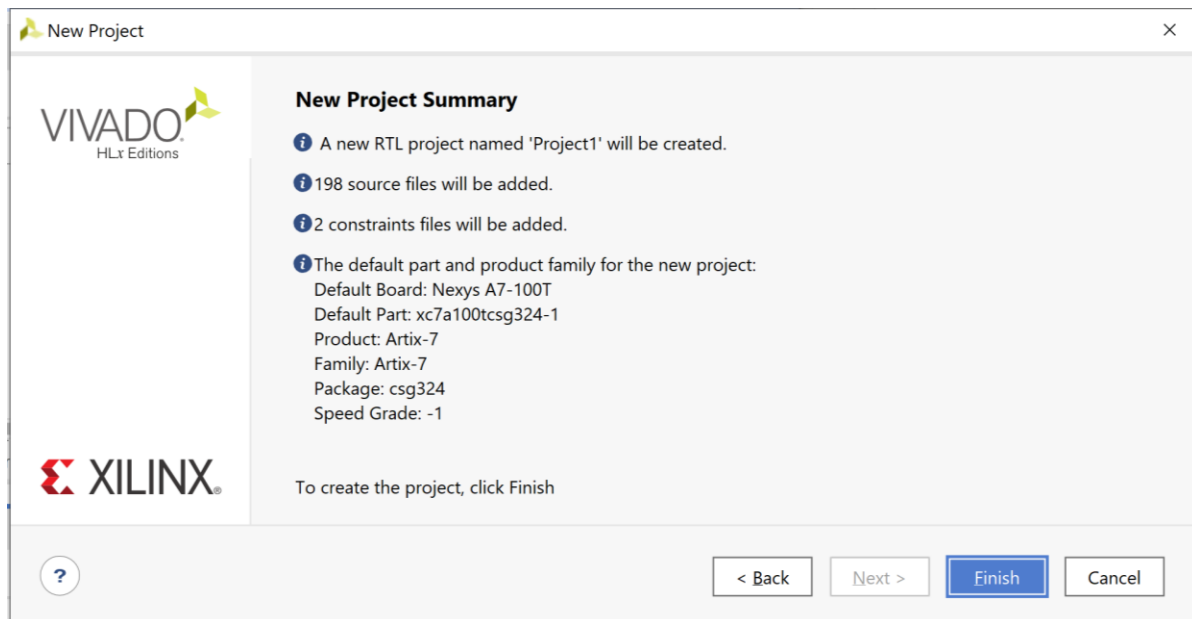


Figura 8. Ventana de resumen del nuevo proyecto

Tenga en cuenta que una vez que el proyecto termine de configurarse, indicará que existen archivos con errores de sintaxis - esto se solucionará en el siguiente paso.

Paso 5. Configuración del proyecto: Establecer *rvfpganexys* como *Top Module*, *common_defines.vh* como *Global include*, añadir el fichero *boot_main.mem* al proyecto e incluir los directorios de Pulp Platform.

Establecer *rvfpganexys* como *Top Module*: Establezca el módulo *rvfpganexys* como el Top Module (módulo superior). En el panel Fuentes (*Sources*), desplácese hacia abajo en Fuentes de Diseño (*Design Sources*), haga clic con el botón derecho del ratón en el módulo *rvfpganexys* y seleccione Establecer como Superior (*Set as Top*) (véase la Figura 9). También puede encontrar el módulo *rvfpganexys* escribiendo este nombre en el cuadro de búsqueda, como se muestra en la citada figura. Esto establece *rvfpganexys* como el módulo de más alto nivel en la jerarquía y el objetivo a ser sintetizado e implementado en la FPGA. Después de establecer *rvfpganexys* como el módulo superior, la jerarquía se actualizará.

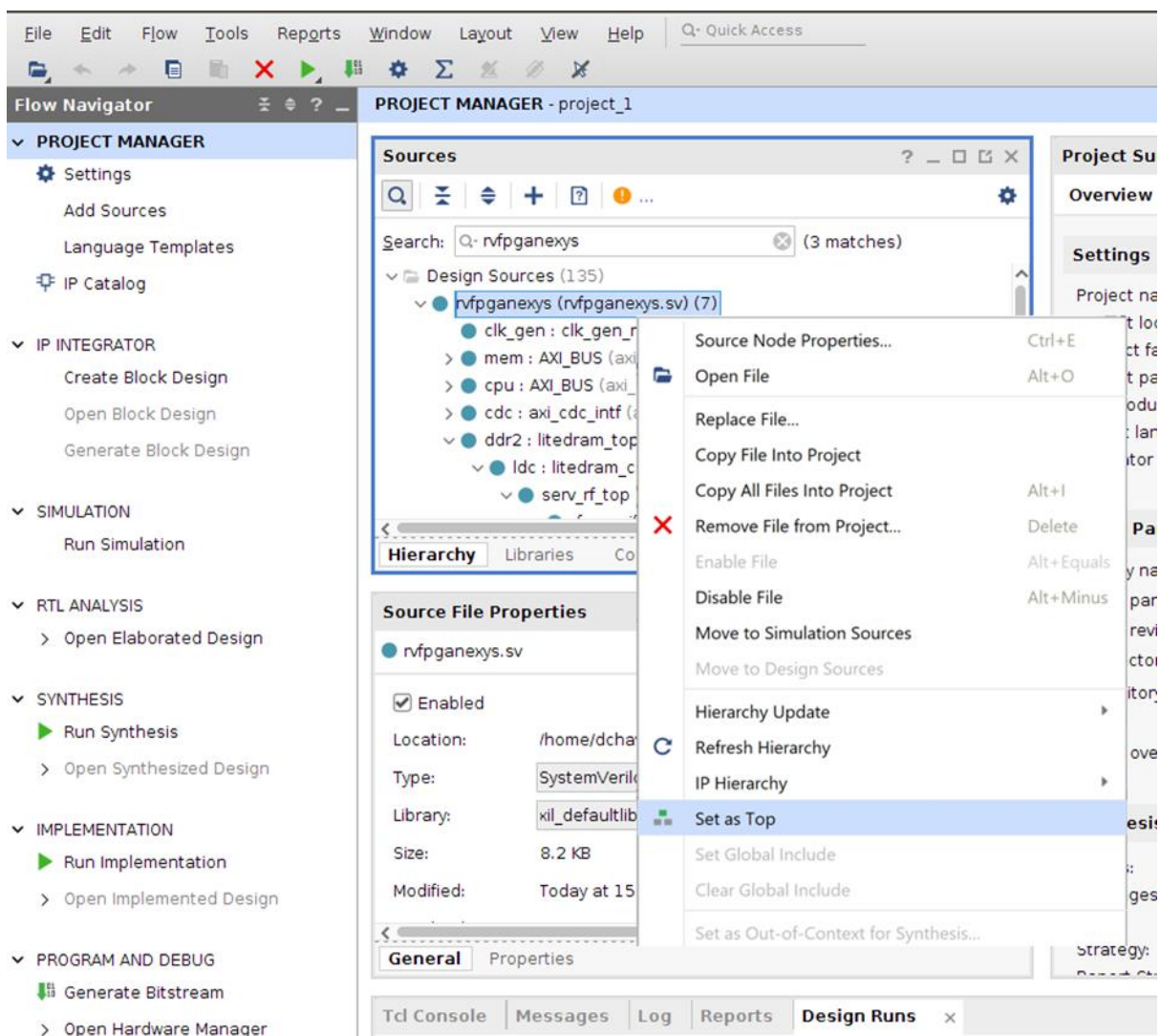


Figura 9. Establecer *rvfpganexys* como módulo superior

Establecer *common_defines.vh* como *Global include*: Ahora, aún en el panel de Fuentes (*Sources*) bajo Fuentes de Diseño (*Design Sources*), expanda el grupo de archivos No-módulos (*non-module files*) y haga clic en *common_defines.vh*. Las propiedades del archivo se abrirán en el panel Propiedades del archivo de origen (*Source File Properties*), justo

debajo del panel Fuentes. Haga clic en Inclusión Global (*Global Include*) para marcar esa casilla (véase la Figura 10). La jerarquía se actualizará e incluirá ese archivo en Fuentes de Diseño/Inclusión Global (*Design Sources/Global Include*). Tenga en cuenta que los Archivos de error de sintaxis desaparecerán en el siguiente paso.

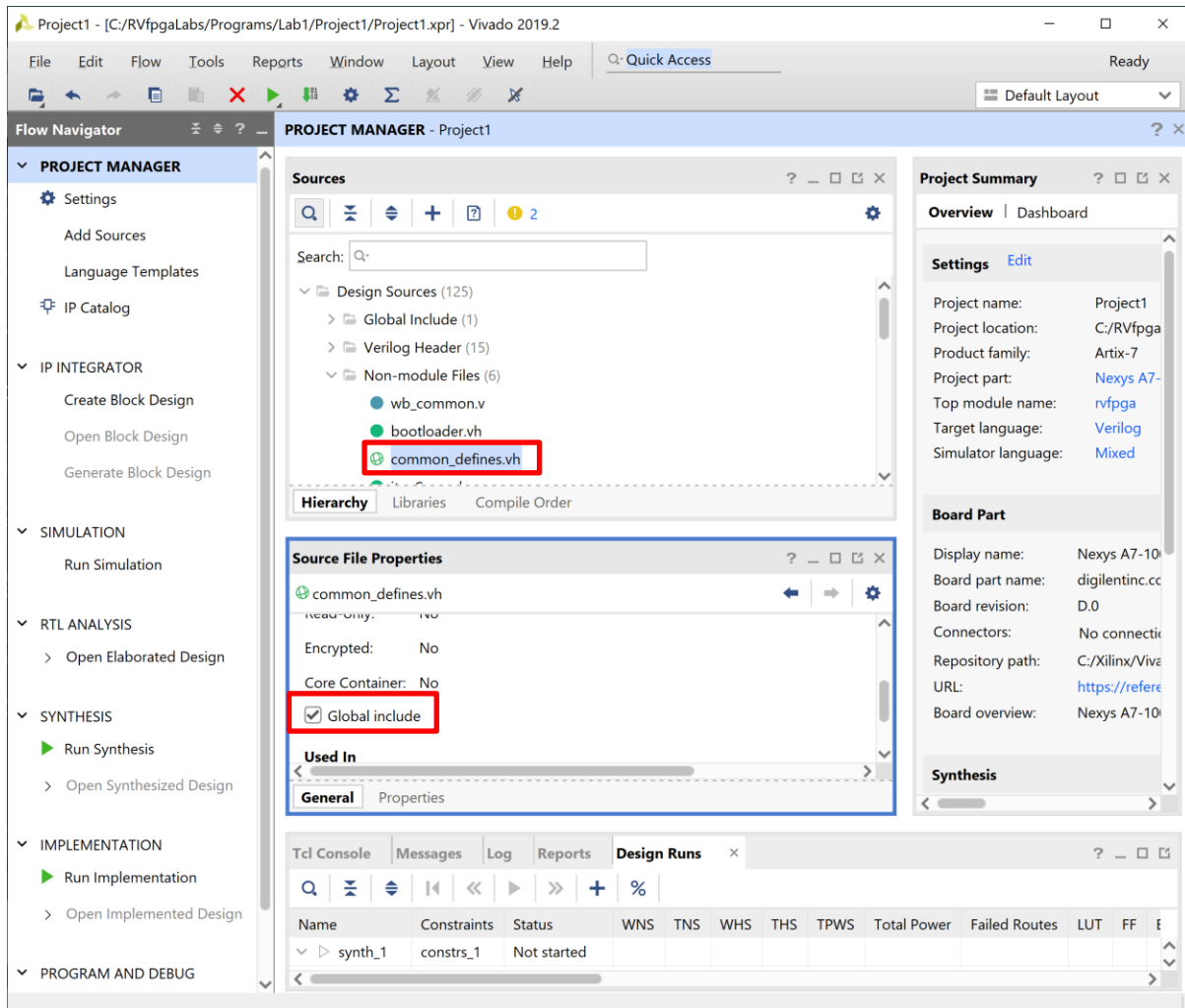


Figura 10. Establecer common_defines.vh como archivo de inclusión global

Añadir el fichero *boot_main.mem* al proyecto: En la ventana Flow Navigator, haga clic en Add Sources, mantenga la opción por defecto (“Add or create design sources”), y haga clic en Add Files (ver Figure 11). Navegue a la carpeta *[RVfpgaPath]/RVfpga/src/SweRVolfSoC/BootROM/sw* y seleccione el fichero *boot_main.mem* (como se muestra en la Figure 11). La jerarquía de módulos se actualizará y se incluirá el nuevo fichero en Design Sources/Memory File.

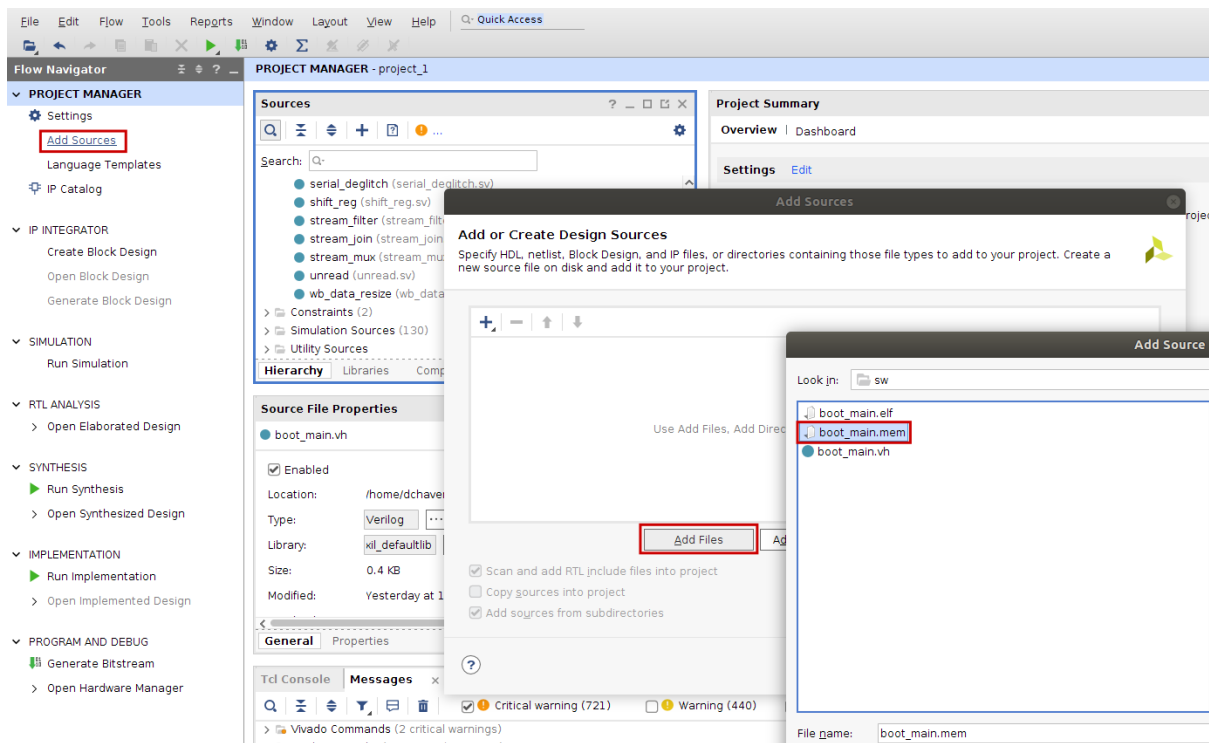


Figure 11. Añadir el fichero *boot_main.mem*

Este archivo (*boot_main.mem*) se utiliza para inicializar la Boot ROM del SoC. Para ello, se invoca como parámetro en el archivo `[RVfpgaPath]/RVfpga/src/rvfpganexys.rv`:

```
25 module rvfpga
26   #(parameter bootrom_file = "boot_main.mem")
```

La Sección 6.A de la Guía de Inicio contiene más información sobre este fichero.

Incluir los directorios de Pulp Platform: Para terminar, incluya los dos directorios de la Pulp Platform (ver Figure 12). En la ventana Flow Navigator haga clic en *Settings*, y en la

ventana que se abre haga clic en *General* y luego en *Verilog options* (⋮). En la nueva ventana, añada los dos siguientes directorios haciendo clic en **+** y navegando a los siguientes directorios:

```
[RVfpgaPath]/RVfpga/src/SweRVolfSoC/Interconnect/AxiInterconnect/pulp-platform.org__axi_0.25.0/include
[RVfpgaPath]/RVfpga/src/OtherSources/pulp-platform.org__common_cells_1.20.0/include
```

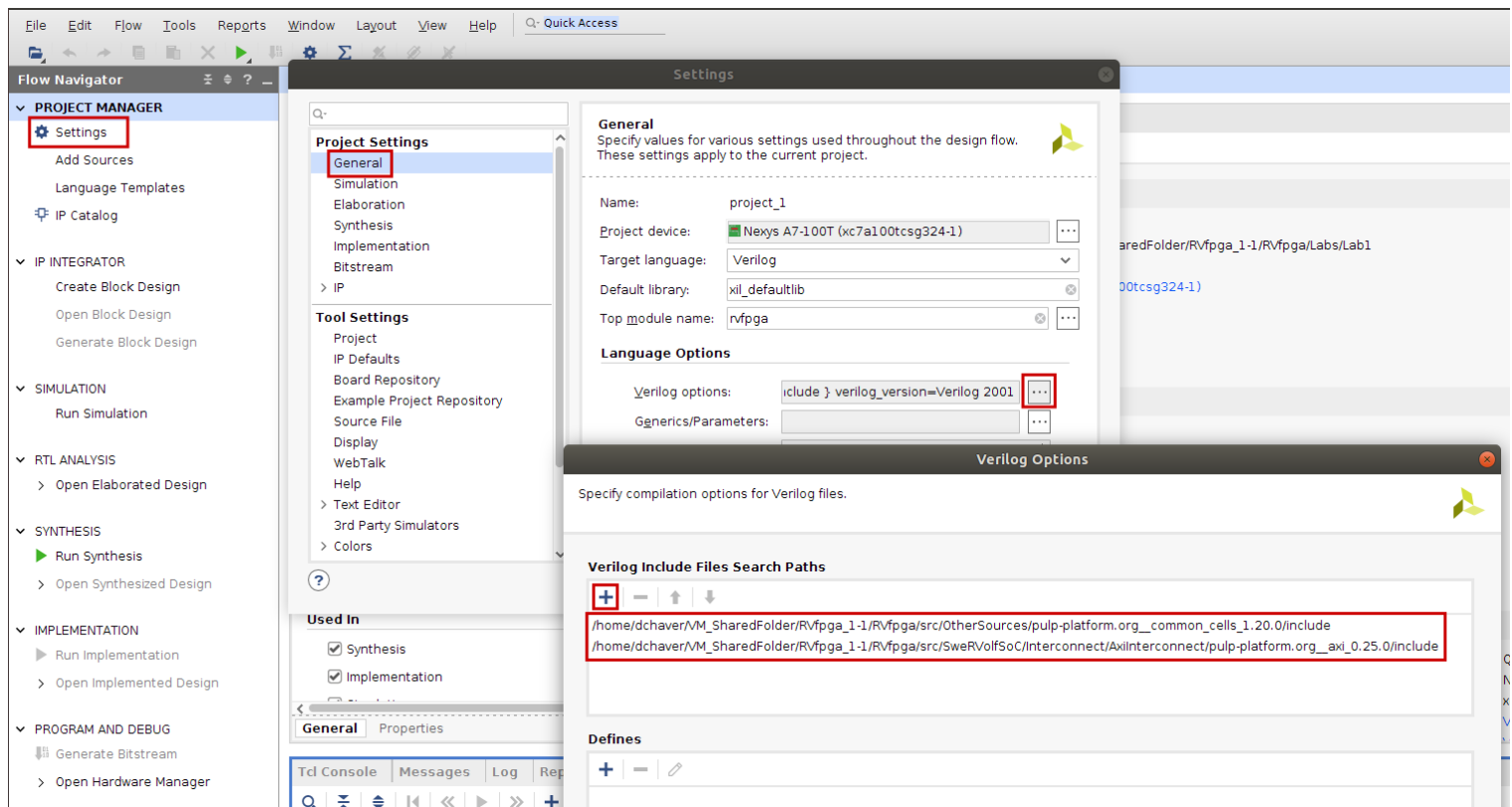


Figure 12. Incluir directories para la Pulp Platform

Paso 6. Generar el *Bitstream*

Haga clic en Flujo (*Flow*)→ Generar *Bitstream* (*Generate Bitstream*), como se muestra en la Figura 13. Puede aparecer una ventana que comuniqué que no hay resultados de implementación disponibles y solicite permiso para lanzar la síntesis y la implementación (ver Figura 14). Haga clic en Sí (Yes). A continuación, haga clic en OK en la ventana de Lanzamiento de ejecuciones (Launch Runs) (ver Figura 15). Este paso sintetiza RVfpgaNexys (conforme a lo definido en los archivos Verilog y SystemVerilog del proyecto), lo mapea en la FPGA y crea el *Bitstream*. Este proceso típicamente toma 20-50 minutos, dependiendo de la velocidad de su computadora.

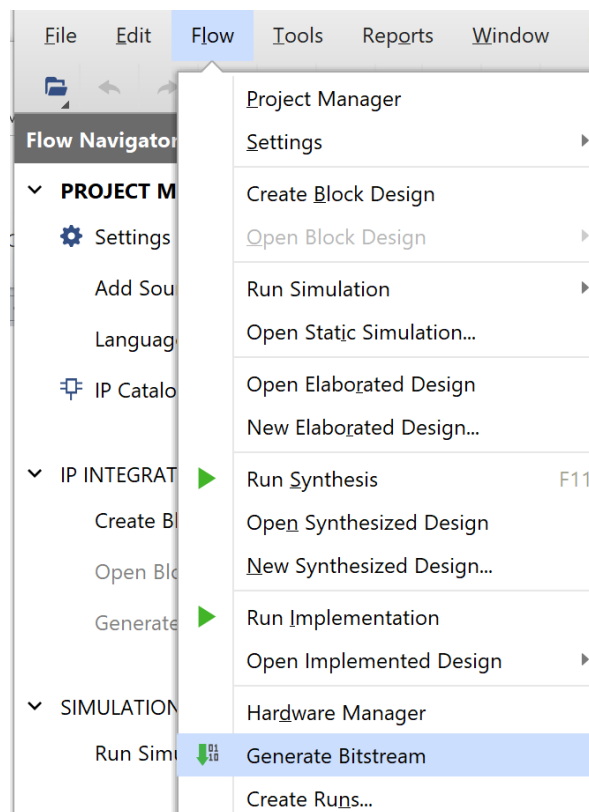


Figura 13. Generación del *Bitstream*

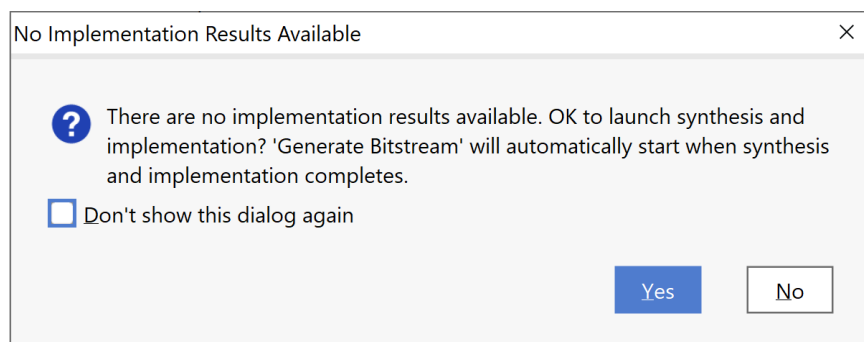


Figura 14. Ventana de lanzamiento de la síntesis y la implementación

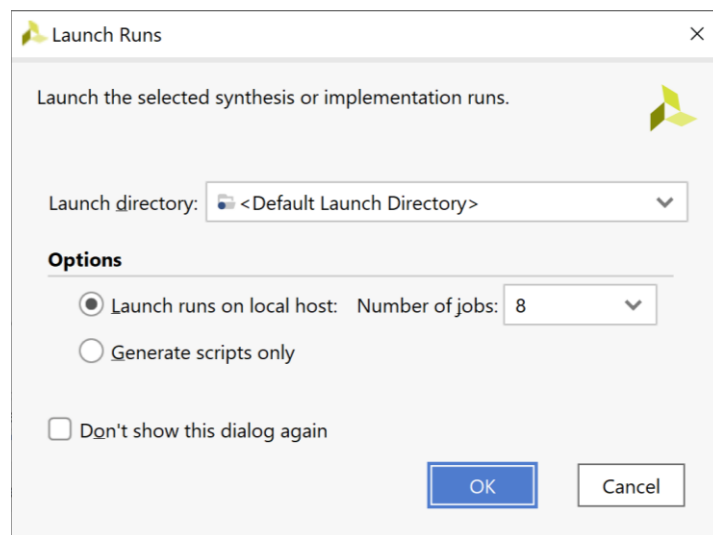


Figura 15. Lanzamiento de ejecuciones

Después de que se haya generado el *Bitstream*, aparecerá una ventana como la que se muestra en la Figura 16. Haga clic en el botón **X** de la esquina superior derecha para cerrar la ventana.

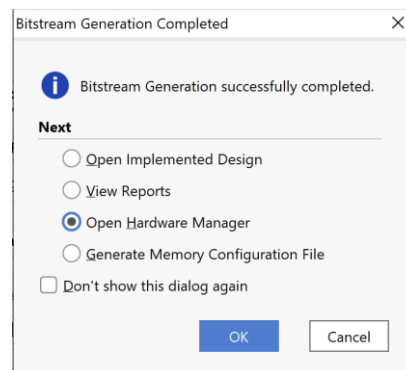


Figura 16. Generación del *Bitstream* finalizada

Ahora que ha construido el sistema RVfpgaNexys por usted mismo, podrá reconstruirlo después de modificarlo en las prácticas 6-10. Por ahora comenzará a usar el sistema RVfpgaNexys que acaba de crear, para descargar y ejecutar programas en él usando PlatformIO.

Se recomienda usar PlatformIO para descargar RVfpgaNexys en la placa Nexys A7. Este método se describió en detalle en la sección 6.A de la Guía de inicio de RVfpga (GSG). Como también se describe en la GSG para los diferentes ejemplos (6.B a 6.H), después de descargar RVfpgaNexys en la FPGA de la placa Nexys A7, se utilizará PlatformIO para descargar y ejecutar/depurar programas en RVfpgaNexys.

También puede usar Verilator, un simulador HDL, para simular la ejecución de programas sobre RVfpgaSim, tal y como se describe en la sección 7 de la Guía de inicio de RVfpga. Estas simulaciones a nivel RTL permiten visualizar señales de hardware de bajo nivel según se ejecuta el programa de software. Se utilizará Verilator en gran medida en las prácticas 6-10, conforme se vaya extendiendo el Sistema RVfpga y se prueben y depuren los cambios realizados.