	Universidad de Costa Rica Escuela de ingeniería Eléctrica Laboratorio 1	EIE Escuela de Ingeniería Eléctrica
IE0424: Laboratorio de Circuitos Digitales II-2019		

Objetivo General

Introducir las herramientas que serán utilizadas durante el curso mediante una práctica introductoria sobre la síntesis de código HDL en un FPGA.

Objetivos específicos

- Familiarizar al estudiante con el ambiente integrado de desarrollo Vivado de Xilinx.
- Presentar al estudiante una implementación en HDL de un microprocesador de la arquitectura RISC-V.
- Presentar al estudiante un ejemplo sencillo en código C para ser ejecutado en el microprocesador.

Propuesta del problema

Se debe corroborar el funcionamiento correcto de un código HDL que implementa un procesador de la arquitectura RISC-V. Se da también un código en C que debe ser compilado a la arquitectura señalada. Este código es un contador infinito que es desplegado en 8 leds en la tarjeta de desarrollo.

El código para esto está disponible en https://gitlab.com/jsdanielh/inicio-ie424. Se debe contar con una cuenta en Gitlab para accesarlo.

Anteproyecto

- De los conceptos aprendidos en los cursos de Estructuras de Computadoras Digitales, repase y mencione los siguientes conceptos:
  - ¿Qué es pipelining?
  - ¿Cuáles son las etapas clásicas de un pipeline de un procesador RISC?
  - ¿Qué son riesgos que atentan contra el pipeline? Mencione y explique algunos.
  - ¿En qué consiste el riesgo de read after write?
- Investigue acerca de la arquitectura RISC-V. ¿Qué elementos distinguen esta arquitectura de cualquier otra arquitectura RISC? ¿Cuál es el set de instrucciones de una arquitectura RISC-V de 32 bits?
- Investigue qué es compilación cruzada (cross compiling). ¿Para qué es necesario?

- Investigue cómo funcionan los LEDs de siete segmentos de la tarjeta Nexys 4 DDR y describa los pasos que se necesitan para desplegar números.

Requisitos

- Contar con Vivado previamente instalado.

Arquitectura

La implementación del procesador RISC-V que se va a usar en este laboratorio se llama picorv32. Este es código fuente abierto que se puede localizar en https://github.com/cliffordwolf/picorv32.

Procedimiento

Primero, es necesario construir el *toolchain* para la arquitectura RV32I. Para no compilar el toolchain desde cero, se va a utilizar un ambiente de desarrollo basado en contenedores (Investigación extra: Investigar cuáles son la diferencias entre máquinas virtuales y contenedores).

Asimismo, se usará un administrador de ambientes de desarrollo llamado V*agrant*. Para ello, es necesario instalar en su sistema los siguientes paquetes usando:

```
$ sudo apt install vagrant docker.io
$ sudo usermod -a -G docker $USER
```

Nota: El último comando agrega el usuario al grupo *docker*. Para que este efecto, proceda a hacer *logout* y luego a hacer *login* de nuevo.

Obtenga el código inicial del proyecto. Para ello, ejecute en una terminal:

```
$ git clone https://gitlab.com/jsdanielh/inicio-ie424.git
```

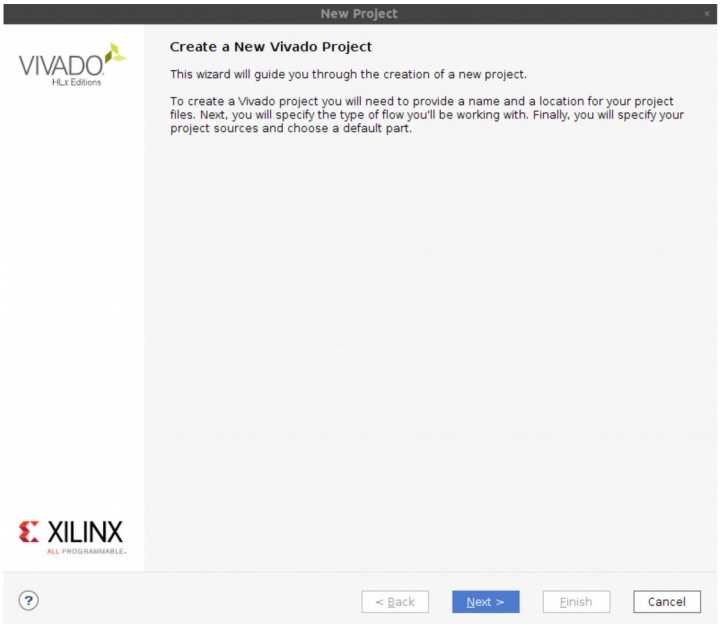
Compile el código que se correrá en el procesador que se va a sintetizar. Para ello, levante el ambiente de desarrollo (que por debajo levantará un contenedor) y compile el código fuente en *src/firmware* para la arquitectura RISC-V de 32 bits. Esto lo puede hacer ejecutando los siguientes comandos:

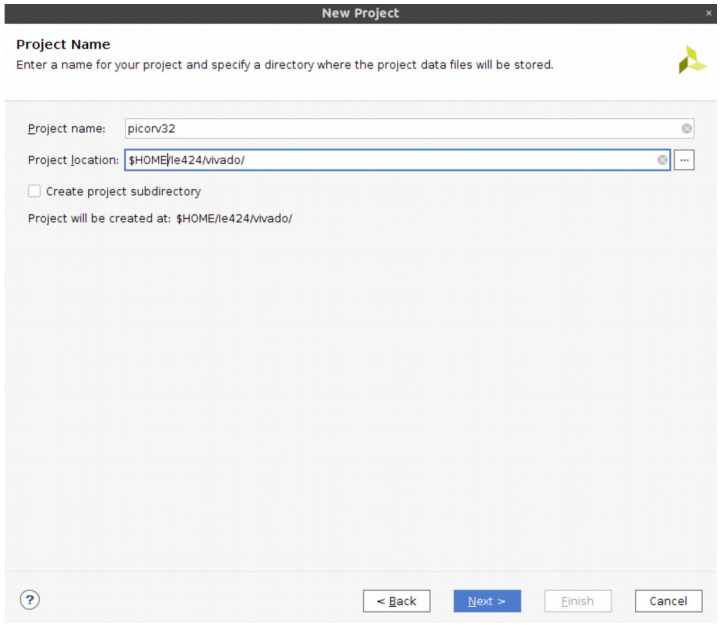
```
$ cd inicio-ie424 # El directorio que contiene el código clonado
$ vagrant up # Este proceso puede tardar varios minutos
$ vagrant ssh
$ cd ws/src/firmware/
$ export PATH=$PATH:/opt/riscv32i/bin/
$ make
```

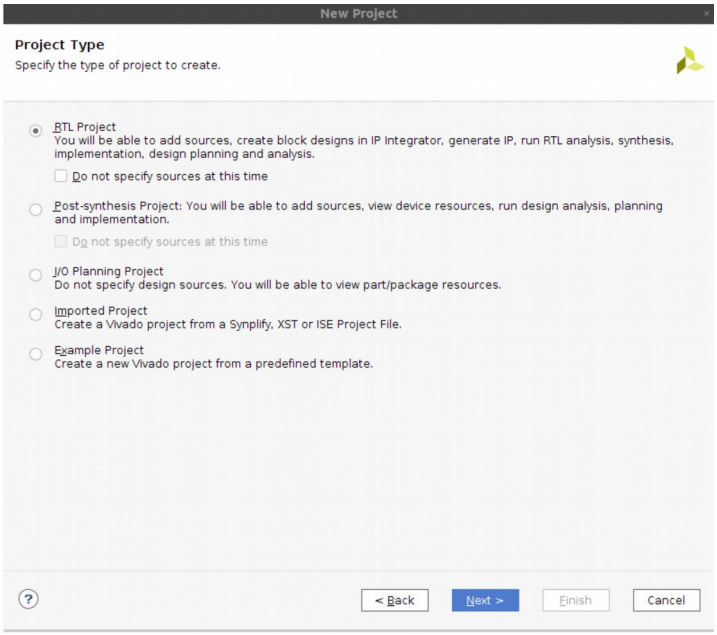
Asegúrese que los comandos anteriores no hayan presentado errores. En caso de éxito, se le debió haber generado el archivo *firmware.hex* dentro del directorio *src/firmware*.

Inicie Vivado y cree un nuevo proyecto. Para ello, siga los siguientes pasos:

1. Haga clock en ‘Create project’ y siga las instrucciones que se presentan:





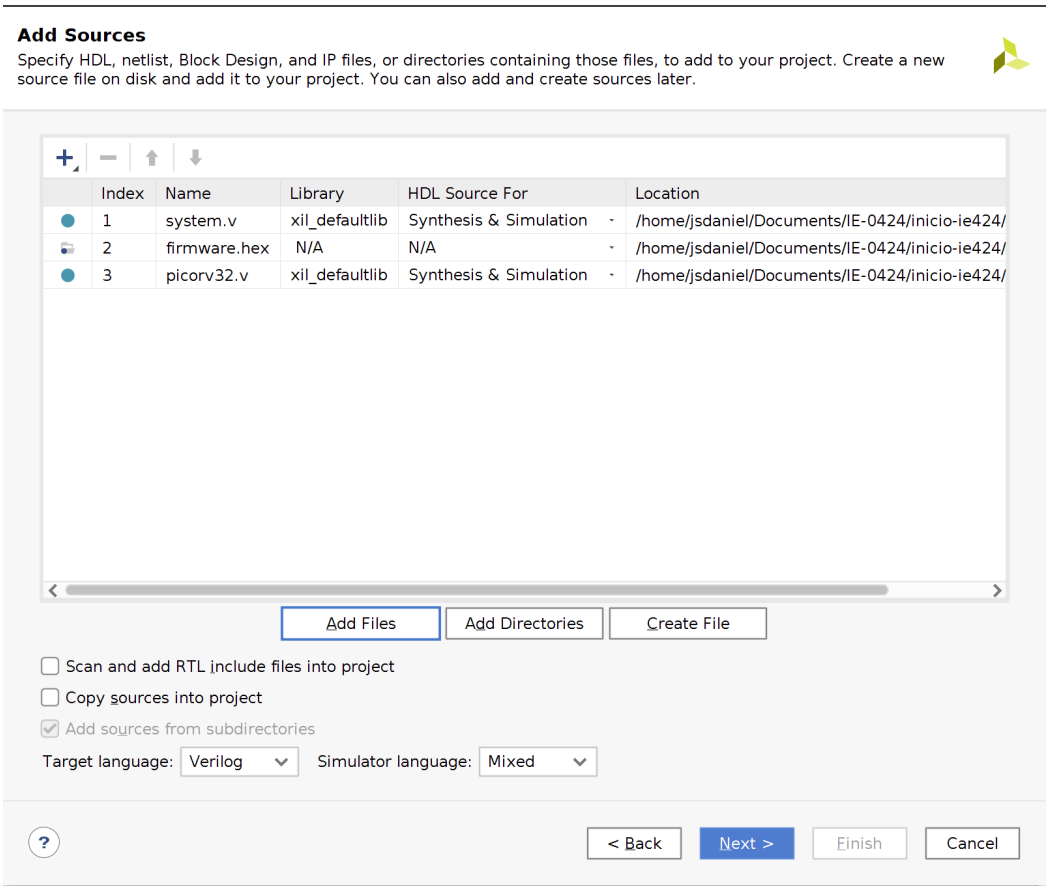


2. Agregue los siguientes archivos como fuentes:

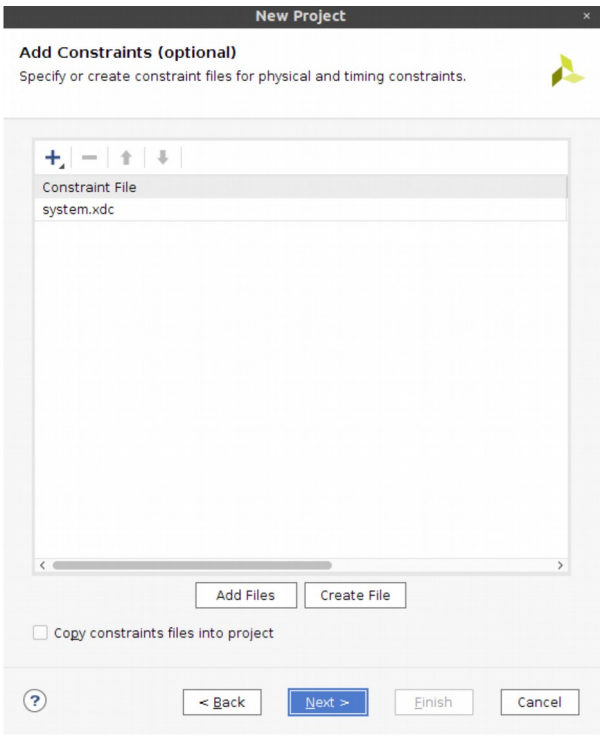
   a) src/picorv32/picorv32.v

   b) src/verilog/system.v

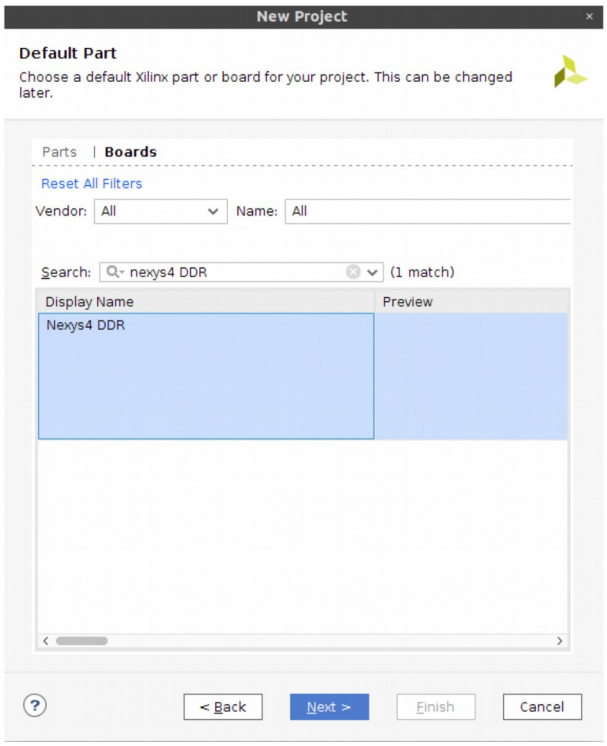
   c) src/firmware/firmware.hex



3. Agregar los siguientes archivos como constraints:
    a) src/constraints/system.xdc

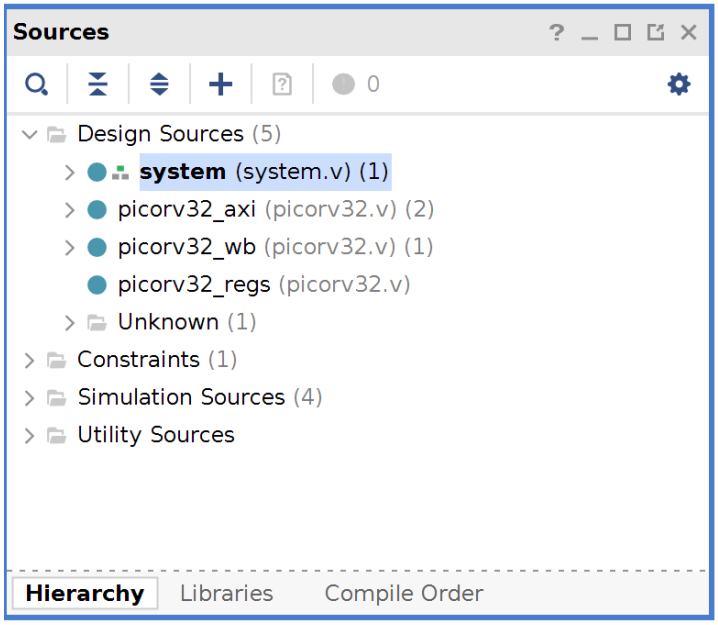


4. Seleccionar la tarjeta Nexys4 DDR como parte por defecto.

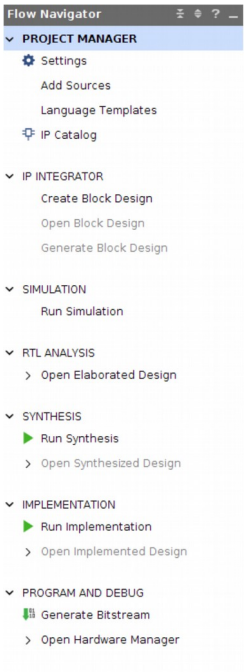


5. Continuar con las instrucciones que se le presenten.
6. Asignar el módulo *system* como top.

a) Para ello, haga click derecho sobre el módulo *system* y seleccione '*Set as Top*'.



Una vez generado el proyecto, haga click en '*Generate Bitstream*' para generar el archivo *.bit* a utilizar en la tarjeta de desarrollo.



Finalizada la generación del archivo .bit:

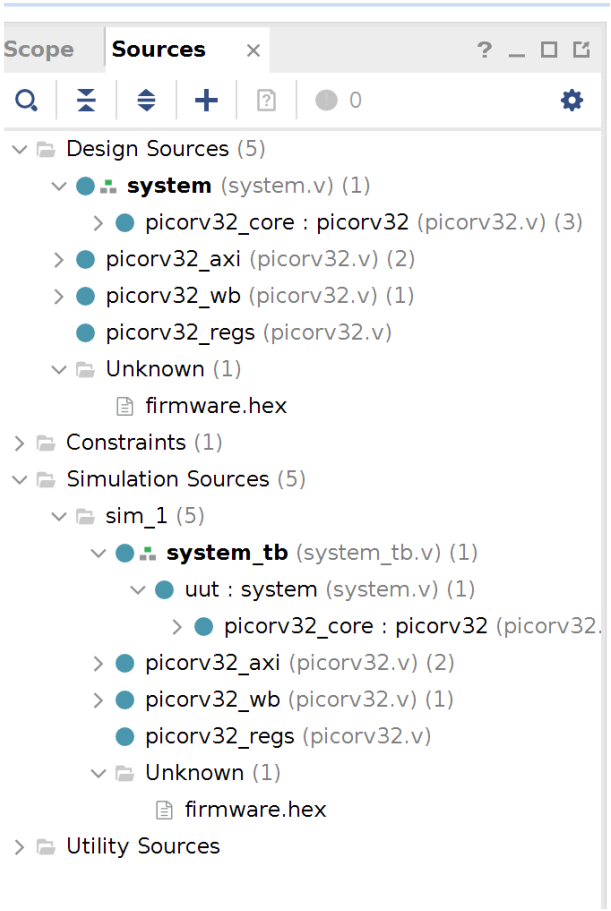
1. Conecte la tarjeta de desarrollo al computador.
2. Abra el 'Hardware Manager'.
3. Seleccione 'autoconectar dispositivo'.
4. Seleccione 'Programar dispositivo'.

5. Al haberse programado el dispositivo se debe aplicar la señal de reset.
    a) Esto se hace con el switch V10.
6. En caso de éxito, deberá observar un contador en 8 de los LEDs contiguos a los switches.

Note que cada vez que realiza alguna acción dentro de Vivado, en la ventana 'Tcl Console' se imprimen ciertos comandos y ciertas salidas. El programa se puede manejar utilizando esta consola y es la base para permitir crear el proyecto con un script (opcional).

Para hacer simulaciones, realice los siguientes pasos:

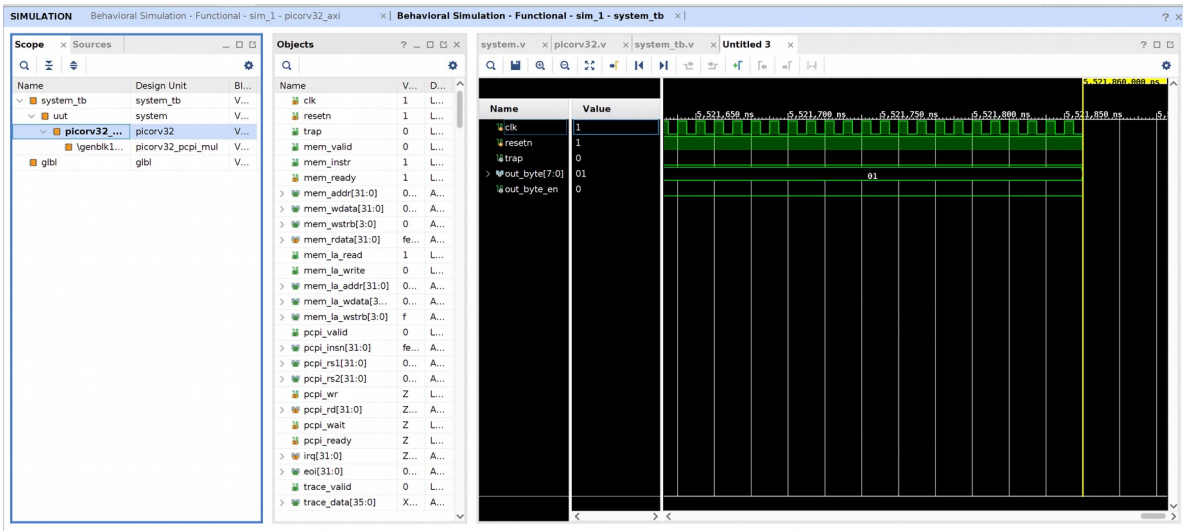
1. Agregue los siguientes archivos como archivos de simulación:
    a) src/verilog/system_tb.v



2. Procure que todos los archivos se muestren como en la imagen anterior. Si *firmware.hex* no aparece como archivo de simulación, agréguelo como archivo de simulación.
3. Luego de esto, proceda a lanzar una simulación por comportamiento.
    a) Para ello vaya al panel izquierdo bajo las opciones de 'Simulación'.

b) Dé click en ‘Correr simulación’ y luego escoja la opción de ‘Simulación por comportamiento’.

4. En caso de éxito, la simulación se levantará y deberá ver una ventana como esta:



5. Para correr la simulación utilice los botones de control en la parte superior. Luego de correr la simulación utilice el botón de ‘Reiniciar’ y luego proceda a correr la simulación por algún tiempo. Puede usar el botón de ‘Correr’ y luego pararlo después de un tiempo o el botón de ‘Correr por X tiempo’.

6. Utilice el panel de ‘Scope’ y ‘Objects’ para navegar los diferentes módulos, registros y cables del diseño. Por ejemplo, navegue el procesador Picorv32 y agregue cables o registros de forma tal que los pueda ver en el visualizador de ondas.

Ejercicio 1 (Obligatorio):

Describa la interacción que existe entre el firmware, el procesador Picorv32 y los LEDs. ¿Cómo inluye el firmware en la cuenta que se despliega en los LEDs? ¿Cómo está compuesto el espacio de memoria que ve el procesador Picorv32?

Ejercicio 2 (Obligatorio):

Puede notar que el número que se despliega en los LEDs cambia muy lentamente. Modifique el código en la para que este cambio sea más rápido.

Utilice una simulación que demuestre los efectos del cambio realizado.

Ejercicio 3 (Obligatorio):

Realice los cambios necesarios para desplegar el contador utilizando el display de 7 segmentos de la tarjeta de desarrollo. Para esto deberá de cambiar el archivo de constraints. Investigue para qué sirve el archivo de constraints y cómo se puede modificar este archivo desde la interfaz gráfica. Realice los cambios necesarios.

Ejercicio 4 (Obligatorio):

Proceda a lanzar una simulación.

1. Utilizando una simulación demuestre que la implementación del procesador sigue un diseño de pipelining. Identifique las etapas del pipeline y demuestre para las primeras 10 instrucciones cuáles etapas se llegan a ejecutar. Justifique los resultados obtenidos.
2. Utilizando una simulación identifique un caso de read after write. Indique si el caso se maneja correctamente y si tiene algún efecto en el pipeline. Describa cómo se maneja el caso.

Ejercicio 5 (Opcional):

1. Identifique los warnings de las distintas etapas e indicar si son o no de importancia. ¿Cómo se podrían corregir?

Referencia

Rojas, E. (2019) *Síntesis de un procesador RISC-V en tarjetas de desarrollo Nexys 4 DDR*. San José, Costa Rica: Escuela de Ingeniería Eléctrica, Universidad de Costa Rica.