Real-Time Imaging and Control

Lab

MSCV/ESIREM

Antoine Lavault
antoine.lavault@u-bourgogne.fr

Lab 1 - Quickstart with Vivado and VHDL

This goal of this lab are :

- Find technical informations in a manual or datasheet
- Get to know the basic functions of Vivado IDE
- Generate a bit-stream and integrating it on a FPGA board
- Learn the basics of FPGA development flow.
- Use a test-bench as part of a VHDL workflow.

You will use Windows for all of the lab sessions.
Sections marked with a star (⋆)

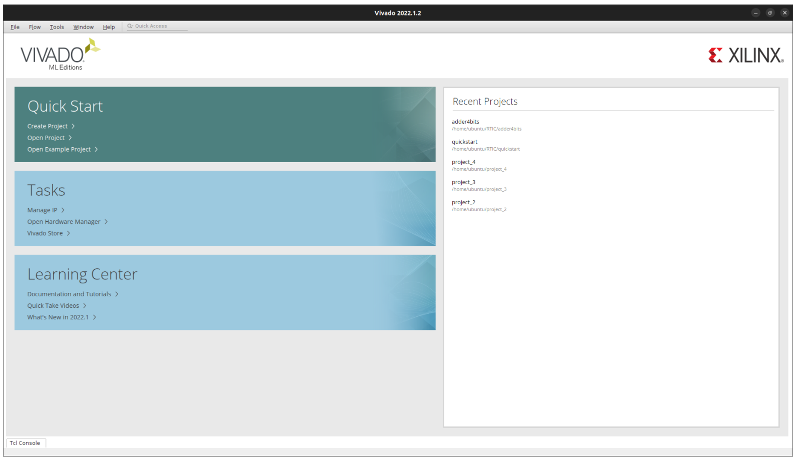
1 Meet the Nexys 4

The Digilent Nexys 4 board reference manual is available here : `https://digilent.com/reference/programmable-logic/nexys-4/reference-manual`
Fill in the quiz

2 Vivado IDE

2.1 Create a new project with Vivado 2021.1

1. Download the require files from Teams. You should have:
   - A file named `quickstart.vhd`
   - A file named `cstr.xdc`
   - A folder named `board_files`
2. Place the contents of the `board_files` folder into. `C:/Xilinx/Vivado/2021.1/data/boards/b`
   Create the folder if it doesn't already exist.
3. Open Vivado 2021.1 and create a new project (figure 1)
4. Create a new RTL project (figure 2)



adder4bits
/home/ubuntu/RTIC/adder4bits

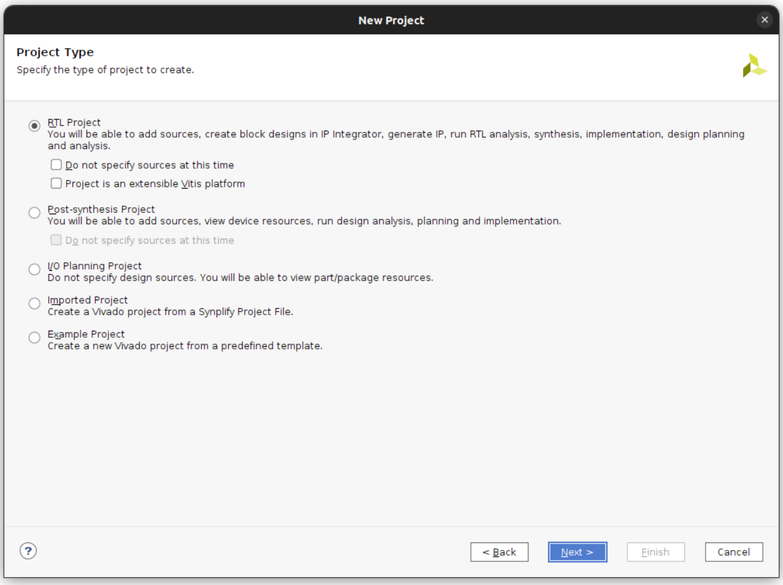
quickstart
/home/ubuntu/RTIC/quickstart

project_4
/home/ubuntu/project_4

project_3
/home/ubuntu/project_3

project_2
/home/ubuntu/project_2

Figure 1: Welcome Screen of Vivado



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

□ Project is an extensible Vitis platform

○ Post-synthesis Project
You will be able to add sources, view device resources, run design analysis, planning and implementation.

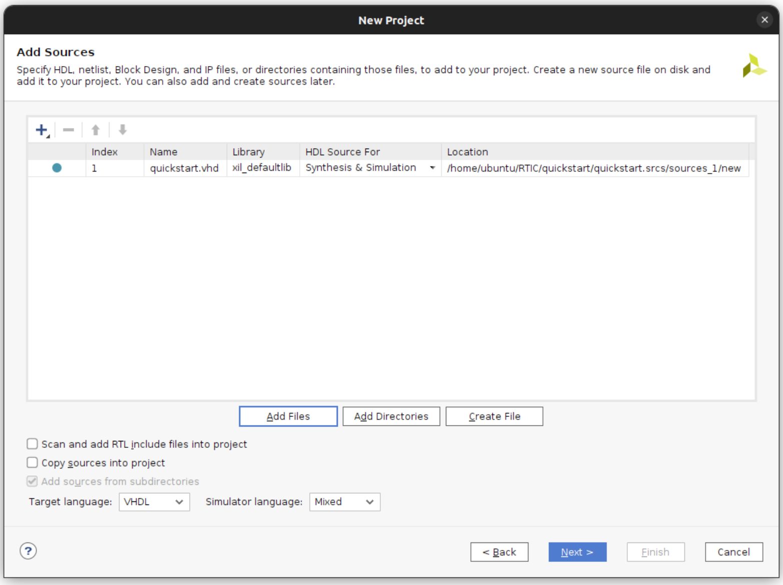
□ Do not specify sources at this time

○ I/O Planning Project
Do not specify design sources. You will be able to view part/package resources.

○ Imported Project
Create a Vivado project from a Synplify Project File.

○ Example Project
Create a new Vivado project from a predefined template.

Figure 2: Options to create a project



Add Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those files, to add to your project. Create a new source file on disk and add it to your project. You can also add and create sources later.

	Index	Name	Library	HDL Source For	Location
●	1	quickstart.vhd	xil_defaultlib	Synthesis & Simulation	/home/ubuntu/RTIC/quickstart/quickstart.srcs/sources_1/new

□ Scan and add RTL include files into project

□ Copy sources into project

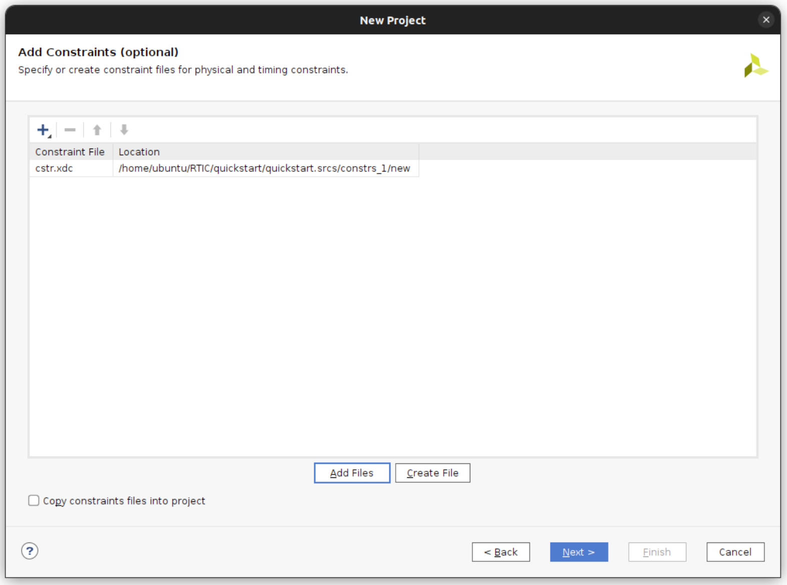
☑ Add sources from subdirectories

Target language: VHDL Simulator language: Mixed

Figure 3: Adding a source file to the project

5. Add the `quickstart.vhd` file to the project (figure 3)

6. Add the `cstr.xdc` file to the constraints (figure 4



Add Constraints (optional)

Specify or create constraint files for physical and timing constraints.

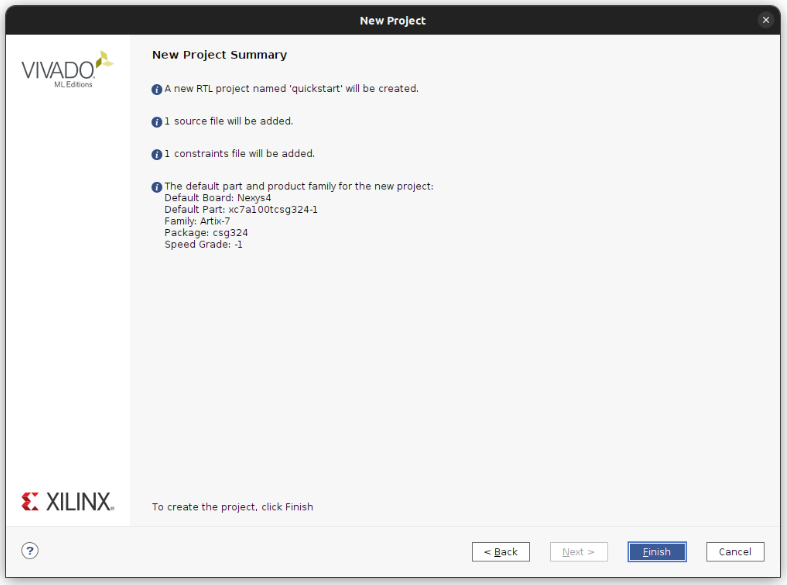
Constraint File	Location
cstr.xdc	/home/ubuntu/RTIC/quickstart/quickstart.srcs/constrs_1/new

□ Copy constraints files into project

Figure 4: Adding a constraint file to the project

7. Choose the `Nexys4` in the hardware target (under the "Board" tab)

8. The creation assistant is now done. Check your project summary to the figure 5



New Project Summary

A new RTL project named 'quickstart' will be created.

1 source file will be added.

1 constraints file will be added.

The default part and product family for the new project:
Default Board: Nexys4
Default Part: xc7a100tcsg324-1
Family: Artix-7
Package: csg324
Speed Grade: -1

To create the project, click Finish

Figure 5: Project Summary

2.2 Development Workflow

1. First, check the RTL Analysis. You can check the schematic produced by the tool.

2. Then, launch the synthesis tool.

3. Then, the implementation.

4. Finally, generate the bit-stream file.

2.3 Programming the Nexys 4

Now that we have a compatible bitstream, we will program the Nexys 4 board.

> WARNING ⚠:
> the board will flash its LEDs with an intensity proportional to the activated switches (MSB on the leftmost switch).
> Use the row of switches from right to left until you find an acceptable brightness to avoid discomfort and short-term glare.

1. The card is plugged in and the power is switched on.

2. Open the "Hardware Manager" and choose "Auto connect". You should see Vivado recognise the card.

3. Choose the "Program Device" option.

4. Once programming is acheived, you can play with the switches and button.

The expected behaviour is as follow:

- The LEDs above the switches are on when the switch is in the "high" position.

- The RGB LEDs should blink with an intensity proportionnal to the binary number encoded by the switches

- The north and south buttons will change the color as well as the east and west and finally, center too.

3 VHDL and Simulations

For the following exercise, you will have to complete a snippet of code for a 4-bit adder to match an already available test bench called `4bitadder_tb.vhd`.

3.1 Project Setup

Create a new project for the 4bit adder. Import the source file and the test-bench.
Keep the Nexys 4 as the target hardware.

3.2 Adding a test-bench

Use the provided test-bench to test the adder.

1. Check if the provided code file actually passes the test-bench.

2. Modify the code of `4bitadder.vhd` to output the correct values if needed if need be.

3.3 Hardware implementation

Make sure your code passes through the synthesis and implementation process.
We would like to make use of the switches and LEDs on the board to control the 4-bit adder. We especially want to :

- use the right-most switches (SW3-SW0) to give the value to a
- use the left-most switches (SW15-SW12) to give the value to b
- use the switch 8 (SW8) as the carry in.
- Output the results on the left-most LEDs (LED15-12)
- Output the carry out result on the right-most LED (LED0)

The switches should read a binary number with the MSB to the left (eg. SW15 encodes the MSB when SW12 encodes the LSB). Same for the LEDs.
For this, use the `I/O Ports` tab in the `Implementation` step.

Once you are done, ask for a lab assistant to check your I/O setup.