Introduction

In a lot of digital designs (DAQ, Trigger, ..) the FPGAs are used. The aim of this exercise is to show you a way to logic design in a FPGA. You will learn all the steps from the idea to the test of the design.

This project sets up your FPGA board for use with Xilinx® Vivado™ and shows you the steps in starting project files. It also gives basic background knowledge on using digital circuits. This is a starter project with very little hands-on work with your board, but it is a good reference to understand the HDL(Hardware Descriptor Language) tahts I needed to work with any FPGA .

In this exercise you will:

Program a FPGA from the design idea up to the implementation and test the same. The FPGA board used is Xilinx Artix-7(BASYS-3) based on a small FPGA , with multiple additional interface components like audio CODEC, switches, button, seven-segments display, LEDs …

You need to use a tool (Vivado) for the BASYS 2 Virtex7 Xilinx FPGA programming and the HDL (Hardware Descriptor Language-HDL) can be either VHL or Verilog .In this exercise you will using Verilog .

You need the Vivado tool to work with the FPGA Board. The Vivado tool is preinstalled on the pc /laptop which will be used for connecting to the BASYS 3 Board (Artix-7 FPGA) start-up menu.

The instructions to use Viavdo to starta project and then creade a VHLd/Verilof file is as below:

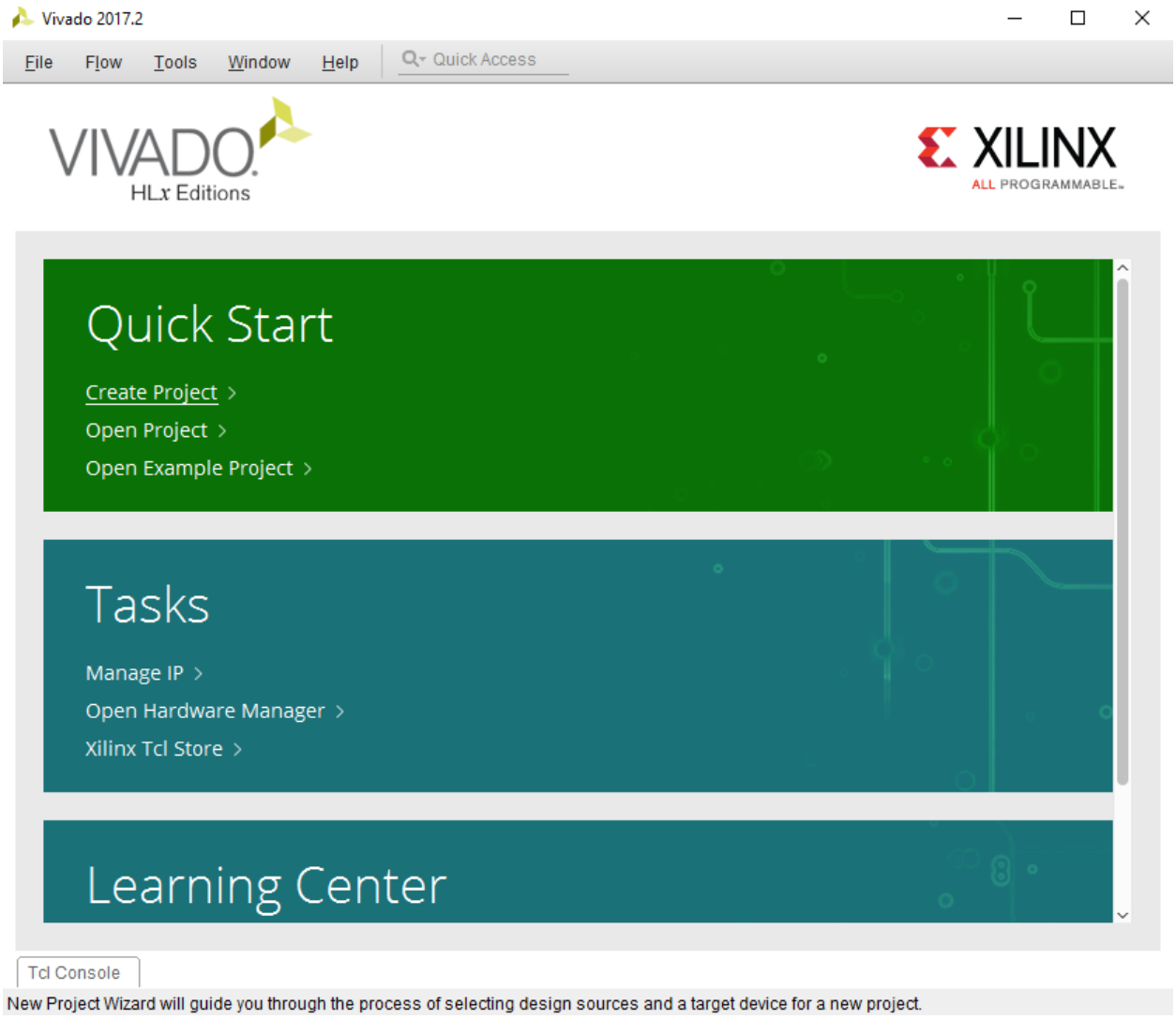


Figure 3. Vivado Start-up Window

Open Create Project Dialog

Click on Create New Project. This will open the new project wizard as shown in Figure 2. Click on Next to continue creating your first project.

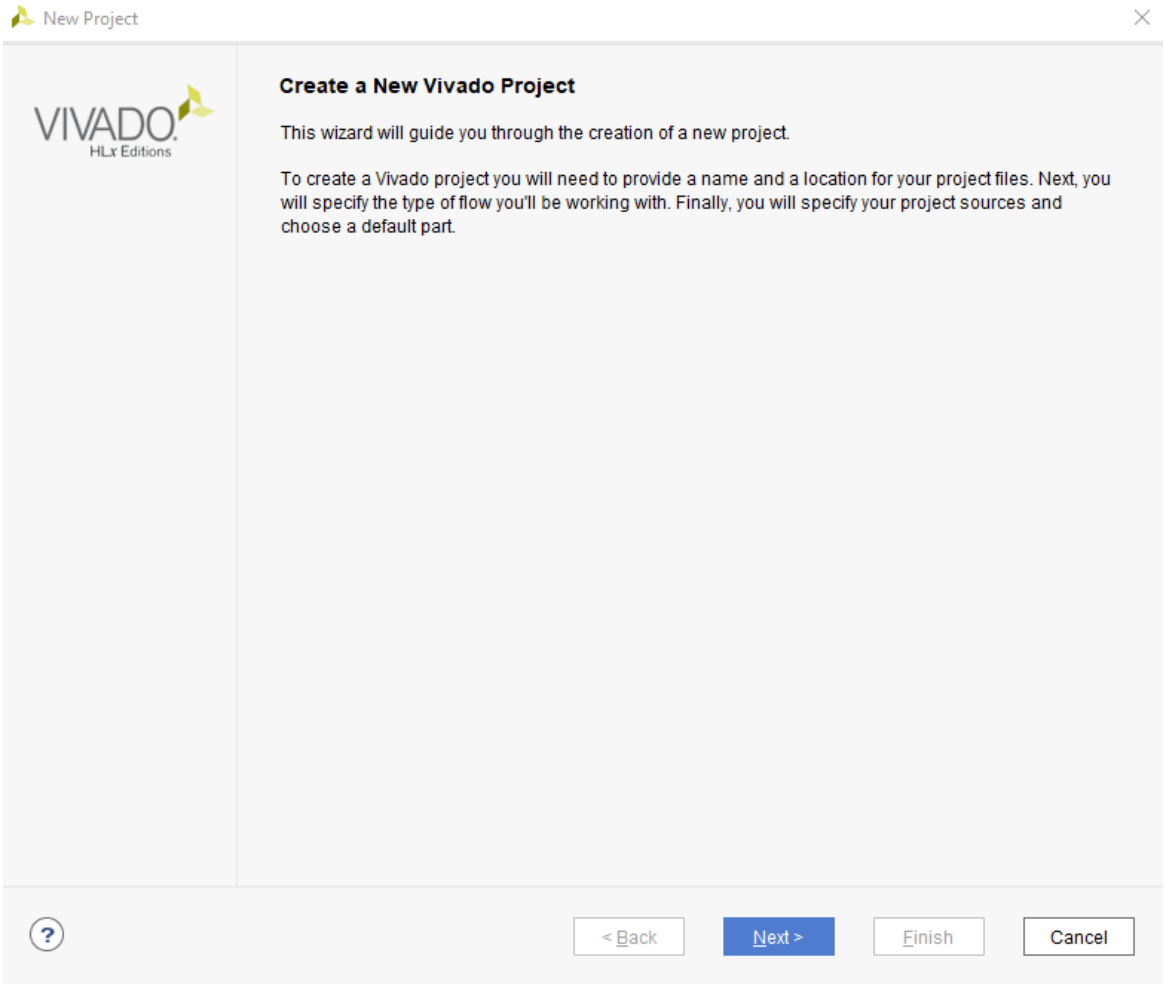


Figure 4. Create Project Dialog

Set Project Name and Location

Enter a name for the project, in this case `project_1`. Actually most of the times, it is more helpful to include both the project number and class project name in the project name. That way when you need to reference the project later, you don't need to remember both what the project was and the project number.

Mind the spaces in the path string

NOTE: it is also recommended that the path of location and working directory does not contain white space. (i.e., `C:\Document and Settings\...` is not recommended as there are spaces in the path. Having white space in the file path might cause XST to fail.)

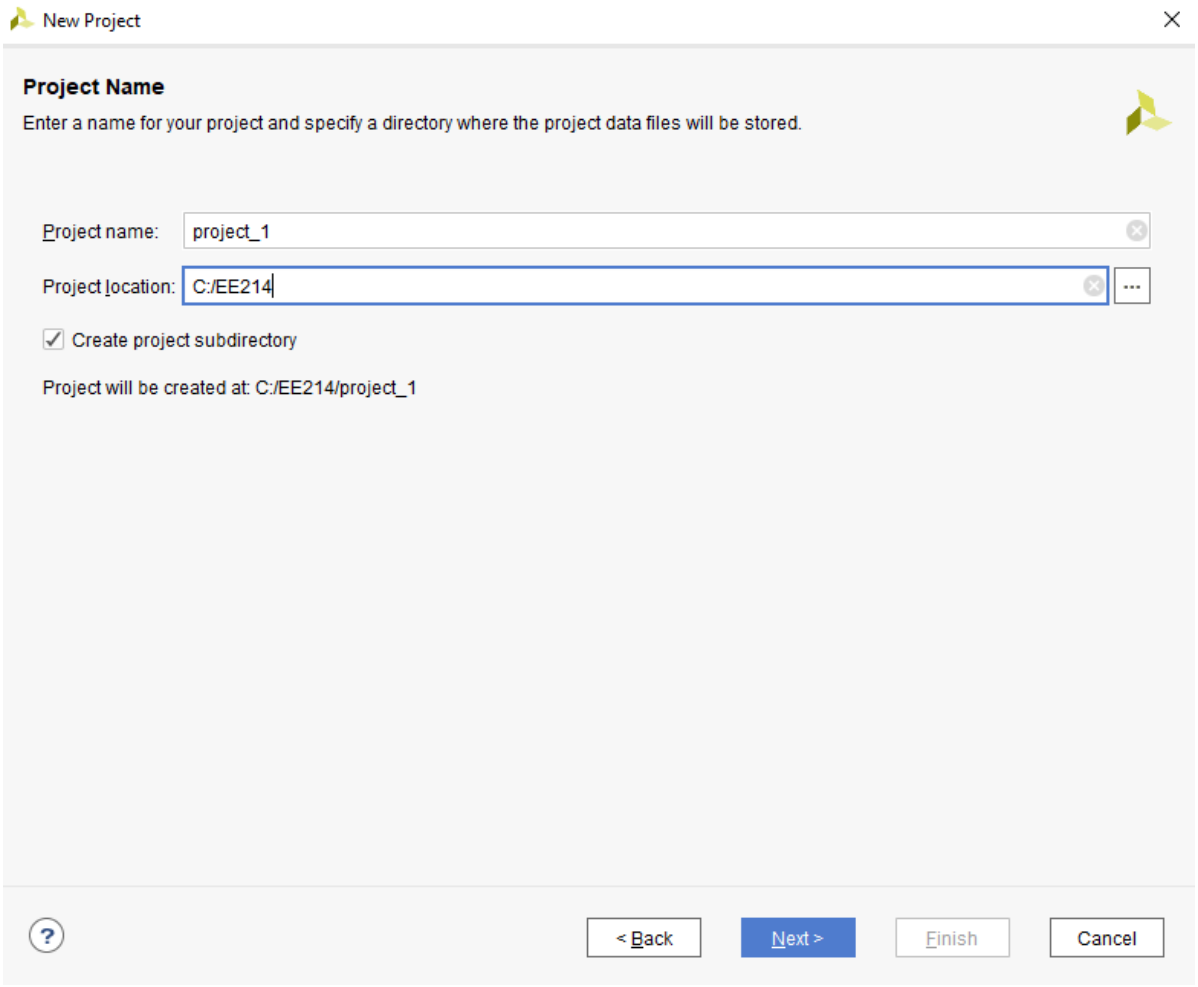


Figure 5. Enter Project Nam

Select Project Type

For this project, we will use the project type RTL Project. Select the RTL Project radio button and hit Next.

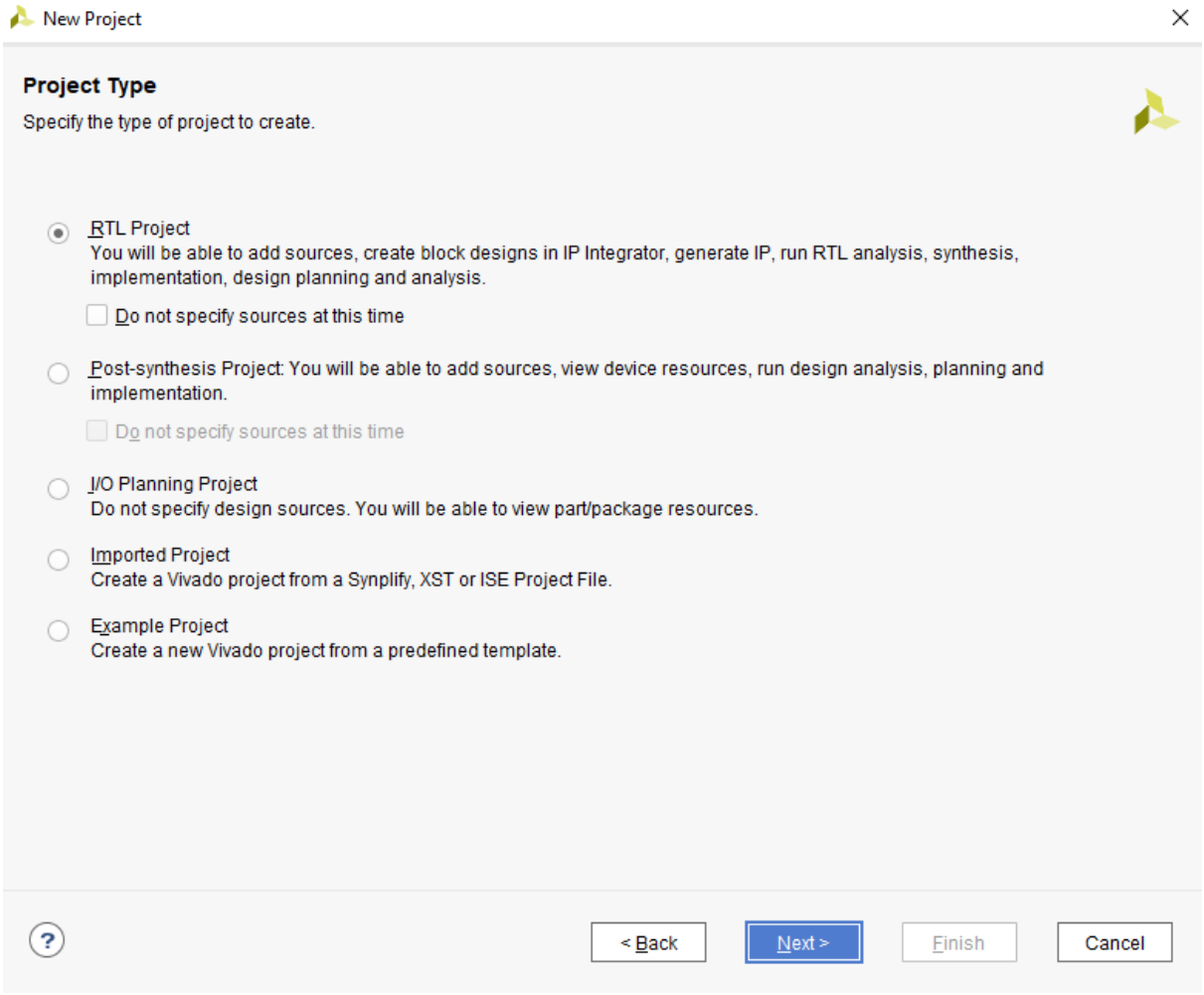


Figure 6. Select Project Type

Add Existing Sources

Next, the project wizard will prompt you to choose source files for your project. In this project, you will add sources files later. Just skip this step by clicking Next to continue. I will show you how to add the source files later in the project. Click **Next** to continue.

Add Sources

In the future, if you have existing source files at the time you create the project, you can click Add Files button to add them or Add Directories button to add all files under selected directories. You can also create new source files by clicking Create File in this screen. However, you can certainly add or create them later outside of the New Project Wizard.

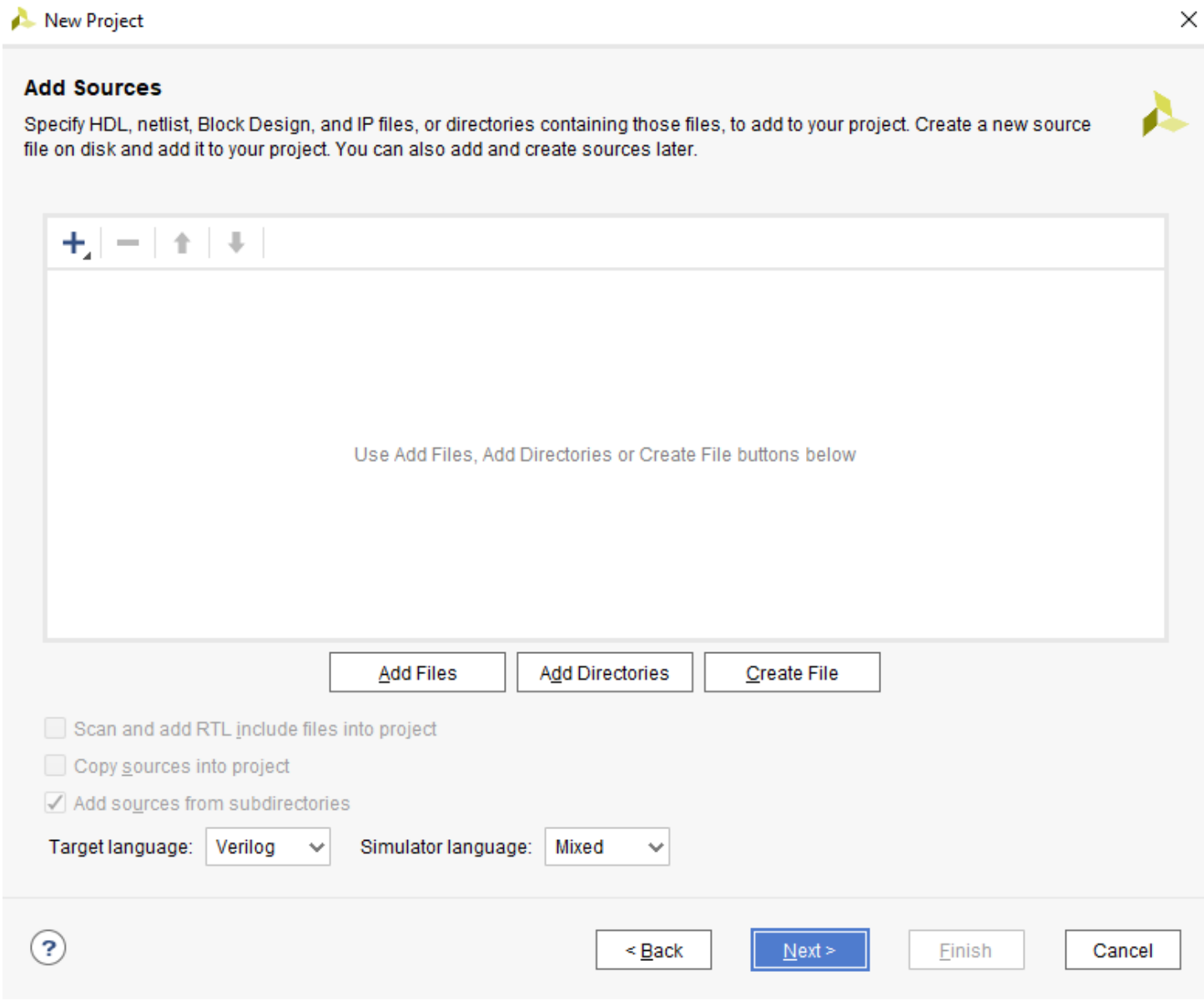


Figure 7. Add Source

Add Constraints

Next, the project wizard will prompt you to add or create constraints file for your project. In this tutorial, you will create the constraint files later. So, at the moment, just skip this step by clicking Next to continue.

Xilinx Design Constraints (XDC)

Xilinx Design Constraints is a file format used for describing design constraints, such as I/O delays, clocks, timing exceptions, and physical constraints. For more information about constraints, you can search Xilinx user guide on using constraints on Xilinx Documentation Page.

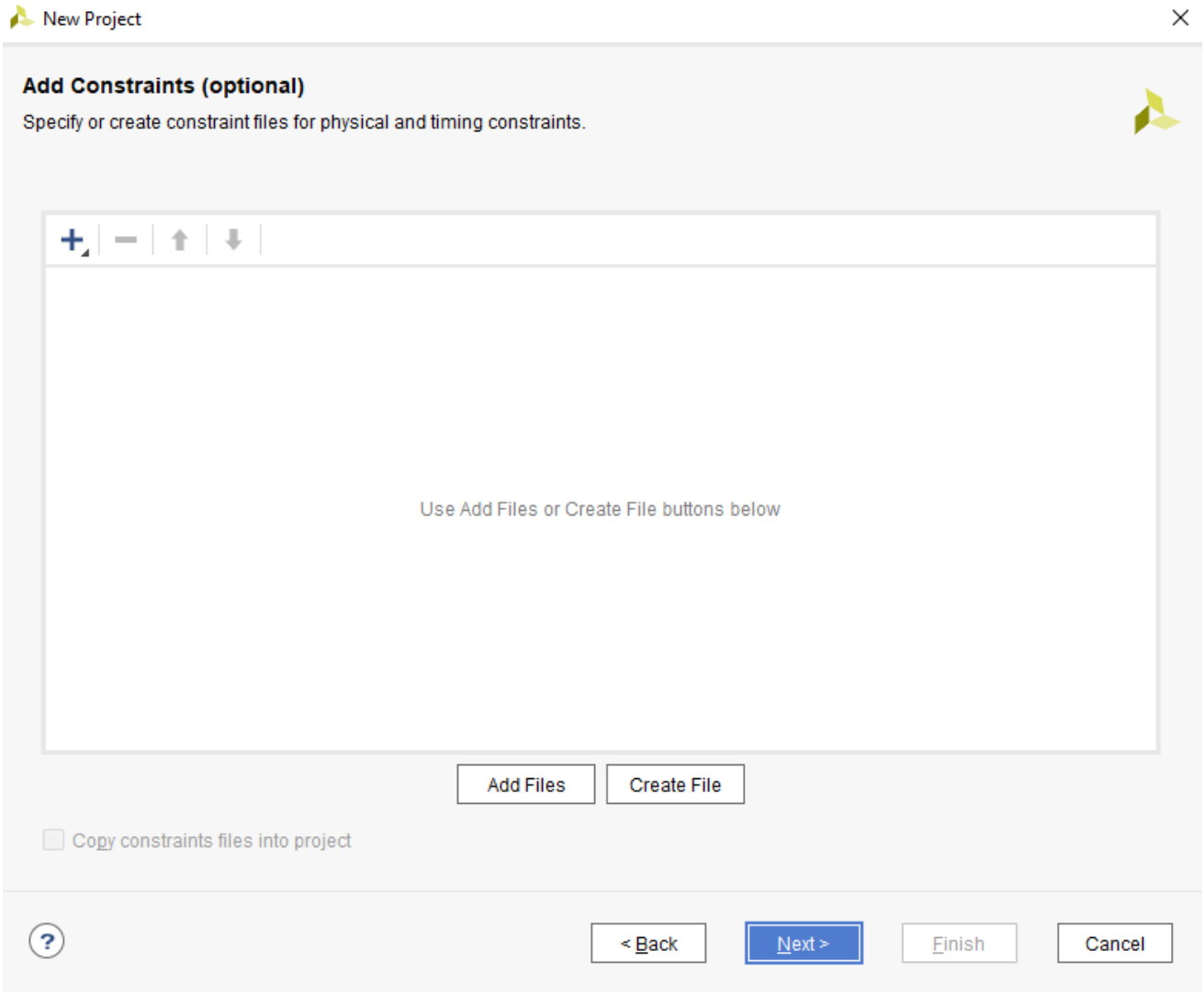
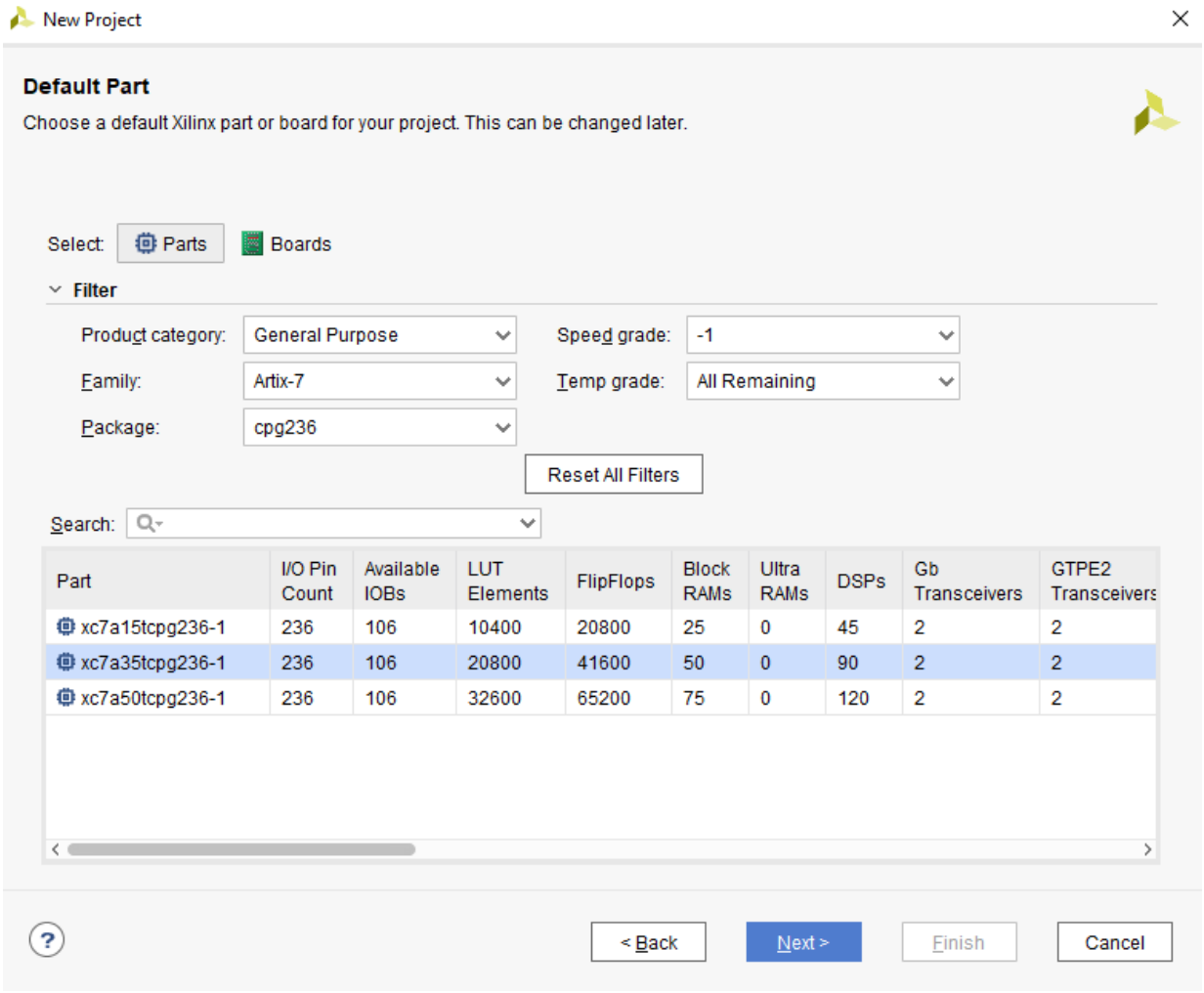


Figure 8. Add Constraint Files

Select Parts

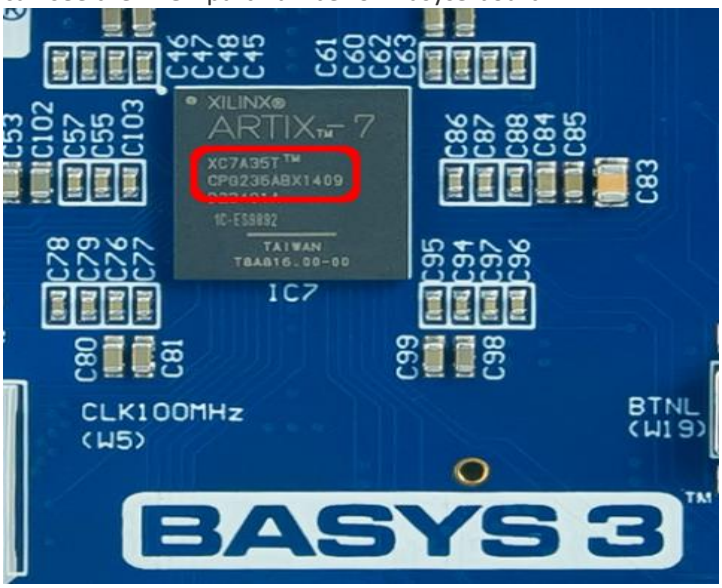
The last bit of information that you need to enter is the details about the FPGA chip on your FPGA board. You will need to know the family, package, speed grade, and temperature grade of your FPGA chip. The table below shows the detailed information of the Artix 7 FPGA Chip on Basys3 board.

Part Number	XC7a35tcpg2361C
Family	Artix-7
Package	cpg236
Speed grade	-1
Temperature Grade	C

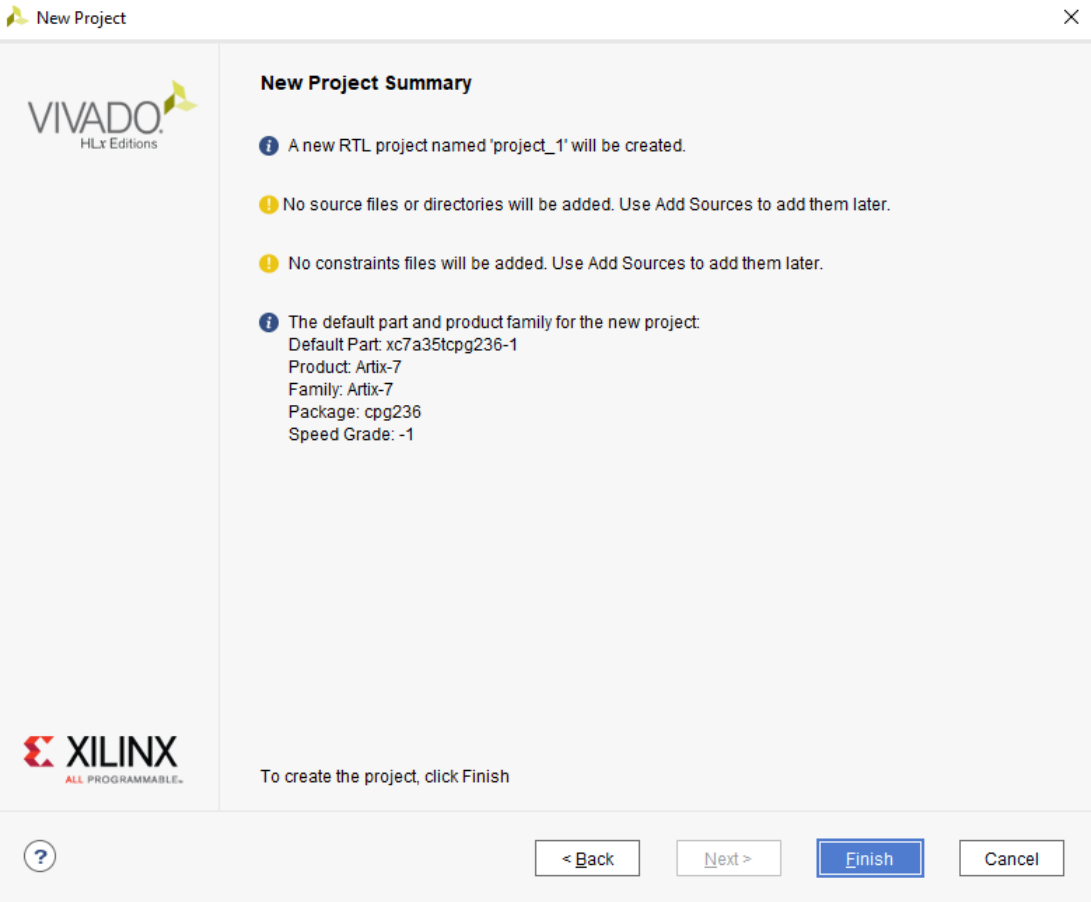


How to find your FPGA Parts

Usually, you can find the part number directly from the marking on the back of IC. In the picture below, you can see the FPGA part number on Basys3 board



On the last page of the Create Project Wizard, a summary of the project configuration is shown. Please verify all the information in the summary and make sure that the correct FPGA part is selected for your project, the project name is correct, the list of files to be included are listed. After verifying all the configuration information, click Finish to finish creating an empty project



To create a verilog design source file for your project, in the Sources panel, select Design Sources, right click on it and select Add Sources...

Design Sources

There are multiple ways to describe a logic circuit design. This series of tutorials focus on using Verilog HDL language to describe a digital system. You may refer to page VERILOG HDL: THE FIRST EXAMPLE for some basic knowledge on Verilog HDL.
Besides Verilog HDL, Vivado tools can take different design sources including VHDL codes, netlist in EDIF and NGC format, DCP design checkpoint files, TCL scripts, etc. For more information on other design input sources, you can refer to Xilinx Vivado Design Flows Overview on Xilinx Document Page

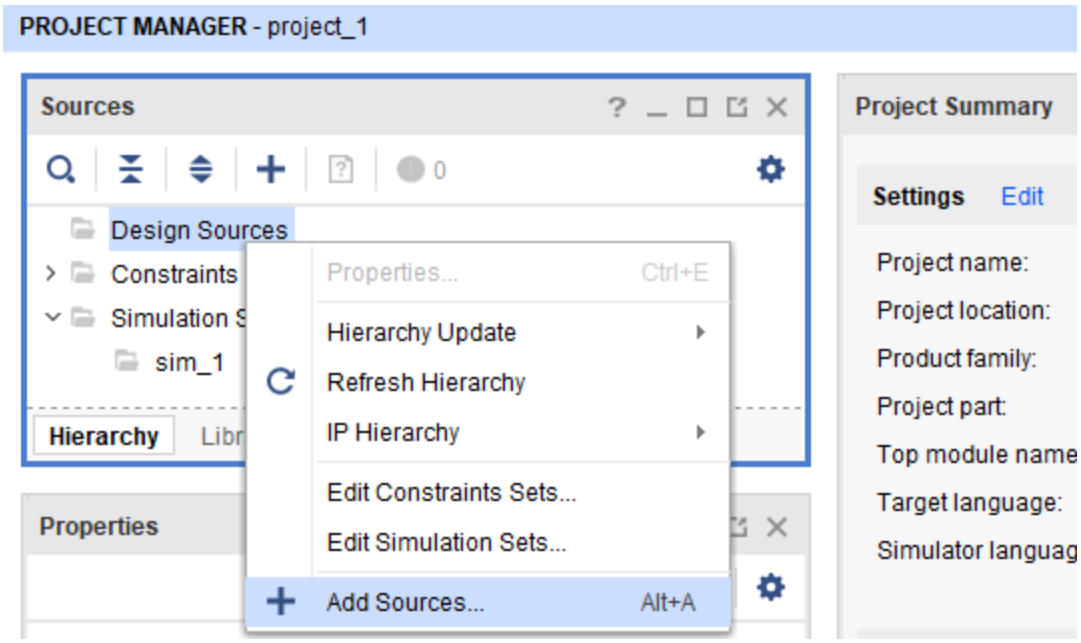


Figure 14. Add Design Sources

An Add Sources dialog will appear as shown in Figure 15. Select Add or create design sources and click Next.

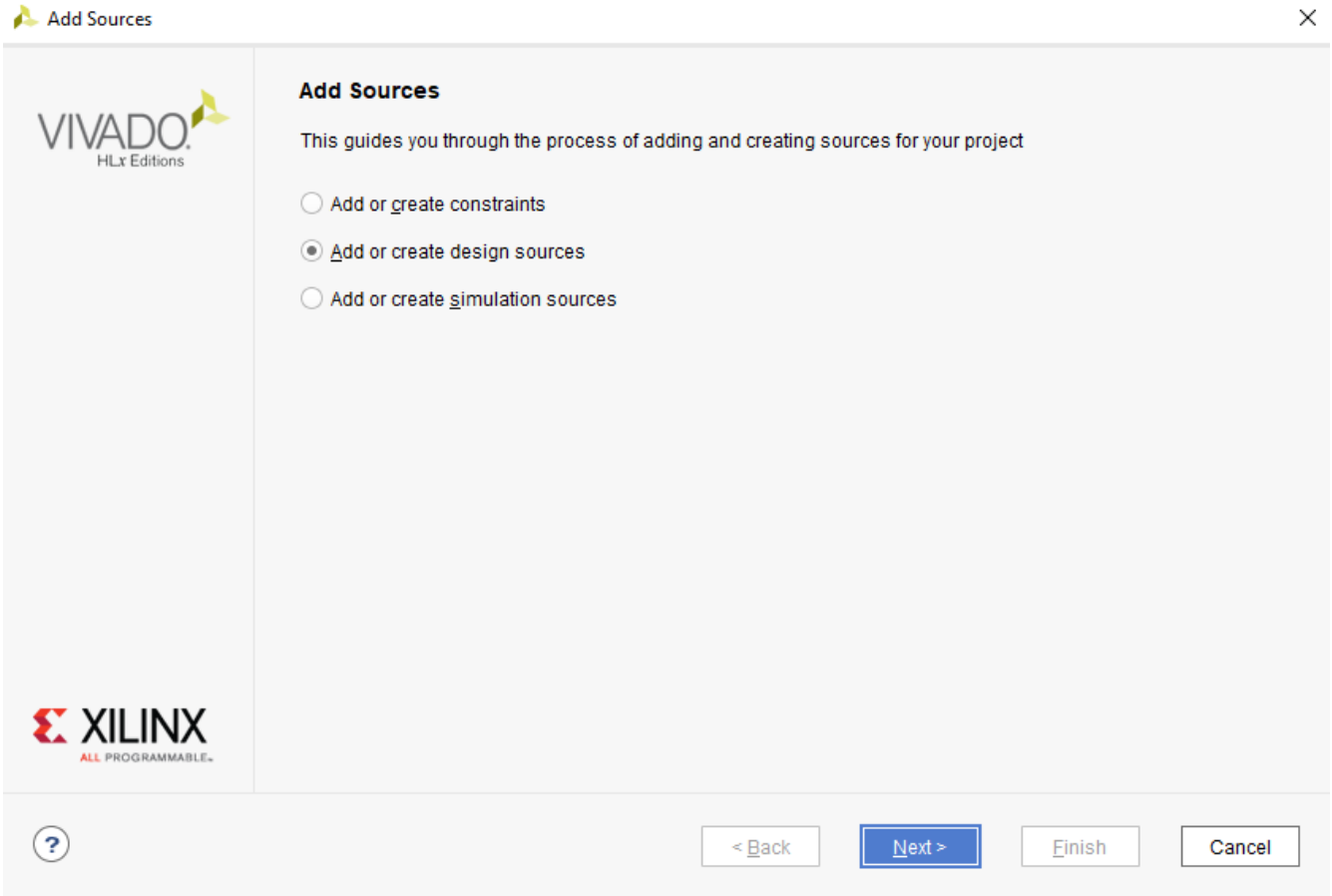


Figure 15. Add or create design sources using Add Source Dialog

Now, you are landed on the page Add or Create Design Sources in Add Sources dialog. You can click on Create File, put project1_demo as file name and click ok in the pop-up dialog. The newly created file will appear in the list, as shown in Figure 16. Click Finish in Add Sources dialog to finish adding the file.

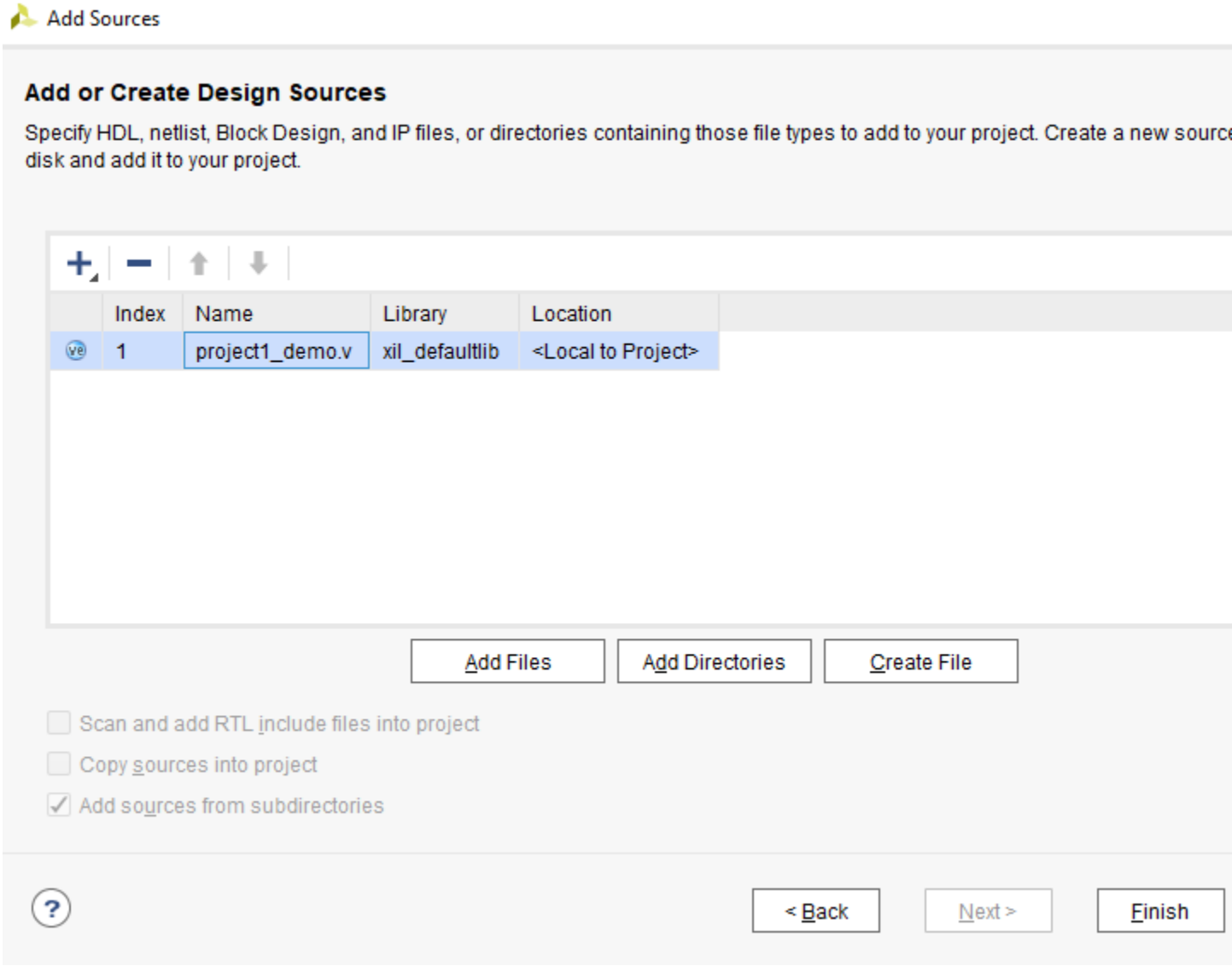


Figure 16. Create Design Source File

Skip the Define Module dialog by clicking OK to continue. You wil see project1_demo listed underneath Design Sources folder in Sources panel. Double click project1_demo to open up the file and replace the content with the codes below.

Download and Add Source

Instead of creating a source file and typing all the codes, you may also download the project1_demo.vfile using button PROJECT1_DEMO.V and add it to your project using the Add Sources button in Add Sources dialog.

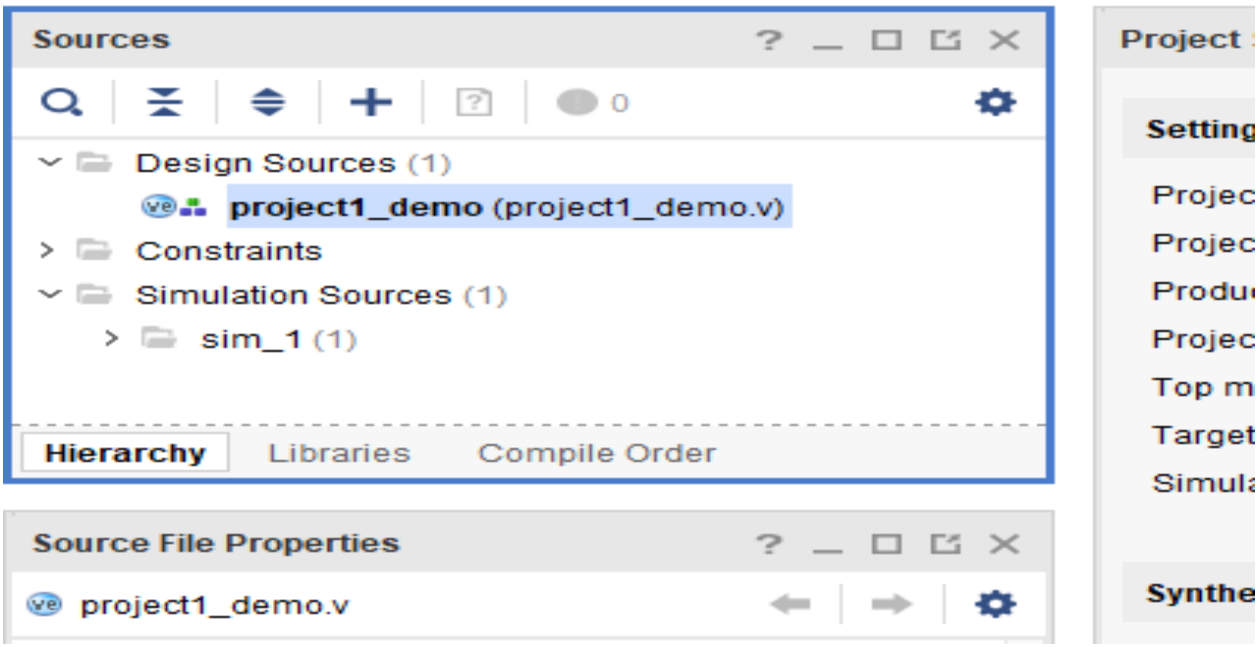


Figure 17. project1_demo appears in design sources

```
Copy
`timescale 1ns / 1ps

    module project1_demo(
        input clk,
        output [15:0] led,
        output [7:0] seg,
        output [3:0] an
    );

    reg [23:0] counter = 24'd0;
    reg divclk = 1'b0;
    reg [3:0] round_counter = 4'd0;
    reg [15:0] led_reg;
    reg [7:0] seg_reg;

    /* Clock Divider: 100MHz -> 10Hz (100ms) */
    always @(posedge clk)
```

```
    begin
        if (counter == 24'd4999999) begin
            divclk <= ~divclk;
            counter <= 24'd0;
        end
        else begin
            divclk <= divclk;
            counter <= counter + 1'd1;
        end
    end

    always @(posedge divclk)
    begin
        round_counter <= round_counter + 1'd1;
    end

    always @(posedge divclk)
    begin
        if (round_counter == 4'd0) begin
            led_reg <= 16'hFFFE;
            seg_reg <= 8'hFE;
        end
        else begin
            seg_reg <= {seg_reg[6:0], seg_reg[7]};
            led_reg <= {led_reg[14:0], led_reg[15]};
        end
    end

    assign led = led_reg;
    assign an = 4'h0;
    assign seg = seg_reg;

    endmodule
```

Create Design Constraints

Design sources, such as Verilog HDL files, only describes how the digital system work internally. You also need to provide proper constraints to your design to make it work with the physical board you have.

To create new design constraint file, in the Sources panel, expand Constraints, select constrs_1, right click on it and select Add Sources...

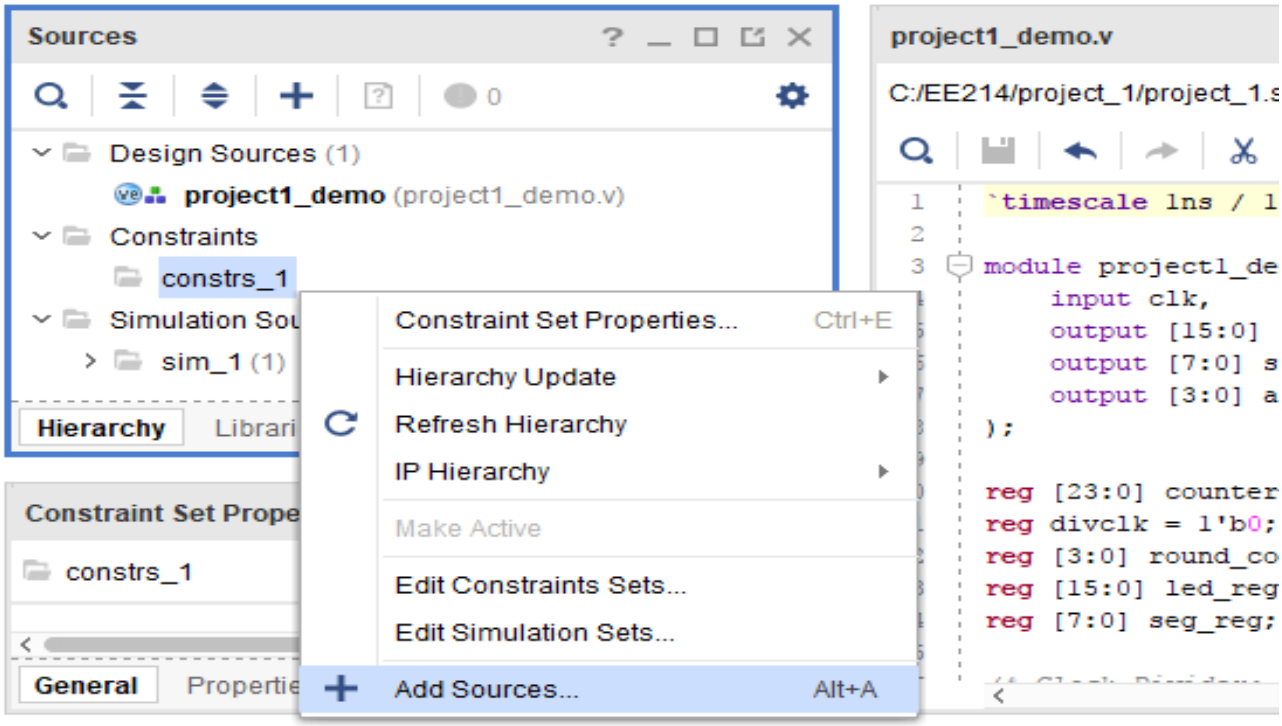


Figure 18. Add Source to Design Constraints

An Add Sources dialog will appear as shown in Figure 19. Select Add or create constraints and click Next.

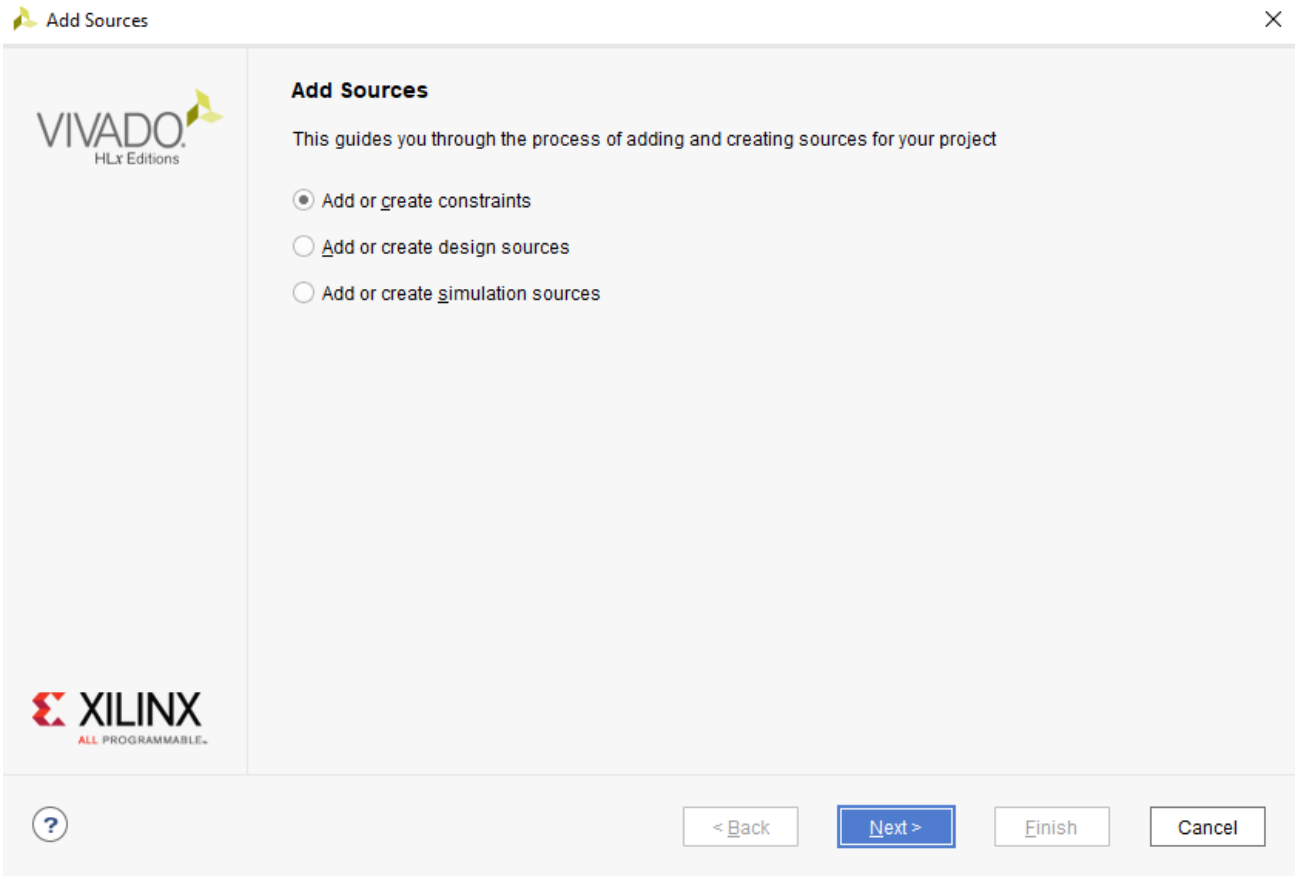


Figure 19. Add or create design constraints using Add Source Dialog

Now, you are landed on the page Add or Create Constraints in Add Sources dialog. You can click on Create File, put project1 as file name and click ok in the pop-up dialog. The newly created file will appear in the list, as shown in Figure 20. Click Finish in Add Sources dialog to finish adding the file.

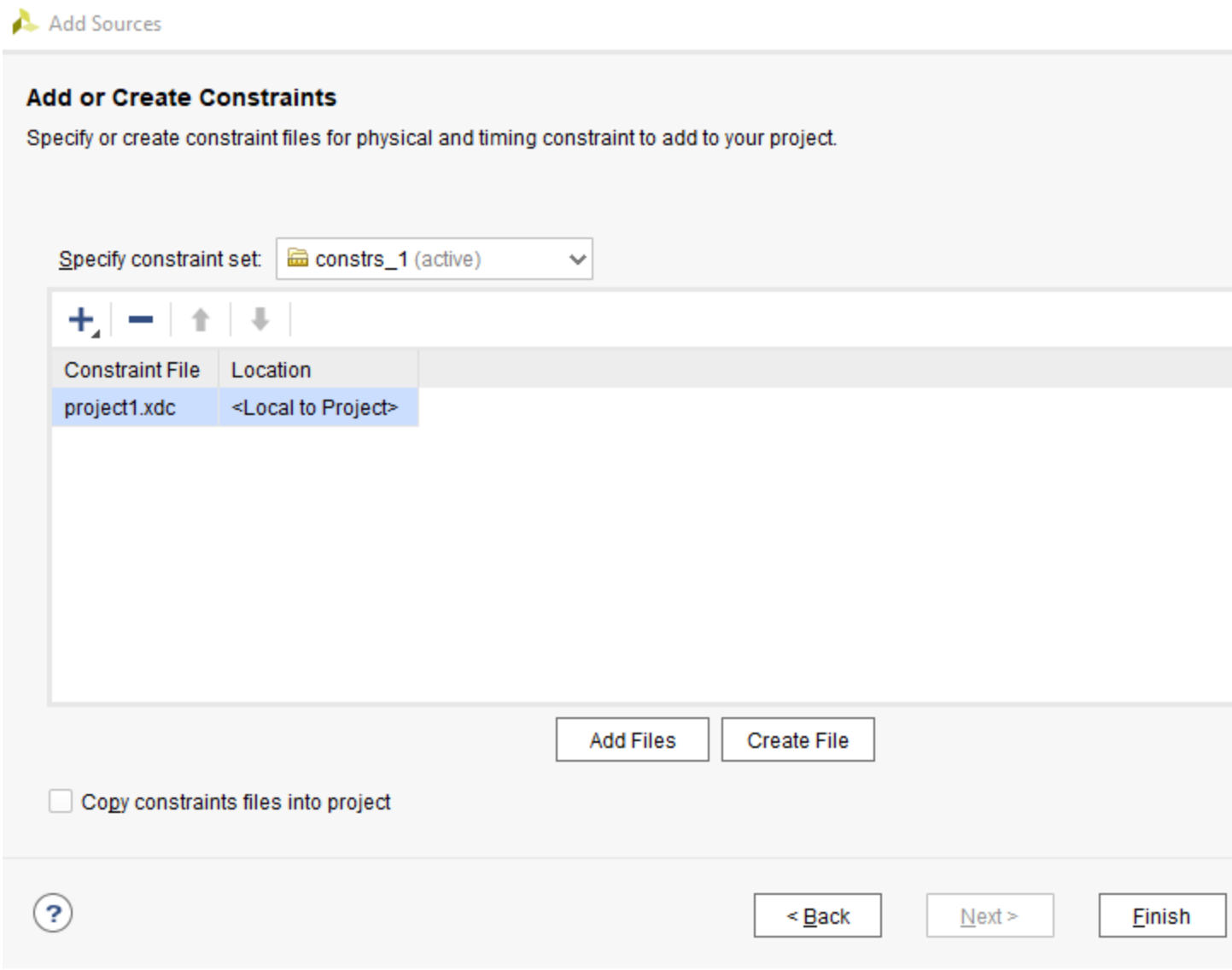


Figure 20. Create constraint file

You wil see project1.xdc listed underneath Constraints/constrs_1 folder in Sources panel. Double click project1.xdc to open up the file and replace the content with the codes below.

Download and Add Constraints

Instead of creating an empty constraint file and typing all the codes, you may also download the project1.xdc file using button PROJECT1.XDC and add it to your project using the Add Files button in Add Sources dialog as shown in Figure 21.

Figure 21.

Double click to edit project1.xdc

```
Copy
# Clock signal
    #Bank = 34, Sch name = CLK100MHZ
    set_property PACKAGE_PIN W5 [get_ports clk]
    set_property IOSTANDARD LVCMOS33 [get_ports clk]
    create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]

    # LEDs
    set_property PACKAGE_PIN U16 [get_ports {led[0]}]
    set_property PACKAGE_PIN E19 [get_ports {led[1]}]
    set_property PACKAGE_PIN U19 [get_ports {led[2]}]
    set_property PACKAGE_PIN V19 [get_ports {led[3]}]
    set_property PACKAGE_PIN W18 [get_ports {led[4]}]
    set_property PACKAGE_PIN U15 [get_ports {led[5]}]
    set_property PACKAGE_PIN U14 [get_ports {led[6]}]
    set_property PACKAGE_PIN V14 [get_ports {led[7]}]
```

```
    set_property PACKAGE_PIN V13 [get_ports {led[8]}]
    set_property PACKAGE_PIN V3 [get_ports {led[9]}]
```

```
    set_property PACKAGE_PIN W3 [get_ports {led[10]}]
    set_property PACKAGE_PIN U3 [get_ports {led[11]}]
    set_property PACKAGE_PIN P3 [get_ports {led[12]}]
    set_property PACKAGE_PIN N3 [get_ports {led[13]}]
    set_property PACKAGE_PIN P1 [get_ports {led[14]}]
    set_property PACKAGE_PIN L1 [get_ports {led[15]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {led[*]}]

    #7 segment display
    #Bank = 34, Sch name = CA
    set_property PACKAGE_PIN W7 [get_ports {seg[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[0]}]
    #Bank = 34, Sch name = CB
    set_property PACKAGE_PIN W6 [get_ports {seg[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[1]}]
    #Bank = 34, Sch name = CC
    set_property PACKAGE_PIN U8 [get_ports {seg[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[2]}]
    #Bank = 34, Sch name = CD
    set_property PACKAGE_PIN V8 [get_ports {seg[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[3]}]
    #Bank = 34, Sch name = CE
    set_property PACKAGE_PIN U5 [get_ports {seg[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[4]}]
    #Bank = 34, Sch name = CF
    set_property PACKAGE_PIN V5 [get_ports {seg[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[5]}]
    #Bank = 34, Sch name = CG
    set_property PACKAGE_PIN U7 [get_ports {seg[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[6]}]
    #Bank = 34, Sch name = DP
    set_property PACKAGE_PIN V7 [get_ports {seg[7]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[7]}]
```

```
    #Bank = 34, Sch name = AN0
    set_property PACKAGE_PIN U2 [get_ports {an[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[0]}]
    #Bank = 34, Sch name = AN1
    set_property PACKAGE_PIN U4 [get_ports {an[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[1]}]
    #Bank = 34, Sch name = AN2
    set_property PACKAGE_PIN V4 [get_ports {an[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[2]}]
    #Bank = 34, Sch name = AN3
    set_property PACKAGE_PIN W4 [get_ports {an[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[3]}]
```

Step 4: Synthesize and Implementation

Synthesize

In *Synthesize* process, Verilog HDL codes are compiled and translated into netlist by a program called *a synthesis tool*. You can start Synthesize process by clicking on Run Synthesis button in the Flow Navigatorpanel, as highlighted in red in Figure 22.

When synthesis is running, select log panel which is located at the bottom of Project Manager. As the synthesis tool is running, the program will print out a lot of information about what the program is doing at the moment. All the errors occurring during the synthesis process are all described in the log.

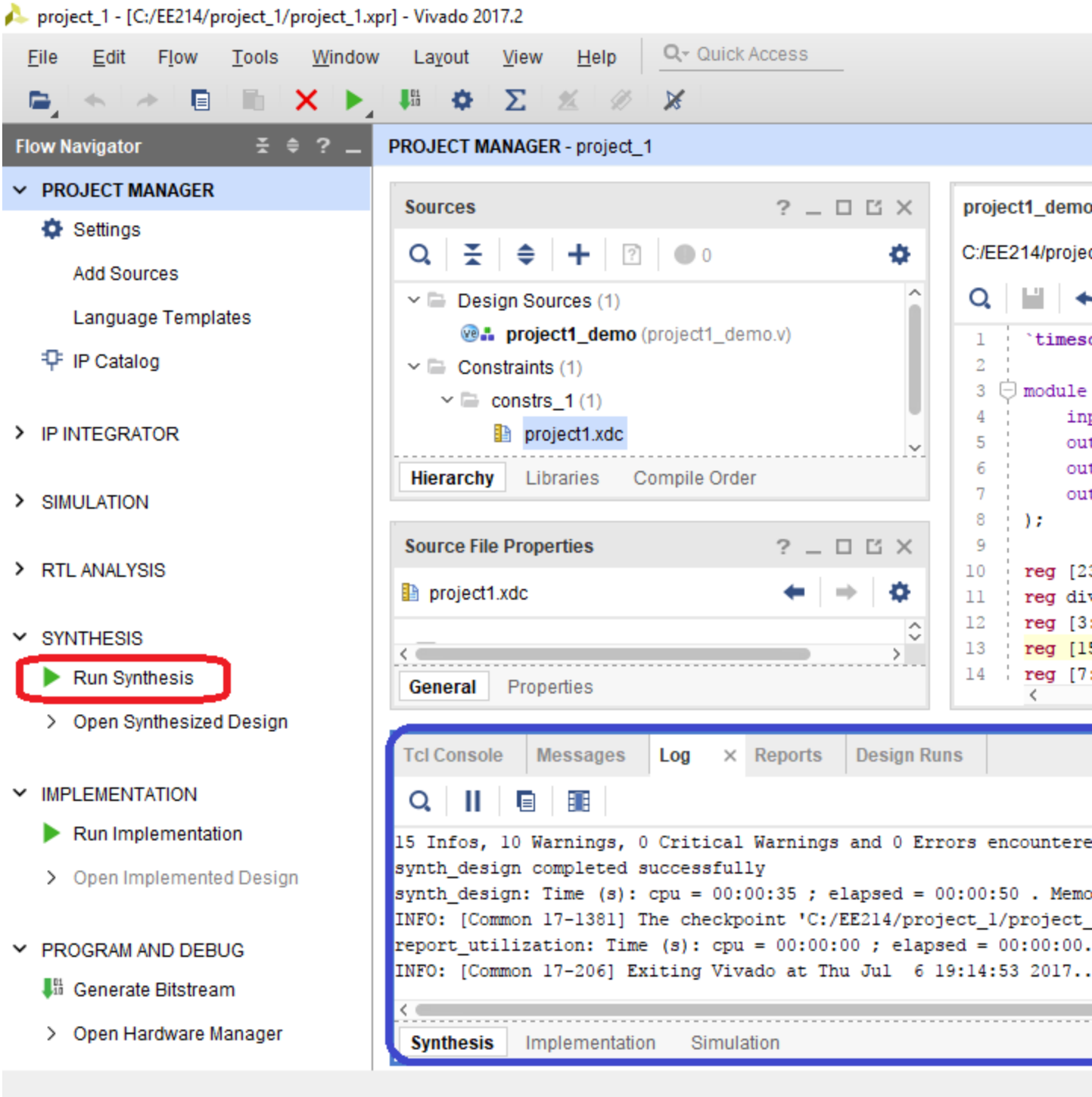


Figure 22. Start Synthesis process and monitor the systhesis log

Implementation

After the design is synthesized, you need to run the *Implementation* process. The implementation process maps the synthesized design onto the FPGA chip targeted by the design. Click Run Implementation button in the Flow Navigator panel, as highlighted in red in Figure 23.

When implementation is running, select log panel which is located at the bottom of Project Manager. All the errors occurring during the implementation process are all described in the log.

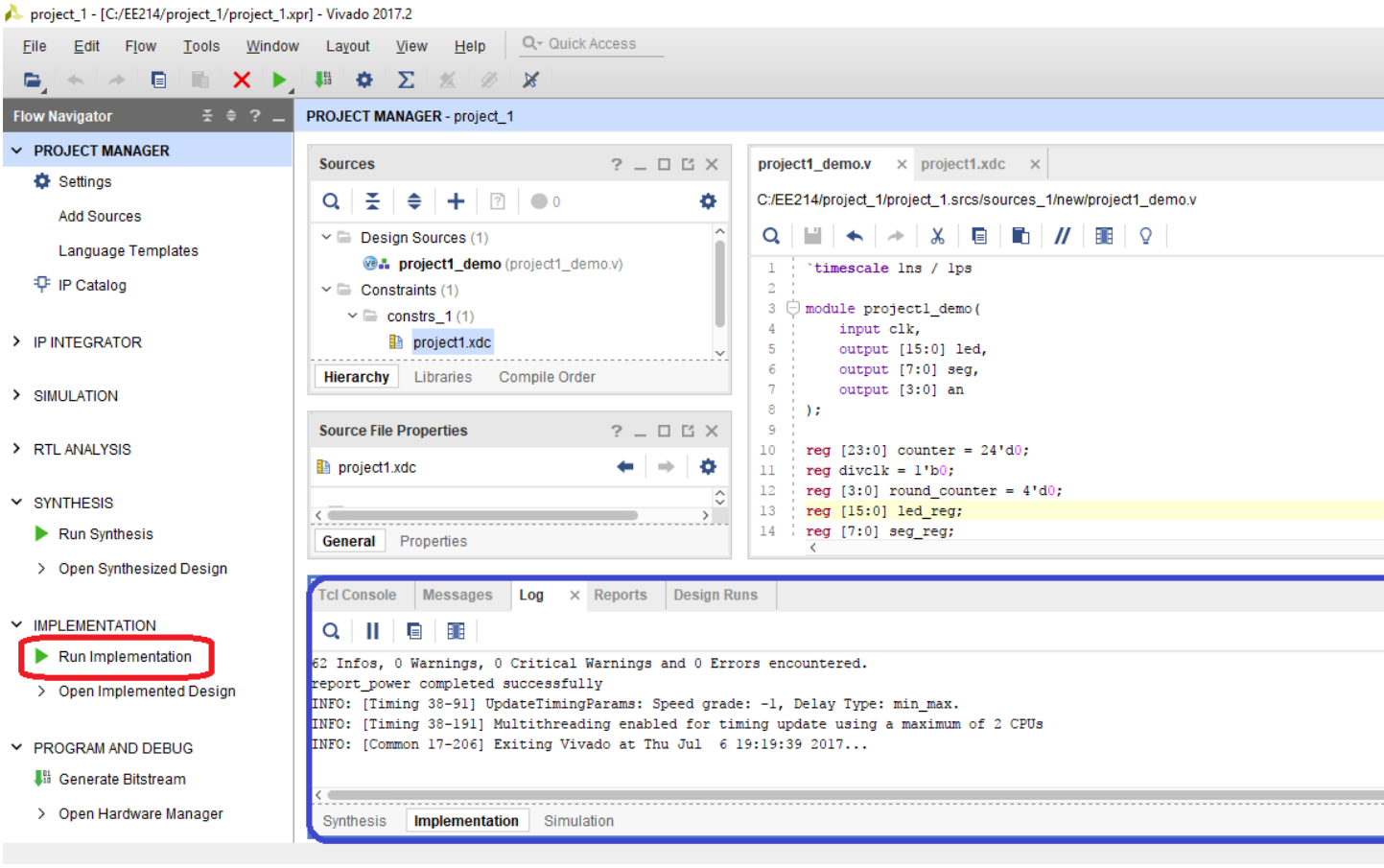


Figure 23. Start Implementation process and monitor the implementation log

Generate Bitstream

After the design is successfully implemented, you need to run Generate Bitstream located in the Flow Navigator panel, as highlighted in red in figure 24. The process translates the implemented design into a bitstream which can be downloaded onto your FPGA board.

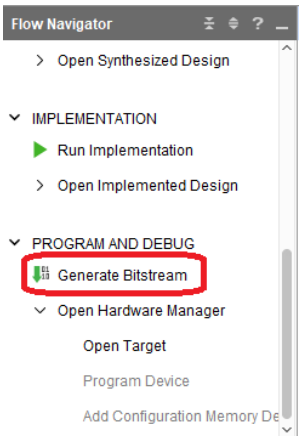


Figure 24. Generate Bitstream

Step 5: Download Bitstream

Open Hardware Manager

After the bitstream is successfully generated, you can program your FPGA board with the bitstream using Hardware Manager. Click Open Hardware Manager located at the bottom of Flow Navigator panel, as highlighted in red in Figure 25.

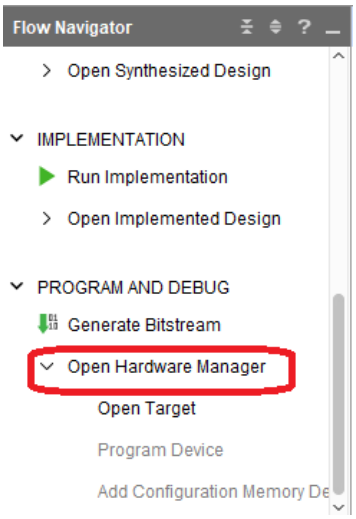


Figure 25. Open Hardware Manager

Connecting Your Board

Connect your FPGA board to your PC with a micro-USB cable, and then click on Open target link underneath Hardware Manager. Select Auto Connect as shown in Figure 26 to ask Vivado identify your FPGA board automatically.

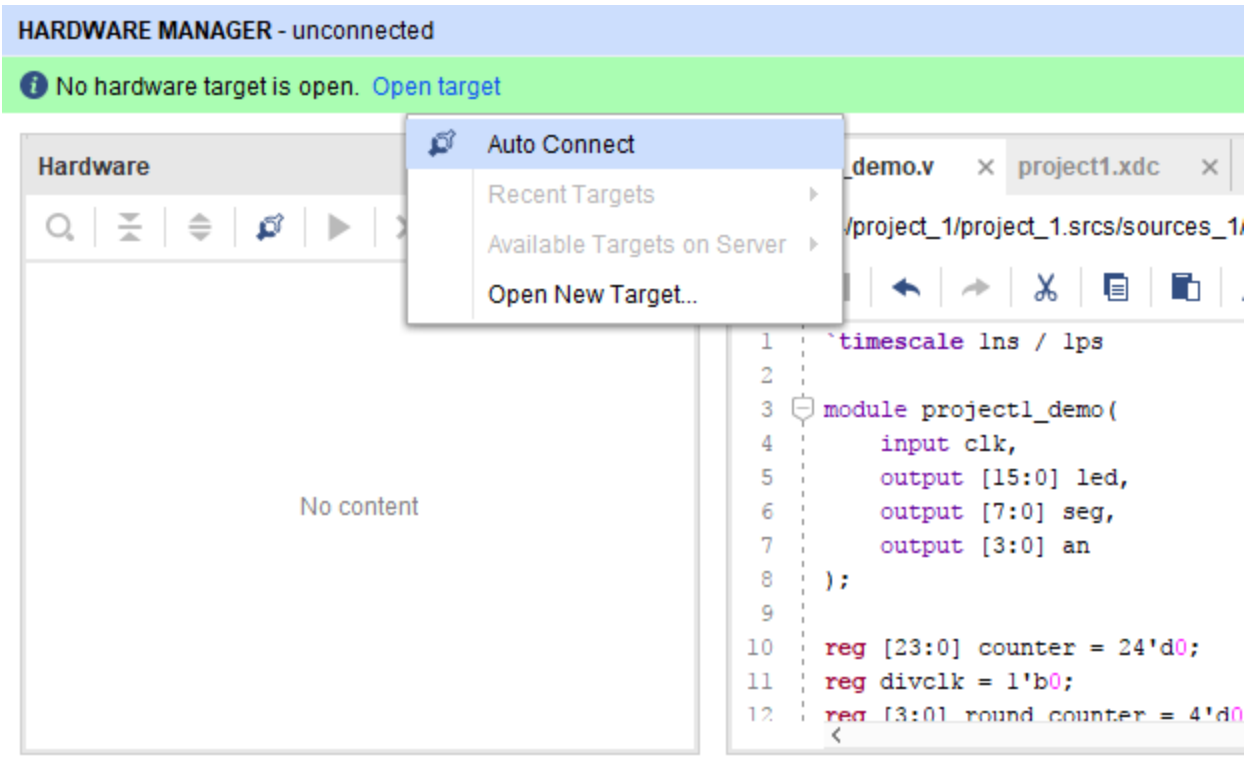


Figure 26. Open Hardware Manager

Verify Your Board is Identified

If Vivado detects your Basys3 board successfully, you will see the FPGA is detected under Hardware panel located at the top left corner of Hardware Manager. The part number of FPGA should be correctly identified as xc7a35t for Basys3 board.

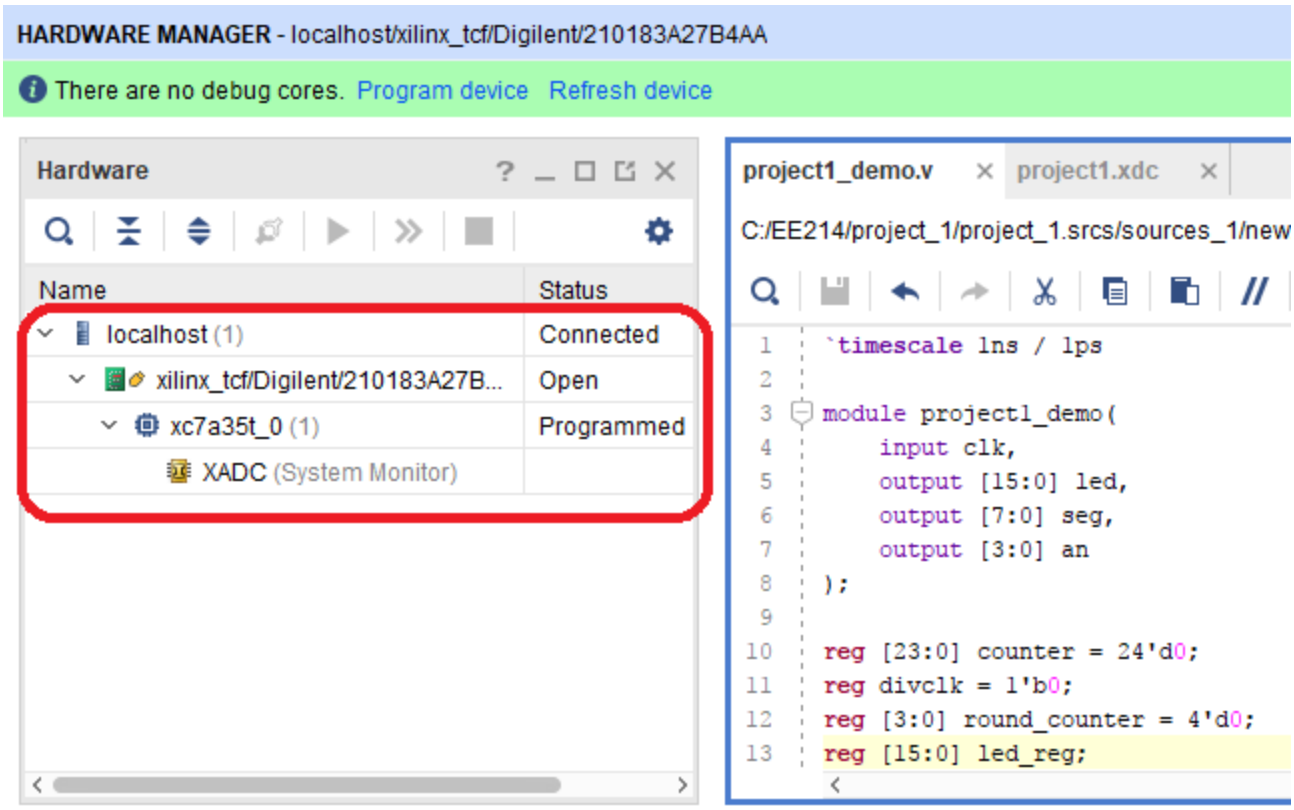


Figure 27. Vivado finds Basys3 board and connects to it successfully

Download Bitstream

Select the FPGA xc7a35t, right click and select Program Device.... A Program Device pop-up dialog window will show with the generated bitstream file selected in the text box corresponding to Bitstream file. Click on Program to download the bitstream to your FPGA.

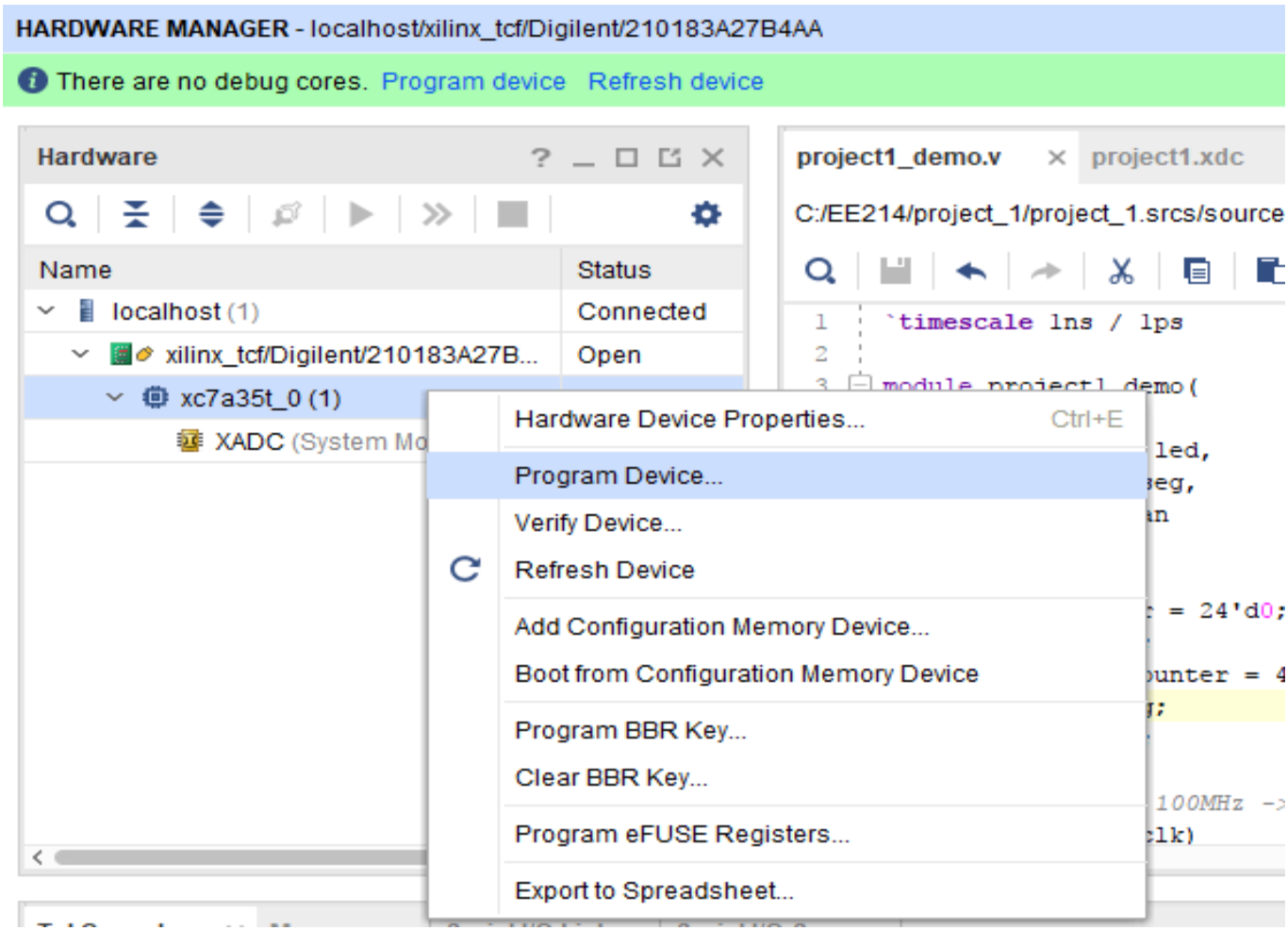


Figure 28. Program Device

Verify the Demo Project

You will see the demo operating and showing the green LEDs on except for one LED that is off moving across the LEDs. The seven-segment display will change through the different segments. When this is complete you have finished this project!