B. Appendix 2 – VIVADO Quick Start Tutorial

VIVADO Quick Start Tutorial, adapted for version 15.4

Starting the VIVADO Development Environment

Double click the desktop icon, or go to **Start→Programs→ Xilinx Design Tools → Vivado 15.4**

Accessing Help

To open Help:

- From the main menu launch the **Help/Documentation and Tutorials** from the Help menu. It contains information about creating and maintaining your complete design flow in VIVADO.

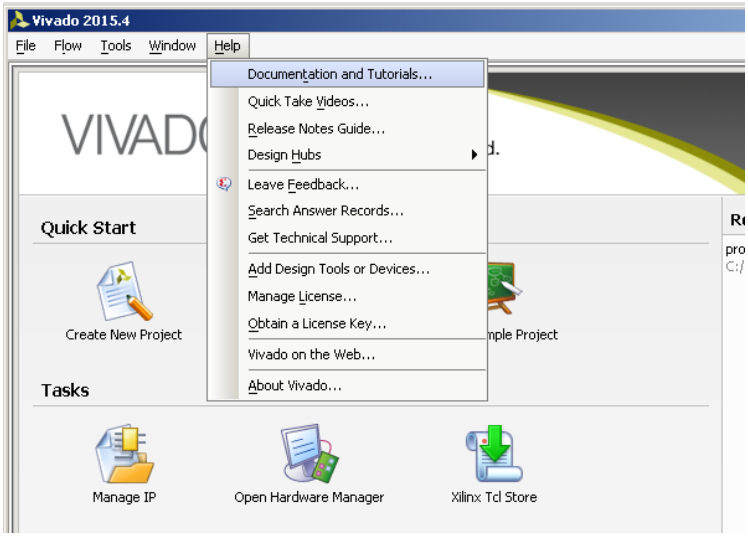


Figure B-1: VIVADO Help Topics

Create a New Project

Create a new VIVADO project, which will target the FPGA device on the Basys 3 development board, Artix 7 FPGA.
To create a new project:

1. Select **File → New Project ...** The New Project Wizard appears. Click **Next**.
2. Type **test_env** in the Name field.
3. Enter or browse to a location (directory path) for the new project (remember the laboratory rules). A test_env subdirectory is created automatically (*Create project subdirectory* must be checked). Click **Next**.
4. You have to choose the type of project. Select *RTL* Project. Cheek the *Do not specify sources…* you will create them later. Click **Next**.

5. Choose the properties of the target device:

   - Product Category: **All**
   - Family: **Artix-7**
   - Package: **cpg236**
   - Speed Grade: **-1**
   - Temperature grade: **C**
   - Choose the **xc7a35tcpg236-1** in the table below.
   - **Next**…

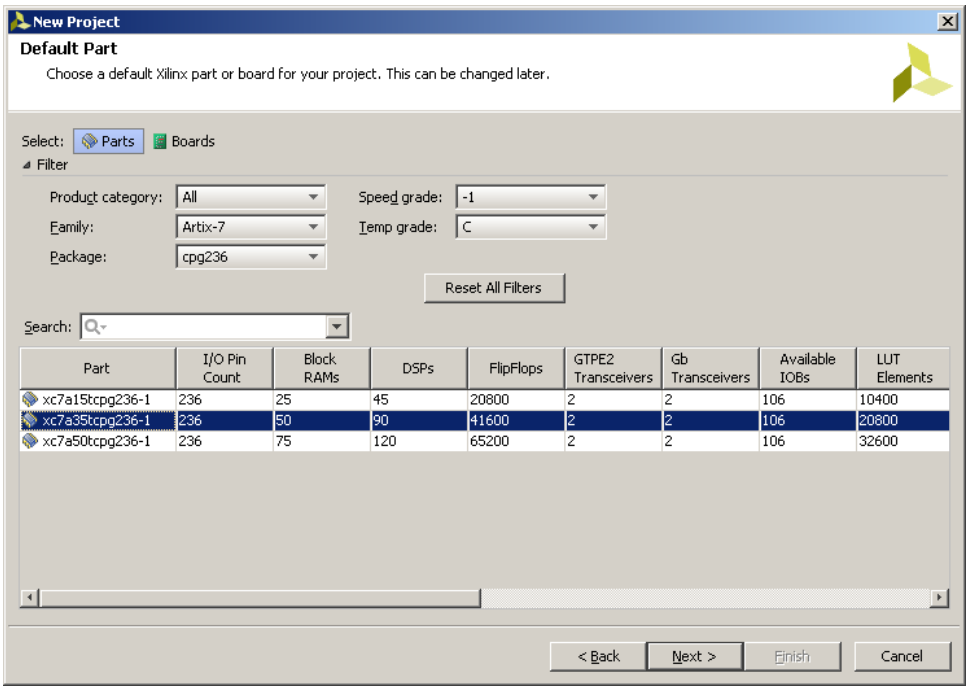


Figure B-2: Choosing the properties of the development board

6. Click Finish. The new project will open

Creating a VHDL Source

Create a VHDL source file for the project as follows:

1. Click the menu **File\Add Sources** or in the **Flow Navigator** pane (usually situated on the left) at **Project Manager / Add Sources**.
2. Select *Add or create design sources*, click **Next**.
3. Press *Create file*. On the next dialog select *File type*: VHDL, introduce the file name *test_env (! It is not mandatory to have the same file name as the project name)*, leave the *File location* unchanged. This name (*test_env*) will also be given to the entity created in this file. Click **OK**.

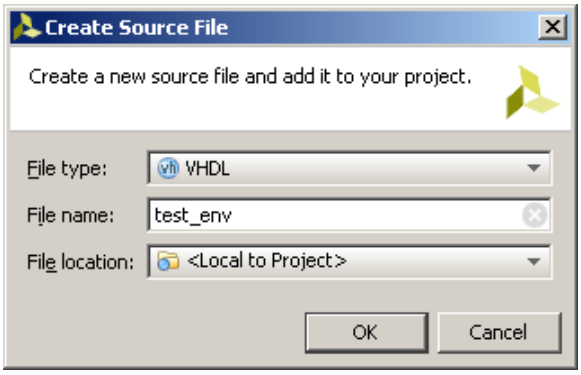


Figure B-3: Creating a VHDL source file

4. Press Finish. The new file / entity will be created and a new dialog will appear for defining the entity's ports: **Define Module**.
5. You now have to declare the ports of the top level (main) entity, completing the information for the ports as in the next figure. **Attention: These ports are particularly defined for the Basys 3 development board, being enough for the majority of the laboratory designs for this semester.** Press **OK**.

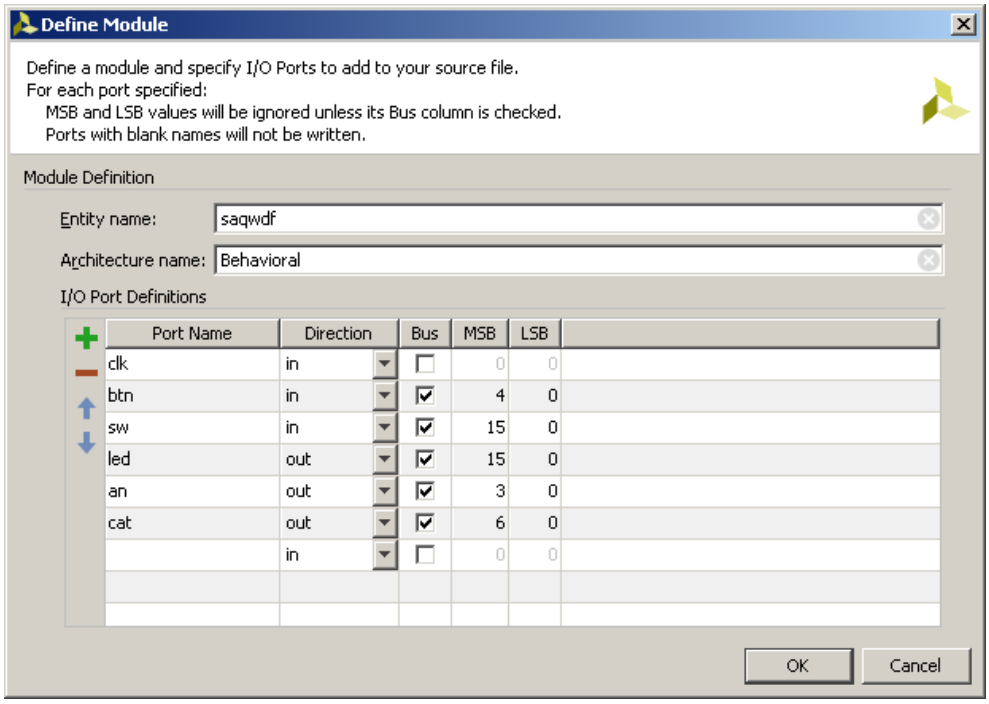


Figure B-4 Port definition through the VIVADO Interface

6. The new source file has been created.

Remark: You can skip the define module step. In this case you will have to declare (modify/correct) the ports definitions in the entity from the source file.

The source file containing the entity *test_env* and its architecture is displayed in the VIVADO environment, and in the **Hierarchy (Sources Pane)** appears as **Top Module**

for the current design. If the editing window of the new source file does not appear, double click the test_env in the Sources Pane.

Remember, in projects containing multiple source files, if one accidentally changes the top module entity, you can reset it as a top module by right click on a source in **Hierarchy**, and select **Set as Top**.

Attention: Access the **Project Summary**, at the **Synthesis** and **Implementation** you should see in the *Part* field the name of the chosen device. For Basys 3 it must be **xc7a35tcpg236-1.** If it does not match it means that you have skipped step configuration properties (step 5) in the **Creating a new project** part. In the main menu select **Tools/Project Settings**, in the **General** section, Project Device field, press "…" and re-introduce the correct properties of the targeted device.

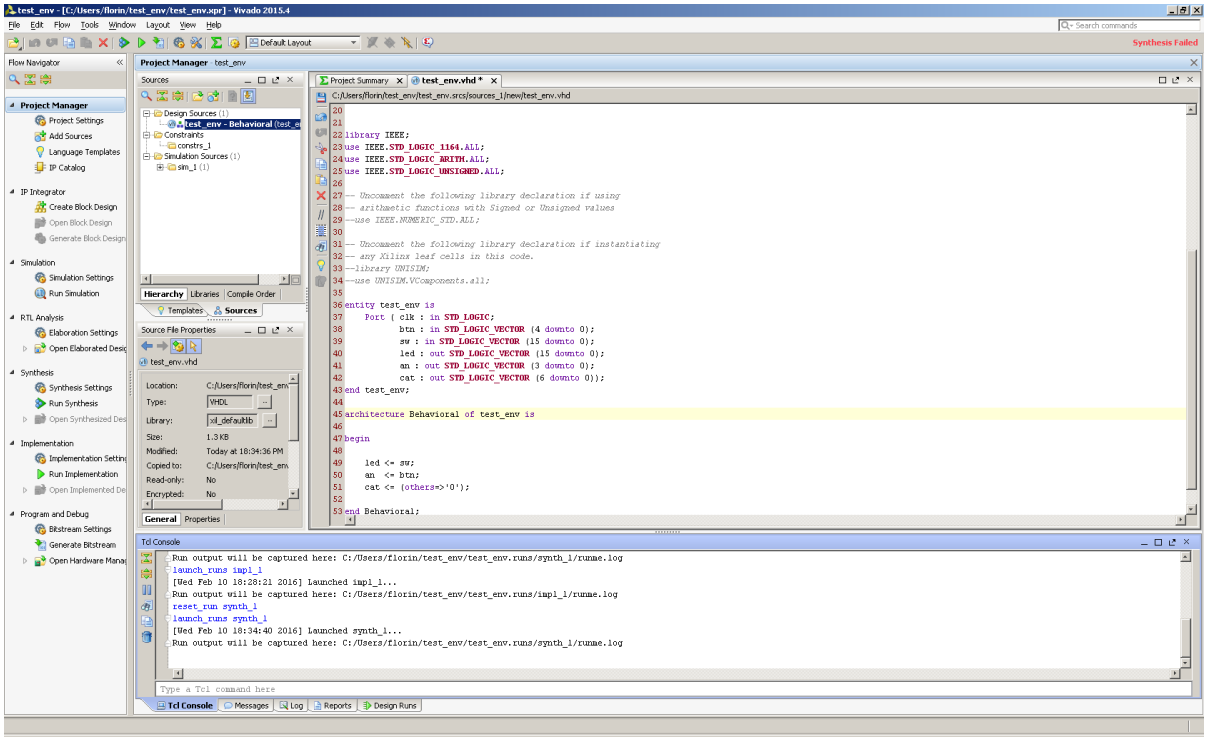


Figure B-5: The new VIVADO Project

Make shore that the following libraries are included in the source file header:

use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

If they are missing from any VHDL file, include them!

Using Language Templates (VHDL) – optional (*You will probably have to use this in the future…*)

Language Templates includes VHDL synthesizable examples that you can use in your designs. The "Light Bulb" takes you directly to Language Templates tab" or you can do the following"

1. Place the cursor under the begin statement of your architecture.
2. Open Language Templates by selecting the menu **Windows → Language Templates…**
3. Navigate in the hierarchy "+", to the coding examples: **VHDL → Synthesis Constructs → Coding Examples → …**
4. Select the desired component in the hierarchy, and copy it in the source file.
5. Close the **Language Templates**.
6. Change the signal names so that they will match the signals in your entity.

Editing the VHDL Source Code

1. Add component and/or signal declarations between the architecture and the begin statements.
2. Add the rest of the code (component instantiation, behavioral description, etc.) after the begin statement and before the end statement.
3. For the first example add the following statements after begin.

```
led <= sw;
an  <= btn(3 downto 0);
cat <= (others=>'0');
```

4. Save the file by selecting **File → Save or Ctrl + S**.
5. Select the top-level entity in the Hierarchy pane: *test_env*.
6. Synthesize the project, in Flow Navigator (left) **Run Synthesis**.
7. Correct the errors, if they appear in the bottom part of the VIVADO environment section **Console / Messages**. Start from the top with the first error.
8. Visualize the resulting circuit in a schematic form (relevant for the next projects, that will be more complex): in the **Flow Navigator** pane click the **RTL Analysis – Elaborated Design → Schematic.** The data-path of the top-level module will appear, double click on the components to view their internal structure. **You should recognize at a least a part of the declared components!** This is a first method to verify that your code is correct and implements the desired circuits.

You have now created the VHDL source for the "test_env" project with no errors.

Assigning Pin Location Constraints

Specify the pin locations for the ports of the design so that they are connected correctly on the DDB.

1. You can find the user constraints file for the Basys 3 Board here. Download the file and add it to your design. Open the constraints file and see the syntax for every port.
2. Click the menu **File\Add Sources**. Or in the **Flow Navigator** pane go to **Project Manager / Add Sources.**
3. Select *Add or create constraints*, click **Next**.
4. Press *Add File*. On the next dialog select the downloaded constraints file. Press **Ok**.

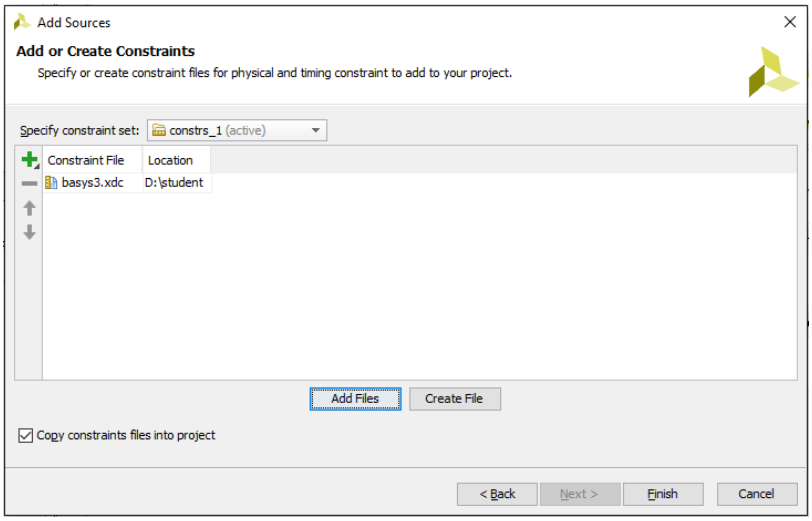


Figure B-6: Adding the Constraints File to the Project

5. Check the **Copy constraint file into project** checkbox and press Finish. The constraints file is added to the project and visible in the **Sources** Pane under **Constraints**.
6. Right Click on the constraints file and press *Set as Target Constraint File in order to associate it with the targeted device on the development board.*
7. Double click the constraints file to edit / view its content. A usual constraint (between a port from the top-level entity and a pin on the development board) is defined by two lines. Example for the center button:

```
set_property PACKAGE_PIN U18 [get_ports {btn[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {btn[0]}]
```

where U18 represents the pin name from the targeted device associated with the center button; and btn[0] is the port from the top-level entity which we want to be associated to the center button.

For the future, you can add new ports to the constraints file by adding 2 lines in the constraints file. The pin name can be found in the reference manual of the Basys 3 development board or written on the board:

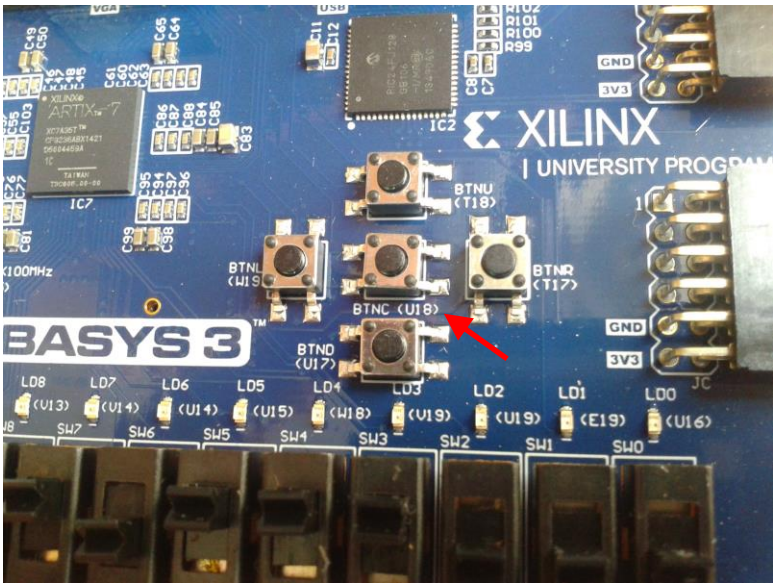


Figure B-7:The pin location names written on the development board. Red arrow: for the center button the location is U18.

Implement Design and Verify Constraints

1. Implement design: in the menu **Flow / Run Implementation** or in the **Flow Navigator** pane **Implementation / Run Implementation**.
2. If the implementation completes with no errors you can proceed to the next step, otherwise correct all the errors and re-run the implementation process.

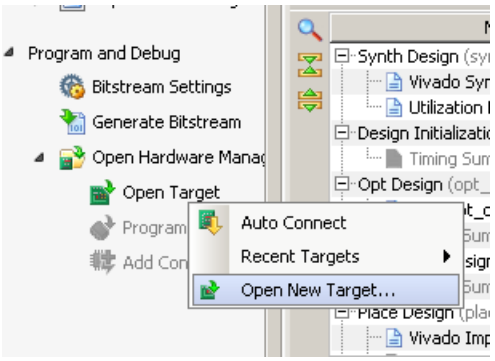
Generate Programming File

In the **Flow Navigator** pane at **Program and debug / Generate Bitstream** or in the **Flow** menu press **Generate Bitstream**. The bit file for the development board configuration is created.

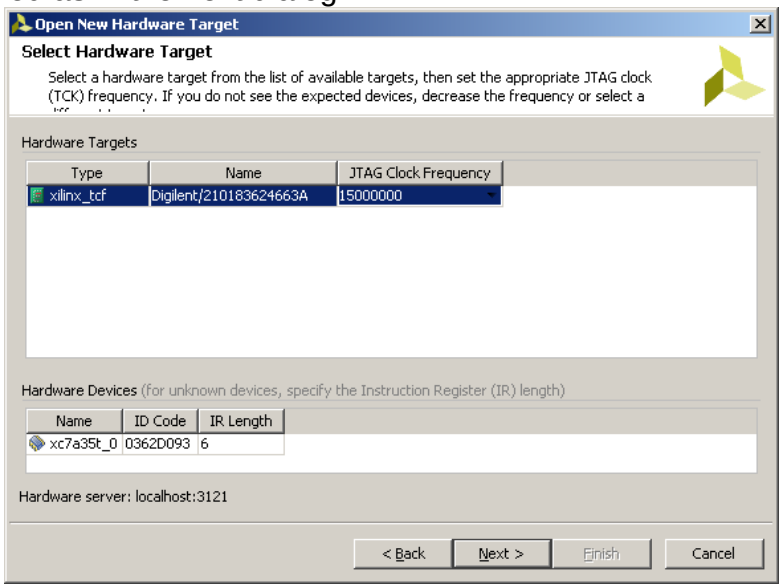
If there are no errors at this time the file “test_env.bit” should be in the project folder. Otherwise you have to correct all the errors.

Download Design to the Basys 3 development board

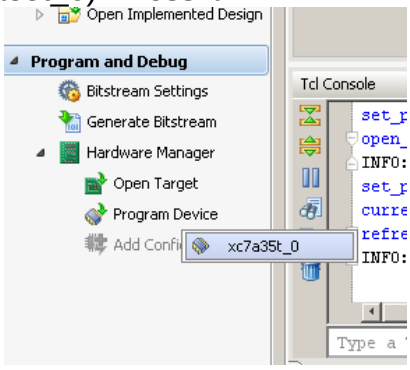
1. Connect the Basys 3 development board to the USB port.
2. In the **Flow Navigator** pane Program and Debug section click on Open Target and in the new menu click Open New Target.



3. **Next**. *Local server* in the Connect to field. **Next**. The device should be recognized as in the next dialog.



4. **Next. Finish.**
5. If the board has been recognized, then **Program Device (Flow Manager / Program and Debug)** is active. Press it ...
6. A pop-up menu will appear, showing the connected devices – only one in our case (Basys 3 – xc7a35t_0). Press it.



7. Press **Program** in the next dialog.
8. At this moment the design is loaded on the development board. Test its functionality (buttons, switches, LEDs).
9. For re-programming, if the Program Device is not active, click on **Open Target / Auto Connect**

Possible problems when connecting the board and solutions:

Problem: The Basys 3 board is not recognized:

a) Try a different USB port (front or rear of the computer). If a driver install process initiates, call your TA. You will need administrative privileges.
b) Try a new programming cable
c) Try a new board (report this to your TA)
d) Restart VIVADO / the workstation
e) Change the workstation