Steps To Create a Project in Vivado

1.

Create a new project from the UI specifying a location Note: Do not use spaces in file names or folders but use '_' (underscore) instead.

2.

Set RTL project-flagging the check box "*Do not specify sources at this time*".

3.

Select the "*Boards*" tab and scroll to the **Xilinx Pynq Z2** board in the list.

4.

Create the board design (graphical components)

> **Note: Never use the same file name for all the different file types and components else Vivado will not process the design correctly (this is a limitation of the IDE)**

5.

First add to the diagram (using the '+' button) the *Zynq 7 processing. system* component.

6.

Execute the *Designer assistance* link green bar "*Run block automation*".

> **Note: this process assigns the right settings to the DDR speeds and should not be omitted.**

7.

Select "*Sources*" and "*Add sources*" to add your custom VHDL file script.

On the popup window select "*Add design sources*", then "*Create file*". Add the file name of the custom component, then press the "*Finish*" button and confirm the next popup without

modifications A file with the default content is created but don’t care as its content will be replace by our custom VHDL definition.

8.

Open the file for editing from the design sources list and fully replace its content. with the custom component definition.

9.

After updating (project update is automatic on every change) the custom component is available for drag and drop on the block diagram.

Note: it is possible to repeat the same procedure if other custom components are needed in the design.

10.

Customise the design and set the connected pins to be external, as well as renaming the meaningful signals and pins to mnemonic names according to-the project (not mandatory but strongly suggested).

11.

After the design has been completed with all the connections and settings, validate the design with the top button bar and correct it until the validation passes with no errors.

12.

From the “*Sources*” tab right click on the *Design* icon on top of the tree and launch the “*Create HDL wrapper*” selecting the radio check “*Let Vivado…*” When the HDL wrapper finishes check the warning list. and confirm (ignore them, if any), as we expect that are not critical warnings related to the project.

Note: After-the MDL wrapper generation under the *Sources* tab the project tree has charged with the design wrapper on top.

13.

Now we can launch the *Synthesis* process with the command "*Run Synthesis*"from the flow navigator on the left bar of the program window.

> **Note: this process may be long so be patient.**

14.

If the synthesis ends with no errors now we should add some constrains (connecting the board pins to the external signals of the design)

> **Note: To see the exact name of the pins on the board we want to connect, see the board hardware documentation**

After defining the constraints the program asks for the constraints file name: write it in the text box; after saving the file appears in the "*Sources*" tab under the "*Constraints*" tree. The file can also be edited later, if needed, to add or modify the constraints definitions.

> **Note: Remember that after every change to the initial design architecture the last synthesis process become out of date and should be relaunched.**

15.

At this point we should run the board implementation with the command "*Run implementation*" When the process ends, launch the "*Generate bitstream*" to create the bit stream file; it is a binary file with the same name of the file wrapper .bit. This is one of the files we should import in the Jupyter notebook to control the FPGA with Python.