

AXI VIP Tutorial 1

Introduction

The goal of this tutorial is to demonstrate how to use the Xilinx AXI Verification IP (VIP) to test and verify your custom AXI lite peripherals using the Xilinx Vivado Simulator. We will show you how to connect the VIP to your design and use it to write to and read from registers of the AXI GPIO IP.

Step 1: Create a new project and block diagram

1. Create a new project for the **Nexys 4** board.
2. Create a new block design and name it **design_1**.
3. Add and wire the **AXI Verification IP**, **AXI GPIO**, and **Constant** IPs to the diagram as shown in Figure ?? below. Make sure that the port names match the figure.
4. Configure the **basic settings** tab of the **AXI VIP IP** as shown in the following figure. Leave the advance settings at their default values. This will set the VIP into AXI lite mode.

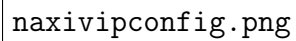
The image shows a large, empty rectangular box representing the AXI VIP configuration window. The text 'naxivipconfig.png' is positioned at the bottom left of the box.

Figure 2: AXI VIP configuration window.

5. Configure the **AXI GPIO IP** as shown in the following figure. We will be using the VIP to set the LEDs and read from the push buttons.

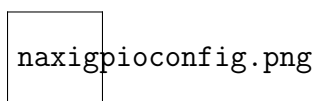
The image shows a small, empty rectangular box representing the AXI GPIO configuration window. The text 'naxigpioconfig.png' is positioned at the bottom left of the box.

Figure 3: AXI GPIO configuration window.

6. Configure the **Constant** IP as shown in the following figure. The constant value of 5 will simulate pushing buttons 0 and 2.

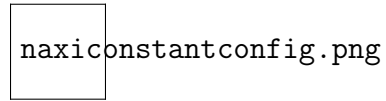
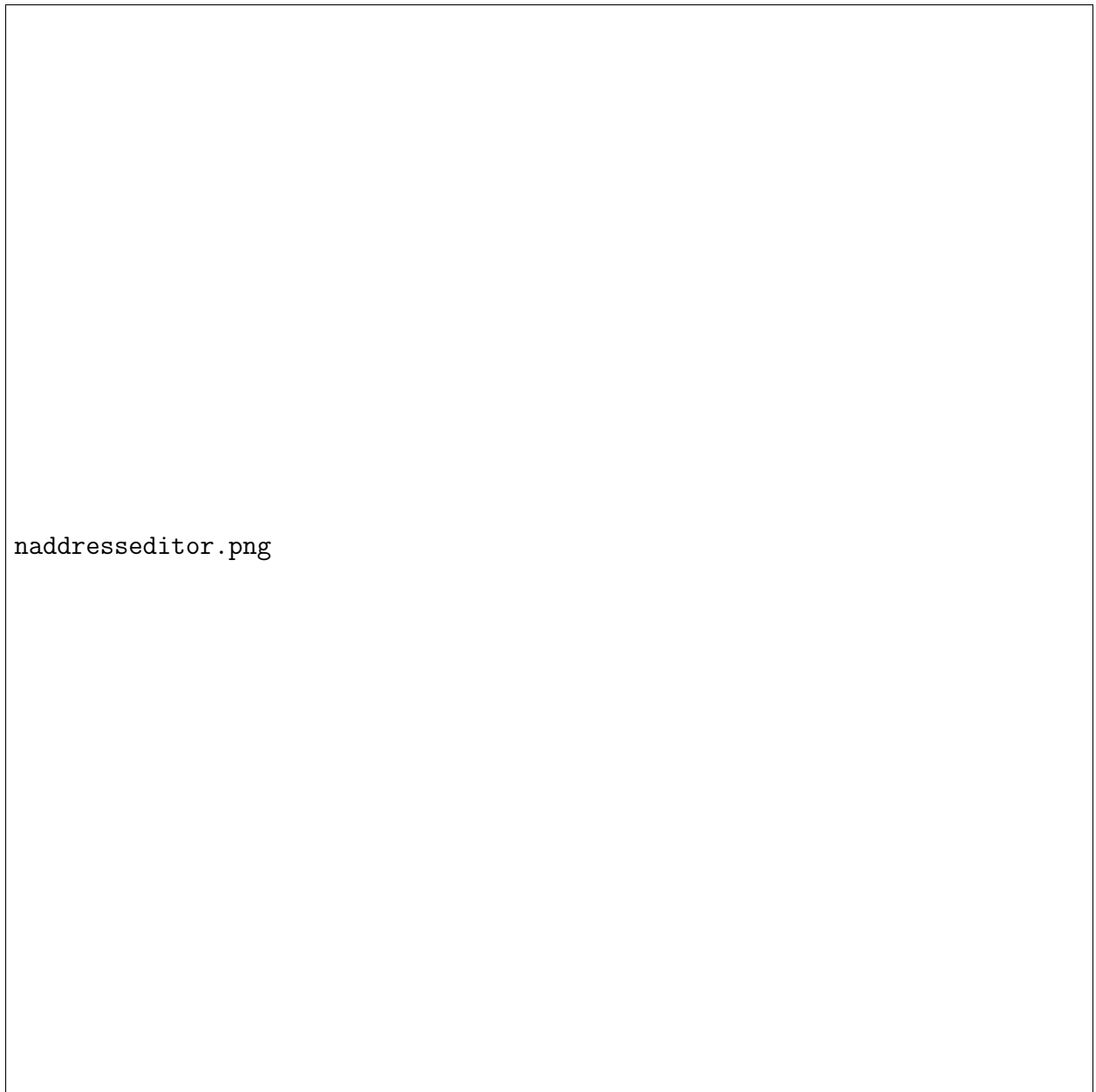


Figure 4: Constant configuration window.

7. Go the **address editor** tab and set the offset address to **0x40000000**.



naddresseditor.png

Figure 5: Address Editor

8. Validate the block design, create the HDL wrapper, and generate the block design.

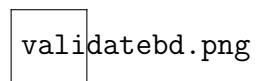
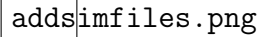


Figure 6: Generating the block design files.

Step 2: Creating the testbench

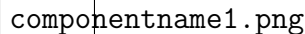
1. Import **tb.sv** as a **simulation source**.



addsimfiles.png

Figure 7: Importing testbench files.

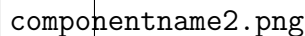
2. Open the tb.sv file in Vivado text editor. Replace the **<vip_component_name>** in line 4 and 15 with the vip component name. Refer to Figure ?? to determine what the component name is for your design.



componentname1.png

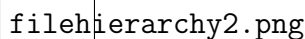
Figure 8: Replace lines 4 and 15 with the appropriate component name for your VIP.

Note: Syntax checker errors can be ignored, refer to simulation log for errors instead.



componentname2.png

Figure 9: Example of what lines 4 and 15 should look like.



filehierarchy2.png

Figure 10: How to find the component and instance name.

3. Go to line 60, and replace **<bd_instance_name>** and **<vip_instance_name>** with the bd instance name and vip instance name. Refer to Figure ?? to determine what the instance names are for your design.

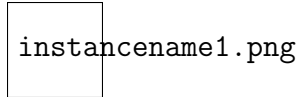


Figure 11: Replace line 60 with the appropriate instance names for your BD and VIP.

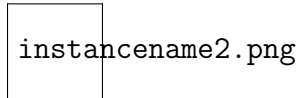


Figure 12: Example of what line 60 should look like.

4. Set tb instance under Simulation Sources as the top instance if it isn't already. The simulation source hierarchy should look like the figure below.

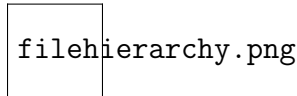


Figure 13: Simulation source hierarchy.

Step 3: Simulate the testbench

1. Before simulation, go to tb.sv and analyze line 66 to 75. This is where our test vectors are located. We will be writing to the output register of the GPIO bank that is connected to the LEDs. Then we'll set the push button GPIOs to be inputs and then wait for all writes to finish. We will then read the GPIO data register for the push buttons, and print the value of that register to the tcl console.

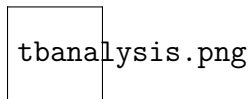


Figure 14: TB test vectors.

2. Start the simulation by clicking "Run Simulation" on the navigation pane. Add the AXI bus signals to the simulation window and click relaunch simulation. Make sure that you select the VIP instance in the scope window.

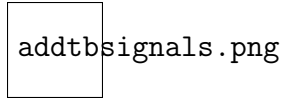
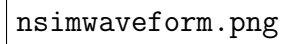


Figure 15: Adding AXI signals to the simulation.




Figure 16: Relaunching the simulation.

3. Analyze the tb results...

The image shows a rectangular frame intended for a simulation waveform plot. The text 'nsimwaveform.png' is positioned on the left side of the frame, indicating the source of the plot data.

nsimwaveform.png

Figure 17: Simulation waveform.



simruntclconsole.png

Figure 18: Tcl console print out of register.

Troubleshooting

1. If the Vivado simulator bugs out and starts throwing errors that are not related to your source files, you can try to fix it by resetting your behaviour simulation.

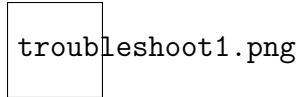


Figure 19: Resetting the simulation.

2. If you want to create more advanced AXI testbenches with the Vivado AXI VIP refer to their example design.

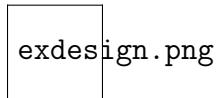
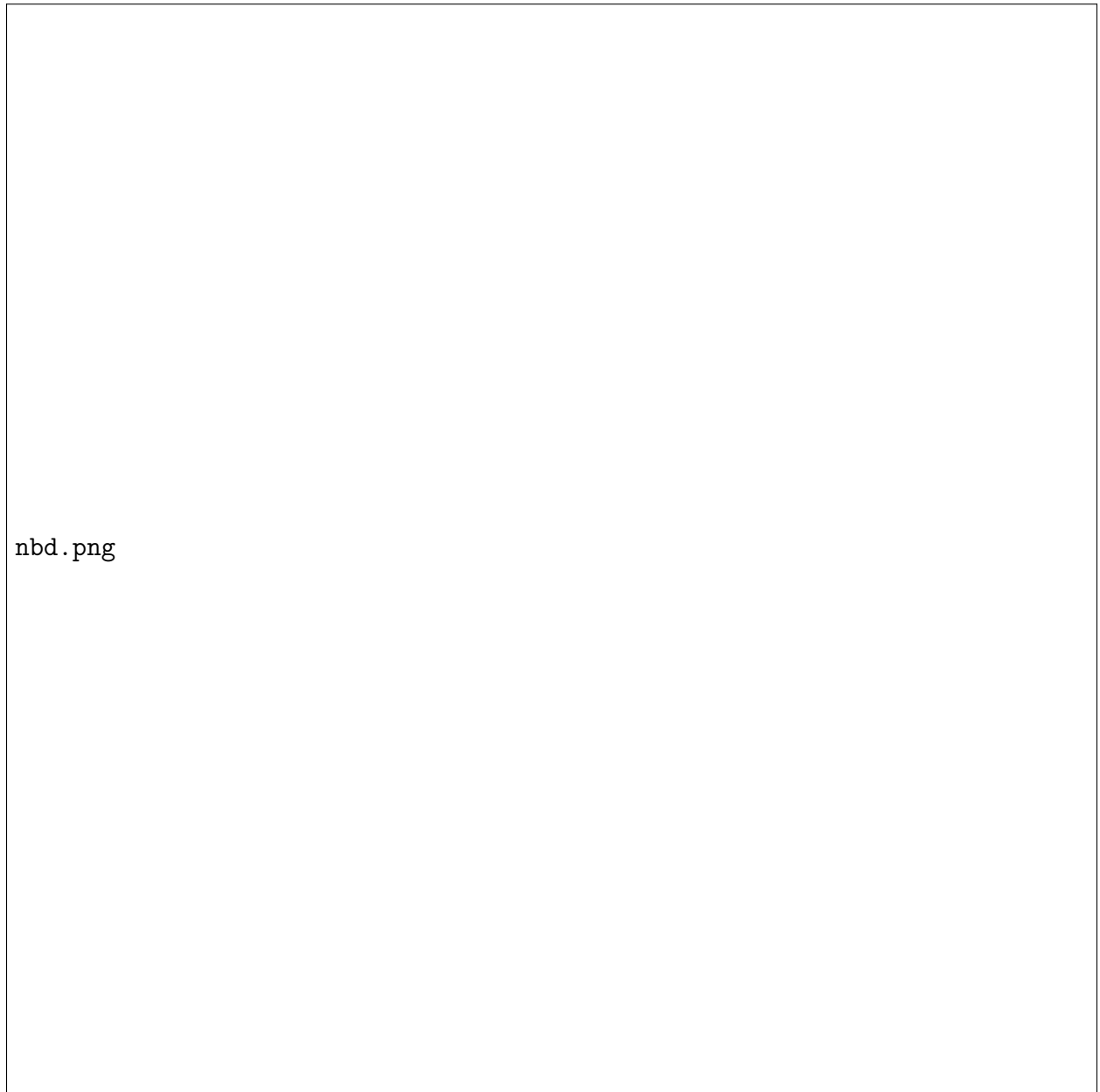


Figure 20: Advanced example design for the AXI VIP.



nbd.png

Figure 1: Block design.