

Hardware Debugging

Introduction

In this lab you will use the `uart_led` design that was introduced in the previous labs. You will use Mark Debug feature and also the available Integrated Logic Analyzer (ILA) core (in IP Catalog) to debug the hardware.

Objectives

After completing this lab, you will be able to:

- Use the Integrated Logic Analyzer (ILA) core from the IP Catalog as a debugging tool
- Use Mark Debug feature of Vivado to debug a design
- Use hardware debugger to debug a design

Procedure

This lab is broken into steps that consist of general overview statements providing information on the detailed instructions that follow. Follow these detailed instructions to progress through the lab1.

Design Description

The design consists of a uart receiver receiving the input typed on a keyboard and displaying the binary equivalent of the typed character on the LEDs. When a push button is pressed, the lower and upper nibbles are swapped when using the ZedBoard or the upper 4-bits are displayed with the Zybo. The block diagram is as shown in **Figure 1**.

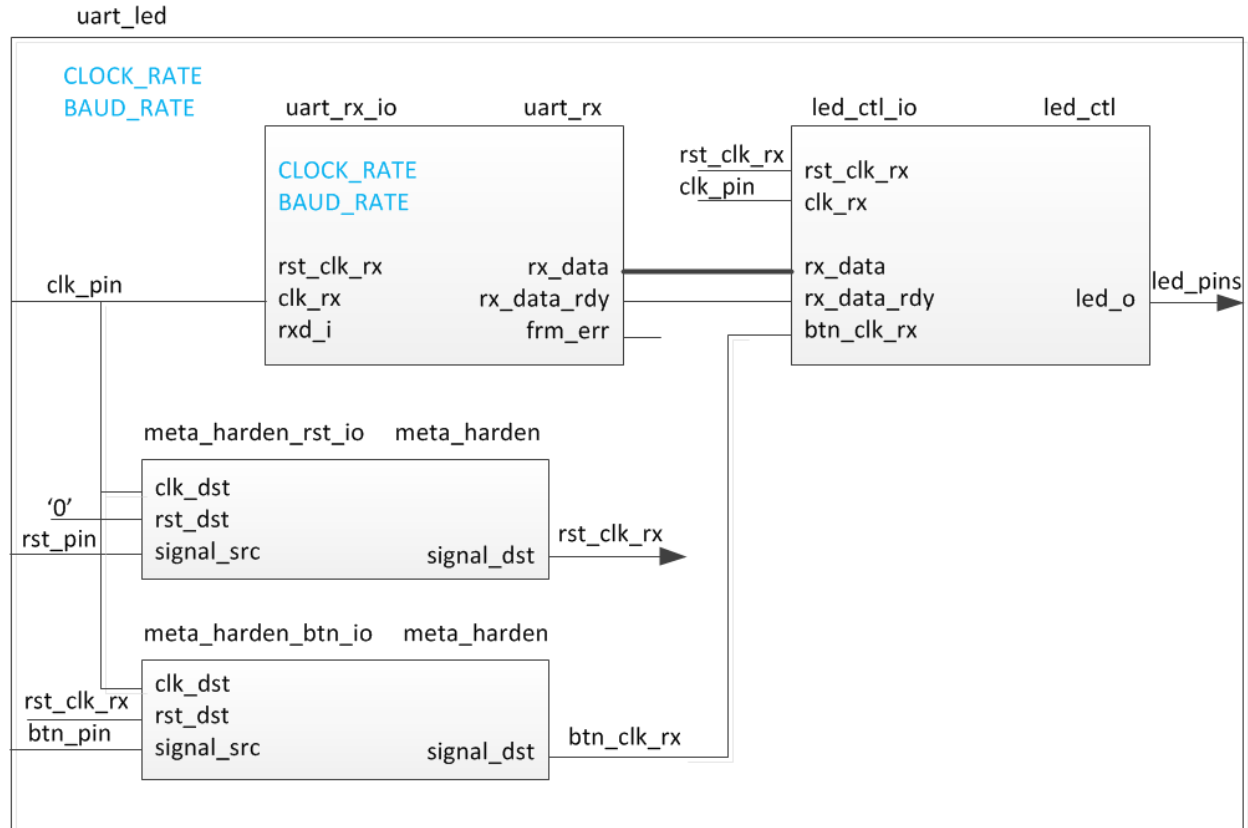
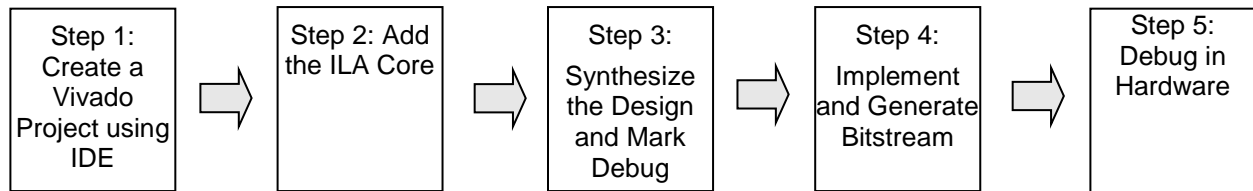


Figure 1. The Completed Design

General Flow



Create a Vivado Project using IDE

Step 1

- 1-1. Launch Vivado and create a project targeting the ZedBoard or the Zybo and using the Verilog HDL. Use the provided Verilog and Xilinx Design Constraints source files from the `<2014_2_zynq_sources>\<board>\lab6` directory.**

References to `<2014_2_zynq_labs>` is a placeholder for the `c:\xup\fpga_flow\2014_2_zynq_labs` directory and `<2014_2_zynq_sources>` is a placeholder for the `c:\xup\fpga_flow\2014_2_zynq_sources` directory.

Reference to `<board>` means either the **ZedBoard** or the **Zybo**.

- 1-1-1.** Open Vivado by selecting **Start > All Programs > Xilinx Design Tools > Vivado 2014.2 > Vivado 2014.2**
- 1-1-2.** Click **Create New Project** to start the wizard. You will see *Create A New Vivado Project* dialog box. Click **Next**.
- 1-1-3.** Click the Browse button of the *Project location* field of the **New Project** form, browse to `<2014_2_zynq_labs>`, and click **Select**.
- 1-1-4.** Enter **lab6** in the *Project name* field. Make sure that the *Create Project Subdirectory* box is checked. Click **Next**.
- 1-1-5.** Select **RTL Project** option in the *Project Type* form, and click **Next**.
- 1-1-6.** Select **Verilog** as the *Target Language* in the *Add Sources* form.
- 1-1-7.** Click on the **Add Files...** button, browse to the `<2014_2_zynq_sources>\<board>\lab6` directory, select all the six Verilog files (*led_ctl.v*, *meta_harden*, *uart_baud_gen*, *uart_led*, *uart_rx*, and *uart_rx_ctl*), click **OK**. Make sure the source files are copied into the project and then click **Next**.
- 1-1-8.** Click **Next** to get to the *Add Constraints* form.

You will see `uart_led_timing.xdc` and `uart_led_pins.xdc` have automatically been included.

Make sure the source files are copied into the project. Click **Next**.

1-1-9. In the *Default Part* form, using the **Parts** option and various drop-down fields of the **Filter** section, select the **XC7Z020CLG484-1** for the ZedBoard or **XC7Z010CLG400-1** for the Zybo Click **Next**.

1-1-10. Click **Finish** to create the Vivado project.

Add the ILA Core

Step 2

2-1-1. Click **IP Catalog** under the *Project Manager* tasks of the *Flow Navigator* pane.

2-1-2. The catalog will be displayed in the Auxiliary pane.

2-1-3. Expand the **Debug & Verification > Debug** folders and double-click the **ILA** entry.

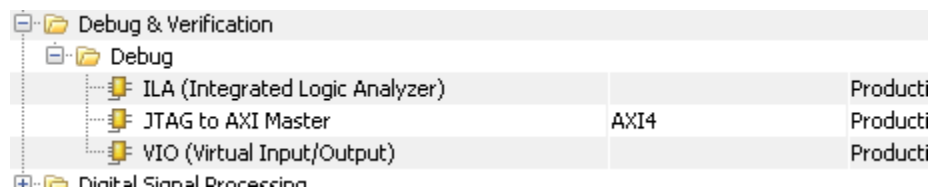


Figure 2. ILA in IP Catalog

We will be connecting the ILA core/component to the LED port.

2-1-4. Change the component name to **ila_led**.

2-1-5. Change the *Number of Probes* to **2**.

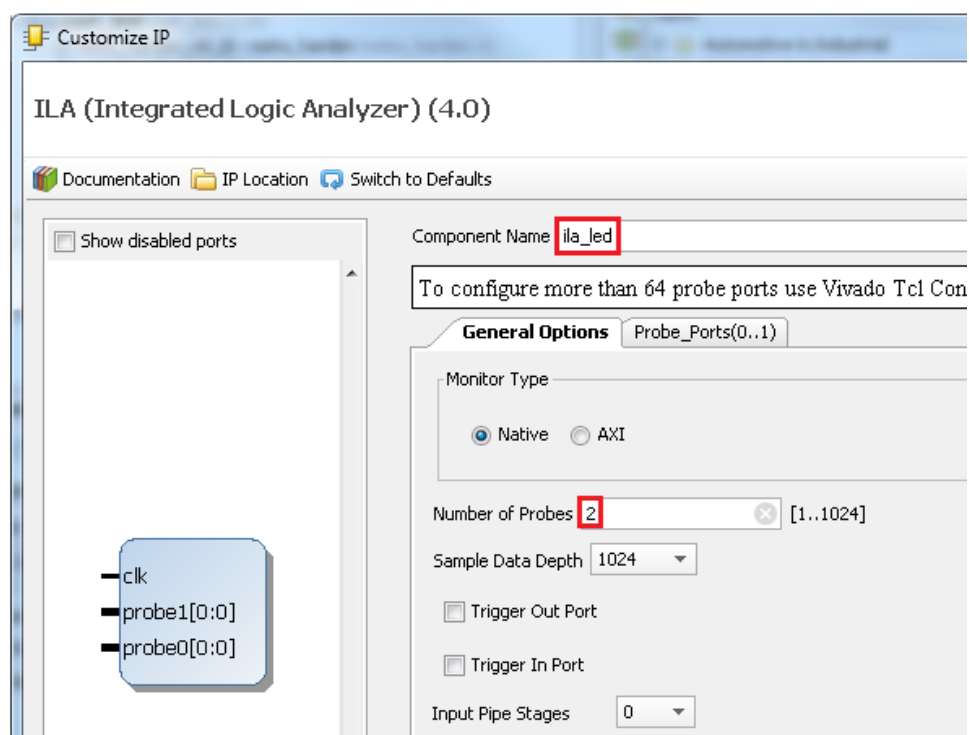


Figure 3. Setting the component name and the number of probes field

- 2-1-6.** Select the *Probe Ports* tab and change the PROBE1 port width to **8** for the ZedBoard and **4** for the Zybo, leaving the PROBE0 width to **1**. Leave the Number of Comparators as default for both probes.

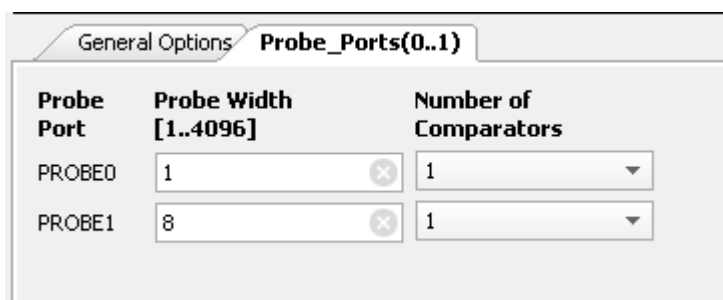


Figure 4. Setting the probes width for the ZedBoard

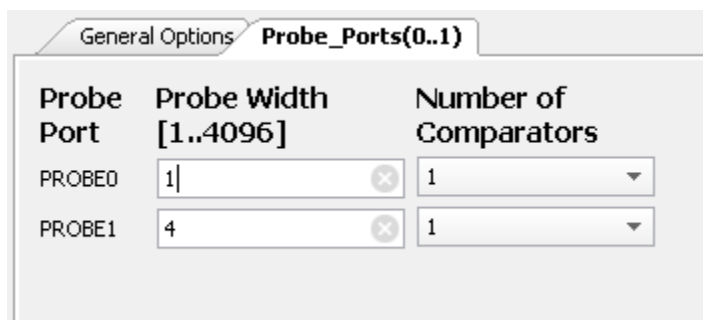


Figure 4. Setting the probes width for the Zybo

2-1-7. Click OK.

The Generate Output Products dialog box will appear.

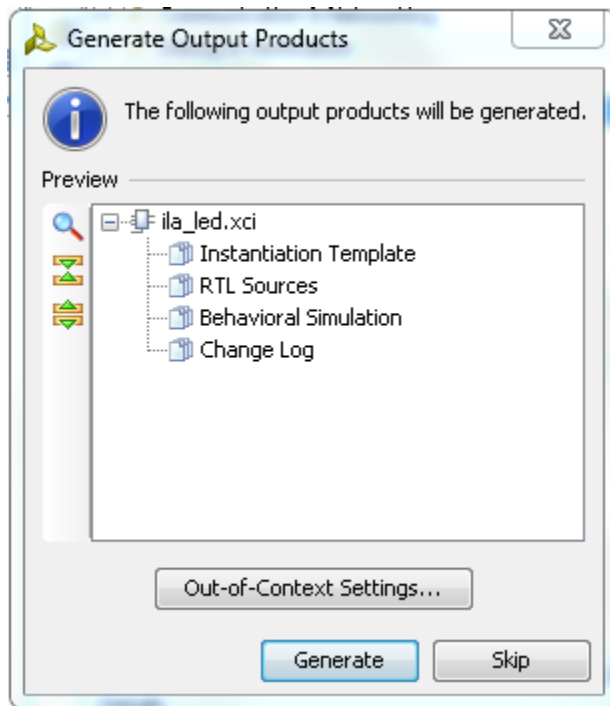


Figure 5. The Generate Output Products

2-1-8. Click the **Generate** button to generate the core including the instantiation template. Notice the core is added to the *Design Sources* view.

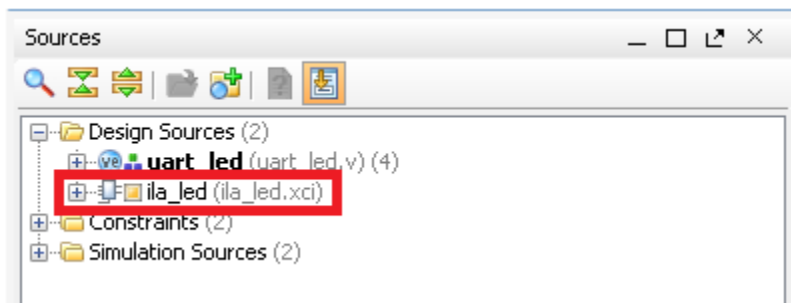


Figure 6. Newly generate ila core added in the design source

2-1-9. Select the **IP Sources** tab, expand **IP > ila_led > Instantiation Template** and double-click the **ila_led.veo** entry to see the instantiation template.

2-1-10. Instantiate the **ila_led** in design by copying lines 57 – 61 and pasting around line 104 (before “endmodule” on the last line) in the **uart_led.v** file.

2-1-11. Change *your_instance_name* to **ila_led_i0**.

2-1-12. Change the following port names in the Verilog code to connect the ila to existing signals in the design:

```
.CLK(CLK)          .CLK(clk_pin)
.PROBE0(PROBE0)    .PROBE0(rx_data_rdy)
.PROBE1(PROBE1)    .PROBE1(led_pins)
```

```
105 ila_led ila_led_i0 (
106     .clk(clk_pin),          // input wire clk
107     .probel(led_pins),      // input wire [7 : 0] probe1
108     .probe0(rx_data_rdy)    // input wire [0 : 0] probe0
109 );
110
111 endmodule
```

Figure 7. Instantiating the ILA Core in the uart_led.v

2-1-13. Select **File > Save File**.

Notice that the ILA Core instance is in the design hierarchy.

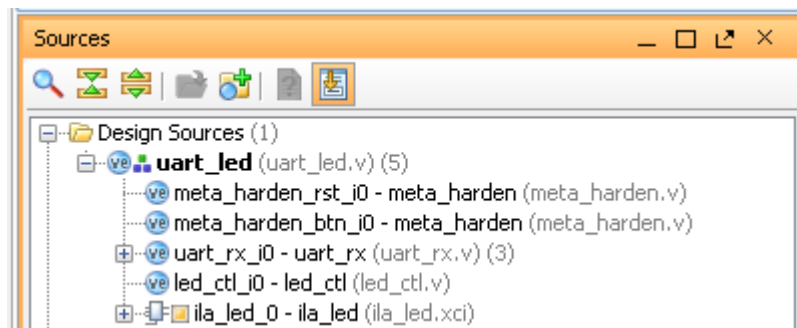


Figure 8. ILA Core added to the design

Synthesize the Design and Mark Debug

Step 3

3-1. Synthesize the design. Open the synthesized design. View the schematic. Add Mark Debug on the rx_data bus between the uart_rx_i0 and led_ctl_i0 instances.

3-1-1. Click on **Run Synthesis** under the *Synthesis* tasks of the *Flow Navigator* pane.

The synthesis process will be run on the uart_led.v and all its hierarchical files. When the process is completed a *Synthesis Completed* dialog box with three options will be displayed.

3-1-2. Select the *Open Synthesized Design* option and click **OK** as we want to look at the synthesis output.

3-1-3. Click on **Schematic** under the *Synthesized Design* tasks of *Synthesis* tasks of the *Flow Navigator* pane to view the synthesized design in a schematic view.

- 3-1-4. Select the **rx_data** bus between the *uart_rx_i0* and the *led_ctl_i0* instances, right-click, and select **Mark Debug**.

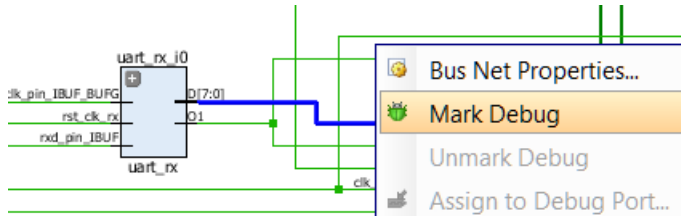


Figure 9. Marking a bus to debug

- 3-1-5. Confirm the debug nets selection. Click **OK**.

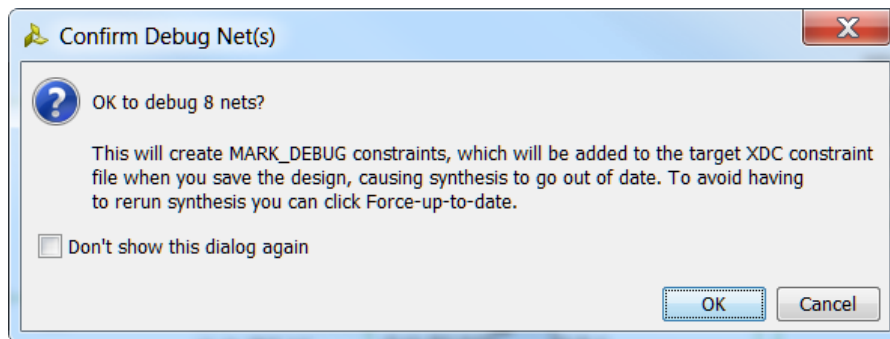


Figure 10. Confirm Debug Net(s) dialog box

- 3-1-6. Select the **Netlist** tab and notice that the nets which are marked/assigned for debugging have a debug icon next to them.

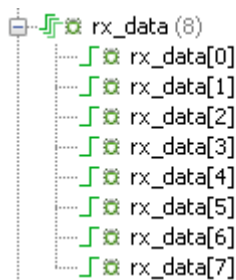


Figure 11. Nets with debug icons

- 3-1-7. If not already in the layout, select the **Debug** layout or **Layout > Debug**.

Notice that the **Debug** tab is visible in the Console pane showing Assigned and Unassigned Debug Nets groups.

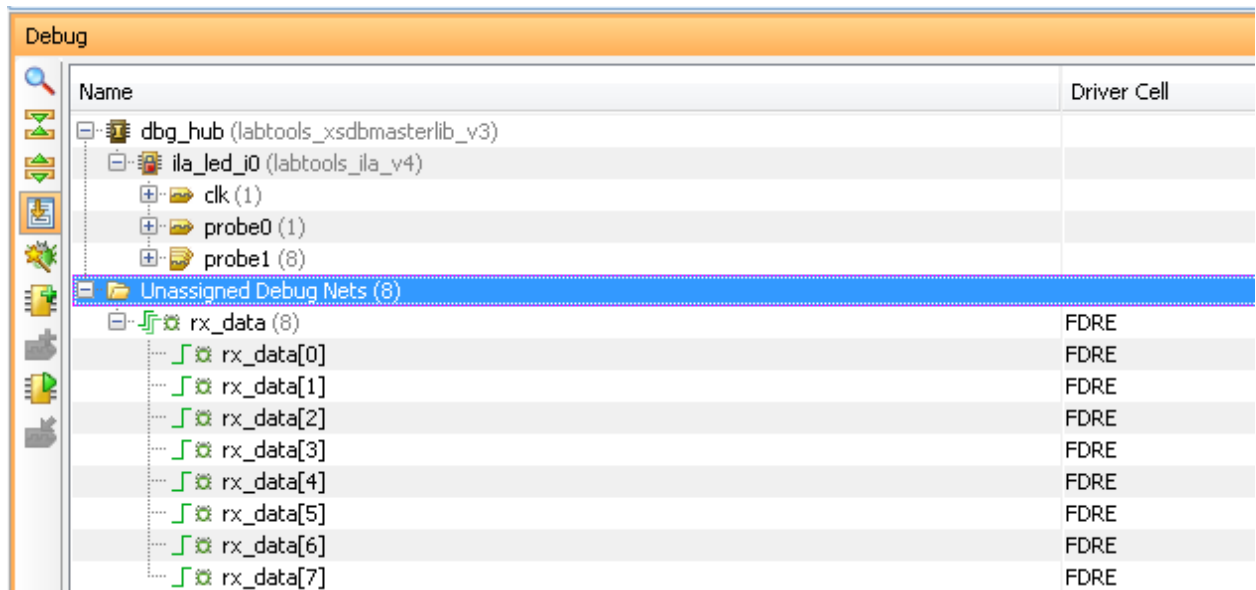






Figure 12. Debug tab showing assigned and unassigned nets

- 3-1-8.** Either click on the  button in the left vertical tool buttons of the Debug pane, or right-click on the *Unassigned Debug Nets* and select the **Set up Debug...** option.
- 3-1-9.** In the Set Up Debug wizard click **Next**.
- 3-1-10.** There is an error bar stating that there is no identified clock domain for the nets. Press Ctrl and select/highlight all three nets. Click on the **Select Clock Domain** button  on the side bar to assign a clock domain to the signals.
- 3-1-11.** In the Select Clock Domain window, there is an info message that nothing was found for GLOBAL_CLOCK. Click **OK** to close the box.
- 3-1-12.** In the Select Clock Domain window, select **ALL_CLOCK** from the drop down menu  and **uncheck** Display unique nets. Notice a list of clocks found. Select **xlnx_opt_** as the clock domain. Click **OK** to exit the window.
- 3-1-13.** The nets are now listed with a xlnx_opt_ selected as the clock domain. Since some of the nets are already assigned, we can remove them from this setup. Select and delete the led_pins(*) and rx_data_rdy nets. Click **Remove Nets**  on the side bar to remove the previously mentioned nets from the list.

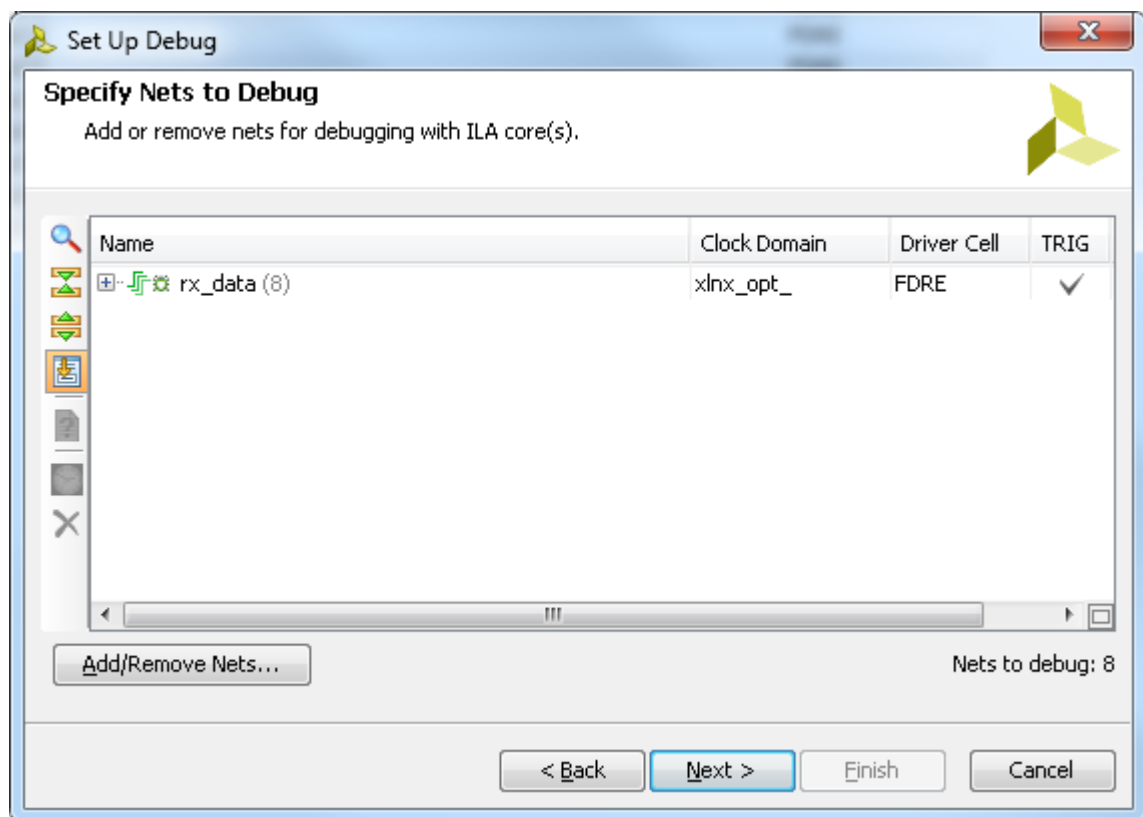


Figure 13. The remaining nets after removing already assigned nets in the Set Up Debug wizard

3-1-14. Click **Next** and again **Next** (leaving everything as defaults) then **Finish**.

3-1-15. In the Synthesized Design Schematic, click on the net on the output side of the IBUF for the input pin named clk_pin. Hover over the now highlighted net and notice the name is xlrx_opt_. This is the clock net selected for the debug nets earlier.

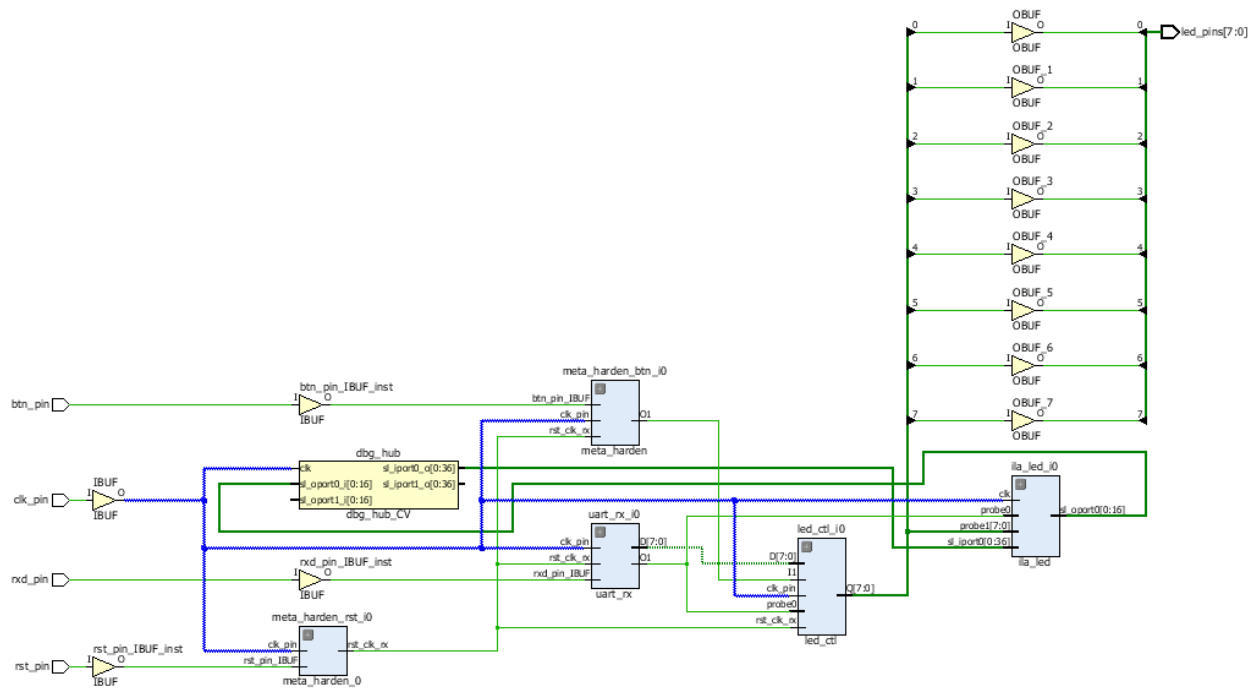


Figure 14. Locating xlnx_opt_ in the design for the ZedBoard

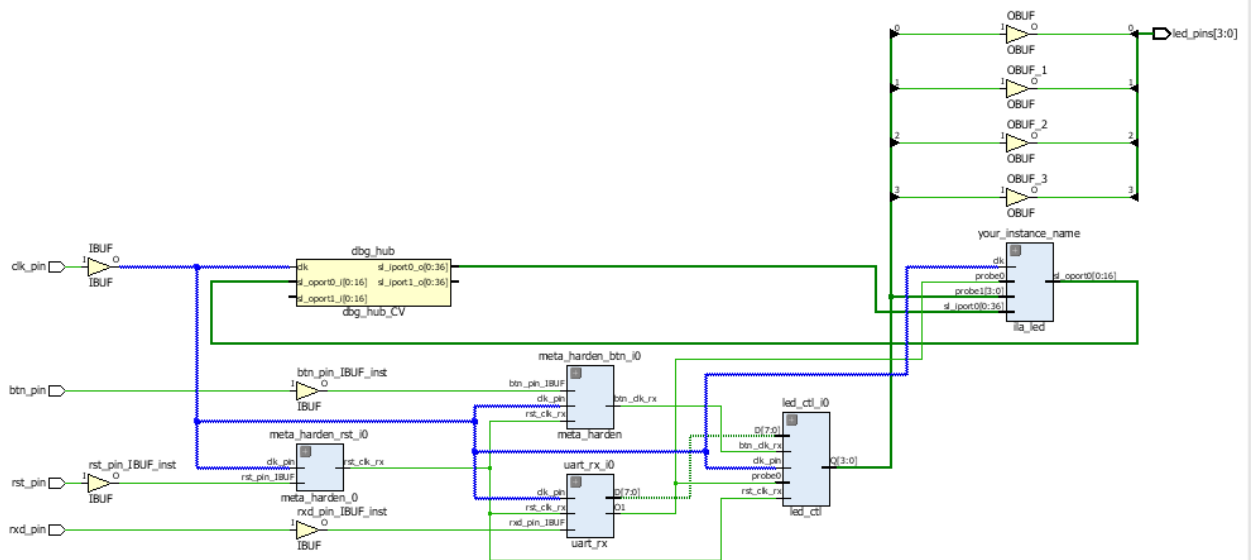


Figure 14. Locating xlnx_opt_ in the design for the Zybo

3-1-16. Right click on `uart_led_pins.xdc` in the sources pane and select **Set as Target Constrain File**. This will save the changes to the file.

3-1-17. Select **File > Save Constraints**. There may be a warning about synthesis going out of date from a saved constraints file, acknowledge it by clicking **OK**.

3-1-18. Open `uart_led_pins.xdc` and notice the debug nets have been appended to the bottom of the file.

- 3-1-19.** Perform this step if synthesis is not already up-to-date, in the **Design Runs** tab, right-click on the *synth_1* and select **Force Up-to-Date** to ensure that the synthesis process is not re-run, since the design was not changed.

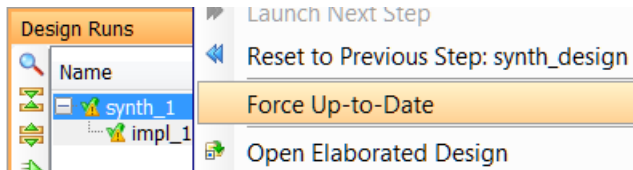


Figure 15. Forcing the design to be up-to-date

Implement and Generate Bitstream

Step 4

4-1. Generate the bitstream.

- 4-1-1.** Click on the **Generate Bitstream** to run the implementation and bit generation processes.
- 4-1-2.** Click **Yes** to run the processes.
- 4-1-3.** When the bitstream generation process has completed successfully, a box with four options will appear. Select the **Open Hardware Manager** option and click **OK**.

Debug in Hardware

Step 5

5-1. Plug-in the PmodUSB UART module into the top-row of the JA PMOD connector on the ZedBoard or the JE PMOD connector on the Zybo. Connect the module to the host machine using a micro-USB cable. Connect the board and power it ON. Open a Hardware Manager, and program the FPGA.

- 5-1-1.** Plug-in the PmodUSB UART module into the top-row of the JA PMOD connector of the ZedBoard. If using the Zybo, plug the PmodUSB UART module to the top-row of the JE PMOD connector.
- 5-1-2.** Connect the micro-USB cable between the PmodUSB UART module and the host USB port.
- 5-1-3.** Make sure that the Micro-USB cable is connected to the JTAG PROG connector (next to the power supply connector). Connect the power jack and turn ON the power.

5-1-4. Select the *Open Hardware Manager* option.

The Hardware Manager window will open indicating “unconnected” status.

5-1-5. Click on the **Open New Hardware Target** link.

You can also click on the Open Recent Hardware Target link if the board was already targeted before.

5-1-6. Click **Next** to see the Hardware Server Settings form.

5-1-7. Click **Next** with the JTAG Clock Frequency of **15000000** selected.

The JTAG cable which uses the diligent_plugin should be detected and identified as a hardware target. It will also show the hardware devices detected in the chain.

5-1-8. Click **Next** and click **Finish**.

5-1-9. The Hardware Manager status changes from Unconnected to the server name and the device is highlighted. Also notice that the Status indicates that it is not programmed.

5-1-10. Select the device and verify that the **uart_led.bit** is selected as the programming file in the General tab.

5-2. Start a terminal emulator program such as TeraTerm or HyperTerminal. Select an appropriate COM port (you can find the correct COM number using the Control Panel). Set the COM port for 115200 baud rate communication. Program the FPGA and verify the functionality.

5-2-1. Start a terminal emulator program such as TeraTerm or HyperTerminal.

5-2-2. Select an appropriate COM port (you can find the correct COM number using the Control Panel).

5-2-3. Set the COM port for 115200 baud rate communication.

5-2-4. Right click on the target FPGA and select **Program Device...** . Click **Program**.

The Hardware Manager will open showing the **Debug Probes** window. Notice that there are two debug cores.

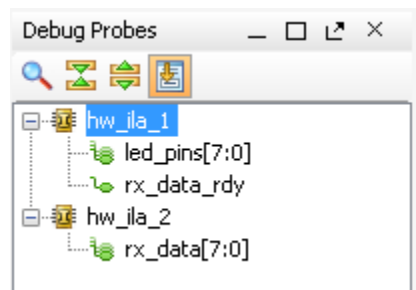


Figure 16. Debug probes

The Hardware Manager status window also opens showing that the FPGA is programmed having two ila cores with the idle state.

| Name | Status |
|--------------------------------------|----------------|
| localhost (1) | Connected |
| xilinx_tcf/Digilent/210248470364 (2) | Open |
| arm_dap_0 (0) | Not programmed |
| xc7z020_1 (3) | Programmed |
| XADC (System Monitor) | |
| hw_ila_1 (ILA) | Idle |
| hw_ila_2 (ILA) | Idle |

Figure 17. Hardware Manager status for the ZedBoard

| Name | Status |
|------------------------------------|----------------|
| localhost (1) | Connected |
| xilinx_tcf/Digilent/21027954024... | Open |
| arm_dap_0 (0) | Not programmed |
| xc7z010_1 (3) | Programmed |
| hw_ila_1 (ILA) | Idle |
| hw_ila_2 (ILA) | Idle |
| XADC (System Monitor) | |

Figure 17. Hardware Manager status for the Zybo

- 5-2-5. Select the XC7Z020 or XC7Z010, and click on the **Run Trigger Immediate** button to see the signals in the waveform window.

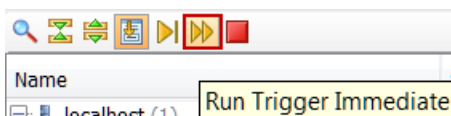


Figure 18. Opening the waveform window

Two waveform windows will be created, one for each ila; one ila is of the instantiated ILA core and another for the MARK DEBUG method.

- 5-3. **Setup trigger conditions to trigger on a write to led port (rx_data_rdy=1) and the trigger position to 512. Arm the trigger.**

- 5-3-1. In the **Debug Probes** window, right click on **hw_ila_1** and select **Add Probes to Basic Trigger Setup**. This adds all the probes for that ILA into the **ILA – hw_ila_1** trigger setup window.

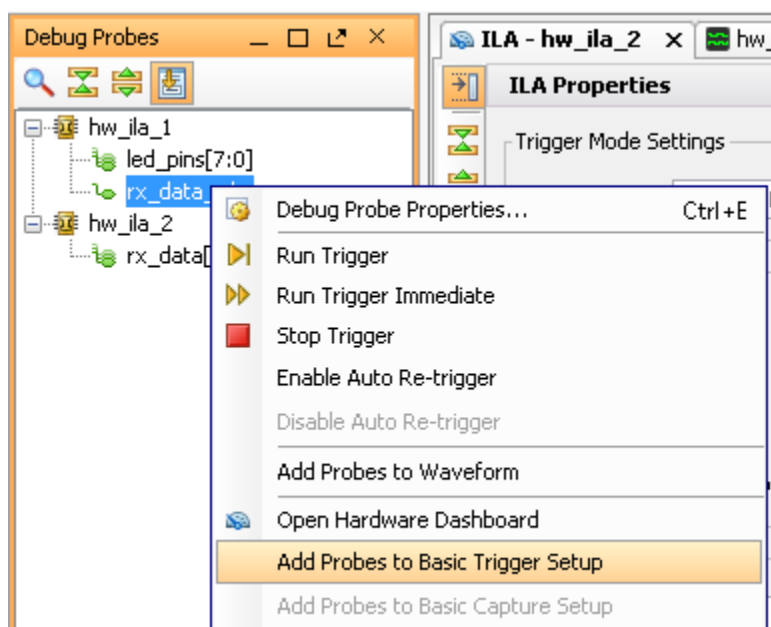


Figure 19. Adding a signal to trigger setup

5-3-2. Click on the **rx_data_rdy** compare value (== [B] X) and change the value from x to 1. Click **OK**.

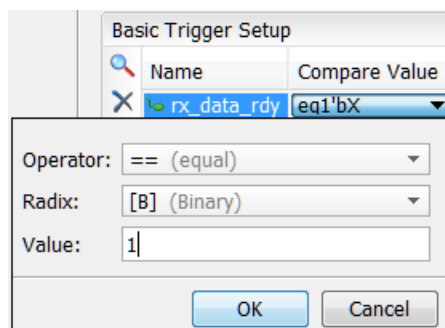


Figure 20. Setting the trigger condition

5-3-3. Set the trigger position of the *hw_ila_1* to **512**.

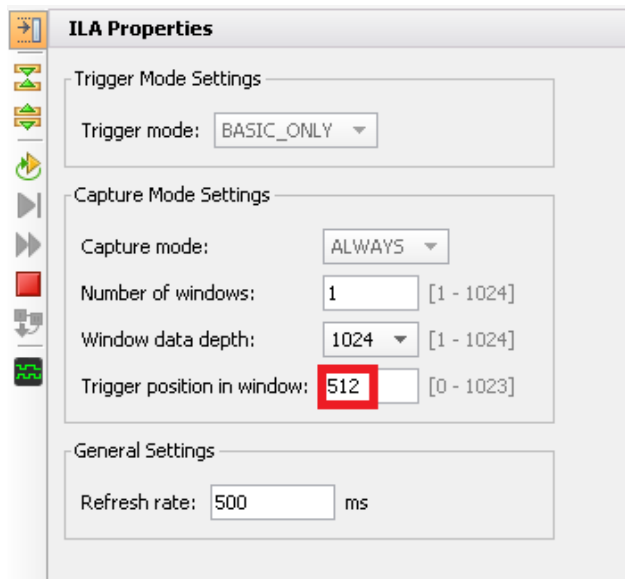



Figure 21. Setting up the trigger position

5-3-4. Similarly, right click on *hw_ila_2* in the **Debug Probes** window and select **Add Probes to Basic Trigger Setup**. Set the trigger position of the *hw_ila_2* to 512.

5-3-5. Highlight *hw_ila_1* in the Debug Probes window and click on the Run Trigger () button and observe that the *hw_ila_1* core is armed and showing the status as **Waiting For Trigger**. Also note that the *hw_ila_2* status is Idle.

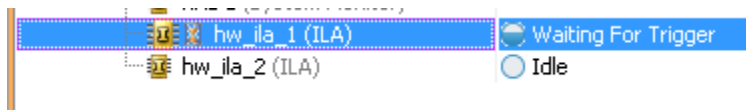


Figure 22. Hardware analyzer running and in capture mode

5-3-6. In the terminal emulator window, type a character, and observe that the *hw_ila_1* status changes from Waiting For Trigger to Idle as the *rx_data_rdy* became 1.

5-3-7. Select the *hw_ila_data_1.wcfg* window and see the waveform. Notice that the *rx_data_rdy* goes from 0 to 1 at 512th sample and the *led_ctl_i0* [7:0] bits also changes from 0 to the bit pattern corresponding to the character you typed.

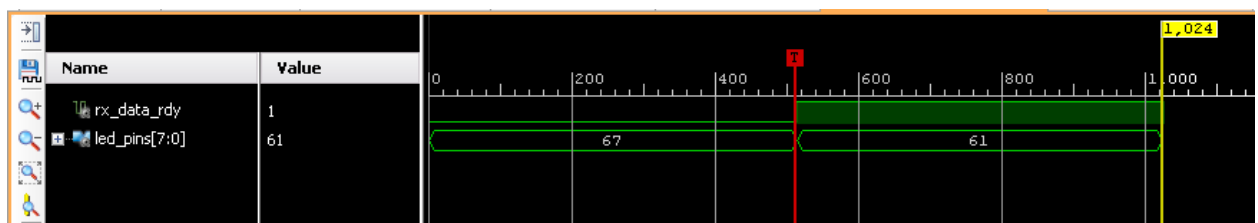


Figure 23. Zoomed waveform view for the ZedBoard

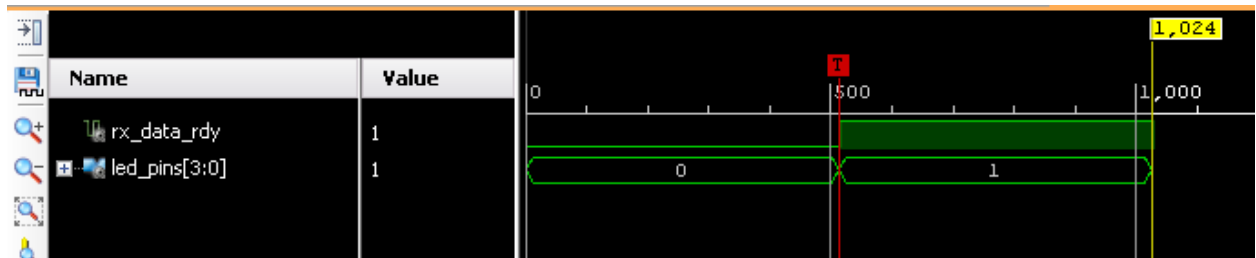


Figure 23. Zoomed waveform view for the Zybo

- 5-3-8.** Add the `hw_ila_2` probes to the trigger window of the `hw_ila_2` and change the trigger condition for `rx_data[7:0]`'s Radix from Hexadecimal to Binary. Change `XXXX_XXXX` to `0101_0101` (for the ASCII equivalent of U). Click **OK**.

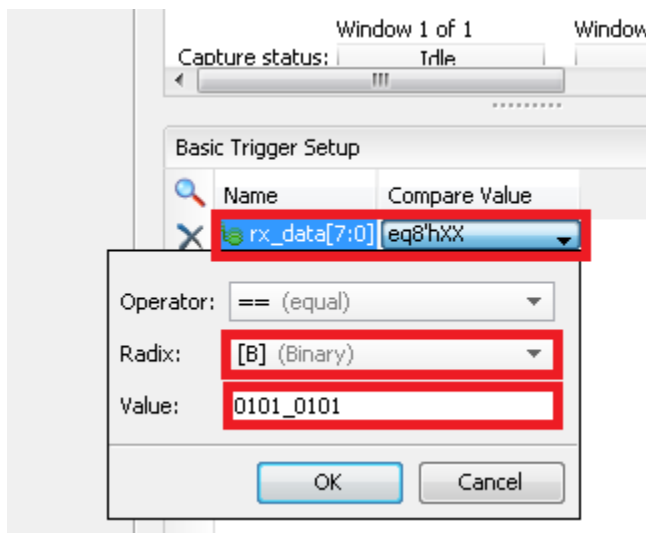


Figure 24. Setting up trigger condition for a particular input pattern

- 5-3-9.** In the Hardware Analyzer window, right-click on the `hw_ila_2` and select **Run Trigger**, and notice that the status of the `hw_ila_2` changes from *idle* to *Waiting For Trigger*. Also notice that the `hw_ila_1` status does not change from *idle* as it is not armed.
- 5-3-10.** Switch to the terminal emulator window and type **U** (capital letter; on keyboard "shift+u") to trigger the core.
- 5-3-11.** Select the corresponding waveform window and verify that it shows 55.

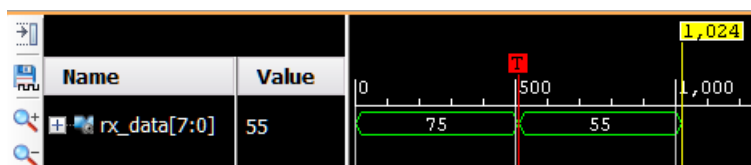


Figure 25. Second ila core triggered

- 5-3-12.** When satisfied, close the terminal emulator program and power OFF the board

5-3-13. Select **File > Close Hardware Manager**. Click **OK** to close it.

5-3-14. Close the **Vivado** program by selecting **File > Exit** and click **OK**.

Conclusion

You used ILA core from the IP Catalog and Mark Debug feature of Vivado to debug the hardware design.