JACOBS
UNIVERSITY

Jacobs University Bremen

CA-ECE-803: Digital design

Title: Intro to Xilinx Synthesis

Lab report: 2

Experiment conducted by:
Betelhem Nebebe
Maria Namachanja
Neeha Hammad

Instructor: Prof. Dr. Fangning Hu
Execution date: February 23, 2021

1. Introduction

The goal of this lab is to gain experience with the Xilinx Synthesis tools, learn how to write UCF files for Spartan 3 boards and gain an understanding of how FPGAs implement logic functions.

2. Synthesize and Test a Design

To synthesize a design, first launch the Project Navigator. Head to **File -> New Project**. Once you have created a project and added the source files, your project will now be ready to be synthesized.

To synthesize your files, select your top level file and right click on the icon in the **"Process View"** box (in the lower half of the left side of the screen) labeled **"Synthesize - XST"**. Select the green **"Run"** button from the context menu. You should see the console fill with output and a progress wheel should appear in the lower right corner.

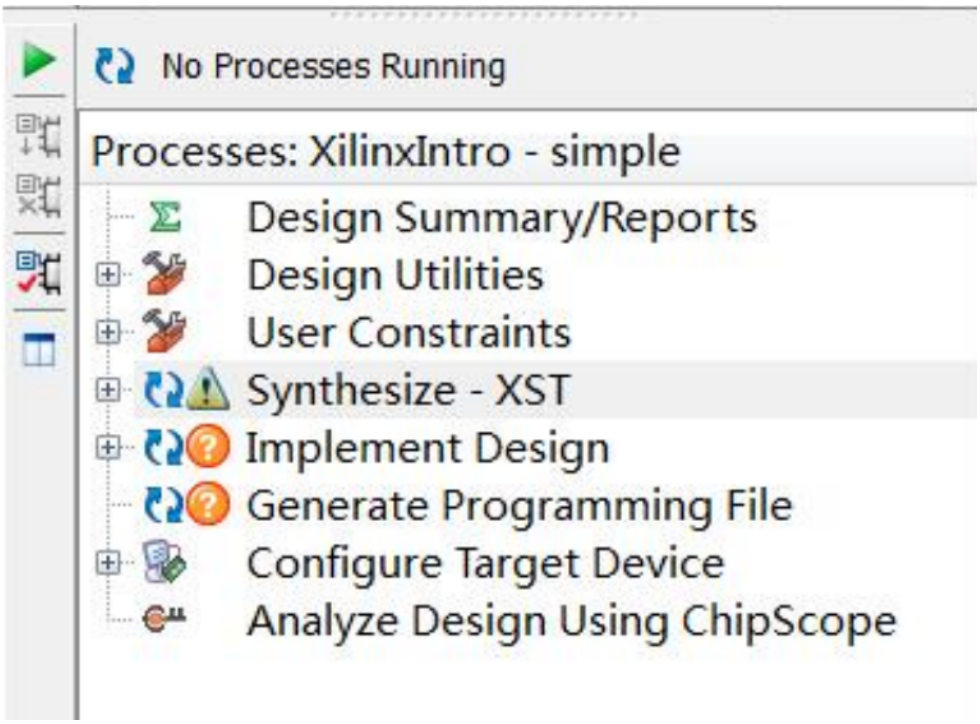
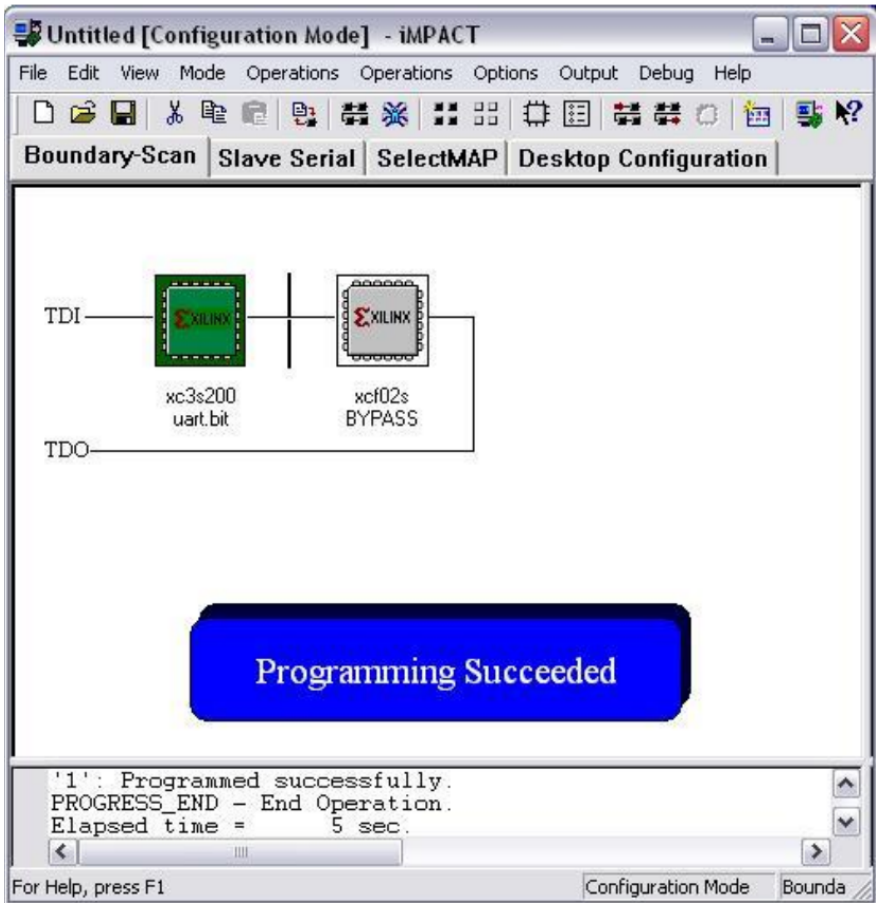


Figure 1: Process View Box

Synthesis can either throw an error, throw a warning, or resolve successfully. If synthesis throws errors or warnings, they will appear in the console window as well as their appropriate tab of the output window.

Once the synthesizing is successful, it is now time to test the design. Make sure your programming cable is connected correctly and that your Spartan 3 board is connected to the power supply before you continue.

Now, click on **Generate Programming File** from the sub-menu in **Project Navigator** to go to **iMPACT**. Once iMPACT is launched, go to **File -> Initialize Chain.** Now, you will see another window where you can assign a new configuration file. Navigate to your .bit file (it will be in your project directory) and select open. ALWAYS select bypass or cancel for the second chip. To download right click on the left chip and select "Program..." A dialog box should pop up. Click "okay". If everything went well, your program should now be on the board and running.



For better understanding, we can also modify the UCF file and add more switches and LEDs to see the differences in the outcome.

NET "switches<0>" LOC="F12" ;

NET "switches<1>" LOC="G12" ;

NET "switches<2>" LOC="H14" ;

NET "switches<3>" LOC="H13" ;

NET "switches<4>" LOC="J14" ;

NET "switches<5>" LOC="J13" ;

NET "switches<6>" LOC="K14" ;

NET "switches<7>" LOC="K13" ;

NET "buttons<0>" LOC="M13" ;

NET "buttons<1>" LOC="M14" ;

NET "leds<0>" LOC="K12" ;

NET "leds<1>" LOC="P14" ;

NET "leds<2>" LOC="L12" ;

NET "leds<3>" LOC="N14" ;

NET "leds<4>" LOC="P13" ;

NET "leds<5>" LOC="N12" ;

NET "leds<6>" LOC="P12" ;

NET "leds<7>" LOC="P11" ;

Note: We were present in the lab but couldn't take pictures of the equipment.

3. Synthesis Errors/Warnings

One of the warnings we noticed was regarding the button numbers. The code included buttons 3 and 2 as well even though we were not using those buttons. To fix this, you have to modify the following line of code in the vhd file:

buttons : in std_logic_vector(3 downto 0);

Since we only require buttons 0 and 1, we can change the line to this:

buttons : in std_logic_vector(1 downto 0);

4. UCF File

Xilinx UCF is a constraints file format. It is used to apply various constraints on the design to the Xilinx tools, including but not limited to pin locations, timing, and area. For instance, in this case, we were specifying the buttons, switches, and leds in the UCF file, along with their respective locations from the board.

When mapping your RTL (VHDL) design to a physical device, you will minimally need clocks and pin constraints at the top level, but constraints for real designs can go much further than this.

5. Other Problems Encountered

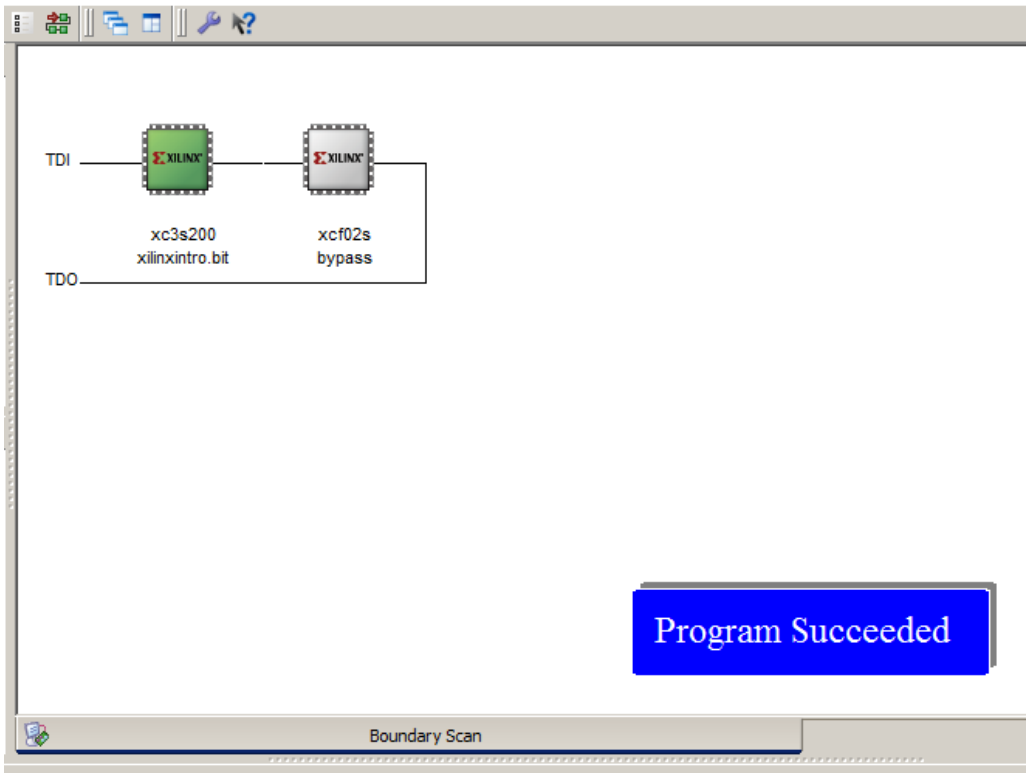
Initially, upon launching the iMPACT software, we were not seeing an option to assign a new configuration file. This was happening because we were using the wrong pin on our Spartan 3 board. This was fixed once we switched the pin on our board.

Additionally, we had a problem finding the Process View window at first. We simply asked the TAs where to find it. For future reference, it is the box shown in **Figure 1.**

Conclusion

The end goal of this lab which was to have a simple design on the board that displays whether the switches are on or off on the LEDs was achieved.

As seen below we completed the tutorial's steps successfully.



References

http://fpga-fhu.user.jacobs-university.de/?page_id=402
https://electronics.stackexchange.com/questions/108683/usage-of-ucf-file-and-clock-divider