

Lab Assignment 8

Programming FPGA using Simulink

The goal of this lab is to develop a digital design using Simulink, and download the design and run it on the FPGA. You will be using the Simulink HDL Coder to program the FPGA on the ZedBoard. This lab builds upon the subsystems built in lab 6. We will control the counter through switches, use pushbuttons to load the counter with a predefined value, and change its counting direction and counting speed.

Instructions

- Each lab assignment consists of a set of pre-lab questions, the actual time in the lab, and a lab report.
- The pre-lab assignments prepare you for the challenges you will be facing in the actual lab. So that each lab member gets the same benefits, pre-lab assignments have to be solved individually. Feel free to discuss problems with class and lab mates, but complete the assignment on your own.

Reading List

The following reading list will help you to complete the pre-lab assignment. The readings will also help you in subsequent lab assignments. Please complete the readings that are marked **[Required]** before attempting the pre-lab assignments.

- [1] Chapter 2 of Simulink tutorial, how to program FPGA
 - <https://blackboard.neu.edu/>
- [2] Mathworks materials on HDL coder
 - https://www.mathworks.com/products/hdl-coder/?s_tid=hp_fp_list

Pre-Lab Assignment

We have only very limited time for the lab available. To make efficient use of the lab time, you will need to prepare for it. First, go through the assigned reading above and get an overview of the schematics and the manuals mentioned. Second, approach the questions below to apply your knowledge. Please submit the solution to the pre-lab assignments in PDF format via blackboard.

Pre-Lab 8.1 Increment / Decrement Counter

Using the principles of lab 6, design a counter that increments by one if the PBNTU button is pushed, and decrements by one the PBTND button is pushed. Note: on the HDL counter, you can choose to use an enable port, as well as a direction port.

- What signal input is necessary to increment the counter by one? Recall that an HDL counter in Simulink increments the count during each simulation cycle when the enable input is high.
- What signal input is necessary for the counter to count down by one? Record your answers to both questions.
- Implement a push button counter in Simulink that can count up and down by pressing a push button. In your report, include a printout of your design. Report in your pre-lab writeup your findings from the simulation and errors you may have encountered.


Hint: Parts of the On/Off Button subsystem (see lab 6.3) will help you realize the control for the push button counter.

Pre-Lab 8.2 Change Counting Speed

The cascaded counter used in lab 6.2 counts at a constant speed. What changes are necessary to control the speed of the second counter? Discuss the purpose of the reset port in the first counter. Please describe your proposed solution in your pre-lab writeup.

Simulink Simulation Hint:

Simulink Design Settings:

- For each of your Simulink designs, set the solver to a discrete solver. Open the configuration settings in menu entry *Simulation* → *Model Configuration Parameters*. In the dialog, select:
 - Type: Fixed-step
 - Solver: Discrete (no continuous states)
 - Fixed-step size: 1
- On your scopes, click on the *Settings*  icon, click on the *History* tab, and **uncheck** option *Limit data points to last 5000*. You should remove the scopes when downloading into the FPGA.
- Your outermost inputs and constants should be set with the appropriate **data types** (*Boolean*, *int8*, etc) and a **sampling time** of *1*. The rest of the inputs and outputs in your subsystems can either be left to inherit the data type and sampling time, or set to the appropriate data type if necessary. You can also use the *Data Type Conversion* block if needed.

Lab Assignment

Connecting to ZedBoard

1. Login into the Windows desktop PC (we will refer to this system as the host) using your myneu credentials.
2. Make sure your Zedboard is powered (plugged in and the power switch on the board is in the ON position), and has been given time to boot up (generally takes a few minutes). Then connect the Zedboard to the PC host via Ethernet to the RJ45 connector.
3. Make an SSH connection to the Zedboard, where the IP address is 192.168.1.10, and port number of 22.

Don't work as root, you already created your user accounts, use that!

Opening your required tools

1. Open the "System Generator" from All Programs -> Xilinx Design Tools -> ISE Design Suite 14.4 -> System Generator.
2. With Matlab open, open the Simulink library browser by clicking on its icon or typing simulink in Matlab command line.

Overall Lab Goal

This lab will guide you through the steps necessary to develop a pushbutton-controlled counter, similar to the one implemented in software in Lab 3. This time, you will design the counter in Simulink, and run it on the FPGA. The overall functionality and switch assignment is as follows:

Switches:	define the start value of the counter
PBTNL:	load counter value from switches
PBTNC:	start or stop counting
PBTNU:	increase speed
PBTND:	decrease counting speed
PBTNR:	change the direction (i.e., increment vs. decrement)

Do not try to design this all at once. The design would be too complex. As with any design, it is a good idea to start with small steps. The following steps guide you through the steps individually.

This lab assignment will ask you to reuse the components built in Labs 5 and 6. Only minor modifications should be needed. If you find yourself designing a lot of new logic, you probably did not take full advantage of the components developed in the previous labs.

Lab 8.1 Start / stop counter

Implement an 8-bit counter that outputs its value to the LEDs, and can be controlled by pushbutton PBTNC to start/stop counting. Save this model as LedCount.slx. Validate the functionality on the ZedBoard.

Hint: utilize the cascaded counter design (lab 6.2) as a basis for the counter. Use the result of lab 6.3 (turning LED on/off with a pushbutton) to enable (start) / disable (stop) the counter.

Define the counting speed to be 2Hz. Report in your lab report how to set the parameters of the counter.

Lab 8.2 Change Direction

Expand the design of `LedCount.slx` to toggle the direction of counting with each push of PBTNR. Use the same design principles you applied in part 6.4 for enabling/disabling the counter. Save this model as `LedCount_dir.slx`. Validate the functionality on the ZedBoard.

Lab 8.3 Load Value

Expand the design of `LedCount_dir.slx` to allow the user to set a starting value of the 8-bit counter through the switches (the number is entered as a binary). The 8-bit counter should load the value from the switches only if the button PBTNL is pushed. Otherwise, the 8-bit counter should count freely, as designed before. Save this model as `LedCount_load.slx`. Validate the functionality on the ZedBoard.

Note: you can configure the 8-bit counter to expose a load port. It will then also automatically expose a “load enable” port. If the load enable is 1, then the counter is loaded with the value present at the load port. Note that the load enable port should be active for only one cycle. Hint the control of the load enable is very similar to the control in Pre-Lab 7.1 increment/decrement Counter.

Lab 8.4 Change Counting Speed

Extend the design `LedCount_load.slx` to control the counting speed of the 8-bit counter. The PBTNU pushbutton should increase the counting speed, and PBTND pushbutton should decrease the speed. Save this model as `LedCount_speed.slx`. Validate the functionality on the ZedBoard.

Hint: Expose the reset port of the first counter in the cascaded counter design. Implement a circuit to trigger a reset when the counter reaches a certain value even if the counter is set to “free running more”. Validate that this design is working identical to the range limited counter.

Reminder: Downloading the design to the FPGA takes about 15 minutes. Simulate your design in Simulink and make sure it works before you download it to the platform.

Laboratory Report

You should follow the lab report outline provided on Blackboard. Your report should be developed on a wordprocessor (e.g., OpenOffice or LaTeX), and should include graphics when trying to present a large amount of data. Include the output of compiling and running your programs. Upload the lab report on blackboard. Include screenshots of your design in the appendix of your lab report.