# ENEL 453 Lab 2: Enhancements to Lab 1

## 1. Revision History

| Version | Comments |
|---|---|
| v1 | Initial document |

Path: C:\Users\donen\Dropbox\U of C\Teaching\ENEL 453\ENEL 453 F2020\Labs\Lab 2\ENEL 453 Lab 2 v1.docx

## 2. Introduction

This is the second lab project of the course and its focus for the team is to become more familiar with your design and VHDL. You will extend design of Lab 1 to add some functionality. The requirements will be given at a high-level, to encourage you to complete more conceptual design.

The prerequisite knowledge for Lab 2 is a solid understanding of your Lab 1 design and a solid understanding of using the Quartus and ModelSim tools and debugging. It is vital by the end of Lab 2, for you to become comfortable with completing a moderately complex FPGA design in VHDL (writing RTL code and instantiating it within higher level modules); debugging in Quartus and exploring a design with the RTL Schematic; and writing testbenches and simulating effectively with Modelsim. Otherwise, Labs 3 and 4 will be extremely challenging for you.

The basic elements of this project are to:

- Review all lab and lecture videos to date.
- Develop a high-level design approach to meet the project requirements. This is most likely a block diagram of the entire design. The instructor and TAs will not be able to help you effectively without you communicating your design intentions with a block diagram.
- Design and write the new modules required for the project and write their testbenches, including an updated version for the top level. Use both Quartus and ModelSim effectively to complete your design. Build up your design with verified modules (i.e. use unit testing), before integrating them into the top level (for system-level testing), otherwise your debugging will be painful.
- Modify the .QSF file as needed. Download your FPGA configuration to the DE10-Lite to verify the design in physical hardware.
- Be prepared to do extensive debugging, work effectively as a team, and start your work in a timely manner so that you can get help from the office hours and lab sessions.
- Upload the required videos and documents to the D2L Dropbox before the deadline.

## 3. Technical Requirements

The project technical requirements are as follows.

**3.1.** Increase the functionality of Lab by adding two additional modes of operation for the 7-segment displays. The modes of operation will include:

1. Hexadecimal output (existing Lab 1 functionality)
2. Decimal output (existing Lab 1 functionality)
3. Stored output based on a switch or pushbutton input (new for Lab 2)
4. Hardcoded "5A5A" output (new for Lab 2)

You must provide testbenches for the modules you add/modify.

**Stored output:** Under user control (e.g. when a pushbutton is pressed), the current 7-segment display value will be stored. Then when operational mode 3 (stored output) is selected, then the stored value will be presented on the 7-segment displays.

Example: The user is in mode 1 and uses the slide switches to display **F3** on the 7-segment displays (SW(7 downto 0) is 11110011). The user then presses a pushbutton to store **F3** in a register. The user then goes into mode 2 and displays **11** using the slide switches (SW(7 downto 0) is 00001011). Then without changing binary value on the slide switches, the user goes into mode 3 and system displays the stored value of **F3** (SW(7 downto 0) is 00001011).

**Hardcoded "5A5A" output:** When the user enters operational mode 4, the system will display 5A5A, regardless of slide switch positions. This is a test mode to output "engineering values" that is useful for debugging purposes (reduces the dependence on input circuitry).

Hints:

1. Since you have 4 outputs based on the 4 operational modes, you'll probably need to modify your multiplexer to take in the 4 inputs possible inputs to the 7-segment displays. You will have to study the design and determine where and how you can take in any additional control inputs for this added functionality, if required to control the modified multiplexer. Include a testbench for this component.
2. Since you have to store a value, you probably need to use a register. See Lecture 8 for how to create a register with a Load signal. You will have to study the design and determine where and how you can take in any additional control inputs for this added functionality, if required. Include a testbench for this component.
3. It will be useful for you to draw out the design of top_level on a piece of paper (use the RTL Viewer to help visualize your circuit from Lab 1), then draw the modifications required to add and connect any additional modules, then code what you drew. *Drawing out circuits has been reported by many students as very helpful for designing with FPGAs.*

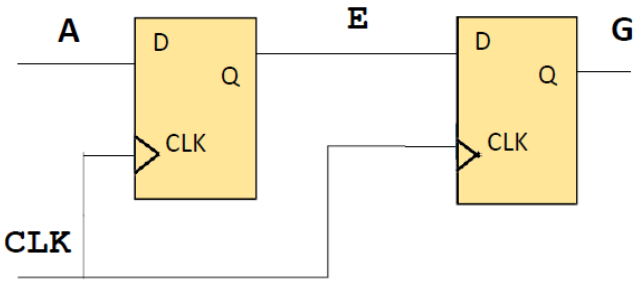
**3.2.** The top level must be only structural code, there must be no logic or RTL code in it. This is good design practice and will allow you to complete unit testing on the lower level components to verify that they are working before they are brought into the top level for system integration.

Hint: When designing lower level components, such as the new multiplexer or the register module, it is useful to first code in Quartus and complete a synthesis check to make sure that what you coded can be made into hardware, then double-check the schematic visually in the RTL Viewer. After the synthesis check, write a testbench and simulate this code to check that it will work correctly. Especially for beginners, it is very tricky to write synthesizable code, so you do synthesis checks first, then simulate. When a designer becomes experienced, they can write the code and check with simulation and simply do synthesis checks periodically to make sure problems didn't inadvertently enter the design.

**3.3.** The system reset must be an asynchronous active-low reset_n as it was for Lab 1. This applies to any new modules you add to the design.

**3.4. Synchronous Design:** The design must be synchronous, meaning that whenever you use a clocked process, it must always be rising_edge(clk). Not falling_edge(clk) nor rising_edge(some_other_signal), etc. Further to synchronous design, all asynchronous inputs to the design must be synchronized (already covered by the synchronizer and debounce circuits in the next two requirements). No Latches are permitted to be used in the design.

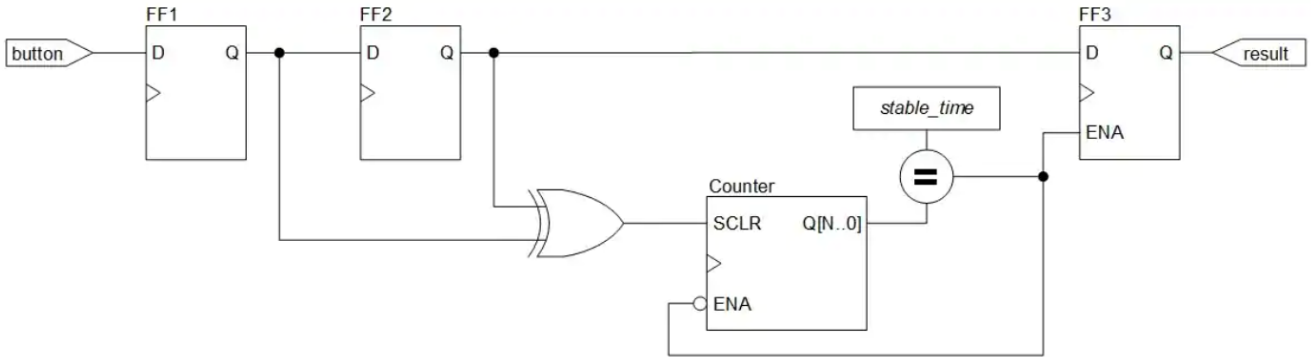
**3.5.** The slide switch inputs (i.e. SW(9 downto 0)) must be synchronized with the 2 Flip Flop synchronizer (see Lecture 8). Synchronization allows asynchronous signals to be made safe within the design, i.e. so that the internal Flop Flops' timing is not violated. With the example figure of the 2 Flip Flop synchronizer below, the A input would connect to the SW input (one for each signal, so 10 in total), G would connect to whenever that SW input would go in the design, and CLK would connect to the system clk signal.



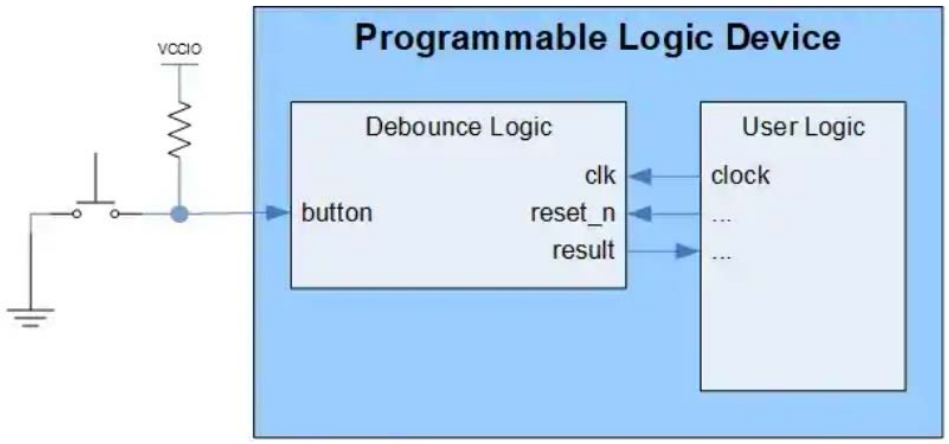

Hint: Basically insert the synchronizer at the SW inputs. You may consider using a register block for the entire synchronizer for all SW inputs or individual Flip Flops for each individual SW signal, this is your choice as a designer.

**3.6.** The remaining Pushbutton (the other one is used for reset_n), must be debounced using the debounce.vhd module provided to you. Modify this debounce code to provide protection against a bounce time of 30 ms. Be prepared to explain your calculation. You probably will find it useful to use this Pushbutton input in this design project (DO NOT debounce the reset_n signal). This debounce code has been slightly modified by the instructor but the original source and explanation is here: https://www.digikey.com/eewiki/pages/viewpage.action?pageId=4980758#DebounceLogicCircuit(withVHDLexample)-ExampleVHDLCode

**debounce.vhd RTL schematic**




**How to use debounce.vhd in your design**




You will also be provided the instructor's testbench for debounce module, tb_debounce.vhd. This testbench is an example of a self-checking testbench and is provided "as is." You may need to modify this testbench code to suit your purpose (change the bounce time? change the clock frequency?), but it should help you understand the functionality of the debounce module.

## 4. Resources Provided

- DE10-Lite kit.
- Reference code that demonstrates: debouncing a switch/pushbutton input;
- Lab and lecture videos (especially Lecture 8 for register and synchronizer design).

## 5. Deliverables

The team will upload the following to the D2L Dropbox by **11:59 pm Sunday October 25**.

1. VHDL code for their project:
    a. RTL code, i.e. the design code, for the complete project.
    b. The testbenches for all modules within the top_level (except BCD and 7-segment displays), i.e.:
        i. Updated multiplexer
        ii. Register for holding the stored value
        iii. 2-Flip Flop synchronizer
        iv. Debounce circuit
        v. Any other modules you created, if any
    c. A testbench for the top_level, demonstrating the system operating in all 4 modes: Hexadecimal, Decimal, Stored, Hard-coded, and showing Reset behavior. Also showing debounce behavior.
2. The .sdc and .qsf files for their project.
3. A Design Record which is a PDF document that contains the following screenshots. Please refer to the Lab 1 Design Record Example document to see the required format and instructions on how to obtain the screenshots.
    a. Your RTL schematic (after the required changes have been completed).
    b. Your Slow 1200mV 85C Model Fmax Summary – we want to see the maximum frequency of your design.
    c. Your Messages Window.
    d. ModelSim simulation waveforms for top_level, multiplexer, registers, debounce, and synchronizer.
    e. Note, you do not have to show the useful instructions on how to obtain the screenshots, as shown in the example document. You just have to show the screenshots, the document title, and the student names.
4. Three videos of **no more than 3 minutes each**, delivered by the respective leads. All students in a team must present at least one video. You must feature your face in the video, as you are speaking. The videos must be recorded in "one-take." No video editing, splicing, cutting, speeding up or slowing down, etc. The videos must be uploaded to the D2L Dropbox in MP4 format. You must follow the Lab Submission Procedure to D2L Dropbox, provided as a PDF and a video to explain the instructions, in the Labs folder in D2L. This will allow the video to be played within the D2L Dropbox player and will make it more convenient for the marker and for you. *Keep in mind that the marker is someone who knows VHDL and FPGA design, is familiar with the requirements of the project, and is familiar with a reference solution. This should help you be efficient with your presentation because you won't have to explain simple things like what is a signal assignment etc.*

a. Design Lead Video:
    i. Use Quartus and your face must be present throughout the presentation. You can start a Zoom session and share the screen with yourself and this should make your face visible in a small picture.
    ii. Introduce yourself and your role.
    iii. Using Quartus: Explain the design and how it works, starting with an overview of the RTL schematic and dive into the design modules as needed, especially how you achieved the additional functionality required by this lab project. Then explain the RTL code, starting with top_level and explain the lower level modules as needed. Be sure to explain your modification of debounce to meet the required protection from Pushbutton bounce time.
b. Simulation Lead Video:
    i. Use ModelSim and your face must be present throughout the presentation. You can start a Zoom session and share the screen with yourself and this should make your face visible in a small picture.
    ii. Introduce yourself and your role.
    iii. Using ModelSim: Explain the testbench code, starting with tb_top_level and show and explain its simulation waveforms in ModelSim. Then repeat this explanation for any other lower level testbenches, if not adequately covered by the top_level testbench simulation discussion. Be sure to explain how the testbench exercises the design and how the simulation displays the correct functionality of the design (this is the purpose of the testbench).
c. Implementation Lead Video:
    i. Introduce yourself and your role, while looking into the camera, so that your face is visible.
    ii. Focus the camera on your DE10-Lite board and demonstrate the required functionality of the system, while explaining what you are doing and the results you are seeing and how it confirms the required behavior: the system operating in all 4 modes:
        1. Hexadecimal;
        2. Decimal;
        3. Stored;
        4. Hard-coded; and
        - Debounce behavior; and
        - Reset behavior.

# 6. Grading Rubric

Lab 2 will be weighted at 15% of the final course grade. The grades of the 3 presentations will be averaged with equal weighting for all the students in the team. Example:

| | |
|---|---|
| Design Lead Video: | A (4.0) |
| Simulation Lead Video: | B+ (3.3) |
| Implementation Lead Video: | B+ (3.3) |
| Grade assigned to all students of the team: | 3.53 (10.6/3 = 3.5333… rounded to two decimal places) |

The Design Record will be used primarily as a reference but a poor-quality Design Record will detract from the presentation grades.

## 6.1. Presentation Rubric

The video presentations will be assessed on a grade-point scale for a single grade out of 4.00. The grade will be interpreted from the description below.

| Grade | Description |
|---|---|
| 4.00 (A) | Excellent: superior performance, showing comprehensive understanding of subject matter. *To earn an "A" the submitted work must fully complete and fully meets the project requirements. The work is of such high quality that it could serve as an exemplar for the project from both the technical and communication perspectives.* |
| 3.70 (A-) | Excellent: superior performance, showing comprehensive understanding of subject matter. *The submitted work is fully complete and meets the project requirements. The work is very high quality from both the technical and communication perspectives.* |
| 2.70 to 3.30 (B- to B+) | Good: clearly above average performance with knowledge of subject matter generally complete. *The submitted work is fully complete and meets the project requirements. The work is very high quality but has minor weakness or weaknesses either technically and/or in communications.* |
| 1.70 to 2.30 (C- to C+) | Satisfactory: basic understanding of the subject matter. *Submitted work is complete but has significant weakness or weaknesses either technically and/or in communications. Generally indicates insufficient effort or accomplishment is moderate.* |
| 1.00 to 1.30 (D to D+) | Minimal pass: marginal performance; generally insufficient preparation for subsequent labs or courses in the same subject. *Weak effort demonstrated by missing elements or superficially completed elements either technically and/or in communications.* |
| 0.00 (F) | Fail: unsatisfactory performance or failure to meet course requirements. *Clear lack of effort or ability to accomplish the project requirements.* |

Notes: Grade can be reduced down to 0.0 for not complying with requirements or for unprofessional behavior. Excessive video lengths will be penalized one letter grade (e.g. B+ to B) for every 15 second interval overlength of the stated limit.

Late penalty: one letter grade per day late (e.g. A- to B+, not A- to B-), according to the D2L Dropbox timestamp, based on the latest submission for the project. Students are responsible for retaining the D2L Dropbox submission confirmation email. Submissions must be fully complete to be eligible for grading of the lab project. This includes code and design files, videos, and Design Record, otherwise the late penalty will apply to the entire lab project.