Lab #6: State machine and stopwatch

Physics 127BL   Winter 2024

Lab report due **Thursday, February 22, at 11:55 P.M.**

Please read the lab report and homework guidelines handout on the course web page.

Introduction

In this lab, we will build a digital debouncing circuit that will produce a single clock cycle pulse for each flip of a slide switch on the Altera board. This handy circuit will be used in future labs where single button presses or switch flips may be used to control logic. This design will be the first time that you will build a *state machine*, a type of circuit we will use often in the rest of the course. In addition, you will learn how Quartus makes modules out of your code so you can make more flexible and modular designs.

Using what we have learned, we will then build a digital stopwatch. This exercise will show you how to design a circuit from scratch. We will also see how to use libraries in the Quartus software when we incorporate the switch debouncer from the first part of the lab.

1 Slide switch counter

Here, we will use a counter to measure the bounce coming from the slide switches. On our FPGA boards, the push buttons (`KEY[3..0]`) have already been debounced, and will be the preferred way to control logic in general. However, the slide switches are not debounced internally, so we will use them to observe "bouncing" behavior and create our own *debouncer*.

1. Load the project from `lab6_part1`. In this directory you will see the slide switch, push button, and clock inputs, as well as the 7-segment outputs.

   Design a circuit to: a) synchronize the switch logic to the clock, b) produce a single pulse for each rising edge of the slide switch signal (using the *edge detector* from Homework #3), c) count the pulses, and d) display the count.

2. By testing the circuit with many flips of the switch, make a rough estimate of the probability and number of bounces coming from your switch. You may find that the speed at which you move the switch contributes to the probability.

2 Debouncer Circuit

We will now design a circuit to debounce the push-button input, using the assumption that after a transition the logic level should stay at that transition value for at least a fixed amount of time.

1. Design a *state machine* for this task, following the relevant discussion in the textbook.

2. Much of the code for the state machine has been entered in the text module `seqdebounce` in the `lab6_part2` directory. Open the module and edit the truth table to complete the state machine. Text entry with a truth table has been chosen for defining the state machine because this is the simplest method to enter all the possible combinations for the flow of states. Please note how the inputs and outputs are defined, as well as the state machine variables. In the future, you will be designing state machines from scratch.

3. For the purposes of our class, our state machines will be defined ("encoded") according to your design. Although it's possible for Quartus to automatically encode them, we want to make sure this feature is disabled in the software so as to not override your design. Select `Assignment→Settings→Compiler Settings→Advanced Settings (Synthesis)`, then make sure the State Machine Processing is set to "UserEncoded" and *not* to "Auto".

4. Test the debouncing circuit for several time delays, spanning about 2^{10} to 2^{20} clock cycles. At what time delay do you no longer see switch bounce? This delay is a few times larger than the time scale for the bounce events. Note: you can use a time delay up to about 0.1 s before affecting the functionality of the switch, since it would be hard to flip more than a few times per second.

3 Modular Design

We will now redesign the circuit from part 2 above with the debouncing function and edge detector being modules.

1. Copy your working project `lab6_part2` to a directory named `lab6_new` and load the project from this directory.

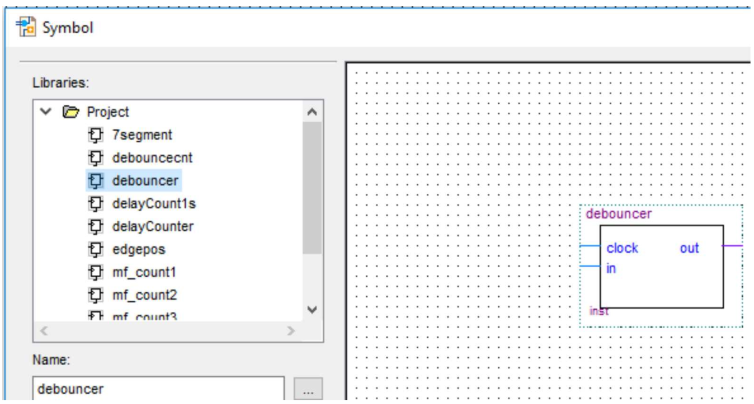
2. Select `File→New` and choose to make a "Block Diagram/Schematic file". This creates a new schematic diagram module, which you should rename "debouncer" using the `File→Save As` menu entry.

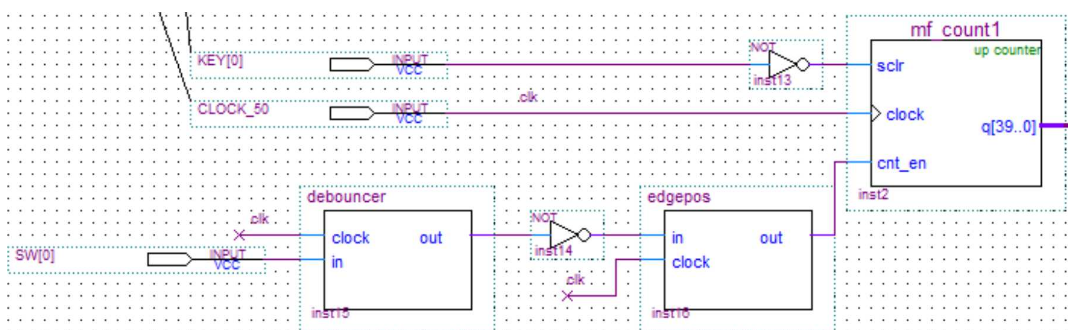
3. Using `File→Open`, go back to your top level "Counter" module and select (via left click-drag) the part of the circuit that functions as the debouncer. Right click on a part and select "Copy". Change screens to the "debouncer" module, then right click "Paste". Using the pin tool, create input pins "clock" and "in", and an output pin "out". Connect these pins to the appropriate wires.

4. Now we will create a block diagram symbol that you will be inserting later in your top-level design. First save the file with `File→Save`. Then choose `File→Create/Update→Create Symbol Files for Current File` and the design will be precompiled and checked for errors. If compilation errors occur, you must fix them at this time.

5. Change screens to the top-level "Counter" module, and insert your "debouncer" module. This is done with the symbol tool, choosing the file `Project`→`debouncer` (see the figures below). Insert this into your design and wire the inputs and outputs. Note that if you want to debounce more than one switch, it is easy to do so by simply copying and pasting this module multiple times.

6. Repeat the above procedure to create a positive-edge detector module "edgepos".



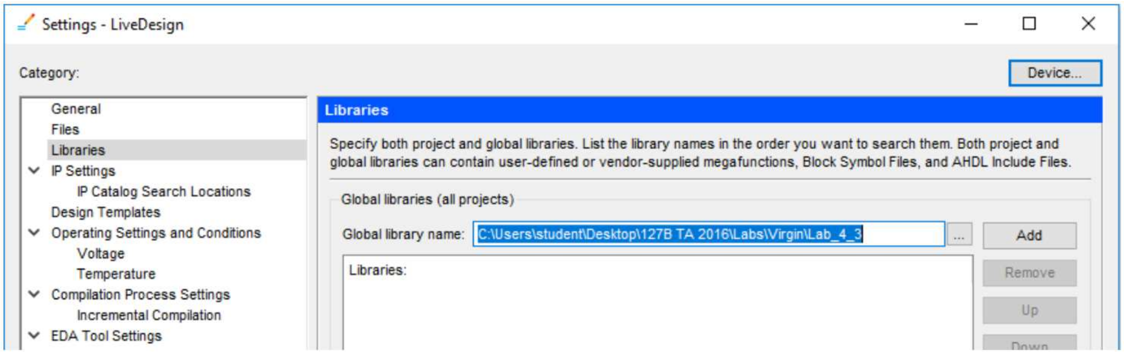


7. Compile and test for proper functioning of the modules. In the next part we will learn how to import these modules as libraries into other designs.

4 Stopwatch

1. Load the project from `lab6_part3`. This directory has a top level entity named "TopLevel" that is entirely blank. You will need to create input and output pins for the stopwatch using the names of pins declared in previous projects. The 7-segment converter module is also in this directory. You can cut and paste code from previous projects into "TopLevel".

2. The module for debouncing a switch and making a positive-edge pulse can be taken from your last lab by using it as a library: select `Assignments`→`Settings`, then click `Libraries` under the left hand column of user settings (see the figure below). Enter the path to your library `lab6_new` by using the browser ("…" symbol). Don't forget to add this path to your libraries by clicking "Add". The path will then be displayed in the "Libraries:" box. The library modules can be accessed in the standard way with the symbol tool (`AND` gate symbol).



3. Design the stopwatch to display tenths of seconds, seconds, minutes, and hours, with the gaps between `HEX` displays splitting the time units. For example, hours on `HEX[7..6]`, minutes on `HEX[5..4]`, etc. Have the stopwatch cycle through 3 states of a state machine with a push-button. The 3 states are counter-clear, counter-start, and counter-stop.

4. How is it possible to quickly simulate the stopwatch, or test the "hours" counter?