

Sequential Design: A Digital Lock

This is the first sequential design assignment. The task is to design and implement an electronic lock for a door. The input combination is a sequence of three 4-bit digits.

You have four switches and a pushbutton. Set the switches to the first number of the combination, press the button, change the switches to the second number, press the button, etc. If you successfully enter the appropriate three numbers, a signal goes high to unlock the door (in your case, to light one of the LEDs). The correct combination can be encoded as part of the design (or entered separately – see options below).

You will also want a signal to reset the lock. A second button can generate this signal. If the reset signal goes high, the door locks. I'll leave some buttons and resistors for you to use. An alternative to a reset switch is to *time out* after a given period of time has passed.

Your assignment is as follows.

1. This specification was vague; such as you might get from a client. Formalize the specification with truth tables or logic equations. Justify any design decisions that you make.
2. Code the design in Verilog and simulate it. Give a justification for your choice of test sequence. Why does it do a good job of testing the design?
3. Download the design onto the Xess board. Demonstrate your working design to the TA.

Email the following

1. A design document, with the design decisions you have made and justification for those.
2. The project itself as a zip file
3. Simulation results

Optional

- Add a door-closing timeout. After the door opens, start counting and when a set time has elapsed, close the door.
- Add an idle timeout circuit. If the user has begun entering a combination but let too much time pass, reset the combination sequence.
- Allow the user to enter a combination instead of hard-coding it in the Verilog. You might do this with another pushbutton to signal that this is a new combination.

More Details

For this assignment you'll need a clock running at a modest frequency, but we may not have discussed counters by lab time. You'll need to divide the 100MHz clock provided on the XSA-100 board down to some frequency more useful for this task. We'll provide some Verilog to generate a slower clock.

You can use the pushbutton on the board but you'll need some logic to synchronize it with the clock and debounce it. We'll talk about this in class.

The XSA-100 board clock is on pin 88 of the FPGA.

Other useful pin information (from the schematic diagram):

Seven-Segment Display

Top, s[6]	P49
Upper Left, s[5]	P57
Upper Right, s[4]	P46
Middle, s[3]	P60
Lower Left, s[2]	P62
Lower Right, s[1]	P39
Bottom, s[0]	P67

Pushbutton (normally high)	P93
----------------------------	-----

DIP switches (normally high)

1	P54
2	P64
3	P63
4	P56

Authors: Anselmo Lastra and Aleksandra Krstic