University of California at Berkeley
College of Engineering
Department of Electrical Engineering and Computer Sciences

CS 150, Fall 2000

J. Wawrzynek, N. Weaver
Modified by R. Fearing, D. Young, and N. Aboobaker
Xilinx F1.5 Version: T. Smilkstein

Lab 3
Finite State Machine on Xilinx

1 Objective

Put your combination lock from Lab 2 onto a chip. To do this, you will:

1. Use Xilinx Flow Engine to compile and route your design.
2. Use Xilinx Hardware Debugger to do real-time hardware debugging and watch internal signals

1.1 Overview

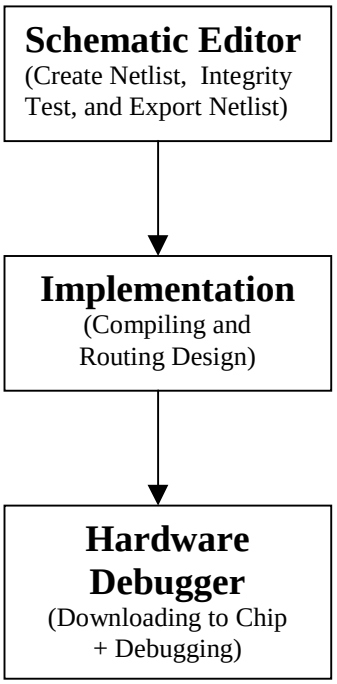

Figure 1. Steps taken in this lab to compile, download, and test design

For a design to be useful, it must make its way into hardware. To do this we will be using Xilinx Field Programmable Gate Arrays (FPGAs), chips that can be configured as many different circuits. Specifically, we will be using the Xilinx XC4000 Design Demonstration Board, which contains an XC4005E FPGA, switches, and light-emitting diodes.

These boards are very expensive and not easily replaced. **Be very careful with these. Make sure that they stay on the anti-static mat. Make sure also that the ground plug is plugged in. Before touching the Xilinx board, ground yourself-- touch any metal you can find or the static mat.**

Do not take the Xilinx boards or anything else from 204B Cory!

2 Prelab

1. Answer the questions on the checkoff sheet.
2. Make sure your Lab 2 design works and the labels in your schematic match the labels shown in the example.
3. Also check that none of your filenames are longer than 8 characters. We're using new software so this might or might not be a problem (it's better to be safe though).
4. Make sure your schematic does not use any components from the built-in library. If you followed the instructions in Lab 2, this should not be a problem.
5. Make sure none of your symbols or components have the same name as components in the (xc4005e) library, such as DFF or CLB. If you did, rename your components.

2.1 Xilinx Interface

To compile your design for the Xilinx, some interface components need to be added. We have done the busy work for you. (See section 3.1 below)

2.1.1 I/O Pads

I/O pads and buffers are special cells in the (xc4005e) library, and connect the Xilinx to the outside world of buttons, LEDs, and other chips:

IPAD	Input Pad
OPAD	Output Pad
IBUF	Input Buffer
OBUF	Output Buffer

Each IPAD and OPAD is connected to a particular pin on the Xilinx, designated by the LOC attribute. For example, in our schematic, the IPAD in the upper-left corner, SW5-1, has the attribute LOC=P27, indicating it connects to pin 27 of the Xilinx.

2.1.2 Debounce

Your lock expects **ENTER** and **RESET** to be high for exactly one clock cycle every time it is pressed. The **DEBOUNCE** circuit, a simple state machine, ensures this.

3 Add the Xilinx Interface Circuits

We have entered the required interface circuits; you need to copy them into your design. You need to open the TA-provided design and save it as the second sheet in your combination lock schematic.

As with one-sheet schematics, nets with the same label on different sheets of the same schematic are connected implicitly.

1. The TA macro library must be added to your library list to make access to some of the symbols possible. Select **File → Project Libraries…** (or ctrl + L), then, if a library called lab3file is in the list on the left, select it and click on "**Add** >>". If it isn't in the list, click on "**Lib Manager…**". Select **Library → Attach…**, and go to **U:\cs150\lab3file\lib**. A library called **lab3file** should appear in the right list window. Press **OK**. The **Project Libraries** window should come back up with the library you just added in the list window on the left. Click on the new library, then click on "**Add**>>" to add it to your project libraries. As a check, make sure it has shown up in your project manager file list.
2. Use **File → Open** to open **U:\cs150\lab3file\lab32.SCH**. Then save it under the name **lab32.SCH** to your own directory using **File → Save as….** (Remember to add to project if "Non-project" is displayed in title bar of schematic).
3. From the schematic editor, run **Options → Create netlist**, **Options → Integrity test**, and **Options → export netlist**.

4 Preparing your Schematic to be downloaded to hardware

The next step after successfully exporting your netlist is preparing it for downloading to the chip.

The Xilinx Flow Engine, which is started by clicking on the Implementation box (located under the Design Entry box), will compile and route the exported (EDIF) netlist into a bit file that can be downloaded to the Xilinx chip.

In the dialog box that comes up after clicking on Implementation, set the Device to 4005EPC84, and Speed to 1. Then run. A new window will pop up showing the progress of the compilation and routing. To look at the results of a compilation, click on the "**Versions**" tab in your project manager, right click on the revision that you need information on, and select "**View Implementation Log**" or "**Invoke Interactive Flow Engine**". Both will show you a log of the compilation. You can also access the log by clicking on the "**Reports**" tab and double clicking on "**Implementation Log File**". Look through the log and see what information it gives you.

There is a tool that let you see graphically the resulting mapping of the logic in the Xilinx chip. Select **Tools → Implementation → EPIC Design Viewer/Editor**. You should take a look, since it's pretty neat to see. By selecting nets in the EPIC list, you can see how the compiler has routed your wires.

5 Download and Test the Circuit

5.1 Connect the power cables and XChecker cables

There should be power supplies at all of the stations. The power supplies have a cable with two plugs, red and black. The black is ground, and the red is +5V.

1. Plug in the power supply.
2. Ground yourself.
3. Connect power to the board. Match black plug to black, red to red.
4. The decimal point on CR3 (the left seven-segment LED) should light, indicating the power is on.

You'll also need to plug in the XChecker cable. It has a rectangular plug connected to a bunch of wires. It only fits in one direction. You shouldn't have to force it. (One of the openings in the plug is filled with epoxy. It corresponds to the area without a pin on the board.) Ask your TA if you have any questions here.

5.2 Download the circuit

Xilinx's Hardware Debugger is a real-time hardware debugging tool. With it, you can step through clock cycles and observe the actual values of internal nodes. If you check the TA provided schematics, you'll notice that it includes a "readback" block. This is used to recover the values of internal nodes. It is a component in the CS150 directory.

To bring up the Hardware debugger , click on the "**Programming**" block in the program manager and select the hardware debugger. If it complains about the cable not being connected correctly, make sure the board has power and the XChecker cable is attached. You can also check the cable configuration by choosing **Cable → Communcations**. The XChecker cable is hooked up to **COM1** and the transfer rate should be 115 Kbps.

Choose **Download->Download Design** or double-click on the name of the **.bit** file in the diagram to download the design to the Xilinx board. In this lab, a zero should appear on the CR4 (the right 7-seg LED), indicating that the machine is in State 0.

5.3 Observe the circuit in operation

The clock signal for the combination lock design comes from the XChecker cable and is fed into pin 8 of the Xilinx chip (see schematic). This is useful for debugging because the software allows us to

manually control the clock. To manipulate the clock, we'll use the Hardware Debugger "**Debug Control Panel**". From the control panel, we can also watch the internal signals and have the waveforms displayed like in ViewTrace.

5.3.1 Turn on the clock

Since the clock is coming from the XChecker cable, the lock will not operate properly until the clock is running.

1. Open the Debug Control Panel by choosing **View->Control Panel** from Hardware Debugger.
2. On the far right, click the circle next to **Start** in the Clock Control section of the control panel. You can also apply single clock pulses by clicking the "Apply" button while Stop is selected.

5.3.2 Test the circuit

While the clock is running, test out your circuit changing the combination and pressing the enter button.

1. Test a successful combination using the buttons and switches on the design demonstration board.
2. Test an unsuccessful combination.

Make sure SW 4-7 (marked RST on the board) is closed. Otherwise, the RESET button won't work.

Have your TA check off your working lock when done with the rest of the lab.

5.3.3 View the internal variables

Open the Debug Control Panel.

1. Click on the Groups button. Combine state2, state1, and state0 into a group called "state". You'll need to click the New button to make a new group, and the Save button for the changes to take effect.
2. Click on the Displays button to display waveforms for state, error, open, and enter. To select "state" for display, select "Groups" from the Display section (at the top of the Display Signals dialog box). To select error, open, and enter for display, "Signals" must be selected. After clicking OK, a waveform window will pop up.
3. Go through both a successful and unsuccessful attempt at opening the lock, taking a snapshot (click the Read button) after each state transition (for this lab, you can leave the clock running while you do this).
4. To read the value of the groups, you can click with the left mouse button on the part of the waveform you want to see. Also, pressing the plus symbol will expand the group waveform into separate single waveforms.

Show your TA the generated waveform for check-off.

Name:_____Name:_____Lab Section:

6 Checkoffs

6.1 Prelab questions (refer to TA schematic)

1. What is ENTER connected to (include direct and indirect sources)? Why?

2. What do you expect to see when the lock is opened?

3. Where does the clock signal come from and where is it output (which LED)?

4. Why are CODE0 and CODE1 not debounced?

6.2 Questions

1. In terms of buttons, switch settings, and lights illumination, give instructions for opening the lock.

2. Explain the three wire-wrap wires on the board. Why are they there? Why does SW4-7 have to be closed?

3. By looking at the report files, how many CLBs does your design use?

1. Working Xilinx Lock	**TA: _____ (40%)**
2. Questions answered	**TA:_____ (20%)**
3. Debugger waveforms	**TA:_____ (40%)**
Turned in on time	**TA:_____ (x100%)**
Turned in 1 wk late	**TA:_____ (x50%)**