

# EECS 373 – Fall 1998

## Lab 6: Introduction to Xilinx FPGAs

### **Introduction:**

FPGAs (Field Programmable Gate Arrays) are high-capacity general-purpose reprogrammable digital logic devices. In this lab, you will design, simulate, and test a logic circuit that uses some of the input/output devices found on the EECS 373 Expansion Board. To provide a gentler introduction to the Xilinx design process, this lab will not involve the MPC823.

### **Goals:**

1. To introduce you to Xilinx FPGAs and the associated software tools used to design, simulate, and implement FPGA-based logic circuits.
2. To familiarize you with some of the hardware found on the EECS 373 Expansion Board.

### **Specifications:**

Your circuit will use the eight DIP switches on the expansion board to input an 8-bit binary value. When you press the pushbutton switch “S1”, the value will be displayed in hexadecimal on the two-digit seven-segment numeric display marked “D2”. The display will be updated only while the pushbutton switch “S1” is pressed; when S1 is not pressed, the display will continue to show the value from the last update. In a similar fashion, when pushbutton switch “S2” is pressed the value of the DIP switches will update the bar-graph display (D1) and one of the two digital-to-analog converters (DAC0). Again, the bar graph and DAC will only be updated when S2 is depressed, and will hold their previous value when S2 is not pressed.

### **Details:**

You will be drawing schematic diagrams which describe your circuit such that it can be implemented on the in the Xilinx FPGA. The “EECS373” library contains two parts you will need: “PROCESSOR\_IO” and “BOARD\_IO”. These parts represent the I/O pins and buffers that you will need to pass signals into and out of the Xilinx chip. The convention used on these components is that signals leaving the chip are on the left-hand-side of the component, and signals entering the chip are on the right.

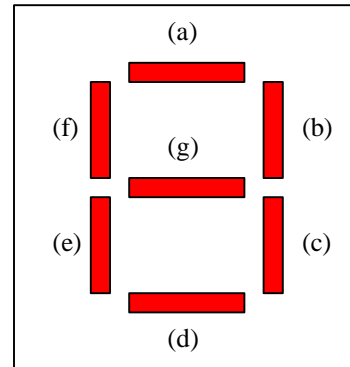
For this lab, the only signal you will need from the PROCESSOR\_IO component is GCLOCK1. This is the system bus clock, which runs at 20 MHz. The other signals discussed below can all be found on the BOARD\_IO component.

**DIP Switches** – The eight DIP switch signals are combined into a bus called DIP\_SW[7:0]. The left-most switch on the board is DIP\_SW7 and the right-most switch is DIP\_SW0. Note that with the Xilinx tools, bus elements are broken out without using brackets or parentheses. When the DIP switch is in the down position (pointing toward the near edge of the board), the corresponding signal is HIGH. Your circuit should be configured so that the switch for the most significant input bit is on the left and, for each switch, up is 1 and down is 0.

**Pushbuttons** – As suggested by the bubbles on the component, pressing a pushbutton produces a LOW signal.

**Seven-Segment Numeric Display** – This is a multiplexed display. Each digit has an independent anode (positive terminal), but shares the seven cathode (negative terminal) connections (one for each segment) with the other digit. To display one digit, simultaneously drive its anode high and some or all of the cathodes low; the segments corresponding to the low cathode signals will

illuminate. The other digit's anode must be low at this point, or the same segments will illuminate on that digit as well. Though you can only drive one digit at a time, if you alternate between the two digits at a sufficiently high frequency (at least 100 Hz) your eye will be unable to detect the flickering and will see both digits as continuously on. (This is basically the same principle that let you control the apparent intensity of the LED in lab 5.) Note that if you alternate too rapidly (say over 1 MHz) the segments will not have enough time to turn on fully and the display will be dim or unreadable. The signal "7SEG\_ANODE1" is the anode for the digit on the left. The seven segment control lines (cathodes) are named "7SEG\_SEG\_A" through "7SEG\_SEG\_G". The accompanying figure shows the locations of the segments.



**Bar-graph Display** – The bar-graph display is a 10-segment LED display (D1). The left-most position is a status indicator. When it glows (weakly) the Xilinx chip has been configured. The second position from the left is unused. The remaining positions are controlled via the LED\_BAR\_OUT[7:0] bus. Of these remaining positions, LED\_BAR\_OUT7 is the left-most, LED\_BAR\_OUT0 is the right-most. These signals are active-low.

### **General Notes On Designing With Xilinx:**

The first steps taken by the Xilinx tools are to convert your schematic to a netlist, and then to convert this netlist to Boolean equations. These equations are then simplified. What this means is that the actual format of your schematic (the gates you choose, the number of levels of logic you use) makes very little difference – the tool-set will always generate internal logic that represents your schematic in its most simplified form. As a result, your circuit schematic should be designed in such a way as to reduce your design time and ease verification. For example, it is completely acceptable to use 4-to-16 decoders, followed by other logic, to do complex conversions.

The Xilinx Libraries Guide can be accessed using the DynaText Browser. From the Start Menu:

**In the lab:**

Start ? Programs ? Xilinx Foundation Series ? Online Books

**On a CAEN machine:**

Start ? Courseware ? EECS373 ? Xilinx Foundation Series ? Online Books

**-- OR --**

Start ? Engineering & Science ? Xilinx Foundation Series ? Online Books

Select "Xilinx Books" in the "Collections" pane. Then double-click on "Library Guide".

The functional selection guide will provide you with the quickest access to the information you need.

It is important that you simulate your design prior to coming to the lab. I recommend that you do your design, then implement it (i.e., generate a downloadable bitstream file) *prior* to doing simulation. By implementing your design first, you know that whatever you have designed is compatible with the Xilinx tools. Once you are assured of this, then check to make sure that the circuit works as you want it to. Otherwise you may spend a lot of time to get a "working" design, only to find that it can't be implemented by the Xilinx tools. Additionally, the Xilinx tools will point out syntactic and conceptual errors by removing circuitry that you thought was correct.

### ***Prelab Questions:***

1. Design a circuit that takes an 8-bit value and drives a multiplexed LED display as described above.
2. Study the data sheet for the DAC0830 (paying close attention to Figure 1). Describe how you would drive the DAC with the output of an 8-bit latch that is internal to the Xilinx. Use the simplest method possible.

### ***Lab Procedure (start in CAEN/finish in the lab):***

1. Start the Xilinx tools by double-clicking on the “Xilinx Foundation” icon on the desktop (or selecting “Xilinx Foundation Project Manager” from the Start Menu.
2. Create a directory in your AFS space for your Xilinx projects.
3. Create a new project (you may need to go to the File menu for this) called “lab6” in the directory you have just created. The remaining fields in this box should be:

Family	XC4000E
Part	XC4010EPG191
Speed	3

4. We now need to add the special EECS373 library to your new project. Under the File menu, select “Project Libraries”. Look for “eecs373” in the list of attached libraries (these are libraries that the software knows how to find). If the library is present in this list, skip down to step 4(b).
  - (a) If the software doesn’t know how to find our library, we have to tell it. Select “Lib Manager” from the Project Libraries dialog. Select Library ? Attach. The EECS373 library can be found in the following directory:  

```
k:\f98\eeecs373\xilinx\
```

After selecting the library, click OK. The library should now appear in the list of attached libraries.
  - (b) Select “eecs373” from the list of attached libraries and click “Add” to make it accessible from within your project.
5. Note the window at the upper-right of the project manager window. This diagram indicates the design flow you will be using: Schematic – Simulation – Implementation. Also note the message window that makes up the lower portion of the project manager. There will be a lot of messages here, but the one’s you need to keep an eye out for are going to be red or blue.
6. Click on the “Schematic Editor” button in the design flow window. Once the schematic editor window opens up, take a few minutes to familiarize yourself with the locations of common functions like “symbols toolbox” (used to place components on the sheet), “draw wires”, “draw busses”, etc. Note that the Undo function is only one operation deep, so you will probably want to save your work often.
7. Select the AND gate symbol on the toolbar on the left. This will open the symbols toolbox. There are two ways to select the library symbol you want:
  - a) In the small box at the bottom of the toolbox, type the name of the component (you do not need to press return). As you type, the list of component names will scroll, and the closest name completion to what you have typed will be selected.
  - b) Use the scroll bars to find the component, then click on the component name.

Once the component you want is highlighted, move the mouse from the toolbox, and you will see that the component is under the mouse, ready to be placed. Any time the toolbox is active, you will see the AND gate-style cursor. When using this cursor, clicking on any component on the sheet will create a new copy for you to move and place on the sheet.

8. Place a few components on the sheet. Now select the “draw wire” tool and make some connections. You can single-click to attach a wire to a component pin; double-clicking will join to a wire already on the sheet.
9. Immediately below the “Hierarchy” menu are three toolbar buttons. The “+|-“ button is actually two buttons used to zoom into and out from the current center of the sheet. The second button will zoom to an area that you define with the mouse. The last button will zoom out until the entire sheet is visible.
10. As you add components to the sheet, you may find yourself running out of room. You can use the File ? Page Setup dialog box to change the sheet size and orientation. An “A” size sheet is 8.5x11 and is the easiest to print, but holds very little circuitry. Start your sheet size no smaller than “B”.
11. Once you have completed your schematic, it is time to convert it to a file suitable for download into the Xilinx chip. Make sure you have saved all the sheets in your design, then minimize the schematic editor and return to the Project Manager. From the Project Manager Flow diagram, select “Implement”. Your schematic has changed since the last time you implemented the design, so click “OK” when prompted to update the netlist.
12. After a little churning, the Design Manager will open. Select the project that you want to implement and press the big black arrow button. Make sure that the correct part is selected (XC4010E-3-PG191) and then click the “Run” button.
13. The Flow Engine will then process your design. There are four major steps involved:
  - 1) Translate – Translates the EDIF-format netlist into the Xilinx equation-style netlist
  - 2) Map – Reduces the equations in the new netlist into their simplest forms. Partitions the reduced netlist into CLB- and IOB-size blocks.
  - 3) Place & Route – Chooses locations for the CLBs and IOBs, then routes the necessary signals between them.
  - 4) Configure – Creates the BIT file that we will be downloading into the chip.
14. Once the flow engine completes it’s processing, it gives you the opportunity to look at the logfile and to look at the reports. The reports are available from the Design Manager, but the logfile is not, so it is best to review it here. The logfile is useful for getting a general idea of how the run progressed, but to track down problems, you will have to look at the individual tool reports.
15. The two most important reports to look at are the Translation Report and the Map Report. The translation report will tell you which signals were not used (there will often be several in this category). Look here for signals that you created. If they show up here, then you probably have a connectivity problem in your schematic.

The Map Report gives a lot of information (note the table of contents), but sections 1, 2, 4, and 5 are the most important to us. Look in section 1 to get general information about the design. Check section 2 for warnings (ignore the warning about slew-rate-limited outputs). Section 4 tells you about the logic that the tool decided wasn’t really necessary. Look here for your signal names (sometimes they are not really removed, just merged with other logic). This section, with section 5, gives detailed information about what was removed and why. You can use this as a guide to correcting non-syntactic errors in the schematic.
16. If all went well, there should be a file named <project-name>.BIT in the project directory. Minimize the Xilinx tools and start the “Xilinx Download” Hyperterminal Session. Apply power to the hardware and you should see the following message in the terminal window:

```
Xilinx Bitstream Downloader
Beta Version 0.2
Paul Stoffregen, March 1996
```

17. Select “Send Text File” from the Transfer menu (Alt-T, T). Navigate to where your BIT file is located (you may want to select “All Files” for file type in this dialog), select the file and click OK to begin the transfer. After a successful transfer, your window should look like the following figure.

```
XC4000 Series Bitstream
INIT* signal went high.
.....
.....

DONE signal went high after 0x56FC bytes.
Complete startup sequence.

Beta Version 0.2
Paul Stoffregen, March 1996
```

Additionally, when the configuration process completes successfully, the left-most LED of the bar-graph display will glow (dimly).

18. Test the seven-segment and bar-graph display functions.
19. Hook an oscilloscope up to J5 of the Expansion board and test the DAC output function. Measure the output voltage for input values of 0, 16, 32, 48, ... 240, 255. If the deltas between output voltages are significantly larger for some of the intervals, measure some intermediate input values as well.

### **Lab Report:**

1. Briefly summarize the operation of your circuit and how you arrived at your design. Be sure to include a description of the DAC control circuit.
2. Include a printout of your final schematic.
3. Discuss any difficulties you had in designing, implementing, or debugging the circuit.
4. How many CLBs and IOBs did your design require? What fraction of the total number of available CLBs and IOBs does this represent, respectively?
5. What was the longest signal delay in your design?
6. Plot the input value vs. the DAC output voltage. How would you characterize the output of the DAC?

### **Hints:**

- 1) Be sure to name ALL of your signals.
- 2) To debug your actual Xilinx implementation, you can use the bus port labeled “IC\_OUT[31:17]” to drive internal signals onto the test-point connector (marked TP1). From there you can observe them using the oscilloscope or the logic analyzer.
- 3) A comprehensive tutorial for the Xilinx tools can be found at:  
<http://www.ee.upenn.edu/rca/software/xilinx/foundation/foundation.sch1.html>
- 5) You can get help with common mistakes using the tools from:  
<http://www.ee.upenn.edu/rca/software/xilinx/foundation/commistakes.html>