39th ASEE Midwest Section Meeting

A Modular and Specifications Oriented Digital Circuit Design Laboratory

Jason Moore, Mitchell A. Thornton, Ronald W. Skeith
Southern Methodist University / University of Arkansas
Dallas, Texas / Fayetteville, Arkansas

Abstract:

A laboratory for a second undergraduate course in digital logic design is described based on the philosophy of decomposing the circuits into control and datapath portions. Modular designs are strongly encouraged by requiring later designs to need some of the same pieces of earlier designs. The unique aspect of this laboratory is that students are required to generate designs that meet criteria other than correct functionality; both performance and area are specified. This gives the students a good idea of what life as a circuit designer might be like. They are faced with the real-world problem of speed versus size. This often leads to friendly competitions between students on whose design is smaller or faster. The specifications and difficulty of the assignments force students to use proper design techniques. The remainder of the paper will focus on the experiences gained both while teaching and taking the laboratory.

Introduction:

This paper describes a Digital Circuit Design Lab that focuses on modular design and is specification oriented. The lab is a companion course to the Digital Logic course described in [2]. Students are taught to break their designs into modules in a hierarchical fashion starting with initially breaking the entire circuit into a control unit and a data path. Then, students are encouraged to further divide the circuit as they see fit. Students are also given a glimpse into what life as a circuit designer might be like. They are given specifications and a due date for their assignment. The lab instructor strongly suggests that they set a schedule to meet certain milestones and to start early.

Students learn how to use datapath diagrams. A datapath diagram is a block diagram that shows the movement of data through the circuit without worrying about the control logic of the circuit. Below is an example of a datapath block diagram.

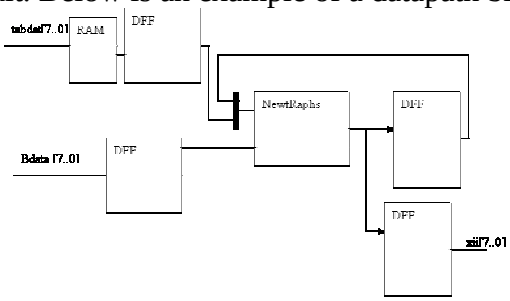
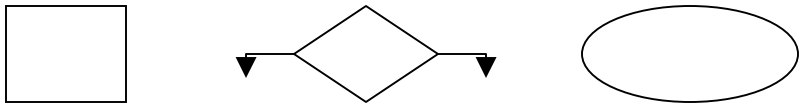


Figure 1: Example of Datapath block diagram

In addition to datapath block diagrams, students also utilize Algorithm State Machine (ASM) charts for the control logic of the circuit. ASM charts are very similar to flow charts used for software design. The three basic components that make up a ASM chart are listed below [1]:



State box **Decision box** **Conditional output box**

When combined appropriately, the above components can create an ASM chart that completely describes the control logic. Although ASM charts are like fingerprints in the sense that no persons' diagram is exactly the same as someone elses', below is an ASM chart that describes the control logic for the datapath shown in the datapath block diagram in Figure 1.

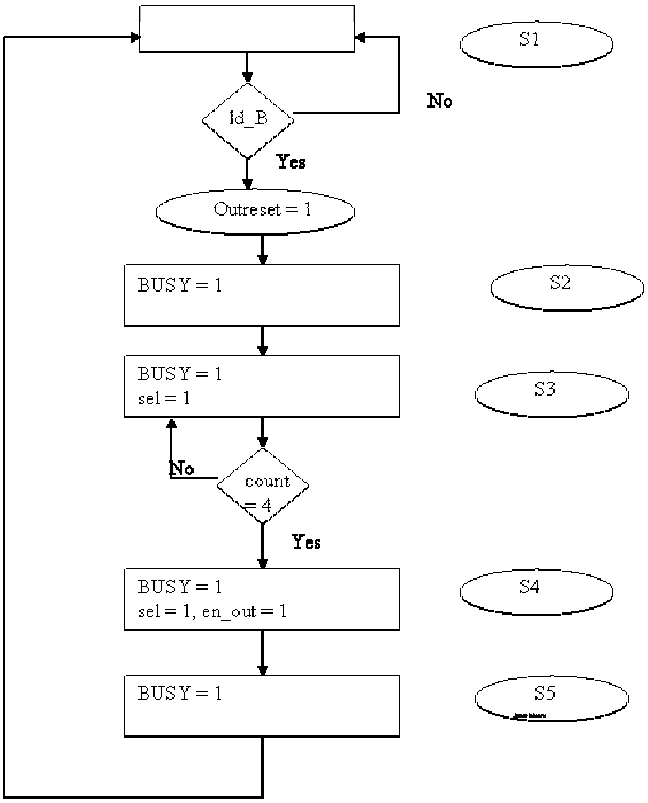


Figure 2: Example of an ASM chart

The remainder of this paper focuses on the method of design taught in the lab and how the lab assignments build on each other not only to give students the experience using design software, but to also teach them how to design in a modular manner and to reuse those modules. Also included in this paper is a course outline for the lab.

Method of design:

The design method taught in the lab is that of top-down, hierarchical modular design. Once the circuit has been divided into a data path and a control unit, students are encouraged to divide the data path into smaller data paths. The control unit is usually coded in VHDL while the data path is generally a combination of schematic capture and VHDL code.

Before starting the design-entry program, students should have already drawn out his or her datapath block diagram. Students should design, build, and test the smaller data paths before using them to build the overall datapath. Since, if the smaller data paths work, then the only thing the student has to worry about is the glue logic. The datapath should definitely be done before the control unit because there is no sense in building a control unit for a data path does not work. The control unit can be simulated directly through the use of the waveform editor. The signals produced by the control unit can be treated as inputs to the datapath circuit and appropriate test waveforms can be generated. This approach allows the student to test their datapath before the design of the control unit of the circuit has occurred.

After designing and testing the datapath, students should start drawing the ASM chart for the controller circuit for his or her datapath. If the ASM chart is drawn properly, the actual coding of the control unit should be very easy for the student. However, a poorly done ASM chart or no ASM chart causes the task of creating a control unit fairly difficult. Many students when given task of designing the control logic for the first time will not draw the ASM chart until after they start trying to write the VHDL code for it. Many of these students end up taking a step back and drawing the ASM chart and starting over with the code while the remainder of the students who have not drawn their ASM chart continue struggling through the VHDL code without an ASM chart. This situation usually does not occur more than once with the student still being able to meet the deadline. A very important issue to point out to students when they are designing their ASM charts is that the ASM chart is not carved in stone but is intended as a guide. Now that the student has used his or her ASM chart to design the control unit of the circuit, it can be tested by comparing the output of the control unit with the signals that were used to simulate the control unit when testing the data path.

Once the datapath and control unit have been tested, all that is left is to put them together. In theory, this task should be as easy as creating symbols for both and connecting the right outputs to the right inputs. However, students sometimes learn that everything is not as easy as it seems in theory. Sometimes, connecting the data path to the control unit requires a small amount of logic between them i.e. an OR gate, AND gate, …, etc. This experience provides the student with an idea of the difficulty that is often encountered in system integration.

Once the circuit is completed, the top level of the schematic should look something like the example below although some students will choose to have multiple smaller data paths and/or control units in the top level schematic.

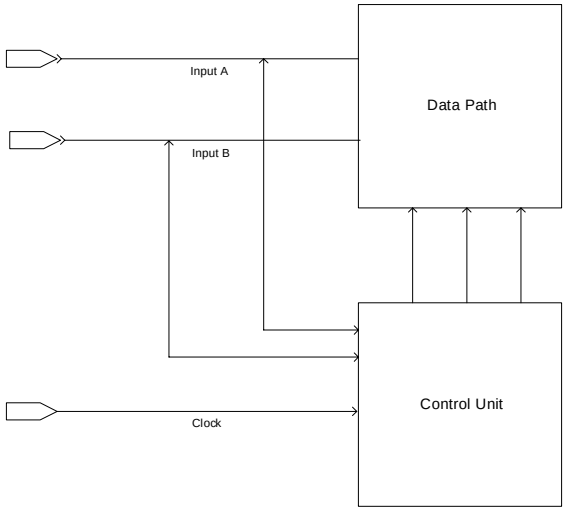


Figure 3: Top level view

Building from one lab to the next:

In laboratory experiments 0, 1, and 3 students become familiar with the design tools. Experiment 0 is quite simply a tutorial out of the back of student's textbooks [1], and there is no design work required on their part. Experiment 1 is an exercise designed to teach the students how to use the schematic capture tool including such things as the different synthesis option available. Experiment 3 teaches them how to use the LPM modules. This is very important since we emphasize design reuse and hierarchy.

In experiment 2 students are given the schematic below and are asked to implement the 2×8 multiplier in one VHDL entity. The students are also given a waveform to test their circuitry and a "golden" waveform to compare their output with.

The students are now familiar with the 2×8 multiplier circuit when in lab 6 they are asked to modify the above schematic to run at 70 Megahertz or greater. Bonus points are offered in 5-point increments for reaching the following plateaus 90, 100, and 110 Megahertz. Additional bonus points are also offered if the student can modify the above circuit to be an 8×8 multiplier. In addition to allowing students to make up for some points lost, the bonus points serve to teach two very important lessons. The first is that you should first meet the specifications before trying to improve on them. The second lesson is a little less obvious since it is a lesson in cost and rewards. Gaining the 5 bonus points from increasing the speed from 100MHz to 110MHz is much more difficult than gaining the 10 points from designing and pipelining the 8×8 multiplier.

The last bonus part becomes particularly important for experiment 8 since the student must implement a Newton-Raphson inverter using the 8×8 multiplier described in experiment 6. In experiment 8, the students are also given an explanation of the theory of the Newton-Raphson inversion approximation and are assigned to work out some examples by hand. Experiment 8 is broken into three parts. The first part is to build the Newton-Raphson circuit using LPM-multipliers. The second part is to replace the LPM-multiplies with the 8×8 multiplier that he or she designed. The final part is to pipeline the circuit to try to achieve maximum speed yet still fit on the device given.

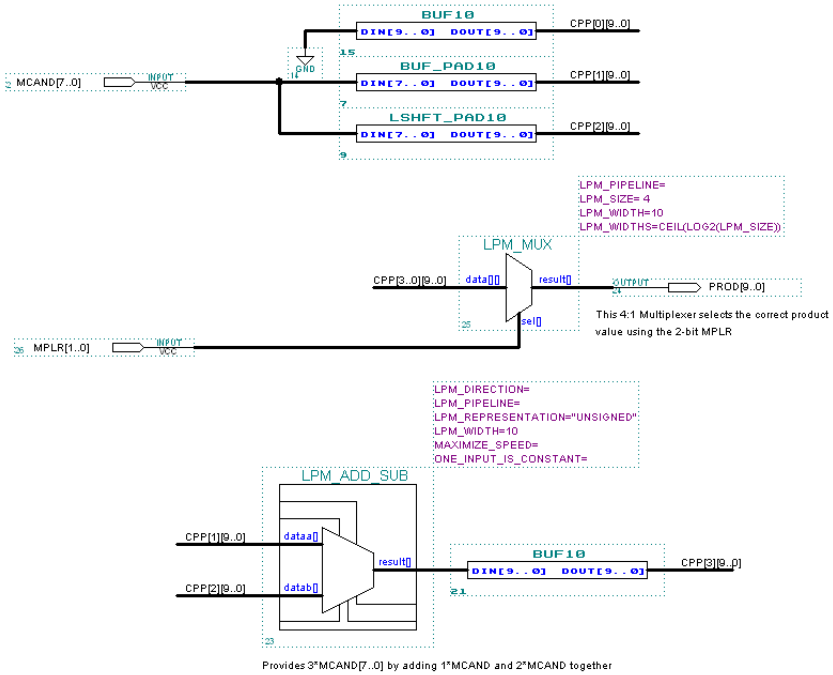


Figure 4: 2×8 Multiplier

Course Outline:

This section of the paper is meant to briefly describe what each lab assignment is meant to accomplish. Since the skills taught are more important than the actual assignments, we will just provide the skill gained from each lab assignment.

Lab 0: Altera Max Plus 2 tutorial and installation guide.
Lab 1: Introduction to schematic capture and simulation.
Lab 2: Design of 2X8 multiplier circuit that will be used in later lab assignments.
Lab 3: Introduction to LPM modules and data paths
Lab 4: Using LPM modules in VHDL
Lab 5: Explore timing issues such as register to register and clock to delay. It also serves as an introduction to circuit pipelining.
Lab 6: Forces the use of pipelining in order to meet the specifications
Lab 7: Sequential logic in VHDL
Lab 8: Introduces the split design of data path and synchronous control unit
Lab 9: Uses previous lab assignments as building blocks to build a more complicated circuit.

Conclusion:

The lab is designed as a companion design experience that accompanies the lecture portion of the class not only by implementing subjects talked about in class, but also by encouraging students to design circuits using methods taught in class. This is accomplished by having the lab assignments build on top of each other by using a previous lab as part of the datapath of a later lab which also allows for further testing of the previous labs since sometimes problems occur in the circuit when used as part of a

larger circuit that do not get exposed by the test vectors used initially. As the semester goes along, students begin to design in a modular way on their own and test their modules before inserting them into the overall data path.

References:

[1] Stephen Brown and Zvonko Vranesic, *Fundamentals of Digital Logic with VHDL Design*, McGraw-Hill Higher Education, Boston, 2000 pp. 504-505.

[2] Mitchell A. Thornton and Aaron S. Collins, "A Second Undergraduate Course in Digital Logic Design: The Datapath+Controller-Based Approach", ASEE Southeastern Section 2003.