Integrating Printed Circuit Boards into a Computer Architecture Course

Abstract

This paper describes the integration of printed circuit board design and fabrication into a computer architecture course. These printed circuit boards utilized 7400-series chips to realize an 8-bit ALU having the functions ADD, OR, ADD, SUB, and set-less-than. Students designed and laid out the boards using electronic tools. Their designs were sent out for fabrication and the resulting boards were populated and tested. This paper presents the methods, integration, and assessment of this effort for student learning.

Introduction

The standard computer architecture course in electrical and computer engineering curricula typically focuses on the logical structure building an ALU and datapath. Laboratory exercises to supplement lecture material often focus on the realization of a part or all of the datapath using electronic capture tools such as Mentor Graphics, or the implementation of the design in a hardware description language such as VHDL. Aggressive, yet often failed attempts are occasionally made to physically fabricate components such as an ALU on a breadboard using 7400-series chips and hook-up wire. However, the work involved is an enormous burden on the student, and potential errors make debugging a nightmare. In this paper, I will give my experiences in physically fabricating an ALU using capture tools, printed circuit board (PCB) layout, PCB fabrication, board population, and testing. This exercise was integrated into the junior-senior level elective of computer architecture.

The course in which this development took place is an upper-level junior/senior elective entitled "Computer Architecture." The course consists of three 50-minute lectures per week and students receive three credits. The textbook for the course is the canonical Patterson and Hennesey text[1] and focuses on the structure of the MIPS CPU. The major topics covered include the instruction set architecture of the MIPS, ALU structure, a single-cycle, multi-cycle, and pipelined 32-bit datapath, and memory hierarchy.

The goal of the PCB exercise was two-fold. One, we wished for our students to have proficiency in designing, laying out, and populating printed circuit boards. The second goal was to use this exercise to increase student comprehension of the ALU presented in the text.

Design and Fabrication

The design for the ALU presented in Patterson and Hennessy utilized a modular "bit-slice" structure. A one-bit circuit calculated all functions of each bit in parallel and a resulting 4:1 MUX selected the correct result given the operational code. For addition and subtraction, a carry-out and carry-in bit were used to construct a ripple-adder.

Since it was desired to have the students learn this design structure, the PCB design paralleled that of the text. Students first assembled a full-adder circuit, then used this in a hierarchical block to form a bit slice. An example bit-slice is shown in Figure 1.

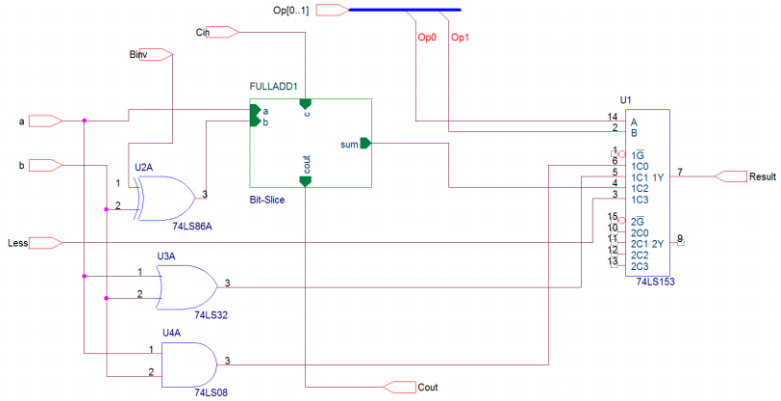

Figure 1 Schematic for standard bit slice, based on the text.

As in the text, these bit-slices were used to form an 8-bit ALU by cascading slices together. Students were then asked to simulate their designs in PSpice and verify the designs.

Modifications had to be made to this basic design to realize the circuit on a PCB using 7400-series chips. One modification was that the 4:1 MUX units are available only in pairs with a single select line via the 74153. In the basic design, students used one chip per bit-slice, essentially wasting the second MUX on the chip. In their optimized design, students re-created either 2- or 4-bit slices to utilize fully each 74153.

A second optimization was in the full-adders. Students' designs for the full-adders went down to the AND/OR/NOT gate level. This requires 7 gates for each full-adder for a total of 56 gates just for the adders. Instead, students re-designed their circuits to use a 4-bit adder, the 7483. This saved approximately 8 chips per board, but the resulting design was less elegant than the text's presentation. Students utilizing the 7483 thus typically created a 4-bit-slice and used two to complete their design. A representative 4-bit-slice is shown in Figure 2.

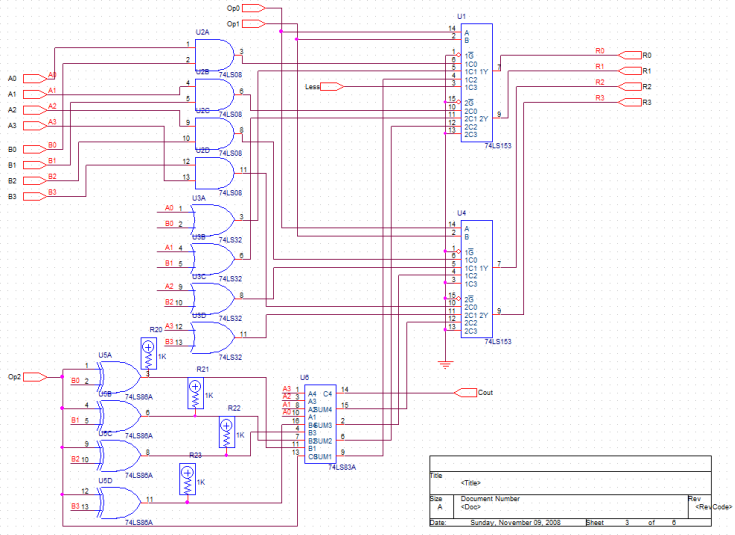

Figure 2 Optimization of design using the 7483 and implementing a 4-bit slice.

The final complication for physical realization of the board was the need for input switches, LED indicators, and a power connector. Students added LED indicators for the two 8-bit inputs, the 8-bit output, and the Carry, Zero, and Overflow flags. These were coupled with the gate design by using off-page connectors. Further, certain gates require pull-up resistors, and students were instructed to add these as well. Schematics for switches and LED's are shown in Figure 3.

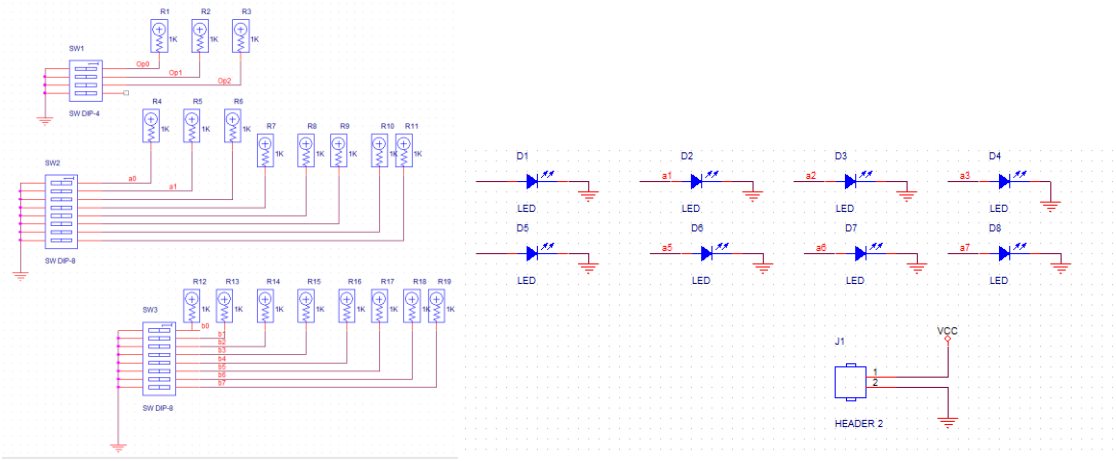

Figure 3 Additional components for the physical realization of the circuit.

When students had completed their schematic circuits, their designs were then exported to OrCAD PCB designer. Students placed components manually, but used the auto-route function of the software to run all traces. Students used traces twice as thick for the power and ground nets, and increased the default spacing for components, traces, pins and vias. Students generated the artwork for their board and sent it to the professor for upload to Advanced Circuits, the PCB fabricator. Turnaround was approximately five days. A student's completed PCB layout is shown in Figure 4.

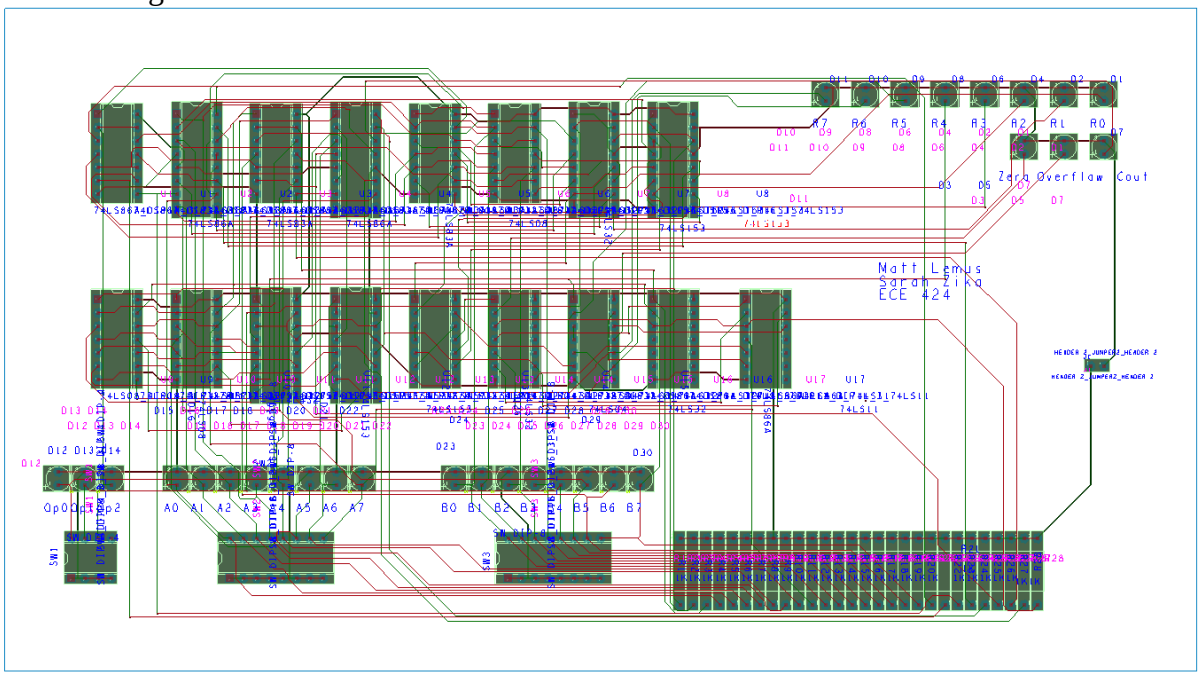

Figure 4 Final layout of ALU for PCB fabrication.

When the boards were received from the manufacturer, each student populated the boards with components by soldering. They then tested the functionality of each operation given several different input combinations. Students then demonstrated the functionality of their boards to the instructor, who graded using a standard rubrick.

Assessment

The efficacy of the PCB exercise was measured using five metrics. Four were self-assessment by the student, and one was by instructor evaluation. The results are summarized in Tables 1-3.

At the end of the course, students were asked to fill out a three-question survey giving their opinions on the efficacy of the PCB exercise. Results are summarized in Table 1.

Table 1 Results from student survey on PCB exercise.

Question	Average (out of 5)	St Dev	n
How much did the PCB exercise help you to learn to design and fabricate printed circuit boards?	4.5	0.5	18
How much did designing the ALU for the PCB help you to understand the design of the 32-bit MIPS ALU?	4.4	0.8	18
I was successful in my design, fabrication, and testing of the ALU on the PCB.	4.3	1.0	18

Overall, students felt that this was a good exercise for helping them gain proficiency in designing PCB's. In the question asking them, "How much did the PCB exercise help you to learn to design and fabricate printed circuit boards?," students responded positively. All students answered either 4 or 5 on their responses with an average response of 4.5. Students also felt that the integration of the PCB exercise into the computer architecture course helped them to understand the computer architecture material. Over half of the students ranked this question as a 5, with an average of 4.4 for student responses. The last question concerned whether students felt they were successful or not in their design. Students average response was 4.3. This was student self-assessment, and instructor assessment is dealt with below.

More interesting than the success of students' projects, though, is a gauge of students' abilities after the experience. Even if a student was not successful with their circuit, they still may have gained valuable skills in the process. To measure the outcome of practical ability in PCB design, students were asked to self-assess on this outcome. Results are shown in Table 2.

Table 2 Outcome evaluation for PCB exercise, student self-assessment.

Question	Yes, Definitely (5)	4	3	2	No, Not at All (5)
Can you design and layout an 8-bit CPU on a printed circuit board?	12	3	3	0	0
Mean = 4.5, Mode = 5		StDev = 0.8		n = 18	

The results show a better overall self-assessment on general ability over success in the specific project. Twelve as opposed to ten students ranked their ability as a 5, and there were no

responses in the lowest two categories. Thus, all students felt they received some proficiency in PCB design, and two-thirds of them felt that they are "definitely able" to design and layout an ALU on a printed circuit board.

Finally, Table 3 shows the result of the instructor evaluation of the boards themselves.

Table 3 Board function evaluation, by instructor.

Total Number of Boards	13
Average Score	95.7%
Fully Operational	7
Minor Problems	3
Significant Errors	2

Of the thirteen boards, seven worked flawlessly. Three had minor mistakes, but these were typically with the carry, zero, and overflow flags, where the operations and results were functional. Two boards had significant problems, but even on both of these, the AND and OR functions were correct.

Lessons Learned

The experience of integration of this subject into a computer architecture class led to several conclusions. The primary conclusion is that incorporating such an exercise has a two-fold benefit. Students gain proficiency with industry-standard tools and gain the ability to design, have fabricated, populate, and test printed circuit boards. Second, this exercise actually reinforces the main lecture material of the structure and design of an ALU.

The drawbacks to this exercise are significant however. A significant cost is incurred for the software ($2000), board fabrication ($50 each), and components ($21 each). Further, the demand on instructor's time must be also recognized. It is estimated that the exercise took up over 100 hours of the instructor's time for the semester, this being in addition to the lecture preparation, homework help, and so on. The exercise does cut into other potential assignments, such as homework and other laboratory exercises, and does use a moderate amount of lecture time. Even with the amelioration of other assignment load, students did typically spend more time on the PCB exercise, and that must also be taken into consideration.

The learning curve for familiarity with the software will be mitigated upon successive implementations of the exercise. However, ever-changing software often stands in the way of knowledge re-use. To ease the cost of software, instructors can consider using freely-available tools. For extreme cost savings, students' board areas could be limited, and multiple smaller boards could be combined on one fabrication run of a larger board. One excellent reference for decreasing the learning curve is found in Mitzner[2].

Though the costs in time and money for this project are significant, it is felt that the overall benefits demonstrated make it worth the time.

References

1. Patterson, D. A., Hennessy, J. L., Computer *Organization and Design: The Hardware/Software Interface*, 3d ed. San Francisco: Morgan Kaufmann, 2005.
2. Mitzner, K., Complete *PCB Design Using OrCAD Capture and Layout*, Burlington, MA, 2007.