Learning to Teach Microprocessors, Work in Progress

Ziad Youssfi
Department of Electrical & Computer Engineering and Computer Science
Ohio Northern University
Ada, Ohio 45810
email: z-youssfi@onu.edu

Abstract

After teaching Microprocessors class at Ohio Northern University a second time, I would like to share the improvements I made that helped engage students more deeply than during my first time teaching the class. My hope is that this sharing helps other faculty and elicits feedback from faculty who teach a similar class.

"Microprocessors" is a required class for juniors and seniors in electrical and computer engineering majors at ONU. The topics covered follow the IEEE Computing Curricula for Computing Engineering recommendations [1], which include basic processor architecture, memory structure, I/O and buses, interrupts, timers and counters, and digital to analogue (DAC) and analogue to digital (ATD) conversion.

In this paper, I discuss implementing new lab experiments to promote problem-based learning (PBL). Since in my teaching the class the first time I found students to have difficulties connecting lab experiments to real-world applications, I introduced in the lab two real-world applications: camera shutter controls and bread-maker controls.

I also included active learning exercises. I found that simple active learning exercises considerably improved students' attention. The exercises did not take away time from covering the course topics — if anything, they actually helped me cover more topics.

Finally, I discuss reviewing binary numbers representation at the beginning of the course to give students the necessary confidence and tools to perform their experiments.

For my next time teaching the course, I propose including a heart-monitor as a real-world application to promote PBL. The heart monitor application would also be Internet connected to promote student thinking about entrepreneurship, the Internet of Things (IOT) and wearable computing.

1 Introduction

In the fall of 2013, I taught the Microprocessors class at Ohio Northern for the first time. The class had been taught in the past through lectures followed by lab exercises. In the lecture, concepts such as instruction execution, interrupts, and timers were explained. The labs would then ask the students to apply these concepts in weekly exercises, for example to generate a specified LED light pattern or to control a stepper motor with certain timing.

As I followed this teaching model, I started wondering whether such lab exercises were too abstract and whether they provided a context of "real-world" applications. Students' feedback confirmed my suspicion: many students wrote they needed "more examples" in their end-of-term evaluations. Although I strived to mention example applications in the lectures, the students were not actually doing these applications in the lab.

In this paper I describe changes I made during my second time teaching the class in 2014 to provide real-world context to the lab exercises. Although these applications are still in development, they are a step in the right direction for turning the Microprocessors labs into problem based learning (PBL) exercises.

I also discuss the inclusion of active learning exercises in the lecture as another important change. As an example, I describe the how students were given the task during the lecture of figuring out instruction execution latencies when loading or storing data from SRAM versus DRAM.

Finally, I discuss the need to review prerequisite material related to binary numbers and binary numbers arithmetic and representations. Even though these topics are taught in a prerequisite Digital Logic class, reviewing them again in the Microprocessors class seemed to increase student confidence and performance.

2 Toward Problem-Based Learning

2.1 Timer and Interrupt: Simulated Control of a DSLR Camera Shutter

To help students connect the concept of microcontroller timer function to a "real-world" application, I created a new lab experiment in which the students are asked to simulate the control of a shutter in a (digital) single-lens reflex (DSLR) camera, illustrated in Figure 1. The amount of light that reaches the digital sensor in a DSLR is controlled by the lens aperture and the shutter's timing. Thus, the shutter mechanism makes an excellent problem through which students can learn how to program the timers of a microprocessor to control photo exposure. In the lab, the students have access to HSC12

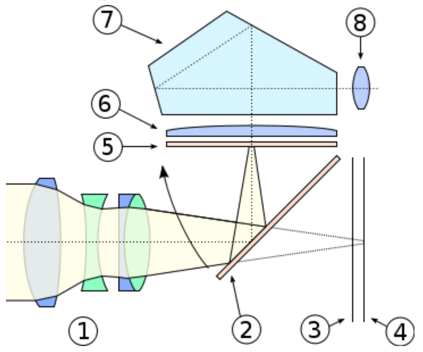
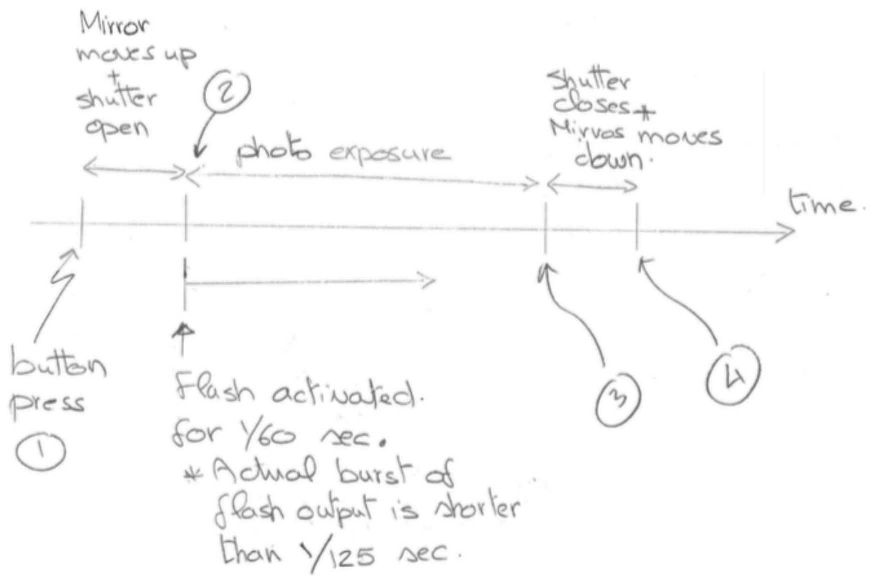


Figure 1. DSLR components: (1) Lens (2) reflex mirror (3) shutter (4) image sensor (5, 6) focusing screen and lens (7) pentaprism (8) viewfinder. When the shutter button is pressed, the mirror swings upward and the shutter is opened simultaneously to expose the sensor plate.

microprocessor development boards and DC steppers motors that can emulate driving the shutter and mirror mechanisms. The students are then asked to move the stepper motor according to a timing diagram given in Figure 2. The requirement is to move the



Mirror moves up + shutter open
2
photo exposure
Shutter closes + Mirror moves down.
time.
button press
1
Flash activated. for 1/60 sec.
* Actual burst of flash output is shorter than 1/125 sec.
3
4

Figure 2. Basic timing diagram for DSLR exposure.

motor in a given direction and degrees to open the shutter within only 50 to 70 ms after a shutter button is pressed on the evaluation board. Using input switches on the evaluation board, the students should emulate photo exposures of the following durations:

Input switches [2-0]	**Photo Exposure Duration in seconds**
000	1/125
001	1/60
010	1/30
011	1/15
100	1/8
101	1/4
110	1/2
111	1

Moreover, the students must also control an LED light on the evaluation board to emulate a camera flash, as in the diagram of Figure 2. Since the flash duration happens simultaneously with the photo exposure but with different timing, the students have to exercise their understanding of the concept of microprocessor interrupt to produce such timing. That is, the students have to program a different timer with different priority to produce an interrupt during or after the photo exposure to stop the flash.

Altogether, the emulated camera shutter and flash allow the students to relate their understanding of all aspects of microprocessor timers and interrupts to a real-world application.

2.2 Interrupts -- Simulated Control of a Bread Maker Machine

Before students can apply their concept of microcontroller timers, they must already have a working knowledge of the concept of microcontroller interrupts—since timers often apply their action through interrupts. So earlier in the semester, and before the camera shutter exercise, I asked the students to emulate interrupts in the control software of a bread-maker machine. In addition to being a personal favorite appliance of mine, the bread-maker machine provides a good context for applying microcontroller interrupts for two reasons.

First, the embedded software control of a bread-maker machine has enough complexity that students must think carefully about consequences and immediacy of interrupt service routine (ISR). For instance, the embedded software program could be reading sensor data for temperature and humidity, actuating a motor at different speeds to emulate kneading, and controlling heating elements to maintain a given temperature. At any point in time, a user can press the pause button to interrupt the main program. The students must then think about providing an ISR that can immediately pause some aspects of the program (e.g. actuating the motor) while maintaining other aspects, such temperature control.

Second, pausing the bread-making program requires the students to learn about enabling and disabling global microcontroller interrupts and about interrupt priority. For instance, pressing the pause button should automatically disable global interrupts so other interrupts would not interfere and an ISR would not interrupt itself. However, in order resume the program, the user has to press the pause button again. Therefore the students learn how to carefully allow an ISR to interrupt itself (i.e. let it call itself again) in order to return the main program. They also learn about interrupt priorities by allowing only interrupts of higher priority than the pause button to proceed.

The students can emulate the interrupt functionality of the bread-maker machine using the HSC12 microcontroller evaluation board, DC motors, LCD display, sensors, and push buttons and switches on the evaluation board.

3 Active Learning Exercises

Teaching and learning about microprocessors and microcontrollers concepts often involves presenting a great quantity of details. But presenting these concepts in a lecture can overwhelm the students and lead them to disengage if they are not involved in some form of active learning exercises. To this end, in my second time teaching Microprocessors, I started applying active learning exercises based on the active learning methods presented by Felder and Brent [2].

Although many forms of active learning exist, I found that for the Microprocessors class, in-class teams to solve a problem or work out a derivation engaged the students effectively. For example, to present the topic of performance of different memory technologies such as SRAM for cache and DRAM for main memory, I organized the lecture as follows:

- I first presented the memory technology in terms of circuits and tradeoff in term of cost and speed, i.e. why SRAM is faster than DRAM but takes more chip area and is more costly than DRAM. We then quickly reviewed instruction execution state diagram for load and store instructions. We had covered this topic earlier in the semester, so only quick review was sufficient.
- I then let student pairs calculate the number of bus cycles and clock cycles required to execute a loop of instructions that loads data from DRAM. After a few minutes, I asked a group to share their solution.
- I then introduced the concept of cache, cache blocks, and cache replacement policy.
- I asked the students to get back into their groups to calculate the number of clock cycles required to execute a single load instruction and then a loop of instructions loading data from cache, assuming that the first loop iteration was a cache miss followed by cache hits.
- Finally, I asked to students to calculate the speedup as a result of using the SRAM cache versus DRAM for executing the loop of instructions.

By letting the students incrementally solve more complex steps of the problem in class, most (if not all) of the students were able to identify the right solution. At the same time, their strong sense of engagement was palpable!

4 Prerequisite Review

One lesson I learned from teaching Microprocessors the first time is that not all students had complete mastery in manipulating binary numbers and their representations. Although binary numbers is a topic covered in a prerequisite digital logic class, some students had trouble with converting between decimal, hexadecimal and binary representations. They also had trouble with binary arithmetic and two's complement representation.

Since binary numbers is such an essential topic in Microprocessors, and to avoid students not having the tools necessary to succeed, I decided when I taught the course a second time to review the topic in the first two lectures. In the third lecture I gave a quiz on the topic. The quiz reinforced the students' skills on the topic and assured me that the students are now ready to proceed with new material. Throughout the course, I felt that all students were much better at manipulating binary numbers than they were during my first time teaching the class.

5 Students' Performance and Surveys

Since I felt student engagement was stronger in the course of the fall of 2014, I was not surprised (and pleased) to see the students performing much better on all metrics: quizzes, homework, and midterm and final exams. Moreover, attendance was stronger than in the fall of 2013. And in the lab, students were much more motivated to complete the experiments. Table 1 shows the improvement in average course grades on a 4.0 scale. The number of students in Table 1 is also broken down by the two programs that require the course: Electrical Engineering (EE) and Computer Engineering (CPE). While student grades improved in the second course, the exam questions were also more conceptually challenging, which confirms my intuitive sense that the students had deeper understanding of the topics.

Table 1. Students' average grades for the two course.

	Class Size	**Avg. Grade by Program**		**Avg. Grade Total**
		EE	CPE	
Fall 2013	16 (4 EE + 12 CPE)	3.08	3.5	3.19
Fall 2014	22 (12 EE + 10 CPE)	3.67	3.9	3.77

In both courses, the students were surveyed at the middle and the end of the term to gather their feedback on the course and the instructor. On each question, the student rate their responses on a scale from 5 to 1 where 5 = "strongly agree" and 1 = "strongly disagree". Here are the set of questions related to the course:

Q1: The course learning outcomes were clearly explained at the outset of the course

Q2: The course learning outcomes were appropriate for a course of its level.

Q3: The course was designed to foster learning of the course material.

Q4: The course provided me with an important skill set needed for further studies in this field.

Table 2 shows the improvement in the student response regarding the course overall between fall, 2013 and fall 2014.

Table 2. Average response to questions Q1 to Q4 regarding the course, 5 = strongly agree, 1 = strongly disagree). N is the sample size. Standard deviation is shown in parenthesis.

Course Questions:	**Q1**	**Q2**	**Q3**	**Q4**
Fall 2013 (N = 10)	3.9 (.83)	3.9 (.83)	3.9 (.83)	3.9 (.83)
Fall 2014 (N = 22)	4.6 (.48)	4.6 (.48)	4.7 (.45)	4.5 (.89)

Two of the most important questions regarding the instructors that the survey asks are the following:

Q1: The instructor motivated me to do my best work.

Q2: The instructor's overall teaching of the course was effective.

Again the students responded by rating each question from 5 = strongly agree to 1 = strongly disagree. Table 3 shows the average student response for question Q1 and Q2 above and their improvements over the two semesters.

Table 3. Average response to questions Q1 and Q2 regarding the instructor. 5 = strongly agree, 1 = strongly disagree. N is the sample size. Standard deviation is shown in parenthesis.

Instructor Questions:	Q1	Q2
Fall 2013 (N = 10)	3.9 (.83)	4.1 (.70)
Fall 2014 (N= 22)	4.5 (.66)	4.5 (.66)

The higher response rate to the surveys in fall 2014 (~100%) also suggests more student engagement in the class.

6 Future Work

Although the lab exercises follow the problem-based learning model, they only emulate real-world applications. That is, the camera shutter exercise actuates a DC motor to emulate the shutter mechanism, but there is no actual shutter, flash, or photo sensor. Similarly, the bread-maker machine emulates the controls of the appliance DC motor and LCD, but there is no actual appliance. Therefore, work remains to offer these applications in their complete form.

A possible complete application that I have been pursuing as a PBL for the course is a heart rate monitor. This type of application would allow the students to apply many Microprocessors concepts such as interrupts, timers, peripheral interfacing, and analog-to-digital conversion. More importantly, it would be a fully complete application, not just emulation. Students would detect and measure their own heart rate with their own application that they develop it.

The challenge of using a heart-rate monitor as a PBL is that the topics it covers are cumulative in nature. The earliest that the students could start development is after the timer concept has been covered, which is usually after the middle of the semester. This leaves the students with only a few weeks to develop and debug the full working application.

7 Conclusion

I see students engage more deeply in the Microprocessors class as I follow the PBL model to create "real-world" applications in the lab, pursue active learning exercises in the lecture, and help students start off with a good basic knowledge of binary numbers. Students' grades improved as well as did their evaluation of the course and instructor. Challenges remain to improve the PBL exercises in the lab, but I think these improvements can be incrementally achieved by building on previous teaching experiences.

References

1. IEEE Curricula, Computer Engineering, available at: http://www.eng.auburn.edu/ece/CCCE/CCCE-FinalReport-2004Dec12.pdf

2. R. M. Felder, R Brent, "Active Learning: An Introduction". ASQ Higher Education Brief, August 2009.