2021 ASEE ANNUAL CONFERENCE
Virtual Meeting | July 26–29, 2021 | Pacific Daylight Time
Paper ID #34473

A Hands-on Learning Approach to Introducing Computer Organization and Architecture to Early-college Students

Dr. D. Cenk Erdil, Sacred Heart University

Dr. Erdil has joined Sacred Heart University's School of Computer Science & Engineering in Fall 2017. Prior to SHU, he has held academic positions at Marist College, Columbia University, and Istanbul Bilgi University. His research interests include using Cloud Computing as Artificial Intelligence Infrastructures, Cyber-Physical Systems and Internet-of-Things, Teaching coding to P-12 students, and Health Informatics. He is the author of numerous peer-reviewed journal and conference publications in grid and cloud computing. In the past, he designed and implemented a cloud-based public health informatics infrastructure. He is a founding member of the School of Engineering at Istanbul Bilgi University, and was the chair of its Computer Engineering Department. He also designed an adaptive resource-matching framework for large-scale, autonomous grid computing environments, using epidemic dissemination protocols. He is the founding director of Engineers Without Borders International, Turkey branch. At the industry, Dr. Erdil has worked in management and software engineering roles for more than a decade at various organizations, including Fidelity National Information Services (FIS), and Turkish Airlines. He is a senior member of the Association for Computing Machinery (ACM), and a senior member of Institute for Electrical and Electronics Engineers (IEEE); and a member of Engineers Without Borders International (EWB-I), American Society for Engineering Education (ASEE), and Association for Information Systems (AIS).

Dr. Kevin N. Bowlyn, Sacred Heart University

Kevin N. Bowlyn is an Assistant Professor at Sacred Heart University. His current research interest is focused on a more efficient method for computing a fast Fourier transform (FFT) algorithm. His research interests are in digital hardware design, digital signal processing, low area-power circuit designs, embedded systems, and computer architecture.

Mr. Joshua Randall, Sacred Heart University

A Hands-On Learning Approach to Introducing Computer Organization and Architecture to Early College Students

D. Cenk Erdil, Kevin N. Bowlyn, and Joshua A. Randall
School of Computer Science & Engineering
Sacred Heart University
Fairfield, CT

Abstract

We present a design and implementation of a lower-level computer organization and architecture course with hands-on components presented as blended-learning modules that are collectively designed to introduce core computer design concepts primarily to college students studying applied science and technology programs, such as computer science and information technology.

With a particular focus on single-board computers and associated hardware modules, students are introduced to core computer components early in their coursework, and encouraged to study advanced engineering concepts as higher elective courses to help them better understand the underlying design of hardware modules. Hands-on activities and problem-based modules are re-designed with the flexibility to be applied in settings that involve all in-classroom cohorts, as well as courses offered in synchronous and/or asynchronous online learning methodologies, which is becoming of particular importance to educators under COVID-19 implications.

First cohort of this newly redesigned course was offered in Fall 2019, when, in the middle of the semester, all instructional methodology had to be switched to fully-online after health measures in October-November 2019. Second and third cohorts of the course are currently being offered in 2020-2021 academic year. Anonymous research data collected with these three cohorts of the revised course show that redesigned of the course improved overall course reviews, while meeting educational goals to introduce students to core knowledge areas in computer organization and architecture.

1 Introduction

This article describes details of design and implementation of a lower-level (core-Tier1) computer organization and architecture course with online hands-on components as common learning environments. We use the following pedagogical approaches for the revised course content: lead-learner, blended course delivery, flipped classroom, and project-based learning.

Online hands-on component of the course has been focused on using a single-board computer, and associated hardware with the aim to provide students contemporary skills in implementing computer orientation and architecture projects with related software components. We have collected research data after the first two cohorts (in the same academic year, two separate sections of the same course).

While we continue to collect research data in subsequent cohorts in (currently) Spring 2021 and (upcoming) Fall 2021 sections, our early student responses show that new design has improved overall course reviews, while achieving curriculum guideline goals for common computer organization and architecture course design. In addition, course materials that include core knowledge areas (KAs) have been kept intact, and student feedback shows that they understand each KA at comparable levels to classical computer organization and architecture course content.

2 Method

In typical computer organization and/or computer architecture courses, knowledge areas are composed of the following concepts [1]:

- Digital logic
- Digital systems
- Machine level representation of data
- Assembly level machine organization
- Memory system organization
- Memory architecture
- Hardware interfacing
- Component communication

- Functional organization (optional)
- Multiprocessing (optional)
- Alternative architectures (optional)
- Performance enhancements (optional)

Modules of our hands-on components are based on the following curricular guidelines: lead-learner, blended course delivery, flipped classroom, and project-based learning. We have used the Englander's Information Technology Approach of the Architecture of Computer Hardware, Systems Software, and Networking, 5th Edition textbook [3]. The following paragraphs explain details of implementation of each curricular guideline in our course design.

In lead-learner, our primary goal is to provide a facilitator role as the person responsible for delivery of the course, and we facilitate an active discussion for each week, and strive to initiate a co-learning environment. There are short time periods where the instructor has to switch back to classical teacher role, go over the presentation slides for a short period of time (typically less than 15 minutes), and then changes the methodology back to lead-learner, rephrasing the understanding of what that particular module has presented to a student's view. This approach provides students opportunities to ask more questions, and increase the overall engagement in class. This methodology has become especially important with changing delivery of course content with multiple modalities under COVID-19 implications, in particular to handle synchronous and asynchronous students within the same cohort.

Blended course methodology allowed us to redesign the hands-on section of the course, with complementary practical applications to the theoretical part, and provide students hands-on skills in working with single-board computers, together with sensors and other digital components, and associated programming activities based on microcomputer platforms such as Arduinos [2]. These alternating hands-on components also helped us flip the class delivery environment from technical to discussion to other modalities, and kept students engaged in overall discussion of fundamentals of computer organization and architecture.

In addition, a project-based component helped students to design and implement a basic computer organization idea, using the fundamental skills they have learned in class. Almost all of the students being in their sophomore year, stated that this class made them excited about their major choice, and indirectly contributed to increasing levels of retention, especially in

early college experiences in computer science and engineering classes. Figure 1 shows demographics of age distribution and class standing in Fall 2019 cohort.

Figure 2 shows distribution of undergraduate majors within the college, with majority of students either already declared or interested in a technology-focused undergraduate degree, including Computer Science, Information Technology, Game Design, Computer Engineering, Cybersecurity, and Electrical Engineering. This course has been an optional course for Computer Engineering and Electrical Engineering students, as those majors have several other computer organization and architecture related courses in their core curricula, respectively.

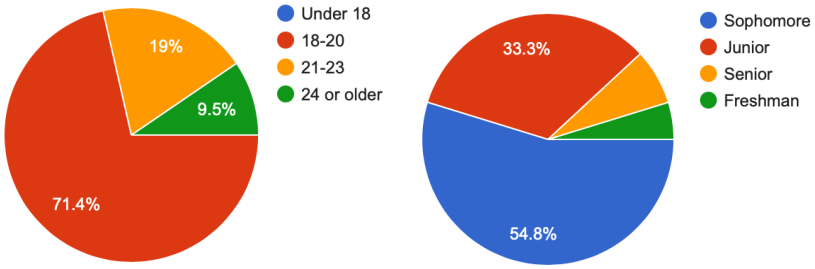

Figure 1: Demographics of age distribution and class standing.

3 Course Design

In the past, this course has been offered at our university by the 40+ year-old department of computer science, which started a major expansion in 2017, and opened three new programs and split its computer science program into three separate undergraduate programs. As part of this expansion, this course has been re-designed, with the following primary goals in mind:

- To re-design major courses in computer science programs as school initiated ABET accreditation process in 2016,
- To align with newly offered undergraduate programs in computer science and engineering within the school,
- To introduce project-based components to a sophomore-level computer organization and architecture class,

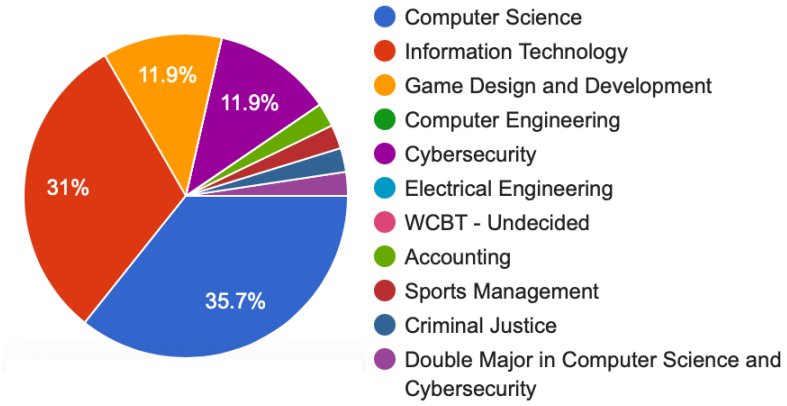

Figure 2: Program of study by percentages (N=42). One student in Fall 2019 cohort is studying two programs: Computer Science and Cybersecurity. WCBT Undecided are those undergraduate students, who have not decided on their major studies yet.

- To increase student retention in early college major programs in STEM,
- To provide a general overview of the fundamentals of computer architecture to a wide variety of students from different majors.

The course learning materials are based on Irv Englanger's "The Architecture of Computer Hardware, Systems Software and Networking - An Information Technology Approach" textbook.

The main course objectives are to get students equipped with the following goals:

- Understand the basics of computer hardware and how software interacts with computer hardware,
- Know the organization of the central processing unit (CPU) and memory hierarchy,
- Use critical thinking to make informed decisions in the selection of hardware,
- Learn and demonstrate how program performance is affected by processor cache sizes,
- Understand how the architecture affects program performance,

- Demonstrate how instructions can be implemented in a chip,
- Learn how to use a low-level assembly language to manipulate registers on a computer architecture performing a specific operation with data,
- Learn how computer organization influences high-level languages, and vice versa.

Following student learning expectations and outcomes has helped us design weekly course activities:

- Identify and analyze core computer organization problems,
- Explore principles of computer architecture,
- Create a program in a low-level assembly language that works with data that fit and exceed processor cache sizes, and measure the performance impacts of doing so,
- Write a simple program in a low-level assembly language to implement a high level program segment,
- Using an Arduino board, control a sensor via software,
- Use Linux operating system and embedded boards to implement basic computer organization artifacts,
- Design a physical computer-based project and implement it.

4 Opinion Survey and Results

In this section we provide results of the opinion survey we have asked our students to respond to at the end of the course. As the results of first-time offering of this course, this course was offered as two cohorts (in two separate sections) with a total of 42 students (N=42). Students have been asked to fill out the survey anonymously, with all the responses provided between 4/15/2020 and 4/27/2020. These responses have been provided together with university standard course evaluation forms (which are also anonymous to instructors), and data have been collected before students have learned their final grades for this course.

Opinion surveys have started with a set of general questions about the course, as Figure 3 shows, and focused on overall experiences in the course, as shown in Figures 4, and 5 show.

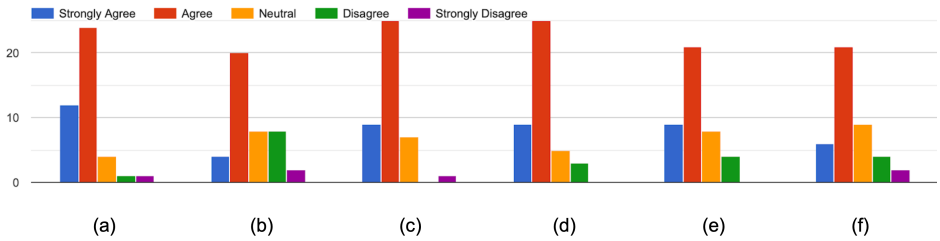

Figure 3: Student responses to the general questions about the course. (a) The level of material covered in the class was suitable for the topic. (b) I was able to follow the material without being too overwhelmed. (c) The course materials contributed to my learning. (d) The objectives of the course were clear. (e) The instructor raised questions or problems that encouraged me to think critically. (f) My interest in the subject matter was enhanced by the instructor's enthusiasm.

5 Discussion and Conclusion

The crucial issue of introducing Computer Organization and Architecture to the early college students is to get them exposed to Computer Science concepts early in college studies. This is particularly important for students who have not declared their major yet, and are looking for more experiences about the technology-focused majors in our college.

By being exposed to engineering and hands-on laboratory-based experiences early in their college experience, students may use critical thinking to make informed decisions in the selection of hardware, which will collectively get them exposed to expenses in computer science and engineering programs. A simplified syllabus for this course has been shared as an attachment.

This article provides details of a revised computer organization and architecture course, offered primarily to sophomore students in a school of computer science and engineering course with six undergraduate major programs. After this first offering of this redesigned course, overall approval rates have improved for this course. This first offering of this course was affected by the COVID-19 implications (eliminated majority of the classroom meetings, as hands-on design experiment experiences were moved to online activities), although overall student evaluations were expressed positively. We are currently collecting additional research data in current Spring 2021 semester, which -similar to Fall 2020 semester-, included online activities with multiple delivery modalities, include synchronous and asynchronous hands-on experiences. We will also offer this same course to a cohort in Fall

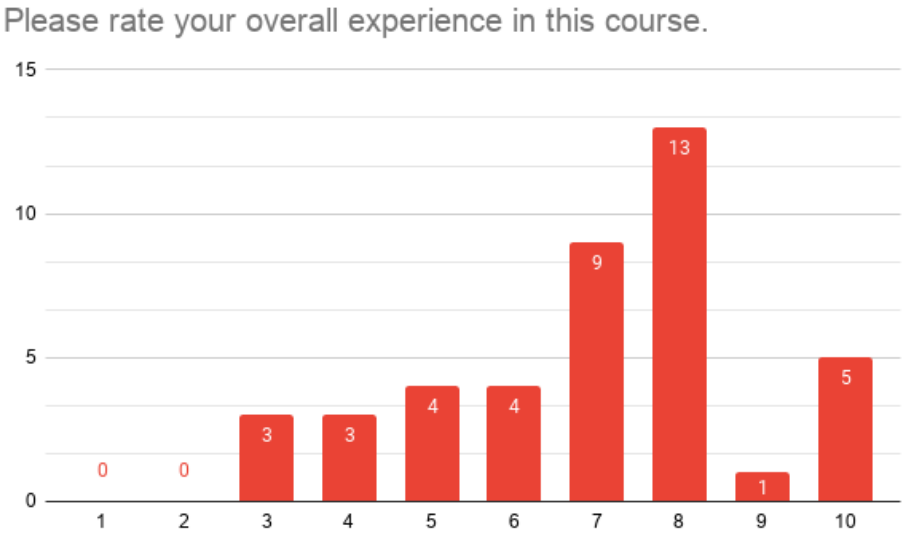

Figure 4: Overall experience of students, on a Likert scale from 1-10 (n=42). Satisfaction displays an overall high-approval rate, with about 23% students below satisfactory levels. After reviewing individual responses and feedback from students, we attribute this to the effect of COVID-19 measures, where about half of the students were not able to attend the classroom, and complete hands-on activities. However, overall student approval rates show design projects overall increased overall student agreement, with regular bell curve peaking around 8/10 (30% approval) which was well received. Moreover, about 12% students were completely satisfied with the course (10/10).

2021 semester, with expected fully in-class experiences after COVID-19 measures are released after herd immunity is reached. We plan to follow-up our results after Spring 2021 and Fall 2021 semester data have been collected, with a comparison of the affects of online learning modalities, and will potentially alter some of the hands-on experiences to handle asynchronous learning activities.

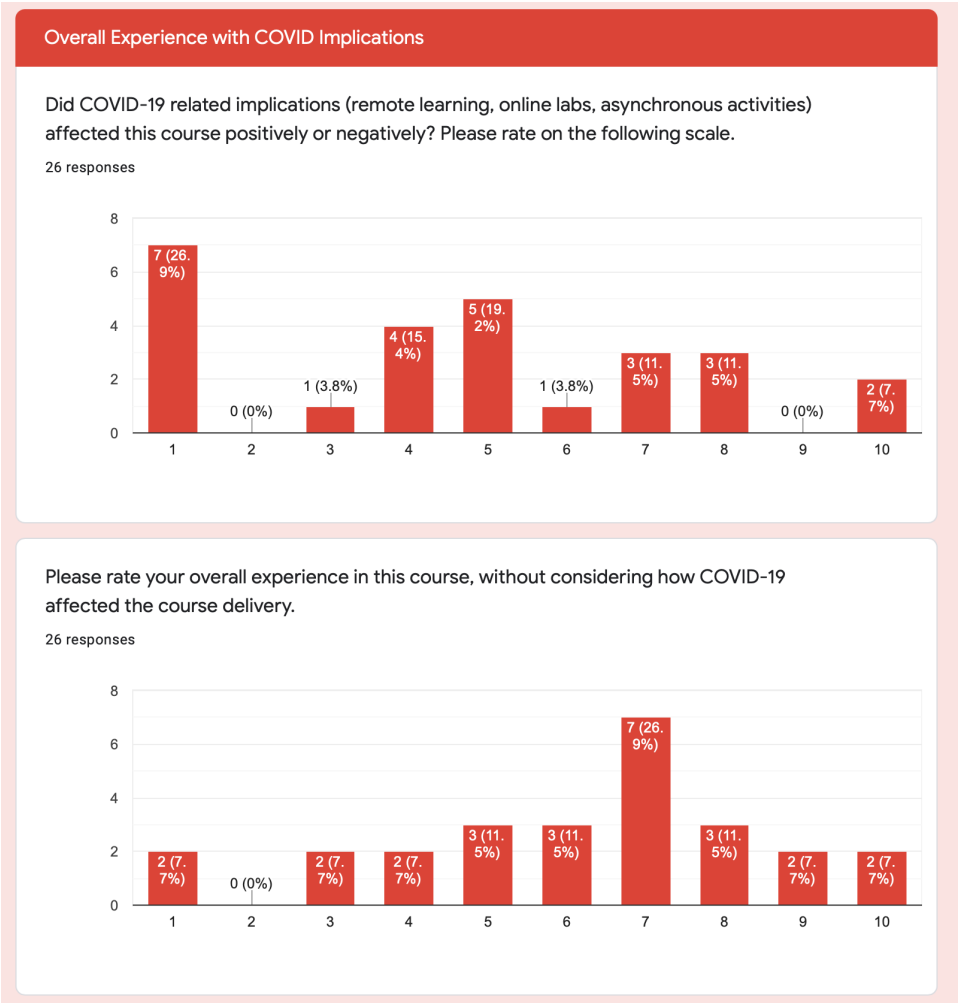

Figure 5: Overall experience of students, on a Likert scale from 1-10, with Spring 2021 cohort, who experienced this course with only an online option in Spring 2021, due to COVID-19 restrictions. (n=26). Satisfaction displays an overall high-approval rate, however you can certainly see the negative effect of online course experiences in this particular course. More details of this data collected will be finalized in about a month, and will be shared during ASEE presentation with attendees in Summer 2021 conference.

References

[1] Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) and IEEE Computer Society. 2013. Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science. Association for Computing Machinery, New York, NY, USA.

[2] Brian W. Evans. Arduino Programming Notebook. 2007. https://archive.org/details/arduino_notebook

[3] Irv Englander. The Architecture of Computer Hardware, Systems Software, and Networking: An Information Technology Approach, 5th Edition. Wiley, January 2014.

CS-215 - Computer Organization and Architecture - Spring 2020 Syllabus

A University Catalog Course Description:

This course presents an overview of computer architecture and computer organization as they relate to computer science. Topics include computer components, interconnection structures, internal memory, instruction sets, number representation in computers, parallel processing, and an elementary introduction to assembly programming.

B Textbooks / Materials:

1. Irv Englander, "The Architecture of Computer Hardware, Systems Software, and Networking - An Information Technology Approach", Wiley, 5e, Jan 2014.

2. Computer Organization and Architecture Tutorials.

   https://www.geeksforgeeks.org/computer-organization-and-architecture-tutorials/

3. Other resources will be provided as needed.

C Course Objectives:

The main goal of this course is to get you equipped with the following: Understand the basics of computer hardware and how software interacts with computer hardware; Know the organization of the central processing unit (CPU) and memory hierarchy; Use critical thinking to make informed decisions in the selection of hardware; Learn and demonstrate how program performance is affected by processor cache sizes; Understand how the architecture affects program performance; Demonstrate how instructions can be implemented in a chip; Learn how to use a low-level assembly language to manipulate registers on a computer architecture performing a specific operation with data; Learn how computer organization influences high-level languages, and vice versa.

D Student Learning Expectations/Outcomes:

- Identify and analyze core computer organization problems,
- Explore principles of computer architecture,
- Create a program in a low-level assembly language that works with data that fit and exceed processor cache sizes, and measure the performance impacts of doing so,
- Write a simple program in a low-level assembly language to implement a high level program segment,
- Using an Arduino board, control a sensor via software,
- Use Linux operating system and embedded boards to implement basic computer organization artifacts.

E Data for Research Disclosure:

Any and all results of in-class and out-of-class assignments and examinations are data sources for research and may be used in published research. All such use will always be anonymous.

F Meeting Schedule:

Table 1 shows a tentative weekly schedule for our Spring 2020 semester.

G Sample CS 215 Lab Experiment

In this course, you will be using a single-board computer (e.g., Arduino, Raspberry Pi, etc.), to design several computer hardware solution as a design experimental project.A few examples are illustrated in Figure 6 and Figure 7.

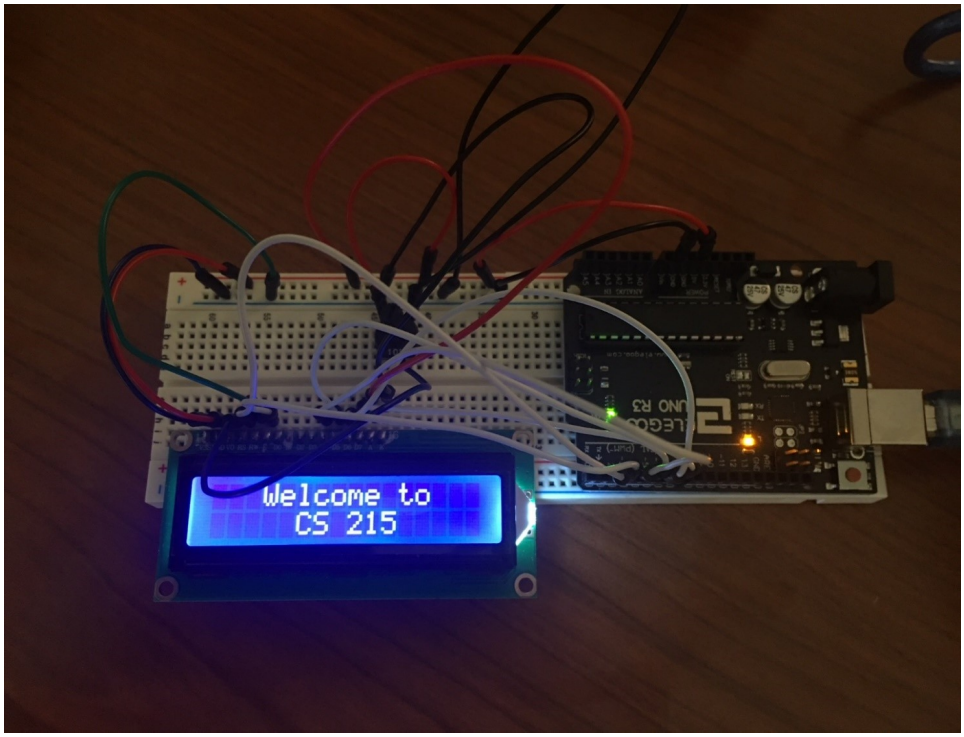

Figure 6: Shows a sample lab experiment using an Arduino and a liquid crystal display (LCD) to display word characters onto a LCD screen.

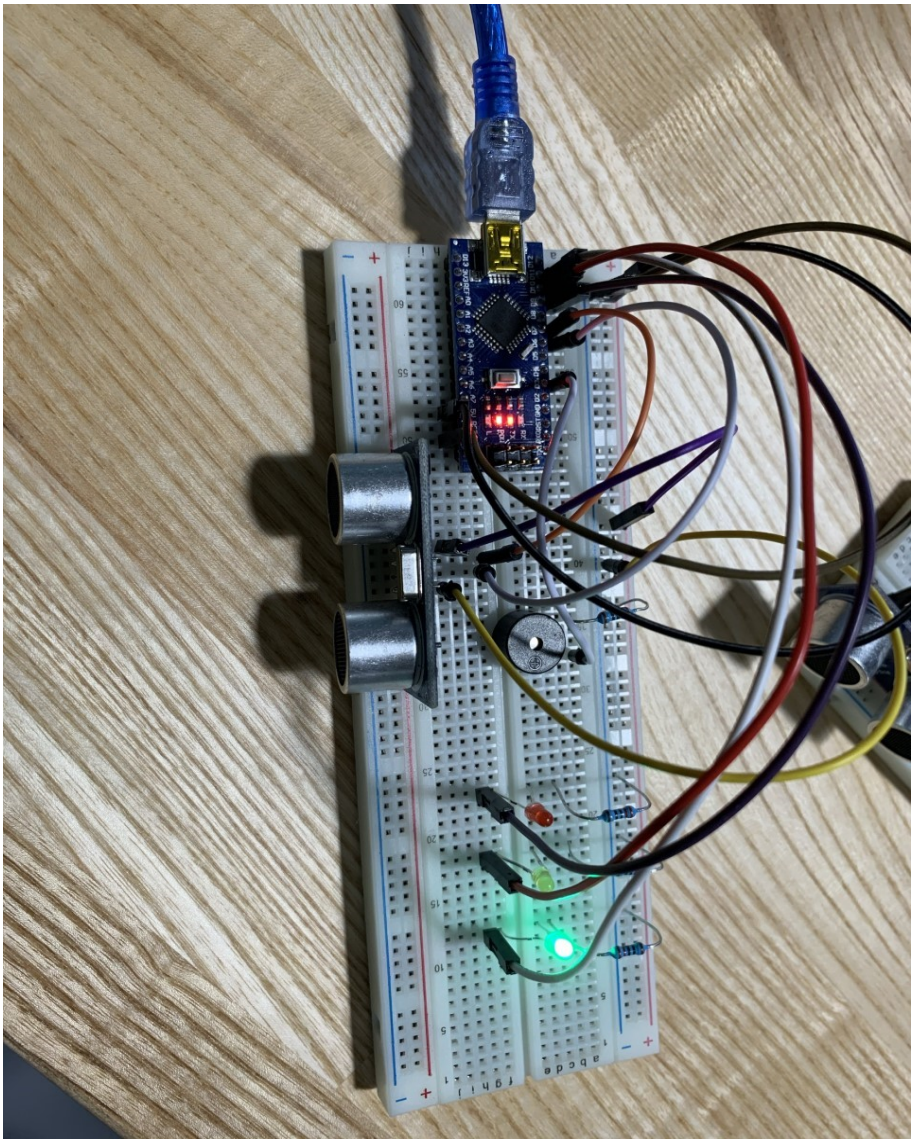

Figure 7: Shows a designed for a car parking assist buzzer using an Arduino, ultrasonic sensors, and a buzzer to trigger off one of the three LED's (Red, Yellow, and Green). When the car is initial approaching, the green LED light would light up. As the car approach closer and reaches a certain distance, the Yellow LED would light up. The Red LED will light up when the car is about less than or equal to 5cm away in which the buzzer should sound off indicating to the driver that he/she should stop.

Table 1: Tentative Schedule.

Weekly Outline			
Week No.	Week	Lecture	Reference
1	1/13 - 1/17	Class mechanics; Intro to Computers and Systems	Chapter 1
2	1/20 - 1/24	Intro to Systems Concepts and Architecture	Chapter 2
3	1/27 - 1/31	Number Systems	Chapter 3
4	2/3 - 2/7	Into to Arduino	
5	2/10 - 2/14	Data Format	Chapter 4
6	2/17 - 2/21	Representing Numerical Data	Chapter 5
7	2/24 - 2/28	Into to Raspberry Pi	
8	3/2 - 3/ 6	Spring Break - No Classes	
9	3/9 - 3/13	Midterm Activity - Take Home Exam	Chapters 1-5
10	3/16 - 3/20	The Little Man Computer	Chapter 6
11	3/23 - 3/27	The CPU and Memory	Chapter 7
12	3/30 - 4/3	CPU and Memory: Design, Enhancement, Implementation	Chapter 8
13	4/6 - 4/10	Input/Output	Chapter 9
14	4/13 - 4/17	Computer Peripherals / Project	Chapter 10
15	4/20 - 4/24	Modern Computer Systems / Project	Chapter 11
16	4/27 - 4/31	Finals	TBA