A Microcontroller-based DSP Laboratory Curriculum

Dr. Ying Lin, Western Washington University

Ying Lin has been with the faculty of Engineering and Design Department at Western Washington University since September 2010 after she taught for two years at SUNY, New Platz. She received her MS in Applied Statistics and Ph.D. in Electrical Engineering from Syracuse University, NY, respectively. Her teaching interests include first-year Intro to Electrical Engineering, circuit analysis sequence, and upper-division communication systems and digital Signal Processing courses. Her research areas focus on statistical signal processing for wireless sensor network applications and secure communications in wireless networks.

Prof. Todd D. Morton, Western Washington University

Todd Morton has been teaching the upper level embedded systems and senior project courses for Western Washington University's Electronics Engineering Technology(EET) program for 25 years. He has been the EET program coordinator since 2005 and also served as department chair from 2008-2012. He is the author of the text 'Embedded Microcontrollers', which covers assembly and C programming in small real-time embedded systems and has worked as a design engineer at Physio Control Corporation and at NASA's Jet Propulsion Laboratory as an ASEE-NASA Summer Faculty Fellow. He has a BSEE and MSEE from the University of Washington.

A Microcontroller-based DSP Laboratory Curriculum

Abstract

In this paper, we present digital signal processing (DSP) hands-on laboratory coursework which was developed based on a low-cost embedded microcontroller (MCU) platform. Recent advances in MCUs (e.g. ARM Cortex M MCUs) have made the embedded microcontroller an option for most DSP applications and therefore a practical option for the DSP laboratory. The selected MCU tool uses the same ARM Cortex-M4 platform as used for the embedded microcontroller courses in our program with the addition of the ARM CMSIS DSP library. Our work was inspired by the needs of creating meaningful hands-on DSP lab experiments in the allotted one term period (ten weeks) and by the goal of improving student success in implementing DSP-based culminating projects that meet desired goals within realistic constraints. The benefits of integrating the MCU tools in the DSP course are very promising. It permits more practical DSP laboratories and DSP-based capstone projects that render richer design experiences and makes meeting realistic design constraints feasible. Furthermore, it provides an integrated laboratory curriculum structure between embedded microcontroller and DSP courses which reduces students' unnecessary effort of learning new tools in different courses. Consequently, students can focus more on the subject matter and additional concepts can be introduced in the lab and enhanced learning outcomes can be added to the DSP course. These MCU- based DSP laboratory coursework were offered in Spring 2016. The assessment results from this course demonstrated the effectiveness of these developed lab coursework in achieving additional learning outcomes.

I Introduction

In most modern electrical and computer engineering curricula, digital signal processing (DSP) is an important area and an integral part of the curricula[1]. A well-perceived pedagogical approach to teaching DSP is through fusing DSP theory covered in lectures with well-designed hands-on laboratory coursework. This synthesizing approach enhances students' understanding of DSP concepts and permits students practicing real-time DSP algorithms through lab activities. The benefits of integrating lab coursework with lectures have been noted in the literature in various studies (e.g., the situated learning [2,3]). As such, the hands-on laboratory experiments are critical in order to be effective complement to lectures[4].

Conducting real-time DSP algorithms relies on hardware and/or software tools which can support these real-time tasks. Popular choices of such tools typically fall into two main categories. One is the use of dedicated and expensive DSP hardware, such as the Texas Instrument's TMS320C6713DSK board with the CCS software development tool. The other alternative is using an inexpensive embedded microcontroller (MCU) platform that is equipped with DSP functionalities[2]. Both types of tools have advantages and limitations.

As we reported in the preliminary work[5], the dedicated DSP platforms render powerful processing capabilities for DSP applications, but they are no longer the best solution to most designs that must meet realistic constraints such as cost, size, and power consumption. Moreover, students need to learn new hardware and software development tools before they can use such a platform to conduct hands-on lab activities. The effort and time spent to learn the tools often pose a great challenge. For instance, in our program at Western Washington University, the DSP course is a 10-week one-quarter course. Much time has to be allocated to allow students to become proficient at using the new DSP tools (TMS320C6713DSK board and CCS IDE). In addition, studies have shown that learning a new IDE or using an inappropriate IDE may impose additional cognitive load to students[6]. As such, the level of lab content that students could accomplish within the course span often was reduced due to the significant time and effort spent in learning the tools.

To solve this issue, since 2016, we have switched from the dedicated DSP hardware to a MCU-based DSP platform for the hands-on DSP lab coursework. In particular, the DSP class adopts the same MCU platform and KDS IDE software tool as those used in previous embedded MCU courses (see Table 1 for a summary of these courses). Based on the new MCU platform, we have developed a number of hands-on laboratory experiments for the DSP course.

Course Number/Title	**Scope of the Course**
EE244 Embedded Microcontrollers I	Introduction to microcontrollers, software development using KDS, assembly language programming
EE344 Embedded Microcontrollers II	Simple multitasking, peripheral drivers, C language programming, using GIT in KDS
EE444 Embedded Systems	Real-time kernels, team-based software development, and advanced applications

Table 1[5]: Summary of Courses using the KDS IDE software tool and the MCU platform

This unified approach of adopting common hardware and software tools for multiple courses greatly reduces students' time to learn how to use different tools, thus permits more time and effort to be focused on understanding DSP concepts and attaining more learning outcomes.

Our objectives have been to move to a common hardware and software platform that meet the requirements of the DSP course, the embedded microcontroller courses, and the culminating project to allow the students more experience with hands-on laboratory experiments tied to the expanded learning outcomes and to better prepare students to design culminating projects that meet realistic constraints.

The developed DSP hands-on laboratory coursework has been designed to enhance students' understanding of important DSP topics and to practice real-time DSP algorithm implementations. The selected MCU platform is suitable for implementing these real-time algorithms. Although MCUs in general have disadvantages in performance, ease of development, and power efficiency compared to dedicated DSP hardware[7], the benefits of saving students' time and effort in learning a new hardware platform and IDE along with its low cost outweigh the disadvantages.

The lab exercises focus on topics including spectrum analysis through DFT/FFT, FIR and IIR digital filtering, impact of fixed-point and floating point on digital filtering, and adaptive filtering. Students use both stand-alone C programs and the ARM CMSIS DSP code library to program the selected MCU platform.

Our first trial in Spring 2016 has shown to be successful. The developed laboratory coursework was carried out by students who took the DSP class in Spring 2016. We have noted promising improvements by using the MCU platform compared to the use of a traditional dedicated DSP hardware platform. In particular, students were able to complete more complex DSP algorithms and more lab content in the 10-week term of study and attained more learning outcomes. The effectiveness of the MCU-based laboratory coursework were demonstrated through student surveys and Lab/Exam performance presented in the Assessment Results section.

In this paper, we will first provide an overview of the DSP course, followed by a review on the selected MCU platform and the software development tools. Next we will describe the detailed hands-on laboratory coursework for the DSP course along with their targeted learning outcomes. Assessment results that testify the effectiveness of the developed laboratory coursework and proposed future improvements will be presented as well. Finally, preliminary results of the impact this new system has had on the culminating project will be discussed.

II Overview of the DSP Course

The DSP course at Western Washington University is offered as an EE junior-level course in a ten-week period. Prior to taking the DSP course, students have taken the Discrete Systems course which exposes students with topics such as sampling, DFT, and Z-transforms. The DSP course emphasizes topics include Fast Fourier Transform (FFT), spectrum analysis, digital filtering (FIR and IIR filters), and the implementation real-time DSP algorithms through hands-on laboratory activities.

The DSP course consists of weekly three-hour lectures and a two-hour lab session. The laboratory activities are divided into software simulation labs using MatLab and the MCU-based real-time DSP implementation labs. The lab activities facilitate the attainment of desired learning outcomes. Table 2^5 lists the desired learning outcomes of the DSP course. These seven outcomes were identified in fall 2015 based on our program level course assessment. Among these outcomes, outcomes #1 ~ #3 and #5~#6 were existing learning outcomes when the dedicated DSP platform was used for hands-on lab activities. Outcomes #4 and #7 are newly introduced after the MCU-based DSP platform has been adopted.

The use of the selected MCU platform which students have been familiar with prior to taking the DSP class offers promising benefits such as achieving additional learning outcomes (#4 and #7) and allowing students to practice more complex lab activities (e.g., Lab #4 in Table 3). In the past, this level of lab activity complexity was not achievable due to the lack of sufficient time when the dedicated DSP platform was adopted for laboratory coursework. For the same reason,

the additional learning outcomes #4 and #7 had been planned in the curriculum for several years, but only become attainable after the MCU platform was used.

Figure 1 shows the relationships among the lab coursework and the targeted learning outcomes.

Learning Outcomes	
1.	Implement Fast Fourier transform (FFT) to analyze the frequency spectrum of digital signals.
2.	Design and implement FIR filters and IIR filters and analyze filter performances including passband ripples/gain, passband roll-off (transition region width), and stop band attenuation.
3.	Obtain filter impulse response and frequency responses given FIR and IIR filter structures, difference equations, or transfer functions.
4.	Recognize the impact of quantization and fixed-point and floating-point hardware.
5.	Apply DSP techniques to design solutions for practical applications.
6.	Use software tools for DSP modeling and simulations.
7.	Implement DSP algorithms in real-time embedded hardware.

Table 2[5]: Desired learning outcomes for the DSP course

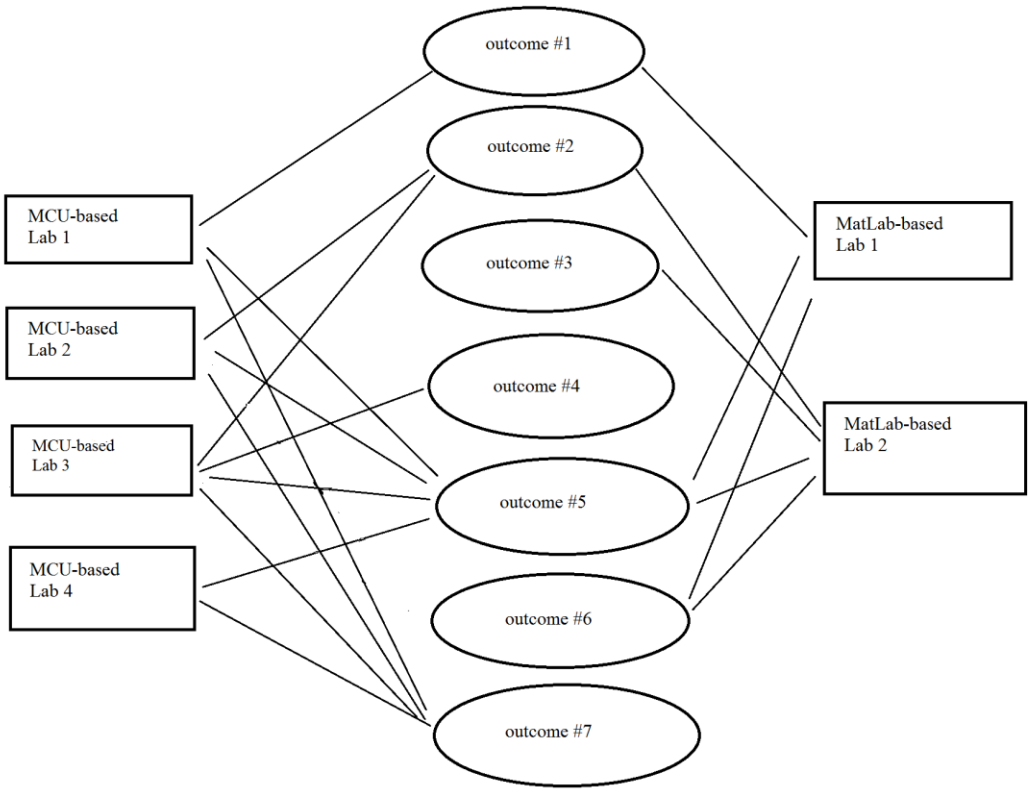

Figure 1: Connections among hands-on lab coursework and learning outcomes

III Review of Adopted MCU Platform and Software Tool for the MCU-based DSP Lab Coursework

As noted in our preliminary work[5], the hardware platform is the Freescale tower system with the Kinetis TWR-K65F180M board (ARM Cortex-M4 MCU) along with a custom-made CODEC tower board to support high-fidelity audio signal processing applications. The software platform is based on the KDS IDE with the ARM CMSIS DSP library.

Prior to taking the DSP class, students have worked with the hardware platform and the KDS IDE in three embedded microcontroller courses (listed in Table 1[5]) and are familiar with these tools. So there is no need to take additional training on how to use such tools, which eliminates a long learning process to learn a new IDE.

While the initial development work and the developed DSP laboratory coursework described below is using the TWR-K65F180M hardware, there are many other hardware options from NXP that can be programmed with the same KDS development tools. For example, we have now ported the original firmware to the FRDM-K66F development board, which is much smaller than the TWR-K65F180M and includes a CODEC on-board. We also have a version that uses the on-board ADC and DAC (physical circuits/chips to convert analog signals to digital formats or vise versa) in place of the CODEC (i.e., the custom-made CODEC board), which allows the students to be exposed to a more traditional ADC architecture that is better suited to demonstrate the effects of aliasing. These firmware are available to the students through KDS project imports and GIT repositories.

Students will need to be introduced with the ARM CMSIS DSP code library, which can be accomplished in manageable time and effort. As we have noted, using the selected platform and software development tool permits students in the DSP course to focus immediately on the DSP concepts without having to spend time learning how to use new hardware or software.

The developed MCU-based hands-on lab activities are carried out based on the aforementioned platform, software development tool, and firmware.

IV: Developed MCU-based Hands-on DSP Laboratory Coursework

Compared to the previous lab coursework that were carried out on the dedicated DSP platform, the MCU-based hands-on DSP laboratory experiments have been designed to accomplish a higher attainment level of the existing learning outcomes and the attainment of new learning outcomes (i.e., the seven outcomes in Table 2). In the following, we elaborate the detailed hands-on lab coursework.

Four lab experiments have been developed and their contents are closely aligned with the topics covered in lectures. The labs are also sequenced according to the schedules of these topics. They are aimed to I) enhance students understanding of the DSP theory and II) provide students opportunities to practice and implement real-time DSP algorithms by using real-life audio signals as the input such as music, human voice signals, or audible tones.

In Table 3, we highlight the pertaining topics and the scope of each lab experiment.

MCU-based Lab	**Concepts/Topics Related**	**Scope of the Lab Experiment**
#1	• FFT • spectrum analysis	Given a set of different input signals, implement a FFT algorithm to conduct spectrum analysis.
#2	• FIR filter • Frequency response	FIR filter design and implementation and frequency response analysis.
#3	• IIR filter • Filter implementation structures • Impact of fixed-point and floating -point	IIR filter design and implementation and frequency response analysis. Explore the impact of using fixed-point in IIR filtering. Compare the difference between fixed-point and floating-point IIR filtering.
#4	Real-time adaptive filtering	Integration of digital filtering and FFT to realize real-time adaptive filtering implementation based on input signal spectrum analysis. Generate in-line filter coefficients which would vary according to the changes of the input signal dynamics and frequency spectrum contents.

Table 3: Summary of the developed MCU-based lab coursework.

For comparison, the summary of the labs using the dedicated DSP platform (TMS320C6713DSK board) in previous DSP courses (offered prior to Spring 2016) is provided in Table 4.

Lab	**Concepts/Topics Related**	**Scope of the Lab Experiment**
#1	Introduction to TMS320C6713DSK board and the CCS IDE	The DSK board basics, CCS functionalities, programming tips
#2	FFT using CCS	Spectrum analysis of different input signals
#3	FIR filter Implementation Frequency response	FIR filter design and implementation and frequency response analysis.
#4	IIR filter Filter implementation structures Floating point operation	IIR filter design and implementation (floating point) and frequency response analysis.

Table 4: Summary of previous Lab coursework using the dedicated TMS320C6713DSK board in the DSP courses prior to Spring 2016

Although the number of lab experiments are the same, clearly, the developed MCU-based labs permit students to attain more content and practice DSP topics with more depth and complexity, e.g., the fixed-point IIR filtering in lab #3 and the adaptive filtering in lab #4 described in Table 3. As such, more learning outcomes have been achieved through these MCU-based hands-on lab coursework.

Next, we provide abbreviated details of each lab experiment.

MCU-based Lab #1:

Lab objectives: Familiarize students with the template source code and practice conducting signal spectrum analysis through FFT.

Lab tasks:

- Select/create a test input audio signal and conduct N-point FFT to analyze its spectrum; Identify the spectrum peak location and relative amplitude, verify that the spectrum result is correct by comparing to the actual input signal spectrum in theory.
- Study the impact of sampling frequency and the number of FFT points on the frequency spectrum and compare the results with the theory covered in lectures.

Major ARM CMSIS DSP library functions adopted:

- `arm_rfft_q31(),arm_cmplx_mag_q31(), arm_max_q31()`

MCU-based Lab #2:

Lab objectives: Implement fixed-point FIR filtering using the MCU platform and study the frequency response of the filter and the impact of filtering.

Lab tasks:

- Select/create a test input audio signal, design a FIR filter with any type such as low-pass, high-pass, band-pass or notch filter using MatLab or generated in-line inside the c code, implement the FIR filter and verify that the filtering has been successful.
- Analyze the filter frequency response.
- Conduct spectrum analysis of both the input and filtered signals.

Major ARM CMSIS DSP library functions adopted:

- `arm_fir_init_q31(), arm_fir_q31()`

Note that students were encouraged to adopt various methods to verify the filtering results, e.g., check the output waveform from an oscilloscope, check the frequency spectrum of the output, or use the Kinetis expression window or memory to check the values of certain variables.

MCU-based Lab #3:

Lab objectives: implement and compare fixed-point and float-point IIR filtering using the MCU platform and study the frequency response of the filter and the impact of filtering.

Lab tasks:

- Select/create a test input audio signal, design an IIR filter with any type such as low-pass, high-pass, band-pass or notch filter using MatLab or generated in-line inside the c code (if possible), implement the IIR filter using both fixed-point and floating-point and verify that the filtering has been successful.
- Analyze the impact of fixed-point IIR filtering due the scaling of filter coefficients, filter gain, input signal value, and overflow or saturation.
- Analyze the filter frequency response.
- Conduct spectrum analysis of both the input and filtered signals.

Major ARM CMSIS DSP library functions adopted:

- `Floating point IIR filtering: arm_biquad_casd_df1_inst_f32() arm_biquad_cascade_df1_init_f32() arm_biquad_cascade_df1_f32()`
- `Fixed-point IIR filtering: arm_biquad_casd_df1_inst_q31() arm_biquad_cascade_df1_init_q31() arm_biquad_cascade_df1_q31()`

MCU-based Lab #4:

Lab objectives: Implement real-time adaptive filtering by fusing spectrum analysis and filtering.

Lab tasks:

- Create a test input audio signal which consists of a desired signal portion and an additive single tone noise component, design a variable low pass FIR filter which changes its filter coefficients as the frequency of the tone varies, and implement the filter.
- Verify that the filtering has been successful and the noise component has been eliminated.

Major ARM CMSIS DSP library functions adopted:

- `arm_fir_init_q31(), arm_fir_q31(), and FFT functions`

Note that in lab #4, in order to accomplish the tasks, students will need to 1) identify where the single tone noise is located using FFT and 2) program their code so that it can automatically generate the filter coefficients rather than using the values pre-calculated in MatLab. As such, as the noise component changes, the code would generate the corresponding new filter coefficients in real-time to filter out the noise in a real-time and adaptive manner.

In the next Section, we will discuss some assessment results to further demonstrate the effectiveness of the developed MCU-based lab coursework.

V: Assessment Measures and Assessment Results from the DSP Course

To gauge the effectiveness of the selected MCU-based platform and the developed lab experiments compared to the traditional dedicated DSP hardware case, we have adopted the following assessment measures:

- Students' feedback in the form of survey questionnaires were collected from EE students who took the DSP class in Spring 2016. This group of students was the first group to use the MCU platform to conduct the hands-on lab experiments. Their feedback helped to assess how students feel about the effectiveness of the proposed MCU-based platform compared to the DSP-based platform. Student responses are listed in Table 5.

  There were thirteen students who responded the questionnaires. As shown from the students survey results, all students have acknowledged that the MCU platform is easy and convenient to use for conducting the DSP hands-on laboratory exercises. A majority of students feel that these hands-on lab activities have helped to enhance their understanding of the DSP topics covered in lectures, which might not be a good direct measure of student's understanding of topics. However, it shows a relatively high level of students' self-efficacy which can improve learning performance[9, 10].

  Students also supported the use of this platform for future DSP offerings except for one student who pointed out that the selected K65 board might be too powerful for most senior design projects. As noted in Section III, we are currently investigating a similar but smaller size MCU board (i.e., the FRDM-K66F development board) as the alternative platform for the DSP laboratory coursework. This board could be a better option for some senior design projects compared to the K65 board.

  Constructive suggestions were also offered by some students from their survey results. For instance, one student recommended to cover more details about the custom-made CODEC during lectures. Another hoped to know more about communications between the MCU and the CODEC module. Regarding the lab content, some suggested to include more challenging real-life applications such as voice recognition using the MCU-platform.

  All these suggestions will be considered in the future DSP courses.

- Moreover, we tracked students' performance in completing the DSP lab coursework, in particular Lab #3 and Lab #4.

We found that 100% of the teams finished Lab #3 about the fixed-point IIR filtering. The grades of the final exam (Question 1. (3)) on the impact of fixed-point on IIR filtering were also used as an assessment measure. 90% of students answered this problem correctly. The lab performance and the final exam grades indicated that the new learning outcome #4 has been attained at a satisfactory level.
Lab #4 performance showed that students were able to practice more complex real-time DSP algorithms to solve more dynamic problems. Out of the nine teams of the entire class, 7 teams accomplished the required tasks. Two teams took the challenge to a new level and implemented an arbitrary FIR filter in their code which can produce various types of filters such as low-pass, high-pass, and so on depending on the spectrum features of the noise component of the input signal. Being able to complete this lab along with the first three MCU-based hands-on lab experiments testifies that students have attained learning outcome #7.

VI: Senior Capstone Project Improvement Goals and Assessment Measures

As noted in our preliminary work[5], the capstone project courses are designed to meet the ABET EAC[8] Criterion 5 and student outcome (c) and, to assess the effectiveness of this work, we focus on the following outcome:

1. Students are able to successfully complete a design project that uses DSP to meet realistic constraints.

The assessment of this outcome showed that between 2011 and 2015 there were 18 student designs that should have used DSP. Of the 18, five used dedicated DSP hardware, two used MCU-based hardware, and 11 avoided using DSP by using analog signal processing. Only three of the student designs met realistic constraints and only two of these three successfully completed their project[5].

The previous analysis concluded that most students in the past five years that should have used DSP to meet realistic constraints either avoided projects that required DSP, did not meet the constraints, or did not successfully complete their project because of their limited laboratory experience in the DSP course.

We now have had one cohort of seniors that did take the DSP course using the new MCU-based hardware platform. The group will not complete their projects until the end of Spring quarter 2017 so successful completion is yet to be determined. However, we did assess the system-level design decisions made by these students. There were a total of 16 senior projects. Out of those, five of the projects should have used MCU-based DSP to meet realistic constraints. This year all five of the seniors chose to use MCU-based DSP over dedicated DSP hardware or analog signal processing. When describing the DSP design choice, one student wrote “They will mostly be

simple FFT calculations". In the past, no student would have thought that FFT calculations using MCU-based hardware was simple.

This is a promising result and indicates that students are more likely to use DSP in their projects now. It seems reasonable to conclude that students would be more inclined to include DSP in their projects given their DSP laboratory coursework experience using the selected MCU platform.

At the end of Spring 2017 quarter, we will be assessing the successful completion for these student projects to compare with previous results.

VII. Conclusions

In this paper, we present a series of MCU-based hands-on DSP lab coursework. These lab experiments are designed based on the Kinetis TWR-K65F180M board with ARM Cortex-M4 MCU and the KDS software development tool. Prior to taking the DSP course, students in our program have been familiar with the MCU and the software in three other embedded system courses. The developed lab coursework has tied closely to the topics covered in the DSP course and facilitated the attainment of desired learning outcomes. The assessment results based on student's feedback, lab performance, and final exam grades have shown that these hands-on lab experiments are beneficial and help to achieve a higher level of attainment of desired learning outcomes. Compared to the previous adopted dedicated DSP platform, the MCU-based platform permits the attainment of several new learning outcomes. This advantage comes from the fact that significant amount of time and effort have been saved from learning new hardware platform and corresponding software development tool, as such, students are able to focus more on the lab tasks and course content with minimal training needed on the tools.

Questions	**Agree**	**Disagree**	**Additional Comments**
Do you feel that the time spent to familiar yourself with the k65 tower is minimal and acceptable?	100% (13)		
Do you feel that hands-on lab exercises enhance your understanding of DSP topics such as frequency spectrum analysis and digital filtering?	85% (11)	15% (2)	- "Some of the labs can be better structured and can be more challenging and extensive" - "More DSP practical applications/projects"
Do you feel that hands-on labs facilitate your understanding of the impact of fixed-point on filtering?	85% (11)	15% (2)	
	Yes	**No**	**Additional Comments**

Did you run into problems due to unfamiliarity of the hardware platform (i.e., K65 tower) when working on DSP labs?	100% (13)	0% (0)	
Would you suggest using the K65 tower system as the platform for future DSP classes?	92% (12)	8% (1)	- "Good to see that the K65 board can do DSP" - "The board is too powerful for most senior design projects that students do"

Table 5: Student survey results from the Spring 2016 DSP class.

References

1. Cameron H.G. Wright, Thad B. Welch, and Michael G. Morrow, "Real-time DSP using "See-Through.", Computers in Education Journal, Vol 6, No. 2, April – June 2015, pp 31-39.
2. Aditya Johri and Barbara M. Olds, "Situated Engineering Learning: Bridging Engineering Education Research and the Learning Sciences", Journal of Engineering Education 100.1 (2011): 151-185.
3. Jennifer Round and Barbara Lom, "*In Situ* Teaching: Fusing Labs & Lectures in Undergraduate Science Courses to Enhance Immersion in Scientific Research", Journal of Undergrad Neuroscience Education, 2015, Summer, 13(3): A206 – A214.
4. Donald S. Reay, "Digital Signal Processing Using the ARM Cortex-M4", John Wiley & Sons, 2016.
5. Todd Morton and Ying Lin, "Work-in-Progress: An Integrated DSP and Embedded Microcontroller Laboratory Curriculum", Proceedings of ASEE Annual Conference, 2016.
6. Murat Pasa Uysal, "Evaluation of Learning Environments for Object-oriented Programming: Measuring Cognitive Load with a Novel Measurement Technique", Journal of Interactive Learning Environments, 2016, Vol 24, Issue 7, pp 1590-1609.
7. Leon Adams, "Choosing the Right Architecture for Real-Time Signal Processing Designs", White Paper, Texas Instruments, SPRA879, November 2002.
8. ABET, "Criteria for Accrediting Engineering Programs", 2016 – 2017.
9. Jacquelynne S. Eccles and Allan Wigfield, "Motivational Beliefs, Values, and Goals", Annual Review of Psychology, 2002, Vol 53: 109-132.
10. Albert Bandura, "Self-efficacy Mechanisms in Human Agency", American Psychologist, 37, 122-147.