AC 2012-4807: UTILIZING A SYSTEM-ON-CHIP PROJECT AS A CAPSTONE EXPERIENCE IN A MICROPROCESSORS COURSE

Prof. Scott James Schneider, University of Dayton

Scott J. Schneider is an Associate Professor of electronic and computer engineering technology at the University of Dayton. Schneider has extensive industrial experience in the areas of software development, embedded real-time system design, and automotive technologies. He also performs research in the areas of computer and software engineering pedagogy.

Mr. Seth Jarek Peterson, University of Dayton

Seth J. Peterson is a fifth-year undergraduate student of electronic and computer engineering technology at the University of Dayton. Peterson has experience in the areas of software development and embedded hardware design. He enjoys enhancing the capabilities and functionality of both hardware and software in his spare time.

Utilizing a System-on-Chip Project as a Capstone Experience in a Microprocessors Course

Abstract

In an introductory microprocessors course, students utilize a Field Programmable Gate Array (FPGA) and the Verilog Hardware Description Language (HDL) to study microprocessors. Students utilize both structural and behavioral Verilog code to develop the fundamental building blocks of a processor, culminating in the design and implementation of a simple ALU including an instruction memory and program counter. Once they have studied the fundamental internal architecture of a microprocessor, the students then employ a small 8-bit embedded processor through a system-on-chip design process. Working with the embedded processor allows the students to study the instruction set architecture (ISA) of a microprocessor utilizing assembly language programming.

This paper focuses primarily on the capstone design project that the students complete within this microprocessor course. This project requires the students to develop a complete microprocessor system for audio recording and playback utilizing the embedded processor and any necessary interface hardware. Therefore, to complete the project each student is responsible for both hardware and software development; a true co-design experience. During this process, the students must also determine how each subsystem will be tested in support of the final project. The developed microprocessor system must include interfaces for analog-to-digital conversion for audio recording, data memory for audio file storage, and digital-to-analog conversion for audio playback. Additionally, the system must include a user interface for performing the record and playback operations. The complete system requirements will be detailed in this paper along with an overview of one student team's design solution.

Introduction

An embedded system is a computer system that has a dedicated, specific function typically operating within a non-computational system. Often, embedded systems must perform real-time tasks, may or may not utilize a software operating system and have stringent cost and performance constraints. As a result, embedded systems often are resource limited, having less processing power and a smaller amount of memory than general purpose computing systems. Embedded systems are prevalent in a variety of industries including the automotive, aerospace, and consumer electronic markets.[1]

The growing need for educating electrical and computer engineering students in the area of embedded system design has been well documented.[2] This need, along with strong support of the Industrial Advisory Committee of the Electronics and Computer Engineering Technology Program (ECT) at the University of Dayton (UD), has led to the redevelopment of several computer engineering technology courses and to a new course sequencing to have a stronger embedded systems focus. The revised course sequence and major associated embedded system topics are outlined in Figure 1. Note that these courses are all required for the ECT major.

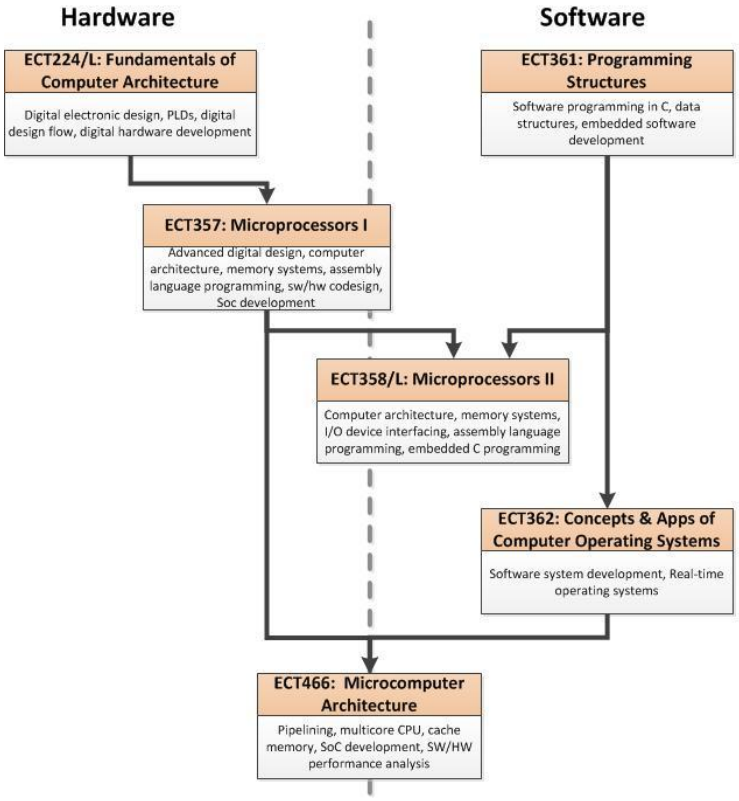

Figure 1: Embedded systems curriculum

Within the course sequence, system-on-chip (SoC) technology has become an important focus, being studied in multiple courses. Overall, there has been a movement towards utilizing SoC technology in embedded devices throughout industry and within engineering programs.[3, 4, 5] This movement follows an earlier trend of using programmable logic devices (PLDs) in place of fixed function logic integrated circuits (ICs).[6, 7, 8] A SoC can be defined as a single IC which contains multiple discrete components that are integrated into a single cohesive system. Often, these components are provided by vendors as validated intellectual property (IP) cores that can be combined together with an engineer's unique design in a similar fashion to how individual ICs are combined on a printed circuit board to complete a circuit. These IP cores are often referred to as soft-cores since they are implemented from Hardware Descriptive Language (HDL) code and not hard wired into the silicon of the IC. Having these components combined within a PLD provides the engineer a faster and more flexible development experience. SoC development can also provide an increase in system performance and reliability since the system resides completely within the constrained PLD environment while also reducing the manufacturing costs over using fixed function ICs.[9, 10]

As an example, a SoC for an embedded device may contain IP cores for one or multiple processing units, serial and parallel communication interfaces and memory systems. In addition to the ability to use soft-core IPs, many modern PLDs also contain hard-core IP for processors, memory systems and serial communications that are fixed within their silicon providing even better performance. Both soft-core and hard-core microprocessors are used within the Electronic and Computer Engineering Technology Program at UD.

Microprocessors 1 Course Overview

In the embedded systems course sequence at UD, the Microprocessors I course is utilized as an advanced digital design course with an emphasis on understanding the fundamental building blocks of a microprocessor. Also within this course, students begin to learn about and work within the hardware and software co-design process as they develop with a soft-core microprocessor core. This course is taken during their fourth semester as a required ECT course and students are expected to already have a strong background in digital electronics design and to be familiar with PLDs.

This course utilizes Field Programmable Gate Arrays (FPGAs) and the Verilog HDL. The student exercises are performed using the Digilent BASYS2 FPGA development boards, pictured in Figure 2, and the Xilinx WebPACK development environment.[11, 12] The BASYS2 board uses the Xilinx Spartan 3E-100 FPGA coupled to multiple input/output devices including a USB port, a VGA port, a PS/2 port, multiple LEDs, pushbuttons and switches as well as 4 peripheral module expansion slots. Each student must purchase a BASYS2 board as most of the homework assignments are implementation based exercises that must be demonstrated to the instructor for full credit. Note that the BASYS2 board is also required in the prerequisite course, ECT224, Digital Computer Fundamentals.

Figure 2: The BASYS2 FPGA development board

During the first part of the course, students learn advanced digital design techniques using the Verilog HDL with combinational and sequential logic. Next, the students learn about advanced arithmetic circuits, the Arithmetic Logic Unit (ALU) and memory storage technologies and systems. Associated with these lessons is a sequence of three assignments that build upon each other in the creation of a simple 4-bit ALU and processor system. This ALU incorporates a simple addressable memory system and introduces students to an instruction set and the instruction cycle. The complete block diagram for the simple ALU system is shown in Figure 3.

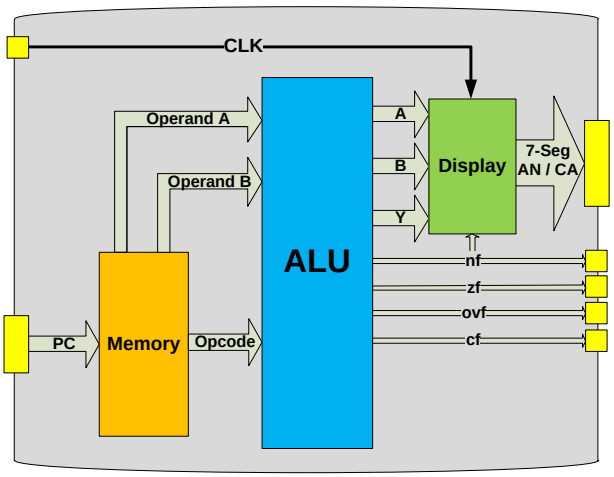

Figure 3: Final 4-bit ALU system block diagram

After studying the internal architecture of a microprocessor, students are introduced to the PicoBlaze 8-bit soft-core processor. The PicoBlaze processor is an 8-bit RISC processor that was designed specifically for the Xilinx FPGAs. Some features of the PicoBlaze include a 1k-word instruction memory, a 64-Byte scratchpad RAM, support for 256 input and 256 output peripherals, operational speeds up to 200 MHz (100MIPS) and support for pipelining and interrupts for improved performance and responsiveness.[13] A complete system diagram of the PicoBlaze processor is shown in Figure 4. Given that the PicoBlaze has a simple architecture and only 57 instructions, it is easy for students to quickly learn how to implement and interact with it within the FPGA. Furthermore, challenging topics such as interrupts and addressing are easily demonstrated within this streamlined processing structure. Software programs for the PicoBlaze are written in assembly language. The Mediatronix pBlazeIDE integrated development environment (IDE) can be used for software development with the PicoBlaze; however, in this class, the KCPSM3 batch assembler is used.

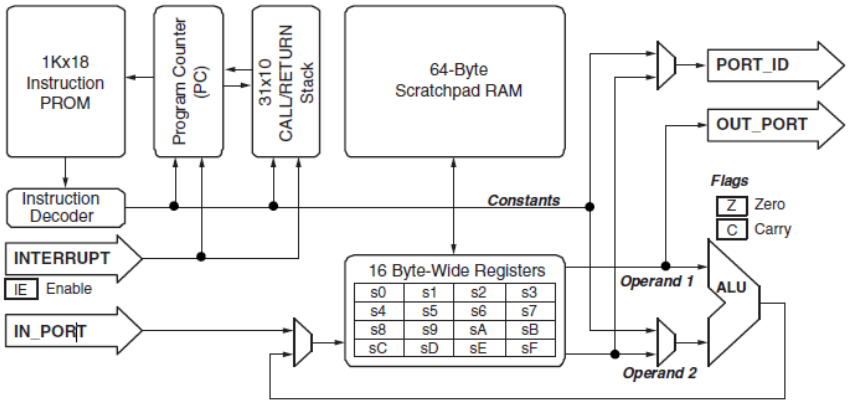

Figure 4: PicoBlaze soft-core microprocessor block diagram

The final topics covered in the class are parallel and serial communication interfaces including the Personal System/2 (PS/2) and Serial Peripheral Interface (SPI) interfaces, and data conversion and signal processing techniques. The course culminates with a cumulative capstone design project. The complete course topic and assignment list is shown in Table 1.

Topic	Assignment
LSN1 - Digital Design Flow	
LSN2 - Intro to Verilog HDL	Code development exercises
LSN3 - Combinational Circuit Applications	Computer-based logic minimization
LSN4 - Behavioral Models in Verilog	Timing analysis and code development
LSN5 - Arithmetic Circuits and the ALU	Simple 4-bit ALU
LSN6 - Sequential Circuit Applications	Add I/O to ALU
LSN7 - Registers and Memory Systems	Develop ISA and add memory to ALU
LSN8 - Microprocessor Architecture	
LSN9 - Embedded Soft Core Processors	PicoBlaze HW/SW co-design application
LSN10 - Communication Interfaces	Interfacing SPI memory
LSN11 - Data Conversion & Signal Processing	Interfacing ADC and DAC ICs
Capstone Design Project	Develop an audio recorder

Table 1: Microprocessors I course topic list

Capstone Embedded System-On-Chip Project

The students are tasked to complete a capstone design project during the last four weeks of the Microprocessor I course. This project requires students to integrate many of the concepts and assignments covered throughout the class into a final SoC system. Every student must develop a simple audio recording, storage, and playback device. The decision was made to make the final project related to audio since a majority of students own personal digital audio devices and therefore, such a project would be relevant to the students and not simply academic in nature. Furthermore, previous offerings of this course had allowed the students to select their own final project topic and a majority of those projects were related to music or audio in some manner.

The students work in groups of 2 to 3 students where each team member is expected to maintain their own lab notes for the project. These notes along with a report detailing the team's design and analysis, the correctness and professionalism of the Verilog and Assembly code developed and a demonstration of their working prototype are all used in calculating the final project grade. Each student's final project grade is also evaluated by their team members as a peer evaluation that acts as a multiplier of the team's overall project grade. The complete project grade distribution is documented in Table 2.

	Grading Criteria	% of Grade
Team Grade	Report Professionalism	5%
	System overview (block diagram)	10%
	Hardware design and analysis discussion	23%
	HDL code correctness and professionalism	17%
	Software design and analysis discussion	20%
	Assembly code correctness and professionalism	15%
	Prototype demonstration	5%
Individual Grade	Laboratory notes	5%
	Project Total	100%
	Individual grade = Project Total * Average Peer Evaluation	

Table 2: Project grade evaluation

The final embedded SoC project uses the Verilog HDL, the BASYS2 development board, and the Xilinx PicoBlaze soft-core microprocessor. The problem statement provided to the students states that the system should be able to record audio, store it in a memory structure and play the recorded audio back upon user request. The PicoBlaze microprocessor is to be used as the system master to handle the interaction with and between the peripheral devices. A block diagram of the desired embedded audio system is provided to the students and is shown in Figure 5. Note that the location of the data storage component, internal or external to the FPGA, was left to the discretion of the students.

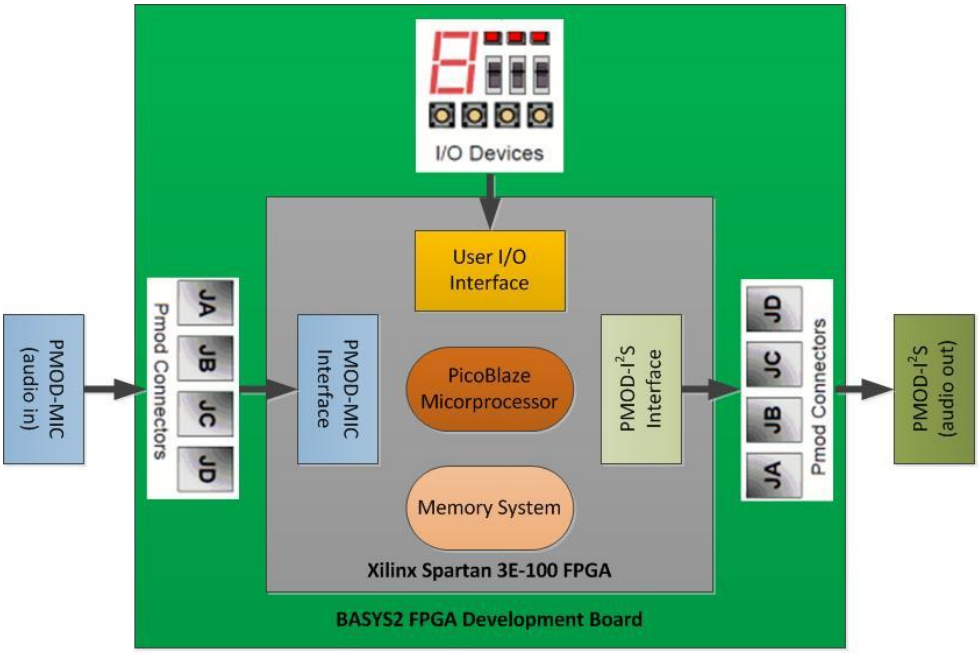

Figure 5: Embedded audio system block diagram

The Digilent BASYS2 board, pictured in Figure 2, contains four parallel peripheral module connectors (PMOD) that may be used to interface the audio input and output modules. The Digilent PMOD-MIC peripheral board is used for the audio input for the system.[14] The PMOD-MIC uses the National Semiconductor ADCS7476AIM 12 bit A/D converter which is interfaced using the SPI protocol. The Digilent PMOD-I^{2}S peripheral board is used for the audio output for the system.[15] The PMOD-I^{2}S uses the Cirrus Logic CS4344 Stereo D/A converter which is interfaced using a variant of the I^{2}C protocol. The PMOD-MIC and PMOD-I^{2}S peripheral devices are pictured in Figure 6.

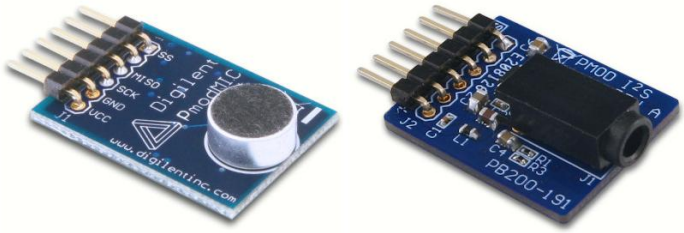

Figure 6: Digilent PMOD-MIC and PMOD-I^{2}S peripheral devices

The details of what components or operations are going to be handled in the software running on the PicoBlaze microprocessor or implemented in hardware using Verilog code is left up to the student teams since this is a true hardware and software co-development project. They are expected to utilize knowledge gained throughout the semester in making their design decisions. Table 3 maps the various design details in the project to the associated course material and assignments. In designing the audio input and output component the students must calculate the audio sampling rate which will also have an impact on the size and speed of the memory system selected. The user interface for the audio system must make use of the pushbuttons, switches, LEDs and seven-segment displays

Project Design Component	Associated Course Material & Assignments
Hardware/Software co-design	LSN1, LSN9 and PicoBlaze assignment
PicoBlaze microprocessor	LSN8, LSN9 and PicoBlaze assignment
Audio input and output	LSN9, LSN10 and associated assignments
Memory system	LSN7 and memory system assignment for ALU
User interface	LSN3, LSN6 and I/O interface project for ALU

Table 3: Mapping of project component to course material and assignments

Overview of Student Project

To highlight the types of solutions that the students in the Microprocessors I course develop for the embedded audio system, a brief overview of one such solution is provided here. The audio system developed by this student team is shown in Figure 7. This solution provides just one possible solution that uses internal block RAM on the Spartan 3e FPGA for the memory system and implements the input/output interfacing within hardware instead of in software running on the PicoBlaze. This solution uses 8-bit audio sampled at 22 kHz and utilizes internal hard-core dual port block RAM modules for the audio storage.

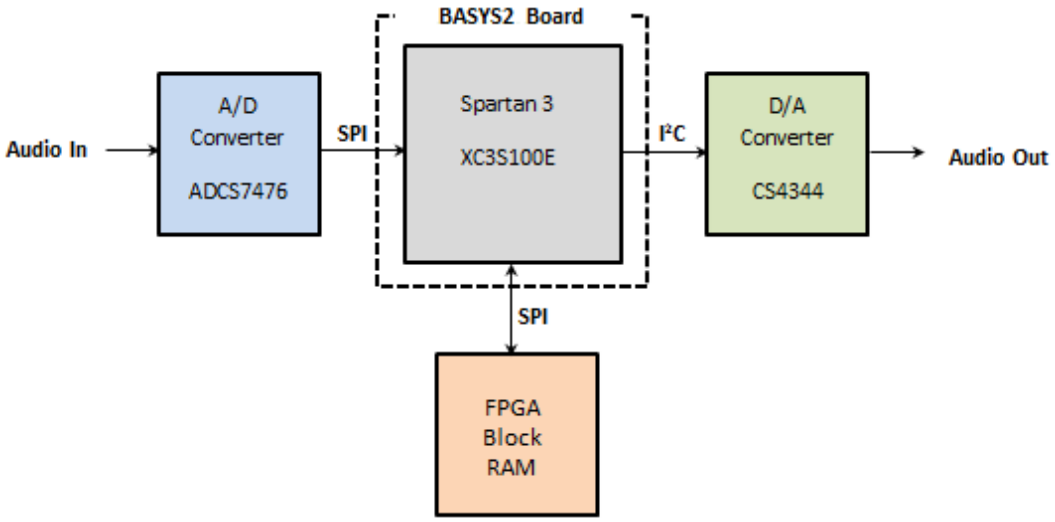

Figure 7: Block diagram of student developed embedded audio system

The I^2C interface module used to interface the D/A converter was provided to the team by the course instructor since the students had not worked with this communication protocol yet. The student team was, however, responsible for correctly interfacing this I^2C module within their system and for implementing the SPI communication interface to the A/D converter. A diagram of their functional layout is shown in Figure 8. Note that the students had to develop a custom hardware interface to control the flow of audio data between the SPI interface module, the I2C interface module and the Block RAM. The PicoBlaze was responsible for these data-path control operations.

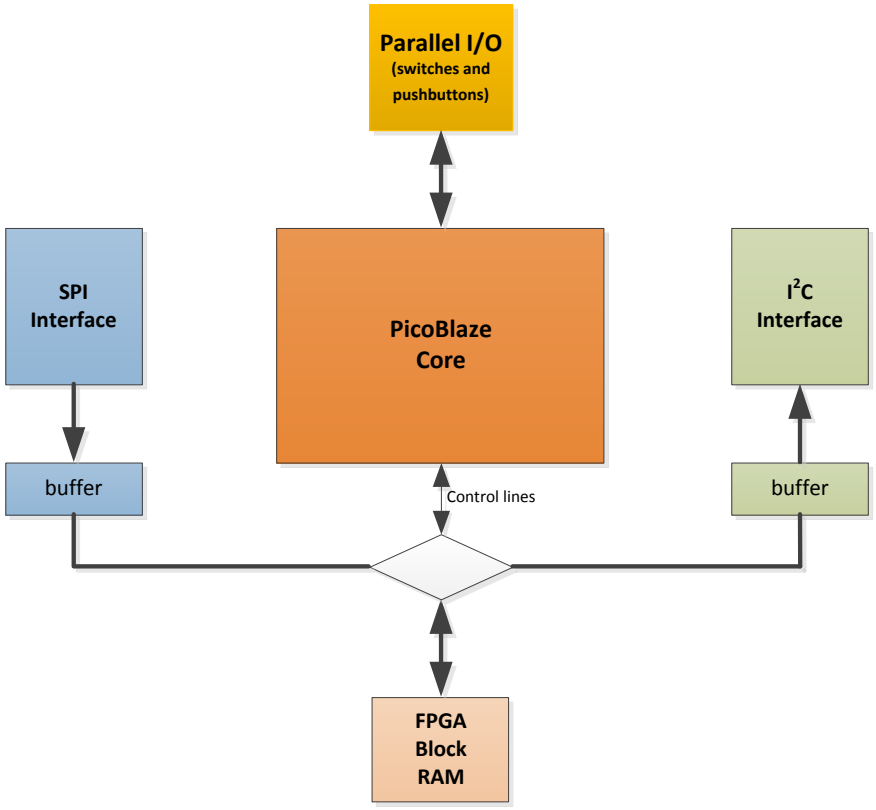

Figure 8: Software functional diagram for PicoBlaze

The students developed a software flowchart for interfacing the user controls and for the system-level state machine. Furthermore, all external interfaces where tested using both simulation and hardware analysis techniques with the results documented within the design report. For example, to validate the SPI communications interface module, an oscilloscope plot was captured and is shown in Figure 9.

This student design project provided a unique solution which demonstrated for the students the real-world tradeoffs and challenges when working with SoC design, especially in determining the hardware and software architectures. The students realized some of the limitations associated with using only the on-chip block RAM modules and the impact these made on the audio quality. Likewise, the students only decided to implement the input\output interfaces in hardware after an initial attempt to do so in software proved more challenging than initially anticipated, especially with respect to testing and debugging.

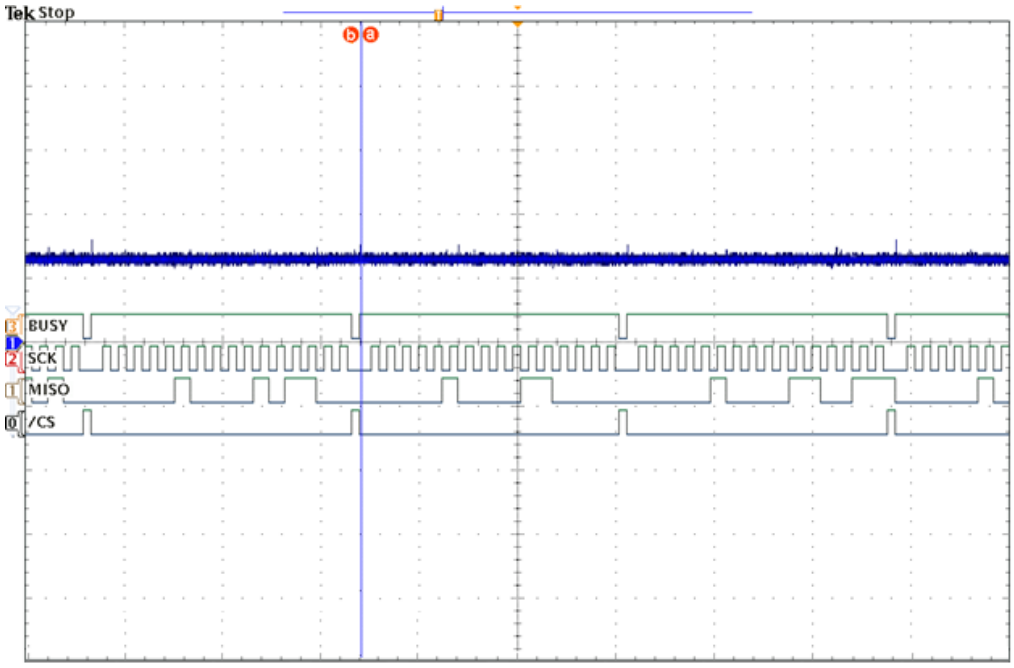

Figure 9: Oscilloscope capture of the A/D converter's SPI communications

Conclusion

Embedded system design is an important and evolving discipline within electrical and computer engineering and, as such, is playing a more important role within engineering curriculums. In the electronic and computer engineering technology program at UD, a major emphasis has been placed upon embedded system design, including the use of SoC development within multiple courses. In the Microprocessors I course, SoC development has been used to provide students exposure to the hardware/software co-design process. Using a cumulative design project related to audio has provided relevance to the course material and a challenging real-world design platform. The example overview of a student design project provided a glimpse into some of the design issues that the students face during this capstone project and how they overcome them.

Bibliography

1. Berger, A., "Embedded Systems Design," CMP Books, 2002.
2. Anderson, M., "Help Wanted: Embedded Engineers, Why the United States is losing its edge in embedded systems…," *IEEE-USA Today's Engineer Online*, February 2008. Available: http://www.todaysengineer.org/2008/Feb/help-wanted.asp
3. Nagvajara, P. and Kizirian, R., "Design Projects for Programmable Embedded System-On-Chip Course," *Proc. 2011 ASEE Annual Conference & Exhibition*, Vancouver, BC Canada, June 2011.
4. Hall, T.S., Hamblen, J.O., "System-on-a-programmable-chip Development Platforms in the Classroom," *IEEE Transactions on Education*, Nov 2004, pp502-507.

5. Slivosky, L. and Liddicoat, A., "Transforming the Microprocessor Class: Expanding Learning Objectives with Soft Core Processors," *Proc. 2007 ASEE Annual Conference & Exhibition*, Honolulu, HI, June 2007.
6. Nickels, K., "Pros and Cons of Replacing Discrete Logic in Introductory Digital Logic Courses," *Proc. 2000 ASEE Annual Conference & Exhibition*, St. Louis, MO, June 2000.
7. Nickels, K., Aminian, F. and Giolma, J.P., "Bridging the Gap Between Discrete and Programmable Logic in Introductory Digital Logic Laboratories," *Proc. 2001 ASEE Annual Conference & Exhibition*, Albuquerque, NM, June 2001.
8. "The Advantages of Migrating from Discrete 7400 Logic Devices to CPLDs," Xilinx whitepaper #202 (v1.3), January 2005.
9. "Comparing and Contrasting FPGA and Microprocessor System Design and Development," Xilinx whitepaper #213, July 2004.
10. "Field Programmable Controllers for Cost Sensitive Applications," Xilinx whitepaper #167, December 2002.
11. Digilent BASYS2 Board Reference Manual (11/11/10) [Online]. Available: http://digilentinc.com/Data/Products /BASYS2/Basys2_rm.pdf
12. Xilinx ISE Suite Software Manuals and Help (12/14/10)[Online]. Available: http://www.xilinx.com/support/documentation/sw_manuals/xilinx12_4/manuals.pdf
13. PicoBlaze 8-bit Embedded Microcontroller User Guide (6/22/11) [Online]. Available: http://www.xilinx.com/support/documentation/ip_documentation/ug129.pdf
14. Digilent PMOD-MIC datasheet (4/20/09) [Online]. Available: http://digilentinc.com/Products/Detail.cfm?NavPath=2,401,517&Prod=PMOD-MIC
15. Digilent PMOD-I2S datasheet (12/21/10) [Online]. http://digilentinc.com/Data/Products/PMOD-I2S/PmodI2S_rm.pdf
16. Digilent PMOD-SF datasheet (9/26/06) [Online]. http://digilentinc.com/Data/Products/PMOD-SF/PMod_SF_%20rm.pdf