Design and Verification of an Application Specific Integrated Circuit (ASIC)

Senior Project II



Kevin Cao, Whitley Forman, Dhruvit Naik,

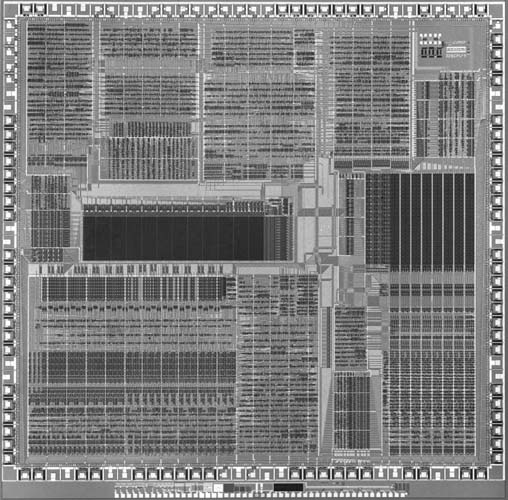
Zachary Nelson\* and Julie Swift

\*Team Leader

Dr. Orlando Hernandez and Dr. Larry Pearlstein

May 2016

**Fulfillment Page**



Submitted to the Faculty of the Electrical and Computer Engineering Department

of The College of New Jersey

Written and Researched by:

Kevin Cao

Whitley Forman

Dhruvit Naik

Zachary Nelson

Julie Swift

In partial fulfillment of the Bachelor of Science degree in Electrical and Computer Engineering

**Acknowledgements**

The team would like to acknowledge Ian Patel’s work on helping to set up the realistic FPGA test environment. Ian wrote the PSoC microcontroller code to generate an I2S audio stream and capture it back from the FPGA. He also wrote Matlab code that produced different types of waveforms in I2S format. We would also like to thank MOSIS for sponsoring the chip fabrication aspect of the project through the MOSIS Educational Program (MEP).

**Abstract**

The goal of this project is to gain experience in VLSI design by designing a chip that will process digital streaming audio data. More specifically, we will implement a 512-tap digital finite impulse response (FIR) filter, which will be applied to an input stream in order to create an output stream. We used the I2C protocol to allow a host to control the chip and the serial I2S protocol for transferring digital audio streams in and out.

Our hardware design was represented using Verilog register-transfer level (RTL) code. Development has been done using Xilinx ISE Design Suite 14.7. Test-benches were also designed and implemented using Verilog. The end goal of the project is to implement the design on a field-programmable gate array (FPGA). We will bring up our FPGA design with a realistic environment including an audio source, audio sink, and a microcontroller for reading and writing registers. We also plan to send a simple CMOS integrated circuit design for fabrication by MOSIS that will help us gain experience in physical chip design and prepare future groups to fabricate our full design.

**Keywords:** Application-specific integrated circuit (ASIC), Very large scale integration (VLSI) I2S, I2C, Digital filtering

**Table of Contents**

List of Tables 5

List of Illustrations 6

Nomenclature 7

Introduction 8

Specifications 10

Chapter 1: Background 11

Chapter 2: System Design 13

Chapter 3: I2S Interface 16

Chapter 4: Digital Filtering 26

Chapter 5: Register Block 33

Chapter 6: I2C Interface 41

Chapter 7: Simulation/Verification 11

Chapter 8: FPGA Implementation 11

Chapter 9: EDA Tools and Physical Design 11

Chapter 10: Conclusion 57

References 58

Appendix A: Project Overview 59

Biography 59

Engineering Standards and Realistic Constraints Form 59

Engineering Standards, Specifications, and Codes 59

Modern Engineering Tools 59

Appendix B: Management 61

Schedule 59

Meeting Minutes 59

List of Contacts 59

Material List 59

Financial Budget 59

Appendix C: Source Code 81

Appendix D: Test Benches 125

Appendix E: Industry Specifications 125

**List of Tables**

Table 1.1: Comparison of MOSIS Academic Account Types 12

Table 2.1: Interface Signals for I2S Input ??

Table 8.1: Verilog Signals to FPGA Mapping ??

**List of Illustrations**

Figure I.1: Process of Designing a System on a Chip 8

Figure 1.1: Multi-Project Wafer 11

Figure 2.1: Top-Level Drawing of Chip 13

Figure 2.2: Detailed Top-Level Drawing of Chip 14

Figure 2.3: FPGA Testing Environment 15

**Nomenclature**

**ASIC**: Application Specific Integrated Chip

**BIST**: Built in Self-Test

**CIF**: Caltech Intermediate Form

**DIP**: Dual in-line package

**EDA**: Electronic Design Automation

**FPGA**: Field Programmable Gate Architecture

**FSM**: Finite-State Machine

**GDSII**: Graphic Data System II

**I2C**: Inter-Integrated Circuit

**I2S**: Integrated Interchip Sound

**IC**: Integrated Circuit

**LSB**: Least Significant Bit

**MEP**: MOSIS Educational Program

**MPW**: Multi-Project Wafer

**MSB**: Most Significant Bit

**OCP**: Open Cavity Plastic

**PCB**: Printed Circuit Board

**RO**: Read Only

**RTL**: Register-Transfer Level

**RTR**: Ready to Receive

**RTS**: Ready to Send

**RW**: Read/Write

**WO**: Write Only

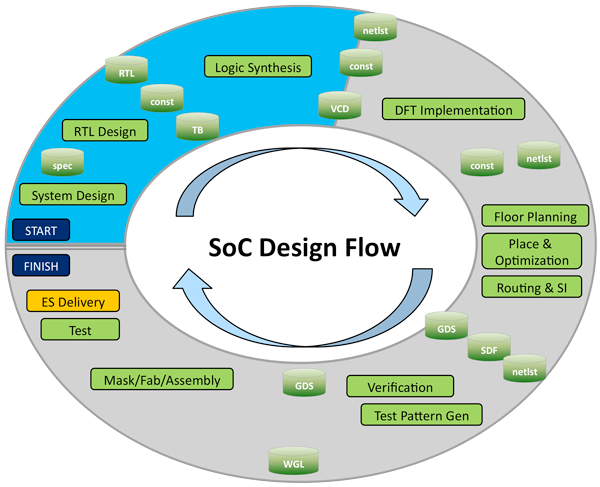
**XFC**: Transfer Complete

**VLSI:** Very Large Scale Integration

**Introduction**

The goal of this project is to design a chip that is capable of processing digital streaming audio data and apply a 512-tap filter to the input stream. The design was implemented and tested on a Nexys 4 Artix-7 FPGA board. We also submitted a simple circuit to MOSIS for fabrication in order to gain experience with using EDA tools. Two senior project groups for next year will be using certain aspects of our project. One group will be taking the Verilog code that we designed and tested and will fabricate a chip out of it. This group can also use the information that we recorded about using the EDA tools. The other group will be designing a RISC processor that may reuse some of the code that we have written. It should be noted that the terms VLSI and ASIC will be used interchangeably throughout the report and refer to designing the chip.

In industry, some of the advantages of an FPGA are it can be reprogrammed, there is no manufacturing involved, and it is as simple as just downloading code onto a board. Some of the advantages of an ASIC include being a fully custom design so that the device is manufactured to the designer’s specifications, a lower unit cost when producing in high volume, a smaller chip since there are no unwanted components included, higher clock rates, and lower power dissipation. We chose our application to be audio processing because digital audio filtering has a wide range of real world applications. The chip’s application is relatively simple because the bulk of the time was spent going through the chip design process since this is the first chip any of the students have designed.



**Fig I.1:** Process of Designing a System on a Chip

Even though we will not be following every step, Fig. I.1 shows the general process for designing a chip. The first task is system design where the requirements of the system are documented and the major functionality of the chip is described. The second task is RTL design which involves writing code in a hardware description languages (HDL) such as Verilog in order to create a high-level representation of the chip. The next step is logic synthesis which involves turning the RTL design into a netlist of logic gates. Floor planning involves determining the location of the major blocks on the IC schematic. Place and optimization is placing all of the electronic components, circuitry, and logic elements into a limited amount of space in an effective manner. Routing is connecting all of the different components and must follow the rules and limitations of the fabrication process. This step is mostly done automatically by an EDA tool but sometimes has to be done manually. It should be noted that testing and verification will be performed throughout the entire process. The last step is submitting a verified design to be fabricated (commonly in the form of a GSDII file).

**Specifications**

**General:**

* Maximum Clock Rate: 100 MHz
* Clock frequency will be a minimum of 1200 times the audio sampling rate

**I2S:**

* I2S input and output interfaces comply with I2S standard, included here in Appendix E
* Maximum Serial Clock Rate: 1.44 MHz
* Audio input sample rates ranges of 8 kilosamples/sec - 48 kilosamples/sec
* Digital audio bit clock and word select lines will be controlled from I2S master, which can be our chip, or the external I2S source
* Audio input and output must have same sample rate – our chip will be the timebase master on the output I2S interface
* Audio input and output will be two 16 bit channels (i.e. one stereo pair)

**Filter:**

* Filter Design: FIR Filter
* 512-tap filter, 16-bit coefficients, all independently settable
* Programmable filter coefficients to achieve different filter types
* Maintain integer headroom of 4 bits

**I2C:**

* 8-bit slave address space
* 12-bit register address space
* Data transfer rate of up to 400 kbits/second desired (both I2C standard and fast modes supported)
* Slave only capability
* Write operation (Burst write functionality)
* Read operation (Single read, optional burst read request)
* User selectable slave address with strap pins
* Simple strobe interface to register block (read and write)

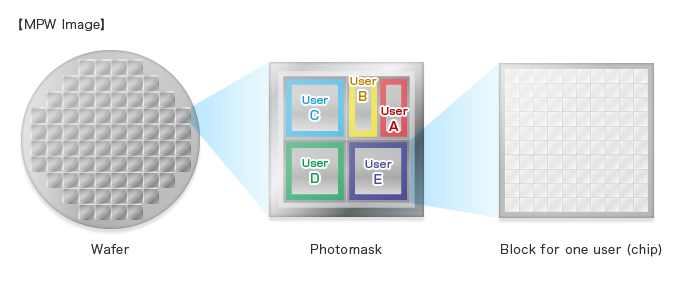
**Register:**

* Simple strobe interface to I2C block
* Provides read/write access to control/status registers, including the following
  + Source select bit (I2S vs. BIST)
  + Filter bypass bit (pass through or filter input streams)
  + 512 16-bit filter coefficients
  + Overflow/saturation detector
  + Audio FIFO overrun/underrun

**Chapter 1: Background – Z. Nelson**

*1.1 Who is MOSIS?*

The simple chip that we are designing will be produced by MOSIS, at no cost to The College of New Jersey. MOSIS was the first well-known multi-project wafer (MPW) service, and was established by DARPA (Defense Advanced Research Projects Agency) in 1981. The acronym MOSIS stands for Metal Oxide Semiconductor Implementation Service and the company has processed over 60,000 IC designs over the last 30 years [1]. The MOSIS Service is known for MPWs, which is how they are able to keep the cost of fabrication low. A MPW is when multiple IC designs are shared on a single wafer and an illustration of this concept in shown in Figure 1.1. This means that designs from private companies and students could be on the same wafer. The reasoning behind this approach is that the cost of fabrication can be kept at a reasonable price if the cost of mask making, wafer fabrication and assembly are shared throughout multiple projects. This idea of a MPW is also attractive because designers can create a prototype of their design without making a huge investment. It should also be noted that MOSIS offers using a single project for a wafer (dedicated run) for all processes and can start the fabrication at any time.



**Fig 1.1:** Multi-Project Wafer

*1.2 Fabrication Process*

MOSIS offers three different options for an accredited college or university to use. These options are registering for a Commercial account, a MOSIS Educational Program (MEP) Instructional account, or a MOSIS Educational Program (MEP) Research account. A comparison of these three accounts is listed on MOSIS’s website and is also shown in Table 1.1. Our advisors registered for a MEP Instructional account and we will either be using a GlobalFoundaries 180 nmCMOS (7HV) process or GlobalFoundaries 180 nm(7RF) process depending on the availability.

**Table 1.1:** Comparison of MOSIS Academic Account Types

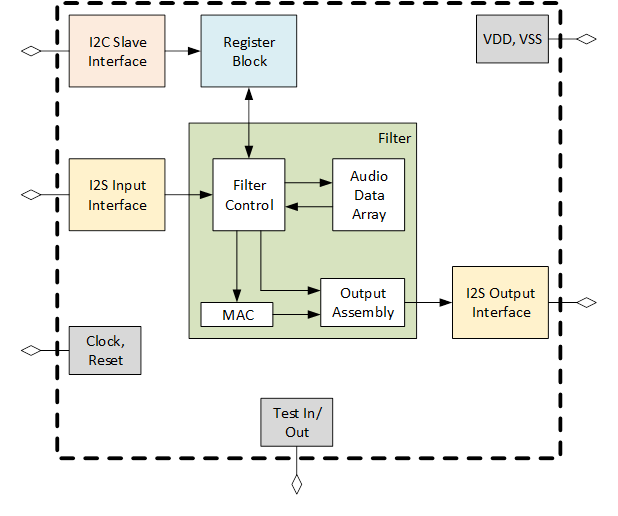
|  |  |  |  |
| --- | --- | --- | --- |
| **Topic** | **Commercial** | **MEP Instructional** | **MEP Research** |
| Primary Purpose | Customers pays for fabrication and packaging | Classroom instruction | Unfunded research |
| Available Processes | All | 1. ON Semi 0.50CMOS (C5N)  2. GlobalFoundaries 180 *nm* CMOS (7HV) | 1. ON Semi 0.50CMOS (C5N)  2. GlobalFoundaries 130 *nm* SiGe BICMOS (8HP)  3. GlobalFoundaries 130 *nm* CMOS (8RF-DM) |
| Size Limits | No restrictions | 5 deliverable parts of a project no larger than 1.5 mm x 1.5 mm | Less than 16 |
| Number of Submissions per Year | Unlimited | Annual request subject to review | One submission per academic year per institution after approval of proposal |
| Run Restrictions | All MPW runs except for MEP-only | MEP-only and space available for COM runs | Space available for COM runs |
| IP Access through MOSIS | Yes | No | Yes |
| Fabrication Costs | Customer pays fabrication cost | Free | Free |
| Packaging | No restrictions, customer pays | Free ceramic and OCP packaging; lids cannot be sealed. Fully encapsulated packaging not available. | No restrictions, customer pays |

*1.3 Submitting a Design to MOSIS*

When submitting a MPW run, MOSIS has specified certain steps that need to be taken in order to submit a design. The first step is to submit a new project request by logging onto their website. The designer then needs to assign their Export Control Classification Number (ECCN) before they make a fabrication request. Next, a fabrication request needs to be made specifying the process that will be used. The last step is to submit that actual design layout to MOSIS in either Caltech Intermediate Form (CIF) or Graphic Data System II (GSDII) format. Both of these formats are used to describe the layout of the integrated circuit and will be generated by an EDA tool.

**Chapter 2: System Design – Z. Nelson**

The system design of our project started in April 2015 when we wrote a senior project proposal. The proposal discussed the basic functionality that should be accomplished and how the tasks should be broken down. A top level block diagram of the chip is shown in Fig. 2.2. The chip was broken down into five major modules. These modules included the I2S input interface, I2S output interface, filter, register block, and I2C slave interface. Kevin’s responsibility was the I2S interface, Dhruvit’s was the filter, Julie’s was the register, and Whitley’s was the I2C interface. Over the summer,

**

**Fig 2.1:** Top Level Drawing of Chip

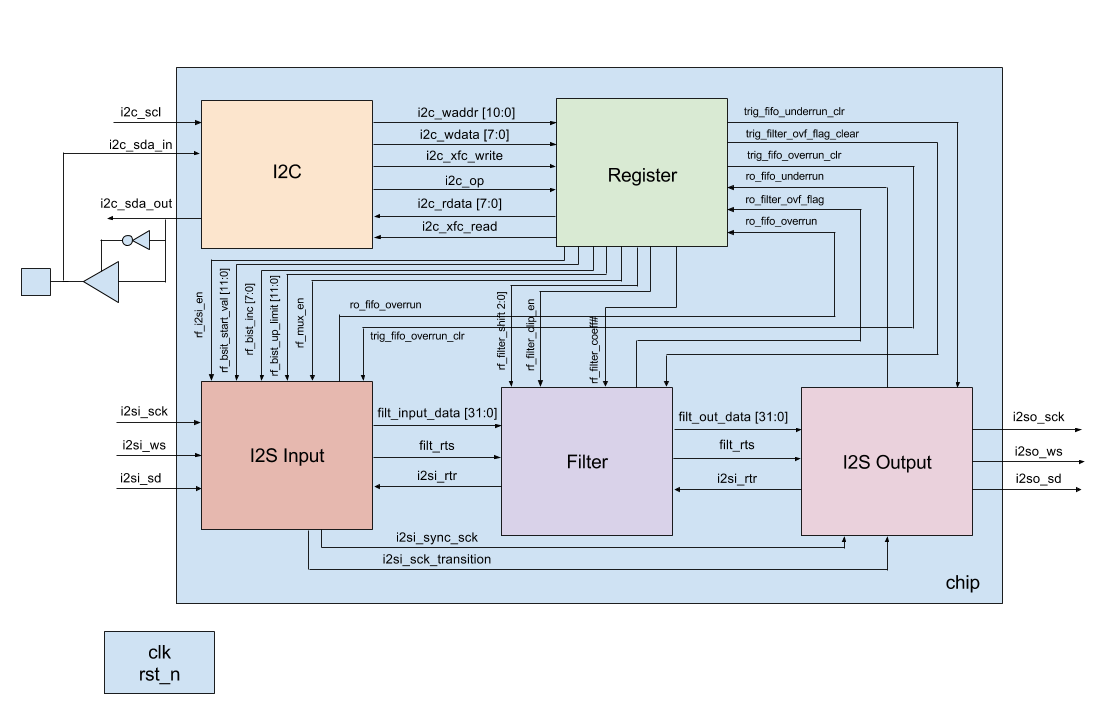
The main purpose of the I2S input interface was to convert a serial audio stream into 16-bit parallel data. This module also included a Built in Self-Test (BIST) feature that would use a predefined sawtooth signal as the audio data stream. After the audio stream was converted into parallel data, it was put through a 512-tap FIR filter in the filter module. The filter audio signal was then converted back to the I2S serial standard in the I2S output interface. The combination of these three modules was how data flowed within our chip.

**Data Flow:** I2S Input Interface 🡪 Filter 🡪 I2S Output Interface

The control flow of the chip started with the I2C slave interface module reading or writing register values to the chip. The register block is where the actual filter coefficient values were stored along with other information such as chip info, control bits, and status bits. The filter block would then read the 512 filter coefficients from the register block in order to implement the FIR filter.

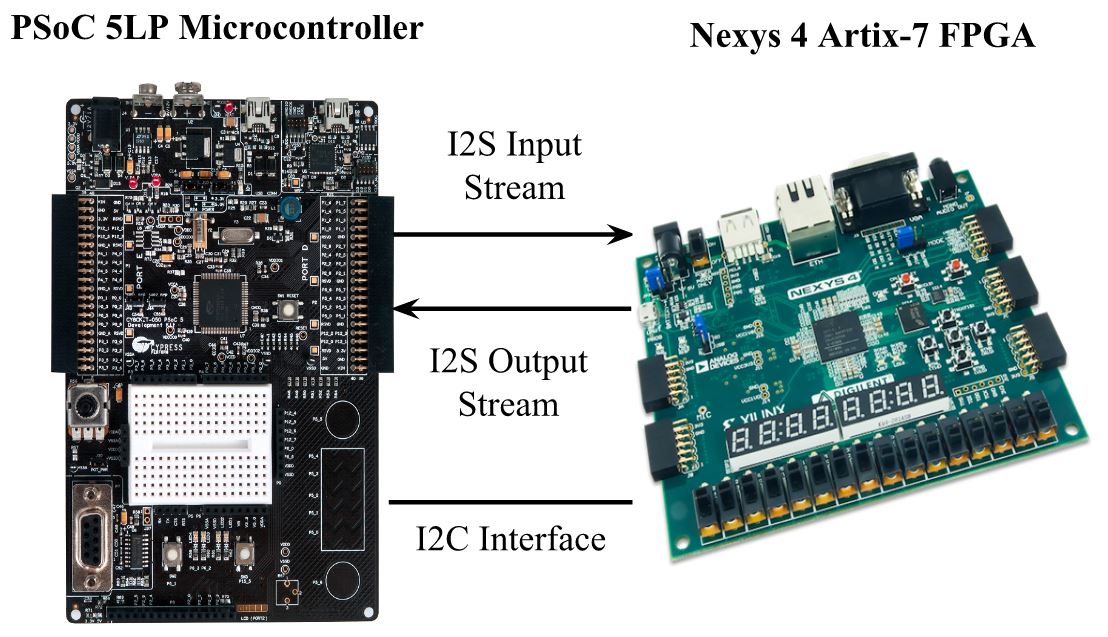
**Control Flow:** I2C Slave Interface 🡪 Register Block 🡪 Filter

The next step in the system design process was creating block documents that stored the details of each module. These block documents can be found on the GitHub repository. After the interfaces for each module were agreed upon, we created a detailed schematic of the top-level module of our chip (refer to Fig. 2.2).

****

**Fig. 2.2:** Detailed Top-Level Drawing of Chip

The PSoC microcontroller will be used to generate the audio stream and provide I2C read/write access for the chip. Since we will create the audio stream in the PSoC environment, we should know exactly what the output stream will be based on the filter coefficients we are using. The microcontroller will also have the ability to read and write filter coefficients to the chip. This is an important feature of our project because the user will be able to manually set and check the values of all 512 filter coefficients. A diagram of the PSoC microcontroller interfaced with the FPGA is shown in Fig. 2.3.



**Fig. 2.3:** FPGA Testing Environment

**Chapter 3: I2S Interface – K. Cao**

**Chapter 4: Digital Filtering – D. Naik**

**Chapter 5: Register Block – J. Swift**

**Chapter 6: I2C Slave Interface – W. Forman**

**Chapter 7: Simulation/Verification – K. Cao and D. Naik**

**Chapter 8: FPGA Implementation - Zachary Nelson**

The FPGA implementation was primarily a way to show that our design worked in realistic setting. The first step of the FPGA portion of the project was purchasing a Nexys 4 Artix-7 FPGA. After we received the FPGA, we needed to assign pins to the inputs and outputs of our design. Table 8.1 shows how we mapped the signals to the FPGA pins. The pin assignments were done using the PlanAhead software that is included in the Xilinx ISE Design Suite.

**Table 8.1:** Verilog Signal to FPGA Mapping

|  |  |  |  |
| --- | --- | --- | --- |
| Signal Name | Direction | FPGA Pin | PlanAhead Site |
| clk | in | internal (crystal) | E3 |
| rst\_n | in | reset button | C12 |
| i2si\_sck | in | JB3 | V11 |
| i2si\_ws | in | JB4 | V15 |
| i2si\_sd | in | JB7 | K16 |
| i2so\_sck | out | JB8 | R16 |
| i2so\_ws | out | JB9 | T9 |
| i2so\_sd | out | JB10 | U11 |
| i2c\_addr\_bits | in | JA9:JA7 | DC18, C17, G13 |
| i2c\_scl | in | JA1 | B13 |
| i2c\_sda\_in | in | JA2 | F14 |
| i2c\_sda\_out | out | JA3 | D17 |

Since we now had synthesized Verilog code and pin assignments, we could generate a programming file and download it to the FPGA. A FGPA programming guide was created and the steps are listed below.

**Step #1:** Click the Synthesize button in Xilinx ISE. (This will take several minutes)

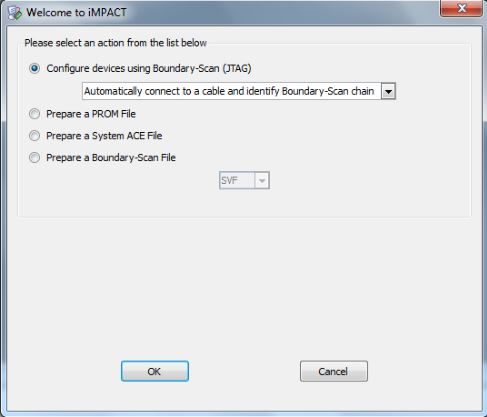
**Step #2:** Click the Implement Design button in Xilinx ISE. (This will take several minutes)

**Step #3:** Click the Generate Programming File button in Xilinx ISE. (This will take several minutes)

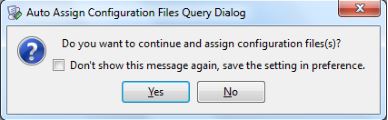
**Step #4:** Plug in FPGA via USB.

**Step #5:** Under “Configure Target Device”, click the Manage Configuration Project (IMPACT) button. The ISE iMPACT tool window will now open.

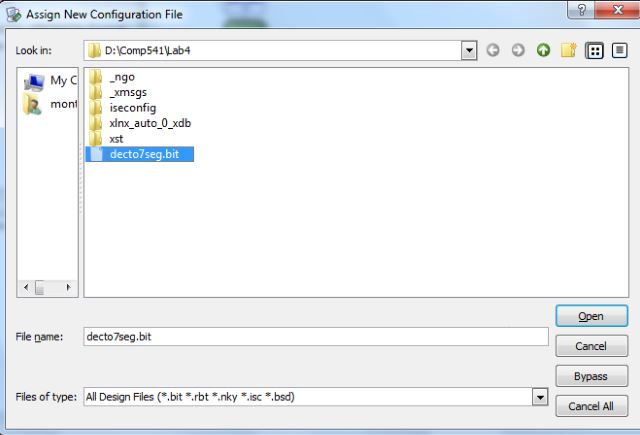
**Step #6:** Run the wizard by clicking Edit\Launch Wizard. The “Welcome to iMPACT” window will open.

****

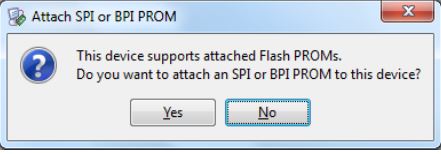
Click OK.



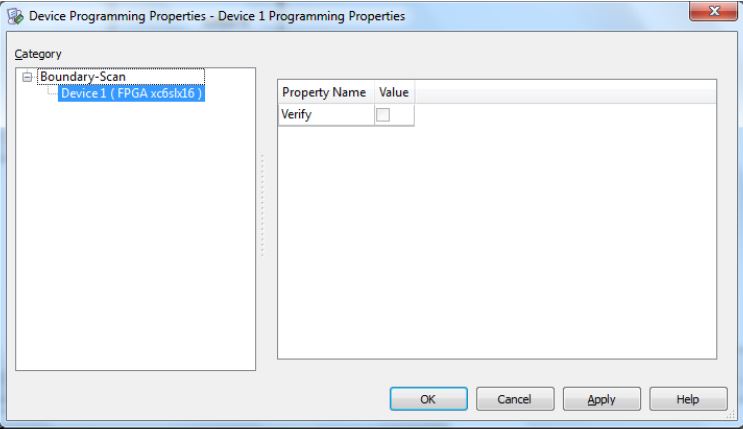
Click Yes.



Select the .bit file that you generated in Step 3.



Select No.

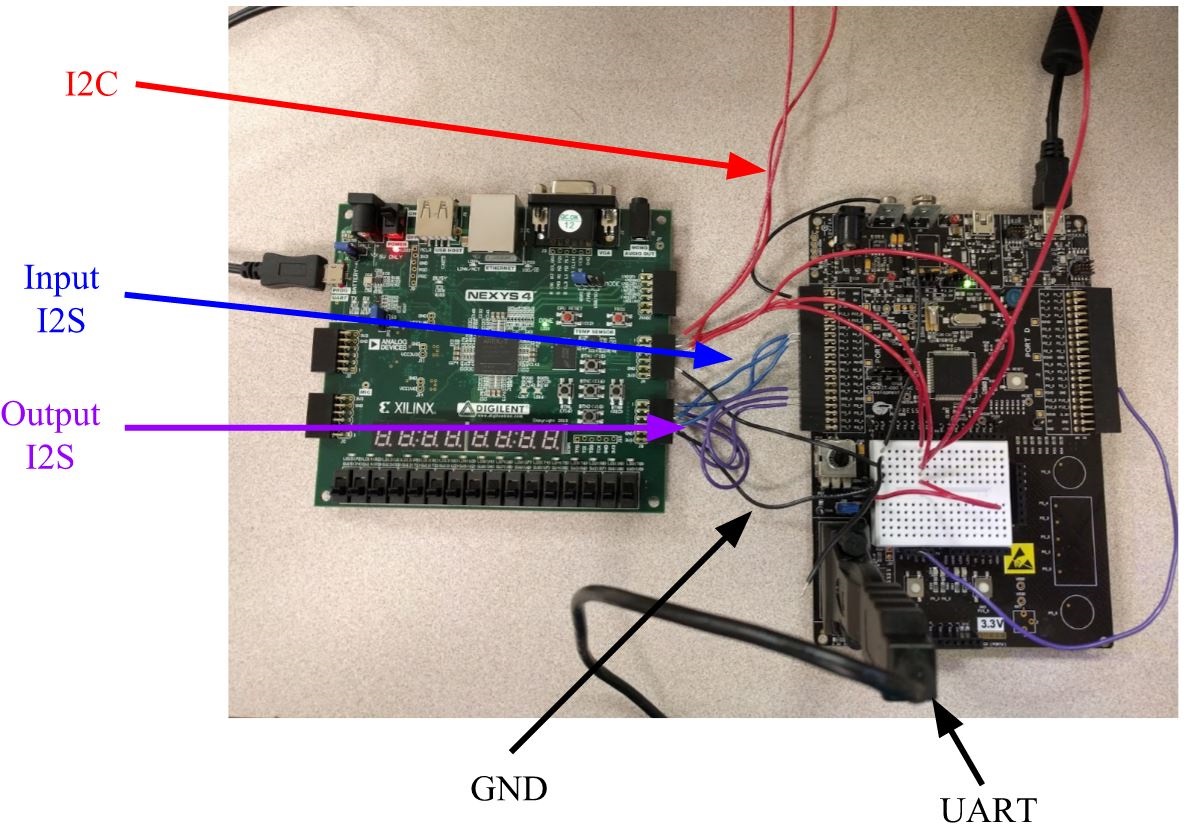


Click OK.

**Step #7:** If there are no errors, you will see an “Identify Succeeded” message. Right-click on the green Xilinx box and click program.

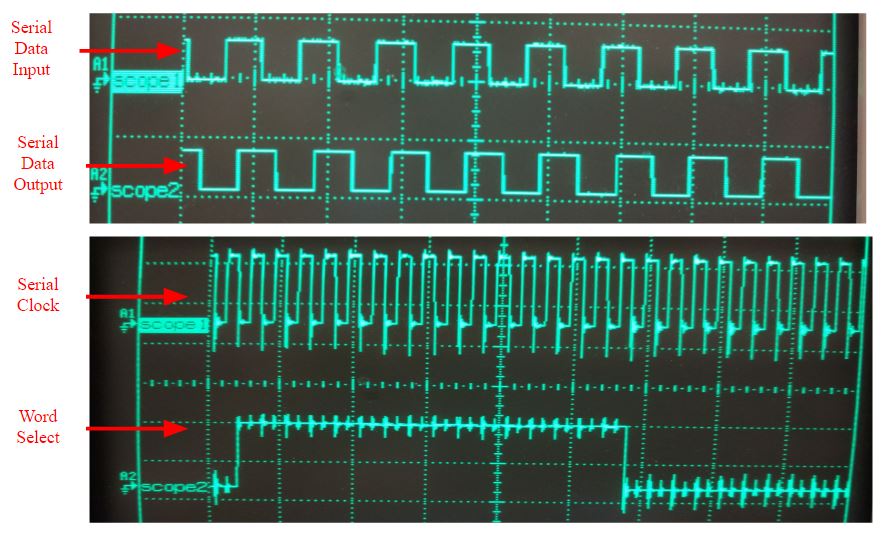
**Step #8:** Look at the port mappings in rtl\_description.xlsx and configure test setup.

Since our FPGA was now successfully programed, we needed to create I2S and I2C PSoC programs that could test the FPGA. Fig. 8.1 below shows the FGPA wired up to two PSoC microcontrollers (the I2S one is showed in the picture). The I2S program was created by Ian Patel and the I2C program was created by Whitley Forman. The I2S program was originally intended to generate and capture back I2S data to verify that the filtering component of the design was working. However, there was some kind of bug in the I2S capture that we could not fix. For some reason, we could not get accurate data captured back to the PC. The I2S program was successfully able to generate different types of input waveforms including sinusoidal, square, and triangular waves.



**Fig. 8.1**:FPGA Testing Set Up

Without the I2S capture working we were still able to do some testing of the FPGA. We first set the default filter coefficients to create a pass-through filter (the output signal is the same as the input signal). We then applied a square wave to the input since it can easily be observed on an oscilloscope. Fig. 8.2 show the serial clock and word select that was taken from the chip. Note that the input and output serial clock and word select matched in frequency. Fig. 8.2 also shows the square wave input and the corresponding output. It can be seen that output exactly matched the input besides being slightly delayed. This simple test gave us some physical proof that our designs data flow worked not only in simulation but on hardware.



**Fig. 8.2**:FPGA Waveforms (Pass-Through Filter)

**Chapter 9: EDA Tools and Physical Design – D. Naik**

**, J. Swift, K. Cao**

* Documentation for next year
* Problems with libraries, tips

**Chapter 10: Conclusion**

**References**

[1] Mosis.com, ‘About Us’, 2015.[Online]. Available: https://www.mosis.com/what-is-mosis. [Accessed: 11 November 2015].

[2] nxp.com, ‘*I2C-bus specification and user manual - UM10204*’, 2015. [Online]. Available: <http://www.nxp.com/documents/user_manual/UM10204.pdf>. [Accessed: 10 November 2015]

**Appendix A: Project Overview**

Biography 4

Engineering Standards and Realistic Constraints Form 4

Engineering Standards, Specifications, and Codes 4

Modern Engineering Tools 4

**Biography:**

*Kevin Cao:*

* Kevin is from Morris Plains, NJ and is a Computer Engineering major who is planning to enter the workforce after graduating at TCNJ. Kevin has interned as a software engineer at LGS Innovations, located in Florham Park, NJ.



*Whitley Forman:*

* Whitley is from Ocean Grove, NJ and is an Electrical Engineering major who is continuing his education in the electrical field and will be obtaining his Master Electrician’s license after graduation. He is planning on using his new knowledge and experience with his current business to expand into new ventures.

*Dhruvit Naik:*

* A resident of Mount Laurel, NJ, Dhruvit is a senior Computer Engineering major at TCNJ. He plans on entering the workforce after graduation and continuing his education in the coming years. He is the Vice-President of a startup, ThinkSOAS, INC.

*Zachary Nelson:*

* From Cream Ridge, NJ, Zachary is a Computer Engineering major who has enrolled in a Ph.D. program at Johns Hopkins University specializing in controls. He has experience as a software engineering intern at Teletronics Technology Corporation and an undergraduate student researcher at TCNJ as part of the MUSE program.

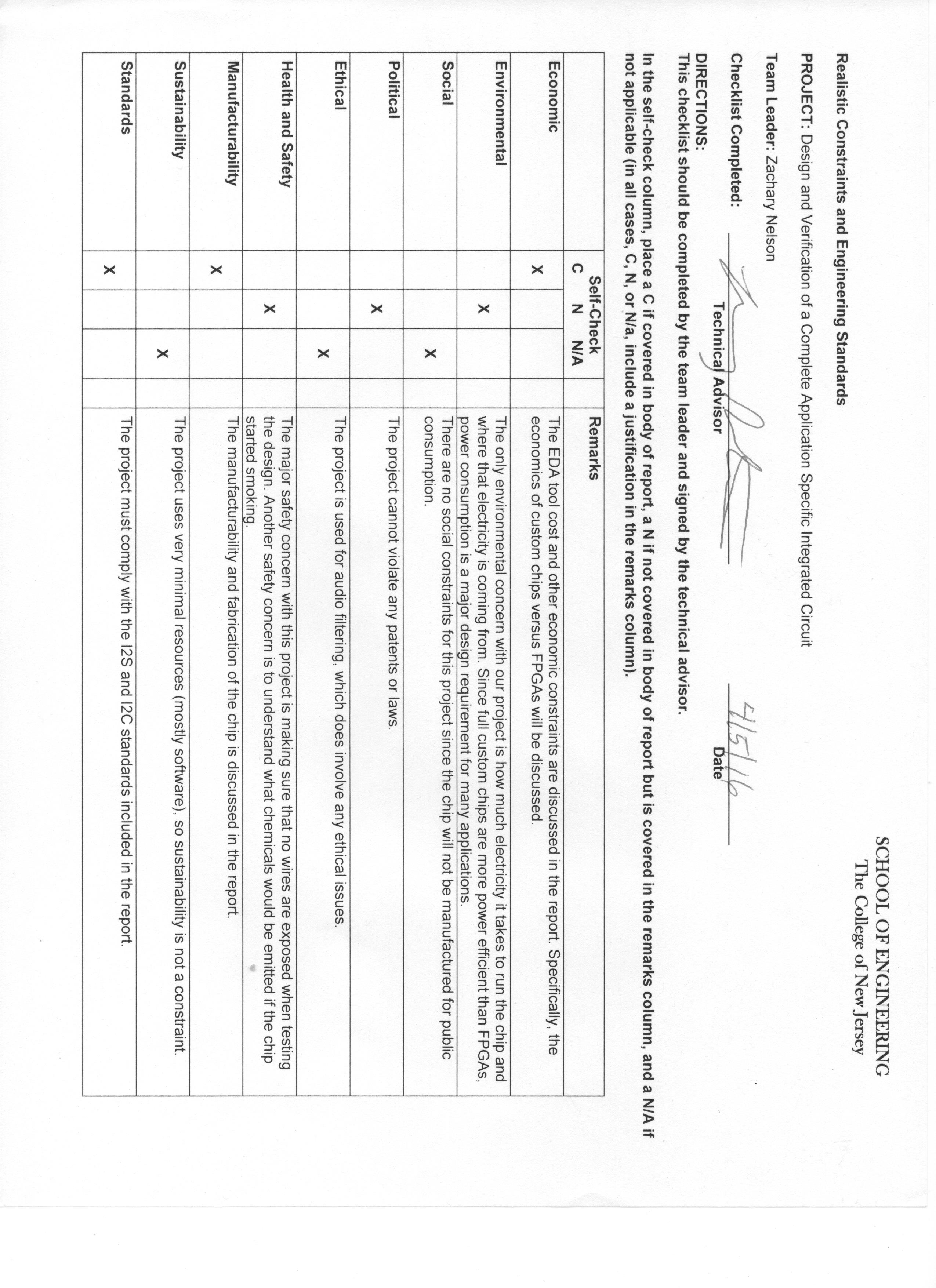


*Julie Swift:*

* From Robbinsville, NJ Juliann is a Computer Engineering major who is planning on entering the workforce after college. She has experience as a AutoCAD Designer interning at Linearization Technology and a Software Development Life Cycle Analyst interning at Educational Testing Services. She participated in the undergraduate student research program, MUSE, at TCNJ.



**Engineering Standards and Realistic Constraints Form:**



**Engineering Standards, Specifications, and Codes:**

* *I2C:* A serial communication standard that we used for uploading and reading registers to our FPGA. The details of this standard can be found in Appendix E.
* *I2S:* A serial communication standard for transferring digital audio. The I2S bus uses a bit clock line, word clock line, and data line. The details of this standard can be found in Appendix E.
* *Verilog:*A hardware description language (HDL) that can be used to design digital systems (such as an FPGA). This language is a high-level abstraction of a design that can be synthesized into low-level logic gates.

**Modern Engineering Tools:**

* *CORE 9*: System design software that we used to model the basic structure of the system. We also used it to keep track of the system requirements and use cases.
* *DropBox*: Cloud storage service that we used to store all confidential materials.
* *Git/GitHub*: Revision control software that allows a group to keep multiple versions of source code. We stored all documents and code in the public GitHub repository.
* *Linux RedHat Operating System*: Operating system that was required for all Mentor Graphics software (ModelSim and Pyxis).
* *Microsoft Project*: Project management tool that we used to keep track of the group’s schedule and progress.
* *ModelSim HDL Simulator*: Simulation and verification tool that we used to test our Verilog code.
* *Pyxis Custom IC Design Platform*: Multiple pieces of software that we used for the design of our simple chip.
* *Xilinx ISE Design Suite*: Design environment for Xilinx FPGA products. We used this tool for writing Verilog code, writing Verilog test benches, debugging Verilog code, synthesizing our design, and programming the FPGA.

**Appendix B: Management**

Schedule 4

Meeting Minutes 4

List of Contacts 4

Saftey Considerations 4

Materials List 4

Financial Budget 4

**Schedule:**

**Meeting Minutes:**

**Chip Requirements (June 11th Meeting)**

**Members in Attendance:** Dr. Pearlstein, Julie Swift and Zachary Nelson

* **Overall Goal**
  + **Produce an Audio Processing Integrated Circuit**
* I2S Interface
  + Audio Input: 2 channel stereo channel I2S (master interface)
  + Support audio input sample rates of 8kilosample/sec – 48kilosamples/sec
  + Output is the same sample rate as the input and I2S
  + Digital audio bit clock and the word select (ws) line will be controlled from master
  + Input and Output will be 2 channel 16 bits
* I2C Interface
  + Support single master configuration
  + 7-bit addressing and we will consume entire I2C address space
* Uses an external clock input
  + Clock frequency will be a minimum of 1200 times the audio sampling rate
  + Maximum clock rate will be 100 MHz
* Has an external reset pin
  + Power on reset
* Register Block
  + 512 bit frequency coefficients
  + 10k register bits
  + Register fields will include
    - Source select bit
      * Allows user to select between I2S and BIST (built in self test)
    - Filter Bypass Bit
      * 0 doesn’t bypass, 1 bypass
    - 512 16-bit signed coefficients stored as 2’s compliment
      * The effective radix point of the coefficients (4 bit number for data point, 4 bit number for coefficient)
    - Read only status register bits
      * Overflow/saturation detector – audio clipping
      * Input FIFO overrun
      * Output FIFO underrun
    - Control Bit Fields
      * 1 sticky bit to clear overrun
      * 1 sticky bit to clear underrun (stay until you clear them)
      * Clear Overflow Flag
      * Filter Order to Support
        + 9 bit number to represent 1 to 512
  + Presents an array of registers for hosts control and status monitoring (through I2C read and write operations).
* Provide built in self test function
  + Test gadgets for I2S
    - Test by converting I2S to audio coeffs
    - Audio coeffs to I2S
* Chips made by service call MOSIS
  + IBM7RF process, geometry, drawn gate length, 180nm gates, mixed signals,
  + Chip Area: no more than 3 by 3 squared mm
* Filter Audio
  + Implement an FIR filter on the input data based on the coefficients stored in the programmable registers
    - Programmable from 1-512 taps
    - Filter Order Control (9-bit number)
      * Support filter from 1-512 taps
    - Maintain intermediate precision of 4
* Produce a microcontroller platform to configure the audio input and output modules
  + UDA 1380
  + Produce a test fixture to show that the chip works
    - Sample analog audio and covert it to I2S
    - Receive I2S and convert it to analog audio
    - Allow user to create filter coeffs and upload them to the chip that was designed
    - Has to have software to configure the analog audio subsystem
    - Parametrize low pass filters, high pass, band pass, and comb filters
      * Use a button on a PSOC?
      * Could use slider on PSOC to change the frequency and upload the new coeffs to the chip
    - Create a board that we can plug the chip into to do testing
      * Bread board?
    - Will not have to design a printed circuit board for this chip

**June 18th Senior Project Meeting**

**Armstrong 144, 3:30 P.M – 5:05 P.M.**

**Members in Attendance:** Dr. Pearlstein, Zachary Nelson, Julie Swift, Whitley Forman and Dhruvit Naik

* Went through the CORE requirements hierarchy for the chip **(System Design Step)**
  + Zach will make the top-level requirements the elements from the chip block diagram so that the requirements are broken down in a more logical way
  + Zach will make an Enhanced Functional Flow Block Diagram (EFFBD) on CORE with the functions that were generated as part of creating the requirements
* Discussed the design flow of creating an integrated circuit
  + A figure illustrating this is on GitHub under Chip-Design/proj\_asic/docs/design\_flow.png
  + Major Steps
    - System Design
    - RTL Design
    - Logic Synthesis
    - Design for Test (DFT) Implementation – may be able to skip this
    - Floor Planning
    - Place and Optimization
    - Routing
    - Verification
* Deadline for the chip
  + November 30th: We would get the chip back in the Spring semester and be able to see if it works
  + March 2016: Would get the chip back after graduation and would implement our design on a FPGA during the Spring semester instead
* Dropbox will be used for storing private files
  + The rest of the code and documents will be stored on the public GitHub account
* Assigned Verilog modules to group members **(RTL Design Step)**
  + Everyone will start working on the modules over the summer
  + Zach: i2s\_in.v and i2s\_out.v
  + Julie: register.v
  + Whitley: i2c\_slave.v
  + Dhruvit: filter.v
  + Kevin: chip.v
* Julie and Whitley need to hand in the NDA forms (electronically or in person)
* Kevin and Dhruvit need to accept the invite to the GitHub account

**July 7th I2S Senior Project Meeting**

**Dr. Pearlstein’s Office, 2:00 P.M – 5:10 P.M.**

**Members in Attendance:** Dr. Pearlstein and Zachary Nelson

* Register block bits will either start with “ro”, “rf”, or “trig”.
  + ro label for input.
  + rf label for outputs.
  + trig label when the bit causes something to trigger.
* Add the bit i2s\_in\_en to the I2S\_IN block.
  + Low to high: coming out of reset.
  + High to low: goes back into a reset state.
* Need variables to characterize the BIST saw-tooth wave.
  + rf\_bist\_start\_val (16 bit signed value) – start value.
  + rf\_bist\_inc (8 bit integer) – increment between 0 and 255.
  + rf\_bist\_upper\_limit (16 bit signed value) – upper limit.
* We will create a synthesized sclk by dividing the system clock.
* Make all the flops clock with the system clock (always statements must be with clk).
* We will have a register holder that takes in one bit at a time.
* Add trig\_i2sin\_fifo\_overrun\_rst bit
  + Example:

if (ro\_fifo\_overrun) 🡪 happens when rts=1 and rtr=0

ro\_fifo\_overrun = 1

else if (trig\_i2sin\_fifo\_overrun\_rst)

ro\_fifo\_overrun = 0

**July 30th Senior Project Meeting**

**Dr. Pearlstein’s Office, 2:00 P.M – 3:40 P.M.**

**Members in Attendance:** Dr. Pearlstein and Zachary Nelson

* Discussed how register.v will use register addressing to access data.
* Discussed cell utilization and a rough approximation about how much area our chip will use (60% utilization)
* Went through **register\_map.xlsx** with Dr. Pearlstein
  + Added new registers
  + Fixed register addressing
  + Added missing default values
* Added register fields that enable clipping for the filter and one that shifts the number of bit positions after the filter accumulator.
* Reviewed the block diagram of the **i2si.v** block.
  + This diagram seems to be correct and the block can start being designed and verification tests can be created.
* Discussed details of how the **i2so.v** will work.
  + Clock divider will go in this block
  + Zach wrote down a quick block diagram
* Block Documents Status
  + register.v – not started
  + i2si.v – near completion
  + i2so.v – not started
  + filter.v - not started
  + i2c.v - not started
* Zach and Kevin will switch roles.
  + Zach will be responsible for chip.v and can help with all blocks (especially i2s blocks)
  + Kevin will be responsible for the i2s blocks.

**September 2nd Register Senior Project Meeting**

**Dr. Pearlstein’s Office, 1:00-2:30**

**Members in Attendance:** Dr. Pearlstein, Zachary Nelson, and Julie Swift

* Project Goals Slide
  + Why are we doing this project?
    - This is not a common project for undergraduates because of the high cost associated with the fabrication of a chip.
    - Describe the MOSIS Education Service
    - Free fabrication for 3mm by 3mm
  + Also refer to this as VLSI
  + Explain the difference between an FPGA and an Integrated Circuit
  + Why did we choose to process audio as our application?
    - Complex and interesting enough
    - Not too hard so that this project is able to be completed
  + Take out parameterized filters
  + Put a picture of a chip
* Chip Overview Slide
  + I2S is digital but represents an analog signal
* DropBox is for confidential files while GitHub is for everything else
* I2C Slide
  + sda\_in and sda\_out are both interfaces
* Filter Slide
  + Fix the summation sign
* DFT Slide
  + We will most likely not be doing this task but it is good to have as background information.
* Gate Level Simulation
  + We will not do gate level simulation that intensively.
* Place and Route
  + We will not manually do the place and route, the EDA tools will do it for us.
  + We may be doing manual floor planning.
* Create a Block Level Testbenches Slide
  + We will have tb for each individual block as well as the overall chip
* Discussed interfaces for the register.v module
* Discussed a block diagram of the register.v module.
  + Ask Julie or Zach for more specifics
* MOSIS information
  + We can order one lot of 40 chips
  + We will fab using the Global Foundries 180 nm CMOS (7HV) process
  + MOSIS technology code for the 7HV process is GF\_7HV
  + Customer Submission date for 7HV is **March 7th, 2016**

**September 2nd Senior Project Meeting**

**Armstrong Hall 137, 3:30 - 4:15 P.M.**

**Members in Attendance:** All Members

* Add a presentation slide on input/output cells
* The Format of the weekly meeting will be:
  + Discuss what the team has done over the past week
  + Discuss what the team is going to do over the next week and beyond
  + The most important thing is to discuss any roadblocks that we have
* One goal is to have the design implemented on a FPGA by the end of the fall semester
* Some tasks that can take place during the spring semester:
  + Place and Route
  + Verification
* We need to adopt a project management tool and create a schedule that is more sophisticated than the one that Dr. Katz has provided us with
  + One potential tool that we will look into is Microsoft Project
  + Be specific enough for about 3 milestones per week
* Manage the amount of time being put into senior project
  + Put the needed amount of time into the project but if we find ourselves each putting much more than 6 hours/week, we may need to narrow the scope of the project.
* Grading for Senior Project
  + 2 students will be graded by Dr. Hernandez and 3 students will be graded by Dr. Pearlstein
  + The rubrics for Senior Project I and II was distributed by Dr. Katz in an email
* What FPGA will we use for the project?
  + We could use one that is already in stock
  + We could purchase one with the $500 that the group has
* Julie and Dhruvit will be responsible for EDA Installation
  + Step 1: Figure out the EDA requirements and install OS
  + Step 2: Install EDA
  + Step 3: Simple test case with libraries from MOSIS and doing a place and route test
* We need to discuss who will be responsible for configuring the test fixtures.
* We will hold off on the website design until we hear more information.

**For Next Week:**

* We need to create a detailed plan/schedule for the project using a tool like Microsoft Project
* Everybody needs to finish their block documents for their specific modules
* Everybody needs to add the information about their module to the CORE 9 project

**September 9th Senior Project Meeting**

**Armstrong Hall 137 and 144: 4:00 – 5:15 P.M.**

**Members in Attendance:** All Members

**Next Week’s Work Plans:**

* **Zach**
  + Fix Schedule
  + RTR/RTS and XFC Protocol
  + Buying an FPGA and seeing if the school one will work
  + CORE for I2S Blocks
  + Start coding one of the I2S submodules (BIST Generator?)
* **Whitley**
  + Start coding the I2C module
  + CORE 9 for I2C Requirements
* **Julie**
  + EDA Tool Installation
  + Complete Register Block Documentation
  + Start coding the Register module
  + CORE 9 for Register Requirements
* **Kevin**
  + Complete one submodule for the I2S Block (Deserializer?)
  + Create test benches and test this submodule
  + CORE 9 for I2S Blocks
* **Dhruvit**
  + EDA Tool Installation
  + Start coding the Filter module
  + CORE 9 for the Filter module

**Meeting Notes:**

* The RTS/RTR and XFC protocols need to be in one place
  + Zach assigned as the owner to this
  + Should be finished by next Wednesday
* EDA Tools should be installed and a simple place and route test should be completed by next week
  + We will be downloading Mentor Tools
  + An action item is to contact by email or in person Mike about installing the tools
* Things that need to be added/changed for the Microsoft Project schedule
  + Creating test benches
  + Create test cases for individual blocks
  + Creating test cases
  + Correction: Microcontroller not “board” will be linked to the chip
  + Shorten the milestone names
  + Block level synthesis
  + Full chip simulation
* CORE requirements and use cases can be integrated to write test specifications
  + The new CORE license works
* Dr. Hernandez will give feedback on the schedule
* Dhruvit and Julie have access to 144-B
  + This is where we will be installing the EDA tools
  + Can we remotely connect to this computer-Ask Mike?
* Chip.v
  + Will instantiate everyone’s modules
  + We will need to create input/output submodules
* Create model in another environment that takes in the register states and inputs and produces the correct output
  + An end to end test case
  + This is used to verify that we are producing the correct outputs in our chip
  + Does not need to be the most sophisticated model due to time
* Test benches will be stored on the “tb” folder on GitHub
* FIFO will be very similar for all blocks
  + The only difference will be the size and the width
* Need to buy crystal oscillators (10, 20, or 100?)
* Whitley will do board design in spring semester
* PSoC configuration will take 1-2 weeks of time
  + Whitley may be assigned to this task

**September 23rd Senior Project Meeting**

**Armstrong Hall 137: 3:30-4:00 P.M.**

**Members in Attendance:** Dr. Orlando Hernandez, Zachary Nelson, Julie Swift, Dhruvit Naik and Kevin Cao

**Last Week’s Work:**

* Zach
  + Cleaned up RTS/RTR and XFC Protocols
  + Updated CORE 9 Files
  + Updated Schedule on Microsoft Project
    - Everyone agreed on due dates
    - Created a calendar on Google
  + Continued working on i2si\_bist\_gen.v
* Kevin
  + Continued working on i2si\_deserializer.v
  + Looked into setting up the project website
* Dhruvit
  + Continued working on filter\_convolution.v
* Julie
  + Continued working on register block document and register.v
* Whitley
  + Continued working on i2c\_slave\_deserializer.v

**Next Week’s Due Dates:**

* Friday, September 25th
  + Install Linux on Machine in Room 144B (Dhruvit and Julie)
* Monday, September 28th
  + Finish Design and Testing: filter\_convolution.v (Dhruvit)
  + Install EDA Tool on Machine in Room 144b (Dhruvit)
* Wednesday, September 30th
  + Finish Design and Testing: i2c\_salve\_deserializer.v (Whitley)
  + Finish Design and Testing: i2si\_bist\_gen.v (Zach)
  + Finish Design and Testing: i2si\_deserializer (Kevin)

**Meeting Notes:**

* We should have a testing specification when we perform testing
  + Multiple smaller unit tests are better than a couple of large comprehensive tests for our blocks
* Dr. Hernandez offered to present his presentation slides on Verilog to the group
  + All of the Verilog modules will have 3 always statements
  + The group will look over the slides and let Dr. Hernandez know if this will be beneficial

**September 30th Senior Project Meeting**

**Armstrong Hall 137: 3:30-3:30**

**Members in Attendance:** All Members

**Last Week’s Work:**

* Linux Installed
  + Permission errors
  + Need to email passwords
  + Root password: icchip2015
* Zach
  + Bist generator
  + Rts and rtr

**Next Week’s Due Dates:**

* Last Week:
  + Finish Design and Testing of i2si\_deserializer.v (Zach and Kevin)
    - 1 week for the deserializer.v
  + filter\_convolution.v (Dhruvit)
  + Testing of Whitley’s block
    - 1 week for testing
* Friday, September 2nd
  + Create Web Page (Kevin)
  + Finish Design and Testing of filter\_accumulator.v (Dhuvit)
* Wednesday, September 7th
  + Finish Design and Testing of i2so\_serializer.v (Zach and Kevin)
  + Finish EDA Tool Installation (Dhruvit and Julie)
  + Place and Route Test (Dhruvit and Julie)
  + PDR Presentation #2 (All)

**Meeting Notes:**

* Zach will make a test fixture
* Action items
  + Investigate the use of mentor tools for place and route ()
* Asynchronous vs synchronous reset
  + We will use asynchronous
  + Add to rtr protocols

**January 26th Senior Project Meeting**

**Armstrong Hall 148: 2:00PM – 2:45PM**

**Members in Attendance:** All Members

**Goals for Next Week:**

* Kevin
  + Fix BIST issues
  + Testbenches with different clock ratios
  + Learn how to use Mentor Tools
* Dhruvit
  + Fix Filter Warnings
  + Mentor Tools
* Whitley
  + Register/I2C Subsystem
  + Subsystem Test
* Zach
  + I2S/Filter Subsystem
  + Register Help
  + FPGA
* Julie
  + Register/I2C Subsystem
  + Subsystem Test
* Ian
  + PSoC Program

**February 2nd Senior Project Meeting**

**Armstrong Hall 148: 2:00PM – 3:00PM**

**Members in Attendance:** All Members

**Goals for Next Week:**

* Updated Microsoft Project Plan
  + Plan 2.0
* Coordinate with Ian
* Whitley
  + Last Week
    - Got the submodule test finished
  + Next Week
* Julie
  + Last Week
    - Got the submodule test finished
  + Next Week
    - Continue working on
* Kevin
  + Last Week
    - More testing on I2S
    - Looked into EDA tools
    - Matching tools with different stools
  + Next Week
    - Get libraries installed
    - Place and Route
    - Working with Dhruvit
* Zach
  + Last Week
    - Fixed issues
  + Next Week
    - Coordinate with Ian
    - Create New Plan
    - FPGA Implementation
* Dhruvit
  + Last Week
    - EDA tools are stables
  + Next Week
* Steps
  + Coding
  + Simulation/Verification
  + Synthesis 🡪 netlist
  + Physical Design
  + Place and Route
  + GSDII Masks File

OR you can start

* Manual schematic capture of IO cells

**February 10th Senior Project Meeting**

**Armstrong Hall 148: 2:00PM – 3:00PM**

**Members in Attendance:** All Members

* Zachary
  + Last Week
    - Created the presentation
    - Worked on the I2S/Filter Subsystem
    - Updated the top-level drawing
    - Created an updated schedule
    - Added all code to chip.v
      * All Verilog code is in chip.v and is synthesizable
      * Need help with a testbench
  + Next Week
    - Top-level testbench
    - Work on chip.v
* Whitley and Julie
  + Last Week
    - Register and I2C blocks working together
    - Loading register values works
  + Next Week
    - Top-level testbench
    - Implement FPGA for I2C
* Kevin
  + Last Week
    - Helped I2S/Filter module
    - EDA Tools tutorials
* Dhruvit
  + Last Week
  + Next Week
    - More verification of filter block

**March 1st Senior Project Meeting**

**Armstrong Hall 148: 2:00PM – 2:45PM**

**Members in Attendance:** All members

* Zach
  + Last Week
    - Uploaded Ian’s PSoC Program
    - Updated chip.v
    - Worked TCF poster
    - Updated schedule
    - Worked with PlanAhead Tutorial
    - Created document for final report
    - Realistic constraints form
    - Xilinx ISE will work
  + Next Week
    - Work with PlanAhead
    - Help finish Ian’s I2S PSoC Program
    - Work on TCF Poster
  + Questions
    - When should we give you a TCF poster draft by?
      * Before break
    - What speed is FPGA? -3, -2, -1
      * Look in the data sheet
    - Can you help with PlanAhead?
      * Will meet with Dr. Pearlstein
    - How do we specify the I/O ports?
      * Will meet with Dr. Pearlstein
* Julie
  + Last Week
    - Looked into EDA tools from Princeton
    - Updated register Verilog code
    - Updated register Excel document
  + Next Week
    - Physical Design
* Dhruvit
  + Last Week
    - Filter model in Java
    - Fixed issues with filter Verilog code
  + Next Week
    - Physical Design
* Kevin
  + Last Week
    - Top-level simulation/verifications
    - I2S debugging
  + Next Week
    - Continue verification
* Whitley
  + Last Week
    - Created multiple test benches with different register values
  + Next Week
    - Continue verification
* Additional Comments
  + Source and sink could come from UDA 1300
  + Might need to buy this

**March 8th Senior Project Meeting**

**Armstrong Hall 148: 2:00PM – 2:30PM**

**Members in Attendance:** Zach, Dhruvit, Kevin, Whitley, Dr. Hernandez, and Dr. Pearlstein

* Zach
  + Last Week
    - Finished TCF Poster
    - Learned how to use Plan Ahead software
    - Set up clock signal and specified I/O ports
    - Fixed Realistic Constraints Form
  + This Week
    - Warnings for Xilinx ISE
    - Finish design in Plan Ahead
    - Get poster printed
* Dhruvit
  + Last Week
    - Top-Level Simulation
    - Looked at Physical Design presentation
  + This Week
    - Get a plan for the physical design
* Julie
  + Last Week
    - Looked at Physical Design presentation
  + This Week
    - Get a plan for the physical design
* Kevin
  + Last Week
    - Top-Level Simulation
    - Text file comparison
  + This Week
    - Install Model-Sim (Dhruvit might help)
    - Fix warnings
    - Clean up code
* Whitley
  + Last Week
    - Top-Level Simulation
  + This Week
    - I2C PSoC Program
    - Fix I2C always block
* Automate testing process using make file

**March 22nd Senior Project Meeting**

**Armstrong Hall 148: 2:00PM – 3:00PM**

**Members in Attendance:** Zach, Dhruvit, Kevin, Whitley, and Dr. Pearlstein

* Zach
  + Last Week
    - Fixed Abstract
    - Worked with I2S PSoC program
    - Fixed some RTL issues
  + This Week
    - Work with Ian on PSoC program
    - Keep updating top-level
    - Get clock verified on FPGA
    - Open drain and resistor for I2C
    - Output voltage on Artix board
    - DA conversion in PSoC
* Dhruvit
  + Last Week
    - Researched how to use EDA tools
    - Having trouble logging into machine while on campus
  + This Week
    - Continue working with physical design
* Kevin
  + Last Week
    - Downloaded ModelSIM
    - I2S block works without optimization
  + This Week
    - Continue fixing synthesis warnings
* Whitley
  + Last Week
    - Started I2C PSoC program
  + This Week
    - Continue I2C PSoC program
    - Test I2C on FPGA

**March 29th Senior Project Meeting**

**Armstrong Hall 148: 2:00PM –**

**Members in Attendance:** Zach, Dhruvit, Kevin, Whitley, and Dr. Pearlstein

* Zach
  + Last Week
    - Got all correct port mappings
    - Fixed all but one FPGA warnings
    - Ian finished PSoC program and show me how to use it
      * I tested it and it seems to work
    - Wrote FPGA guide
    - Created a realistic test environment with I2S input stream
      * Nothing on I2S output (might be a reset issue)
    - Updated website
  + This Week
    - Further testing and debugging of FPGA
    - More code documentation
* Dhruvit
  + Last Week
    - Drew a transistor in EDA tools
  + This Week
    - Design rule check
    - Research on how to make inductor
    - Power and ground routing
    - MOSIS GSDII file requirements
    - Implementation warning
* Kevin
  + Last Week
    - Fixed warnings
  + This Week
    - Help Dhruvit
* Whitley
  + Last Week
    - Fixed I2C warnings
    - PSoC I2C program
  + This Week
    - I2C program
    - Default register state
* Julie
  + Last Week
    - Playing around and learning the EDA tool
    - Fixed overrun/underrun setting that Kevin told me to correct
    - Added comments to code
  + This Week
    - Help Dhruvit

**April 5th Senior Project Meeting**

**Armstrong Hall 148: 2:00PM – 3:00PM**

**Members in Attendance:** Zach, Dhruvit, Kevin, Whitley, Dr. Pearlstein, and Dr. Hernandez

* Zach
  + Last Week
    - Set up final report and presentation
    - Added comments to code
    - Fixed chip output
    - Passed sine, triangular, and square wave through FPGA
    - Started a testing document for FPGA
  + This Week
    - Work on reset warning
    - Further testing of FPGA (maybe with filtering and I2C)
  + Questions
    - VDDIO on PSoC? 🡪 Should be fixed now
* Dhruvit
  + Last Week
    - Made some changes to register.v, filter\_mux.v, filter\_round\_truncate.v
    - Created a new testbench that does not use I2C
    - Worked with EDA tools
    - Worked on FPGA warning
  + This Week
    - Create different filter coefficient combinations to test on FPGA
    - Continue working with physical design of a chip
* Kevin
  + Last Week
    - Testing of chip.v
    - Worked with EDA tools
  + This Week
    - Continue working with physical design of a chip
* Whitley
  + Last Week
    - Worked on I2C PSoC program
  + This Week
    - Finish I2C PSoC program
    - Test I2C PSoC program on FPGA
* Julie
  + Last Week
    - Worked with EDA tools
  + This Week
    - Continue working with physical design of a chip

**List of Contacts:**

* Ann Zsilavetz - Computer Science Program Assistant (zsilave2@tcnj.edu)
* Chris Collins - Lab Technician (collinsc@tcnj.edu)
* Katie Thacker - CORE University Program Coordinator (kthacker@vitechcorp.com)
* Michael Mensch – Information Technology (mensch@tcnj.edu)

**Safety Considerations:**

The major safety concern with this project involved interfacing the microcontroller with the FPGA. By taking the following precautions, no events compromising the safety of the team occurred. Since both the FPGA and microcontroller used 3.3V logic, we did not have to worry about any problems with high voltage. We made sure there were no loose wires by keeping our wiring as organized as possible. When working with the FPGA, we did not allow any food or drinks around the board in order to prevent damage. We were also aware of the fact that either the microcontroller or FPGA could start smoking. These chemicals could potentially be dangerous and we were aware to contact Chris Collins if this ever occurred.

**Material List:**

*Software Materials*:

* CORE 9 University
* Dropbox
* Git/GitHub
* Linux RedHat Operating System
* Microsoft Project 2013
* Mentor Graphics Software
* ISE Design Suite 14.7

*Hardware Materials*:

* CY8CKIT-050 PSoC 5LP Development Kit
* Nexys 4 Artix-7 FPGA Board
* Stock Room Materials (wires, wire cutters, etc.)

**Financial Budget:**

|  |  |  |
| --- | --- | --- |
| **Item** | **Cost** | **Comment** |
| CORE 9 University | $0.00 | Vitech’s CORE in the Classroom program |
| Dropbox | $0.00 | Free up to 2GB of storage |
| Git/GitHub | $0.00 | Free software |
| Linux RedHat Operating System | $0.00 | TCNJ Subscription |
| Microsoft Project 2013 | $0.00 | Free from the Computer Science Department’s DreamSpark Program |
| Mentor Graphics Software | $0.00 | TCNJ Subscription |
| ISE Design Suite 14.7 | $0.00 | Free software |
| CY8CKIT-050 PSoC 5LP Development Kit | $0.00 | School already owns |
| Nexys 4 Artix-7 FPGA Board | $192.41 | Purchased from Digilent |
| **Total Budget** | $500.00 |  |
| **Total Costs** | $192.41 |  |
| **End Balance** | $307.59 | Well under-budget |

**Appendix C: Source Code**

chip.v (top-level module) 4

Filter Code 4

filter.v 4

filter\_accumulator.v 4

filter\_mux.v 4

filter\_stm.v 4

filter\_storage.v 4

filter\_round\_truncate.v 4

I2C Code 4

i2c.v 4

i2c\_deserializer.v 4

i2c\_sequencer.v 4

i2c\_serializer.v 4

I2S Input Code 4

fifo.v 4

i2s\_in.v 4

i2si\_bist\_gen.v 4

i2si\_deserializer.v 4

i2si\_mux.v 4

i2si\_synchronizer.v 4

I2S Output Code 4

i2s\_out.v 4

i2so\_serializer.v 4

Register Code 4

chip\_reg.v 4

register.v 4

trig\_generator.v 4

**Appendix D: Test Benches**

chip.v (top-level module) 4

**Appendix E: Industry Specifications**

I2C Specification 4

I2S Specification 4