Introduction of Real-World Signals and Systems into ECpE DSP Laboratory Curriculum

Iowa State University
Department of Electrical & Computer Engineering

Team 14 - Fall 2020
Client & Advisor: Matthew Post
Email: sddec20-14@iastate.edu
Website: http://sddec20-14.sd.ece.iastate.edu/

Samuel Burnett
Electrical Engineering
burnett@iastate.edu

Isaac Rex
Electrical Engineering
irex@iastate.edu

Emily LaGrant
Electrical Engineering
elagrant@iastate.edu

Mitchell Hoppe
Software Engineering
mhoppe@iastate.edu

Brady Anderson
Electrical Engineering
bcra@iastate.edu

Max Kiley
Electrical Engineering
mkiley@iastate.edu

Executive Summary

Development Standards & Practices Used

Standard industry convention was used in all hardware and software designs. These include:

- Best practice followed for schematic
  - Use standard signal flow model
  - Separate high level design blocks in to multiple sheets
  - Use standard practice symbols and circuit block drawings
- Best practice followed for PCB layout
  - Keep high impedance traces short
  - Component layout for best routing scheme
  - Keep layout intuitive for end user
- Best practice followed for Power Delivery
  - High efficiency switching regulators
  - Low-noise shielded components
  - Ground stitching, via fencing
- Best practice followed for software
  - Self-documenting code
  - Consistent variable naming conventions according to Python, C, and VHDL standards
  - Atomic Git commits whenever possible
- Standard format used for lab and manual writeups
  - All labs and manuals written in LaTeX
  - 12pt base font
  - Computer Modern Serif font
  - Each lab includes summary of deliverables at end

Summary of Requirements

- Create a platform (CyDAQ) to supplement hardware portion of labs in signals, systems, and communication classes
  - Include a general purpose platform, expandable to many classes (including both lab activities and lecture demos)
- Develop labs for that utilize CyDAQ hardware to connect concepts to real world application for EE 224 and EE 324
  - Test labs to ensure that each lab is completable by an average student in 2 to 4 hours
- Create a set of final projects for each class
- Implement design elements (filter design, controller design, etc.) in labs

Table of Contents

Introduction

Conceptualizing the signals and systems curriculum at Iowa State University can be as frustrating as it can be rewarding for students. In lecture, professors work hard to help students wrap their minds around this difficult curriculum; however, in the lab, the curriculum has fallen behind and currently lacks both the conceptual support for the lectures and real-world applications that helps prepare Iowa State students for the professional world. In order to revitalize the signals, systems, and communication classes at Iowa State, a redesign of the signal processing laboratory curriculum is needed.

Our team seeks to alleviate student's frustration with the signals and systems curriculum by redesigning the signal processing laboratory curriculum to connect core concepts to real world hardware. To do this, we utilize a hardware teaching platform called CyDAQ. CyDAQ, is a hardware platform designed for laboratory curriculum that contains onboard analog filters, signal acquisition and generation hardware, multiple input sources, and a wide array of general purpose expansion ports called PMODs. The CyDAQ baseboard, shown in Figure 1, is designed as a hardware hat for the Xilinx Zybo Z7.

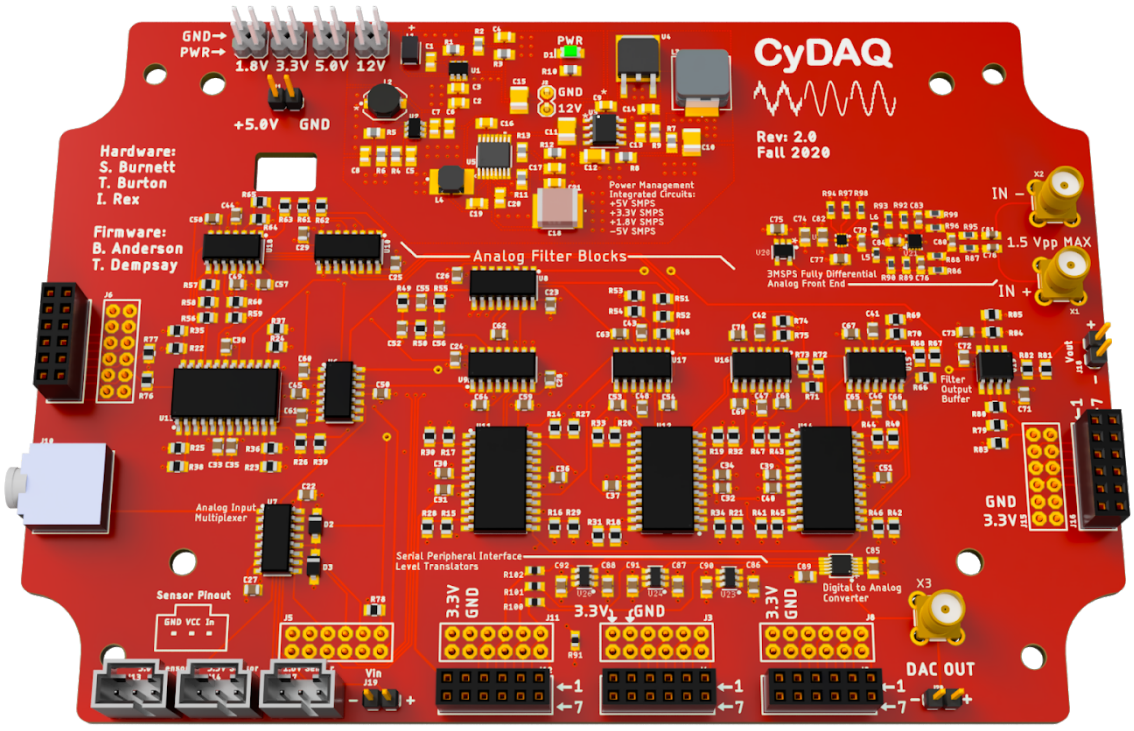


Figure 1: CyDAQ Baseboard

Conceptual Sketch

Figure 2 below helps conceptualize the multiple pieces that make up the CyDAQ teaching platform.

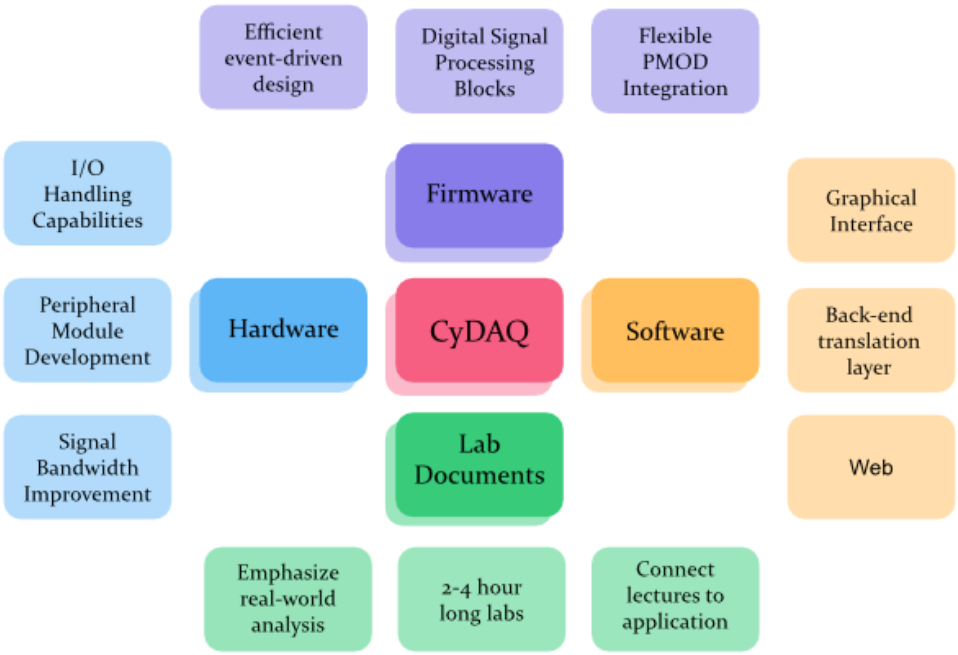


Figure 2: Conceptual Sketch

Functional Requirements

Hardware

- Inputs for a variety of signal sources:
  - General purpose analog differential input
  - 3.5mm TRS auxiliary input jack
  - Special purpose sensors (provided by the team or the ETG)
- Outputs for signal sampling, measurement, and hardware interfacing:
  - General purpose output for signal measurement
  - Stream samples to computer for processing and storage
  - General purpose analog output for interfacing with external hardware
  - General purpose power supply rail for powering external hardware
- On board signal processing
  - Built-in analog filters
  - Ability to implement digital processing of signals
- Multiple input channels for simultaneous analog and digital processing

Software

- The UI must allow the user to configure the CyDAQ's filter specifications and input source
- The UI must allow the user to configure the CyDAQ's DAC by uploading samples of a function
- The UI must allow for developer debugging, having functions of configuration and access not available to the GUI

Non-functional Requirements

Hardware

- Provide design examples of PMOD extensions for adding future functionality
- Implement firmware in a well documented scalable manner to facilitate future expansion
- The final, assembled CyDAQ should cost less than $350
- Provide schematics and board files that conform to industry convention for teaching and reproduction purposes
- All hardware must be able to survive use by inexperienced students
- Externally exposed hardware should survive repeated ESD events

Software

- The firmware of the FPGA must be able to configure and sample the CyDaq board to meet the requirements of current or potential lab applications with an acceptable amount of stability.
- The UI must not allow users to configure the board in ways that are mathematically impossible or unreasonable.

Lab Manual Requirements:

- Labs should have hardware elements and a connection to a real world application
- Labs should be related to and reinforce the course and lecture material
- Labs should be completable by an average student in 2-4 hours
- Final design project should be completable by an average student in 6-8 hours
- All labs should include a summary of deliverables section at the end

Engineering Constraints:

- Bandwidth limited serial communication for ADC and DAC expansions (50 MHz max)
- Voltage input limits (detailed in implementation section below)
- Matlab has limited ability to work with a Python backend

- Labs must be fully functional with and without a full set of stand electrical engineering benchtop equipment (i.e., oscilloscope, power supply, multi-meter), but can use the Digilent Analog Discovery 2 (discussed in implementation section below)
- Time constraints, especially in the face of general purpose rubrics and a plethora of assignments unrelated to our project

Implementation

Operational Environment

The CyDAQ will be operating in a standard lab environment, and is expected to be handled by dozens of people each day. In this environment, every-day wear and tear is expected as well as the breaking of various parts. Although food and drink are not allowed in the lab environment, CyDAQ's exposure to food and drink is expected. To counteract these effects, the CyDAQ is designed to be placed in a transparent blue polycarbonate box, along with the Zybo FPGA, as shown in Figure 3. This transparent casing ensures that damages are kept to a minimum while also allowing students to see the internal components.

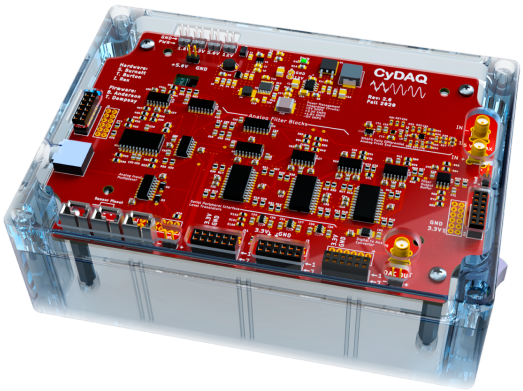


Figure 3: CyDAQ

Limitations

The following limitations are important to note as part of the scope of the project:

- The final assembled CyDAQ shall cost less than $350
- The CyDaq will have a minimum sample period of 1 μs
- All CyDAQ extension modules necessary for *all* labs must fit inside the lab's cabinet storage space
- Input voltage range is limited to $\pm 5V$
- DAC output voltage range is limited to $\pm 5V$
- Filter bandwidth is 1 kHz to 40 kHz

Related Work & Literature

Data acquisition devices are ubiquitous in industry for test engineering, but few are targeted directly at the educational market. The major manufacturers are Digilent and National Instruments. Our project will be loosely benchmarked against NI's Elvis III, as the Digilent

Analog Discovery (DAD), shown in Figure 2, is complementary to CyDAQ and will be used alongside it in labs.

The DAD is a USB oscilloscope and function generator. It is used to provide access to basic and necessary EE tools in labs that do not have access to the full set of industry standard tools in most circuits labs. Digilent also provides the WaveForms software with the DAD which will be used to facilitate learning by generating signals to real hardware systems, measuring those real systems, and visualizing their output in both the time and frequency domains. Adding the DAD's feature set to CyDAQ's provides the complete feature set necessary to implement context based labs and demos for improved teaching and learning.

Figure 4: Digilent Analog Discovery 2 (DAD)

Economics

We were able to significantly increase the functionality of the hardware platform with minimal impact to the overall cost. Moreover, the peripheral module expansion board facilitates the reuse of the revision 1 hardware, requiring no modifications and preventing significant loss and waste. Both the peripheral module and revision 2 board can be mass produced with the fabrication files and materials we have generated, significantly reducing the per-unit cost to the department. For a detailed cost analysis, see Appendix II.

Hardware

Analog Signal Chain Upgrades

The ADC

Hardware Revision 2 of CyDAQ involved two signal handling upgrades. The first is the addition of an analog front-end composed of a high bandwidth, fully differential amplifier, 3MSPS analog to digital converter, and high stability analog reference. The FDA incorporates a multiple feedback filter and 2nd order LC lowpass filter -3dB corner frequency of 1.5MHz and attenuation of 60 dB/decade. This anti-aliasing filter is maximally flat and supports a 3MHz sampling frequency for operation into the lower AM bands.

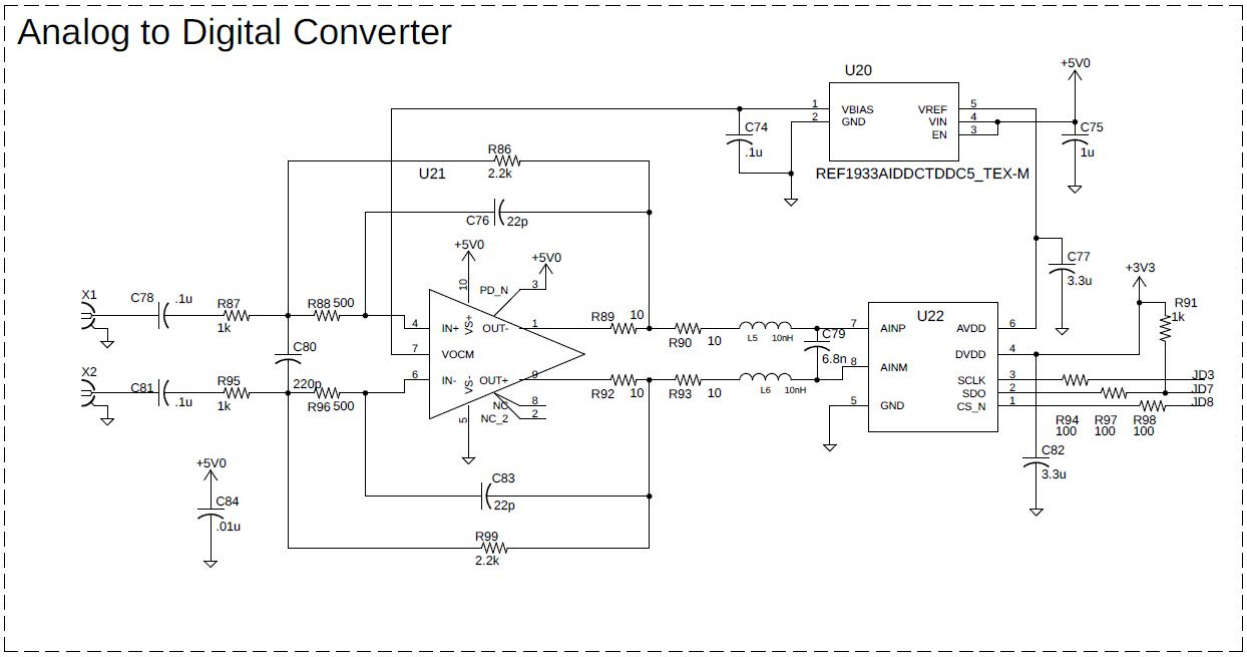


Figure 5: Analog Front-end Schematic

The ADC operates over Serial Peripheral Interface (SPI) at clock speeds up to 50MHz. Both the FDA and the ADC utilize the same analog reference IC, allowing for maximum common mode noise rejection through a shared reference and mid-supply common mode with drift tracking and local bypassing. The fully differential inputs feature coaxial SMA connectors with DC blocking capacitors and 50 Ohm matched terminations. The inverting input can be terminated with a 50 Ohm load for single-ended operation.

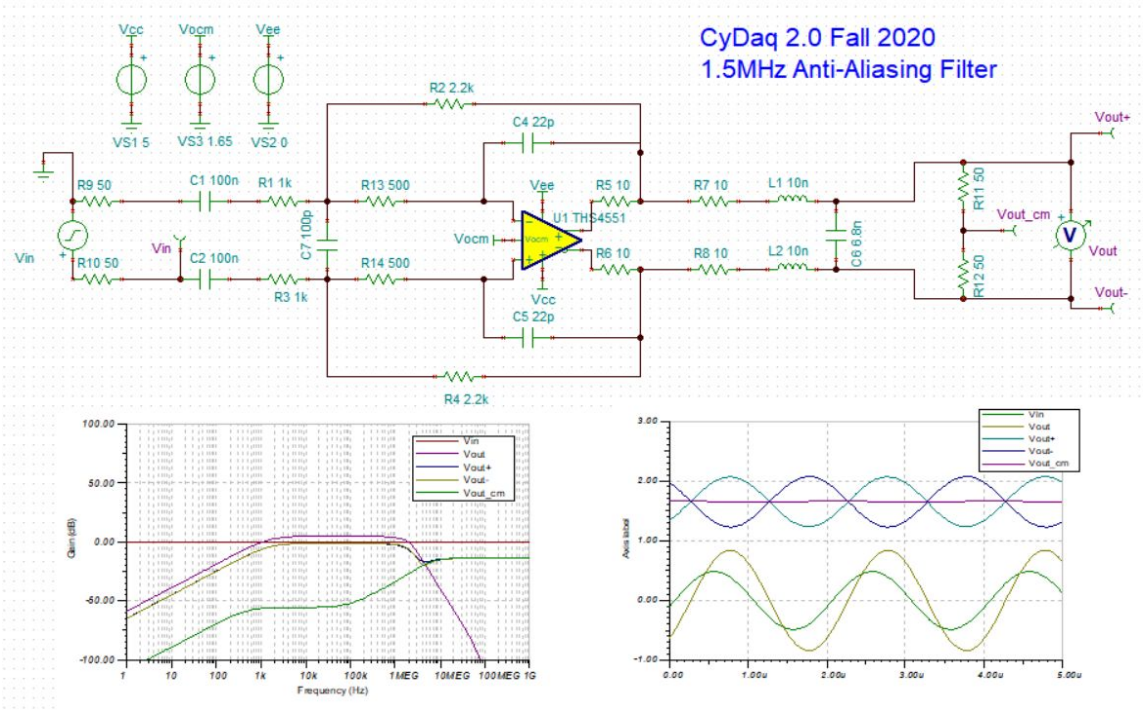


Figure 6: Anti-Aliasing Filter Design

The DAC

The second major hardware design effort was the addition of a SPI communication based digital to analog converter (DAC). The DAC operates at a clock rate of 50MHz and is capable of 16 bit waveform generation. The output is internally buffered and is commensurate with the system signal amplitude of 5Vpp. It has a theoretical maximum output of 1MHz that is dependent on software control optimization. High speed level translators handle logic level communication mismatch between the FPGA and the DAC itself, allowing for 5V output from the DAC and a 3.3V system logic level.

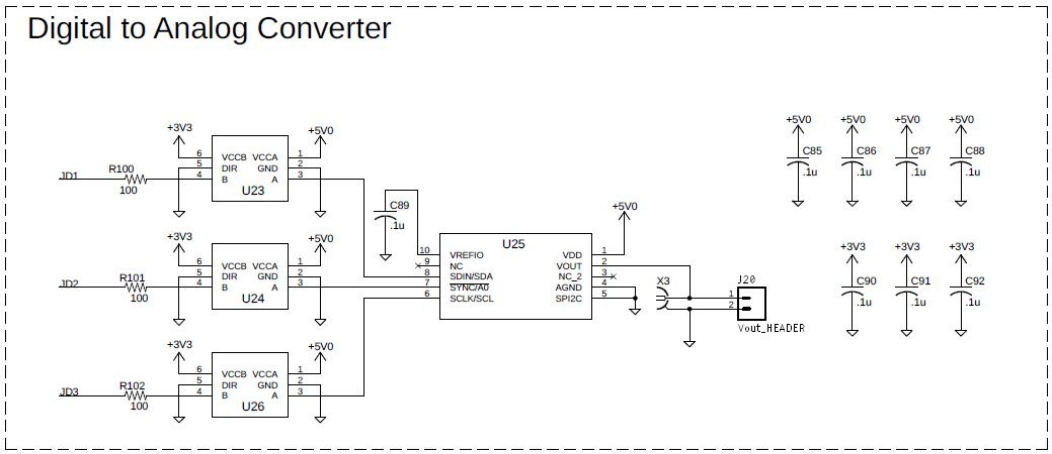


Figure 7: DAC Schematic Capture

PMOD Expansion

The final effort in hardware development was to generate a peripheral module (PMOD) that would expand the functionality of the now-obsolete CyDAQ revision 1 hardware to match revision 2 capabilities. This PMOD not only allows for the continued use of legacy equipment, but provides a tangible design standard for further peripheral hardware development and implementation. Both the ADC and DAC circuits are fully implemented on the same GPIO headers that the new design utilizes, so one firmware build will work for both revisions. The PMOD includes a 5VDC boost converter SMPS that allows the DAC to operate at 5Vpp, and also provides supply rails to the FDA and the analog reference IC.

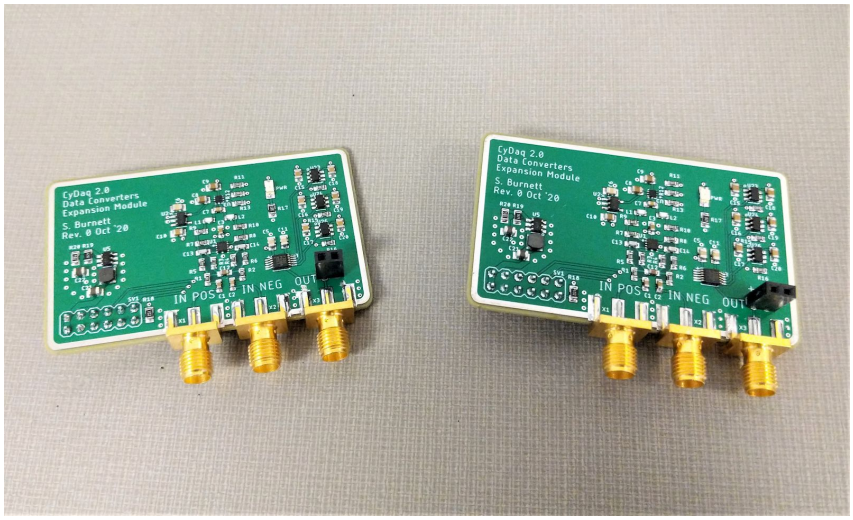


Figure 8: PMOD Expansion Boards

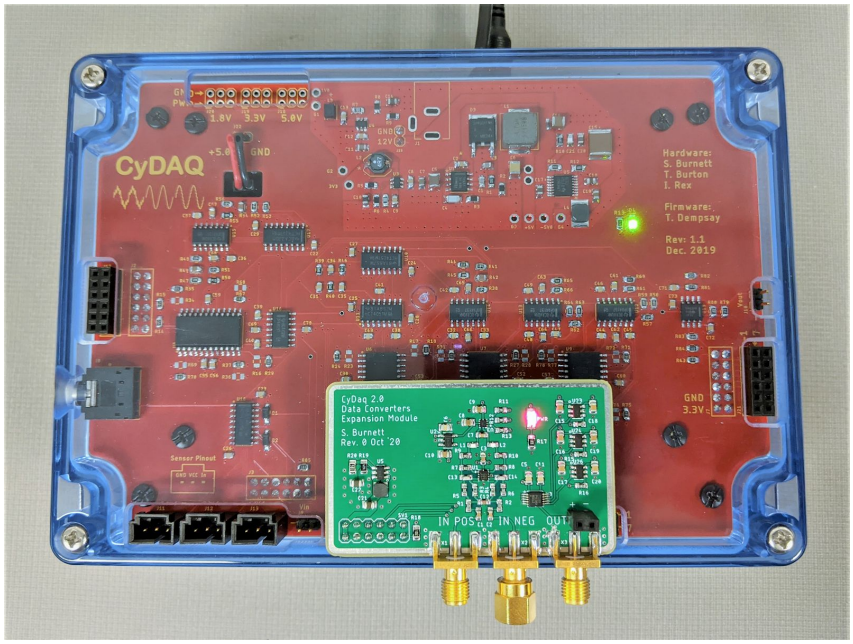


Figure 9: PMOD on CyDAQ Rev 1

Testing and Validation

We were able to verify all functional requirements of the hardware revision and the peripheral module, including SPI communication data rate, input filter bandwidth, sampling functionality, analog signal generation, impedance matching gain conditions, logic level translation, power supply startup, and reference stability. Key results are tabulated below.

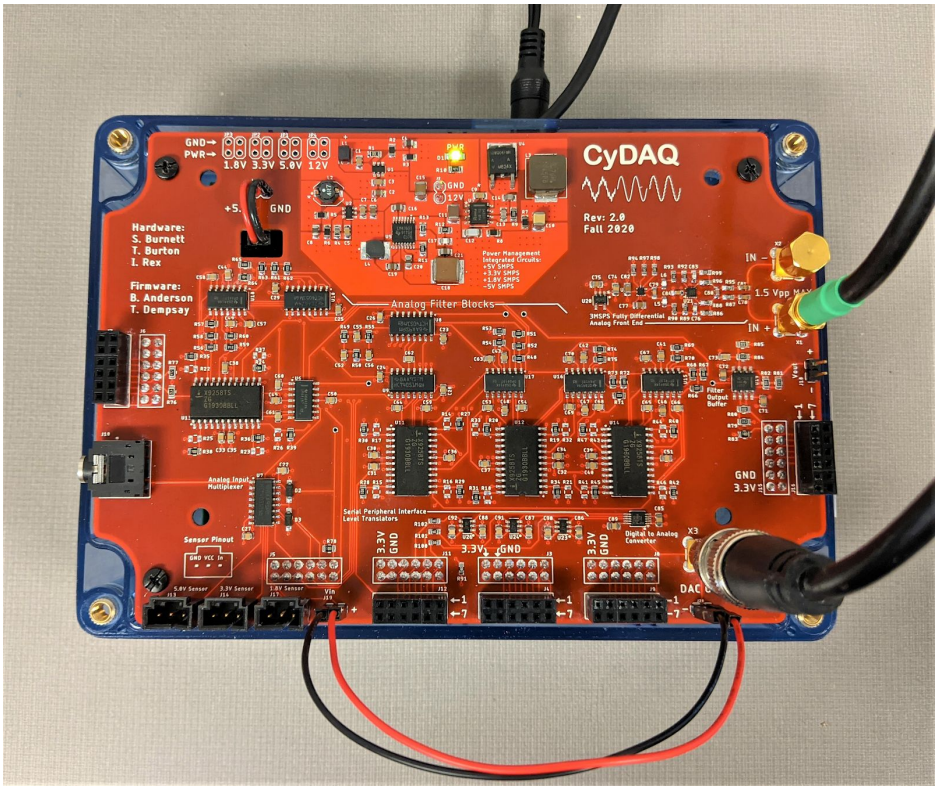


Figure 10: CyDAQ Rev 2 Under Test

Function	**Test**	**Result**
SPI MOSI for DAC Control	Transmit DAC Control Data	50MHz Transfer Successful
SPI MISO for ACD Sampling	ADC Sample Data	50MHz Transfer Successful
DAC Signal Output	16 bit Waveform Generation	16 Bit Waveform Recovered
Anti-Aliasing Filter Design	-3db Corner <= 1.5MHz	1.46 MHz -3dB Measured
ADC Waveform Input	Sinusoidal Input (var. freq.)	Sampled Sinusoid Recovered
ADC Common Mode	1.65 VDC Expected	1.649 VDC Measured
Analog Measurement Reference	3.3 VDC Expected	3.299 VDC Measured

Software

Front-End

Introduction

The front-end of the Mixer uses MATLAB App Designer to quickly and easily make a graphical representation of the parameter selection process that is also available in the command line. MATLAB is able to connect to Python by adding the location of the Python installation and the Python libraries to the Environment Variables in Windows.

Features

1. Students choose the parameter needed for the current lab they are working on in a step-by-step process.
2. Mixer displays the current configuration on the left-side of the window to show their currently selected parameters.
3. Mixer will save and load parameters from previous sessions to reduce the number of times the user goes through the selection process.
4. Mixer allows the user to go back to a previous choice that they have made to make a different selection if they choose.

Back-End

Introduction

The Python backend handles serial communication for the CyDAQ, maintaining the parameter dependencies, and the API that the Mixer uses to keep track of the progress that the user is making during the configuration process.

Features

1. Adheres to the parameter dependencies when making parameter choices.
2. Stores and saves previous configurations for the next session
3. Has a command line interface for all necessary functions in case the Mixer is not available for use.

Implementation

Python represents the dependencies of the parameters with a JSON object containing a graph. The graph contains nodes representing a choice that the user makes, each node has children that describe the next node to traverse to. Choices that the user makes will determine the child that is navigated to. A Python class ‘Constructor’ has been created to walk the user through the parameter decisions that the user has to make. The class gives a ‘Ticket’ containing the choice that the user makes along with its constraints. In return the function ‘input’ is used for the response for the given ticket. A new ticket will only be created if the given input is valid. The MATLAB front-end as well the command line tool takes advantage of this object. Python uses the pyserial library to communicate with CyDAQ.

Testing

The only software tests that have been made are for particular Python functions that are especially prone to error, and functions where errors are especially harmful to the flow of the program. All other functions in the Python back-end have been developer tested in the majority of test cases. For obvious reasons, UI testing the MATLAB front-end is not economically feasible for the scope of this project. So manual testing has been done on the part of the developer to determine if the right behavior is occurring given the current state of the Mixer.

Firmware

Overview

The CyDAQ is powered by a Xilinx Zybo FPGA board with a dual integrated ARM processor. The board features the reprogrammable logic of an FPGA, while also allowing high-level C software to handle system configuration and high-level logic operations. This flexibility makes the board ideal for an ever-evolving piece of lab equipment, but it does make designing new features an involved process.

Figure 11: Zybo Z7

The reprogrammable logic fabric is used to interface with devices on the CyDAQ hardware board. It is configured as a Block Design in Xilinx Vivado, and makes use of variou prebuilt and CyDAQ-custom IP cores.

The dual ARM processor is used for high-level control of the CyDAQ and communication with the lab PCs. It also configures IP cores implemented in the reprogrammable fabric. Its control firmware is written in C and developed using Xilinx SDK.

System Features

CyDAQ contains a wide variety of features, some of which are based on previous iterations of the CyDAQ design. An overview of all CyDAQ features is given below; more detailed information about new features added in the latest design is given later in this section.

Tunable Hardware Filters

The CyDAQ has a bank of hardware high-pass, band-pass, and low-pass filters that allow students to filter waveforms just prior to ADC sampling. They can be tuned with digital potentiometers controlled over an I2C bus.

Selectable Filters and Inputs

The CyDAQ's logic I/O can also be used to select from a variety of input sources for sampling, as well as which filter is currently applied to the input signal. Inputs include a 3.5mm audio in; 1.8V, 3.3V, and 5V sensor inputs; and a fully differential input on the SPI ADC.

Sampling Signals with XADC and SPI ADC

The CyDAQ features two analog-to-digital converters (ADCs) which can be used to sample signals for later manipulation on a lab PC. The Zybo Z7's built-in XADC can sample signals up to 500 kHz, while the external SPI ADC can capture signals over 1MHz. Both ADCs are driven by a timer in the ARM processor, which periodically runs an interrupt service routine (ISR) to store a sample and prime the ADC for sample capture again.

Generating Signals with SPI DAC

In addition to capturing signals, CyDAQ can also generate waveforms with its built in SPI digital-to-analog converter (DAC). Like the ADC systems, the DAC is driven by a timer in the ARM processor which runs an ISR to periodically update the DAC output. Datasets can be uploaded to the CyDAQ through the Mixer PC application and iterated over to create arbitrary waveforms for any lab application.

Lab PC Communication

The CyDAQ operates in conjunction with The Mixer GUI program. The Mixer runs on a lab PC and uses USB to UART communication to send and receive data from the CyDAQ. This communication link allows students to easily configure the CyDAQ for their needs, receive sample data from captured signals, and upload datasets for signal generation.

Newly Implemented Features

CyDAQ is an iterative project - our team is developing the third iteration of the CyDAQ design as we fix issues and add new features. This section provides a discussion of the newly added features and revised features that have been designed over the course of this project.

Interrupt Driven Sampling

The previous CyDAQ design used a polled approach to sample the XADC; the high-level C control software would periodically check the XADC to see if new sample data was available. The approach was highly complex and could not support parallel operations such as sampling with an ADC while generating with the DAC. Going forward, the firmware uses a timer-and-interrupt based design to only interact with the XADC when data is available for capture. The firmware now spends most of its time listening for commands from the lab PC, and spends a little time interacting with the ADC and DAC–only when necessary.

External SPI ADC

The XADC has been sufficient for most CyDAQ applications thus far, but with only 500kHz of bandwidth, some high-frequency applications like AM radio labs were infeasible with the XADC. We have added a new external ADC chip with over 1MHz bandwidth, doubling the sampling capacity of CyDAQ and opening up the potential for high-frequency labs in the future.

Samples are received from the external ADC over a 50 MHz Serial Peripheral Interface (SPI) bus, controlled by a Quad SPI Master in the FPGA's programmable logic fabric. The ADC takes only 96ns to capture a new sample, so samples are available basically immediately upon request from the FPGA. Requests are scheduled by a timer in the ARM processor to guarantee a specific sample rate, much like the new XADC sampling process.

External SPI DAC

High-frequency sampling makes CyDAQ a valuable lab tool, but many interesting lab applications also require function generation. We have addressed this need by adding an external DAC to the SPI bus with a maximum generation rate of 1-2MHz. The DAC is driven by a timer in the ARM processor like the XADC and SPI ADC. Datasets can be uploaded to CyDAQ for the DAC to iterate over, allowing arbitrary waveform generation for maximum flexibility.

Labs

Requirements

All labs should conform to the following requirements:

- Have hardware elements and a connection to an identifiable real world application
- Be completable by an average student in 2-4 hours
- Follow class learning objectives
- Be visually appealing, consistent, and made in LaTeX
- Serve as an example for additional ECpE course labs
- Final design projects should be completable by an average student in 6-8 hours
- 4+ labs created for both EE 224 and EE 324 (8+ labs total)

Lab Documents

Introduction to CyDAQ and DAD – EE 224

In this lab, students are given an overview of the CyDAQ platform and introduced to the basic usage. This is meant to be an introductory lab for students to get comfortable with the platform and basic concepts like signal generation and measurement. It should help new students cement their understanding of the lecture concepts by providing some real-world context of the theory. This includes use cases for the basic signals they learn about (step function, sinusoids, exponentials, etc.) and an introduction to real-world design challenges like EMI and signal integrity.

Introduction to Noise Filtering – EE 224

In this lab, students are given a noisy audio file, and are expected to improve the quality of the audio file using CyDAQ. First, students must use Matlab to plot the original .wav file in the frequency domain. Then, they must use their knowledge of filters to decide which filter best fits their needs: low pass, high pass, band pass, or band stop. They must apply that filter, and be able to hear a noticeable difference between the original audio file and the improved audio file. Finally, students work with the Nyquist Sampling Theorem to determine the minimum sampling rate required.

The Introduction to Noise Filtering lab is a straightforward lab designed to help students understand the basics of filtering. This connects filtering to Fourier Transforms and helps make connections between EE 230 and EE 224. This lab also helps students understand the power of both filtering and the Nyquist Theorem as students are able to easily hear the difference between an unfiltered or undersampled audio file and a properly filtered or sampled audio file.

Introduction to Image Restoration – EE 224

The image restoration lab is an open-ended lab where students are expected to find their own image to restore. The lab document gives students ~10 methods to restore images using Matlab. Students are expected to pick, at minimum, 3 methods, and show the before and after each stage. The methods they choose should help students solve specific issues found in their image such as blurring or noise. The only requirement for students' chosen methods is that there is a visible difference in quality by the end.

This lab hopes to help students with the decision making process of design and implementation and teach them how to find and use the right tool. Since each photo they bring will be unique, the choice of methods they chose should reflect that. Having students choose their own best method helps bring up important questions in the lab such as "why did you choose to follow the steps that you did?" and "what makes different filters and restoration methods effective?"

Figure 12: Left: Low-quality Lenna, Middle: Original Lenna, Right: Restored Lenna

Heart Rate Monitor Design Project – EE 224

In this project lab for EE 224, students are expected to use a pulse sensor and the CyDAQ to sample and measure a heart rate. The sensor chosen for this lab is the Adafruit Pulse Sensor Amped, as shown in Figure 8 to the right. Students are expected to use the datasheet for this sensor to correctly set-up the sensor then take a sample of their heart rate.

Figure 12: Adafruit Sensor Amped

Using that sample, students then calculate the heart rate in both beats per minute (BPM) and Hz. Students also need to calculate the time between beats and the length of time of an

average beat. They then need to display the heart rate waveform in Matlab, and compare the results to a typical heart rate waveform.

The key to this lab is the simple concepts tied in with the more difficult EE 224 ideas. Since most students will enter the lab already knowing what a pulse is and how it is useful, students will then be able to connect that knowledge to concepts that they are less familiar with such as those in EE 224. Having a simple real-world application also helps students focus on the steps it takes to design something since the report instructions are fairly open-ended.

System Dynamics – EE 324

This lab was designed to be Lab 1 in EE 324. This lab allows students to run in depth experiments on two dynamic systems. Their knowledge is solidified as they are asked to derive and explain the differential equations that represent this system.

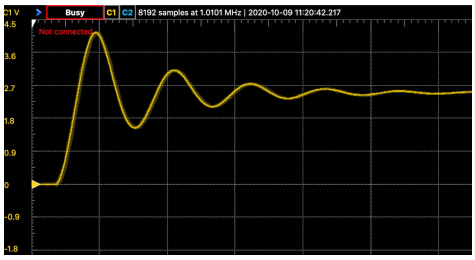


Figure 13: LRC Circuit Step Response

The first dynamic system is an LC circuit that students will build and assemble on a breadboard. Then they will use a Digilent Analog Discovery (DAD) in order to generate a step response and observe the response using the onboard oscilloscope. Next, students will simulate the circuit using MATLAB's Simulink. Students will be able to overlay and compare the data that was received from their circuit measurements and the simulation.

In the second dynamic system, students will utilize a mechanical spring mass damper system and observe the vertical oscillations of this system. These mechanical systems will be outfitted with an analog distance sensor which will send data to a CyDAQ. Using this data, students will be able to derive characteristics of the dynamic system.

Figure 14: Mechanical Oscillator

Controls I: Power Supply Bode Analysis – EE 324

This lab is designed to supplement the end of EE 324 when students get an introduction to basic control theory. One of the concepts they learn about is stability analysis using a Bode plot. This lab will teach students how to measure a real system Bode plot, familiarize them with a very fundamental building block circuit, and introduce them to industry tools like application notes.

The first part of this lab is explaining and building a linear regulator. Though EE 230 provides enough circuit background to explain the operation of the regulator, they are also given Texas Instruments application notes references to help give some more background.

The experienced circuit designer will know that the circuit as given in Figure 11 has low stability margin. The second part of this lab shows students how to measure the loop response of the circuit and observe the gain and phase margins on a Bode plot. They will then stabilize the circuit by adding various compensation components and once again measuring the loop response.

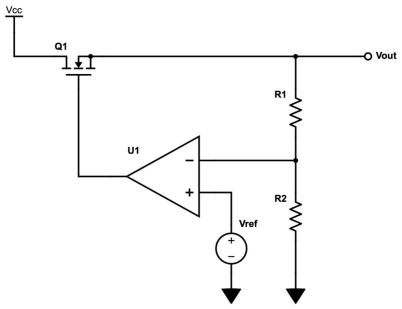


Figure 15: Voltage Regulator Circuit

The final purpose of this lab is to give students a link between the frequency domain analysis and the time domain analysis. To this end, students will measure the transient response of the circuit to a step load. This is an industry standard test procedure for characterizing the response of a power supply. Figure 12 shows measurement examples for a Bode plot and transient response.

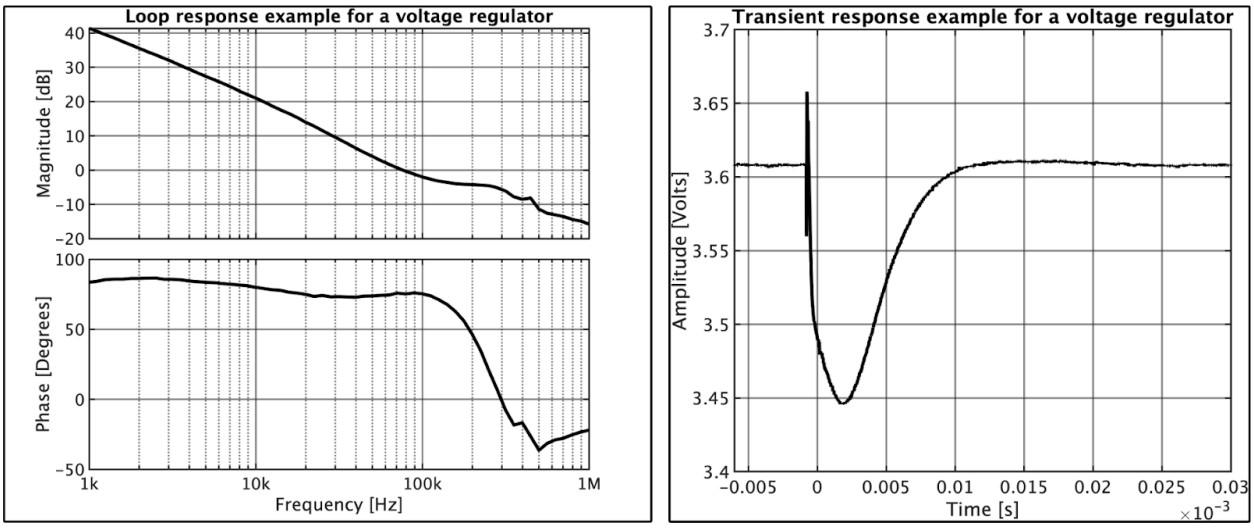


Figure 16: Bode plot and transient response measurement examples

Controls II: Controller Design and Root Locus – EE 324

As one of the last labs in EE 324, this one employs the use of all the skills the student has accumulated. The student will be tasked with designing a PID controller to balance a ball on a beam. The system in question is shown in Figure 17. A ball sits on a beam with sensors on the end to measure the distance of the ball. A servo tilts the beam to adjust the ball to a set position and hold it in place.

The student will use Simulink's design tools to model and simulate this system. They will insert a PID block from the Simulink controls library and visually to the linearized step response.

This will give them Kp, Ki, and Kd values that they can upload to CyDAQ to see how the real system performs.

They will also use root locus analysis to explain the system's behavior around a linearization point. They will analyze the system both before and after the controller is in place. The idea is to connect the root locus plot to a real-world system and observe the physical meaning of the branches of the root locus.

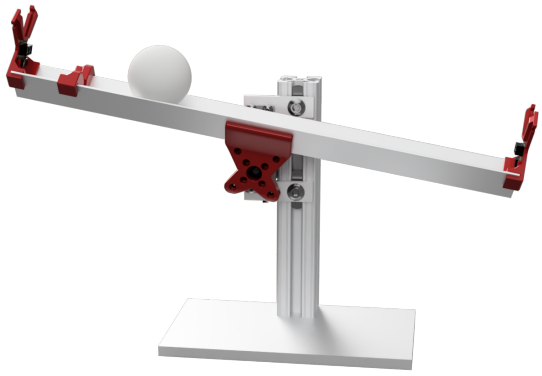


Figure 17: Ball & Beam Balance System

Power Systems Load Frequency Control Design Project – EE324

Since many students in EE 324 take the course as a prerequisite to additional power engineering courses, it's important that these students see how the course applies to power systems. A power system consists of generation, transmission, and distribution systems. When there is a change in load, the frequency and voltage adjust. However, these adjustments must stay within pre-specified limits using a control system.

In this lab, students will use this information to model the generator, load, and turbine then proceed to design the control system for them. This lab heavily uses the application of Laplace transforms to achieve a minimal steady state frequency error within their system.

Testing

Due to COVID-19, the lab documents and CyDAQ platform were not able to enter the labs in the Fall of 2020. Instead, only internal team testing was available. Each lab was tested by members of the team who (1) did not write the lab documents, and (2) do not have extensive experience with signals and systems curriculum.

The following considerations were taken into consideration during testing:

1. Are all steps clear to individuals with little signals and systems experience?
2. Are all documents written with enough detail to satisfy those reading, but not too much detail to overload the reader?
3. Do all documents relate to the course overview and learning objectives for each class?
4. Is any additional hardware or equipment needed?
5. Do all answer keys include explanations and commented code?
6. Are all lab documents written in similar formats with similar language?

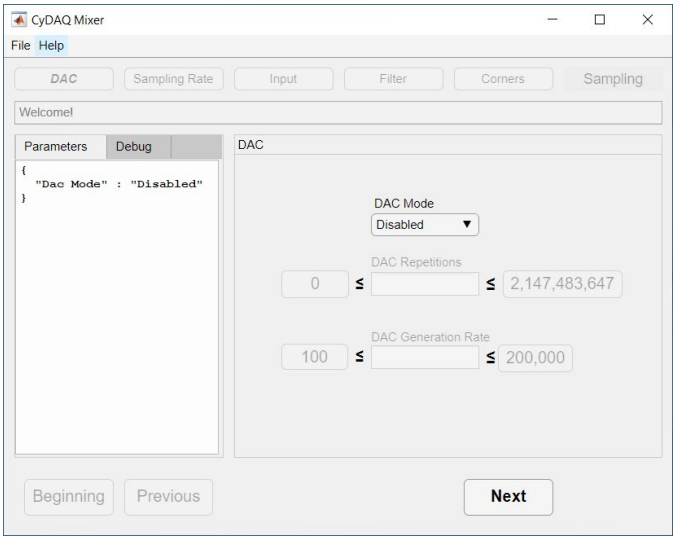
It is through this testing that the CyDAQ team was able to develop documentation to help Teaching Assistants in the laboratory environment.

Conclusion

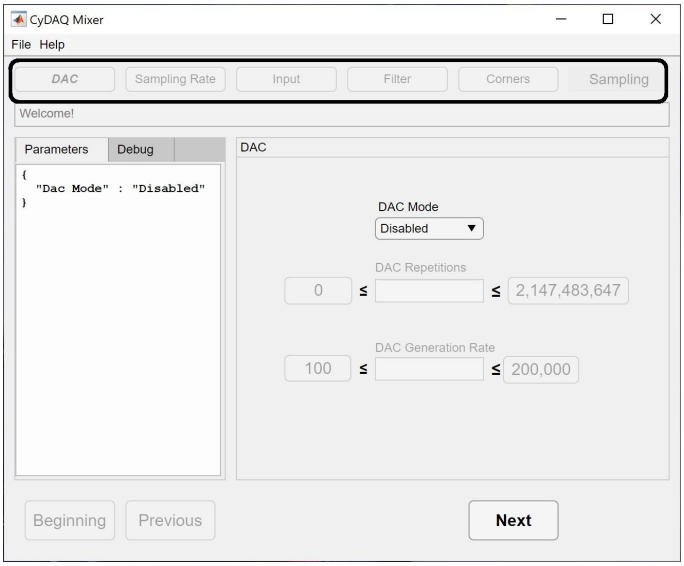
In order to revitalize the signals and systems curriculum at Iowa State University, our team completely redesigned and reworked the laboratory curriculum. At the heart of this redesign is the hardware platform, CyDAQ. Through the past year, the CyDAQ team has implemented new firmware, new software including a completed GUI interface, new hardware revision including the addition of an analog front-end, ADC, and differential amplifier, and eight fully designed labs. This new lab curriculum focuses not only on real-world application, but also allows students to learn different design processes as they are given the opportunity to create design projects. The CyDAQ team is excited to see CyDAQ enter the curriculum in Spring 2021 and has the potential to expand its use to classes all across the ECpE department.

Appendix I: Mixer Operation Manual

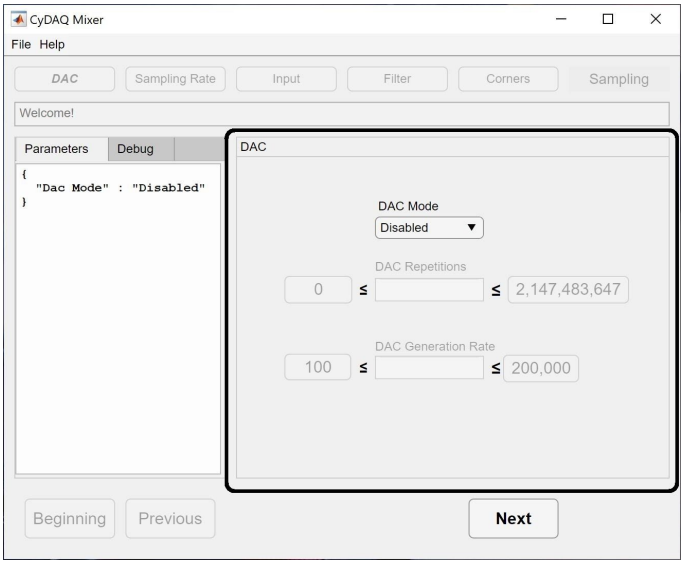
There are two main components to the software that the students interact with to control the CyDAQ. The Python back-end and the MATLAB front-end. The back-end handles the serial communication between the lab computer and the CyDAQ, and in the background handles the process for maintaining the requirements that the parameters have. As some parameters have constraints that must be adhered to, some are incompatible with others, and some are required in the presence of other parameters.



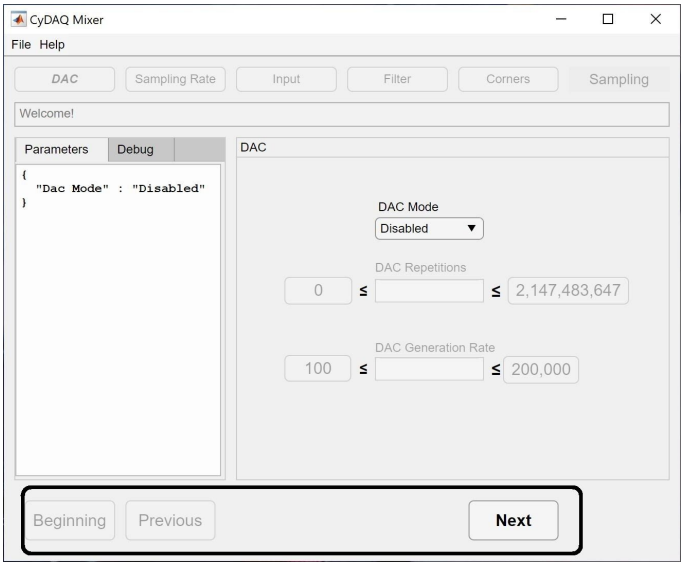
The parameters are picked by the students in a step-by-step process, in order to simplify the process, and to adhere to the constraints of the parameters. The parameters that the students choose are: DAC, Sampling Rate, Input, Output, Filter, and Corners.



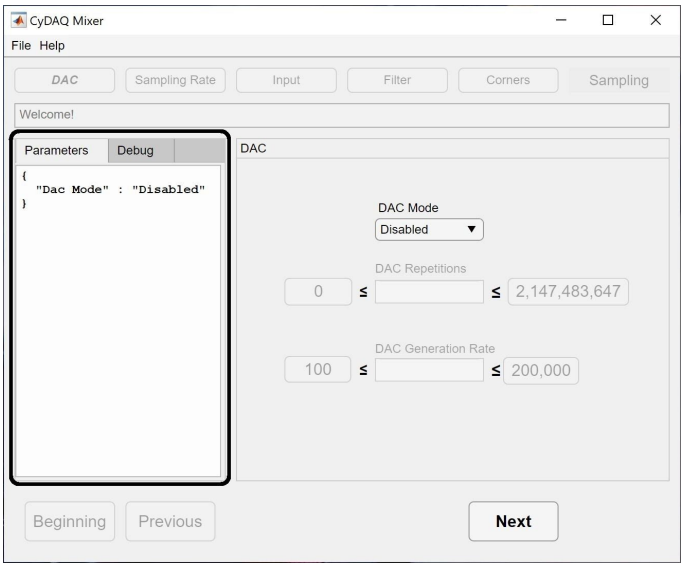
The right figure shows the Navbar at the top of the window. These buttons show the progress that the user is making while selecting parameters. If applicable, the user may click on the button to go back to that portion of the parameter selection process to make a change if desired. The current page that the user is on has the Navbutton bolded and italicized.



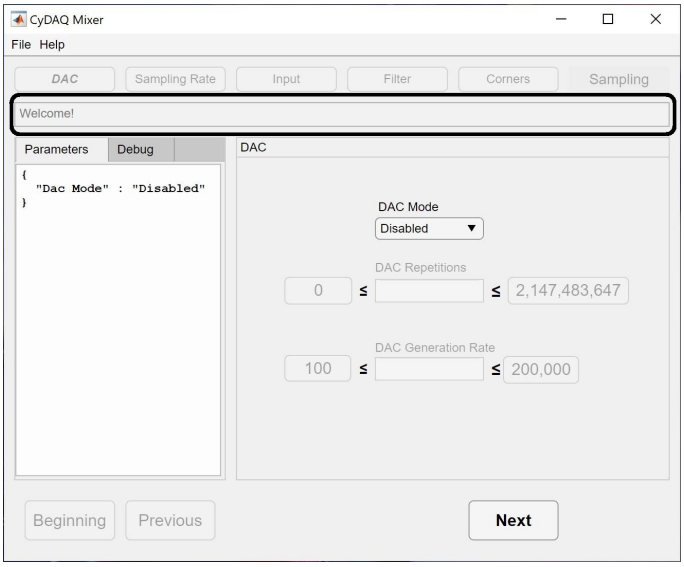
This figure highlights the current page that the user is on. The page contains form items that the user has to fill in correctly with the appropriate parameter to make forward progress. The page may contain one or more form items as multiple choice boxes, a number input, or a dropdown menu. The choices that the user makes will affect the future parameter choices that they will have. For example, in the figure DAC Mode is selected to be 'disabled', therefore, the user does not have the option to select DAC Repetitions, or DAC Generation Rate.



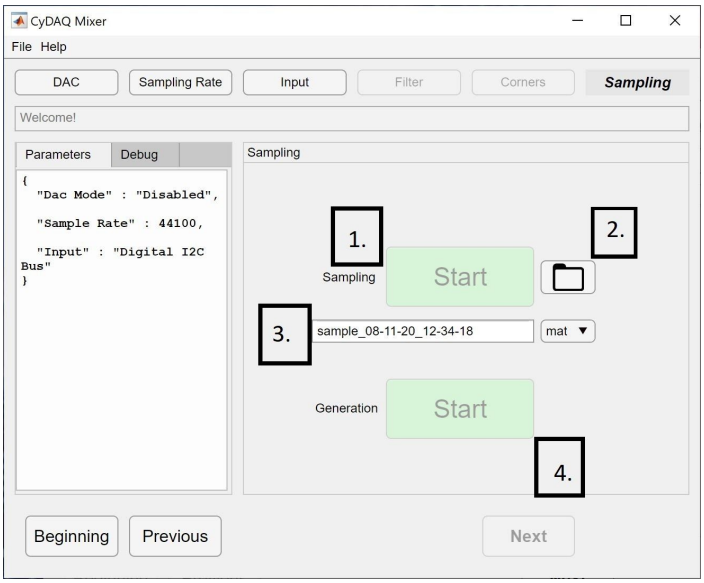
Below are three navigation buttons that are used to make forward or backward progress in the parameter selection process. These buttons are disabled or enabled if the action is allowed. The text of the button and its function should be apparent.



The Sidebar contains two tabs for the users, developers, and TA's to use. The 'Parameters' tab reflects the choices that the user has made in JSON from and will update in real time. The debug tab will display messages and errors that occur for troubleshooting and development.



The messages bar tells the user errors and other information that should be available in its immediacy. So that mistakes made by the user are not perpetuated and corrected as soon as possible.



1. The sampling Start and Stop button. When the user stops sampling, the mixer will fetch the samples from the CyDAQ and not allow any sampling until completed.
2. The button will open the file explorer to the location that the samples will be saved to.
3. The user is allowed to input a filename that the samples will be saved to, and the user may choose between a csv or a mat file extension.
4. The generation button starts and stops the DAC. It cannot be used if the parameters selected for ‘DAC Mode’ is Disabled’.

Appendix II: Scrapped Attempts

Free RTOS

The CyDAQ performs a variety of functions, many of them at high speeds in parallel for long periods of time. Designing a system that can reliably share CPU time between tasks is not easy, especially as more time-sensitive features get added in future revisions of the CyDAQ. At first glance, this seems to be a problem neatly solved by a real-time operating system, or RTOS. RTOS architectures feature automatic task scheduling and switching, task priority systems, and easily configurable task periods - all apparently perfect for a high-performance application such as the CyDAQ. The CyDAQ firmware needed to be improved from its previous polled approach, so we decided it would be worthwhile to design the next generation of firmware around FreeRTOS, which has builds targeting the Zybo Z7.

Unfortunately, we quickly discovered that FreeRTOS is not designed for high-frequency applications like the CyDAQ, but is intended more for low speed robotics applications that update at 1 kHz or less. In fact, the automatic task scheduler built into FreeRTOS is not capable of running much faster than 1 kHz, which is not acceptable in an application which must switch between tasks at a rate close to 1 MHz. FreeRTOS does provide functions to manually switch context to a different task, but manually controlling the running task defeats the purpose of using an RTOS in the first place. In fact, such an application has largely reduced functionality because of the additional overhead FreeRTOS requires - an acceptable tradeoff if the automatic task switching can be leveraged, but largely useless otherwise.

With these considerations in mind, we stepped back and decided FreeRTOS would not improve the CyDAQ firmware design. Instead, we opted to use Interrupt Service Routines (ISRs) triggered by timers to achieve reliable task switching without as much overhead as FreeRTOS would have required. The end result is a reliable system that is much faster than FreeRTOS and much more efficient than the previous CyDAQ firmware design.

Original Design Doc

Introduction of Real-World Signals and Systems into ECpE DSP Laboratory Curriculum

DESIGN DOCUMENT

Team Number: 14

Client: Matt Post

Advisers: Matt Post

Team Members/Roles:
Brady Anderson - Embedded Systems Engineer
Sam Burnett - Hardware Design Engineer
Mitchell Hoppe - Software Engineer
Max Kiley - Manager of Information
Emily LaGrant - Testing and Integration Engineer
Isaac Rex - Hardware Design Engineer

Team Email: sddec20-14@iastate.edu
Team Website: sddec20-14.sd.ece.iastate.edu

Revised: 2/23/20 -- Version 1

Executive Summary

Development Standards & Practices Used

Standard industry convention was used in all hardware and software designs. These include:

- Best practice followed for schematic
  - Use standard signal flow model
  - Separate high level design blocks in to multiple sheets
  - Use standard practice symbols and circuit block drawings
- Best practice followed for PCB layout
  - Keep high impedance traces short
  - Component layout for best routing scheme
  - Keep layout intuitive for end user
- Best practice followed for Power Delivery
  - High efficiency switching regulators
  - Low-noise shielded components
  - Ground stitching, via fencing
- Best practice followed for software
  - Self-documenting code
  - Consistent variable naming conventions according to Python, C, and VHDL standards
  - Atomic Git commits whenever possible
- Standard format used for lab and manual writeups
  - All labs and manuals written in LaTeX
  - 12pt base font
  - Computer Modern Serif font

Summary of Requirements

- Create a platform (CyDAQ) to supplement hardware portion of labs in signals, systems, and communication classes
  - Include a general purpose platform, expandable to many classes (including both lab activities and lecture demos)
- Develop labs for that utilize CyDAQ hardware to connect concepts to real world application for EE 224 and EE 324
  - Test labs to ensure that each lab is completable by an average student in 2 to 4 hours
- Create a set of final projects for each class
- Implement design elements (filter design, controller design, etc.) in labs

Applicable Courses from Iowa State University Curriculum

List all Iowa State University courses whose contents were applicable to your project.

Electrical Engineering: *EE 201, EE 224, EE 230, EE 285, EE 321, EE 324, EE 422, EE423, EE 475, EE 476, EE 333*

Computer Engineering: *CPRE 281, CPRE 288, CPRE 381, CPRE 488*

Computer Science: *COMS 309, COMS 319, COMS 311*

English: *ENGL 314*

New Skills/Knowledge acquired that was not taught in courses

Technical Knowledge/Skills:

- Hardware design experience
- Hardware manufacturing (assembling, soldering)
- Integrating various aspects of a project into a final product
- Testing and safety procedures for hands-on experiments
- Lab writing

Non-technical Knowledge/Skills:

- Organization and communication skills
- Creative design
- Empathy and emotional intelligence
- Self reflection
- Ethical Leadership Skills
- Cultivating Strategic Communities
- Integrating E-business Solutions
- Orchestrating Leveraged Infrastructures
- Bleeding-Edge Quality Vectors
- Creating Seamless Initiatives
- Delivering One-to-One Sprints

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Acknowledgement

Our team would like to thank the Electronics and Technology Group for assistance with equipment, technical advice, and financial aid. We would also like to thank ECpE faculty members Dr. Andrew Bolstad, Dr. Julie Dickerson, and Dr. Shawana Tabassum for their technical support. Without their contributions, our project would not have been possible.

1.2 Problem and Project Statement

Conceptualizing the signals and systems curriculum at Iowa State University can be as frustrating as it can be rewarding for students. In lecture, professors work hard to help students wrap their minds around this difficult curriculum; however, in the lab, the curriculum has fallen behind and currently lacks both the conceptual support for lecture and the real-world application that helps prepare Iowa State students for the professional world. In order to revitalize the signals, systems, and communication classes at Iowa State, a redesign of the signal processing laboratory curriculum is needed.

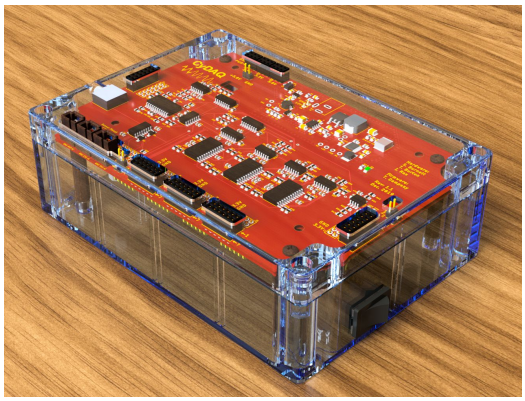


Figure 1: CyDAQ board

At the heart of this redesign is the CyDAQ. CyDAQ, shown in Figure 1, is a hardware teaching platform designed for laboratory curriculum. This board contains onboard analog filters, signal acquisition hardware, and a fully featured powerful embedded computing platform called the Zybo Z7. The Zybo Z7 enables expansion of the CyDAQ base platform by adding external special purpose hardware modules.

CyDAQ has the potential to solve a wide variety of issues found in Iowa State's signal processing course, including but not limited to:

Problem

- Disconnect between lecture material and real world application
- Design skills and techniques not emphasized
- Very difficult concepts to internalize and solidify knowledge for future use in class and industry
- Current labs are pure simulation/Matlab without hardware or real-world connections

Solution

- Show real world application of concepts through hardware experiments
- Introduce design projects that allow the student to learn design process
- Create intuitive sense of concepts by relating them to physical phenomena
- Use CyDAQ and Matlab together to show how these tools can be used in the analysis and design process

CyDAQ is a powerful learning tool that not only has the ability to revitalize the signals, systems, and communication curriculum but also has the potential to spread across the entirety of ECpE curriculum, both inside lecture and labs. By the end of this project, our team hopes to accomplish the following:

- Design a minimum of 4 labs for EE 224 to be implemented in the Spring 2020 semester
- Design a minimum of 4 labs for EE 324 to be implemented in Spring 2021 semester
- Design a minimum of 2 projects for each EE 224 and EE 324
- Develop a student friendly GUI for CyDAQ platform
- Design and deploy several hardware extensions for CyDAQ, as needed for the designed labs
- Develop lab documents emphasizing real-world analysis and design processes
- Design CyDAQ as a complete learning platform that can be expanded to more classes and labs in the future

1.3 Operational Environment

The CyDAQ will be operating in a standard lab environment, and is expected to be handled by dozens of people each day. In this environment, every-day wear and tear is expected as well as the breaking of various parts. Although food and drink are not allowed in the lab environment, CyDAQ's exposure to food and drink is expected. To counteract these effects, the CyDAQ is designed to be placed in a transparent blue polycarbonate box, along with the FPGA. This transparent casing ensures that damages are kept to a minimum while also allowing students to see the internal components.

1.4 Requirements

Functional Requirements:

Hardware:

- Inputs for a variety of signal sources:
  - General purpose analog differential input
  - 3.5mm TRS auxiliary input jack
  - Special purpose sensors (provided by the team or the ETG)
- Outputs for signal sampling, measurement, and hardware interfacing:
  - General purpose output for signal measurement
  - Stream samples to computer for processing and storage
  - General purpose analog output for interfacing with external hardware
  - General purpose power supply rail for powering external hardware
- On board signal processing
  - Built-in analog filters
  - Ability to implement digital processing of signals
- Multiple input channels for simultaneous analog and digital processing
- Provides interface for expanding the core system to suite future needs

Software:

- The firmware of the FPGA must be able to configure and sample the CyDaq board to meet the requirements of current or potential lab applications with an acceptable amount of stability.
- The UI of the board will allow the user to configure and sample the board using the different I/O and filters available on the board.
    - The UI must not allow users to configure the board in ways that are mathematically impossible or unreasonable.
    - The UI must have a GUI to make it easier for lab students to configure the board.
    - The UI must allow for developer debugging, having functions of configuration and access not available to the GUI

Economic Requirements:

- The final, assembled CyDAQ should cost less than $350

Lab Manual Requirements:

- Labs should have hardware elements and a connection to a real world application
- Labs should be completable by an average student in 2-4 hours
- Final design project should be completable by an average student in 6-8 hours

1.5 Intended Users and Uses

The intended users for all products of this project are electrical and computer engineering students. CyDAQ use will begin in the EE 224 lab during week 10 and continue for the remainder of the Spring 2020 semester. The planned topics and general uses for the CyDAQ in lab curriculum can be found in *Table 1*. This lab curriculum is intended to helpEE 224 students understand the learning objectives found in the course's syllabus.

Table 1:

Week*	EE 224 Lab Topic
10	CyDAQ basics, signal acquisition, time and frequency analysis
11	Digital filtering of a noisy signal
13	Aliasing
14 & 15	Two week project to filter a signal and provide actionable information

*Weeks not included have non CyDAQ labs

By Spring 2021, integration of labs for EE 324 will begin. These labs will be functionally the same as those for EE 224, but covering EE 324 topics. The content of the labs will cover the learning objectives for EE 324: analysis, design, and implementation of Laplace and Z transforms, signal filters, and feedback system.

1.6 Assumptions and Limitations

Assumptions:

- The CyDAQ will fit in a polycarbonate box

- The design team will create 4+ lab activities for both EE 224, EE 324, and potentially additional classes
- The CyDAQ will be used by students in a lab environment for educational purposes.
- The CyDAQ will always be available for students to access in the signals and systems labs

Limitations

- The final assembled CyDAQ shall cost less than $350
- The CyDaq will have a minimum sample interval of 0.0000010 seconds per sample
- All CyDAQ extension modules necessary for *all* labs must fit inside the lab's cabinet storage space
- Input voltage range is limited to $\pm 5V$
- DAC output voltage range is limited to $\pm 5V$
- Filter bandwidth is $1\ kHz$ to $40\ kHz$

1.7 Expected End Product and Deliverables

There are three primary categories of deliverables for this project: hardware interface, software interface, and lab manuals.

The hardware interface consists of the main CyDAQ platform and "PMOD" hardware expansions as required by individual labs. The main CyDAQ platform is the core of the project, offering students a hardware interface for studying signal processing and control concepts. It is a complete, self-contained unit intended to be stationed at each lab bench: one per computer. It contains the main CyDAQ board with six analog filters, a full hardware interface as detailed in section 1.4 and, a Zybo Z7 embedded system platform for digital control and processing. The initial, functional revision will be due by 3/23/2020. A second revision of the main CyDAQ platform will be due by 12/2020.

Some labs may require additional hardware (e.g., an FM radio receiver, a microphone/speaker, external push-buttons, etc.). These will be delivered in the form of a "PMOD" expansion module. These modules are boards designed to connect to the CyDAQ's expansion ports to increase the scalability of the platform. They will be designed alongside their appropriate lab and can be used to future expansion of the platform. The PMODs will be due on a per-lab basis–see the lab section below for due dates.

The software interface will be a front-end GUI and back-end command line used to set up and communicate with the CyDAQ. The GUI will be an intuitive interface written in Python for quickly setting up the CyDAQ and transceiving data. The command line will be a lower level interface for developing and scripting of the CyDAQ and Zybo. This interface for the CyDAQ will be provided for labs in the form of a .EXE file, such that no dependencies will have to be installed on the lab computers.

The lab manuals will be a full description of a lab experiment for EE 224 and EE 324. They will be written to fulfill the learning objectives of the respective course and provide the student with an intuitive connection between the theory and the application. There will be a minimum of four labs and two projects per class. The lab descriptions will be accompanied by a full solution set per lab.

2. Specifications and Analysis

2.1 Proposed Approach

FPGA Integration using FreeRTOS real-time operating system to handle signal routing, sampling, digital signal processing, and digital to analog conversion tasks simultaneously.

Peripheral module development not limited to:

- Additional digital and analog inputs/outputs with signal conditioning and level shifting
- Radiofrequency couplers, baluns, and antenna interfaces
- Tactile user input and control interfaces
- Signal visualization, indication devices
- Specialized sensor interfaces

Hardware improvements to include on-board digital to analog converter, additional configurable power delivery, and programmable light emitting diodes.

Lab resource development in concert with relevant faculty and in support of established curricular learning objectives. Emphasis will be placed on hands-on intuition of signal acquisition, processing and feedback using the Cydaq as a facilitative device.

2.2 Design Analysis

Current Design Characteristics

Hardware:

- +5V, -5V, +3.3V and +1.8V power delivery
- I2C protocol based filter block configuration
- 100kHz stable passthrough bandwidth
- 40kHz programmable filter bandwidth
- 12bit analog to digital conversion at 1Ms/s
- Limited to single channel input
- 6 programmable filter banks
- 1 channel passthrough

Firmware:

- Bare-metal C application on Zybo PS
- Simple command string formatting with little error detection
- Single-threaded operation, data acquisition and transmission must be done sequentially instead of in parallel
- Data sampling in Zybo PL fabric via Xilinx XADC IP core

Software:

- Rudimentary graphical interface with limited functionality
- Single-threaded tasks for sampling the XADC

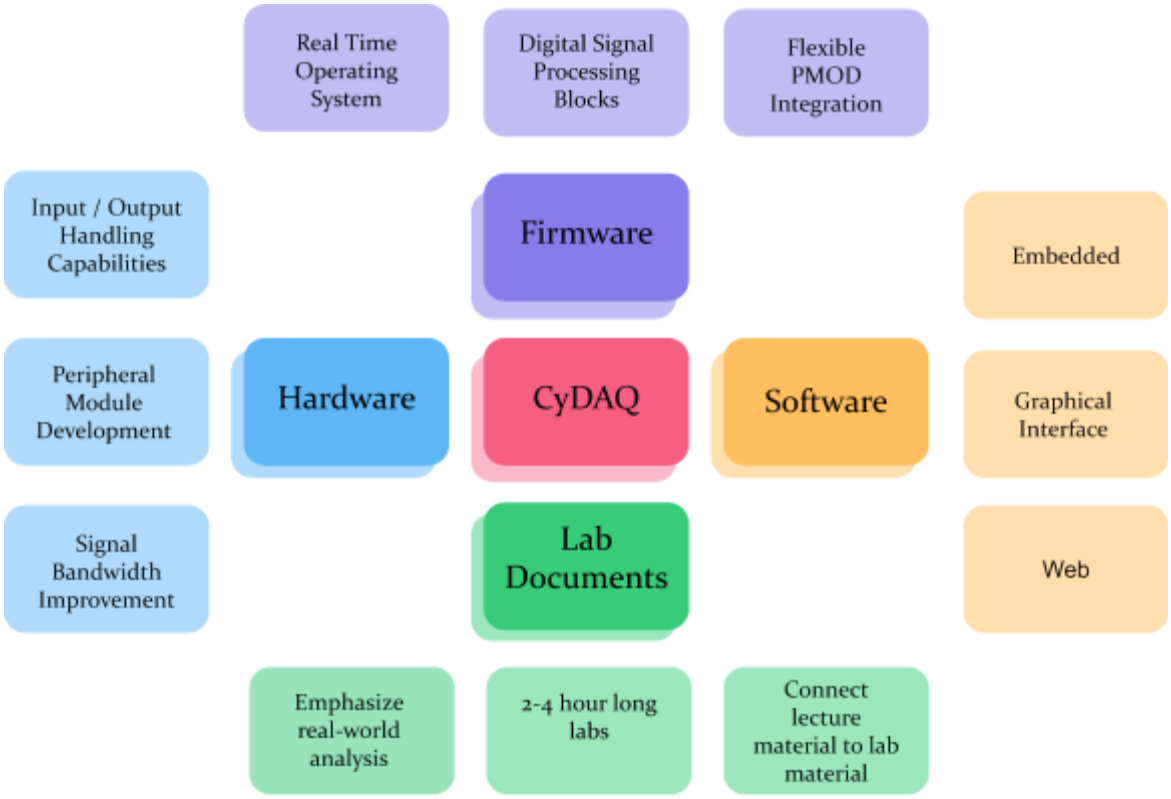
- Python TKinter for the graphical components.
- JSON for configuration and data storage

2.3 Development Process

A combination of the waterfall and agile development processes will be used to coordinate the team's progress. High-level development will follow the waterfall process, in that CyDAQ system requirements will be determined early on before the implementation and testing phases begins. This is primarily because faculty will be integrating CyDAQ and it's associated labs into their courses, so the feature set of CyDAQ must be determined early to give them sufficient integration time.

Day-to-day development will follow the agile process model; this will enable team members to independently work on tasks in their focus area, while ensuring consistent progress is achieved each week. A Trello board will be used to coordinate team members and to keep tasks flowing through the development pipeline.

2.4 Conceptual Sketch



3. Statement of Work

3.1 Previous Work And Literature

Data acquisition devices have existed for a long time, but few are targeted directly at the educational market. The major manufacturers are Digilent and National Instruments. Our project will be loosely benchmarked against NI's Elvis III. More research needs to be done in this area.

The distinguishing features of the CyDAQ platform are: low cost, purpose built for the educational environment, smaller footprint, and lab's developed along side the CyDAQ for optimal integration.

3.2 Technology Considerations

The Zybo Z7 is an affordable FPGA board, but lacks some of the features of more advanced Xilinx boards, such as the expanded I/O of the ZedBoard, or the real-time digital video transcoding of the Zynq UltraScale+ MPSoC chipset.

Using a Real-Time Operating System (RTOS) to schedule data acquisition will expand the capability of the sampling interface, but will introduce significantly more software complexity and overhead than writing a bare-metal application.

The hardware filter signal flow allows for great flexibility of signal acquisition, but does not have any means of outputting a signal other than the direct input passthrough.

3.3 Task Decomposition

All team members have been assigned technical and non-technical roles within the group this creates a natural delegation of tasks within each of the technical divisions of the project. Non-technical work is distributed as needed and all members understand the need for collaboration and synergistic workflows.

3.4 Possible Risks And Risk Management

Team members being unknowledgeable for completing a particular task is a possible risk. Not all experience in the required areas is covered in the skillset of the group. The solution to this is group members will learn new skills and technical abilities as the need is encountered.

Adding hardware components to an expensive FPGA platform carries the possibility of damaging its electronic systems. Team members must be knowledgeable about the design of hardware components and their pinouts before attempting to connect them to the Zybo.

3.5 Project Proposed Milestones and Evaluation Criteria

3.6 Project Tracking Procedures

The group will use Trello to track current work items, assign members to specific items, and track completion rates for the work items. New work items are created from consensus from the group during the weekly technical meetings. Code changes will be tracked through git version control. Code reviews and versioning will be handled through Gitlab. Where new changes have to be approved by other members of the group before they go out to production.

3.7 Expected Results and Validation

The ultimate outcome of this project will be an increase in student's understanding and enjoyment of the signals and system curriculum. This will be measured by performing test implementations in the labs during the Spring and Fall 2020 semesters.

4. Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline

• A realistic, well-planned schedule is an essential component of every well-planned project

• Most scheduling errors occur as the result of either not properly identifying all of the necessary activities (tasks and/or subtasks) or not properly estimating the amount of effort required to correctly complete the activity

• A detailed schedule is needed as a part of the plan:

– Start with a Gantt chart showing the tasks (that you developed in 3.3) and associated subtasks versus the proposed project calendar. The Gantt chart shall be referenced and summarized in the text.

– Annotate the Gantt chart with when each project deliverable will be delivered

• Completely compatible with an Agile development cycle if that's your thing

How would you plan for the project to be completed in two semesters? Represent with appropriate charts and tables or other means.

Make sure to include at least a couple paragraphs discussing the timeline and why it is being proposed. Include details that distinguish between design details for present project version and later stages of project.

4.2 Feasibility Assessment

Realistic projection of what the project will be. State foreseen challenges of the project.

4.3 Personnel Effort Requirements

Include a detailed estimate in the form of a table accompanied by a textual reference and explanation. This estimate shall be done on a task-by-task basis and should be based on the projected effort required to perform the task correctly and not just "X" hours per week for the number of weeks that the task is active

4.4 Other Resource Requirements

Identify the other resources aside from financial, such as parts and materials that are required to conduct the project.

4.5 Financial Requirements

Hardware development support including DAC and PMOD prototyping and PCB revision expenses.

5. Testing and Implementation

Testing is an **extremely** important component of most projects, whether it involves a circuit, a process, or a software library

Although the tooling is usually significantly different, the testing process is typically quite similar regardless of CprE, EE, or SE themed project:

1. Define the needed types of tests (unit testing for modules, integrity testing for interfaces, user-study for functional and non-functional requirements)
2. Define the individual items to be tested
3. Define, design, and develop the actual test cases
4. Determine the anticipated test results for each test case 5. Perform the actual tests
6. Evaluate the actual test results
7. Make the necessary changes to the product being tested 8. Perform any necessary retesting
9. Document the entire testing process and its results

Include Functional and Non-Functional Testing, Modeling and Simulations, challenges you've determined.

5.1 Interface Specifications

– Discuss any hardware/software interfacing that you are working on for testing your project

5.2 Hardware and software

– Indicate any hardware and/or software used in the testing phase

– Provide brief, simple introductions for each to explain the usefulness of each

5.3 Functional Testing

Examples include unit, integration, system, acceptance testing

5.4 Non-Functional Testing

Testing for performance, security, usability, compatibility

5.5 Process

– Explain how each method indicated in Section 2 was tested

– Flow diagram of the process if applicable (should be for most projects)

5.6 Results

– List and explain any and all results obtained so far during the testing phase

- – Include failures and successes
- – Explain what you learned and how you are planning to change it as you progress with your project
- – If you are including figures, please include captions and cite it in the text

• This part will likely need to be refined in your 492 semester where the majority of the implementation and testing work will take place

-**Modeling and Simulation**: This could be logic analyzation, waveform outputs, block testing. 3D model renders, modeling graphs.

-List the **implementation Issues and Challenges**.

6. Closing Material

6.1 Conclusion

In order to revitalize the signals and systems curriculum at Iowa State University, a major rework of laboratory curriculum is required. The CyDAQ is at the heart of this rework and can lead to students successfully applying lecture content to real-world application. This new lab curriculum focuses not only on real-world application but also allows students to learn different design processes as they are given the opportunity to create design projects. CyDAQ has the potential to solve these issues found in the curriculum and more as CyDAQ has the potential to expand its use to classes all across the ECpE department.

6.2 References

None at this time.

6.3 Appendices

None at this time.

Appendix III: Bill of Materials

Qty	Value	Package	Parts	Description	BOM	MANUFACTURER PART NUMBER	MF	DIGI-KEY PART-NUMBER
1	470nF	C0805	C19	CAP CER 0.47UF 25V X5R 0805		TMK212BJ474KD-T	Taiyo Yuden	587-1290-1-ND
1	10.2k	R0805	R7	RES SMD 10.2K OHM 1% 1/8W 0805		CRCW080510K2FKEA	Vishay Dale	541-10.2KCCT-ND
1	1.96k	R0805	R8	RES SMD 1.96K OHM 1% 1/8W 0805		CRCW08051K96FKEA	Vishay Dale	541-1.96KCCT-ND
1	.01uF	C0603K	C84	CAP .01UF 0603		C0603C103J5REC7411	KEMET	399-17568-1-ND
2	.1uF	C0603K	C78, C81	CAP .1UF 0603		C0603C104M4RACTU	KEMET	399-1099-1-ND
35	0.1uF	C0805	C22, C23, C24, C25, C26, C27, C28, C29, C30, C31, C32, C33, C34, C35, C36, C37, C38, C39, C40, C41, C42, C43, C44, C45, C46, C47, C48, C49, C50, C51, C52, C55, C56, C71, C72, C74, C85, C86, C87, C88, C89, C90, C91, C92	CAP CER 0.1UF 50V X7R 0805		CL21B104MBCNNNC	Samsung Electro-Mechanics	1276-2450-1-ND
1	1.2nF	C0805	C14	CAP CER 1200PF 50V C0G/NP0 0805		CL21C122JBFNNNE	Samsung Electro-Mechanics	1276-2986-1-ND
2	1.8k	R0805	R80, R82	RES 1.8K OHM 0.1% 1/8W 0805		ERA-6AEB182V	Panasonic Electronic Components	P1.8KDACT-ND
4	10	R0603	R89, R90, R92, R93	RESISTOR, American symbol		RC0603FR-0710RL	Yageo	311-10.0HRCT-ND
6	100	R0603	R94, R97, R98, R100, R101, R102	RESISTOR, American symbol		RC0603FR-07100RL	Yageo	311-100HRCT-ND
1	1M	R0805	R12	RES SMD 1M OHM 1% 1/8W		CRCW08051M00FKEAC	Vishay Dale	541-4135-1-ND

				0805				
1	100k	R0805	R11	RES SMD 100K OHM 1% 1/8W 0805		CRCW0805100KFKEAC	Vishay Dale	541-3978-1-ND
20	100k	R0805	R48, R51, R52, R54, R56, R57, R58, R60, R61, R62, R63, R65, R66, R67, R68, R70, R71, R72, R73, R75	RES SMD 100K OHM 5% 1/8W 0805		CR0805-JW-104ELF	Bourns Inc.	CR0805-JW-104ELFCT-ND
5	100nF	C0805	C1, C3, C5, C7, C9	CAPACITOR, American symbol				
2	10k	R0805	R3, R6	RES SMD 10K OHM 1% 1/8W 0805		CRCW080510K0FKEA	Vishay Dale	541-10.0KCCT-ND
18	10k	R0805	R19, R20, R24, R27, R28, R29, R30, R31, R34, R35, R36, R38, R39, R40, R42, R45, R47, R78	RES SMD 10K OHM 5% 1/8W 0805		CR0805-JW-103ELF	Bourns Inc	CR0805-JW-103ELFCT-ND
18	10k	R0805	R14, R15, R16, R17, R18, R21, R22, R23, R25, R26, R32, R33, R37, R41, R43, R44, R46, R83	RES SMD 10K OHM 5% 1/8W 0805	DNU	CR0805-JW-103ELF	Bourns Inc.	CR0805-JW-103ELFCT-ND
16	10nF	0805_H0.75	C53, C54, C57, C58, C59, C60, C61, C62, C63, C64, C65, C66, C67, C68, C69, C70, C73	CAP CER 10000PF 50V X7R 0805		CL21B103KBANNNC	Samsung Electro-Mechanics	1276-1015-1-ND
2	10nH	WE-MK_0603	L5, L6	FIXED IND 10NH 500MA 400 MOHM		FIXED IND 10NH 500MA 400 MOHM	Würth Elektronik	732-6263-2-ND
1	10pF	C0805	C13	CAP CER 10PF 50V C0G/NP0 0805		CL21C100JBANNNC	Samsung Electro-Mechanics	1276-1109-1-ND
2	10uF	C0805	C2, C6	CAP CER 10UF 35V X5R 0805		C2012X5R1V106K085AC	TDK Corporation	445-14417-1-ND
1	10uF	C1210	C11	CAP CER 10UF 50V X7R 1210		C3225X7R1H106M250AC	TDK Corporation	445-14933-1-ND
1	12.4k	R0805	R2	RES SMD 12.4K OHM 1% 1/8W 0805		CRCW080512K4FKEA	Vishay Dale	541-12.4KCCT-ND
1	15uH	IFSC-1515A_IND_IFSC_151	L4	FIXED IND 15UH 1.25A 222 MOHM		IFSC1515AHER150M01	Vishay Dale	541-4328-1-ND

		5AH						
1	160k	R0805	R64	RES SMD 160K OHM 5% 1/8W 0805		RC0805JR-07160KL	Yageo	311-160KARCT-ND
3	18k	R0805	R79, R81, R84	RES 18K OHM 0.1% 1/8W 0805		ERA-6AEB183V	Panasonic Electronic Components	P18KDACT-ND
3	1k	R0603	R87, R91, R95	RES SMD 1K OHM 1% 1/10W 0603		RC0603FR-071KL	Yageo	311-1.00KHRCT-ND
1	1u	C0805	C75	CAP CER 1UF 16V Y5V 0805		C0805C105Z4VACTU	KEMET	399-8011-1-ND
1	20k	R0805	R9	RES SMD 20K OHM 1% 1/8W 0805		CRCW080520K0FKEAC	Vishay Dale	541-4133-1-ND
2	2.2k	R0603	R86, R99	RES SMD 2.2K OHM 1% 1/10W 0603		RT0603FRD072K2L	Yageo	13-RT0603FRD072K2LTR-ND
1	2.2uF	C0805	C20	CAP CER 2.2UF 16V X5R 0805		EMK212BJ225KG-T	Taiyo Yuden	587-1293-1-ND
1	2.2uH	IND_NLCV32TR	L1	FIXED IND 2.2UH 1.2A 115 MOHM		NLCV32T-2R2M-PFR	TDK Corporation	445-3650-1-ND
1	220p	C0603K	C80	CAP CER 220PF 25V X7R 0603		C0603C221K3RACAUTO	KEMET	399-C0603C221K3RACAUTOTR-ND
2	22p	C0603K	C76, C83	CAP CER 22PF 25V NP0 0603		C0603C220J3GAC7867	KEMET	399-15376-2-ND
2	22uF	C0805	C4, C8	CAP CER 22UF 10V X7S 0805		C2012X7S1A226M125AC	TDK Corporation	445-14560-1-ND
1	22uF	C1210	C15	CAP CER 22UF 16V X5R 1210		GRM32ER61C226ME20L	Murata Electronics	490-1881-1-ND
1	255k	R0805	R13	RES SMD 255K OHM 1% 1/8W 0805		ERJ-6ENF2553V	Panasonic Electronic Components	P255KCCT-ND
1	2k	0805	R85	RES 2K OHM 0.1% 1/8W 0805		ERA-6AEB202V	Panasonic Electronic Components	P2.0KDACT-ND
2	3.3u	C0805	C77, C82	CAPACITOR, American symbol		C0805C335K8PACTU	KEMET	399-3129-1-ND
2	30	R0805	R1, R4	RES SMD 30 OHM 0.5% 1/10W 0805		RR1220Q-300-D	Susumu	RR12Q30DCT-ND

1	30.9k	R0805	R5	RES SMD 30.9K OHM 1% 1/8W 0805		CRCW080530K9FKEA	Vishay Dale	541-30.9KCCT-ND
1	33k	R0805	R53	RES SMD 33K OHM 5% 1/8W 0805		CR0805-JW-333ELF	Bourns Inc.	CR0805-JW-333ELFCT-ND
1	33pF	C0805	C17	CAP CER 33PF 50V C0G/NP0 0805		C0805C330J5GACTU	KEMET	399-1115-1-ND
2	39k	0805	R49, R50	RES SMD 39K OHM 5% 1/8W 0805		RC0805JR-0739KL	Yageo	311-39KARCT-ND
2	4.7k	0805	R76, R77	RES SMD 4.7K OHM 5% 1/8W 0805		CR0805-JW-472ELF	Bourns Inc.	CR0805-JW-472ELFCT-ND
2	4.7uF	C0805	C16, C21	CAP CER 4.7UF 25V X5R 0805		TMK212BJ475KG-T	Taiyo Yuden	587-1782-1-ND
1	4.7uH	IND_5022	L2	FIXED IND 4.7UH 2.3A 81 MOHM SMD		LTF5022T-4R7N2R0-LC	TDK Corporation	445-6196-1-ND
1	4.7uH	IHLP-3232DZ	L3	FIXED IND 4.7UH 8A 22.6 MOHM SMD		IHLP3232DZER4R7M11	Vishay Dale	541-1353-1-ND
1	470	R0805	R10	RES SMD 470 OHM 5% 1/8W 0805		RC0805JR-07470RL	Yageo	311-470ARCT-ND
1	47uF	C1210	C10	CAP CER 47UF 16V X5R 1210		GRM32ER61C476KE15L	Murata Electronics	490-6539-1-ND
1	47uF	C2220	C18	CAP CER 47UF 16V X7R 2220		CGA9N3X7R1C476M230KB	TDK Corporation	445-7935-1-ND
2	500	R0603	R88, R96	RES 500 OHM 0.1% 1/10W 0603		RT0603BRC07500RL	Yageo	13-RT0603BRC07500RLTR-ND
2	51k	R0805	R69, R74	RES SMD 51K OHM 5% 1/8W 0805		CR0805-JW-513ELF	Bourns Inc.	CR0805-JW-513ELFCT-ND
1	6.8n	C0603K	C79	CAP CER 0603 6.8NF 16V X7R 10%		C0603C682K4REC	KEMET	C0603C682K4RECT-ND
1	8.2nF	C1206	C12	CAP CER 8200PF 25V C0G/NP0 1206		12063A822JAT2A	AVX Corporation	478-1503-1-ND
1	9.1k	R0805	R55	RES SMD 9.1K OHM 5% 1/8W 0805		RC0805JR-079K1L	Yageo	311-9.1KARCT-ND
1	91k	R0805	R59	RES SMD 91K OHM 5% 1/8W 0805		RC0805JR-0791KL	Yageo	311-91KARCT-ND

3	CD74HCT4051	SOIC127P600X144-16	U7, U8, U9	IC MUX/DEMUX 8X1 16SOIC		CD74HCT4051M	Texas Instruments	296-9276-5-ND
1	CD74HCT4053	SOIC127P600X144-16	U10	IC MUX/DEMUX TRIPLE 2X1 16SOIC		CD74HCT4053M96	Texas Instruments	296-26030-1-ND
1	DAC80501Z	DGS10_TEX-M	U25	DAC 12 BIT 3MSPS		DAC80501ZDGSR	Texas Instruments	296-DAC80501ZDGSRCT-ND
1	GREEN	LED-1206	D1	LED GREEN CLEAR CHIP SMD		LTST-C230KGKT	Lite-On Inc.	160-1456-1-ND
1	LM43601PWPR	PWP16	U5	IC REG BUCK ADJ 1A 16HTSSOP		LM43601PWPR	Texas Instruments	296-40221-1-ND
1	OPA2197	SO08	U19	IC OPAMP GP 2 CIRCUIT 8SOIC		OPA2197IDR	Texas Instruments	296-44769-1-ND
5	OPA4197	SOIC127P600X144-14	U6, U15, U16, U17, U18	IC OPAMP GP 4 CIRCUIT 14SOIC		OPA4197IDR	Texas Instruments	296-45059-1-ND
1	REF1933AIDDCTDDC5_TEX-M	DDC5_TEX-M	U20	IC REFERENCE 3V3 FIXED 1V8 BIAS		REF1933AIDDCT	Texas Instruments	296-38444-1-ND
2	SBR80520LT3G	SOD123	D2, D3	DIODE SCHOTTKY 20V 500MA SOD123		SBR80520LT3G	ON Semiconductor	SBR80520LT3GOSCT-ND
1	SJ1-3523N	CUI_SJ1-3523N	J10	CONN JACK STEREO 3.5MM R/A		SJ1-3523N	CUI Devices	CP1-3523N-ND
3	SN74LVC1T45DBVRG4	DBV6	U23, U24, U26	IC TRNSLTR BIDIRECTIONAL SOT23-6		SN74LVC1T45DBVRG4	Texas Instruments	296-52733-2-ND
1	TPS54531	DDA8_3P1X2P4-L	U3	IC REG BUCK ADJUSTABLE 5A 8SOPWR		TPS54531DDAR	Texas Instruments	296-40801-1-ND
2	TPS56339	TSOT23-6_DDC	U1, U2	PWR MGMT SPECIALIZED REGULATOR		TPS56339DDCR	Texas Instruments	296-53568-1-ND
1	VS-50WQ04FNPBFDPAK-L	DPAK-L	U4	DIODE SCHOTTKY 40V 5.5A TO252AA		VS50WQ04FNPBF	Vishay	VS-50WQ04FNHM3GI-ND
1	Value	RUG0008A	U22	ADC 12BIT 3MHZ SPI		ADS7047IRUGR	Texas Instruments	296-48819-1-ND

1	Value	RUN0010A	U21	Differential Op Amp 150MHZ		THS4551IRUNR	Texas Instruments	296-48683-1-ND
4	X9258	SOIC127P1030X250	U11, U12, U13, U14	IC DGT POT 100KOHM 256TAP 24SOIC		X9258TS24IZ-2.7	Renesas Electronics America Inc.	X9258TS24IZ-2.7-ND
1	A_XADC	FEMALE_HEADER-2X6	J15	RECEPTACLE STRIP 12P 2.54MM PITC	DNU	RS2-12-G	Adam Tech	2057-RS2-12-G-ND
1	BRKOUT_JB	FEMALE_HEADER-2X6	J9	RECEPTACLE STRIP 12P 2.54MM PITC	DNU	RS2-12-G	Adam Tech	2057-RS2-12-G-ND
1	BRKOUT_JC	FEMALE_HEADER-2X6	J4	RECEPTACLE STRIP 12P 2.54MM PITC	DNU	RS2-12-G	Adam Tech	2057-RS2-12-G-ND
2	BRKOUT_JD	FEMALE_HEADER-2X6	J12, J16	RECEPTACLE STRIP 12P 2.54MM PITC	DNU	RS2-12-G	Adam Tech	2057-RS2-12-G-ND
1	BRKOUT_JF	FEMALE_HEADER-2X6	J7	RECEPTACLE STRIP 12P 2.54MM PITC	DNU	RS2-12-G	Adam Tech	2057-RS2-12-G-ND
1	ZYBO_JB	FEMALE_HEADER-2X6	J8	RECEPTACLE STRIP 12P 2.54MM PITC	DNU	RS2-12-G	Adam Tech	2057-RS2-12-G-ND
1	ZYBO_JC	FEMALE_HEADER-2X6	J3	RECEPTACLE STRIP 12P 2.54MM PITC	DNU	RS2-12-G	Adam Tech	2057-RS2-12-G-ND
1	ZYBO_JD	FEMALE_HEADER-2X6	J11	RECEPTACLE STRIP 12P 2.54MM PITC	DNU	RS2-12-G	Adam Tech	2057-RS2-12-G-ND
1	ZYBO_JE	FEMALE_HEADER-2X6	J5	RECEPTACLE STRIP 12P 2.54MM PITC	DNU	RS2-12-G	Adam Tech	2057-RS2-12-G-ND
1	ZYBO_JF	FEMALE_HEADER-2X6	J6	RECEPTACLE STRIP 12P 2.54MM PITC	DNU	RS2-12-G	Adam Tech	2057-RS2-12-G-ND
1	ZYBO_Power	1X02_LOCK	J1	Generic Header-pin footprint for 0.1 inch	DNU	50579002	Molex	WM2800-ND
1	INPUT_PWR	1X02_LOCK	J2	Generic Header-pin footprint for 0.1 inch	DNU	705530001	Molex	WM4900-ND
4	AUX PWR	2X02	JP1, JP2, JP3, JP4	PIN HEADER	DNU	PH2-04-UA	Adam Tech	2057-PH2-04-UA-ND
2	Vout_HEADER	1X02_LOCK	J18, J20	Generic Header-pin footprint for 0.1	DNU	PH1-02-UA	Adam Tech	2057-PH1-02-UA-ND

				inch				
1	Vin_HEADER	1X02_LOCK	J19	Generic Header-pin footprint for 0.1 inch	DNU	PH1-02-UA	Adam Tech	2057-PH1-02-UA-ND
1	3V3_SENSOR	1X03_LOCK	J14	Generic Header-pin footprint for 0.1 inch	DNU	705430037	Molex	WM4825-ND
1	1V8_SENSOR	1X03_LOCK	J17	Generic Header-pin footprint for 0.1 inch	DNU	705430037	Molex	WM4825-ND
1	5V0_SENSOR	1X03_LOCK	J13	Generic Header-pin footprint for 0.1 inch	DNU	705430037	Molex	WM4825-ND
3	BU-SMA-V	BU-SMA-V	X1, X2, X3	TOP LAUNCH SMA CONNECTOR	DNU			