AERO ENGINE CONTROLS: TORQUE MOTOR FLUID DELIVERY SYSTEM

By

Ross Boe

David Bostwick

Zhimin Zou

Final Report for ECE 445, Senior Design, Spring 2012

TA: Alex Suchko

02 May 2012

Project No.21

Abstract

The following is a final report for ECE 445 at the University of Illinois at Urbana-Champaign. This report covers the design and testing procedures of a positional drive control system implemented with a torque motor as per the request of Aero Engine Controls, the company overseer for the project. The project in question is a small scale representation of a torque motor subsystem contained within an industrial fluid delivery system implemented in aerospace applications.

Terms and Keywords

- AEC: Aero Engine Controls
- CAN: Controller Area Network
- DMM: Digital Multi-Meter
- EMF: Electromotive Force
- FDS: Fluid Delivery System
- LCD: Liquid Crystal Display
- MISO: Master-In Slave-Out
- MOSI: Master-Out Slave-In
- PIC: Peripheral Interface Controller
- POT: Potentiometer
- PWM: Pulse Width Modulation (or Pulse Width Modulated)
- SPI: Serial Peripheral Interface

Contents

1. Introduction 1
   1.1 Purpose of Project 1
   1.2 Project Functions and Features 1
   1.3 Block Diagram 2
   1.4 Block Diagram Component Descriptions 2
      1.4.1 Power Supply 2
      1.4.2 PC “EEC Interface” 2
      1.4.3 Torque Module PIC (Node) 2
      1.4.4 Drive Circuit 2
      1.4.5 Torque Motor Model 3
      1.4.6 Position Feedback Sensor 3
      1.4.7 Arduino Motor Recognizer 3
2. Design 4
   2.1 General Design Alternatives 4
   2.2 Equations, Simulations, and General Circuits 5
      2.2.1 Fluid Resistance Simulation on Torque Motor 5
      2.2.2 Current Control Simulation 6
      2.2.3 Simulation Calculations 6
      2.2.4 PID Simulation 7
   2.3 Description of Design 7
      2.3.1 Microcontroller 7
      2.3.2 PIC ADC Software 9
      2.3.3 PIC PWM Software 10
      2.3.4 LCD Software 10
      2.3.5 SPI Software 10
      2.3.6 Drive Circuit 11
      2.3.7 System Control 12
      2.3.8 System Control Dynamics 14
      2.3.9 Bulk Voltage Correction Factor 14

2.3.10 Modularity of Design 14
3. Verification of Design 17
3.1 Testing Procedures 17
3.1.1 Power Supply Test 17
3.1.2 Drive Circuitry Tests 17
3.1.3 PIC Board Testing 17
3.1.4 Arduino Uno Test 18
3.1.5 System Performance Test 18
3.2 Measurements and Results 18
3.3 Discussion of Results 18
3.3.1 Power Supply Test 18
3.3.2 Drive Circuitry Tests 18
3.3.3 PIC Board Testing 18
3.3.4 Arduino Uno Test 19
3.3.5 System Performance Tests 19
4. Costs 20
4.1 Parts 20
4.2 Labor 21
4.3 Total Cost 21
5. Conclusion 22
5.1 Accomplishments 22
5.2 Uncertainties 22
5.3 Ethical considerations 22
5.4 Future work 23
References 24
Appendix A Requirement and Verification Table 25
Appendix B Schematics and Pin-Outs 28
Appendix C PID Transient Response 32
Appendix D Drive Circuit and Positioning Response 38

1. Introduction

1.1 Purpose of Project

The objective of this project was to design a positioning control system for a torque motor. It is intended for use on a larger scale as part of an aerospace applied fluid delivery system for Aero Engine Controls (AEC). Additionally, the torque motor drive system is meant to be modular; the drive system should be interchangeable with a stepper motor drive system developed in previous semesters for AEC. The system receives positioning commands from a POT knob, which is analogous to a FDS controller (a fluid percent flow rate controller) and outputs a variable current though use of drive circuitry to position the motor as desired based on the initial input. A lever arm attached to a spring is mounted onto the motor shaft in an effort to simulate fluid resistance, while maintaining an anhydrous system. The torque motor used in the project is equipped with a position sensor, which provides the positioning information of the motor. This information is fed back to the PIC controller. The PIC takes this position sensor information and adjusts the motor controls (by varying the driving current) to correct any positioning error.

1.2 Project Functions and Features

- Insure the reliability of the fluid delivery system
- Provide an accurate estimate of the amount of fluid being delivered
- Improved control over the amount of fluid delivered
- Provide motor status to a higher level controller
- Dynamic positioning response for motor through use of the position sensor
- Programmable PIC controller for position and torque modulation
- Functional with a lower or higher operating voltage
- Defines a node for a distributed system for the FDS

1.3 Block Diagram

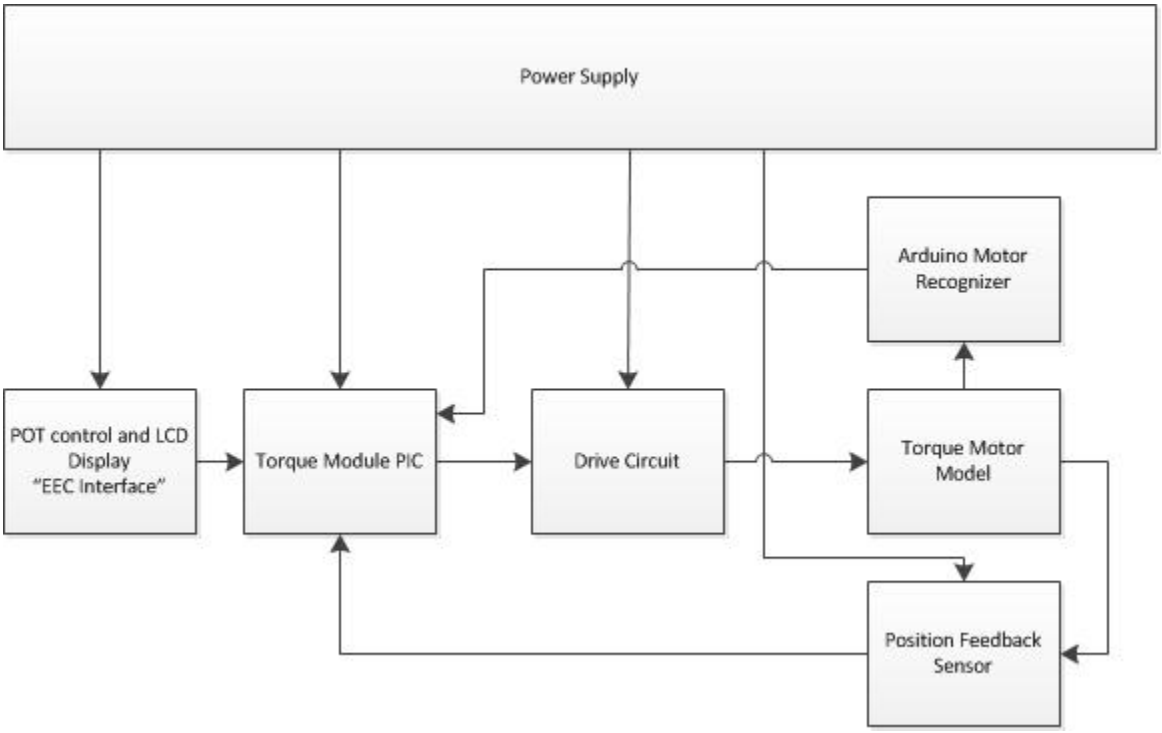

Figure 1.3.1 Block Diagram of Design

1.4 Block Diagram Component Descriptions

1.4.1 Power Supply

The power supply will be the primary voltage source for all components and simulates the 28V supply rails used in the industry counterpart of this project. The entire design runs off of this rail voltage. The 28V rail is delivered to buck converters which provide 3.3V and 5V outputs for the PIC and drive circuit, respectively.

1.4.2 PC "EEC Interface"

This is a user-input controller on the vehicle for which the FDS is implemented. It constantly sends and receives requests to and from the torque module PIC. This component consists of four LEDs and a POT in order to interface with the user.

1.4.3 Torque Module PIC (Node)

This is a programmable PIC controller which interprets actuation signals from the high level controller and sends low level actuation signals with which to drive the torque motor model.

1.4.4 Drive Circuit

This is the actuation hardware required to interpret control signals from the PIC board into usable analog signals in order to manipulate the motor position. The drive circuitry takes in PWM data signals from the PIC and interprets them to produce a current, which in turn drives the motor.

1.4.5 Torque Motor Model

This is a current driven brushed DC torque motor which will act against a compressing spring force to simulate fluid resistance within the delivery system. The motor will operate at 14.5V maximum and 6A maximum (87 Watts). It is the mechanical system for which positioning control is implemented.

1.4.6 Position Feedback Sensor

This is a position sensor which will feed positional and torque information back into the PIC controller in DC voltage for dynamic positioning response.

1.4.7 Arduino Motor Recognizer

This is an Arduino which communicates with the PIC using a SPI. It helps facilitate the modularity and compatibility of the design with other motor drive systems (e.g. a stepper motor drive system).

2. Design

2.1 General Design Alternatives

An alternative design in the drive circuit was considered in the beginning of the design process. Figure 2.1.1 is the alternative bidirectional drive circuit design. This design allows the motor to move two directions (e.g. clockwise and counter-clockwise rotation). However, the additional direction of rotation introduces a new problem in the drive circuit. For example, if MOSFET Q3 falling edge delay is longer than the rising edge delay of Q1, a surge of current can go through Q3 and Q1. A bidirectional H bridge design also introduces additional problems in controlling the motor. Driving the circuit in two direction means there must be two PWM signals coming from the PIC; one for forward direction drive and the other for reverse direction drive. When driving the motor backwards, the system is extremely unstable due to the force of the spring occurring in the same direction. The inertia of the lever arm then becomes the only reason for shaft to stay in the original position. During forward directional drive, this issue is non-existent because the spring torque and motor torque are in opposing directions. This reverse direction drive problem makes positional control much more difficult. As a result, the group decided to go with a single side drive circuit due its simplicity, reliability, and reduction in hardware.

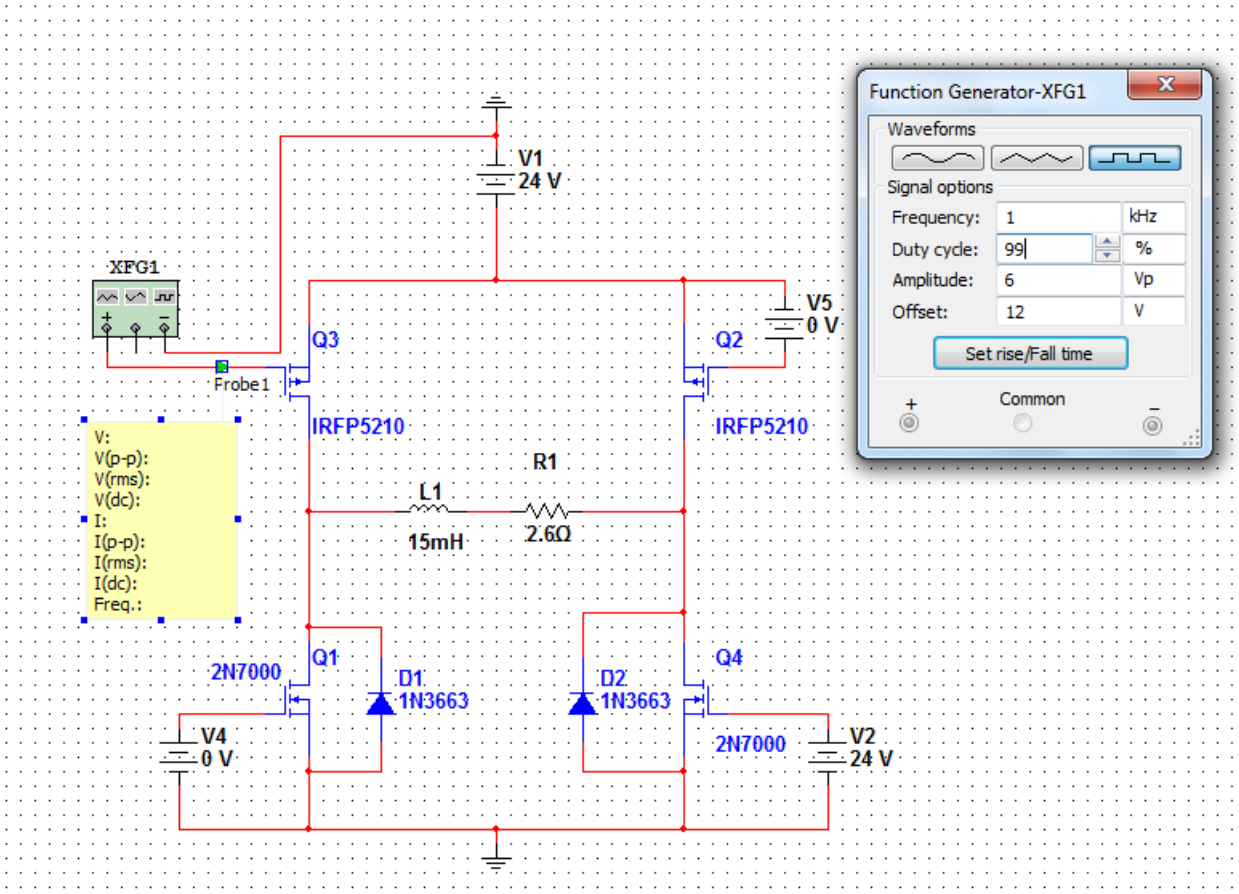

Figure 2.1.1 Dual Directional Drive Circuit

2.2 Equations, Simulations, and General Circuits

2.2.1 Fluid Resistance Simulation on Torque Motor

As aforementioned, fluid resistance on the valve in the FDS is represented by a lever arm and spring so as to provide proof of concept while maintaining a dry model for lab work. The motor has a mounted lever attached to its rotor shaft. This pulls a spring mounted near the end of the lever arm, which acts as a torque load analogue to the fluid resistance. The torque delivered can be calculated given the spring constant, angle of the lever, and length of the lever. See Figure 2.2.1 for a visual aid.

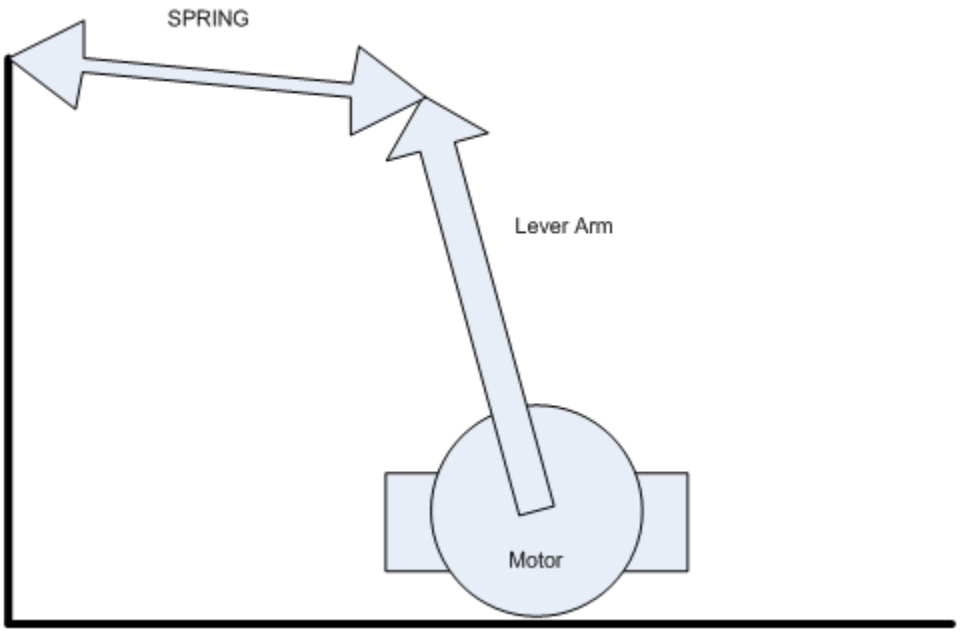

Figure 2.2.1 Visual Representation of the Torque Motor Model

2.2.2 Current Control Simulation

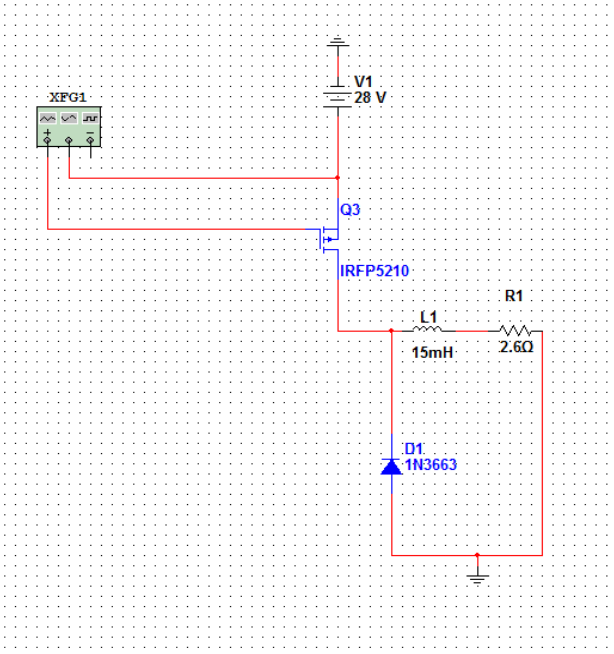

Figure 2.2.2 Current Control Simulation Diagram

The circuit above is a basic buck converter with its inductor replaced by a model of a DC motor without the back EMF. The motor is modeled with an inductor and a resistor; the values are taken from the data sheet for the motor. Since ω is approximately zero in this case, one can conclude the voltage drop on the motor is approximately zero based on equation (1) below. As one can see by adding the fly-back diodes, the circuit is effectively a buck converter. The simulation is done without protection elements to show the concept of controlling current by using a buck converter. In this simulation, MOSFET Q3 is pulsed by a function generator with different PWM duty cycles.

$$v_b = k_v \omega \tag{1}$$

The first set of tests set the driver, a square wave function generator to 50% duty in this case. Next, the frequency of the wave is varied from 1 kHz to 10 kHz. Then the ripple of the current passing through the motor, the resistor and inductor are compared.

As one can see from Figure 2.2.9 in Appendix C, there is little to no ripple looking at a 10ms time frame in the 10 kHz case. In contrast, there are clearly ripples in the 1 kHz current case. The current in the 1 kHz wave fluctuate from 0.5 A to 1 A. Higher frequency is clearly the winner when it comes to current control.

2.2.3 Simulation Calculations

From equations (2) and (3), one can see the duty ratio in a buck converter is directly proportional to its output current over input current. This relationship was tested by running at 5 kHz and varying the PWM duty ratio from 1% to 99% with a step size of 25%.

$$\frac{Iout}{Iin} = D \tag{2}$$

$$\frac{Vs*D}{R} = Iout \tag{3}$$

In the circuit tested, the function generator is floating on the high side source and is driving the PMOS. The function generator gave an error when the polarity is flipped with the source and gate. Since the PMOS simulated has a Vgs threshold of about -4V, the duty ratio used on the function generator has to be subtracted out of one to get an equivalent duty ratio.

From Figures 2.2.3 – 2.2.7 in Appendix C, one can see that the output current is linearly related to the duty ratio. These two simulations show that one can accurately control current if the PWM is running at a high frequency. Since the duty ratio is linearly related to the output current, to have high resolution in output current the duty ratio resolution must be high.

At 99% $I_{99\%}$=5.4778A, one can predict at 50%, $I_{50\%}$=2.7389A.

The simulated result current is 2.7217A

$$\%error = |I\ simulated - I\ predicted| / I\ simulated = 0.63196\% \tag{4}$$

This is very accurate considering the circuit is not running at its potential frequency.

2.2.4 PID Simulation

A discrete PID loop simulation is set up to analyze the effect of the integral term the PID loop. Figures 2.2.12 – 2.2.14 in Appendix C are plotted from an excel simulation. The derivative gain is zero.

As one can see from Figure 2.2.12 , at t = 0.015 the position overshoots the commanded position due to the integral term. At t =0.05 the integral term stablilizes the position.

Furthermore, as seen in Figure 4.3.2 and Figure 2.2.14, the higher the integral term coefficient factor, the faster the position get closer to the desired 50 degrees.

2.3 Description of Design

2.3.1 Microcontroller

The general sequence of operation of the microcontroller is shown in Figure 2.3.1 below.

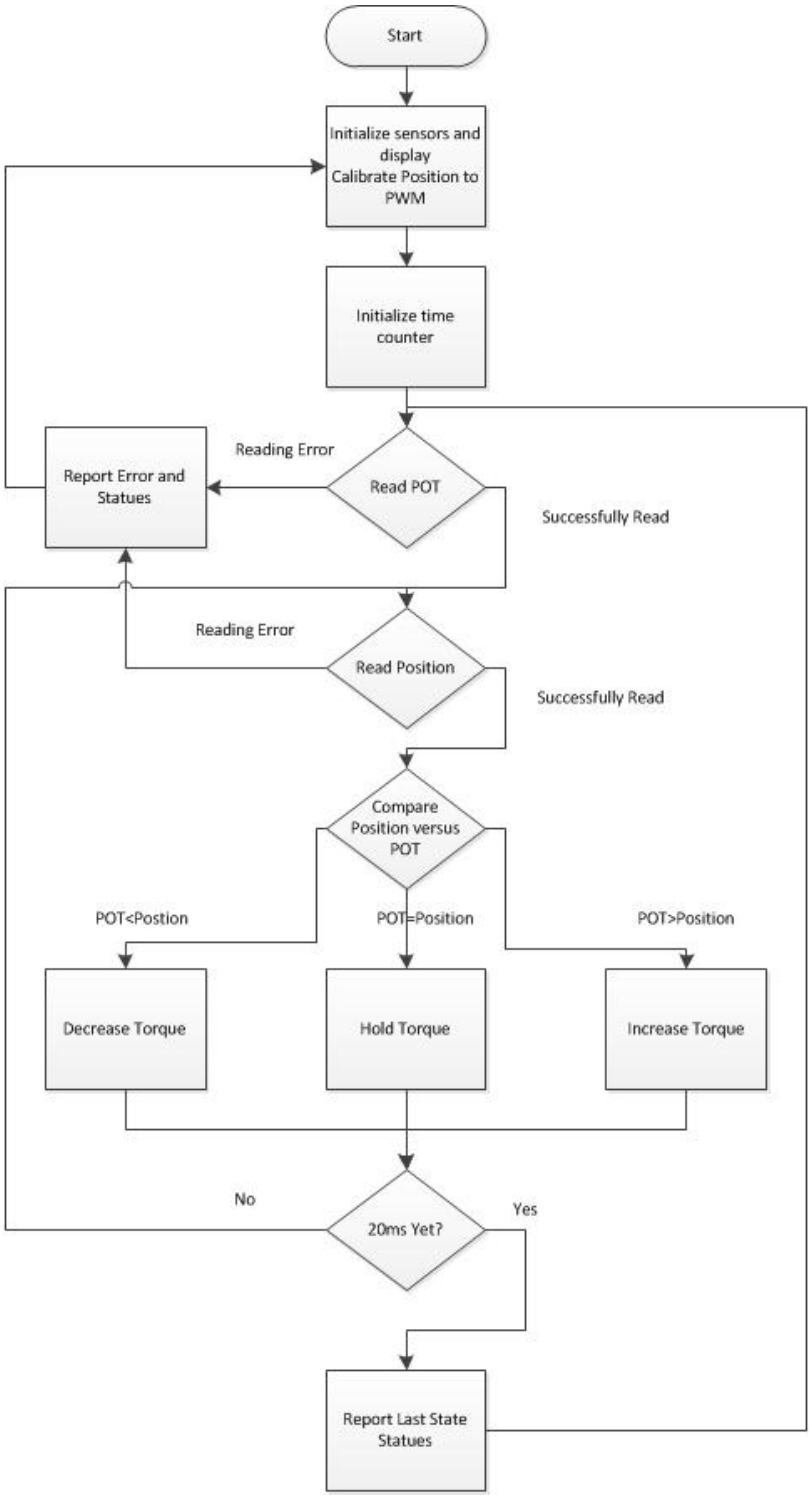

Figure 2.3.1 Microcontroller Flow Diagram

In a realistic situation, the fluid delivery system will interface the EEC controller via CAN bus. The EEC will send out a request for the FDS status and current fluid flow percentage out of the maximum fluid flow percentage. After the request is sent out from the EEC, 20ms is given to the FDS to report back to the

EEC. The FDS should send out two status messages. The first will contain the state of the FDS, these statuses are: OFF, HOLDING, TRANSITION, MAX, and DRIVE SAFE. State OFF means the FDS is not responding. State HOLDING means the valve or lever has achieved its requested position and is holding the position. State TRANSITION means the valve or lever is still in transition to achieve the requested position. MAX means the valve or lever is at its peak position or 100% fluid flow rate. Finally, DRIVE SAFE state means the FDS lost communication with the EEC.

In this project, the EEC is represented by a POT. The POT value is sampled as a percentage of the max voltage that can be measured by the analog input. The FDS then processes the requested value and the motor position sensor value and adjusts the motor accordingly into the correct position. Since the FDS does not have an EEC to interface with, it reports its state by LEDs. In this project, the off state and drive safe are included. Since these states are based on the CAN communication status, it is unlikely for the POT to malfunction. The FDS being put into a safe drive state can be simulated by a push button. When a button is pressed, the FDS responds as if it lost communication with the EEC and changes its current state to safe drive. The FDS constantly updates its states and position, but does not report the state unless the 20ms interval is up. This ensures that the EEC won't be flooded by FDS status report messages.

In the TRANSITION state, the motor can only produce variable torque in one direction. This is because the lever is pulled backwards by the spring, which delivers a constant force in the opposite direction. The torque motor can change its position by balancing the force the motor produces and the constant force of the spring on the lever.

AEC specified that the torque motor must be controlled by the PIC24HJ256GP610A from microchip. This is because of the PIC's multiport capability, such as the ECAN port and SPI ports. The Explorer 16 Development Board DM240001 was suggested as well. The final PCB design of the microcontroller is closely based on the design of the Explorer 16 Development board. See Figure 2.3.2 in Appendix B for PIC pin out. Additionally, this part is also up to industry standards for both safety and reliability.

2.3.2 PIC ADC Software

The PIC samples multiple analog peripherals through its ADC ports. These peripherals are the POT, bulk voltage, and position sensor. A 10kΩ potentiometer is connected to pin AN5 on the PIC as seen in Figure 2.3.3 in Appendix B. A 470 ohm resistor is placed in between the potentiometer and AN5 to ensure no surge of current will flow into the pin. The bulk voltage is sampled through a voltage divider into pin AN0. Lastly, the position sensor voltage is sampled into RB2.

Manual sampling is used due to an unknown error with automatic sampling mode causing the sample in buffer to not update in desired time. The sampling clock is chosen to be 2Tcy, which is two times the system clock period. 1 ms is given for the ADC unit to convert the sampled information.

The potentiometer and bulk voltage are sampled from ADC unit one. Sampling is done by switching channels between AN5 and AN0. The position sampling is done similarly to the POT sampling, since it has DC potential output. Since the position sample information is crucial to the control system, it is implemented into the second ADC unit on the PIC.

2.3.3 PIC PWM Software

The PWM is crucial to the drive circuit since the duty ratio controls the output current; see equation (5). The PWM frequency is also extremely important. As the frequency increases, ripple on the output current decreases. Too high of frequency will result in high power loss in switching on and off the FETs. In this case, a PWM frequency of just 12.5 kHz is chosen, since the simulation gave very good results in current control under these conditions.

$$D = \frac{I_{out}}{I_{in}} \tag{5}$$

$$\text{PWM Period} = [(PRy) + 1] \bullet TCY \bullet (\text{TMRy Prescale Value}) \tag{6}$$

$$\text{PWM Resolution (bits)} = \log_2 (\text{Timer Clock Frequency/ PWM Frequency}) \text{ bits} \tag{7}$$

Equation (6) and (7) are obtained from [2].

The PWM output on the PIC is done by using the comparator on the PIC. The clock is compared with an input register on the PIC. Timer 2 is used as the clock. The clock pre-scale value is set to 1:1. TCY is the oscillator period times two, PRy is 500, and TMRy is set to 1 to 1. This means the PWM value is 8000Hz; see equation (9). This frequency is perfect for our application since it is right above 10 kHz. PWM resolution can be calculated using equation (10).

$$\text{PWM Period} = [(500) + 1] \bullet (2.5e\text{-}7) \bullet (1) = 1.25e\text{-}4 \text{ s} \tag{8}$$

$$\text{PWM Frequency} = 1/(\text{ PWM Period}) = 8000\text{Hz} \tag{9}$$

$$\text{PWM Resolution (bits)} = \log_2 (8000{,}000/8000) \text{ bits} = 9 \text{ bits} \tag{10}$$

$$\text{Duty Cycle Steps} = 2^9 = 512\ Steps \tag{11}$$

The duty ratio is controlled by the ratio shadow register of comparator RS1 to the PWM period which is defined by PRy. This is updated through the ADC interrupt. Each time the POT ADC conversion is done the duty ratio is updated.

$$Duty\ Ratio = \frac{OC1RS}{PRy} \tag{12}$$

Since the torque motor has a single side drive circuit, one PWM is sufficient to control the motor.

2.3.4 LCD Software

The LCD TSB1G7000-E driver is given in the AEC repository written by previous semesters Stepper Motor Group. The LCD has its own read and write register, which is connected to PORTE on the PIC. Pin RD5 is connected to the read and write control register of the LCD. Pin RB15 is connected to the register select (RS) of the LCD. Pin D4 is connected to the enable (E) register of the LCD.

2.3.5 SPI Software

The SPI software in this project is a modified SPI driver from the AEC repository. Originally, the driver was used for inputting information to a controller for the stepper motor. This project's version of the SPI

driver retained all functions of the previous driver for the modular design. An extra read function is added specifically to get the motor information from an Arduino, which is a motor tag reader. The PIC operates in master mode and the Arduino operates in slave mode. SPI2 ports from the PIC are used for this operation.

2.3.6 Drive Circuit

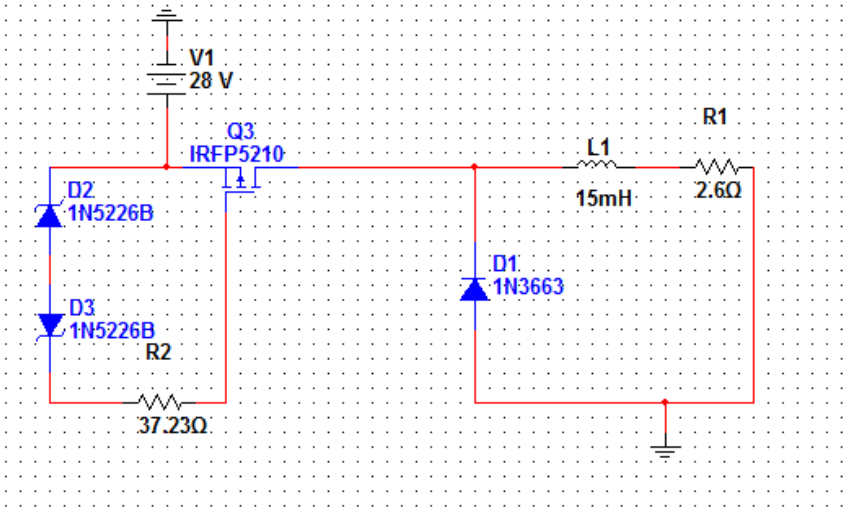

Figure 2.3.4 Drive Circuit

The driver circuit is made of two components, the H bridge driver chip and the single-side drive circuit. The L298N H bridge driver chip is used in the drive circuitry. The single side drive circuit is made up of a PMOS on the high side of the motor and a fly back diode. The PMOS is used in order to secure a constant source voltage from the battery. No boot strapping is required. Since the torque motor can only apply force in one direction, the PIC outputs a single PWM wave. The PMOS will turn on with Vsg greater than 3V. Since the source of the PMOS is connected to a constant voltage source of 28V, a gate voltage of 0V would open the source to drain of the PMOS. A gate voltage of 28V would close the source to drain of the PMOS. This means the PWM signal duty ratio will be inversely proportional to the magnitude of current flowing through the PMOS. Inverters are added in between the PIC output PWM and the bridge driver chip to correct this issue. The inverters are connected to 5V for operation.

Since the motor is inductive, the PWM induced square wave voltage will be smoothed out into a constant DC current based on the PWM duty ratio. This is done by converting the single sided drive circuit into a buck converter by adding a diode from ground to the input of the motor (see Figure 2.3.5 in Appendix B). The motor can be modeled as an inductor in series with the resistor. One can clearly see from Figure 2.3.5 that the drive circuit is essentially a buck converter. Two zener diode (MAZ41500MFTB-ND) must be placed between the source and gate to prevent the gate voltage from going outside -10V to 10V. Zener diodes with a 10V breakdown voltage were selected to fit the circuit. Lastly, in order to stop current from going back into the driver chip, resistors were placed on the gate of the MOSFET. In order to find the resistance needed these resistors, the input capacitance and charge on each MOSFET were used to calculate the voltage, as shown in Equation (13). Protection designs are borrowed from designs in [1].

$$V = \frac{q}{C} = \frac{35nC}{1880\,pF} = 18.62V \tag{13}$$

Specific device rating of the PMOS was obtained from its datasheet. The input capacitance C_{iss} is 1880 pF, and the total gate charge Q_g is 35 nC; these values were found on the data sheets of the MOSFETs. Using the voltage calculated from equation (13) and a current of about 0.5 A, the amount of resistance needed can be calculated, as shown in Equation (14). Therefore, the decision is to use a resistance of 50Ω on each gate.

$$R_{gate} = \frac{V}{I} = \frac{18.62}{0.5} = 37.23\Omega \tag{14}$$

In order to power the driver chip, 5V logic that comes from the output of the buck converter (LM2825N-5-ND) is connected to pin nine logic supply and pin six chip enable on the driver chip. Since the PMOS gate needs to be pulled to 28V to allow a Vsg of 0V, the bulk voltage of 28V is connected directly to the Vs pin four of the driver chip. In addition, a 100nF Capacitor is placed on both the pin four bulk supply and pin nine logic supply of the driver to filter the AC ripple on the DC source. Lastly, a current sensing resistor of 20 ohms is placed on pin one.

2.3.7 System Control

The FDS sends command to the drive circuit based on a look up table and a PID control loop. At the startup of the FDS, the PIC will calibrate its look up table. This look up table is consisting of PWM duty ratio values when given position values. During the startup calibration, the PIC slowly increases the PWM duty ratio until the maximum position is found. Once the maximum position is found, the PIC slowly decreases the PWM ratio and records its position in a look up table. These positions won't fill in all positions on the look up table since the PIC is commanding increments of PWM duty ratio instead of position. Some positions will have no duty cycle ratio value associated with them, since theposition is non-continuously sampled (i.e. there are position gaps between recorded data points). The solution is to average the nearest known positions to fill in these empty tables. The algorithm to average these values is shown in Figure 2.3.6 below.

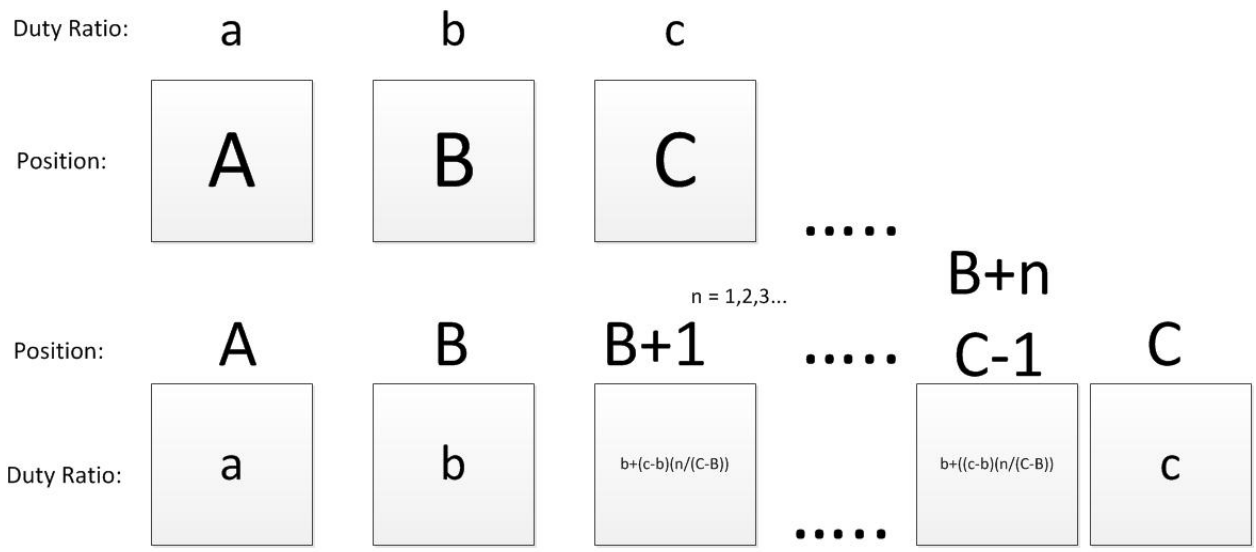

Figure 2.3.6 Positional Averaging Algorithm

Once the averaging is complete, the look up table of duty ratio to position is completely filled. This look up table will be used as a linearizer of duty ratio to the measured position as shown in Figure 2.3.7. In real application, the FDS would be calibrated on a test setup of a pressurized fluid line. Once the calibration is done, the look up table can be stored in the PIC. This look up table then can be used as long as the fluid pressure and value size stays relatively the same.

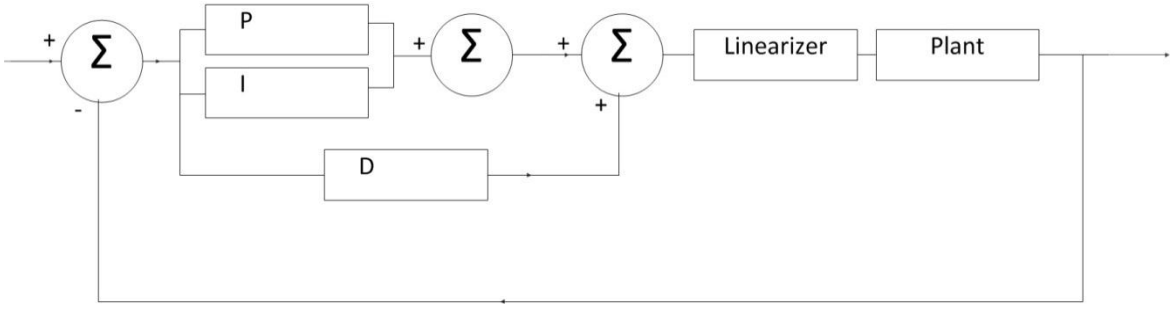

Figure 2.3.7 PID Control Diagram

Once the FDS is initiated, the position command is read by the ADC from the POT each program cycle. This command is the ideal output, or set-point (SP). During the same cycle, the DC sensor voltage is sampled from the ADC and converted into a position value. This value is the process variable (PV). The difference of SP and PV, SP – PV, is the position error. The integral of the error is found discretely by multiplying the error by the time interval of the sampling cycle, which is 10ms. Gain coefficients, Kp and Ki, will be multiplied with the proportional error and integral error respectively.

The sum of the error is converted to a new desired angle by adding PV on to the PI error. This new desired position will then be used to calculate a derivative error term for damping purposes (i.e. to prevent excessive positioning overshoot). The derivative term is calculated by finding the difference between the error of the current cycle and the error of the previous cycle(s), and divides the difference

by the period of the ADC sampling cycle. This error will then be added to the PI error after being scaled by the derivative gain coefficient, Kd. This overall PID error is converted to a new desired angle again by adding PV on to the PID error.

Finally, the new desired angle is placed into the look table and linearized into a PWM that drives the motor into the corresponding position.

2.3.8 System Control Dynamics

The nature of a PID based control is self-correcting. The integral error accumulates when the motor is out of position; this accumulation is then used to correct the position. Figure 2.3.8 in Appendix B shows the dynamic response of the FDS. The Y-axis is the voltage of the position sensor, which corresponds linearly to the position. The X-axis is time. As one can see from Figure 2.3.8 in Appendix B, at 150ms the position the motor has over-shot the desired position; the integral term pulled the motor back and eventually settled into desired position. The derivative term served as a damping term. When there is a large difference between the current desired position and measured position as well as previous desired position and measured position, there will be a large negative derivative factor in the sum of PID error. This prevents over shoot of the motor. The nature of PID control will compensate for any external factor that may offset the motor, such as voltage fluctuation and pressure fluctuation.

2.3.9 Bulk Voltage Correction Factor

An accurate proportional is the key to fast response time. If the motors initial guess with the proportional term is very close to the desired position, then less time will be needed for the integral term to accumulate to compensate for the error.

When the rail or bulk voltage drops, less current will flow into the motor with the same duty ratio. To compensate for this effect, an additional ADC channel is opened in order to sample the bulk voltage. The bulk voltage value is translated into a percent offset on the PWM duty ratio; see equation (15).

$$Duty\ Ratio = Duty\ Ratio[New\ Desired\ Position] + ((NormalBulkVoltage - CurrentBulk\ Voltage)/NormalBulkVoltage) * VoltageCorrectionFactor \quad (15)$$

2.3.10 Modularity of Design

The main Fluid Delivery System can be broken down into three modules, which are shown below in Figure 2.3.8. These modules are the user interface, Torque Motor Control, and Motor Interface. The user interface consists of the LCD screen and the potentiometer. The user sends out an analog position signal and receives the desired position and current position on the LCD. The torque motor control consists of the PIC which links the user interface to the motor interface. The PIC sends two PWM signals to the drive circuitry for the torque motor and receives the analogue position signal and SPI motor ID signal. The motor interface depends on the type of motor that is connected.

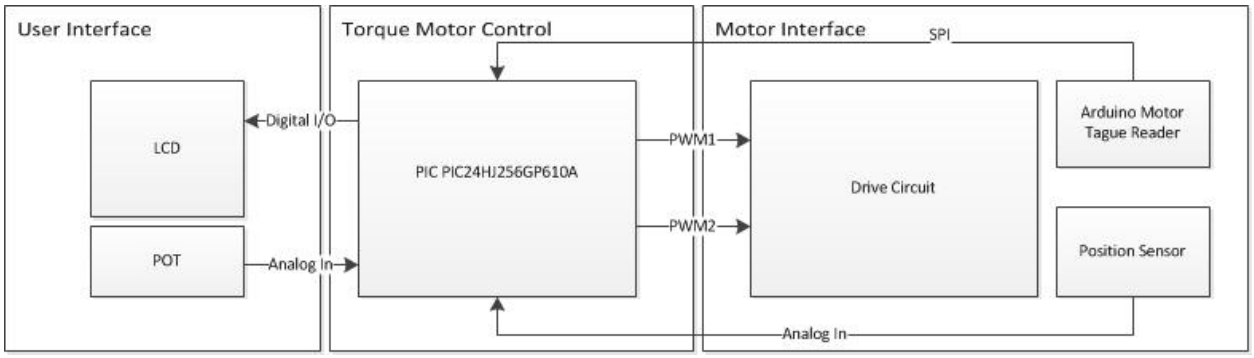

Figure 2.3.8 Modular Design

In the case of a torque motor, the SPI motor ID signal is generated by the Arduino Uno, which is enabled as a slave. The clock signal of the PIC dictates the timing for the data transmission. The Arduino is put into slave mode by changing the bits of the control register in the Arduino. The PIC is the master and it enables Arduino Uno as slave by making the slave enable line low. Once the Arduino is initialized, the PIC will receive the data that identifies the motor.

The SPI interface consists of four lines as shown in Figure 2.3.9: a master-in, slave-out line (MISO), a master-out, slave-in line (MOSI), a clock signal, and a slave enable line. The MISO line is connected to pin 12 on the Arduino, the MOSI line is not used, the clock signal is connected to pin 13, and the slave enable line is connected to pin ten. The Arduino is powered by a five volt output from the power supply chip in the drive circuit, and the PIC and Arduino must have their grounds connected to each other in order to communicate with the same reference point.

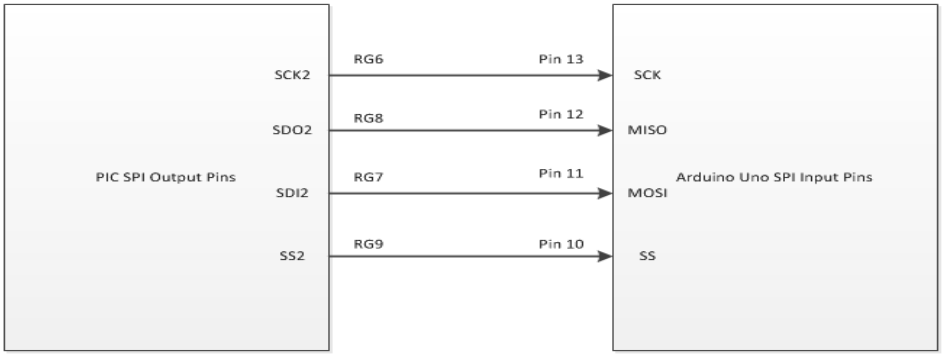

Figure 2.3.9 SPI Ports

For a torque motor, the letter “A” is sent from the Arduino telling the PIC that a torque motor is connected. This signal tells the PIC to use the PWM signals to control the torque motor. Since SPI is a standard interface, it is good choice for modular design. Modular design enables different types of motors to be controlled. The motor interface block can be taken out of the system and be replaced by another motor such as a stepper motor as shown in Figure 2.3.10 below. If a stepper motor is connected, the PIC will use SPI to control the stepper motor drive circuit, and the PWM waveforms will be disabled.

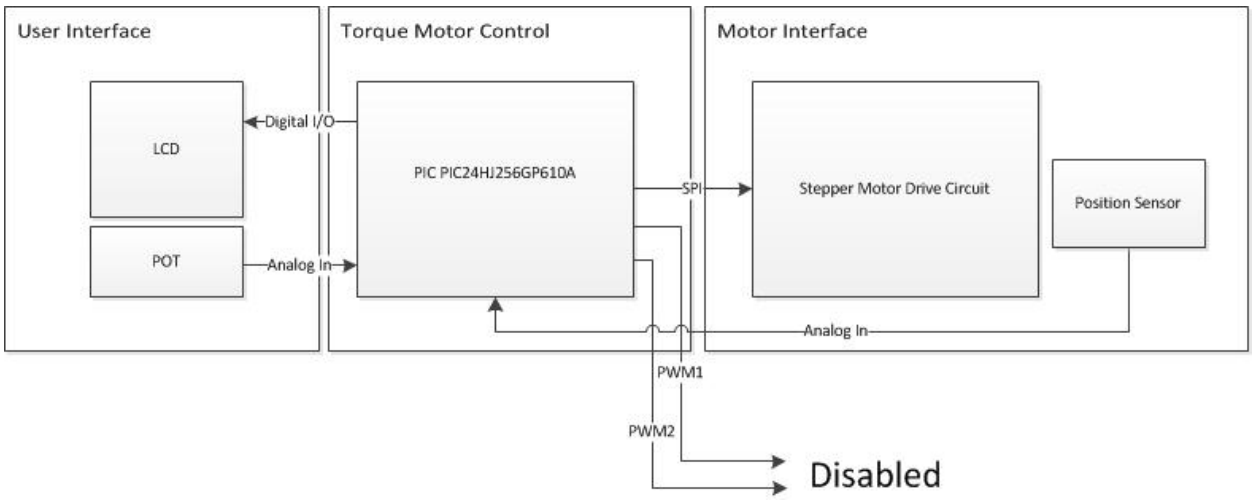

Figure 2.3.10 Modular Design Ripped for Stepper Motor Control

Both motors utilize SPI and will have feedback; the motor interface module is easily interchangeable. The PIC is capable of controlling both of the motors depending on the configuration that is connected. Therefore, the modular design of the circuit provides greater flexibility to the user on which type of motor can be used.

3. Verification of Design

3.1 Testing Procedures

For correct operation and functionality of this design, there are four key elements which need to work correctly: the power supply, the drive circuitry, the PIC board, and the Arduino Uno. Additionally, overall system performance tests needed to be conducted to show that all of the key elements worked properly with each other to produce the desired functionality. Further details about the testing procedures and verification can be found in Appendix A.

3.1.1 Power Supply Test

The master component of the power supply is a 28 V DC input to simulate the rail voltage of one of Aero Engine Controls' systems. An adjustable Kenwood regulated DC supply was used to provide the necessary input voltage.

One of the project goals was to make the system run entirely off of the 28 V rail. However, several other input voltage levels were needed to power other components of the system such as the drive circuit, PIC, and Arduino. Values of both 3.3 V and 5 V were needed as a part of the supply for these components. As such, two buck converters were used to step down the rail voltage to these values. The outputs of these buck converters were then measured with a digital multi-meter to verify appropriate voltage levels.

3.1.2 Drive Circuitry Tests

In order for the drive circuit to function properly, it is necessary for any PWM signal input to the gate to produce a corresponding "PWM" voltage response waveform with identical frequency and duty ratio, or the input will not produce the expected current output. To test this, a PWM signal was input to the MOSFET gates. The resulting drain to source voltage waveform along with the PWM input waveform were then observed on an oscilloscope to verify both the frequency and duty cycle match. This duty cycle directly corresponds to the average output current of the drive.

Additionally, the current output corresponding to the input PWM should be nearly constant in amplitude so as to provide the motor with constant torque. As such, any fluctuation in current should be regulated to within two tenths of an amp, peak to peak. This was also measured on the oscilloscope.

3.1.3 PIC Board Testing

As the "brains of the operation," the PIC interprets user input from the potentiometer and converts it into a PWM signal so as to generate the corresponding current needed to position the motor at the desired set point. Therefore, the duty ratio of the PWM must fluctuate evenly with the POT (i.e. 50% POT location must correspond to 50% duty ratio). This in turn will generate an angular change equivalent in percentage to that of the POT input. Please see appendix A for more details on specific inputs tested. Results were verified using an oscilloscope.

Furthermore, to integrate the torque motor module into a pre-existing system, it was necessary for the PIC to communicate with the Arduino using a SPI bus. To verify, the output clock from the PIC was

measured to be 50 kHz the the PIC was able to output and read hexadecimal “A” to communicate that the system in place uses a torque motor.

3.1.4 Arduino Uno Test

The Arduino is the communication device which allows for compatibility with multiple types of motor systems (e.g. torque motor, stepper motor, etc.). As such, the MISO line to/from the Arduino needed to be able to send and receive hexadecimal “A,” which communicates that it is connected to a torque motor module.

3.1.5 System Performance Test

For the system as a whole to be a success, accurate positioning control must be attained. As such, when the rail voltage is well regulated to 28 V, the lever arm must be able to increase to its maximized angle upon inputting a maximum POT position.

Additionally, the system should be able to position itself quickly. For any input from the POT, the system should move into a new steady state position within 700 milliseconds. This was verified using an oscilloscope once again. See appendix A for further explanation.

3.2 Measurements and Results

For the project demonstration, the overall system requirements were met, so individual module testing was unnecessary. However, data for tests that were completed are shown in Appendix D.

3.3 Discussion of Results

3.3.1 Power Supply Test

Multi-meter measurements of the power supply voltage are taken from the 5V to ground and 3.3V to ground. The 5V supply is measured at 5.01V and 3.3V supply was measured to be 3.29V. These values are confirmed to be within tolerated range of $\pm$0.2V from the rated voltage value. The PIC boot up and the Arduino Uno boot up also confirms the 3.3V supply and 5V supply is working.

3.3.2 Drive Circuitry Tests

The oscilloscope measurement of the drain to ground voltage is shown on Figures 3.2.1-3.2.3 in Appendix D. Channel one in Figures 3.2.1-3.2.3 is the drain to ground voltage. Channel two in Figures 3.2.1-3.2.3 is the PIC’s PWM signal. The frequencies indeed match up. Channel three of Figures 3.2.1-3.2.3 is the current input for the torque motor. The current is indeed within $\pm$0 .2 A from the peak to dips of the wave for all cases.

3.3.3 PIC Board Testing

Channel two of Figure 3.2.1-3.2.3 are generated by the PIC. This confirms the duty ratio of PWM waveform can indeed be controlled.

The LCD is confirmed to be working since Figure 3.3.1 in Appendix D shows the LCD is displaying the position information and desired position information.

In order to confirm the position sensor is working, the potential meter is set to maximum position corresponding to 100% fluid flow. The maximum angle of the lever is then recorded to be 55 degrees.

The desired angle value can be calculated by multiplying the measured fluid percent by the maximum angle to find the measured angle. The sensor is confirmed to be accurate within ± 2 degrees with input measured at 25%, 50%, and 75% in Figure 3.3.2-3.3.4 in Appendix D. Voltage sampling is confirmed to be working.

When dropping the bulk voltage on the power supply is dropped to 24V, the warning is confirmed to be displayed on the LCD as shown in Figure 3.3.5 in Appendix D.

3.3.4 Arduino Uno Test

By connecting the oscilloscope to the MISO, CLK, and SLAVE enable, one can see the Arduino is running as a slave corrected. In Figure 3.3.6 in Appendix D, channel one is the MISO, channel two is the slave enable, and channel three is the clock. The Arduino is set to output letter 'A'; this corresponds to '01000001' in binary. This output is corrected shown on the scope data in channel one; it is in synch with the slave enable line and clock.

The PIC is confirmed to receive the letter 'A' and is displaying the received SPI input in Figure 3.3.7 in Appendix D.

3.3.5 System Performance Tests

The position sensor is connected to the oscilloscope and voltage is measured. Figure 3.3.8 in Appendix D shows the transient response of the FDS. Voltage is linearly related to the position of the motor. The motor settles into threshold voltage of ± 0 .5V which is proportional to ± 4 degrees within 700 ms in all tested cases.

The range of position is confirmed to be 99% to 0% when tested on 28V bulk supply.

The integral term is confirmed to be working since the measure lever position is closer to desired position when the integral coefficient is set to be greater than zero.

4. Costs

4.1 Parts

Table 1. Parts Cost

From Microchip				
Part	Product Number	Quantity	Cost ($)	Actual Cost with Shipping ($)
PIC24H 100P to 100P TQFP Plug-In Module	MA240012	1	18.75	
Explorer 16 Demo Board	DM240001	1	97.49	
9V Wall Mount Power Supply	AC002014	1	15	152.14
Prototype PICtail Plus Daughter Board	AC164126	1	15	23.09
PICkit 3 In-Circuit Debugger	PG164130	1	33.71	43.44
TRULY LCD 16x2 3.3V (TSB1G7000)	LCD0016	1	9.72	12.08
TRULY LCD 16x2 3.3V (TSB1G7000)	LCD0016	2	9.72	17.35
From SonceBoz				
Part	Product Number	Quantity	Cost ($)	Actual Cost with Shipping ($)
Torque Motor 4236	4236R000	1	239	239
From DigiKey				
Part	Product Number	Quantity	Cost ($)	Actual Cost with Shipping ($)
POT 10K	91A1-G28-B15L-ND	1	8.12	17.88
Buck Converter with 5V output	LM2825N-5.0-ND	1	37.8	
Buck Converter with 3.3V output	LM2825N-3.3-ND	1	37.8	75.6
MOSFET P-CH 50V 9.7A TO-220 AB	IRF9Z20PBF-ND	4	1.71	
MOSFET N-CH 40V 12A TO-220	FDP8447L-ND	4	1.34	
Buck Converter with 12V output	LM2825N-12.0-ND	1	37.8	57.86
IC DRIVER FET FULL 80V 20-SOIC	HIP4081 AIBZT CT-ND	1	5.1	10.15
IC OPAMP 1.2MHz Quad	296-1391-5-ND	1	0.53	
Diode Schottky 20V 200MA	568-1617-1-ND	10	0.339	6.46
From Mouser				
Part	Product Number	Quantity	Cost ($)	Actual Cost with Shipping ($)
SMT LED PLCC-2	ASMT-UBB5-NS8Q2	6	0.8	
8Mhz Oscillator	XT9M20ANA8M	1	0.95	
Push Button Switch	TL6100BF300RP	5	0.98	7
From Part Shop				
Part	Product Number	Quantity	Cost ($)	Actual Cost with Shipping ($)
Full Bridge Driver	L298N	1	4.45	
Hex Inverter	SN74LS04NE4	1	0.54	

.1uF Cap	RDED72E104K3M1C11A	12	0.8	
50uF Eletrolytic Cap	EEU-FR1H151	4	0.63	
Various Resistor		25	0.14	
Diode Rectifier 5A	C4D05120A	1	7.2	
10V Zener Diode	BZX79C10	2	0.13	
From ECE Store				
Part	Product Number	Quantity	Cost ($)	Actual Cost with Shipping ($)
Arduino Uno	A000066	2	25	54
Total Parts Cost				**716.05**

4.2 Labor

Labor Cost: ($30/hour)(50 days of labor)(3 hours/day)(3 workers) = $13,500 (16)

4.3 Total Cost

Total Cost: Parts Cost + Labor Cost = $716.05 + $13,500 = $14,216.05 (17)

5. Conclusion

5.1 Accomplishments

In addition to the basic ability of position the motor and holding the motor in a desired position, the motor is able to compensate for external forces and factors. The FDS will put the motor into the desired position and hold it steady within a reasonable amount of error when the bulk voltage fluctuates. The FDS also holds the position when additional external forces are applied on the motor.

The FDS was designed to be part of a modular configurable system. This requires that it must be able to be tuned for various situations and platforms. The calibration system is the first part that will help achieve this. The FDS is able to learn the force to position relationship and linearize it for the given situation. Various vehicles require different types of response. Additionally, a tuning program is built into the FDS. This tuning program allows the user to change the Ki and Kd term on the FDS PID loop. The PID loop can be tuned to balance its response and accuracy. This will benefit the testing phase of the FDS since the FDS do not require to be reprogrammed to be tuned.

5.2 Uncertainties

In a realistic situation, the pressure on the fluid valve varies with time. The FDS must be able to adjust to this change fast enough in order to hold the position of the shaft.

5.3 Ethical considerations

In order to ensure that the torque motor module meets ethical standards, a variety of ethical considerations must be made. According to the IEEE code of ethics number one, the project designers must accept responsibility in making decisions consistent with the safety, health, and welfare of the public and to disclose promptly factors that might endanger the public or the environment. The torque motor module was designed for aerospace applications, and therefore it is very important that all parts of the design operate correctly and communicate properly with each other. If one part of the design does not work, then the motor will not operate properly. This will cause safety concerns as the necessary fluid flow will not be provided. Redundant safety circuitry was developed in order to ensure that the design is the safest and most reliable it can possibly be. As such, a safe-drive state is implemented in order to correspond to the situation where the PIC has lost communication with the EEC. The safe-drive state will be defined by the particular vehicle that the torque motor module is implemented in. Also, the amount of torque that is actually delivered to the motor must match the amount specified by the user from the potentiometer. If there is a mismatch, then the product will not function as specified or expected, thus causing safety concerns to arise as aforementioned. Additionally, highly accurate positioning at a fast rate to regulate fluid flow quickly and properly was implemented. Minimal overshoot was design for so as to not waste fluid, which protects the environment. Warning features were also added as a precaution, should the system performance start declining due to bulk voltage drop. In addition, aircraft grade approved components must be used as an appropriate safety measure. Furthermore, this project falls under the IEEE code of ethics number five regarding improvement in the understanding of technology. This design builds off of previous design work and seeks to further develop the technology for fluid delivery systems. Finally, IEEE code of ethics number

ten, which states that colleagues and co-workers must be assisted in their professional development, must be followed. The leadership and mentorship of AEC to the project helped facilitate this growth for all team members.

5.4 Future work

The current FDS is robust but the response time could be faster. A more intuitive control loop can be programmed into the PIC. In additional to bulk voltage sampling and positional feedback, a fluid pressure sensor can be implemented into the FDS. The fluid pressure can be used as a feed forward control for the control loop.

Fine tuning of the PID loop can be done on Simulink. A transfer function can be found for the motor on a particular fluid system. With this transfer function, Simulink can accurately tune the Ki, Kp, and Kd term for the desired response.

References

[1] S. Lloyd, M. Mcfadden, D. Jennings, "Supporting Documentation for the OSMC Project," *OSMC*, Vol. , no. , pp. 23, March 30 2002.[]. :http://www.robotpower.com/downloads/OSMC_project_documentation_V4_25.pdf. [Accessed 2/12/2012]

[2] Microchip, " PIC24HJXXXGPX06A/X08A/X10A Data Sheet," *Microchip*, Vol. , no. , pp. 314, 18 February 2011.[Data Sheet]. Microchip:http://ww1.microchip.com/downloads/en/DeviceDoc/70592C.pdf. [Accessed 3/23/2012]

[3] Microchip, "Explorer 16 Development Board," *Microchip*, Vol. , no. , pp. 50, 31 October 2005.[User Guide]. Microchip:http://ww1.microchip.com/downloads/en/DeviceDoc/Explorer%2016%20User%20Guide%2051589a.pdf. [Accessed 3/23/2012]

[4] Atmel, "8-bit Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash," *Atmel*, Vol. , no. , pp. 448, October 2009.[Data Sheet]. :http://www.atmel.com/Images/doc8161.pdf. [Accessed 3/23/2012]

Appendix A Requirement and Verification Table

Table 2. System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. Drive Circuitry is working a. The FET on the drive circuit is manipulating the PWM waveform properly. b. The output current can be manipulated by the PWM waveform and is remaining stable for the PWM input.	1. a. The source to drain on the FET in the drive circuit has the same frequency and duty ratio as the PWM waveform from the PIC. b. The output current is increased by the duty ratio and must remain a constant value within $\pm$0.2 A from the peaks to the dips when measuring from the oscilloscope.	Y
2. Power Supply is working a. The buck converter is outputting the correct value for the Arduino and driver chip. b. The buck converter is outputting the correct value for the PIC.	2. a. 5 V is being provided to power the Arduino. The power supply provides a constant DC value of within $\pm$0.2 V measured with a multimeter given that a constant rail voltage of 28V is applied. b. 3.3 volts is being provided to power the PIC. The power supply provides a constant DC value of within $\pm$0.1 V measured with a multimeter given that a constant rail voltage of 28 V is applied.	Y
3. PIC Board is working a. The PIC board is producing a PWM waveform based on the value produced from the potentiometer. b. The LCD display is functioning properly. c. Voltage sampling is working d. The PIC produces a warning signal when it should.	3. a. The duty ratio of the PWM waveform can be controlled to set values of 0-50% duty ratio shown on the oscilloscope. b. The LCD lights up and displays the potentiometer percent voltage and sensor percent voltage compared to the values received from the debugger. c. When the lever is set to	Y

	25%, 50%, and 75% of the maximum angle, the lever goes to the specified angle within $\pm 2^{\circ}$. d. "Warning" appears on the screen when the bulk voltage is pulled too low, e.g. around 24 volts.	
4. Arduino Uno SPI is functioning properly and operating in slave mode. a. The Arduino is sending the correct values via the MISO line. b. The Arduino is receiving clock and slave enable lines from the PIC, and it is receiving the correct values.	4. a. The Arduino Uno is sending "A" or '01000001' to the PIC via the MISO line. This will be verified using the oscilloscope. b. The Clock signal and its frequency will be determined by the PIC board output. The Slave enable line should go low to enable the Arduino, and this line will be determined by the output of the PIC when the Arduino transfers the "A" via the MISO line.	Y
5. PIC SPI is functioning properly and operating in master mode. a. There's a clock feeding out. b. There's a slave enable line feeding out. c. The MISO line for the PIC is working properly.	5. a. The frequency of the clock is 50 kHz. b. The slave enable line goes low when data is ready to be sent from the Arduino to the PIC. c. By connecting the MISO and MOSI lines together, the PIC should output an "A" and read an "A".	Y
6. System Performance a. Lever is put into desired position within specified time. b. The PIC board is able to be properly calibrated. c. The PIC is calibrated for the full range of the sensor's measurement angle.	6. a. When the potentiometer is set from 0% to 55%, the system response on the oscilloscope flattens within 700 milliseconds. b. Proportional term should be $\pm 10\%$ within the desired angle. Measurements of the position is read from the LCD which is based on the sensor feedback. c. Turn the potentiometer to	Y

		99% and 0% to see if the measurement on the LCD of the position sensor reaches the desired measurement within $\pm$1% with 28 volts bulk voltage.	

Appendix B Schematics and Pin-Outs

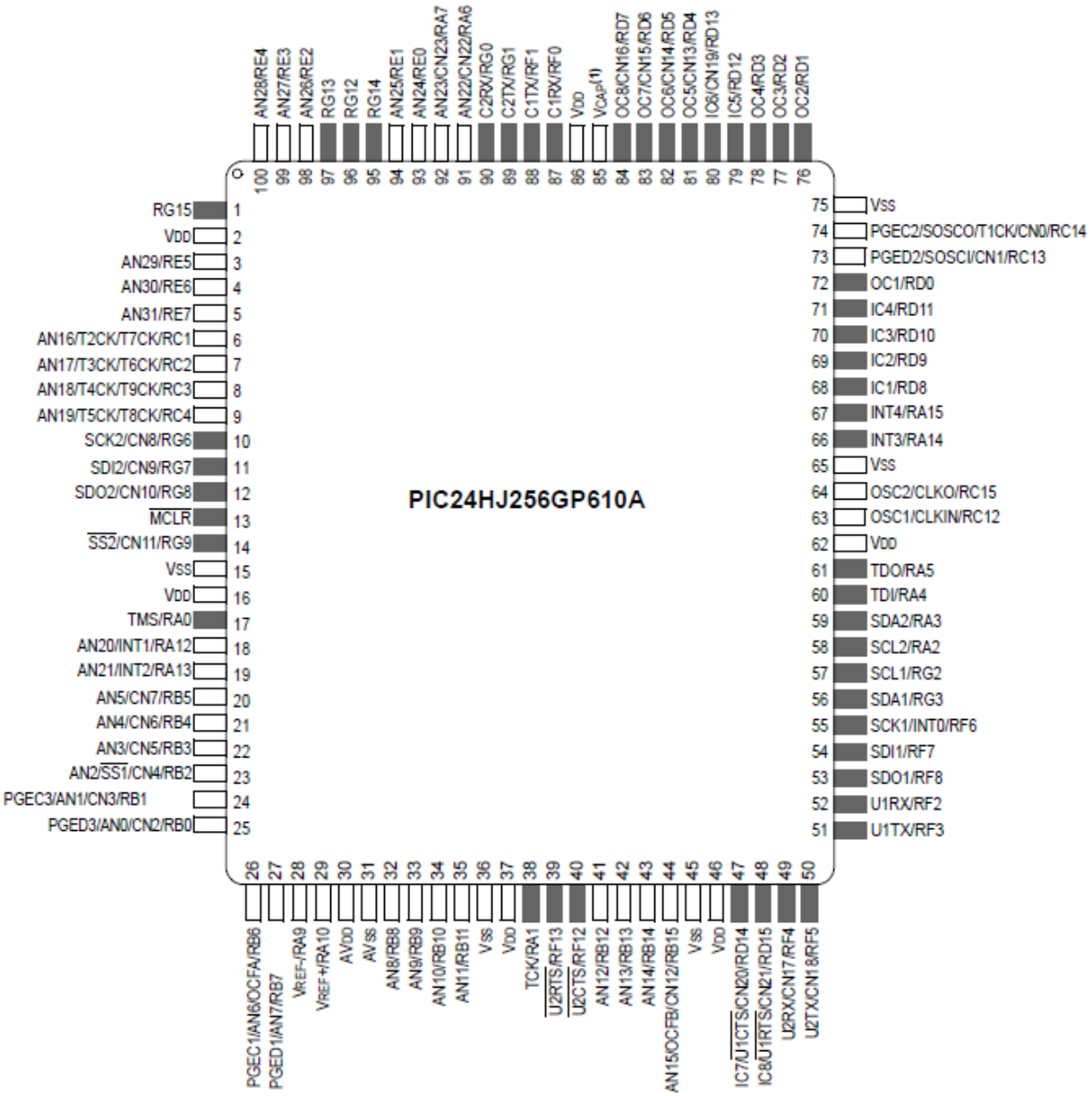

Figure 2.3.2 PIC Pin-Out Diagram [2]

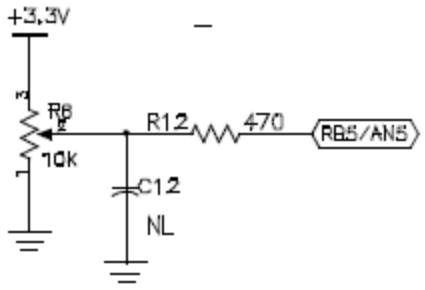

Figure 2.3.3 Potentiometer [3]

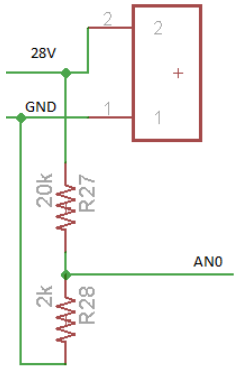

Figure B.1 Bulk Voltage

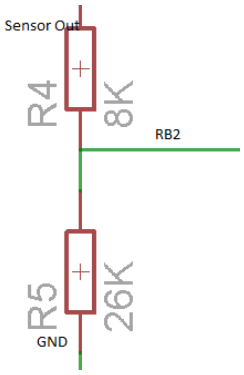

Figure B.2 Sensor Voltage

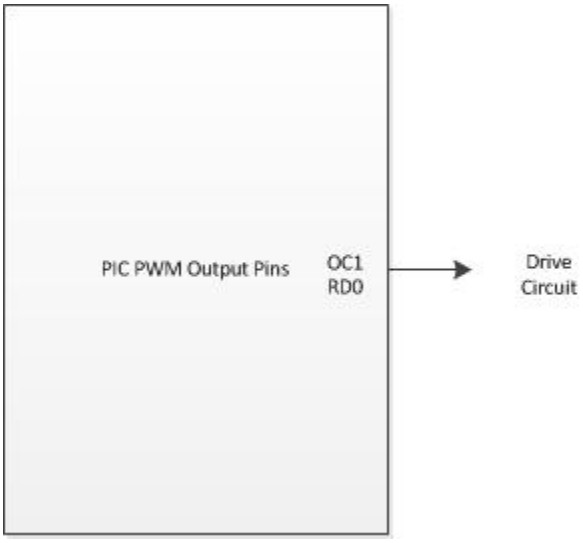

Figure B.3 PWM Pin Out

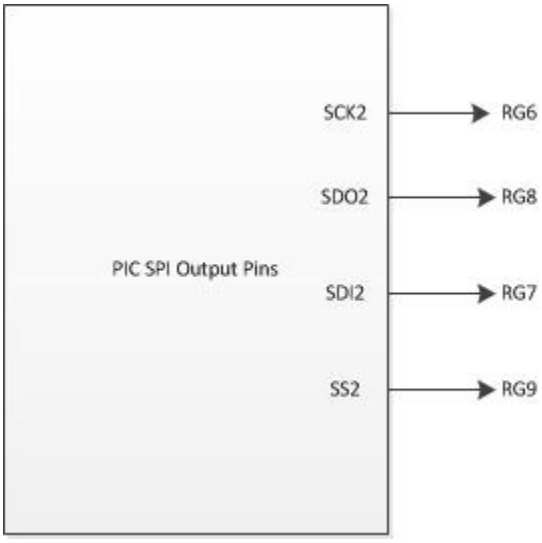

Figure B.4 SPI Ports

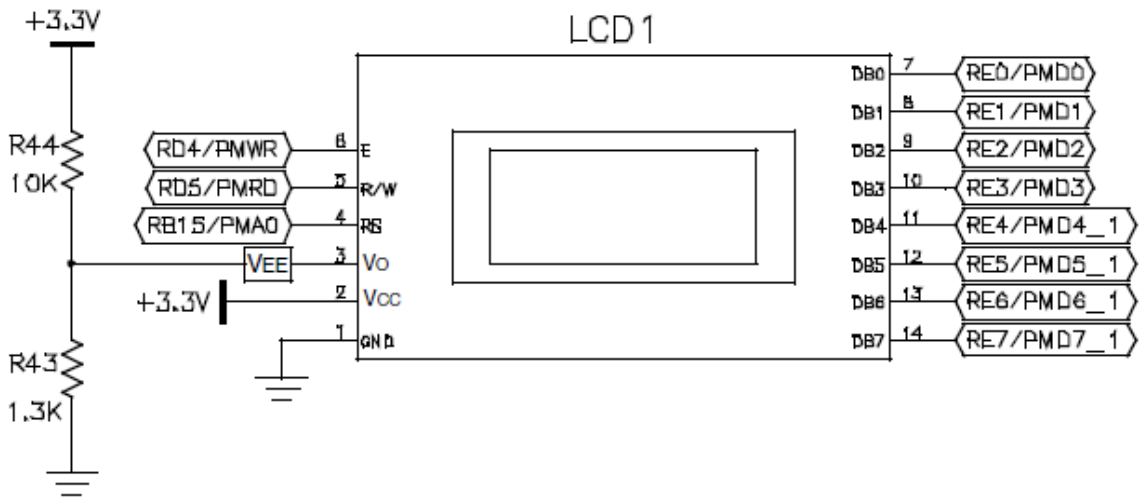

Figure B.5 LCD Schematic [3]

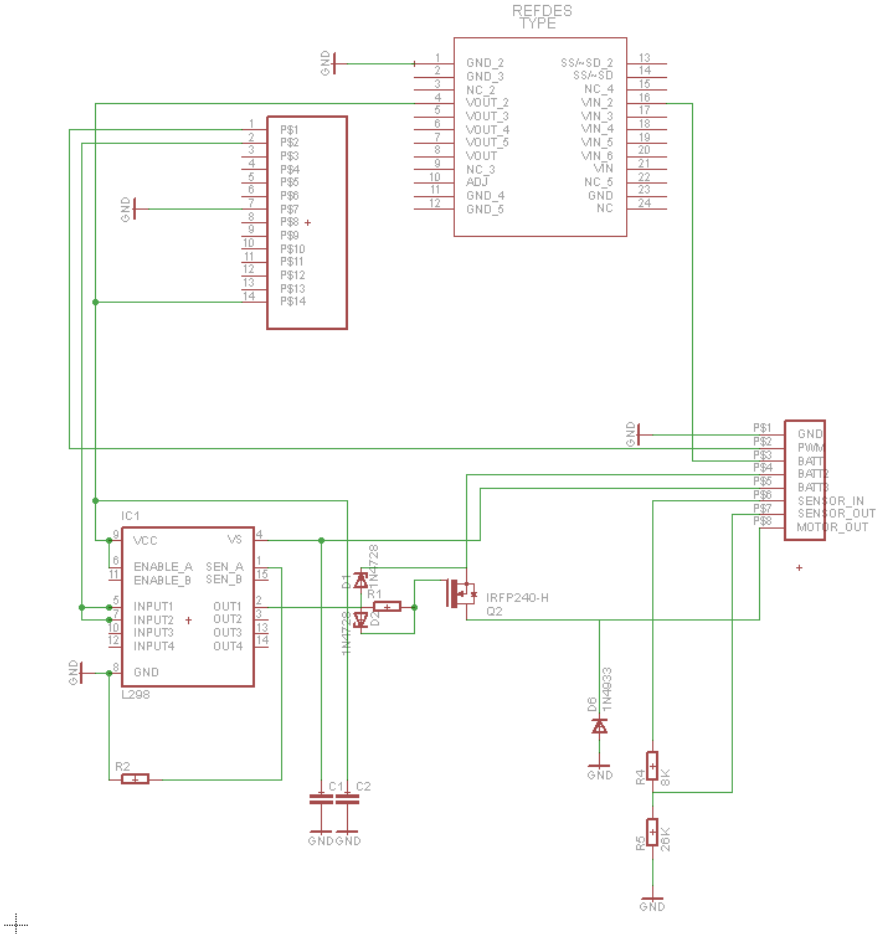

Figure 2.3.5 Schematic for the drive circuitry system

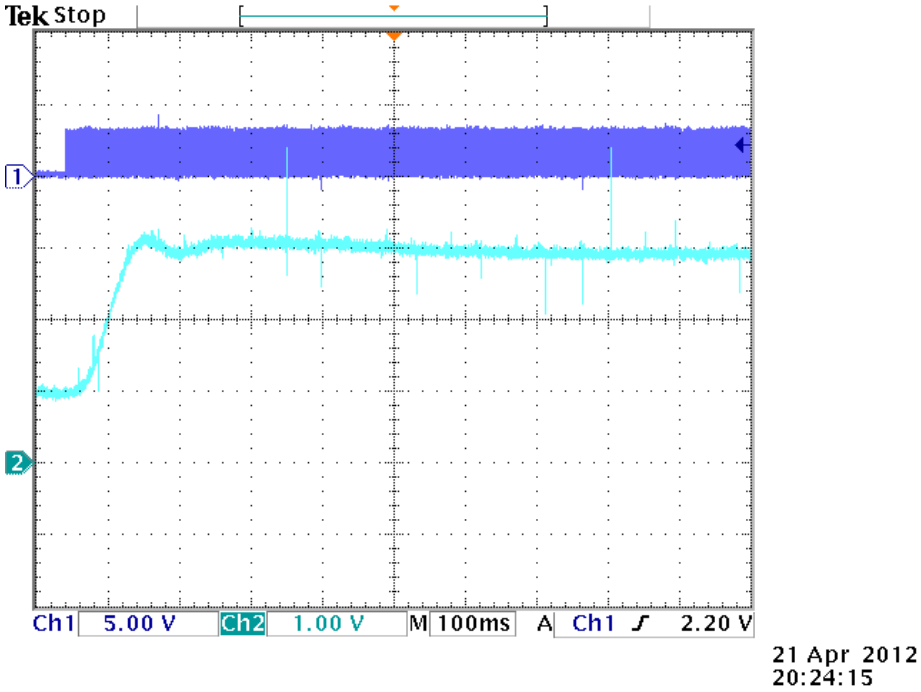

Figure 2.3.8 Zero to 90% Input Response

Appendix C PID Transient Response

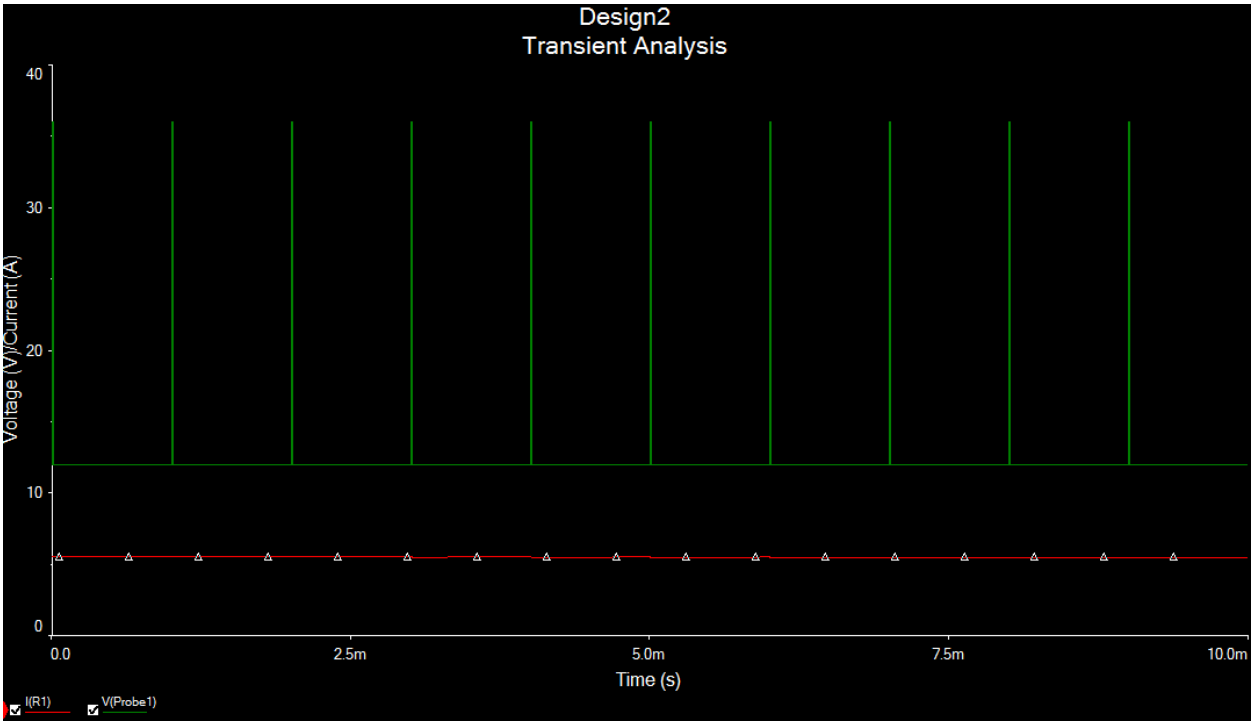

Figure 2.2.5 Current analysis for a voltage with 1% Duty Ratio (99% Duty Ratio Equivalent)

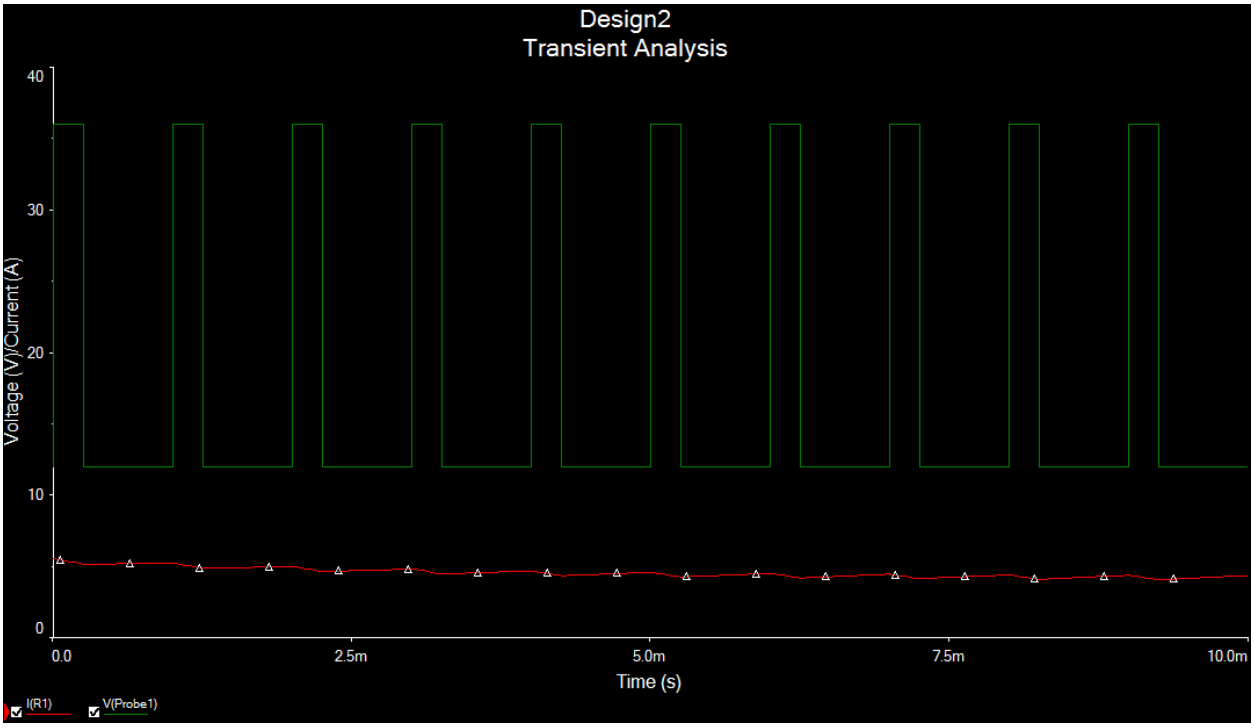

Figure 2.2.6 Current analysis for a voltage with 25% Duty Ratio (75% Duty Ratio Equivalent)

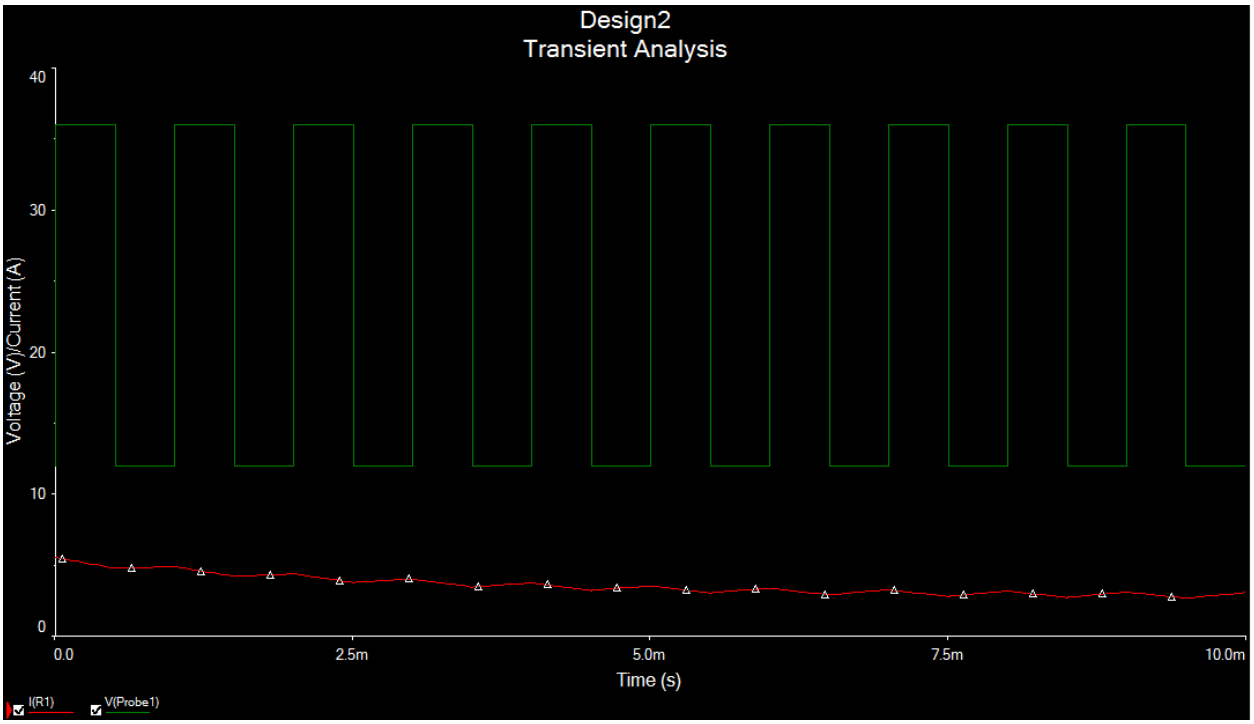

Figure 2.2.7 Current analysis for a voltage with 50% Duty Ratio (50% Duty Ratio Equivalent)

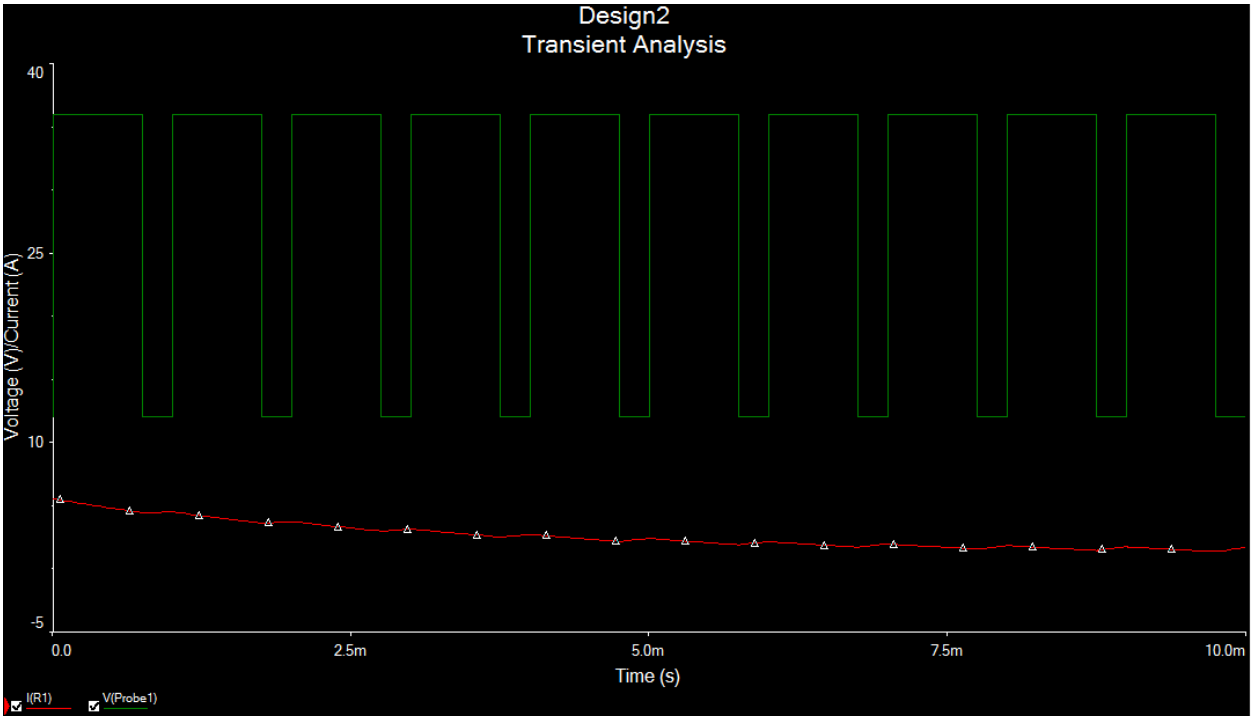

Figure 2.2.8 Current analysis for a voltage with 75% Duty Ratio (25% Duty Ratio Equivalent)

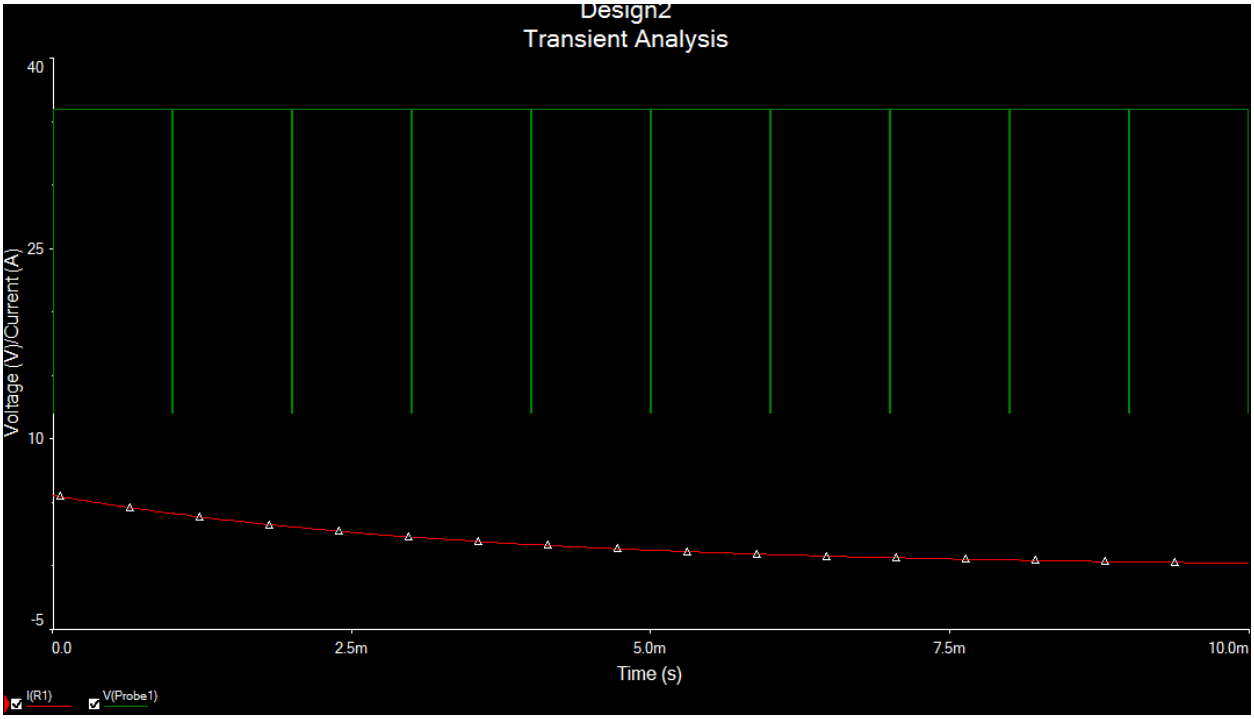

Figure 2.2.9 Current analysis for a voltage with 99% Duty Ratio (1% Duty Ratio Equivalent)

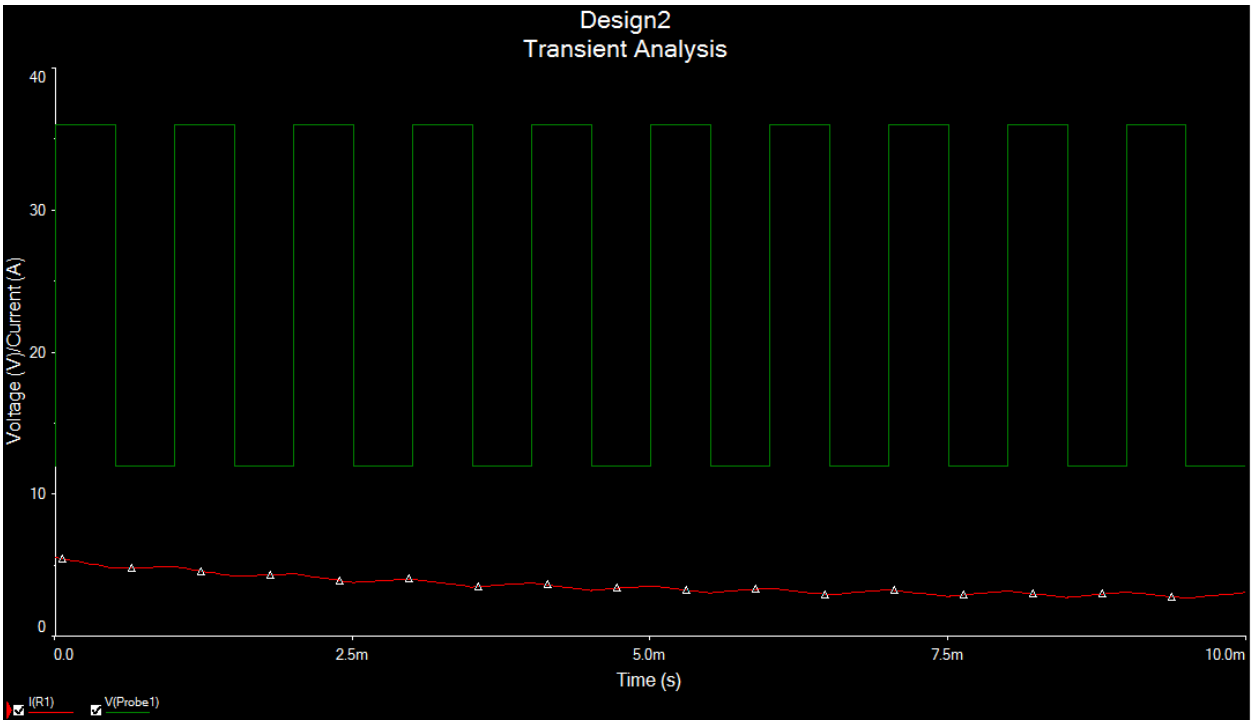

Figure 2.2.10 Voltage and current transient analysis at 1 kHz

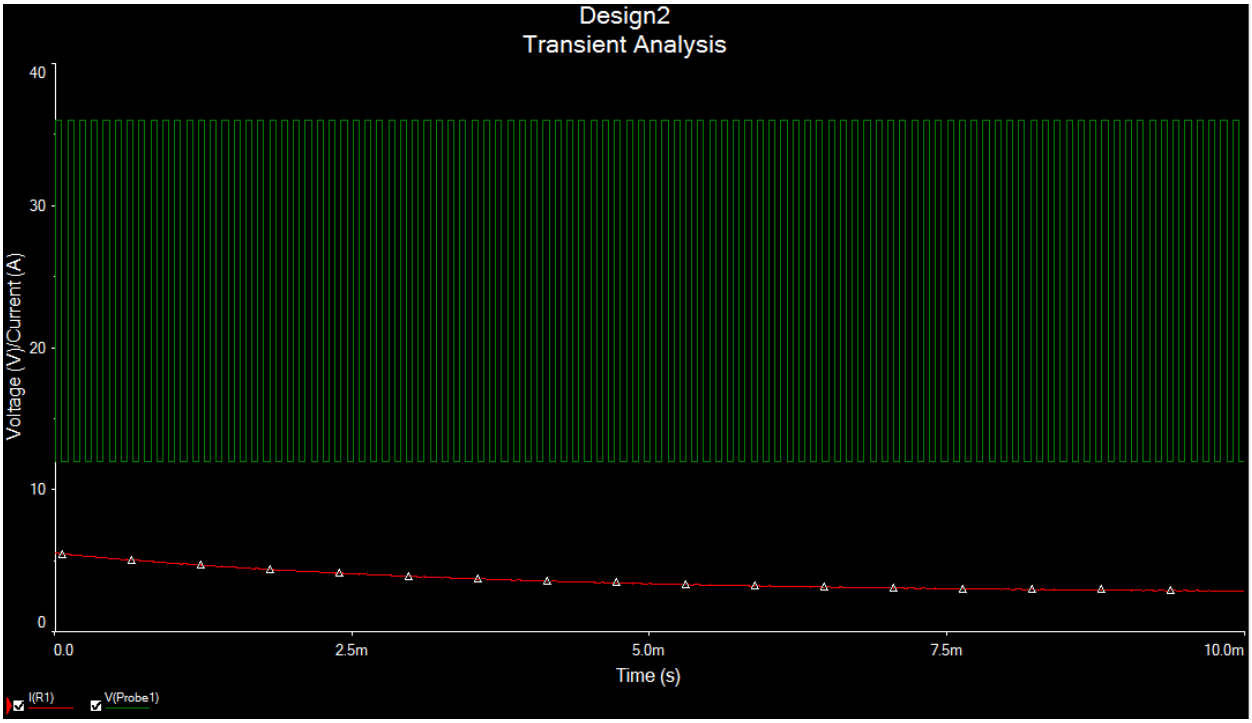

Figure 2.2.11 Voltage and current transient analysis at 10 kHz

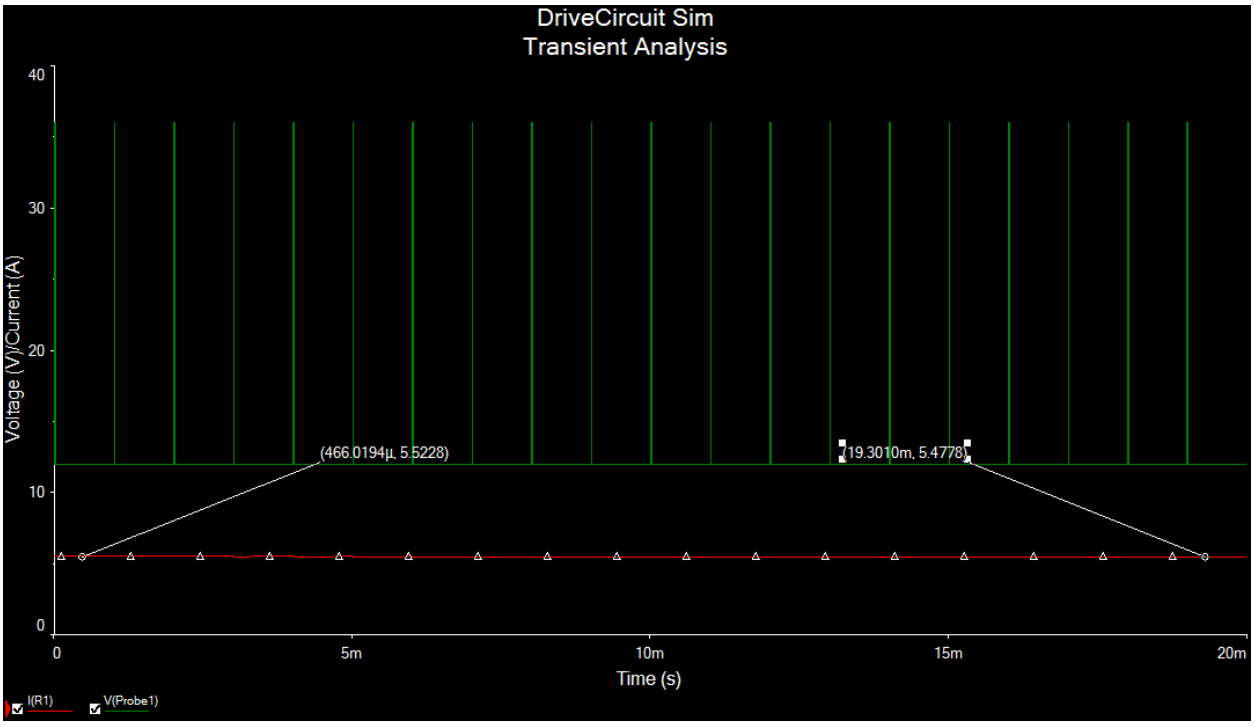

Figure 2.2.12 Current analysis for a voltage with 99% equivalent Duty Cycle

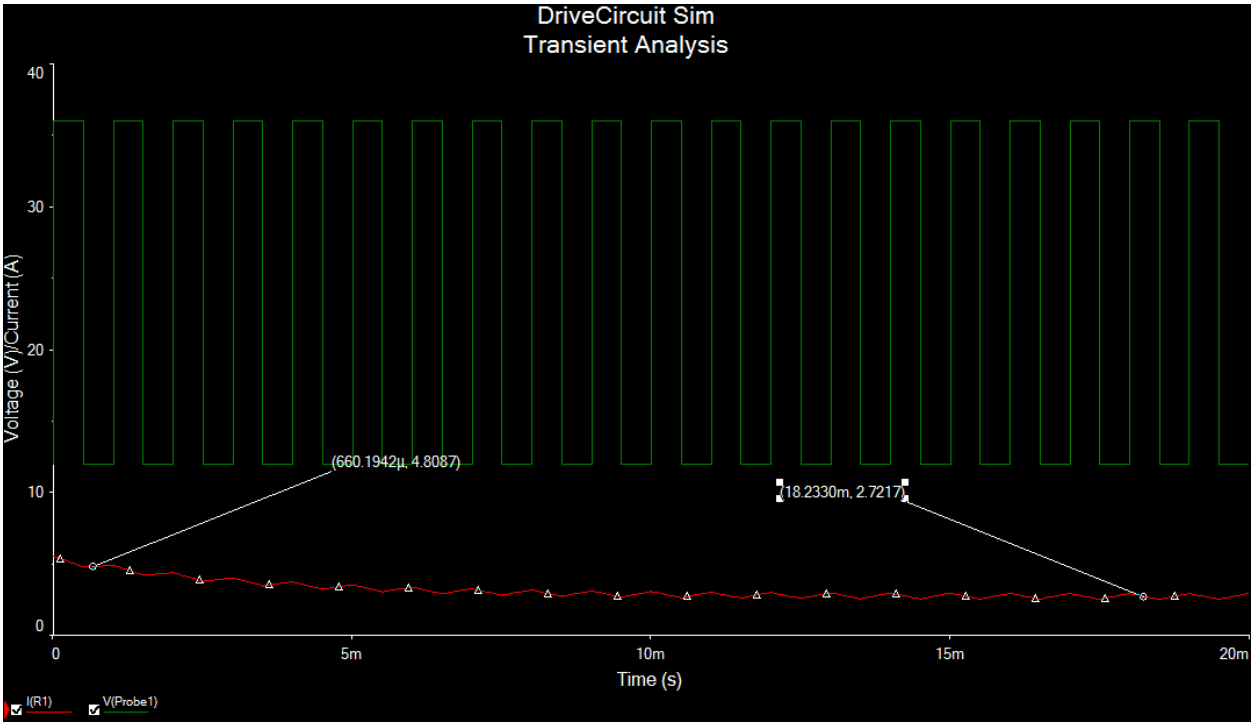

Figure 2.2.13 Current analysis for a voltage with 50% equivalent Duty Cycle

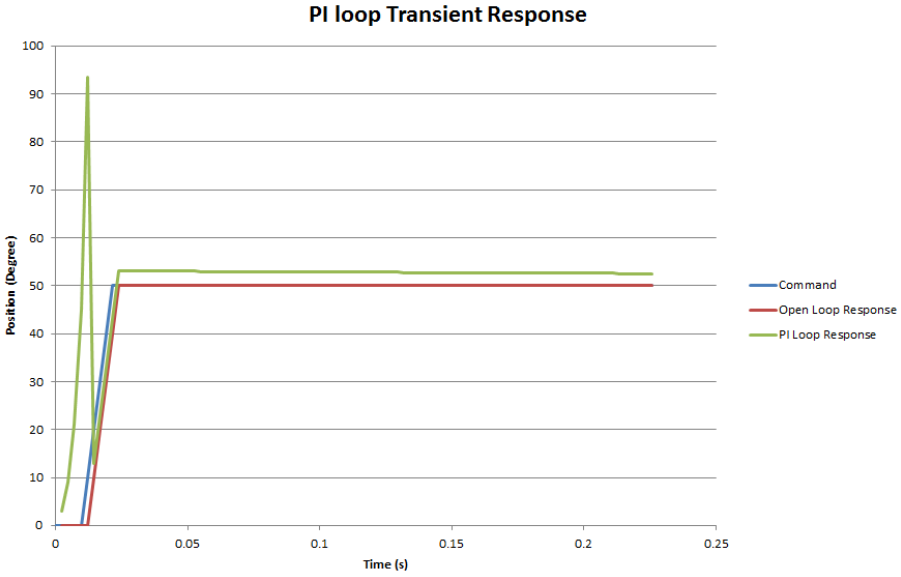

Figure 2.2.14 PI Response with Kp=1 and Ki=1

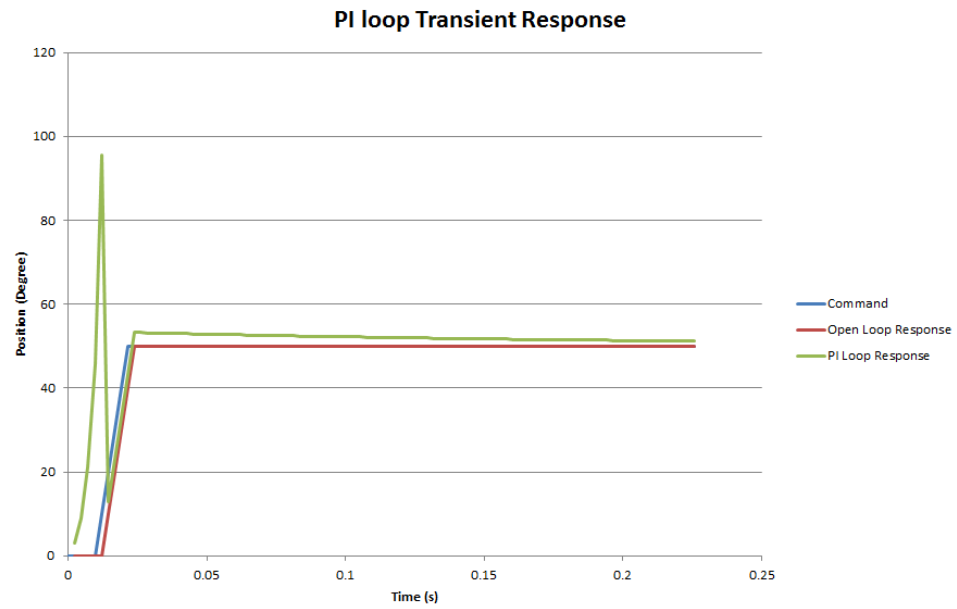

Figure 2.2.15 PI Response with Kp=1 and Ki=5

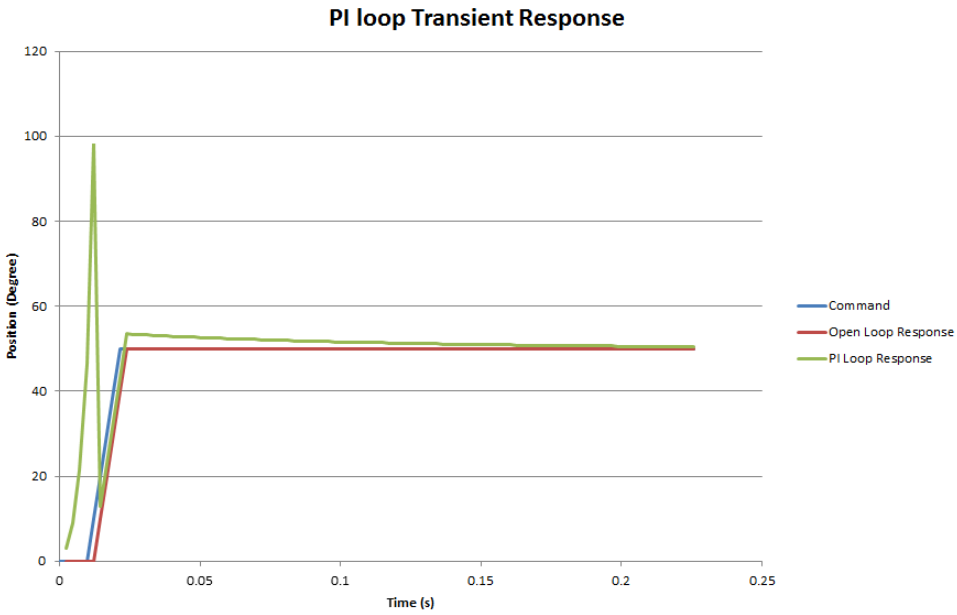

Figure 2.2.16 PI Response with Kp=1 and Ki=10

Appendix D Drive Circuit and Positioning Response

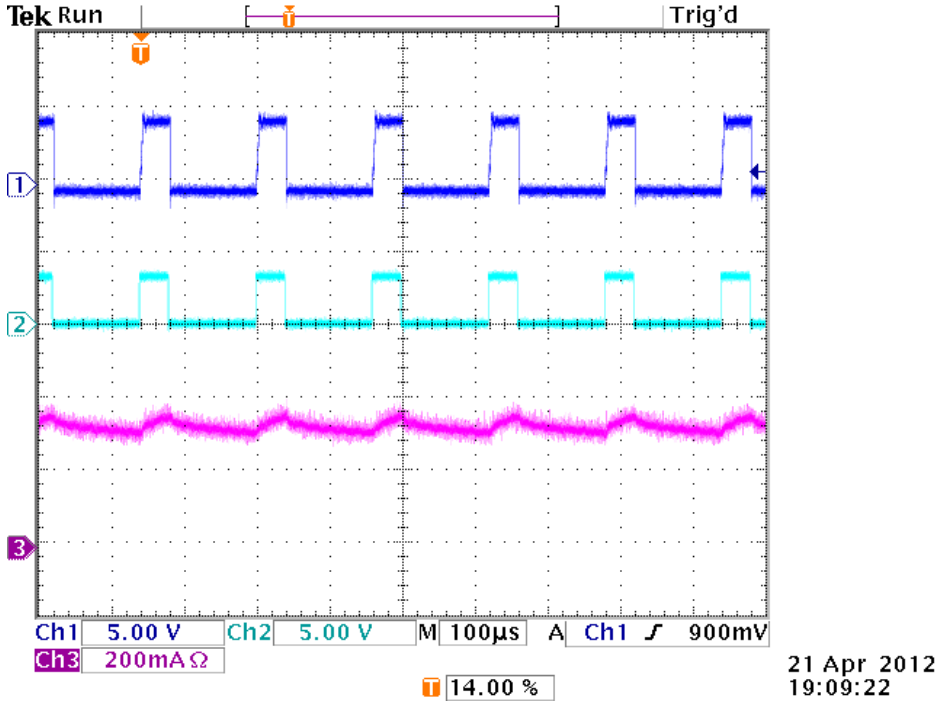

Figure 3.2.1 Drive Circuit Test with 25% PWM Duty Cycle

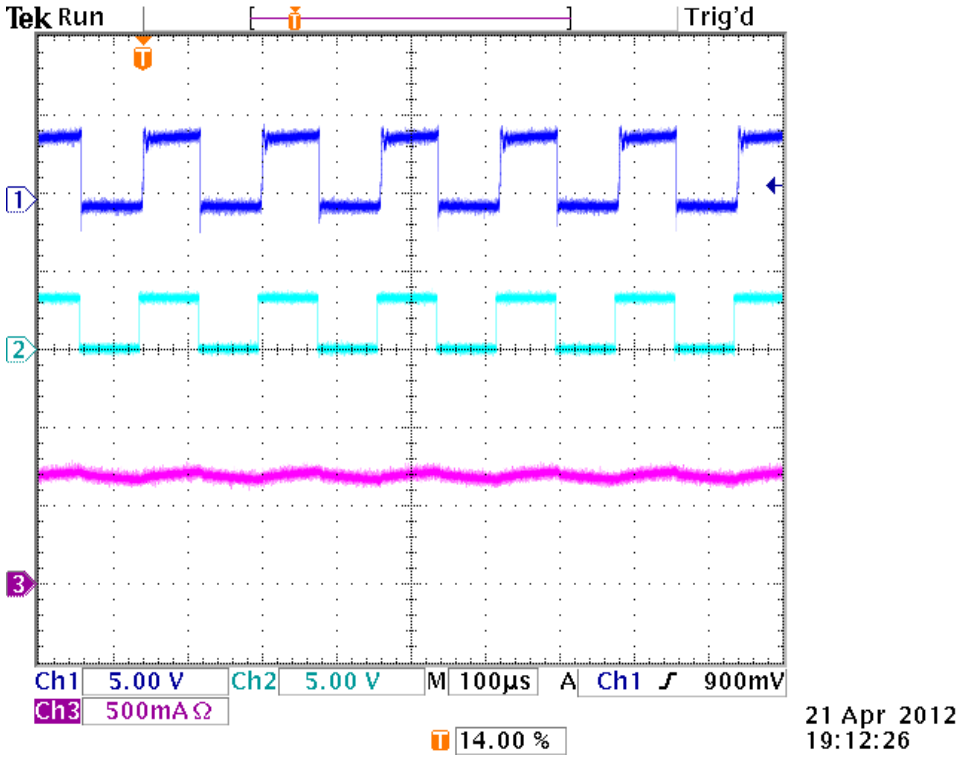

Figure 3.2.2 Drive Circuit Test with 50% PWM Duty Cycle

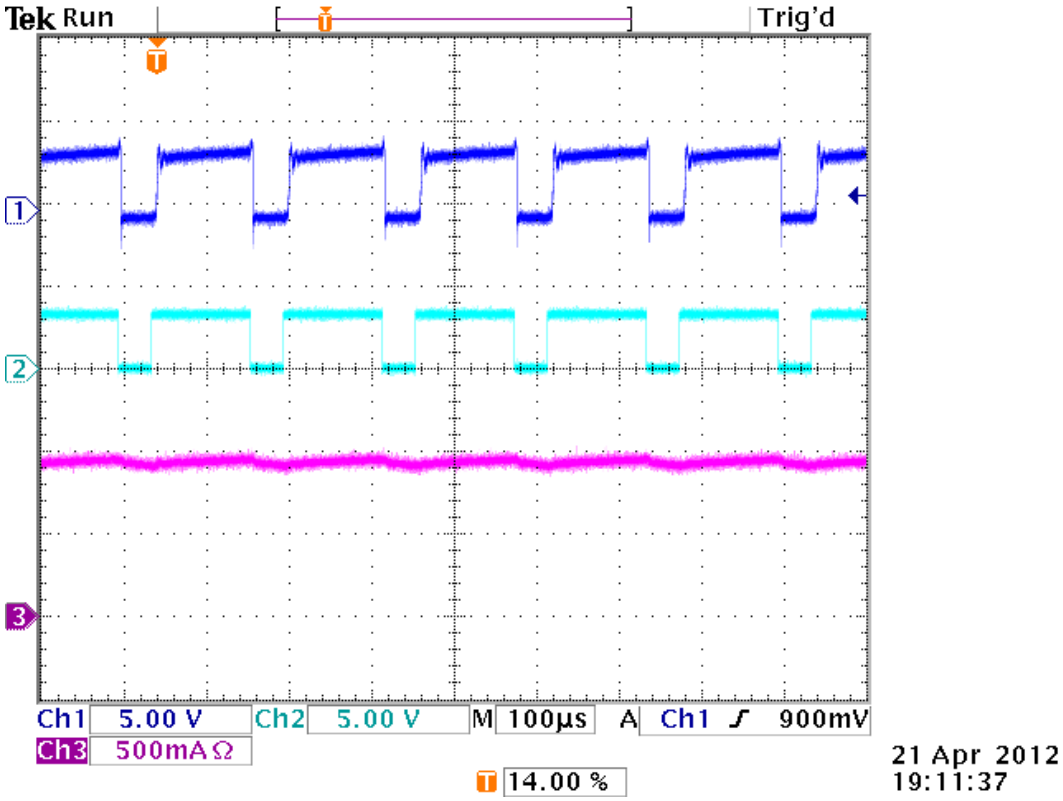

Figure 3.2.3 Drive Circuit Test with 75% PWM Duty Cycle

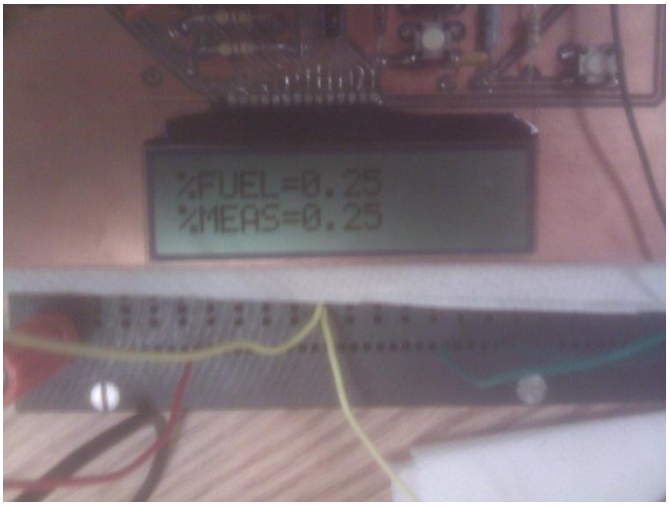

Figure 3.3.17 LCD Display

Figure 3.3.2 Desired 25%

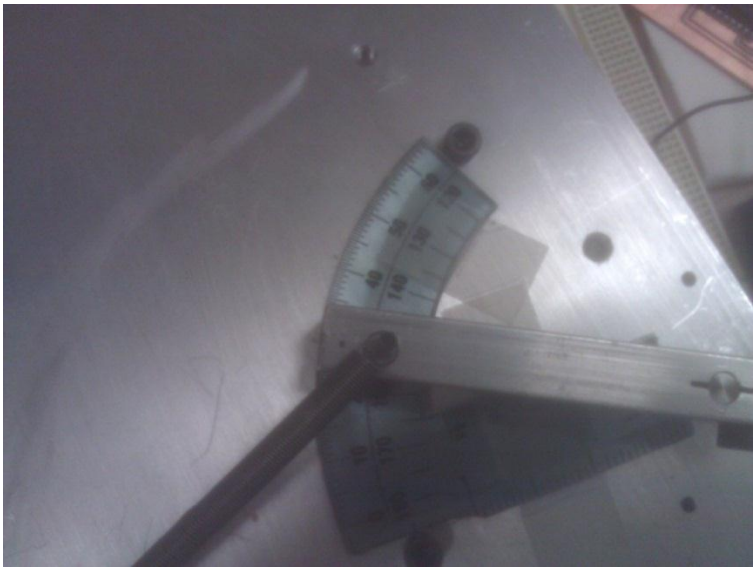

Figure 3.3.3 Desired 50%

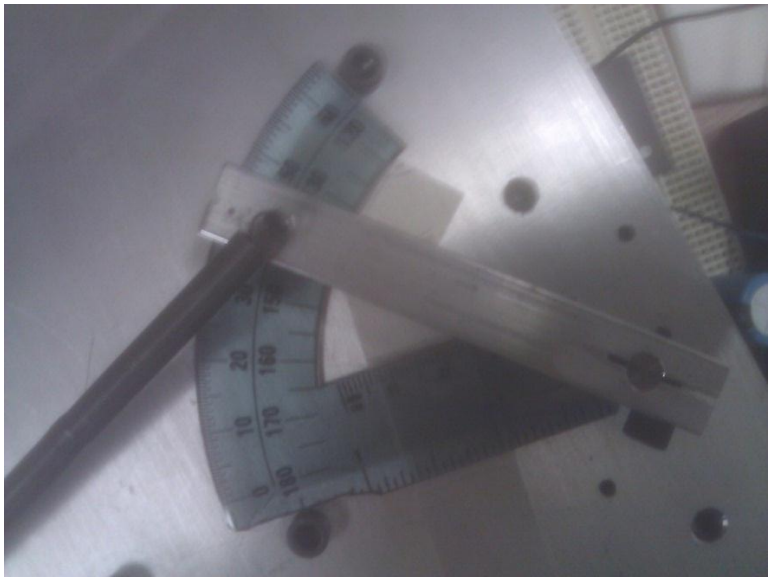

Figure 3.3.4 Desired 75%

Figure 3.3.5 Bulk Voltage Warning

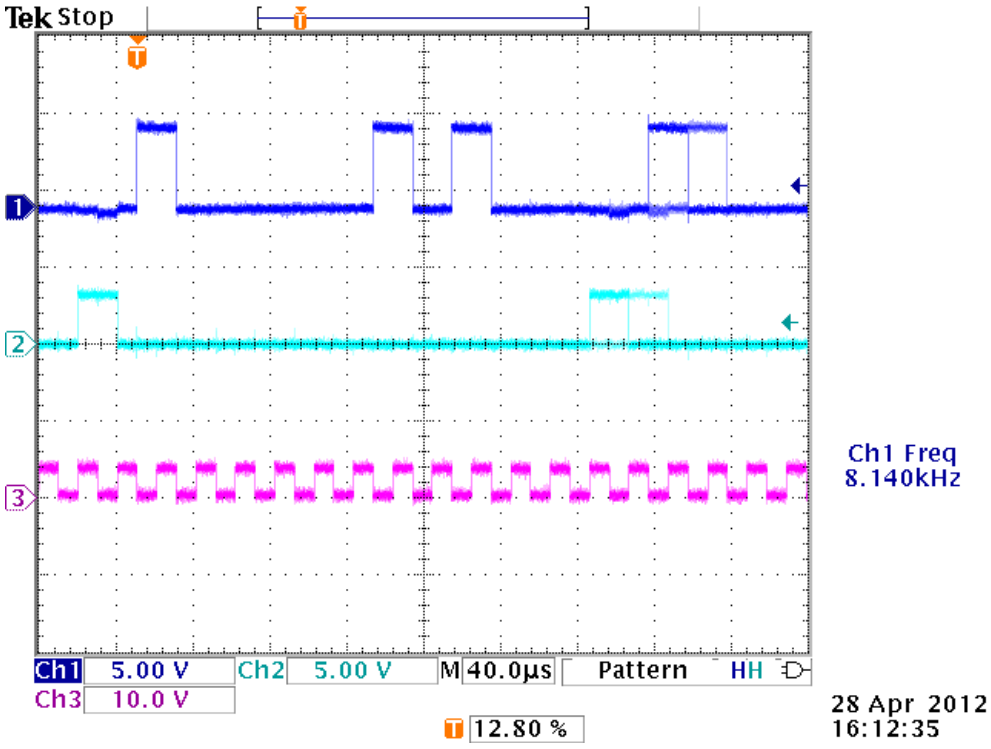

Figure 3.3.6 Arduino SPI MISO Out and PIC Out

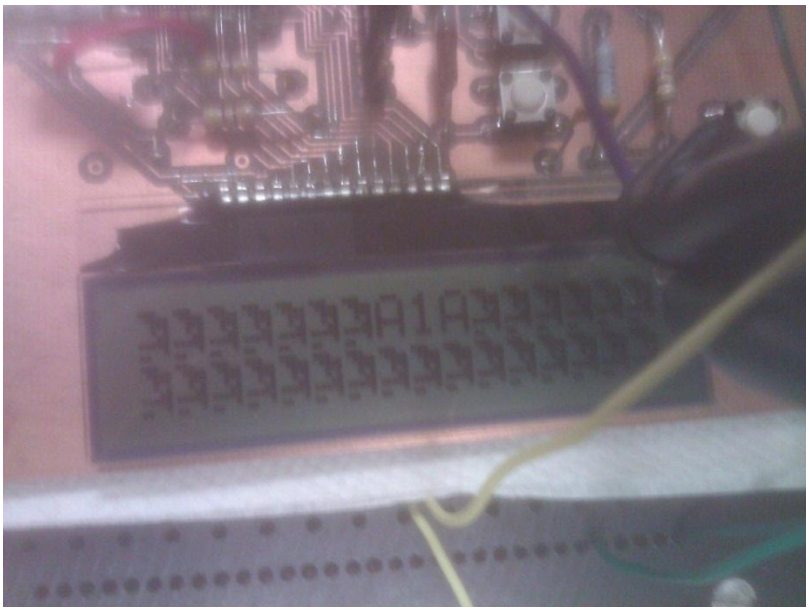

Figure 3.3.7 PIC SPI Receive

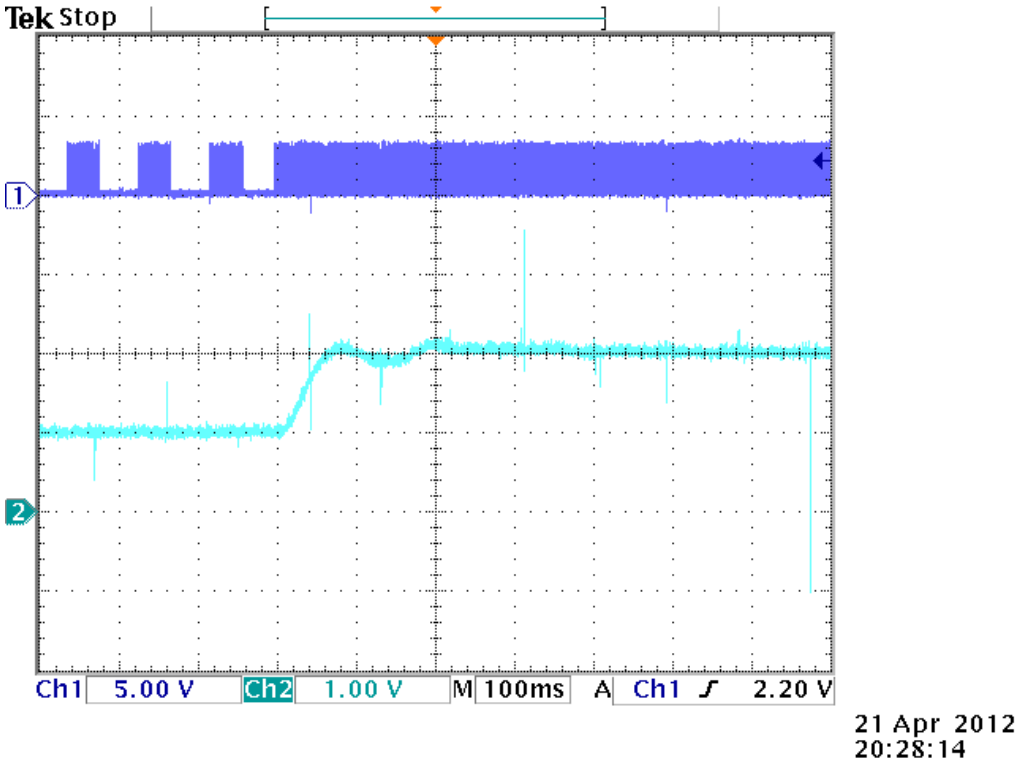

Figure 18 System Transient Response f0 - 44 Degree Command