Chalk Robot

By

Neil Christanto
Leonard Lim
Enyu Luo

Final Report for ECE 445, Senior Design, Spring 2013

TA: Mustafa Mukadam

01 May 2013

Project No. 12

Abstract

We designed an autonomous chalk robot to draw the outline of a simple BMP-formatted image on a sidewalk. This is an idea proposed by MIT Lincoln Lab. After the image is chosen, a sequence of programs will be executed to create an outline of the image, convert to vectors, and give instructions to the motor control module where to go.

The robot uses a Pandaboard as the main core for the image processing and instructions generation. A motor control board that has an ARM Cortex-M3 processor is responsible for controlling the movement of the motor, utilizing the PID control system. Ultrasonic based position sensing is used to provide feedback to the motor control board about the current position and direction of the robot.

Upon testing, the robot draws the outline correctly even though a few lines do not match the original image completely. That being said, there are still a lot of improvement that can be implemented on this robot.

Table of Contents

1. Introduction

We designed an autonomous chalk robot to draw the outline of a simple BMP-formatted image on a sidewalk. This is an idea proposed by MIT Lincoln Lab. After the image is chosen, a sequence of programs will be executed to create an outline of the image, convert to vectors, and give instructions to the control module where to go. Ultrasonic based position sensing is used to provide feedback to the motor control board about the current position and direction of the robot.

1.1 Statement of Purpose

Currently the only way to draw pictures or words on the floor or sidewalk is to actually draw it by hand. There are a few disadvantages to this method. First of all, it is time consuming and back breaking for a person to crouch down and chalk the floor. Also, the end product often depends largely on the skill and artistic flair of the individual drawing the image. It is also difficult for the average person to replicate images or logos with a great deal of accuracy. The chalk robot aims to automate this process, making chalking the sidewalk a much simpler task.

1.2 Objectives

Goals:

> The goal of this project is to develop a robot system that can draw an image on the ground with chalk. This will be a two-wheeled mobile robot with a chalk mechanism. The robot will be able to choose a preloaded image and move accordingly on the floor to draw the image outline.

Features:

- Position tracking relative to the starting point
- Bidirectional motor control allowing for forward, backward motion, turning-on-the-spot
- Control of the chalk mechanism
- Drawing outline of image file
- Up to 1.6 m x 1.6 m range of drawing

Benefits:

- Users can use our robot to easily print chalk advertisement on the sidewalk
- Image outline can be chalked with high level of similarity

1.3 High Level Project Division

On a high level, the whole project is divided into three main components: image processing, motor control, and position sensing. The high level block diagram is given in figure 1.1. Detailed description of each subproject will be explained under the 'Design' section.

1.3.1 Image Processing

The entire image processing is done on a Pandaboard running Ubuntu. After an image is chosen, a C++ based program will create an outline of the image. Then another program converts the outline into a Scalable Vector Graphics. A python script then parses the SVG file and generates movement instructions to be given to the motor control board.

1.3.2 Motor Control

The motor control functionality utilizes the ARM Cortex-M3 processor. It controls the power to the motors to move the robot to the appropriate coordinate in order to draw the image. With feedback from the quadrature encoders, the board will be able to control the distance and velocity of the robot. The ARM processor will also handle the movement of the chalk armature as well as the display of the Liquid Crystal Display (LCD).

1.3.3 Position Sensing

An ultrasonic based position sensing is used to determine the current position of the robot. This mechanism utilizes the time-of-flight of the ultrasonic pulse from the transmitter to the receiver. With the speed of sound is known and the time-of-flight measured, we can calculate the distance the robot is from each transmitter. Four transmitters will be placed on each corner. Using triangulation, we can determine the x and y coordinates of the robot.

1.3.4 Inter-board Communication

The communication between the Pandaboard and the Motor Control Board uses a UART protocol. Since there is an I/O voltage difference between the Pandaboard and the ARM processor, a voltage conversion circuit is made to support this communication. The feedback from the position sensing mechanism to the Motor Control Board is done via I^2C interface.

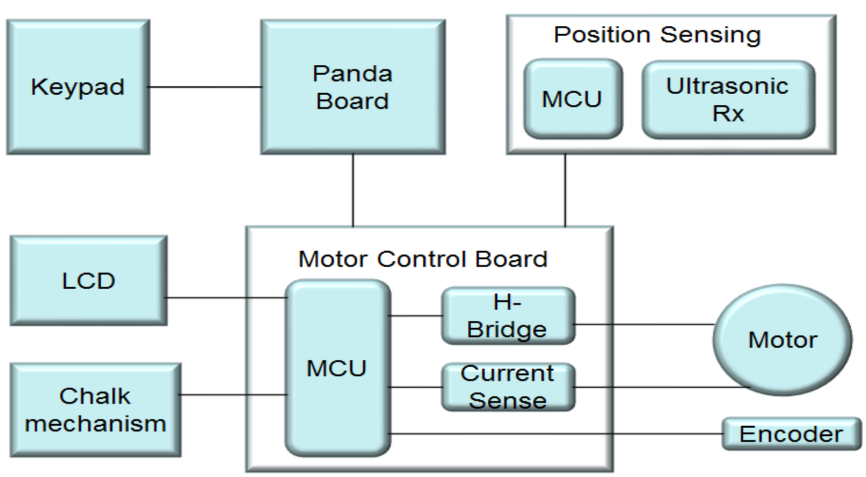


Figure 1.1: High level block diagram

2. Design

The project is divided into three main parts, as mentioned previously. These parts are first developed independently to maximize productivity. After every part is done, a communication protocol is developed to ensure correct and reliable communication between the three subprojects.

2.1 Image Processing

The image processing for our robot is done using programs installed on a Pandaboard running Ubuntu operating system. The input to the image-processing block is a simple image in bitmap format preloaded into the Pandaboard. By going through a few steps outlined in the figure 2.1, the image processing would produce a list of ordered vectors to be sent to the motor control board for use.

2.1.1 Outline

The simple bitmap image is first read and then processed in its raster form. Firstly, we would convert the color image into a greyscale image by averaging their R, G and B channels to obtain a single grayscale value between 0 and 255 (0 corresponds to black and 255 corresponds to white). Then, we would use K-means to determine the threshold to which we would split the greyscale pixels into black and white. With the black and white image, we would then proceed to generate the outline by comparing each pixel to its neighbor. An outline would then be formed on the edge of the black-white space. The K-means and outlining flowcharts are given in Appendix B.

2.1.2 Vectorization

The raster black outline image would then be put into another program, where it would be converted from a bitmap to a Scalable Vector Graphics (SVG) format. This is accomplished through the use of Potrace, which is a free program distributed online. The individual vectors in the .svg file are then read, interpreted and converted into the right form for transmission to the Motor Control Board.

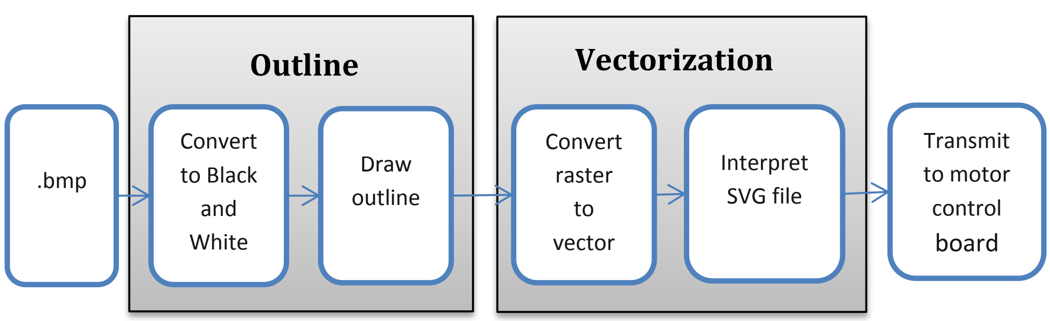


Figure 2.1: Flowchart for image processing

2.2 Motor Control

The main function of the Motor Control Board is to control the power to each motor in order to move the robot to the appropriate x-y location. It consists of a microcontroller, which sends PWM to the H-bridge driver. The H-bridge then determines the voltage applied to the motor. Current sensing is done to either control or limit the current. Feedback from separate encoders is used for velocity control of each individual motor. The microcontroller also receives position and direction information from the position sensing network to compensate for any error arising from the encoders.

2.2.1 H-bridge

The H-Bridge is the driving circuit to the motors. An integrated circuit LM18200T is used here and can handle 3 A of continuous current. The motor stall current at 18 V is 1.5 A. Thus, a H-bridge of 3 A is sufficient in driving the motor. The IC also comes with its own current sensing and temperature flag integrated within. The H-Bridge draws current from a separate battery that supplies 11.1 V. The ground is separated by the digital logic ground via a 100 nH inductor.

2.2.2 Voltage Regulators

We used two LM2596 switching voltage regulators. One of them supplies 3.3 V and the other supplies 5 V. The 5 V regulator is set up with the appropriate values of inductance and capacitance so as to be able to supply an output current of 2 A. The 5 V voltage regulator supplies power to both the Pandaboard and the position sensing receiver board.

2.2.3 Microcontroller

The microcontroller used in this board is TI's Stellaris LM3S9D92. It has an ARM Cortex M3 core. It has eight PWM outputs, two QEI along with the standard communications, ADCs and other peripherals which makes it suitable for motor control of two separate wheels that this robot needs.

2.2.4 Chalk Control

The chalk mechanism consists of two servo motors and a lever. The chalk goes to one end of the lever, while it is pivoted at the other end. One servo grips the chalk, while the other lifts the lever. When drawing is required, the servo releases the lever to the floor, allowing gravity to press the chalk on the drawing surface. When the robot is not drawing, the lever is lifted up.

2.2.5 Control Algorithm

Each of the wheels has an independent PID controller for velocity. The velocity is sampled at 20 ms intervals. The PID is tuned to $k_p = 3.5$, $k_d = 2.5$, $k_i = 0$. This is tested by a fixed sequence velocity commands, and outputting the instantaneous velocity via UART to a computer for analysis.

For position control, each wheel has a fixed acceleration, maximum velocity, and deceleration sequence. The time and distance where the acceleration stops and for the deceleration to start is calculated such that when the velocity goes back down to zero, the wheel will have covered the desired distance.

For the robot to move in a straight line, both wheels are fed with the same velocity curve. For Turning on the spot, each wheel is fed with an opposite direction with the same velocity magnitude curve. The distance the wheels move will be proportional to the desired turning angle as it moves along the circumference of the turning circle.

Curves that the robot will be drawing are defined by cubic Bezier curves. Each curve is defined by a start point, two control points and an end point. The Bezier equation is rewritten such that there is less multiplications needed for efficient processing on the microcontroller. All operations are performed in fixed point.

$$\mathbf{B}(t) = (1-t)^3\mathbf{P}_0 + 3(1-t)^2t\mathbf{P}_1 + 3(1-t)t^2\mathbf{P}_2 + t^3\mathbf{P}_3 \ , \ t \in [0,1].$$

$$= a\,t^3 + b\,t^2 + c\,t + P_0$$

$$c = 3(P_1 - P_0), \qquad d = 3(P_2 - P_1), \qquad b = d - c, \qquad a = P_3 - P_0 - d$$

The length of the Bezier curve is first calculated by adding the lengths of 32 straight line segments of the curve. Very short lengths will be drawn with 4 line segments. Longer lengths will be drawn with 8, 16 or 32 line segments.

2.2.6 Real Time analysis

For the control systems to work correctly, we analyzed the processing time of the tasks running on the microcontroller and prioritized them accordingly. It is crucial that the tasks meet real time requirements, and do not skip samples. Table 2.1 is a list of tasks and their execution times. It is noted that all of the execution times are only a small fraction of their period. The total processor utilization of the five periodic tasks is only 47%. This is less than the utilization bound for 5 periodic tasks, where $U(5) = 5\left(2^{1/5} - 1\right) = 74\%$. Thus, real time requirements are achieved.

Table 2.1: Task analysis

Tasks	Priority	Period	Execution time
Servo Control	0 (Highest)	1.5 ms	3.5 µs
ADC	1	40.0 µs	2.5 µs
PID Loops	3	20.0 ms	12.0 µs
Communications	5	80.0 ms	5.0 µs
Misc. Tasks	7 (lowest)	0.5 s	<200.0 ms
Main Program	-	-	-

2.3 Position Sensing

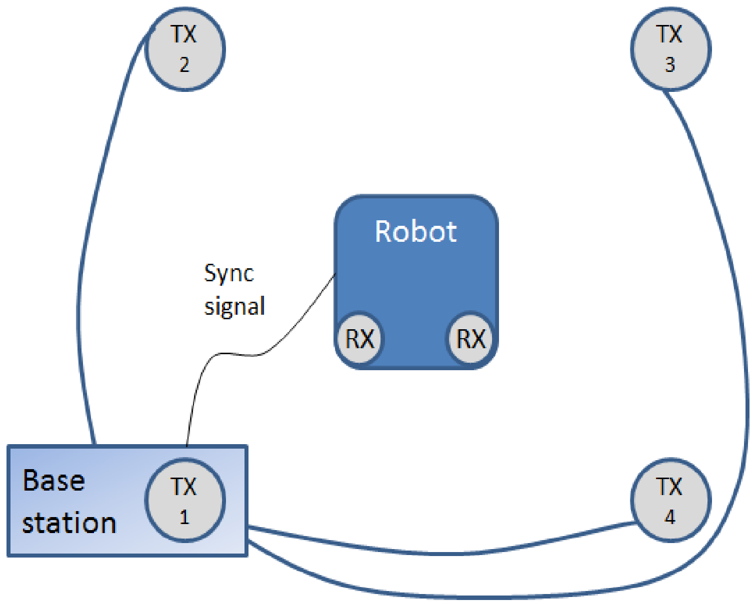


Figure 2.2: Position Sensing overview

The job of the position sensing is to tell the robot its position relative to the placement of the transmitters. The goal is to send the position (and direction) every 80 ms. The time-of-flight approach is going to be used heavily in the distance calculations, so every delay produced by the module itself (microcontroller I/O delay for example) needs to be taken into consideration.

2.3.1 Base Station

The base station's job is to handle the synchronization of all the transmitter and receiver modules. The main component is a microcontroller. An interrupt will trigger periodically to send a reference pulse to the receiver module via a long wire. When the reference pulse is sent, four timers with different expiration will start (each timer corresponds to each transmitter). Since the first transmitter is combined with the base station, the first transmitter will start transmitting at the reference pulse. There are three additional outputs from the microcontroller corresponding to the remaining three timers, and those outputs are hooked to an interrupt input at each transmitter's microcontroller. When each timer expires, it means that it is time for a particular transmitter to send a pulse.

The four timers are configured to have different expiration times set at regular intervals. It is configured this way so that the receiver can determine which timer is sending the pulse. That being said, there is a limit of how short the interval between two consecutive pulses can be. The following equations are used to determine the lower limit of the interval between two consecutive pulses.

$$\text{Speed of sound } (\text{m.s}^{-1}) = 331.3 \times \sqrt{1 + \frac{T}{273.15}}\,;$$

T is the surrounding temperature in Celsius.

For the following equations, we are using the worst case (slowest) speed of sound (at sea level) which is 340 m/s. The transmitters are put forming a square of 2.4 m × 2.4 m because the beam angle is only 80 degree wide (it is configured this way so that the transmitters can cover an area of 2 m by 2 m, even though our drawing area is 1.6 m by 1.6 m). So the worst case travel distance for the ultrasound is from the transmitter in one corner to the other corner where the robot is the furthest. Then the worst case travel time is the worst case distance divided by the worst case speed of sound.

$$\text{Worst case travel time} = \frac{\sqrt{2}\tan(50°) + \sqrt{2}}{340} = 9.12\text{ms}$$

So we assign a 10 ms window for each transmitter to send a pulse. However, we need to take into account the fact that there is going to be a sound reflection. By probing using an oscilloscope, it is found that the longest time it needs for the reflection to be damped down is around 8 ms. So we assign a 20 ms window for each transmitter to send a pulse.

In that window, only one assigned transmitter will send a pulse. Since there are four transmitters, the time required for all transmitters to send a pulse is 80 ms. Therefore, the reference pulse is sent every 80 ms.

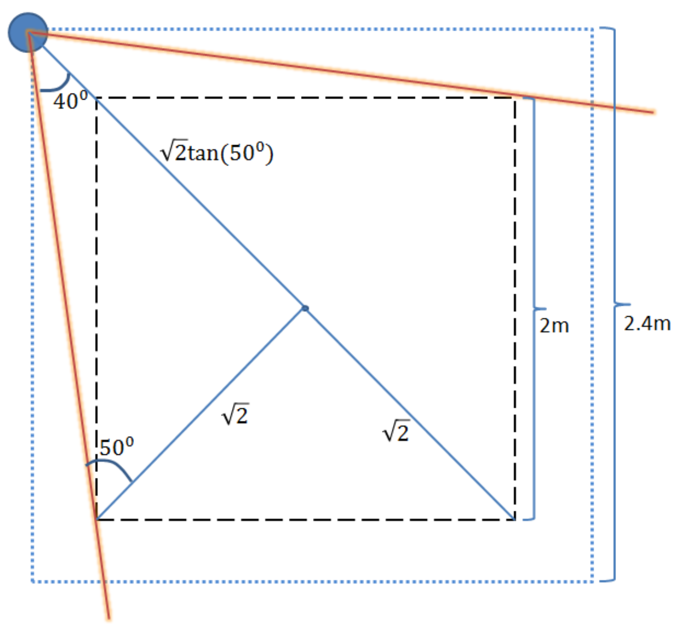


Figure 2.3: Transmitters placement

2.3.2 Ultrasonic Transmitters

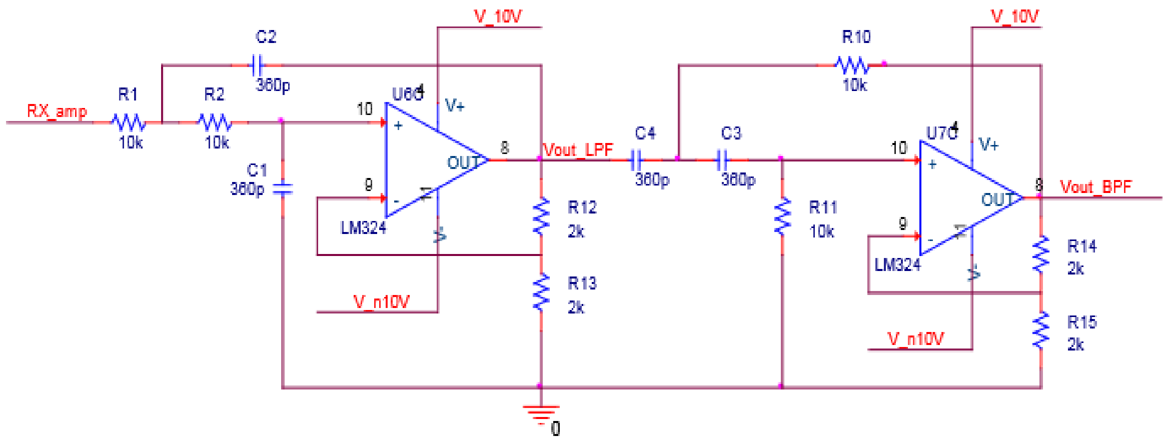
The transmitters will be connected to the base station using long wires. They would first wait for an interrupt to be triggered by a signal from the base station, indicating the start of their transmission window. To deliver a pulse with the strongest power, the ultrasonic transmitters have to be driven by a 40 kHz signal with 20 V peak-to-peak. To realize this, each transmitter has a microcontroller to output a square wave generated by the PWM module. Since the microcontroller can only support a digital logic output between 0 V and 3.3 V, a H-bridge is used, supplied at 10 V to enable a switching between +10 V and -10 V at the output of the bridge, to produce a 20 V peak-to-peak wave.

The square wave is sent only for a limited time to emulate a pulse. In this case, the microcontroller will only send two periods of square wave to the transmitter. Other than the PWM, the microcontroller will also make use of another timer to count up to what is equivalent to two periods of 40 kHz square wave, and after that timer expires, the microcontroller stops sending the wave.

2.3.3 Receiver Module

This module presents the biggest challenge in the position sensing mechanism. The main components consist of a microcontroller and two ultrasonic receivers. For the basic position sensing, we would require a single ultrasonic receiver. However, we are using two receivers so that we can also do direction sensing. The two receivers are placed as wide apart as physically possible so as to minimize the error in calculating out direction. The algorithm to do direction sensing will be further explained in section 2.3.4.

The ultrasonic receiver is most sensitive at 40 kHz. Hence, a second-order band pass filter with a gain of four is made by cascading a second-order low pass filter and a second-order high pass filter with cutoff frequency at 40 kHz, each with a gain of two. The schematic is shown in figure 2.4.



RX_amp = input from the RX, amplified
V_10V = +10V supply
V_n10V = -10V supply
Vout_BPF = Band pass filter output, connected to another amplifier

Figure 2.4: Band pass filter schematics (40 kHz center frequency)

Before and after the band pass filter, we added high gain amplifiers. The total gain of the two amplifiers plus the band pass filter is chosen to be high enough so that a very weak signal will be at least 800 mV peak-to-peak (400 mV amplitude). The noise is less than 250 mV peak-to-peak (125 mV amplitude).

The last stage is a Schmitt trigger that has a band of 300 mV with 300 mV upper threshold and 0 V lower threshold. The op-amp is supplied at 10 V for Vcc and ground for -Vcc so that the output range is between 0 V and 10 V (less than 10 V in reality due to saturation). The Schmitt trigger calculation is explained in Appendix C. Finally the output of the Schmitt trigger is regulated by a 3.3 V Zener diode so that the final output that is hooked to the input of the microcontroller to be processed is either 0 V or 3.3 V.

After the reference pulse is received, a timer in the microcontroller will start counting. The microcontroller expects four incoming pulses before it receives the next reference pulse. Since we know the time-of-flight of the pulses and the order of which the beacons transmit the pulse, we can determine the position of the robot relative to each beacon. The speed of sound depends on the surrounding temperature, but variation is minimal.

The calculation for the triangulation is shown in Appendix C. The equations show that two x and y values can be calculated in one period of the pulses transmission. The average value is taken, producing a more reliable result. This also reduces the error since the two values that we get are from opposite transmitters. Hence, even when the speed of sound varies and the two readings of each x and y coordinate will be further apart, the average will not vary that much.

2.3.4 Direction Sensing

The role of the direction sensing is to give information about where the robot is facing. It is integrated within the position sensing mechanism since it uses the same ultrasonic receivers in the position sensing mechanism. Hence, the performance of this block relies heavily on the accuracy of the position sensing mechanism.

After the positions of the two ultrasonic receivers are known, the direction of the robot can be determined using a simple arc tangent formula. This is possible only if we know which sensor is placed where (to the left or to the right of the robot), which can be done by simply assigning the receivers to particular input pins on the microcontroller.

Finally, the positional and directional data would be sent to the Motor Control Board together. Therefore, every time the receiver module communicates with the motor control board, it gives both position of the chalk and the direction of the robot. The communication is done using I^2C protocol.

2.4 Inter-board Communications

The image processing block will communicate with the motor control board using UART. The vectors will be sent from the Pandaboard to the microcontroller one at a time. Handshaking will be used to determine if the microcontroller is done with the previous vector and ready to receive the next one. A simple circuit is used to convert between the 1.8 V logic of the Pandaboard and the 3.3/5 V logic used by the Motor Control Board. A schematic of this is shown in Appendix B.

The position sensing block will communicate with the Motor Control Board using I^2C. After each stroke is drawn, the Motor Control Board will request the position from the position sensing block, and the transfer of position and direction will begin. The Motor Control Board is configured as a master receiver and the position sensing board is configured as a slave transmitter.

3. Design Verification

The detailed list of requirement and verification is attached at the end of the report as Appendix A. The following sections are the verification procedures of some important requirements. The verification results are attached as Appendix D.

3.1 Image Processing

The ultimate requirement for the image processing is to send a list of vectors to the Motor Control Board. Also, it is important that the list of vectors sent accurately represents the input image. To check the accuracy of the output vectors, we would plot the output vectors on python and compare them to the input image. The python plot should have a similar shape to the input image.

To check for successful UART communication with the Motor Control Board, we would program the LCD on the Motor Control Board to output according to the data sent from the Pandaboard.

3.2 Motor Control

The motors are tested for functionality by applying a small voltage across the terminals and making sure that the motors are able to turn in both directions. The H-Bridge is tested using a 12 Ω, 20 W resistor as the load, and varying the PWM, supply voltage, and currents across the load. The real time task execution time is obtained by programming the tasks to toggle output pins and probed using an oscilloscope. The velocity PID controller is tested by programming a fixed velocity sequence and have the microcontroller UART the instantaneous velocity out and plotted in Matlab. The plots are attached in Appendix D. Position accuracy is obtained by programming the robot to move in a straight line with a distance of four feet. We would then measure the error between the desired and actual four feet mark.

3.3 Position Sensing

The requirement for the base station is to instruct the transmitters to send a pulse at a steady interval of 20 ms between consecutive transmitters. To verify this, we would program the microcontroller so that the timer expiration that is connected to the transmitter causes a toggle in the output pin of the MCU. We would then probe each output pin corresponding to each timer on the oscilloscope.

For the transmitter, we would require it to transmit a 40 kHz signal driven with 20 Vpp square wave. To emulate a pulse, two periods of the square wave is transmitted. To verify this, program the microcontroller to output two PWM signals that corresponds to 40 kHz signal and two periods. This PWM signal needs to be 180 degree out of phase of each other because it will drive an H-bridge supplied at 10 V to produce 20 Vpp to drive the transmitter.

Testing the receiver module is a little bit involved. But the main goal is to receive an interrupt once for every pulse that the transmitter transmits. When there is an ultrasonic pulse being transmitted, the receiver will receive multiple pulses due to reflections off the environment. To verify this, make the calculation in the interrupt handler toggle a particular output pin and probe this output pin on the oscilloscope. Every time there is a TX transmission, this output should toggle only once.

4. Costs

4.1 Parts

Motor and Chalk Control

Part name	Part number	Qty	Unit Cost	Total Cost
Microcontroller Evaluation Kit	EKS-LM3S9D92	1	$107.66	$107.66
Voltage Regulator	LM2576	2	$2.91	$5.82
Geared DC Motor	GM9434G807	2	$250.00	$500.00
Encoder	E6B2-CWZ3E	2	$39.95	$79.90
Op amp	LM324	1	$0.41	$0.41
H-bridge	LMD18200	2	$8.10	$16.20
Servo	HS-311	1	$7.99	$7.99
Micro Servo	HXT900	1	$2.69	$2.69
11.1V 2100mAH Battery	VNR1577	2	$19.99	$39.98
LiPo Battery Charger	---	1	$49.99	$49.99
			Total:	$810.64

Position and Direction Sensing

Part name	Part number	Qty	Unit Cost	Total Cost
Ultrasonic Transmitter	MA40S4S	4	$7.17	$28.68
Ultrasonic Receiver	MA40S4R	2	$7.38	$14.76
Microcontroller	LPC1114fn28	5	$2.55	$12.75
Op Amp	MC33179	2	$0.42	$0.84
Op Amp – single ended	MC34074	2	$0.60	$1.20
5V to ±10V Converter	MAX680	2	$4.82	$9.64
5V Linear Regulator	UA7805	4	$0.27	$1.08
3.3V Linear Regulator	UA78M33	4	$0.27	$1.08
3.3V Zener Diode	1N4728A	2	$0.20	$0.40
Half H-bridge Driver	SN754410	4	$2.55	$10.20
11.1V 2100mAH Battery	VNR1577	2	$19.99	$39.98
			Total:	$120.61

Image Processing and User Interface

Part name	Part number	Qty	Unit Cost	Total Cost
Alphanumeric LCD	HD44780	1	$17.95	$17.95
Numeric Keypad	---	1	$6.47	$6.47
Pandaboard A3	750-2152-021(A)	1	$180.00	$180.00
SD Card 8GB	B000OF2F36	1	$4.98	$4.98
			Total:	$209.40

4.2 Labor

Name	Hourly Rate	Hours/week	Weeks	Total*
Neil Christanto	$40	14	12	$16,800
Leonard Lim	$40	14	12	$16,800
Enyu Luo	$40	14	12	$16,800
Total				**$50,400**

*Total Labor (per person) = Hourly Rate X Total Hours X 2.5

4.3 Grand Total

Total Parts	Total Labor	Grand Total
$1,140.65	$50,400	$51,540.65

5. Conclusion

5.1 Accomplishments

Through this project, we have successfully implemented a functional autonomous chalk robot. The robot can take in a simple bmp image, locate its position accurately in the workspace and move accordingly to draw the desired image. We are able to get position sensing accuracy of ±3 cm and we can draw line segments up to an accuracy of ±1 cm.

5.2 Uncertainties

Despite the working project, one uncertainty lies in the position sensing part of the project. Ultrasonic waves and time of flight calculations are used to determine the position. However, the speed of sound changes as the surrounding temperature changes. While the variation is not a lot, re-calibration may be needed if we want an accurate reading in a new environment.

5.2 Ethical considerations

In line with the IEEE Code of Ethics,

1. With the building of our robot, we will be mindful to take responsibility and make decisions to ensure the safety and welfare of the public. We will disclose promptly factors that, when operating the robot, might endanger the public or the environment. At no point will we allow the robot to operate without human supervision.

    1. *to 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;*

    9. *to avoid injuring others, their property, reputation, or employment by false or malicious action;*

2. We will be honest and realistic in reporting our work and will not falsify or fabricate results.

    3. *to be honest and realistic in stating claims or estimates based on available data;*

3. Through this project, we would learn various real-world applications and technologies. These skills serve to improve our technical competence.

    6. *to maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations;*

4. We will be critical of technical work, both of our own and of others. Also, we will always cite all work that we use, giving credit where credit is due.

    7. *to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others;*

5. We will assist teammates and fellow students in their professional development through knowledge sharing and support them in following the IEEE code of ethics.

   10. *to assist colleagues and co-workers in their professional development and to support them in following this code of ethics.*

5.3 Future work

While our robot is perfectly functional, it is definitely not the most polished of a product. There are many ways in which it can be further improved and worked on.

One possible direction is to improve on the user interface. We could leverage on smart devices like smart phones and tablets by designing an app and allowing the user to control the robot via their devices. The user could either select a pre-drawn image, or do a simple sketch on their touchscreen. The image drawn can then be sent to the robot for image processing and control. We could also make the input more flexible, allowing the user to vary the size and dimension of the workspace.

On the position sensing end, one possibility is to add ultrasonic receivers on each of the beacons and allow for automatic calibration of the position sensing network. This would make setting up much more convenient and could then open the possibility of having workspaces of varying dimensions. Another auto-calibration algorithm can also be developed so that we don't have to use reference pulse. As we have four transmitters on each corner, we have redundancies. The robot can then observe the incoming pulses and determine the necessary time offset for each transmitter.

Another improvement would be to improve the control algorithm of the motors. Instead of using line segments to approximate the curve, we would directly control the motors using a more continuous scheme. This would create an output curve that is much smoother than what we currently have. Also, we would like to integrate position sensing on the fly so that the robot can correct itself, even when it is drawing.

References

[1] Bjerknes, Jan D., Wenguo Liu, Alan FT Winfield, and Chris Melhuish. "Low Cost Ultrasonic Positioning System for Mobile Robots." (n.d.): n. pag. Web. <http://www.ias.uwe.ac.uk/~a-winfie/Bjerknes_etal_TAROS07.pdf>.

[2] Shima, Jim. "DSP Trick: Fixed-Point Atan2." *DspGuru.com*. N.p., 23 Apr. 1999. Web. 26 Feb. 2013. <http://www.dspguru.com/dsp/tricks/fixed-point-atan2-with-self-normalization>.

[3] "NXP LPC111X Datasheet." N.p., 20 Feb. 2013. Web. <http://www.nxp.com/documents/data_sheet/LPC111X.pdf>.

[4] "NXP LPC111X User Manual." N.p.: n.p., 24 Sept. 2012. PDF.

[5] "Scalable Vector Graphics (SVG) 1.1 (Second Edition)." W3C. N.p., 16 Aug. 2011. Web. <http://www.w3.org/TR/SVG>.

[6] "LM2576/LM2576HV Series SIMPLE SWITCHER® 3A Step-Down Voltage Regulator." Texas Instruments, Nov. 2004. Web. 27 Feb. 2013. <http://www.ti.com/lit/ds/symlink/lm2576.pdf>.

[7] "LMD18200 3A, 55V H-Bridge." Texas Instruments, May 2004. Web. 27 Feb. 2013. <http://www.ti.com/lit/ds/symlink/lmd18200.pdf>.

[8] "Stellaris® LM3S9D92 Evaluation Kit User's Manual." Texas Instruments, 5 July 2011. Web. 27 Feb. 2013. <http://www.ti.com/lit/ug/spmu174/spmu174.pdf>.

[9] "Stellaris® LM3S9D92 Microcontroller." Texas Instruments, 22 Jan. 2012. Web. 27 Feb. 2013. <http://www.ti.com/lit/ds/symlink/lm3s9d92.pdf>.

[10] "LM124/LM224/LM324/LM2902 Low Power Quad Operational Amplifiers." Texas Instruments, May 2004. Web. 27 Feb. 2013. <http://www.ti.com/lit/ds/symlink/lm324-n.pdf>.

[11] "OMAP™ 4 PandaBoard System Reference Manual." Pandaboard.org. N.p., 29 Nov. 2010. Web. <http://pandaboard.org/sites/default/files/board_reference/pandaboard-a/panda-a-manual.pdf>.

[12] "HD44780U (LCD-II)(Dot Matrix Liquid Crystal Display Controller/Driver)." N.p., n.d. Web. 15 Feb. 2013. <http://www.sparkfun.com/datasheets/LCD/HD44780.pdf>.

[13] "IEEE Code of Ethics." IEEE. N.p., n.d. Web. 27 Feb. 2013. <http://www.ieee.org/about/corporate/governance/p7-8.html>.

Appendix A Requirement and Verification Table

Motor Control

Requirement	**Verification**	**Status**
H-Bridge		
1. Transistors in the H-bridge turn on with a supply voltage between 12 V to 24 V.	1. Set the PWM pin to high and vary the supply voltage from 12 V to 24 V with a 50 Ω. Use a digital multi-meter to ensure that the output voltage is more than supplied voltage minus 2 V.	Verified
2. Maximum steady state current of 2 A per motor channel.	2. Set the PWM pin to high, and use 10 Ω load across the output of the H-bridge. Increase the supply voltage until the voltage across the load reaches 20 V. Use a digital multi-meter to confirm this voltage drop.	Verified
3. Peak instantaneous current of 3 A per motor channel.	3. Connect 5 Ω load across the output of the H-bridge. Use a function generator to generate pulses of 16 V for 100 μs every 10 ms (square wave of 100 Hz with 1% duty cycle). Use the oscilloscope to measure the voltage across the load. The voltage reading should be 15±0.2 V.	Verified
Encoder		
1. Encoder output channel A and B are 90±10 degrees out of phase from each other. For clockwise rotation, one channel should lead while for counter-clockwise rotation, the other channel should lead.	1. Connect both output channels to an oscilloscope. Rotate the encoder in a clockwise direction. Measure the phase difference between the two channels. Repeat for counter-clockwise rotation. Make sure the leading of the channel is correct according to the rotation of the encoder.	Verified

DC Motor		
1. Motor can turn in both directions.	1. Connect the motor input pins to a DC power supply. Output 5 V from the power supply. The wheel connected to the shaft should rotate in one direction. Now switch the connection of Vcc and ground. The wheel should rotate in the opposite direction.	Verified
Control Algorithm		
2.a) Able to control velocity from 0 to 512 encoder pulses per second. b) Velocity control within ±10% of desired velocity at steady state.	2.a) Using microcontroller, set a fixed velocity to make the wheel rotate. Then by reading from the encoder, make the microcontroller output the instantaneous velocity every 10 ms via UART (check using hyperterminal) or DAC (check using oscilloscope). b) Using microcontroller, send fixed velocities of 150, 175, 200, 225, 250, 275, 300 pulses per second and measure the actual velocity. The measurement is done using the same method: read from encoder and give the output via UART or DAC.	Verified Verified
3. Position Control to an error of ±1 cm.	3. Use microcontroller to tell the robot to go to a certain position (preprogram the coordinate). The final position has to be within ±1 cm of the exact coordinate.	Verified

Chalk Control

Requirement	Verification	Status
1. When writing, the part of the chalk touching the ground should be within ±1 cm of the middle of the two wheels.	1. Put some kind of mark on the wheels (ink is possible) so that they will leave mark as the robot moves. Send commands to draw some squiggly lines over 1 to 2 m. The chalk line should be exactly in between the two ink line.	Verified
2. Servo that controls the chalk armature can be directed to 3 different positions (let go of the lever completely, lift the lever completely, and lift the lever to 1±0.2 inch off the ground).	2. Using function generator, output a PWM to control the servo, directed to the three different positions. When the lever is let go completely, the chalk should be in contact with the ground. When the lever is lifted all the way up, the last contact switch is activated. For the last command, measure the distance of the lever from the ground with a ruler. It should be at 1±0.2 inch off the ground. After the tests are completed, output a PWM to the servo using the microcontroller. Then repeat the same test.	Verified

Position Sensing

Requirement	Verification	Status
Base Station		
1. Interrupt is triggered every 80±0.1 ms (use MCU timer) to send a reference signal.	1. The interrupt service routine toggles a particular GPIO on the MCU. Probe this GPIO pin using the oscilloscope and make sure that it toggles every 80±0.1 ms.	Verified
2. The three timers (for TX2, TX3, and TX4) have expiration at 20 ms, 40 ms, and 60 ms after the reference signal is sent (with ±0.01 ms tolerance).	2. Matched output 2 should toggle 20 ms after the reference pulse. Output 3 should toggle 20 ms after output 2 toggles, output 4 should toggle 20 ms after output 3 toggles.	Verified
3. After the last timer expires, none of the timers should expire again. The timers are restarted at the next reference pulse.	3. After matched output 4 toggles, nothing should happen. Probe in the scope for all of the MCU's matched output to make sure that they don't toggle again after output 4 toggles.	Verified

Ultrasonic Transmitter		
1.a) Generate a 40±1 kHz square wave right after a signal from the base station is received.	1.a) The PWM timer with 40 kHz and 50% duty cycle should start right after the MCU receives a signal from the base station as an interrupt (take the toggling of the matched output from the base station as the source of interrupt). Probe the output of the MCU PWM on the oscilloscope and make sure that it is a square wave of 40 kHz and 50% duty cycle.	Verified
b) The 40 kHz square wave needs to be at most 20 V.	b) Using 10 V as a supply voltage, drive the H-bridge with the PWM output from the MCU and probe the output at the oscilloscope. The output should alternate between at most +10 V and -10 V, producing a 20 V peak-to-peak voltage. Make sure the frequency is still 40±1 kHz.	Verified
2. Stop sending the wave after two periods.	2. Have a second timer that starts at the same time the PWM timer starts. Make this timer's expiration equivalent to two periods of 40 kHz wave. Once this timer expires, disable the H-bridge. Use oscilloscope to probe the output of the H-bridge and make sure that the wave stops after 2 periods.	Verified
3. Transmitter works when driven by the MCU.	3. Hook the transmitter to the output of the H-bridge. Drive the transmitter with 40 kHz 20 V peak-to-peak square wave. Place the receiver very close to the transmitter and probe the receiver using oscilloscope. Make sure the receiver produces a noticeable steady sine wave at around 40 kHz.	Verified

Receiver Module - Bandpass Filter		
1. Bandpass filter with a high gain centered at 40±1 kHz to filter out noise and amplified the received signal at 40 kHz.	1. Make a signal generator produce a low voltage to simulate the receiver (around 20 mV peak-to-peak) and probe the output of the filter on the oscilloscope. Vary the frequency from 20 kHz to 60 kHz. The output should be the strongest at around 40 kHz.	Verified
2. After the bandpass filter, the signal that comes out needs to be at least 800 mV peak-to-peak.	2. Make the transmitter transmit a 20 V peak-to-peak signal at 40 kHz driven by the function generator and place the receiver far from the transmitter. To test the worst case, place them 4 m apart, then probe the output of the bandpass filter. The output should be at least 800 mV peak-to-peak.	Verified
3. When there is no ultrasound transmission (the transmitters are not active), the output of the bandpass filter needs to be at most 250 mV peak-to-peak.	3. Leave the receiver powered up with no transmitters transmitting anything. Probe the output of the bandpass filter on the oscilloscope. The noise needs to be at most 250 mV peak-to-peak.	Verified

Receiver Module - Schmitt Trigger					
1.a)	Weakest signal should produce an alternating GND to Vcc output.	1.a)	The weakest signal is at least 400 mV in amplitude. This should be enough to trigger a Vcc output. The lower threshold is 0 V, so when the signal goes below zero the output will be driven to -Vcc supply of the op amp (we choose a single supply op amp, so -Vcc supply is connected to ground). Probe the output using oscilloscope.	Verified
b)	Noise should produce a GND output.	b)	The output of the bandpass filter when there is nothing being transmitted has at most 125 mV amplitude, so this should not make the output of the Schmitt trigger to go to Vcc; the ouput should stay at GND level. Probe the output using oscilloscope.	Verified
2.	Input to the microcontroller needs to be in a range of -0.5 V to +5.5 V. Input is considered logic high if the voltage is at least 0.7 Vdd and the input is considered logic low if the voltage is at most 0.3 Vdd, with Vdd = 3.3 V.	2.	Put a zener diode with a break down voltage between 3.3 V and 5.5 V after the Schmitt trigger. Probe that terminal using an oscilloscope, the peak voltage should be between 2.31 V (0.7 Vdd) and 5.5 V, the low signal should be between -0.5 V to 0.99 V (0.3 Vdd).	Verified
Receiver Module - MCU					
1.	Have one timer that starts after the MCU received the reference signal from the base station. This timer should be able to count up to 80 ms equivalent amount of time to avoid timer overflow. The time-of-flight calculation will be based on this timer.	1.	Taking the fact that the clock frequency of the MCU is 48 MHz, a 32-bit timer will be able to count up to 80 ms equivalent amount of time.	Verified

2.a) Trigger an interrupt every time there is a pulse coming in. b) The calculation in the interrupt handler needs to be triggered only once per TX transmission.	2.a) Connect the clamped output of the Schmitt trigger to a GPIO pin on the MCU and configure that pin to be a source of interrupt. The interrupt handler is invoked on the rising edge input of the GPIO pin. b) To make sure that the interrupt is only triggered once per pulse, have a timer that starts counting after receiving the first interrupt, and make its expiration equivalent to around 10 ms. This interrupt handler then disables the reception of the next interrupts. The interrupt will be re-enabled and the timer will stop counting when the timer expires.	Verified Verified
3. Calculate the position of the robot with a maximum error of 5 cm.	3. Using oscilloscope, calculate the delay it takes from the beginning of the ultrasonic transmission to the time when the voltage at the GPIO input pin reach 2.31 V. This delay should be taken into account when we calculate the time-of-flight. Using the time-of-flight information from each transmitter to calculate the x and y coordinate. The end result should be within 5 cm of the actual measured position.	Verified

Direction Sensing

Requirement	Verification	Status
1. Calculate the direction of the robot relative to the placement of all four beacons (give each beacon an ID number for example) within 10 degree. Full rotation is 360 degree.	1. We have two ultrasonic receivers with the chalk in the middle. Given the reading from the two receivers, determine the angle using trigonometric (arc tangent) knowing which receiver is which (give each receiver an ID number as well). To calculate the arc tangent, use the atan2 math function. Move the robot around and see the direction this function outputs. The result should be within 10 degree of the actual result (measured with a real compass).	Verified

Image Processing

Requirement	Verification	Status
Outlining		
1. Black and white image generated from the thresholding should retain the general shape of the original image.	1. Using a simple test image (basic shapes), compare the print out of the black and white image to a print out of the original image. This is going to be a subjective judgment, but at least the outline should be noticeably matched.	Verified
2. Outline should trace the thresholded black and white image.	2. Using a simple test image, compare the print out of the outline to the black and white image. Again, this is going to be a subjective judgment, but the outline should trace the black and white image well. Ensure there are no bumps on the outline.	Verified

Vectorization		
1. Vectored image should look like the outline.	1. Using a simple test image, compare the vectorized outline with the raster outline. For a simple square or circle, the vectorized outline should match exactly with the raster outline.	Verified

User Interface

Requirement	**Verification**	**Status**
LCD		
1. LCD connected properly to Motor Control Board.	1. When the Motor Control Board is on, LCD will draw power from the Motor Control Board and displays appropriate status.	Verified
2. LCD can display multiple files preloaded on the Pandaboard.	2. Have multiple files preloaded in the Pandaboard. The LCD should display one file at a time (image file name displayed on the LCD will change according to the user input).	Verified
Numeric Keypad		
1. Numeric Keypad connected properly to Pandaboard.	1. With the keypad connected to the Pandaboard via USB, toggle the 'Num Lock' key and the indicator light should toggle on and off.	Verified
2. Numeric Keypad can provide correct input to Pandaboard.	2. Connect the Pandaboard to a regular screen and open a word processor application. With both ‘Num Lock’ on and off, check that the inputs to the word processor application are consistent to what is being pressed.	Verified
3. When integrated with the LCD, the numeric keypad can scroll through files.	3. With 'Num Lock' on, scroll up and down with number ‘8’ and ‘2’. The LCD should display the current selected file name and change as we scroll with the keypad.	Verified

Communication Protocol

Requirement	Verification	Status
UART (Pandaboard <-> Motor Control Board)		
1. Setup a UART communication protocol with 115200 baudrate, 8-bit word length, 1 stop bit, and no parity bit check.	1. Program the Pandaboard with the required UART configuration, then send a byte. Probe the TX pin of the Pandaboard on the oscilloscope, we should see starting bit '0' followed by the transmitted byte in reverse order, then a stop bit '1'. Do the same with the Motor Control Board. The bitrate should be around 115 kHz.	Verified
2. Voltage divider for 3.3 V to 1.8 V logic levels generates 1.8 V logic levels corresponding to 3.3 V logic levels.	2. Using a Square wave(0 to 3.3 V) from the function generator, check that the V_{out} is a scaled replica (0 to 1.8±0.2 V)	Verified
3. Line converter for 1.8 V to 5 V logic level produces the right logic output for given logic input.	3. Using a Triangle waveform (0 to 1.8 V) generated from function generator as V_{in}, check that V_{out} is a square wave of 5±0.5 V. Check the transition point is around Vin = 0.9±0.2 V.	Verified
4. Line converter for 1.8 V to 5 V logic level produces the appropriate logic output at a frequency around 120 kHz.	4. Output a square wave (0 to 1.8 V, 150 kHz) from the function generator. Using an oscilloscope, compare the shape of the V_{in} and V_{out}. V_{out} should still look like a square wave.	Verified
I2C (Position Sensing <-> Motor Control Board)		
1. I2C communication standard mode (100 kHz), configure it to transmit 7 bytes which contain x, y coordinate, and the direction.	1. Use the IDE to step through the code and make sure that exactly seven bytes are sent; no extra interrupt after the 7 bytes are sent until the next request of position.	Verified

Appendix B Software Flowcharts and Circuit Schematics

Image Processing Software Flowcharts

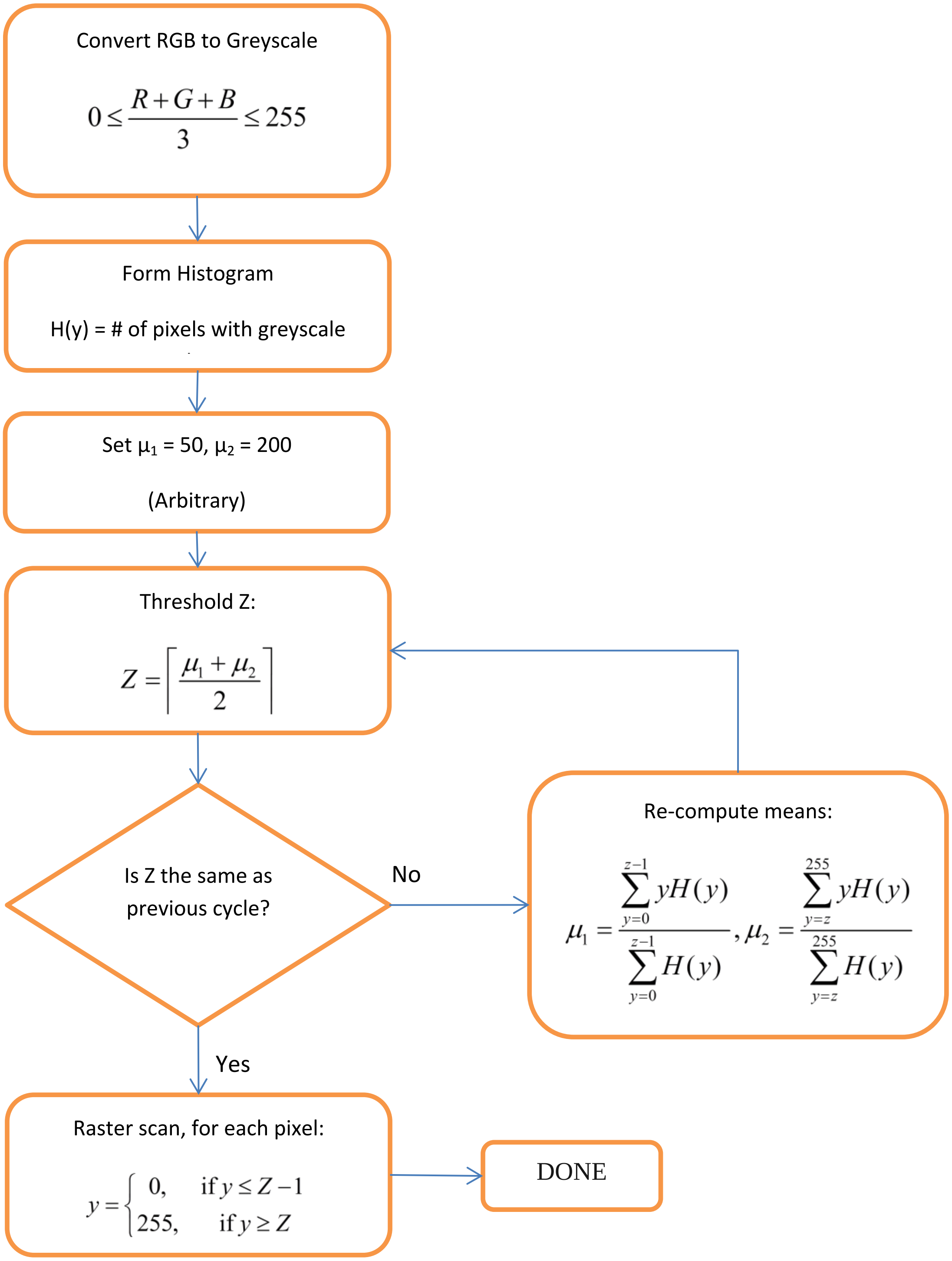


Figure B.1: K-means threshold algorithm

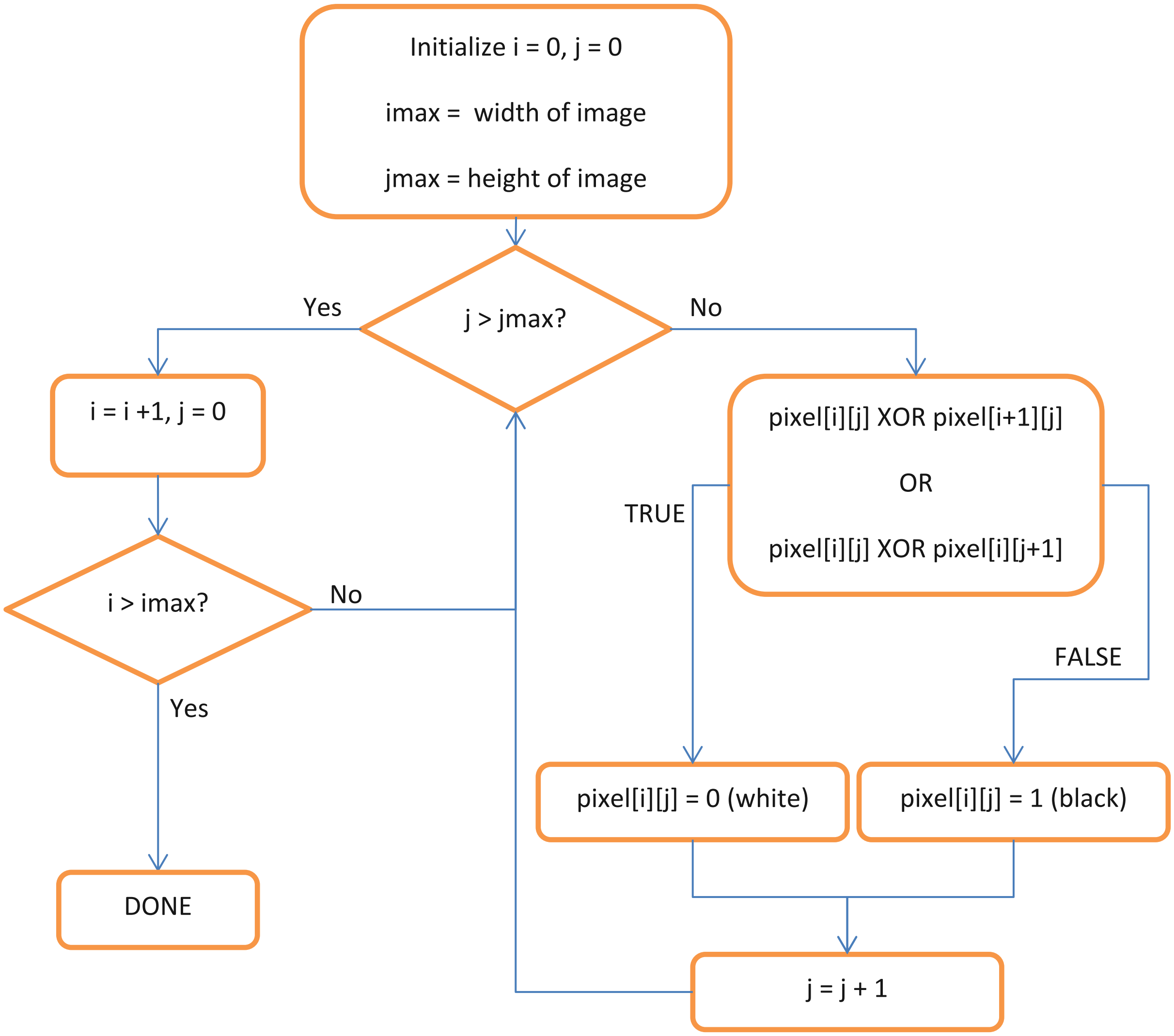


Figure B.2: Image outline algorithm

Motor Control Board Schematics

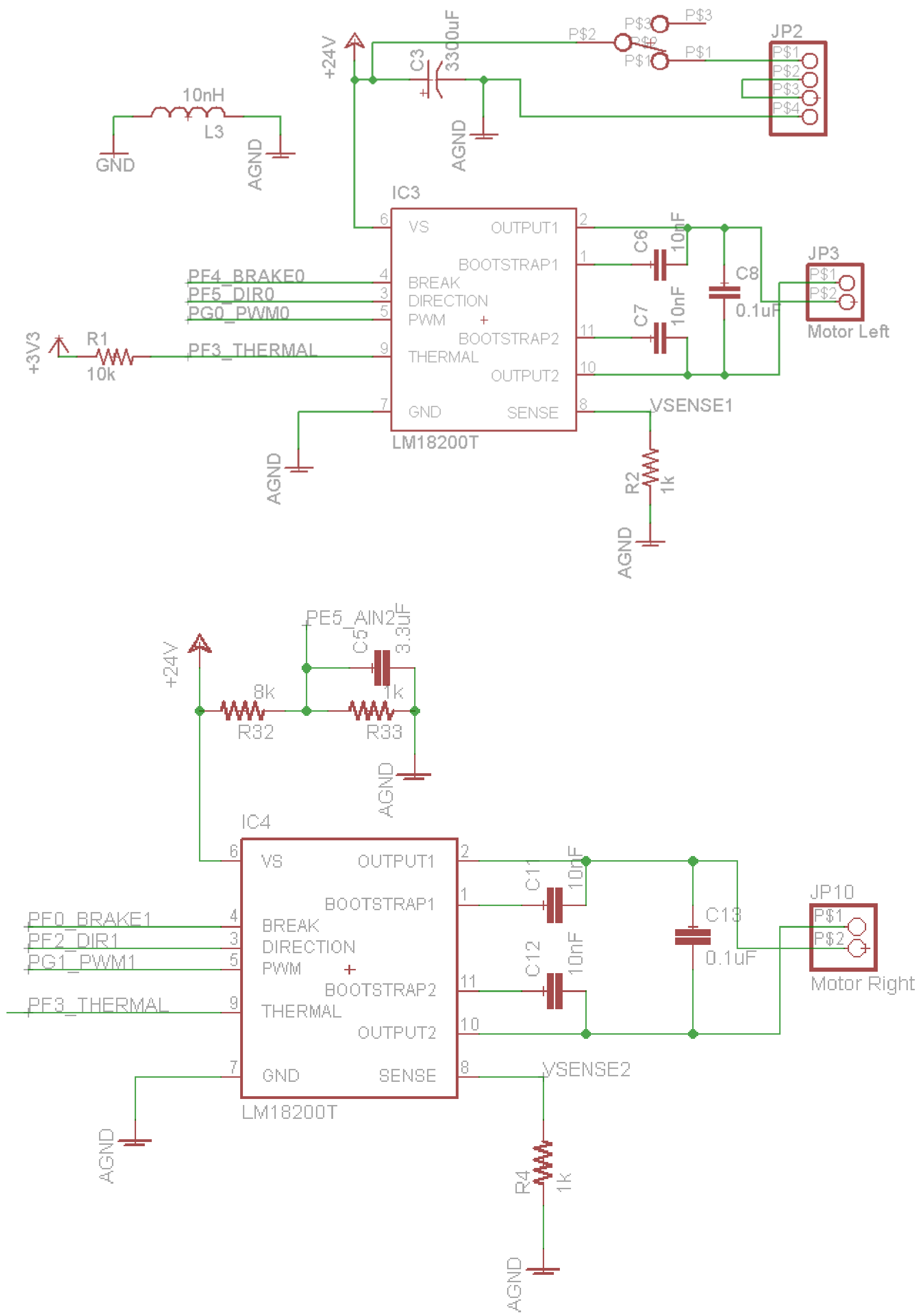


Figure B.3: Schematics for H-Bridge (left and right motor)

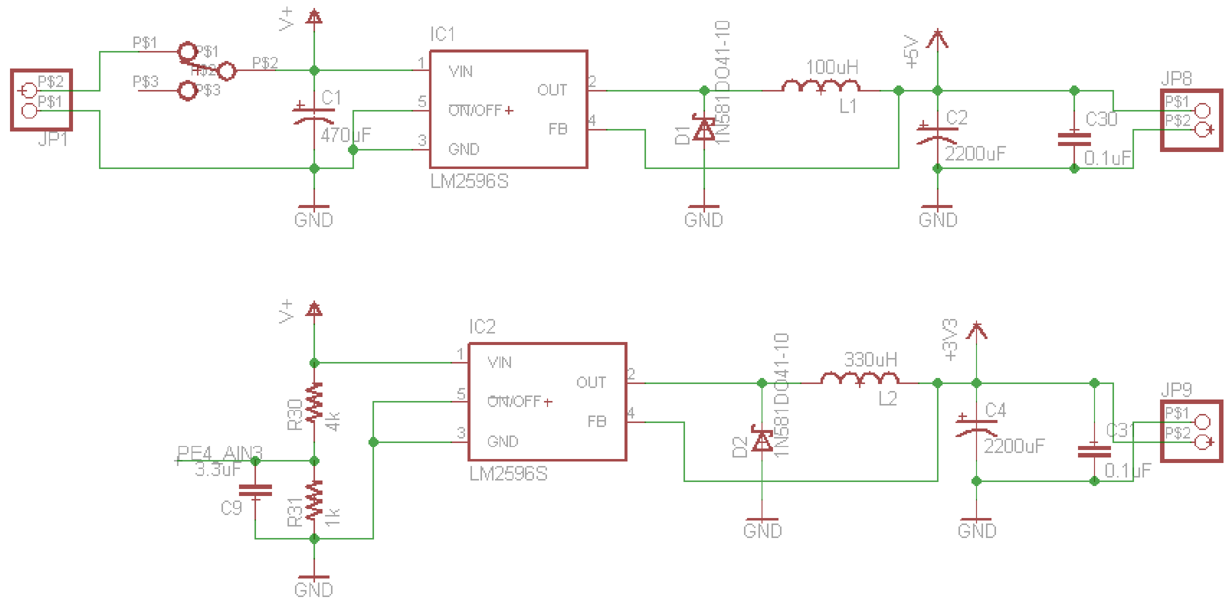


Figure B.4: Schematic for Switching Voltage Regulator

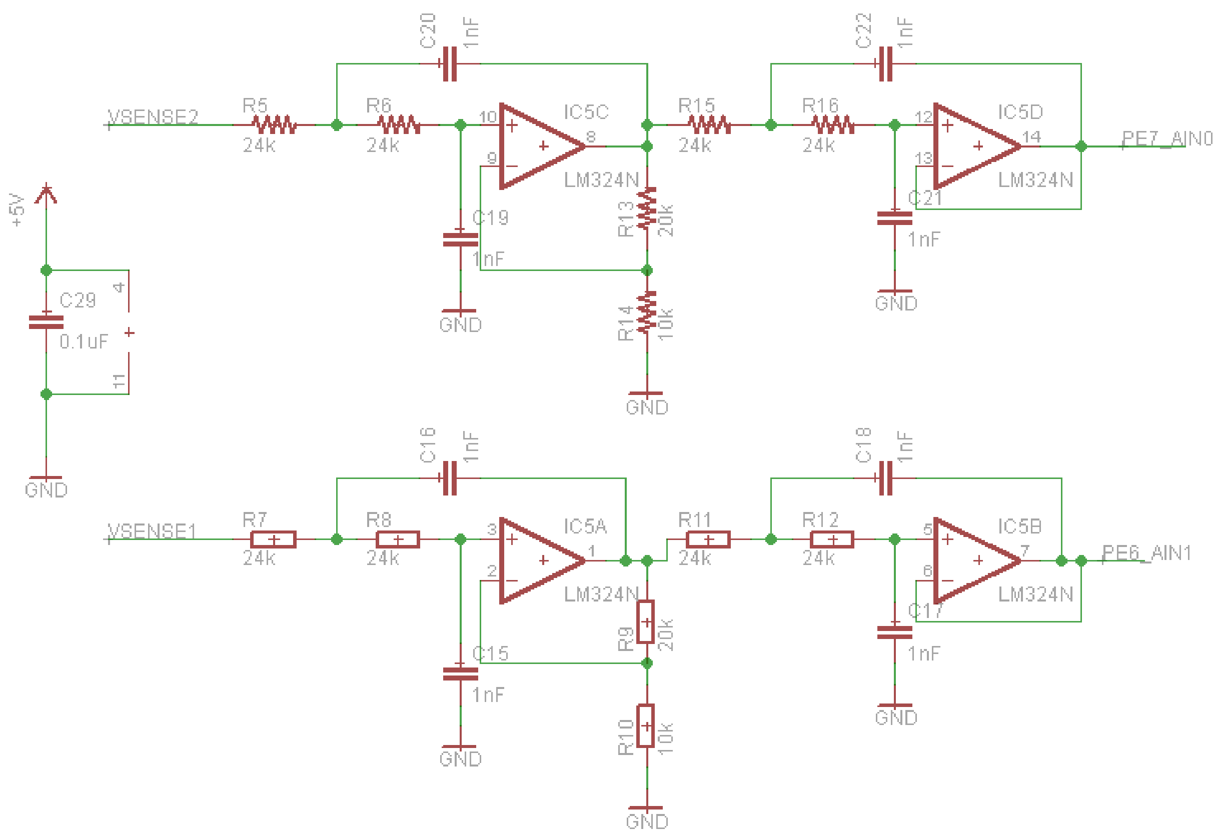


Figure B.5: Schematic for Current Sensing and Low Pass Filter

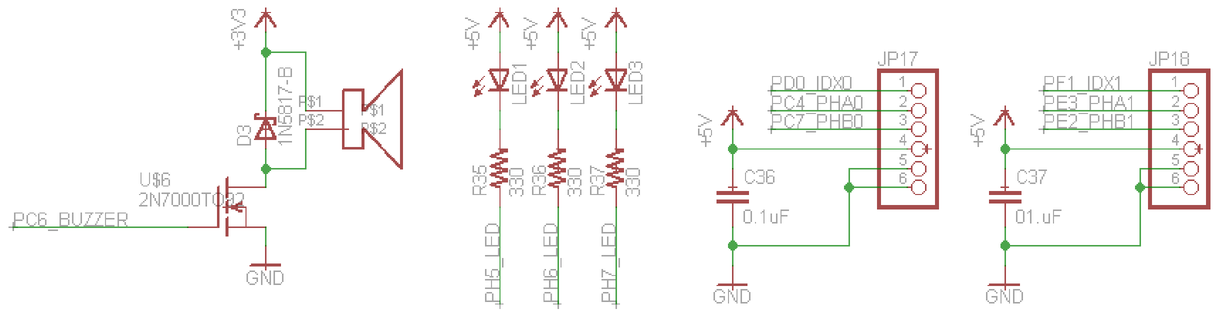


Figure B.6: Schematic for Buzzer, LEDs and Encoders

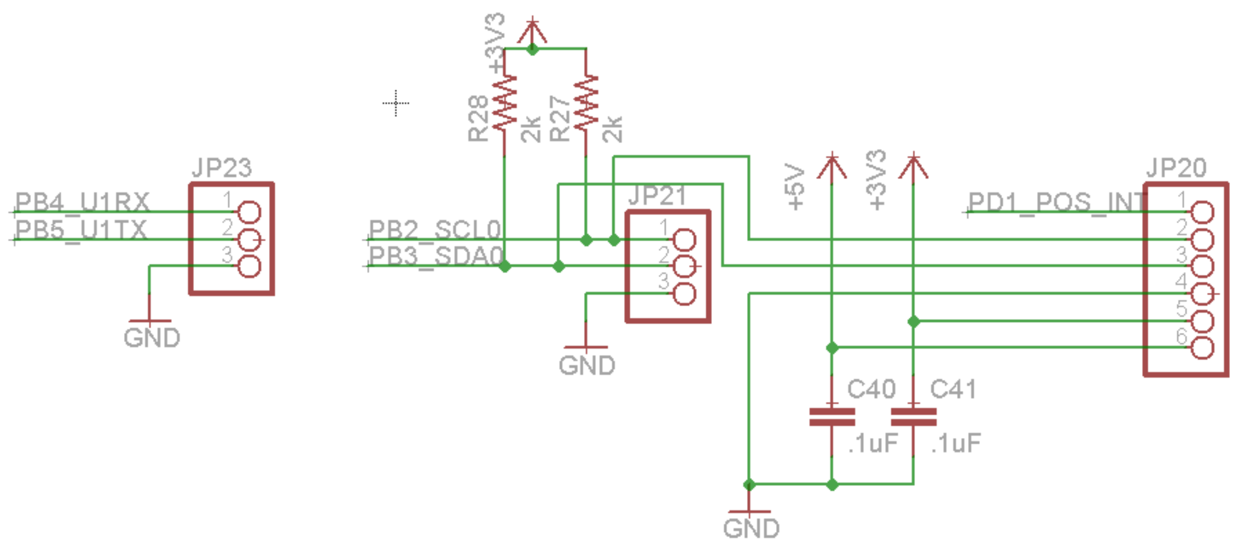


Figure B.7: Schematic for UART and I^{2}C Communication

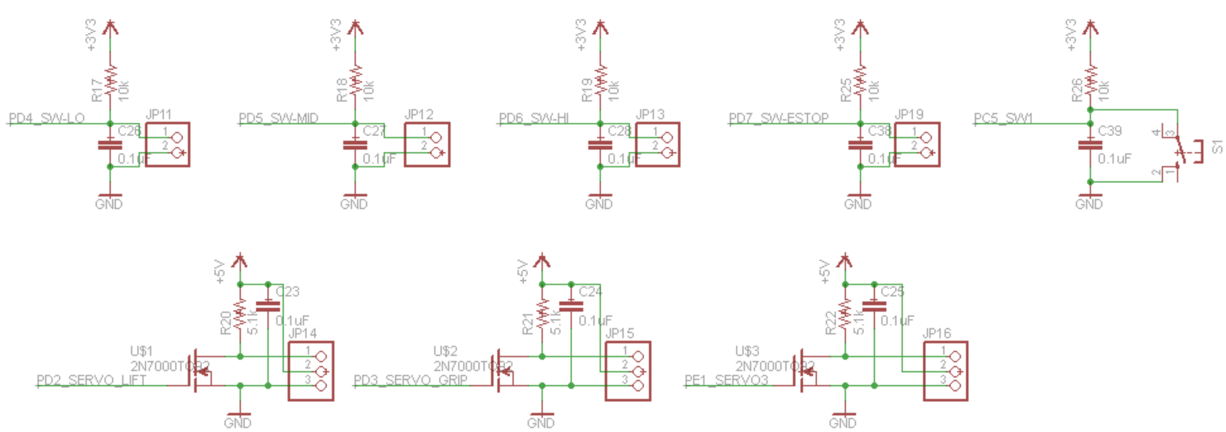


Figure B.8: Schematic for Contact Switches and Servo Output

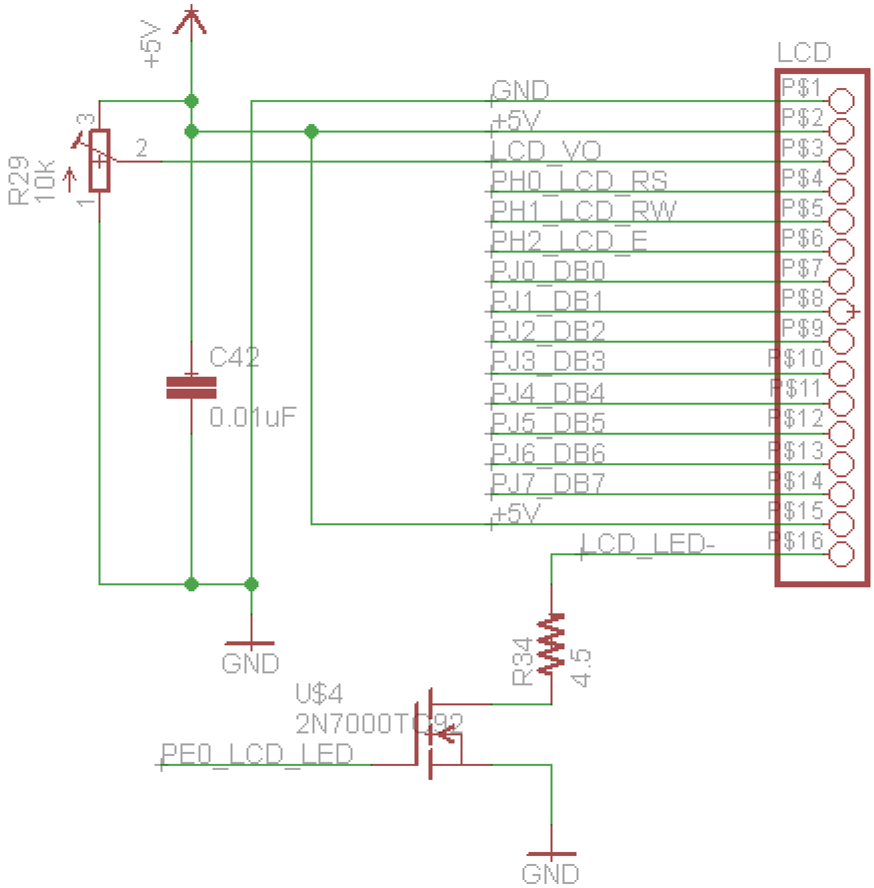


Figure B.9: Schematic for LCD Interface

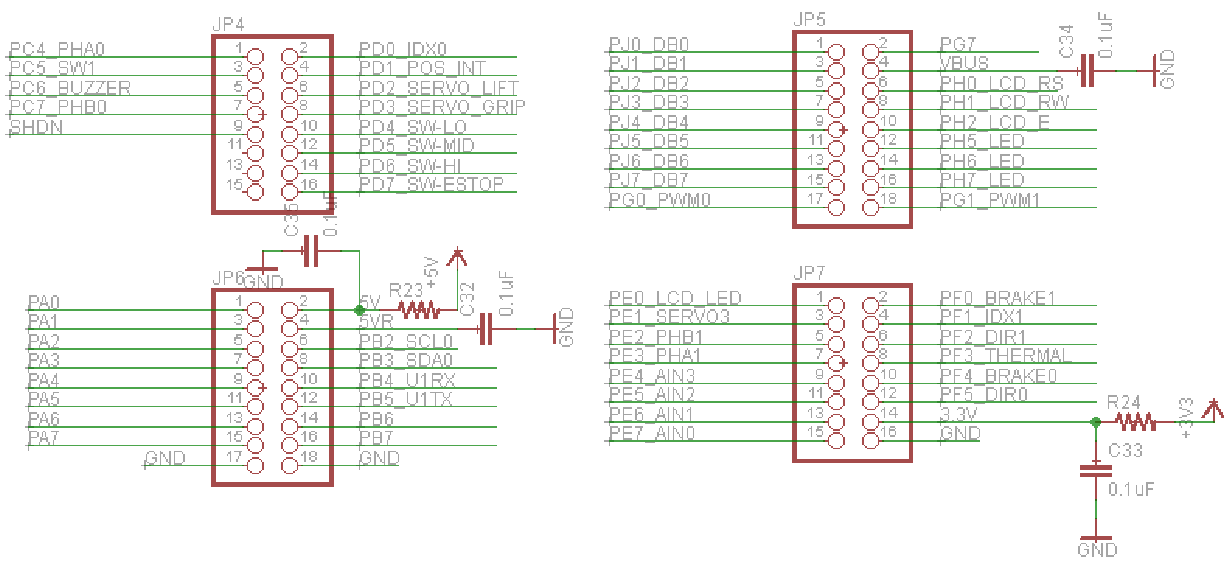


Figure B.10: Schematic for Microcontroller Board

Position Sensing Circuit Schematics

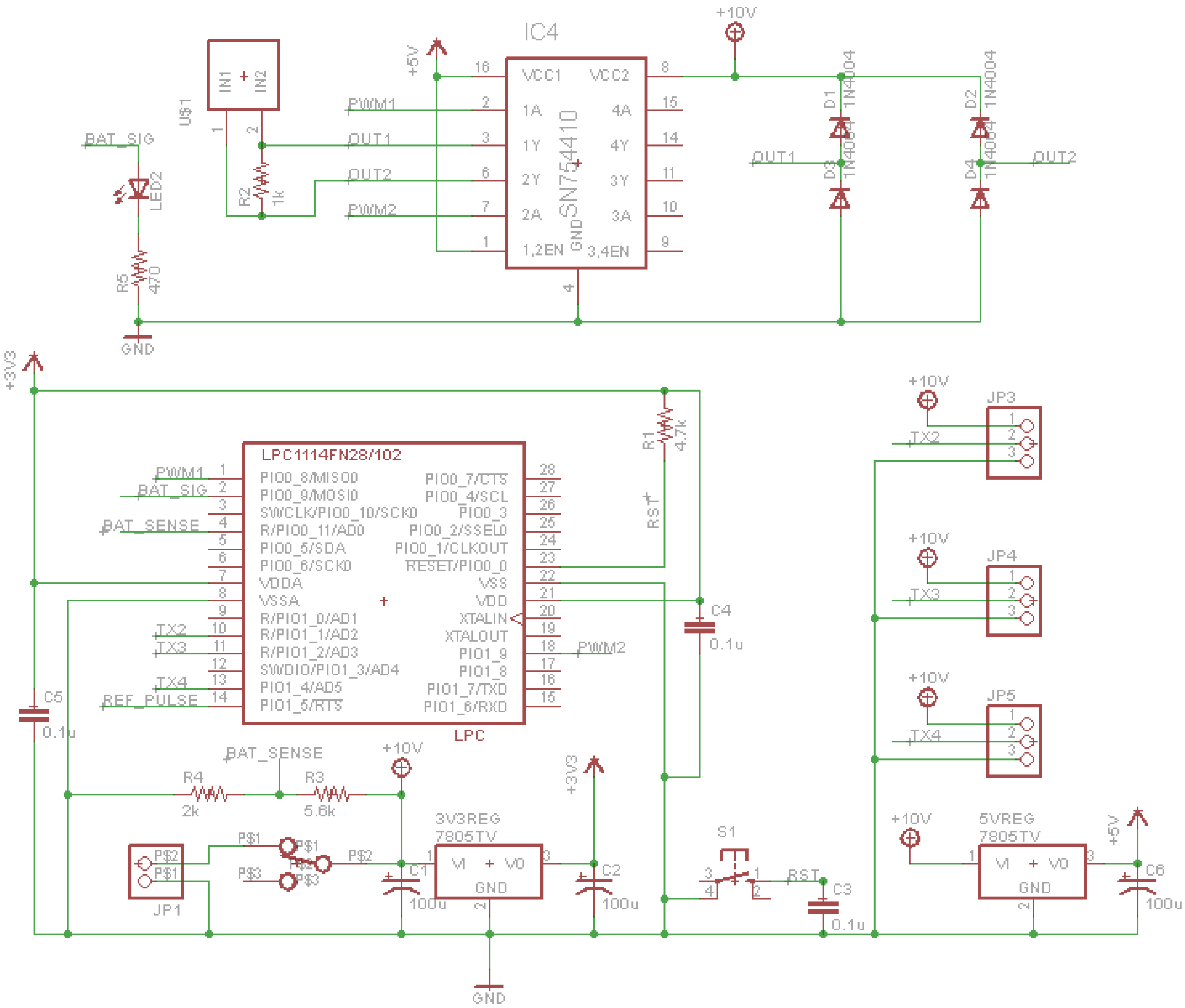


Figure B.11: Base station schematics (one transmitter attached)

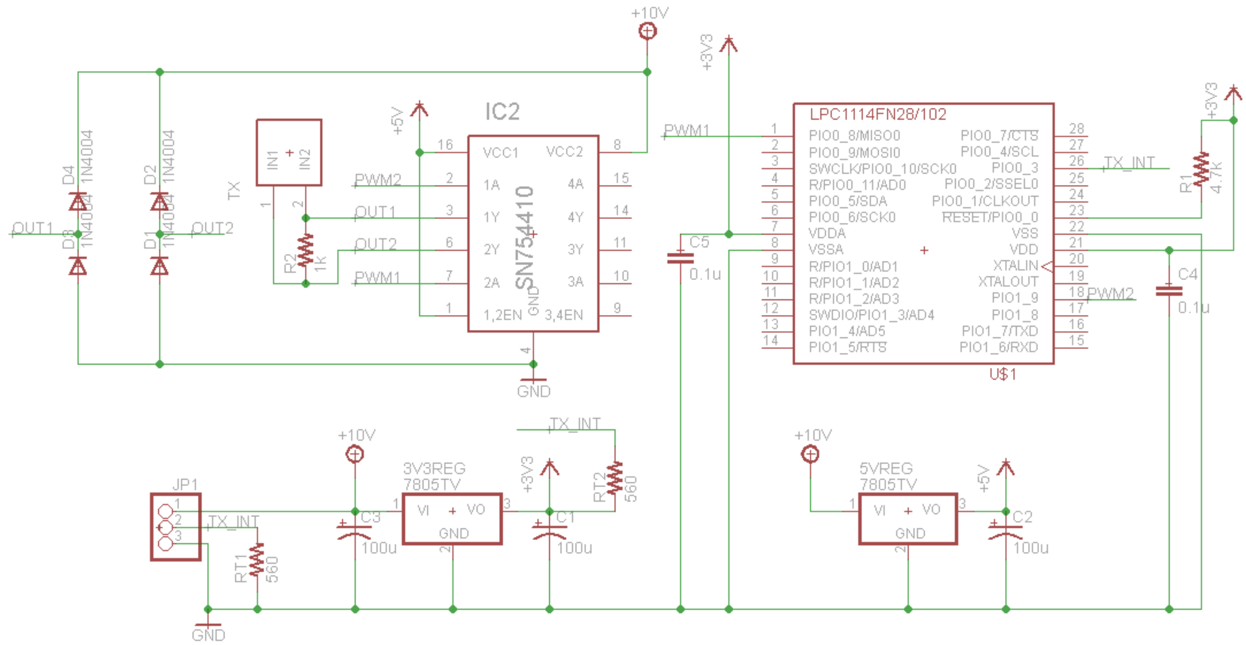


Figure B.12: Transmitter schematic

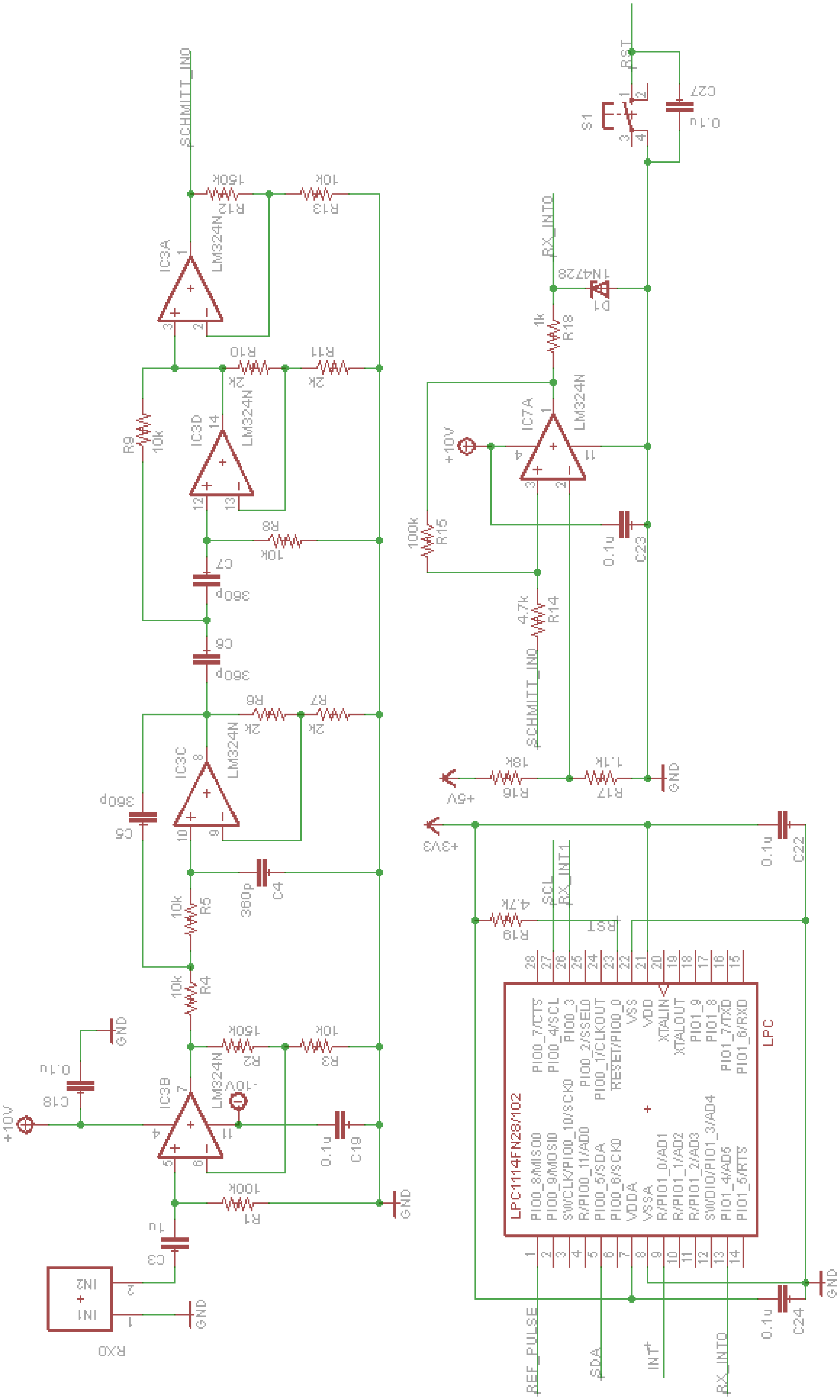


Figure B.13: Receiver module (one of the two ends) schematic

Pandaboard – Motor Control Board Logic Conversion

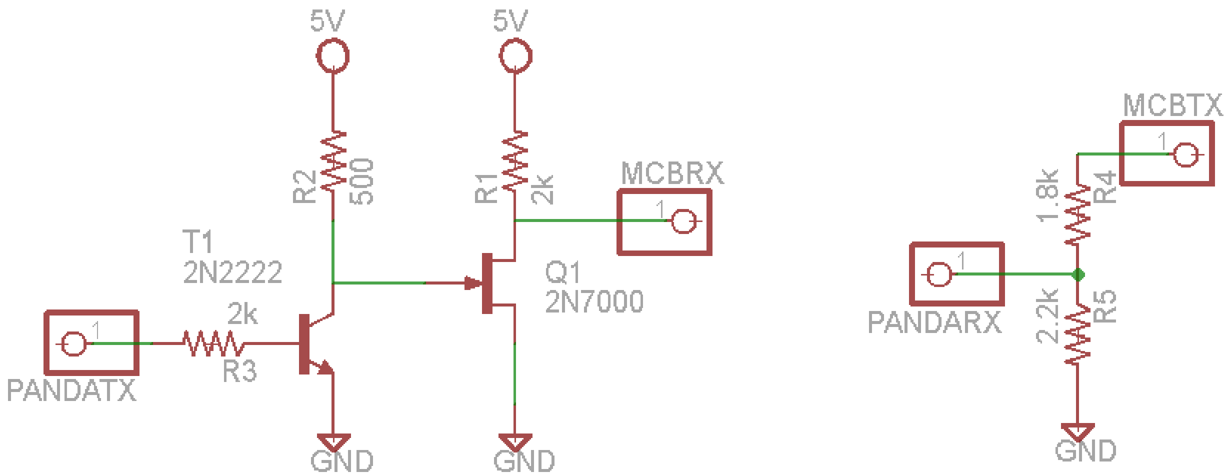
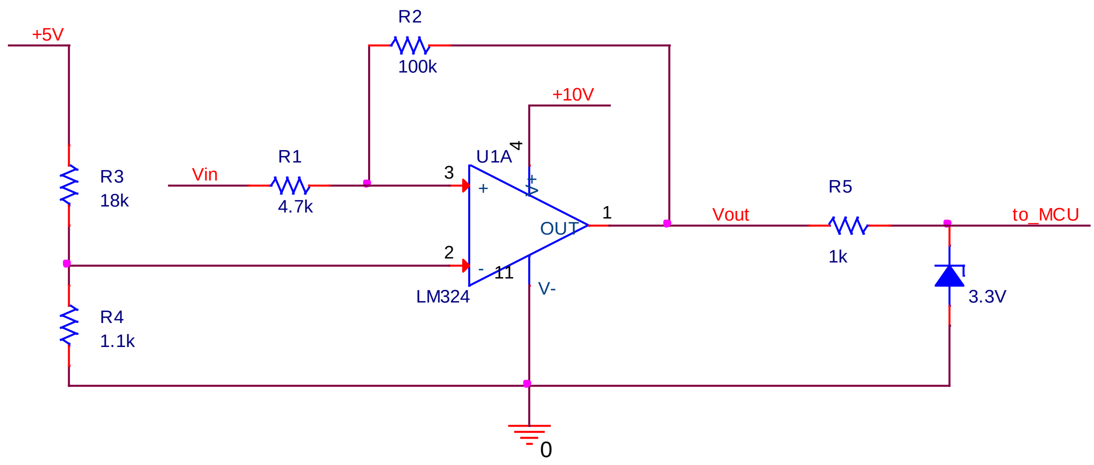


Figure B.14: 1.8 V to 5 V conversion circuit (left) and 3.3 V to 1.8 V conversion circuit (right)

Appendix C Equations and Calculations

Schmitt Trigger



Range = 0.3 V

At V_{out} = 0 V:

$$V^+ = \frac{R_2}{R_1+R_2}V_{in} \; > \; V^- \text{ to make } V_{out} \text{ goes high.}$$

$$V_{in} > \left(1+\frac{R_1}{R_2}\right)V^-$$

At $V_{out} = V_{max}$ (saturation)

$$V^+ = \frac{R_2}{R_1+R_2}V_{in} + \frac{R_1}{R_1+R_2}V_{max} < \; V^- \text{ to make } V_{out} \text{ goes low.}$$

$$V_{in} < \left(1+\frac{R_1}{R_2}\right)V^- - \frac{R_1}{R_2}V_{max}$$

We found out that the highest the output can achieve is 7.3 V, so V_{max} is 7.3 V.

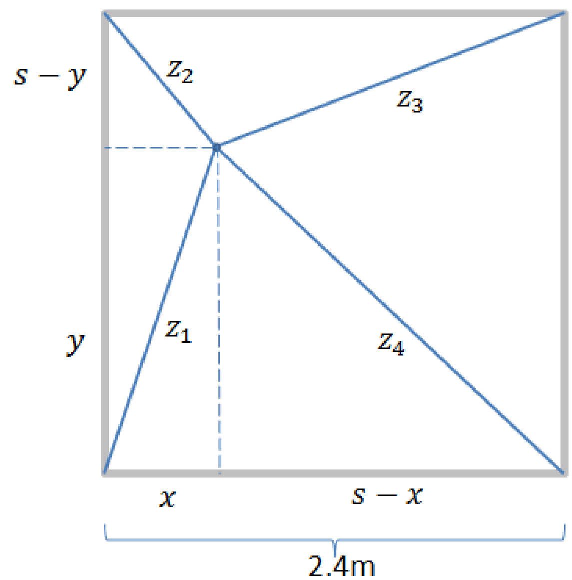
The range becomes $\frac{R_1}{R_2} \times 7.3 = 0.3$ so $\frac{R_1}{R_2} = 0.0411$.

We choose R_1 to be 4.7 kΩ and R_2 to be 100 kΩ.

We want the upper threshold of the Schmitt trigger to be 300 mV; if V_{in} is greater than 300 mV, we want V_{out} to go high. That means $V^- = \frac{0.3}{1+{}^{4.7}/_{100}} = 0.2865\,V$.

To get 0.2865 V, we use a voltage divider. We choose R_3 to be 18 kΩ and R_4 to be 1.1 kΩ.

Triangulation



$$x^2 + y^2 = {z_1}^2 \qquad \text{eq1}$$

$$(s-y)^2 + x^2 = {z_2}^2 \quad \rightarrow \quad s^2 - 2sy + y^2 + x^2 = {z_2}^2 \qquad \text{eq2}$$

$$y^2 + (s-x)^2 = {z_4}^2 \quad \rightarrow \quad y^2 + s^2 - 2sx + x^2 = {z_4}^2 \qquad \text{eq3}$$

$$(s-x)^2 + (s-y)^2 = {z_3}^2 \quad \rightarrow \quad 2s^2 - 2sx - 2sy + x^2 + y^2 = {z_3}^2 \qquad \text{eq4}$$

Subtracting eq1 from eq2 gives us $\quad {z_2}^2 - {z_1}^2 = s^2 - 2sy \quad \rightarrow \quad y = \frac{{z_1}^2 - {z_2}^2 + s^2}{2s}$

Subtracting eq1 from eq3 gives us $\quad {z_4}^2 - {z_1}^2 = s^2 - 2sx \quad \rightarrow \quad x = \frac{{z_1}^2 - {z_4}^2 + s^2}{2s}$

Subtracting eq3 from eq4 gives us $\quad {z_3}^2 - {z_4}^2 = s^2 - 2sy \quad \rightarrow \quad y = \frac{{z_4}^2 - {z_3}^2 + s^2}{2s}$

Subtracting eq2 from eq4 gives us $\quad {z_3}^2 - {z_2}^2 = s^2 - 2sx \quad \rightarrow \quad x = \frac{{z_2}^2 - {z_3}^2 + s^2}{2s}$

Appendix D Figures of Verification Results

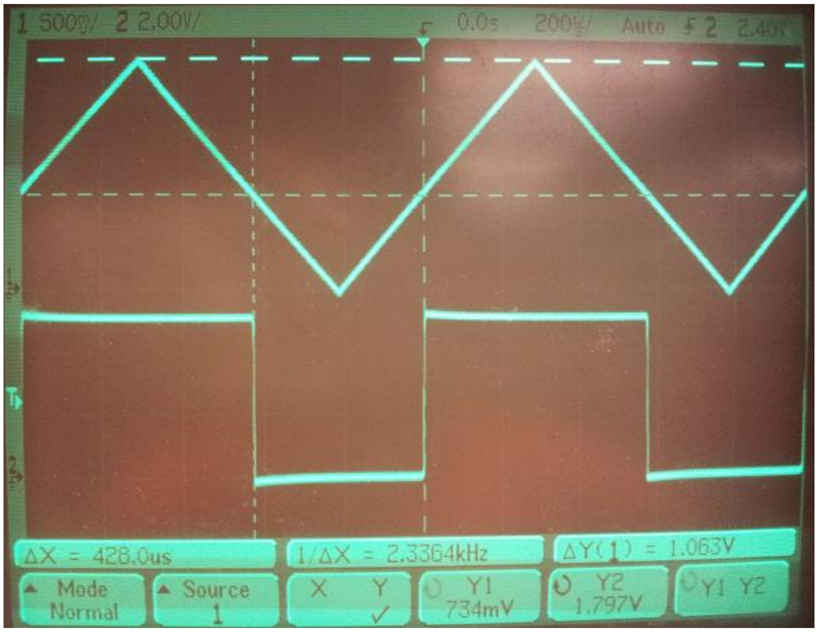


Figure D.1: UART communication verification #3 (transition voltage at ~0.7 V shown by Y1 cursor)

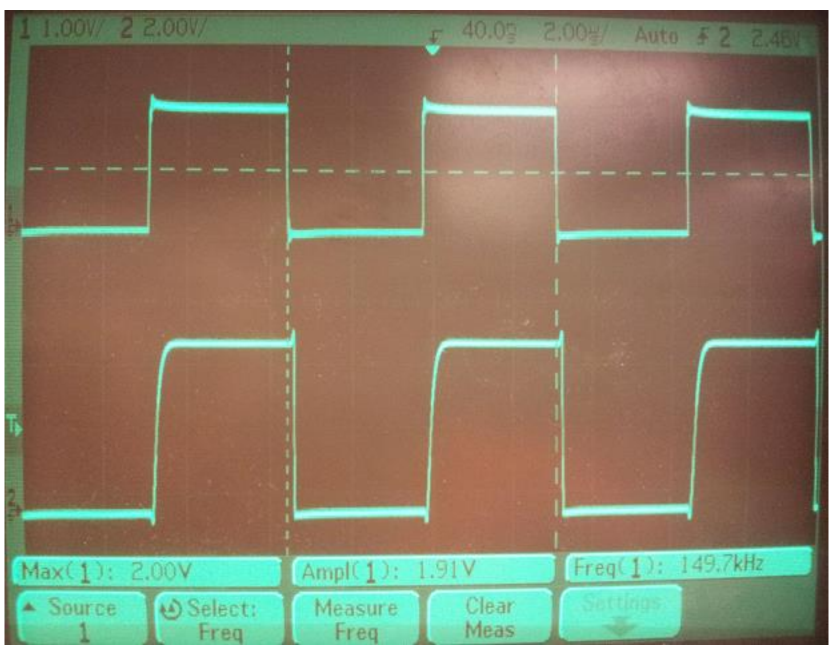


Figure D.2: UART communication verification #4

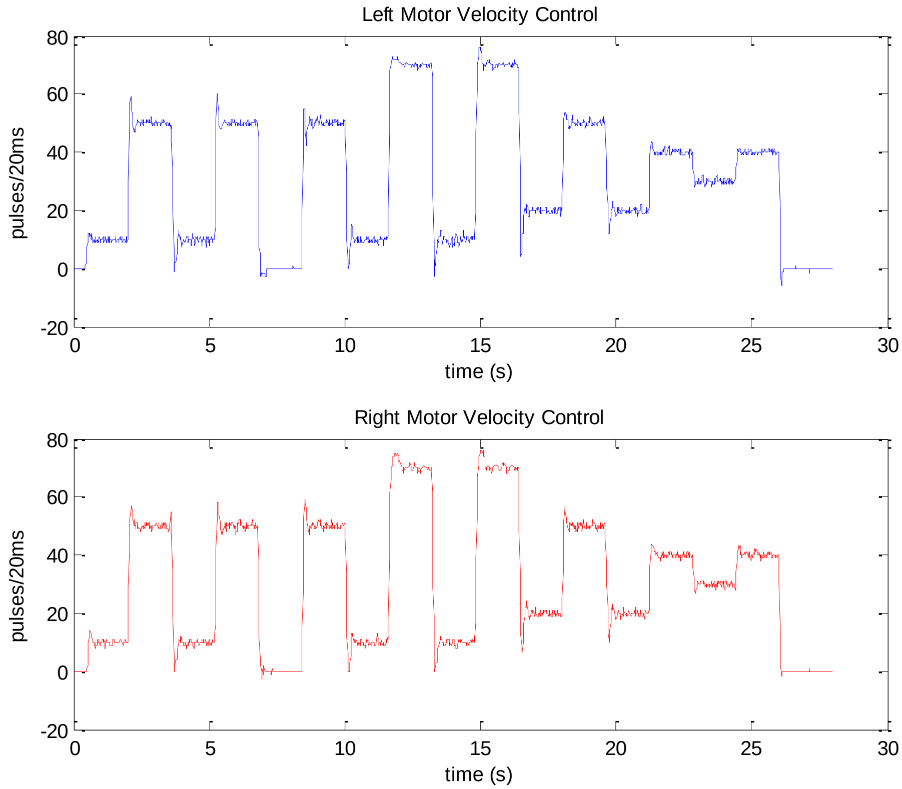


Figure D.3: Velocity output of test sequence

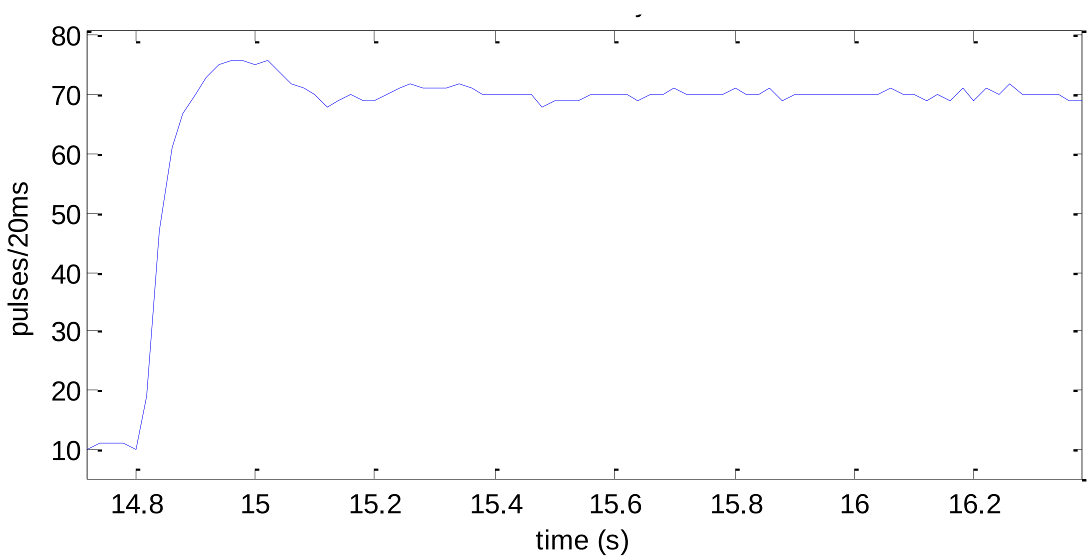


Figure D.4: Velocity Step Response

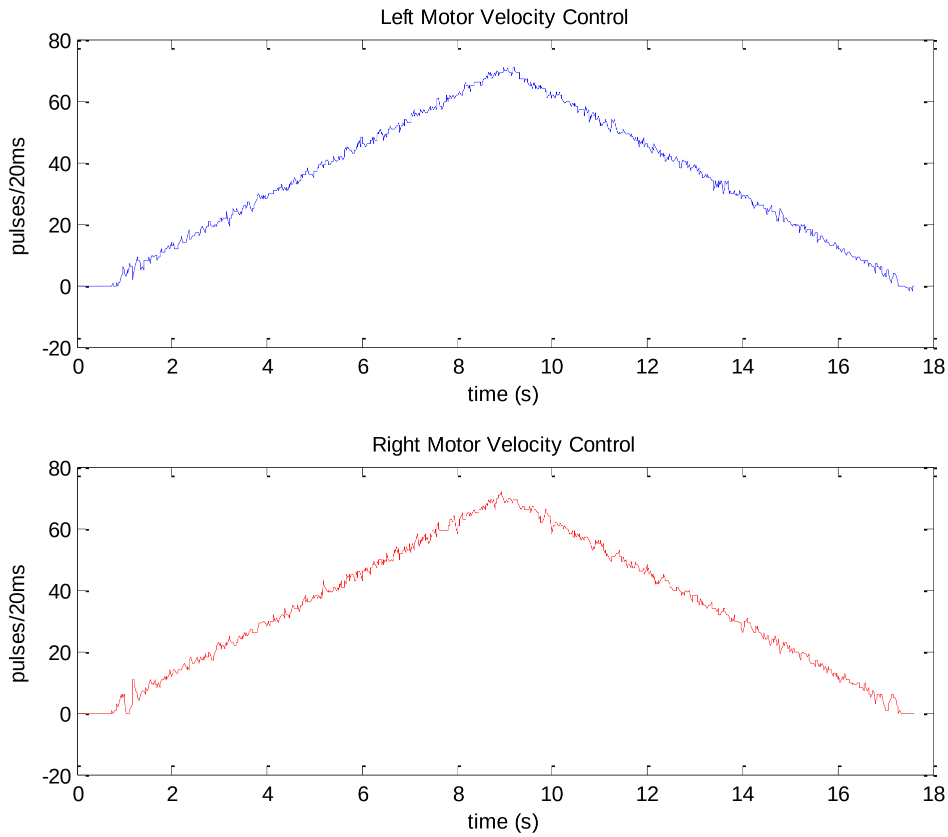


Figure D.5: Velocity Ramp Response

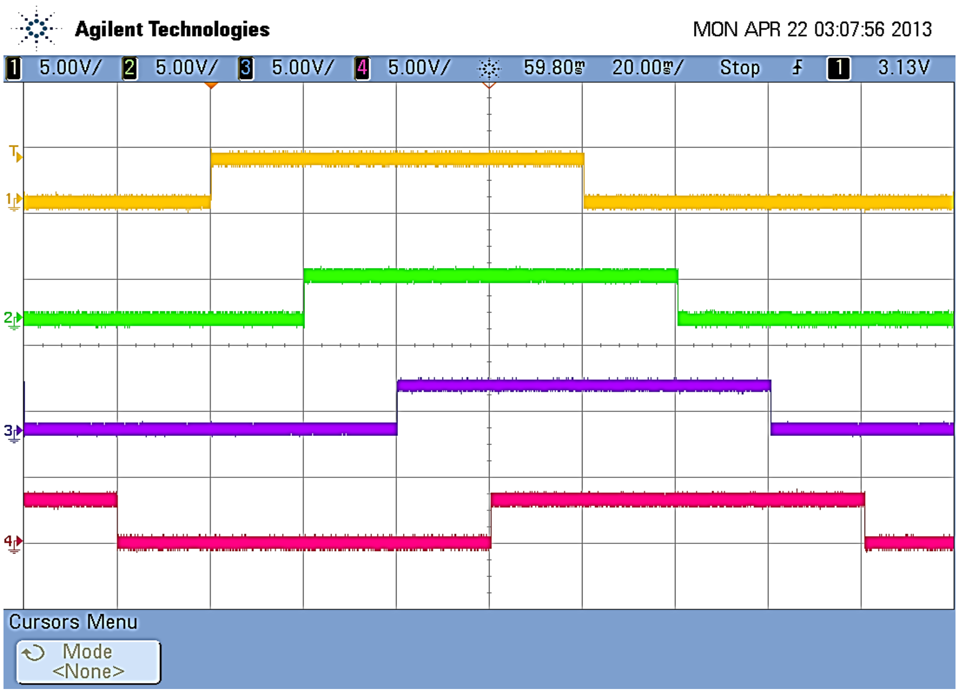


Figure D.6: Base Station verification (TX toggling interrupt with 20 ms spacing)

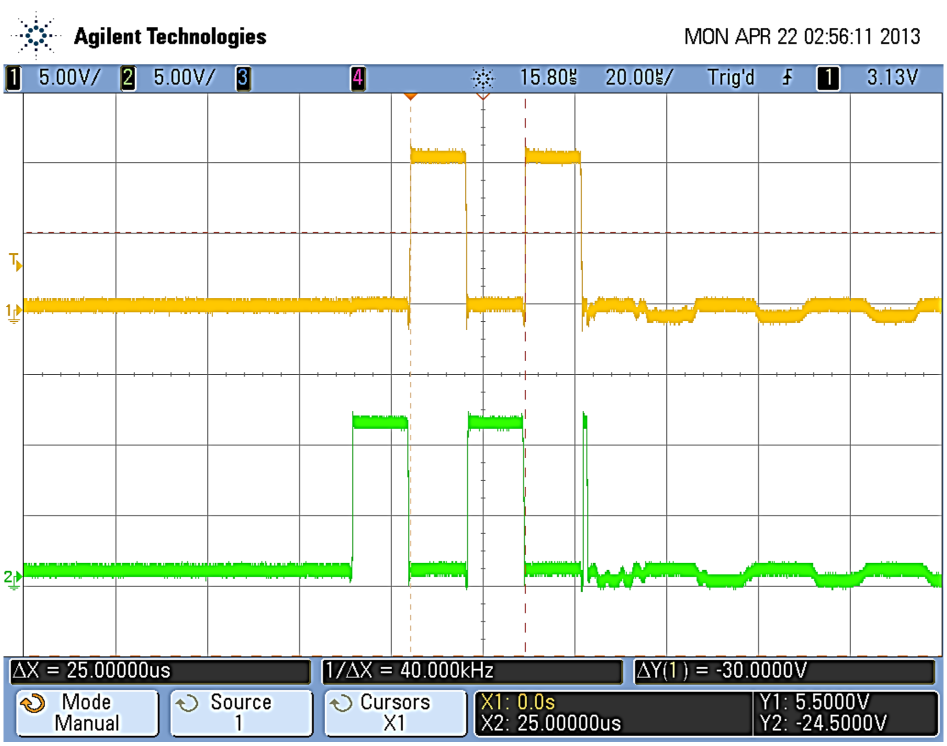


Figure D.7: Transmitter verification (40 kHz, 10 V PWM out of phase to produce 20 Vpp)

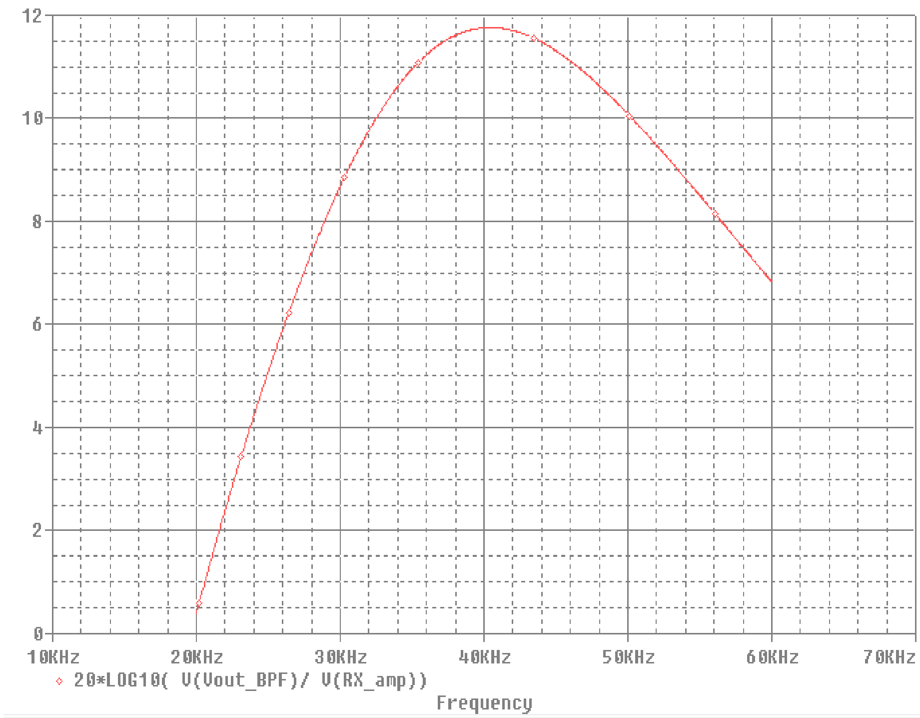


Figure D.8: Receiver Module band pass filter frequency response

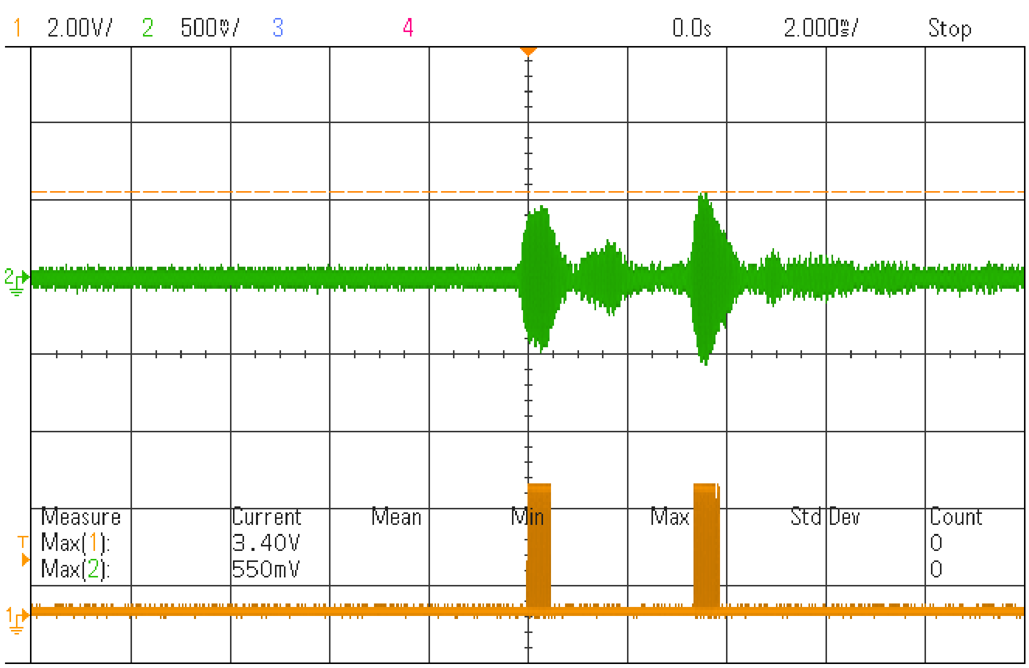


Figure D.9: Receiver Module Schmitt Trigger verification

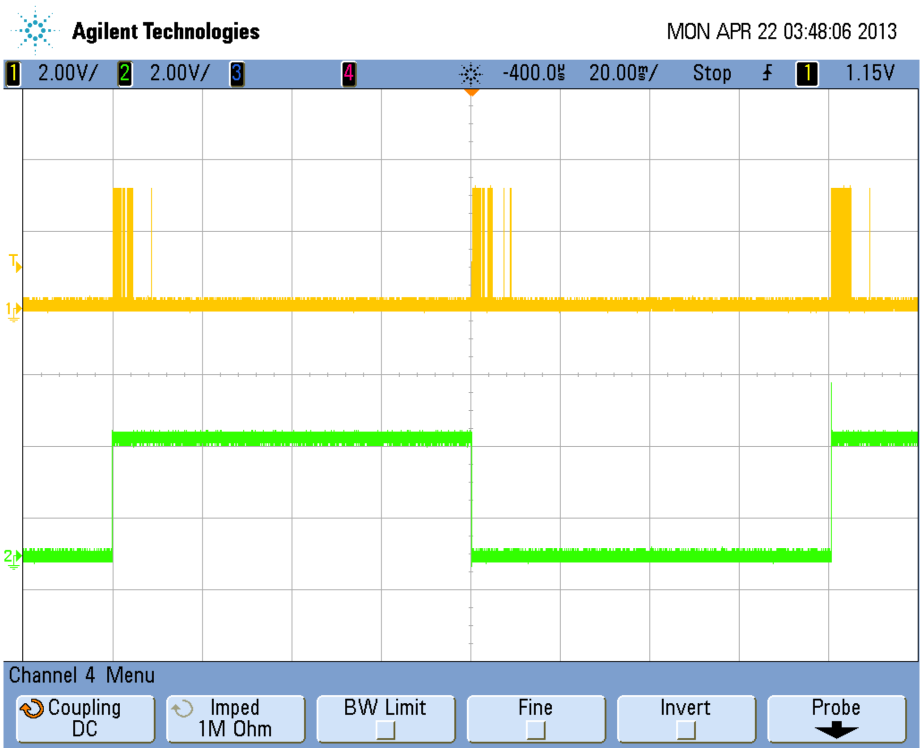


Figure D.10: Receiver Module MCU verification (interrupt triggered once every incoming TX pulse)