CONTROL AND SENSOR DEVELOPMENT ON A FOUR-WHEEL PYRAMIDAL REACTION WHEEL PLATFORM

A Thesis

Presented to the

Faculty of

California Polytechnic State University

San Luis Obispo

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science in Aerospace Engineering

By

Jeffery Jay Logan

October 2008

COMMITTEE MEMBERSHIP

TITLE: CONTROL AND SENSOR DEVELOPMENT ON A FOUR-WHEEL PYRAMIDAL REACTION WHEEL PLATFORM

AUTHOR: Jeffery Jay Logan

DATE SUBMITTED: October 2008

COMMITTEE CHAIR: Dr. Eric A. Mehiel

COMMITTEE MEMBER: Dr. Kira J. Abercromby

COMMITTEE MEMBER: Dr. Jordi Puig-Suari

COMMITTEE MEMBER: Dr. Albert A. Liddicoat

ABSTRACT

CONTROL AND SENSOR DEVELOPMENT ON A FOUR-WHEEL PYRAMIDAL REACTION WHEEL PLATFORM

By
Jeffery Jay Logan

The Pyramidal Reaction Wheel Platform, or PRWP, is used to simulate three-axis controls in a torque free space-like environment. The primary purpose of the system will be to evaluate the effects of conjoining sensors to maximize pointing accuracy. Furthermore, the system will incorporate a star tracker in conjunction with a Simulated Star Field (SSF) to better estimate the PRWP orientation. For the sake of this document, however, the goal is to implement a gyroscope, wheel rate sensors, and a make-shift accelerometer—to the PRWP—and integrate a controls algorithm such that three-axis controls are achieved for the PRWP.

Three sensors were either better integrated into the system or added altogether. Tachometers were created as a form of hardware circuitry to measure each wheel rate with an accuracy of approximately 2.5 Hz (nearly 15 radians per second). The TAC board circuitry converted each motors encoder output into a speed by use of a frequency to voltage converter. Additionally, although three gyroscopes had been implemented previously, the system was better incorporated into the model such that it was directly transformed via a ROBOSTIX ADC converter before being relayed to SIMULINK via a Bluetooth link. The MEMS gyroscopes allowed for very accurate rate measurements—with a minimum resolution of approximately 0.25 radians per second. Finally, a makeshift accelerometer was incorporated into the system for the purpose of system

identification. The accelerometer was incorporated into the system by utilizing a discrete time derivative of the gyroscope readings. However, thankfully a system of two accelerometers can be later utilized to achieve an accuracy of approximately 6 degrees per second-second in the x-axis and 2-3 degrees per second-second in the y- and z-axes.

A controls test was performed where the starting location was q_o=[0, 0, sqrt(2)/2, sqrt(2)/2] and the target location was q_c=[0, 0, 0, 1]. At 80 seconds, the pointing accuracy was 70 degrees around the target and the system was unable to settle during the 80 second trial. The inaccuracy was because of the low frequency of operation of the system—1 Hz. Additionally, the platform reacts slowly to sensor readings and commands. The coupling of these issues causes the pointing accuracy to high. Furthermore, through experimental testing, the maximum wheel rate was found to be approximately 6400 RPM at a duty cycle of 50% at an 8000Hz PWM application due to the Pololu MD01B design limitations: low voltage range (up to 16V), low limit current limiter (5A), and high susceptibility to overheating for large currents.

To my Parents Carmen and Lloyd,
for all of the support and consulting.

To my Brother Eric,
for your contagious happiness
and unrelenting brotherly love.

To my longtime friend Erika Fitzgerald,
for all the love and motivation.

To all my friends,
for telling me to “get ‘er done”.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1: PROJECT INTRODUCTION

1.1 Introduction of Project

1.1.1 Project Purpose

The Pyramidal Reaction Wheel Platform, or PRWP, tests control algorithms and aims towards maximizing pointing accuracy in a simulated space environment. The PRWP aims to achieve its tri-axial environmental simulation by concentrating its center of gravity, CG, above a nearly frictionless spherical air bearing.

The system will use several sensors to deduce orientation including: wheel tachometers to relay wheel rates, three Micro-Electrical to Mechanical System (MEMS) rate gyros to deduce satellite orientation, and an accelerometer for system identification. By coupling the sensors with the simulated environment, the final system will be able to be tested to deduce an optimal method for orbit determination and satellite orientation to be utilized in missions. It is hoped that the research generated by this project will assist in further increasing the pointing capabilities of outwardly observing micro-satellites.

1.1.2 Project Goals

The ultimate goal of the project is to allow for cheap yet accurate pointing capabilities for use of observation satellites. Many outwardly observing satellites could utilize the eventual research produced by this project to integrate and couple three-axis control with orientation-deducing sensors to improve overall sensor accuracy. The eventual use of this project could be applied to lower budget orbiting telescopes such as the future Cal Poly Longview concept.

Before the ultimate goal can be met several flow down goals were generated. First, the system needs to be in reasonable order while not isolated. For this portion the system needs to be shown to partially work through extensive exterior interaction. Second, the system needs to become fully operational and more isolated from exterior motives. This step requires that the system is completely isolated from wiring to exterior power/command sources. The only interaction shall be data transfer between the platform and the off board computer generated via Bluetooth link.

1.2 Thesis Overview

When this project began the PRWP was not in a optimal working condition. Two reaction wheels would not spin and none could receive and execute wireless commands, due to many planned changes the controls algorithms needed significant modifications, and the wheel rates could not be deduced without external interaction by a computer. Furthermore, limited ROBOSTIX Analog to Digital Conversion (ADC) ports created conflicts which inhibited controllability of the system. Additionally, the system's onboard signal routing created significant electrical interference. Finally, the system needed a revamped safety system to be operated following the summer of 2007.

The goal of this thesis was to utilize SIMULINK to generate control commands and deliver the command to the system via blue tooth to an onboard computer (ROBOSTIX and GUMSTIX combo) for further execution by the PRWP. The control algorithms incorporate the MEMS rate gyros to detect orientation, and four tachometers to detect wheel rates—to generate relevant commands to the system.

There are several flow down goals to this thesis including: designing and manufacturing the necessary hardware and software to allow for accurate wheel rate and directional sensing, updating the control algorithms, considering the addition of an accelerometer sensor, realigning and correctly utilizing the MEMS rate gyro data, and converging the components into a working and nearly isolated system. Throughout this thesis the approaches taken, designs chosen, implementation suggestions, testing methods, and operational results are investigated.

1.3 Literature Review

This thesis assumes the reader has basic knowledge of ridged-body dynamics, electronics, modern control theory, an understanding of both SIMULINK and MATLAB, and a pre-existing understanding of a quaternion, quaternion math, quaternion rotation sequences. For this purpose, several educational resources will be utilized throughout this document.

1.4 State of the Art Review

Other simulators similar to the PRWP generally facilitate a torque free environment. These simulators use the torque free environment to display control capabilities involving formation flying or precision pointing by negating both the influence of gravity and friction on results. These torque free simulators fall into the category of Distributed Spacecraft Attitude Control System Simulators, or DSACSS, and are used to demonstrate and test high risk systems prior to mission engagement.

Virginia Tech has developed a DSACSS system to research formation flying.[1] The Virginia Tech design is very similar to the PRWP in that it utilizes momentum exchange devices for control, a fine weight balancing system for proper CG placement, and an air bearing to reduce friction. While the PRWP is a “Tabletop” style configuration, the Virginia Tech style can also test a “Dumbbell” arrangement. The Virginia Tech DSACSS combines two Whorl-I platforms in formation by connecting them, in parallel, to form a dumb-bell formation called Whorl-II, see Figure 1.1

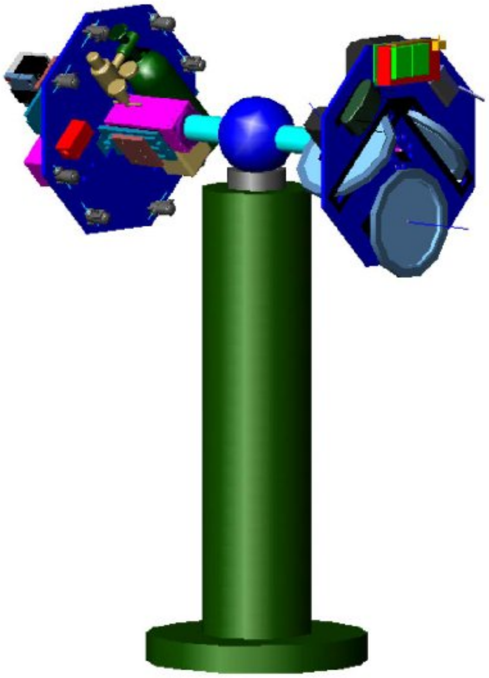


Figure 1.1. Whorl-II dumbbell configuration.[1]

The Virginia Tech model is used to not only perform analysis on formation flight, but also to analyze the effects of mixed control schemes and Magnetic Bearing Research. To clarify, mixed control schemes include not only the internal actuators—reaction wheels, CMGs, or magnetic torques, but also external actuators—thrusters. The tabletop testing also attempts to utilize magnetic bearings rather than air bearings to minimize friction. Although the Virginia Tech DSACSS does not combine different aspects to measure pointing accuracy, the overall concept is very similar to the PRWP. Both projects utilize nearly the same design concept to understand spacecraft dynamic and control theory.

A second example of a state of the art simulator is provided through Honeywell's development of a spacecraft test bed which allows for a high-agility slew and scan capabilities via six 225 ft-lb CMGs, as seen in Figure 1.2. The platform further offers structural control, vibration isolation, payload-steering, utilizes structural dampers for jitter minimization, and couples the system with high resolution sensors. Furthermore, the system utilizes an air bearing—similar, but larger, to that utilized by Virginia Tech and throughout this document—to simulate and calculate the platforms capabilities. The aim of the project is to analyze the test bed's "ability for auto coding MATLAB/SIMULINK models, attitude-control and estimation design, structural design, and disturbance-mitigation measures."[2] It also shows "the pointing performance of a structure driven by a momentum control system."[2]

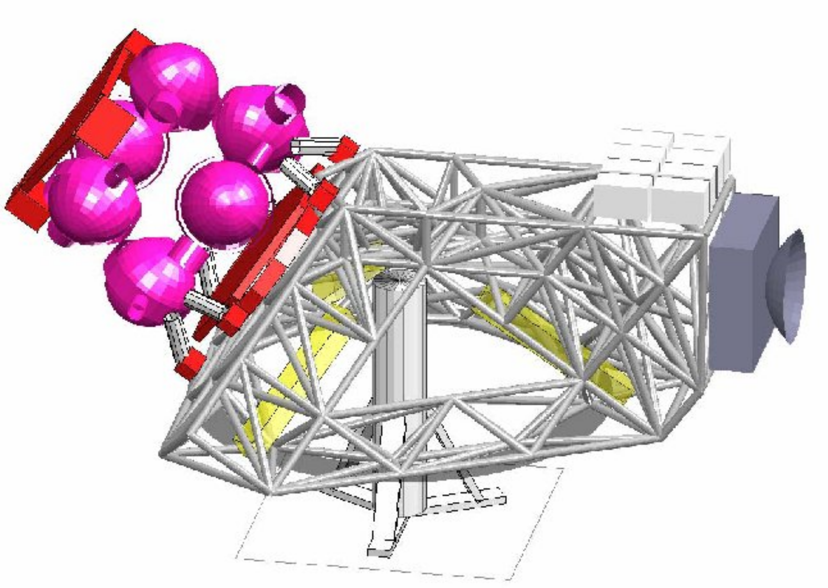


Figure 1.2. 2003 Testbed Structure with 6 225 ft-lb-sec CMGs.[2]

CHAPTER 2: INTRODUCTION TO PLATFORM COMPONENTS

The PRWP components which are relevant to this thesis can be divided into seven subcategories including: the platform structure, the air compressor, the Reaction Wheels, the ROBOSIX and GUMSTIX, the fine weight balancing system, the sensors, and the controls. Following is a short introduction to each for completeness.

2.1 Platform Structure

Upon receiving the project, the platform of the PRWP was comprised of an aluminum structure, as depicted in Figure 2.1. The structure had two shelves for the purpose of mounting hardware, while the actual structure utilizes a rail system to house a rough weight balancing system. Below the structure, Figure 2.2, was the male portion of

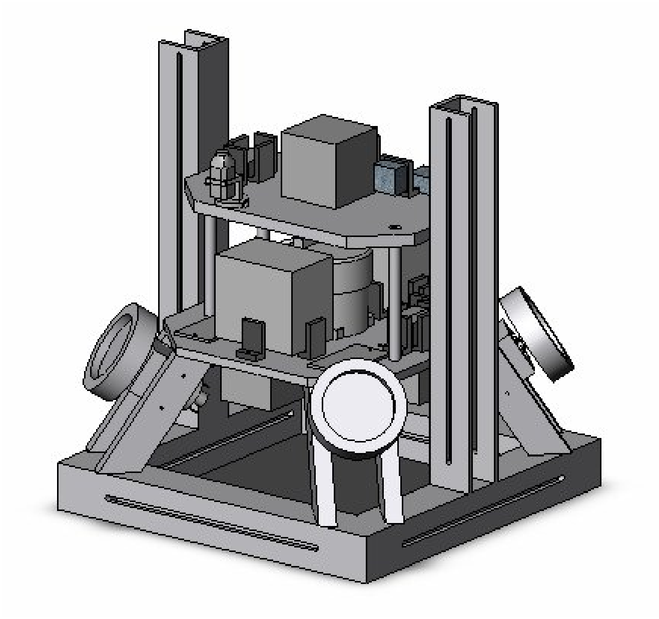


Figure 2.1 SolidWorks Model (Isometric)[3]

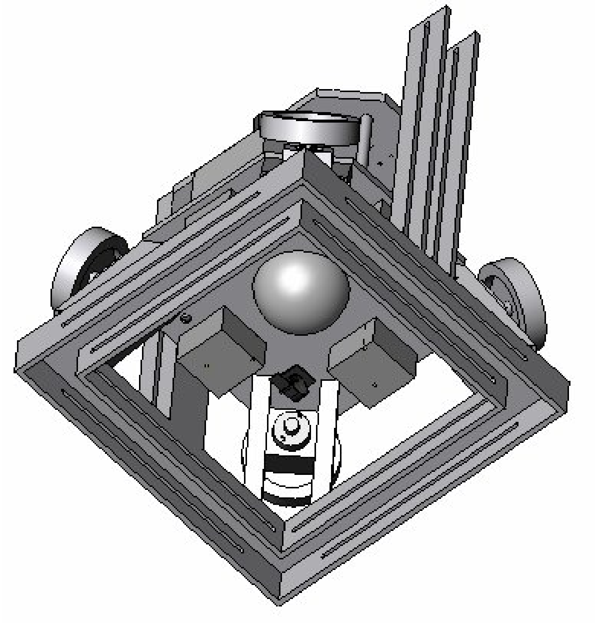


Figure 2.2 SolidWorks Model (Bottom)[3]

the spherical air bearing. The overall purpose of the structure was to act as a component hub for the entire system.

The structure housed several components throughout. On the middle shelf: two 12V-5Ah batteries, an eventual Northrop Grumman LN-200 Inertial Mass Unit (IMU), four finely machined reaction wheels, and the x-axis component of the fine balancing system were bolted to the flat surface. Furthermore, the top shelf housed three MEMS gyroscopic sensors and the y-axis component of the fine balancing system. The base of the PRWP acted, and still acts, as a supporting system for the electrical circuitry, which can also be seen in both Figures 2.2 and 2.3. Additionally, the z-component of the fine balancing system connected both the top and bottom shelves of structure vertically. Finally, four reaction wheel systems comprised of a finely machined cylindrical mass—each with connected via shaft to a Faulhaber 3863-24C DC motor, and an adjoining IE2-64 encoder which were each connected to the slanted corners of the pyramid.

2.2 Compressor, Support Pole, and Air Bearing

The compressor is in a sound reduction chamber—as seen in Figure 2.3—which provides a consistent pressure of 60 psi to the system. The air pressure introduced creates a small boundary layer between the male and female portions of the air bearing, as seen in Figures 2.4 and 2.5. The boundary layer nearly negates the effects of friction on the system at the pivot point.

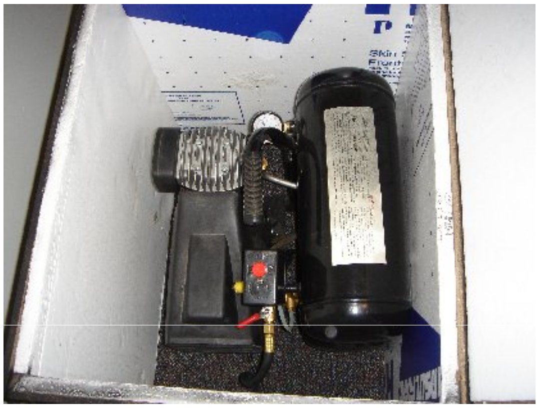


Figure 2.3 Air Compressor Utilized

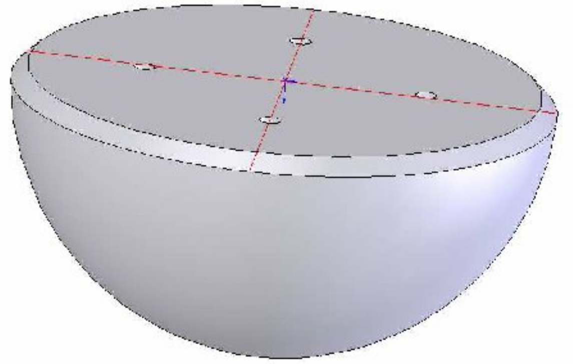


Figure 2.4 Male Air Bearing Component[3]

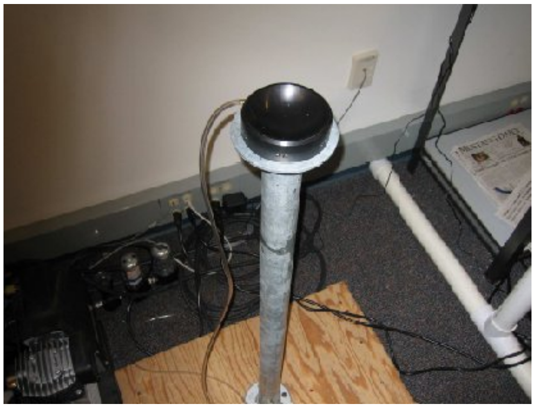


Figure 2.5 Inherited PRWP Support and Female Air Bearing Component[3]

Finally, to support the PRWP a two inch, in diameter, cylindrical post connects to both a solid surface and to the female portion of the air bearing.

2.3 Reaction Wheels

The reaction wheels, see Figure 2.6, are the main drivers of the system. By utilizing the law of conservation of angular momentum, the each reaction wheel increases and decreases its angular velocity to propagate an equal, but opposite, force through the ridged system that it is attached. Furthermore, by applying known angular rates to each individual wheel, a corresponding reaction can be calculated for the overall system. In theory, three reaction wheels—one perpendicular to each plane—could provide a momentum exchange to the x-, y-, and z- axes; thus applying directional forces to each.

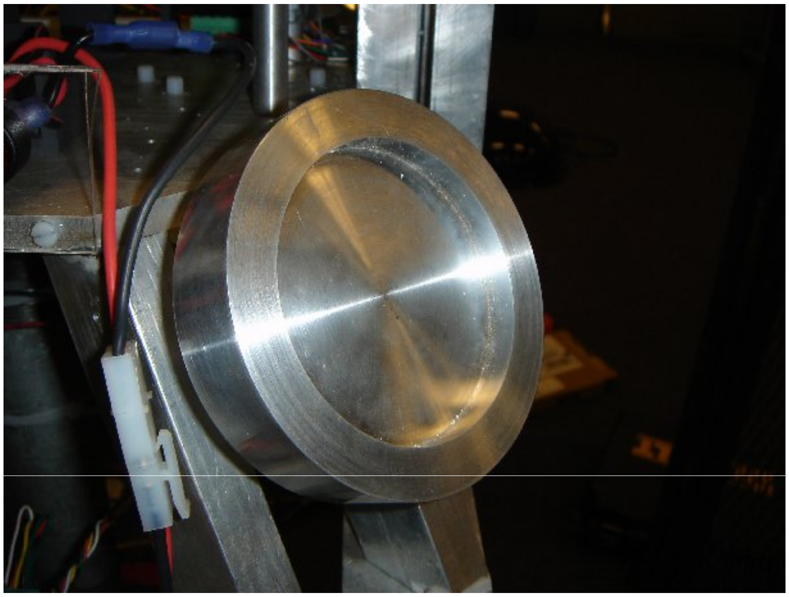


Figure 2.6 One Reaction Wheel

For the PRWP, the reaction wheel concept has been extended to enhance control authority and saturation times, while additionally adding redundancy to the system. By using four reaction wheels a redundant wheel is added into the system. By inclining the wheels, see Figure 2.7, control authority can be fine tuned to maximize angular

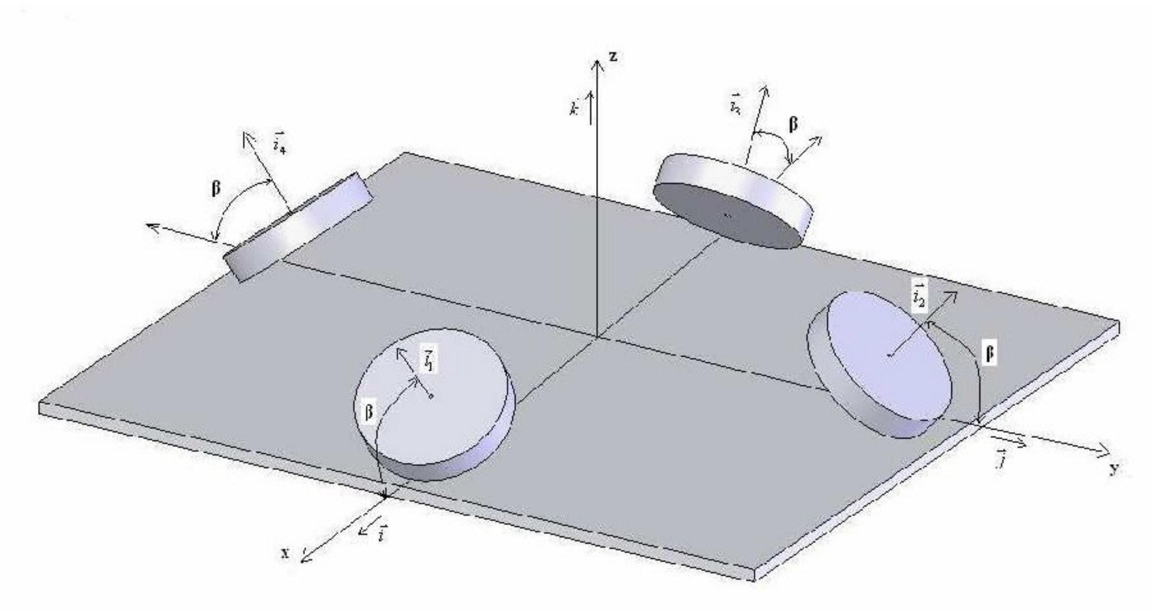


Figure 2.7 Reaction Wheel Pyramid[3]

momentum exchange on one or any of the axes. These inclined configurations are often referred to as a four reaction wheel pyramid configuration. For the PRWP the reaction wheels are sloped at 63 degrees from vertical. Although the optimum configuration for equal tri-directional control is 53 degrees, the higher angle value allowed for more z-axis control authority which assisted in holding the platform upright.

2.4 ROBOSTIX and GUMSTIX

Onboard data handling management occurs by utilizing both a ROBOSTIX, Figure 2.8, and GUMSTIX, Figure 2.9. The ROBOSTIX collects, modulates, generates and routes the information from the PRWP to the GUMSTIX which then

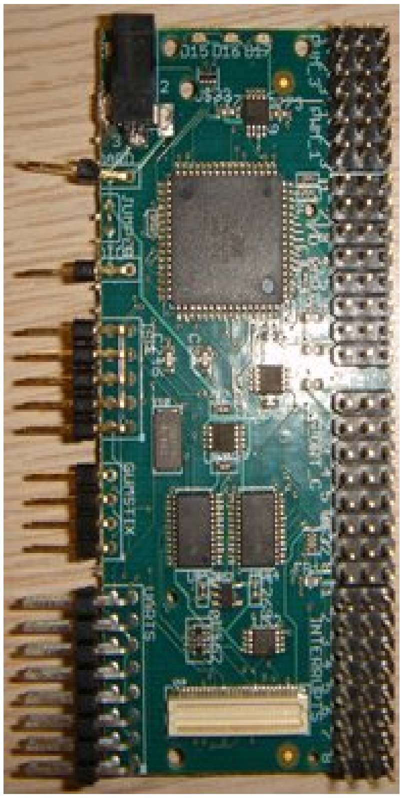


Figure 2.8 ROBOSTIX Utilized

Figure 2.9 GUMSTIX Utilized

transmits the signals via Bluetooth to the user, and vice versa. Both the ROBOSTIX and GUMSTIX operate on 5 volts. The ROBOSTIX offers six pulse width modulated outputs, eight 10-bit analog to digital converter (ADC) ports, four all purpose digital I/O ports, and a 400 MHz processor.

2.5 Weight Balancing System

The weight balancing system on board the PRWP is comprised of two separate systems: a manually adjusted rough balancing system, and an automated fine balancing system. The manually adjusted system consists of six counter weights which are approximately adjusted to translate the CG of the PRWP to approximately within one centimeter over the air bearing pivot point. The fine balancing system then further uses a control algorithm, motor driven actuators, three uni-axial guiding tracks, and three similar counter weights. The fine balancing system performs a series of commands conjoined with a control law to center the CG above the pivot point. Upon centering the CG, the fine balancing system returns a calculated platform inertia matrix such that the reaction wheel controls can be retuned accordingly.

2.6 Sensors

The PRWP was comprised of several sensors which include: three uni-axial MEMS gyros, a camcorder and an IMU. The MEMS gyros, Figure 1.11, measure and record the angular rates of the platform. The camcorder is used as a SST and is used to visually deduce the current location of the PRWP. Finally, the IMU will output an angular quaternion of the platform, with extreme accuracy. Other sensors will need to be developed for the PRWP to become fully operational.

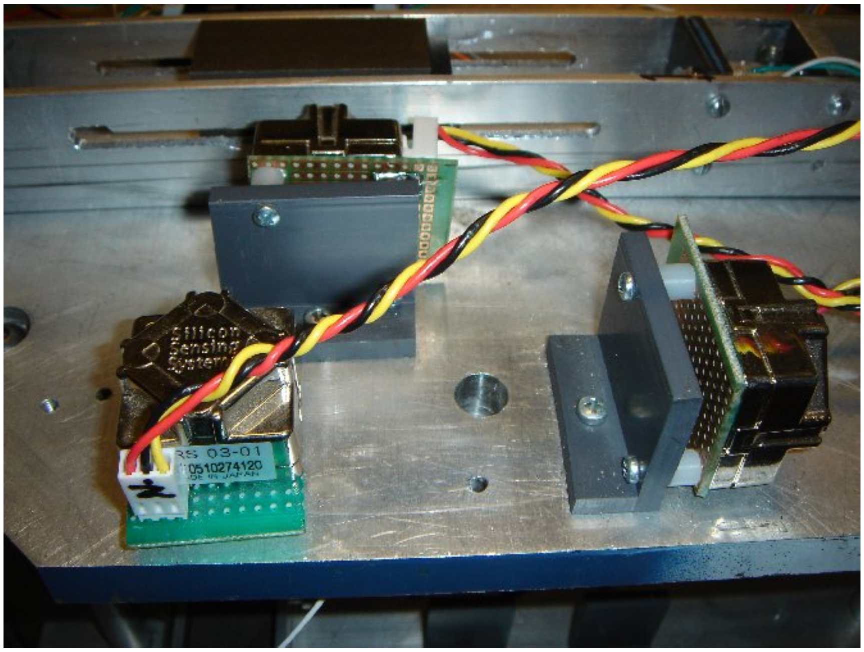


Figure 2.10 MEMS Gyroscopes

2.7 Control

The controls onboard the PRWP enable the ability to point at specific locations by utilizing micromanaged moment exchange between the reaction wheels and the PRWP structure. Through the on board sensors, the control algorithm accounts for the dynamics of the PRWP, calculates a currently quaternion, compares that quaternion to that of a control target quaternion, and calculates a wheel output to increase/decrease the wheel velocity to approach the target orientation. The entire control algorithm is parsed such that the SIMULINK model operates in real time. Although the modified version is discussed in the upcoming chapters, a detailed explanation on the original control law can be found in Carson Mittelsteadt's thesis titled "Results on the Development of a Four-Wheel Pyramidal Reaction Wheel Platform".[3]

CHAPTER 3: REQUIREMENTS AND CO-PROJECTS

3.1 Primary Requirements

The primary requirements for this project are as follows: 1) wiring and power should be better managed and distributed throughout the platform; 2) sensors should exist such that system identification and real-time controls can occur; 3) z-axis controls shall be shown to be possible, and 4) all sensors should maintain most data after passing the ROBOSTIX ADC.

3.2 Requirements Flow-down

3.2.1 Sensors

To meet the requirements corresponding to the sensors, several modifications and additions will be made to the current PRWP. Three differing types of sensors should be introduced into the system including: 1) circuitry geared towards tracking each of the four reaction wheels spin rates and directions, 2) accelerometers will be selected for use by system identification and 3) MEMS gyroscopes need to be slightly modified to accommodate for wireless operation. Following is a short list to introduce the requirements for each sensor.

The first and most complicated sensor needing to be introduced involves hardware which tracks and delivers wheel rates to the on-board ROBOSTIX. The primary purpose of this hardware is to remove the Nyquist limitations resulting from the previous iteration of this project as performed by Carson Mittelsteadt.[3] As stated in Carson's report, the limitation only allowed for angular measurements of up to approximately 480 radians per second before the signals became sporadic. Because only lower wheel rates could be read, the wheels would saturate quicker, reducing control durations. Therefore, the hardware should be designed such that by incorporating it, the problem will be averted. Furthermore, the hardware should deduce wheel direction and wheel speed and combine both pieces of information into a single signal for later modification. The sensing hardware should also be incredibly flexible to control frequency, allow the user to eventually operate the PRWP as desired. Additionally, the hardware should observe the limitation of ADC ports on the ROBOSTIX by use of the previously described switch board circuit. Finally, the hardware should allow for proper connection to each motor encoder.

To allow for system identification an accelerometer will need to be incorporated to the system. The accelerometer needs to measure the x-, y-, and z- axis components of tangential acceleration with respect to the pivot point of the PRWP. The output should then be easily modified to allow for an angular acceleration to be calculated. The final output should then be fed into an ADC port to be accessed by the system identification SIMULINK simulation. Finally, the accelerometer should have a voltage to 1g translation of at least 0.5 volts per g to allow for higher resolution on its measurements.

Although gyroscopes have previously been incorporated onto the PRWP, slight modifications should be made to allow for some changed incorporated into the system. The primary change to the gyros shall involve the wiring due to inadequate wire connections and improper wire insulation. The gyros must continue to measure angular velocity by utilizing angular momentum to spin an interior sensing system. Additionally, the sensor should be properly calibrated. The gyro sensing system should also be readjusted to ensure proper alignment of each axis. Furthermore, the gyros must continue to operate under a +5V power supply.

Finally, all sensors must deliver a signal that lies within in the ROBOSTIX ADC reading capabilities—a voltage between 0 and +5V with respect to the PRWP ground.

3.2.2 Sensor Routing Components

To acuminate for the large influx of data to the ROBOSTIX, several signal routing components must be considered. It is necessary that all proposed sensor signals (not the IMU or SST) shall be pushed through one of the eight ADC conversion pins through proper signal routing. Signal routers must also operate on 0 and 5V rails. Electrical disturbances of the original circuitry must also be avoided and the complexity simplified. This updated “main board” should also act as a power hub which provides a 5V regulated source to the PRWP from a 14.8V battery which also provides 6.6 Ah of current.

3.2.3 Command Transfer

Because commands will be calculated from a computational hub external to the PRWP system, data must be easily sent to and read from the ROBOSTIX. To allow this, two SIMULINK blocks need to be designed: one for sending commands and the other for reading the sensors. The blocks must be user friendly and easily incorporated into any SIMULINK simulation. Furthermore, the blocks must operate at a frequency greater than one hertz for control purposes.

3.2.4 Controls

SIMULINK control systems have previously been derived by Carson Mittelsteadt; however, many modifications will need to be made to support the proposed changes to the system. New calibration techniques and data conversion methods should be designed for proper data management. Additionally, command transfer blocks must be incorporated into the system. Furthermore, all reaction wheels—instead of the inherited two—must be enabled for full and enhanced command authority. Additionally the controls must be made to act in real time. The inherited system did not act in real time, but acted at an extremely fast frequency, thus reducing the calculated error by brute force. To push the system to act in real time, a simulation parser should be incorporated such that system time will manage simulation time—enabling real-time control applications. The controls pointing accuracy should be limited by both the resolution ROBOSTIX ADC. Furthermore, the operating times of each iteration of the control loop will dictate commanded pointing accuracy. As the number of control calculation and applications increase per second, the overall accuracy will be increase. In theory iteration time is primarily limited by sensor settling times and the combined time to both send and receive data. In application the limiting factor should be introduced via the send and receive SIMULINK blocks.

3.3 Concurrent Changes

Throughout the duration of the design, application, integration, testing, and post test analysis of this thesis, one other project was simultaneously occurring. Seth Silva's thesis involving the Fine Weight Balancing—which doubles as a System Identification procedure—and his software modifications to the ROBOSTIX, GUMSTIX, and SIMULINK aspects of the PRWP.[4]

CHAPTER 4: ANALYSIS, DESIGN, & IMPLIMENTATION

4.1 Design Goals

Upon further inspection, the ROBOSTIX ADC ports work as an upper limit on input data. Overall, the ROBOSTIX will need a pin corresponding to each sensors output including: the accelerometer’s three uni-axial outputs—where each is a directional component of tangential acceleration, three gyroscope outputs—one for each axis, a +5V reference input, and all four wheels direction and corresponding speed. This totals to a necessity of 15 ADC inputs. To remedy this, a type of velocity-meter and switch board duet will need to be designed to reduce the 8 pins required for wheel directions and rates to only 1 ADC port.

Furthermore, as described in Chapter 3 the requirements for design define a direction for the design goals of this project. So in summation: a main board must be designed to reduce clutter, a velocity-meter must be design to track wheel rate and direction, the limit of eight ADC pins creates a necessity for a switch board type circuit board, accelerometers need to be selected for system identification, and the control algorithms need to be updated to support all proposed changes.

4.2 Main Board Design

4.2.1 Goals

The main board has two primary goals: 1) the board needs to act as a power regulator for the 0 to 5V components throughout the system—including most sensors and the ROBOSTIX/GUMSTIX combo and 2) the main board needs to collect/distribute signals and properly route each to and from the ROBOSTIX for processing and delivery.

4.2.2 Show Old Design

Although the board did properly route signals and act as a power regulator, the original board was cluttered with multiple boards and free wires, as seen below in Figure 4.1. Furthermore, the board contained several components which were obsolete. For example, the power regulator utilized two ICs which unnecessarily attempted to fix the voltage to 5V, when the regulator-capacitor combo already managed this task. Additionally the board was actually three boards connected together—not including the ROBOSTIX/GUMSTIX. The design had a separate board for the power regulator, a second for signal routing, and third for switch and mounting purposes—this caused excessive wiring between boards, which could be compiled into a single easily

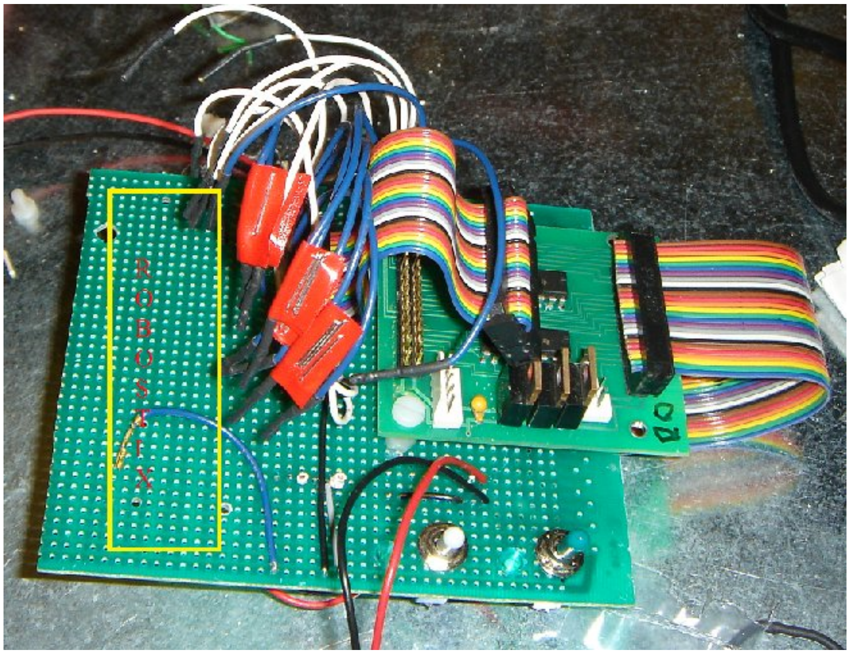


Figure 4.1: Initial Main Board (top view) without ROBOSTIX/GUMSTIX

replaceable board. Finally, the labeling on the system was poorly done and made it extremely difficult for the continuation of the project.

Beyond the cosmetic issues the board had one significant dynamic issue. The system of wires and boards created extremely high electrical disturbance to the system. Once an aluminum safety cage had been incorporated into the system, a user could simply step foot inside and their static discharge could fire one, or more, of the reaction wheels.

4.2.3 Show Proposed Design

To fix the many issues involved a new main board was created, see Figure 4.2 below. As seen the design combines the three previously utilized boards into a single board. The board was created using Pad2Pad—a program which acts as a library and building space for circuitry. Overall, the board was designed to be 4.8" wide and 3.6" in height and utilizes a duel layer surface. The top layer of the board acts as a signal routing system, while the bottom layer supplies a +5V and common to all points throughout the board. Furthermore, as seen, the board is divided into 5 sections.

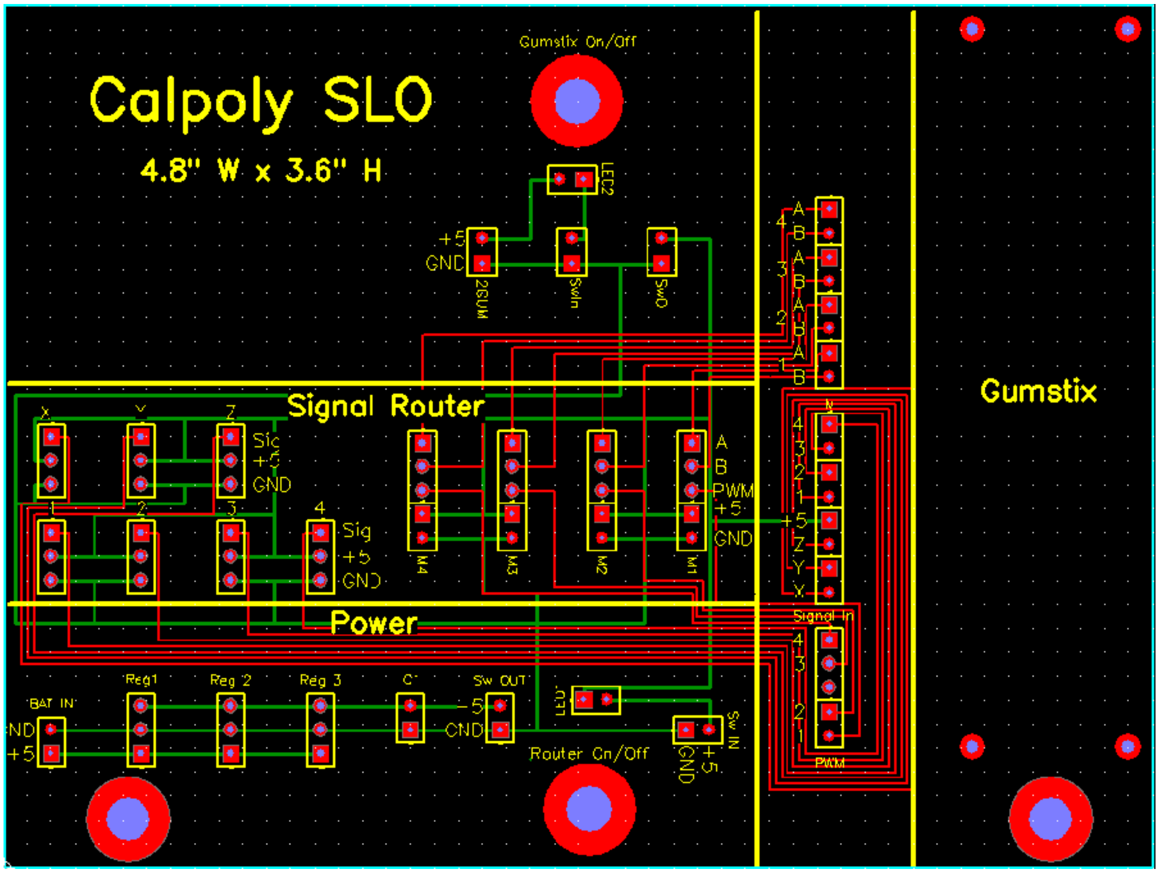


Figure 4.2: Main board circuit layout.

To help clarify the purpose of each component, a short description of power flow will be described. The routing signals are on the top of the board, as labeled as red lines. Additionally, power sources—ground and positive voltages—within Figure 4.2 are marked as green lines. Initially, in the "Power" section of the board, a rechargeable battery connects to the "Bat IN" header. As designed, the voltage is then regulated through three 5V 7805A linear voltage regulators—"Reg1", "Reg2", and "Reg3"—to ensure proper power to the board. A 2.2 microfarad tantalum comparator, "C1", is then used to further smooth out the signal to a consistent supply. The power is then delivered through a switch, "Router On/Off", to allow a user to control when power is delivered to the system. When the router switch is activated, 5 volts is routed to the X, Y, Z axis gyros, the four velocity-meters, and the direction and pulse width signals to the motors. A pulse width signal is also pulled from the ROBOSTIX towards these connection points for the motors. Following the signal routing section, power is delivered in through a second switch, “GUMSTIX On/Off”. When activated this second switch delivers power to the ROBOSTIX/GUMSTIX combo.

The new main board design also has six mounting points. The two larger points on the bottom left and right allow the board to be bolted onto the bottom of the frame of the PRWP, while the four smaller points, in the GUMSTIX section, are where the ROBOSTIX and GUMSTIX are mounted.

4.3 Velocity-meter Board Design

4.3.1 Goals

The motor outputs two square waves where each represents signal A and signal B. Both outputs have the same frequency and duty cycle to signifying motor speed; however, the waves utilize a 90 degree phase shift to represent direction. If motor A leads motor B, the direction of the wheel is clockwise, while if B leads A the wheel is spinning counter clockwise. The velocity-meter board, or TAC board, aims to convert the A and B outputs of each motor into a single signal representing both speed and direction.

Originally, due to the Nyquist Limitation on the wheels, the system forced to limit the wheels to speeds to below 490.8750 radians per second (approximately 4680 RPM) because it could not correctly detect higher wheel rates. Since the motors maximum rates are 6700 RPMs each wheel would saturate at only 70% of their maximum rates, at

20,000Hz. These limited saturations highly limited the test times of the PRWP. Therefore the final product of Carson's only utilized a single command.

4.3.2 Show Proposed Design

As previously mentioned the TAC boards, Figure 4.3, are the most complicated of the sensor circuitry. The aim of the design was to combine a speed signal and a direction signal into a velocity which swings about a steady state voltage. For simplification, the board can be thought of as divided into several sub components, as seen in Figure 4.4. The input of the TAC is on the top left hand corner and consists of four signals: A, B,

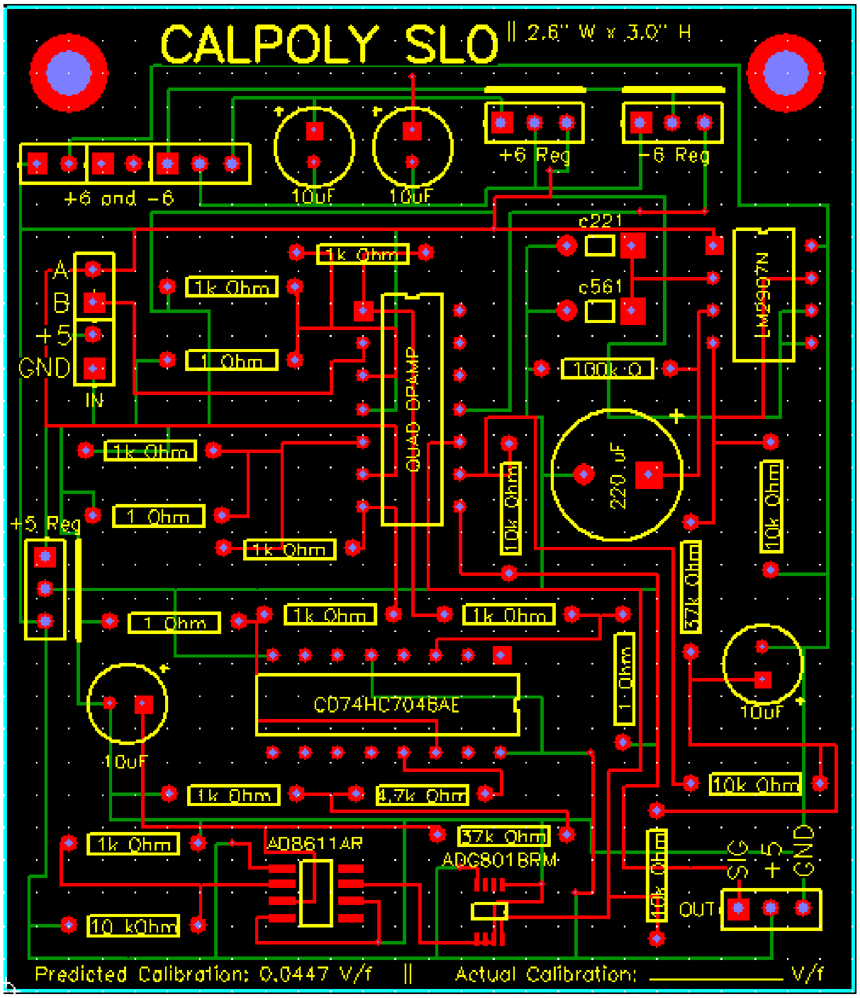


Figure 4.3: Velocity-meter circuit layout.

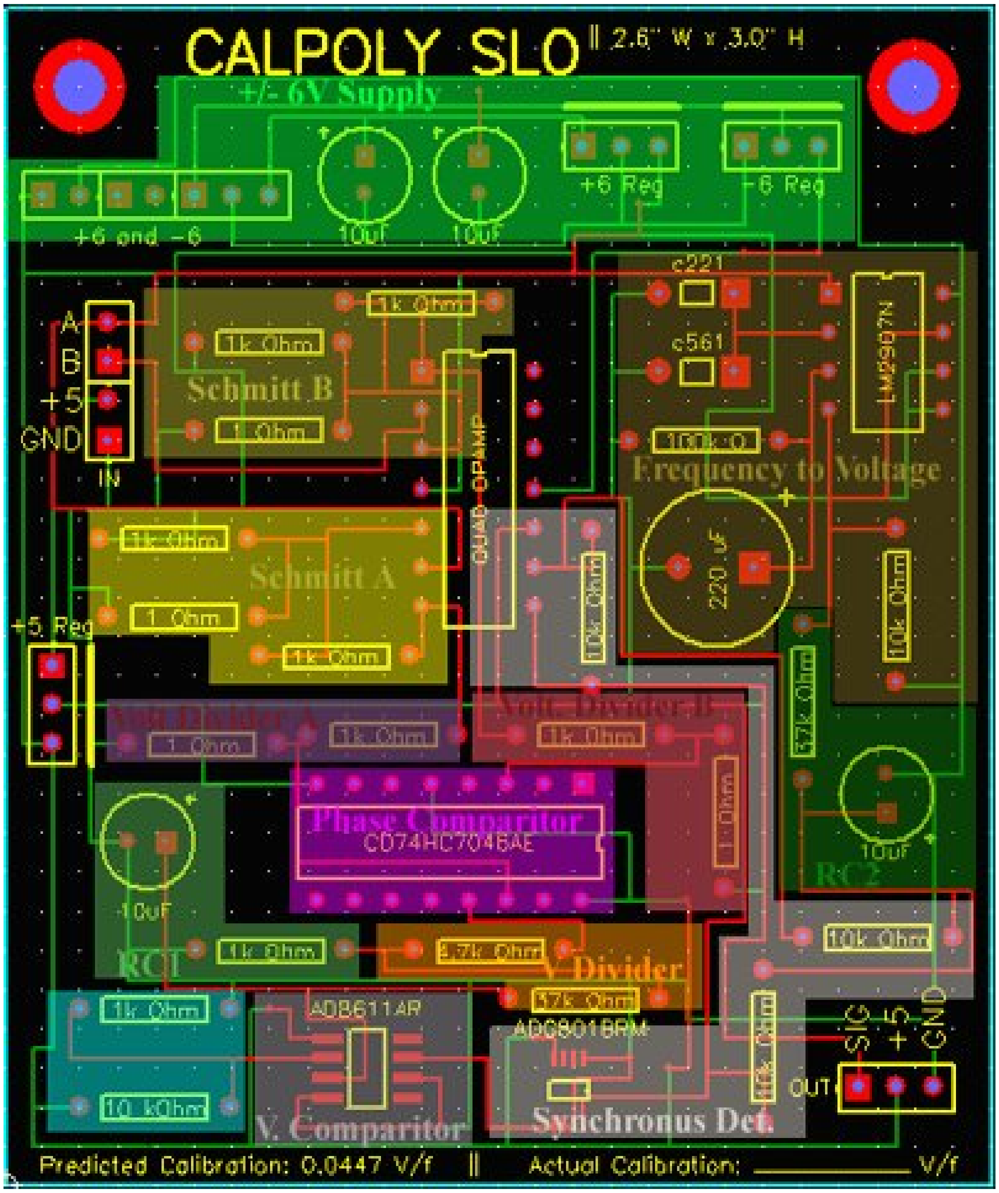


Figure 4.4: TAC board break down.

+5, and GND which are from the motor encoders. Initially, both the A and B signals are fed into a non-inverting Schmitt trigger (each using an LM327N operational amplifier) to ensure the signals are square. The amplified square waves are then pushed through independent voltage dividers which half the signal to slightly less than 3 volts. From this point the signal should route in one of two differing locations—either both signal A and B to a phase locked loop, or PLL CD74HC7046AE, or only signal A to a frequency to voltage converter (LM2907N). By utilizing the PLLs phase comparator two (PC2) pins, PLL can calculate the difference in phase between two waves of equal duty cycle and frequency. According to the data sheet, the PC2 will represent clockwise motion by outputting approximately +3.125V when signal A leads signal B by 90 degrees. If however, signal A lags signal B—also by 90 degrees—a lower +1.875V signal is produced by the PLL. The output signals are then divided to 0.536V, for counter-clockwise motion and 0.164 V for clockwise motion. An RC circuit is then utilized to remove noise from the signal before delivering it to a voltage comparator (AD8611ARM). The voltage comparator utilizes a 0.45V reference voltage to discriminate between the outputs of the PLL. The comparator's output is dependent on its respective TLL ratings. For this particular comparator the output is 3.8 V for clockwise, while it is 0.2V for counter clockwise—this signal represents direction. The output of this voltage flows into the synchronous detector. Signal A is also routed to a frequency to voltage converter before the PLL. Because the encoder utilizes the frequency to measure motor speed, the frequency to voltage converter is used to convert the motor speed to volts. Several resistors and compactors are used in conjunction to the frequency to voltage converter to produce a smoother, more noise-free signal. This signal also feeds into an RC circuit for noise removal before encountering the synchronous detector. The synchronous detector is a component that utilizes the voltage comparator's output to either activate or deactivate a normally open (NO) SPST (ADG801BRM) switch connected to an operational amplifier. If the switch is activated, or the direction is counter-clockwise the speed value is inverted, otherwise the output is left without further modification. Thus, the speed and direction are combined into a single wave with amplitude dependent on the operational amplifiers power supplies. The signal, +5 and ground signals are then passed out to the switchboard for further modification.

Because it is desired to conserve as much data as possible, the TAC boards were designed to produce a 10VPP signal which swung about 0V. To properly power the operational amplifier to produce a +5 to -5 velocity signal, a DC/DC power converter (DCH010512D) will be used to convert a 5V power supply into both a +12V and -12V supply. From there each resulting voltage is run in parallel to 10 microfarad capacitors to minimize each supply's voltage. Each voltage is then run through a corresponding +6V (L7806ABV) or -6V (MC7906CT) voltage regulator. Finally, the +6V is routed to the operational amplifiers V_{DD} pin, while the -6V is routed to the V_{EE} pin. Notice also that a +5 volt regulator is used to ensure that a consistent, but safe voltage is delivered to some of the more sensitive components.

Each TAC board also utilizes two 0.2" mounting locations. The actual location of which is along the base of the PRWP structure. Finally, notice that the final design is 2.6" wide and 3.0" tall.

4.5 Switch Board Design

4.5.1 Goals

The goal of the switch board is to mux together the four velocity-meter signals, such that all four signals are combined into one. By utilizing this board the ROBOSTIX should have enough pins available to acuminate for all integrated sensors.

4.5.2 Show Proposed Design

As seen in Figure 4.5, the system utilizes a system of switches to control which TAC is being read at a given instant. Furthermore, instead of utilizing ADC ports to transmit TAC signals, the system uses three all purpose I/O ports as an 8-bit logic for reading switch orientations, as seen in Table 4.1. The system of switches, Figure 4.6,

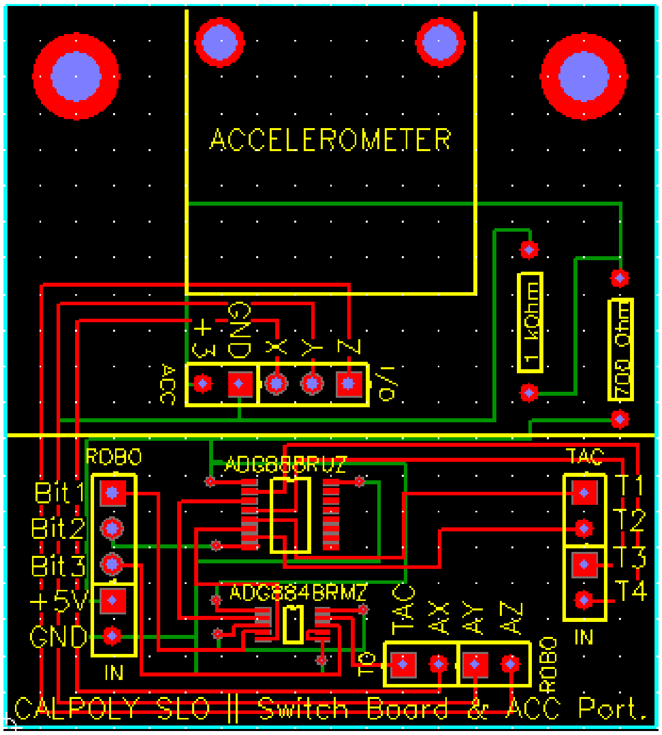


Figure 4.5: Switchboard circuit layout.

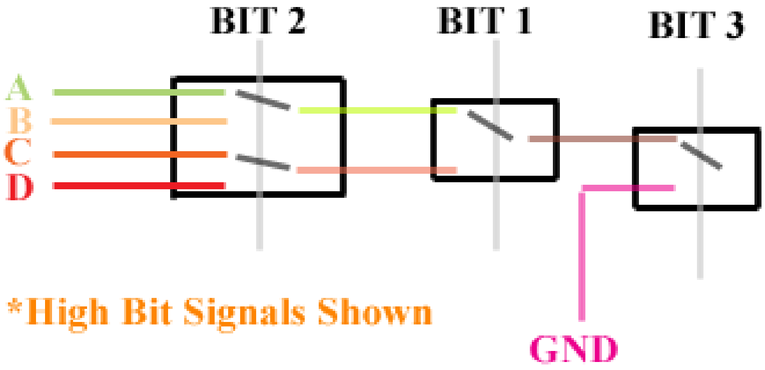


Figure 4.6: Switch Diagram

Table 4.1: 8-Bit Logic

Bit 1	Bit 2	Bit 3	Signal
0	0	0	GND
0	0	1	Wheel 1
0	1	0	GND
0	1	1	Wheel 2
1	0	0	GND
1	0	1	Wheel 3
1	1	0	GND
1	1	1	Wheel 4

utilizes a DTDP switch (or ADG888RUZ), on bit 2, to differentiate between the original four TAC signals—where a high value represents TACs 2 and 4, while a low value represents TACs 1 and 3, see Table 4.1. From there the two remaining TAC readings flow into a dual SPDT (ADG884BRMZ), on bit 1, to differentiate between the remaining two signal possibilities. The signal then passes the other end of the dual SPDT switch, which utilizes a high voltage to route the desired TAC signal and a low voltage to route a ground signal. This final switch exists to act as a buffer between signals. The bit logic is inputted via the "ROBO IN" header. Finally, the signal is passed out of the board to the main board's "1" signal port.

The switch board can also operate as the housing for a three-axis accelerometer. The accelerometer could operate at 2.7-3.3V, so a voltage divider is utilized to reduce the 5V signal to approximately 3V. The outputs are then routed to the "TO ROBO" header with the destination of the main board. Notice also that four mounting locations exist on this board, the two smaller for mounting the accelerometer, while the two larger are used to mount the board as a whole.

The output of the switch board is nearly identical to that of each TAC output, with minimal voltage loss.

4.6 Controls Design

The control algorithms, for the PRWP, had previously been developed as a first iteration solution to test the feasibility of control on the system, see Figure 4.7. The goal of this design process was to combine the sensors and the controls, while pushing for wireless capabilities and enabling the capacity to utilize all four reaction wheels. These modifications will allow for adjustable command rates, full tri-axial controllability, reaction wheel redundancy, and therefore an increased control period prior to wheel saturation. The control system should also be designed such that it operates in real time. Furthermore, the controls system should utilize the Bluetooth connection, ROBOSTIX/GUMSTIX, sensors, and reaction wheels to essentially act as a closed looped system.

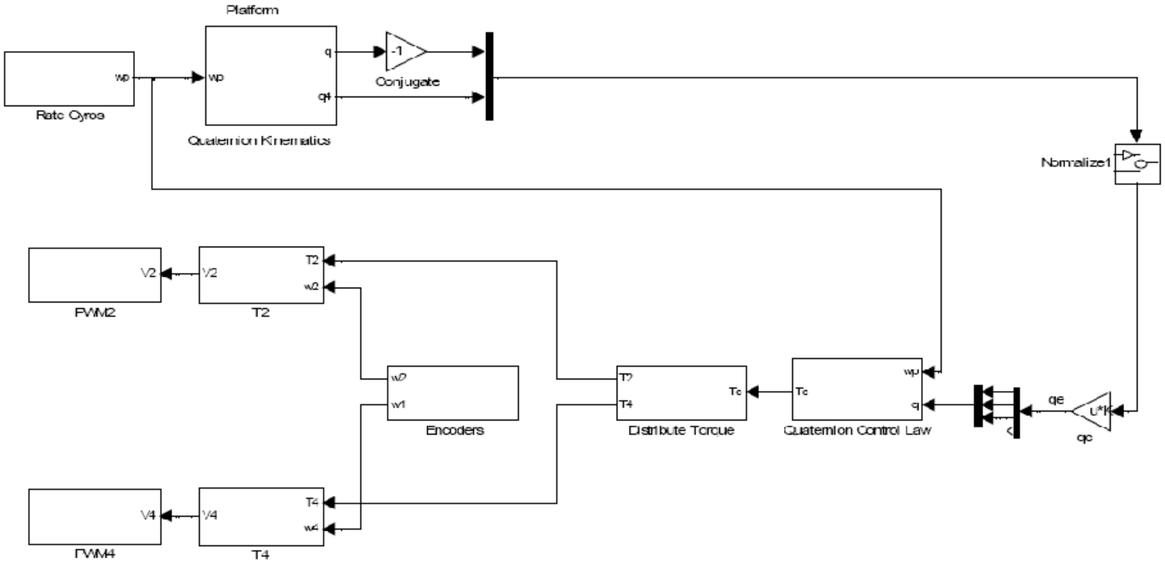


Figure 4.7: Control Diagram Prior to Update[3]

To assist in understanding the system, the upgrades will be considered in the order of signal flow, Figure 4.8 (for the complete updated version, refer to APPENDIX B.2.1). The first upgrade involves how the data is gathered and processed prior to being applied to the control algorithm. First, the "Read" block with gather all of the ROBOSTIX ADC outputs for the sensors and deliver each to the SIMULINK model as a single signal. The signal is then demuxed and sorted accordingly. The three gyroscope signals then pass through a "Gyro Conversion" block to translate the ADC signal, of each axis, into radians per seconds. The TAC board signals are gathered by utilizing the switch board logic at each iteration and

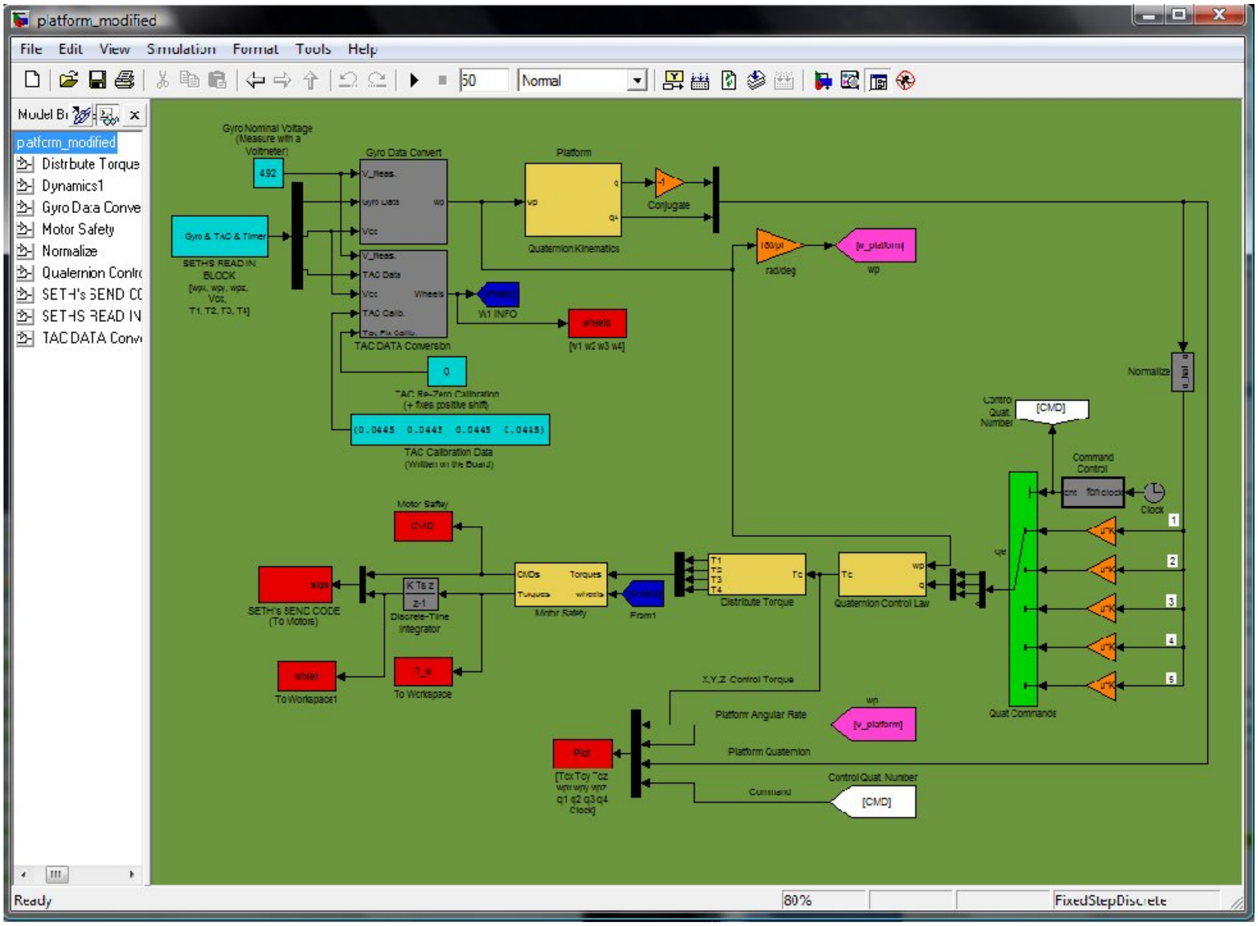


Figure 4.8: Modified SIMULINK

then passed through a block which converts each ADC reading into a wheel rate, in radians per second. Finally, because the accelerometer readings are not needed for the controls simulation, the values are terminated.

A lot of the original controls are used. The platforms angular rates, the gyro readings, are passed through the "Platform Quaternion Kinematics" block where they are combined and translated into a quaternion. From there the quaternion is inverted. The quaternion is then normalized to remove error, and then amplified by one of the five different control error matrix gain as described in Equation 4-1 below.

Equation 4-1

$$q_e = \begin{bmatrix} q_4 & q_3 & -q_2 & -q_1 \\ -q_3 & q_4 & q_1 & -q_2 \\ q_2 & -q_1 & -q_2 & -q_3 \\ q_1 & q_2 & q_3 & q_4 \end{bmatrix}$$

Each error quaternion gain results in the difference in orientation between the current location and the commanded location. A switch operates based upon simulation time to dictate which command is being issued to the PRWP. The vector portion of the resulting quaternion error and the platform angular rates are then pushed through a control law which applies control gains according to the PRWP inertia and desired natural frequency and dampening ratio. The block then utilizes the amplified signals to calculate a command torques along each of the three axes. The command torques are then distributed amongst the reaction wheels.

Another potential update is the additionment of wheels 1 and 3 to the system. This allows for full three axis control and increases control durations—the total momentum potential is doubled. The original encoder blocks simulate the physical TAC boards which were incorporated into the system. The encoder utilizes direct wiring to sample the raw data of the system via SIMULINK. Since the TAC boards are to be introduced into the system architecture and the system is desired to be wireless, the encoder block can be completely removed. This allows the reaction wheels to theoretically operate as high as 6700 RPM with accurate readings prior to each controlled attitude maneuver.

A safety block then compares the current wheel rate to the previous rate to ensure the motors will not be overstressed by the command torque difference. The calculated reaction wheel torques are compared to the current wheel rates measured by the TACs. If it is deemed safe for the wheel rate shift then the CMD output is a 0, while it is otherwise a 1 and the wheel power is temporarily cut off until the next command iteration. Finally, the wheel rates are issued to the "Send" block to deliver the signal to the ROBOSTIX, main board, microcontrollers and the wheels.

Notice that if the Bluetooth network is connected (see Appendix B), the SIMULINK controls should be automatically connected to the system via the MATLAB interface. Furthermore, it has been assured that the information is in proper form when delivered to and from the simulation. Finally, if the PuTTY client is properly set, a sample control sequence can be initiated by simply running "Simulation.m". The MATLAB file can also be easily modified to change each control quaternion, lines 102 through 142, by changing the values of qc1, qc2, qc3, and qc4. Finally, to ensure the system control gains are correct, a system identification program can be run to generate a 3x3 inertia matrix, I. Because only rough z-axis rotations will be performed, an inertia matrix is not necessary, but if full three-axis control is desired, a well calculated inertia matrix must be found.

4.6 Power Improvements

4.6.1 Battery Update

The original battery utilized for the PRWP was a 1.2 Ah, 12V, Ni-MH rechargeable battery; however, at only14.4W this battery did not provide nearly the adequate amount of power. The ROBOSTIX would not boot properly when the main board switches were activated; furthermore, the voltage for the components was below 1V. To correct for this, the battery was significantly improved to a 14.8V, 6.6Ah, Lithium Ion rechargeable battery. The updated battery provides nearly 98W of power to the system. The upgraded battery will power all non-DC motor/microcontroller components throughout the system.

It is important to note that a Lithium Ion grade battery charger was also purchased. Users must be very careful with the Lithium Ion batteries in the following manner. The batter must never be placed in series with another batter. The battery must also be charged with a charger designed for it specifically. Additionally, short circuits must be avoided. Finally, the battery must be charged with the charger for a number of hours equal to 6.6 over the charging current rate, in Ah (assuming the charger is at the same voltage). If the procedures previously described are not followed, there is a possibility for an explosion to occur—damaging not only the equipment but individuals in close proximity to the equipment.

4.6.2 Wiring

All connectors utilized for the components described pervious were hand made with 23 gage color coded, insulated, wire. A terminal crimp was applied the both ends of each wire to allow each to be locked into the according housing. Each wire was linked to each corresponding housing to form the typical cable utilized connected the system components.

The wires connecting each TACH board to both the encoder and main board power outlets are somewhat complex. The wires are created in a "T" formation. The fork allows for power to be distributed to not only the TACH board but also to the encoder of each motor. Across the top of the formation, the A and B outputs from the encoder are passed to the TACH board for analysis. Additionally, a 47ohm resistor is utilized between the +5 component and the encoder V_{cc} input to insure a current below 15mA. If a current greater than 15mA is attained, the encoder hardware may be destroyed. The wire is comprised of a single two row, six pin header—to connect the encoder, a two pin housing—which connects to the main board, and a four pin housing—which connects directly to the TACH boards.

Further wiring is comprised of individual wires with crimps on each end, and protective shrink wrap. These wires are utilized to connect each TACH board output to their corresponding inputs on the Switch Board. An additional single wire connects the Switch Board TACH output to the main board's signal in on port "1". Finally, additional single wires are utilized to connect grounds of both the lithium ion power supply and the motor's power supply.

4.7 System Structure Layout

The system architecture is rather complex, see Figure 4.9. The off-board computer acts as a central hub for the SIMULINK driven controls and the PuTTY client which utilizes

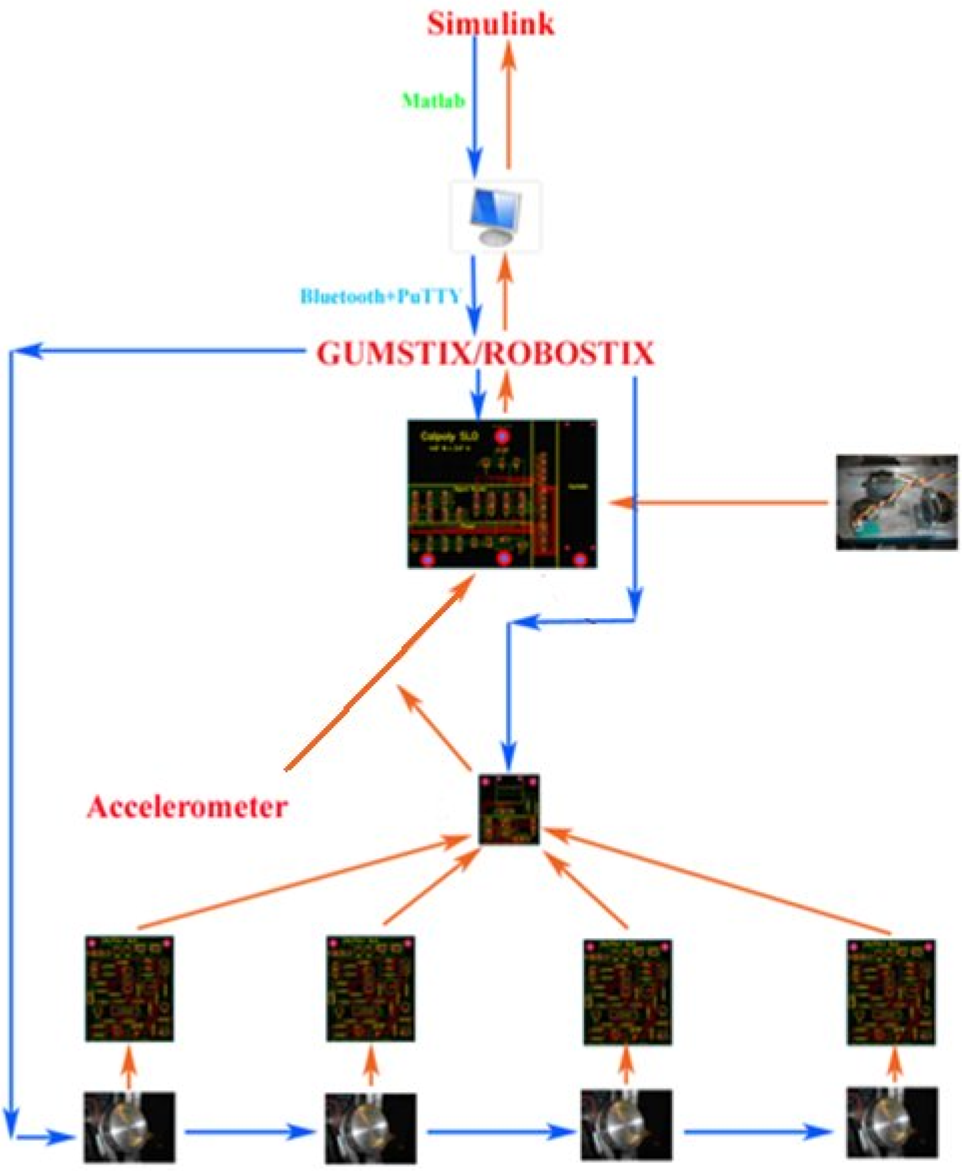


Figure 4.9 Proposed System Architecture (Blue=SEND, Orange=READ)

a blue tooth connection to both send and read signals. The signals are then routed via the main board to both the motors as pulse width modulators (PWM) to govern wheel rates. The bit logic pins are routed directly from the ROBOSTIX all purpose I/O pins to the switch board to allow for the switch board logic. The main board also supplies a 5V and ground source to these circuits as supplied by the Lithium Ion battery.

The read route delivers all sensor signals back to the SIMULINK model for control purposes. As seen, each wheel rate is calculated by each TAC board, and each TAC board signal is then compiled into the switch board. The accelerometer's x, y, and z axes readings are also sent to the switch board to be delivered to the system and then to the Accelerometer modification board in preparation for the ADC. The switch board then modifies the TAC signals via the designed switch board logic and passes the output into the TAC signal modification board as ADC preparation. Each signal is then delivered to the main board which routes each to the ROBOSTIX ADC pins. The ROBOSTIX converts the analog signals to digital signals and passes the resulting values to the SIMULINK model through the use of the GUMSTIX, a Bluetooth connection, and the "Read" SIMULINK block.

There are several different power supplies being passed between subcomponents. The TAC boards then collects four signals from the DC motor output including: a +5V source, a GROUND, an A signal and B signal. All unnamed circuit components then utilize a +5 and GROUND voltage as prepared by the ROBOSTIX. An encoder is mounted on each motor, which connects to the reaction wheel and is powered by two 12V, 5Ah power sources in series. The two power supplies are then grounded together by connecting a TAC ground output to that of the ROBOSTIX. Finally, everything is then mounted to the structure of the PRWP.

4.8 Component Construction and Implementation

4.8.1 Main Board

The main board was bolted onto top of the base of the primary structure of the PRWP, as seen in Figure 4.10. Because a large number of signals are being routed around the board Pad2Pad was used to design and outsourced for fabrication. Boards were shipped directly to the PRWP building site for soldering and integration. Before being secured to the PRWP, the ROBOSTIX and GUMSTIX combo was bolted onto the main board.

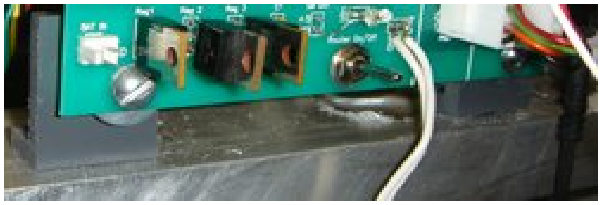


Figure 4.10: Main Board Mounting Location

The soldering of the board was also relatively easy. First the power regulator's capacitor and LEDs were added to the board, second all the headers were added, third the power regulators, and last the on board switches. The TAC and motor output headers were added into the system on the back side of the board to minimize wire clutter.

The main board wire connections consists of two-duel pin headers, seven tri-pin headers, two four pin headers, five five-pin headers, and one eight pin header—each which utilized a corresponding cable as described in the Wiring section.

4.8.4 TAC Board

The TAC boards were each bolted onto an L-shaped aluminum support which was then mounted onto the top of the base of the PRWP, see Figure 4.11. The TAC board were designed and ordered through Pad2Pad due to its surface mounted components and complexity. Because the Pad2Pad order crafts the board, no board fabrication was required.

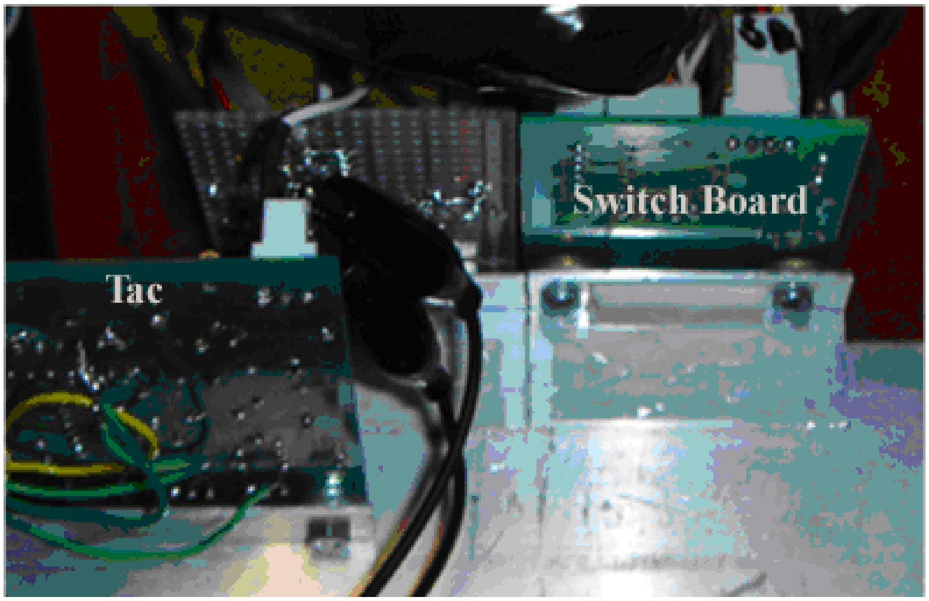


Figure 4.11: TAC Board Mounting Locations

The TAC boards had the most complicated proposed soldering procedure. First, the face mounted components were implemented onto the system dotting each solder point with a light coating of solder. Each face mounted component was then carefully placed with tweezers and a magnifying glass to ensure that each was properly oriented and mounted. The soldering iron was then pressed against the top of each pin until the solder underneath melted—securing the component. Following the face mounted components, the DIP compatible components were implemented—including the PLL, frequency to voltage converter, and the LM324N quad operational amplifier. Next, the resistors and ceramic capacitors were soldered onto the board in the typical fashion followed by the electrolyte capacitors. The headers were then applied to the system. Finally, the DC/DC power supply converter was soldered on along with the +5V, +6V, and -6V voltage regulators.

The TAC boards utilized a four pin and three pin header for the input and output, respectively—where each wire was made in accordance to the Wiring section.

4.8.5 Switch Board

The switch board was mounted to the system in a similar fashion and orientation—along the PRWP structural base—as the TAC boards by utilization of an L-shaped aluminum support. The board was printed and delivered from Pad2Pad, as described above. and therefore required no fabrication besides the design—via software—and the soldering of the physical board. It utilizes one three pin, one four pin, and two five pin headers in conjunction with the according wiring, as previously described to connect to the system.

The soldering procedure for the switch board is as follows: implement the face mounted components—as previously described, solder on the resistors, and finally add the header components. Notice also that the accelerometer is mounted to the system in accordance to the Accelerometer portion of this section.

CHAPTER 5: SIMULATION & TEST RESULTS

5.1 Main Board

5.1.1 Successes

The main board concept was successful. It combined all the components of the previous circuitry into one board which significantly reduced clutter. Most importantly the main board reduced static discharge interference. The board also properly routes all the signals, the ROBOSTIX fits on as intended, and it fits into the previously created mounting location with ease. The power regulator regulates power to 5V, especially with the additionment of the Lithium Ion battery. All components are accessed more easily since some headers are on the back side of the board while others are on the front. The switches on the main board also work as intended. However, the LEDs were ignored, because they would burn out too quickly.

A voltmeter was used to test the resistances between contact points to ensure proper connectivity between routing points. Through this method it was shown that two points were not properly connected. The fix for this will be covered in the issues section following.

5.1.2 Issues

Although the main board performed its job, it needed several modifications and changes as the project evolved. The ROBOSTIX power connector had a problem directly related to power draw. The components such as the gyroscopes would avert power away from the ROBOSTIX during its boot process. Fortunately, due to the additionment of the new battery, power draw is no longer an issue.

Another problem with the main board was that the housings were too wide for the header pins on the ROBOSTIX. This essentially meant that each was sanded down to fit accordingly. This was easily fixed by ordering and using narrower housings.

The labeling on the board became incorrect as the board evolved. The original headers for the TAC inputs were modified to also include the potential for the use of an accelerometer. Another labeling issue was created during the original printing of the board. Unfortunately, the +5V and GND battery inputs on the main board are switched such that +5V is actually ground and GND is actually +5. This issue was on the Pad2Pad side as original design has the points correctly labeled. This was easily fixed by manually relabeling the points.

The main board had two faulty connections. Motor 3 had a faulty connection to the PWM so a wire was soldiered between that and the corresponding ROBOSTIX input. Additionally, wheel 1's B directional output was incorrectly attached and needed to be soldiered to the corresponding header pin from the ROBOSTIX.

The power regulators on the main board experience high temperatures. Because the initial regulator is significantly reducing the voltage of the battery, it can easily burn a user, and damage the regulator itself. More importantly however, when the first regulator becomes overheated it allows voltages larger than 5V to pass into the system, potentially damaging hardware. Furthermore, the three power regulators were incorrectly connected to each other causing the second and third regulator to be easily bypassed. Finally, the high temperatures continued even when the power regulator switch was powered off. Although the system was not powered the regulators continued to regulate the power to 5V and would dissipate the batteries charge via heat. To correct this, the middle regulator was removed from the system, the lines connecting the inputs and outputs were scratched out, large heat sinks were placed on each regulator—along with thermal paste—and the two regulators were put in parallel instead of series, see Figures 5.1 and 5.2 below. These corrections allowed the load to be equally distributed and significantly lowered the temperature and reliability of the voltage regulators. To correct for the power discharge problem it was made sure that the battery was consistently unplugged and recharged after and between testing.

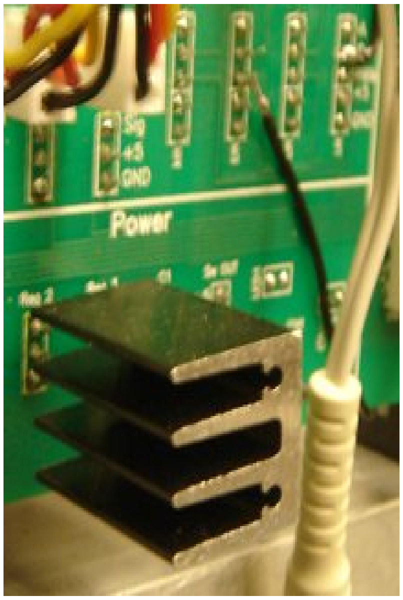


Figure 5.1: Voltage Regulator and Connection Modifications

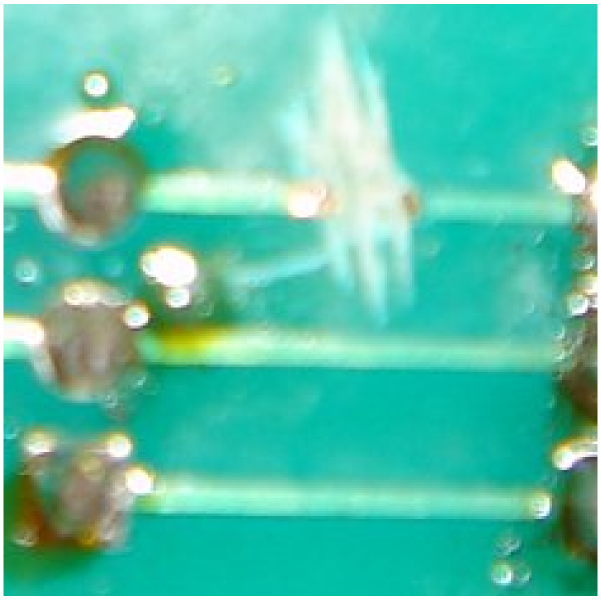


Figure 5.2: Example of Power Line a Modification

5.2 Automated Calibration, Manual calibration, and ADC Decryption

5.2.1 Automated Calibration

The purpose of the automated calibration program and simulation (Appendices C.1.4 and C.2.4) is to simplify sensor calibration. A user can utilize the automated calibration by inserting “calibrate()” to the beginning of their MATLAB code and the sensors will be calibrated accordingly.

The automated calibration is not complicated. The program samples several hundred points and finds the average of them. Assuming the system is static, the values are set to zero. The zero values are then later used by each corresponding ADC Decryption block—“Gyroscope Data Convert”, “Accelerometer Data Convert”, or “Tachometer Data Convert”—as a reference point for comparison.

5.2.3 Manual Calibration

The actual TAC results do not follow a linear trend because the microcontroller’s ability to amplify the 5V PWM signal into a 24 volt signal is not linear. The current Pololu MD01B board was designed to operate in conditions of up to 36 volts; however, due to design flaws the company suggests that voltages should not surpass 16V. Furthermore, the current regulators onboard restrict the potential of the higher duty cycles. This occurs because as the duty cycle increases, the wave’s Vpp decreases. This causes the wheel rates to plateau, see Figure 5.3. Furthermore, for PWM rates of 1000 Hz, the duty cycle values

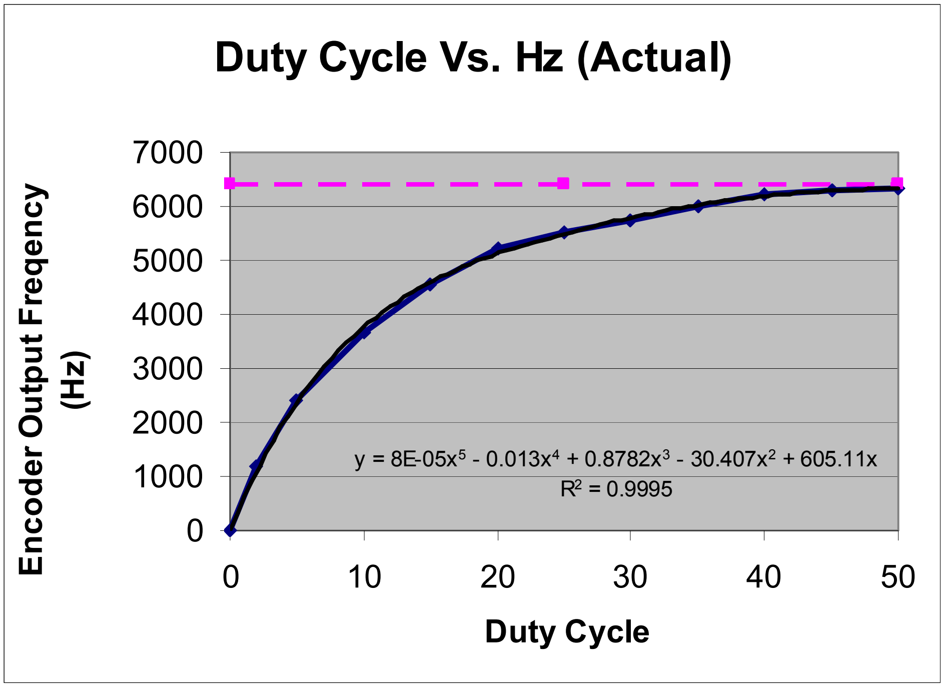


Figure 5.3: Duty Cycle vs. Hz (Experimental)

larger than 15% create relatively insignificant increases in wheel rates. For the produced results the frequency was increased to 8000 Hz, and allowed for significant duty cycles up to approximately 45%. The increase in frequency forces the microcontroller to maintain more data, so the higher 8000 Hz will be utilized. Unfortunately, by increasing the frequency higher, not significant difference in duty cycle resolution is obtained. Finally, each TAC boards potentiometer is adjusted such that each TAC follows the same trend. This is done by running each wheel at its maximum wheel rate, and adjusting the output voltage of each to be similar (currently the TACs have a nominal to maximum range of 582 mV or 119 ADC values). The duty cycle converts to the wheel rate for each wheel as described by Equation 5-1 below.

$$\text{Frequency}_{\text{Encoder}} = 8\text{E-}05x^5 - 0.013x^4 + 0.8782x^3 - 30.407x^2 + 605.11x,$$
where x=Duty Cycle

Equation 5-1: Wheel Duty Cycle to Frequency Conversion

Eventually, the microcontrollers will be replaced. The new microcontrollers should be able to handle much larger voltages, and allow enough current to properly amplify the motor driving signal.

5.2.3 ADC Decryption

The ROBOSTIX ADC converts a 0 to 5 voltage into a 0 to 1024 integer such that its data can be transferred via Bluetooth for later decryption and utilization. Because of the scope, each value of the ADC is a maximum of 4.88mV; however, to accurately monitor the actual sensor values, a nominal +5 volt source is fed into one of the ADC ports. Furthermore, since the value of the source is known, it acts as a reference for recalculating the actual conversion between an ADC value and volts. By utilizing the automated calibration, each recorded ADC value is compared to the nominal value. Furthermore, by converting the resulting ADC value into a voltage, the sensor data can be rediscovered. As seen in Appendix C.2.5 the integer can be reconverted back into a voltage and further utilize the sensors specs to determine useful data.

The gyroscopes first divide the resulting voltage by 0.02 to convert to degrees per second, and then multiply by $\pi/180$ to convert to radians per second. Finally the, value in multiplied by negative one, because the sensor reads opposite of the actual direction of motion.

The accelerometers will divide the resulting voltages (one for the x, y, z axes) by the resolution of 1 gravity to voltage. The values are then converted to feet per second-second by multiplying by 32.2. Next, the values re divided by the corresponding lever arm. Finally, the orientation of the axes is translated to that of the platform and each value is multiplied by -1 to represent actual platform movement.

A much different method was used to decrypt the switch board's TAC readings following the ADC. The wheels were each ran at various duty cycles, and the results generated for ADC reading to each duty cycle. From there, the corresponding wheel rate was compared to each ADC value, see Figure 5.4. Furthermore, notice the dead-zones on each wheel. These zones are values where the motor is unable to turn the wheels despite the applied duty cycle. Due to the angular inertia of the wheels, duty cycles below two were not resulted in a zero ADC reading. The read values were then used to decrypt the ADC values as read from the switch

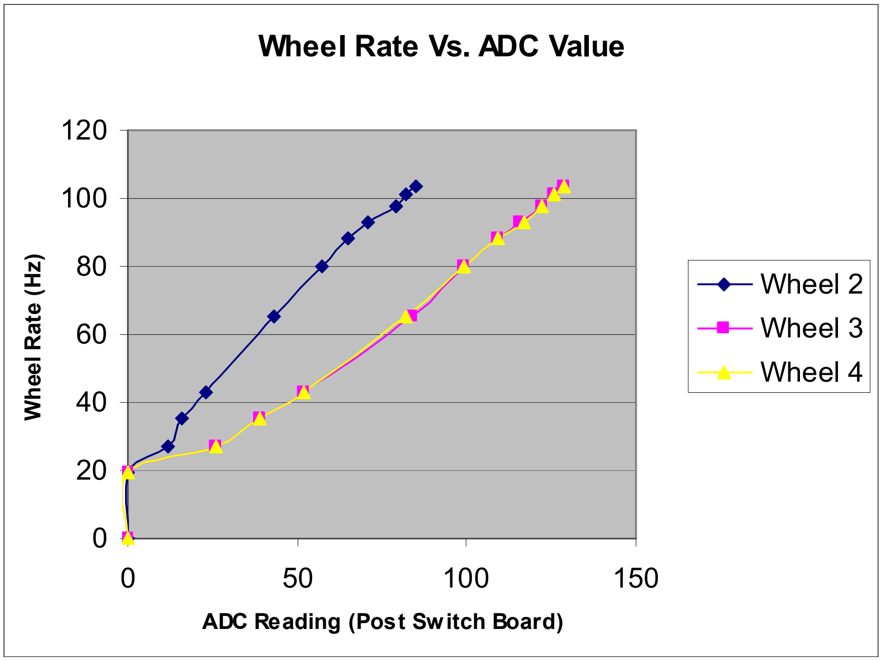


Figure 5.4: Actual ADC to Wheel Rate Conversions

board by use of Equations 5-2, 5-3, and 5-4 for wheels 2 through 4. Due to the lack of encoder, wheel 1 was ignored. The result is then multiplied by 2π to get radians per second.

$$\text{Rate}_2 = 4\text{E-09}(\text{ADC}_2)^6 - 1\text{E-06}(\text{ADC}_2)^5 + 9\text{E-05}(\text{ADC}_2)^4 - 0.0037(\text{ADC}_2)^3 + 0.0439(\text{ADC}_2)^2 + 1.9382(\text{ADC}_2)$$

Equation 5-2: Wheel 2 ADC to Rate (Hz) Conversion

$$\text{Rate}_3 = -8\text{E-}07(\text{ADC}_3)^4 + 0.0002(\text{ADC}_3)^3 - 0.0218(\text{ADC}_3)^2 + 1.4271(\text{ADC}_3)$$

Equation 5-3: Wheel 3 ADC to Rate (Hz) Conversion

$$\text{Rate}_4 = -3\text{E-}07(\text{ADC}_4)^4 + 9\text{E-}05(\text{ADC}_4)^3 - 0.0082(\text{ADC}_4)^2 + 1.0554(\text{ADC}_4)$$

Equation 5-4: Wheel 4 ADC to Rate (Hz) Conversion

The “Gyroscope Data Convert”, “Accelerometer Data Convert”, and “Tachometer Data Convert” blocks can all be incorporated into any system (Appendix C). Their input is the corresponding ADC value, while their output is the converted value.

5.3 TAC Boards

5.3.1 Overview

The TAC boards did not work as intended. Unfortunately, the received boards were printed incorrectly and contained several errors which caused some components on the board to be essentially useless. Some of the problems include an incorrect signal pass to pin 6 instead of pin 5 on the face mounted SDSP switch and the connection between GND and +5 along the top of the board. Other substantial issues involved the fact that motor 1 did not produce suitable results (it flat-lined instead of outputting a 5Vpp square signal), each board had a current draw of 250 mA instead of the needed 40 mA, the PLL was not producing any sensible results, and the DC to DC converter was not producing a desirable voltage. Therefore many fixes needed to be made to salvage each TAC board.

5.3.2 Software Modifications

The method utilized to fix the lack of direction detection was to incorporate direction detectors (Embedded Code in APPENDIX C.2.1) into SIMULINK which would check the current command, last command, and then find the direction based upon that input data and the TAC rate from the ADC value of the frequency to voltage converter. Furthermore, to convert duty cycle into RPM (see APPENDIX C.2.6), experimental data was gathered for recorded wheel rates at each duty cycle and can be found in Appendix D. The data will be utilized by the system identification simulation to better balance and understand the system dynamics.

5.3.3 Physical Hardware Fixes

To salvage the board several fixes needed to be made. First, four line cuts were made to the board: the 5V power supply line was severed, the lines were severed prior and following the point where GND connects to +5V, and the frequency to voltage converters power supply line was severed. Second, a 100 kilo-ohm potentiometer was placed on the board to allow for further optimization to the circuitry. The potentiometer will allow for the user to select a scope on the output of the TAC readings. It is currently set to be between 0 and 600mV to represent a 0% to 40% duty cycle, respectively (see Section 5.6). Third, unnecessary components were not soldered onto the board; this included the Schmitt Triggers, synchronous detectors, operational amplifiers, voltage comparators and PLL. Fourth, 23 gage wires were utilized to create connections around the unused components, see Figure 5.5. As seen several modifications were made to the bottom of the board. The

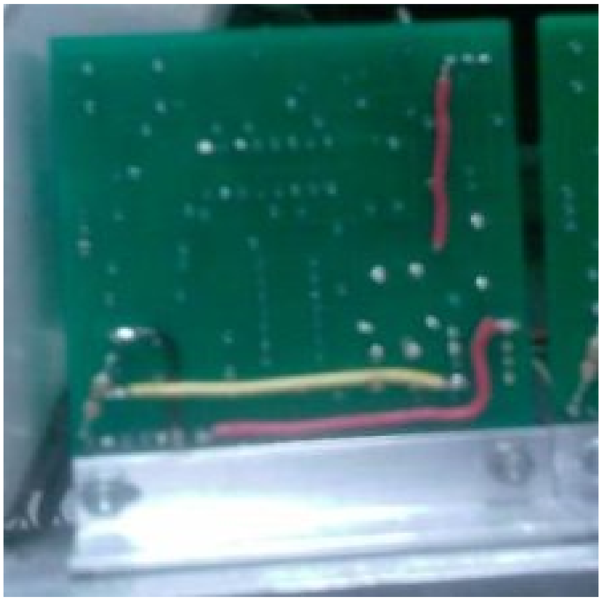


Figure 5.5: Wire updates to Each TAC (bottom)

DC to DC converter was manually sent a +5V source and forced to share its ground with that of the rest of the system. Furthermore, the frequency to voltage converter was connected directly to the positive output of the DC to DC supply and also fed the A signal for conversion. Finally, the output of the frequency to voltage converter was sent directly to the output of the board. Not seen in the figure are the top surface modifications which connect the potentiometer to the 100 kilo-ohm resistor source and drain ports. Fifth, the 220 microfarad capacitor was exchanged for a 0.01 microfarad capacitor to significantly reduce response time to the system—from an order of seconds to milliseconds. Sixth, for C1 only the 220 pica-farad capacitor was used. Finally, each board utilized a 10 ohm resistor to limit the current flow to approximately 40 mA.

5.3.4 Simulated Results

By utilizing the TAC simulation as found in Appendix C.2.2, results can also be simulated. The TAC simulation recreates each of the 13 subsections of the TAC board shown in Section 4.3.2 above. The simulation does not consider a noisy environment and further assumes optimal circuitry with no power loss. Additionally, the operational amplifiers, utilized in the simulated solution, do not check for a depth of charge beyond the potential of the rails.

As seen in Figure 5.6, for a simulated run where the wheel goes from zero to five Hz, the simulated response time is approximately three seconds. This is primarily due to the directional components of the circuit. Because these components are bypassed, this settling time should be seen to be much smaller.

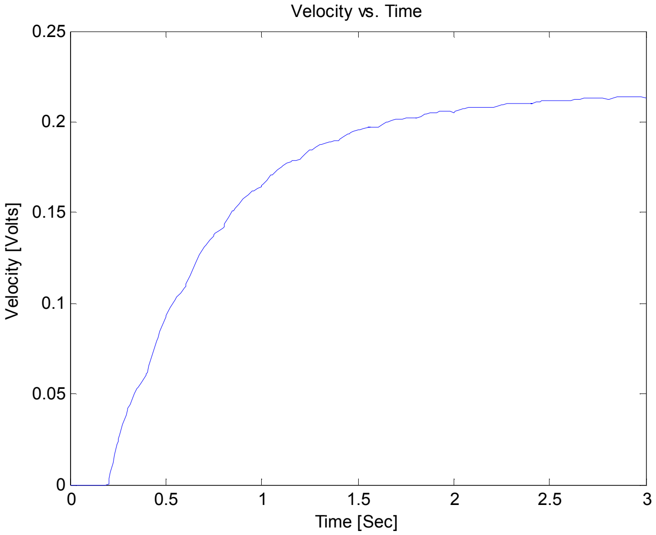


Figure 5.6: Simulated Wheel Rates for 5 Hz

5.3.5 Theoretical Results

Although the directional components of the TAC board suffered the most misprint issues, several modifications were made to salvage each TAC board. To assist in future System Identification processes, the TAC boards were still created and optimized to fit the PRWP needs. Instead of incorporating the 100 kilo ohm resistor as originally intended, a 100 kilo-ohm potentiometer was coupled with the 780 pica-farad capacitor. The potentiometer allowed the maximum recordable frequency, f$_c$, to cover a very large range of frequencies, Equation 5-5, while a 12.8 V power supply was utilized to

$$f_{c_{\min}} = \frac{1}{2*\pi*R1*C1} = \frac{1}{2*\pi*100{,}000\Omega*\left(780*10^{-12}\right)\mathrm{F}} = 2{,}040\ \mathrm{Hz}$$
$$f_{c_{\max}} = \frac{1}{2*\pi*R1*C1} = \frac{1}{2*\pi*50\Omega*\left(780*10^{-12}\right)\mathrm{F}} = 4.080\ \mathrm{MHz}$$

Equation 5-5: Calculating the maximum and minimum operation frequencies.

power the frequency to voltage converter to increase resolution for ADC conversion. The maximum output voltage can be set to 1.4336 volts if the potentiometers are maxed out, see Equation 5-6. Finally, the settling time of the circuit will be approximately 0.0001 seconds,

$$\mathrm{V_{out}}=\mathrm{fc*Vcc*R1*C1}$$
$$\mathrm{V_{maxout}}=112*12.8*100{,}000\Omega*\left(10*10^{-9}\right)\mathrm{F}=1.4336\ \mathrm{V}$$

Equation 5-6: Calculating the output Voltage.

as seen in Equation 5-7 causing little issues with lag. Furthermore, theoretically the circuit

$$\tau = \left(0.01*10^{-6}\right)F*10{,}000\Omega = 0.0001s$$

5-7: Calculating Response Time

follows a linear trend while converting frequencies to voltage, see Figure 5.7 below. Finally, the resolution of the readings to the ADC is approximately 3mV to an increment of 10 RPMs, or 0.167 Hz.

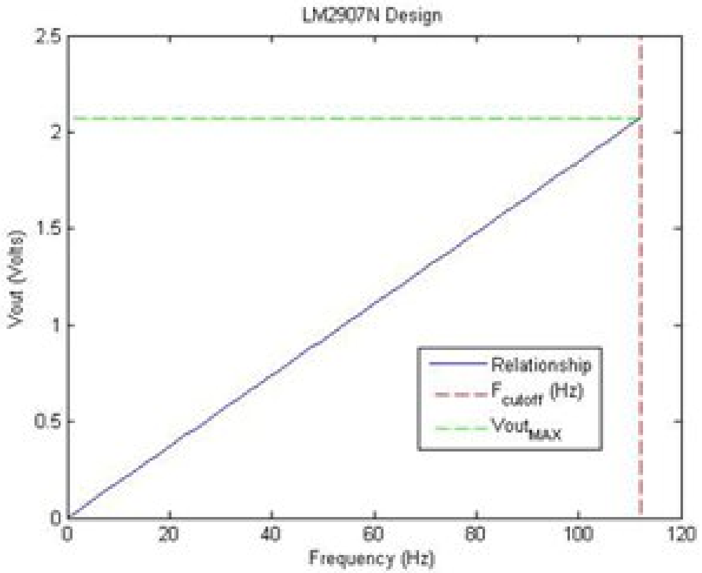


Figure 5.7: LM2907 Design Trend (Maxed Potentiometer)

5.3.6 Actual Results

The actual wheel rates following the ADC encountered minimal noise, see Figure 5.8. As seen, the wheel accelerates to its commanded rate, then becomes nearly constant

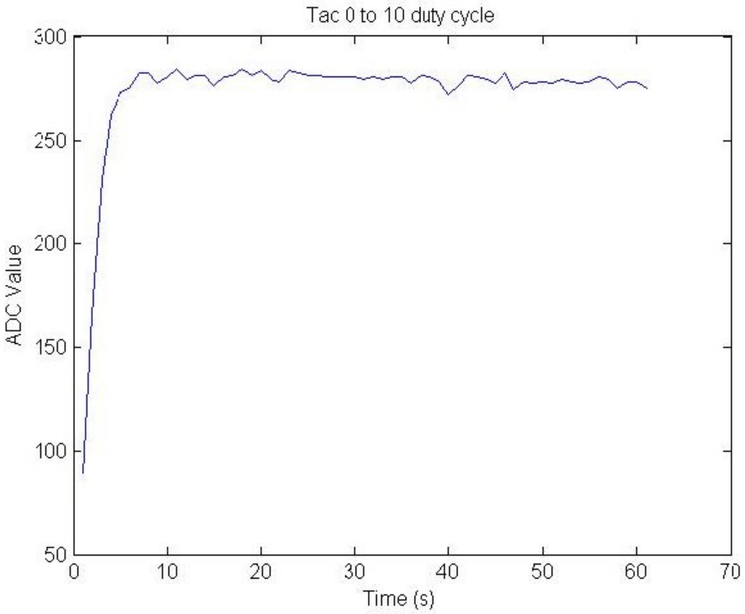


Figure 5.8: Wheel ADC vs. Time (Actual)

at the desired rate. Furthermore, as the duty cycle increases the error induced by the noise is significantly reduced by the plateau effect previously described; however, for very low duty cycles, the noise can induce errors of up to 8 Hz, or 480 RPM, see Appendix D for more details.

5.4 Gyroscopes

The MEMS gyroscopes work as intended. When angular velocity is positive, the systems output increases in voltage from its nominal value, while if the angular rate is negative the gyro output decreases in voltage from its nominal value. The nominal values of each gyro differ slightly depending on the test, but generally fall around 2.36 volts. Furthermore, the gyroscopes have a higher capability for accuracy because they measure in increments of 20mV per degree per second. This reading is roughly ten times the 2.6 mV resolution of the ADC converter. Furthermore, one ADC reading is approximately equivalent to 1 degree per second.

Despite the accuracy of the gyroscopes each encountered two problems over time. First off, the readings would receive noise which would falsely suggest a dynamic system. For example, when calibrating, the PRWP is put upright and kept in a static position with no wheel movement or angular rotations applied, but the system still picks up noise which was less than 2 degrees per second in amplitude, see Figure 5.9. Fortunately, the noise has minimal impact to system accuracy because each single ADC reading represents much less

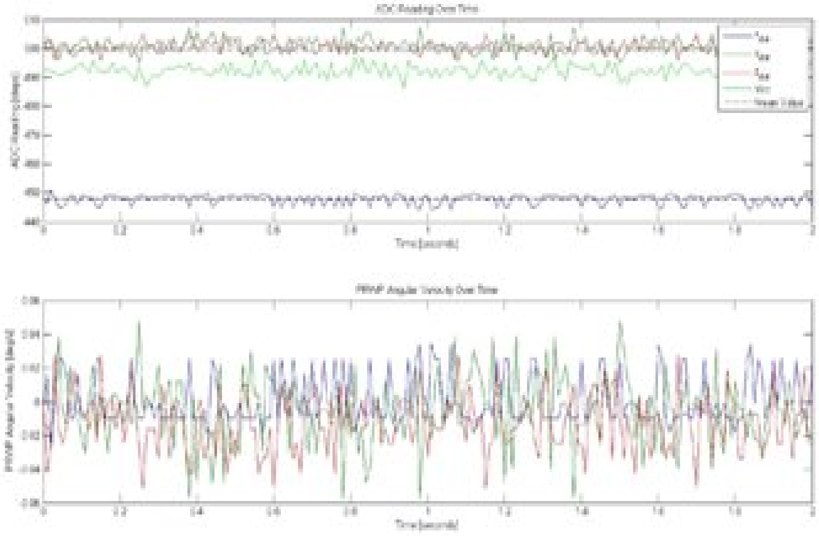


Figure 5.9: Gyroscopic Noise Plot (no applied torque)

than 1 degree per second. Another source of error derived from drift. As seen in Figure 5.10 at the four minute point the quaternion has drifted to a value of as much as 0.2 of a

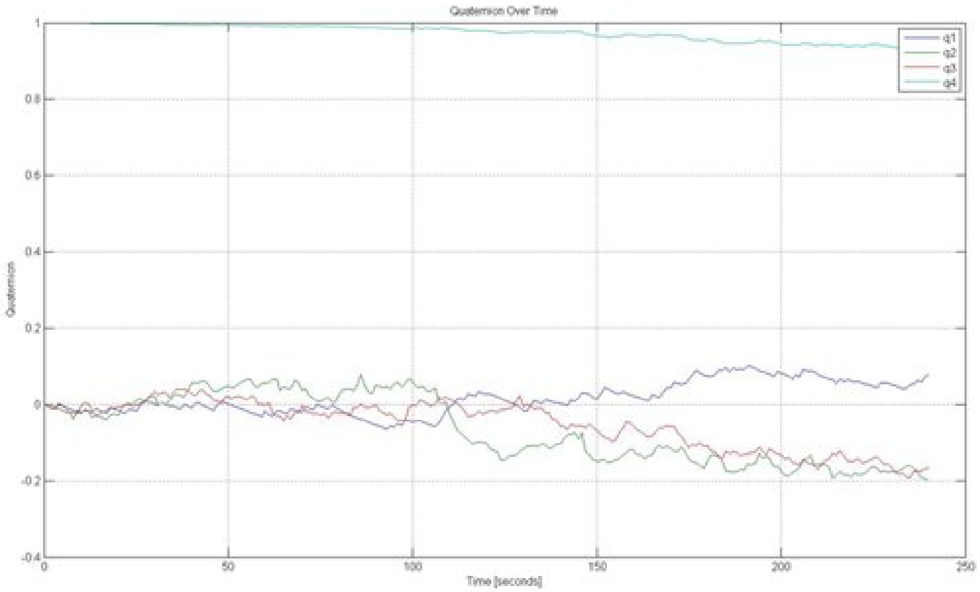


Figure 5.10: Gyroscopic Drift Plot

quaternion; keep in mind that the actual platform was stationary. This error builds over time and causes the measurements to be increasingly inaccurate. Note also that this was the worst case scenario and that perturbations can also be induced when the compressor is pressurizing.

To insure the gyroscopes accurately read data in a dynamic system, a series of manual rotations were induced to the system. The platform was rotated about each axis between 45 and 90 degrees over a one second time span and then back to its original position over the next second. The results were recorded—see Figure 5.11 for the z-axis results—for each axis and compared to the physical rotation. It was found that giving the loss of resolution—on the rotations caused by operating at 1 Hz—the results were actually very accurate.

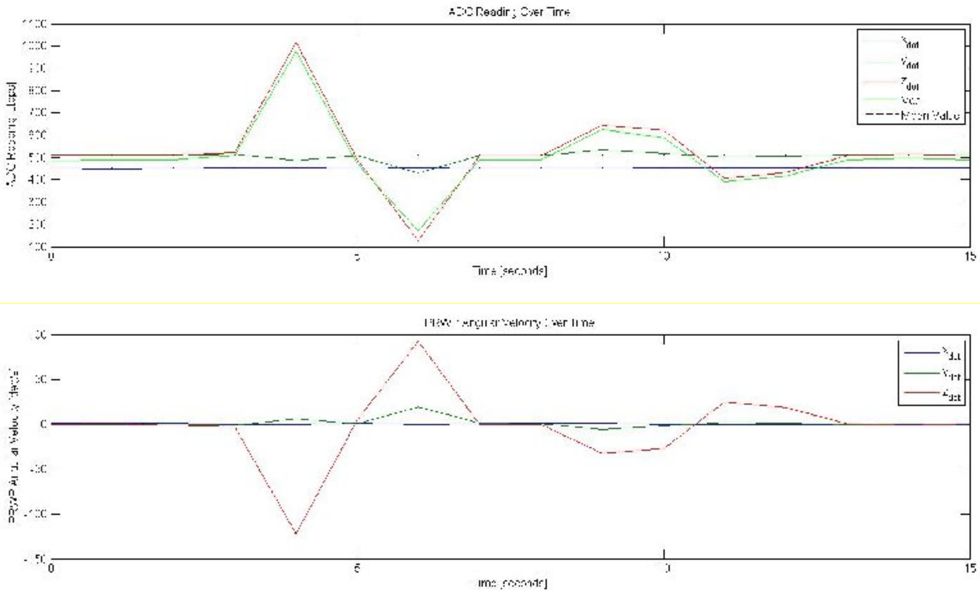


Figure 5.11: Z-axis actual sensor response for 1 Hz Operation

5.5 Accelerometer Sensor Usability Analysis

Unfortunately, the closest three-axis accelerometers utilize an excessive resolution of 800mV per g, which caused the post ADC values to be unaffected by small angular accelerations but highly affected by noise. It can be found that a single ADC reading represented approximately 25 degrees per second-second, Equation 5-8.

Notice that ΔADC is the change in the ADC reading from the static system calibration, P is the measured static voltage of the accelerometers divided by the corresponding static ADC value—usually around 0.003 volts per ADC value in resolution, and a by utilizing an approximate lever arm to the interior of the base of the pyramidal structure, (lx, ly, lz) = (2.7362 .8071 5.2953) inches with respect to the accelerometer coordinates.

$$\alpha_x\left(\frac{\text{Rad}}{\text{s}^2}\right)_{platform} = -\frac{(\Delta ADC_x * P_x)\ \text{Volts} * \dfrac{1\ \text{g}}{0.8\ \text{Volts}} * 32.174\dfrac{\frac{\text{ft}}{\text{s}^2}}{\text{g}}}{\sqrt{(l_y)^2+(l_z)^2}}$$

$$\alpha_y\left(\frac{\text{Rad}}{\text{s}^2}\right)_{platform} = -\frac{(\Delta ADC_y * P_y)\ \text{Volts} * \dfrac{1\ \text{g}}{0.8\ \text{Volts}} * 32.174\dfrac{\frac{\text{ft}}{\text{s}^2}}{\text{g}}}{\sqrt{(l_x)^2+(l_z)^2}}$$

$$\alpha_z\left(\frac{\text{Rad}}{\text{s}^2}\right)_{platform} = -\frac{(\Delta ADC_z * P_z)\ \text{Volts} * \dfrac{1\ \text{g}}{0.8\ \text{Volts}} * 32.174\dfrac{\frac{\text{ft}}{\text{s}^2}}{\text{g}}}{\sqrt{(l_x)^2+(l_y)^2}}$$

Equation 5-8: Calculating Angular Acceleration from ADC Value by Axis

As can be seen through this example, for a very small ADC value of 1, the system is highly impacted. Furthermore, because noise amplitudes can be large as 4 ADC values peak to peak, it would be very common to see angular accelerations of +/- 50 degrees per second-second when the actual platform was steady.

To remedy the problem three paths could be taken. The first option would be to significantly *increase* the lever arm (see Equation 5-5). This would reduce ADC data loss substantially; however, to measure to 2 degrees per second-second accuracy, the diagonal of the lever arms would have to be approximately 3 feet. However the lever arm length could be reduced, while maintaining the same results by incorporating a newer accelerometer which represents measurements via larger voltage values. The accelerometer should still measure approximately +/- 1.5g to prevent limitations on the measurements, but should also operate under a 5V limitation.

By coupling the systems and shifting the accelerometer(s) to the furthest point on the system, the lever arm would be (lx, ly, lz) = (14, 1, 5.2953) inches from the pivot point. The shift in the accelerometer location increases the accuracy to $(\alpha_x, \alpha_y, \alpha_z) = (13.3,\ 4.8032,\ 5.122)$ degrees per second. A new 5V 1.65 V/g accelerometer can increase the ADC accuracy on the x-axis to 6 degrees per second-second. Furthermore, the recommended accelerometer will operate on 5 volts and successfully measure the +/- 1.5g accelerations.

As a temporary fix, a different solution will be used. Because the gyroscopes were reading with a 1 degree per second-second sensitivity, any small errors induced by running the output through a discrete time integrator should be minimal (as the operating rate increases)—especially since the system is actually running in real time. Therefore, the ‘Gyro Calibration’ block has one input—the gyro measures—and two output measures including the x, y, z angular velocity and the approximate angular acceleration of the platform.

5.6 Switch Board

The switch board operates as intended. It utilizes the I/O ports from the ROBOSTIX to switch between and record ten measurements from each TAC—for each cycle— such that they can later be averaged and sent to SIMULINK. To test the switch board, prior to ordering the components a SIMULINK model was created (see Appendix C.2.3). The

SIMULINK model incorporates the simulated TAC boards and simulated signals and bit values to mux and demux TAC signals, see Figure 5.12 below. To initially test the circuit board a DC power supply was attached to each TAC pin as 1V, 2V, 3V and 4V. Furthermore, a 5V source and ground was then applied to each bit pin, and each TAC output was recorded—via voltmeter—as the switch logic was induced to the board. The boards were shown to properly switch the signals in that the correct voltage was delivered to the system.

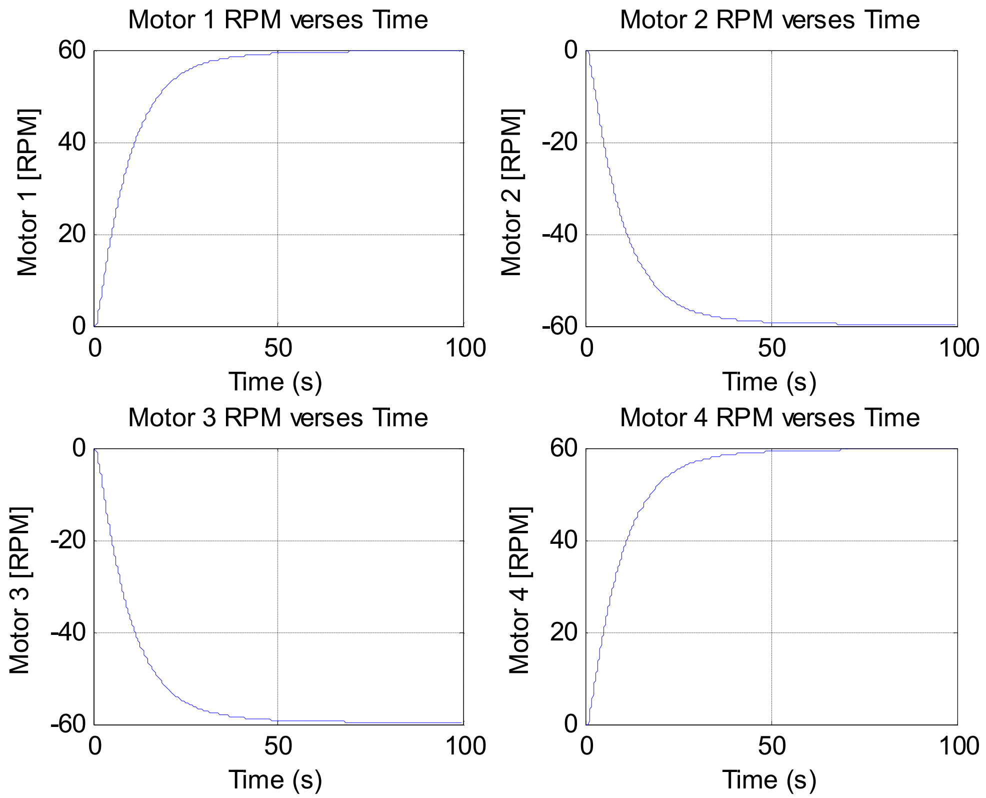


Figure 5.12: Simulated Demuxed Motor Rates for (M1,M2,M3,M4)=(1,-1,-1,1)

5.7 Controls

Because an accurate inertia matrix is currently unknown for the platform, the control algorithms (Section 4.6 and Appendix C.2.1) were ran for a single z-axis maneuver. The system initially was pointed upwards to meet the initial conditions of the platform position ([0,0,sqrt(2)/2, sqrt(2)/2] quaternion). From this point "simulation.m" was run, which began with calibration, simulated the maneuver, and finally analyzed and presented the results. For each test, all four reaction wheels shared approximately the same torques to rotate around the z-axis; therefore, wheel one was utilized despite its lack of an encoder. Additionally, because the requirements for controls were simply to depict a proof of concept, for this thesis, only a single target quaternions was considered—q_c=[0 0 0 1], from an initial point of q_o=[0 0 sqrt(2)/2 sqrt(2)/2]. The simulation was 80 seconds in duration to depict how the system damps over time. These commands were chosen such that if the appropriate inertia matrix was utilized, approximately 90 degree rotation would occur. Furthermore, the final point is the actual origin. This is preformed such that errors caused around the initial measurement would be avoided.

Upon testing the system, it was seen that the controls work as intended. The z-axis inertia was estimated to be 0.2776 kg-m^2 by trial and error as beginning with an inertia generated by a solid model as seen as follows:

$$I = \begin{bmatrix} 0.5524 & -0.0452 & 0.0008 \\ -0.0452 & 0.5531 & -0.0015 \\ 0.0008 & -0.0015 & 0.5952 \end{bmatrix}.$$

Furthermore the gains used were as follows:

$$C = \begin{bmatrix} -4.4194 & 0 & 0 \\ 0 & -4.4248 & 0 \\ 0 & 0 & -2.3809 \end{bmatrix} \text{ and } K = \begin{bmatrix} 1.1049 & 0 & 0 \\ 0 & 1.1062 & 0 \\ 0 & 0 & 0.5952 \end{bmatrix}.$$

As seen in Figure 5.12, due to the duration of the test and the low control rate of 1 Hz, the wheels quickly near saturation (just over 700 radians per second). It can be seen that the reaction wheels perform their directional changes by speeding up and slowing down. This causes the platform to rock back and forth around the z-axis, Figure 5.13, until the system stabilizes. As seen, the platform is settling to approximately -80 degrees, not -90 (negative due to a change in coordinate measure). Overall, these results can be improved by increasing the control rates beyond 1 Hz and by utilizing an experimentally achieved inertia matrix to properly calculate the C and K gains (each currently utilize a 20000 multiplier).

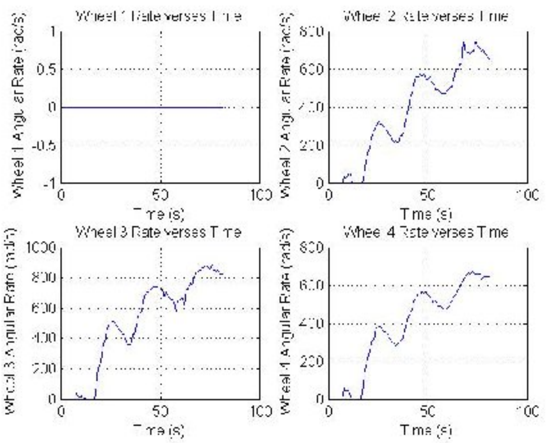


Figure 5.13: Actual Reaction Wheel Rates over Time

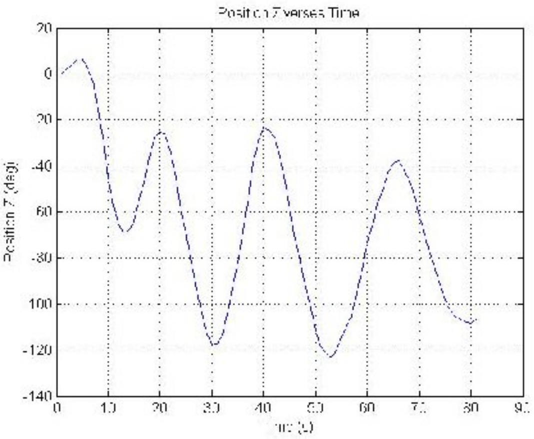


Figure 5.14: Actual Position over Time

Furthermore, the platform encountered minimal angular rates, as seen in Figure 5.14. As seen, the platform is rotating with a maximum rate of 12 degrees per second.

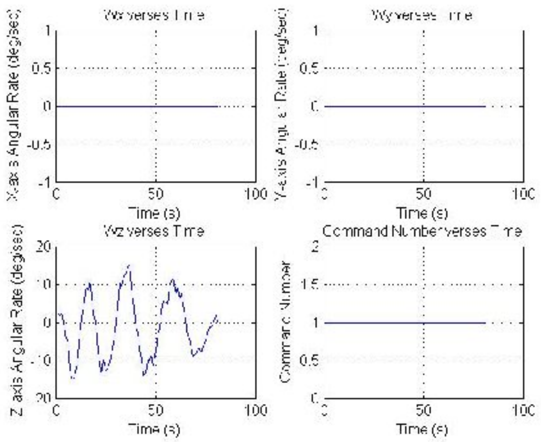


Figure 5.15: Actual Platform Angular Rates over Time

Additionally, notice that the only one command is being issued (bottom right of Figure 5.14), and the x- and y- axis rates are being ignored to isolate z-axis data. Before progressing, it is important to note that the angular rate is slowly dampening out to zero as time extends. Therefore, if a case was run with more control applications, a settling point would be achieved. Unfortunately, this cannot be performed because at 1Hz the wheels become saturated at the end of this test trial. This could be fixed by increasing the control applications per second.

Finally, the quaternions of the platform can be seen in Figure 5.15. As seen, the quaternions do not depict any out of the ordinary or unexpected behavior when compared to that of the physical system’s responses. Furthermore, notice that the actual quaternions are approaching the result for a -80 degree slew about the z-axis.

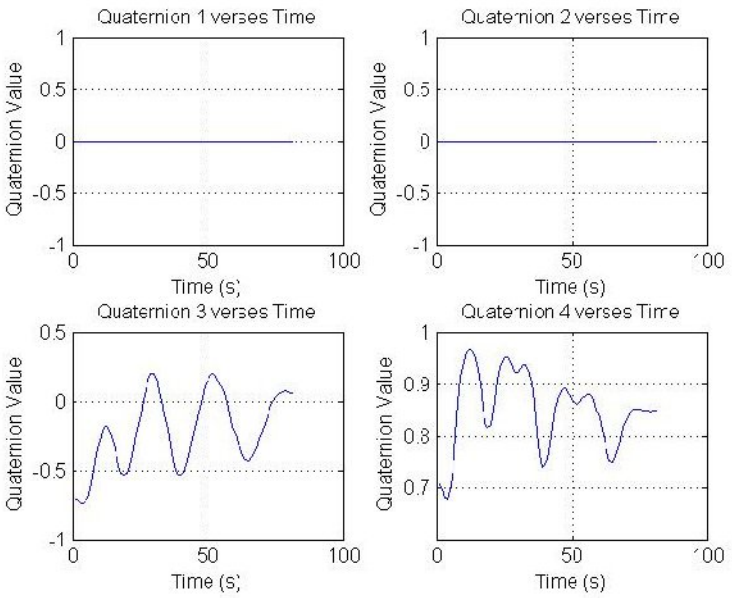


Figure 5.16: Actual Platform Quaternions over Time

After gathering the data the pointing error in the quaternions was found to be $q_{e_{final}} = [0 \quad 0 \quad 0.0681 \quad 0.9977]$. This error translates to approximately 70 degrees in the z-axis (+/- 35 degrees), as seen in Figure 5.13. As previously stated, this error would be significantly reduced as time progressed and the system settled. Furthermore, because the wheels became saturated, the actual settling time could not be deduced for the 1 Hz. Note that the pointing accuracy will be increased as the project evolves and the command rates increase and the actual inertia matrix becomes known.

CHAPTER 6: CONCLUSIONS AND RECOMMENDATIONS

6.1 Conclusion

. By utilizing the calibrated sensors, finalized software, and working hardware the platform has been shown to be controllable about the z-axis. Overall, the main board has been shown to properly route signals, distribute power, and act as a central hub with minor modifications. The 14.8V, 6.6Ah lithium ion system battery and a set of two 12V, 5Ah lead-acid motor batteries has been shown to be adequate in powering the system. The power is dissipated through the main board's system of two 5V regulators. The final product is mounted onto the base frame of the reaction wheel platform, and the ROBOSTIX/GUMSTIX combo is connected to that. Calibration was utilized via the using the "calibration()" command in a MATLAB .m file. Each calibration run takes 140 seconds, and collects one data point per second for each of the gyroscope axes, accelerometer axes, and TAC boards. The calibration utilizes a static system to create nominal values for later comparison. Despite the error created by the motor controller signal application limitation, the TAC boards have been utilized as a first iteration to accurately read wheel rates for all working wheels (2, 3, and 4), with some modifications. Each TAC board reads only wheel rate rather than wheel velocity, the directional components of each board were not in working order. Furthermore, each TAC board represents wheel rate on a scale of 0 to 811 mV following calibration. The switch board has been shown to allow for the muxing of the TAC signals into a single TAC reading for ADC conversion. Additionally, experimental data has been utilized to create a relationship between the ADC reading and wheel rates for each wheel. The gyroscopes have been properly calibrated and integrated into the system, and each signal is currently being differentiated to represent accelerometer readings. The gyroscopic data is accurate to approximately 2 degrees per second, while the make-shift accelerometer is accurate to approximately the same value with little differentiation error due to a time keeping block to limit operation rates to real time. Finally, the control loop has been shown to be able to perform rough z-axis rotations. The controls did not settle—the final measurement reflected a 70 degree error which centered on the target location. This would easily be fixed by increasing the rate of operation. Overall, the controls have been shown to be feasible to the now working wireless system.

6.2 Recommendations

Several recommendations can be made concerning the future progress beyond the scope of this document. First, a final iteration on circuitry to ensure it evolves with the project. The same circuitry could perhaps be miniaturized further than currently designed. Secondly, the star tracker (camera) and simulated star field could to be introduced into the overall system and properly implemented into the controls algorithms. Additionally, an IMU sensor should be implemented into both the physical system and controls algorithm.

Additionally, the two accelerometers should which are composed of the sensor and sensor housing—shown in Figures 6.1 and 6.2 respectively—should be incorporated into the system on top of the platforms pillars, see Figure 6.3 for approximate placement. This could be done by placing the two dual axis accelerometers at the selected location (see Section 6.1), utilizing the accelerometer calibration block (Appendix C.1.5 and C.2.5), updating the calibration constants for lever arms, and calibrating as usual. The accelerometers should be used to replace the current makeshift SIMULINK version. Although four outputs from the accelerometers can be used, only three ADC ports are available so one value for each axis will be utilized, see Figure 6.3. Finally, these components have been ordered and can be found in the ATL.

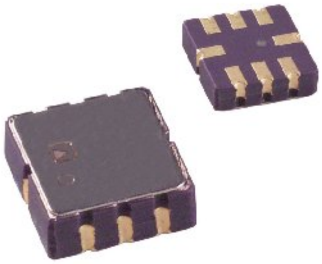


Figure 6.1: Analog Devices ADXL213AE Duel Axis 1.5V/g, +/-1.2 g, 5V, MEMS Accelerometer (8-CLCC Package)

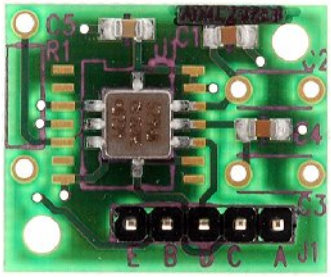


Figure 6.2: Analog Devices ADXL213EB Accelerometer Housing

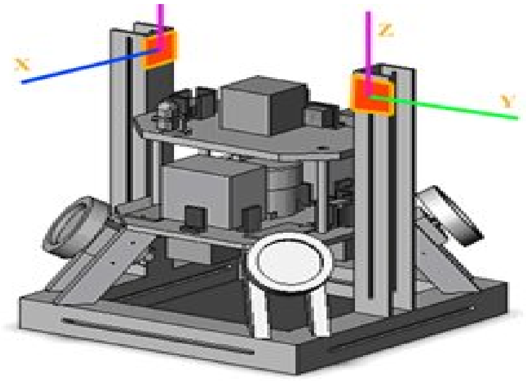


Figure 6.3: Desired Accelerometer Placement (Y-axis in Green, Z-axis in Purple, and X-axis in Blue)

Additionally, the reaction wheels should be reintroduced into the system to analyze the full capabilities of all four reaction wheels. To achieve this last portion one will need to install a new DC motor and encode to reaction wheel 1. From there an additional TAC board should be incorporated into the system for a more accurate wheel measurement for reaction wheel 1. This measurement can be easily calibrated via the calibration() function produced in Appendix C.1.4.

The onboard microcontrollers should be replaced. Since the current Pololu MD01B chips cannot handle voltages greater than 16 volts, and has strict current limitations, a newer microcontroller must be selected for each wheel. The Pololu MC33887, see Figure 6.4, can operate correctly up to voltages of 28V. This voltage is more than suitable for each reaction wheel's DC motor to operate. Furthermore, the issue with limited

Figure 6.4: New Pololu MC33887 Microcontroller

current should be bypassed by allowing the larger voltages.

By incorporating these proposed changes, and recalibrating the TACs, the system will allow for full utilization of each reaction wheel. Furthermore, the accelerometer readings will be more accurate and can replace the SIMULINK makeshift accelerometer.

Finally, by re-installing the fourth reaction wheel into the system and utilizing an inertia matrix via System Identification, full three axis controls will be possible. However, the controls algorithms should eventually be hardcoded into the ROBOSTIX to allow for control iterations much greater than 1Hz. This final step will allow the system to stabilize quicker and more accurately.

6.3 Questions and Contact Information

If any questions arise concerning the contents of this thesis, do not hesitate to contact the author via email at jeffjlogan@hotmail.com. Questions applicable are for a thesis copy, or for a piece of software that could not be found including code, SIMULINK files, etc.

LIST OF REFERENCES

[1] Schwartz, Jana Lyn. " The Distributed Spacecraft Attitude Control System Simulator: From Design Concept to Decentralized Control." Title page for ETD etd-07122004-165209. 07 July 2004. Digital Library and Archives. 10 Sep 2008 <http://scholar.lib.vt.edu/theses/available/etd-07122004-165209/>.

[2] Peck, Cavender, Mason, Andrew. "AN AIRBEARING-BASED TESTBED FOR MOMENTUM-CONTROL SYSTEMS." American Astronomical Society 03-127. 1-20.

[3] Mittelsteadt, Carson O. "Results on the Development of a Four-Wheel Pyramidal Reaction Wheel Platform." Thesis.

[4] Silva Seth. “Applied System Identification for a Four Wheel Reaction Wheel Platform.” Thesis

[5] Brogan, William L. Modern Control Theory. Upper Saddle River: Prentice Hall, 1990.

[6] Kuipers, Jack B. Quaternions and Rotation Sequences - A Primer with Applications to Orbits, Aerospace and Virtual Reality. New York: Princeton UP, 2002.

[7] Magrab, Edward B., Shapour Azarm, and Balakumar Balachandran. An Engineers Guide to MATLAB. Upper Saddle River: Prentice Hall PTR, 2007.

[8] Rizzoni, Giorgio. Fundamentals of Electrical Engineering. New York: McGraw-Hill Science, Engineering & Mathematics, 2008.

[9] Vallado, David A., and Wayne D. McClain. Fundamentals of Astrodynamics and Applications. New York: Springer, 2007.

APPENDIX A: PART LIST BY COMPONENT

A.1 Main Board

Table A.1 Main Board Component List

Type	Part	#Total	Cost ($)	Total $	DIGIKEY #
Header					
	2	12	0.34	4.08	22-23-2021
	3	24	0.43	10.32	22-23-2031
	4	4	0.51	2.04	22-23-2041
	5	10	0.59	5.9	22-23-2051
	8	4	0.86	3.44	22-23-2081
Housing					
	2	12	0.25	3	22-01-3027
	3	24	0.32	7.68	22-01-3037
	4	4	0.39	1.56	22-01-3047
	5	10	0.46	4.6	22-01-3057
	8	4	0.68	2.72	22-01-3087
Capacitor					
	C1	1	4	N/A	478-1867-ND
Voltage Regulator					
	pos 5	7805A	3	N/A	N/A

A.2 TAC Board

Table A.2 TAC Board Component List

Type	Part	Units	# per b	Buffer	# Total
Resistors					
	1	Ohm	4	2	22
	510	Ohm	1	1	6
	1	kOhm	9	2	47
	4.7	kOhm	1	5	10
	10	kOhm	6	2	32
	36	kOhm	2	2	12
	100	kOhm	1	2	7
CMOS Switch-SPST					
	ADG801BRM	--	1	2	7
Capacitor					
	220	uF	3	2	17
	10	uF	4	2	22
	220	pF	1	2	7
	560	pF	1	2	7
Zener Diode					
	5.1	Volt	2	2	12
Op Amps					
	QUAD	QUAD	1	2	7
Frequency to Voltage					
	LM2907N	F2V	1	2	7
Voltage Comparitor					
	AD8611ARM	V-Com	1	1	6
Voltage Regulator					
	7805A Equivalent	5V Reg	1	2	7
		(-6V) Reg	1	0	5
		6V Reg	1	0	5
	DCH010512D	DC/DC	1	0	5
Headers/Housings					
	4 Way	Housing	1	2	7
	3 Way	Housing	1	2	7
	4 Way	Header	1	2	7
	3 Way	Header	1	2	7

A.3 Switch Board

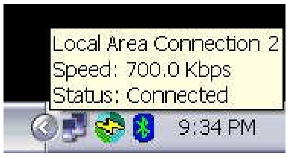
Table A.3 Switch Board Component List

Type	Part	Units	# per b	Buffer	# Total
Resistors					
	750	Ohm	1	2	4
	1	kOhm	1	2	4
CMOS Switch-SPDT					
	ADG884BRMZ	DUAL	1	2	4
CMOS Switch-DPDT					
	ADG888YRUZ	DUAL	1	2	4
Headers/Housings					
	5 Way	House	3	2	8
	4 Way	House	2	2	6
	5 Way	Header	3	2	8
	4 Way	Header	2	2	6

APPENDIX B: TEST PROCEDURE

In order to test the platforms control algorithms it is recommended that the user does the following:

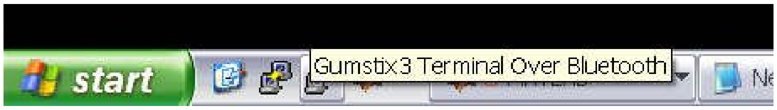
1) Plug in the power source to the main board or ROBOSTIX depending on if the system needs to be wireless or not.
2) Turn on the Commanding computer which is located near the safety cage.
3) Wait for a blue tooth connection to be made:



Local Area Connection 2
Speed: 700.0 Kbps
Status: Connected

Figure B.1: Active Bluetooth Connection

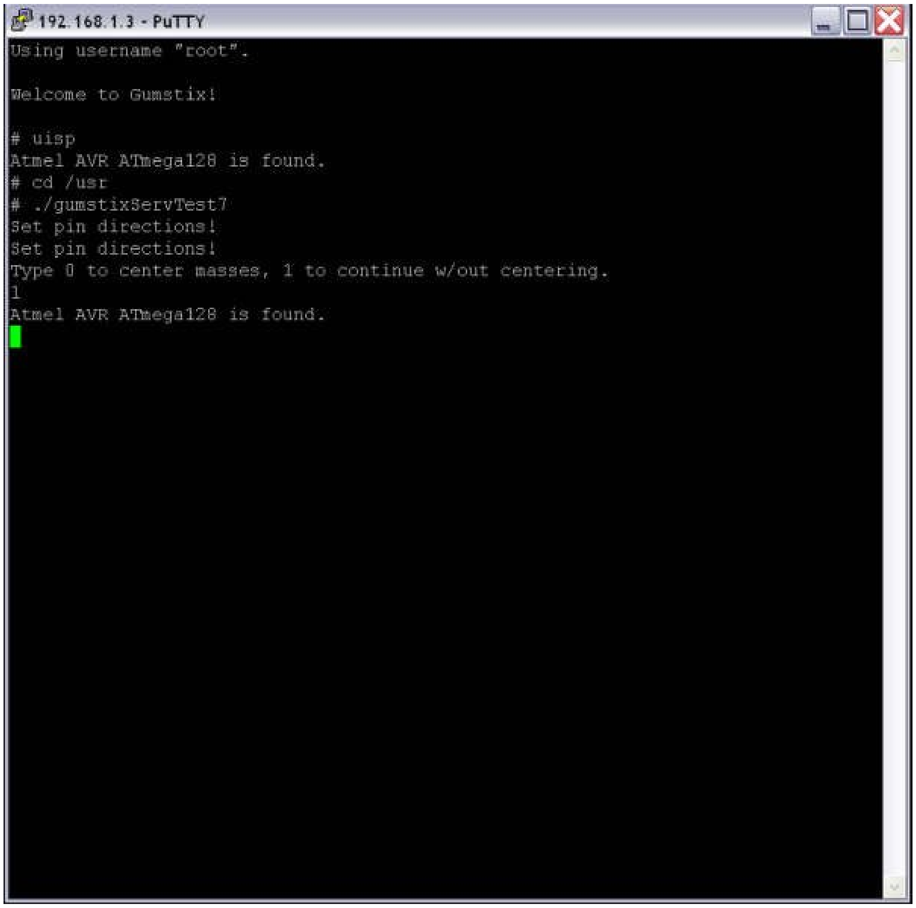
4) Upon connection open the PuTTY client:



Gumstix3 Terminal Over Bluetooth

Figure B.2: PuTTY Client Terminal

5) When the PuTTY client opens type the following:
   a. uisp
   b. cd /usr
   c. ./GUMSTIXServTest7
   d. Followed by a ‘0’ if you would like to “center” the fine weight masses or ‘1’ if you would like to continue without centering.
   e. The window should appear as follows:



```
Using username "root".

Welcome to Gumstix!

# uisp
Atmel AVR ATmega128 is found.
# cd /usr
# ./gumstixServTest7
Set pin directions!
Set pin directions!
Type 0 to center masses, 1 to continue w/out centering.
1
Atmel AVR ATmega128 is found.
```

Figure B.3: Sample PuTTY window (with active connection).

f. If errors persist start over by pressing control+c to return back to # and return to the beginning of step 5.

6) Initiate MATLAB and navigate the workspace to C:\Documents and Settings\aero\My Documents\MATLAB\PRWPCode\System.

7) Turn on the reaction wheel power by moving the large red switch—which is located above the TAC boards opposite of the main board—to ground (yes, it is backwards).

8) Close and lock the front door of the safety cage.

9) Open and run “simulation.m”.

10) The simulation will ask if you would like to calibrate. Calibration takes 4 minutes and should be run at the beginning of each set of tests. It will calibrate automatically.

    a. Type ‘y’ for yes or ‘n’ for no.

11) Observe the results.

12) Repeat as necessary or type ‘poweroff’ in the terminal client to turn the system off when testing is complete.

NOTE: If a connection is not established to the GUMSTIX turn the power on and off.

APPENDIX C: FULL CODE AND SIMULINK

C.1 Code

C.1.1 Controls

```
simtime=200;
%Ts = 0.025;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Inertias

%Wheel
outerdia = 101.65*10^-3; %m
innerdia = 76.15*10^-3; %m
shaftdia = 15.56*10^-3; %m
outerwidth = 24.62*10^-3; %m
innerwidth = 9.06*10^-3; %m
shaftwidth = 17.32*10^-3; %m
density = 2700; %kg/m^3

outermass = outerwidth*pi*(outerdia/2)^2 * density;  %kg
innermass = 2*innerwidth*pi*(innerdia/2)^2 * density;  %kg
shaftmass = shaftwidth*pi*(shaftdia/2)^2 * density; %kg

TotalMass = outermass-innermass+shaftmass; %kg

J1 = .5*outermass*(outerdia/2)^2 - .5*innermass*(innerdia/2)^2 +
.5*shaftmass*(shaftdia/2)^2; %kg-m^2 inertia of the wheel

J=[J1;0;0];

%Platform Inertia
I=[ 0.552430 -0.045171  0.000847;
   -0.045171  0.553103 -0.001499;
    0.000847 -0.001499  0.595233];

I1=I(1,1);
I2=I(2,2);
I3=I(3,3);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Quaternion control law
% zeta = -2;
% Wn = 5;

zeta = -.02;
Wn = .005;

C=10000*[I1*2*zeta*Wn, 0,            0;
   0,            I2*2*zeta*Wn, 0;
   0,            0,            I3*2*zeta*Wn];
```

```
K=10000*[I1*2*Wn^2, 0,              0;
   0,         I2*2*Wn^2,      0;
   0,         0,              I3*2*Wn^2];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%initial quaternions
q1=0;
q2=0;
q3=0;
q4=1;

global q_init q4_init
q_init=[q1;q2;q3];  %IC for integral
q4_init=q4;         %IC for integral

q=[q4;q1;q2;q3]; %for Quat to Euler transformation block

%commanded quaternions
qc1=0;
qc2=0;
qc3=0;
qc4=1;

%qcp1=[qc1 qc2 qc3 qc4]';

%%Quaternion matrix for error output
qcp1=[qc4,  qc3, -qc2, -qc1;
   -qc3,  qc4, qc1, -qc2;
    qc2, -qc1, qc4, -qc3;
    qc1,  qc2, qc3,  qc4];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

qc1=0;
qc2=0;
qc3=sqrt(2)/2;
qc4=1;

%qcp2=[qc1 qc2 qc3 qc4]';

qcp2=[qc4,  qc3, -qc2, -qc1;
   -qc3,  qc4, qc1, -qc2;
    qc2, -qc1, qc4, -qc3;
    qc1,  qc2, qc3,  qc4];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
qc1=0;
qc2=0;
qc3=0;
qc4=1;

%qcp3=[qc1 qc2 qc3 qc4]';
```

```
qcp3=[qc4,  qc3, -qc2, -qc1;
   -qc3,  qc4, qc1, -qc2;
    qc2, -qc1, qc4, -qc3;
    qc1,  qc2, qc3,  qc4];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Distributed Torque
alpha = 57;  %degrees

B = (90 - alpha)*pi/180;

%%ALL REACTION WHEEL CASE
% Aw = 0.5*[1   0  .5  .5;
%           0   1  .5 -.5;
%           -1  0  .5  .5;
%           0  -1  .5 -.5];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Aw = 0.5*[1   0  .5  .5;
           0   1  .5 -.5;
           -1  0  .5  .5];



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Motor Torque
%3863 24C
Ke = 0.00349*60/(2*pi); %V/rad/sec
Km = 0.0333;  %N-m/A
Vmax = 24;  %volts
R = 0.62; %ohms         Winding Resistance
Lw = 194*10^-6; %H      Inductance
Bw = 0; %Nm/rad/sec     Viscous Friction
GR = 1; %Gear Ratio

G = (6700/24)*(pi/30);  %rad/s/Voltage
Tc = 1;  %Time Constant - Seconds

Threshold = 50*(pi/30);%RPM
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Wheel Dynamics
Rw1 =  [cos(B)  0   -sin(B);
        0       1   0;
        sin(B)  0   cos(B)];

Rw2 =  [0       1   0;
        cos(B)  0   -sin(B);
        sin(B)  0   cos(B)];

Rw3 =  [-cos(B) 0   sin(B);
        0       1   0;
        sin(B)  0   cos(B)];
```

```
Rw4 =  [0       1    0;
        -cos(B) 1    sin(B);
        sin(B)  0    cos(B)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Simulate
%t=sim('platform_modified', simtime);

% %...
%
clf
%Quaternions
figure(1)
subplot(2,2,1)
plot(Plot.signals.values(:,7))
title('Quaternion 1 verses Time')
xlabel('Time (s)')
ylabel('Quaternion Value')
grid on

subplot(2,2,2)
plot(Plot.signals.values(:,8))
title('Quaternion 2 verses Time')
xlabel('Time (s)')
ylabel('Quaternion Value')
grid on

subplot(2,2,3)
plot(Plot.signals.values(:,9))
title('Quaternion 3 verses Time')
xlabel('Time (s)')
ylabel('Quaternion Value')
grid on

subplot(2,2,4)
plot(Plot.signals.values(:,10))
title('Quaternion 4 verses Time')
xlabel('Time (s)')
ylabel('Quaternion Value')
grid on

%Platform Rate & Clock
figure(2)
subplot(2,2,1)
plot(Plot.signals.values(:,4))
title('Wx verses Time')
xlabel('Time (s)')
ylabel('X-axis Angular Rate (deg/sec)')
grid on

subplot(2,2,2)
plot(Plot.signals.values(:,5))
title('Wy verses Time')
xlabel('Time (s)')
```

```
ylabel('Y-axis Angular Rate (deg/sec)')
grid on

subplot(2,2,3)
plot(Plot.signals.values(:,6))
title('Wz verses Time')
xlabel('Time (s)')
ylabel('Z-axis Angular Rate (deg/sec)')
grid on

subplot(2,2,4)
plot(Plot.signals.values(:,11))
title('Command Number verses Time')
xlabel('Time (s)')
ylabel('Command Number')
grid on

%Wheel Rates
figure(3)
subplot(2,2,1)
plot(wheel.signals.values(:,1))
title('Wheel 1 Rate verses Time')
xlabel('Time (s)')
ylabel('Wheel 1 Angular Rate (rad/s)')
grid on

subplot(2,2,2)
plot(wheel.signals.values(:,2))
title('Wheel 2 Rate verses Time')
xlabel('Time (s)')
ylabel('Wheel 2 Angular Rate (rad/s)')
grid on

subplot(2,2,3)
plot(wheel.signals.values(:,3))
title('Wheel 3 Rate verses Time')
xlabel('Time (s)')
ylabel('Wheel 3 Angular Rate (rad/s)')
grid on

subplot(2,2,4)
plot(wheel.signals.values(:,4))
title('Wheel 4 Rate verses Time')
xlabel('Time (s)')
ylabel('Wheel 4 Angular Rate (rad/s)')
grid on

% %Animate
% figure(4)
% animate(Plot);
```

C.1.2 TAC Board

```
%Jeff Logan
%7-11-08
%Tests Velocimeter Fidelity

format compact
clear all
clc
clf

phaseshift=0;  %0 for clockwise 1 counter-clockwise
simtime=500; % [Sec]
freq=1; % [Hz] (max=111.67 Hz--motor's physical limitation
         %       min=0.5 Hz {first 30 RPMs}--caused by v_ref limitation)
Vcc=5;  % [V]

%% SCHMITT TRIGGERS
%Resistors
R1st=1;       % [Ohm]
R2st=1000;    % [Ohm]
R3st=1000;    % [Ohm]

%% Voltage Divider 1 & 2
R1vd12=1000;    % [Ohm]
R2vd12=1;       % [Ohm]

%% Voltage Divider 3
R1vd3=10000;   % [Ohm]
R2vd3=1000;    % [Ohm]

%% RC Circuit
Rrc=37000;     % [Ohm]
Crc=220;       % [uF]

%% Comparitor
VinH=3.4;  %@5V
VinL=0.25; %@5V

%% Comparitor
VinH_Switch=2;    %@5V
VinL_Switch=0.8;  %@5V

%% Syncronus Detector
R1sd=10000;    % [Ohm]
R2sd=10000;    % [Ohm]
R3sd=10000;    % [Ohm]

%% Calculations
T=1/freq;      % [s^-1]
if phaseshift==0
    phase=0;   % [Sec]
elseif phaseshift==1
    phase=T/2; % [Sec]
```

```
else
    disp('ERROR: YOU ARE A MORON.');
end

t=sim('Velocimeter', simtime);
figure(1)
plot(t,v.signals.values)
xlabel('Time [Sec]')
ylabel('Velocity [Volts]')
title('Velocity vs. Time')
figure(2)
v2f=1/0.0447; %Voltage to Frequency Conversion
v2RPM=v2f*60; %Voltage to RPM Conversion
plot(t,v.signals.values*v2RPM)
xlabel('Time [Sec]')
ylabel('Rotations Per Minute')
title('RPM vs. Time')

% clc
% disp('Velocimeter Test Run Complete:')
% fprintf('\nThe reference velocity for the Voltage comparator is: %g
[Volts]!\n',max(v_ref.signals.values))
% disp('Please Review Figure 1 for the Voltage vs. Time Results')
```

C.1.3 Switch Board

```
%Jeff Logan
%7-11-08
%Tests Velocimeter Fidelity

format compact
clear all
clc
clf

%% Constants
simtime=500; % [Sec]
Vcc=5;  % [V]
FreqCollect=10; % Collection Rate [Hz]
               % 1 cycle = 1 data signal flop
f=FreqCollect;
T=1/f; % Time per cycle [seconds];
tlag=T/4; %Time lag between signals;

%Conversion Factor
conv=1/0.0447; %Voltage to Frequency Conversion
v2RPM=conv*60; %Voltage to RPM Conversion

%% Bit Source (I/O Pins 1-3 on ROBOSTIX
%%%Bit Digits are as follows 3 2 1%%%
% Bit 1 Signal Generation: 1s
bit1f = f;        % [Hz]
bit1w = 50;         % [% of period]
bit1p = .5*1/bit1f; % [seconds]

% Bit 2 Signal Generation: 2s
bit2f = 2*f;        % [Hz]
bit2w = 50;         % [% of period]
bit2p = .5*1/bit2f; % [seconds]

% Bit 3 Signal Generation: 4s
bit3f = 4*f;        % [Hz]
bit3w = 50;         % [% of period]
bit3p = .5*1/bit3f; % [seconds]

%% MOTOR Source
%%RAW SOURCE%%
%Motor 1
freq1=1; % [Hz] (max=111.67 Hz--motor's physical limitation
         %       min=0.5 Hz {first 30 RPMs}--caused by v_ref limitation)
%Motor 2
freq2=-1; % [Hz] (max=111.67 Hz--motor's physical limitation
         %       min=0.5 Hz {first 30 RPMs}--caused by v_ref limitation)
%Motor 3
freq3=-1; % [Hz] (max=111.67 Hz--motor's physical limitation
         %       min=0.5 Hz {first 30 RPMs}--caused by v_ref limitation)
%Motor 4
freq4=1; % [Hz] (max=111.67 Hz--motor's physical limitation
         %       min=0.5 Hz {first 30 RPMs}--caused by v_ref limitation)

%%Calculations%%
%Motor 1
T1=abs(1/freq1);      % [s^-1]
if freq1>=0
    phase1=0;   % [Sec]
else
    phase1=T1/2; % [Sec]
end
%Motor 2
T2=abs(1/freq2);      % [s^-1]
if freq2>=0
    phase2=0;   % [Sec]
else
    phase2=T2/2; % [Sec]
end
%Motor 3
T3=abs(1/freq3);      % [s^-1]
if freq3>=0
    phase3=0;   % [Sec]
else
    phase3=T3/2; % [Sec]
end
%Motor 4
T4=abs(1/freq4);      % [s^-1]
if freq4>=0
    phase4=0;   % [Sec]
else
    phase4=T4/2; % [Sec]
end

%% TAC BOARD COMPONETS
%%SCHMITT TRIGGERS
R1st=1;         % [Ohm]
R2st=1000;      % [Ohm]
R3st=1000;      % [Ohm]

%%Voltage Divider 1 & 2
R1vd12=1000;      % [Ohm]
R2vd12=1;         % [Ohm]

%%Voltage Divider 3
R1vd3=10000;     % [Ohm]
R2vd3=1000;      % [Ohm]

%%RC Circuit
Rrc=36000;       % [Ohm]
Crc=220;         % [uF]

%%Comparitor
VinH=3.4;   %@5V
VinL=0.25;  %@5V

%%Comparitor
```

```
VinH_Switch=2;    %@5V
VinL_Switch=0.8;  %@5V

%%Syncronus Detector
R1sd=10000;    % [Ohm]
R2sd=10000;    % [Ohm]
R3sd=10000;    % [Ohm]

%% Simulate
t=sim('V_all_in_2', simtime);

%% Results: Data Separation
%Pre-Allocate :,1=time value
%             :,2=signal value
Motor1a=10*ones([length(t),2]);
Motor2a=10*ones([length(t),2]);
Motor3a=10*ones([length(t),2]);
Motor4a=10*ones([length(t),2]);

%Sort Data
M1c=1; M2c=1; M3c=1; M4c=1;
for i=1:length(t)
    if motor.signals.values(i)==1
        Motor1a(i,1)=t(i);
        Motor1a(i,2)=v.signals.values(i);
        M1c=M1c+1;
    elseif motor.signals.values(i)==2
        Motor2a(i,1)=t(i);
        Motor2a(i,2)=v.signals.values(i);
        M2c=M2c+1;
    elseif motor.signals.values(i)==3
        Motor3a(i,1)=t(i);
        Motor3a(i,2)=v.signals.values(i);
        M3c=M3c+1;
    elseif motor.signals.values(i)==4
        Motor4a(i,1)=t(i);
        Motor4a(i,2)=v.signals.values(i);
        M4c=M4c+1;
    end
end

%Finalize Resort
M1=zeros([M1c-1,2]);
M2=zeros([M2c-1,2]);
M3=zeros([M3c-1,2]);
M4=zeros([M4c-1,2]);

%Motor 1
j=1;
for i=1:length(t)
    if Motor1a(i,2)~=10
        M1(j,1)=Motor1a(i,1);
        M1(j,2)=Motor1a(i,2);
        j=j+1;
    end
```

```
end
%Motor 2
j=1;
for i=1:length(t)
    if Motor2a(i,2)~=10
        M2(j,1)=Motor2a(i,1);
        M2(j,2)=Motor2a(i,2);
        j=j+1;
    end
end
%Motor 3
j=1;
for i=1:length(t)
    if Motor3a(i,2)~=10
        M3(j,1)=Motor3a(i,1);
        M3(j,2)=Motor3a(i,2);
        j=j+1;
    end
end
%Motor 4
j=1;
for i=1:length(t)
    if Motor4a(i,2)~=10
        M4(j,1)=Motor4a(i,1);
        M4(j,2)=Motor4a(i,2);
        j=j+1;
    end
end

%% Results: Plot
%%TAC OUTPUT
% figure(4)
% subplot(2,2,1)
% plot(t,sig1.signals.values)
% title('Motor 1 Voltage verses Time (B4 SB)')
% xlabel('Time (s)')
% ylabel('Motor 1 [Volts]')
% grid on
%
% subplot(2,2,2)
% plot(t,sig2.signals.values)
% title('Motor 2 Voltage verses Time (B4 SB)')
% xlabel('Time (s)')
% ylabel('Motor 2 [Volts]')
% grid on
%
% subplot(2,2,3)
% plot(t,sig3.signals.values)
% title('Motor 3 Voltage verses Time (B4 SB)')
% xlabel('Time (s)')
% ylabel('Motor 3 [Volts]')
% grid on
%
% subplot(2,2,4)
% plot(t,sig4.signals.values)
% title('Motor 4 Voltage verses Time (B4 SB)')
% xlabel('Time (s)')
% ylabel('Motor 4 [Volts]')
% grid on


%%Switch Board OUTPUT
figure(1)
subplot(2,2,1)
plot(M1(:,1),M1(:,2))
title('Motor 1 Voltage verses Time')
xlabel('Time (s)')
ylabel('Motor 1 [Volts]')
grid on

subplot(2,2,2)
plot(M2(:,1),M2(:,2))
title('Motor 2 Voltage verses Time')
xlabel('Time (s)')
ylabel('Motor 2 [Volts]')
grid on

subplot(2,2,3)
plot(M3(:,1),M3(:,2))
title('Motor 3 Voltage verses Time')
xlabel('Time (s)')
ylabel('Motor 3 [Volts]')
grid on

subplot(2,2,4)
plot(M4(:,1),M4(:,2))
title('Motor 4 Voltage verses Time')
xlabel('Time (s)')
ylabel('Motor 4 [Volts]')
grid on


%%RPM PLOTS post Switch Board
figure(2)
plot(t,v.signals.values)
xlabel('Time [Sec]')
ylabel('Rotations Per Minute')
title('RPM vs. Time')

subplot(2,2,1)
plot(M1(:,1),M1(:,2)*v2RPM)
title('Motor 1 RPM verses Time')
xlabel('Time (s)')
ylabel('Motor 1 [RPM]')
grid on

subplot(2,2,2)
plot(M2(:,1),M2(:,2)*v2RPM)
title('Motor 2 RPM verses Time')
xlabel('Time (s)')
ylabel('Motor 2 [RPM]')
```

```
grid on

subplot(2,2,3)
plot(M3(:,1),M3(:,2)*v2RPM)
title('Motor 3 RPM verses Time')
xlabel('Time (s)')
ylabel('Motor 3 [RPM]')
grid on

subplot(2,2,4)
plot(M4(:,1),M4(:,2)*v2RPM)
title('Motor 4 RPM verses Time')
xlabel('Time (s)')
ylabel('Motor 4 [RPM]')
grid on



% %%BIT IMPACT
% figure(3)
% subplot(2,2,1)
% plot(t,bit1.signals.values)
% title('Bit 1 verses Time (I/O pin 1)')
% xlabel('Time (s)')
% ylabel('Bit 1 Value [Volts]')
% grid on
%
% subplot(2,2,2)
% plot(t,bit2.signals.values)
% title('Bit 2 verses Time (I/O pin 2)')
% xlabel('Time (s)')
% ylabel('Bit 2 Value [Volts]')
% grid on
%
% subplot(2,2,3)
% plot(t,bit3.signals.values)
% title('Bit 3 verses Time (I/O pin 3)')
% xlabel('Time (s)')
% ylabel('Bit 3 Value [Volts]')
% grid on
%
% subplot(2,2,4)
% plot(t,motor.signals.values)
% title('Motor Number verses Time (Translator)')
% xlabel('Time (s)')
% ylabel('Motor Number')
% grid on



%% Results: Display Relative Information
clc
disp('Velocimeter Test Run Complete:')
fprintf('\n The reference velocity for the Voltage comparator is: %g
[Volts]!\n',max(v_ref1.signals.values))
fprintf('\n The period of a full data flop is: %g [sec]!\n',T)
fprintf(' The lag time between motor readings is: %g [sec]!\n\n',tlag)
```

```
disp('  NOTE: See Figure 1 for graphed results.')

% (c) Jeff Logan 7/29/2008
```

C.1.4 Automated Calibration

```
function [gyro, accel, tac]=calibrate()
%Place this function in the System Identification Folder in My
%Documents/MATLAB/Seth's Code/System Identification

simtime=150; %seconds

disp('Initiating Calibration.  This will take two minutes, and thirty
seconds.')
disp(' NOTE: Do not move the system during calibration.')

%% Run Sim
sim('Calibration',simtime)

%% Nominal Values
disp('Updating calibration constants...')
gyro  = mean(ADCwp);
accel = mean(ADCap);
tac   = mean(ADCtac);
disp('Calibration Complete!')
```

C.1.5 Calibration Analysis

```
%Calibration Analysis

clf
%% Initial Conditions
q1=0;
q2=0;
q3=0;
q4=1;

global q_init q4_init
q_init=[q1;q2;q3];  %IC for integral
q4_init=q4;         %IC for integral

%%
simtime=10;
rate=1;
[gyro,accel,tac]=calibrate();
%disp('Running Test Sim.  This will Take 2 and a half minutes')
tout=sim('GYRO_TEST',simtime);
beep
disp('Sim Complete!')

VccADC=mean(ADCVcc.signals.values);
wpADC=mean(ADCwp.signals.values);
tacADC=mean(ADCtac.signals.values);
%apADC=mean(ADCap.signals.values);

%% ADC Analysis
disp('Analyzing the Data')
fprintf('\nPin 0''s (x.) mean value is: %g.',wpADC(1));
fprintf('\nPin 1''s (y.) mean value is: %g.',wpADC(2));
fprintf('\nPin 2''s (z.) mean value is: %g.',wpADC(3));
fprintf('\nPin 3''s (Vcc) mean value is: %g.',VccADC);
fprintf('\nPin 4''s (Tac1) mean value is: %g.',tacADC(1));
fprintf('\nPin 5''s (Tac2) mean value is: %g.',tacADC(2));
fprintf('\nPin 6''s (Tac3) mean value is: %g.',tacADC(3));
fprintf('\nPin 7''s (Tac4) mean value is: %g.',tacADC(4));
%fprintf('\nPin 8''s (x..) mean value is: %g.',apADC(1));
%fprintf('\nPin 9''s (y..) mean value is: %g.',apADC(2));
%fprintf('\nPin 10''s (z..) mean value is: %g.\n\n',apADC(3));

%% wp vs t
%ADC
figure(1)
subplot(2,1,1)
plot(tout,ADCwp.signals.values)
xlabel('Time [seconds]')
ylabel('ADC Reading [steps]')
title('ADC Reading Over Time')
hold on
plot(tout,gyro(1),'k--');
legend('x_d_o_t','y_d_o_t','z_d_o_t','Mean Value')
hold on
```

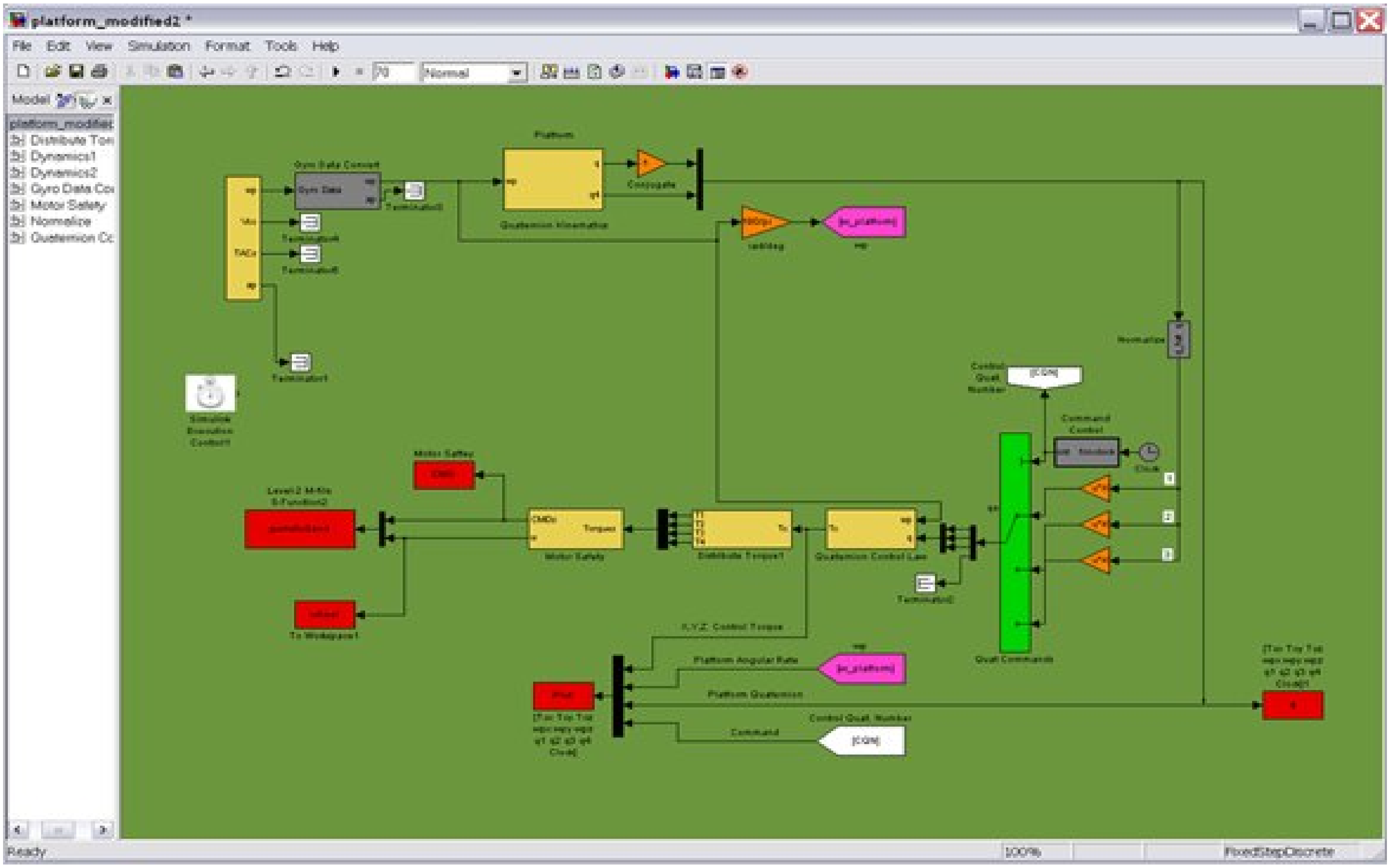
```
plot(tout,gyro(2),'k--');
hold on
plot(tout,gyro(3),'k--');

%wp
subplot(2,1,2)
plot(tout,wp.signals.values)
xlabel('Time [seconds]')
ylabel('PRWP Angular Velocity [deg/s]')
title('PRWP Angular Velocity Over Time')
legend('x_d_o_t','y_d_o_t','z_d_o_t')

% %% Quaternions v t
% figure(2)
% plot(tout,q.signals.values)
% xlabel('Time [seconds]')
% ylabel('Quaternion')
% title('Quaternion Over Time')
% legend('q1','q2','q3','q4')
%
%% TACs
%ADC
figure(3)
subplot(2,1,1)
plot(tout,ADCtac.signals.values)
hold on
plot(tout,ADCVcc.signals.values(1,:),'g')
xlabel('Time [seconds]')
ylabel('ADC TAC Value [steps]')
title('TAC ADC reading Over Time')
hold on
plot(tout,tac(1),'k--');
legend('TAC 1','TAC 2','TAC 3','TAC 4','Vcc','Mean Value')
hold on
plot(tout,tac(2),'k--');
hold on
plot(tout,tac(3),'k--');
hold on
plot(tout,tac(4),'k--');

%tac
subplot(2,1,2)
plot(tout,TAC.signals.values)
hold on
xlabel('Time [seconds]')
ylabel('Wheel Velocity (RPM)')
title('Wheel Velocity Over Time')
legend('TAC 1','TAC 2','TAC 3','TAC 4')

%% Alpha v t
%ADC(POST ORGION CONVERSION)
figure(4)
subplot(2,1,1)
plot(tout,ADCap.signals.values)
xlabel('Time [seconds]')
ylabel('Angular Acceleration (WRT pivot)')
```

```
title('Acceleration Over Time')
hold on
plot(tout,accel(1),'k--');
legend('a_x','a_y','a_z','Mean Value')
hold on
plot(tout,accel(2),'k--');
hold on
plot(tout,accel(3),'k--');

%ap
subplot(2,1,2)
plot(tout,ap.signals.values)
xlabel('Time [seconds]')
ylabel('Angular Acceleration (rad/s^2)')
title('Acceleration Over Time')
legend('ax','ay','az')
```

C.2 Full SIMULINK

C.2.1 Controls

Figure C.1: Main Window



Embedded Code:

Control Quaternion Timer:

```
function cnt= fcn(clock)

cnt = 1;

if clock > 30
    cnt = 2;
end
if clock > 60
    cnt = 3;
end
```

Direction Deduction:

```
function [CMD,wout]  = fcn(wplan,wlast)

%% Direction Portion
if sign(wlast)+sign(wplan)==-2
    %Continues Negative
    wout=-1*abs(wplan);
elseif sign(wlast)+sign(wplan)==0
    %Switches Direction
    if wplan>0
        %Switches from negative to positive
        wout=abs(wlast);
    else
        %Switches from positive to negative
        wout=-1*abs(wlast);
    end
else
    %Positive Direction case
    wout=abs(wlast);
end

difference=abs(abs(wplan)-abs(wlast));
safedif=40; %hz
safe=safedif*2*pi;

if abs(difference)<=safe
    CMD=0; %keep going
else
    CMD=1; %break!
end
```

Figure C.2: Demuxing the Receive Block

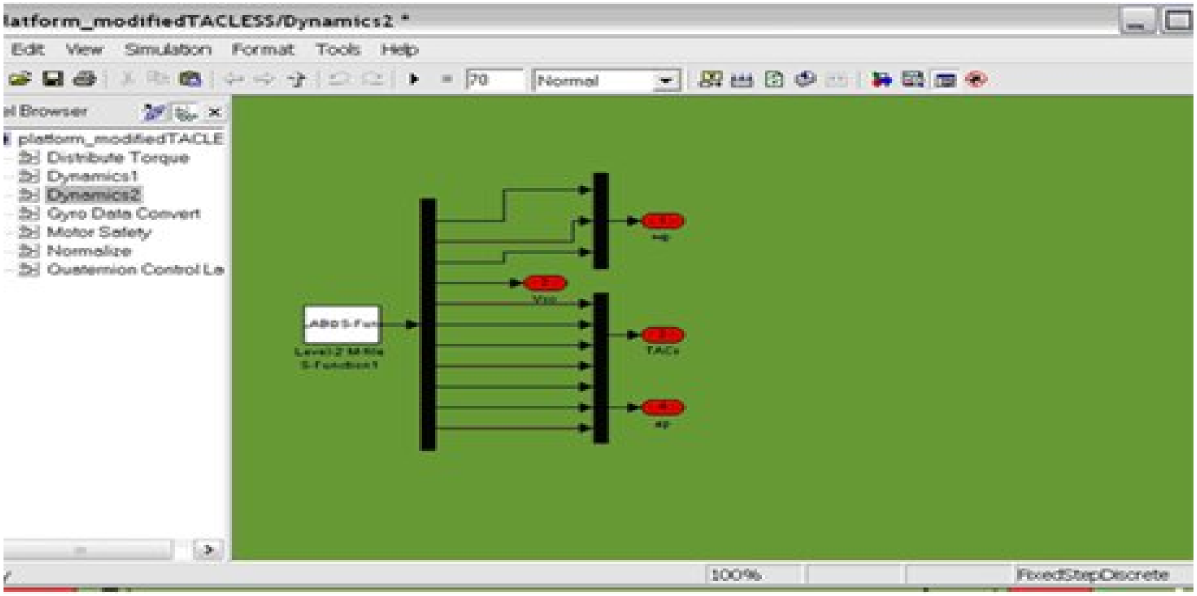


Figure C.3: Quaternion Dynamics

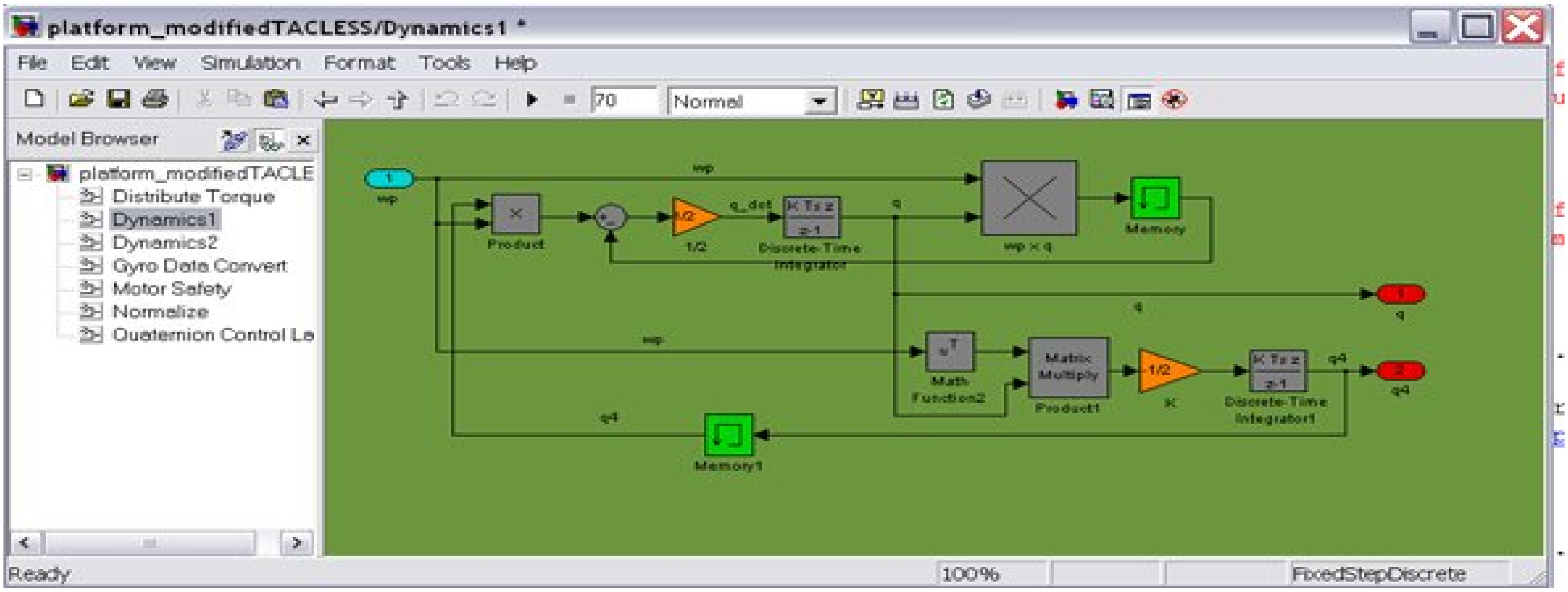


Figure C.4: Quaternion Normalization

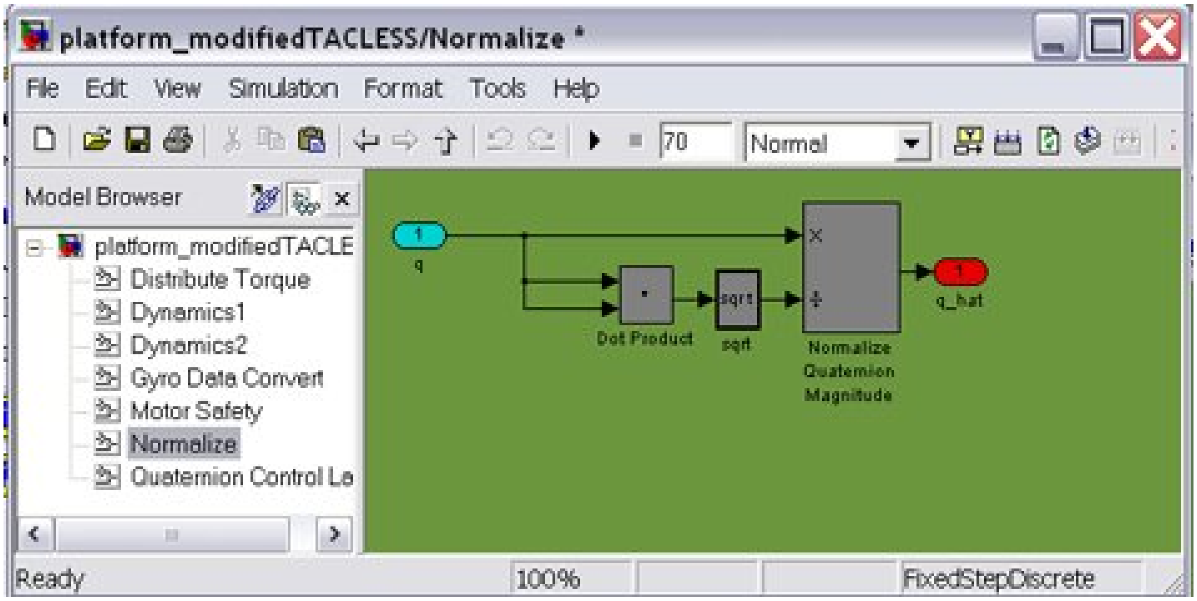


Figure C.5: Control Gain Application

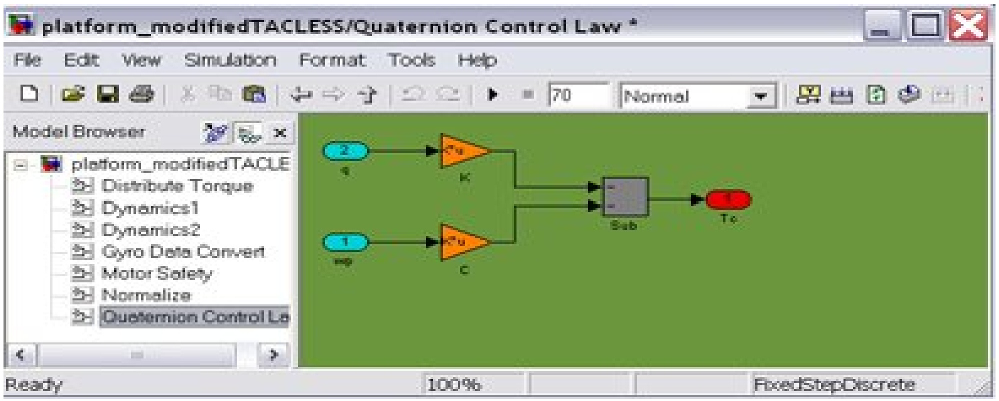


Figure C.6: Body Torque to Wheel Distribution

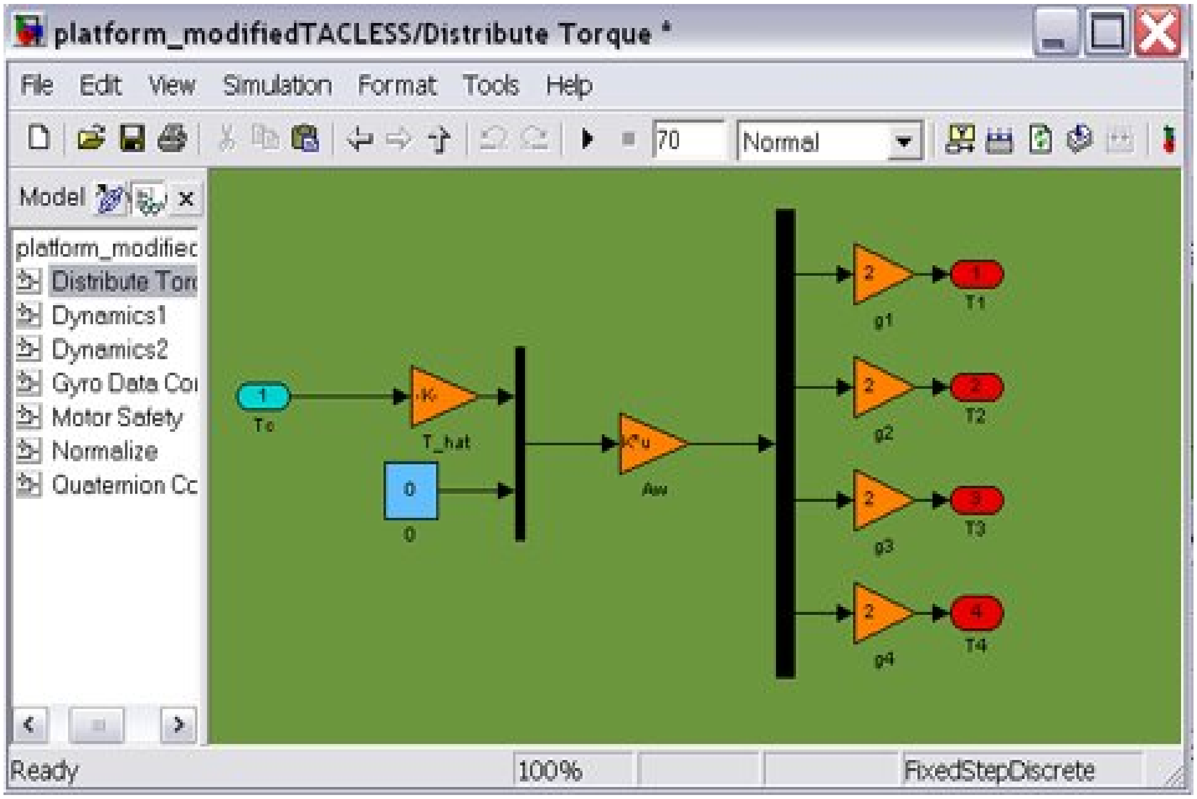
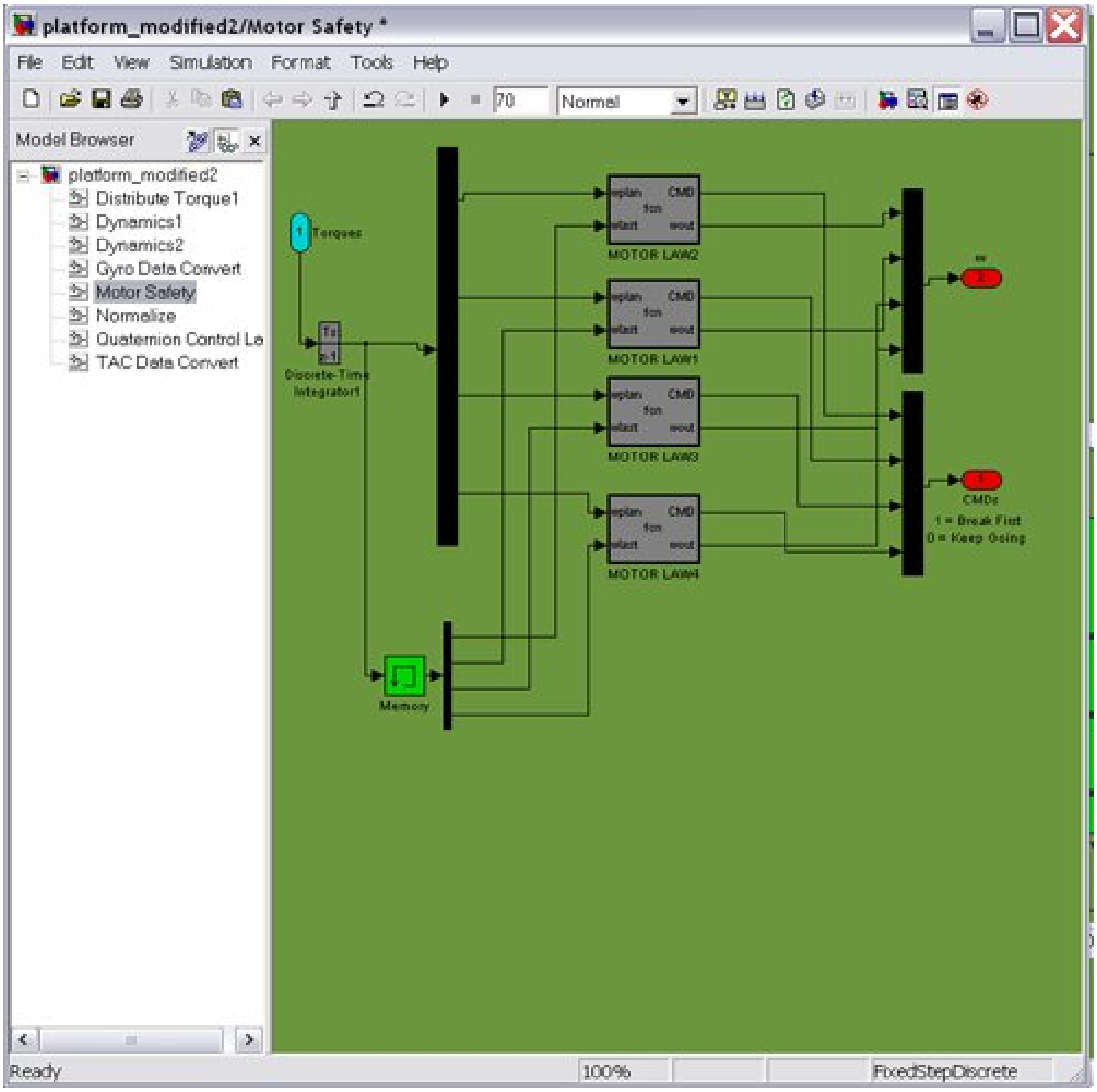
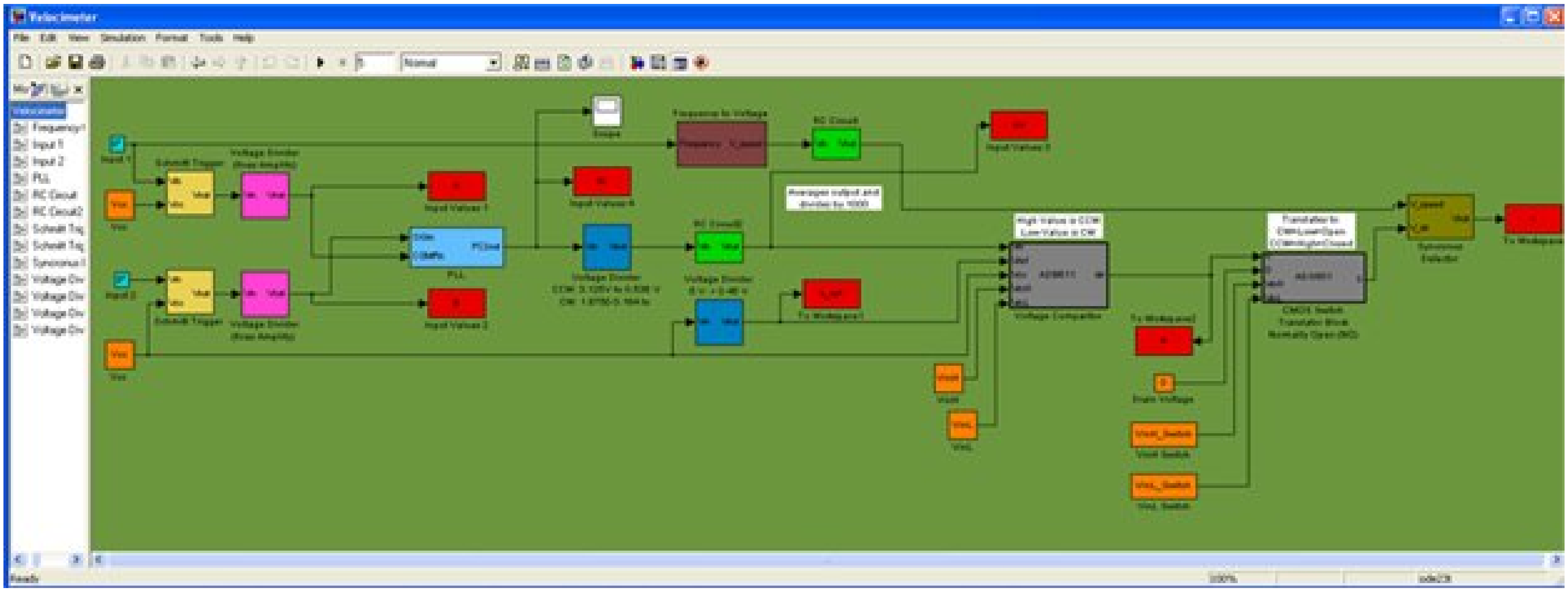


Figure C.7: Motor Safety and Direction Deduction



C.2.2 TAC Board

Figure C.8: Main Window



Embedded Code:

SDSP Switch:

```
function dir = ADB611(Vin,Vref,Vcc,VinH,VinL)
% This block supports the Embedded MATLAB subset.
% See the help menu for details.

dir=0;
if Vin>=Vref %Counter Clockwise Condition
    dir=VinH;
    if Vin >=4;
        disp('Your comparitor is melting!')
    end
elseif Vin < Vref
    dir=VinL; %Clockwise Condition
end
```

Voltage Comparator:

```
function S = ADG801(G,D,VinH,VinL)
% This block supports the Embedded MATLAB subset.
% See the help menu for details.

S=D;
if G>=VinH %Close the switch
    S=5;
elseif G>VinL&&G<VinH
    S=inf;
elseif G < VinL %Open the switch
end
```

Figure C.9: Frequency to Voltage

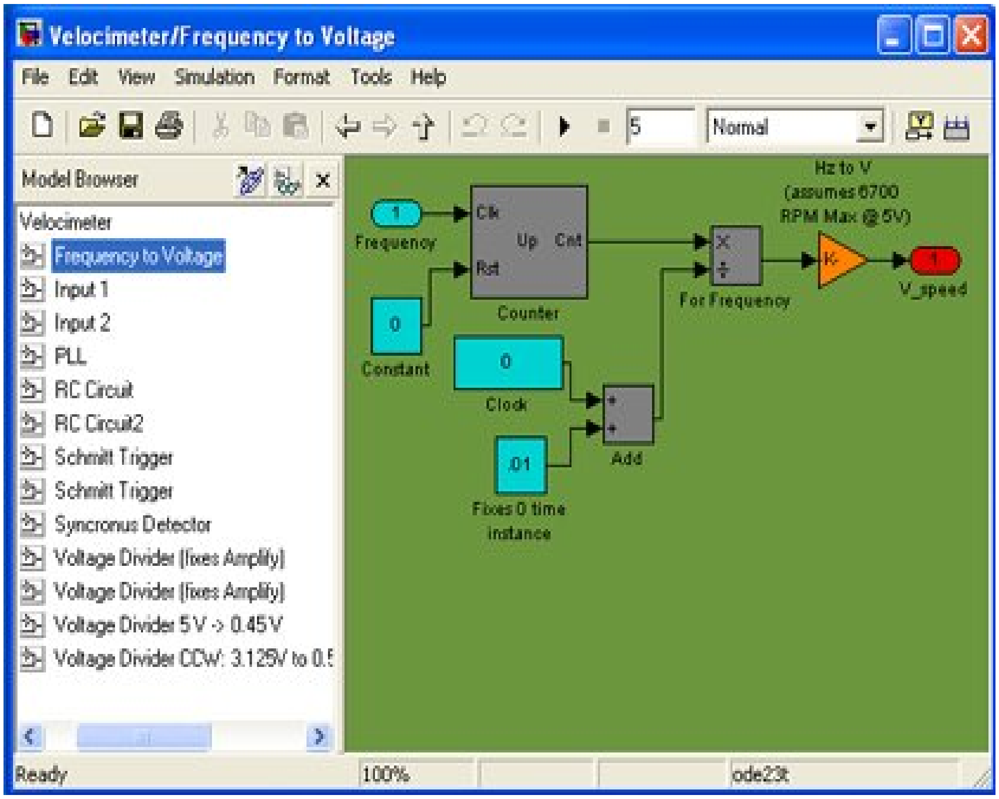
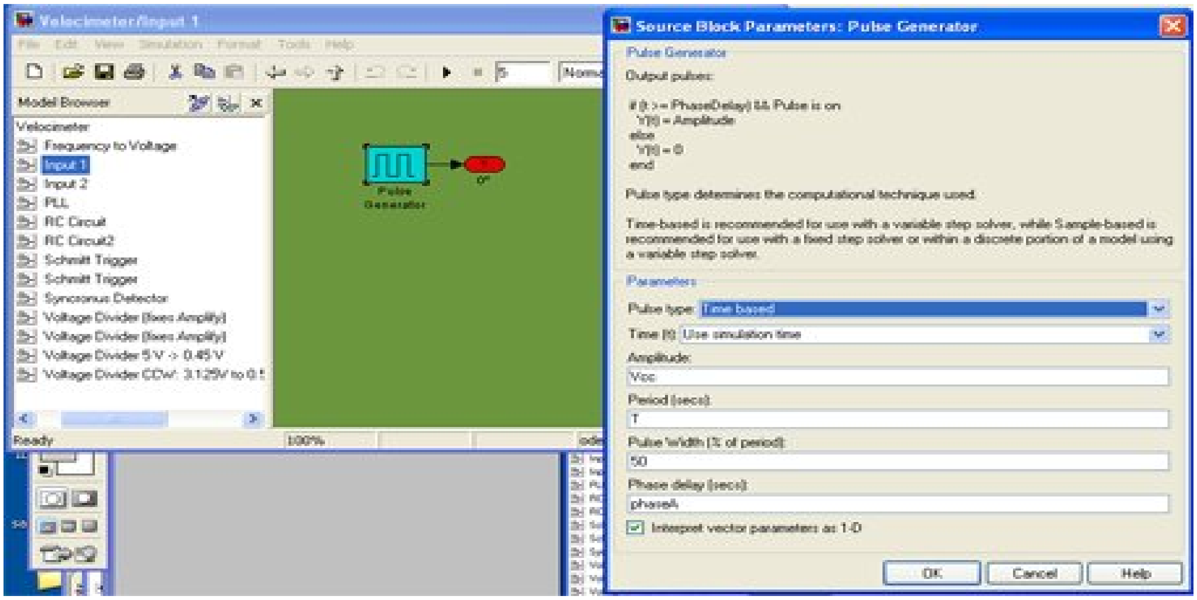


Figure C.10: Pulse Generator


Velocimeter/Input 1
File Edit View Simulation Format Tools Help
Model Browser
Velocimeter
Frequency to Voltage
Input 1
Input 2
PLL
RC Circuit
RC Circuit2
Schmitt Trigger
Schmitt Trigger
Syncronus Detector
Voltage Divider (fixes Amplify)
Voltage Divider (fixes Amplify)
Voltage Divider 5 V -> 0.45 V
Voltage Divider CCW: 3.125V to 0.5
Pulse Generator
Ready
100%
Source Block Parameters: Pulse Generator
OK
Cancel
Help

Pulse Generator

Output pulses:

```
if (t >= PhaseDelay) && Pulse is on
  Y(t) = Amplitude
else
  Y(t) = 0
end
```

Pulse type determines the computational technique used.

Time-based is recommended for use with a variable step solver, while Sample-based is recommended for use with a fixed step solver or within a discrete portion of a model using a variable step solver.

Parameters

Pulse type: Time based

Time (t): Use simulation time

Amplitude: Vcc

Period (secs): T

Pulse Width (% of period): 50

Phase delay (secs): phaseA

☑ Interpret vector parameters as 1-D

Figure C.11: Phase Lock Loop Simulation

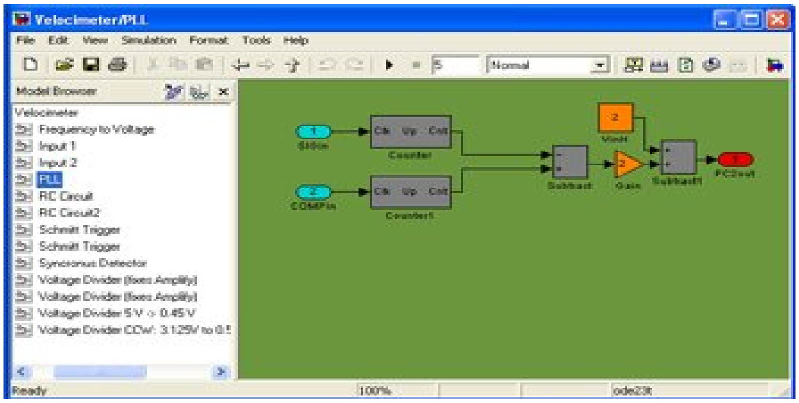


Figure C.12: RC Circuit 1 and 2

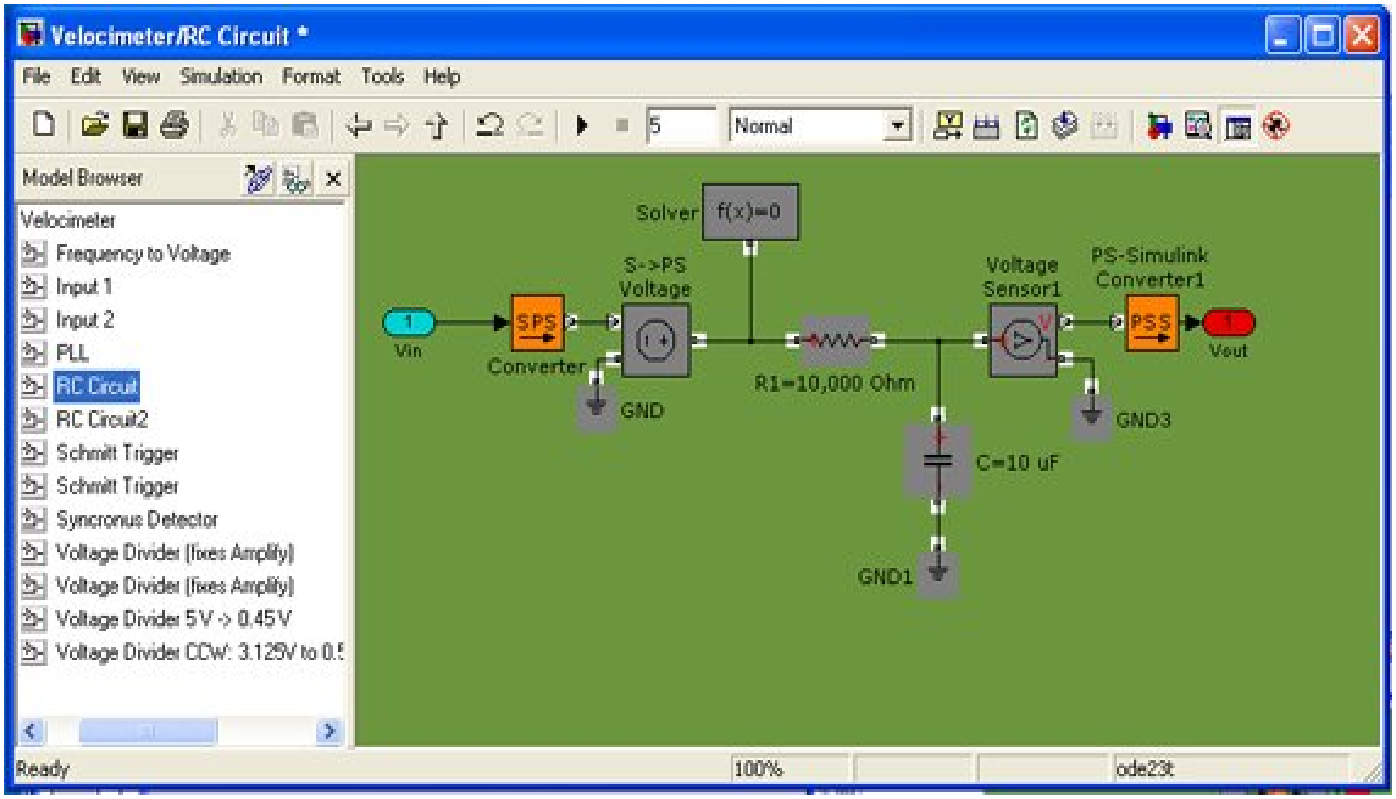


Figure C.13: Schmitt Trigger's A and B

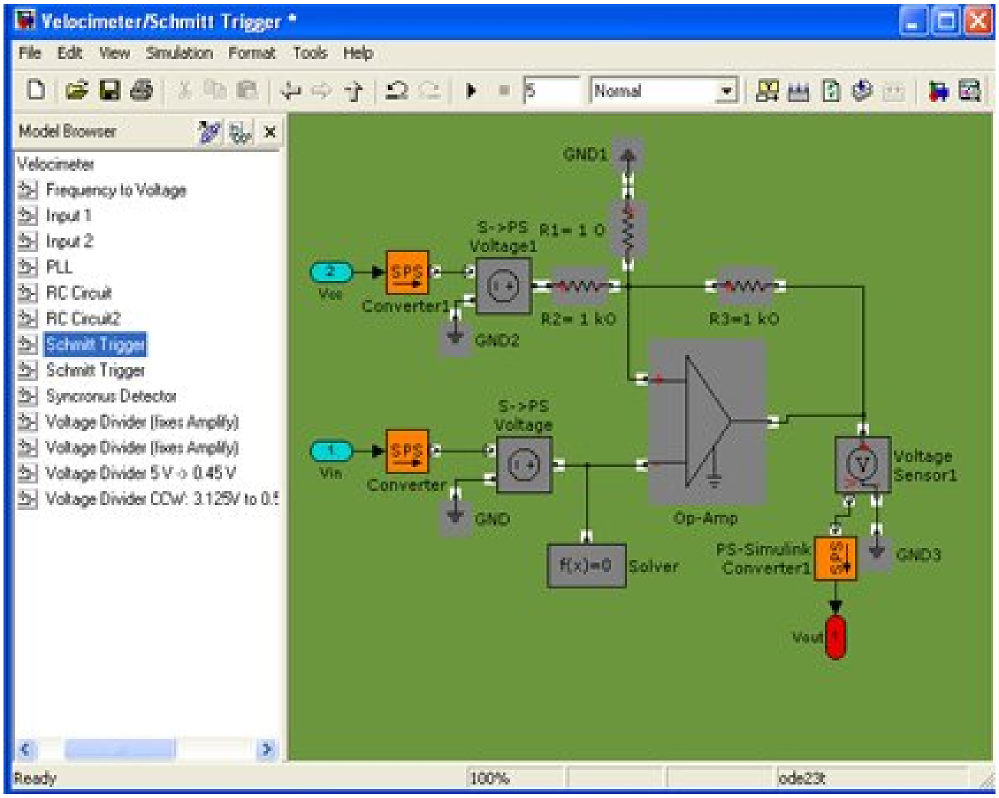


Figure C.14: Synchronous Detector

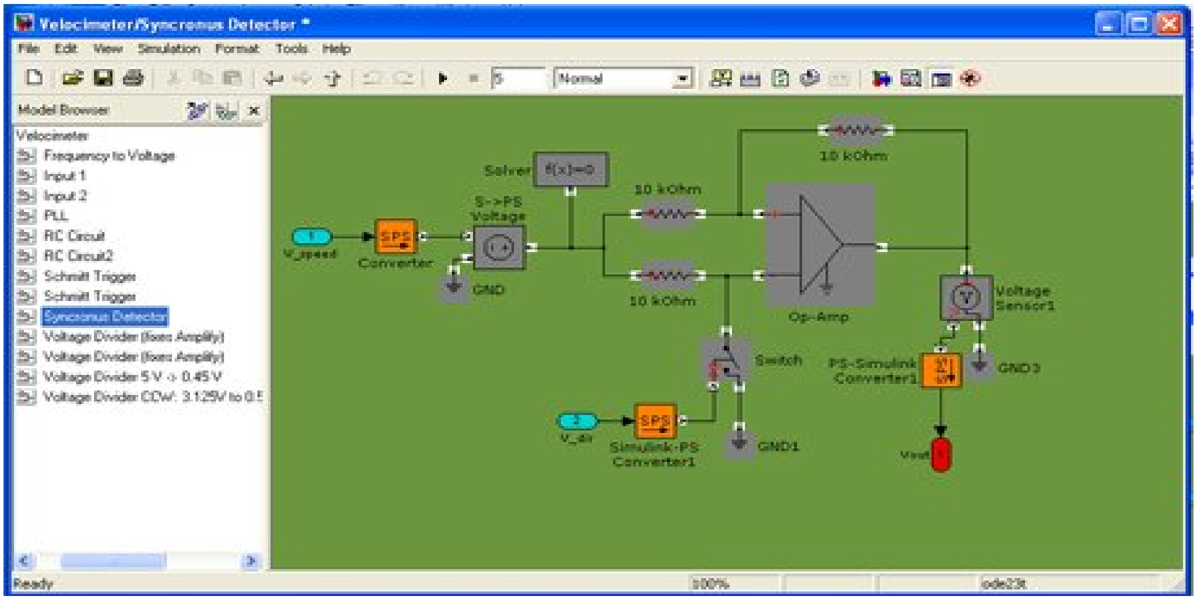


Figure C.15: Voltage Divider Post-Schmitt

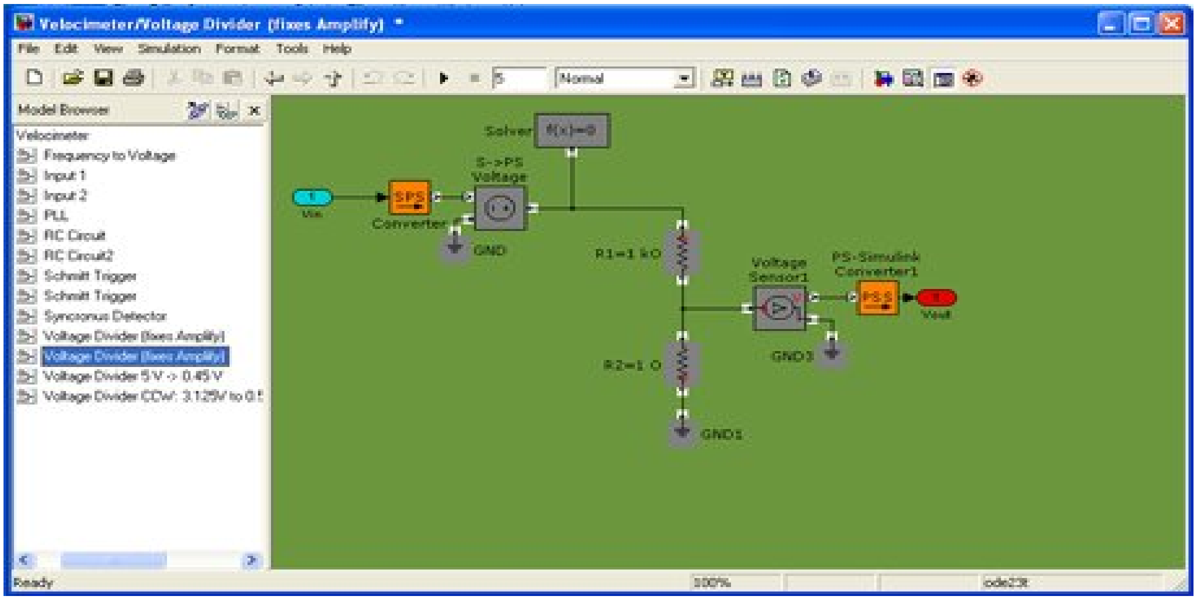
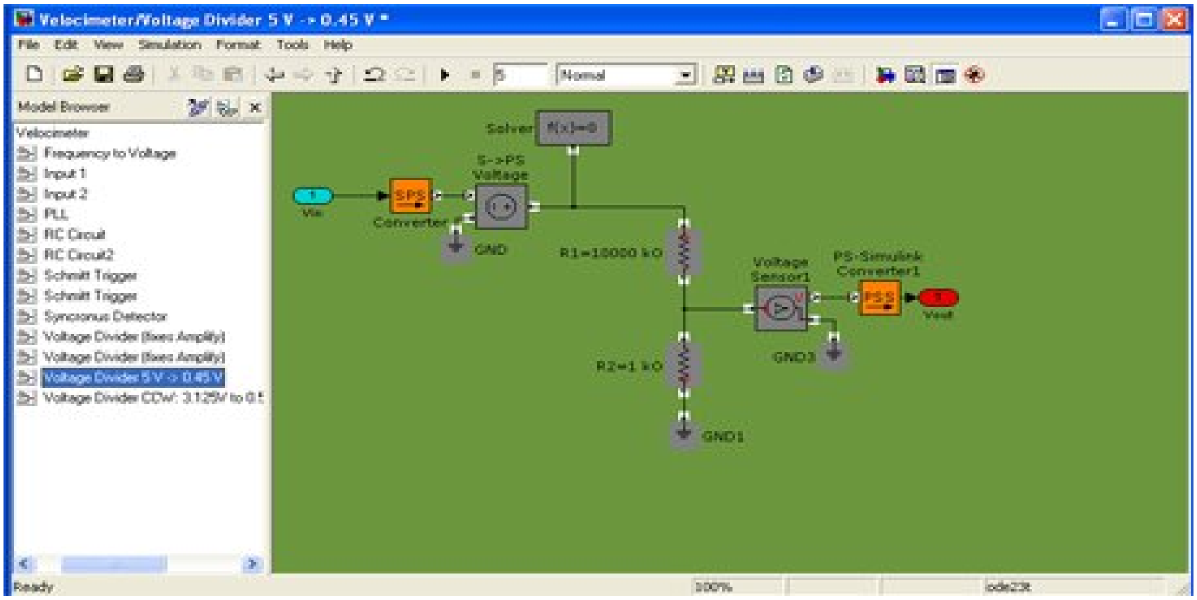
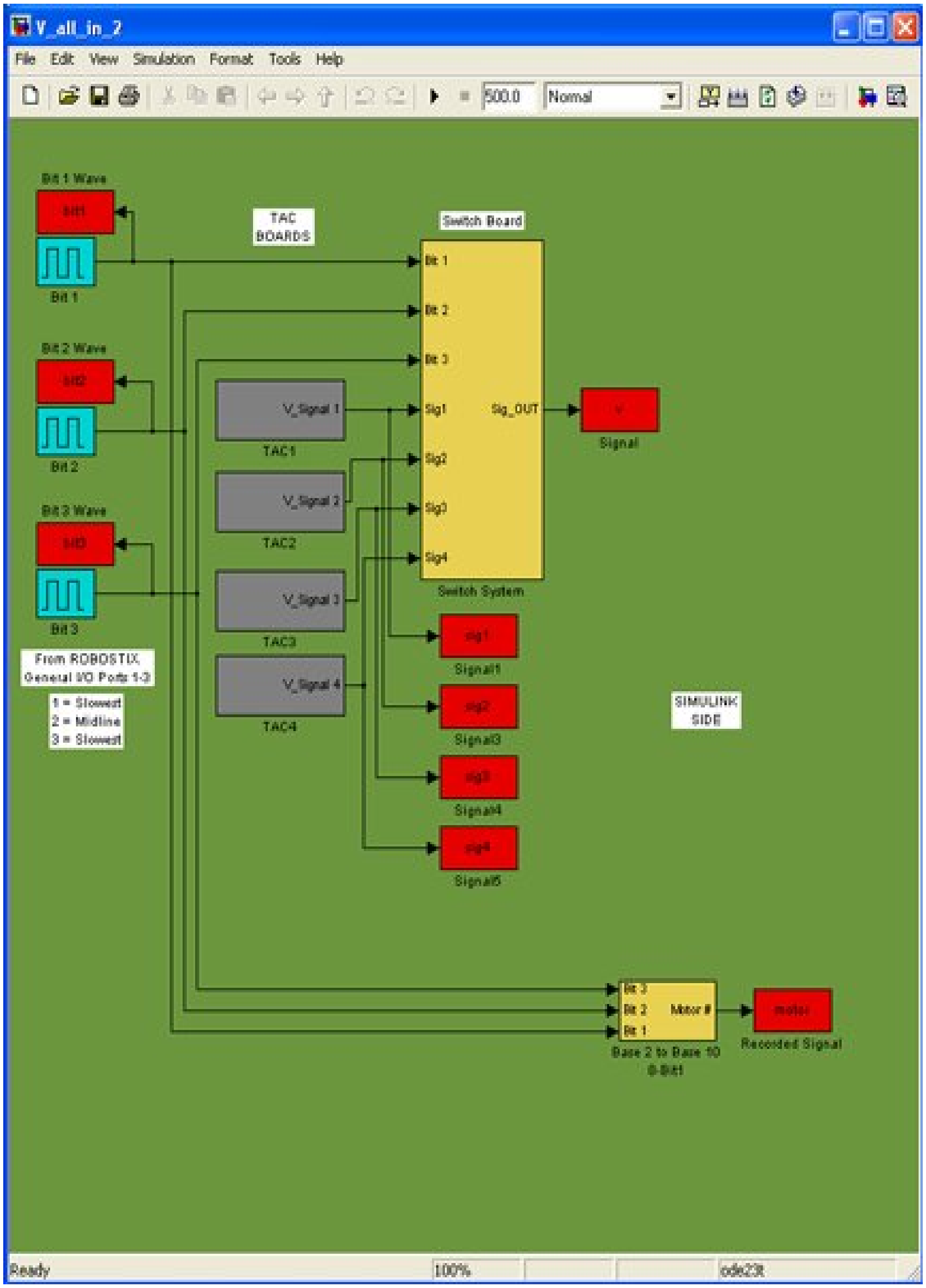


Figure C.16: Reference and Directional Voltage Divider



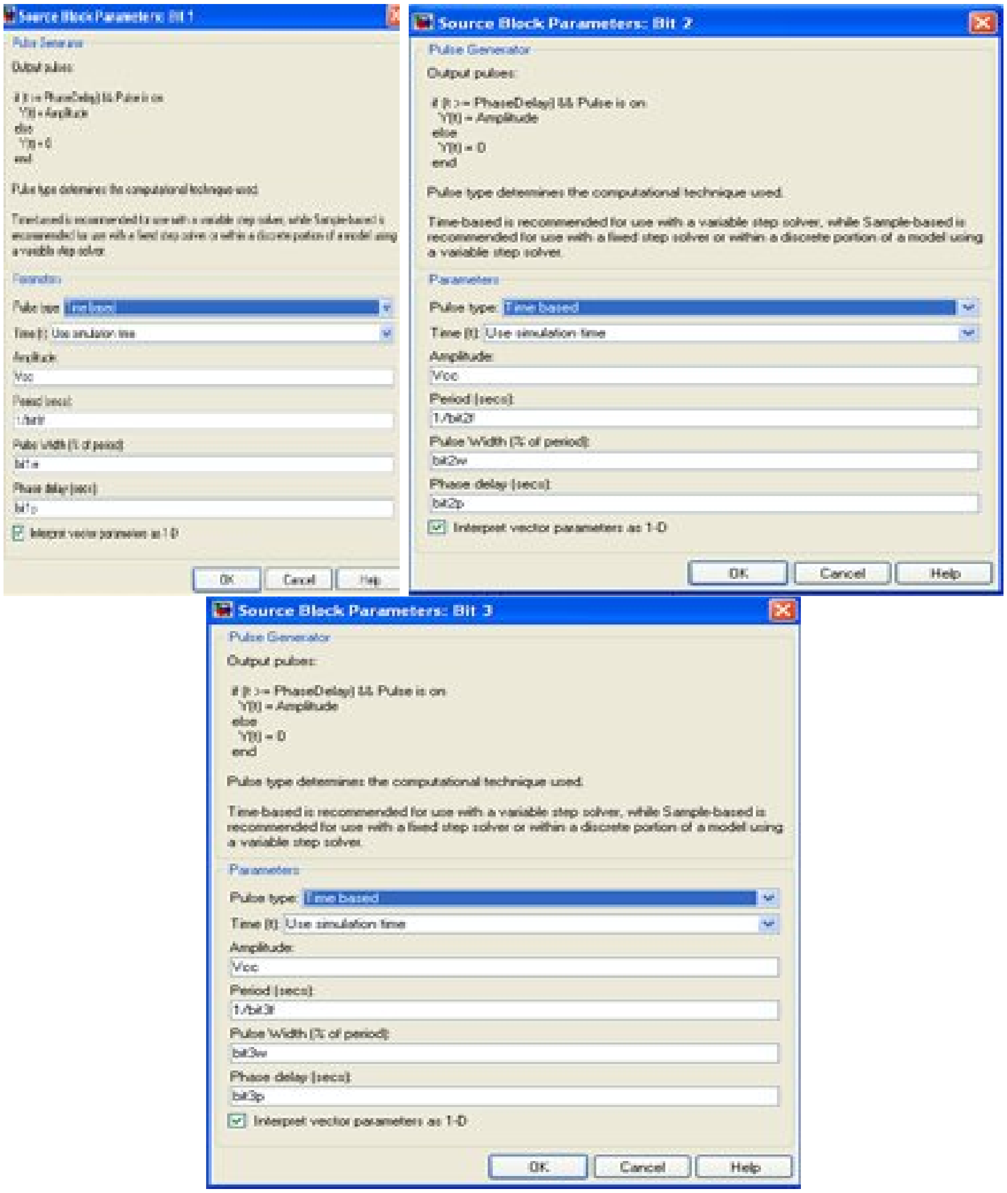
C.2.3 Switch Board

Figure C.17: Main Layout



Note: Each TAC (1 through 4) utilizes the exact same SIMULINK as the TAC boards listed previously.

Figure C.18: Simulated Logic Bit Input



Source Block Parameters: Bit 1

Pulse Generator

Output pulses:

```
if (t >= PhaseDelay) && Pulse is on
  Y(t) = Amplitude
else
  Y(t) = 0
end
```

Pulse type determines the computational technique used.

Time-based is recommended for use with a variable step solver, while Sample-based is recommended for use with a fixed step solver or within a discrete portion of a model using a variable step solver.

Parameters

Pulse type: Time based

Time (t): Use simulation time

Amplitude:

Vcc

Period (secs):

1/bit1f

Pulse Width (% of period):

bit1w

Phase delay (secs):

bit1p

☑ Interpret vector parameters as 1-D

Source Block Parameters: Bit 2

Pulse Generator

Output pulses:

```
if (t >= PhaseDelay) && Pulse is on
  Y(t) = Amplitude
else
  Y(t) = 0
end
```

Pulse type determines the computational technique used.

Time-based is recommended for use with a variable step solver, while Sample-based is recommended for use with a fixed step solver or within a discrete portion of a model using a variable step solver.

Parameters

Pulse type: Time based

Time (t): Use simulation time

Amplitude:

Vcc

Period (secs):

1/bit2f

Pulse Width (% of period):

bit2w

Phase delay (secs):

bit2p

☑ Interpret vector parameters as 1-D

Source Block Parameters: Bit 3

Pulse Generator

Output pulses:

```
if (t >= PhaseDelay) && Pulse is on
  Y(t) = Amplitude
else
  Y(t) = 0
end
```

Pulse type determines the computational technique used.

Time-based is recommended for use with a variable step solver, while Sample-based is recommended for use with a fixed step solver or within a discrete portion of a model using a variable step solver.

Parameters

Pulse type: Time based

Time (t): Use simulation time

Amplitude:

Vcc

Period (secs):

1/bit3f

Pulse Width (% of period):

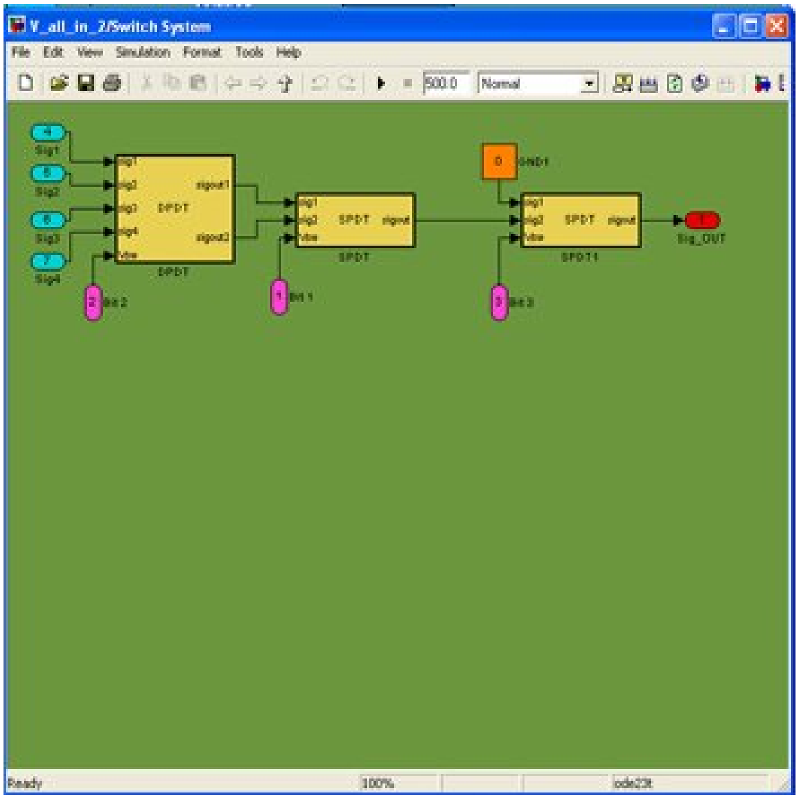
bit3w

Phase delay (secs):

bit3p

☑ Interpret vector parameters as 1-D

Figure C.19: Switch Logic



Embedded Code:

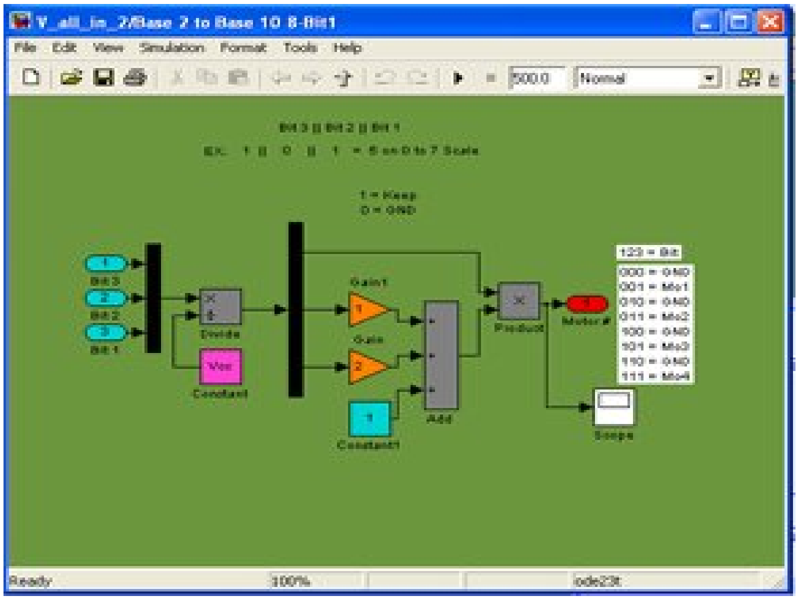
DPDT:

```
function [sigout1, sigout2] = DPDT(sig1,sig2,sig3,sig4,Vsw)
%Simulates a Single-Pole, Double-Toggle Switch

if Vsw>2.2 % HIGH
    sigout1=sig2;
    sigout2=sig4;
else        % LOW
    sigout1=sig1;
    sigout2=sig3;
end
```

SPDT:

```
function sigout = SPDT(sig1,sig2,Vsw)
%Simulates a Single-Pole, Double-Toggle Switch

if Vsw>2.2 % HIGH
    sigout=sig2;
else        % LOW
    sigout=sig1;
end
```

Figure C20: Switch Logic for Demuxing Signals



Bit 3 || Bit 2 || Bit 1

EX. 1 || 0 || 1 = 5 on 0 to 7 Scale

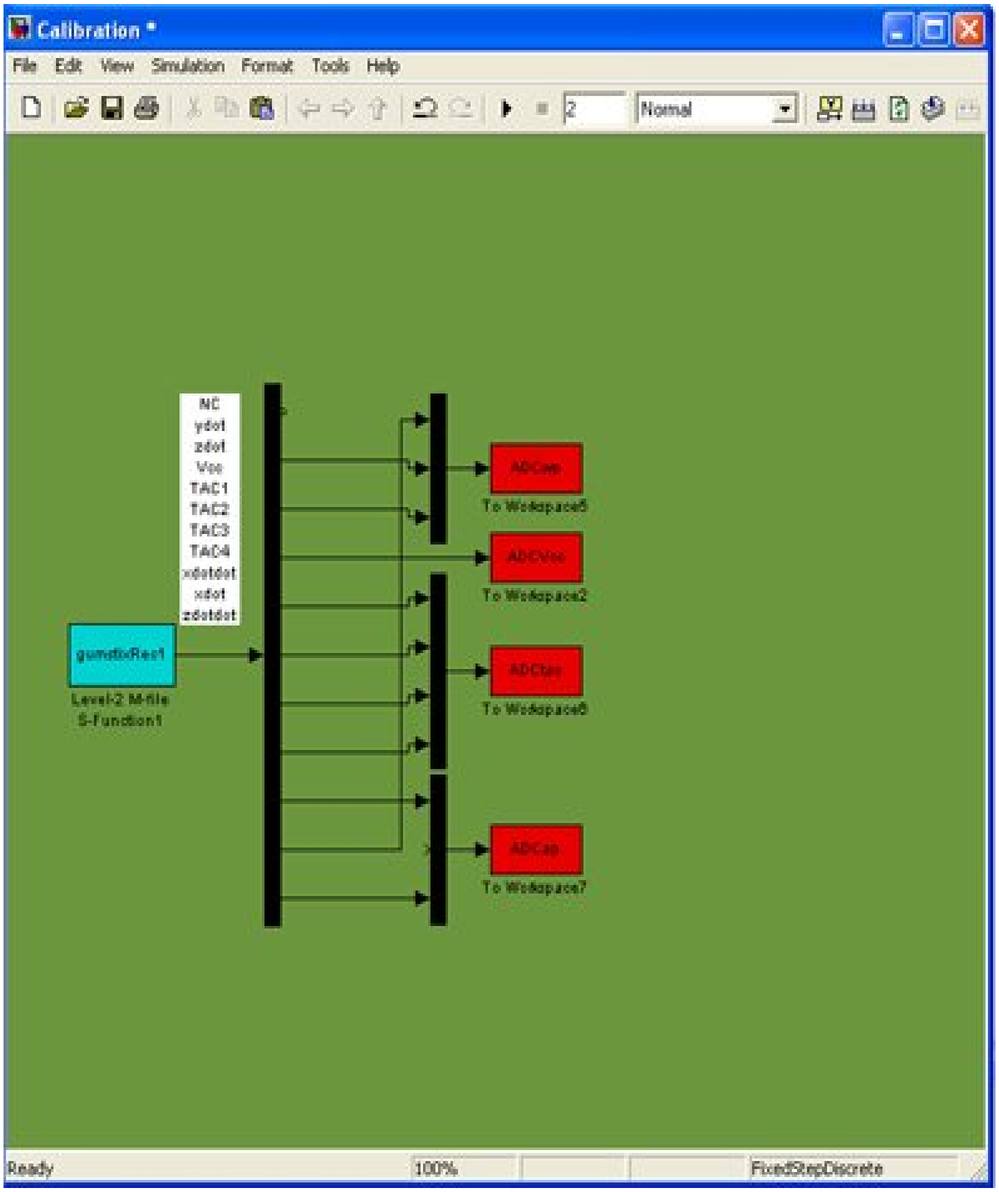
1 = Keep
0 = GND

123 = Bit

000 = GND
001 = Mo1
010 = GND
011 = Mo2
100 = GND
101 = Mo3
110 = GND
111 = Mo4

C.2.4 Automated Calibration

Figure C.21: Main: 200 Iterations for Calibration



C.2.5 Calibration Analysis

This simulation generates and tests the conversion blocks for each reading following auto calibration.

Figure C22: Calibration Analysis Main

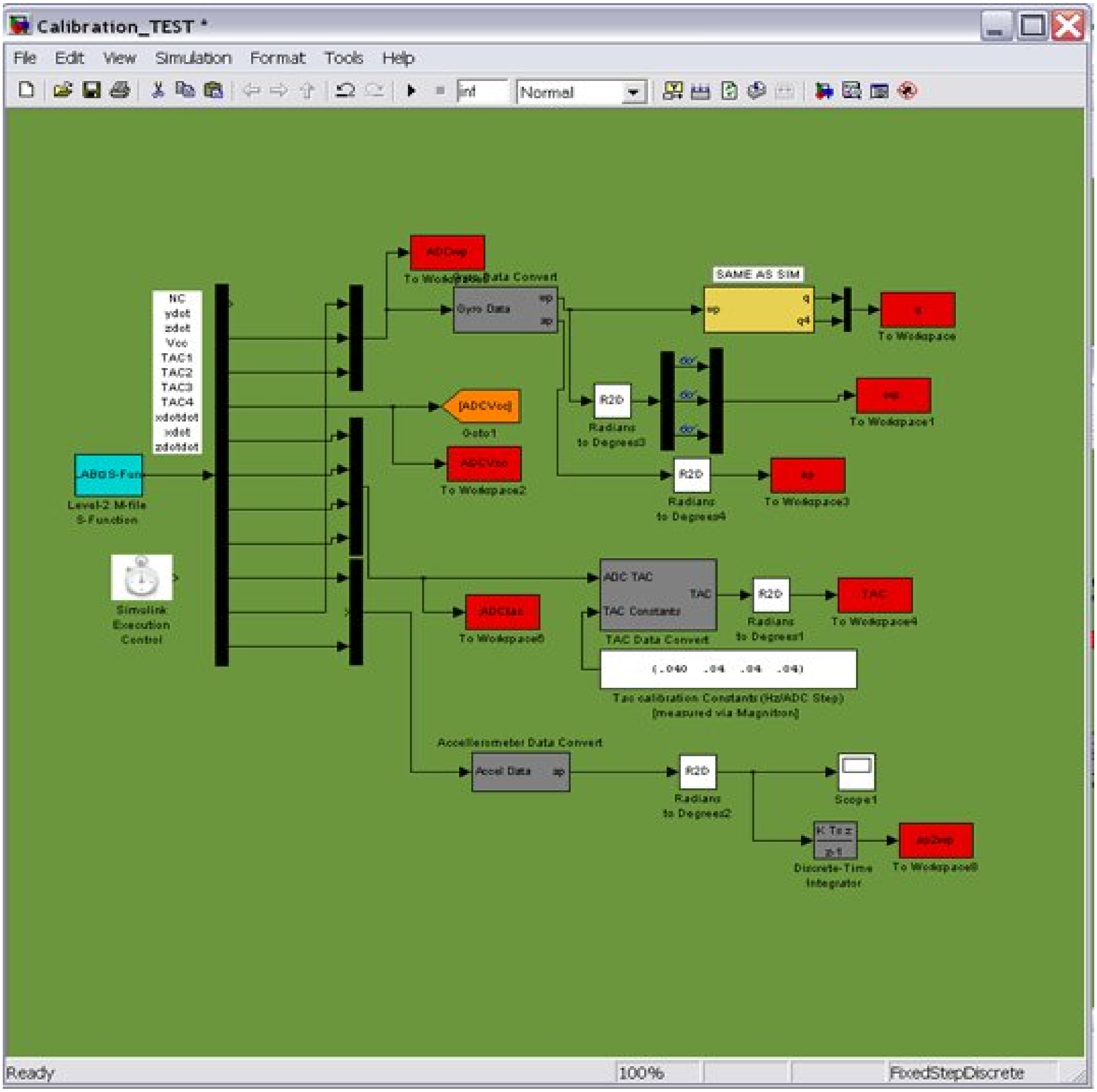
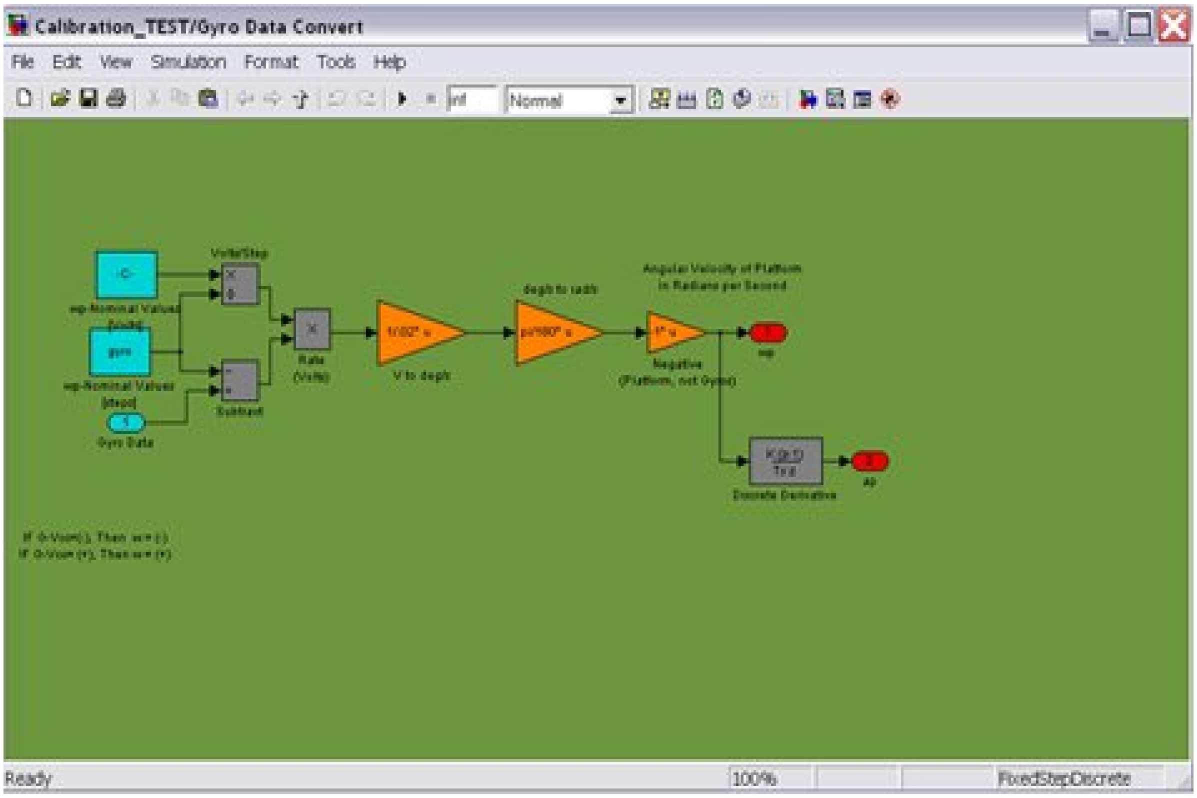
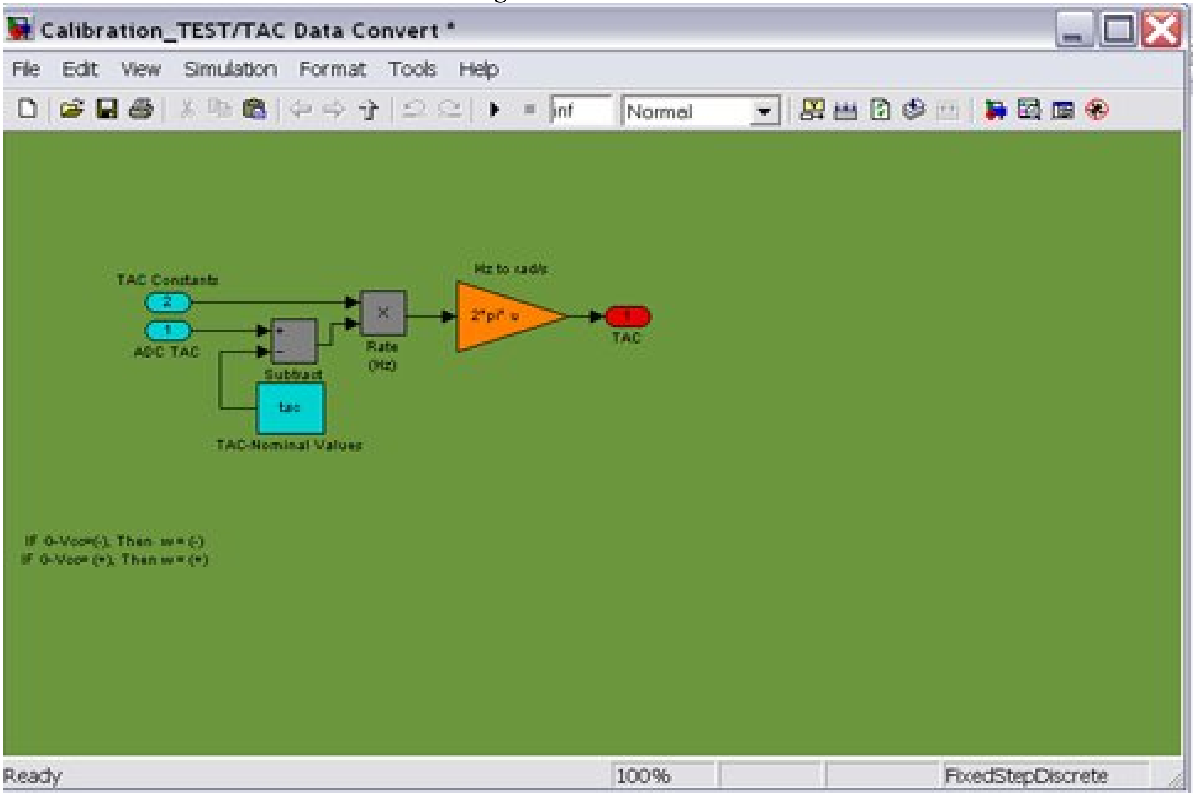


Figure C.23: Gyro Conversion Test


Calibration_TEST/Gyro Data Convert
File Edit View Simulation Format Tools Help
inf
Normal
Volts/Step
-C-
wp-Nominal Values [Volts]
gyro
wp-Nominal Values [steps]
1
Gyro Data
Subtract
Rate (Volts)
1/.002* u
V to deg/s
deg/s to rad/s
pi/180* u
Angular Velocity of Platform in Radians per Second
-1* u
Negative (Platform, not Gyro)
1
wp
K (z-1) / Ts z
Discrete Derivative
2
Ap
Ready
100%
FixedStepDiscrete

If 0-Vcom(-), Then w = (-)
If 0-Vcom (+), Then w = (+)

Figure C.24: TAC Test



If 0-Vcom(-), Then w = (-)
If 0-Vcom (+), Then w = (+)

Figure C.25: Accelerometer Test

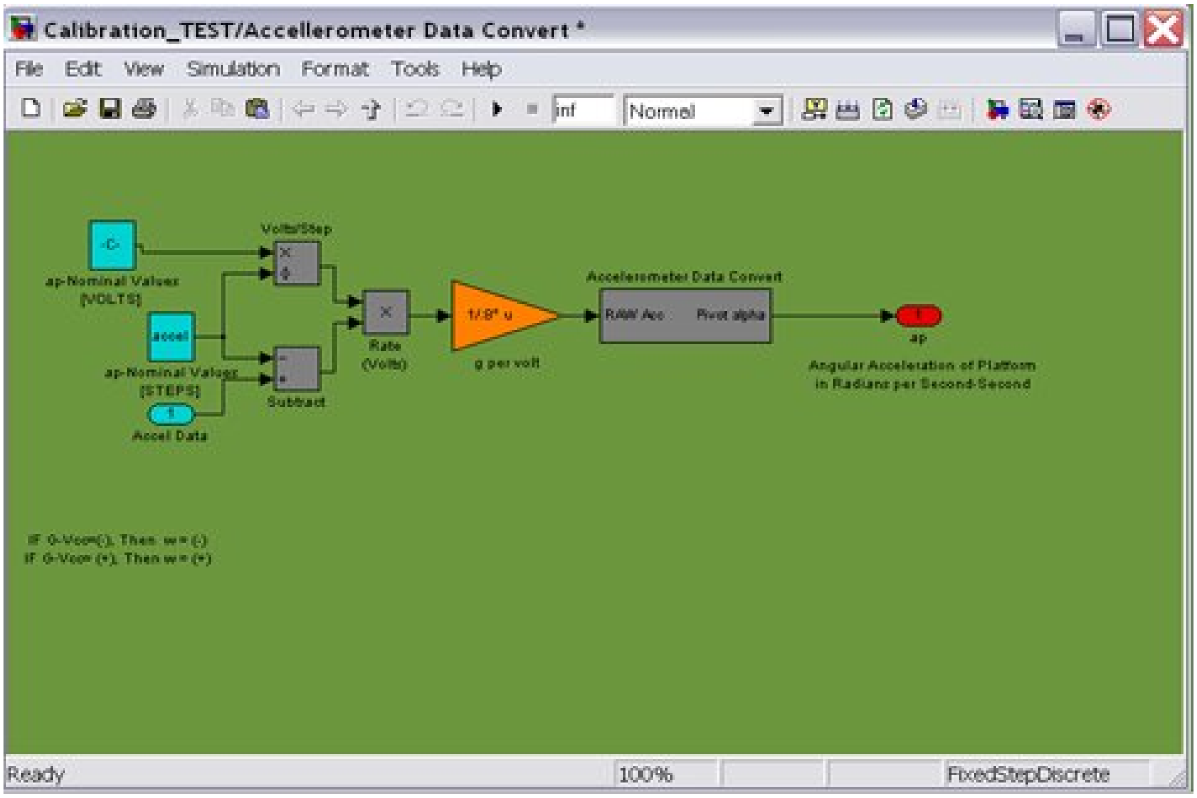
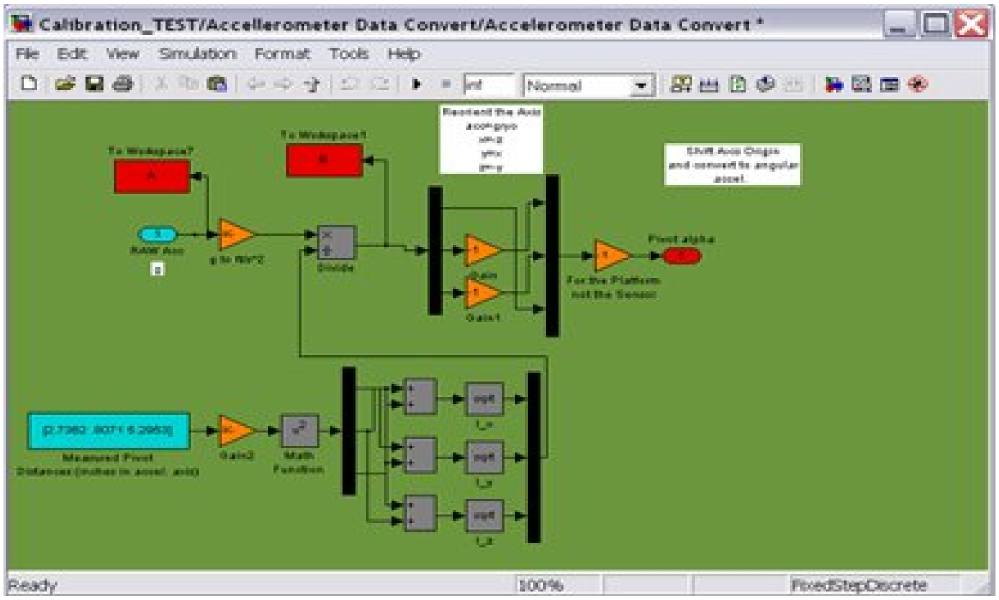

Calibration_TEST/Accellerometer Data Convert *
File Edit View Simulation Format Tools Help
inf
Normal
-C-
ap-Nominal Values
[VOLTS]
Volts/Step
accel
ap-Nominal Values
[STEPS]
1
Accel Data
Subtract
Rate
(Volts)
1/.8* u
g per volt
Accelerometer Data Convert
RAW Acc
Pivot alpha
1
ap
Angular Acceleration of Platform
in Radians per Second-Second
IF 0-Vcc=(-), Then w = (-)
IF 0-Vcc= (+), Then w = (+)
Ready
100%
FixedStepDiscrete

Figure C26: Accelerometer Data Convert



C.2.6 TAC Conversion

These components translate TAC duty cycle to RPM and vise versa as described by the data in Appendix D.

Figure C.27: ADC to RPM Conversion Main

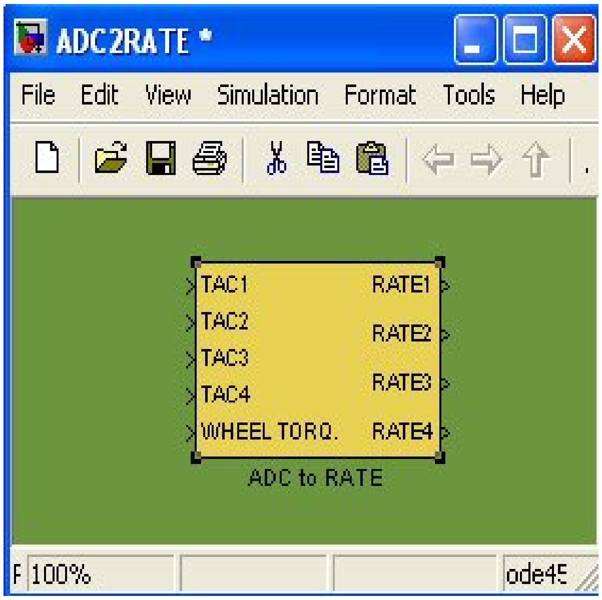


Figure C.28: ADC to RATE block

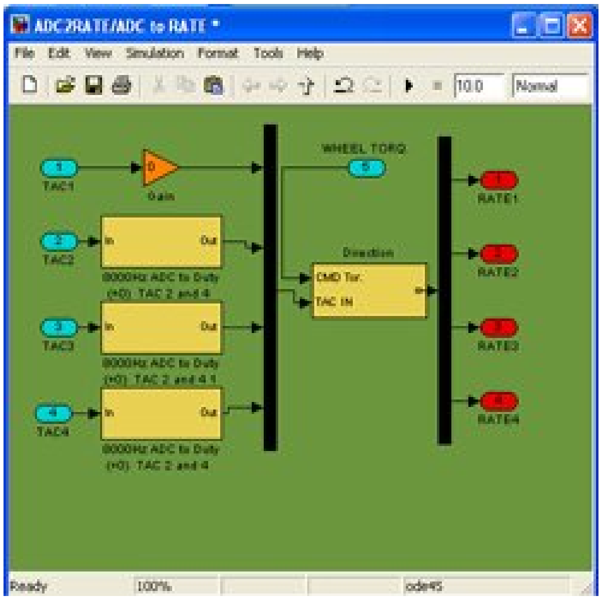


Figure C29: ROBOSTIX PWM 8000Hz Send Conversion based upon Experimental Results

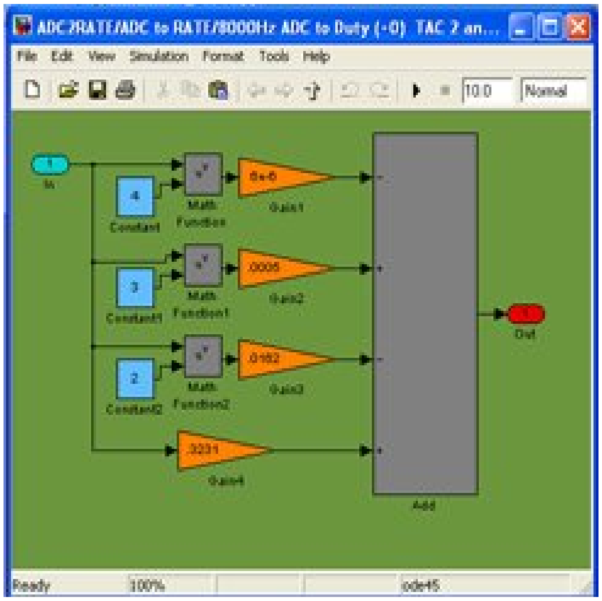
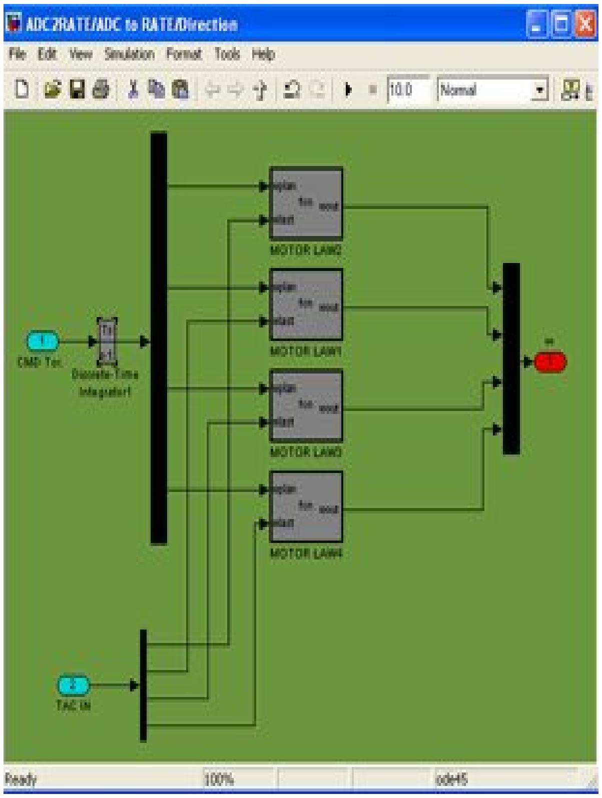


Figure C3: Direction Sensor For Each Wheel



Embedded Code:

MOTOR LAW:

```
function [CMD,wout]  = fcn(wplan,wlast)

CMD=1;
wplan=wplan*sign(wlast);
```

APPENDIX D: WHEEL RATE TEST RESULTS

D.1 1000 Hz

Table D.1 Magnetometer Test

Magnet Test			
Duty Cycle	Hz	Rad/s	RPM
WILL NOT SPIN		WILL NOT SPIN	
1	7.5	47.12389	450
1.2	9.7	60.9469	582
1.4	11.9	74.76991	714
1.6	14.1	88.59291	846
1.8	16.3	102.4159	978
2	18.5	116.2389	1110
2.2	20.2	126.9203	1212
2.4	21.9	137.6018	1314
2.6	23.6	148.2832	1416
2.8	25.3	158.9646	1518
3	27	169.646	1620
4	36.25	227.7655	2175
5	45.45	285.5708	2727
6	54.65	343.3761	3279
7	63.85	401.1814	3831
8	73.05	458.9867	4383
9	82.25	516.792	4935
10	91.45	574.5973	5487
11	100.65	632.4026	6039
12	109.85	690.2079	6591
12.1	110.77	695.9884	6646.2
12.2	111.69	701.769	6701.4

Table A.3 Encoder Oscilloscope Analysis

Encoder Wave Length Test			Measured Volts	
Duty Cycle	RPM	HZ	Duty Cycle	Volts
WILL NOT SPIN			WILL NOT SPIN	0.61
1	500	8.333333	1	0.71
2	1500	25	2	0.72
3	2500	41.66667	3	0.73
5	4049	67.47638	4	0.74
7	5000	83.33333	5	0.75
9	5319	88.65248	6	0.75
11	5556	92.59259	7	0.76
12	5714	95.2381	8	0.77
13	5814	96.89922	9	0.77
15	5848	97.46589	10	

NOTE: Orange values in Tables D.1 and D.2 were physically measured, while the black values were either calculated or extrapolated.

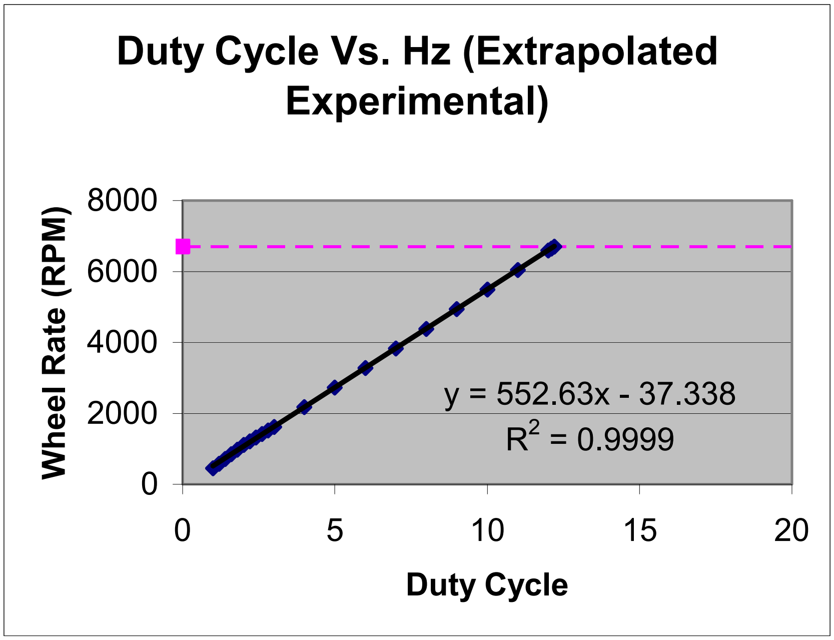


Figure D.1 Switch Board Component List

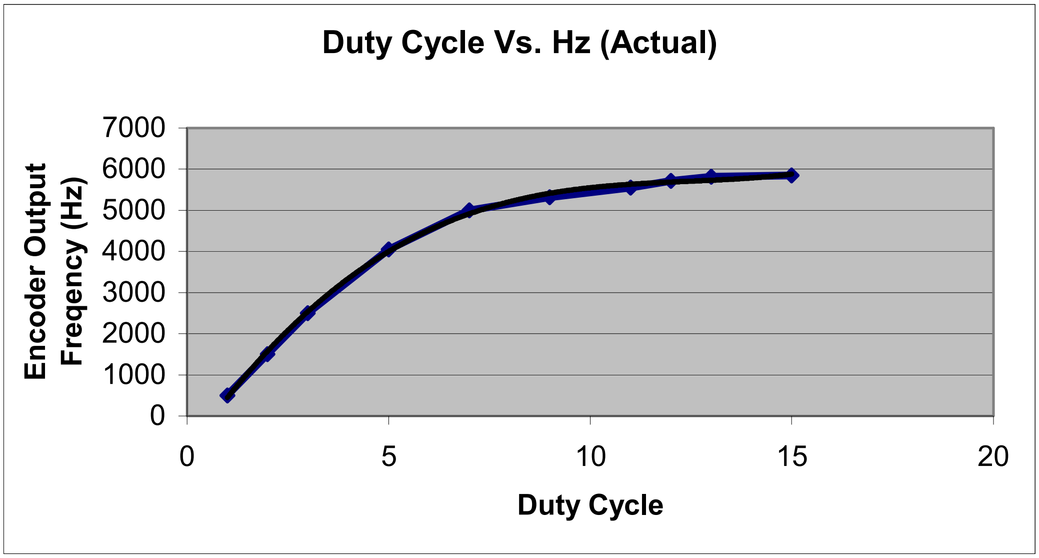


Figure D.2 Switch Board Component List

D.2 8000 Hz

Table D.3 Encoder Wave Length Test (Via Oscilloscope)

Duty Cycle	Recorded Val	Time Const	Wave Length (usec)	Encoder	HZ	RPM	Voltage	ADC (+3)	ADC (+0)
0	0	0	0	0	0	0	16.1	3	0
2	4	200	0.0008	1250	19.53125	1171.875	68.4	14	11
5	3.9	100	0.00039	2564	40.0641	2403.846	184	38	35
10	5.1	50	0.000255	3922	61.27451	3676.471	312	64	61
15	4.1	50	0.000205	4878	76.21951	4573.171	403	83	80
20	3.6	50	0.00018	5556	86.80556	5208.333	451	92	89
25	3.4	50	0.00017	5882	91.91176	5514.706	487	100	97
30	8.16	20	0.0001632	6127	95.74142	5744.485	512	105	102
35	7.8	20	0.000156	6410	100.1603	6009.615	531	109	106
40	7.55	20	0.000151	6623	103.4768	6208.609	544	111	108
45	7.45	20	0.000149	6711	104.8658	6291.946	556	114	111
50	7.4	20	0.000148	6757	105.5743	6334.459	565	116	113

NOTE: Orange values in Tables D.1 and D.2 were physically measured, while the black values were either calculated or extrapolated.

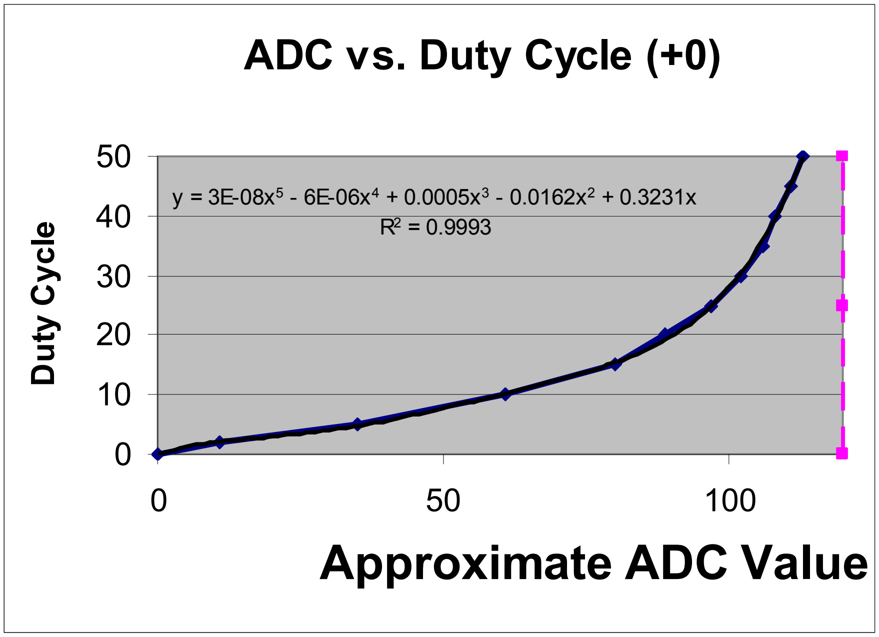


Figure D.3 Switch Board Component List

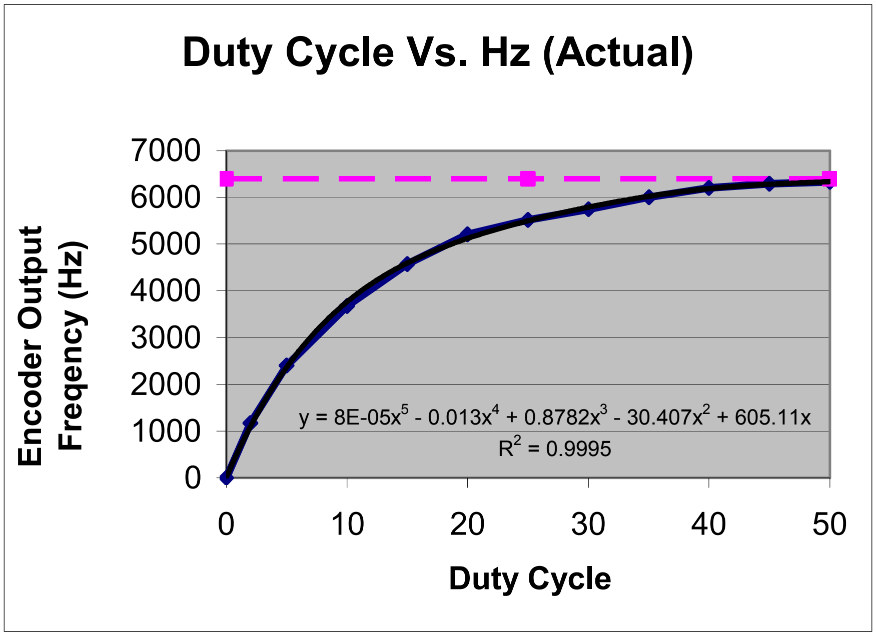


Figure D.4 Switch Board Component List

D.3 ADC Conversion Numbers

Duty Cycle	Encoder	HZ	RPM	Rad/Sec	ADC 1	ADC 2	ADC 3	ADC 4
0	0	0	0	0	0	0	0	0
2	1250	19.53125	1171.88	122.72	0	15	22	0
5	2740	42.80822	2568.49	268.97	0	28	45	5
10	4157	65.10417	3906.25	409.062	0	43	60	21
15	5128	80.12821	4807.69	503.4604	0	54	68	44
20	5663	88.47678	5308.61	555.916	0	63	72	63
25	5952	93.00595	5580.36	584.3736	0	67	77	76
30	6250	97.65625	5859.38	613.5923	0	73	82	83
35	6494	101.461	6087.66	637.4985		75	90	90
40	6614	103.3399	6200.4	649.304		76	96	96

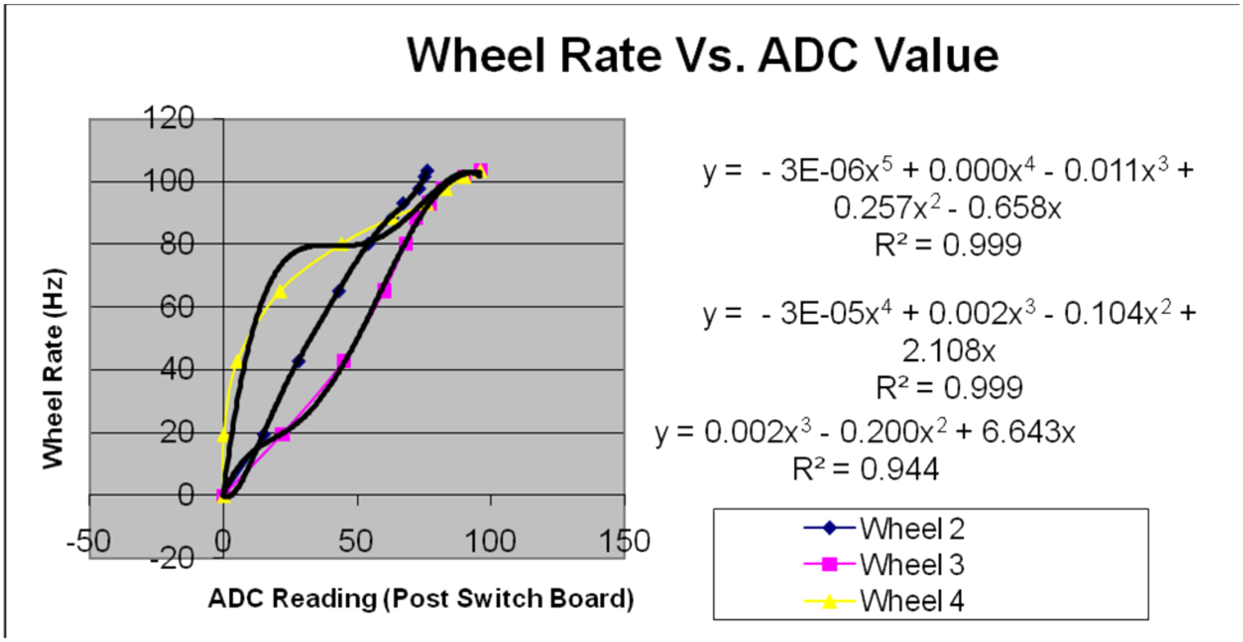


Figure D.5 ADC to Wheel Rate

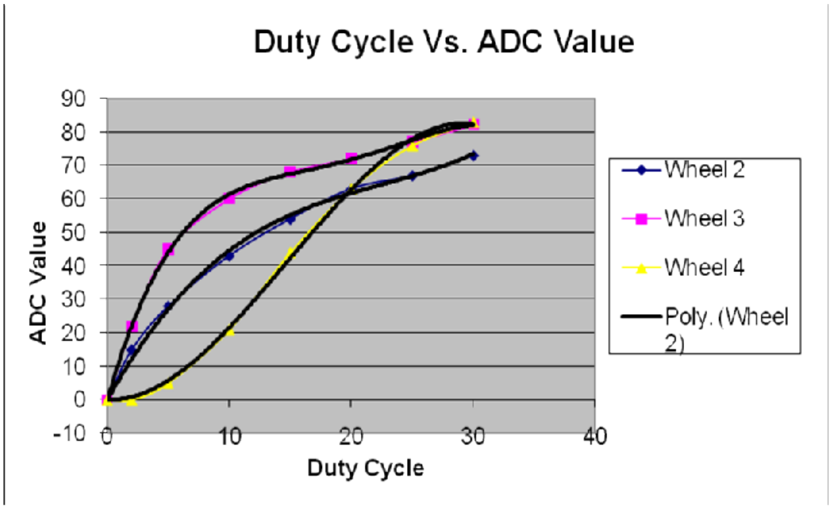


Figure D.6 ADC to Duty Cycle