MASTER THESIS

Hardware and Software Implementation of a Control Pod for Airborne Wind Energy

***Estudis**: Enginyeria en Electrònica/Màster d'Enginyeria Electrònica*

***Autor**:* Josep Albert Caba Monton

Director/a: Daniel Toal

***Any**: 2011-2012*

To my family. Thanks for always being there for me.

ACKNOWLEDGMENT

I would like to acknowledge the advice and guidance of Dr. Daniel Toal. I would also like to thank Dr. Edin Omerdic for all the help and time spent teaching and helping me. Finally, I want to thank Joseph Coleman, for his priceless help and unconditional support.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

ABSTRACT

Implementation of the electronic part of a parafoil controller for wind farming at high altitudes was performed. The system consists in a lightweight kite controlled by an airborne pod which allows the system to be elevated to high altitudes by the wind, negating the requirement for propulsion or a static tower. Implementation includes the choosing of the electronic parts/devices within the system, and also designing the software to allow the system meet the projects specifications.

The control part of the system is provided by an FPGA and a microprocessor. These operate the airborne sensors and actuate two servo motors. These control kite's orientation in order to get the system to the desired position or maintain it, based on some sensors inputs- primarily an inertial measurement unit, through the programming of an FPGA and microprocessor.

The proper positioning of the pod is achieved, either with a manual control, based on a joystick with a human pilot or an automatic one, based on sensors inputs. This method was shown to be an effective way of controlling a control pod based parafoil system as demonstrated by laboratory bench tests and recent flight test in the field.

1. INTRODUCTION

Airborne Wind Energy (AWE) is the sector which investigates the production of energy from wind at high altitudes, much higher altitudes than conventional wind turbines, because of the stronger wind which leads to increased energy production.

This project is based on an AWE system that uses lightweight parafoils, used to maintain the kite on the air without any energy consumption, by the effect of the lift produced by the air, which translates in a more effective energy production system and cheaper to implant, as no additional infrastructure is needed.

1.1 WIND FARMING AT HIGH ALTITUDES

As it is well known, the speed of the wind increases proportional to the altitude, so the higher the altitude in which the wind turbine is situated, the more the energy that can be produced from it, and for longer periods(Canale et al.,2010), as its seen in the following figure:

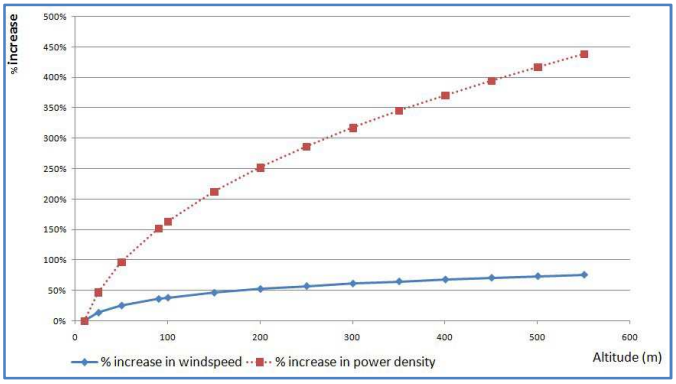


FIGURE 1. INCREASING WINDSPEED AND POWER DENSITY IN RELATION TO ALTITUDE(O'GAIRBITH 2010)

This is especially interesting in Ireland, featuring some of the most powerful and consistent winds in Europe.

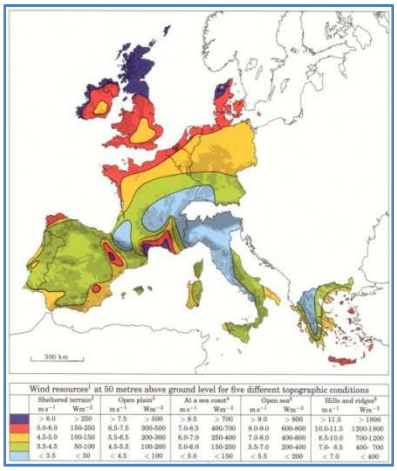


FIGURE 2.EUROPEAN WIND ATLAS, (1989), RISO NATIONAL LABORATORY

Another important feature is the fact that as the power needs increases, so does the size of the terrestrial wind turbines, in order to produce more energy, translating into increasing engineering and material costs(O' Gairbhith, 2009). This increasing costs are making the wind farming to stagnate, so one possible solution to this problems could be the airborne wind energy.

1.2 PROJECT INTRODUCTION

Taking all the advantages introduced in the previous section(1.1), the idea surrounding this project is to develop the electronic subsystem of a parafoil control pod, that is able to remain airborne using very little or no energy apart from the wind. Being able to control its position in an automatic way, using the feedback of an MEMs inertial measurement unit, or manually, by the use of a joystick is the aim of this project, which into the future it will enable the extraction of wind energy at higher altitudes than currently possible with terrestrial tower turbines.

The main problems found on AWE systems is the low stability they present, leading to have to implement an active stabilisation system and being able to follow a reference trajectory, so they can be used in the generation of electricity.

This project's aim is the development of the airborne part, including all the design regarding the pod and the communication system within all the system, but not designing the ground based system such as the winch, whose rotation generates electricity, as is going to be explained in the next section(1.3).

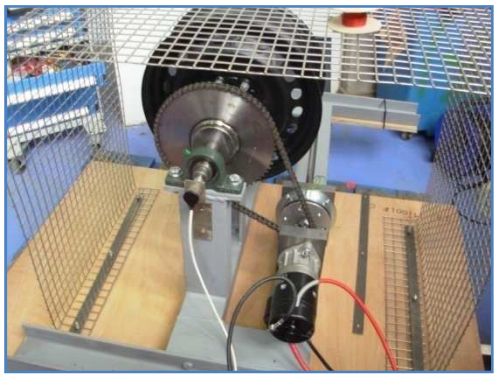


FIGURE 3.WINCH WITH RECOVERY MOTOR

1.3 System Description

In the first place, the parafoil is launched, the tension present in the tether by its dynamic properties, making the rope to unwind from the ground drum. This drum is at the same time connected to a generator, so the rotation of this drum produces electric energy.

When the kite is in the highest possible altitude, the tension present in the tether is reduced by modifying the foils aerodynamic profiles. During this phase, a motor helps the parafoil to return to its original position, making the system to work in a cyclical manner. The power used during the recovery phase is much lower than the one produced(Coleman, 2011).

On the following figure is shown the power and recovery phases:

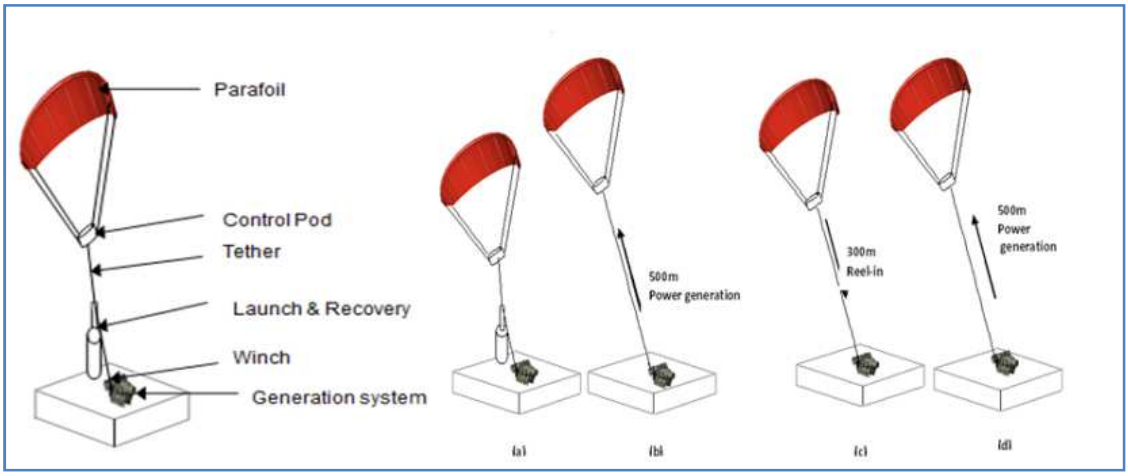


Figure 4.System Outline(Coleman,2011)

2. HARDWARE

The purpose of this chapter is to describe the different parts that are needed to allow the airborne to fulfill its objective: being able to stay in the air, and control its position. For this purpose there are two motors which modify the orientation of the kite by pulling or drop the kite's rope. The motors(and the position for instance) is controlled manually with LabVIEW's user interface or by joystick. In order to know the orientation, acceleration and speed of the kite and the airborne there are implemented two inertial measurement units (IMU). With the purpose of exchanging information with the ground station there is a Wifi router. All the control part is within the Single-Board RIO (SbRIO).

The first draft of this prototyping is expected to be able to keep online for at least 30 minutes, for this reason, the battery selection will be deeply described.

The following figure shows the general hardware layout of the project :

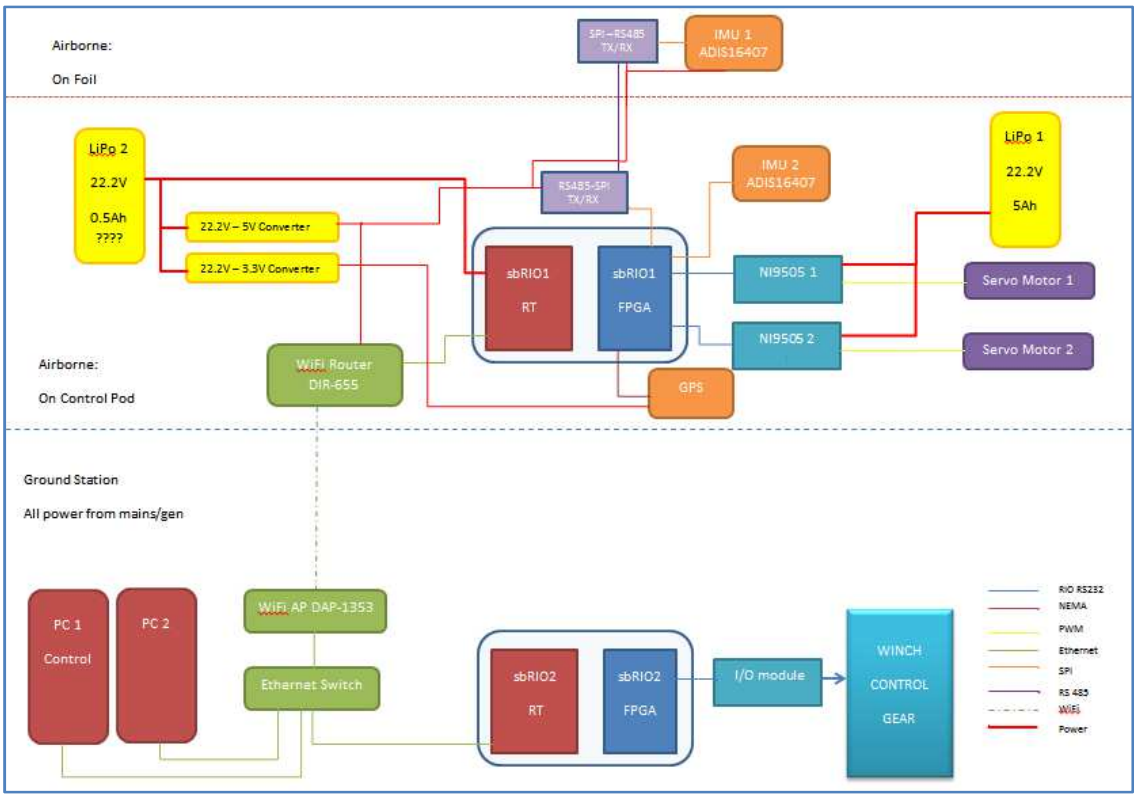


FIGURE 5. AIRBORNE AND GROUND STATION HARDWARE LAYOUT

2.1 AIR PART GENERAL DIAGRAM

On the airborne system, the position of the pod must be tracked, for this purpose a GPS will be used . The translational body accelerations and rotational rates on the pod and on the kite are acquired through the use of two IMUs , one situated at both of these locations. The airborne layout is shown in the following figure:

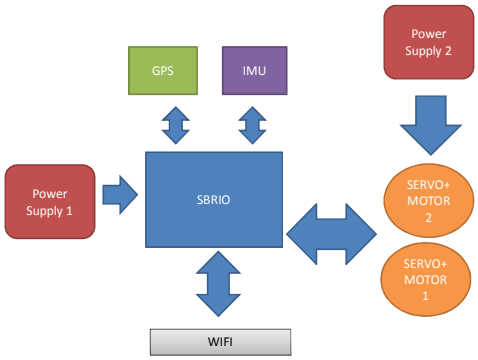


FIGURE 6.AIRBORNE LAYOUT

2.1.1 SBRIO

2.1.1.1 DESCRIPTION

The Single Board RIO, from National Instruments, is an advanced embedded control and data acquisition system designed for applications that require high performance and reliability . It features an embedded real-time processor for stand-alone or distributed operation. There is also an FPGA chip that provides the possibility of adding logic to the system.

An SbRIO-9602 was chosen for this project due to its multiple I/O (110) , and its dimensions and weight. It was also chosen because it operates over TCP/IP protocol, which simplifies the communication with other devices, like computers. The digital I/O are 3.3V TTL but 5V tolerant.

The processor is 400Mhz , 256 MB of non-volatile storage and 128 MB of DRAM for deterministic control and analysis. It has one Ethernet port (10/100 Base-T) and an RS232 port.

The FPGA has a capacity of 2M gate reconfigurable I/O .

The operating temperatures range from -20 to 55 ºC . The XT version for extreme temperatures has not been considered necessary as the application falls within the previously specified temperature range.

2.1.1.2 WHY SBRIO?

SbRIO is a National Instruments plataform that operates with Vxworks as OS. This allows it to interface with LabVIEW, enabling a rapid graphical and more intuitive programming environment. Speed of development and reliability are the main motivations behind a SbRIO/LabVIEW approach.

Another important feature is that it permits flexible embedded hardware , as there are a lot of expansions modules available from National Instruments, that allow prototyping deployment for almost all possible applications while staying in the same platform. For example, LabVIEW has Softmotion Module that allows advanced control regulation with better performance and flexibility. The human interface is also an important factor, as LabVIEW allows the rapid development of intuitive interfaces for the user .

The LabVIEW Real-Time OS provides the ideal environment for time-critical applications that require deterministic performance. Through precise timing and task prioritization, deterministic tasks such as closed-loop motion control can be easily developed in LabVIEW and deployed to SbRIO . The LabVIEW Real-Time OS also provides an optimized environment designed to ensure applications run reliably.

The SbRIO implementation can be split into three different steps. The first one is to implement the FPGA interface needed for the project. The next one, is to link the FPGA interface to the Real-Time Interface , where the SbRIO develops real-time deterministic algorithms to fulfill the specified tasks. The last step is to link the two previous interfaces with LabVIEW Windows , to build the user-interface and any PC run control tasks, as shown in the following figure :

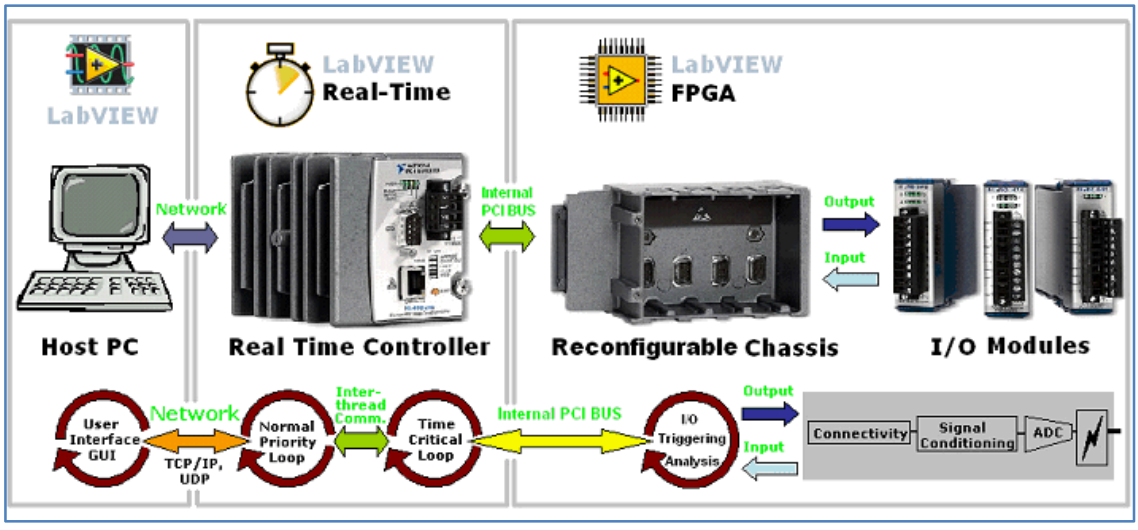


FIGURE 7.INTERFACE LABVIEW WITH SBRIO.
NATIONAL INSTRUMENTS, *WHY TO CHOOSE SBRIO?,VIEWED 1ST NOVEMBER 2011* HTTP://WWW.TRACNOVA.COM/TRACNOVA-PUB/CRIO.PDF

2.1.1.3 REAL-TIME INTERFACE

To be considered "real-time", an operating system must have a known maximum time for each of the operations that it performs (or at least be able to guarantee that maximum most of the time). Some of these operations include OS calls and interrupt handling. Operating systems that can absolutely guarantee a maximum time for these operations are referred to as "hard real-time", while operating systems that can only guarantee a maximum most of the time are referred to as "soft real-time". To fully grasp these concepts, it is helpful to consider an example.

The main point is that, if programmed correctly, an RTOS can guarantee that a program will run with very consistent timing. Real-time operating systems do this by providing programmers with a high degree of control over how tasks are prioritized, and typically also allow checking to make sure that important deadlines are met.

LabVIEW Real-Time is a tool used for performing deterministic floating point processing of the data acquired in the FPGA application. The FPGA interface pallet makes it easy to perform real-time communication between the FPGA and the real-time or Windows host application. FPGA registers are created simply by defining a front-panel control or indicator on the FPGA application, and can then be written or read by the host at high speed while the applications are running.

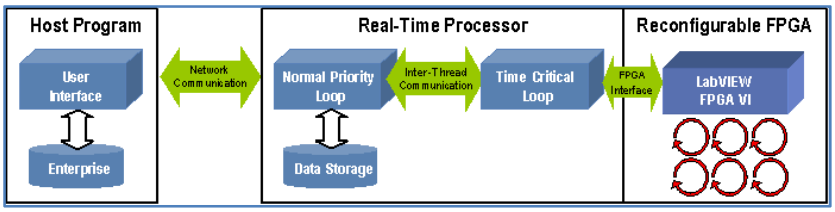


FIGURE 8.REAL TIME INTERFACE WITH SBRIO'S FPGA MODULE

NATIONAL INSTRUMENTS, *LABVIEW WITH CRIO TUTORIAL,VIEWED 1ST NOVEMBER 2011 < HTTP://CATS-FS.RPI.EDU/~WENJ/ECSE446S06/LABVIEWCRIOTUTORIAL.PDF?Q=~WENJ/ECSE446S06/LABVIEWCRIOTUTORIAL.PDF>*

2.1.1.4 FPGA INTERFACE

LabVIEW FPGA application development is the first step involving interaction with the I/O modules and analyzing the acquired signals. The FPGA makes it easy to create advanced applications requiring high speed control, precise timing, triggering and synchronization. LabVIEW FPGA offers intuitive function blocks (called Virtual Instruments or VIs) that make the process of acquiring and generating both analog and digital signals straightforward. Functions from the FPGA Device I/O pallet can be used for any application requiring interaction with signals; ranging from a simple analog input reading from a thermocouple to generating PWM signals for precise control of a servo motor. The hot swappable Input/Output modules acquire the signals, perform signal conditioning and communicate the digital signals to the FPGA.

2.1.2 IMU

2.1.2.1 OBJECTIVES OF THE SENSOR

The IMU provides the body translational accelerations $(\dot{u}, \dot{v}, \dot{w})$, body angular velocities: (P, Q, R) and body magnetic fields on the airborne pod and also on the kite.

2.1.2.2 DESCRIPTION

The ADIS16407 device is a complete inertial system that includes a triaxial gyroscope, a triaxial accelerometer, a triaxial magnetometer, and pressure sensors. Each sensor in the ADIS16407 includes signal conditioning that optimizes dynamic performance.

One of the main advantages of this IMU is the simplicity of its communication , as all the calibration and configuration is done in the factory, so the communication with it is the only feature that has to be solved. This communication is implemented through the serial peripheral interface(SPI). The different information is stored in different registers which can be accessed, also there is a configuration register which can be modified in order to modify the behavior of the sensor.

2.1.2.3 SPI PROTOCOL

The SPI or serial peripheral interface is a full-duplex serial protocol. The data exchange requires synchronization to an interface clock signal. There is one master and it can have one or multiple slaves. The role of the master is to initialize the communication, specify a clock for the communication (SCLK), and send a command (read or write) to the slave when the communication has been established.

To begin a communication , the master has to make an edge (descendant or ascendant depending on the default active state) , and send and/or read the bites during the ascending/descending edge (also depending on the configuration) of/from the slave , as specified in the following figure:

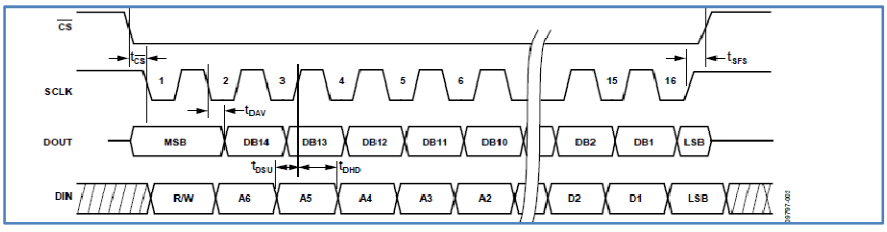


FIGURE 9.SPI COMMUNICATION EXAMPLE

The CPOL and CPHA are the configuration that will depend the edge of actuation of the SPI. In the current case the CPOL=CPHA=1 , meaning that the base value of the clock is one , and data is captured in clock's rising edge and data are propagated in the falling edge.
The current configuration and implementation are detailed later in the third chapter (Software) , within the IMU section.

2.1.2.3.1 DISTANCE PROBLEM

As stated in the beginning of the chapter, two IMUs sensors will be used: one on the kite and one on the pod, to exactly know the state of both elements at every moment. The SbRIO will be on the pod, which creates a wiring distance problem between the SbRIO and the kite's IMU , because the distance (around 5m.) is too long for the SPI protocol , as it is specifically for PCB (<1 feet) .

With byte lengths ranging from 8 to 12 bits and multiples thereof, and data rates ranging from 1 to 20 Mbps, the standard SPI configuration allows for short propagation times and hence only short distances in order to maintain synchronicity between the interface clock and the data transmitted in both directions. Over long distances, however, the transmission cable introduces significant propagation delay into the signal path, causing the SPI to be incompatible for this cases. To solve this , and to avoid using an SPI extender, because they are very expensive, RS485 transceivers have been used in order to convert normal signal into a differential signal , allowing long distance transmission.

The solution has been implemented through an RS485 transceiver : sn65hvd30. Four units are required in each side of the communication (8 in total), three(yes 3 sending and one receiving,4 in each part , so 8 in total) drivers and one receiver in the SbRIO side , and the opposite in IMU side.

On the following picture the diagram of the connection is shown (notice that CS is not implemented for simplification of the figure.

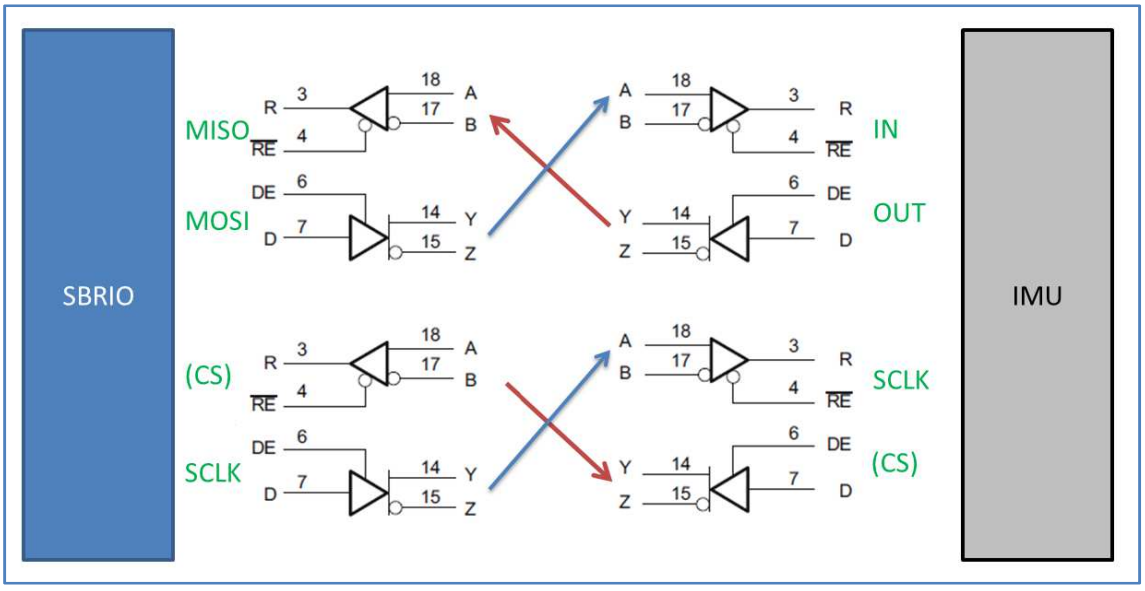


FIGURE 10. TRANSCEIVERS CONNECTION DIAGRAM

2.1.2.4 SCHEMATIC

As stated above, the system consists of two IMU , one on the kite and one on the control pod, the wiring diagrams for these are illustrated below:

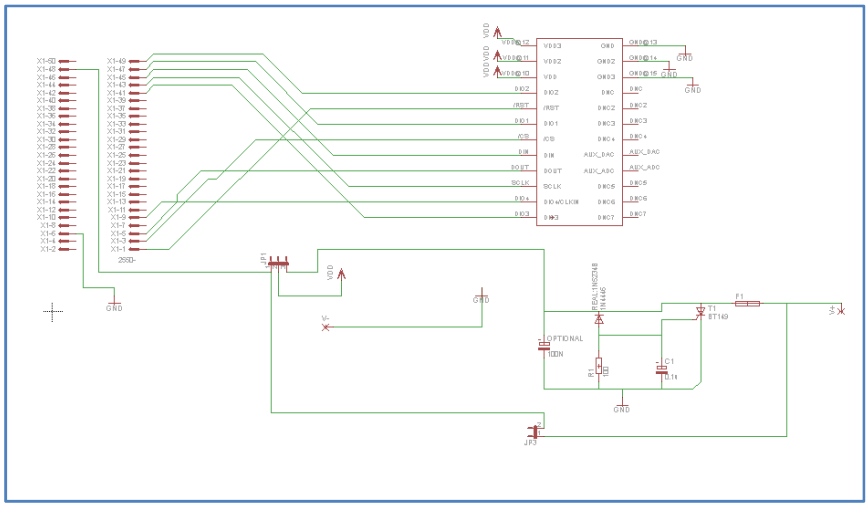


FIGURE 11.SCHEMATIC OF AIRBONE'S IMUS

The IMU situated in the pod does not require the use of transceivers , as they are placed less than 30 cm away from the FPGA inputs. The power supply comes straight from the SbRIO output , as it has an isolated, overvoltage protected 5V supply pin. Even with this protection there is also the possibility to add overcurrent protection by closing the jumper to the third pin. This protection circuit when detects an overcurrent , an before the fuse acts , the thyristor is activated and redirects the current to the resistance , preventing it reaching the IMU.

The NI SbRIO 5V power supply has a total of 2 A external load at 5 V. This total includes 200 mA per installed C Series module. So in this case two C Series modules are installed, so:

$$2\,A - (2 \times 0.2) = 1.6\,A \text{ is available for use}$$

Each pin in the headers is rated for 2 A, but a typical 28 AWG ribbon cable is rated for only 225 mA per conductor.

For this reason, and more than 500mA is being drawn (there are only two 5V connector per header) , in the final board the power to supply the elements comes from converter coming from the battery supplying the SbRIO, because if not , there is a voltage loss over the cable due to heating effects, so it is internal resistance is increased (as more current than the maximum the cable has been designed is being drained) that results in voltage losses.

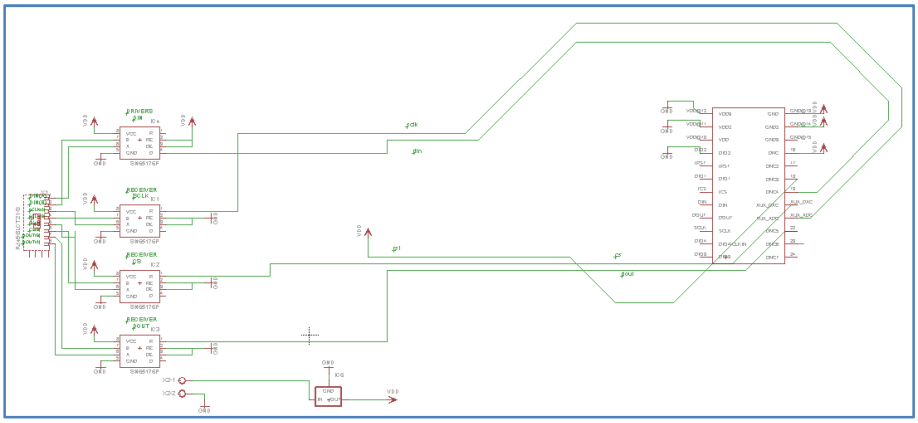


FIGURE 12.SCHEMATIC OF KITE'S IMU

On the kite IMU a converter is provided to supply the IMU. The power comes from SbRIO's battery. There are four transceivers , three configured as receivers (SCLK,DIN,CS) and one configured as a driver (DOUT). The outputs of the transceiver (each one is formed of two signals as it is converted into a differential signal) are connected to a RJ45 header , than connects the Kite's IMU board to receiver part situated on the pod.

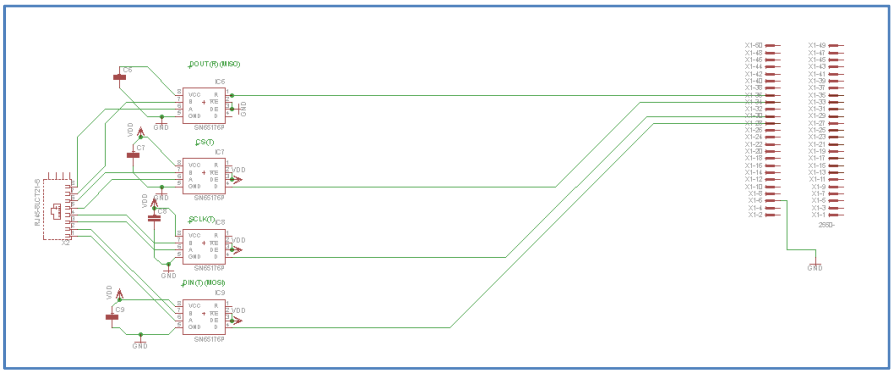


FIGURE 13.SHEMATIC OF KITE'S IMU RECEIVING PART

The receiver part is formed by four transceivers , three configured as drivers (SCLK,DIN,CS) and one as receiver (DOUT). The data arrives into the RJ45 connected to Kite's IMU board , and therefore connected to SbRIO FPGA input. There is also a capacitor situated in each IC to filter the noise caused by IC commutations.

2.1.2.5 PCB

The PCBs of the previously stated schematics are shown above:

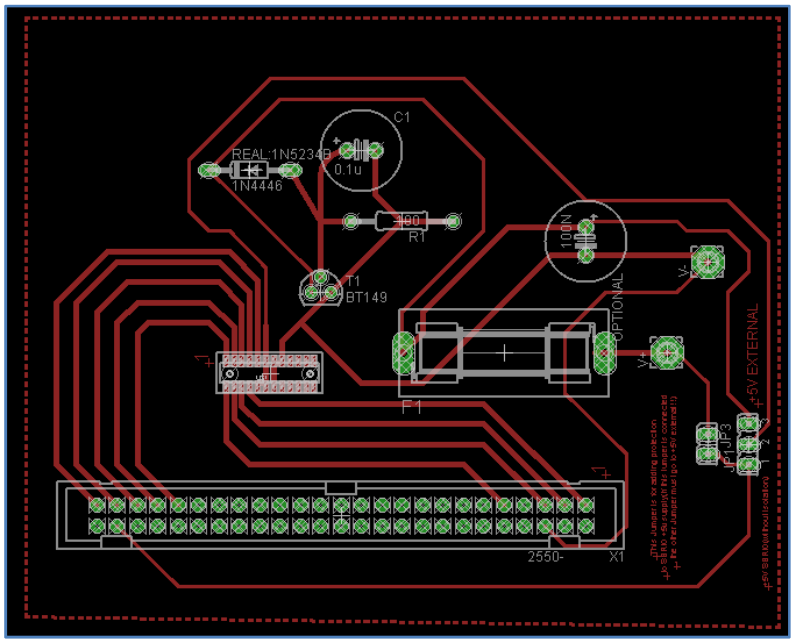


FIGURE 14.SBRIO'S IMU PCB

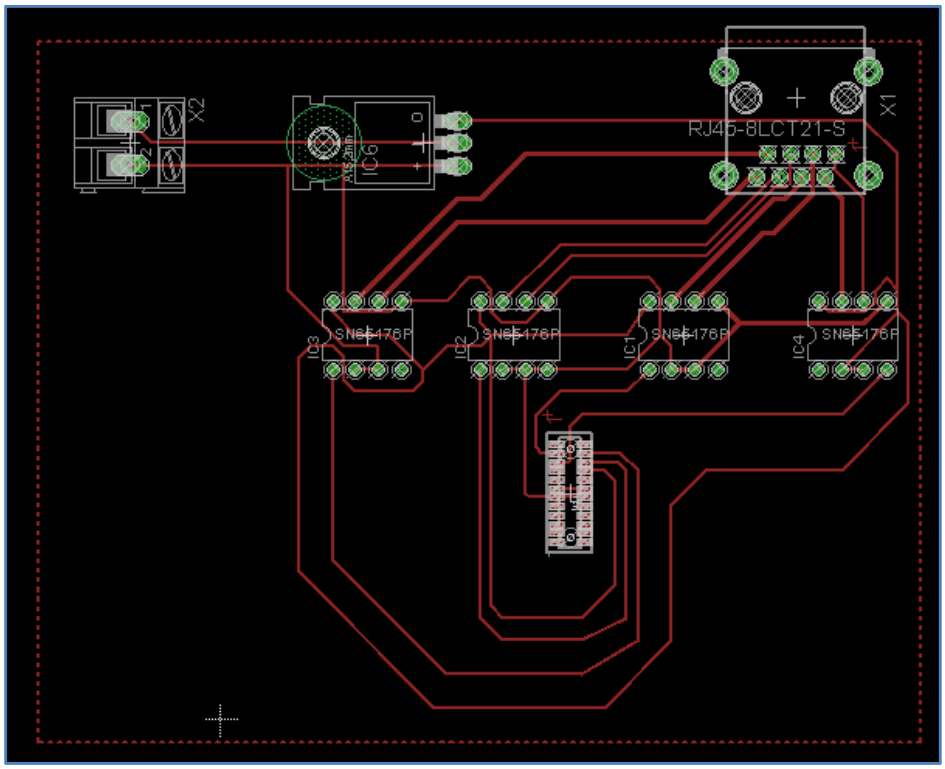


FIGURE 15.KITE'S IMU PCB

2.1.3 MOTOR

2.1.3.1 OBJECTIVE

The motors have to provide actuation to the foil , allowing it to be steered and its lift force altered.

2.1.3.2 DESCRIPTION

In the project two DC motors are used . A brushed DC motor design was chosen , for several reasons , but the main one was the power to weight ratio. Another important feature was the price: a brushed motor is a lot cheaper than a brushless motor because its market is more mature , so there is more offer that translates in cheaper prices. Also it provides high torque and low speed(this feature is amplified by the gearhead).

In the future , if the project accomplishes the targets of the prototype phase , and a commercial use starts to be developed , the study of implementing a brushless motor instead of a brushed motor will be studied , as the first one would allow longer periods of activity from the kit without maintenance .

2.1.3.3 WHY THIS MOTOR

The motors RE40 from Maxon , reference number : 148867 have been chosen for the project. It is a 150W motor with a nominal voltage of 24V.

The primary advantage of this motor is that it has the highest power to weight ration: its nominal torque (170mNm) is more than enough for the application, never needing a bigger torque than 0,1308Nm.

Another point to have in consideration , is the fact that the voltage levels are standard in the polymer lithium batteries that are going to be used in the project(for longer details refer to Batteries chapter) , and the current can be also fulfilled by the battery.

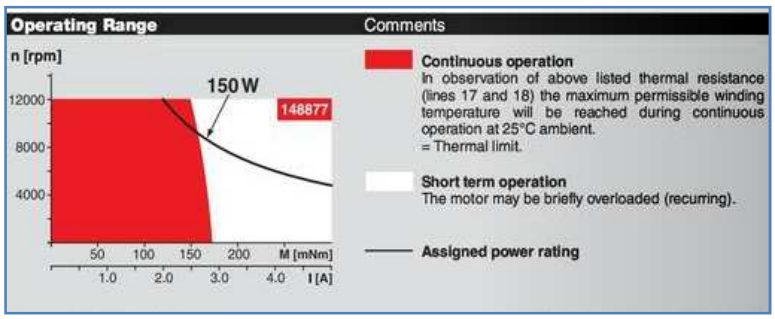


FIGURE 16.RE40 OPERATING RANGE

MAXON,*RE40DATASHEET*,1ST DECEMBER 2011 HTTP://TEST.MAXONMOTOR.COM/DOCSX/DOWNLOAD/CATALOG_2005/PDF/05_083_E.PDF

2.1.3.4 GEARHEAD

The gearhead is the responsible of increasing the torque and reducing the speed. In the project the following planetary gearhead has been chosen:GP 42C reference 203115.

This has been chosen because it provides the necessary conversion ratio for the project, as it divides motor speed by 15, but increases its torque by the same amount. The maximum torques and motor speed are discussed in greater detail within the power consumption section(2.1.8).

2.1.3.5 PROBLEMS

One of the main disadvantages of the RE40 motor is as mentioned is fact that is a brushed motor design. This reduces the overhaul lifetime of the motor as the brushes may need to be replaced periodically. This motor design does however provide the lightest weight solution.

2.1.3.6 DC MOTOR MODEL

In this section , a dynamic model of a direct current motor is described by using basic electromagnetics relationships.

By Ampere's law it is know that a force F will be produce whose direction is orthogonal to both the current and flux and whose magnitude is given by :

$$F = N \cdot B \cdot l \cdot i$$

Where :

l is the length of wire within the field

i is the current flowing in the wire

N is the number of turns of the coil

The field B is produced by a two or more pole steel magnet. This part is know as the stator.

In a dc motor we will have a coil , supported by a shaft in which it rotates. This is know as the rotor, as can be seen on the figure below:

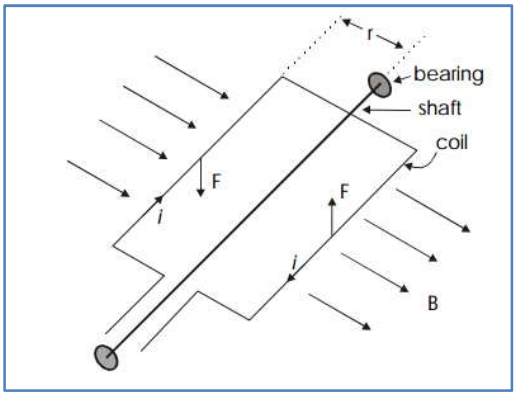


FIGURE 17.DC MOTOR ROTOR

Each part of the rotor is facing to a different pole, making that both parts are rotating on the same way, and the net effect of the two forces generates what is known as torque, with the following equation:

$$T = 2 \cdot F \cdot r$$

Making the substitutions from the first equation we have that:

$$T = 2 \cdot N \cdot B \cdot l \cdot i \cdot r$$

N,B,l and r are a characteristic of each motor, so they can be considered a constant, so:

$$T = K \cdot i \quad (1)$$

To avoid that when 180 degrees are reached the esu of the rotation changes, a commutator is added to the motor, so when it reaches this point, it reverses the current, making it to be opposite to the previous one, translating in a turning in the same way of the motor for all the positions of rotation.

On the other hand, this rotation will produce an e.m.f. which is given by:

$$e = -\frac{d\lambda}{dt}$$

Where:

λ is the current flux

$$\lambda = N \cdot \Phi$$

Where:

Φ is the magnetic flux

$$\Phi = 2 \cdot B \cdot l \cdot r \cdot \cos\theta$$

Where:

Θ is the angle between the pole-faces

Leading to the following expression:

$$e = 2 \cdot N \cdot B \cdot l \cdot r \cdot \theta$$

Note that once again N,B,l and r are constants, and θ can be translated into angular speed multiplied per time, getting the following equation:

$$e = K \cdot \omega \quad (2)$$

The Dc motor can be approximated by the following diagram:

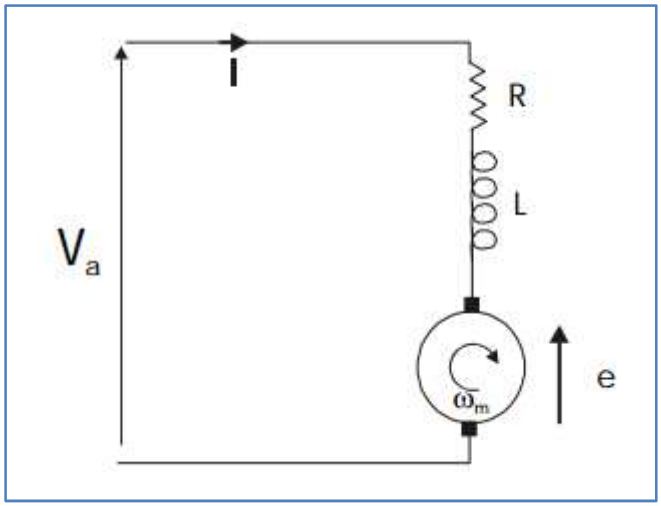


FIGURE 18.DC MOTOR MODEL

Where:

Va : Source Voltage

R: Rotor Electric Resistance

L: Rotor Coil self-impedance

e: e.m.f

Using Kirchoff's law, we get the following equation:

$$Va - R \cdot i - L\frac{di}{dt} - e = 0 \rightarrow Va = R \cdot i + L\frac{di}{dt} + e \quad (3)$$

(1),(2) and (3) form the basic equations of a DC motor.

2.1.4 NI9505

2.1.4.1 DESCRIPTION

The NI9505 is a module developed for CompactRIO and SbRIO that creates a highly customizable drive for the motor. It can control brushed DC motor of 150W .The main objective of this extension module is to help the developer to create a regulation loop for the module , including position speed and current control in the FGPA level of the SbRIO , and send the command as a PWM signal to the driver part , formed by an H-Bridge. The layout of the elements is shown below:

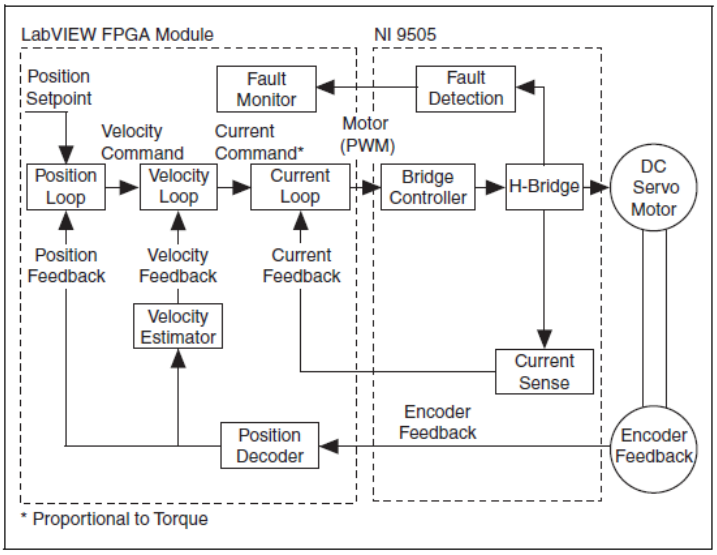


FIGURE 19.NI9505 BLOCK DIAGRAM

NATIONAL INSTRUMENTS,NI9505 OPERATING INSTRUCTIONS, 31ST *DECEMBER 2011*
<HTTP://TEST.MAXONMOTOR.COM/DOCSX/DOWNLOAD/CATALOG_2005/PDF/05_083_E.PDF>

The diagram is as follows , in the FPGA interface a PID controller can be implemented by the user to regulate the position , with a current limitation (current loop) and this translates in a velocity command , translated into a PWM signal sent to H bridge , that translates into voltage and current sent to the DC motor. The NI9505 uses as feedback the position of the encoder , and the current drawn by the motor.

The NI9505 has also implemented several protective features, and it disables the driver on the following cases:

Undervoltage	<6 V
Overvoltage	>32 V
Reverse polarity	-30 V
Motor terminal (MOTOR±) short to ground	Yes
Motor terminal (MOTOR±) short to V_{SUP}	Yes
Tem perature fault trip point	115 ºC

TABLE 1.NI9505 PROTECTION

The NI9505 can deliver voltages up to 32V , and nominal currents until 7.3A with temperatures below 40ºC . Current spikes of up to 12A for 2 seconds maximum can be sustained by the module. Although there must be at least 10 seconds between these spikes.

The optimum PWM frequency to communicate with the H-Bridge is 20kHz.

ENCODER FEEDBACK

The encoder features and mode of operation are explained in the following section, with focus in the interaction between the NI9505 and the encoder.

The encoder being used has three outputs : phase A,B and index. As the distance between the encoder and the SbRIO is short (<1m) no differential encoder should be needed , although the fact remains that even if the distance is very short a lot of noise can be seen due to proximity to motor and 50Hz interferences mainly coming from the lightning system, so the final decision was to implement a differential encoder(figure shown below) in order to gain robustness .

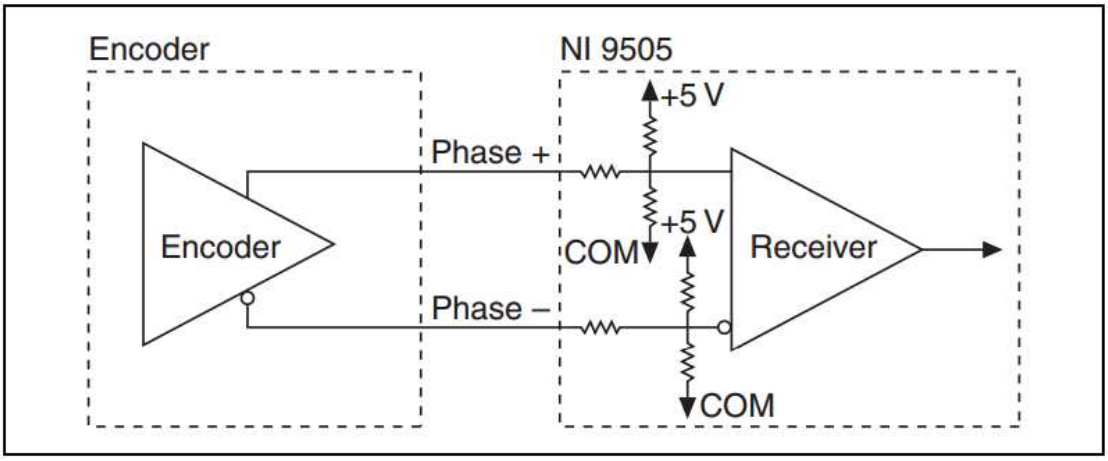


FIGURE 20.ENCODER-NI9505 CONNECTION DIAGRAM

NATIONAL INSTRUMENTS,NI9505 OPERATING INSTRUCTIONS, 31ST DECEMBER 2011
<HTTP://TEST.MAXONMOTOR.COM/DOCSX/DOWNLOAD/CATALOG_2005/PDF/05_083_E.PDF>

The encoder should also be shielded in order to minimize the noise acting in it .

CURRENT FEEDBACK

The way that the NI9505 measures the motor current, is by using a shunt in each of the possible motor directions, and calculating through and ADC(previously connected to an amplifier) the voltage present, and making the conversion into current based on the resistance value, as illustrated in the following figure:

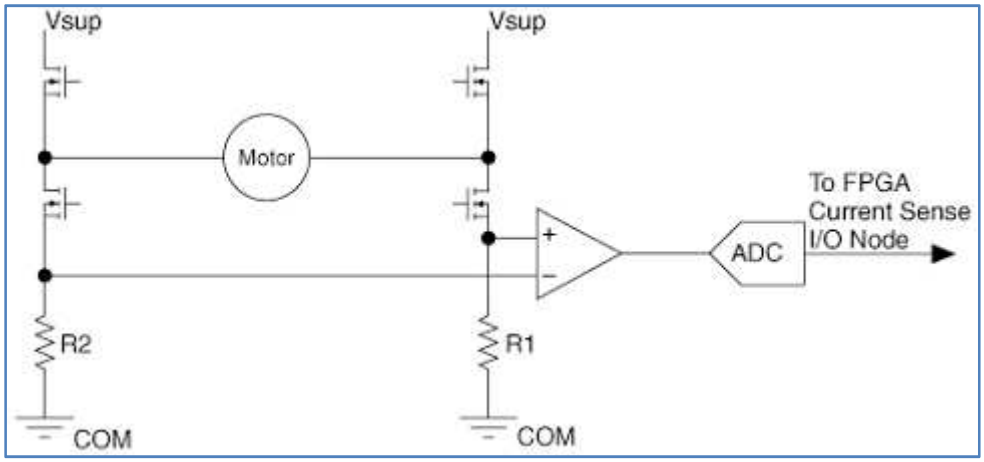


FIGURE 21.NI9505 CURRENT SENSOR SCHEMATIC

NATIONAL INSTRUMENTS,UNEXPECTED CURRENT READING ON THE NI9505, 23TH *FEBRUARY 2012* <HTTP://DIGITAL.NI.COM/PUBLIC.NSF/ALLKB/812D5CAD54BAF478862576C7007D6C83>

During the ON time of the PWM,all the current goes through R1, based on the state of the transistors, as stated below:

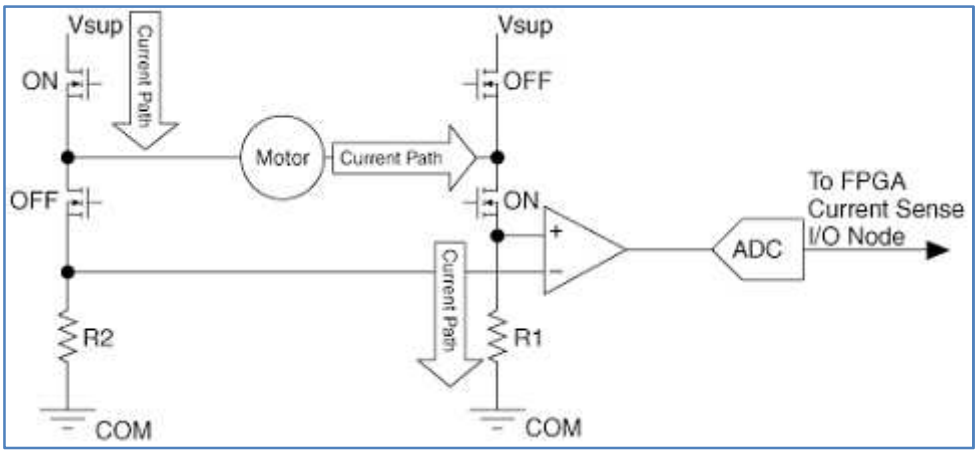


FIGURE 22.CURRENT SENSE DURING ON PWM PERIOD

NATIONAL INSTRUMENTS,UNEXPECTED CURRENT READING ON THE NI9505, 23TH *FEBRUARY 2012* <HTTP://DIGITAL.NI.COM/PUBLIC.NSF/ALLKB/812D5CAD54BAF478862576C7007D6C83>

The problem occurs when the PWM signal is OFF, as the current goes through R2, and as can be seen on the below figure, the way of the current makes the voltage drop negative, making the ADC conversion get an incorrect value, as it is converting a negative value.

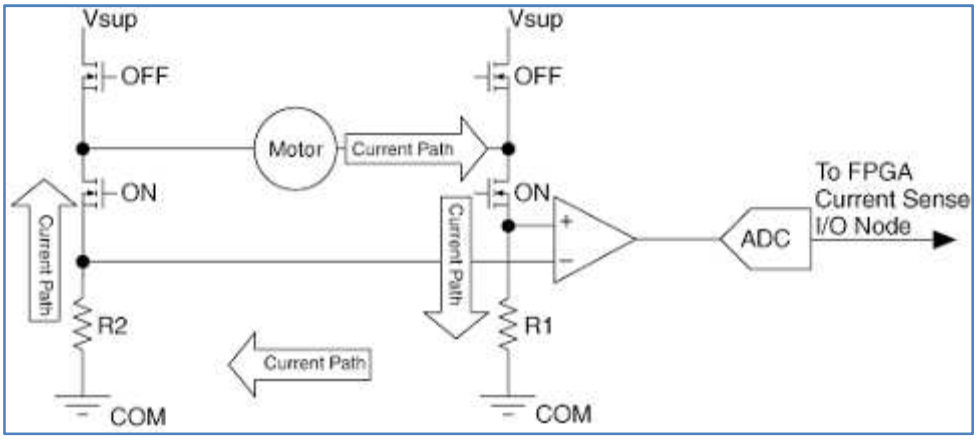


FIGURE 23. CURRENT SENSE DURING OFF PWM PERIOD

NATIONAL INSTRUMENTS,UNEXPECTED CURRENT READING ON THE NI9505, 23TH *FEBRUARY 2012* *<HTTP://DIGITAL.NI.COM/PUBLIC.NSF/ALLKB/812D5CAD54BAF478862576C7007D6C83>*

For this reason, the reading has to be done during the ON period, and in the middle of it, as will be explained on the next section.

PROBLEMS WITH CURRENT FEEDBACK

The problem with this way of reading the current , is that if the inductance of the motor is not high enough , the measure is not reliable . As the inductance of the motor is much smaller than the minimum inductance of 500 µH , the current readings contain some inaccuracy.

This happens because as has been stated in the DC motor model section, the inductance within the coil is the component which opposes current changes ($L \cdot \frac{di}{dt}$), so if this inductance is big enough, the slope that the current will present will be slow enough in order to make a valid current measurement during the sample point, but if for instance, like in this case, this inductance is smaller than the minimum specified by National Instruments, the effect of avoiding a big current modification done by the coil, won't be strong enough to have a valid measurement during the sample point, as it is stated on the figure below:

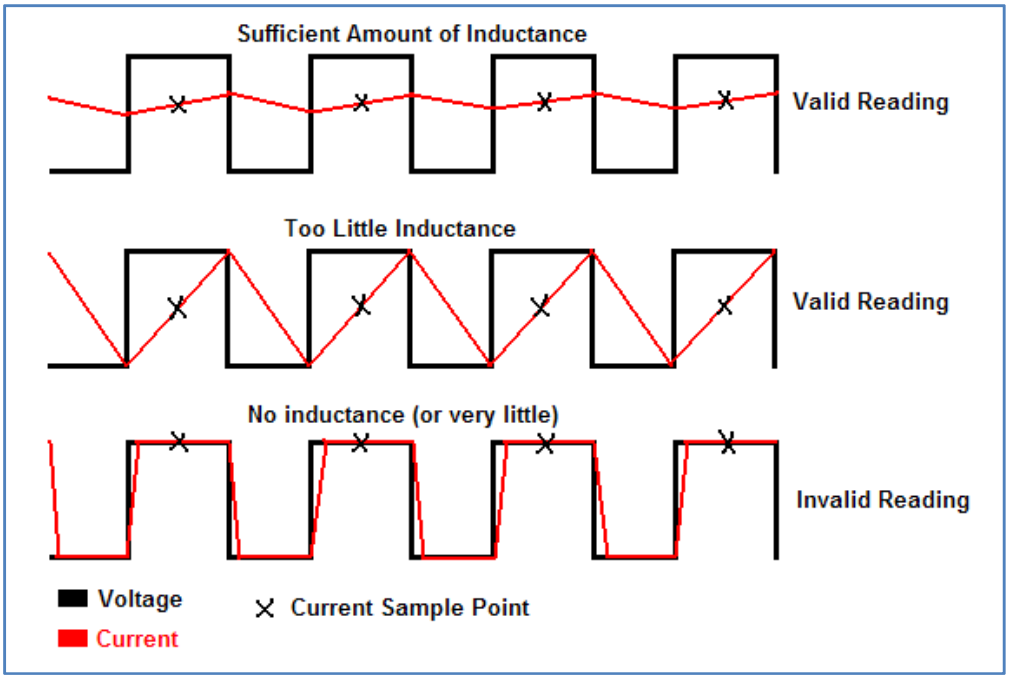


FIGURE 24. RELATION BETWEEN INDUCTANCE AND CORRECT CURRENT MEASURES

This impacts on reliable readings of the NI9505 current sensor for currents less than 200 mA , as has been observed experimentally.

POSSIBLE SOLUTIONS

The chosen solution is to implement another current sensor, because it has been seen that the one present on the NI9505 is not correctly sensing the current. In order to accomplish this goal, there are two options: one is to use a shunt resistor to sense the current coming from the battery, and the other one, is to use a Hall Effect sensor. The shunt resistor was chosen as that provided the simplest solution.

The implementation procedure for the shunt resistor is to use a resistor to measure the voltage drop on it coming from the power supply, and using Ohm's laws, translate the voltage to current. For the project a 0.01 Ohm shunt has been used, in order to keep power consumption of the sensor low, as the maximum consumption on the worst case is going to be:

$$P = I^2 \cdot R = 7.2A^2 \cdot 0.01\Omega = 0.5184W$$

The disadvantage of using a low value shunt resistor is logically, that falling voltages on the resistor are going to be very small, for instance:

Current(A)	Voltage(V)
0	0
1	0.01
5	0.05
7.2	0.072

TABLE 2.CURRENT-VOLTAGE EQUIVALENCE

So in order to have a good enough resolution, the voltage coming from the shunt needs to be amplified. For this purpose MCP6231 is going to be used, as it has a good enough bandwidth (300 KHz). A differential amplifier configuration is used for the AO:

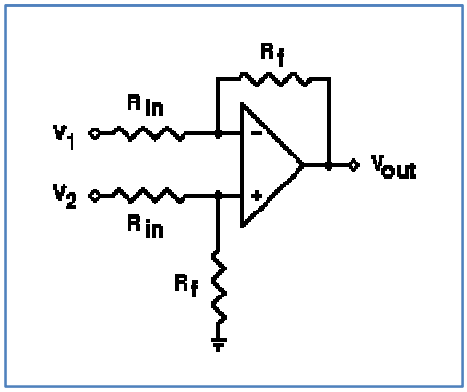


FIGURE 25.INSTRUMENTATION AMPLIFIER

In order to reduce the CMRR and noise interference and resistors of 0.1% tolerance are going to be used to mount the circuit, in order to reduce the possibility of making the system unstable. The output of the AO for this setup is the following:

$$Vout = \frac{Rf}{Rin}(V2 - V1)$$

The values resistor have been chosen to have a gain of approximately 25 (24.8175), with a resistor valued of Rf and Rin of 34kΩ and 1,37KΩ respectively.

With this gain, the voltage to be converted is going to be:

Current(A)	Voltage(V)
0	0
1	0.25
5	1.25
7.2	1.8

TABLE 3.CURRENT-VOLTAGE EQUIVALENCE

The ADC chosen to convert the analog voltage into a digital value is MCP3001, because it uses SPI protocol, the conversion time is more than enough for the purpose, its low power consumption, and finally, its low price. The resolution is of 10 bits, more than enough for the application.

This element works with modes 0,0 and 1,1 of the SPI protocol, with a maximum speed of 2.8MHz (with a power supply of 5V), as such a speed will not be needed, and to reduce power consumption , the working speed is going to be of approximately 0.2MHz, so there will be a reading frequency of 20Khz approximately, translated is going to be , one current measure per PWM period , the same as with the current sensor inside the NI9505.

The final diagram of the current sensor is the following:

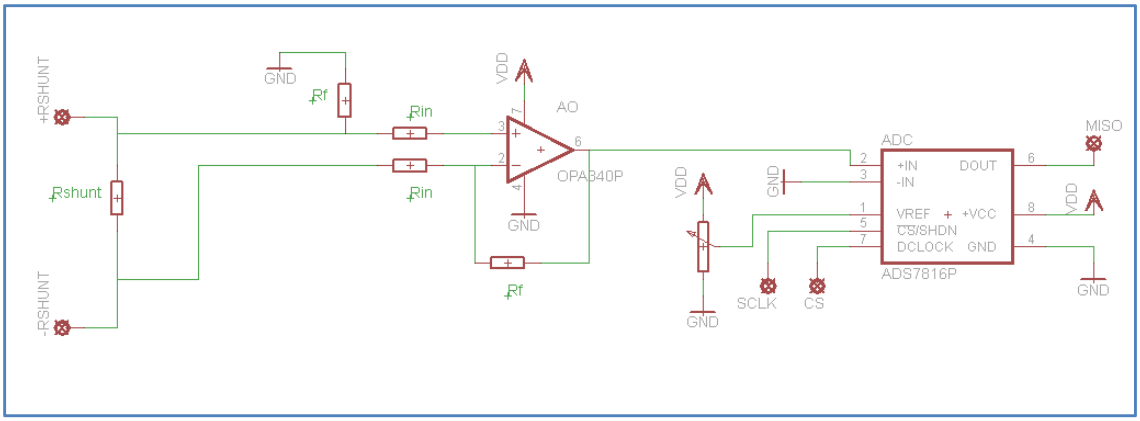


FIGURE 26.CURRENT SENSOR DIAGRAM

To use the sensor the SPI protocol for the sensor is needed thirteen falling rising edges of SCLK. The mode used is the 1,1, so the data is propagated on the falling edge. The two first bytes sent from the ADC are not used, and the following 10 are the data converted, starting with the most significant bit, as can be shown on the following figure:

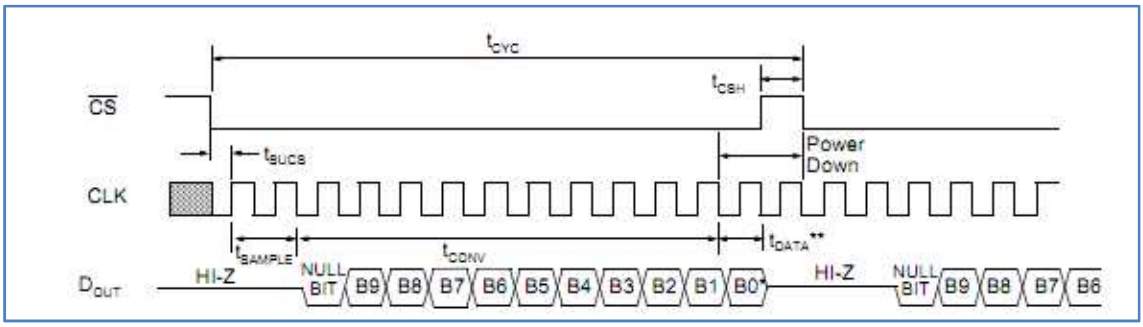


FIGURE 27.ADC SPI PROTOCOL

As outlined above, with a gain of approximately 25, the maximum current to be sensed is 1.8 volts. The potentiometer of the diagram exposes that Vref can be chosen, so if its value is 2 volts, the value conversion in going to be:

$$Digital\ Output\ Code = \frac{1024 \cdot Vin}{2V}$$

Translating into:

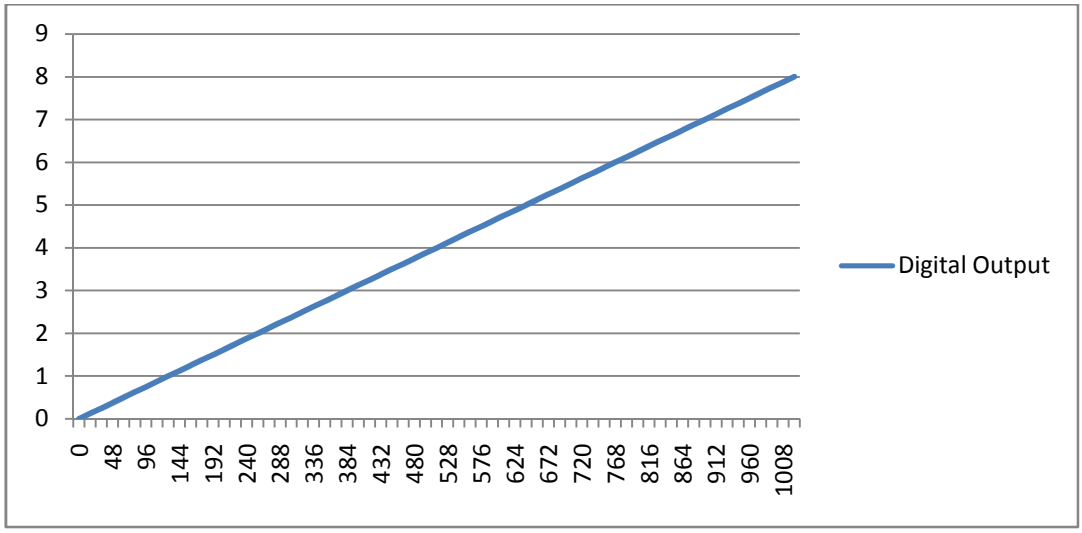


FIGURE 28.ADC-TRANSDUCTION FROM DIGITAL OUTPUT INTO CURRENT

So the resolution of the current sense is:

$$LSB = \frac{Vref}{2^{10}} = \frac{2}{1024} = 1.953125\,mV$$

This translates into a current of:

$$Current\ Resolution = \frac{LSB}{0{,}01 \cdot 24{,}8175} = 7.87\,mA$$

This current resolution is more than enough for the current sensing stage, as increments of about 0.1 Amps want to be detected.

Once implemented and in order to establish the accuracy of the system the following schematic is mounted:

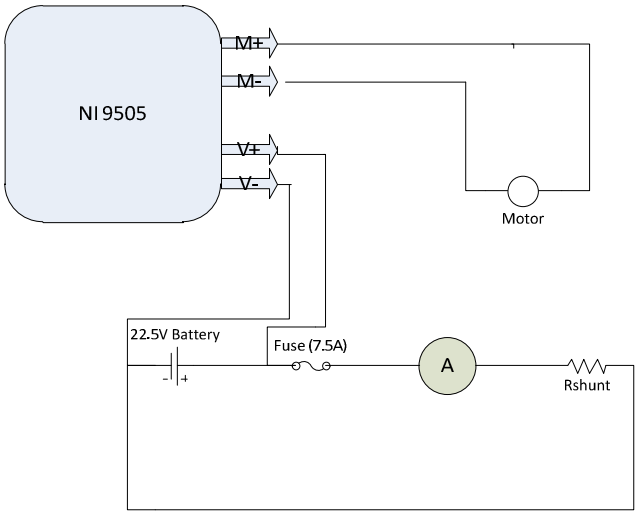


FIGURE 29.CONNECTION BETWEEN THE MOTOR, NI9505 AND CURRENT SENSOR

For very small currents (<0.2 A) the sensor current implemented inside the NI9505 is more accurate than the shunt sensor. This fact is due to small noise present in the input of the AO, which are almost impossible to eliminate, and are amplified 25 times, so they translate into small accuracy for small current.

To solve this problem, and have a reliable system, the NI9505 sensor is used when less than 0.15A currents are sensed by it, if the currents are greater, the current sensor from the shunt is used.

Another problem to be solved , is that the shunt current sensor cannot detect the direction of the motor , so the "polarity" of the current sensor inside the NI9505 is going to be used to indicate the direction , independently of which sensor is used. It is not possible to indicate the direction of the sensor because the NI9505 is a proprietary system with no direct access to the H-bridge given.

COMMUNICATION WITH THE MOTOR

The communication between the SbRIO and the motor is done as mentioned before with the NI9505. This component is mainly a power driver, with an H-Bridge inside, and already implemented isolation to protect the control part from the actuation part. The control has been done in the FPGA part, as will be explained on following sections, the main output which is a PWM signal going inside the H-bridge.

H-BRIDGE, HOW DOES IT WORK?

Firstly, its name comes from it shape, as can be seen on the below figure. Speed and direction are controlled with the currents flowing through the motor in the specified direction ruled by the position of the relays in the bridge. In this example, with switches "A" and "D" closed, the motor will operate in a clockwise direction. With "B" and "C" closed, the motor will operate in the counterclockwise direction.

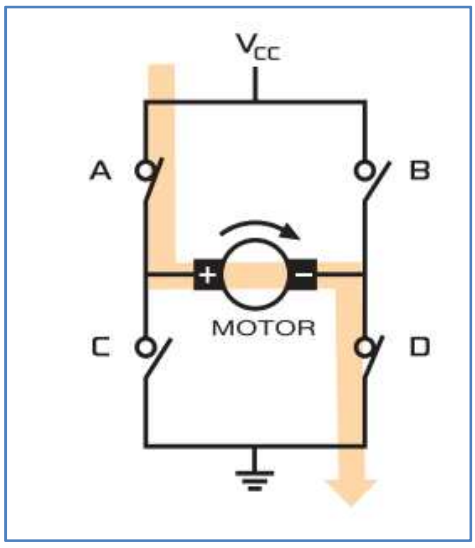


FIGURE 30.H-BRIDGE REPRESENTATION WITH CLOCKWISE DIRECTION

The output from the FPGA has two signals that need to be explained: Motor direction, and PWM (Motor). The first one, if it is true, determines that the direction must be clockwise, so “A” and “D” switches stay ON, and “B” and “C” remain OFF. If the value is false, the direction has to be counterclockwise, so “A” and “D” switches stay OFF, and “B” and “C” remain ON.

The PWM (Motor) signal, as it name says, corresponds to the PWM signal coming to H-Bridge. The more time that the signal stays at high level (High Duty-cycle), the more the speed of the motor, because the switches (“A” and “D” if CW, “B” and “C” if CCW) will stay more time on (the time the PWM signal remains at high level) making the motor to remain ON more time. So during the high level of the PWM signal the motor will be ON, and during the low level, the motor will be OFF. It has to be remembered that due to the inductance present in the motor, the motor will not be stopped when in the low level, because the current will still be flowing. The greater the period of the duty cycle that the motor is ON, the higher the speed.

2.1.5 ENCODER

2.1.5.1 DESCRIPTION

The encoder being used is the MR-1024 CPT from Maxon Motors. It is an incremental encoder, with 1024 counts per cycle, a maximum working frequency of 320 kHz, maximum speed of 18750 RPM and a differential output.

2.1.5.2 INCREMENTAL ENCODERS

The general definition of an encoder is an electromechanical device that can measure angular speed or position. The most common way to measure it is using optical sensors, because they have a longer life expectancy, as they do not suffer any kind of mechanical wear or fatigue. The method of operation is as follows: an electrical signal is generated which can be converted into motion direction and speed or position. An incremental encoder generates a pulse for each incremental step in its rotation.

In order to sense the position two channels are used, each one generates an identical square wave with a phase difference of 90 degrees, to indicate position and direction of the rotation. For example, if A leads B, the motor is rotating in a clockwise direction. In the opposite case, the motor is rotating in a counter-clockwise direction. Therefore, by monitoring both the number of pulses and the relative phase of signals A and B, both the position and direction of rotation can be tracked.

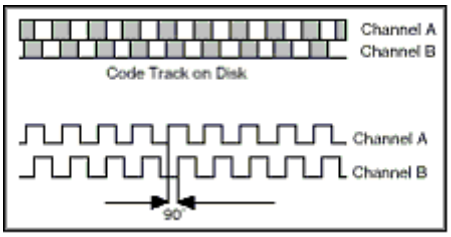


FIGURE 31.TWO-PHASE ENCODER OUTPUT SIGNALS

NATIONAL INSTRUMENTS, UNEXPECTED ENCODER MEASUREMENTS: HOW-TO GUIDE, 25TH MARCH *2012* <HTTP://ZONE.NI.COM/DEVZONE/CDA/TUT/P/ID/7109>

In some encoders there is a third output called index, which generates a single pulse per revolution.

2.1.5.3 OUTPUT

The output format, as commented before is differential. So each one of the outputs previously specified (A, B and index) have two complementary outputs, as shown on the following figure:

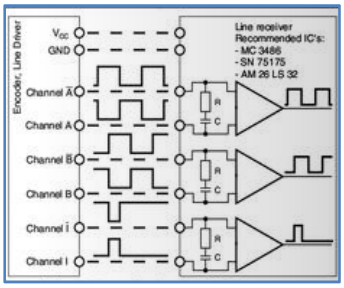


FIGURE 32.MR-1024 CPT ENCODER OUTPUT SIGNAL

This is implemented to avoid signal loss due to long distance and the capacitive effect of the long cables that can cause degradation of the signal and the possibility of the SbRIO being unable to read the signal. In the current case, it is not strictly necessary, but providing this differential output adds robustness to the system.

2.1.6 GPS

2.1.6.1 OBJECTIVE

The objective of the GPS is to know the exact position of the kit, in order to be able to modify its position to the one with optimum wind speeds.

2.1.6.2 DESCRIPTION

The GPS chosen for this project is GPS-610F, from RF solutions. The main characteristics of this GPS are a good signal reception, with an external antenna (it already has the MCMX connector for this purpose) and the possibility to add a DGPS input for more accuracy. The advantages of DGPS will be discussed later on.

Another important characteristic is the low power consumption: ~23mA at 3.3V, the small time before start sending current position: Open sky hot start 1 sec, cold start 29 sec, and finally an standard connector can be used to interface it with the PCB without the needing of using non-standard (expensive, hard to get) connectors, as it happens with multiple GPS modules.

The update rate is of 1Hz and the transmission done through serial interface with configurable speed (4800/9600/38400/115200bps, Default: 9600).

The data sentence is the standard: NMEA 183.

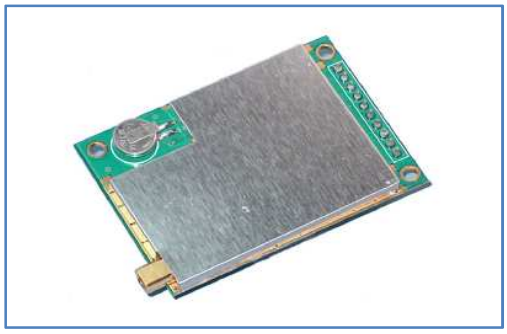


FIGURE 33.GPS 610-F PICTURE

ANTENNA

The GPS bought needs an external antenna in order to receive the satellite signals. For this purpose the 2J402U antenna has been used.

FIGURE 34.2J402U EXTERNAL ANTENNA

2J ANTENNA CONCEPTOR, 2J402U DATASHEET, 25TH MARCH *2012*
<HTTP://WWW.2J-ANTENNAE.COM/IMAGES/PRODUCTS/2J402U.PDF>

The reasons to use this, is because it does not have a big power consumption(around 100mA), it can stand temperatures from -40 up to 85 degrees, has an MMCX connector(the one used by the GPS) and finally, because it has a good gain, reaching the 27dB,as stated below:

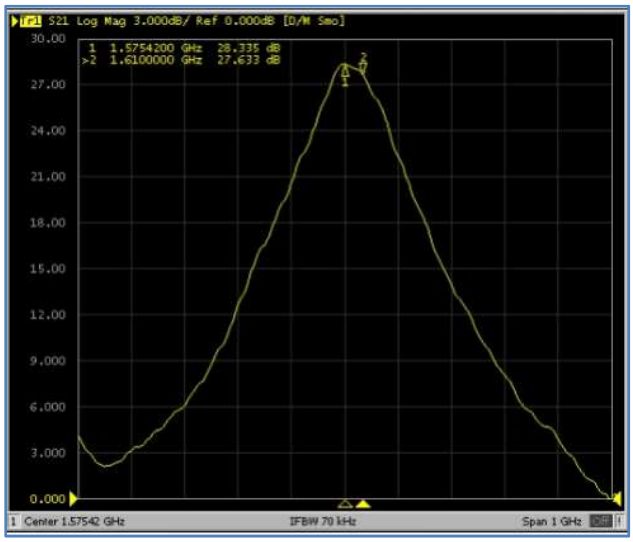


FIGURE 35.ANTENNA'S GAIN AT 5V

2J ANTENNA CONCEPTOR, 2J402U DATASHEET, 25TH MARCH *2012*
<HTTP://WWW.2J-ANTENNAE.COM/IMAGES/PRODUCTS/2J402U.PDF>

NEMA SENTENCE

The data specification that will be used is the NMEA 0183. It consists of sentences, the first one defines the type of the rest of the sentence, which is defined by the NMEA standard. The ones that will be used are discussed below:

- GGA: Provides 3D location and accuracy data
- GSV: Indicates the number of findable satellites for the GPS, and ability to track its data.
- RMC: Indicates position, velocity and time

SERIAL PROTOCOL

The serial protocol is an asynchronous serial communication protocol standard. It is commonly used to communicate electronic components with the PC, but also from sensors to MCU, and to/from a long list of different elements.

It uses two inputs, one is for entering data, and the other is the output data. The communication is done by sending frames. These are complete and non-divisible packets of bits, including information as the overhead (start bit, stop bit and CRC) and the data wanted to be communicated. In serial communication the frame consist in one start bits, 8 bits of data (most of the cases), a parity bit and a stop bit, as shown on the figure below:

FIGURE 36.SERIAL COMMUNICATION EXAMPLE

The start bit indicates the start of a new frame, the parity bit is used for checking transmission errors, but is it an optional feature so it doesn't have to be implemented for all types of communications.

Bit time is the basic unit time between each bit, so the transmitter sends out one bit, waits the bit time, then sends the next one.

So in order that a communication can be established between the transmitter and the receiver, both must know if there is a start or/and stop bit, if is there a parity bit and if so, which type of parity is there, and finally the transmission speed or time between bits.

2.1.6.4 DGPS

The DGPS (Differential Global Positioning System) is a technology complementary to GPS that increments its precision. It is commonly used by the nautical sector.

As it is well known, GPS system has an average error of 15 meters. As a possible expansion of the project, the possibility of adding a DGPS module to the current GPS module is possible, that will add a lot of precision of the project (about 1 meter average error or less).

DGPS operates by calculating the receiver's inaccuracy by gauging it to a well-known receiver station , the station sends radio signal to all DGPS equipment with the correction , so it tells actually the real(or at least more accurate) position of the equipment being used. Actual Irish stations can be found on the following link: http://www.commissionersofirishlights.com/cil/aids-to-navigation/marine-dgps.aspx that refers to Northern Lighthouse Board whose principal concern is the safety of the mariners at sea.

The main inconvenience of this technology is the price.

2.1.6.5 SCHEMATIC

The schematic of the GPS board is formed by a converter that supplies 3.3v to the GPS, a DB9 and a MAX3232. The MAX3232, works as a level transceiver in order to translate the logic levels coming from the serial port of the SbRIO (see above figure), to the 3.3v present on the GPS. If this device is not present, the GPS would be damaged, as the voltages of +-12 volts are above the maximum voltage level of the positioning sensor, rated at 3.6V. Comentar pinout GPS

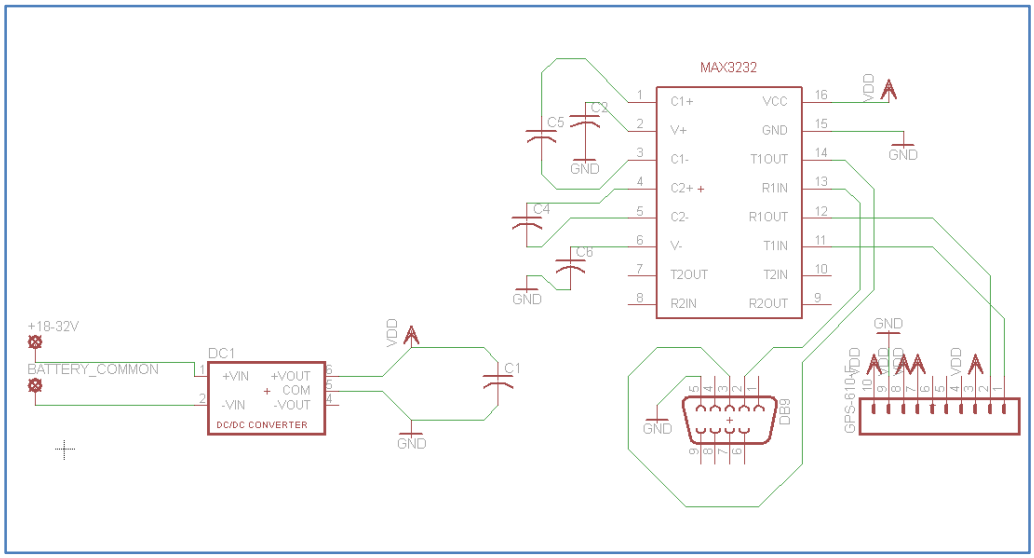


FIGURE 37.GPS BOARD SCHEMATIC

2.1.6.6 PCB

The PCB design developed using EAGLE CADSOFT software is stated below:

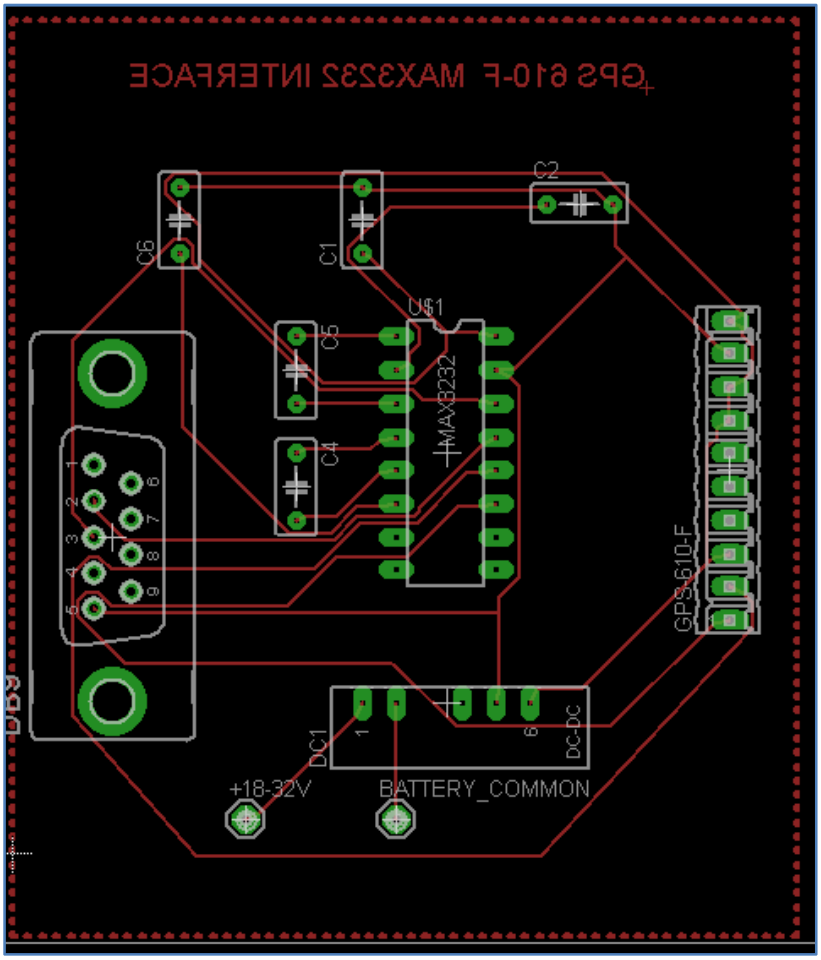


FIGURE 38.GPS BOARD LAYOUT

2.1.7 ISOLATION

In this chapter the isolation from the noisy elements will be discussed.

2.1.7.1 NOISY ELEMENTS

The noisiest element in the project is the motor. The fact that it consumes a lot of power compared with the other elements, and the presence of current spikes, can be translated into noise added to the system.

The current spikes would appear when the motor starts to operate from stopped position, as there is no momentum; the amount of power to start moving is greater. But the most important spike is when the motor is already in movement and the motor has to change the direction of rotation, going from clockwise to counter clockwise, or the opposite.

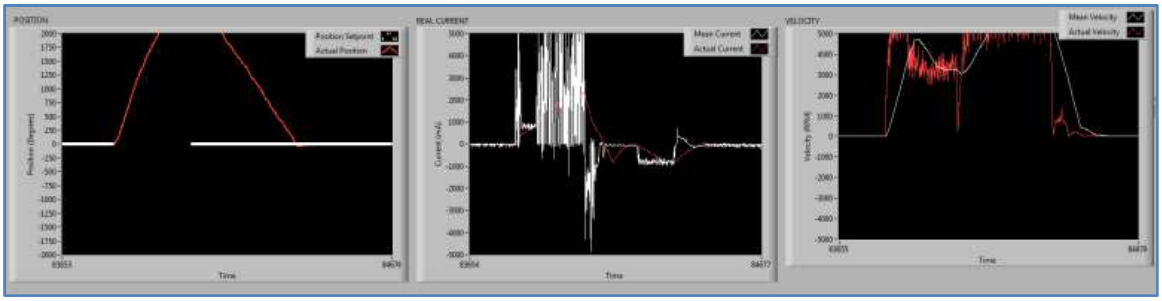


FIGURE 39.POSITION, CURRENT CONSUMPTION AND SPEED WHEN INVERTING MOTOR DIRECTION

Due to these spikes it is necessary to isolate the control electronics from the actuator electronics, two different power supplies are implemented (this fact will be discussed more in depth on the Power Supply chapter), one for the SbRIO and the sensors (IMU,GPS,encoder) and for the router, and the other one only for the actuator part of the system: the motors.

As has been previously discussed, the NI9505 is being used to drive the motors, so this is the driver which needs to be isolated. As it is already isolated (the power part from the control part) , as can be seen on the following figure taken from the NI9505 datasheet , no further isolation would be needed to isolate the motors from the SbRIO .

Safety

Safety Voltages

Connect only voltages that are within the following limits.

Channel-to-COM	0 to +30 VDC max, Measurement Category I
Isolation	
Channel-to-channel	None
Channel-to-earth ground	
Continuous	60 VDC, Measurement Category I
Withstand	750 V_{rms}, verified by a 5 s dielectric withstand test

FIGURE 40. ISOLATION OF THE NI9505

2.1.8 POWER SUPPLY

Following the previous section, it has already been mentioned that there would be two different batteries: one for the actuation part (motors) and the other one for the control and sensing part, to avoid the introduction of noise by the motors.

2.1.8.1 POWER CONSUMPTION OF THE ELEMENTS

In this section, the power consumption of all the elements of the system will be discussed, on the worst cases.

SBRIO POWER CONSUMPTION

Firstly the core of the system, the SbRIO will be analyzed. The consumption of it can be specified by using the formula from the user's guide:

$$Total\ Power\ Consumption = Pint + Pdio + Pcser$$

Where:

Pint is the consumption by SbRIO internal operation

PDIO is the consumption by the 3.3 V DIO

PCSer is the consumption by installed board-only C Series modules

In this case the worst case of power consumption would be:

$$Pint = 6W$$

In SbRIO there are 110 Inputs/Outputs available, with a voltage of 3.3V and a maximum current per channel of 3mA, so:

$$Pdio = 110 \cdot 3mA \cdot 3.3V = 1.089W$$

For the expansion slots, the basic consumption without any expansion module installed is 3.3W, plus 1.1W for any each extra module (1.5W for the NI9505):

$$Pcser = 3.3W + 2 \cdot 1.5W = 6.3W$$

So the total is:

$$Ptotal = 6W + 1.089W + 6.3W = 13.389W$$

Adding 20% for transients conditions:

$$Psbrio = Ptotal \cdot 1.2 = \mathbf{16.0668W}$$

TOTAL CONTROL PART POWER CONSUMPTION

Regarding the datasheet from the different components there is the following power consumption:

Product	Voltage(V)	Nominal Current(A)	Maximum Current(A)	Nominal Power(W)	Maximum Power(W)
DIR-655	12	0.55	2	6.6	24
ADIS 16407	5	0.07	0.084	0.35	0.42
GPS	3.3	0.036	0.07	0.1188	0.231
MR-1024	5	-	0.005	-	0.025

TABLE 4.SENSORS AND COMMUNICATON PART POWER CONSUMPTION

The transceiver used for the kite's IMU also have to be considered. The following power consumptions are extracted from SN65HVD3082E's datasheet:

$$Receiver\ and\ driver\ enabled\ supply\ current\ =\ 900uA$$

$$Receiver\ enabled\ and\ driver\ disabled\ supply\ current\ =600uA$$

$$Receiver\ disabled\ and\ driver\ enabled\ supply\ current\ =600uA$$

$$Driver\ Maximum\ output\ current\ =\ 60mA$$

$$Receiver\ maximum\ output\ current\ =8mA$$

The configuration of the transceivers is the following:

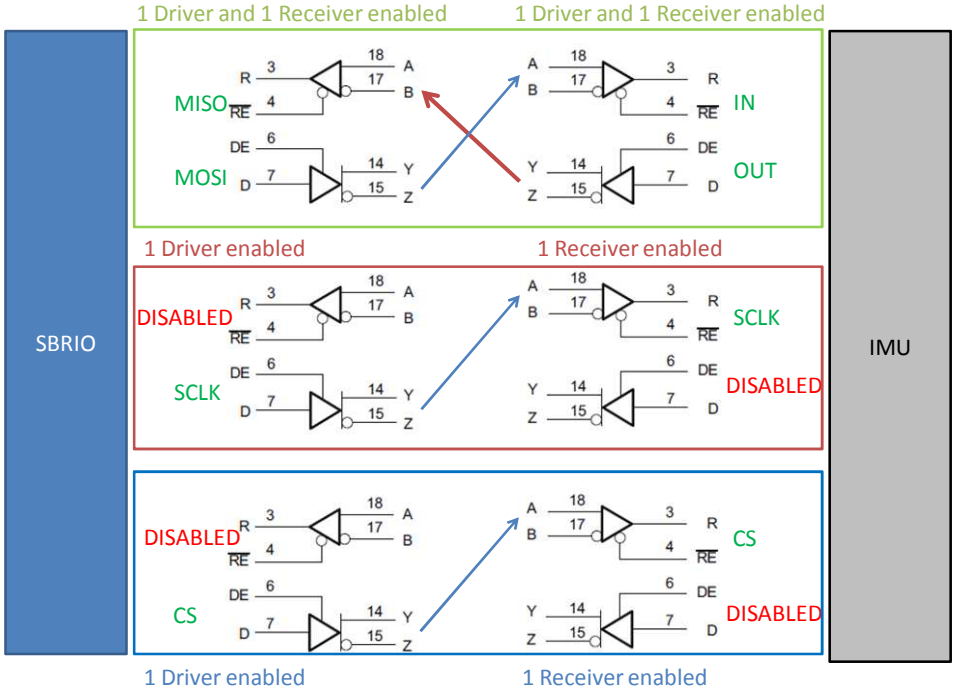


FIGURE 41.TRANSCEIVER CONFIGURATION

Making the calculations:

$$Receiver\ and\ driver\ enabled\ supply\ current\ x\ 2 = 1800uA$$

$$Receiver\ enabled\ and\ driver\ disabled\ supply\ current\ x\ 2\ = 1200uA$$

$$Receiver\ disabled\ and\ driver\ enabled\ supply\ current\ x\ 2\ = 1200uA$$

$$Driver\ Maximum\ output\ current\ x\ 4 =\ 240\ mA$$

$$Receiver\ maximum\ output\ current\ x2\ = 16mA$$

$$Maximum\ Total\ current\ consumption\ of\ the\ Transceivers = 1800uA + 1200uA + 1200uA + 240mA + 16mA = 0.2602A$$

The two current sensors used to add extra feedback to the NI9505 position control power consumption are analyzed:

In the worst case, as has been discussed on current sensing section, the shunt will have a consumption of 0.52W. If to this value the consumption of the AO and the ADC are added:

$$Pcurrent = Pshunt + Pao + Padc$$

$$Pcurrent = 0.52W + (25\mu A \cdot 5V) + (250\mu A \cdot 5V) = 0.521375\ W$$

As there are two different actuator/sensing parts, the total value is:

$$PtotalCurrent = 0.521375 \cdot 2 = \mathbf{1.04275\ W}$$

As the battery for the control part is of 22.5V, there is a need for converters/regulators, as is shown on the following figure:

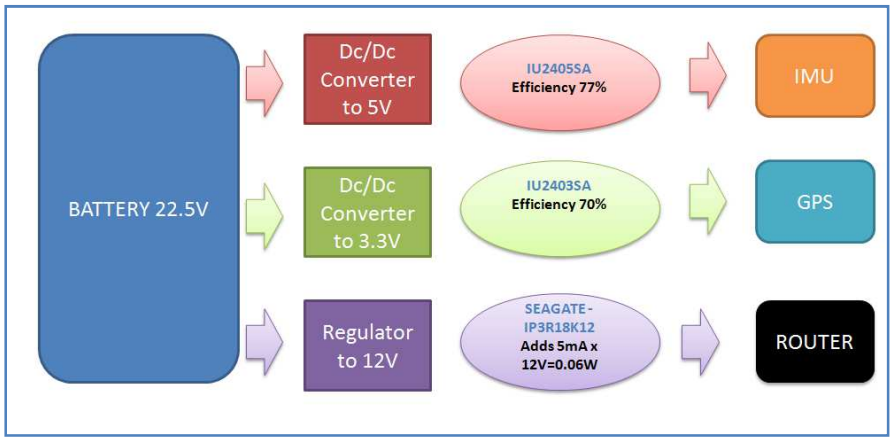


FIGURE 42.SENSING,CONTROL AND COMMUNICATION POWER CONVERSION EFFICIENCY AND CONSUMPTION

So in the worst case, the amount of power consumption is:

Converter	Max Power(W)	Max Current(A)	Efficiency %	Real Max Power(W)
5V(IMUx2,transceivers,Encoder,current)	3.7057	0.52	77	4.558011
3.3V (GPS)	0.231	0.07	70	0.3003
12V(WIFI)	24	2	50	48
TOTAL				**52.858**

TABLE 5.TOTAL POWER CONSUMPTION (WITHOUT ACTUATION PART) USING REGULATORS

The total amount of power dissipation on maximum conditions is:

$$51.0235W + 16.0668W = \mathbf{68.924811W}$$

If instead of a regulator, a DC/DC converter (JCL3024S12) is used for supplying energy to the router, it will be seen as power consumption is heavily decreased, as for a conversion of 22.5 to 12V the converter has an efficiency of 91, so:

Converter	Max Power(W)	Max Current(A)	Efficiency %	Real Max Power(W)
5V(IMUx2,transceivers,Encoder,current)	3.7057	0.52	77	4.558011
3.3V (GPS)	0.231	0.07	70	0.3003
12V(WIFI)	24	2	92	25.92
TOTAL				**30.778311**

TABLE 6. TOTAL POWER CONSUMPTION (WITHOUT ACTUATION PART) USING DC/DC CONVERTERS

By using converters instead of regulators, a battery endurance of more than 50% is gained.

MOTOR POWER CONSUMPTION

It is difficult to estimate the power consumption of the motor, but a test scenario close to reality is established. If it is assumed that there is a switch of position every 30 seconds, with a maximum PWM duty cycle of 50%, an equal load to 2kg and the distance of every switch is equal to 2000 degrees. But the fact is that the concerning is not based on the mass of the load, but on the torque needed for the motor. So for a 2kg load (the radius of the reel is 0.02m):

$$T = F \cdot r \rightarrow F = m \cdot g \rightarrow T = m \cdot g \cdot r = 2kg \cdot \frac{9.81m}{s^2} \cdot 0.02m = 0.3924Nm$$

Taking in consideration that the gear-head reduces the motor torque by 15(while increasing the speed), resulting in a torque on the motor spindle for a 2kg load:

$$T'(2kg) = \frac{T}{15} = 0.02616Nm$$

A 4kg load would result in:

$$T'(4kg) = T'(2kg) \cdot 2 = 0.01308Nm$$

The speeds shown on the graphs also get affected by the gear-head resulting on the following equivalences (the ones shown in the graphs correspond to gearhead's input):

Gearhead's Speed Input(RPM)	Gearhead's Speed Output (RPM)
2000	133.3
4000	266.6
5000	333.3

TABLE 7.EQUIVALENCE BETWEEN GEAR-HEAD'S INPUT AND OUTPUT SPEED

The following power consumption every half minute is present:

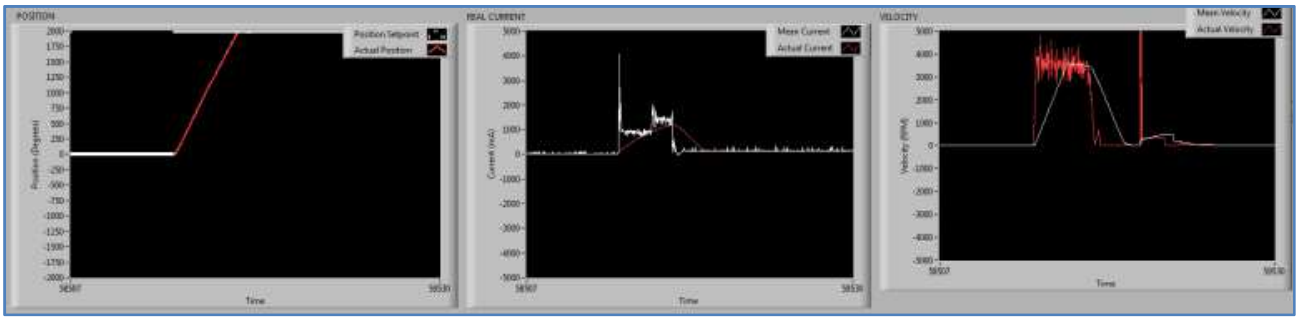


FIGURE 43.MAXIMUM DUTY CYCLE 50%, 0.02616NM LOAD

The time is 3.5 seconds approximately, with an average current of 1000mA on the first 2 seconds, and 1650 mA on the next 1.5 seconds and 167mA in steady state (holding the load), so in this case the consumption of the switching is:

$$Pswitching = 120 \cdot 0.00055h \cdot (1000mA \cdot 22.5V) +$$

$$+120 \cdot 0.000416666667h(1650mA \cdot 22.5V) + 0.883333333 \cdot (167mA \cdot 22.5V) = 6.675$$

This leads to an average current of:

$$Iavge = \frac{P}{V} = \frac{6.675W}{22.5V} = 0.296A$$

If now the scenario is the following: maximum speed configuration of 4000 RPM, a torque 0,01308Nm and the distance of every switch is equal to 2000 degrees, there is the following power consumption every half minute:

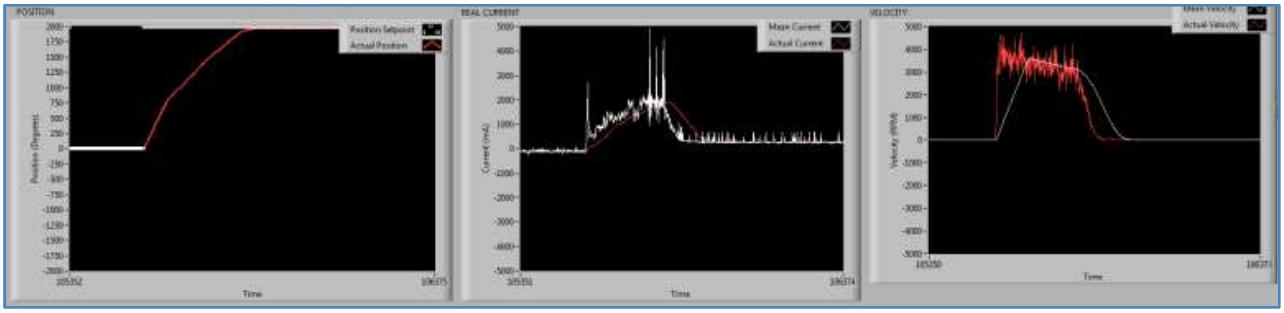


FIGURE 44. MAXIMUM DUTY CYCLE: 50%, 0.01308NM TORQUE

In order to calculate the Power consumption per hour, the following approximation is considered:

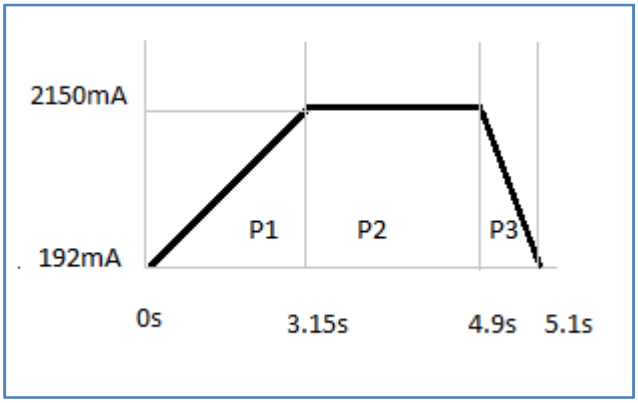


FIGURE 45.POWER CONSUMPTION APPROXIMATION

The power consumption from the first stage comes determined by:

$$P1 = 120 \cdot 22.5V \cdot \frac{(2150mA - 192mA) \cdot 0.000875h}{2} = 2.34241W \cdot h$$

The second stage comes determined by:

$$P2 = 120 \cdot 0.0004861 \cdot 2150mA \cdot 22.5V = 2.8218W \cdot h$$

And the last one:

$$P3 = 120 \cdot 22.5V \cdot \frac{(2150mA - 192mA) \cdot \ 5.55555556 \ \cdot 10^{-5}h}{2} = 0.1487W \cdot h$$

Finally, on steady state there will be a consumption of:

$$Psteady = 22.5V \cdot 192mA \cdot 0.83h = 3.5856W \cdot h$$

So the total consumption per hour is:

$$Ptot = Psteady + P1 + P2 + P3 = 8.8943W \cdot h$$

This leads to an average current of:

$$Iavge = \frac{P}{V} = \frac{8.8943W}{22.5V} = \ 0.3953A$$

The worst case, but possibly the most realistic scenario is to consider that the motor is switching positions without having reached the previous position setpoint. This causes inversions of direction, that need larger current spikes to reach the same speed. Note that the frequency of position switching can be controlled by software, by making the position control iteration slower. By doing this position accuracy is lost, but gained battery endurance. The new scenario is defined by a 0.01308Nm torque getting the following plot:

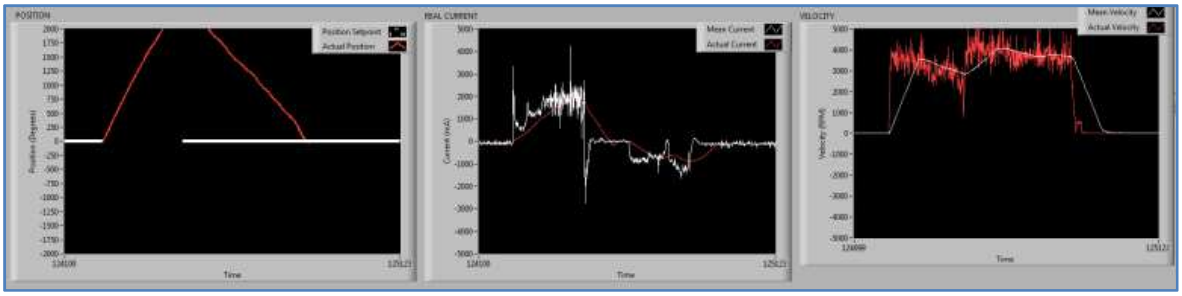


FIGURE 46.MAXIMUM DUTY CYCLE: 50%, 0.01308NM TORQUE

That can be approximated into the following figure:

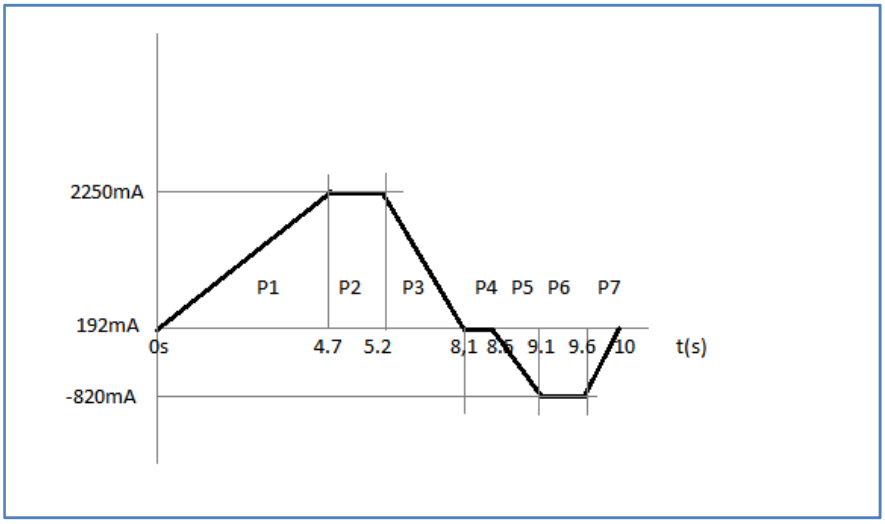


FIGURE 47.POWER CONSUMPTION APPROXIMATION

The power consumption from the first stage comes determined by:

$$P1 = 22.5V \cdot \frac{(2250mA - 192mA) \cdot 0.0013h}{2} = 0.03W \cdot h$$

The second stage comes determined by:

$$P2 = 0{,}000138\text{h} \cdot 2250\text{mA} \cdot 22.5\text{V} = 6.98\text{mW} \cdot \text{h}$$

And the last one:

$$P3 = 22.5V \cdot \frac{(2250mA - 192mA) \cdot \ \ 0.000805h}{2} = 0.018W \cdot h$$

The fourth stage comes determined by:

$$P4 = \ 0.00011h \cdot 192mA \cdot 22.5V = 475.2\mu W \cdot h$$

The next stage:

$$P5 = 22.5V \cdot \frac{(2250mA - 192mA) \cdot \ \ 0.00016h}{2} = 3.7mW \cdot h$$

The one before the last:

$$P6 = 0.000138h \cdot 820mA \cdot 22.5V = 2.54mW \cdot h$$

And the last:

$$P7 = 22.5V \cdot \frac{(2250mA - 192mA) \cdot \ \ 0.00011h}{2} = 0.83\mu W \cdot h$$

The total time per switching period is 10 seconds, so in one hour:

$$Number\ of\ Iterations = \frac{3600}{10} = 360\ iterations$$

So the total consumption per hour is:

$$Ptot = 360 \cdot (P1 + P2 + P3 + P4 + P5 + P6 + P7) = 22.21W \cdot h$$

This leads to an average current of:

$$Iavge = \frac{P}{V} = \frac{22.21W}{22.5V} = 0.987A$$

2.1.8.2 LITHIUM POLYMER

The lithium-polymer uses a dry solid polymer electrolyte. This electrolyte resembles a plastic-like film that does not conduct electricity but allows ions exchange (electrically charged atoms or groups of atoms).

But why has LiPo been chosen over Li-Ion?

Following there is a brief description of the three types of batteries dominating the portable electronics market:

	NiMH	Li-Ion	Li-polymer
Nominal Cell Voltage	1.25V	3.6V	3.7V
Cycle durability	500-1000 cycles	1100 cycles	<1000 cycles
Energy/weight	100Wh/kg	160Wh/kg	100Wh/kg
Energy/size	140-300 Wh/l	140-300 Wh/l	150 Wh/l
Power/weight	250-1000 W/Kg	1800 W/Kg	2800 W/Kg
Charge/discharge efficiency	66 per cent	99 per cent	99 per cent
Energy/consumer-price	1.40 Wh/US$	4 Wh/US$	4 Wh/US$
Self-discharge rate	30%/month	7.5%/month	5%/month
Optimal load current	<0.5C	<1C	<1C
Maximum aggregate voltage	12.5(10 cells)	25.2(7 cells)	25.2(7 cells)
Charge time	<4 hours	<4 hours	<4 hours

TABLE 8.BATTERY COMPARISON

LOW POWER DESIGN, NIMH, LI-ION OR LI-POLYMER? WHAT TO CONSIDER, 22ND FEBRUARY 2012, <HTTP://JARTIUCH.WORDPRESS.COM/2007/11/01/NIMH-OR-LI-ION-OR-LI-POLYMER-WHAT-TO-CONSIDER%E2%80%A6/>

The main characteristic which was being considered for the application is the Wh/kg at a given voltage as this is the most critical feature in an airborne project requiring maximum power to weight ratios. As can be seen, Li Polymer offers a significant weight advantage over alternative cell chemistry options. LiPo is already well established as a robotics power supply technology, making its prices low.

Another point was that there are nominal batteries of 22.5V needed for the motor, and its durability and instantaneous discharge rate was more than enough to feed the actuation and the control stage.

2.1.8.3 BATTERIES DESCRIPTION

The batteries chosen are the Flightpower's EVO-Pro Lite 22.5V 5200mAh with six cells. The main reason is its relation between weight and power. It has a discharge rate of 25C that

allows a maximum continuous current of 130A, and 3 seconds bursts of 260A. The weight is 805 grams.

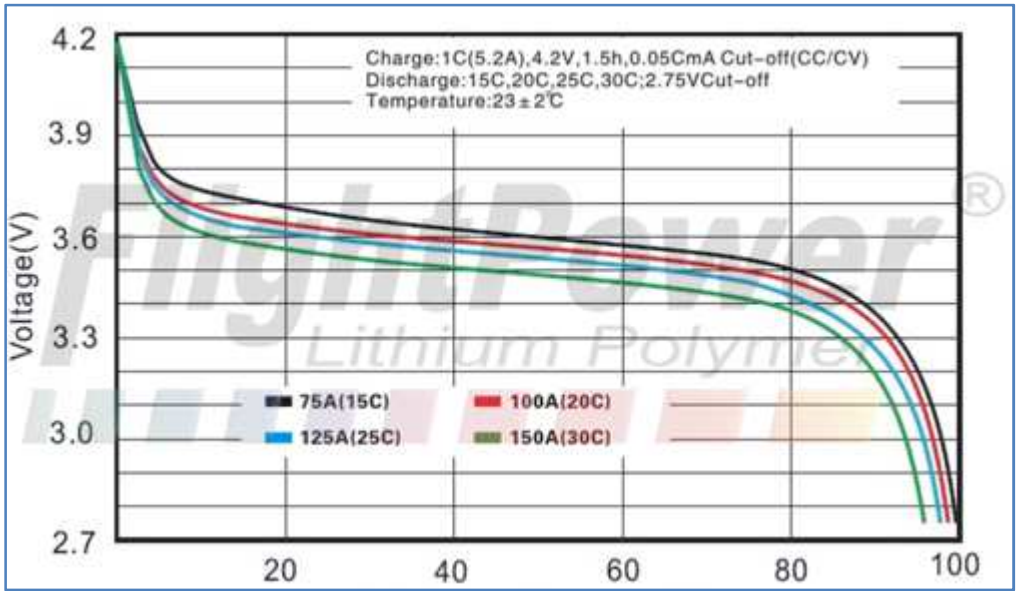


FIGURE 48.BATTERY DISCHARGE CAPACITY (%)

FLIGHT POWER, BATTERY DATASHEET, 25TH MARCH *2012,*
<HTTP://WWW.FLIGHTPOWER.CO.UK/INDEX.PHP?MAIN_PAGE=PRODUCT_INFO&CPATH=6&PRODUCTS_ID=153>

It has to be noted that the provider suggests the user does not allow the battery to discharge below 3V per cell (rchelisite.com) [online], as it can prevent the battery to make any more charging cycle. For this reason, and also to avoid losing the control of the pod caused by a motor failure due to lack of energy, a management of the battery capacity has been implemented, and it will be explained on the battery management section.

2.1.8.4 LIFETIME OF THE BATTERIES

The discharge time can be defined as:

$$discharge\ time = \frac{battery\ capacity}{current\ drained}$$

As it has been defined before:

Product	Voltage(V)	Nominal Current(A)	Maximum Current(A)	Nominal Power(W)	Maximum Power(W)
DIR-655	12	0.55	2	6.6	24
ADIS 16407	5	0.07	0.084	0.35	0.42
GPS	3.3	0.036	0.07	0.1188	0.231
MR-1024	5	-	0.005	-	0.025

TABLE 9.SENSING AND COMMUNICATION PARTS POWER CONSUMPTION

Converter	Max Power(W)	Max Current(A)	Nominal Current(A)	Efficiency %	Real Max Power(W)
5V(IMUx2,transceivers,Encoder,current)	3.7057	0.52	0.4552	77	4.558011
3.3V (GPS)	0.231	0.07	0.036	70	0.3003
12V(WIFI)	24	2	0.55	92	25.92
TOTAL		**2.59**	**1.0412**		**30.778**

TABLE 10.POWER CONSUMPTION WITH CONVERTERS EFFICIENCY

So for the maximum current, for a capacity of 5200mAh:

$$\frac{5200mAh}{\mathbf{2.59}A} = 2\ hours$$

For the typical current and a capacity of 5200mAh:

$$\frac{5200mAh}{1.0412A} = 4.99\ hours$$

For the maximum current, for a capacity of 4200mAh:

$$\frac{4200mAh}{\mathbf{2.59}A} = 1.62\ hours$$

For the typical current and a capacity of 4200mAh:

$$\frac{4200mAh}{1.0412A} = 4.03\ hours$$

For the maximum current, for a capacity of 3300mAh:

$$\frac{3300mAh}{\mathbf{2.59}A} = 1.27\ hours$$

For the typical current and a capacity of 3300mAh:

$$\frac{3300mAh}{1.0412A} = 3.16\ hours$$

The estimation of power consumption must be a current between the nominal and the maximum.

Now, the motor consumption is analyzed (at a maximum speed of 4000RPM):

Scenario	Current Consumption(A)	3300maH Lifetime(h)	4200maH Lifetime(h)	5200maH Lifetime(h)
1	0.296·2	5.57	7.09	**8.78**
2	0.395·2	4.11	5.31	6.58
3	0.987·2	**1.67**	2.12	2.63

TABLE 11.MOTOR POWER CONSUMPTION

Even on the most pessimistic scenario, the power consumption of the motors is low enough to allow almost two hours of operation. As the goal on the prototyping phase is being able to fly for less than half an hour, for testing purposes, with the smallest battery should be more than enough to achieve it.

2.1.8.5 MANAGEMENT OF THE BATTERIES

An important point of the lithium polymer batteries is the level of charge. It is important for the following reasons: Firstly it is critical that the SbRIO or the motors don't stop working during flight operations, and secondly, is that this kind of battery must never go below 3V per cell, because they will not be able to be charged again if this happens.

The most straight forward way to measure the charge of the battery is to measure the voltage between the positive and the negative of the battery. The problem of this solution is that the nominal voltage is 24V, so there is a need of a voltage reference of at least 24V, and this is only possible when the battery it is fully charged, and is voltage that wants to be measured, so it is not a good solution.

As the last option is not possible, the implemented solution consists on measuring the voltage of only one cell. The voltages are exactly the same on each cell, so the only extra calculation that needs to be done is multiply the voltage by 6, the number of cells, to get the total voltage.

As can be stated this method is only an approximation of the total voltage, as the cells can be unevenly charged or discharged, but this should not happen if the battery is working properly, so in order to detect any failures on the battery, an analog converter should be implemented to each cell to detect if the voltages of the cell are the same (at least approximately). We are assuming the voltage in each cell is equal, this may not be the case; they are equal when charged full but may discharge slightly unevenly.

Following, there is the schematichs of batteries level manager:

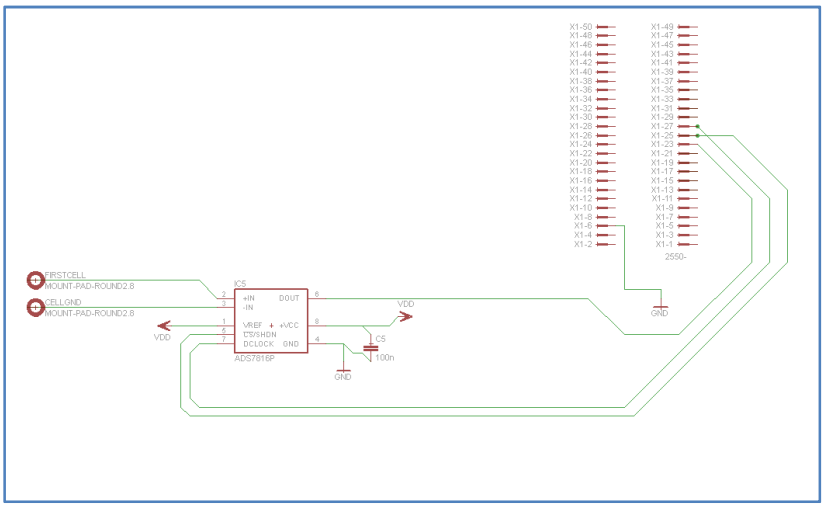


FIGURE 49. BATTERY MANAGEMENT SCHEMATICS

2.1.9 WIFI

The WIFI network is used to communicate the ground station with the Air Pod. This communication is necessary in order that the two parts of the control system know the current state of each other, so they can act in cooperation.

2.1.9.1 WIFI CONFIGURATION

The layout of the elements of the Network are shown on the following figure:

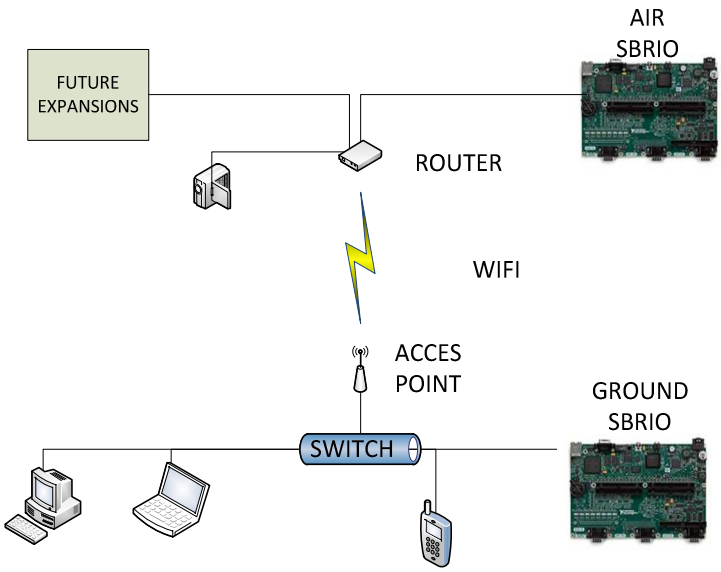


FIGURE 50.WIFI CONFIGURATION

The network can only have one master or router in this case, meaning that it is the center of the network, and the place where all the other components will be connected to. So in order to establish a network, an access point is needed, that will connect to the router. It is situated on the ground. The reason to situate the access point, lighter and with less energy consumption than the router in the ground, is for the possible expansion options regarding the router, which would be explained later on this section.

On the ground station there will be also a switch interconnecting the access point with the ground SbRIO and ground computers. These are the ones where all information of the system would be available, and where is possible to control both stations: the air and the ground ones. Another relevant future feature is the possibility to control a LabVIEW based system using a mobile phone or any flash enabled device with Windows operating system or using NI's web based client. In order that they can communicate with each other, the IPs have to be in the same subnet, as shown in the above diagram:

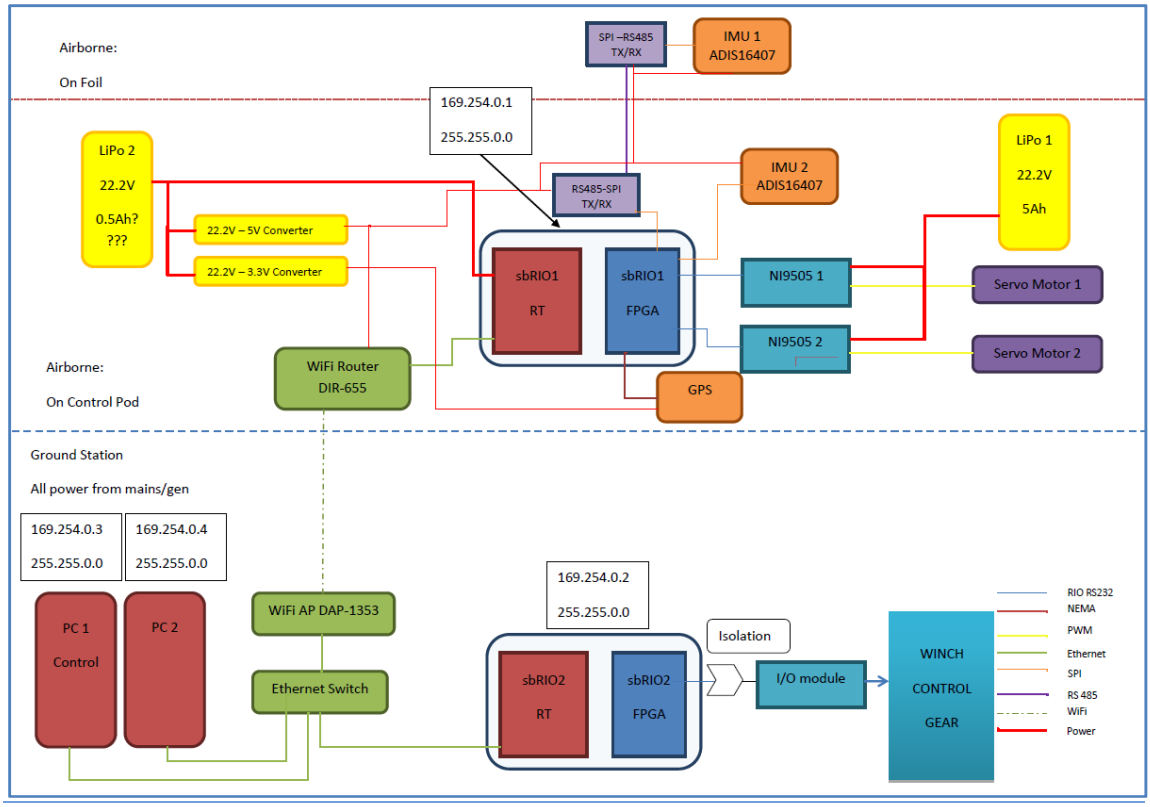


FIGURE 51.NETWORK DIAGRAM

2.1.9.2 DATA TO BE SENT/RECEIVED

The data that is sent from the Air station to the ground station are the following:

- Position of the air pod showing the latitude, longitude and distance from sea level using GPS coordinates.
- The actual axis orientations and accelerations of the kite and the pod using the data coming from IMU.
- The actual position and speed of the motors, using the feedback from the Encoder.
- The actual power consumption and energy left of the air pod.

The data to be received:

- Current limitation of the motor, maximum speed and position setpoint.
- Configure GPS mode.
- Configure IMU mode.

2.1.9.3 DISTANCE PROBLEMS

The distance between both stations would be approximately 150 meters in a straight line of sight.

The ranges of commercial wireless routers can be checked with tool from Radiolabs, where you can specify the characteristics of the router being used and it outputs the real distance availability. Some of commercial routers are checked in the following table:

REFERENCE	Power Consumption	Distance	
WLg-LINK-OEM	3.5 Watts typical, 5 Watts maximum	0.4km	
WLg-xROAD/N	3.6W typical power consumption, 4W for the /NP model	0.3km	
WIZ6000	Under 600mA (3.3V) = 2W	0.2km	

TABLE 12.WIFI COMPARISON

As can be seen, the theorical distance is too close to the maximum practical distance, and as loosing connection between the two stations could be a critical problem, long range Wi-Fi technology would be used.

2.1.9.4 ROUTER USED

The router that will be used in the air part is the DIR-655 from D-Link . It uses the draft 2.0 from the new 802.11n standard. The distance reached of the router is six times longer than the standard ones, by the combination of the 802.11n standard and the use of Xtreme N technology. Also it uses 3 externals antenna, that help in extending the network area.

2.1.9.4.1 EXPANSION OPTIONS

The router is used to be able to use the other Ethernet inputs for expansion slots. One of this possible expansion is an IP camera, in order to be able to see the images from the air pod of the ground or of the foil, and to detect possible problems without having to return the pod to the ground for investigation.

Another expansion possibility is to use an Ethernet-to-serial interface that could allow the use of more serial devices in the future without having to purchase the expensive expansion modules from National Instruments.

2.1.10 FINAL BOARD

The schematics and the layout of the final board are stated in this chapter. In it there are all the elements that have been explained in the hardware chapter, except the sending part of kite's IMU, as it is placed in a different board, as it is placed in the kite.

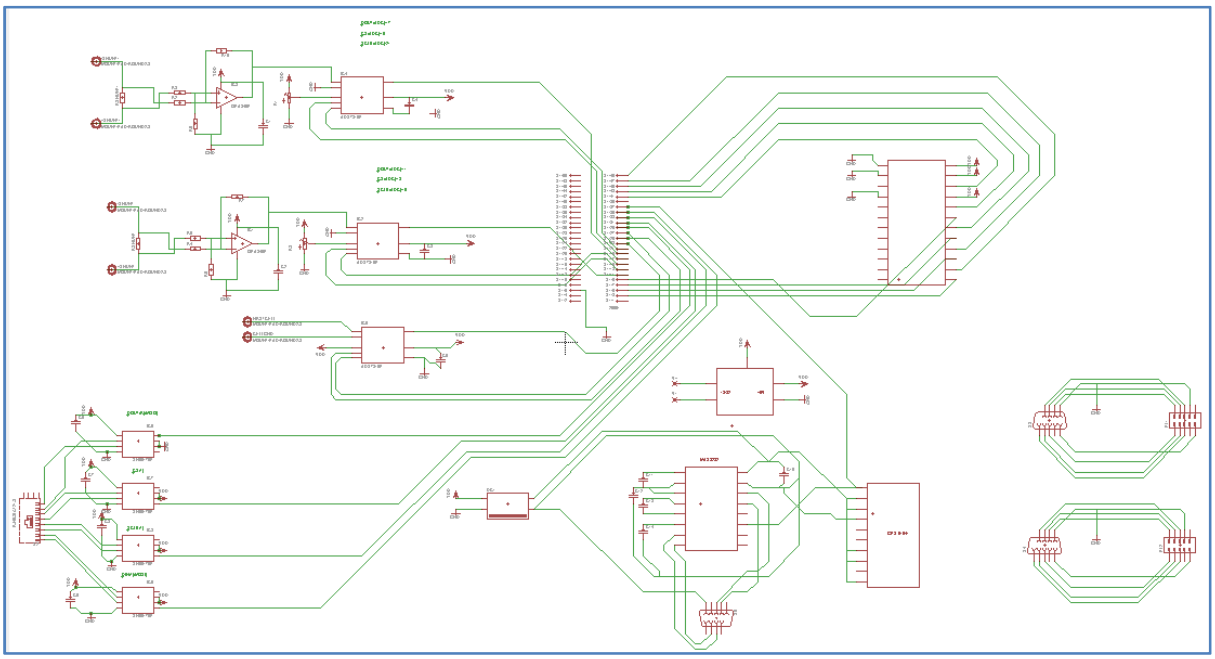


FIGURE 52. FINAL BOARD SCHEMATICS

The following layout has been implemented with Eagle Cadsfot software, based in previously stated schematics:

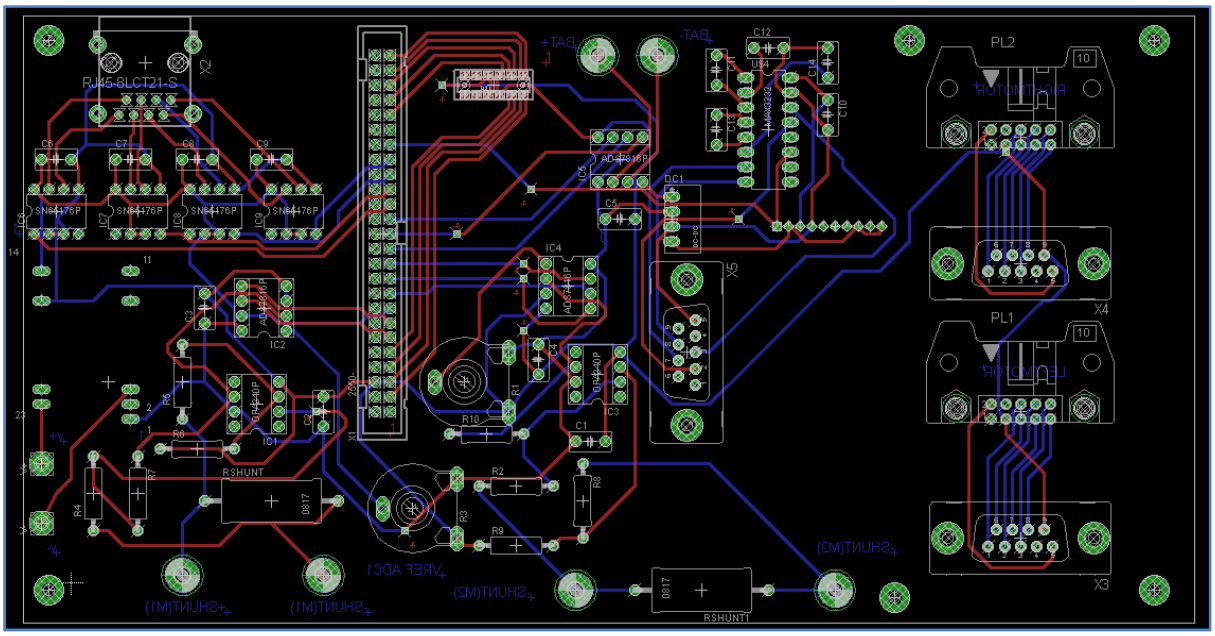


FIGURE 53. FINAL BOARD LAYOUT

3. SOFTWARE

Having examined the hardware development of the project, in this chapter software implementation and solutions will be analyzed:

3.1 IMU

The global software diagram of the IMU sensor is shown below:

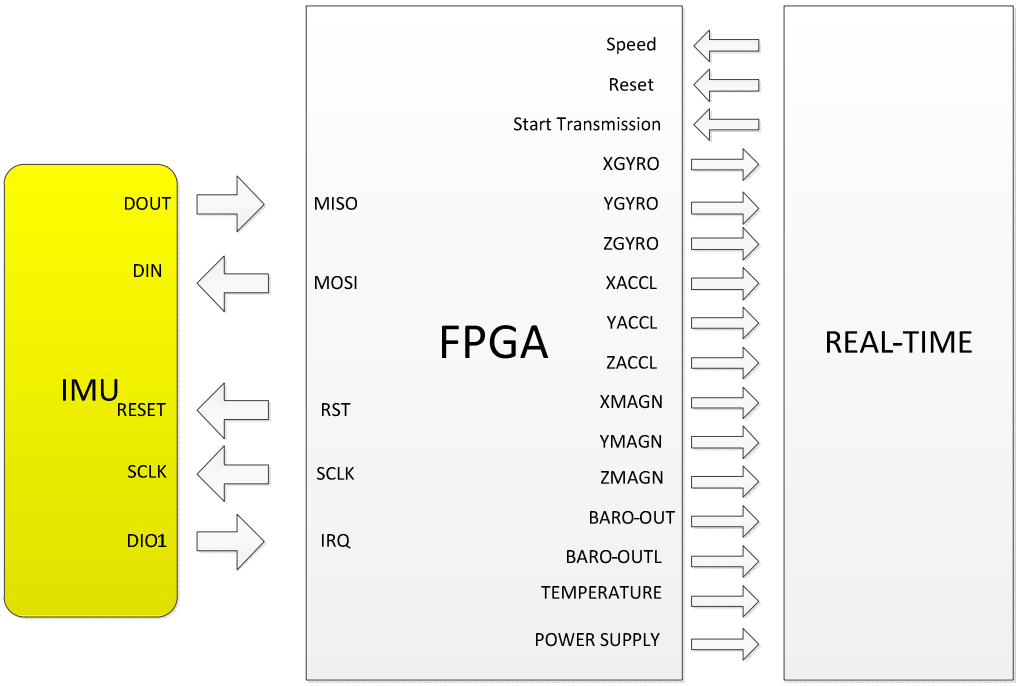


FIGURE 54.IMU-FPGA-REAL-TIME INTERFACE DATA EXCHANGE

The first target to be accomplished is to correctly communicate with the ADIS 16407 (IMU sensor). As explained in the previous chapter, the communication is established through SPI protocol. Once this is successfully done, a conversion of the data received by the sensor must be undertaken. Finally, the important data must be sent to de real-time target, so it can be used by the final user.

3.1.1 FPGA PART

The FPGA establishes communication with the IMU sensor, gets the data from it, and sends it to the Real-Time part.

3.1.1.1 VARIABLES

Variables used in the FPGA are listed below:

Communication Part

Input:

- MISO: Is the input of the SbRIO of the SPI protocol, so it is connected to IMU's data out.

Outputs:

- MOSI : Is the output of the SbRIO of the SPI protocol , so it is connected to IMU's data in.
- CS : Is the enable pin of the SPI protocol , so it is connected to IMU's enable pin.
- SCLK : Is the auxiliary clock used in SPI protocol , to synchronize slave and master .

Data Part

Inputs:

- Speed: Configures the speed of the reception of data, by modifying the frequency of SCLK and CS. Is selected from the Real-Time part.
- Reset: Reinitializes IMU sensor. Is selected from the Real-Time part.
- Start Transmission: When this bit is set, the communication with the IMU sensor is started. Is selected from the Real-Time part.

Outputs:

- XGYRO: Gyroscope output, x axis. Two bytes of data, including gyroscope data, new data and alarm indicator.
- YGYRO: Gyroscope output, y axis. Two bytes of data, including gyroscope data, new data and alarm indicator.
- ZGYRO: Gyroscope output, z axis. Two bytes of data, including gyroscope data, new data and alarm indicator.
- XACCL: Indicates the acceleration on the x-axis. Two bytes of data, including acceleration data in 2 complement new data and alarm indicator.
- YACCL: Indicates the acceleration on the y-axis. Two bytes of data, including acceleration data in 2 complement new data and alarm indicator.
- ZACCL: Indicates the acceleration on the z-axis. Two bytes of data, including acceleration data in 2 complement new data and alarm indicator.

- XMAGN: Indicates the strength of the magnetic field on the x-axis. Two bytes of data, including acceleration data in 2 complement new data and alarm indicator.
- YMAGN: Indicates the strength of the magnetic field on the y-axis. Two bytes of data, including acceleration data in 2 complement new data and alarm indicator.
- ZMAGN: Indicates the strength of the magnetic field on the z-axis. Two bytes of data, including acceleration data in 2 complement new data and alarm indicator.
- BARO-OUT: Indicates the barometric pressure. It is contained in two register (forming a 24 bits register in total), this one refers to the most significant bits. Two bytes of data, including barometric pressure data new data and alarm indicator.
- BARO-OUTL: Indicates the eight less significant bits of barometric pressure.
- TEMPERATURE: Indicates the internal temperature of the IMU sensor. Two bytes of data, including acceleration data in 2 complement new data and alarm indicator.
- POWER SUPPLY: Provides a measurement of the voltage that is in the VDD pins of the device. Two bytes of data, including acceleration data in 2 complement new data and alarm indicator.

3.1.1.2 SPI Protocol : Further Details

SPI MODE

ADIS 16407 is controlled, as has been already explained, using SPI protocol. The mode used is 1,1, meaning that channel select is active at low level, and SCLK is idle at high level. It also means that data is captured in SCLK's rising edge , and data is propagated in falling edge .

To be sure that the bit to read is stable , the readings must be done in SCLK rising edge , and writing in falling edge.

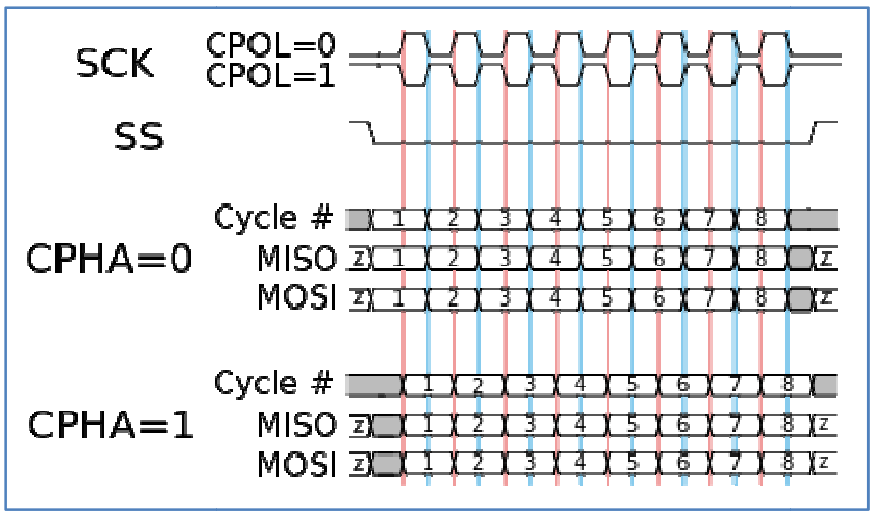


Figure 55.SPI protocol modes

COMMUNICATION PROTOCOL

The channel select is activated at low level, so a falling edge starts communication. After CS has gone low, SCLK generation have to start: each transmission needs an exact number of 16 SCLK falling edges.

The shape of the two signals is the following:

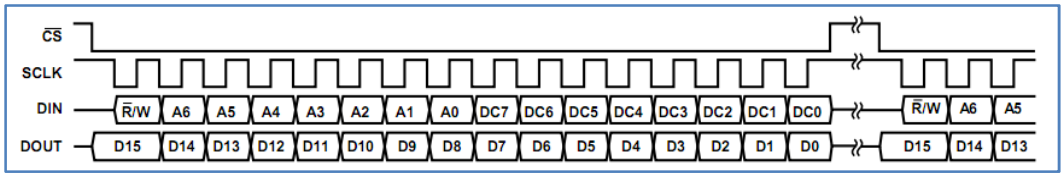


FIGURE 56.ADIS 16407 SPI DATA FORMAT

So for example if interaction with a specific register is wanted, like GPIO_CTRL (auxiliary digital input/output control) , it needs to first be decided if writing or reading is required. If a reading is required a 0 must be inputted in the MSB, followed by the address of the specified register (0x36). If for instance a writing is required, the MSB must be 1, followed by the address(0x36) plus the data that wants to be written in the register , if for example a 0x01 wants to be written , the final value of MOSI must be: 0xB601.

A complete example of reading more than one register will be explained, specifically the three gyroscopes registers x,y,z will be read in this order:

First , a 0x0400 needs to be written in the first SCLK block , so the value of the x-axis gyroscope register will appear in the next SCLK block , in DOUT . After that, a reading of the y-axis register must be performed, so a 0x600 will be written in the second block. So in this block a reading of the x-axis register will be done simultaneously with a y-axis register writing. Finally, in the third block, the y-axis register will be read, and the z-axis written, by forcing a 0x0800 in DIN input.

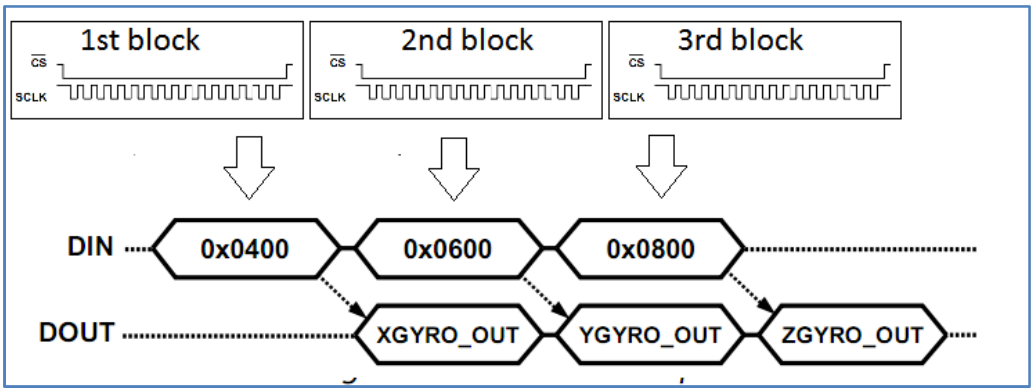


FIGURE 57.WRITING AND READING FOR ADIS16407 EXAMPLE

Some timing considerations must be taken in consideration when designing the signals. Beginning with the time between SCLK blocks which is 9 µs minimum, or the time between SCLK block, which is minimum of 40us.

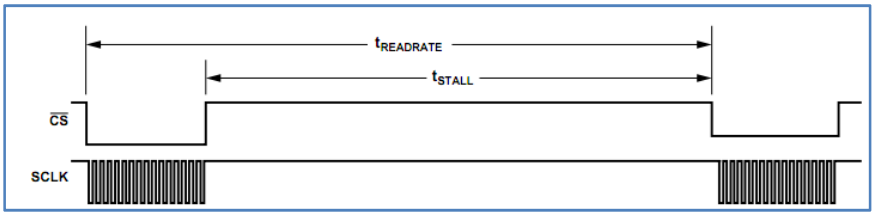


FIGURE 58.TIME STALL AND READ RATE TIME

The minimum time between chip select to SCLK edge is 48.4 ns, and in the opposite case, CS high after SCLK edge, the minimum time is 5 ns. There are also other timing considerations regarding the moment when the data is valid , which will not be analyzed deeply because interaction with the data (read/write)is done in the next valid edge, so no considerations about minimum time must be taking regarding this issue. The minimum SCLK frequency in normal and burst mode (it will be explained in the following chapter) is 0.01MHZ, and the maximum is 2 MHz for normal mode and 1 MHz for burst mode.

NORMAL AND BURST MODE

Everything outlined previously was with reference to normal mode. There also exists the possibility of using a burst mode, especially useful if the gathering of information from all the sensors simultaneously is required.

The working mode is the following:

In the first SCLK block, writing with a 0x4200 value needs to be done, to indicate that a burst mode reading is to be made. After this block, the following registers with this order will appear:

Register	Address	Measurement
SUPPLY_OUT	0x02	Power supply
XGYRO_OUT	0x04	Gyroscope, x-axis
YGYRO_OUT	0x06	Gyroscope, y-axis
ZGYRO_OUT	0x08	Gyroscope, z-axis
XACCL_OUT	0x0A	Accelerometer, x-axis
YACCL_OUT	0x0C	Accelerometer, y-axis
ZACCL_OUT	0x0E	Accelerometer, z-axis
XMAGN_OUT	0x10	Magnetometer, x-axis
YMAGN_OUT	0x12	Magnetometer, y-axis
ZMAGN_OUT	0x14	Magnetometer, z-axis
BARO_OUT	0x16	Barometer/pressure, higher
BARO_OUTL	0x18	Barometer/pressure, lower
TEMP_OUT[1]	0x1A	Internal temperature
AUX_ADC	0x1C	Auxiliary ADC

FIGURE 59.TABLE WITH BURST MODE REGISTERS

So after having written the burst mode in the first SCLK block, from the second until the fifteenth block the different registers will be able to be read, without having to make the channel select go idle as shown below:

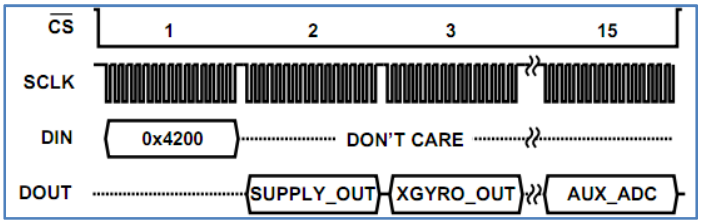


FIGURE 60.BURST MODE

It needs to take in consideration that the minimum time stall for this mode is:

$$Tstall > \frac{1}{fsclk}$$

The main benefit of this mode , is that an idle state after reading each register does not need to be carried out, making the reading much faster(also there is only need to write in the first block , and read in all the others):

The problems of using it, is that the maximum speed of SCLK is reduced by half (from 2MHz of normal mode to 1MHz). Also, the system is less robust to noise interferences, as only one writing is made, so if this is corrupted all data will be lost, rather than just the data from one sensor.

As two different IMUs have to be controlled , one next to the board and another on the kite (a minimum distance of 5m present) , timing considerations must be taken for the second one , as SPI protocol is in-board designed , so it is not reliable over long distances , as has been explained in previous sections. The implication that this has in the software part , is that both modes(normal and burst) need to be implemented , because the burst mode is not the best option for the kite IMU (for speed limitations and interferences present), and burst mode is the most useful for in-board IMU.

3.1.1.3 IMPLEMENTATION

NORMAL MODE:

The first step is to generate the auxiliary clock. It must have 16 falling edges (taking in consideration that the idle state is high level). The implementation done is the following:

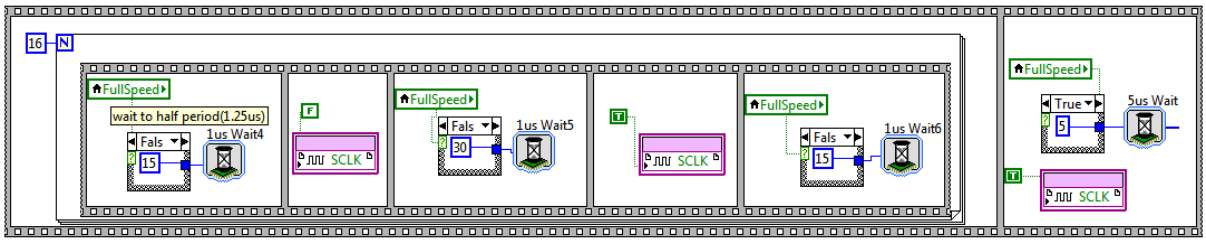


FIGURE 61.SCLK GENERATION CODE

So the upper structure is sequence one, so they will be executed in order, from left to right. The one inside the first sequence is a for iteration, in this case, the code inside will be executed

16 times. Inside it, it can be seen as that there is another sequence structure: in the first frame there is a delay, the amount of which is determined by the speed configuration coming from the Real-Time part. The unit of the delay is microseconds. On the next frame , after the delay has been executed , the bit SCLK will go to low level almost immediately (for this application the rising/falling time of the signals is insignificant) , so there will be another delay configured by speed in third frame , and in the fourth it will go to high level again. On the last frame inside the sequence structure inside the for loop, there is another delay. So after one iteration the signal is the following (for full speed configuration):

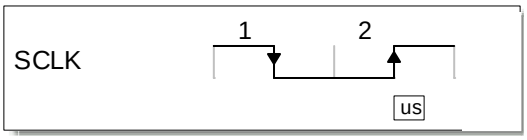


FIGURE 62.ONE PERIOD OF SCLK

After the 16 iterations inside the for loop, the signal is the one shown below:

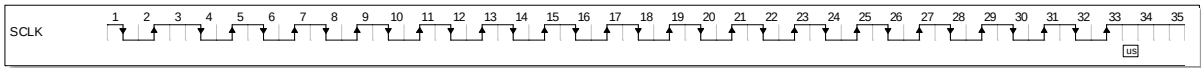


FIGURE 63.SCLK GENERATION OF 16 FALLING EDGES

So the frequency of the SCLK at full speed is:

$$Tsclk = 4us \rightarrow fsclk = \frac{1}{4us} = 250kHz$$

The frequency of the SCLK at low speed is:

$$Tsclk = 60us \rightarrow fsclk = \frac{1}{60us} = 16.\hat{6}kHz$$

So now there are 16 SCLK falling edges (remember that for mode 1,1 each byte is propagated after each SCLK falling edge).

The next step is to generate the CS signal. For this purpose, sequence structure is also used:

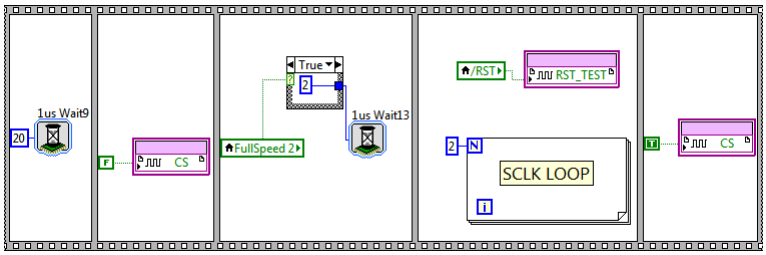


FIGURE 64.CS GENERATION

The first frame includes a delay to respect the stall time of the sensor specification. The minimum stall time is 9 µs, but as speed is not a critical feature in the project, a delay of 20µs has been implemented.

On the second frame, CS is activated (by forcing it to low level), so the transmission is started at this point. On the next frame there is a delay to keep the minimum specified time between CS falling edge and SCLK edge, the time of which depends again in the speed configuration. On

the fourth loop, is where SCLK is generated (also is where Reset is activated), and in the last one, CS goes idle again. So if the two signal generations are combined (without taking in consideration the for loop of SCLK generation), the following figure is obtained for full speed configuration:

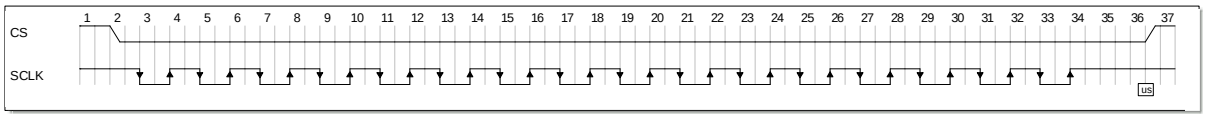


FIGURE 65.ONE SCLK BLOCK WITH CS

So a SCLK block is implemented like this. Now the generation of all the other blocks is needed. As it can be seen in the figure of the CS structure, the SCLK generation is inside a loop of two iterations, because for each register that is wanted to read, first is needed to access it by writing its address to DIN, and in the following SCLK block read the data from DOUT corresponding to the register accessed in the previous block. So the final signal is the following:

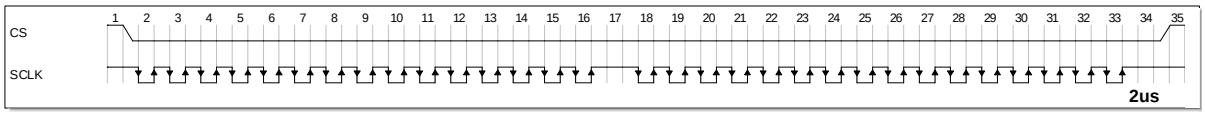


FIGURE 66.TWO SCLK BLOCKS WITH CS

Now a consideration in how to generate MOSI signal must be taken, and when to read the data coming from the sensor with MISO . Let's start with the first one : MOSI generation. The labview code is shown:

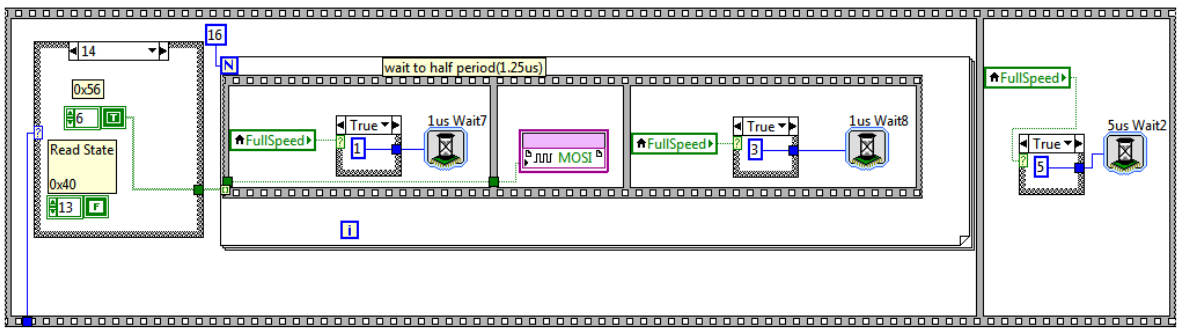


FIGURE 67.MOSI GENERATION CODE

The mode 1,1 has to be remembered , that data is captured in SCLK's rising edge . So in order to have the data ready in the rising edge , there is a need to dispatch it at SCLK's falling edge. The first step is to create a sequence structure that will be executed in parallel with the SCLK generation , so both will start at the same time. At that moment, there is a need to wait until the rising edge of SCLK, so 1µs delay is present . After it , MOSI is then set (this will be discussed later) , once this happens in the rising edge , 3µs have to be waited in order to have the same frequency as SCLK , with a period of 4µs. Both signals are shown below :

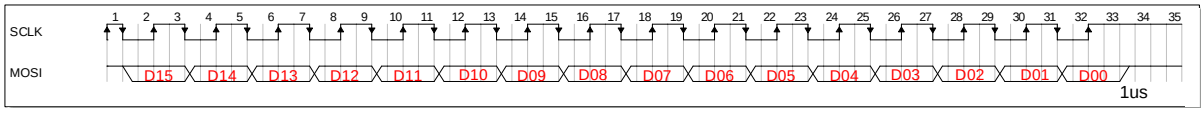


FIGURE 68.MOSI GENERATION SIGNAL

The writing is done by sending a different bit each iteration of the loop structure. On the normal mode, 14 registers are read, plus an extra one if any alarm is set.

The first register is the configuration register, which is written and set to enable the input DIO1 to make it generate a rising edge every time new data is ready. This feature is not implemented in this release, but can be implemented on future releases in order to allow the sensor to work with interruptions. The following registers are the same as in the burst mode, and the last one, the status registers is only read when any alarm is set, and makes a self-test and indicates if the sensor is properly working.

The next step is to read the register value in the next SCLK block. The structure is similar to the two explained before, so no more details will be introduced:

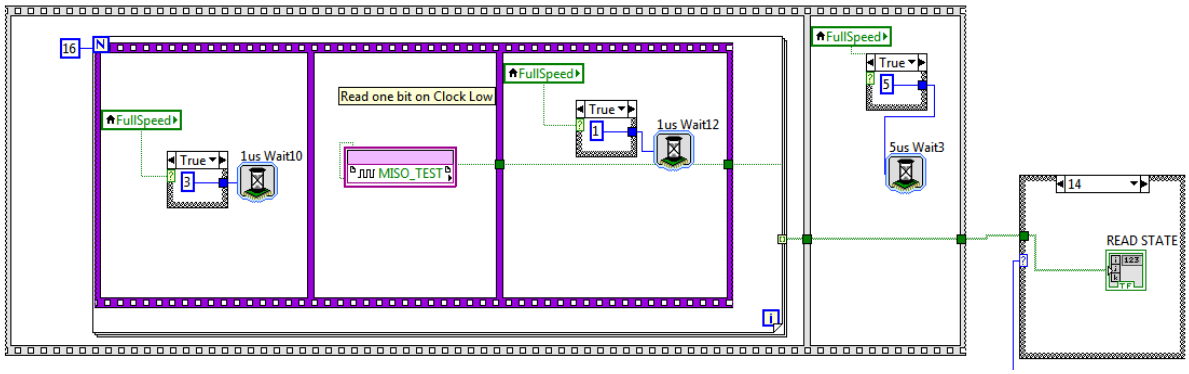


FIGURE 69.READING CODE

Every bit is stored in the specific byte array (on the same order as the registers for the writing case) , and each of the register is stored in a byte with the same name as the one being read , like for example the State register(iteration 14 , the last one) as in the above figure.

The reading has to be done in the rising edge of SCLK, as data is propagated in the falling edge, so at the rising edge the data is stable. The final aspect of a register addressing and reading is the following where the red arrows indicate the exact moment where data is read by the SbRIO:

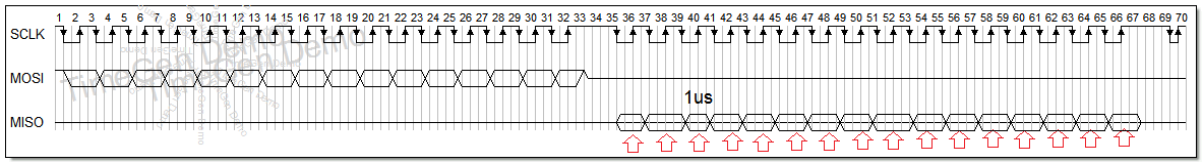


FIGURE 70.MOSI AND MISO IMPLEMENTATION

Now the implementation to make it a forever loop and to capture/write the fifteen different registers is shown:

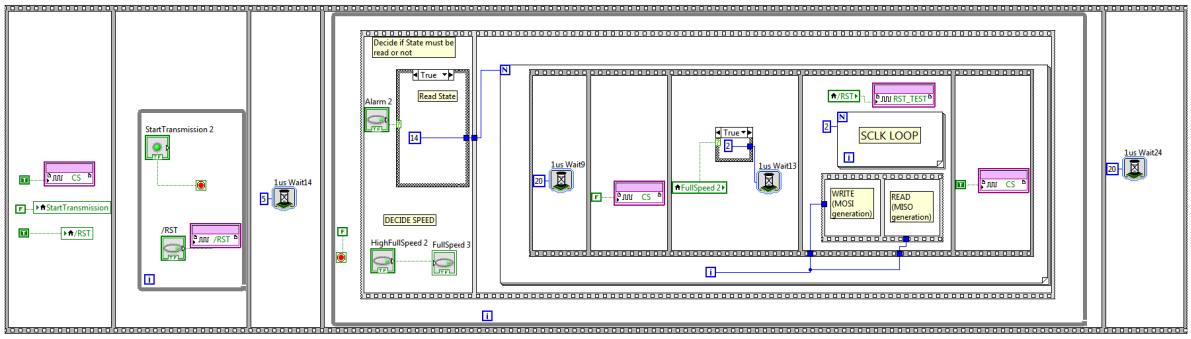


FIGURE 71.IMU IMPLEMENTATION OF THE FPGA PART

On the first frames all variables are set to default value, in the next one there is a waiting until the user from the real-time interface, allows the transmission to start. When this happens, there is a first delay in order to keep a minimum stall time. After that, the forever while loop starts. On the first frame from the structure inside the while loop , there is a case structure to decide if fifteen or fourteen transmissions are done , depending if any register alarm is present , case that an extra register of state will be read. In this frame also is implemented the speed configuration. After this , inside the for loop of the next frame , it has already been explained , the only feature to be explained is the iteration number of the loop coming into the MOSI and MISO generation , that refers to which iteration is currently being executed , so depending on it , a different register is accessed and read.

The final aspect of a successful is shown:

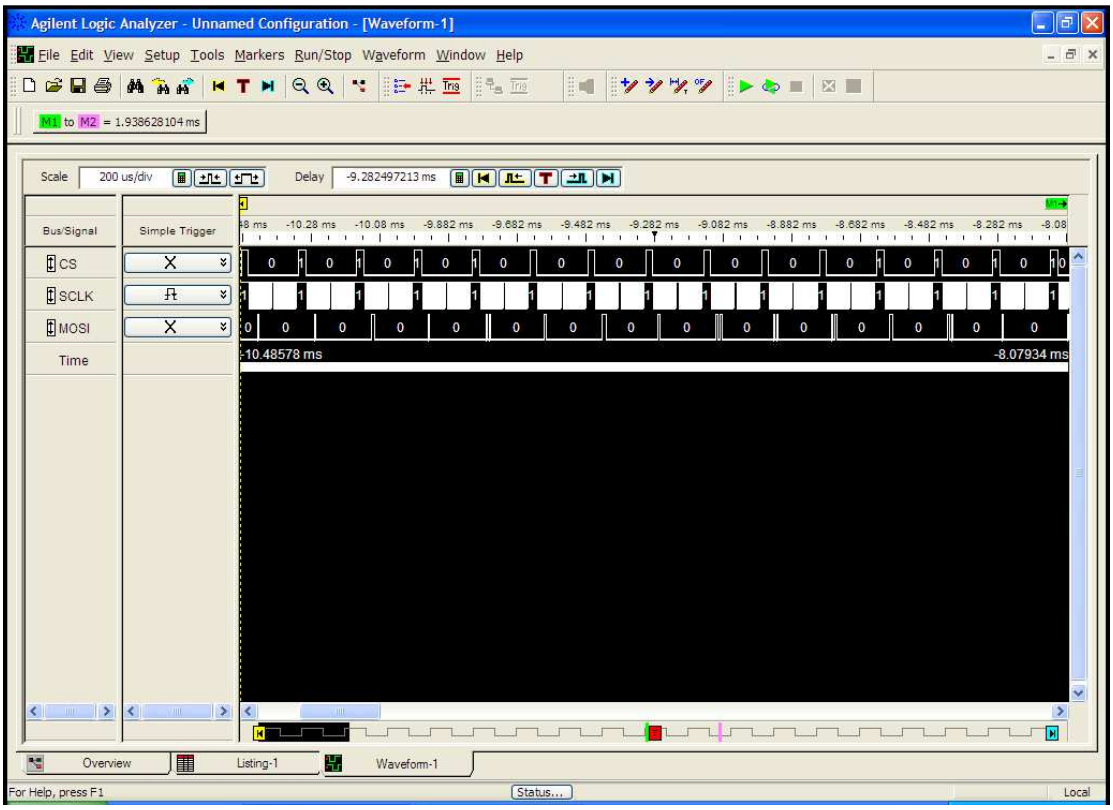


FIGURE 72.REAL SIGNALS OBTAINED

BURST MODE:

The implementation of this mode is very similar to the normal one, with the following differences:

- The SCLK is now generating 15 SCLK blocks, instead of the two of the normal mode for each iteration, because it needs one SCLK block to write the register to indicate the sensor that Burst mode is desired, so in the fourteen next SCLK blocks, each one of the burst registers is read, with the order specified before.
- On the same line as previous point, instead of one writing block , and one reading block per iteration of the Normal mode , now a writing block is set, and fourteen reading blocks.
- If the alarm is set, after having read the burst mode, the normal mode is used in the following iteration to read the status register.

The code of the burst mode is shown below:

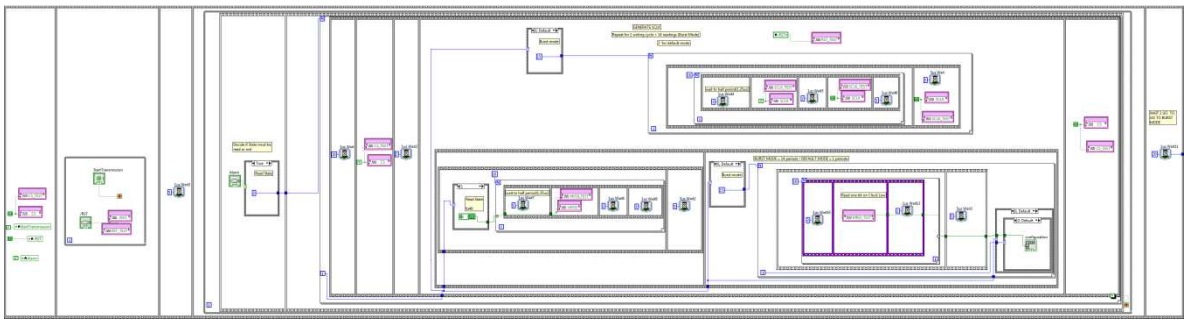


FIGURE 73.BURST MODE CODE

TRANSCEIVER CAPTURES

The main problem of the SPI protocol is that it is very sensitive to wiring distances due to the increment of the parasite capacitance of the circuit over long or medium distances. So as has been explained in the hardware chapter, the addition of transceivers is necessary for kite's IMU as its minimum distance to the SbRIO is 5 meters. The connection over transceivers will make the communication possible, but will also add a speed limitation, based on the delays from the propagation of the signal from the control part or the sensor, which if to large will cause data loss between both parts. The maximum speed of data acquisition is discussed below.

The minimum speed is 1Hz, shown below:

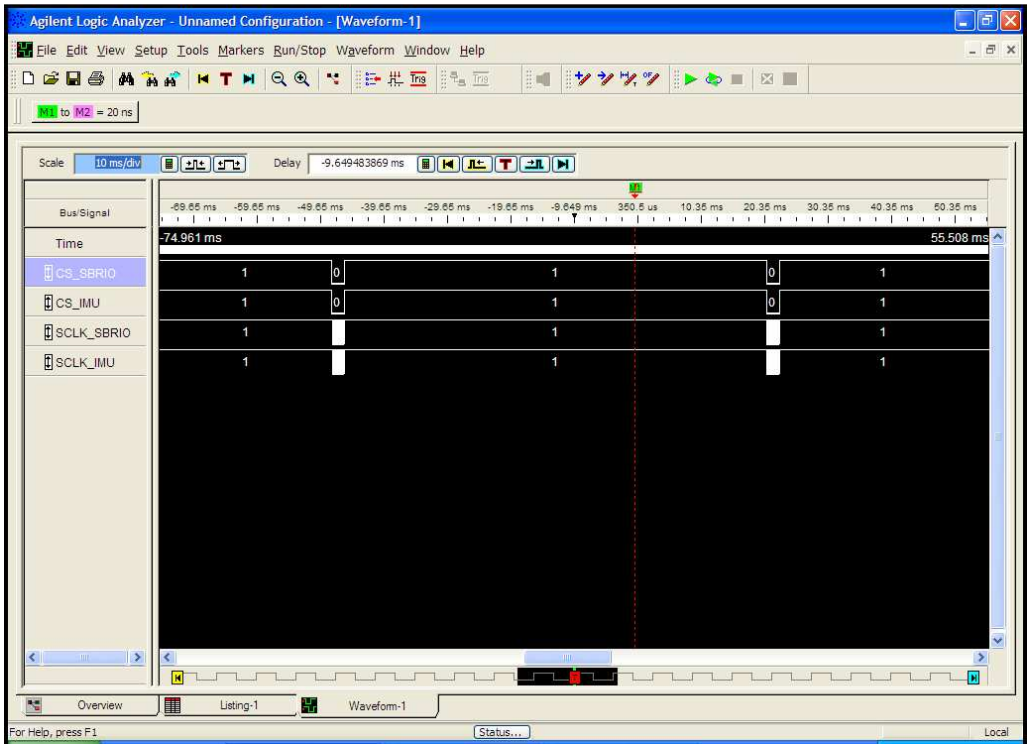


FIGURE 74.SPI SIGNAL AT 1HZ

A zoomed in signal illustrates the delay between both signals:

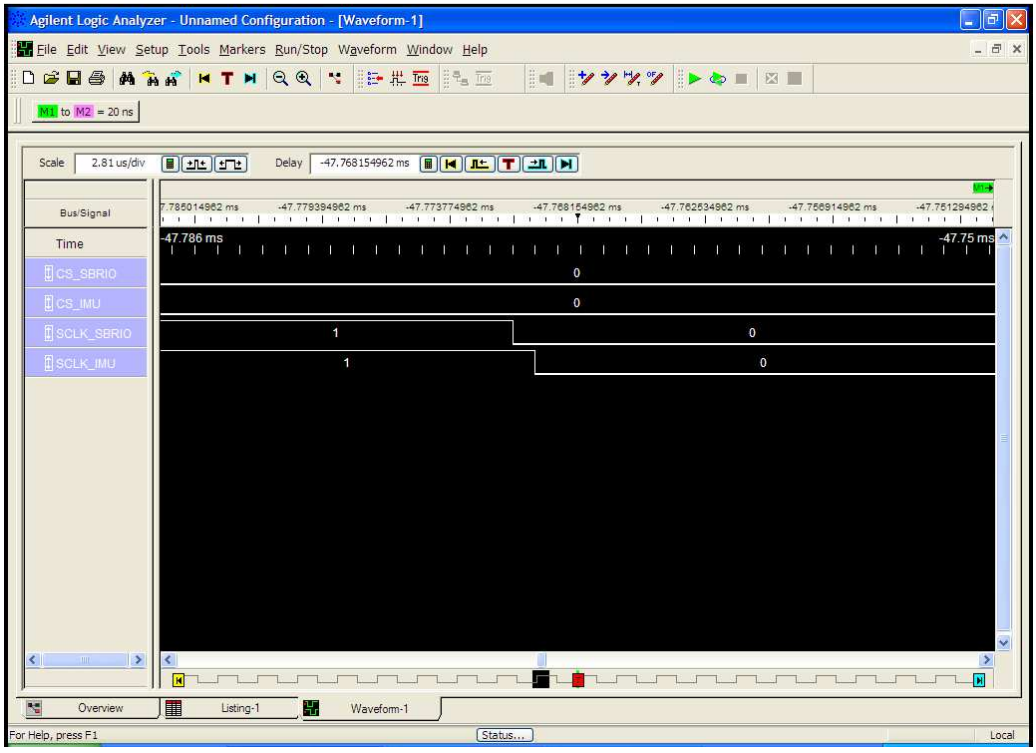


FIGURE 75.ZOOM OF SCLK'S FALLING EDGE AT 1HZ

The delay appreciated is less than 0.01 ms , so at this speed the communication will be robust , as the reading of the data is done in SCLK rising edge , so the data will already be stable in that moment , making the reading correct.

If the speed is configured to achieve the maximum speed:

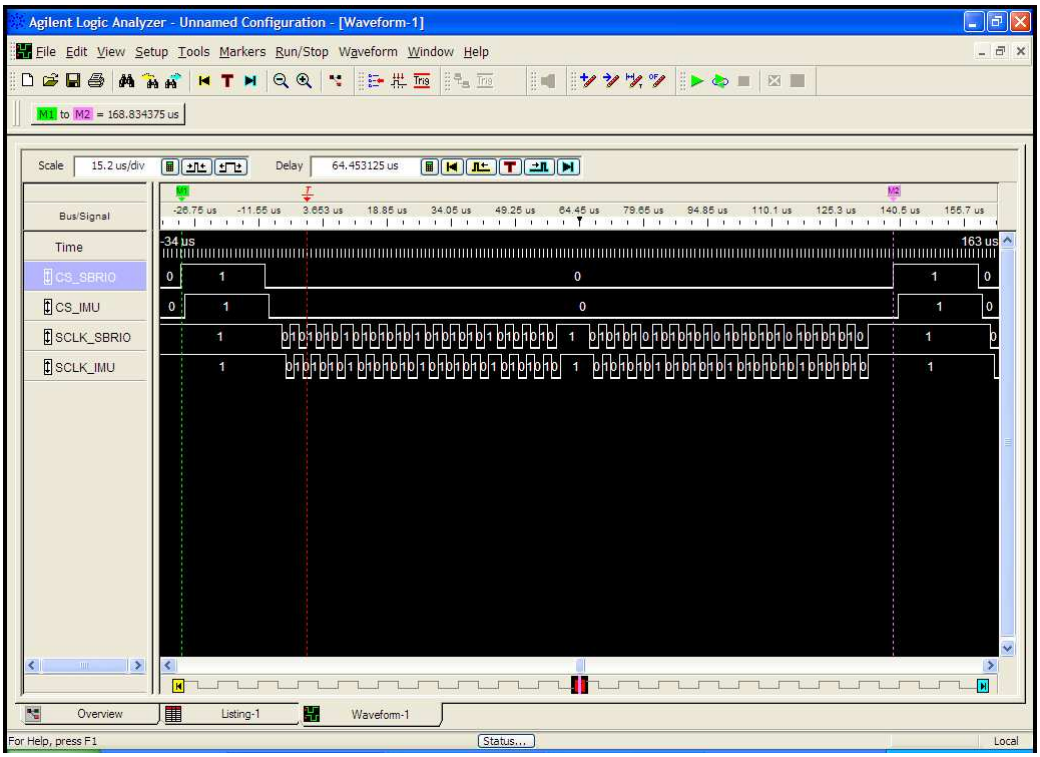


FIGURE 76.SPI SIGNAL AT MAXIMUM SPEED

Making another zoom the following signal is obtained:

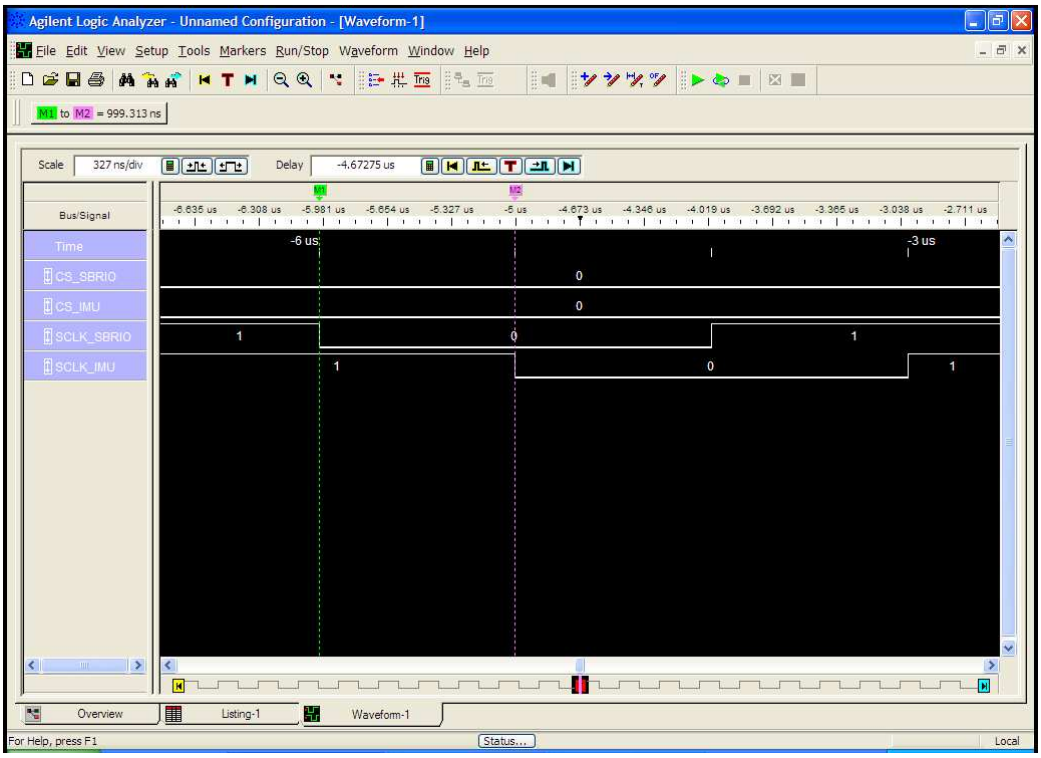


FIGURE 77.ZOOM OF SCLK'S FALLING EDGE AT MAXIMUM SPEED

So now the delay makes that the SCLK gets to the IMU when the SbRIO's SCLK is in the middle between rise and fall edge, making the value in the reading unstable, translating into incorrect readings.

The maximum achievable working frequency is 33.3Hz:

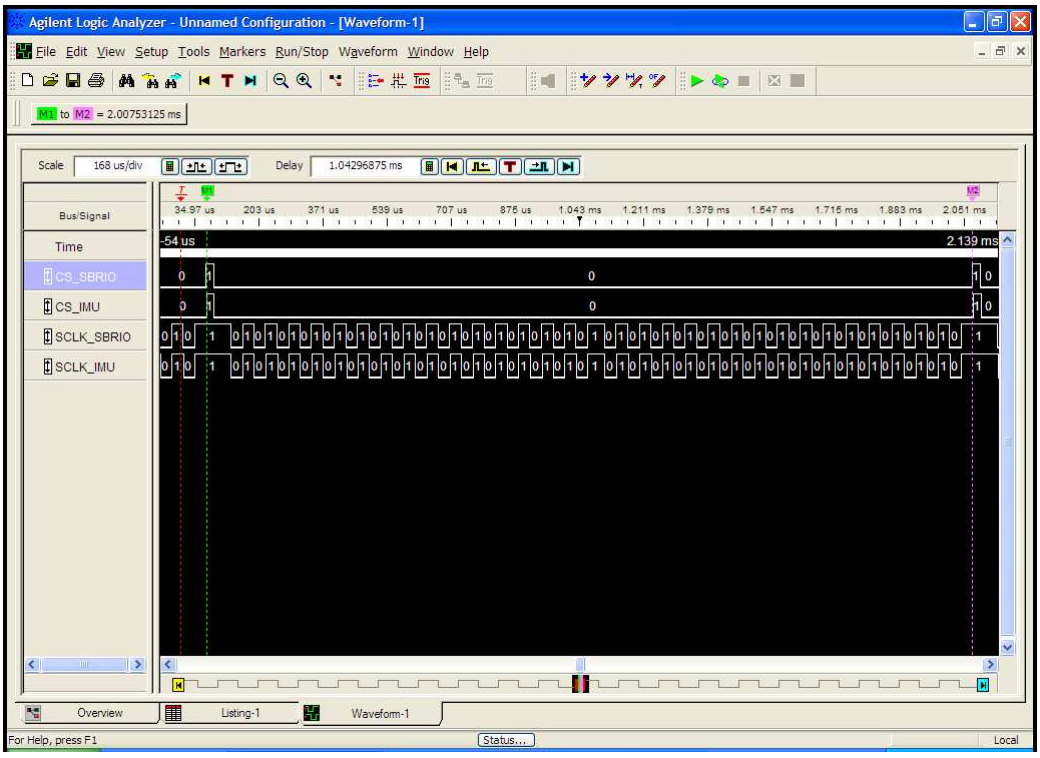


FIGURE 78.SPI SIGNAL AT MAXIMUM WORKING SPEED

If a zoom is made:

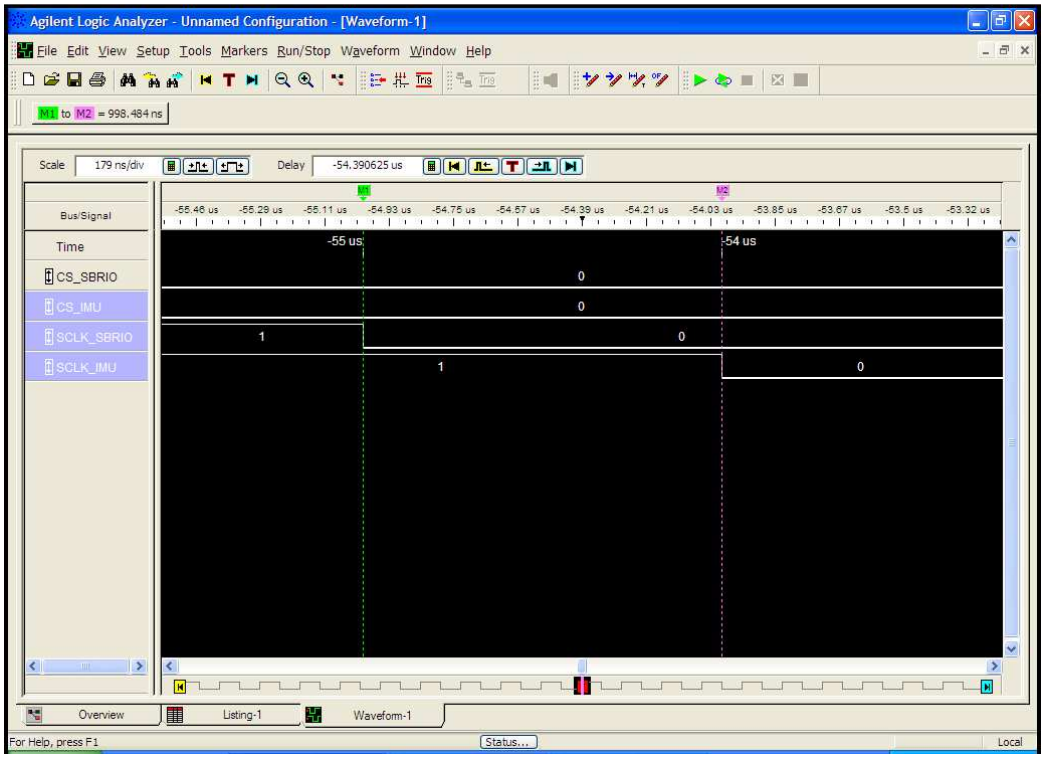


FIGURE 79.SPI SIGNAL AT MAXIMUM WORKING FREQUENCY SCLK'S FALLING EDGE ZOOM

The delay is 998.4 ns, resulting in a rising edge the value that is stable, so the reading will be correct.

3.2 MOTOR+NI9505

The global software diagram of the Motor control and actuation is shown below:

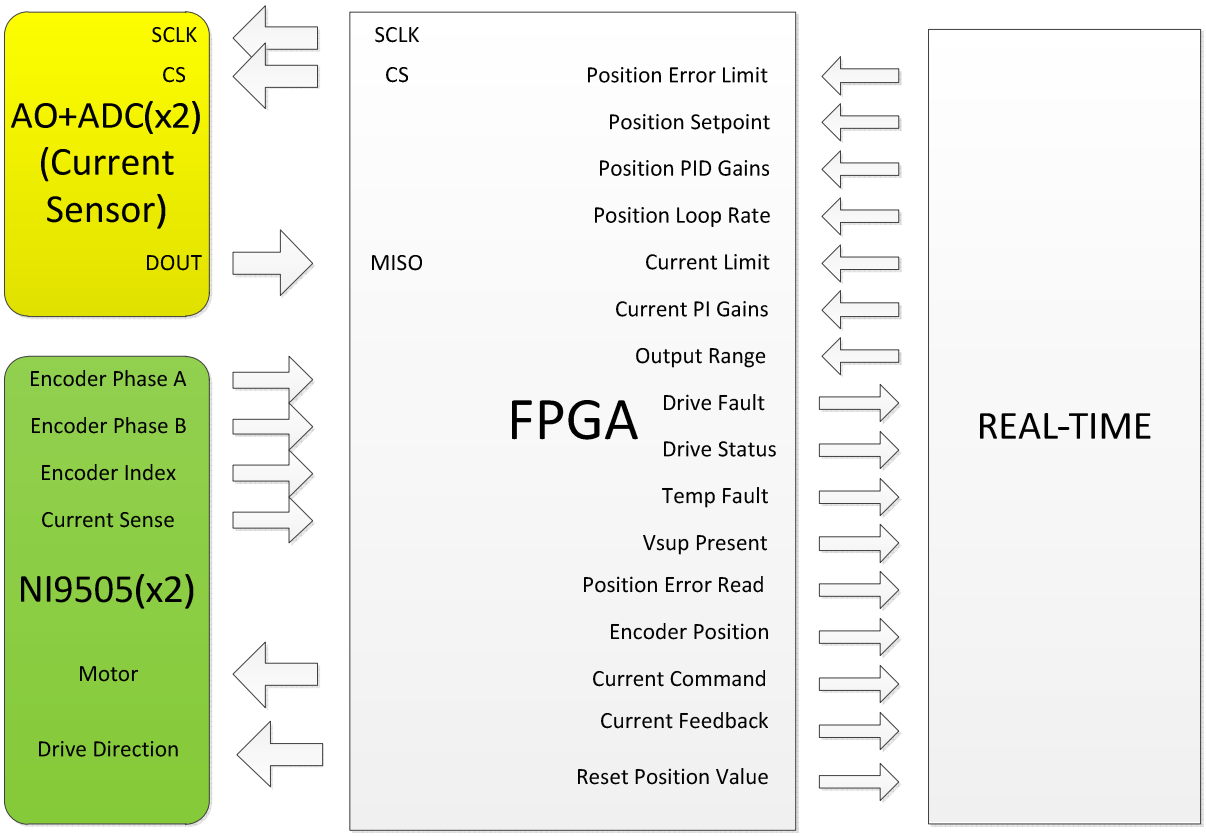


FIGURE 80. INTERACTION BETWEEN THE FOUR MODULES

The main goal of this block is to efficiently control the two motors. For this purpose two NI9505 blocks are used. The control is done, by a position PID, controlled at the same time by a current PI. The current sensing is done simultaneously by two different current sensors, as has been explained in the hardware section. Finally, the important data, such as the current being consumed, the speed, the actual position and other important feedback must be sent to the real-time target, so it can be accessed by the final user.

3.2.1 FPGA PART

The FPGA establishes communication with the NI9505 and the ADC used in the current sensing.

3.2.1.1 VARIABLES

Variables used in the FPGA are listed below:

ADC

Input:

- MISO: Is the input of the SbRIO of the SPI protocol, so it is connected to ADC's data out.

Outputs:

- MOSI : Is the output of the SbRIO of the SPI protocol , so it is connected to ADC's data in.
- CS: Is the enable pin of the SPI protocol, so it is connected to ADC's enable pin.
- SCLK: Is the auxiliary clock used in SPI protocol, to synchronize slave and master.

Ni 9505

Inputs:

- Position Error Limit: Selects the maximum error limit present in the position PID. Is selected from the Real-Time part.
- Position Setpoint: Position (in degrees) where the motor has to go first, and maintain it until it is changed. Is selected from the Real-Time part.
- Position PID Gains: Proportional, derivative and integral gains of the position loop. Is selected from the Real-Time part.
- Position Loop Rate: Time between PID calculations. Is selected from the Real-Time part.
- Current Limit: Maximum current to be asked for from the current loop, to protect from overcurrent, or to work with a maximum desired torque. Is selected from the Real-Time part.
- Current PI Gains: Proportional and integral gains for the current loop. Is selected from the Real-Time part.
- Output Range. Maximum PWM duty cycle (speed) applicable for the current loop. Is selected from the Real-Time part.

Outputs:

- Drive Fault: Indicates if an error has stopped NI9505 module.
- Drive Status: Indicates if the NI9505 is currently enabled, or if it has been disabled shows the error.
- Temp Fault: Maximum or minimum current has been achieved by the NI9505 if the bit is set, so it has been disabled.
- Vsup Present: Is set if the motor from the NI9505 has a valid power supply connected to it.
- Position Error Read: Difference between current setpoint and actual position.
- Encoder Position: Indicates the actual position (in degrees) of the motor.
- Current Command: Indicates the current that has to be given to the motor, assigned by the current loop.
- Current Feedback: Current coming from the current sensor. As has been already explained in the hardware chapter, it will come from the NI9505 sensor, or from the Shunt current measure depending in the situation.
- Reset Position Value: Indicates the Home position of the NI9505.

3.2.1.2 ENCODER LOOP

The encoder is used do mainly determinate position and speed of the motor.

The code of the encoder loop is attached:

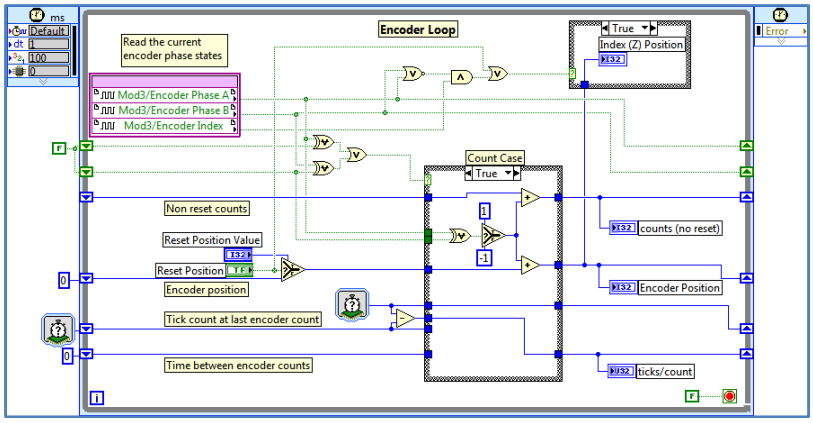


FIGURE 81. TRUE COUNT CASE

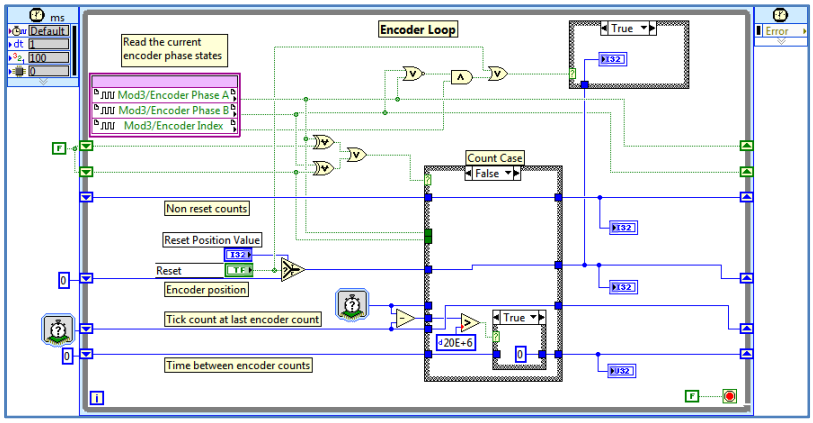


FIGURE 82. FALSE COUNT CASE

When a difference of phase A or B is detected from the last timed loop execution, the count case is true, false in the opposite case. On the first situation, the counts counter is incremented (if a pass per zero has been detected) and the ticks are set to the result of the difference between the total amount of ticks of the last iteration and the current one. This last variable will be used in the real-time part to determine the speed of the motor. The position of the encoder is also determined with the detection of a change of phase (as only phases per revolution of the encoder needs to be known). If no quadrature counts are detected for half second ($20 \cdot 10^6$ticks for a 40MHz clock), the ticks/count variable is set to zero to indicate a null velocity.

3.2.1.3 CURRENT SENSING LOOPS

As has been explained in the hardware chapter there are two different current sensors, so there are two different current sensing loops, and one current PI where one of the two measures is used. The first to be explained is the one implemented with a shunt and an ADC:

ADC Sensor:

The AO and current ranges are explained in the hardware chapter, so it will not be explained in this one. The ADC integrated sensor used is the MCP3001, from Microchip. It uses SPI protocol, as it has been already outlined for the IMU sensor. The main difference between the two methods of gathering the data from the two sensors , is that in this case it is not necessary to write data to the ADC , because once it has a valid SCLK signal , it will start automatically sending the value obtained from the analog to digital conversion, as is shown on the following diagram:

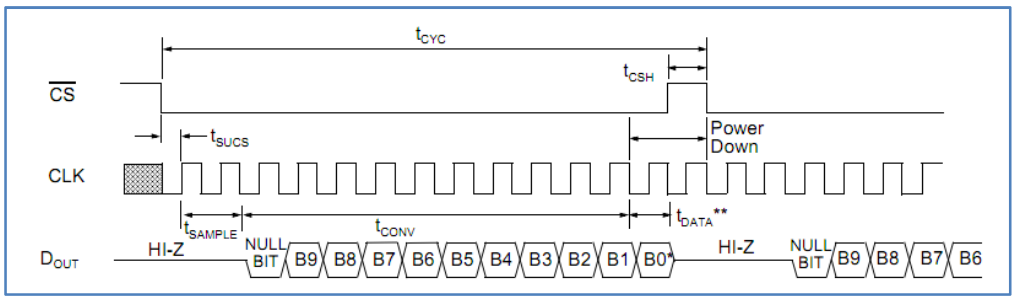


FIGURE 83.COMMUNICATION WITH MCP3001

The mode can be the 0,0 and the 1,1 . For simplicity the last one has been chosen, so the data is propagated in SCLK falling edge. To ensure that the signal is stable at the time of reading, the data will be read in the rising edge of SCLK.

The first three valid bits don't have to be taken into consideration during the conversion. So from the fourth rising edge, the bits have to be registered.

The maximum SCLK frequency is 1 MHz, the minimum time from CS fall and SCLK rise is 100 ns, and from the last SCLK edge to CS rise 50ns. On this case the same speed configuration has been implemented like the one present in the IMU sensor.

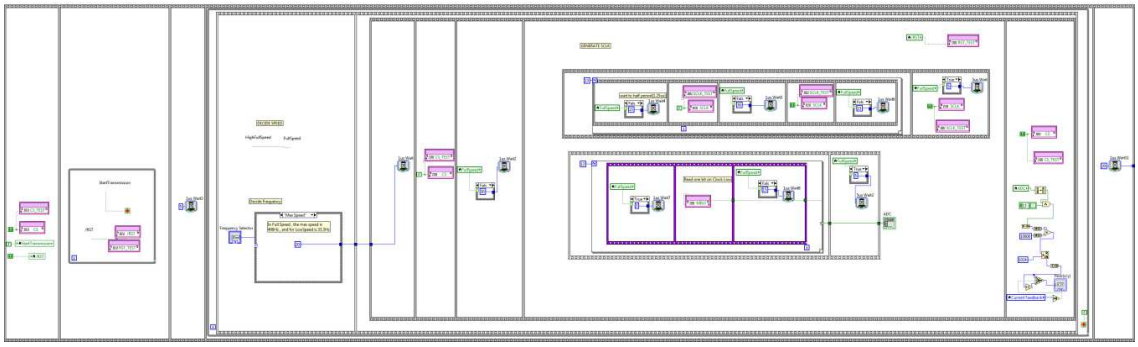


FIGURE 84.ADC READING CODE

As the similarities with the IMU sensors are obvious, only the last frame inside the while loop is going to be analyzed:

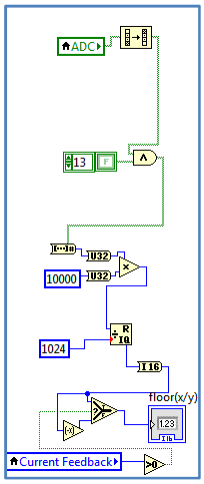


FIGURE 85.ADC DATA CONVERSION INTO CURRENT

After each iteration of the ADC has been executed, a conversion of the obtained value to current is done. As can be seen, the ADC value is being introduced to a AND gate, in order to make a mask and clear the three most significant bits, as they don't have to be read. After it, a conversion into an unsigned word is done (the ADC variable is a string of binary bits), and multiplied by ten thousand, in order to convert it to miliamperes. Is then divided by the total amount of bits of the ADC (2^{10}) , and converted into a signed unit in order to establish the sign, depending in the direction of the motor, which is taken from the NI9505 current sensor, because with the shunt coming from the battery is impossible to know the polarity, as access to the H-bridge is not possible.

NI9505 current sensor:

The current sensor of the NI9505 has to sense the current in the half of the PWM duty cycle (high time) in order to allow the current of the motor to be stable, so the reading is accurate. A trigger is used to determine when the current has to be read, it has to be set 9 ticks before the half of the period , because it is the time it takes to sample the current, so for 9 ticks , the trigger is enabled. The following code is the loop used to generate the PWM, and to set the current trigger. The PWM generation will be explained later on.

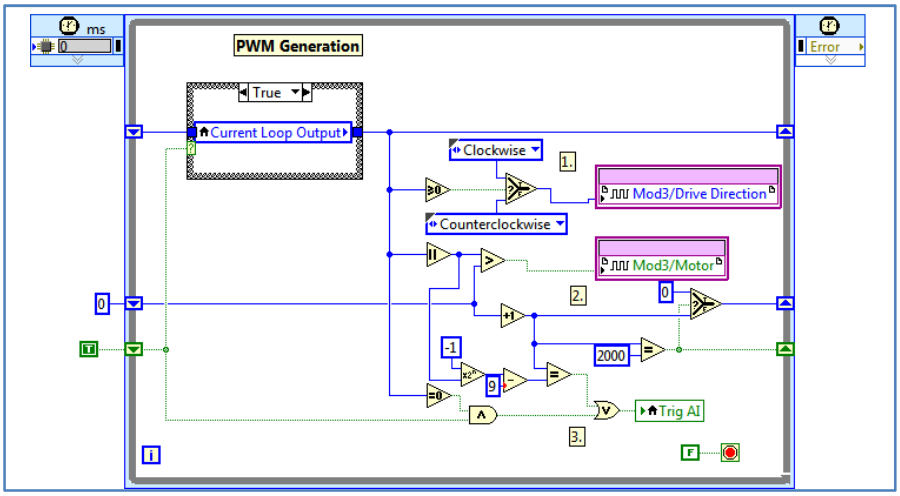


FIGURE 86.PWM GENERATION LOOP (CURRENT TRIGGER ALSO PRESENT)

Once the current trigger is set, the value from the sensor is transferred to the local variable that will be used to control the motor:

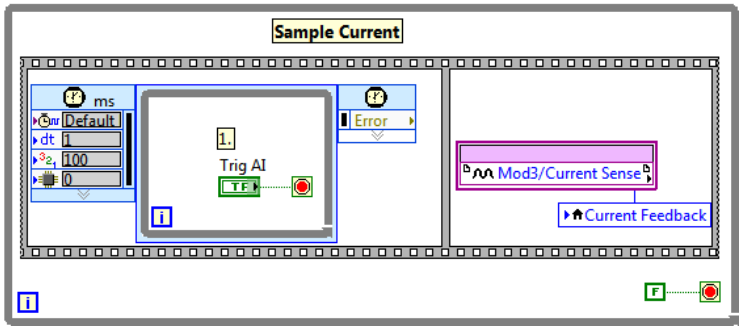


FIGURE 87.SAMPLE CURRENT LOOP

PID CONTROLLER

A proportional integral derivative controller calculates the difference between a specified setpoint and a measured process variable, and tries to make approximate it to zero by adjusting the gains, in order to control a system or a process.

The PID is an algorithm based on three constant parameters:

- Proportional: The present error is based on this parameter.
- Integral: Acts taking the accumulation of past errors in consideration.
- Derivative: Predicts the future error, based on the current rate of change.

By tuning the three parameters in the PID controller algorithm, the controller can provide control action designed for specific process requirements. The response of the controller can be described in terms of the responsiveness of the controller to an error, the degree to which the controller overshoots the setpoint and the degree of system oscillation. Note that the use of the PID algorithm for control does not guarantee optimal control of the system or system stability.

Below is shown the algorithm of a PID controller:

$$u(t) = Kp \cdot e(t) + Ki \int_0^t e(\mathbb{T}) \cdot d\mathbb{T} + Kd \frac{d}{dt} e(t)$$

Where:

Kp: Proportional gain

Ki: Integral gain

Kd: Derivative gain

e: Error

t: Time or instantaneous time (the present)

An example of a motor position control is explained. Figure 88 shows what happens when you add proportional feedback to the motor and gear system. For kp = 1 the motor gets to the position but quiet slowly. If the gain is increased by one unit, the response is faster, but if this gain is increased to five, the motor starts out faster, but it overshoots the target. In the end the system doesn't settle out any quicker than it would have with lower gain, but there is more overshoot. If the gain is increased it will get to a point where the system is unstable, as a lot of oscillation is present, so the setpoint is never reached.

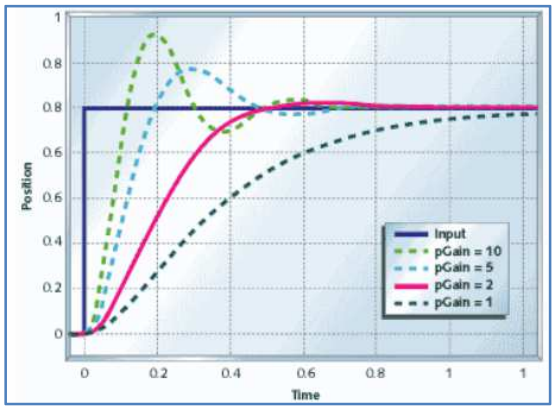


FIGURE 88.ONLY PROPORTIONAL CONTROL PRESENT

In order to avoid the system to never get to the target, integral control is added, and used to add long term precision to a control loop. It is almost always used in conjunction with proportional control. Figure 89 shows the motor and gear with pure integral control (pGain = 0). The system doesn't settle as it needs some proportional gain in order to be stable.

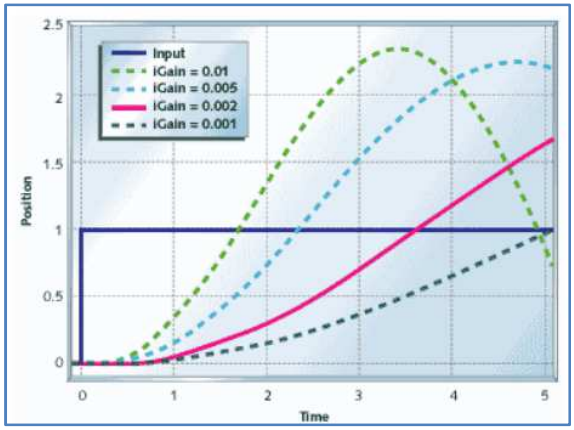


FIGURE 89. ONLY INTEGRAL CONTROL PRESENT

Figure 90 shows the motor and gear with proportional and integral (PI) control. Compare this with Figures 88 and 89. The position takes longer to settle out than the system with pure proportional control, but it will not settle to the wrong spot.

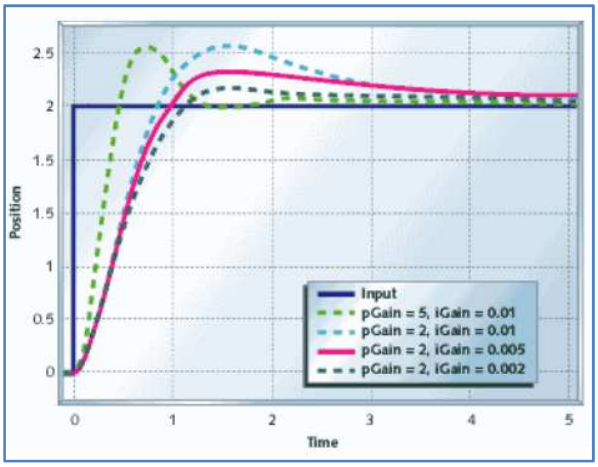


FIGURE 90. PROPORTIONAL AND INTEGRAL CONTROL

Figure 91 shows the response of the system with proportional and derivative (PD) control. This system settles in less than 1/2 of a second, compared to multiple seconds for the other systems.

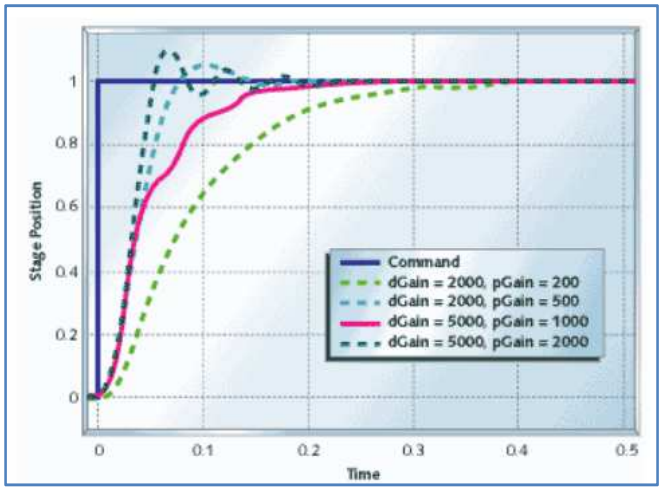


FIGURE 91. PROPORTIONAL AND DERIVATIVE CONTROL

The key in order to have a good control is to tune the three constants with a good enough relation between maximum overshoot and time before the system is stable. An example is shown in the below figure:

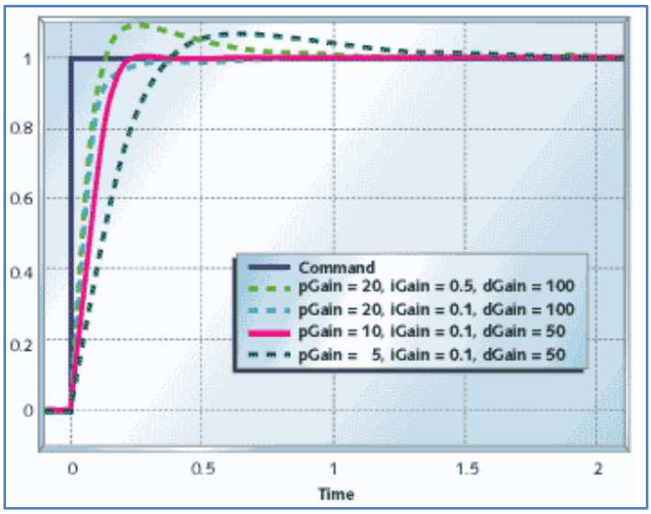


FIGURE 92. PID CONTROL LOOP

CURRENT LOOP

The current loop is the one designated to, given a current command, coming from the position loop , the necessary duty cycle in order to accomplish the current commanded from the position loop. The inputs of the loop are: the current limit, current command, current feedback, PI gains and output range (maximum and minimum speed).

The first step is to coerce the current command if this is greater than the maximum current specified from the real-time target. The output from the coercing module is the setpoint from the PI module. The feedback of the PI (current in this case) is formed by a multiplexer that decides which sensor current to use. If the current is greater than the number specified by the

user in the real-time, the shunt sensor is used, on the opposite case, the NI9505 is used. By default the value to choose between one or another is 150 mA.

The other two inputs of the PI block are the values of the proportional and integral gain, and lastly the maximum speed assigned by the user.

The output from the PI is the duty cycle that is sent to the H-Bridge. It goes inside a timed loop because the controller used to generate the PWM to the motor has a minimum pulse width (high or low) requirement of 2us. With 20 kHz switching which the above example shows, this results in a usable duty cycle range of 4% to 96%. However, it can generate duty cycles of 100% and 0%. If the commanded duty cycle is >96%, it is coerced to 100%, <4% is coerced to 0%.

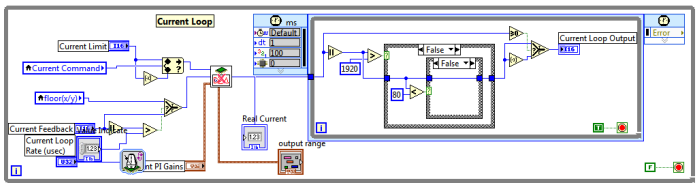


FIGURE 93.CURRENT LOOP CODE

3.2.1.4 POSITION LOOP

The position loop contains the position PID, the loop that has to bring and maintain the position setpoint established by the user in the Real part time. The feedback of the position comes with the variable “encoder position” from the encoder loop. The proportional and derivative gains are included in the “gains” cluster, that can be designed from the real-time part. From the input, the feedback and the gains, the output of the PID is determined, in this case, the current (in mA), which is used in the current loop. The other output is the error between the actual position and the position setpoint.

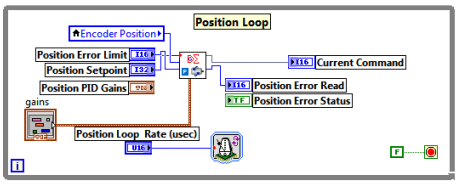


FIGURE 94.POSITION LOOP CODE

3.2.1.5 MOTOR BEHAVIOR

Different tests have been undertaken using the same tuning configuration, different speeds and loads, in order to establish the performance of the system in different situations, by examining the reaction in a change of setpoint of 2000 degrees.

In the first configuration, no load is added to the shaft:

NO LOAD

On the first case, no load is added to the shaft:

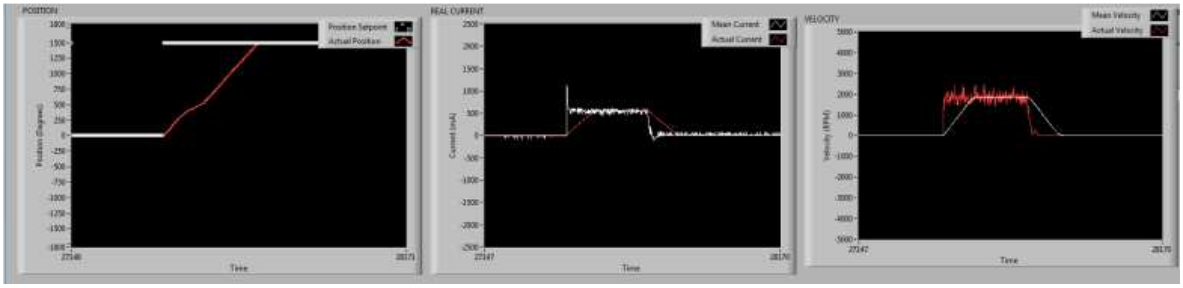


FIGURE 95.MAXIMUM DUTY CYCLE=25%, NO LOAD

As can be seen, the position is reached in 4.3 seconds. There is no overshoot. The initial current spike is of 400mA, approximately, and the average current consumption is 630mA.

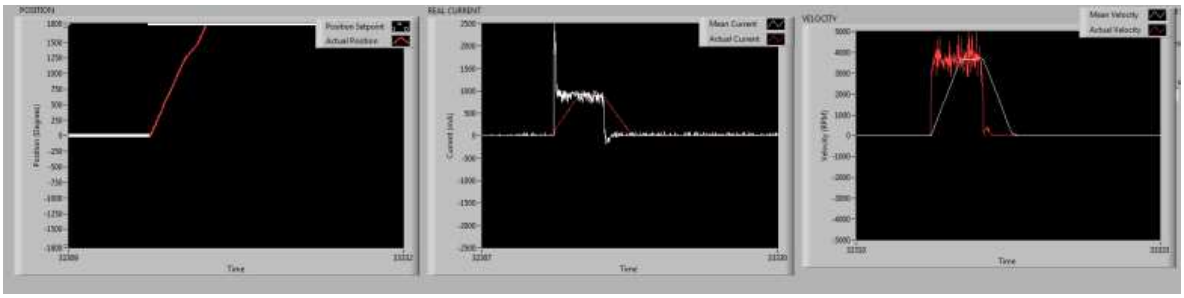


FIGURE 96.MAXIMUM DUTY CYCLE=50%, NO LOAD

Doubling the maximum duty cycle, makes the speed to be the double, making the position reaching almost double faster. On opposite to this speed, the average current consumption is 1000mA, and the spike goes to 2A. All this current consumption shows the price to pay in order to have a faster response.

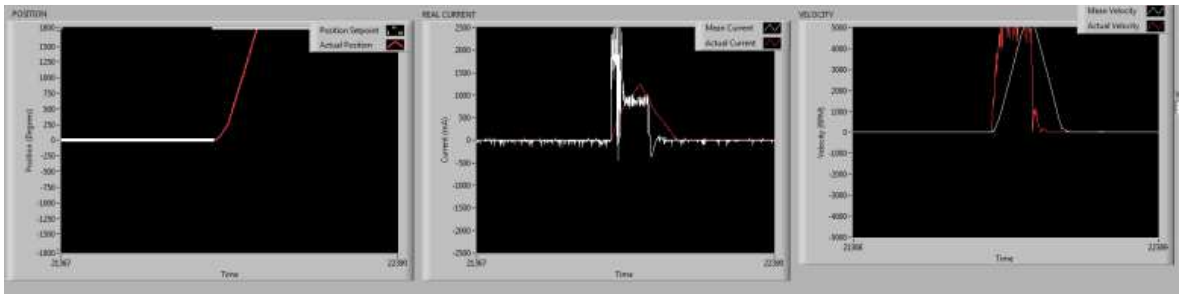


FIGURE 97.MAXIMUM DUTY CYCLE=100%, NO LOAD

With a speed of 5000RPM, the current spike goes to 2.5A, and the average current goes approximately also to 1000mA, so the current consumption is very similar to the previous case. On the above figure seems that it is much logical to go to a speed of 5000 instead of 4000 RPM as its current consumption is very similar, but what happens when an inversion of direction is done online? Let's analyze it by looking the two cases:

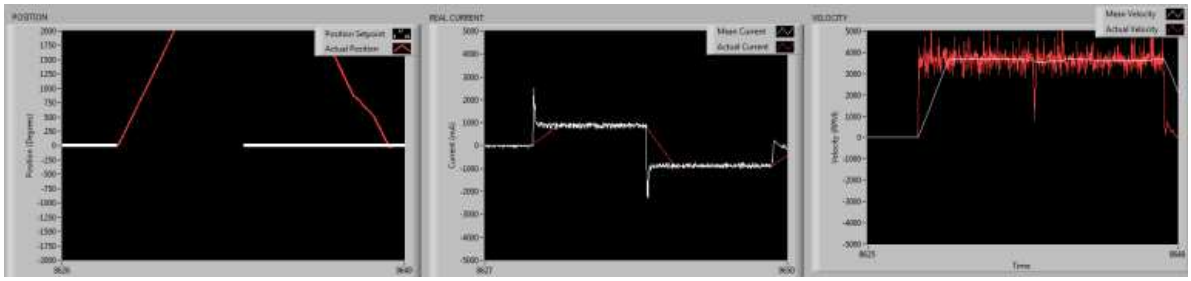


FIGURE 98.MAXIMUM DUTY CYCLE=50%, NO LOAD, INVERSION WHILE TURNING

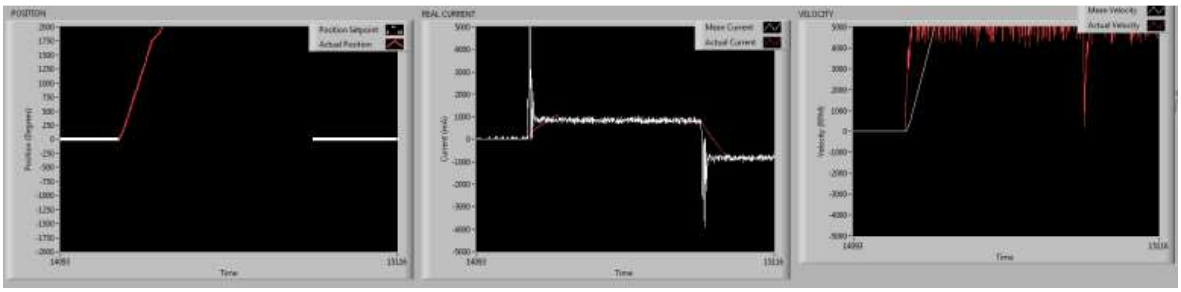


FIGURE 99.MAXIMUM DUTY CYCLE=100%, NO LOAD, INVERSION WHILE TURNING

The current spike in the first case is 5A, more than the double than with the slower one, reaching a maximum of 2.5A. So in this case the speed has a heavy price: almost double spike, causing a more power consuming system and a making it less robust as it can cause driver failure/disablement due to overheating or overcurrent.

0.02616Nm Torque

Now a small load is added to the shaft:

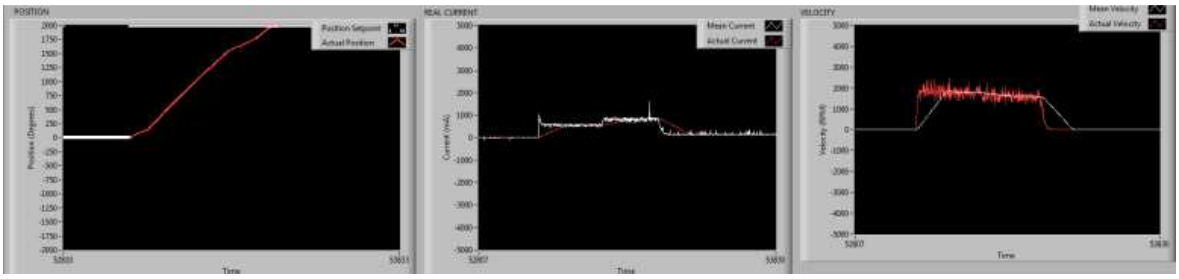


Figure 100.Maximum duty cycle=25%, 0.02616Nm torque

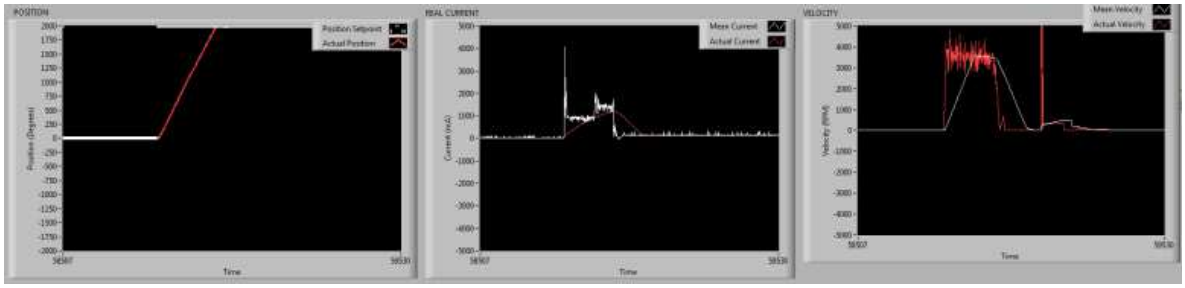


Figure 101.Maximum duty cycle=50%, 0.02616Nm torque

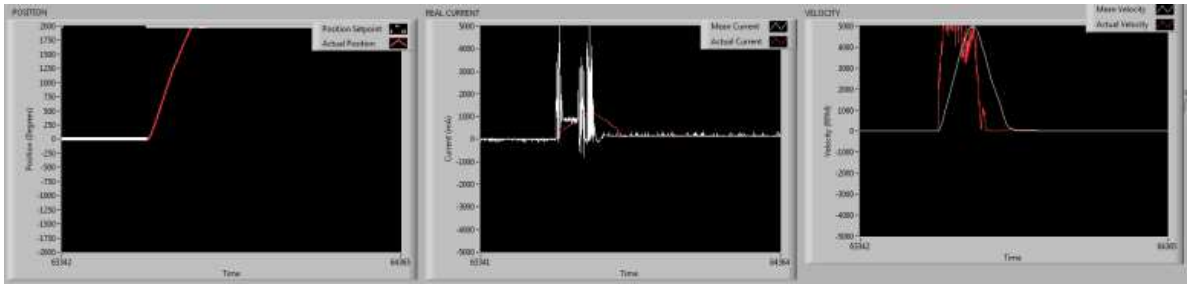


Figure 102.Maximum duty cycle=100%, 0.02616Nm torque

With a torque of 0,02616Nm in the motor shaft, it can be seen that in the slower case there is almost no spike, and the current is quite low, about 900mA, making the system slow but heavy reliable and light power consuming. When the speed is doubled, there is a strong spike reaching 4A and an average current of 1450mA. On the fastest case there are three spikes present, resulting in a lot of power wastage.

With some load, maintaining the system slow makes a lot of difference. It is also has to be taken in consideration that from now on the simulations with maximum speed have been removed as they were causing the system to disable the drivers due to overheating present in the motor , making the tests impossible to be taken.

On the other hand, there is also a lot of difference when the direction of the motor is changed while rotating on the other one:

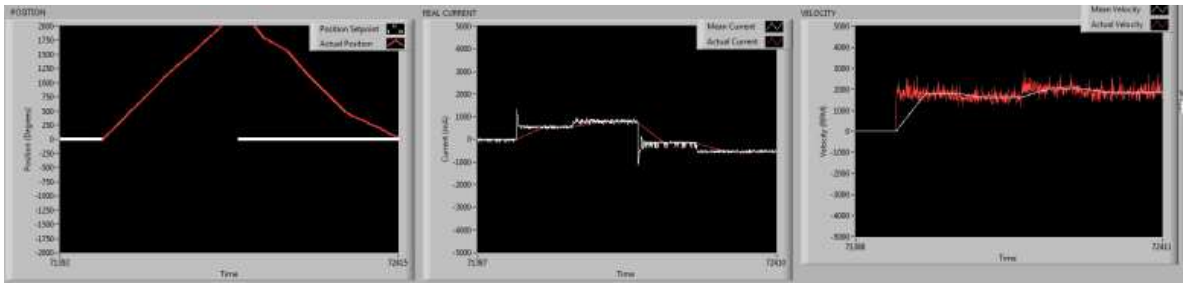


FIGURE 103.MAXIMUM DUTY CYCLE=25%,INVERSION WHILE TURNING

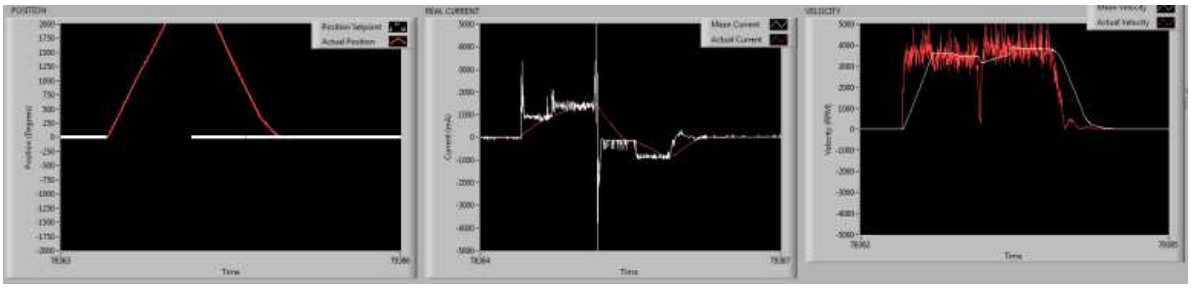


FIGURE 104.MAXIMUM DUTY CYCLE=50%, INVERSION WHILE TURNING

0,01308NM TORQUE

The 0.01308Nm torque simulates the worst possible scenario that can be present on the kite. As expected, the simulation goes in the same line as the ones before, showing that the power consumption is much higher in the faster cases. Also has to be noticed that in 0.01308Nm cases, the speed is not maintained constant, as the loop is based on current control, and as the motor needs a much higher torque, some speed is sacrificed especially in the 4000 RPM case:

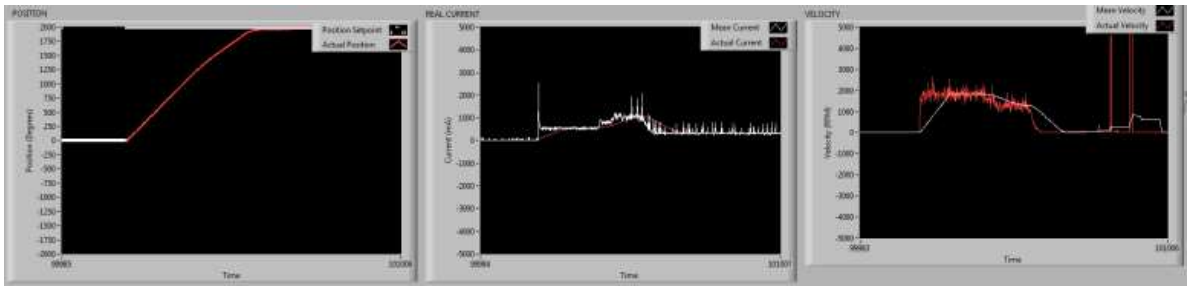


FIGURE 105.MAXIMUM DUTY CYCLE=25%

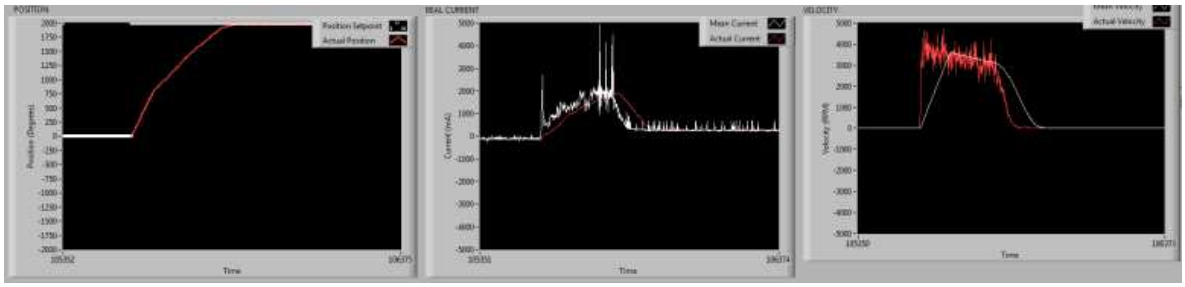


FIGURE 106MAXIMUM DUTY CYCLE=50%

Notice that now there is also a big starting spike for the 2000RPM case.

If the inversion case is analyzed, it turns out that the reverse spikes are more important in this case:

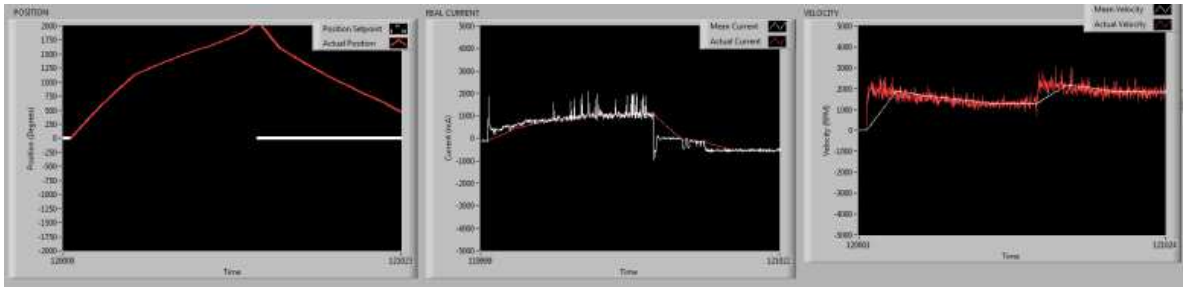


FIGURE 107MAXIMUM DUTY CYCLE=25%, INVERSION WHILE TURNING

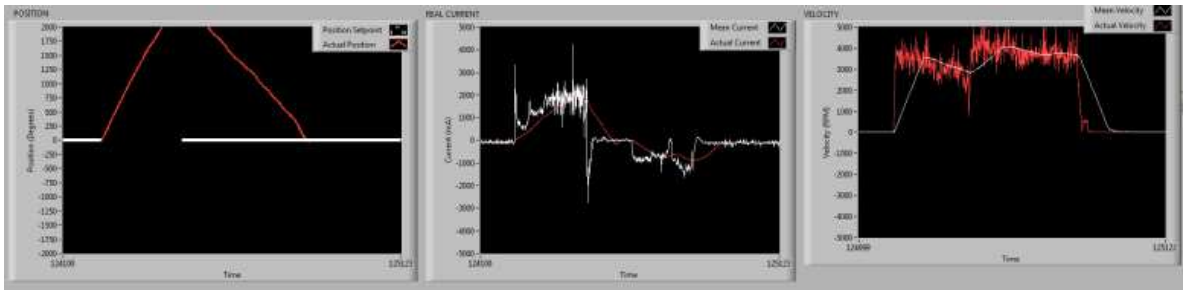


FIGURE 108MAXIMUM DUTY CYCLE=50%, INVERSION WHILE TURNING

This tests shows that the ideal performance for the system is a maximum speed of 2000RPM, as it can reach the position in enough time for this application, and there are almost no spikes in any situation, making the system very durable and reliable, but if more speed is needed it is also noted that a maximum speed of 4000RPM can be achieved only losing some durability, but maintaining almost the same robustness. A maximum speed of 5000RPM is not suggested as there is a gain 1/5 of speed , but the system loses reliability to the point of making the drivers disable making the system impossible to control, as the motors can't be controlled during a small period of time(until the drivers are re-enabled , when the disabling event has disappeared).

3.2.2 REAL-TIME PART

The strategy for tuning the PID has been successfully implemented by executing the following procedure:

The proportional gain is set to 1, and the derivative and integral to zero. As with this value the system is unstable, the proportional value is decreased until the value is stable and noiseless at 0.05.

Now the differential value is set to 100 times the proportional value: 5. With this value , there is a lot of oscillation in the position control , so the derivative value is increased until the relation between oscillation and noise is optimum , reached at kd=10.2.

The integral gain is set to 1, and as the system is slightly oscillating, the value is reduced to 0.01, for a better performance.

With the specified tuning the response of the motor is relative slow, reaching the steady state in 0.5 seconds. This feature can seem to be a problem, but for the application being implemented is better to avoid having big spikes, so the system is more durable and more unlikely to have an error due to overcurrent or overvoltage.

3.3 WIFI CONNECTION

In order to establish connection between the air pod and the ground pod some light software issues must be taken in consideration.

First of all, the two SbRIO need to be on the same subnet, in order to be able to interact and appear on the same project, as stated on the following figure:

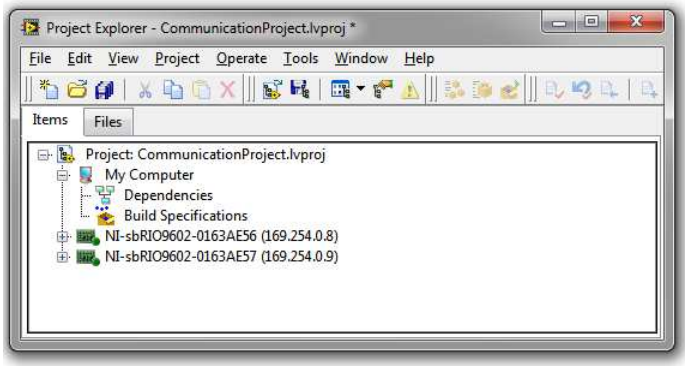


FIGURE 109.PROJECT'S EXPLORER WINDOW

The communication between the two modules, as the ground pod is another phase of the project and has not been started yet, an example of how that communication will occur is shown. The example consists in the sending of a Boolean bit and an integer value from the air pod to the Ground SbRIO. The simple codes of the FGPA modules for both SbRIOs can be seen below:

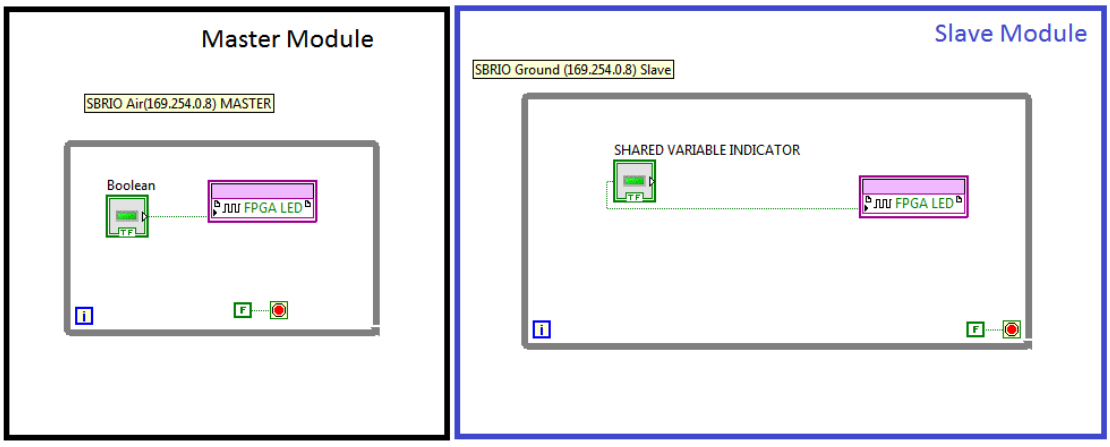


FIGURE 110.AIR AND GROUND POD, FPGA MODULE

In order to be able to send the data, is necessary to transmit the data from the FPGA module to the real-time module, because we need to use a shared variable to make the information visible to the other part, and this is only possible in the real-time interface. So the master module opens its FPGA reference, gets the Boolean value and feeds the “Boolean Variable”, configured as an input as it is the sending part. The integer value is inputted from the user interface, and feeds a shared variable called “GlobalInteger”, also configured as an input.

On the slave module, the shared variables are configured as outputs, correctly showing the value coming from the air pod on its user interface. The code of the real-times modules is shown:

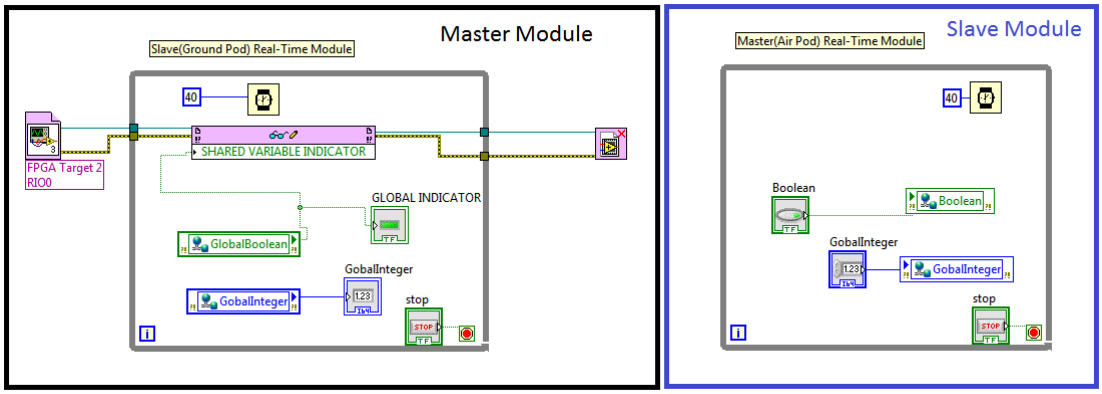


FIGURE 111.AIR AND GROUND POD, REAL-TIME MODULE

3.4 BATTERY CONTROL

The battery control is implemented in a similar way as the current sensor, as can be shown below:

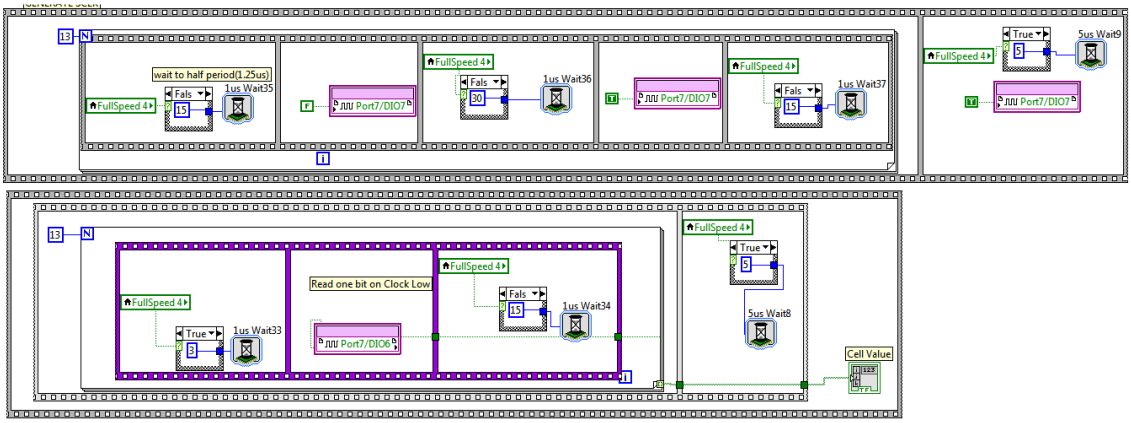


FIGURE 112.ADC LABVIEW CODE

The value obtained from the ADC is gathered in the "cell value" variable, which can be obtained with a configurable frequency. The maximum cell value is 4.2V, so a 5V reference is needed. The output from the ADC is the following:

$$Digital\ Output\ Code = \frac{1024 \cdot Vin}{5V}$$

Translating into:

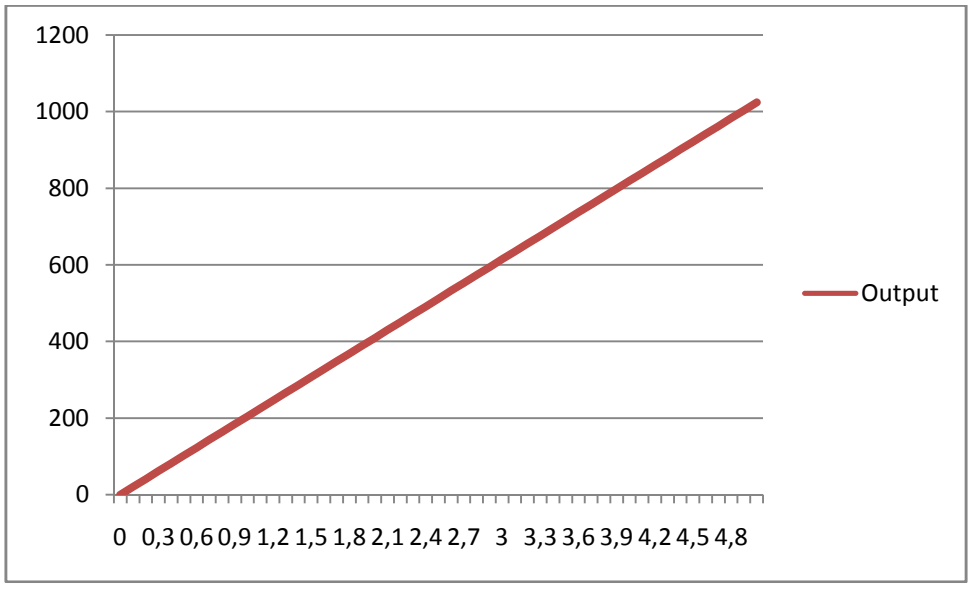


FIGURE 113.ADC OUTPUT - EQUIVALENT CELL CHARGE

So the resolution of the current sense is:

$$LSB = \frac{Vref}{2^{10}} = \frac{5}{1024} = 4.88\,mV$$

In order to convert the obtained value into a readable number, in the real-time module the following code is added:

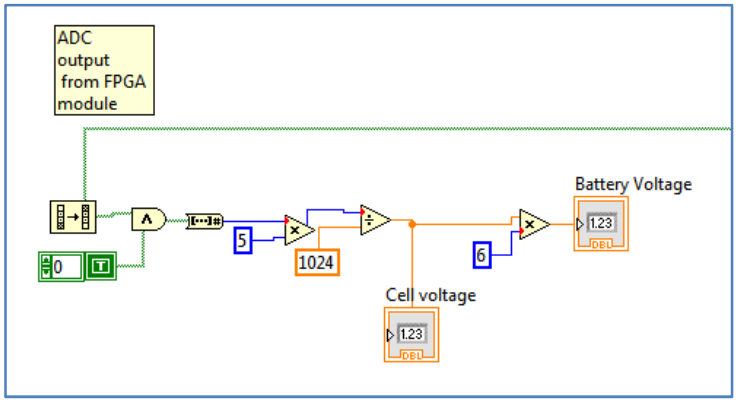


FIGURE 114.ADC REAL-TIME MODULE CONVERSION

The ADC variable that comes from the FPGA module is a Boolean string, first of only the first 10 less significant bits are used, the others are deleted. Then the string is rotated. After that the previous stated formula is implemented, and on the last instance, multiplied by 6 in order to get the battery voltage.

3.5 GPS

As has been previously stated, the conjunction of LabVIEW and SbRIO allow a fast development tool. This is particularly true with the GPS, as there are some modules ready-to-go, as the one used in this project, the code which is shown below:

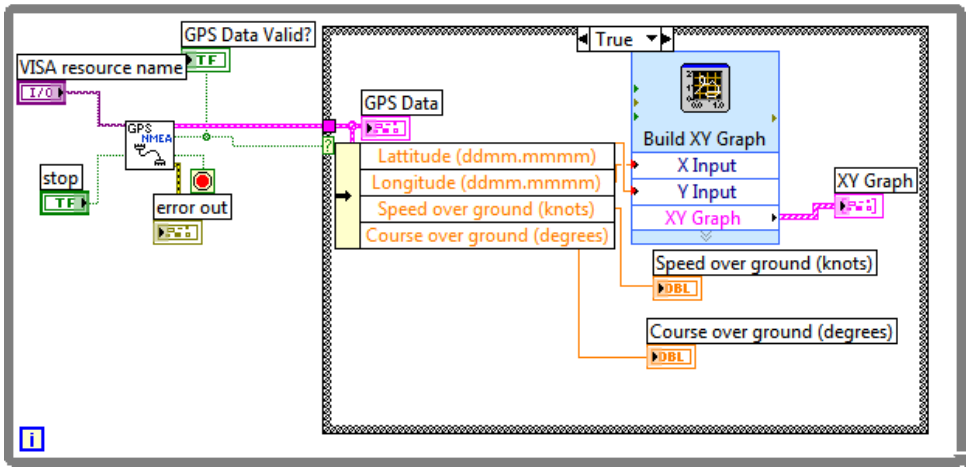


FIGURE 115.GPS CODE

The first step is to establish the serial communication between the FPGA module and the serial protocol of the GPS, as explained on the GPS hardware section. The next phase is to detect if the data is valid, this is done by checking if the valid data is true in the GPRMC sentence, the one that is used to indicate the position, speed and magnetic variation. If the data is in effect valid, then the latitude, longitude, speed over ground and course over ground is parsed from GPS data, by decoding the GPRMC sentence, stated below:

Structure:

$GPRMC,hhmmss.sss,A,dddmm.mmmm,a,dddmm.mmmm,a,x.x,x.x,ddmmyy,x.x

Field	NaME	Example	Description
1	UTC time	092204.999	UTC time in hhmmss.sss format (000000.00 ~ 235959.999)
2	Status	A	Status 'V' = Navigation receiver warning 'A' = Data Valid
3	Latitude	4250.5589	Latitude in dddmm.mmmm format Leading zeros transmitted
4	N/S indicator	S	Latitude hemisphere indicator 'N' = North 'S' = South
5	Longitude	14718.5084	Longitude in dddmm.mmmm format Leading zeros transmitted
6	E/W Indicator	E	Longitude hemisphere indicator 'E' = East 'W' = West
7	Speed over ground	000.0	Speed over ground in knots (000.0 ~ 999.9)
8	Course over ground	000.0	Course over ground in degrees (000.0 ~ 359.9)
9	UTC Date	211200	UTC date of position fix, ddmmyy format
10	Magnetic variation		Magnetic variation in degrees (000.0 ~ 180.0)
11	Magnetic Variation		Magnetic variation direction 'E' = East 'W' = West
12	Mode indicator	A	Mode indicator 'N' = Data not valid 'A' = Autonomous mode 'D' = Differential mode 'E' = Estimated (dead reckoning) mode 'M' = Manual input mode 'S' = Simulator mode
13	checksum	25	

FIGURE 116.GPRMC TABLE

Once the data is parsed correctly, it is inserted into a graph to indicate the previously stated information. The user interface is shown below:

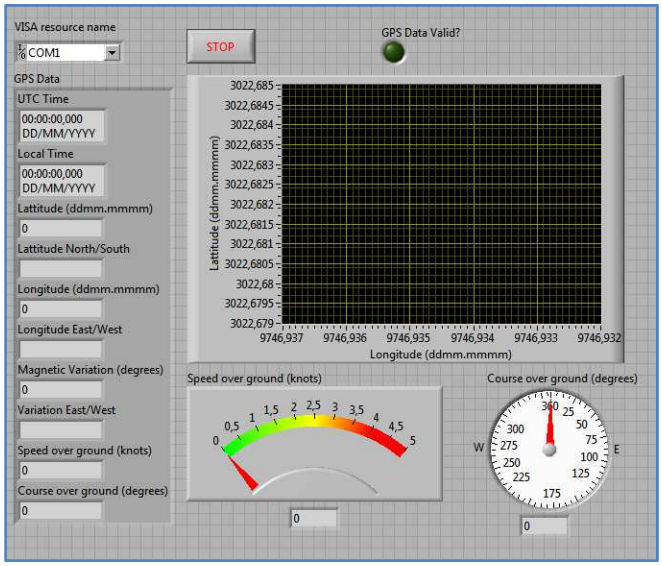


FIGURE 117.GPS USER INTERFACE

3.6 REAL-TIME PART

3.6.1 USER INTERFACE

The user interface has been designed to allow the user to control and be able to know all important information in order to make the pod do the designed function and keep safe at all times.

The control part has two main points: The control and regulation of the motor, and the speed control of the SPI sensors.

The indicator part indicates all the information regarding position, accelerations of the IMU on the kite. Also indicates if any alarm is set within the IMU sensor, and finally shows the current drained, the position and the speed of the motors.
Has to be noted that the following version, has only one motor control as there is not enough space for a second one, so both motors must have exactly the same configuration, but as they are exactly the same motor version this is not a problem. The User interface is shown below:

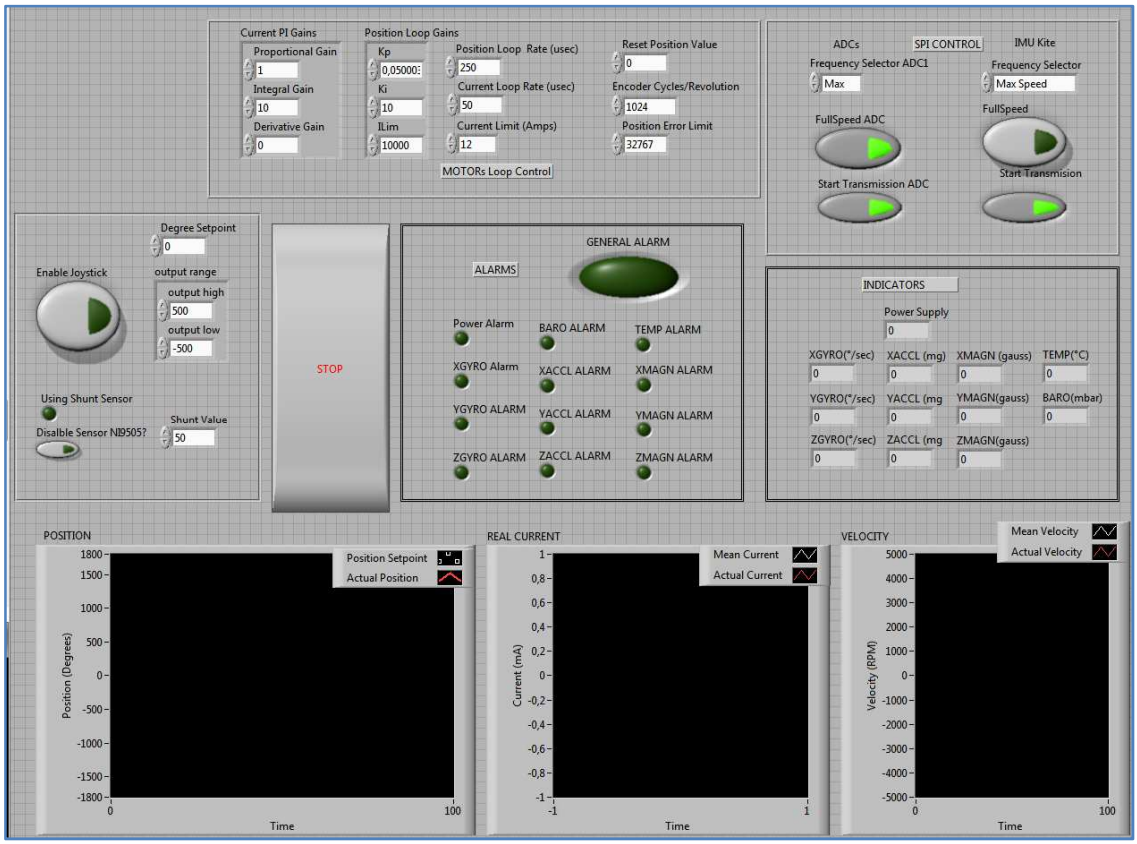


FIGURE 118.REAL-TIME PART, USER INTERFACE

There is a control part in the left which hasn't been commented. On it, there is a button to enable/disable the manual control of the motors. If it is enabled, it is controlled through a joystick, the control of it will be commented in the real-time implementation chapter.
The user interface regarding the joystick control is the following:

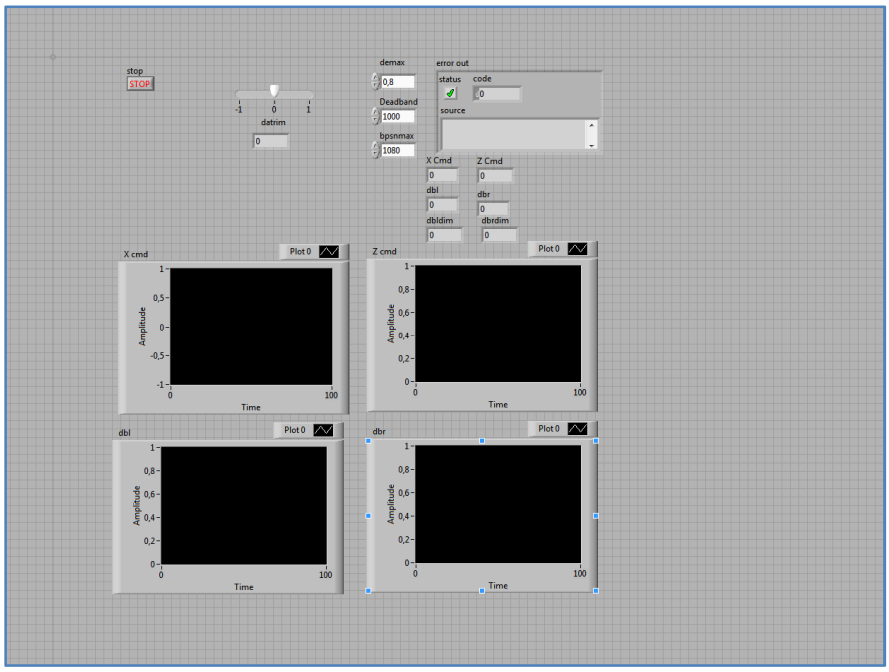


FIGURE 119.JOYSTICK CONTROL VI, USER INTERFACE

On it can be specified the amount of revolutions is done per grade of joystick movement, and also is shown the actual position in the x axis and in the z axis. The x axis movement controls left motor while the z axis controls the right motor.

Another important element for the correct visualization of the state of the airborne system is the orientation. It is the third and last user interface.

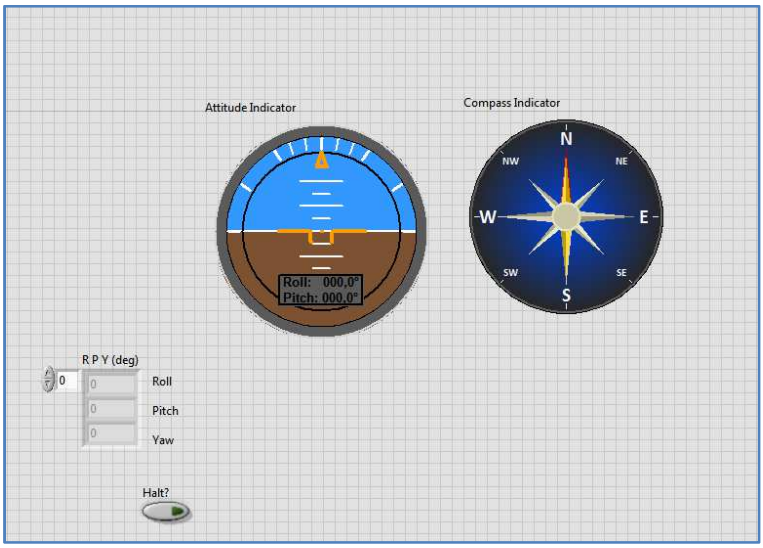


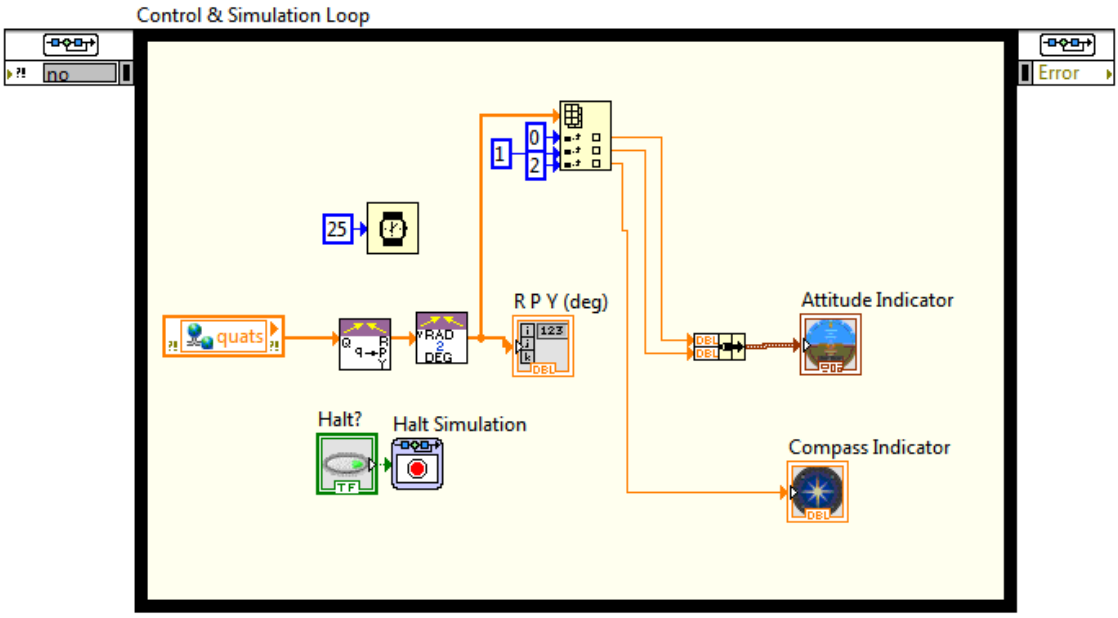
FIGURE 120.AIRBORNE ATTITUDE AND COMPASS INDICATOR, USER INTERFACE

It is based on an attitude indicator, and a compass. The first one indicates the position in reference to the ground of the IMU, and the second the orientation of it.

3.6.2 Real-Time Implementation

3.6.2.1 Attitude and compass indicator

In order to represent the position of the IMU, the angular rates from the gyro are converted to quaternions. These are integrated to give angles from angular rates. After the integration back to Euler angle, these are feed to the artificial horizontal which does the display automatically. The compass is fed with the magnetic fields captured by IMU's magnetometers. The code is shown below, and it is programmed to work in a computer:



3.6.2.2 General diagram

The first thing to do in the general real-time part is synchronize the real-time part with the FPGA module. In order to do this the FPGA target is initialized, and enable the drivers of the two motors, and wait until both are enabled. Once this is fulfilled, the code continues.

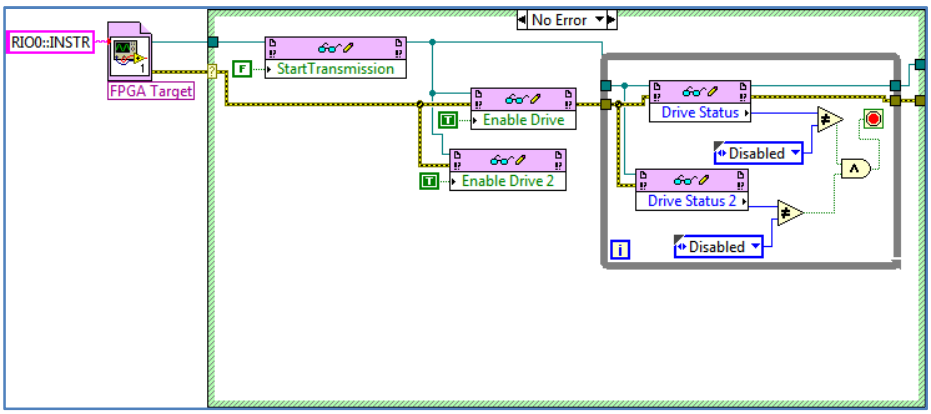


Figure 121.FPGA initialization and waiting until NI9505s drivers are enabled

Below is the aspect of the general diagram:

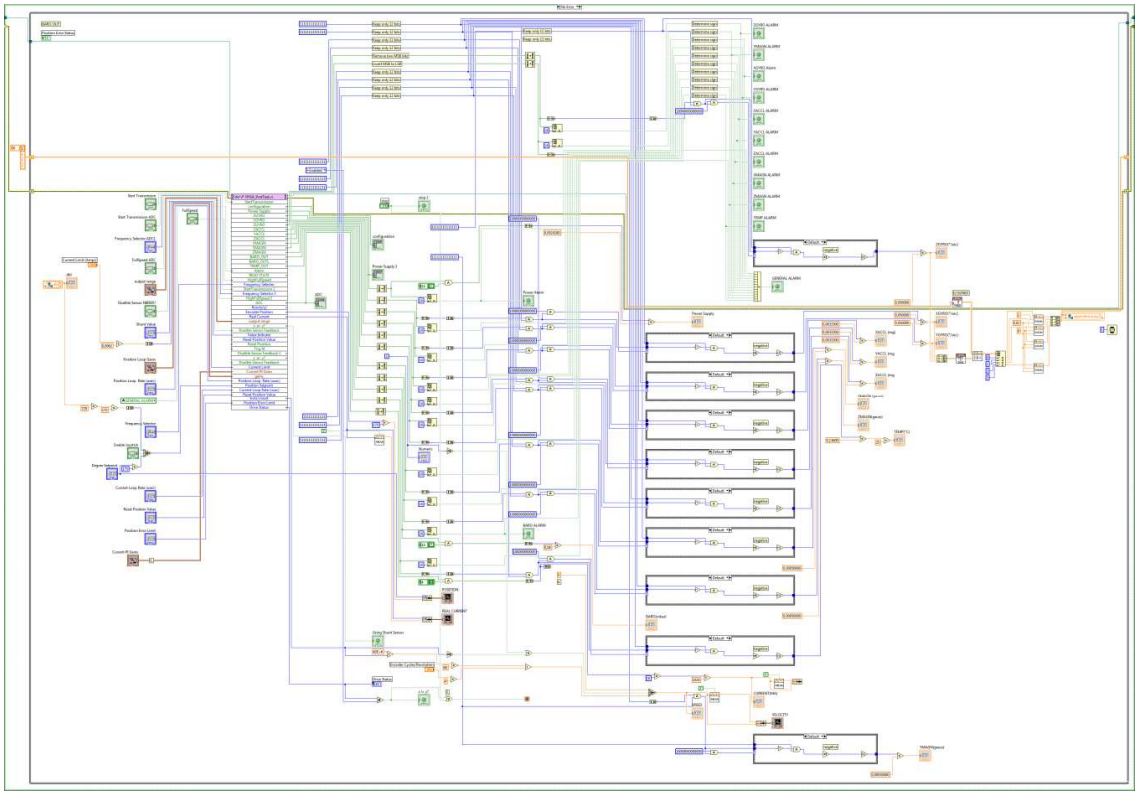


FIGURE 122.REAL-TIME CODE

On the left side there are all the necessary controls for the user interface part. This are connected to the FPGA part through the Read/Write FPGA control. If they have an arrow in the left they are an input to FPGA module, and if it is on the right they are an output.

On the upper right corner there are all alarms for the different registers of the IMU sensor, and on the down-left corner are all the necessary conversions of the IMU registers.

3.6.2.2.1 IMU CONVERSIONS

In order to convert the data coming from the IMU to a valid one understandable for the user, it is necessary to convert it. On the following table there is a list of the units coming from the ADIS 16407:

REGISTER	UNITS PER LSB
XGYRO_OUT	0.05°/sec
YGYRO_OUT	0.05°/sec
ZGYRO_OUT	0.05°/sec
XACCL_OUT	3.333 mg
YACCL_OUT	3.333 mg
ZACCL_OUT	3.333 mg
XMAGN_OUT	0.5 mgauss
YMAGN_OUT	0.5 mgauss
ZMAGN_OUT	0.5 mgauss
BARO_OUT	0.08 mbar
BARO_OUTL	0.0003125 mbar
TEMP_OUT	0.136°C
SUPPLY_OUT	2.418 mV

TABLE 13.UNIT TABLE OF IMU'S REGISTERS

As an example the conversion of the Xgyro register is shown:

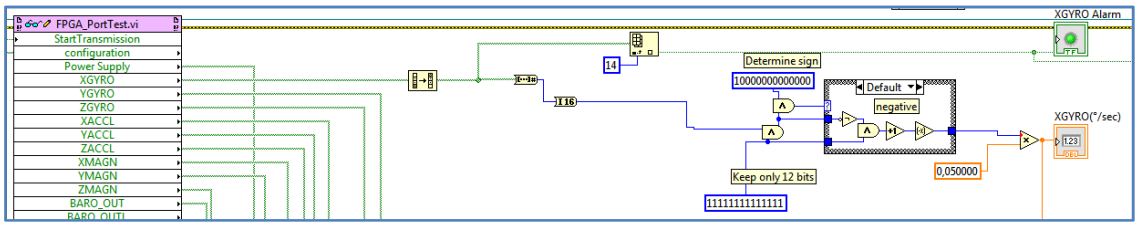


FIGURE 123. XGYRO REGISTER CONVERSION EXAMPLE

The first necessary step is to gather the information within the FPGA register through the Read/Write block in the left. The data coming out this block is a Boolean array, with the MSB at the right, just the opposite of what is needed, so the next step is to invert it. After that, the MSB is converted into the alarm bit. The information that is needed is in the 12 less significant bits, so the first two most significant bits are deleted, and the sign is determined based on the value of the 12th bit.

As has been previously stated, the codification is twos complement. For this reason what is done is to invert the value, and add one to the result. After this decoding is completed, the resulting value is multiplied by 0.5 in this case, based on the values stated on the previous table (value conversion), to get the final value, the one shown on the user interface.

3.6.2.2.2 GRAPHS

In order to visualize the parameters of the motors, the inclusion of graphs has been done. They respectively show the position, current and velocity of the motors (of the left motor in this case, to allow an easier comprehension of the graphs).

The first one to be shown is the position graph:

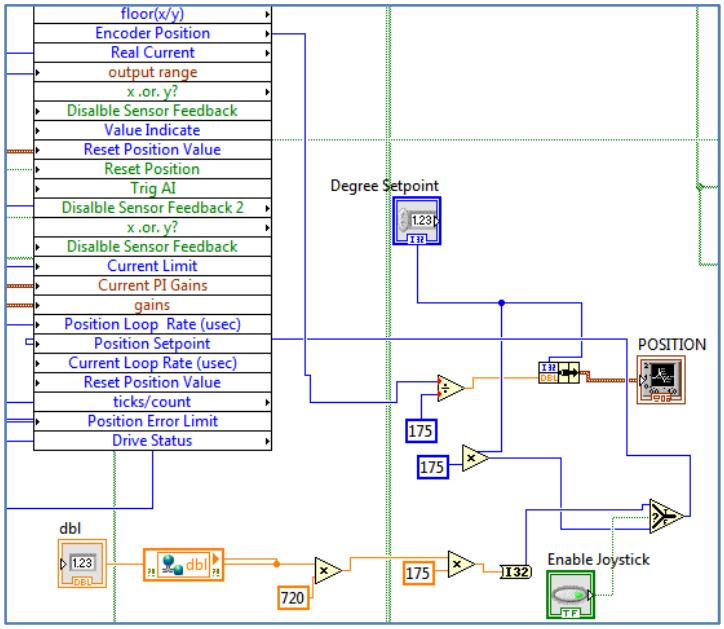


FIGURE 124.POSITION GRAPH CODE

The encoder position is divided by 175 to convert it to degrees, and it is one of the two entries of the graph, and it indicates the actual position. The other entry corresponds to the setpoint. This setpoint can be given by the user interface or by the joystick, depending if the joystick is enabled in the user interface. If the joystick is disabled, the setpoint comes from the Degree Setpoint from the user interface. On the opposite case, the value comes from the joystick VI, through a shared variable. The value of it is multiplied by 720 (on later versions this comes pre-configured from the joystick VI), and it is the amount of dimensional movement caused by each joystick non-dimensional, normalized movement (0 to 1). The value is then multiplied by 175 to convert it to degrees.

The real current graph is easier than the previous one, as it is just the value coming from the FPGA part (explained on the hardware part), and there is also an average block to have a "softer" representation:

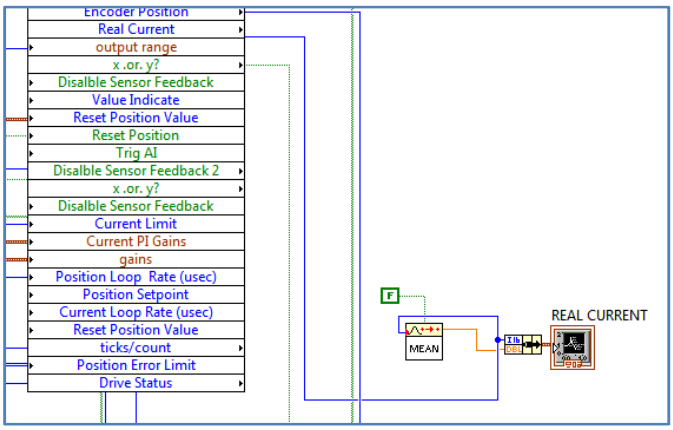


FIGURE 125. CURRENT GRAPH CODE

Finally, the speed graph is shown below:

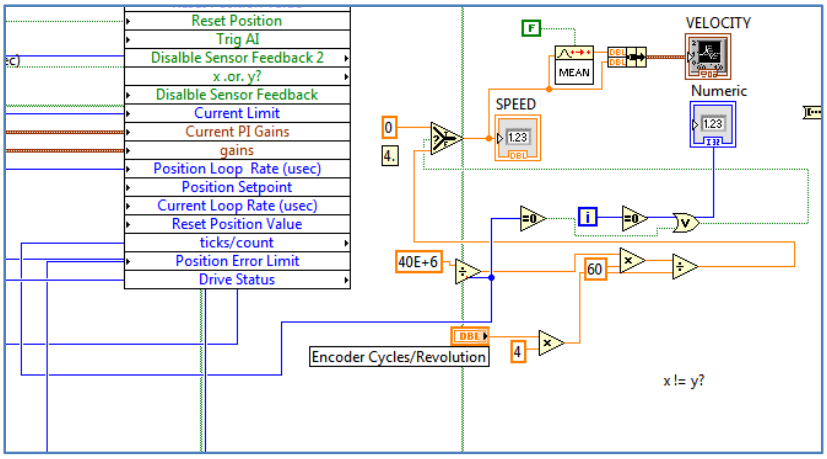


FIGURE 126.SPEED GRAPH CODE

The calculation of the speed is based on the amount of ticks detected by the encoder per cycle. As the clock frequency is 40MHz from the FPGA , the amount of ticks is divided by 40M and the result of it is multiplied by the type of encoder present (1024 ticks per cycle in this

case) resulting on the value of the actual speed. The other input of the graph is the average of the speed value.

4.FUTURE WORK

The prototype still needs a few further steps in order to be able to produce wind energy at high altitudes. The most important one is to finish the development of a flying protocol that will allow the system to be tested under real world conditions, allowing tests of the stability and robustness of the system. Some minor modifications are waiting to be developed, and are summarized by the following points:

- Finish development of a flying protocol
- Test the stability of the system under real world situation
- Develop a safety protocol when the ground pod loses communication with the air pod, so the pod lands avoiding harms to the system
- Develop a safety protocol when one of the motors stop working
- Change the brushed motors to a brushless ones to allow a more durable and precise system
- Optimize power consumption of the motors, by developing a more complex speed protocol to allow lower speeds when the change of position does not require instant modification, translating into less power consumption
- Optimize control, and sensing part, for data gathering only when needed, making the system more durable and saving memory resources
- Exchange the WIFI router to a ZigBee system that will allow lower power consumptions

5.CONCLUSIONS

A complete selection of the electronic components has been completed, and also the required control of the pod, either manual or automatic using sensor's feedback.

Some in-lab tests have been run, and also one outdoor field trial, demonstrating the proper operation of the system. In the below is shown a picture of the box used to run the tests.

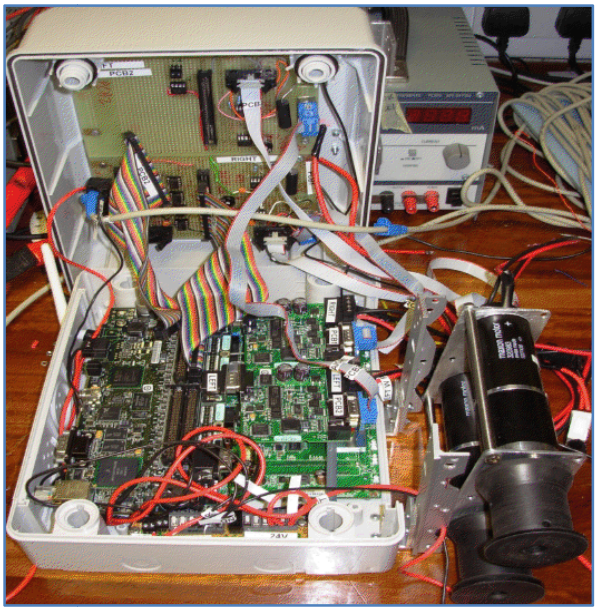


FIGURE 127.TEST BOX

The outdoors tests show limited results, at this point only a proper control of the motors can be achieved, but a test of flying trajectories could not be undertaken, as the automatic flying algorithm is still in development. Illustrated in the figure below is a recent field trial of the control pod as developed in this project. In this test the kite is flown by a human pilot using joystick inputs to control the servo motors which in turn control the position of the kite.

FIGURE 128.OUTDOORS TEST

Wind energy is the most important renewable energy resource in Ireland, based in the characteristics of the island, as has been stated in many studies by the SEAI(Sustainable Energy Authority of Ireland), as the one stated below:

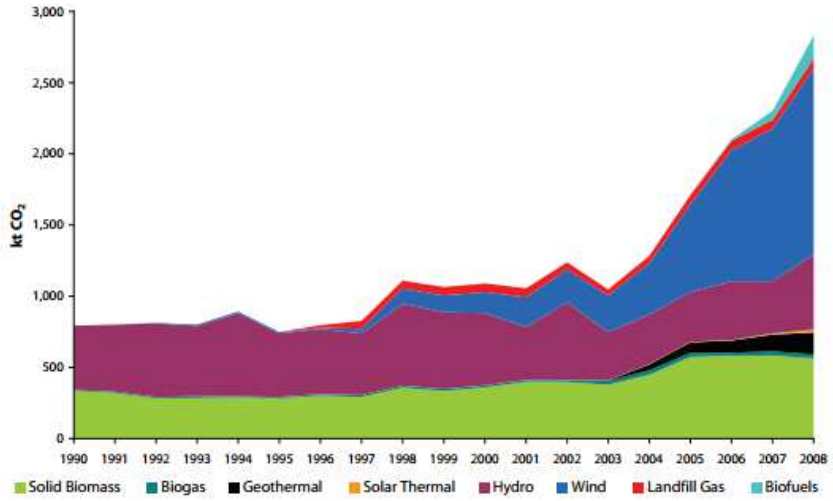


FIGURE 129.AVOIDED CO2 FROM RENEWABLE ENERGY 1990 TO 2008

SEAI, RENEWABLE ENERRGY IN IRELAND 2010 UPDATE, 18TH *APRIL 2012*
<HTTP://WWW.SEAI.IE/PUBLICATIONS/STATISTICS_PUBLICATIONS/SEI_RENEWABLE_ENERGY_2010_UPDATE/RE_IN_IRE_2010UPDATE.PDF>

So all the possible improvements on this field, and different ways of achievement of wind farming at high altitudes, as the reserves of oil are continually depleted. (Hubbert,2010).

The last figure shows the final appearance of the project:

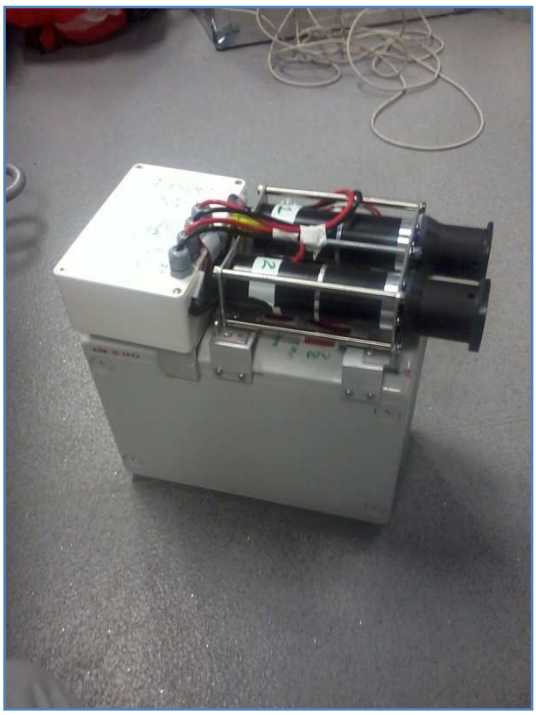


FIGURE 130.MOUNTED POD

6.REFERENCES

Proakis and Salehi, 'Contemporary Communication Systems using Matlab',Ed.Prentice Hall, 131-138

A.Maini, 'Digital Electronics Principles, Devices and Applications',Ed.Wiley, 47-53

P.Peebles, 'Digital Communication Systems',Ed.Prentice Hall, 17-49

T.Floyd and D.Buchla, 'Fundamentals of Analog Circuits',Ed.Prentice Hall, 598-614

R.Larsen, 'LabVIEW for Engineers',Ed.Source, 26-50,103-120,142-163,183-191,216-250, 352-373

L.Clark, 'LabVIEW Digital Sinal Processing',Ed.E.Resources, 127-133

R.Bitter,T.Mohiuddin,M.Nawrocki, 'LabVIEW Advanced Programming Techniques',Ed.CRC, 24-53,144-151,192-195,299-317

RChelsite,2008, 'LiPo Battery Charging & Safety Guide'.[online]
Available at:
<http://www.rchelisite.com/lipo_battery_charging_and_safety_guide.php>
[Accessed 23rd March 2012]

Radiolabs , 'Calculating your 802.11 power output'[online]
Available at:
http://www.radiolabs.com/stations/wifi_calc.html
[Accessed 15th November 2011]

Canale, M., Fagiano, L., Milanese, M., (2010), 'High Altitude Wind Energy Generation Using Controlled Power Kites', IEEE Transactions on Control Systems Technology, Vol. 18 (No. 2), 279-293.

O'Gairbhith, C. (2009), 'Assessing the Viability of High Altitude Wind Resources in Ireland' [online], available at: http://www.carbontracking.com/reports/High_Altitude_Wind_Resource_in_Ireland.pdf [accessed 01 October 2011].

J. Coleman , (2011),'Flight Control of a Tethered Parafoil for Airborne Wind Energy Generation', 7-10,28-34.

T.Kugelstadt,(2011),'Extending the SPI bus for long-distance communication'.[online]

Available at:
http://www.ti.com/lit/an/slyt441/slyt441.pdf [accessed 23 October 2011]

Dynetic Systems, 'Brushless vs Brushed'[online], available at:
http://www.dynetic.com/brushless%20vs%20brushed.htm [accessed 19 November 2011]

Notre Dame control and robotics, 'RS-232 Serial Protocol'[online], available at:
http://patents.ame.nd.edu/microcontroller/main/node24.html [accessed 12 January 2012]

M.Chivers, 'Differential GPS'[online], available at:
http://www.esri.com/news/arcuser/0103/differential1of2.html [accessed 15 January 2012]

National Instruments, Why to choose CompactRIO?,[online],available at:
http://www.ni.com/compactrio/whychoose/ [accessed 11 October 2011]

National instruments, LabVIEW with crio tutorial,[online] available at:
http://catsfs.rpi.edu/~wenj/ECSE446S06/LabViewcRIOTutorial.pdf?q=~wenj/ECSE446S06/LabViewcRIOTutorial.pdf accessed 1 [accessed 11 October 2011]
National Instruments, NI9505 Operating instructions,online],available at:
http://www.ni.com/pdf/manuals/374211f.pdf [accessed 11 October 2011]

National Instruments, Current Reading on NI9505,[online],available at:
http://digital.ni.com/public.nsf/allkb/812D5CAD54BAF478862576C7007D6C83 [accessed 18 October 2011]

Jartiuch Wordpress, NiMG, Li-Ion or Li-polymer?,[online],available at:
http://jartiuch.wordpress.com/2007/11/01/nimh-or-li-ion-or-li-polymer-what-to-consider%E2%80%A6/ [accessed 13 October 2011]

T.Wescott, PID without PhD, [online],available at:
http://igor.chudov.com/manuals/Servo-Tuning/PID-without-a-PhD.pdf
[accessed 19 December 2011]

R.Erickson,D.Maksimovic, 'Fundamentals of Power Electronics, Second Edition',Ed.Springer, 131-170.

R.Malik, 'Electronic Design',Ed.Prentice Hall, 867-879

APPENDIX

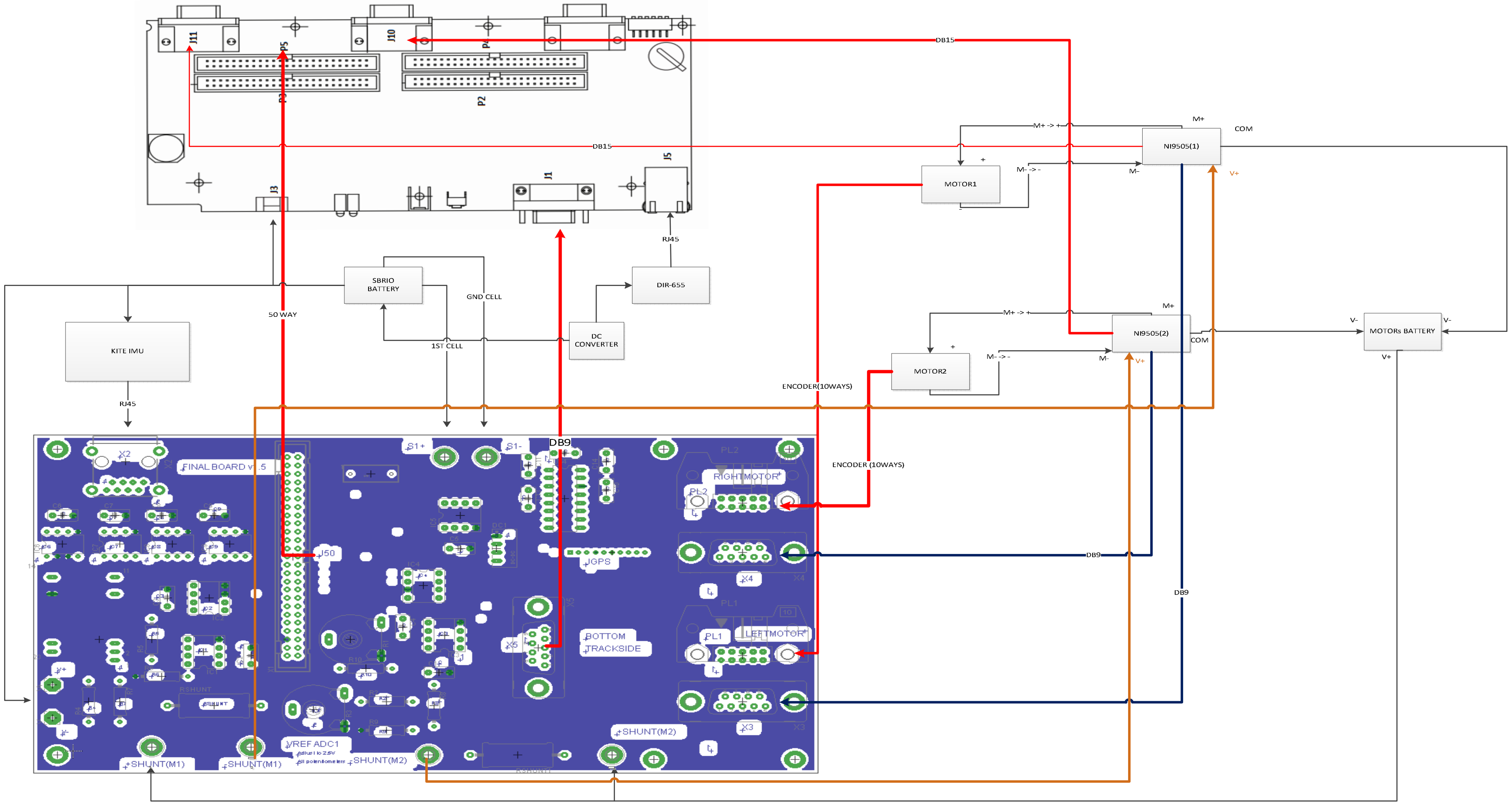


Figure 131. Final board connection diagram

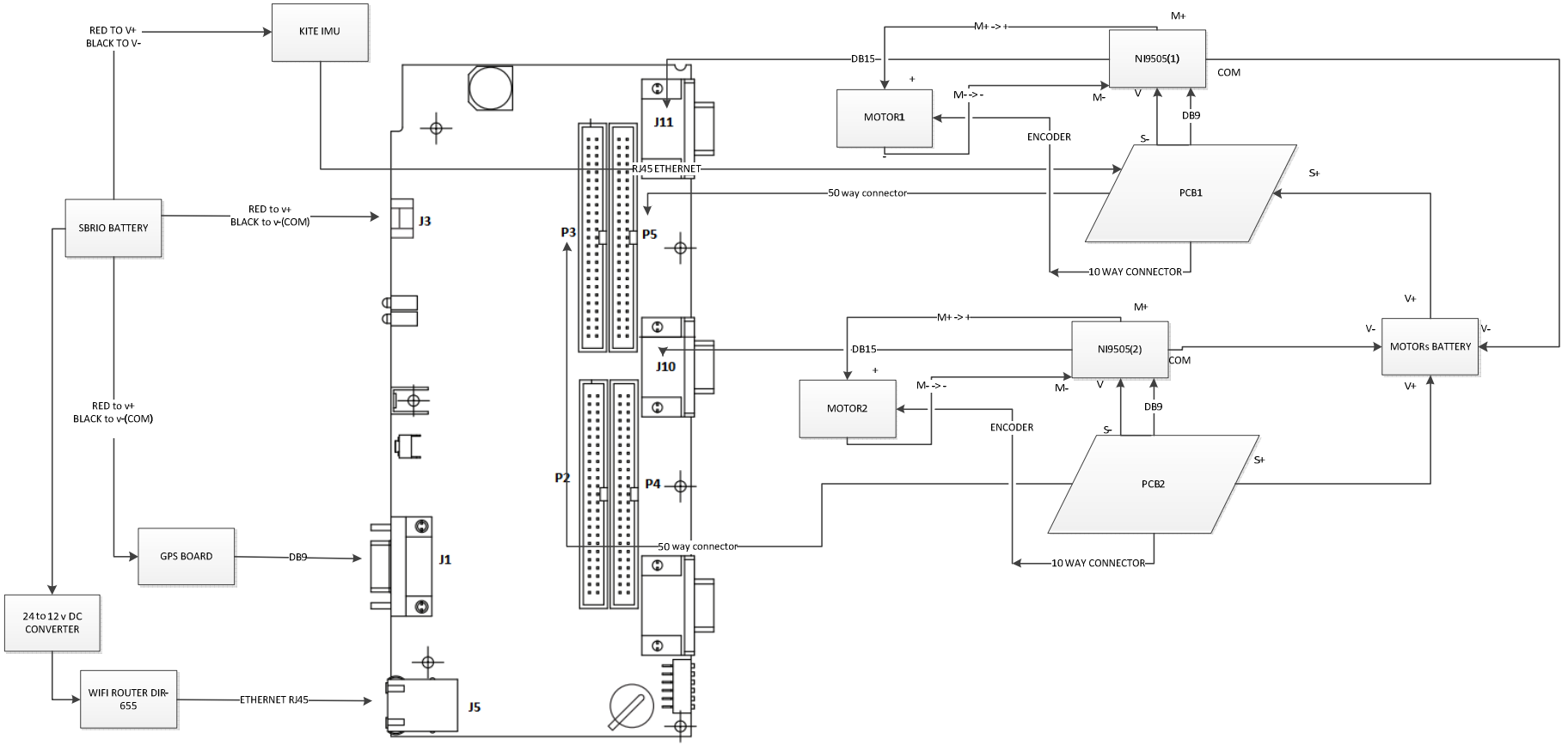
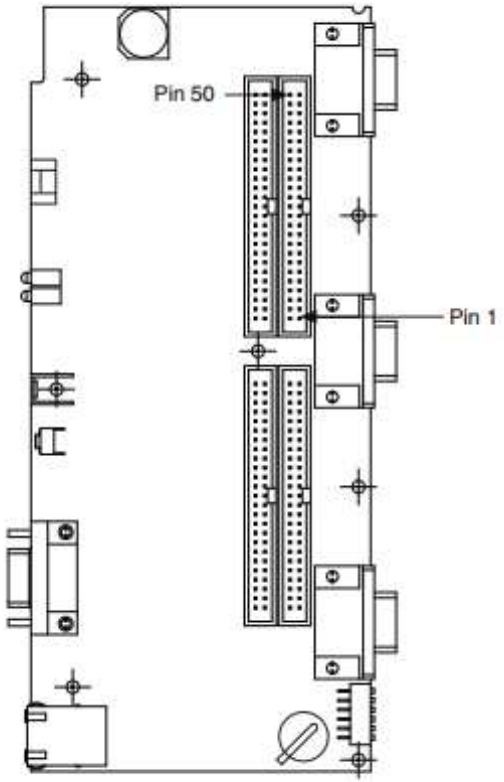
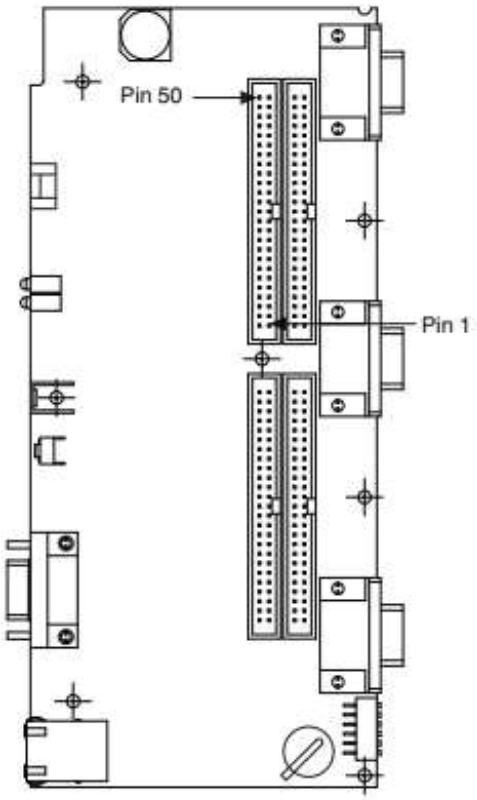


FIGURE 132.TWO BOARDS VERSION CONNECTION DIAGRAM



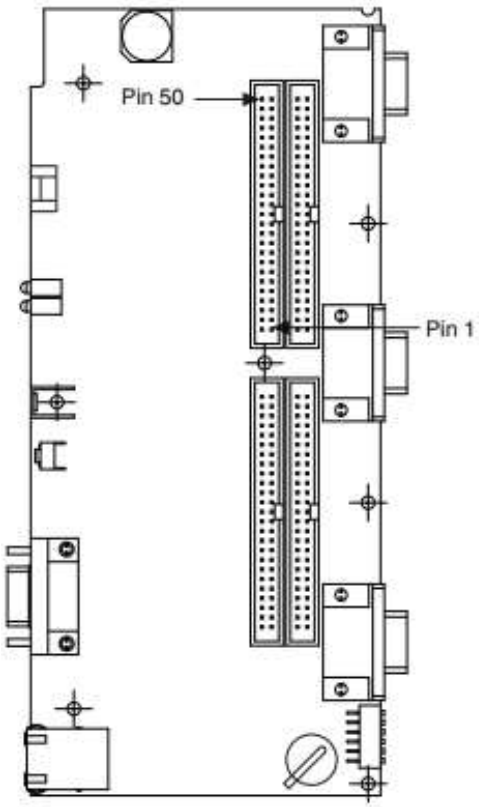
D GND	50	49	Port4/DIO8		
5 V	48	47	Port4/DIO7		
D GND	46	45	Port4/DIO6		
5 V	44	43	Port4/DIO5		
D GND	42	41	Port4/DIO4		
D GND	40	39	Port4/DIO3		
D GND	38	37	Port4/DIO2		
D GND	36	35	Port4/DIO1		
D GND	34	33	Port4/DIO0		
Port4/DIOCTL	32	31	Port4/DIO9		
D GND	30	29	Port3/DIO8		
D GND	28	27	Port3/DIO7		
D GND	26	25	Port3/DIO6		
D GND	24	23	Port3/DIO5		
D GND	22	21	Port3/DIO4		
D GND	20	19	Port3/DIO3	DIN(MOSI)	IMU (Transceiver)
D GND	18	17	Port3/DIO2	SCLK	
D GND	16	15	Port3/DIO1	CS	
D GND	14	13	Port3/DIO0	DOUT(MISO)	
Port3/DIOCTL	12	11	Port3/DIO9		
D GND	10	9	Port7/DIO3		
D GND	8	7	Port7/DIO2	CS	Current Sensor (Motor1)
D GND	6	5	Port7/DIO1	DOUT(MISO)	
Port7/DIO9	4	3	Port7/DIO0	SCLK	
Port7/DIOCTL	2	1	D GND		

FIGURE 133.PCB1 CONECTION:TRANSCEIVER IMU AND MOTOR1 CURRENT SENSOR



D GND	50	49	Port9/DIO8		
5 V	48	47	Port9/DIO7		
D GND	46	45	Port9/DIO6		
5 V	44	43	Port9/DIO5		
D GND	42	41	Port9/DIO4		
D GND	40	39	Port9/DIO3		
D GND	38	37	Port9/DIO2	SCLK	Current Sensor (Motor2)
D GND	36	35	Port9/DIO1	DOUT(MISO)	
D GND	34	33	Port9/DIO0	CS	
Port9/DIOCTL	32	31	Port9/DIO9		
D GND	30	29	Port8/DIO8		
D GND	28	27	Port8/DIO7		
D GND	26	25	Port8/DIO6		
D GND	24	23	Port8/DIO5		
D GND	22	21	Port8/DIO4		
D GND	20	19	Port8/DIO3		
D GND	18	17	Port8/DIO2		
D GND	16	15	Port8/DIO1		
D GND	14	13	Port8/DIO0		
Port8/DIOCTL	12	11	Port8/DIO9		
D GND	10	9	Port7/DIO8		
D GND	8	7	Port7/DIO7	CS	Battery Managment ADC
D GND	6	5	Port7/DIO6	DOUT(MISO)	
D GND	4	3	Port7/DIO5	SCLK	
Port7/DIO4	2	1	D GND		

FIGURE 134. PCB2 CONNECTION: BATTERY MANAGEMENT ADC AND MOTOR2 CURRENT SENSOR



D GND	50	49	Port9/DIO8		
5 V	48	47	Port9/DIO7		
D GND	46	45	Port9/DIO6		
5 V	44	43	Port9/DIO5		
D GND	42	41	Port9/DIO4		
D GND	40	39	Port9/DIO3		
D GND	38	37	Port9/DIO2	GPS Tx	GPS
D GND	36	35	Port9/DIO1	DIN(MOSI)	IMU (Transceiver)
D GND	34	33	Port9/DIO0	SCLK	
Port9/DIOCTL	32	31	Port9/DIO9	CS	
D GND	30	29	Port8/DIO8	DOUT(MISO)	
D GND	28	27	Port8/DIO7	CS	Battery Management
D GND	26	25	Port8/DIO6	DOUT(MISO)	
D GND	24	23	Port8/DIO5	SCLK	
D GND	22	21	Port8/DIO4	CS	Current Sensor (Motor2)
D GND	20	19	Port8/DIO3	DOUT(MISO)	
D GND	18	17	Port8/DIO2	SCLK	
D GND	16	15	Port8/DIO1	CS	Current Sensor (Motor1)
D GND	14	13	Port8/DIO0	DOUT(MISO)	
Port8/DIOCTL	12	11	Port8/DIO9	SCLK	
D GND	10	9	Port7/DIO8	DIN(MOSI)	
D GND	8	7	Port7/DIO7	SCLK	IMU (Board)
D GND	6	5	Port7/DIO6	CS	
D GND	4	3	Port7/DIO5	DOUT(MISO)	
Port7/DIO4	2	1	D GND		

FIGURE 135.FINAL BOARD CONNECTION