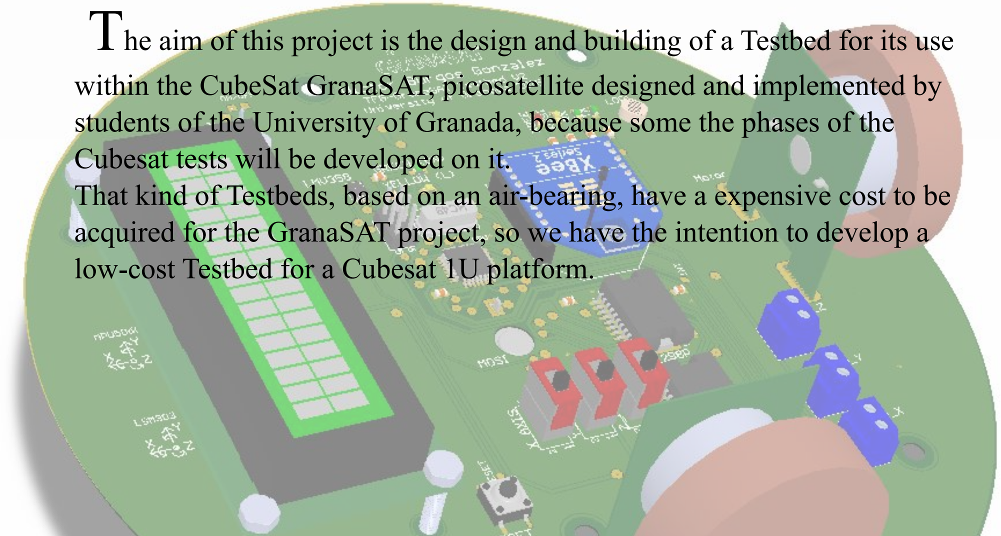


The aim of this project is the design and building of a Testbed for its use within the CubeSat GranaSAT, picosatellite designed and implemented by students of the University of Granada, because some the phases of the Cubesat tests will be developed on it.
That kind of Testbeds, based on an air-bearing, have a expensive cost to be acquired for the GranaSAT project, so we have the intention to develop a low-cost Testbed for a Cubesat 1U platform.

Víctor Burgos González is a telecommunications engineer from Córdoba, Spain. He finished his Bachelor's Studies in 2015 at University of Granada. Víctor starts working with GranaSAT Team in 2013, involved in BEXUS programme, and continued with his Bachelor Thesis designing and developing the Testbed for the GranaSAT Cubesat, specializing his studies in the aerospace field.

Andrés María Roldán Aranda is the academic head of the GranaSAT development team, and the tutor of this project. He is professor in Department of Electronics and Computer Technology at University of Granada.

BACHELOR THESIS

Testbed for a 1U Cubesat

Víctor Burgos González

TELECOMMUNICATIONS
ENGINEERING

2014/15

UNIVERSITY OF GRANADA

Bachelor Degree in Telecommunications Technology Engineering

BACHELOR THESIS

Testbed for a 1U Cubesat

Víctor Burgos González
Academic year 2014/2015

Tutor: Andrés María Roldán Aranda

Universidad
de Granada

GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

"Testbed for a 1U Cubesat"

CURSO: 2014/2015

Víctor Burgos González

GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

“Testbed for a 1U Cubesat”

REALIZADO POR:

Víctor Burgos González

DIRIGIDO POR:

Andrés María Roldán Aranda

DEPARTAMENTO:

Electrónica y Tecnología de los Computadores

D. Andrés María Roldán Aranda, Profesor del departamento de Electrónica y Tecnología de los Computadores de la Universidad de Granada, como director del Trabajo Fin de Grado de D. Víctor Burgos González,

Informa:

Que el presente trabajo, titulado:

“Testbed for a 1U Cubesat”

ha sido realizado bajo su supervisión por Víctor Burgos González, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en

Granada, a 10 de Septiembre de 2015

Fdo. Andrés María Roldán Aranda

Yo, Víctor Burgos González, alumno de la titulación GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN de la Escuela Técnica Superior de Ingeniería Informática y de Telecomunicación de la Universidad de Granada, con DNI [redacted], autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Granada, a 10 de Septiembre de 2015.

Fdo. Víctor Burgos González

Fdo. Andrés María Roldán Aranda

Testbed for a 1U Cubesat

Víctor Burgos González

PALABRAS CLAVE:

Testbed, Cubesat, picosatélite, GranaSAT, Arduino, acelerómetro, magnetómetro, rodamiento, aire, sin fricción, lowcost, MATLAB, impresión, 3D, solidworks, altium, inercia, ruedas, giróscopo, análisis, ESA, mecánica, diseño, electrónica, PCB.

RESUMEN:

Los Cubesats dan a los estudiantes universitarios la oportunidad de tener una experiencia práctica diseñando, implementando y trabajando con la operación de un proyecto aeroespacial real. El objetivo de este trabajo es el diseño y construcción de un Testbed para su uso con el Cubesat GranaSAT, un picosatélite diseñado e implementado por alumnos de la Universidad de Granada, porque algunas fases de test del Cubesat serán desarrolladas en él. Este tipo de Testbeds, basados en un rodamiento de aire, tienen un coste muy elevado para ser adquirido por el proyecto GranaSAT, por ello, tenemos la intención de fabricar un Testbed de bajo coste para una plataforma Cubesat 1U.

El Testbed consiste en un entorno esférico con ficción reducida, que será impreso 3D en ABS con una Prusa i3 Reprap y con el uso de un compresor de aire 1/4" que permitirá que una semiesfera de plástico transparente (donde irán los demás subsistemas) flote sin ningún tipo de soporte externo. Este proyecto tendrá el diseño del sistema, el cálculo de los costes de éste, el desarollo e implementación del diseño, y las aplicaciones de éste Testbed de bajo coste. El autor de este proyecto es uno de los miembros del equipo GranaSAT.

KEYWORDS:

Testbed, Cubesat, GranaSAT, Arduino, accelerometer, magnetometer, Air-bearing, bearing, frictionless, lowcost, MATLAB, 3D printing, printing, 3D, solidworks, altium, reaction, wheels, reactionwheels, gyroscope, analysis, ESA, diseño, mecánica, electrónica, PCB.

ABSTRACT:

Cubesats allow students the possibility to have a true hands-on experience with the design, implementation and operation of a real aerospace experience. The aim of this project is the design and building of a Testbed for its use within the Cubesat GranaSAT, picosatellite designed and implemented by students of the University of Granada, because some the phases of the Cubesat tests will be developed on it. That kind of Testbeds, based on an Air-bearing, have a expensive cost to be acquired for the GranaSAT project, so we have the intention to develop a low-cost Testbed for a Cubesat 1U platform.

The Testbed consists of a spherical reduced-friction environment, 3D printed in ABS with a Prusa i3 RepRap printer, with an air compressed 1/4” inlet that allow the floating of the platforms over a clear plastic hemisphere without external attachments. This project will consist of the system design, cost, performance, operating procedures, and applications of this low cost Testbed. The author of this project is one of the members of GranaSAT team.

Dedicado a

Mis padres, Paco y Encarni, y mi hermano Alberto,
porque sin ellos y sin su apoyo, llegar hasta aquí
hubiera sido imposible.

Agradecimientos:

En primer lugar, me gustaría agradecer a mis padres Paco y Encarni, y a mi hermano Alberto su apoyo durante mi periodo universitario, y en general, durante toda mi vida. Hubiese sido imposible conseguir lo que he conseguido hasta ahora sin su incondicional apoyo y compañía en cada momento. Con este proyecto, cierro mi etapa universitaria, pero abro otra en la que seguiréis siendo los principales protagonistas.

También quiero agradecer a mis compañeros y amigos de Teleco, quiénes me han ayudado en cada momento que lo he necesitado en mis años universitarios., y cada amigo que dió su mejor apoyo hacia mi persona en toda mi vida. Pero especialmente me gustaría agradecer a Mari Jose por su apoyo, paciencia y continuo cariño durante mis últimos cuatro años.

Por último, agradezco a todo el equipo de desarrollo de GranaSAT su colaboración, pero en especial a mi tutor, Andrés María Roldán Aranda por su tiempo dedicado, y comentarios intructivos, no sólo para este proyecto, sino para mi futura vida profesional. Espero que nunca cambies Andrés.

Acknowledgments:

Firstly, I would like to thank my parents Paco y Encarni and my brother Alberto for their support during my university period and, in general, my whole life. It would have been impossible to achieve what I achieved without their unconditional support and company in every moment. With this project, I close my university stage, opening up a new one in which you will also be main characters.

I want to thank to my Teleco's colleagues and friends too, who help me in every moment that I need in my university years, and each friend who do their best support to my person in my whole life. But specially I would also thank to Mari Jose for her support, quiet patience and continuous love in my last four years.

Finally, I would like to thank to the GranaSAT Development Team, but particulary my advisor, Andrés María Roldán Aranda for his provided timely and instructive comments, not only for this project, but for my future professional life. I hope you will never change.

INDEX

Cover Page i

Autorización Lectura vi

Autorización Depósito Biblioteca vii

Abstract ix

Dedicatoria xiii

Acknowledgments xv

Index xvii

List of Figures xxiii

List of Tables xxxi

1 Introduction 1

1.1 Problem Statement . . . 3
1.2 Motivation . . . 3
1.3 Project Objectives . . . 3
1.4 Project Structure . . . 4

2 Background study **7**
2.1 Cubesat introduction . . . 9
2.2 Attitude Determination and Control System background . . . 10
2.3 Coordination and orientation references systems . . . 12
2.4 Spacecraft Motion Equations . . . 13
2.5 System Requirements . . . 18
  2.5.1 Communication System . . . 18
  2.5.2 Testbed PCB and electronic devices . . . 19
  2.5.3 Mechanical parts . . . 19
  2.5.4 Software . . . 20
    2.5.4.1 Telemetry . . . 20
    2.5.4.2 Telecontrol . . . 20
    2.5.4.3 Software onboard . . . 20

3 System Analysis **23**
3.1 ADCS Testbed Technology . . . 23
3.2 Sensors for the ADCS . . . 27
3.3 ADCS Actuators . . . 33
3.4 On-board microprocessor selection . . . 36
3.5 PID control . . . 36
  3.5.1 PID discretization . . . 38
  3.5.2 PID tuning . . . 39

4 System Design **41**

4.1 Electronic Design 41
4.1.1 Electronic Schematics and components descriptions 41
4.1.1.1 Microprocessor 43
4.1.1.2 Display LCD 45
4.1.1.3 Magnetometer and Accelerometer Sensor 47
4.1.1.4 Gyroscope and Accelerometer Sensor 50
4.1.1.5 Wireless Module 52
4.1.1.6 Actuators Drivers 56
4.1.1.7 Power system 58
4.1.1.7.1 Battery 60
4.1.1.7.2 Voltage detectors 61
4.1.1.7.3 DC-DC converters 62
4.1.2 PCB Design 63
4.1.2.1 Design rules PCB 65
4.1.2.2 2D view of the PCB Design 66
4.1.2.3 3D view of the PCB Design 71
4.2 Mechanical Design 74
4.2.1 Hemisphere 75
4.2.2 Air-bearing 77
4.2.3 Air-compressor 84
4.2.4 Reaction Wheels 84
4.2.5 Assembly of the whole parts 88
4.3 Software Design 89
4.3.1 Communication protocol 89
4.3.2 On-board software 92
4.3.3 GS software 94

5 Integration, tests and verification **97**

5.1 Motor characterization 97

5.1.1 Power Supply - RPM Test 97

5.1.2 Electrical parameters measurements 102

5.1.3 Motors step responses 114

5.2 Motors simulation block 118

5.3 Final implementation 124

5.3.1 PCB implementation 124

5.3.2 3D Printing process 127

5.4 Air-bearing tests 132

5.4.1 First model of Air-bearing 132

5.4.2 Second model of Air-bearing 133

5.4.3 Mass of the main objects 134

5.5 Actuators tests 134

5.6 MPU6050 calibration 135

5.7 Final implementation 135

6 Conclusions and Future Lines **141**

A User guide: Using Arduino in Atmel Studio **1**

A.1 How to compile Arduino's code in Atmel Studio 6 1

A.2 Uploading the Arduino Code in Atmel Studio 11

B 3D printing **19**

B.1 Introduction to *RepRap* 19

B.2 Printers used in this project 20

B.2.1 Prusa Mendel i2 21

B.2.2 Prusa Mendel i3 22

B.3 3D printing improvements 24

B.3.1 Relay configuration 24

B.3.2 Adding the LCD control to the printers 27
B.3.3 Adding the fan for the electronics 30

C Gantt Diagram **33**

D Test tables **35**

D.1 Power Supply - RPM Test Tables 35

E Project Budget **37**

E.1 Electronics costs 37
E.2 Mechanics 41
E.3 Software 41
E.4 Human Resources 41

Acronyms **43**

Glossary **47**

References **49**

LIST OF FIGURES

1.1 Logo GranaSAT 1
1.2 BEXUS19 Launch [39] 2
1.3 Flow of the project process 4

2.1 Diagram of the whole system 8
2.2 Different Cubesats depending on the dimension 9
2.3 ADCS for a satellite [15] 10
2.4 ADCS GranaSAT diagram 11
2.5 Earth's Field Simulator 11
2.6 Cubesat Delfi-n3Xt actuators example [42] 12
2.7 Fixed Reference System (FRS) 14
2.8 Sphere Reference System (SRS) 15
2.9 Euler angles definitions 16
2.10 Euler angles in a satellite [15] 16
2.11 Angular momentum example 17

3.1 Drop Tower NASA [41] 24
3.2 ARL Testbed (planar Air-bearing)[47] 25
3.3 Spherical Air-bearings Styles [9] 26
3.4 TACT Dumbbell Style Air-bearing [46] 26
3.5 DSACSS Air-bearing Set [48] 27
3.6 FloatSat project [25] 27
3.7 Star tracker 28
3.8 Horizon Sensor scheme [59] 29
3.9 Horizon Sensor GranaSAT BEXUS [53] 29
3.10 Possible shadow of a larger satellite [65] 30
3.11 Possible reflection in the sun sensor [65] 30
3.12 MEMS accelerometer[45] 31
3.13 MEMS gyroscope[13] 32
3.14 Permanent magnets and Hysteresis rods actuators - GeneSat-1 [1] 33
3.15 Bronze bar for the reaction wheels 34
3.16 Manufactured wheel 35
3.17 Motor connection 36
3.18 Motor dimensions [12] 36
3.19 PID Diagram [64] 38
3.20 ZN Tuning (Ideal process) [43] 39
3.21 ZN Tuning (Real process) [43] 39

4.1 Pinout ATMEGA328P-AU 43
4.2 Schematic for ATMEGA328P-AU 43
4.3 Reset button 44
4.4 LCD Schematic 45
4.5 LCD to I2C converter 46
4.6 LCD to I2C converter schematic [49] 47

4.7 Photo of the I2C converter and the LCD [26] 47
4.8 LSM303DLHC chip [52] 48
4.9 Photo of the the test board for LSM303DLHC 48
4.10 Schematic of the test board for LSM303DLHC [18] 49
4.11 Schematic symbol for Testbed Board for LSM303DLHC 50
4.12 Photo of the test board for MPU6050 50
4.13 Schematic of the test board for MPU6050 51
4.14 Schematic of the test board for MPU6050 [5] 51
4.15 I2C diagram of the system 52
4.16 Module Xbee Pro Series 1 52
4.17 Dimensions of the Module Xbee Pro Series 1 53
4.18 Pinout of the Xbee Module [17] 54
4.19 Schematic of the Xbee Module 55
4.20 Pinout of the L298P [50] 56
4.21 Pin functions of the L298P [50] 56
4.22 L298P Schematic [50] 57
4.23 S1M package[50] 58
4.24 Switch NKK SS22SDP2 58
4.25 Schematic for the power system of the Testbed 59
4.26 MGL2803 battery 60
4.27 MAX8211 Voltage Detector 61
4.28 MAX1771 Step-up Converter 62
4.29 LTC3440 Pinout 63
4.30 LPFK ProtoMat S62 64
4.31 Design Rule Check - Altium Designer 65
4.32 3D View of the top side 71
4.33 3D View of the bottom side 72
4.34 3D View of the right side 73

4.35 3D View of the left side 74
4.36 Clear hemisphere 75
4.37 Air-bearing section to show the support triangles 83
4.38 Black&Decker CP2525 air compressor [8] 84
4.39 Assembly wheel-motor 87
4.40 Testbed assembly 89
4.41 Flow diagram 91
4.42 On-board State diagram 93
4.43 Testbed telecontrol panel designed 94
4.44 Configuration panel used 95
4.45 Sensors ADCS panel used 95

5.1 Tachometer DT-2234B [19] 98
5.2 Motors prepared for the VCC-RPM test 99
5.3 VCC-RPM results for Kysan RF-300CH 99
5.4 VCC-RPM results for the motor without model 100
5.5 VCC-RPM results for Minebea MDN3BT 100
5.6 VCC-RPM results comparison 101
5.7 VCC-RPM results with wheels for Kysan RF-300CH 101
5.8 VCC-RPM results with wheels for Minebea MDN3BT 102
5.9 VCC-RPM results with wheels for the motor without model 102
5.10 Model circuit for DC motor 103
5.11 LCR options to measure Ra and La 104
5.12 Measuring MDN3BT resistance and inductance 104
5.13 Measuring RF300-CH resistance and inductance 105
5.14 Measuring the motor without name resistance and inductance 105
5.15 Mechanical time constant representation [10] 109
5.16 Mechanical time constant graph for MDN3BT 109

5.17 Mechanical time constant graph for RF300-CH 110
5.18 Mechanical time constant graph for no-name motor 110
5.19 Graph for the regression determination of damping ratio (MDN3BT) 113
5.20 Graph for the regression determination of damping ratio (no name motor) . 113
5.21 Graph for the regression determination of damping ratio (RF300-CH) 113
5.22 Key-sight N6705A Source [54] 114
5.23 Output configuration of the N6705A to supply the motors 115
5.24 Step response captures from N6705A 116
5.25 Step response graphs from the data-logger of N6705A 117
5.26 Simulink simulation - Main diagram 118
5.27 Simulink simulation - DC Motor block 119
5.28 Simulink simulation - Output for MDN3BT 120
5.29 Simulink simulation - Output for no name motor 121
5.30 Simulink simulation - Output for RF300-CH 122
5.31 Step response graphs from the data-logger of N6705A 123
5.32 Mixture application on the PCB 124
5.33 First soldered devices and vias 125
5.34 PCB Implementation 126
5.35 Assembly reaction wheel with motor 126
5.36 Calibration of the printer 127
5.37 Result of a bad calibration 128
5.38 Models for the calibration 129
5.39 Printing Testbed V1 130
5.40 Testbed V1 printed 131
5.41 Failed printed part 131
5.42 Printing the final version of the Testbed 132
5.43 Airgun for air-compressor 133
5.44 Final assembly with only Testbed PCB 136

5.45 Final assembly with only Testbed PCB 137
5.46 Final assembly Testbed 138
5.47 Final assembly Testbed 139

A.1 Installation Atmel Studio (1) 2
A.2 Installation Atmel Studio (2) 2
A.3 Installation Atmel Studio (3) 3
A.4 Installation Atmel Studio (4) 3
A.5 Installation Atmel Studio (5) 4
A.6 Installation Atmel Studio (6) 4
A.7 Installation Visual Micro (1) 5
A.8 Installation Visual Micro (2) 6
A.9 Installation Visual Micro (3) 6
A.10 Installation Visual Micro (4) 7
A.11 Installation Visual Micro (5) 7
A.12 Installation Visual Micro (6) 8
A.13 Installation Visual Micro (7) 8
A.14 Compiling the Arduino Code in Atmel Studio (1) 9
A.15 Compiling the Arduino Code in Atmel Studio (2) 9
A.16 Compiling the Arduino Code in Atmel Studio (3) 10
A.17 Compiling the Arduino Code in Atmel Studio (4) 10
A.18 Compiling the Arduino Code in Atmel Studio (5) 11
A.19 Uploading the Arduino Code in Atmel Studio (6) 11
A.20 Uploading the Arduino Code in Atmel Studio (7) 12
A.21 Uploading the Arduino Code in Atmel Studio (8) 12
A.22 Uploading the Arduino Code in Atmel Studio (9) 13
A.23 Uploading the Arduino Code in Atmel Studio (10) 13
A.24 Uploading the Arduino Code in Atmel Studio (11) 14

A.25 Uploading the Arduino Code in Atmel Studio (12) 15
A.26 Uploading the Arduino Code in Atmel Studio (13) 15
A.27 Uploading the Arduino Code in Atmel Studio (14) 16
A.28 Uploading the Arduino Code in Atmel Studio (15) 16
A.29 Uploading the Arduino Code in Atmel Studio (16) 17
A.30 Uploading the Arduino Code in Atmel Studio (17) 17
A.31 Uploading the Arduino Code in Atmel Studio (18) 18
A.32 Uploading the Arduino Code in Atmel Studio (19) 18

B.1 Logo RepRap 20
B.2 Prusa Mendel i2 21
B.3 Prusa Mendel i3 23
B.4 Overheating of the RAMPS 24
B.5 Overheating of the RAMPS (2) 25
B.6 Goodsky Relays used for the solution 25
B.7 Schematic for configuration Relays in Prusa Mendel i2 26
B.8 LCD Smart Controller 27
B.9 LCD Menu Tree 29
B.10 Installed fan and support parts 31

LIST OF TABLES

3.1 Comparison of some kind of sensors [34] 31
3.2 Comparison of gyroscopes [31][37][38] 32
3.3 Different materials for the wheel 34
3.4 Comparison of motors [12][20][40][36] 35
3.5 Comparison of microprocessors [7] 37
3.6 ZN Tuning calculation [28] 40

4.1 Calculations of the resistor for the LEDs 44
4.2 LSM303DLHC Specifications [52] 49
4.3 MPU6050 Specifications [31] 52
4.4 Xbee PRO Series S1 Specifications [17] 53
4.5 Calculations of the resistors for the Xbee LEDs 55
4.6 Rectifier Diode S1M Specifications [23] 58
4.7 ENIX MGL2803 Battery Specifications [23] 60
4.8 MAX1771 Pins function 62
4.9 Main characteristics of CP2525 compressor [8] 84

4.10 Density of different materials 85
4.11 Reaction wheels dimensions 87
4.12 Reaction wheels mass 88
4.13 Data type used [59] 90

5.1 DT-2234B Photo tachometer Specifications [19] 98
5.2 Armature resistance measurements 105
5.3 Start conditions for each motor 106
5.4 Armature inductance measurements 106
5.5 Electrical time constant calculation 106
5.6 Counter-electromotive force calculation for MDN3BT 107
5.7 Counter-electromotive force calculation for RF-300CH 107
5.8 Counter-electromotive force calculation for motor without name 107
5.9 Counter-electromotive constant calculation 108
5.10 Torque constant calculation 108
5.11 Mechanical time constant measurement 109
5.12 Inertia momentum calculation 111
5.13 Friction torque calculation 111
5.14 Parameters necessary for the B calculation (MDN3BT) 112
5.15 Parameters necessary for the B calculation (no name motor) 112
5.16 Parameters necessary for the B calculation (RF300) 112
5.17 Summary motors parameters 114
5.18 Summary object mass 134
5.19 MPU6050 calibration 135

B.1 Quotes requested for 3D printing 21
B.2 Prusa Mendel i2 Specifications 22
B.3 Prusa Mendel i3 Specifications 23
B.4 GOODSKY RWHSH112D Specifications 25

D.1 VCC-RPM test for Kysan RF-300CH 35
D.2 VCC-RPM test for no-model motor 35
D.3 VCC-RPM test for Minebea MDN3BT 36
D.4 VCC-RPM test for Mitsumi M25E-4 (not used) 36

E.1 PCB building cost 37
E.2 Total cost of the PCB implementation 38
E.3 Budget for the electronics devices on PCB 39
E.4 Total cost of the mechanical implementation 41
E.5 Software cost 41
E.6 Human resources cost 42

CHAPTER

1

INTRODUCTION

The following final degree project completed the studies of the Degree in Telecommunications Technology Engineering. The aim of this project was the design and implementation of a Testbed for its use for a 1U Cubesat, GranaSAT.

GranaSAT, whose logo is shown in figure 1.1, is an aerospace project, carried out in the University of Granada, which aims to build a Cubesat by the student's final projects degree. This project is coordinated by the professor Andrés María Roldán Aranda and it allows to the students acquire knowledge about the aerospace field and a real experience in this area. The equipment needed for that project and the laboratories for the student's projects are located in the Sciences Faculty of the University of Granada.

Figure 1.1 – *Logo GranaSAT*

GranaSAT was involved in the BEXUS/REXUS programme defined in its webpage as a programme realized between the German Aerospace Center (DLR) and the Swedish National Space Board (SNSB), and a collaboration with the European Space Agency (ESA) [44]. In the case of GranaSAT, the choice was BEXUS, the part of this programme where the students use an stratospheric balloon to test their systems.

GranaSAT used BEXUS to prove the system which will be included in the future Cubesat. During one year, the team (the author of this document was part of this team) designed an Attitude Determination System (ADS) based on three different methods: a Horizon Sensor, a Magnetometer and a Star Tracker [39].

The experiment was launched in October 2014 in Kiruna (Sweden), where the Esrange Space Center(SSC) is located and the result of this launch for GranaSAT Team was successful. In the Figure 1.2, is shown the launch of this project.

Figure 1.2 – *BEXUS19 Launch [39]*

In order to test all over systems for the Cubesat, a Testbed is needed, a simulation platform designed for the test of a aerospace system, in this case, the GranaSAT's subsystems. This platform will imitate the friction-less environment of the space scenario and it will take the measurements to analyze this scenario with some sensors.

1.1 Problem Statement

The design, implementation and testing of an Attitude Determination and Control System (ADCS) in the University, is a hard task. One of the most important reasons is the microgravity and the reduced friction that ADCS support in the aerospace scenario.

The equipment and components needed for this Cubesat will have been tested for the company which distributes them, but we need a Testbed for the tests of the whole system. That Testbed will enable us to have the hardware loop in the testing process, which is essential for the engineers who build the picosatellite. In this way, they will be able to prove their subsystems and resolve the design and development errors before the mission flight.

1.2 Motivation

The design and implementation of a Testbed like this, for its use in the University of Granada, will provide student the test process and, consequently, the determination of their errors in a simulated micro-gravity, reduced friction environment.

This project will facilitate the GranaSAT development team the tools for the mandatory tests that their subsystems will have to take before the implementation within the Cubesat. When the subsystem of each student was proved in the Testbed, they will have the PASS for the next phase of the tests (thermal, vacuum, shock, vibration).

1.3 Project Objectives

In order to get a successful result for that final degree project, the following objectives have to be fulfill.

- Design and implementation of a Cubesat Testbed which will be able to test the Cubesat ADCS system and subsystems for the Cubesat:
    - Support for the estimation and control systems.
    - Validation for the estimation and control systems.
    - Platform simulating the micro-gravity with a reduce friction Air-bearing.
    - Measure the tested environment with the sensors installed in the Testbed.
- Allow GranaSAT development team to verify and validate the individual components of the Cubesat subsystems developed by students of the UGR.

1.4 Project Structure

In this project, each chapter is directly related with the system engineering process concerned. In the flow diagram of the figure 1.3 there is a scheme of this process and, the appendix C shows how the schedule of this project has flown.

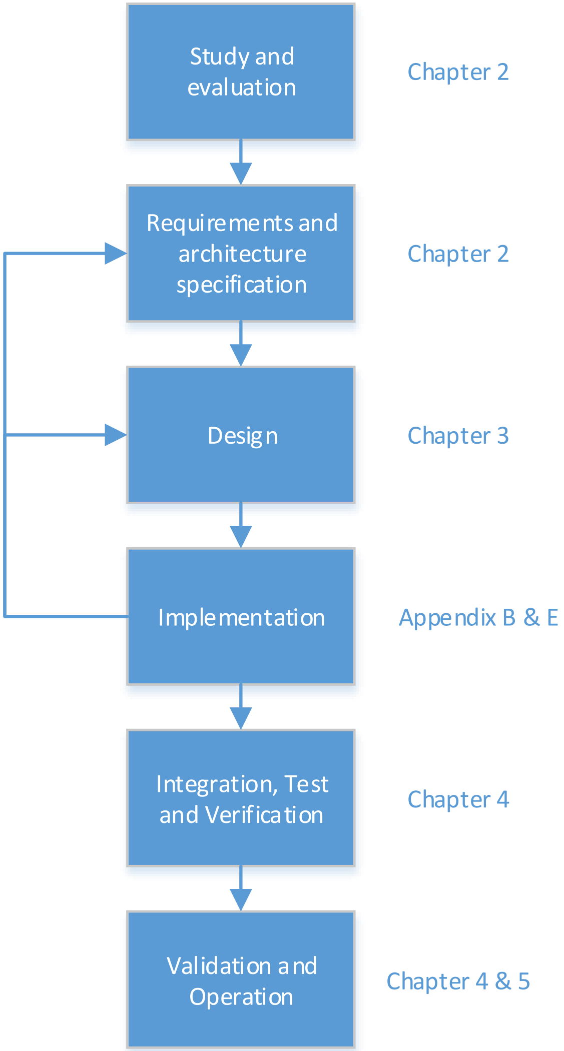


Figure 1.3 – *Flow of the project process*

- **Chapter 2: Background and Analysis:** in this chapter, we will analyze how a Testbed is studied, the different subsystem that it should include, and the theoretic questions that it has to develop. Moreover, a background about the Cubesats and the

ADCS for the GranaSAT Testbed .

Finally, in section 2.5, it will be studied the requirements of each subsystem for our Testbed.

- **Chapter 4: System Design** in this chapter, we will describe the design of the whole system, electronics and mechanical subsystems chosen for the GranaSAT Testbed. Moreover, the software design will be studied in this chapter too.
- **Chapter 5: Integration, test and verification** in this chapter, we will explain the test and verification procedures followed to prove the whole system, and the followed integration to ensure its correct operation.
- **Chapter 6: Conclusions and future lines**: the conclusions of the project and a briefly description of the possible improvements and future work related with this project.

CHAPTER

2

BACKGROUND STUDY

The Testbed for the Cubesat GranaSAT, uses an ADCS to stabilize the system, in our case, the Cubesat, and we have to take into account the references system that we are using, because the ADCS depends directly on the reference system.

The actuators (reaction wheels and magnetorquers) will act to regulate, by a PID control, the attitude of the system, measured by sensors (in our case, an accelerometer-magnetometer and an accelerometer-gyroscope).

The whole system is represented in the diagram of the figure 2.1.

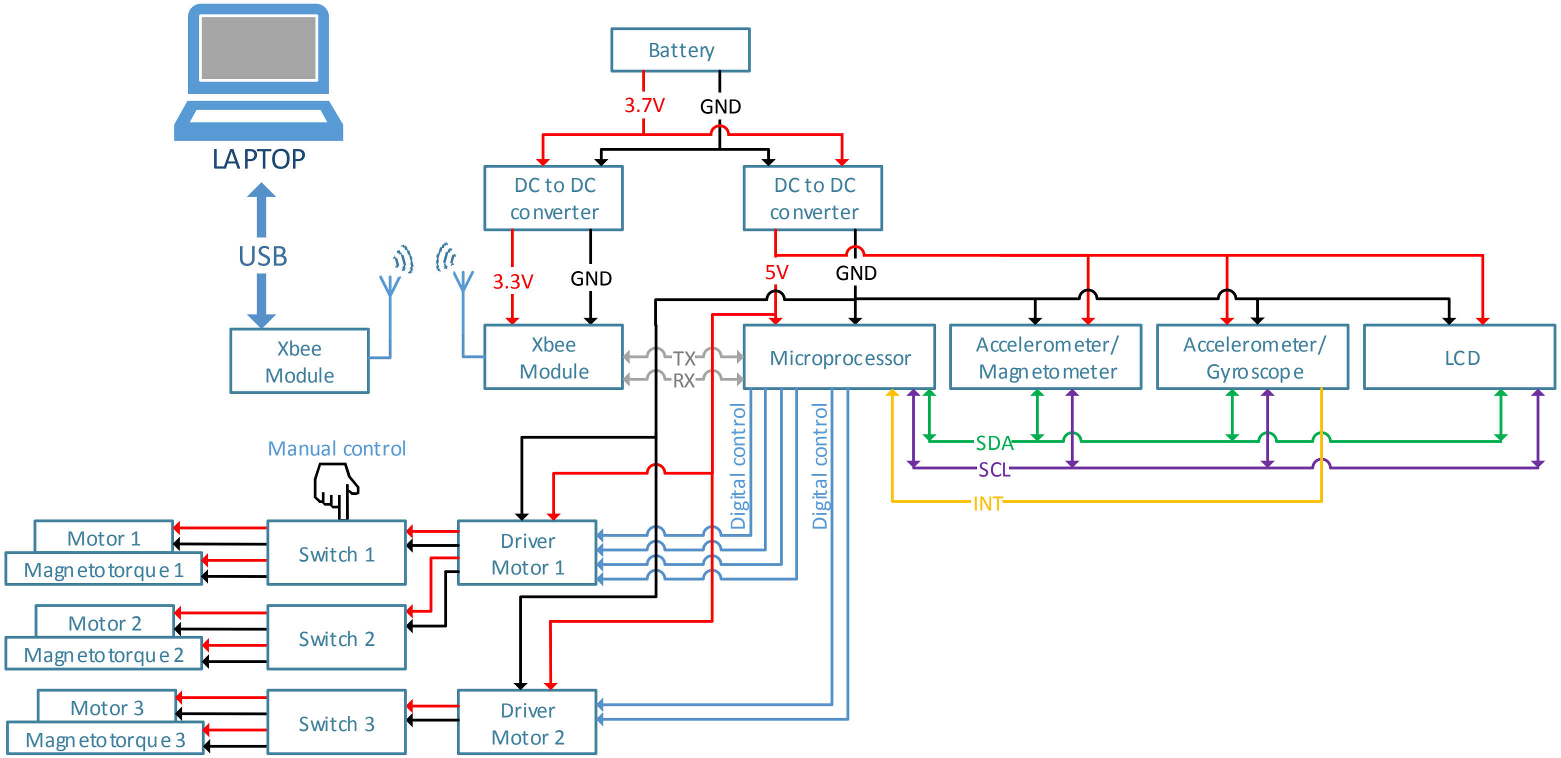


Figure 2.1 – *Diagram of the whole system*

2.1 Cubesat introduction

The aim of GranaSAT project is to build a Cubesat, a picosatellite. A Cubesat is a miniaturized satellite, which has the aim of a determined space research, with a volume of 10 cm cube, and a mass of less than 1.33 Kg [58].

The Cubesats have the origin in 1999, thanks to a collaborative work between Prof. Jordi PuigSuari at California Polytechnic State University (Cal Poly), San Luis Obispo, and Prof. Bob Twiggs at Stanford University's Space Systems Development Laboratory (SSDL). The Cubesat purpose was to provide a standard for design of picosatellites in order to have a reduced cost for these kind of applications and a fster way to develop the aerospace projects [58].

There are different types of Cubesat, depending of its size (see figure 2.2).

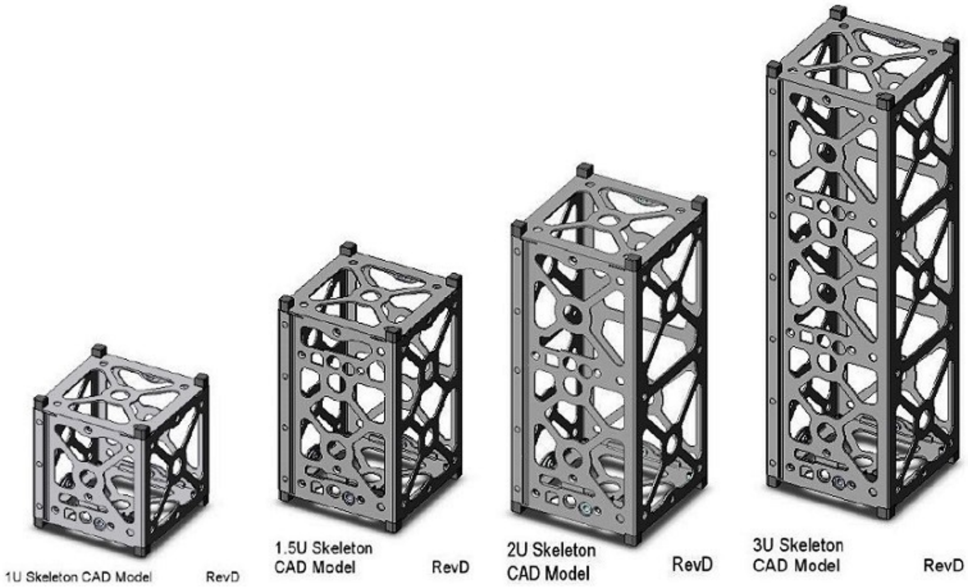


Figure 2.2 – *Different Cubesats depending on the dimension*

Furthermore, these Cubesats have a maximum mass allowed [58]:

1. 1U Cubesat: 1.33kg.
2. 1.5U Cubesat: 2kg.
3. 2U Cubesat: 2.66kg.
4. 3U Cubesat: 4kg.

Note: Maybe they could have larger masses evaluated according to the mission.

The Testbed platform will be designed with the intent to prove the Cubesat GranaSAT (1U type) and verify that every system which be mounted in it, work correctly in the aerospace environment. That restriction about the dimensions and the maximum weight allowed will be relevant for the design of the systems of the Cubesat, moreover, the Testbed systems, because the Testbed has to simulate the environment that the Cubesat will have in the aerospace field.

2.2 Attitude Determination and Control System background

An ADCS for Cubesat is used to regulate, stabilize and orient the vehicle. The attitude of a satellite is its orientation of a spacecraft body with respect to a defined external frame of reference or another entity, for example nearby objects or certain fields. In addition, it can be used a gyroscope or other inertial sensors to determine the orientation of the spacecraft object, so, when the external fixed inertial measurements are not availables, it will possible that determination. A better way to understand the ADCS for a satellite is shown in figure 2.3.

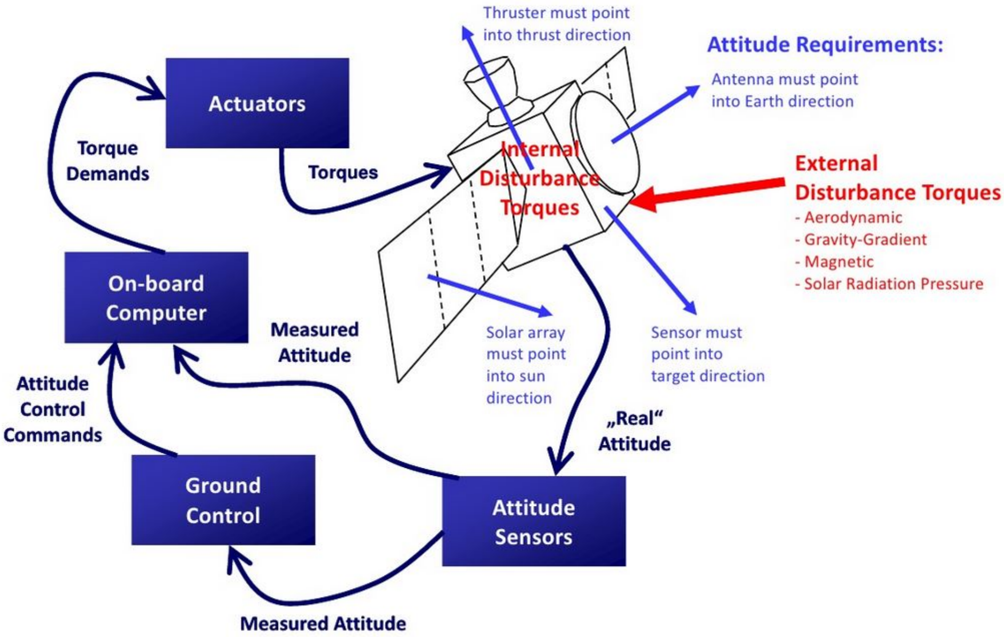


Figure 2.3 – *ADCS for a satellite [15]*

In order to control the attitude, the system generates some torques due to the actuators installed on it, because in the spacecraft exists different disturbance torques that affecting their attitude. In other words, the attitude control is the maintenance of a desired, specified attitude within a given tolerance (see figure 2.4) [16].

One of the most common actuators used in the spacecraft are the thrusters, but they required an significant quantity of fuel for their functionality, so that is a relevant constraint for a Cubesat(or the Testbed), because of its size and its limited weight.

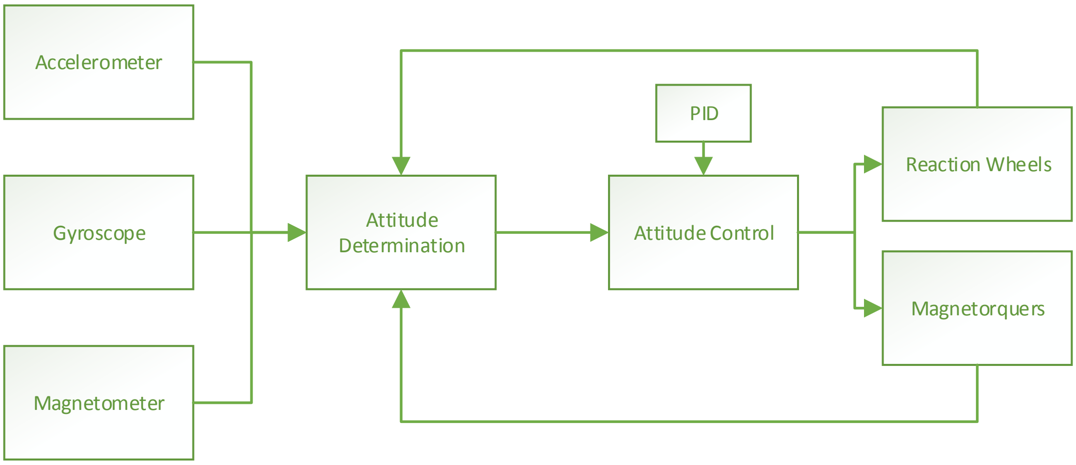
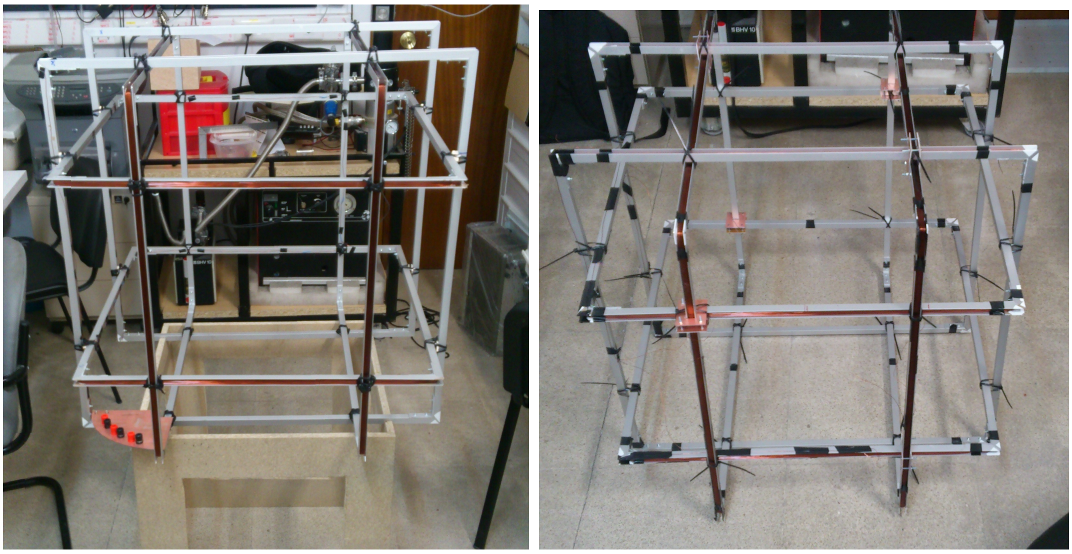


Figure 2.4 – *ADCS GranaSAT diagram*

Other example of actuators are the magnetorquers. They works creating a magnetic field against the Earth local magnetic field. In our Testbed, we will use three magnetorquers, one for each axis, and a 3D simulator of magnetic field (see figure 2.5) with which we will be able to simulate an orbit for our Cubesat and, with the actuators, generate the torque necessary to compensate it.



(a) *Earth's Field Simulator (1)* **(b)** *Earth's Field Simulator (2)*

Figure 2.5 – *Earth's Field Simulator*

Besides the magnetorquers, in our Testbed there will be another kind of actuators, the reaction wheels (see section 3.3 where the actuators are chosen). As well as the magnetorquers, we will place three reaction wheels, which produce inertial torques that not change the angular momentum of the Cubesat. The angular momentum of a fixed body that turn around an axis is the resistance that this body have with the variation of the angular velocity.

In some Cubesats there are installed both systems, **magnetorquers and reaction wheels** (see figure 2.6), because the external disturbances (produced for example by solar radiation, magnetic field from the Earth, gravity gradient torque, etc.), generates angular momentum within the Cubesat, and the reaction wheels store the angular momentum, but not dissipate it. When the reaction wheels reach their maximum storage of angular momentum, external torques, in our case, magnetorquers, reduced it (desaturing the reaction wheels) [60] [14]. Furthermore, the Earth's magnetic field decreases proportional with the distance, for that reason, the magnetorquers are useful only for low Earth orbits.

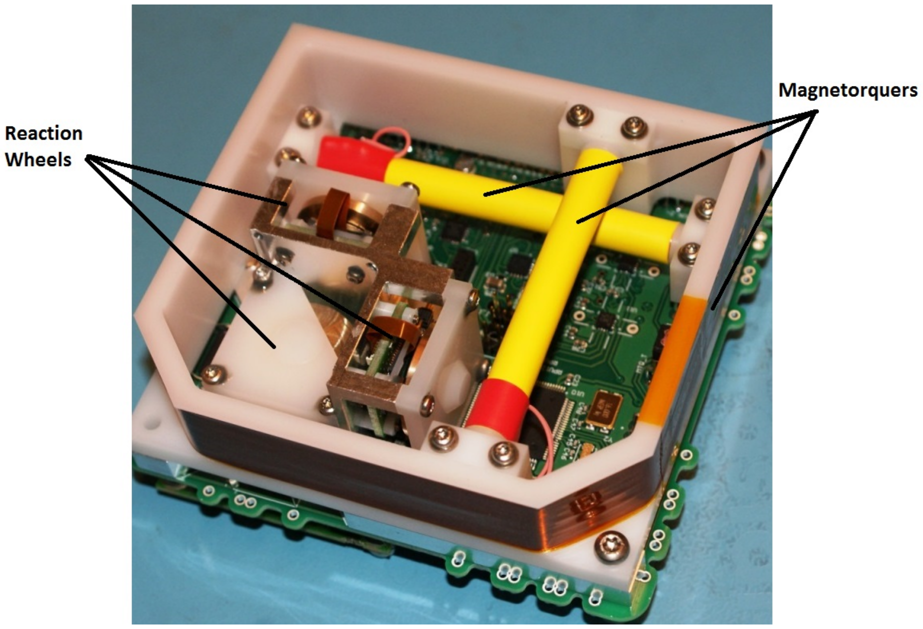


Figure 2.6 – *Cubesat Delfi-n3Xt actuators example [42]*

Thanks to the dimension and weight of the Cubesat (see figure 2.2), it is not necessary to have larger actuators, facilitating the design and the reduction of the power consumption for that actuators.

2.3 Coordination and orientation references systems

In GranaSAT's Testbed, exist differents coordinate systems, basing on the reference point that is treated:

- Fixed Reference System (FRS). This reference system is fixed from the sight of the viewer. The FRS has the point of origin in the center of rotation of the Air-bearing. That reference, will give us the error angle calculated between the FRS and SRS. See figure 2.7.
  - X axis (X_{F}). The X_{F} axis is aligned with the input of the air under pression, and has the same direction to that point. Furthermore, is perpendicular to the gravity

vector.

  - Y axis (Y_F). The Y_F axis is perpendicular to X_F axis and gravity vector too.
  - Z axis (Z_F). The Z_F axis is perpendicular to X_F and Y_F axis, and is parallel to the gravity vector, pointing to the opposite side of the floor.

- Sphere Reference System (SRS). That reference system is not fixed respect the laboratory, thus, is not fixed respect the FRS. It has the point of origin in the center of the PCB. See figure 2.8.
  - X axis (X_S). The X_S axis is pointing to the GranaSAT logo from the center of the Testbed PCB.
  - Y axis (Y_S). The Y_S axis is perpendicular to X_S axis. That axis is pointing to the LCD that is included in the PCB.
  - Z axis (Z_S). The Z_S axis is perpendicular to X_S and Y_S axis, and points toward, pointing to the Air-bearing.

With a view to clarify these axis, see figures 2.7 2.8.

In order to have an easiest way to manage the control attitude, we will work with the Euler angles, which describe a sequence of the three rotations that a body have, with the intention of align two coordinated systems. The Euler angles describe the orientation in 3-dimensional Euclidean space of a rigid body, thus we will need three parameters, normally the change in Ψ is named Yaw, Θ Pitch and Φ Roll, represented in the Figure 2.9.

The calculation of these angles is given with a matrix, which use the trigonometry rulesto determinate the values.

$$Yaw = \begin{pmatrix} cos\Psi & sin\Psi & 0 \\ -sin\Psi & cos\Psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.3.1}$$

$$Pitch = \begin{pmatrix} cos\Theta & 0 & -sin\Theta \\ 0 & 1 & 0 \\ sin\Theta & 0 & cos\Theta \end{pmatrix} \tag{2.3.2}$$

$$Roll = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\Phi & sin\Phi \\ 0 & -sin\Phi & cos\Phi \end{pmatrix} \tag{2.3.3}$$

That angles are represented in a satellite as the figure 2.10 shows.

2.4 Spacecraft Motion Equations

As we have seen in the previous section, the GranaSAT Testbed will have three DOF, then, three motion equations will define the movement of the Air-bearing using the SRS,

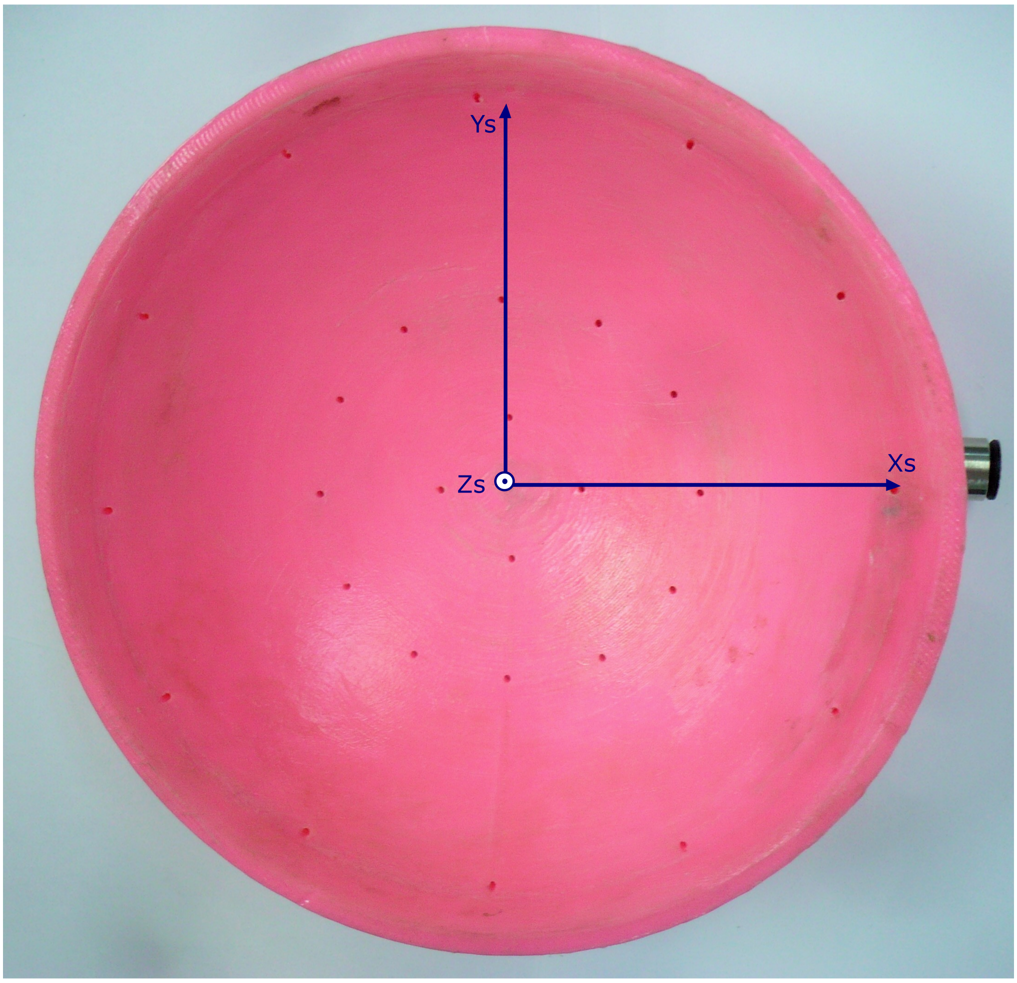


Figure 2.7 – *Fixed Reference System (FRS)*

because is the system of reference that is in continous motion and the mass of the system is inside that range.

The equations will be represented in the vector form, which provides us with the direction and magnitude.

According to the Newton's second law of motion, when a force acts on a mass, an acceleration is produced. With that statement, it is defined the following equation:

$$\bar{T}_{ext} = I\dot{\bar{w}} = \dot{\bar{H}} \tag{2.4.1}$$

Where, if we state that the mass is represented by its moment of inertia (MOI), I, under the acceleration of that mass is the angular acceleration, $\dot{\bar{w}}$, a torque $\bar{T}_{ext}$ is produced in the object, is also equal to the change in angular momentum $\dot{\bar{H}}$.

That affirmation can be explained with an example. The figure 2.11 shows a wheel which has a force tangential to it, which produces a torque along the axis (torque in that case

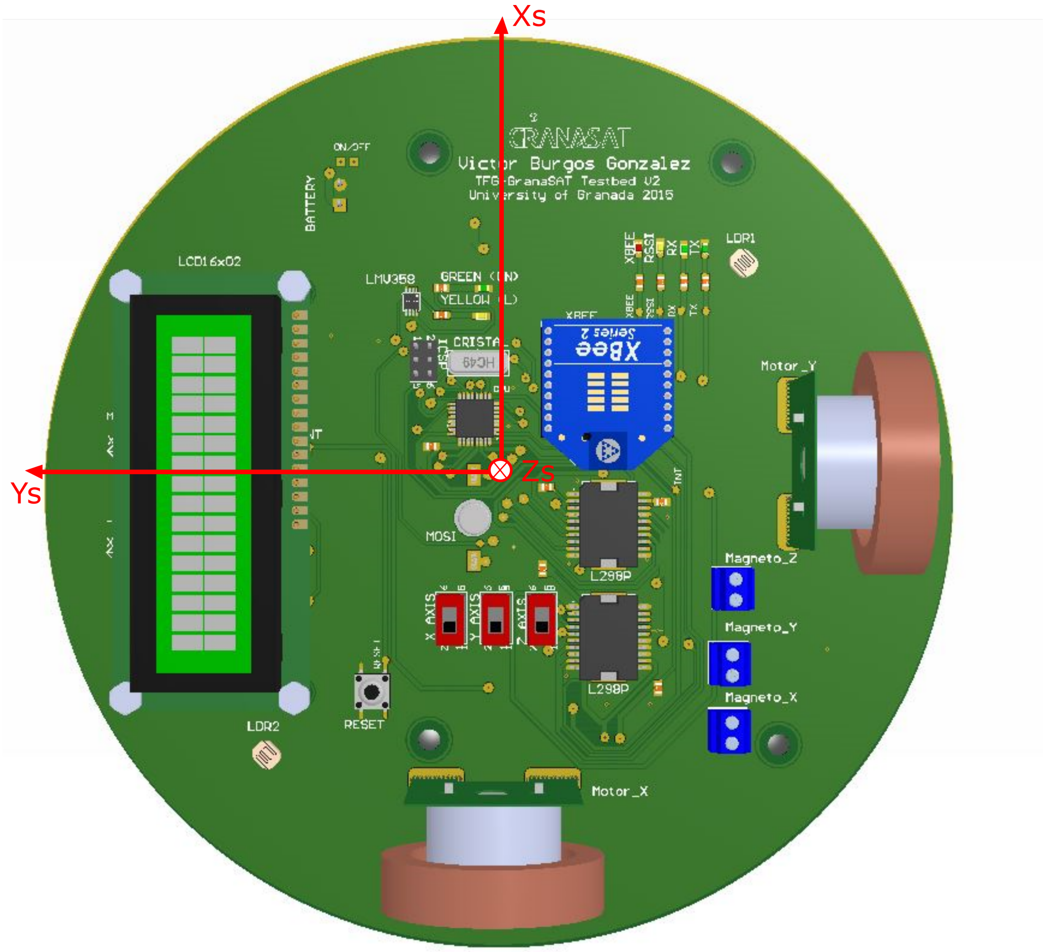


Figure 2.8 – *Sphere Reference System (SRS)*

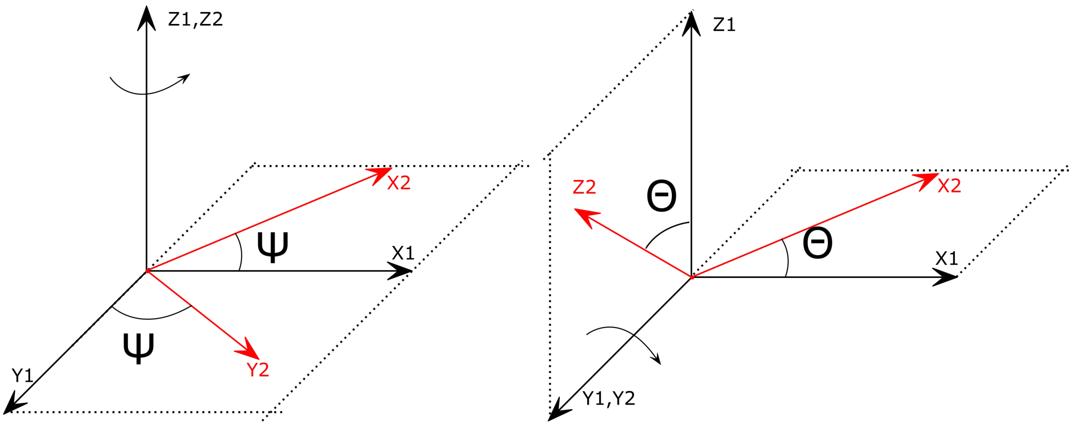
is represented with τ), following the right hand rule.The change in angular momentum (L) with the direction along the axis, will produce the increasing of the angular velocity of the wheel. However, if the angular momentum is perpendicular to the axis of the wheel, the effect will change the direction but not the magnitude of the velocity.

The equation 2.4.1 is taking the fixed inertial reference, because there is no parameters that represent the movement of SRS with respect to the FRS. The Equation 2.4.2 shows the additional part $\bar{w} \times \bar{H}$ for the 2.4.1:

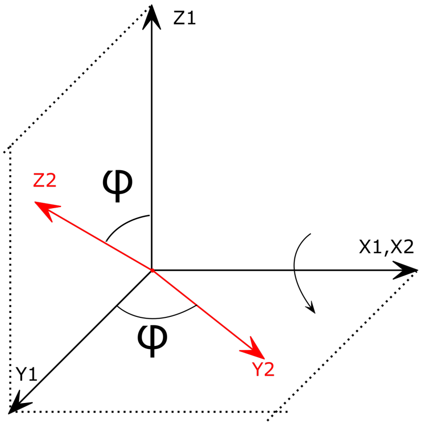
$$\bar{T}_{ext} = \dot{\bar{H}} + \bar{w} \times \bar{H} \tag{2.4.2}$$

Now the $\bar{w}$ and $\bar{w}$, do not represent the change in that parameters, but only the vector of each one. But there are another generation of angular momentum in our system, the actuators.

$$\bar{H}_{total} = \bar{H}_{AB} + \bar{H}_{RW} \tag{2.4.3}$$



(a) *Yaw* **(b)** *Pitch*



(c) *Roll*

Figure 2.9 – *Euler angles definitions*

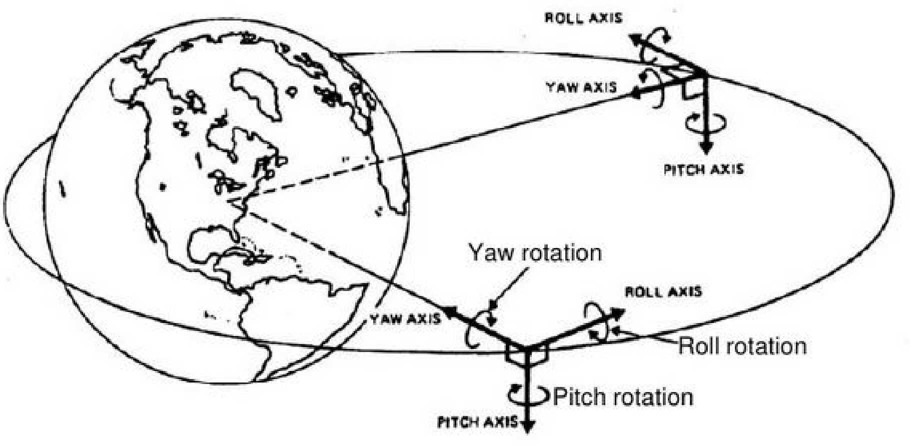


Figure 2.10 – *Euler angles in a satellite [15]*

In the previous equation, we have three terms, the first one is the angular momentum of the Air-bearing, and the $\bar{H}_{RW}$ is the angular momentum of the reaction wheels. The sum of the two terms result in the total amount of angular momentum of the system. But now,

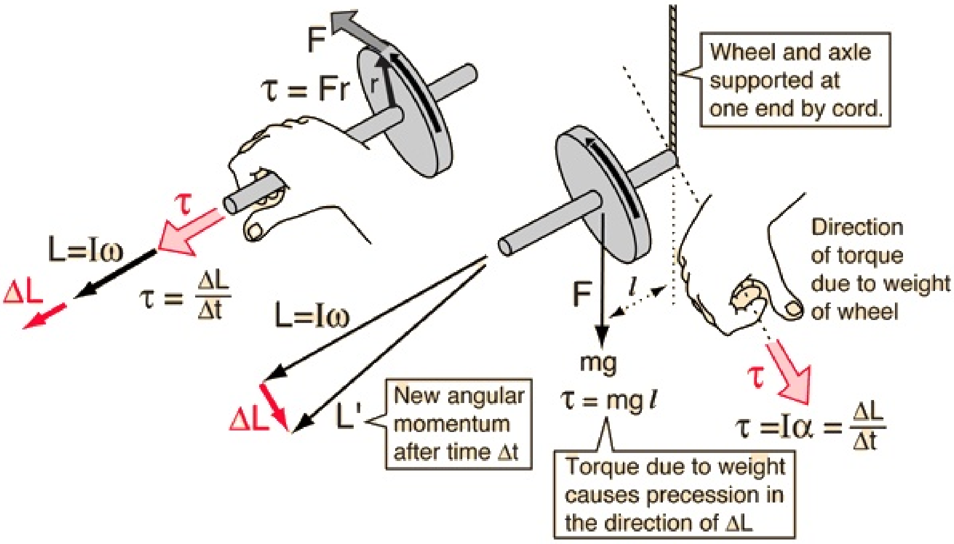

F
τ = Fr
r
τ
L=Iω
ΔL
τ = ΔL/Δt
L=Iω
ΔL
L'
New angular momentum after time Δt
Wheel and axle supported at one end by cord.
F
l
mg
τ = mg l
Torque due to weight causes precession in the direction of ΔL
Direction of torque due to weight of wheel
τ
τ =Iα = ΔL/Δt

Figure 2.11 – *Angular momentum example*

occurs as before, we have a equation where the motion is not included. In order to take account that motion, the equation 2.4.3 is substituted in 2.4.2:

$$\bar{T}_{ext} = \dot{\bar{H}}_{total} + \bar{w} \times \bar{H}_{total} \tag{2.4.4}$$

$$\dot{\bar{H}}_{total} = \dot{\bar{H}}_{AB} + \dot{\bar{H}}_{RW} \tag{2.4.5}$$

Substituting Equation 2.4.5 and Equation 2.4.3 in Equation 2.4.4:

$$\bar{T}_{ext} = \dot{\bar{H}}_{AB} + \dot{\bar{H}}_{RW} + \bar{w} \times (\bar{H}_{AB} + \bar{H}_{RW}) \tag{2.4.6}$$

Now back to the first equation, where the change of angular momentum is equal to inertia times angular acceleration. If the equation 2.4.1 is substituted in Equation 2.4.6, you get the following:

$$\bar{T}_{ext} = I_{AB}\dot{\bar{w}} + DI_{RW}\dot{\bar{\Omega}} + \bar{w} \times (I_{AB}\bar{w} + DI_{RW}\bar{\Omega}) \tag{2.4.7}$$

Equation 2.4.7 shows the complete vector equation of rotational motion in Air-bearing system, where Ω is used to difference the angular velocity of the reaction wheels. D is the direction cosine matrix between the reaction wheel coordinate system and the SRS, but in our case, both reference system are the same, so that matrix has the unit value because the cosine of zero is one.

The values of I_{AB} and I_{RW} are constants. As the SRS is aligned with the principle axes of the Air-bearing, the inertia matrix I_{AB} is a diagonal matrix:

$$I_{AB} = \begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix} kg * m^2 \tag{2.4.8}$$

On the other hand, the reaction wheel inertia matrix, I_{RW}, which include the rotating portions of the flywheel and the motor. Again, we have a diagonal matrix, because of the alignment of its reference system:

$$I_{RW} = I_{flywheel} + I_{motor} = \begin{pmatrix} I_a & 0 & 0 \\ 0 & I_a & 0 \\ 0 & 0 & I_a \end{pmatrix} kg * m^2 \tag{2.4.9}$$

The values of that matrix will be seen in the chapter 3, calculating the values for the flywheel depending on the mass and the geometric parameters and the motor depending on the characteristics described.

The desired parameter of that equation is the angular acceleration of the Air-bearing, $\dot{w}$, so is necessary take out that parameter of the equation 2.4.7. The reaction wheels do not provide external torques, and we assume that there are no more disturbance torques.

$$\dot{\bar{w}} = I_{AB}^{-1}[-(\bar{w} \times (I_{AB}\bar{w}) - I_{RW}\dot{\bar{\Omega}} - (\bar{w} \times +I_{RW}\bar{\Omega})] \tag{2.4.10}$$

The vector $\dot{\bar{w}}$ will give us the value of the three axis of the Air-bearing angular velocities.

2.5 System Requirements

As we have described before, the Testbed for the Cubesat GranaSAT will be a improved tabletop style, which will have some constraints in the pitch and roll axes, in order to achieve the objectives of this project, mentioned in the Section 1.3 of the Chapter 1. The Testbed will be available for the students who need it, so it should be prepared for that issue.

2.5.1 Communication System

It is required a link between the Ground Station and the ADCS Testbed. This link should have the following characteristics:

- Bidirectional option. It is necessary the communication between the GS and the Testbed in two ways because the use of the telemetry (transmission of the data from the sensors in the Testbed to the PC) and the telecontrol (orders transmission from the GS to the Testbed's actuators).
- Bandwidth. The communication device should provide the sufficient bandwidth to

transfer the data, with a minimum of the Cubesat standard.

- Reliability and efficiency. The Testbed requires a efficiency communication system, which enhance the efficiency in the power consumption too.
- The parameters measured with the sensors should be represented in the GS in real time.

2.5.2 Testbed PCB and electronic devices

The designed PCB should have some requirements for a correct operation. That PCB will include the sensors, a microprocessor, the actuators and their drivers, and the power system.

- The PCB power system should provide the required voltages to supply every system on it.
- That power system, will be battery powered because the Testbed is a wireless system, without any cable.
- The battery must have the maximum capacity to turn on the system as long as possible, in the minimum size conceivable.
- The sensors have to offer the maximum accuracy for the measurements, and a compatible communication with the microprocessor.
- The microprocessor ought to permit a correct communication with the GS, and it should have the necessary ports for every sensors, drivers, etc.
- The drivers for the motors have to support the maximum current measured for the motors, which should be so powerful that they faced with the motion necessary to move the Testbed.
- The PCB should have the minimum consumption of energy as possible, because the power supply is a battery and it have to save as much as it can. Moreover, the cost have to be reduced to the minimum possible too.
- The PCB have to fit with the semisphere taking into account that the weight have to be as light as possible, because it will enhance the possibility of the proper functioning.

2.5.3 Mechanical parts

In this project, one of the most important parts is the mechanical design and implementation, because of its difficulty and the constraint, the Air-bearing have to generate a friction-less surface with the Testbed. The main requirements are:

- The designed part for the Air-bearing have to perfectly fit in order to have the less friction between parts when the compressed-air was activated. To enhance the design of these parts, both surfaces should be polished.
- The design of the Air-bearing have to be reduced in cost as much as possible. The Air-bearings available in the market are expensive and we need to improve the design to enhance that problem. In Chapter 3 a comparison is done to choose the best solution.
- The compressed air should have the minimum pressure to move the hemisphere and save the maximum quantity of air in the compressor.

2.5.4 Software

It is necessary a software that allows the functions of telemetry and telecontrol for the ADCS Testbed. So, the software design has three subsections: telemetry, telecontrol and software on board.

2.5.4.1 Telemetry

- Port control in the GS to receive the data.
- Divide and decode the packages received to organize the data in the PC of the GS.
- An error algorithm to ensure that every data is received correctly.
- Real time display of the data received.
- The option of save or load (from a previous session) the data received.
- The system shall provide appropriate viewers for the user to read the data collected (use of graphics, for example).

2.5.4.2 Telecontrol

The system should be prepared for the telecontrol from the GS to the Testbed. That option should be optional, to optimize the data transmission bidirectionally. It will have the same characteristics and requirements than telemetry if we used the same protocol to upload and download the data. The telecontrol must operate to the software on board to change the desired parameters to control the attitude of the ADCS Testbed.

2.5.4.3 Software onboard

- The software onboard must operate to change the system on the Testbed with the microprocessor that the PCB will have.

- In order to have measurements in real time, the algorithm should be as efficient as possible.
- The microprocessor will have to read the data and calculate the outputs for the actuators in the minimum necessary time.

CHAPTER

3

SYSTEM ANALYSIS

In this chapter, it will be analyzed the most relevant parts of the Testbed related with this project. A comparative study will be done in order to select the best option for each part, as much to electronics as to mechanical design. That chapter will be essential to have a proper design, presented in chapter 4.

The first part that will be studied is the Testbed Technology, in order to choose the best mechanical design for the Air-bearing. Later, the electronics parts will be analyzed and finally some issues for the mechanical part, and a resume of the communication system.

3.1 ADCS Testbed Technology

Nowadays, the Testbed technology are in a constant growth. The need to have a low-friction environment, in order to simulate the aerospace environment, is the first condition that the Testbed has to fulfill. With that condition, the subsystems mounted in it, will be tested with a high precision, even with the constraint of the gravity that we have in the Earth. That is why the simulation of that kind of circumstances is not trivial.

There are different types of simulations to achieve the torque-free environment, to have a friction-less ambient simulating the aerospace conditions.

The first method consist of dipping the satellite or its subsystems into a tank of water. The first problem that appears with this method, is the covering that the satellite has to have to ensure that the water does not enter into the system, and because of that encapsulation, the gases generated inside the experiment does not get away.

Another method, but not too common, is the use of drop towers. For that manner of working, you will need a soft net (in some cases a solid like sand) in the end of the tower, to avoid that the satellite being damaged. Moreover, some high-speed cameras are used to take pictures while the satellite is descending, in order to analyze the attitude control. [9] Sometimes, that drop towers are used with the intent to create microgravity(see Figure 3.1). [41]

Figure 3.1 – *Drop Tower NASA [41]*

The third method, and the most common in the aerospace field, is the use of an Air-bearing to create that low-friction atmosphere. That solution can not create a no-gravity environment, but it has a nearly free torque condition. The function of that Air-bearing is to eject air under pressure through some small holes placed on a surface, normally spherical, to generate an air layer between the Air-bearing and the rotor of the Testbed platform. There are two main types of Testbed for the satellites: Planar Air-bearings and Spherical Air-bearings.

- **Planar Air-bearings**.

  One of the most important characteristics of a Testbed, is the degrees of freedom (DOF) that the Air-bearing have. Planar Air-bearing could have one or two DOF(only **translational** type). For that case of Air-bearing, the body that are being tested, carry its own cushion of air hovering upong a polished surface [9].

  An example of that kind of Air-bearing is in the Standford University's Aerospace Robotics Laboratory (ARL), used for the test of robotics for on-orbit construction (see Figure 3.2). The experiment consist of two-link manipulator with two mini arms mounted at its end-point [47].

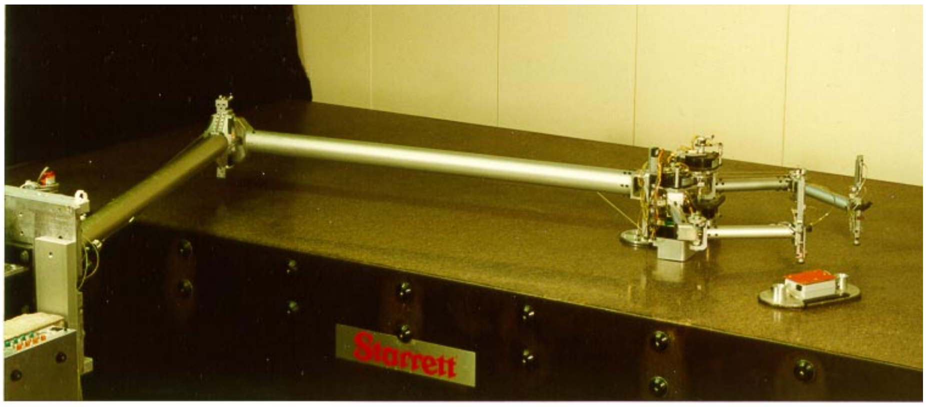


Figure 3.2 – *ARL Testbed (planar Air-bearing)[47]*

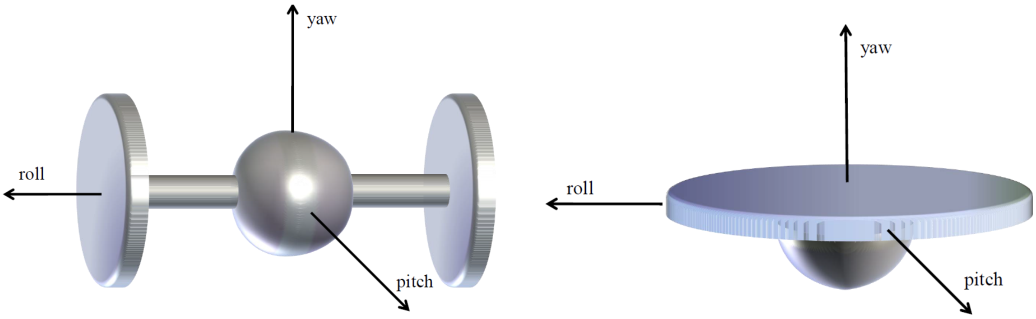
- **Spherical Air-bearings**. Spherical Air-bearings operate rotating a spherical or hemispherical object above a concave structure using compressed air or gas, and perfectly matching with the sphere or hemisphere. In contrast to the planar Air-bearings, which have translational DOF, the spherical Air-bearings have three **rotational** DOF.

  The angular constraint of the Testbed depends on the size, shape and type of Air-bearing. The three different types of spherical Air-bearing are Dumbbell, Tabletop and Umbrella (see figure 3.3).

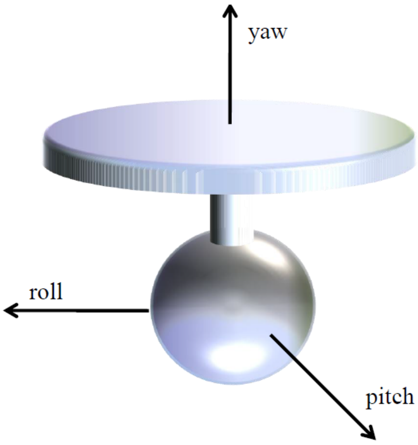
  The description and some examples of each one is presented below:

  - *Dumbbell.* That type of Air-bearing offset the COR and extend two arms, one per side. Dumbbell Testbed allow the 360 of freedom in the yaw and roll axes (see figure 3.3a). An example of that style of Testbed is shown in figure 3.4, an Air-bearing of the University of Michigan known as the Triaxial Attitude Control Testbed (TACT). In that figure, you can see the different actuators that the TACT has: reaction wheels, fans and mass actuators [46].
  - *Tabletop.* In this Testbed, the table where the different subsystems are located, is mounted above a hemisphere directly. With that type of Air-bearing is allowed the 360 motion in the yaw axis (see figure 3.3b), but a limited motion in the pitch and roll axes. The advantages of this solution are the inexpensive fabrication that it has and the little space that it requires. Virginia Polytechnic Institute and State University have built a satellite ADCS Testbed with the use of two spherical Air-bearings: a tabletop style in each side of a dumbbell Air-bearing [48]. Figure 3.5 shows the tabletop air-bearing with the dumbbell part of the Air-bearing base in the background of the photo.
  - *Umbrella.* The last type of spherical Air-bearing is an improvement of tabletop. In order to have a better freedom motion in axes pitch and roll, a rod from a fully spherical bearing is placed, permitting more DOF. As tabletop Testbed, it has a 360 of freedom in the yaw axis, and a bigger table can be mounted on it, without the loss of freedom motion (see figure 3.3c)3.3

The GranaSAT Testbed will be designed as a improvement of tabletop system. Rather



(a) *Dumbbell* **(b)** *Tabletop*



(c) *Umbrella*

Figure 3.3 – *Spherical Air-bearings Styles [9]*

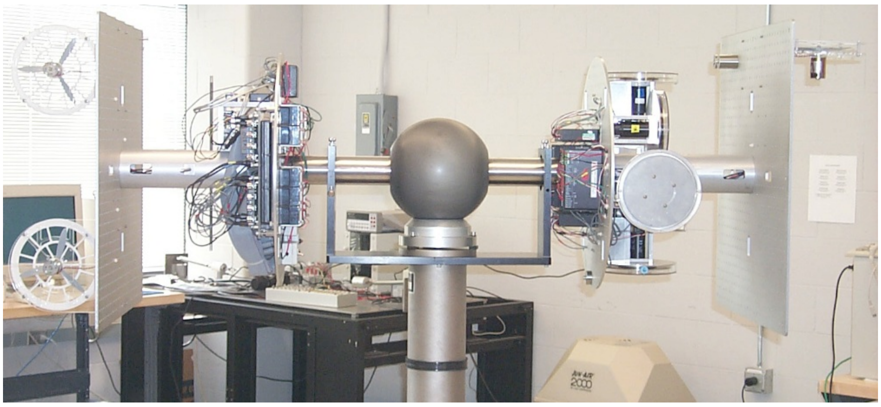


Figure 3.4 – *TACT Dumbbell Style Air-bearing [46]*

than have the subsystems in a table outside of the hemisphere, we have include the subsystems inside of the hemisphere, in order to have more DOF in axes pitch and roll, and less weight discarding the table that tabletop or umbrella have (see figures 3.3c and 3.3b). The design of that Testbed will be fully covered in the next chapter. An example of that kind of Testbed was designed in Universität Würzburg, FloatSat (see figure 3.6).

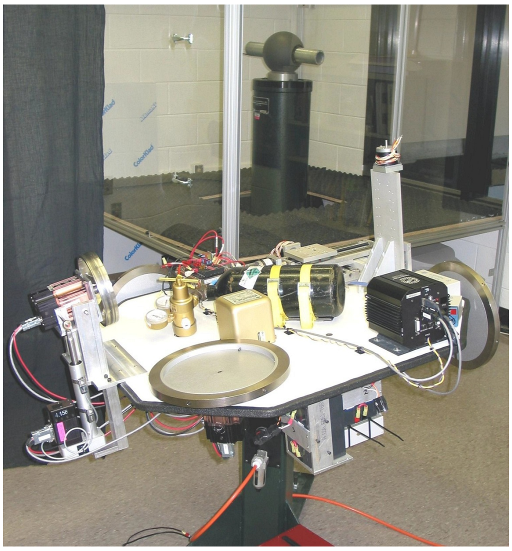


Figure 3.5 – *DSACSS Air-bearing Set [48]*

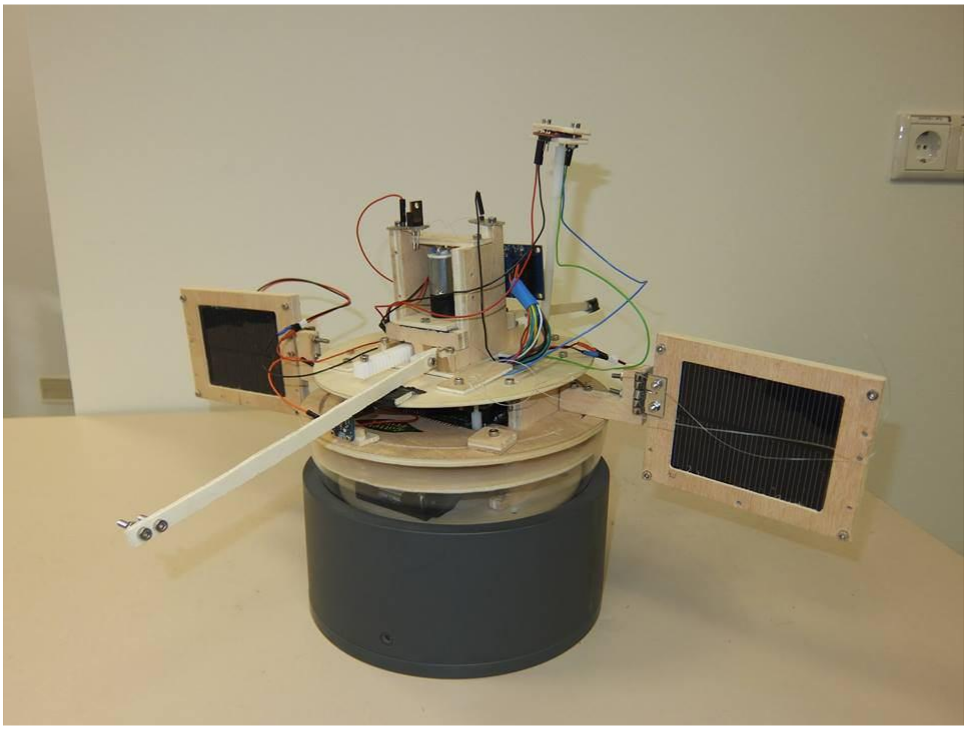


Figure 3.6 – *FloatSat project [25]*

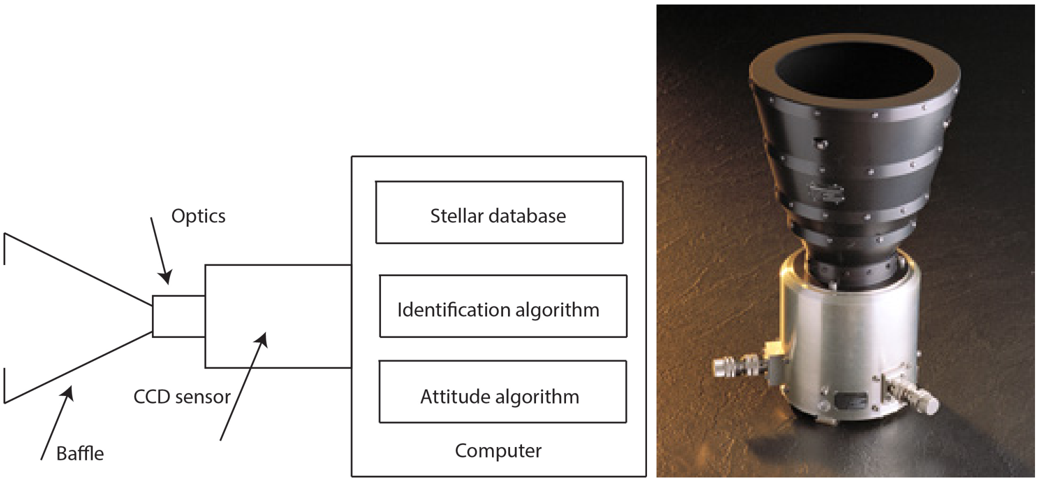
3.2 Sensors for the ADCS

In order to take measures of the actual state of the system, some sensors should be used in the ADCS (see figures 2.3 and 2.4, where a diagram of that kind of systems is represented).

ADCS will use different ways of measurements to acquire the attitude of the system, depending of the kind of sensors, that attitude will be relative or absolute, but the most used are the relative sensors, which use external references to orientate the whole system.

Some of the most common references used by the satellites are from the Sun, stars, Earth's horizon, Earth's Magnetic field, and some inertial measurements as for example the angular velocity of the satellite, in order to obtain the attitude when the external references are not available.

- **Star Sensor**. Also known Star tracker, is an optical device (photocells or a camera) that measures the position of a star group from the sky. The star tracker uses a pattern recognition of the star constellations in the Field of view (FOV) and by means of a star catalogue covering the whole firmament. The orientation of the satellite can be determined finding that pattern by comparing the optical detection (for example an image) with that catalogue. That reference is one of the most accurate (see table 3.1) because they use the fixed stars. The figure 3.7 shows the diagram and a photo of a commercial star tracker.



(a) *Star tracker diagram [39]*

(b) *Commercial Star Tracker - CT-633 [4]*

Figure 3.7 – *Star tracker*

Because of its dimensions(in some cases) and the need of a big quantity of data for the catalogue, that kind of sensor is not very common in Cubesats.

- **Horizon Sensor**. That kind of sensor measures the difference between the darkness of space and the light bouncing of the Earth. In other words, finds the Earth's horizon thus the orientation relatively to the blue planet. Its problem is the low accuracy. Now in the table 3.1 there will be a better comparison with the other sensors, where that affirmation will be confirmed.

  Depending on the kind of the horizon sensor, it uses an infrared diode and a lens or a camera, and it can be a scanner or horizon crossing indicators. Due to size, weight and complexity the last one are considered the most suitable for the Cubesat. Moreover, for the satellites, it has to be used together with other attitude sensors.

The figure 3.9 shows a capture of the GS representing the horizon sensor measurements of the GranaSAT BEXUS Team.

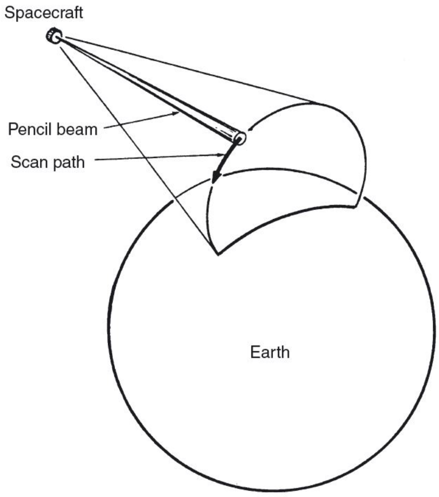
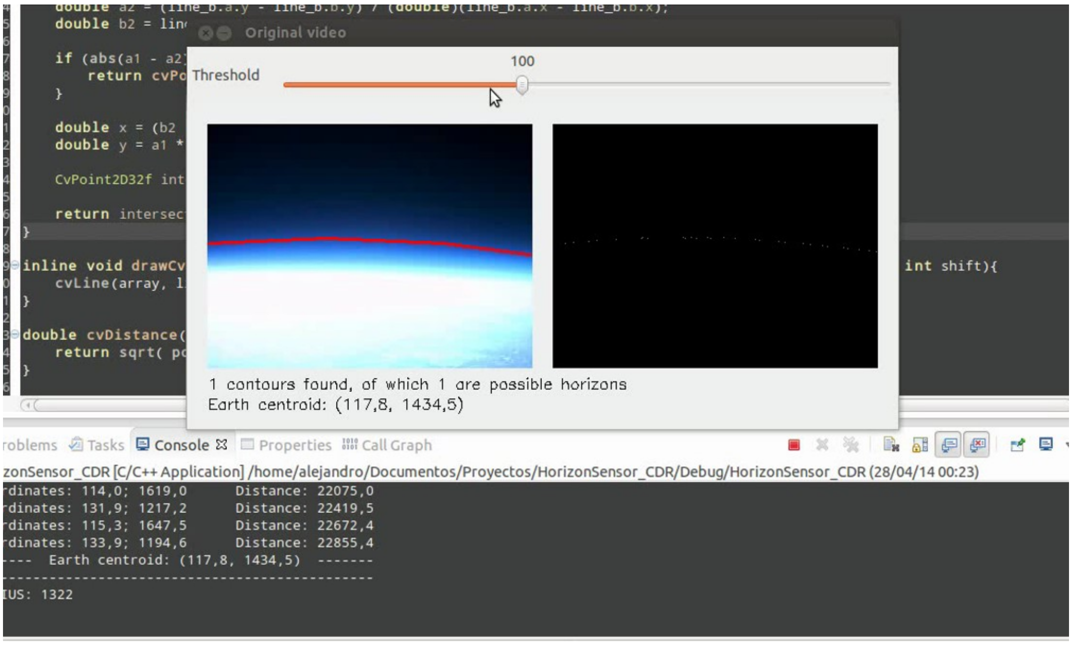


Figure 3.8 – *Horizon Sensor scheme [59]*


```
    double a2 = (line_b.a.y - line_b.b.y) / (double)(line_b.a.x - line_b.b.x);
    double b2 = lin

    if (abs(a1 - a2
        return cvPo
    }

    double x = (b2
    double y = a1 *

    CvPoint2D32f int

    return intersec
}

inline void drawCv                                                        int shift){
    cvLine(array, l
}

double cvDistance(
    return sqrt( po
}
```

1 contours found, of which 1 are possible horizons
Earth centroid: (117,8, 1434,5)

```
zonSensor_CDR [C/C++ Application] /home/alejandro/Documentos/Proyectos/HorizonSensor_CDR/Debug/HorizonSensor_CDR (28/04/14 00:23)
rdinates: 114,0; 1619,0      Distance: 22075,0
rdinates: 131,9; 1217,2      Distance: 22419,5
rdinates: 115,3; 1647,5      Distance: 22672,4
rdinates: 133,9; 1194,6      Distance: 22855,4
----  Earth centroid: (117,8, 1434,5)  -------
-------------------------------------------
IUS: 1322
```

Figure 3.9 – *Horizon Sensor GranaSAT BEXUS [53]*

- **Sun sensor**. The Sun sensor is used for providing a vector measurement of the Sun radiation using photodiodes, photoresistors, solar cells etc. The most relevant problem that the sun sensor has is they need direct sunlight and the solar cells should be replaced by the sun sensors(less power for the satellite). The other problem is the need of the sun, because when the sun is hidden, that sensor has no functionality (see figure

3.10). Similarly, the same problem but with the opposite case, a reflective surface can damage the measurements (see figure 3.11). It can be analog or digital, but the most accurate is the second one [65]. The analog one works depending on the angle and the current flow, this is why is not accurate (1 of accuracy in FOV of 30).

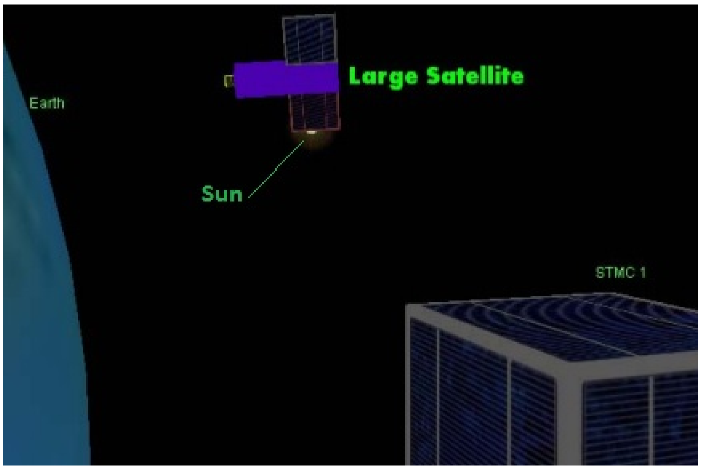


Figure 3.10 – *Possible shadow of a larger satellite [65]*

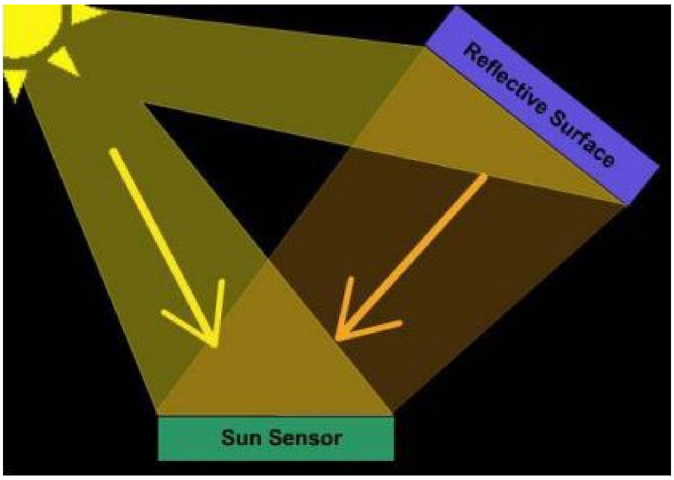


Figure 3.11 – *Possible reflection in the sun sensor [65]*

- **Magnetometer**. It is a sensor which measures the Earth's field. In Low Earth Orbit LEO, where the magnetic field of the Earth is the sufficient strong and 3D magnetometer will provide a proper attitude determination. there are three types: fluxgate, magneto-resistive and magneto-inductive. They should be well calibrated in order to have the control about the field generated within the Cubesat, in that case, Testbed. If the sensor is perfectly calibrated, that option is one of the best because of its good accuracy. The measurements are compared with a model of the Earth's magnetic field (database). They will be really useful because of the use of magnetorquers in our project.

- **Accelerometer**. It is a inertial sensor, and measure translatory accelerations. The MEMS inertial accelerometers consists of a mass-spring system in a vacuum. When it is exercised an acceleration on the sensor, the mass in the spring system is displaced. The description of that functionality is shown in the figure 3.12, a sensor capacitive with MEMS technology. When the mass is moved of its original position, the capacitance changes and, then, the voltage too.

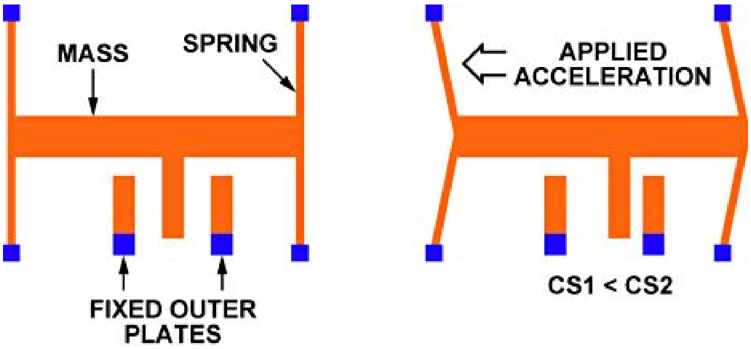


Figure 3.12 – *MEMS accelerometer[45]*

- **Gyroscope**. That kind of sensor will measure the angular velocity of the body about a specified axis of rotation, in this case the Testbed, without the need of a external reference. The detumbling phase that sensor would be really relevant. The detumbling phase is produced when the satellite is launched from the rocket and it starts to travel around the orbit. In that moment, the Cubesat starts to turn on itself with a high angular velocity, so the objective of that phase is to stop that turn, stabilize it, in order to take a proper attitude determination. The stabilization is made thank to the actuators of the system. But in our case, the Testbed will have a manual rotation in order to simulate that situation. The figure 3.13 shows how a MEMS gyroscope can be modeled in a easy way.

Now it is time to get a comparison table of the sensors described above(table 3.1), in order to select the best option for our system.

Type	**Initial attitude acquisition**	**DOF**	**Accuracy**
Magnetometer	Yes	3	1 arcminute
Radio Frequency beacon	Yes	2	1 arcminute
Horizon sensor	Yes	2	5 arcminutes
Sun sensor	Yes	2	1 arcminute
Solar panel	Yes	2	1
Star tracker	Yes	3	1 arcsecond

Table 3.1 – *Comparison of some kind of sensors [34]*

Because of the Cubesat limitations in size and weight, the sensors chosen are the magnetometer, accelerometer and gyroscope. Each sensor will be used in MEMS technology, in

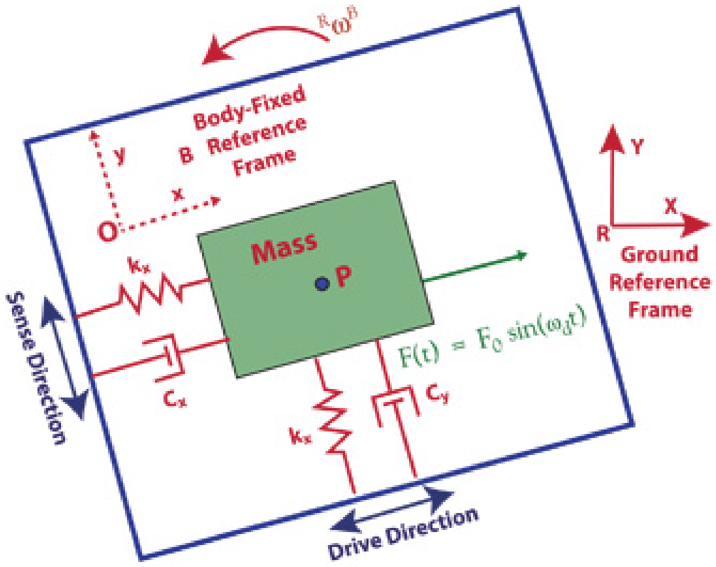


Figure 3.13 – *MEMS gyroscope*[13]

order to have the smallest size possible. Because the studied design for the Testbed, maybe it will be used for the Cubesat if it is possible.

Gyroscope Choice The following table 3.2 presents the comparison of three different 3D gyroscopes MEMS technology, but for the Testbed, it will be used the **test board** of each one, in order to be used it in the breadboard for the simulations and tests, and to have an easier soldering process. However, the future Cubesat will have only the chip sensor.

Parameter	**L3G4200D**	**MPU6050**	**L3GD20H**
Accelerometer incorporated	No	Yes	No
Voltage Supply[1](V)	5	5	5
Temperature range (C)	-40 to 85	-40 to 85	-40 to 85
Scale range (/*sec*)	±250 ±500 ±2000	±250 ±500 ±1000 ±2000	±245 ±500 ±2000
Sensitivity (*mdps*/*digit*)	8.75, 17.50, 70	7.6, 15.2, 30.4, 60.8	8.75, 17.50, 70.00
Output data rate (Hz)	100-800	4-8000	11.9-757.6
Communication protocol	I2C	I2C	I2C
Consumption at 3.3V (mA)	6.1	3.9	5

Table 3.2 – *Comparison of gyroscopes* [31][37][38]

In conclusion, the best option is MPU6050. The features that have not been compared are practically the same values. Moreover, that sensor includes an accelerometer, so it will no be necessary includes one more sensor for the accelerometer. In chapter 4, section 4.1.1.4 MPU6050 will be explained in more detail.

For the magnetometer, see the project of the magnetic part for the Testbed [57].

3.3 ADCS Actuators

To control the position of the Testbed, the ADCS generates torques thanks to the actuators. There are two kinds of actuators:

- **Passive actuators.** That kind of actuator does not require power or fuel, however, it can restrict control over the spacecraft's attitude. For that reason, it is not commonly used. For example:
  - Permanent magnets. They are passive electromagnets and can only align a satellite axis in one orientation relative to the Earth's magnetic field. That kind of actuator is permanent, so, for that reason, they normally are paired with other one.
  - Hysteresis rods. The Earth's magnetic field magnetizes the rods and they uses the hysteresis effect working against the spinning motion. See figure 3.14 where is represented that kind of actuator in a Cubesat and the permanent magnets too.

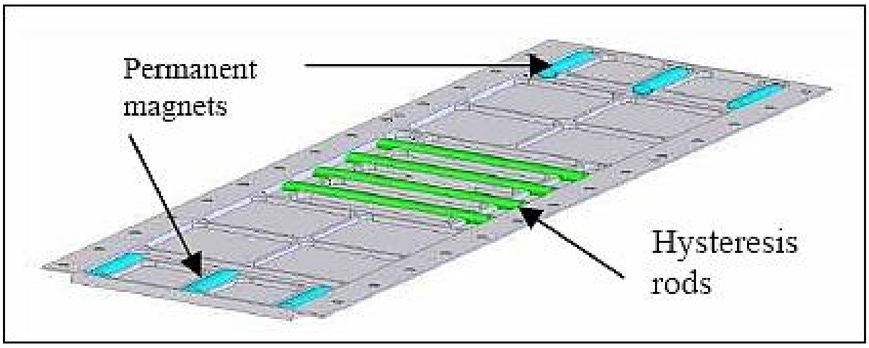


Figure 3.14 – *Permanent magnets and Hysteresis rods actuators - GeneSat-1 [1]*

  - Gravity gradient boom. That boom as a mass at the end of the boom which will be aligned with the center of the Earth.
- **Active actuators.** The opposite kind, active actuators requires fuel or energy to control, but in this case, the control is greater than passive types. For example: magnetorquers, reaction wheels and thrusters. The last one is not really used because of its fuel consumption, which add weight to the Cubesat and it is not recommendable. That type will be explained in more detail below.

For the Testbed, the group chosen is the active actuators, because they have a great control of the attitude. Now, inside this group, the thrusters is discarded, because it will need a determined quantity fuel to control them. Then, for the Testbed, the best two options are the reaction wheels and the magnetorquers, which will be described and we will choose the best materials and options for the Testbed application. In this project, the only actuator which will be studied is the reaction wheels, because the magnetorquers will be studied in the project where the magnetic part for the Testbed is treated [57].

Reaction wheels. Belongs to the active group and they are radial weights that are spun with the use of electric motors. The reaction wheel, will change the Testbed (or Cubesat) angular velocity, changing the angular velocity of the reaction wheel in the opposite direction. Normally there are three reaction wheels (aligned orthogonally), in order to have the control of the three dimensions. That actuator not interacts with external torques and it has a fast acting if the size is the correct. Because of the saturation problem, in order to decrease the stored momentum and required energy, other actuator is normally place with the reaction wheels, in our case, the magnetorquers.

The material of the wheel is the first item that is going to be studied. The table 3.3 shows the different material that we have available to buy, because of its prize and the stock in different stores in Granada.

Material	**Density** $(Kg/m^3$
Copper	8940
Bronze	8890
Iron	7870
Aluminium	2700

Table 3.3 – *Different materials for the wheel*

The best options depending on the density of the material are the copper and bronze, but finally the choice was **bronze** because of the availability in the diameter needed after of the calculation shown in section 4.2.4. The figure 3.15 shows the bronze bar bought. When the reaction wheels were manufactured, the outer surface was cleaned (see figure 3.16), anyway, in chapter 5 that process will be described.

Figure 3.15 – *Bronze bar for the reaction wheels*

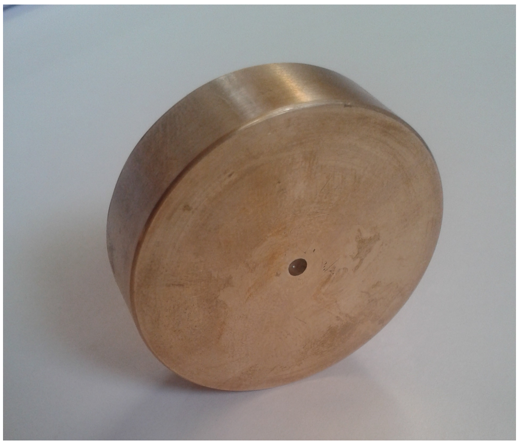


Figure 3.16 – *Manufactured wheel*

The next step is the selection of the motors for the wheels. Because of their sizes and the availability in the laboratory, we used DC Motors from CD players of the recycled PCs available in the Project's Laboratory of the ECTD, so the cost of that motors were zero.

Now, in the table 3.4 there will be a comparison of the motors found on that CD players.

Parameter	**Minebea**	**Mitsumi**	**Kysan**	**Kysan**	**Mabuchi**
Model	MDN3BT	M25E-4	RF-300CH	RF-300CA	RF-300C
Voltage range (V)	0.7-6.0	1.0-7.0	1.0-6.3	0.7-5.0	1.5-6.0
Rated Voltage (V)	2	5	3	2	3
No load speed (RPM)	2305	*	3000	2200	3500
No load current (mA)	16	*	18	18	22
Torque (mN-m)	0.392	1.47	0.42	0.27	0.48
Speed at torque (RPM)	1458	2750	2400	1600	2830
Current at torque (mA)	71	300	65	55	93

Table 3.4 – *Comparison of motors [12][20][40][36]*

Moreover, there where two more motors without name, model or brand, so they should be tested in order to take the characteristics. In chapter 5 there will be a characterization of the motors, with and without the wheel. The characteristics described in table 3.4 are practically similar, so we have to take in account other details: mechanical design, dimensions, connector, etc. And finally the chosen motors were Kysan (RF-300CH model), Minebea and one without name, because the connection and the sizes were the same, as described in figures 3.17 and 3.18.

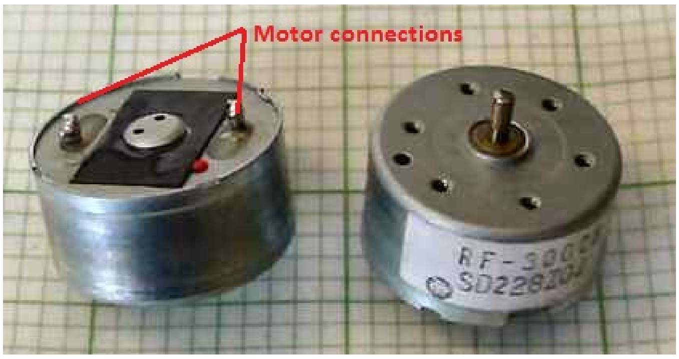


Figure 3.17 – *Motor connection*

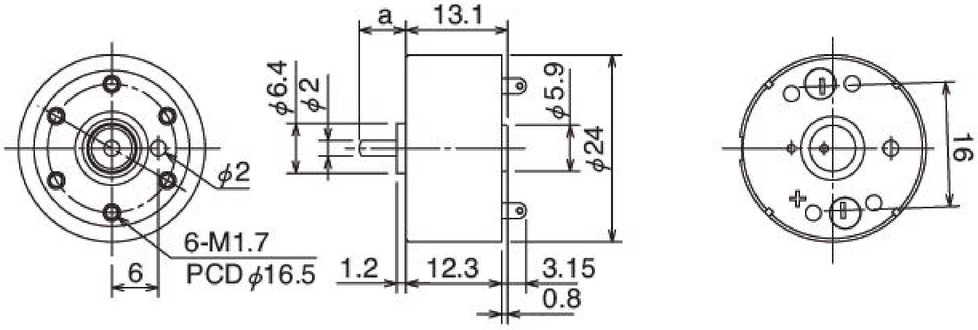


Figure 3.18 – *Motor dimensions [12]*

3.4 On-board microprocessor selection

In order to use the libraries provided by the sensors suppliers, we decided to use Arduino code, but instead of use a complete board of Arduino, the decision was take only the microprocessor AVR chip and the needed circuit components to work with it. Now, it is needed the selection of which microprocessor of Arduino is the best for our project, so the best way to select is comparing the main characteristics of some of them (see table 3.5).

The best option for the Testbed project is the ATMEGA328P-AU, because it has the necessary analog and digital pins and it is the cheapest one. That microprocessor is used in THT model in Arduino UNO, but we will use the SMD model to save weight and space in the PCB. In section 4.1.1.1 the microprocessor will be explained in more detail.

3.5 PID control

A PID (proportional-integral-derivative) controller is a control loop feedback that is commonly used in applications were is needed an industrial control for an actuator, in our case, the reaction wheels. That algorithm will calcule continuously the difference between the set-point (target angle in our case) and the measured angle. That controller has the intecnton to minimize the error between that two values and calculate the new one with the following equation:

Model	**ATmega2560**	**ATmega8-16AU**	**ATmega328P-AU**
SRAM (KB)	8	1	2
ROM (KB)	256	8	32
EEPROM (B)	4096	512	1024
IO Pins	86	23	23
Speed (MHz)	16	16	20
ADC-Bits	10-bit	10-bit	10-bit
PWM	16	3	6
Min Supply Volts (V)	1.8	2.7	1.8
Timers Counters	6	3	3
SPI	5	1	2
TWI (I2C)	1	1	1
UART	4	1	1
ADC channels	16	8	8
Ext Interrupts	32	2	24
Price (€)	17,12	3,72	3,41

Table 3.5 – *Comparison of microprocessors [7]*

$$u(t) = K_p * e(t) + \frac{K_i}{/Ti} \int_0^t e(\tau)d\tau + K_d T_d \frac{de(t)}{dt} \tag{3.5.1}$$

That equation has three parts, represented in the figure 3.19.

Where:

- e(t) is the error of the signal treated, so that is the differences between the signal objective r(t) and the signal that the plant produces y(t).
- u(t) is the controller output and the input to the process
- Kp is the proportional gain.
- Kd is the derivative gain.
- Ki is the integral gain.
- Ti is the constant of integral time.
- Td is the constant for the derivative time.

The diagram of the figure 3.19 has three parts, which are described now:

- The first block is for the proportional control (R), it has the product between the error signal and the proportional constant, with an error in stationary state even zero.

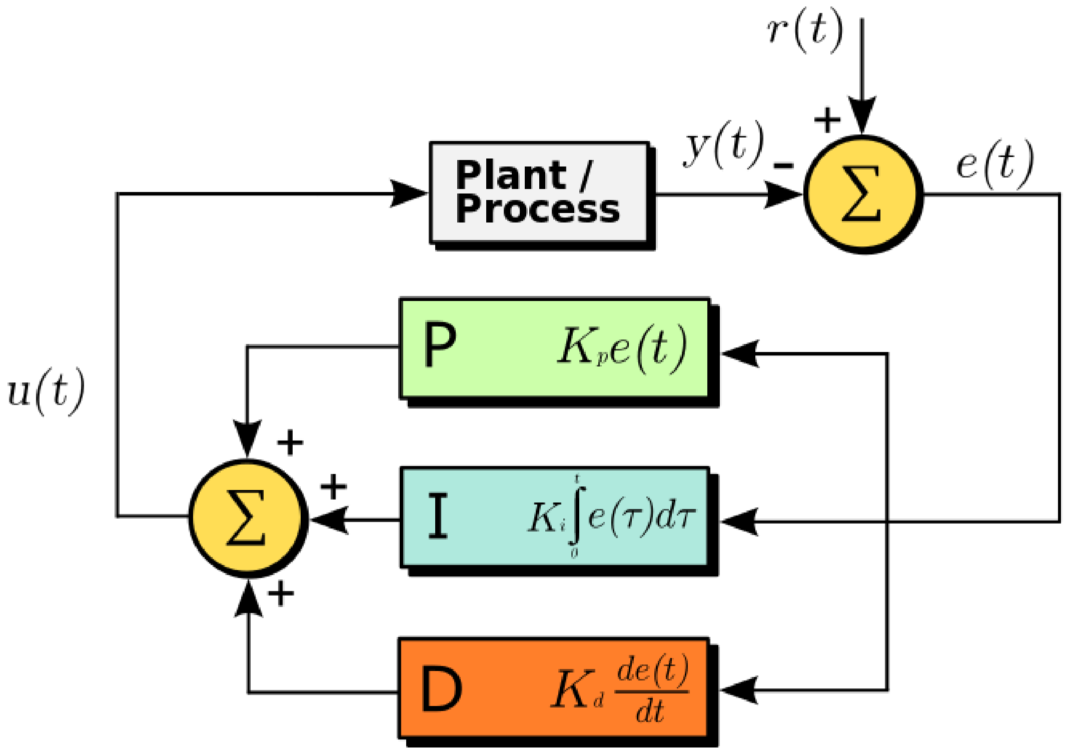


Figure 3.19 – *PID Diagram [64]*

- The second one is the integral control (I), which decrease and delete the error that proportional block generates. It acts as a deviation between the variable and the target point (integrating and add it to the proportional value).

- Finally the derivative control (D) consider the tendency of the error and allow a fast repercussion of the variable when the process perturbation have been done.

So, following the diagram of the figure 2.4 the system would have to works properly.

3.5.1 PID discretization

To use the PID control with the Testbed, it should be discrete, so the equation 3.5.1 should be write in the Z transform way:

$$U(z) = K_p * E(z)[1 + \frac{T}{Ti * (1 - z^-1)} + T_D * \frac{(1 - z^-1)}{T}] \tag{3.5.2}$$

Where T is our discrete sample time, we obtained:

$$\frac{U(z)}{E(z)} = a + \frac{b}{1 - z^-1} + z1 - z^-1 = K_p * [1 + \frac{T}{Ti * (1 - z^-1)} + T_D * \frac{(1 - z^-1)}{T}] \tag{3.5.3}$$

Where:

$$a = K_p; b = \frac{K_p T}{T_i} andc = \frac{K_p T_D}{T} \tag{3.5.4}$$

3.5.2 PID tuning

The tuning of the PID control is the election of the parameters a, b and c of the equation 3.5.4. There are many kinds of tuning for the PID, but, because of the characteristicis of the Tetsbed system, it has been chosen the Zieger-Nichols (ZN). That kind of tuning search values for P, I and D depending on the step response of the plant in open loop, as shows the figures 3.20 and 3.21.

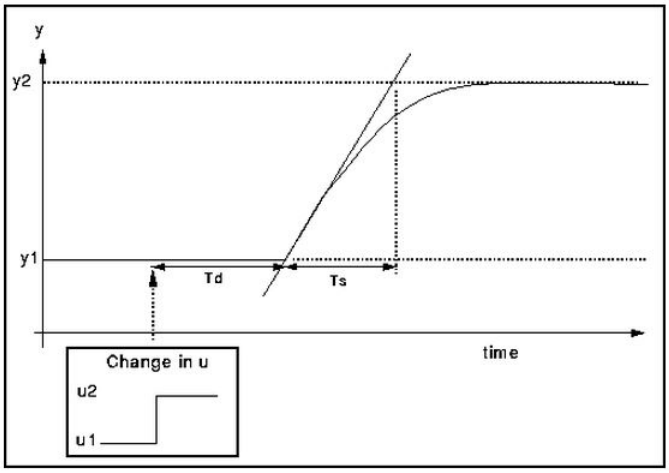


Figure 3.20 – *ZN Tuning (Ideal process) [43]*

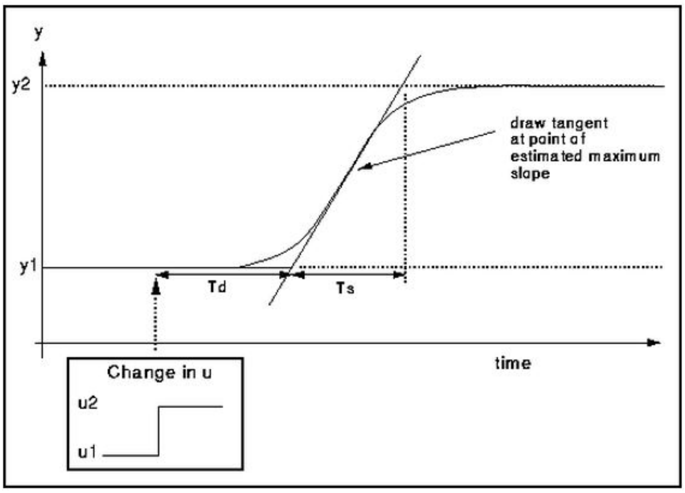
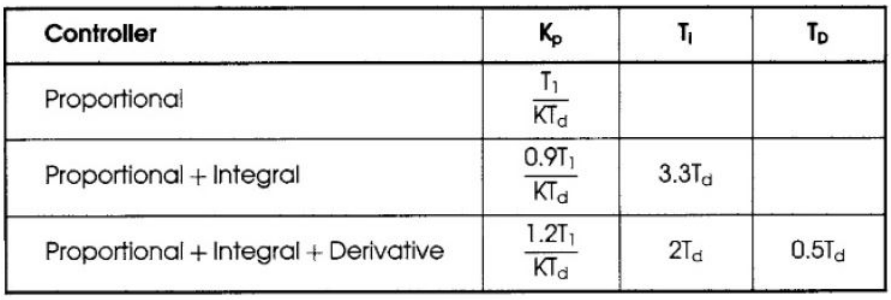


Figure 3.21 – *ZN Tuning (Real process) [43]*

Once the response is measured, is time to calculate the parameters using the table 3.6.



Controller	K_p	T_I	T_D
Proportional	$\frac{T_1}{KT_d}$		
Proportional + Integral	$\frac{0.9T_1}{KT_d}$	$3.3T_d$	
Proportional + Integral + Derivative	$\frac{1.2T_1}{KT_d}$	$2T_d$	$0.5T_d$

Table 3.6 – *ZN Tuning calculation [28]*

For our case, the angular velocity step response will be the decisive data to calculate the PID parameters. In chapter 6 is explained what happened with that calculation.

CHAPTER

4

SYSTEM DESIGN

In this Chapter of the project will be specified the details of the design of every subsystem. In the design of each part, we should apply the requirements given in the Chapter 2, in the Section 2.5. In order to have the best solution for every subsystem, in chapter 3 we have done an Analysis of other projects with different ideas.

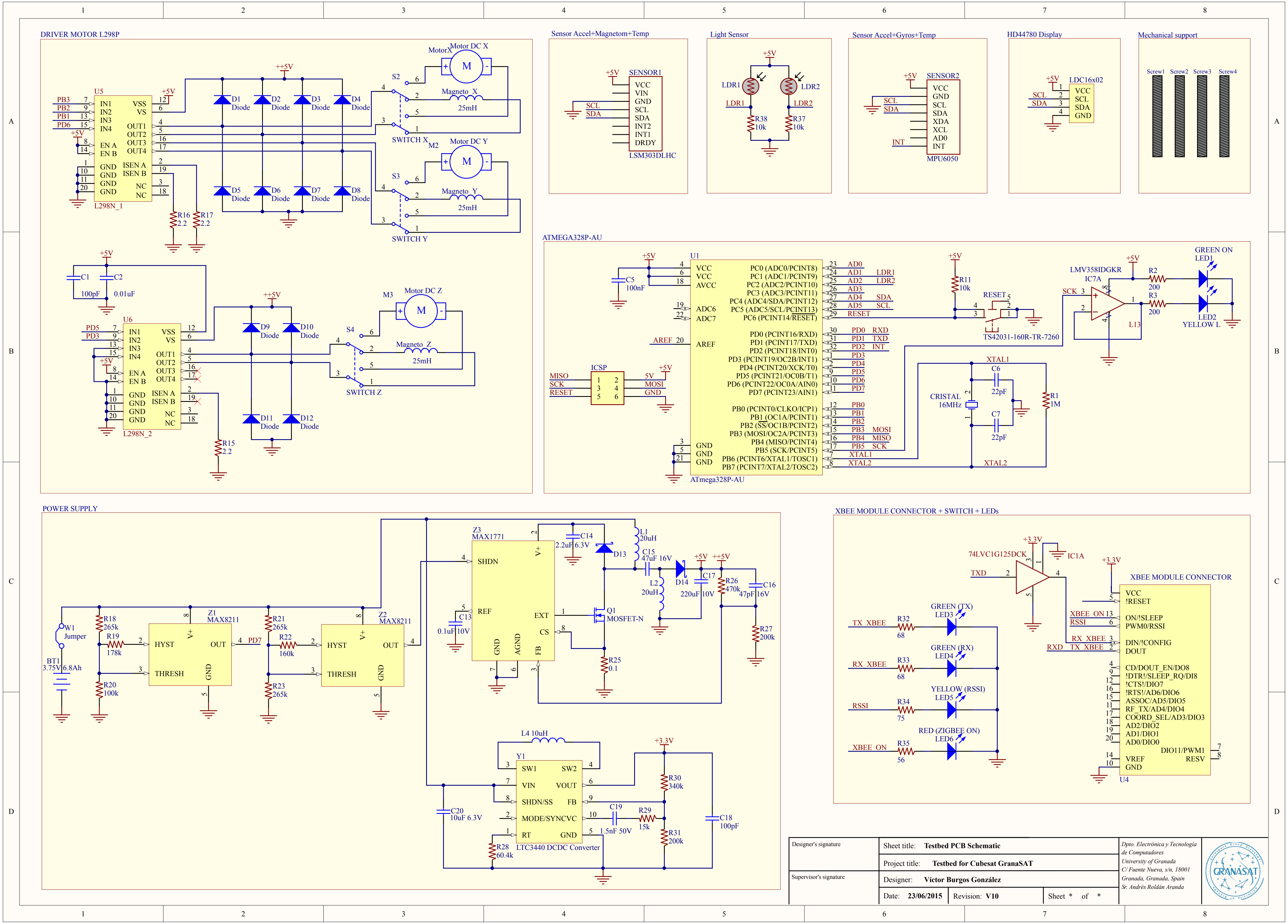
The Block diagram shown in the Chapter 2, in the Figure 2.1, shows every subsystem that the GranaSAT Testbed has. Now, we will see the design of the whole system, and the specifications of each part.

4.1 Electronic Design

The PCB of the Testbed will have the objective of connect the sensors, actuators, LCD to show how is working, the microprocessor that controls everything, and the power supply part. That PCB was designed in Altium Designer 14, a PCB design tool which include the Schematics design, PCB design and 3D model of the PCB. Last option is very useful for the study of the dimension of that PCB, because it can be used to prove if the design fits with the 3D model of the mechanical parts (see 4.2.5).

4.1.1 Electronic Schematics and components descriptions

In the following page is shown the schematic of the PCB, with every subsystem included on it. Then, we will see every part of the schematic designed in more detail.



Designer's signature

Supervisor's signature

Sheet title: **Testbed PCB Schematic**

Project title: **Testbed for Cubesat GranaSAT**

Designer: **Víctor Burgos González**

Date: **23/06/2015** Revision: **V10** Sheet * of *

Dpto. Electrónica y Tecnología
de Computadores
University of Granada
C/ Fuente Nueva, s/n, 18001
Granada, Granada, Spain
Sr. Andrés Roldán Aranda

4.1.1.1 Microprocessor

The microprocessor is from Atmel Corporation, model ATMEGA328P-AU. That microprocessor is the same as Arduino UNO citearduinoUNOweb, but in the Testbed board, it will be in SMD mode to save space and weight for the PCB. The pinout of this microprocessor is shown in Figure 4.1.

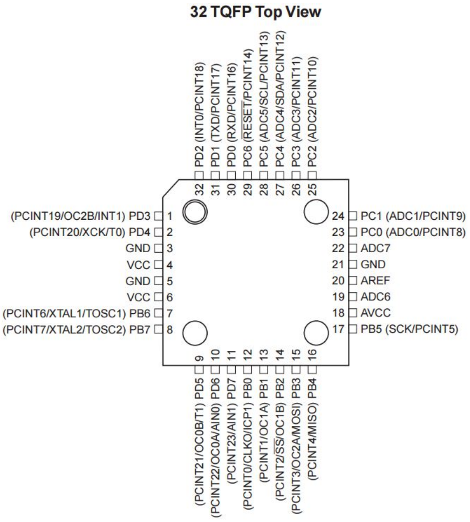


Figure 4.1 – *Pinout ATMEGA328P-AU*

The microprocessor will be the master for the communication with the sensors in I2C protocol, and the wiring of these devices is represented on Figure 4.15.

The schematic designed for the control of the microprocessor is shown in Figure 4.2.

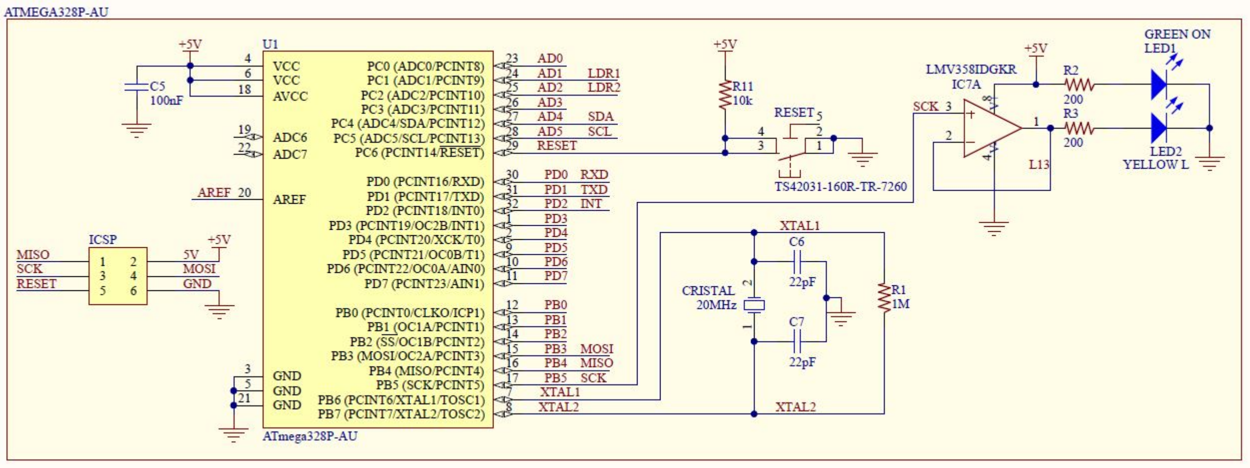


Figure 4.2 – *Schematic for ATMEGA328P-AU*

The first symbol that we can see in the left of the picture above, is the ICSP port. That port is used for the communication between a programmer and the microprocessor, with the objective of program it. The programmer used, as described in the Appendix A.2, is the USBtinyISP.

The microprocessor is programmed for use it as an Arduino UNO, so the external clock installed is of 16MHz. That clock is made with a crystal oscillator and their capacitors connected to the pins ROSC1 and TOSC2 of the ATMEGA328P-AU.

Moreover, to reset the Testbed, there is a button (shown in figure 4.3) with a pull-up resistor of $10k\Omega$.

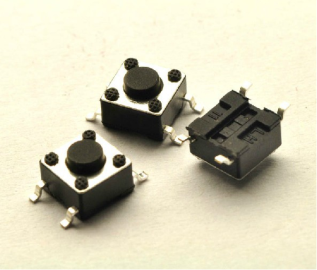


Figure 4.3 – *Reset button*

Finally, the last part of the schematic of the microprocessor is the SMD notificators LEDs:

- LED Blink (yellow color): One of the most common LEDs in an Arduino Board is the LED Blink, a LED connected to the pin number 13 of the Arduino. It is very useful for the tests of the microprocessor, to show how it is working in each moment when a problem appears.
- LED ON (green color): This LED is used to know if the ATMEGA328P-AU is turned ON.

The resistors value used for these LEDs was calculated measuring the voltage and current consumption of each one:

	Green LED	**Yellow LED**
Operating Voltage (V)	2.1	2.1
Datasheet Current (mA)	20	20
Measured Current (mA)	17.4	16.5
Theoretic Value Resistor (Ω)	166.67	166.67
Normalized Value Resistor (Ω)	200	200

Table 4.1 – *Calculations of the resistor for the LEDs*

The Equation 4.1.1 was used for these calculations.

$$R = \frac{V_{dd} - V_{LED}}{I_{LED}} \tag{4.1.1}$$

Where V_{dd} is the voltage supply for the LED and the resistor (5V), V_{LED} is the voltage that the LED supports for a correct operation (2.1V), and I_{LED} is the current consumption that the LED have had in the measurements.

The device that controls the Blink LED (LMV358IDGKR) is an operational amplifier used as a comparator to turn ON/OFF the LED. [29].

4.1.1.2 Display LCD

The LCD will be very useful in the prototype version of the Testbed. It will be used for seeing the data from the sensors and the GS in real time. Moreover, the LCD will be used to facilitate the visualization of the data without the use of the GS, testing the Testbed without the wireless connection.

The Figure 4.4 shows the schematic symbol for the LCD, which consisted of 4 pins. That LCD has a I2C converter incorporated on it, because of the insufficient number of pins that the microprocessor has and its easy functionality (see Figure 4.5).

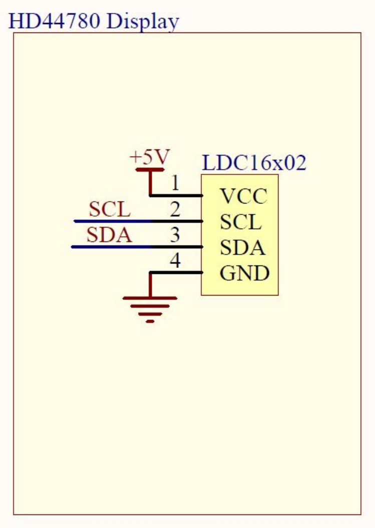


Figure 4.4 – *LCD Schematic*

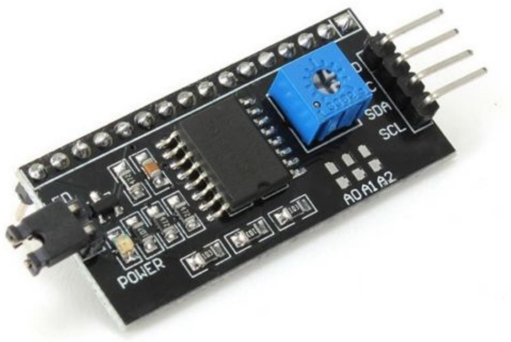


Figure 4.5 – *LCD to I2C converter*

The pinout of these device is the following:

- VCC: 5V voltage supply.
- GND: Pin connected to the ground of the circuit.
- SDA: I2C wire dedicated to the data signal. Connected to the PIN AD4 of the microprocessor.
- SCL: I2C wire dedicated to the clock signal. Connected to the PIN AD5 of the microprocessor.

The LCD to I2C converter has a potentiometer to regulate the contrast of the LCD, and a jumper to choose if the back-light is turned ON or turned OFF. The schematic of this converter is shown in the Figure 4.6. A photo of the converter and the LCD is shown in the Figure 4.7.

The LCD choose for the Testbed has a HD4480 controller, whose datasheet is found on the reference [27].

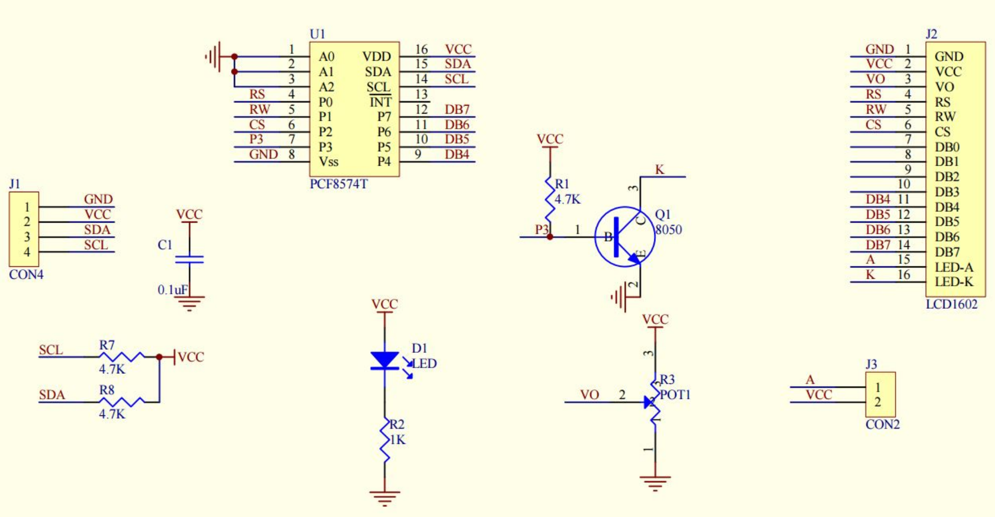


Figure 4.6 – *LCD to I2C converter schematic [49]*

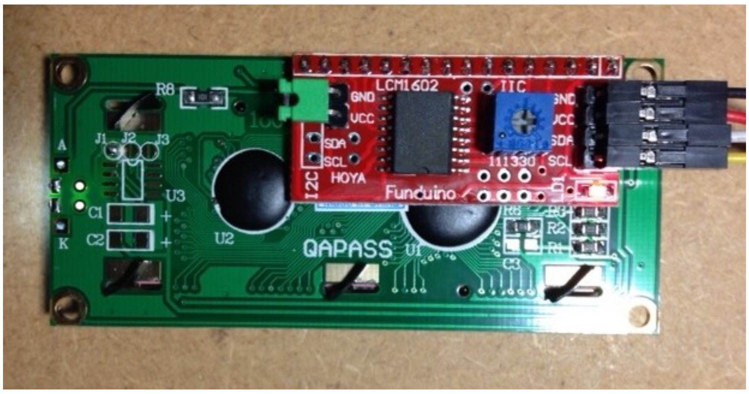


Figure 4.7 – *Photo of the I2C converter and the LCD [26]*

4.1.1.3 Magnetometer and Accelerometer Sensor

For the correcting use of the ADCS, it is necessary some sensors. The magnetorquers work with the Earth's Magnetic Field, so it should be measured by a sensor, in that case, a magnetometer. The magnetometer sensor chosen is the LSM303DLHC model (Figure 4.8), of STMicroelectronics. The datasheet of this sensor is found on the reference [52].

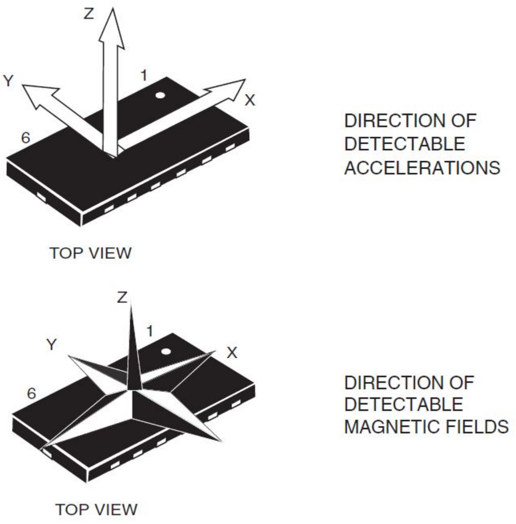


Figure 4.8 – *LSM303DLHC chip [52]*

In order to have an easier soldering process, a test board (GY-511) for this sensor has been used. (see figure 4.9 and 4.10 for the schematic of this test board). That sensor has a 3D digital linear acceleration sensor and a 3D digital magnetic sensor. The following table shows the main characteristics.

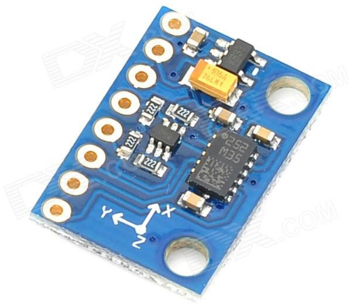


Figure 4.9 – *Photo of the the test board for LSM303DLHC*

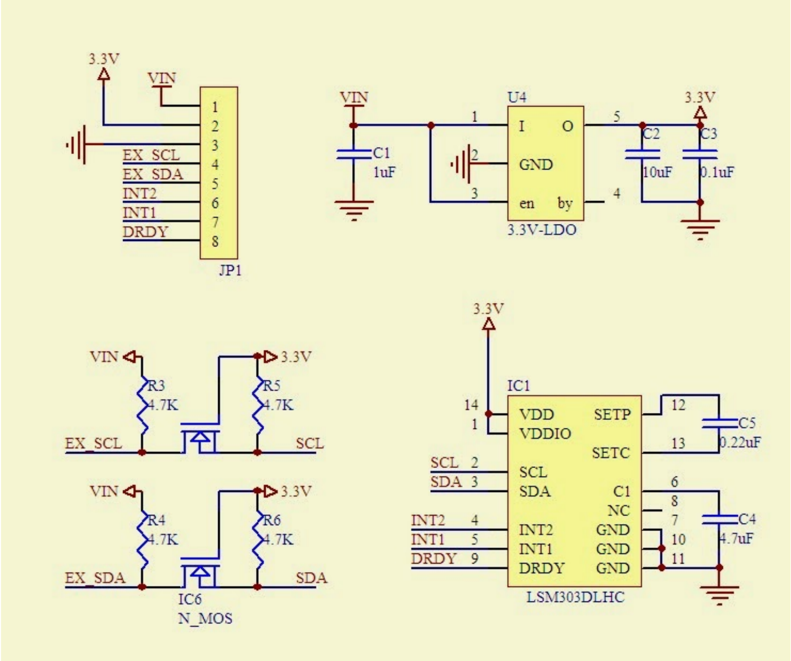


Figure 4.10 – *Schematic of the test board for LSM303DLHC [18]*

Number of channels	3 x accel + 3 x magnet
Magnetic field scale (gauss)	±1.3/±1.9/±2.5/±4.0/±4.7/±5.6/±8.1
Linear acceleration scale (G)	±2/±4/±8/±16
Data output (bits)	16
Interface	I2C
Analog supply voltage for the sensor (V)	2.16-3.6
Analog supply voltage for the board (V)	5
Temperature range (°C)	-40 to +85

Table 4.2 – *LSM303DLHC Specifications [52]*

The schematic symbol that this sensor has is shown in figure 4.11. It has 8 pins through hole, to place the test board shown in figure 4.9.

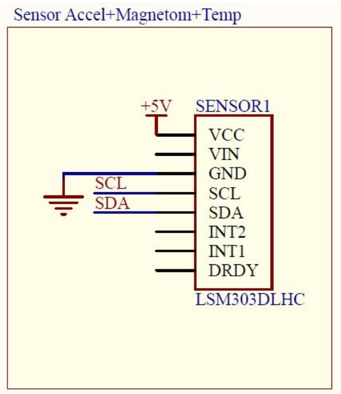


Figure 4.11 – *Schematic symbol for Testbed Board for LSM303DLHC*

4.1.1.4 Gyroscope and Accelerometer Sensor

In order to have the gyroscope sensor on the Testbed PCB, we have include the MPU6050 sensor, which includes an gyroscope, an accelerometer and it is prepared to measure the temperature too. The MPU-6050 devices mixes a 3-axis gyroscope (3D) and a 3-axis accelerometer on the same chip. In addition, it includes Digital Motion Processor™ (DMP™), which processes complex 6-axis MotionFusion algorithms. [6]

As the LSM303DLHC sensor (see previous section 4.1.1.3), we have used a test board for this sensor, GY-521 board in that case (see Figure 4.12). It allow us a better soldering process and an easier prototype version to prove the different subsystems. The schematic of this board is shown in Figure 4.13, where is represented the circuit necessary for the correct operation oh that chip, the MPU6050 sensor.

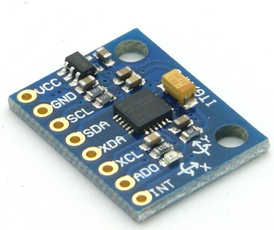


Figure 4.12 – *Photo of the test board for MPU6050*

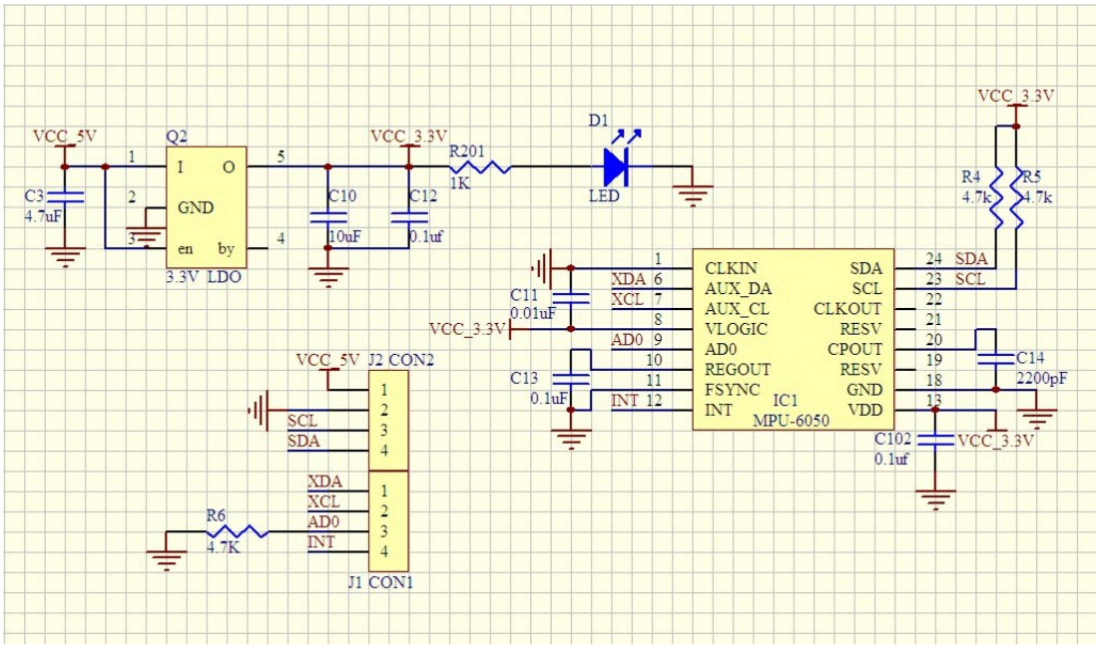


Figure 4.13 – *Schematic of the test board for MPU6050*

And finally, the same as LSM303DLHC, it is placed on the Testbed PCB an 8 pins schematic symbol (see figure 4.14), for the placement of the test board shown in the Figure 4.12.

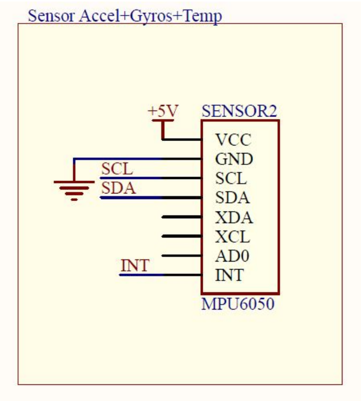


Figure 4.14 – *Schematic of the test board for MPU6050 [5]*

The main characteristics of this sensor are the following:

Number of channels	3 x accel + 3 x gyro
Gyroscope scale (/*sec*)	±250 ±500 ±1000 ±2000
Linear acceleration scale (G)	±2/±4/±8/±16
Data output (bits)	16
Interface	I2C
Analog supply voltage for the sensor (V)	2.375–3.46
Analog supply voltage for the board (V)	5
Logic supply voltage (V)	1.71 to VDD (5)
Temperature range (°C)	-40 to +85

Table 4.3 – *MPU6050 Specifications [31]*

For the communication with the microprocessor, I2C protocol is used, connected in parallel with the LSM303DLHC and the LCD described before. In the Figure 4.15 is shown how this connection is done and the directions that each device has.

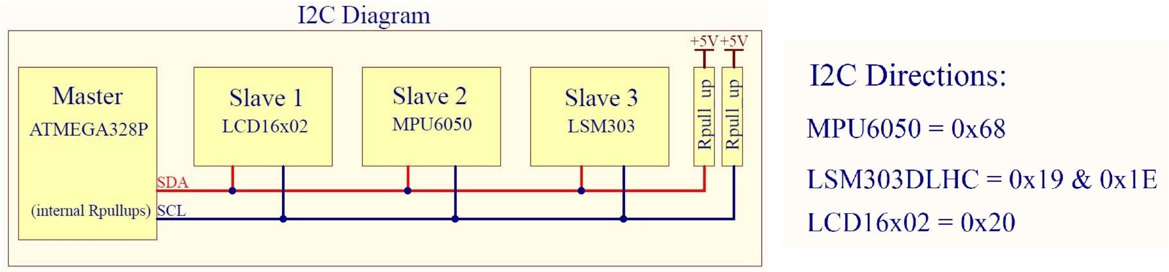


Figure 4.15 – *I2C diagram of the system*

4.1.1.5 Wireless Module

One of the most relevant requirements for the Testbed is the wireless communication. For that issue, the PCB includes a specific device from Digi: Xbee Pro series 1 (see Figure 4.16. That device was chosen for the communication to the Testbed by Carlos Valenzuela, in his Final Project Degree about the Testbed communication [59].

Figure 4.16 – *Module Xbee Pro Series 1*

From the datasheet of the device we can acquire the dimensions of the module Xbee PRO (in our case, the Series 1):

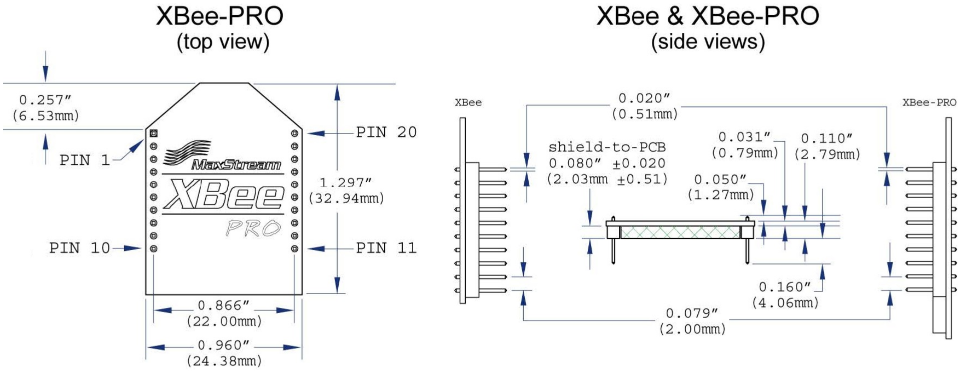


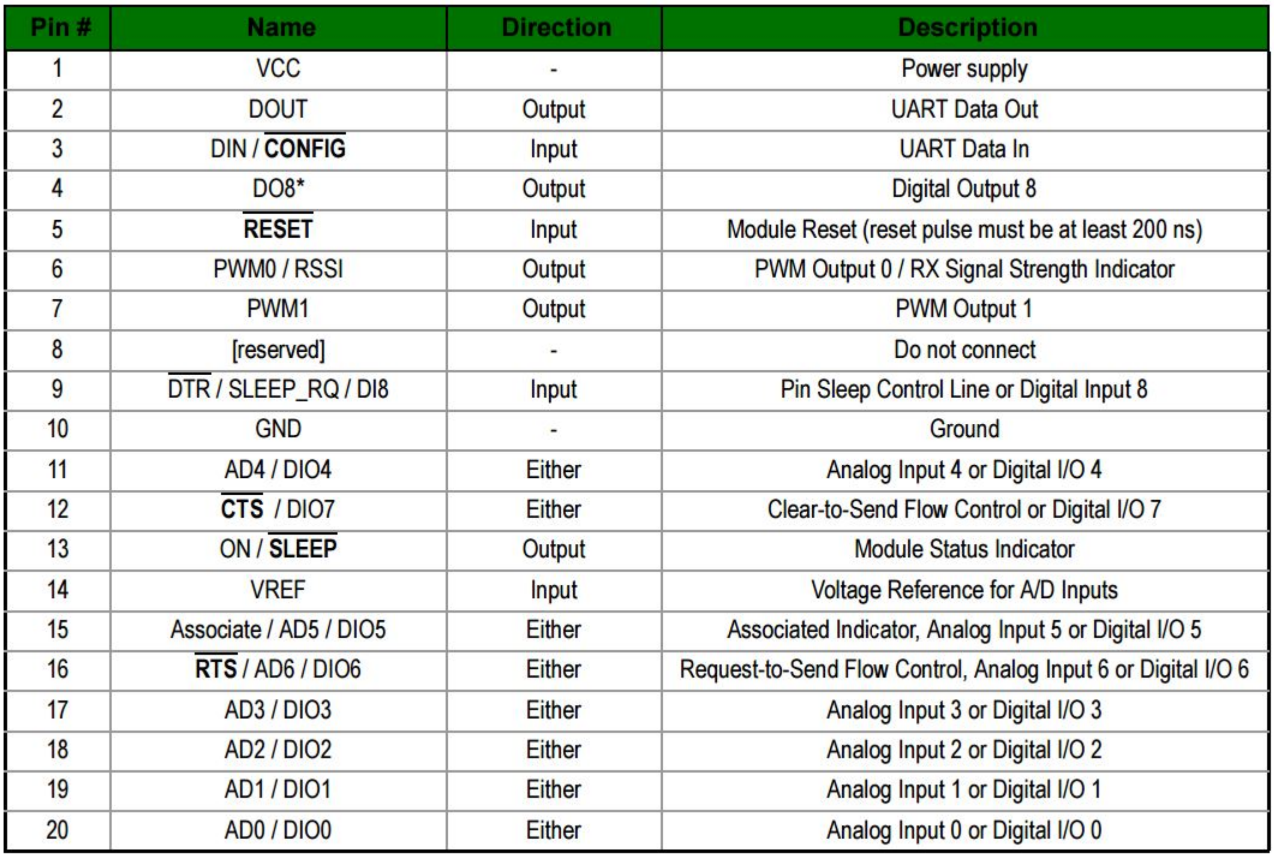
Figure 4.17 – *Dimensions of the Module Xbee Pro Series 1*

Inside the family Xbee, we have selected the PRO Series 1, because of its good implementation to the Point-to-Point Link. The PRO selection is due to the improvements that it has in comparison with the Standard Series in respect of its range, sensibility and transmitted power. Moreover, the Series 2 of this family, has been designed for the communication in more complex networks, so for our link it will be inefficient. [59]

The main features for this module are:

Dimensions (mm)	324.38 x 32.94
Operating Frequency (GHz)	2.4
RF Data Rate (kbps)	250
Indoor Range (m)	Up to 90
Antenna types	Chip, Wire, Whip, UFL, RPSMA
Supply Voltage (V)	2.8-3.4
Serial Interface	3.3V CMOS UART
Logic supply voltage (V)	1.71 to VDD (5)
Temperature range (°C)	-40 to +85
Transmit Current (mA)	150
Receive Current (mA)	55

Table 4.4 – *Xbee PRO Series S1 Specifications* [17]



Pin #	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Output	UART Data Out
3	DIN / **CONFIG**	Input	UART Data In
4	DO8*	Output	Digital Output 8
5	**RESET**	Input	Module Reset (reset pulse must be at least 200 ns)
6	PWM0 / RSSI	Output	PWM Output 0 / RX Signal Strength Indicator
7	PWM1	Output	PWM Output 1
8	[reserved]	-	Do not connect
9	DTR / SLEEP_RQ / DI8	Input	Pin Sleep Control Line or Digital Input 8
10	GND	-	Ground
11	AD4 / DIO4	Either	Analog Input 4 or Digital I/O 4
12	**CTS** / DIO7	Either	Clear-to-Send Flow Control or Digital I/O 7
13	ON / **SLEEP**	Output	Module Status Indicator
14	VREF	Input	Voltage Reference for A/D Inputs
15	Associate / AD5 / DIO5	Either	Associated Indicator, Analog Input 5 or Digital I/O 5
16	**RTS** / AD6 / DIO6	Either	Request-to-Send Flow Control, Analog Input 6 or Digital I/O 6
17	AD3 / DIO3	Either	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Either	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Either	Analog Input 1 or Digital I/O 1
20	AD0 / DIO0	Either	Analog Input 0 or Digital I/O 0

Figure 4.18 – *Pinout of the Xbee Module [17]*

In Figure 4.18 is shown how the pinout of this module is distributed. But, for our case, the only connections requires are: [17]

- VCC: connected to 3.3V.

- GND: connected to the circuit ground.

- DOUT: connected to TXD of the microprocessor. There is a buffer in this line (SN74LVC1G125DCK) to adjust the levels between the Arduino and the Xbee Module [59] [56].

- DIN: connected to RXD of the microprocessor.

So, that connections were done in the Schematic of the project (see figure 4.19).

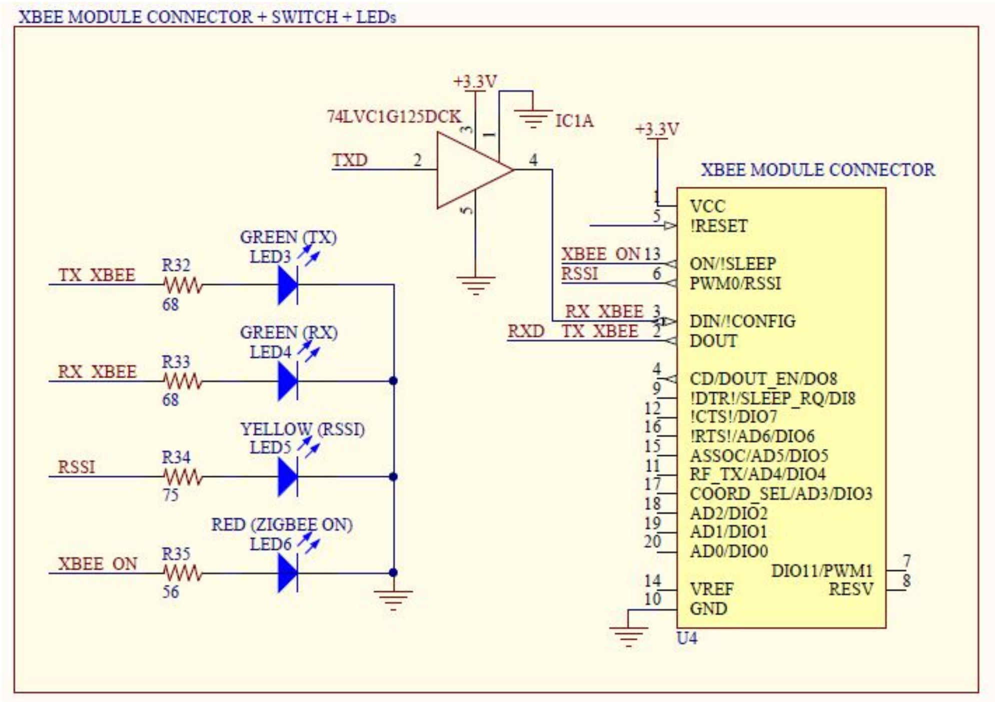


Figure 4.19 – *Schematic of the Xbee Module*

On the other hand, if you want to upload the firmware of the module, the required connections would be: VCC, GND, DOUT, DIN, RTS [17], but we have an USB adapter to do this task, so we can save these connections.

Finally, in order to have more information about the status of the communication, some LEDs are placed. The resistors value, in this case for a VCC of 3.3V, were calculated like in the subsection 4.1.1.1.

	Green LED (TX and RX)	**Yellow LED (RSSI)**	**Red LED**
Operating Voltage (V)	2.1	2.1	2.1
Datasheet Current (mA)	20	20	20
Measured Current (mA)	17.4	16.5	21.43
Theoretic Value Resistor (Ω)	68.966	72.727	55.996
Normalized Value Resistor (Ω)	68	75	56

Table 4.5 – *Calculations of the resistors for the Xbee LEDs*

Again, the Equation 4.1.1 was used for these calculations, but changing the V_{dd} voltage (now is 3.3V).

For more information about the Xbee module and the connection between the GS and the Testbed platform, see the Master Thesis about this issue. [59]

4.1.1.6 Actuators Drivers

To control the actuators of GranaSAT Testbed, there are placed on the PCB two drivers, in this case the model L298P of STMicroelectronics, an SMD dual full-bridge driver, in PowerSO20 package version. That device will accept standard TTL logic levels and drive inductive loads (in my case, DC Motors).[50]

Each motor has 2 input signals to control the direction of the current, and an enable pin to enable or disable the motor independently of the input signals. Moreover, it has a current sense resistor to read the current consumption of each motor.

The pinout of this device is shown in Figure 4.20 and the function for each pin in the Figure 4.21.

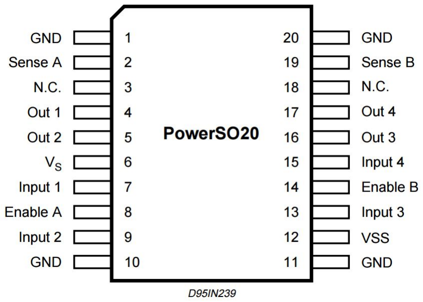
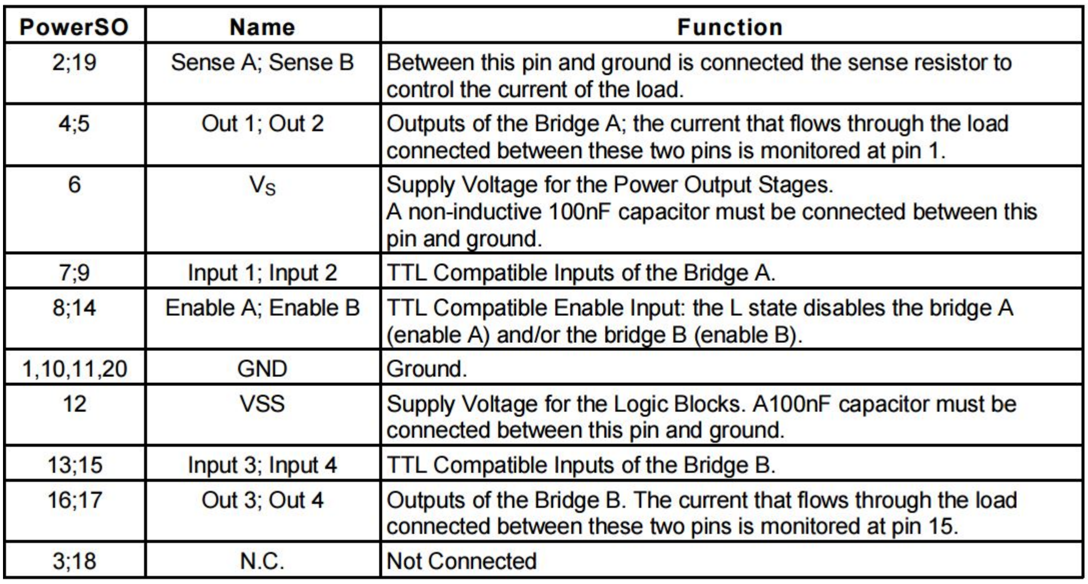


Figure 4.20 – *Pinout of the L298P [50]*



PowerSO	Name	Function
2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
6	V_S	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
1,10,11,20	GND	Ground.
12	VSS	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
3;18	N.C.	Not Connected

Figure 4.21 – *Pin functions of the L298P [50]*

The schematic related to the drivers is shown in Figure 4.22.

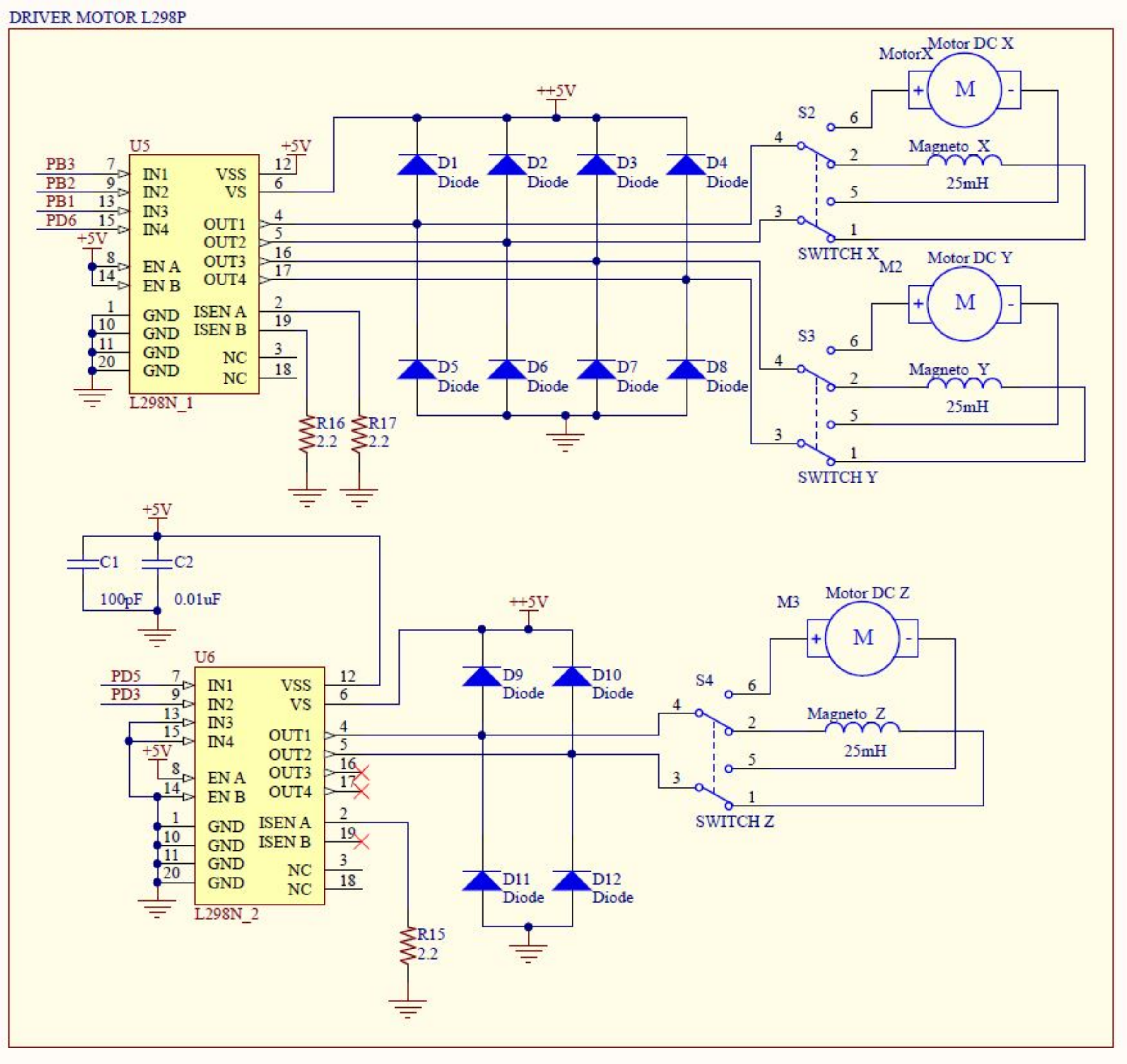


Figure 4.22 – *L298P Schematic [50]*

In order to protect the H-bridge of each driver, some protection diodes (Flyback Diodes) are placed in the outputs of the drivers. The reason of the protection is to create a low-impedance path for the coils (DC Motors) to discharge themselves through them. If they are not placed on the output, the discharge of the coils (stored magnetic energy), when the supply is stopped at each cycle, will generate arbitrarily a high reverse voltage across the MOSFETs of the drivers, which can damage them. That diodes are the model S1M standard SMD, from Fairchild [23].

That rectifier diodes have the following main features shown in Table 4.6:

Maximum Repetitive Reverse Voltage (V)	1000
Average Rectified Forward Current at $T_A = 100$ (A)	1
Storage Temperature Range (kbps)	-55 to +150
Operating Junction Temperature (m)	-55 to +150
Forward Voltage at 1A (V)	1.1

Table 4.6 – *Rectifier Diode S1M Specifications [23]*

The package that the S1M diode has is SMA/DO-214AC, see Figure 4.23.

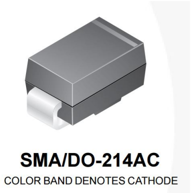
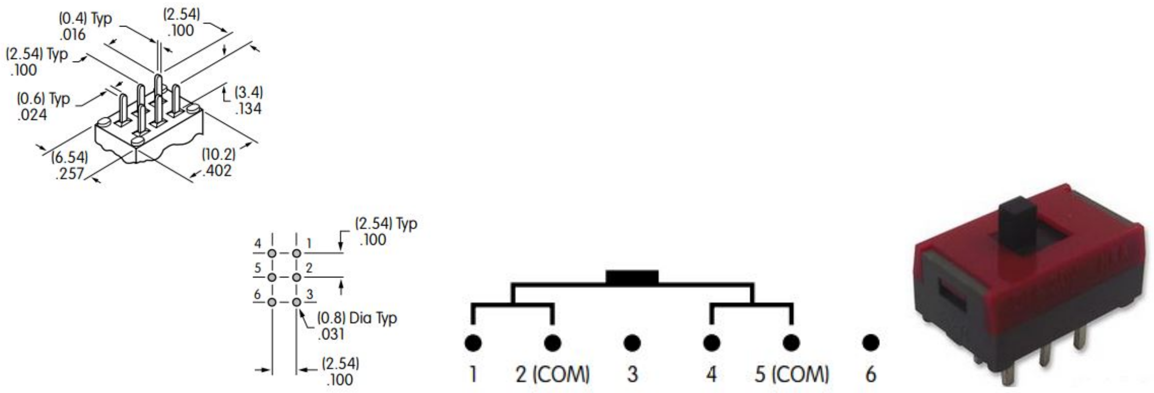


Figure 4.23 – *S1M package[50]*

Finally, to select the actuators (reaction wheels or magnetorquers), there are three switches between the output lines of the driver and the actuators. The switches are of the type DPDT (double pole double throw) and the model SS22SDP2 of NKK switches, and whose diagram, dimensions and a photo can be seen in figure 4.24.



(a) *Switch dimensions* **(b)** *Switch diagram* **(c)** *Switch photo*

Figure 4.24 – *Switch NKK SS22SDP2*

4.1.1.7 Power system

The last section of the Schematic design for the Testbed PCB of GranaSAT project, is the power system. Schematics are shown in Figure 4.25.

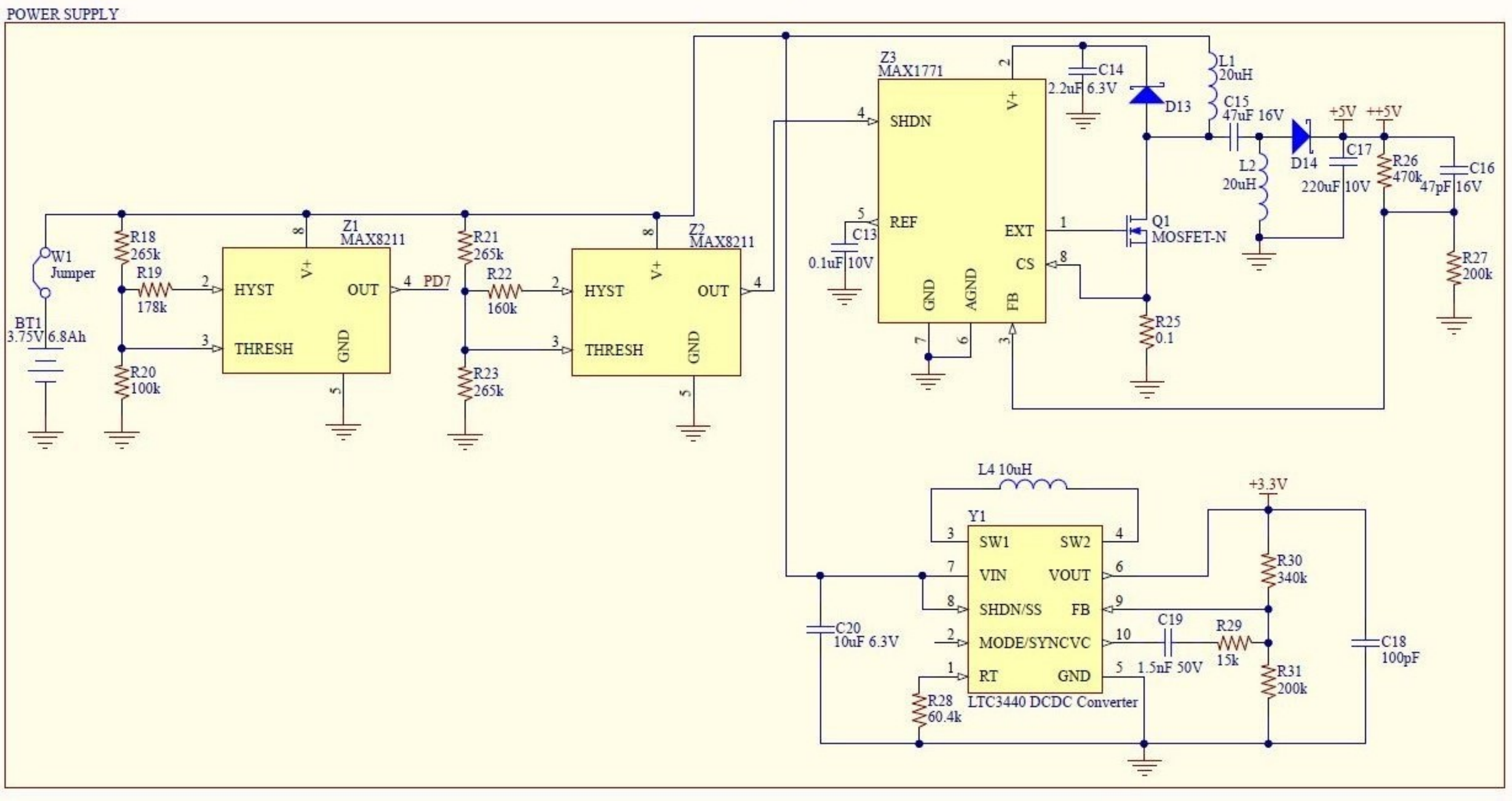


Figure 4.25 – *Schematic for the power system of the Testbed*

4.1.1.7.1 Battery

The first part that is represented in the schematic is the battery. That battery (figure 4.26) is the same as used for the Cubesat GranaSAT-I [21], whose model is MGL2803 from Enix Energies. Some of the main characteristics that the battery has are shown in table 4.7. The battery is connected to the PCB by a PCB terminal connector placed on the bottom layer (see section 4.1.2.

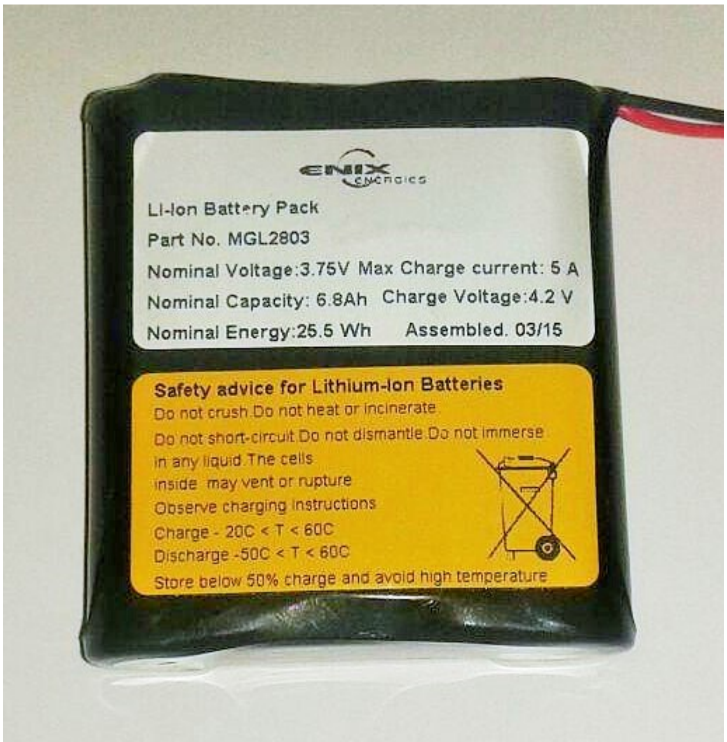

ENIX ENERGIES
Li-Ion Battery Pack
Part No. MGL2803
Nominal Voltage:3.75V Max Charge current: 5 A
Nominal Capacity: 6.8Ah Charge Voltage:4.2 V
Nominal Energy:25.5 Wh Assembled. 03/15
Safety advice for Lithium-Ion Batteries
Do not crush.Do not heat or incinerate
Do not short-circuit.Do not dismantle.Do not immerse
in any liquid.The cells
inside may vent or rupture
Observe charging instructions
Charge - 20C < T < 60C
Discharge -50C < T < 60C
Store below 50% charge and avoid high temperature

Figure 4.26 – *MGL2803 battery*

Nominal voltage (V)	3.75
Nominal capacity 20°C (Ah)	6.8
Technology	Rechargeable Lithium Ion Battery Pack
Discharge Cut off V Nominal (V)	2.8
Maximum recommended discharge current (A)	5
Charge Voltage Nominal (V)	4.2
Charge Termination Current (mA)	150
Discharge temperature range ()	-10 to +60

Table 4.7 – *ENIX MGL2803 Battery Specifications [23]*

The battery charge will be described in the Chapter 5.

The following item in the circuit described before (figure 4.25), is a jumper, which will turn on or off the whole system.

4.1.1.7.2 Voltage detectors

We have now 2 identical devices, MAX8211. That electronic device is a voltage monitor with programmable voltage detection. The application of the first one is detect the voltage that the battery has in each cycle, and notify to the microprocessor (see section 4.1.1.1) when the voltage level was reach. For the second one, the objective is directly turn-off the whole system.

The pinout of the device (in SO-8 package) is shown in the figure 4.27.

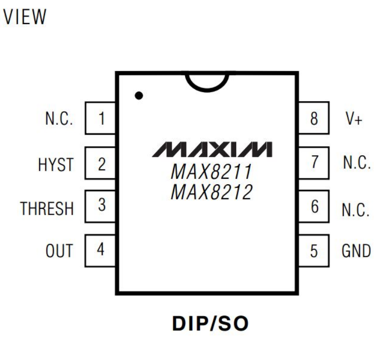


Figure 4.27 – *MAX8211 Voltage Detector*

The schematic of the figure 4.25 shows three resistors for this device. The value of these resistors determine the value for the application of MAX8211. The calculations for the resistors for the first MAX8211 are represented in equations 4.1.2, 4.1.3 and 4.1.4 First, it is necessary fix the value of the R20 resistor (see equation 4.1.2).

$$R20 = 100k\Omega \tag{4.1.2}$$

$$R19 = R20 \times \frac{V_U - V_{TH}}{V_{TH}} = 100k\Omega \times \frac{V_U - 1.15V}{1.15V} \tag{4.1.3}$$

$$R18 = R19 \times \frac{V_L - V_{TH}}{V_U - V_L} = R19 \times \frac{V_L - 1.15}{V_U - V_L} \tag{4.1.4}$$

Fixing the V_U=3.2V and V_L=2.4V values the resistors can be calculated. The values have to be changed to the closest normalised value. Where V_U is the value that represents the start point once the battery have been discharged (when it is charging mode), and V_L is the value from which the system turns off when it is in discharging mode.

The second MAX8211 is calculated in the same way, but with a more limiting values for the threshold voltages.

4.1.1.7.3 DC-DC converters

Finally, the power system has the voltage converters, in order to have 5V and 3.3V output with the 3.75V nominal input voltage from the battery. There are two different converters, one for each desired output. The first one is the MAX1771, from MAXIM. That step-up switching controller provides 90% of efficiency over a 30mA to 2A load [30]. It will be used the adjustable application for this device, in order to have the 5V in the output.

The package mounted on the PCB is SO-8, see figure 4.28. In this figure it is shown the pinout for the MAX1771 too and in the figure 4.8 the function of each pin.

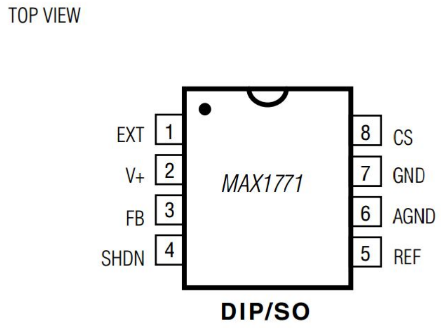
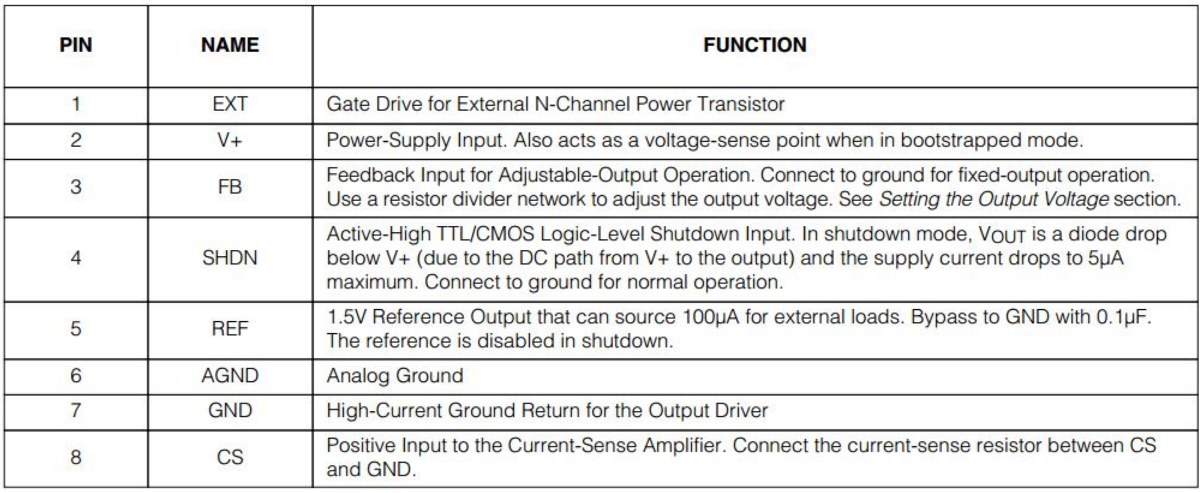


Figure 4.28 – *MAX1771 Step-up Converter*



PIN	NAME	FUNCTION
1	EXT	Gate Drive for External N-Channel Power Transistor
2	V+	Power-Supply Input. Also acts as a voltage-sense point when in bootstrapped mode.
3	FB	Feedback Input for Adjustable-Output Operation. Connect to ground for fixed-output operation. Use a resistor divider network to adjust the output voltage. See *Setting the Output Voltage* section.
4	SHDN	Active-High TTL/CMOS Logic-Level Shutdown Input. In shutdown mode, V_{OUT} is a diode drop below V+ (due to the DC path from V+ to the output) and the supply current drops to 5µA maximum. Connect to ground for normal operation.
5	REF	1.5V Reference Output that can source 100µA for external loads. Bypass to GND with 0.1µF. The reference is disabled in shutdown.
6	AGND	Analog Ground
7	GND	High-Current Ground Return for the Output Driver
8	CS	Positive Input to the Current-Sense Amplifier. Connect the current-sense resistor between CS and GND.

Table 4.8 – *MAX1771 Pins function*

It is not necessary calculate the resistors values because there is an application circuit to supply 5V in the output with the adjustable way (Figure 5 of the Datasheet [30]).

The second one is the LTC3440, from Linear technology. It is a Buck-Boost DC/DC Converter, which has the feature of having the fixed frequency operation with battery voltages above, below or equal to the output voltage, with a 96% of efficiency. [55]

The package chosen is the MSOP (10 leads), which has the pinout represented in the figure 4.29.

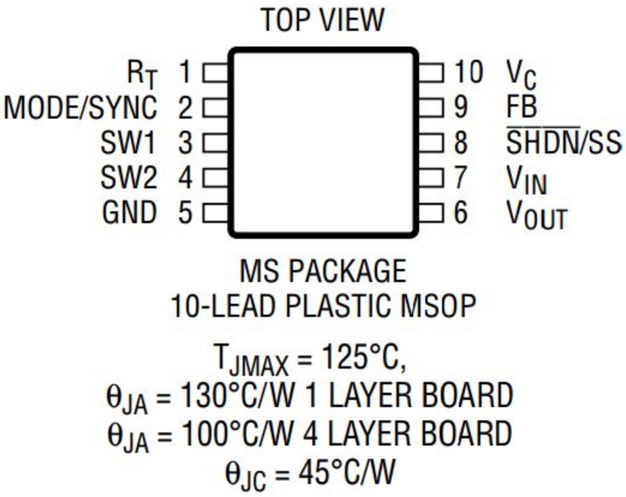


Figure 4.29 – *LTC3440 Pinout*

4.1.2 PCB Design

In this section, it will be presented the design process for the PCB design of the circuit that has been explained in the section 4.1.1. For the PCB design, I have used Altium Designer, the same as the Schematic Design. That tool allows connect the schematic and PCB, linking every component that you place on it. That design has some relevant steps which have to be followed in the right order for a proper PCB design:

- **PCB production Technology**. The first step that you have to take into account when build a PCB is the production technology that you have available for the project. In my case, a prototyping PCB machine was available thanks to the ECTD (LPFK ProtoMat S62 [35]) for the PCB production. So, the solution that we have available is a copper plate with the tracks, holes and vias made with drilling tools. One of the most important features with this technology is the easy PCB production in both sides, top and bottom layer and the drilling facilities that it has to make the holes and vias. The figure 4.30 present the LPFK ProtoMat S62 that we have in the laboratory of the Faculty of Science of the University of Granada.

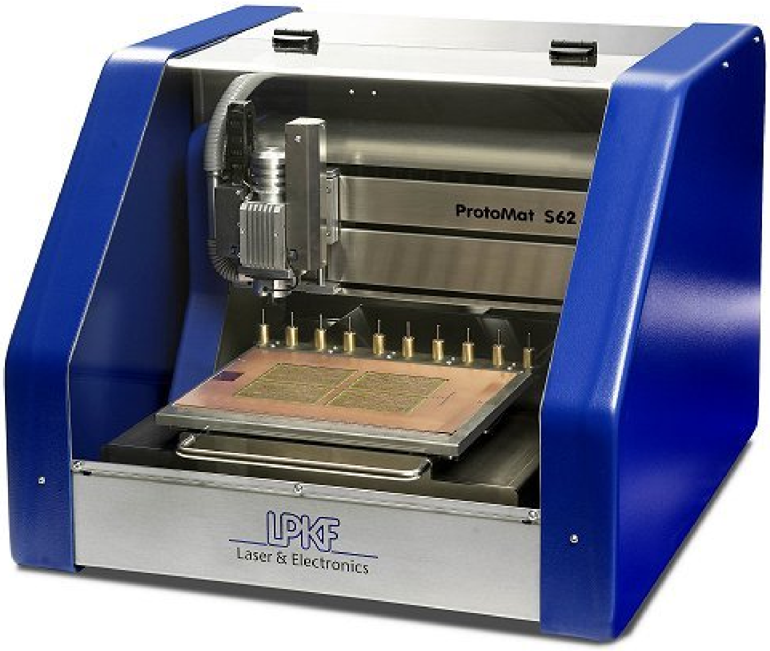


Figure 4.30 – *LPFK ProtoMat S62*

- **Mounting technology**. Once we know what type of technology is available for the PCB production, we have to decide what mounting technology is the best solution for our design. The first issue that we have to consider is the minimum weight for the PCB, because is one of the requirements (see section 2.5), so the SMD technology is the best solution. There are some devices which are not available in SMD, so the PCB will have SMD and THT technology.

- **Package types**. Depending on the device chosen, different packages have been selected, trying to choose the smallest one because of the weight constraint. For the passive devices (resistors, capacitors, etc), in the majority of the cases, the size chosen was 0805 and 0603, but there are some 1206 and 0402, subject to the power consumption of each resistor and the maximum voltage for the capacitors.

- **PCB Library**. As soon as we have selected the components and their package it is time to make the PCB library. The library will include the footprints of every component placed on the PCB and it will be created with the packages found in other libraries or made by the designer (measuring the device with a caliber or taking the measurements from the datasheet information).

- **Transfering the components from the SCH to the PCB**. When the PCB library was finished, the schematic should have every component linked to its footprint. The next step is transfer that information to a PCB, in order to start with the placing and the routing of the footprints. That step will create the netlists for the PCB, which will indicate the connections between the components in order to facilitate the next step.

- **Placing and routing the PCB components**. First, a good placing will facilitate the routing of the components(trace the tracks that connect the components).

- **Verify the PCB design rules and connections**. Altium Designer has a really useful tool: *Design Rule Check* (see figure 4.31). That tool have the objective of test if the rules for the design are being fulfilled. That rules will be explained in the section 4.1.2.1.

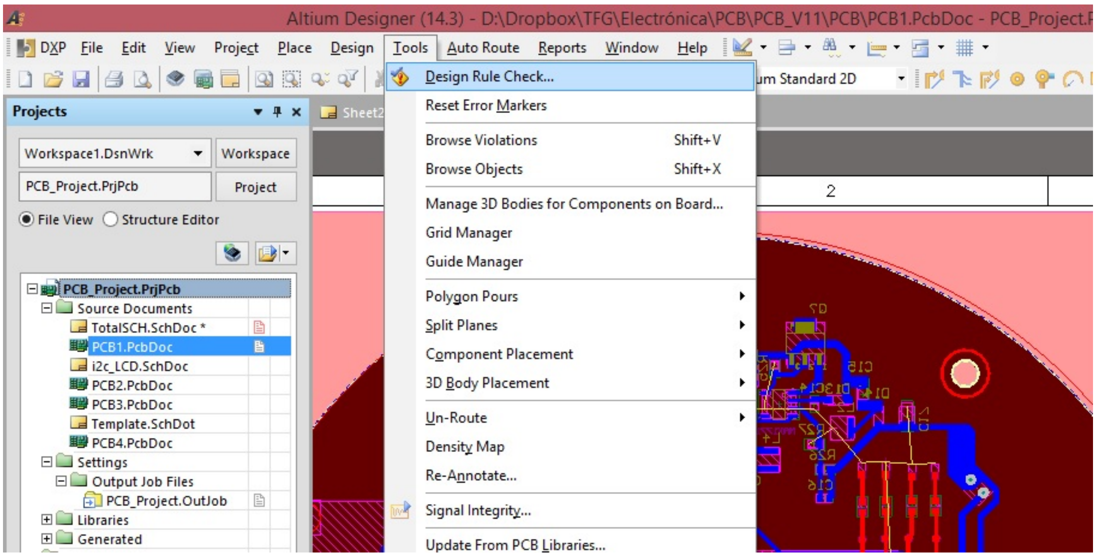


Figure 4.31 – *Design Rule Check - Altium Designer*

4.1.2.1 Design rules PCB

The ProtoMat S62 (PCB prototype machine) has some mechanical requirements that we have to take into account. The affected PCB characteristics for that requirements are:

- **Track**. In every moment, the best option was the maximum width possible for the tracks, in order to reduce the resistance of the track and allow a better operation for the PCB, overall, for the power tracks. The minimum width for the tracks has been 15 mils, because of the operation mode that has the plotter, in order to have a minimum final width of the 10 mils, so the production phase reduces 5 mils approximately from the design. Moreover, it should be avoid the 90 angles, and the shortest tracks possible.

- **Clearance**. The clearance is the minimum spacing between the tracks on a PCB. The clearance should be as minimum 10 mils, in order to avoid the short-circuit between the tracks, because of the production errors that it can exist. Anyway, once the PCB was ended, with a cutter tool you can remark the tracks in order to avoid the shavings generated by the production phase.

- **Hole size**. The constraint of the hole size is imposed by the available drilling tools that we have in the laboratory. The minimum size fixed was 0.6 mm, and the available tools were: 0.6, 0.7, 0.8, 0.9, 1, 1.1, 1.3 y 1.5 mm. The biggest holes were made with a milling cutter drilling the perimeter of the hole.

- **Annular ring**. The annular ring is the copper portion between the hole and the perimeter of the pad. That rule is the same for the vias and THT pads. The minimum diameter of the annular ring is 80 mils, in order to have a good support for the soldering of this lead or via.
- **Text layer**. The production type that we have available has not the silkscreen process. With the intection to give more information of the PCB, a text tool (in both sides) has been created, on the ground plane, to avoid the short-circuits and the errors in the tracks.
- **Board Outline**. One of the most important things of the PCB is its dimension and shape. Because of the hemisphere chosen (see section 4.2), the PCB had to have circular shape, and the dimension was designed taking into account that the battery had to fit between the PCB and the lower part of the hemisphere.

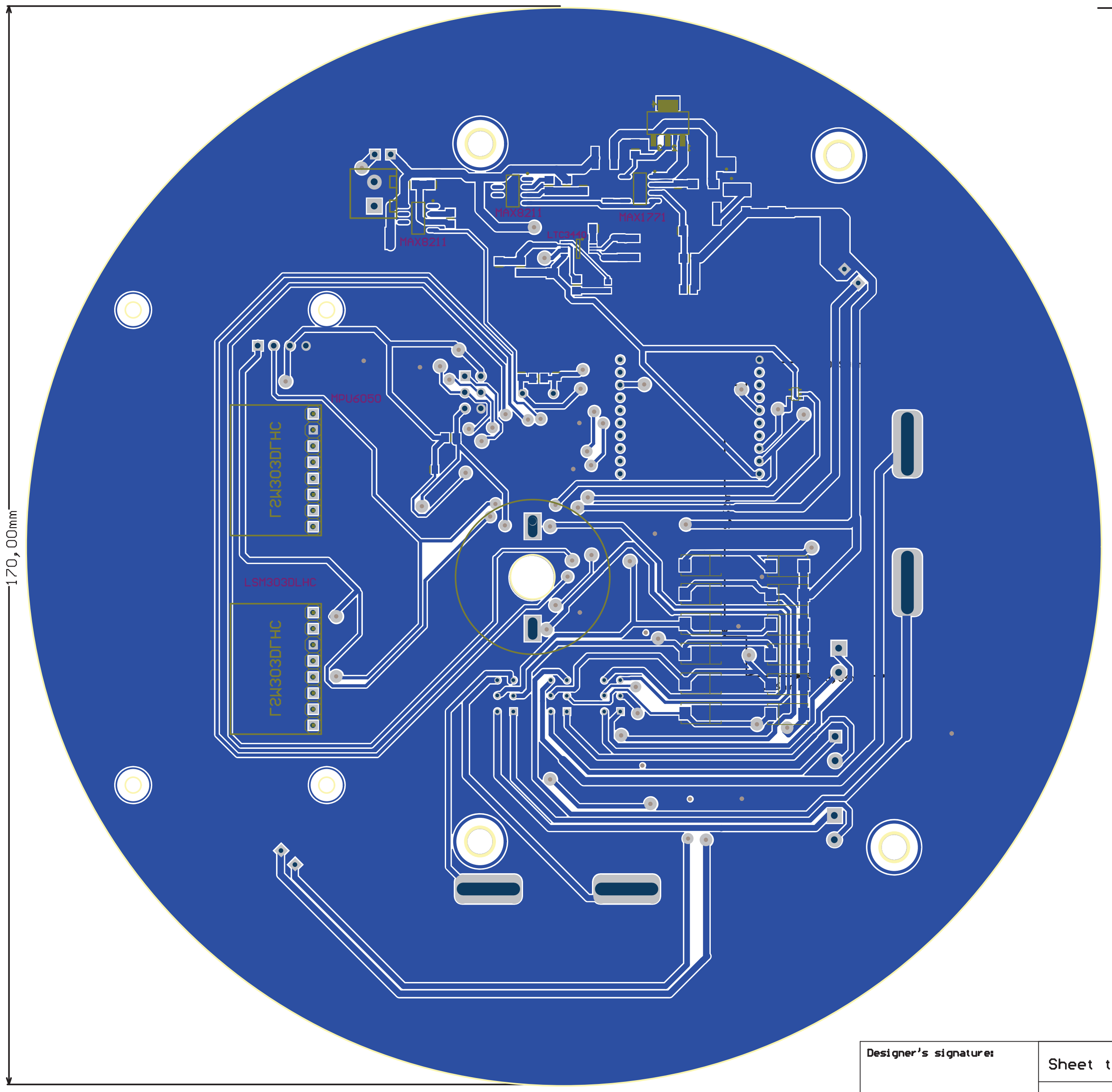
4.1.2.2 2D view of the PCB Design

The dimensions of the PCB were predefined once we had the semisphere od the Testbed. The board had to be

In the following pages will be located the PCB Design Schematics, where is described the tracks, vias, holes, footprints, and the position of each component. In this project, there are four PCBs of the following three types:

- Testbed PCB: Is the main PCB of the project, where the most components described in the subsection 4.1.1 are placed. That PCB has component in top and bottom layer, so it will be included the PDF of both sides.
- Motor PCB: That PCB will be used for support the motors in vertical position (for axes X and Y). There will be two units for that type of PCB, one for each motor, and the third motor will be placed on the center of the Testbed PCB, to control the Z axis (see section 2.3. It will be sold to the Testbed PCB to ensure the correct support and connect it with a pad located on that hole. The motors placed on that PCB have been explaines on section 3.3.
- Magnetorquers PCB: The function of this PCB is only support the magnetorquers mounted in a Cubesat structure 3D printed.

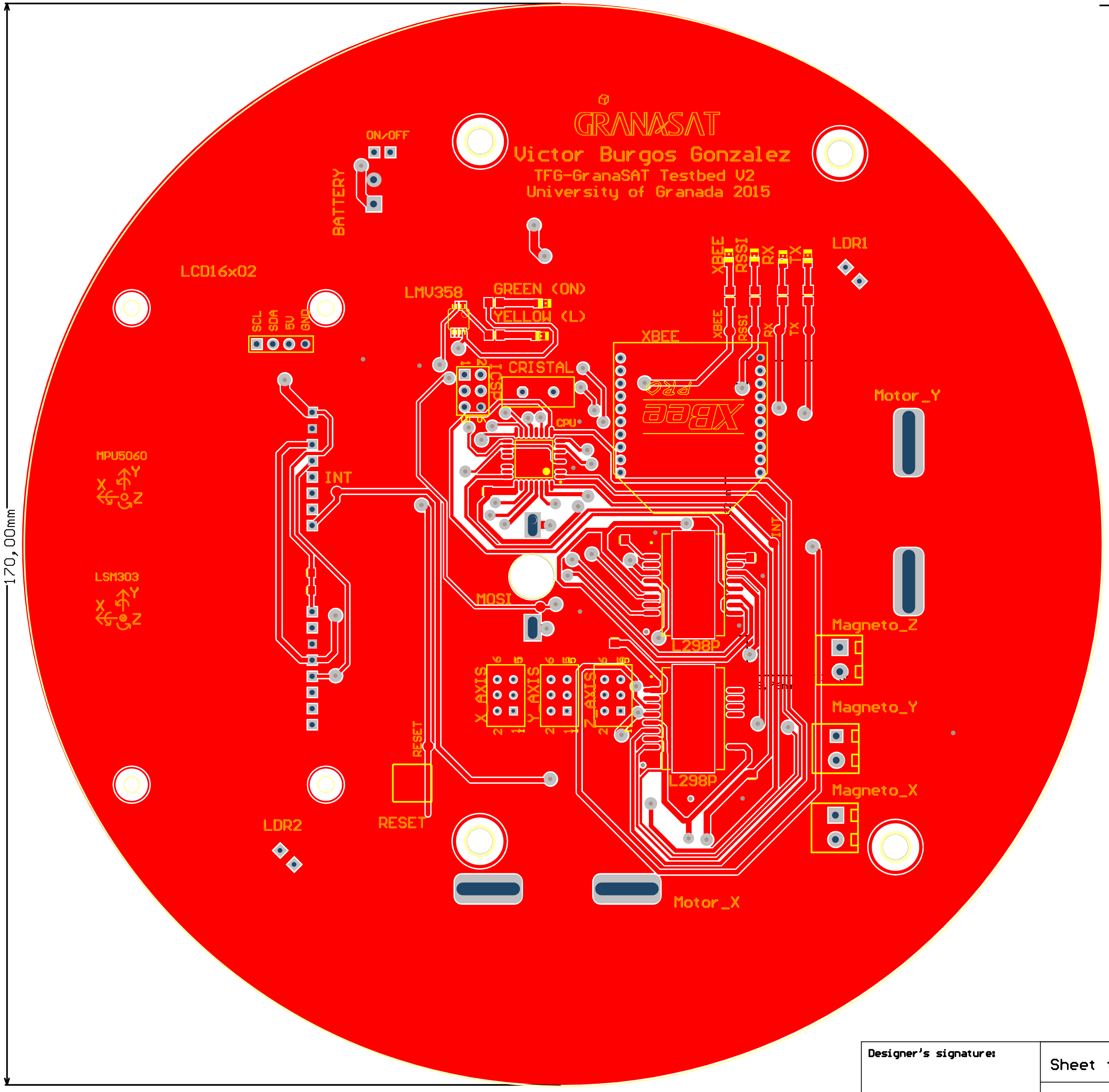
Testbed PCB



Description:
That PCB will be the main PCB for the Testbed and it will connect the Motor PCB to the other subsystems, and it will support the Magnetorquer PCB too with 4 screws.

Designer's signature:	Sheet title: Motor PCB	Dpto. Electronica y Tecnologia de Computadores University of Granada C/ Fuente Nueva, s/n, 18001 Granada, Granada, Spain Sr. Andres Roldan Aranda
	Project title: GranaSAT-I	
Supervisor's signature:	Designer: Victor Burgos Gonzalez	
	Supervisor: Andres Roldan Aranda	**Date: 01.05.2015 Revision: 1** **Sheet 1 of 10**

Testbed PCB



Description:
That PCB will be the main PCB for the Testbed and it will connect the Motor PCB to the other subsystems, and it will support the Magnetorquer PCB too with 4 screws.

Designer's signature:

Supervisor's signature:

Sheet title: Motor PCB

Project title: GranaSAT-I

Designer: Victor Burgos Gonzalez

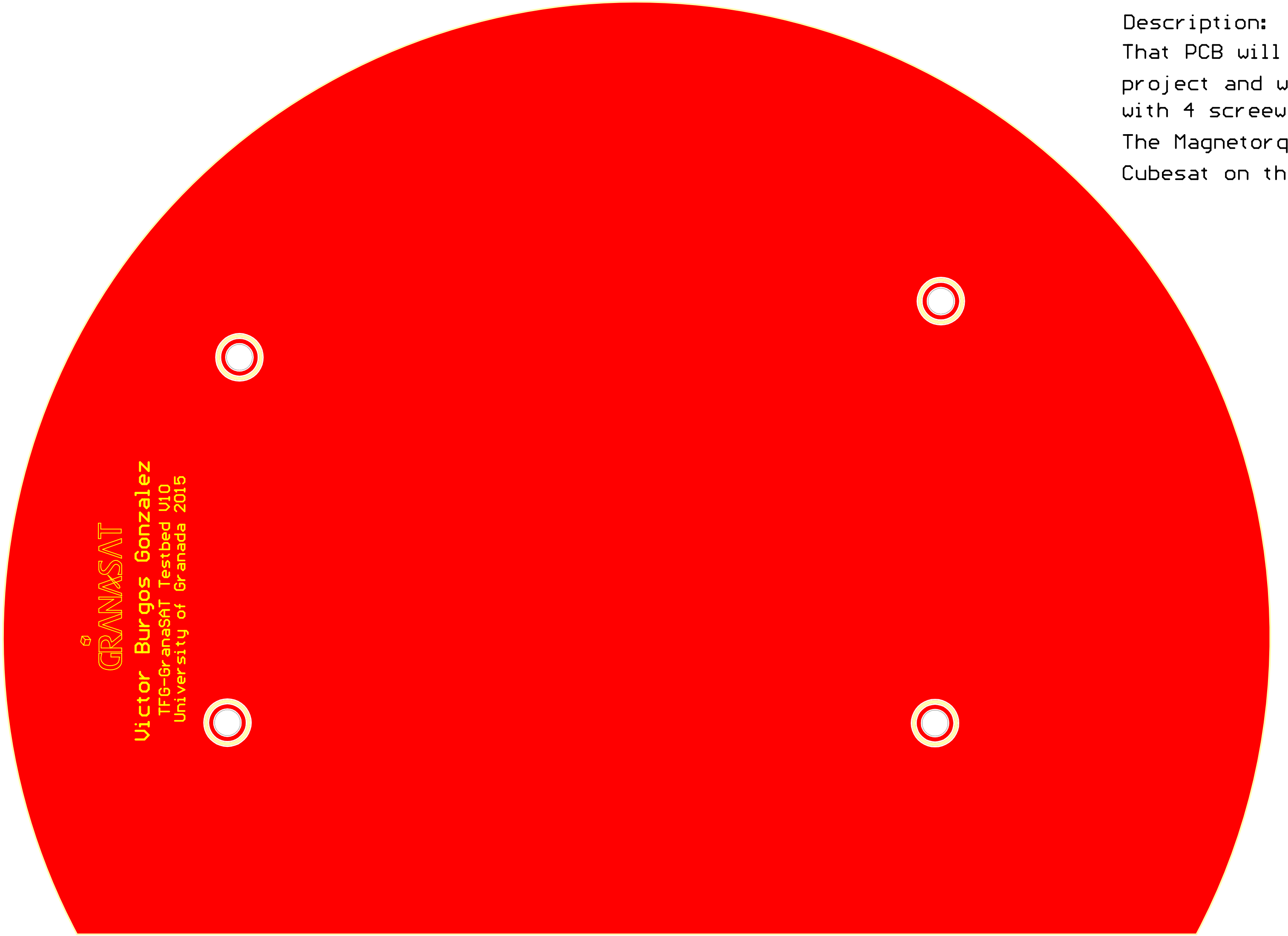
Supervisor: Andres Roldan Aranda

Dpto. Electronica y Tecnologia
de Computadores
University of Granada
C/ Fuente Nueva, s/n, 18001
Granada, Granada, Spain
Sr. Andres Roldan Aranda

Date: 01.05.2015 Revision: 1
Sheet 1 of 10

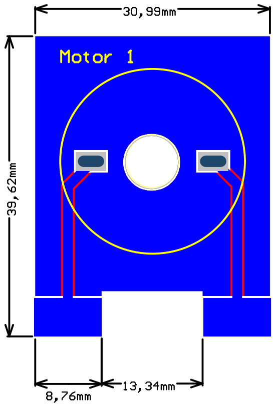
Magnetorquers PCB

Description:
That PCB will support the magnetorquers of this project and will be attached to the Testbed PCB with 4 screews.
The Magnetorquers will be mounted in a 3D printed Cubesat on that PCB.



Designer's signature:	Sheet title: Magnetorquers PCB	Dpto. Electronica y Tecnologia de Computadores University of Granada C/ Fuente Nueva, s/n, 18001 Granada, Granada, Spain Sr. Andres Roldan Aranda	
	Project title: GranaSAT-I		
Supervisor's signature:	Designer: Victor Burgos Gonzalez		
	Supervisor: Andres Roldan Aranda	**Date: 01.05.2015 Revision: 1** **Sheet 4 of 5**	

Motor PCB



Description:
That PCB will be placed in vertical position in order to have the ortogonal to Z axis for the actuators, in this case, X and Y axes.

Designer's signature:

Supervisor's signature:

Sheet title: Motor PCB

Project title: GranaSAT-I

Designer: Victor Burgos Gonzalez

Supervisor: Andres Roldan Aranda

Dpto. Electronica y Tecnologia
de Computadores
University of Granada
C/ Fuente Nueva, s/n, 18001
Granada, Granada, Spain
Sr. Andres Roldan Aranda

Date: 01.05.2015 Revision: 1
Sheet 1 of 10

4.1.2.3 3D view of the PCB Design

In this section, the PCBs shown in the section 4.1.2.2 will be represented in 3D view. That models will be used as reference in order to have an idea of the real size and aspect of the final PCB implementation.

That task has been carried out linking each footprint to a 3D model. The 3D models have been downloaded from `http://www.3dcontentcentral.es/` but some of then have been designed for the author of this project in Solidworks.

The figure show the 3D model of this project, one from each side of the PCB.

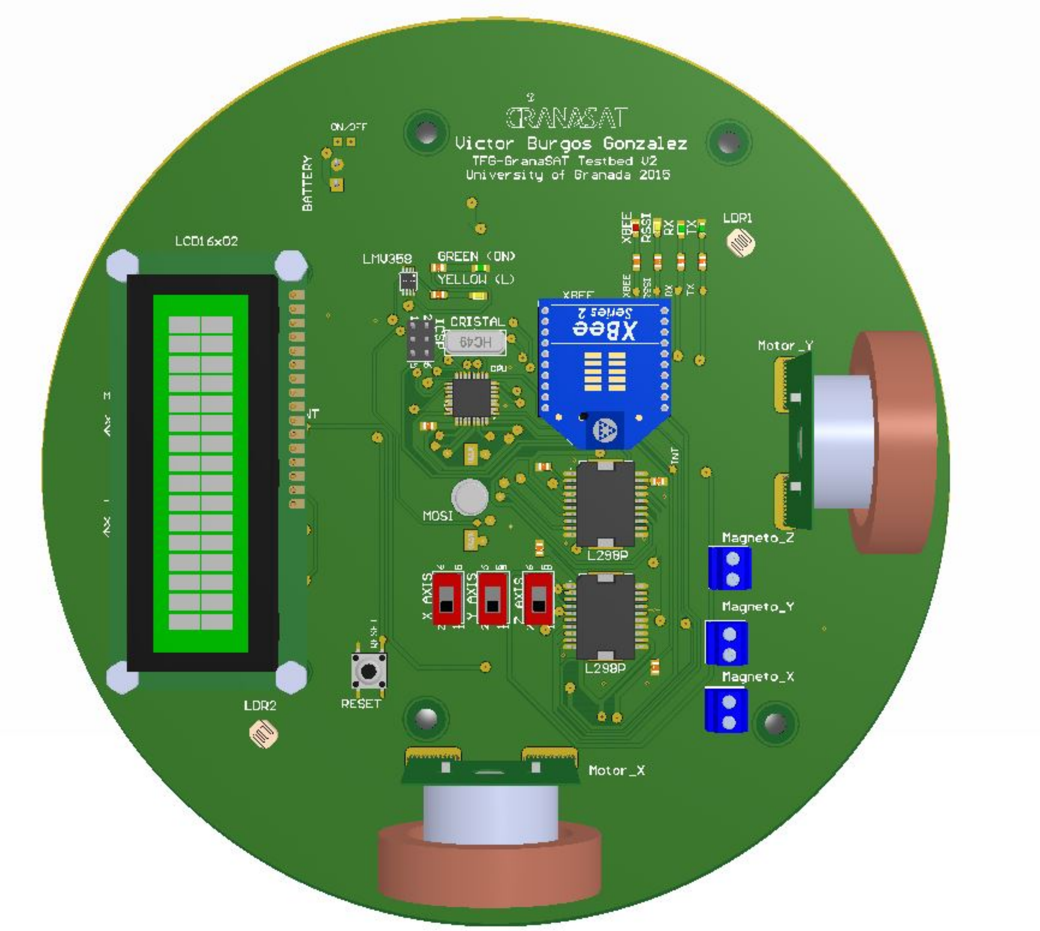


Figure 4.32 – *3D View of the top side*

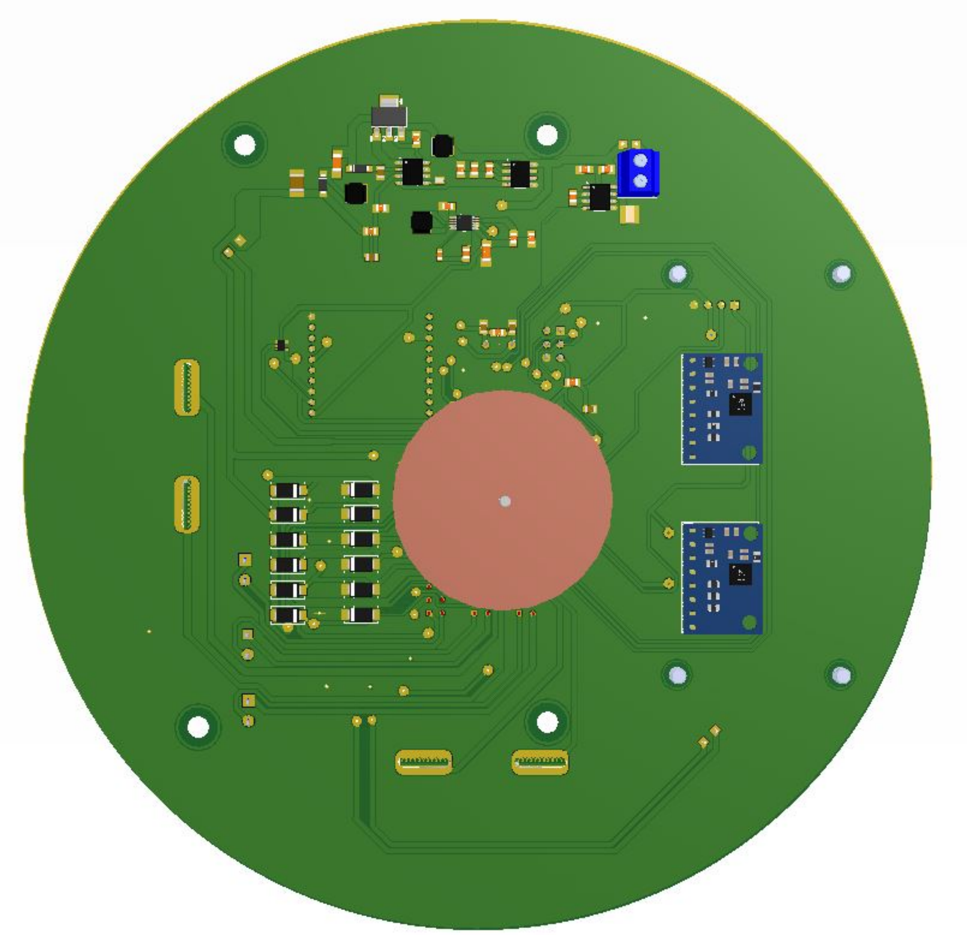


Figure 4.33 – *3D View of the bottom side*

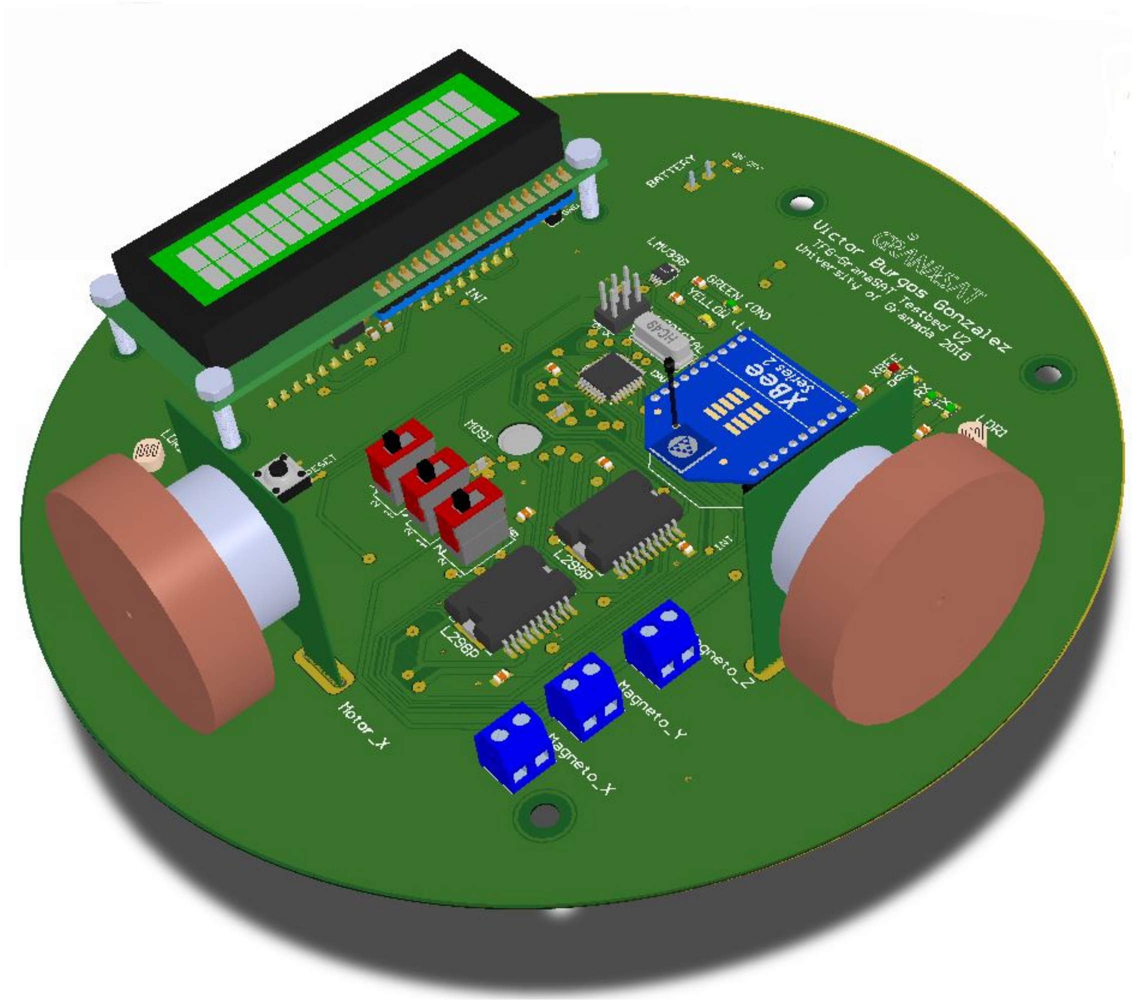


Figure 4.34 – *3D View of the right side*

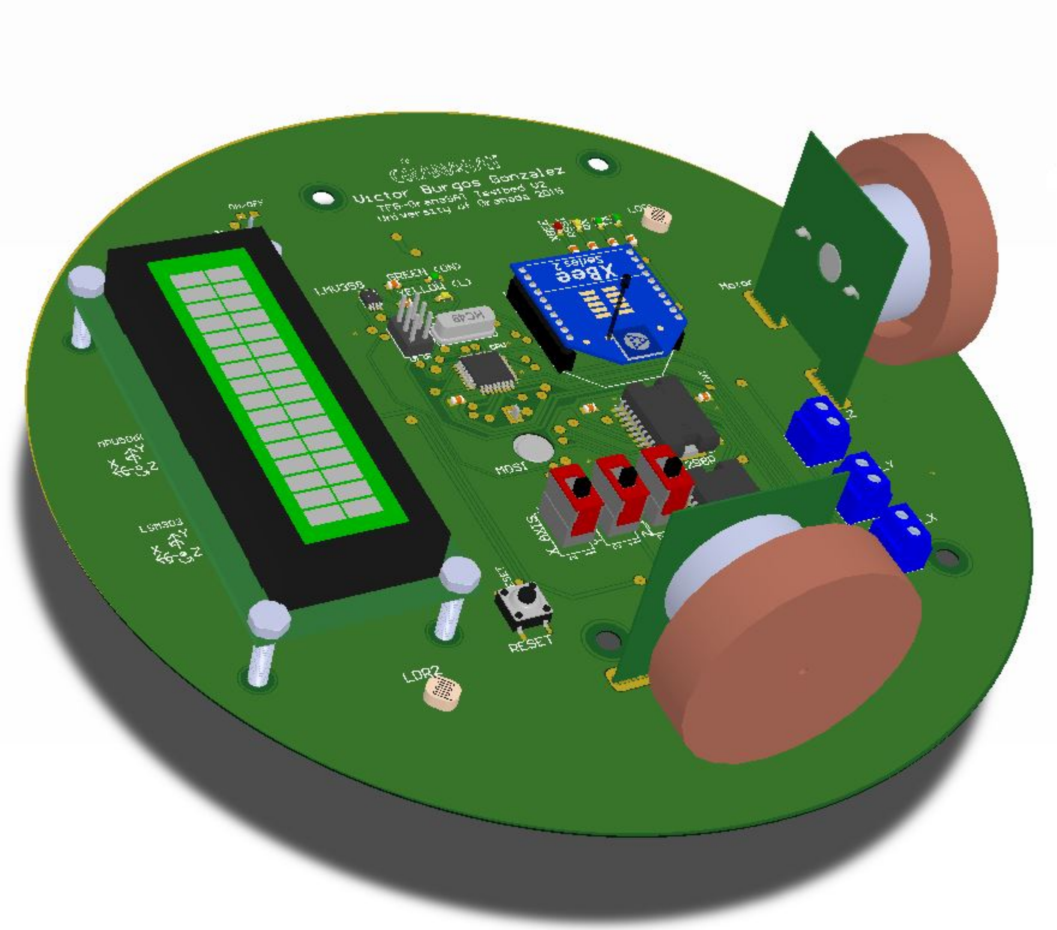


Figure 4.35 – *3D View of the left side*

This `https://www.youtube.com/watch?v=aF2nNklm4gU` have a video of the 3D view of that PCB.

4.2 Mechanical Design

The first step in the mechanical design was acquire the hemisphere where the electronics were located, because that dimension was the decisive for the Air-bearing, in order to perfectly fit with it, and for the PCB design (see section 4.1.2.1).

In order to have a better design and improve the quality of the 3D model, a good option is to make the 3D model in Solidworks of the bought hemisphere, and adapt the Air-bearing to it.

4.2.1 Hemisphere

The hemisphere chosen have the real use for decoration, but it has the necessary characteristics for the correct operation in our project.

The model is a fillable hollow acrylic ball with the hang tabs (see figure 4.36), from Complex Plastics. It is a sphere divided in two hemispheres, using only one for the Testbed.

The hemisphere has the dimensions shown in the schematic of the mechanical design of this part in the following page.

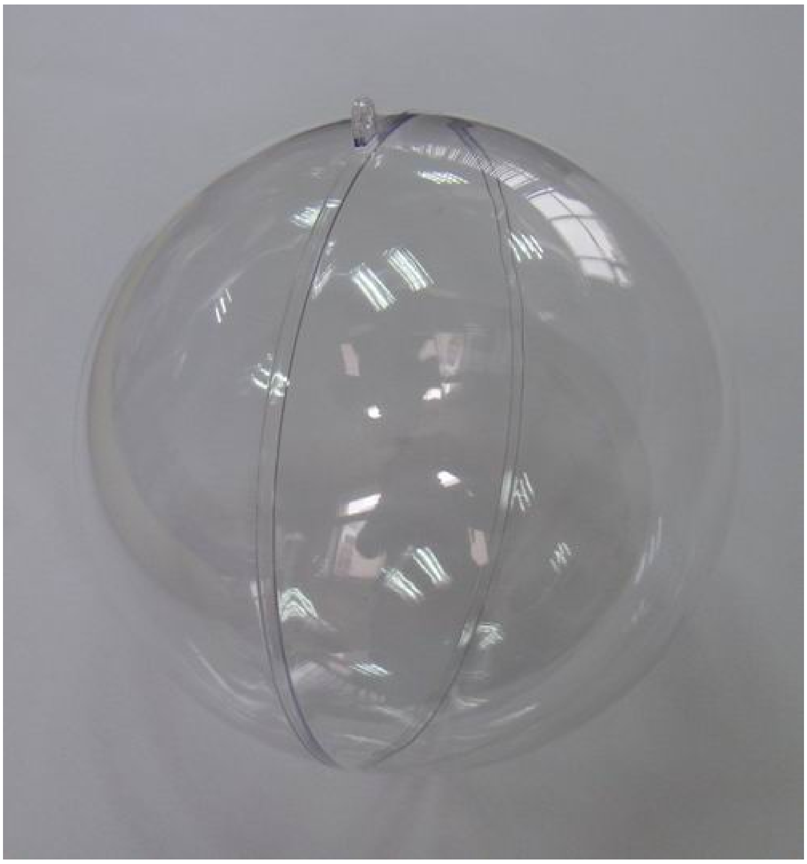
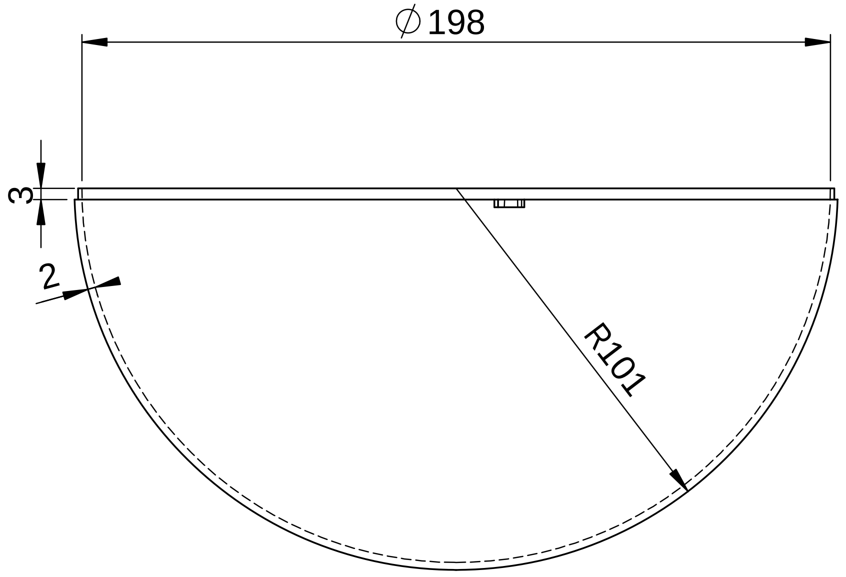
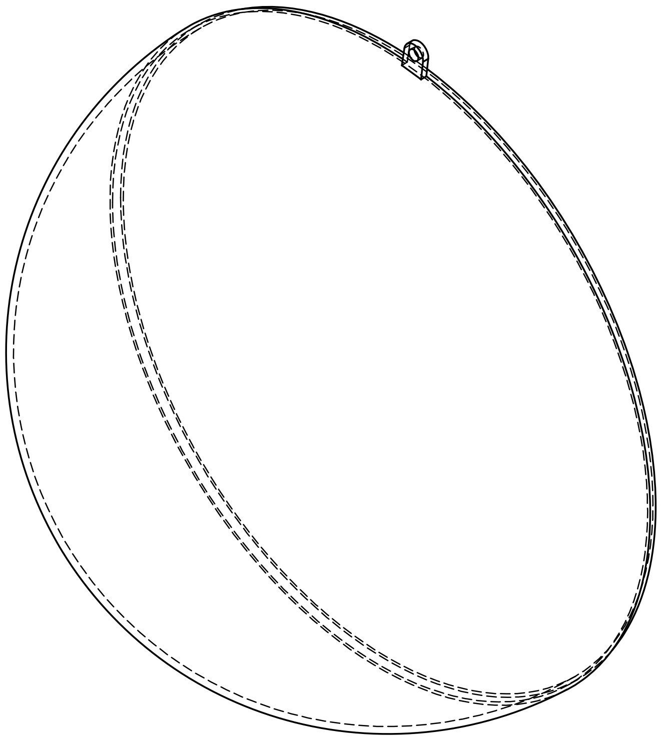


Figure 4.36 – *Clear hemisphere*





Designer: Víctor Burgos González	**TestBed part**	**Date:**	03/01/2015	**GranaSAT** **TESTBED**
Supervisor: Andrés M. Roldán Aranda		**Material:**	Acrylic	
Hemisphere		**Format:** **DIN A4**	**Scale:** 1:2	

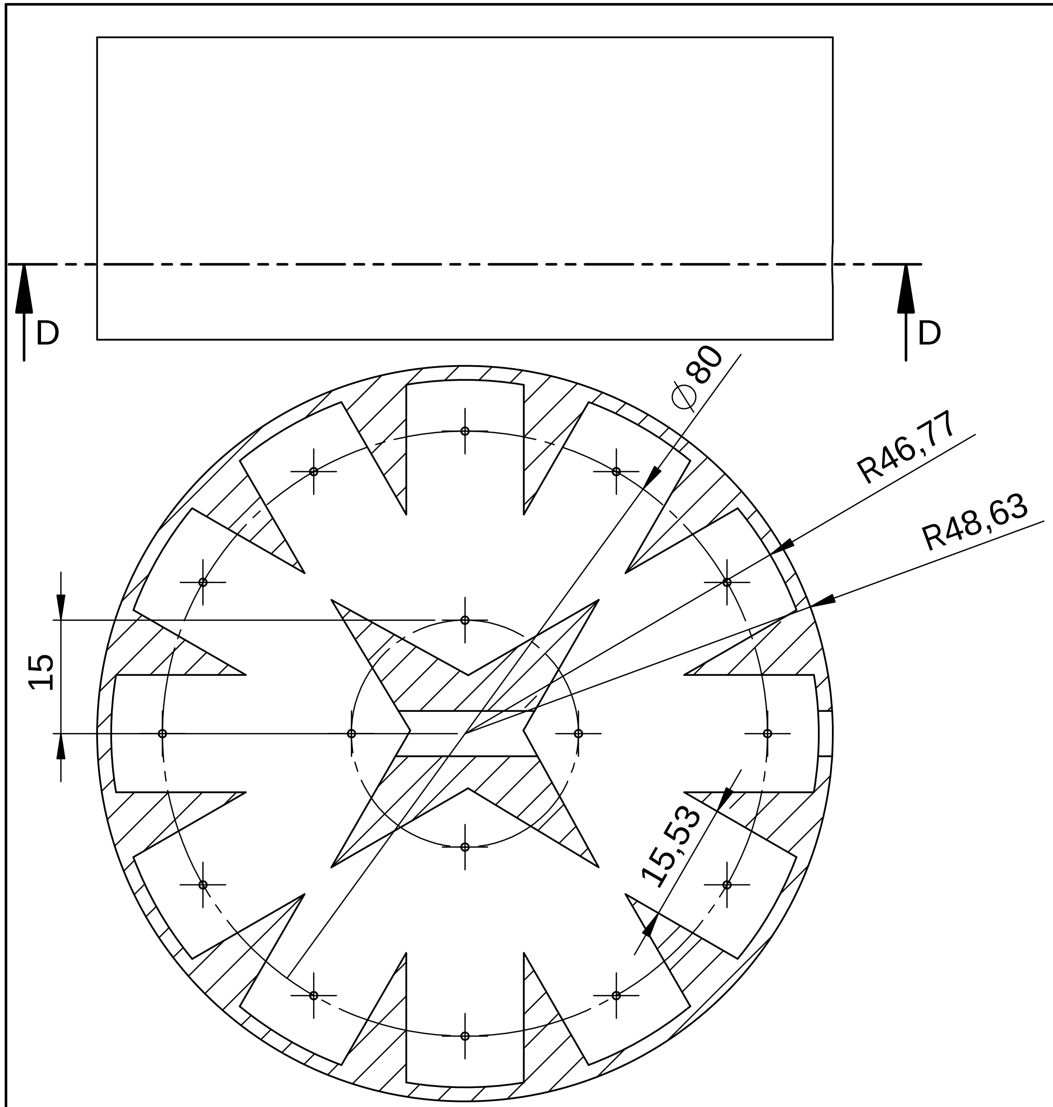
4.2.2 Air-bearing

The Air-bearing is designed with the intention to print it in a 3D printer (see Appendix 5.3.2, where the printing process is described).

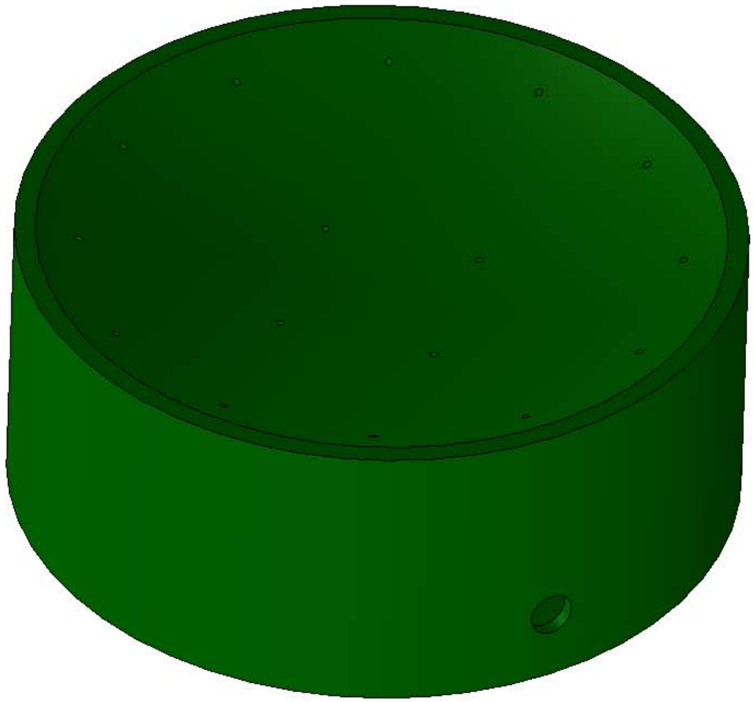
The model that we have studied and adapted to our project is the FloatSat design, seen in chapter 3, section 3.1, using the reverse engineering process.

Once the design was though, the next step is transfer that design to SolidWorks, in order to create a 3D model to print it in the 3D printers that we have in the ECTD laboratories. The following pages show the schematics of the Air-bearing design, with the most relevant dimensions and points of view. The first page is the schematic for the first model that was designed and built. See section 5.4 for more information about that model, and the tests that it had during the project.

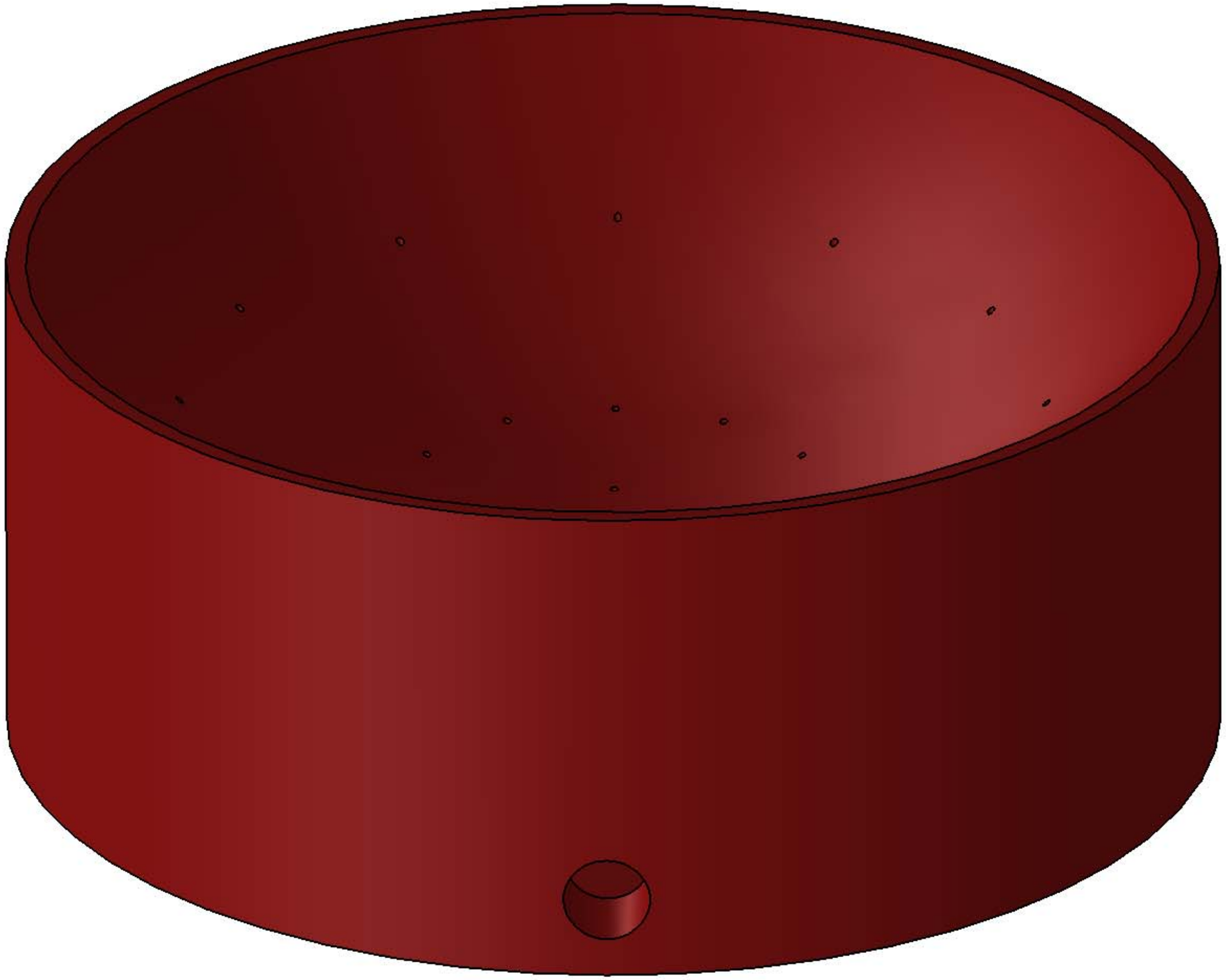
- The first schematic shows a section for the first model that was design for the Testbed. That model have been used for some tests during the project. Representing the dimensions of the main items of that model.
- The second schematic shows the 3D model of the final air-bearing.
- The third one is a section of the inside of that final model.
- The fourth schematic represents the holes distribution designed.
- Finally, the last one, another section for that model.



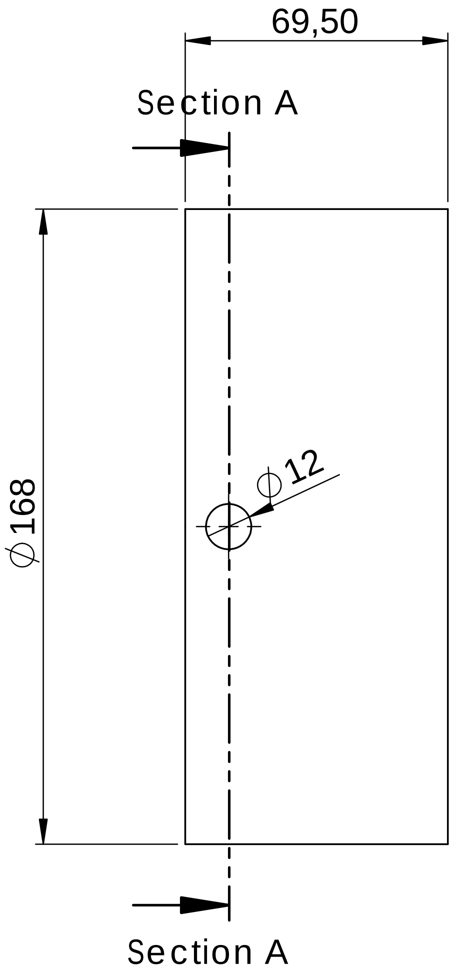
SECTION C
ESCALA 1 : 1

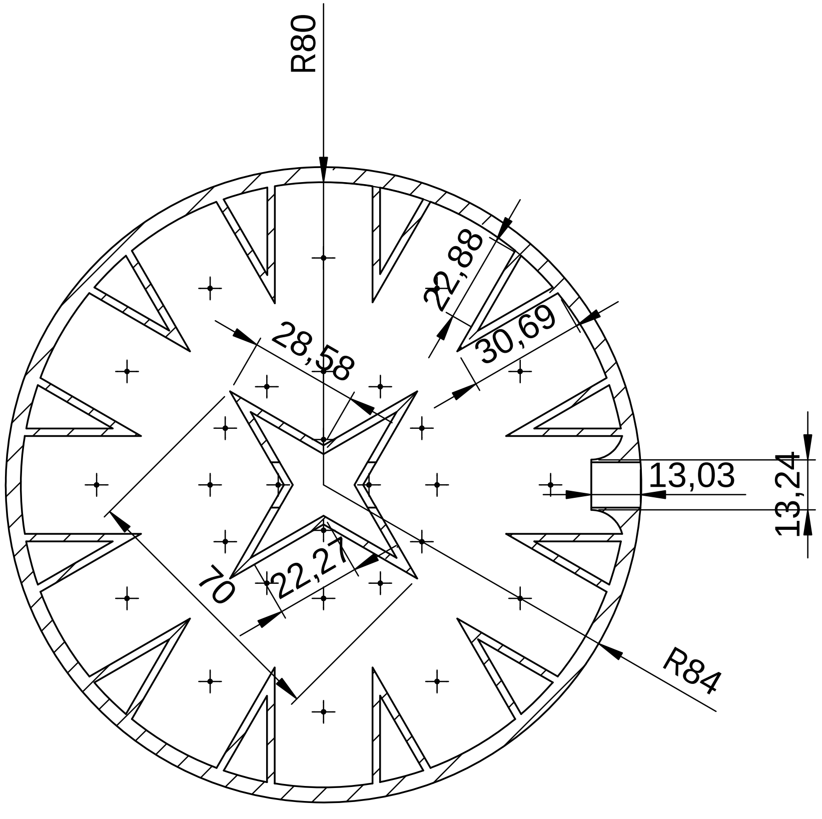


Designer: Víctor Burgos González	**Air Bearing**	**Date:** 12/15/2014		**GranaSAT** **TESTBED**
Supervisor: Andrés M. Roldán Aranda		**Material:** ABS		
Air-bearing 1st model		**Format:** **DIN A4**	**Scale:** 1:1	



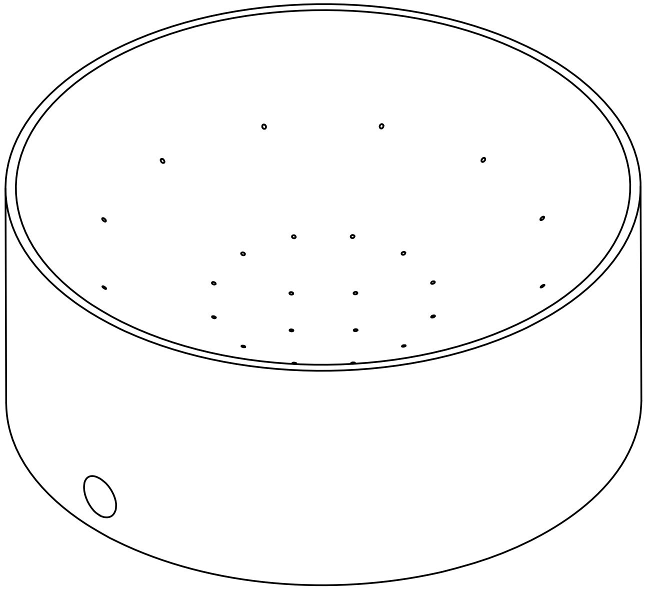
Designer: Víctor Burgos González	**Air-bearing**	**Date:**	15/01/2015	**GranaSAT** **TESTBED**
Supervisor: Andrés M. Roldán Aranda		**Material:**	ABS	
Air-bearing 3Dmodel		**Format:** **DIN A4**	**Scale:** 1:1	

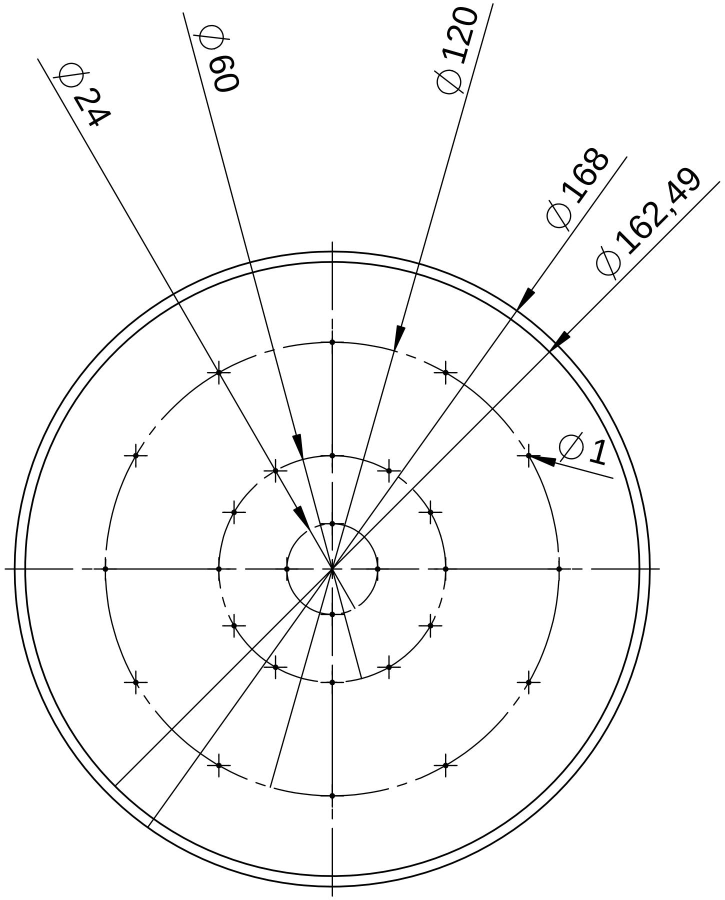




Section A

Designer: Víctor Burgos González	**Air-bearing**	**Date:**	15/01/2015	**GranaSAT** **TESTBED**
Supervisor: Andrés M. Roldán Aranda		**Material:**	ABS	
Air-bearing section		**Format:** **DIN A4**	**Scale:** 1:2	





Designer: Víctor Burgos González

Air Bearing

Date: 15/01/2015

GranaSAT
TESTBED

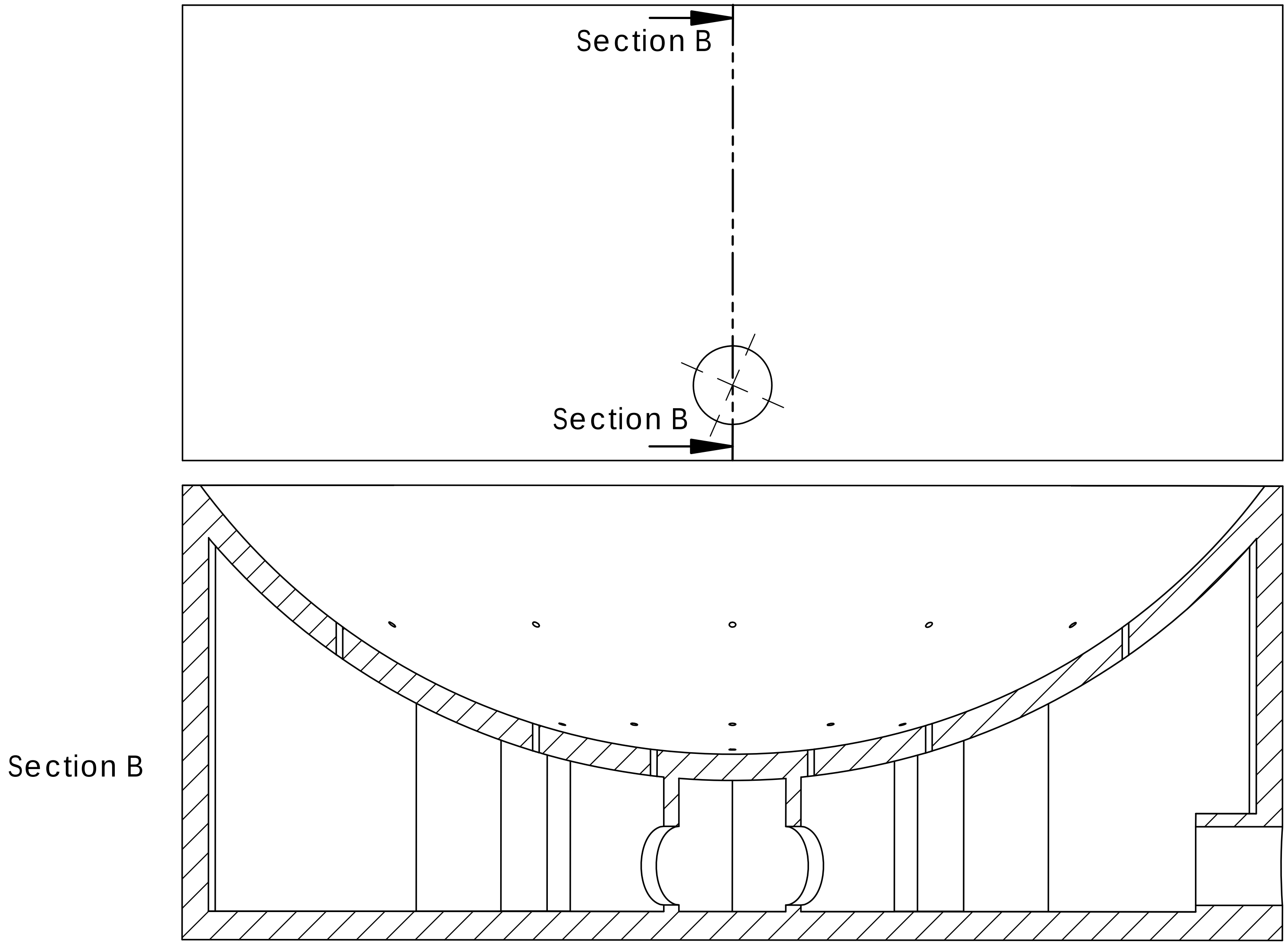
Supervisor: Andrés M. Roldán Aranda

Material: ABS

Holes distribution

Format: **DIN A4**

Scale: 1:2



Designer: Víctor Burgos González	**Air Bearing**	**Date:** 15/01/2015	**GranaSAT** **TESTBED**
Supervisor: Andrés M. Roldán Aranda		**Material:** ABS	
Air-bearing section		**Format:** DIN A4	**Scale:** 1:1

The interactive 3D model, to have a better point of view of the design.

For the design of the Air-bearing, the most relevant limitation that exists is the building process, because of the technology that we are using, 3D printers. The printer works with FFF depositing a filament of a certain material (ABS in our case) on above the same material, in order to create a joint by heat and/or adhesion [2]. Because of this way to build the models (from bottom to top of the model), there are some restrictions in the fabrication to control that the next layer is always on top of each other to form the object.

The triangles that shows the second schematic (section A of the Air-bearing) and in figure 4.37, are the support to avoid that mechanical problems. The calculation of the dimension of each triangle is made with some practical tests measuring how long can extrude the printer without support, depending on the extruder speed and the temperature for the ABS.

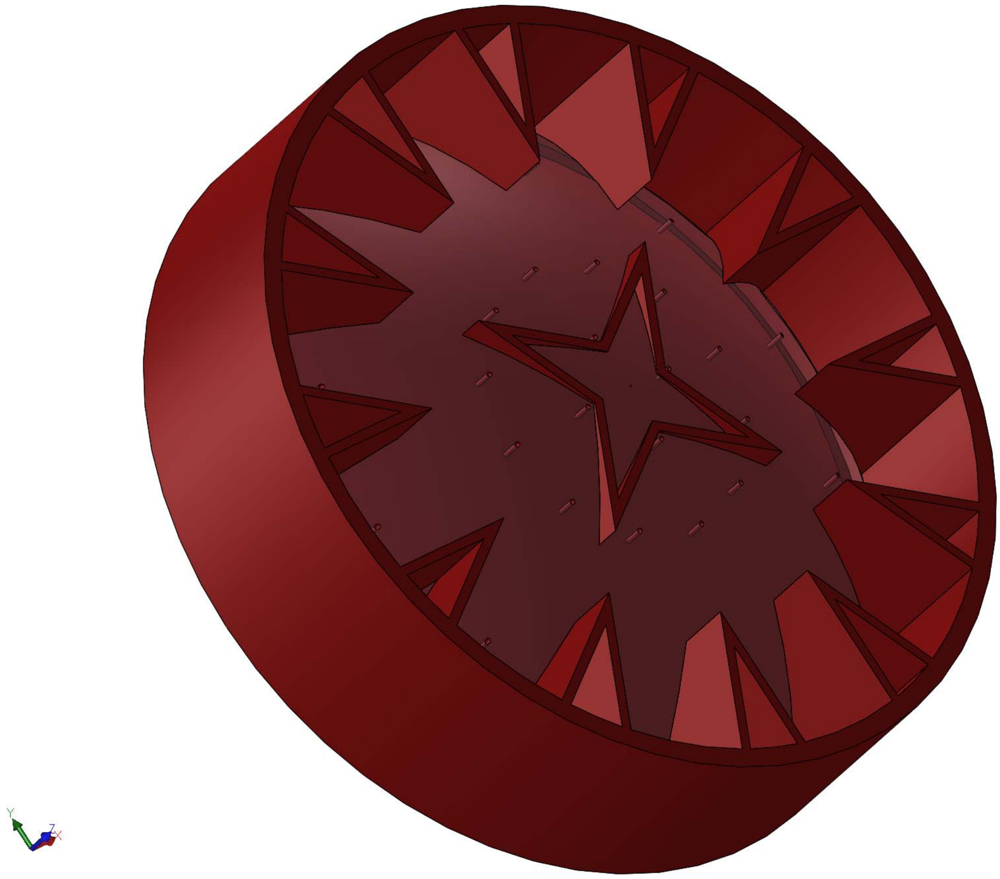


Figure 4.37 – *Air-bearing section to show the support triangles*

4.2.3 Air-compressor

In this section it sill be described the available air compressor that GranaSAT development team hasin the Faculty of Science, in Granada. The model is CP2525, of Black&Decker (see figure 4.38).

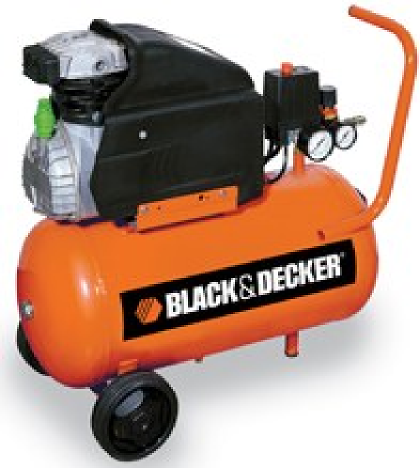


Figure 4.38 – *Black&Decker CP2525 air compressor [8]*

The main characteristics of the compressor are described in the table 4.9.

Oil lubricated
24 liters tank
Pressure regulator and 2 pressure gauges
Plastic shroud protects hot and rotting parts
230V / 50Hz
Maximum working pressure of 8 BAR and the motor has a power of 2HP peak

Table 4.9 – *Main characteristics of CP2525 compressor [8]*

4.2.4 Reaction Wheels

When the reaction wheels are designed for a Cubesat, there are two main limitations, volume and mass. In this case, we are going to use some limitations too, in order to have the most similar possible design for the future Cubesat GranaSAT, but using the choice of the material instead of the mass, because of the limitations in cost and availability of the material.

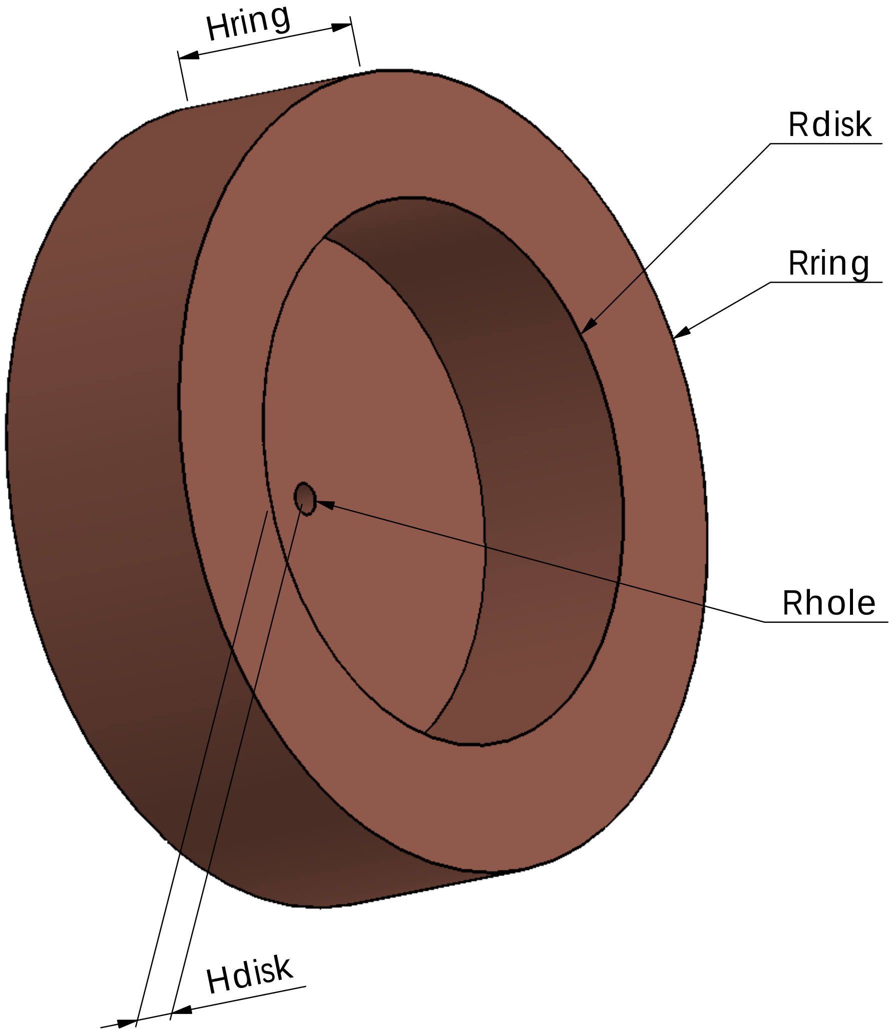
The material chosen is the bronze, because it has a density of $8890 Kg/m^3$, one of the highest densities that we could acquire (see table 4.10) because of its high value and its availability in Granada.

Material	**Density** (Kg/m^3
Copper	8960
Bronze	8890
Iron	7870
Aluminium	2700

Table 4.10 – *Density of different materials*

The second limitation is the volume, the dimension of the wheel. That limitation was fixed for two variables, the availability of the different sizes that we could buy and the dimension of the Cubesat that we will build in a future(see section 2.1), because a 1U-Cubesat has $10 \times 10 \times 10cm^3$ and weights about 1Kg.

In order to name each dimension of the reaction wheel, the following page shows the parameters that are used for the design.



Designer: Víctor Burgos González	**Reaction wheel**	**Date:** 15/05/2015	**GranaSAT** **TESTBED**
Tutor: Andrés M. Roldán Aranda		**Material:** **Bronze**	
Parameters description		**Format:** **DIN A4**	**Scale:** 3:1

The dimensions chosen are the shown in table 4.11.

Parameter	**Designed model**	**Measured built model**
Rring (mm)	18	17.95
Rdisk (mm)	12.2	12.2
Hring (mm)	10	10
Hdisk (mm)	2	2.03
Rhole (mm)	0.7	1.15

Table 4.11 – *Reaction wheels dimensions*

The measured dimension were made with a caliber with accuracy of 0.01mm, because the designed model had some differences with the build model (made by the CIC of the UGR, especially in the Rhole parameter, so in the implementation process was required a special glue to fix the reaction wheel to the motor axis. The assembly of the wheel with the motor is shown in the figure 4.39.

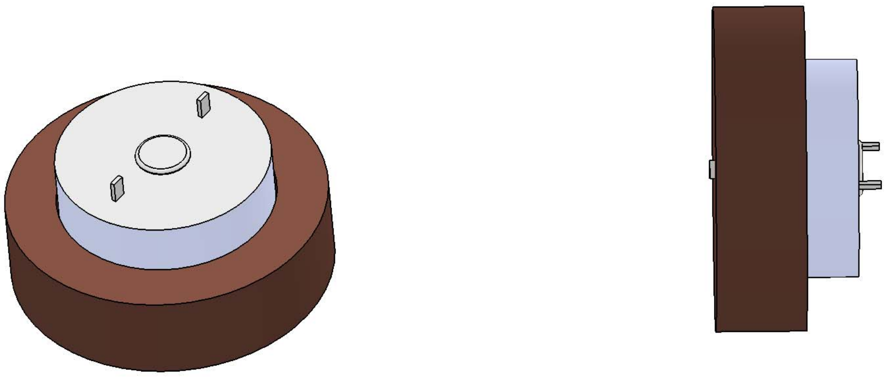


Figure 4.39 – *Assembly wheel-motor*

For the calculation of the wheel mass, the equations 4.2.1 are used:

$$m_{total} = m_{disk} + m_{ring} - m_{hole} = \underbrace{\rho\pi R_{disk}^2 H_{disk}}_{m_{disk}} + \underbrace{\rho\pi(R_{ring}^2 - R_{disk}^2)H_{ring}}_{m_{ring}} - \underbrace{\rho\pi R_{hole}^2 H_{disk}}_{m_{hole}} \tag{4.2.1}$$

Where ρ is the density of the material chosen, in this case bronze with a density of $8890 Kg/m^3$ approximately, m_{total} is the total mass of the reaction wheel, divided in m_{disk}(the mass of the disk), m_{ring} the ring mass and finally the subtraction of the hole mass, m_{hole}. The equation 4.2.1 can be solved now with the dimensions measured and compared with the mass measured with a balance graduated in grams. The table 4.12 shows the comparative of this measurements.

Parameter	m_{disk} (g)	m_{ring} (g)	m_{hole} (g) m_{total} **(g)**	
Design calculation mass	8.31	48.92	0.16	57.06
Build calculation mass	8.43	48.41	0.44	56.40
Build measured mass	-	-	-	57

Table 4.12 – *Reaction wheels mass*

The difference between the measured mass and the calculation mass can be caused by the approximation done in the density, because the bronze is a copper alloy, and the value of its density may differ. Moreover, the balance only has grams accuracy, so maybe the value is rounded.

As the table 4.12 prove, the majority of the mass in the reaction wheel is in the ring of the wheel. The reason of that design is that, with this characteristics, we will have a better inertia momentum with the minimum weight possible.

With the mass calculation we could acquire the inertial momentum that each designed wheel generates, see equations 4.2.2, 4.2.3 and 4.2.4. That equations will be calculated with the mass of the calculation build model.

$$I_{disk} = \frac{(m_{disk} - m_{hole})R_{disk}^2}{2} = 0.628 \times 10^{-6} Kgm^2 \tag{4.2.2}$$

$$I_{ring} = \frac{m_{ring}(R_{disk}^2 + R_{ring}^2)}{2} = 11.403 \times 10^{-6} Kgm^2 \tag{4.2.3}$$

$$I_{total} = I_{disk} + I_{ring} = 12.031 \times 10^{-6} Kgm^2 \tag{4.2.4}$$

As we said before, the most of the inertial momentum is located in the wheel ring.

In the next chapter, when the characterization of the motors was made, it will be calculated the maximum angular momentum with that values of inertial momentum.

4.2.5 Assembly of the whole parts

In this section, will be represented the whole 3D model design of the project, placing every part in its place.

The figure 4.40 shows how the system is assembled, with every 3D model mounted in its correct position.

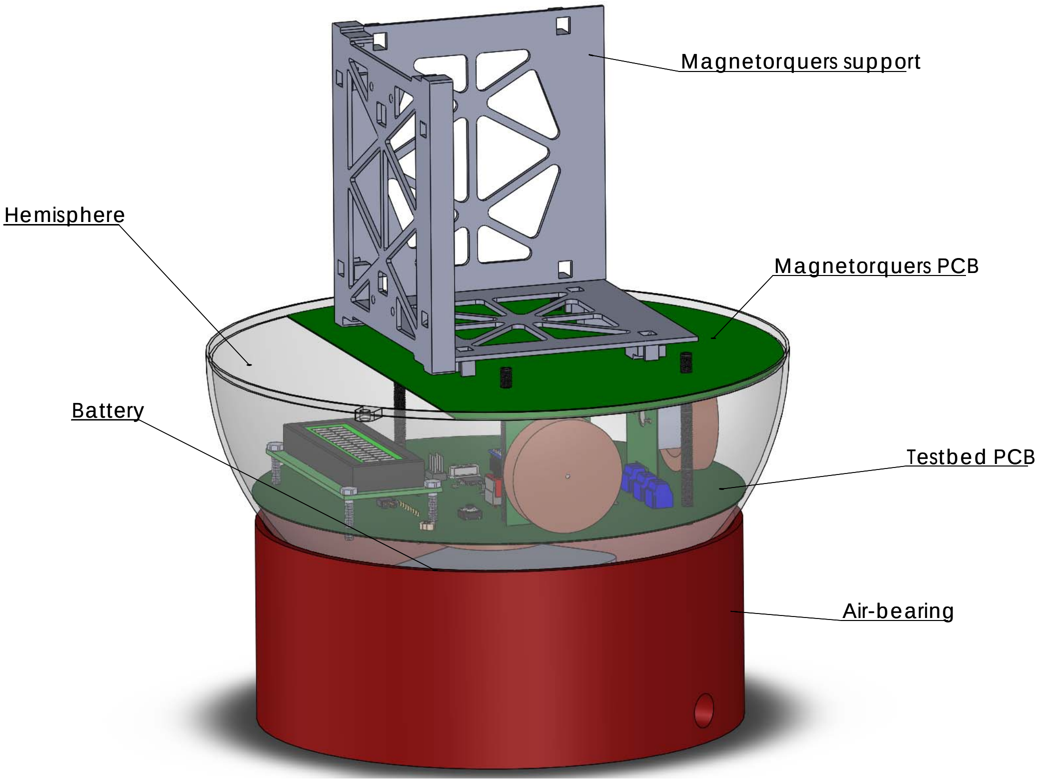


Figure 4.40 – *Testbed assembly*

4

There is a assembly video of the GranaSAT Testbed in this `https://www.youtube.com/watch?v=mojIS9s4XxM`.

For the assembly of Testbed PCB and Magnetorquers PCB four M4 screws have been used, as the figure 4.40 shows.

4.3 Software Design

In this section will be studied the necessary software design for the system. That design has two parts: the onboard software, and the GS software. Both systems have to be synchronized in order to work properly, so the first issue that we have to explain is the communication protocol used.

4.3.1 Communication protocol

The communication protocol was designed for Carlos Valenzuela Morales [59], another GranaSAT team member, anyway here is a summary of the main characteristics.

The first thing that we have to take into account is the quantity of data that we want to send to the GS. That protocol is not only prepared for the Testbed, but also for the Cubesat systems too, so the rest of the GranaSAT team have included their data in this transmission. Each parameter has the data type of the Arduino code more efficient, using the minimum bytes for each value. The table 4.13 shows what type of data can be used in that protocol, but for the GranaSAT Testbed we will only use float and int type.

Data type (Arduino code)	**Description**	**Range**
byte	unsigned, 8 bits	[0, 255]
char	signed, 8 bits	[-128, 127]
int	signed, 16 bits	[-32767, 32767]
unsigned int	unsigned, 16 bits	[0, 65535]
unsigned long	unsigned, 32 bits	[0, 4294967295]
unsigned int (12)	unsigned, 12 bits	[0, 4096]
float	signed, 32 bits	[-3.4028235E38 , 3.4028235E38]

Table 4.13 – *Data type used [59]*

In order to receive the data correctly, the transmission of the parameters should have an order in both systems, and it is used two delimiters for the sent frame. Moreover, it will have a CRC to detect the errors in each frame. Knowing the start and the end of the frame, will facilitate the read of the data and the placing of the values in the dashboard of the GS.

The error control is made with the protocol 804.15.4 of the wireless communication, Xbee module. The errors correction is made in MAC level, with the ACK confirmations from the receptor [51].When the maximum number of times to retransmit the frame without receive the ACK from the receptor occurs, that frame will be incorrigible. The CRC will act in that cases.

The figure 4.41 shows the flow diagram designed for that protocol.

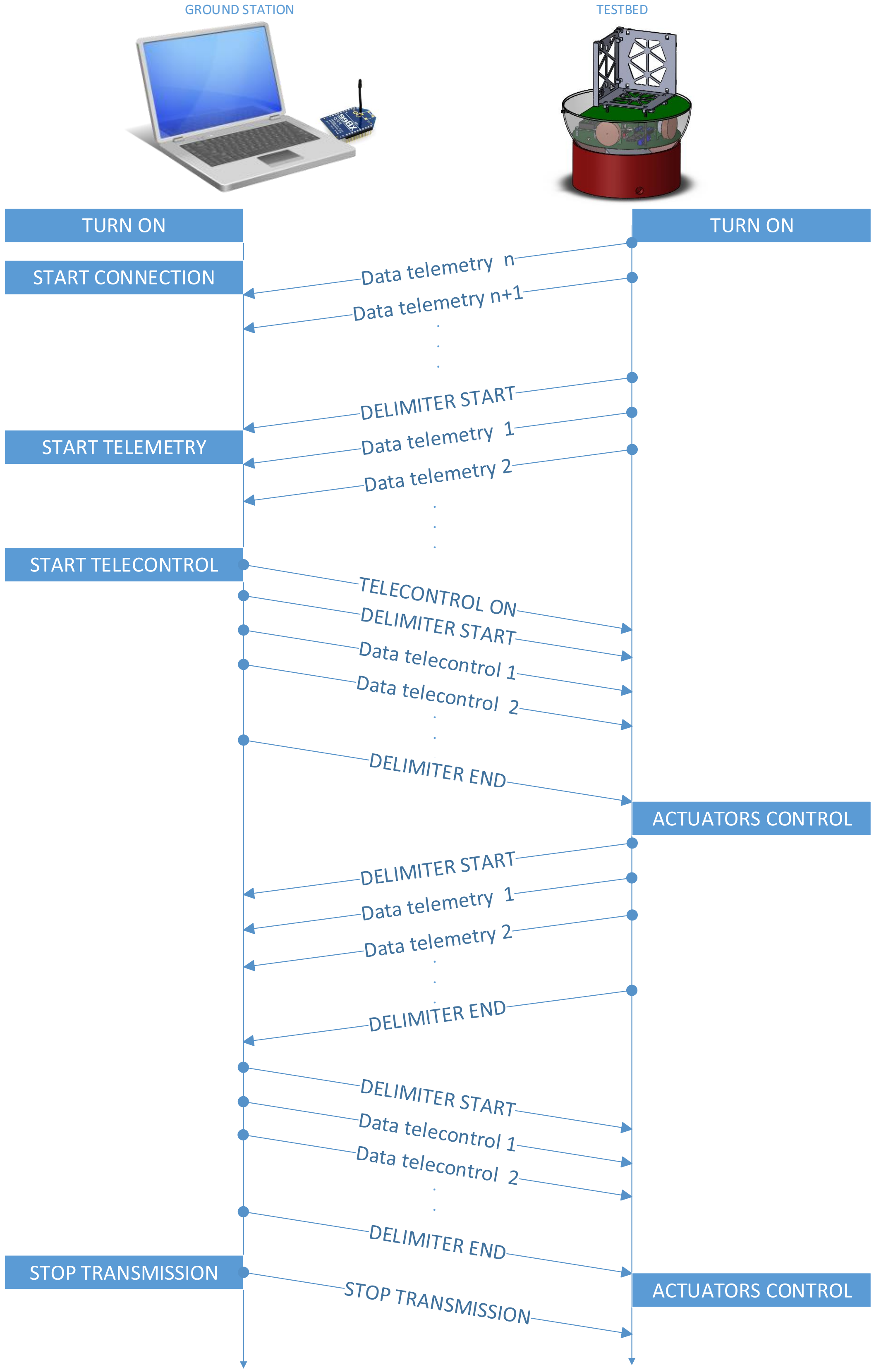


Figure 4.41 – *Flow diagram*

Now, each phase of this communication protocol is described in the following enumeration:

- **TURN ON**. Each system has to be turned on to start the protocol. The GS will be a PC connected to a Xbee Module, and the Testbed will be connected to the battery to start the process flow.
- **START CONNECTION**. Both Xbee modules (GS and Testbed) are connected to each other. That function is in the GS side of the flow diagram but is common to both systems at the same time.
- **START TELEMETRY**. In the dashboard of the GS the reception is started clicking in a button. Before of that phase, the Testbed is sending data but the receptor is not working yet. When the telemetry is started, the GS wait until that the DELIMITER START is received.
- **START TELECONTROL**. Exist a button to turn on the telecontrol. When the telecontrol is activated, a message from the GS to the Testbed is sent to inform of that action, and the delimiter is sent to start the telecontrol data transmission.
- **ACTUATORS CONTROL**. When the telecontrol commands are received in the Testbed, the microprocessor prepares and sends the parameters necessary to control the actuators. Now, the Testbed start to send the next frame of telemetry.
- **STOP TELEMETRY/TELECONTROL**. The transmission of the data is stopped in this phase, clicking on the button STOP in the dashboard of the GS.

4.3.2 On-board software

On-board Software is controlled by the microprocessor in the Testbed PCB and it has the following functions:

- Read the data sensors.
- Display the values chosen in the LCD.
- Control the actuators.
- Transmit the data to the GS with wireless communication.

In order to have a more clear description of the software design, the state diagram of the on-board software is shown in the figure 4.42.

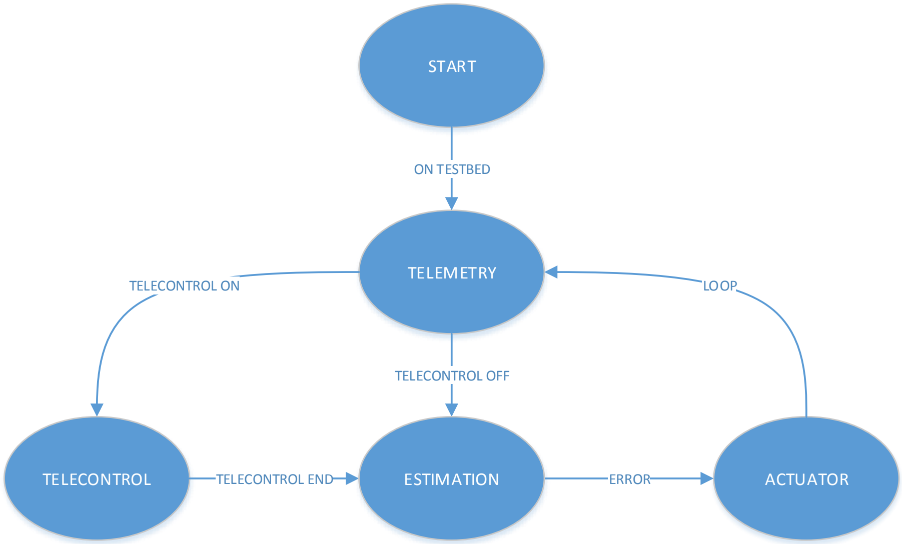


Figure 4.42 – *On-board State diagram*

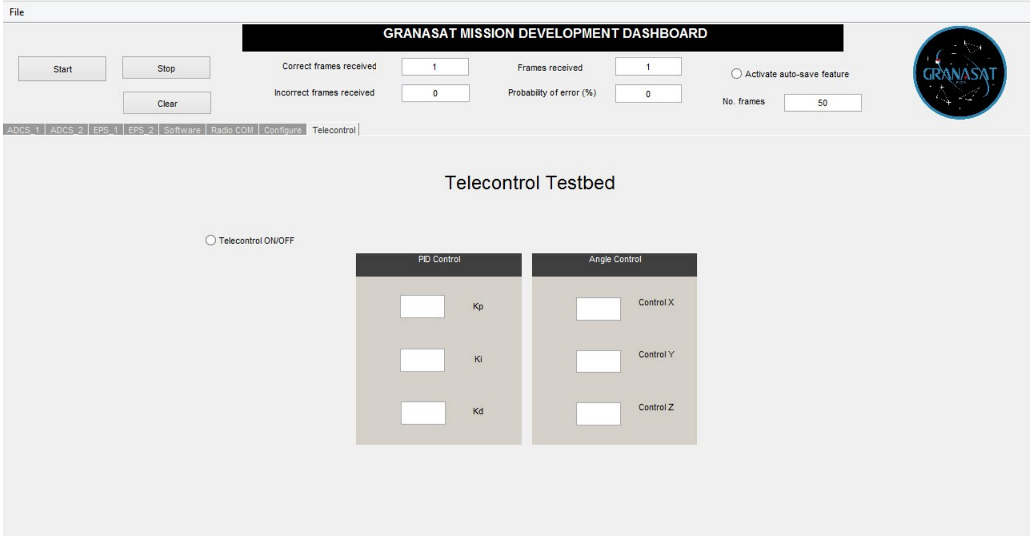
In that state diagram we have five different states, which are going to be described now:

- **START**. In that state, the system is waiting to be turned on. It will only be occupied the first one time.
- **TELEMETRY**. When the system is in that state, it will be measuring the data from the sensors until the reading process is ended and the system will have to decide if the next state is TELECONTROL or ESTIMATION, with a telecommand sent from the GS (see 4.41). When telemetry state ends, the system send the data measured to the GS through the wireless link created with the Xbee modules.
- **TELECONTROL**. If the Telecontrol command from the GS is ON, the TELECONTROL state will be the next in the state diagram, where the system will receive the data from the PC. The system will stay here until the telecontrol data is completely received.
- **ESTIMATION**. The estimation state will calculate the error of the system, using the PID controller. If the last state is TELECONTROL, it will have the telecontrol commands to calculate it, however, if the system comes from the TELEMETRY state, the determination of the error will be calculated with the default values.
- **ACTUATOR**. Finally, when the error is calculated, that state is the responsible for the control of the actuators. It will send the necessary commands and the system will come back to the TELEMETRY state, making a loop.

4.3.3 GS software

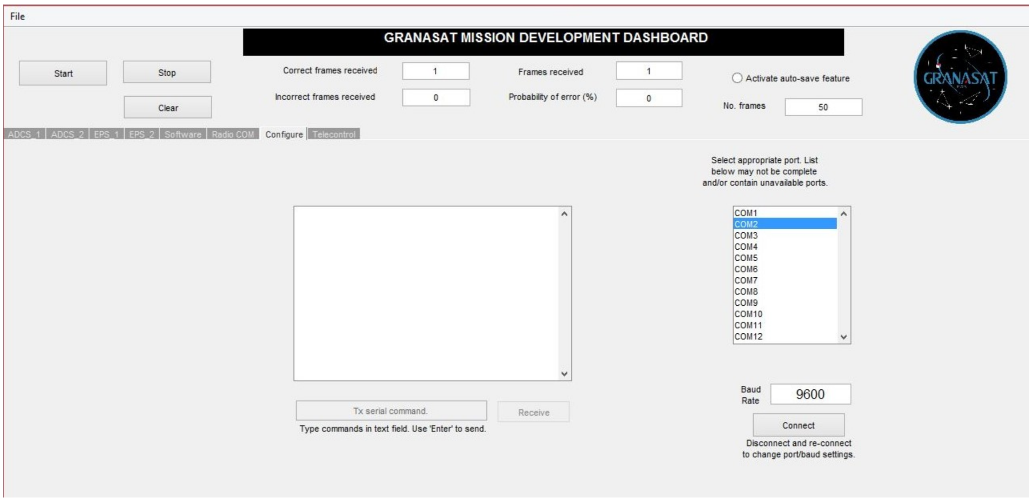
In the GS of our project we will have a graphic interface, a dashboard, showing the telemetry values received from the Testbed and sending the telecontrol commands there. That graphic interface will be developed with MATLAB, where the values will be shown in different tabs where the data will be organized and the configuration for the radio link can be changed.

That dashboard was firstly designed by Carlos Valenzuela [59], but now the design of another new tab was necessary for the telecontrol of the Testbed, and the modifications of the types of values for the sensors chosen. The figures 4.43, 4.44 and 4.45 show the panels used for this project, the first one, specially designed for the telecontrol Testbed. That panel allow us change the PID constants and fix the target angles for each axis. The second figure is used for the configuration of the link and it is adapted to the PC used for the measurements. Finally, the last one, adapted to the data type of the sensors.



Correct frames received: 1
Incorrect frames received: 0
Frames received: 1
Probability of error (%): 0
○ Activate auto-save feature
No. frames: 50
○ Telecontrol ON/OFF

Figure 4.43 – *Testbed telecontrol panel designed*



Correct frames received: 1
Incorrect frames received: 0
Frames received: 1
Probability of error (%): 0
No. frames: 50

Select appropriate port. List below may not be complete and/or contain unavailable ports.

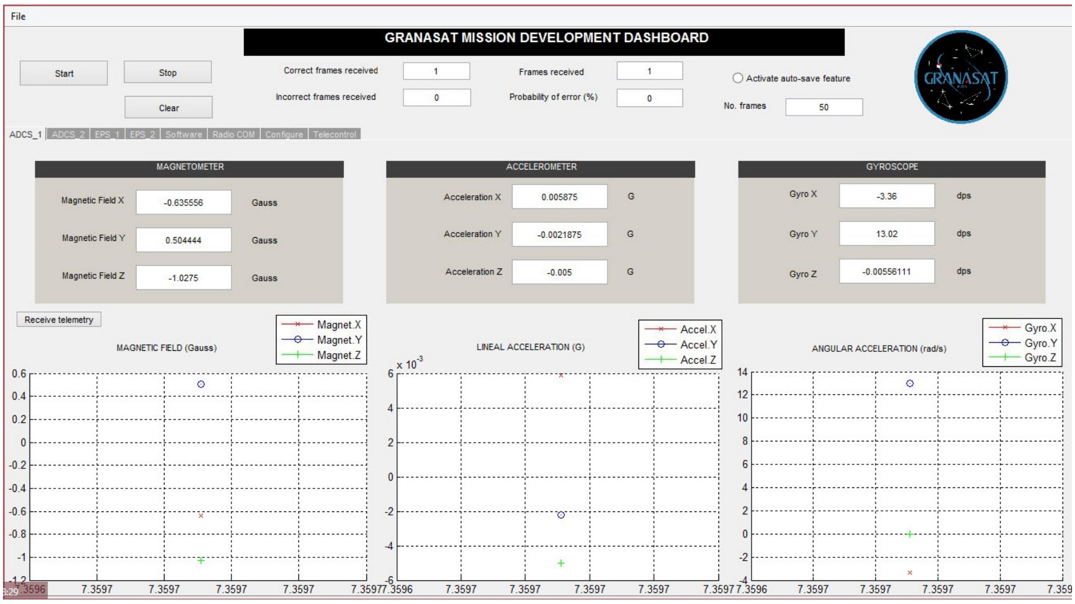
COM1
COM2
COM3
COM4
COM5
COM6
COM7
COM8
COM9
COM10
COM11
COM12

Baud Rate: 9600

Type commands in text field. Use 'Enter' to send.

Disconnect and re-connect to change port/baud settings.

Figure 4.44 – *Configuration panel used*



Correct frames received: 1
Incorrect frames received: 0
Frames received: 1
Probability of error (%): 0
No. frames: 50

Magnetic Field X: -0.635556 Gauss
Magnetic Field Y: 0.504444 Gauss
Magnetic Field Z: -1.0275 Gauss

Acceleration X: 0.005875 G
Acceleration Y: -0.0021875 G
Acceleration Z: -0.005 G

Gyro X: -3.36 dps
Gyro Y: 13.02 dps
Gyro Z: -0.00556111 dps

Figure 4.45 – *Sensors ADCS panel used*

CHAPTER

5

INTEGRATION, TESTS AND VERIFICATION

In this chapter, we will see the integration phase of this project, how each subsystem works with each other. Moreover, a characterization of the whole parameters for the motors will be done and an evaluation of the test taken.

5.1 Motor characterization

In this section, the motors will be evaluated, and some parameters necessary for the study of the reaction wheels will be calculated.

5.1.1 Power Supply - RPM Test

Purpose
One of the most relevant characteristics for the reaction wheels is the angular speed that the motors can achieve. That parameter will be measured with the reaction wheels and without them, to compare both results.

Material and environment needed
The materials used in this test are:

- Regulable power supply source

- Multimeters (to measure the voltage and current accurately)
- Motors
- Reaction wheels
- Tachometer
- PC and Excel Software.

The tachometer is the model DT-2234B [19], which is a photo tachometer type and it has the characteristics shown in table 5.1.

Range (RPM)	5 to 99999
Resolution for <1000RPM (RPM)	0.1
Resolution for >1000RPM (RPM)	1
Accuracy	± (0.05% + 1 digit).
Sampling Time (s)	1
Operating temp (C)	0 to 50
Charge Termination Current (mA)	150
Discharge temperature range (C)	-10 to +60

Table 5.1 – *DT-2234B Photo tachometer Specifications [19]*

That tachometer can save the last value, the maximum value and the minimum value measured. The maximum distance for the detection of the speed is 30 cm. A photography of that tool is shown in figure 5.1.

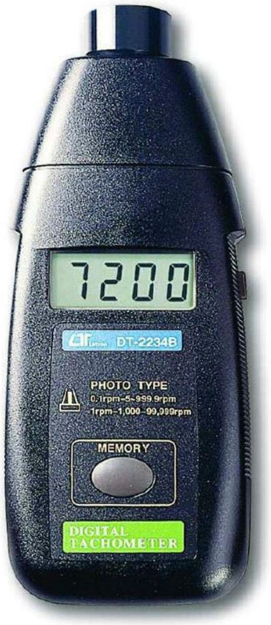
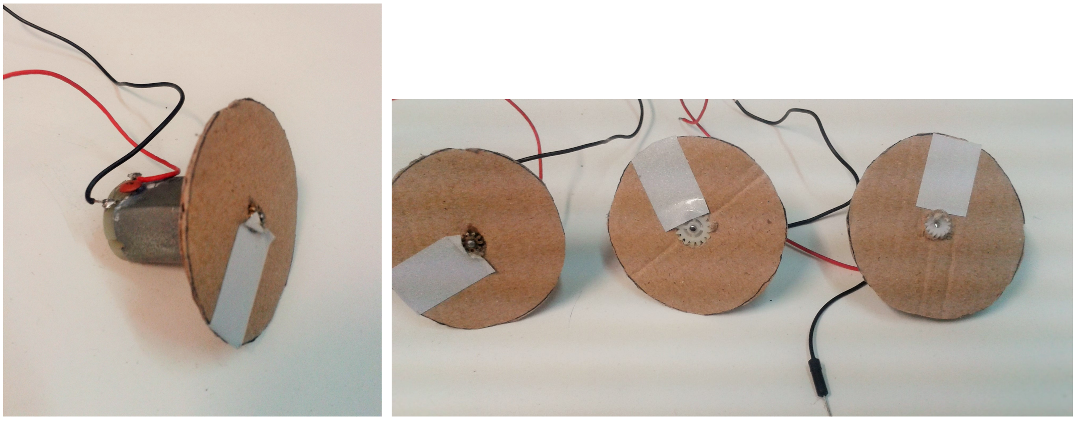


Figure 5.1 – *Tachometer DT-2234B [19]*

Results

The revolution of the wheels is not exactly constant, so in order to have the most accurate value, three measurements was taken and the average of that values was used for the motor characterization. The table D.1 contains the measurements taken with the tachometer DT-2234B for the motor Kysan RF-300CH, located in the appendix D, section D.1. That measurements were made with a piece of cardboard in the motor axis as shows the figure 5.2 with a reflecting sticker that the tachometer's kit includes to detect the angular velocity. Moreover, the figure 5.3 represents that measurements.



(a) (b)

Figure 5.2 – *Motors prepared for the VCC-RPM test*

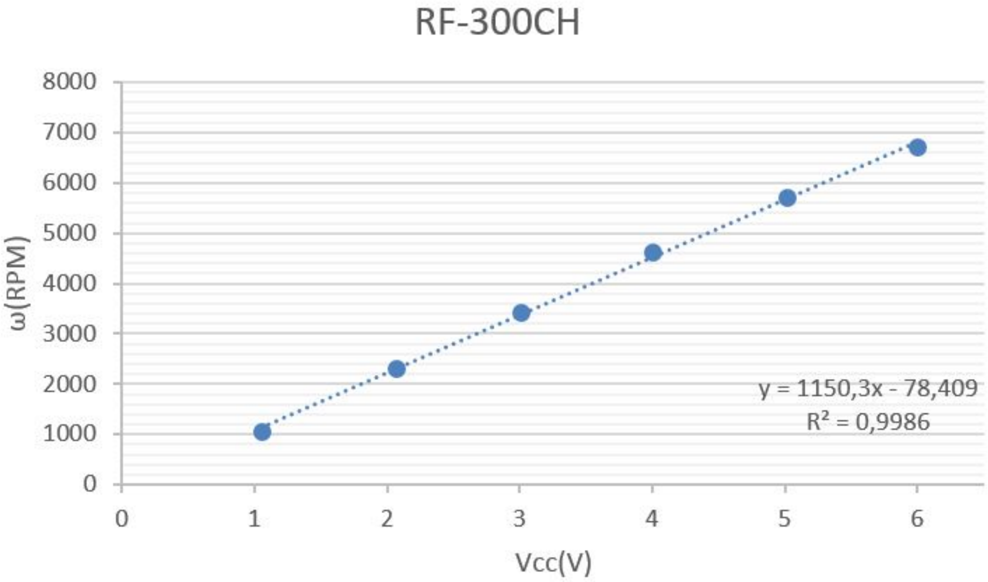


Figure 5.3 – *VCC-RPM results for Kysan RF-300CH*

That graph shows how the tendency of the characterization is practically lineal. The equation represented in the graph proves that evidence.

Now the same measurements were made for the rest of the motors (figures 5.4 and 5.5).

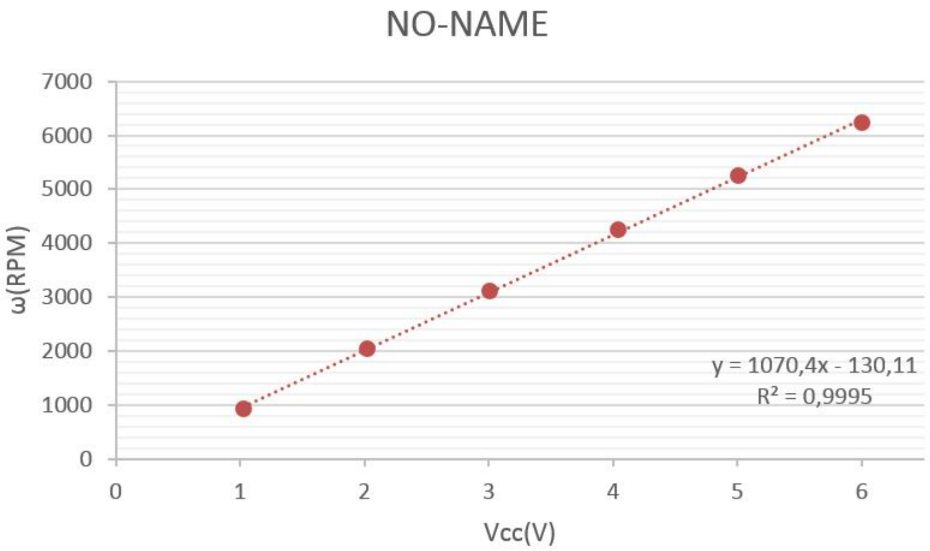


Figure 5.4 – *VCC-RPM results for the motor without model*

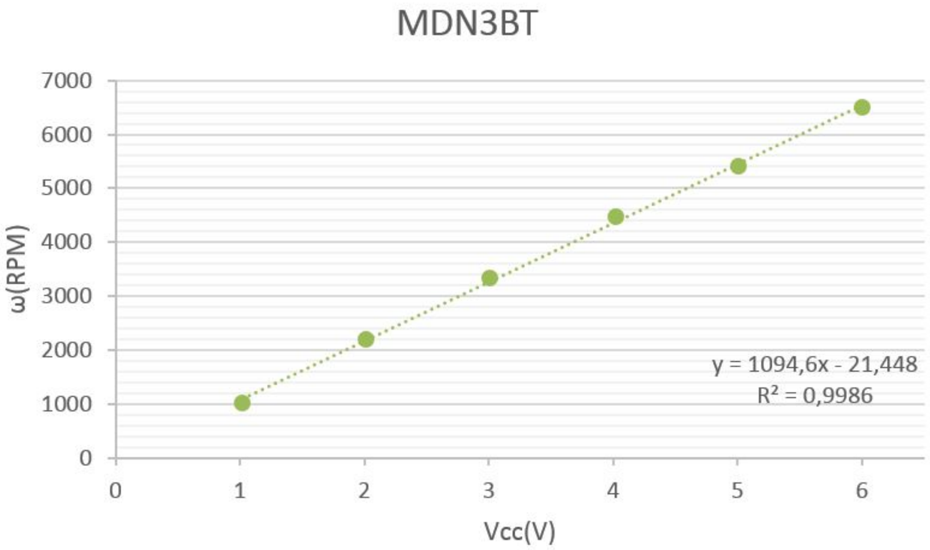


Figure 5.5 – *VCC-RPM results for Minebea MDN3BT*

Now, it is time to compare the three motors graphs. See figure 5.6 to see the comparation.

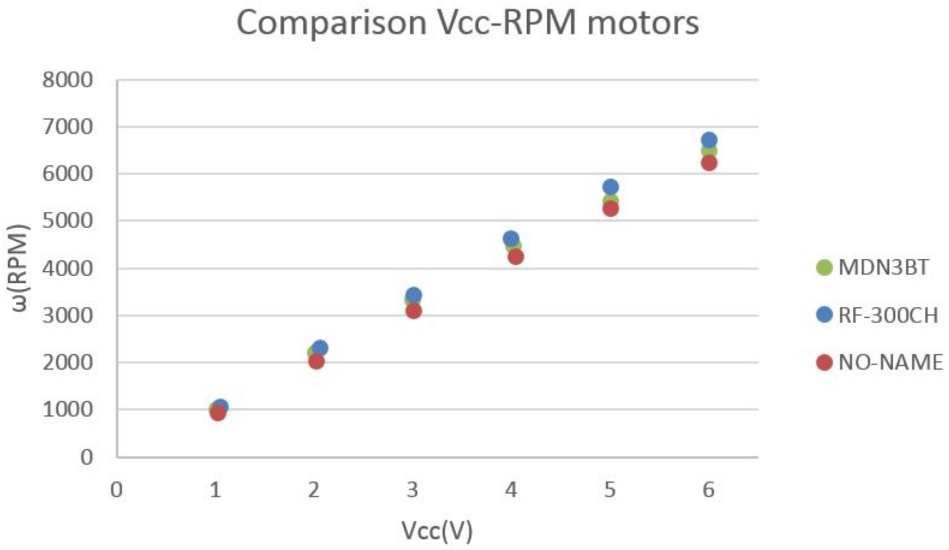


Figure 5.6 – *VCC-RPM results comparison*

Moreover, the tables of the data measured of that motors is in the Appendix section D.1, and the other motors (which have no been chosen for the project) measured in order to choose the best one (see section 3.3).

The second part of this test was measure the same parameters, but in this case, with the wheels mounted on the motor, and finally, see the comparison between the results with and without the wheel, a really useful result to see the behavior of that wheels. The figures 5.7, 5.8 and 5.9 show that results.

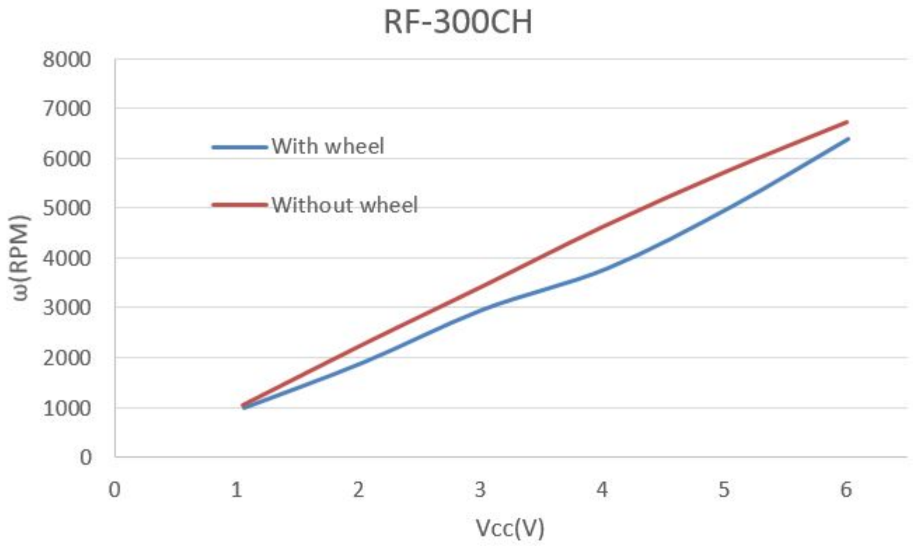


Figure 5.7 – *VCC-RPM results with wheels for Kysan RF-300CH*

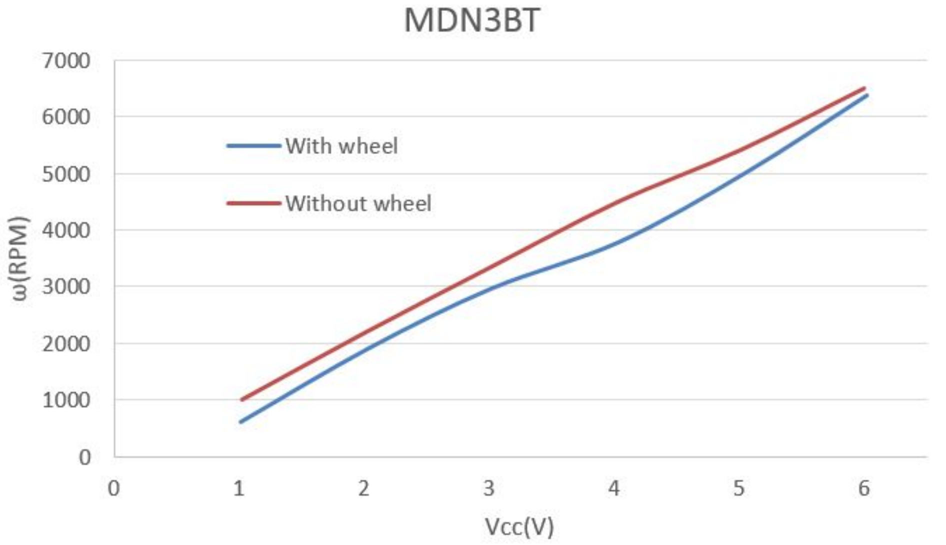


Figure 5.8 – *VCC-RPM results with wheels for Minebea MDN3BT*

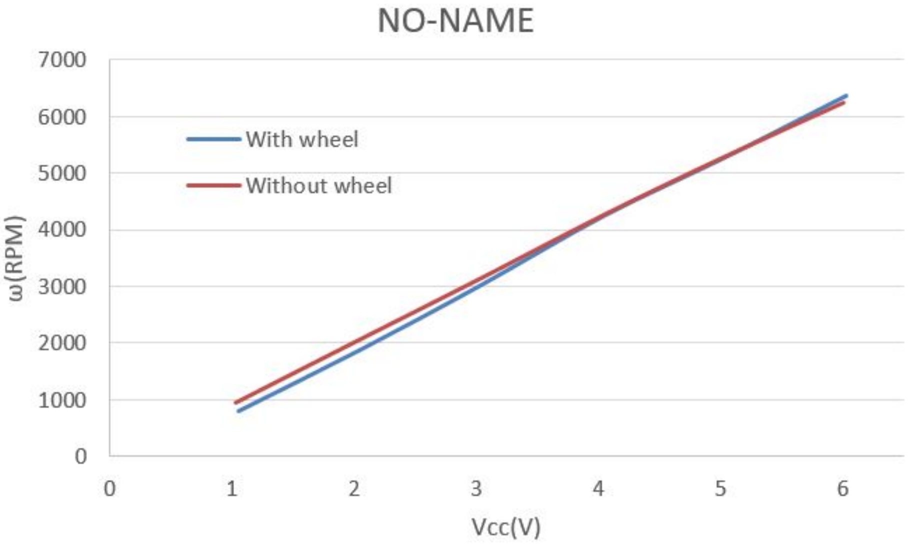


Figure 5.9 – *VCC-RPM results with wheels for the motor without model*

That results will be used for the design of the software which controls the actuator drivers, to enhance the efficiency of the algorithm.

5.1.2 Electrical parameters measurements

In order to simulate the electrical model of the DC motors, some parameters are needed. That simulation will give us an idea of the future response of these motors.The collection of that values is made experimentally, some of then with different methods, with the intention to have a better result comparing the measured values.

First, the DC motor electrical model is represented in figure 5.10.

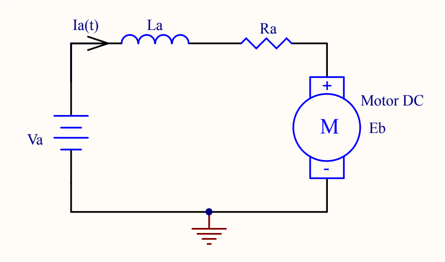


Figure 5.10 – *Model circuit for DC motor*

The parameters that appear in that model means:

- Eb: Counter-electromotive force (counter EMF) of the DC motor. (V)
- Ra: Armature resistance ()
- La: Armature Inductor (H)
- Va: Input voltage (V)
- Ia: Armature current (A)

Now, if we apply the Kirchhoff's circuit laws, we have the equation 5.1.1.

$$Va = Eb + (Ia \times Ra) \tag{5.1.1}$$

Moreover, the transient equation for a DC motor is the equation 5.1.2.

$$Va = La\frac{dI}{dt} + RaIa + K_T\dot{\theta} \tag{5.1.2}$$

Where L/R will be the electric time constant (te), which will be calculated too.

Armature resistance (Ra)

To measure the armature resistance we will use three methods:

- Using a multimeter, measuring in the both terminals of the motor armature.
- With a power supply source, and two multimeters to have a more accurate measures for current and voltage, adjust to the minimum voltage for the supply of the motor, and measure the current consumption. Using the Ohm law, the resistance of that values will be the armature resistance.

- With a LCR meter, measuring in both terminals of the motor armature.

The LCR meter is the model 4192A Impedance Analyzer HP, which gave us the La and Ra values choosing the options shown in figure 5.11. Figure 5.12 shows the LCR meter and the motor been measured.

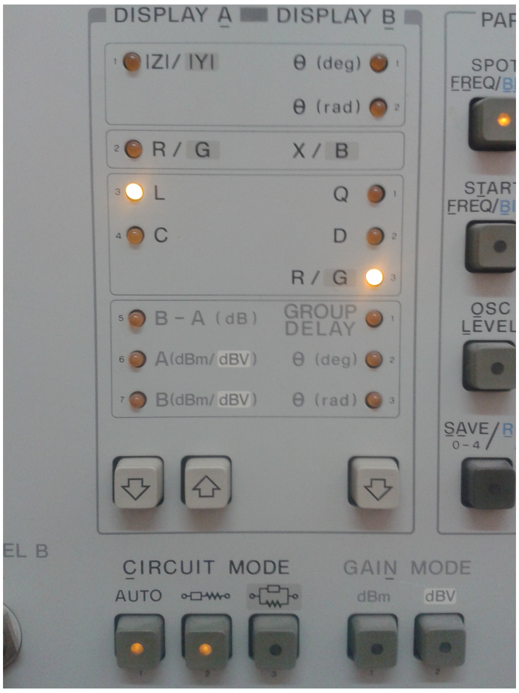


Figure 5.11 – *LCR options to measure Ra and La*

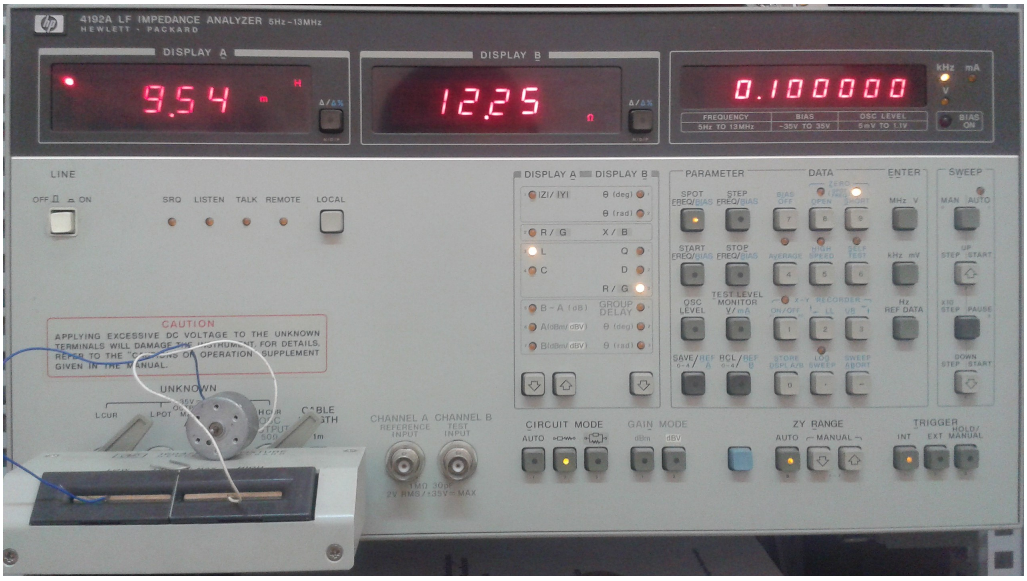


Figure 5.12 – *Measuring MDN3BT resistance and inductance*

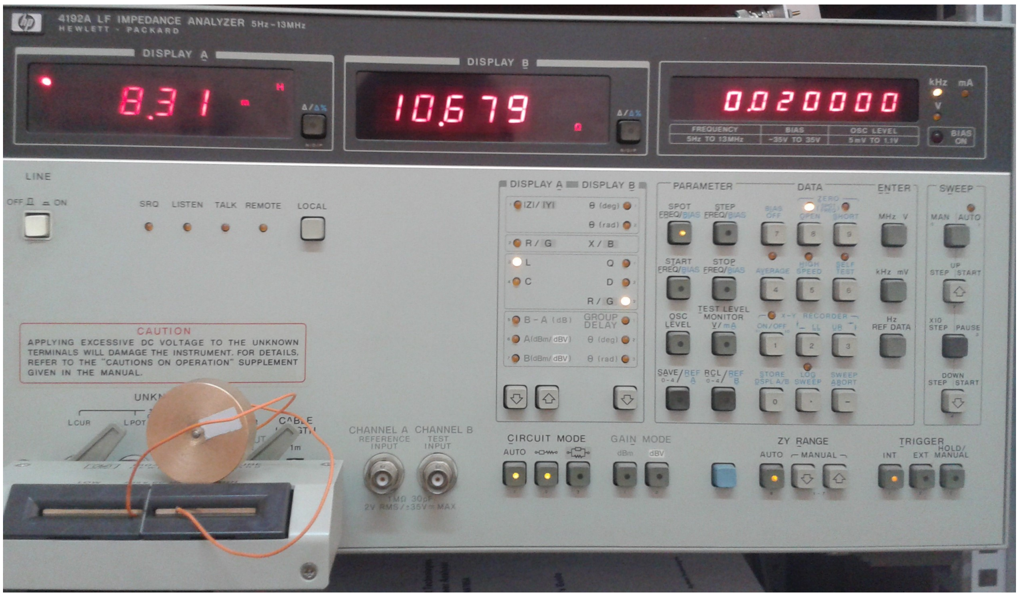

4192A LF IMPEDANCE ANALYZER 5Hz~13MHz
HEWLETT · PACKARD
DISPLAY A
8.31
DISPLAY B
10.679
0.020000
CAUTION
APPLYING EXCESSIVE DC VOLTAGE TO THE UNKNOWN TERMINALS WILL DAMAGE THE INSTRUMENT. FOR DETAILS, REFER TO THE "CAUTIONS ON OPERATION" SUPPLEMENT GIVEN IN THE MANUAL.

Figure 5.13 – *Measuring RF300-CH resistance and inductance*

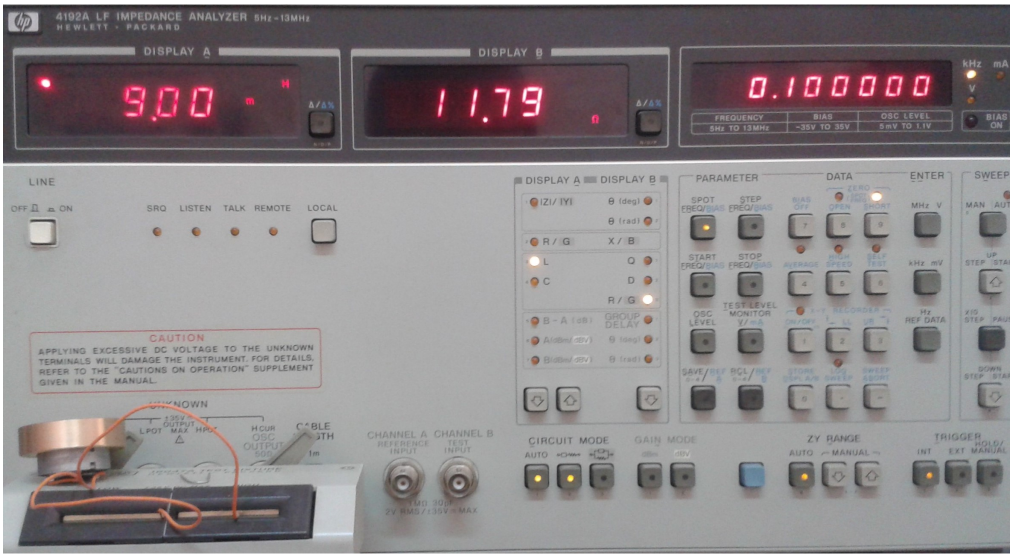

4192A LF IMPEDANCE ANALYZER 5Hz~13MHz
HEWLETT · PACKARD
DISPLAY A
9.00
DISPLAY B
11.79
0.100000
CAUTION
APPLYING EXCESSIVE DC VOLTAGE TO THE UNKNOWN TERMINALS WILL DAMAGE THE INSTRUMENT. FOR DETAILS, REFER TO THE "CAUTIONS ON OPERATION" SUPPLEMENT GIVEN IN THE MANUAL.

Figure 5.14 – *Measuring the motor without name resistance and inductance*

Motor	Multimeter	Power Supply	LCR	Ra_{ave}
MDN3BT	12.25 Ω	12.9 Ω	12.25 Ω	12.47 Ω
RF-300CH	10.92 Ω	10.9 Ω	10.679 Ω	10.83 Ω
NO-NAME	11.14 Ω	12.2 Ω	11.79 Ω	11.71 Ω

Table 5.2 – *Armature resistance measurements*

For the power supply measurements, the values taken for the calculation are shown in table 5.3.

Motor	Istart(mA)	Vstart(V)
MDN3BT	50.2	0.615
RF-300CH	64.1	0.7
NO-NAME	55.2	0.615

Table 5.3 – *Start conditions for each motor*

The start current is the current necessary to start the rotation of the axis for the motor. That value will be maximum just before the friction torque was passed. So the Istart is the minimum current necessary to achieve that situation. The friction torque will be calculated in this section too.

Armature inductance (La)

To measure the inductance of the motor, the LCR used for the resistor will be used (see figures 5.12,5.13 and 5.14). Ant the table 5.4 resume that values.

Motor	**LCR**
MDN3BT	9.54 mH
RF-300CH	8.31 mH
NO-NAME	9.00 mH

Table 5.4 – *Armature inductance measurements*

Electrical time constant (te)

With the previous values for Ra and La, the electrical time constant can be calculated with the equation 5.1.3 and shown in table 5.5.

$$te = \frac{La}{Ra} \tag{5.1.3}$$

Motor	**te** (μs)
MDN3BT	765.2
RF-300CH	767.1
NO-NAME	768.5

Table 5.5 – *Electrical time constant calculation*

Counter-electromotive force (Eb)

The circuit shown in figure 5.10, is the electrical model for the DC motor, where the Eb was represented. Now it is time to calculate that parameter, because we have the values for the Ra and Ia. Following the equation 5.1.1, the following tables have been calculated.

Va(V)	**Ia(mA)**	**Eb(V)**
1.019	17.66	0.7988
2.014	20.42	1.7594
3.004	25.36	2.6878
4.02	34.12	3.5946
5.01	40.5	4.5050

Table 5.6 – *Counter-electromotive force calculation for MDN3BT*

Va(V)	**Ia(mA)**	**Eb(V)**
1.054	15.24	0.8889
2.066	19.12	1.8588
3.012	24.69	2.7445
4	32.00	3.6533
5.01	40.29	4.5734
6	50.4	5.4540

Table 5.7 – *Counter-electromotive force calculation for RF-300CH*

Va(V)	**Ia(mA)**	**Eb(V)**
1.028	12.69	0.8793
2.025	15.49	1.8436
3.007	18.97	2.7848
4.04	24.04	3.7584
5.01	29.95	4.6592
6	38.14	5.5533

Table 5.8 – *Counter-electromotive force calculation for motor without name*

For each calculation, the Ra used was the Ra_{ave} that appears in table 5.2.

Counter-electromotive constant (Ke)

When a DC motor is working, an induced voltage appears proportionally to the product of the flow with the angular speed. Our flow is constant, so the Eb is directly proportional to the angular speed (see equation 5.1.4). The Ke parameter will define the electrical characteristics of the motor.

$$Ke = \frac{Eb}{N}(V/RPM) = \frac{V}{\omega}(V/rad/s) \tag{5.1.4}$$

The Ke value was obtained for each value of Eb and N (or V and ω), and the average value is represented in the table 5.9.

Motor	**Ke** $\left(\frac{V}{RPM}\right)$	**Ke** $\left(\frac{V}{rad/s}\right)$
MDN3BT	0.0008066	0.008713489
RF-300CH	0.000808414	0.008422059
NO-NAME	0.000900611	0.009226577

Table 5.9 – *Counter-electromotive constant calculation*

Torque constant(Kt)

The energy that the motor supplies in its rotation axis is expressed with the equations 5.1.5, 5.1.6 and 5.1.7, depending on Ke.

$$Kt\left(\frac{Nm}{A}\right) = Ke\left(\frac{V}{rad/s}\right) \tag{5.1.5}$$

$$Kt\left(\frac{Nm}{A}\right) = 9.5493 \times 10Ke\left(\frac{V}{kRPM}\right) \tag{5.1.6}$$

$$Kt\left(\frac{oz-in}{A}\right) = 1.3524Ke\left(\frac{V}{kRPM}\right) \tag{5.1.7}$$

Using the equation 5.1.5, the torque constant can be determined thanks to the Ke values calculates before (see table 5.9), which are represented in table 5.10.

Motor	**Ke** $\left(\frac{V}{rad/s}\right)$	**Kt** $\left(\frac{Nm}{A}\right)$
MDN3BT	0.008871223	0.008871223
RF-300CH	0.008605915	0.008605915
NO-NAME	0.009434063	0.009434063

Table 5.10 – *Torque constant calculation*

Mechanical time constant(tm)

That parameter will be used to calculate the Inertia momentum (Jm). When a step function is applied to the motor, it will be generated a transient response. The mechanical time constant is the necessary time to achieve the 63.2% of the final value (see figure 5.15).

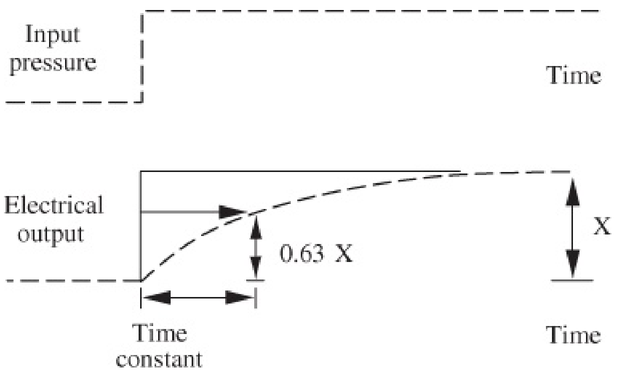


Figure 5.15 – *Mechanical time constant representation [10]*

To measure that parameter the graphs of the section 5.1.3 have been used, only a section of that graphs, obtaining the results shown in table 5.11.

Motor	**tm (s)**
MDN3BT	0.1777
RF-300CH	0.1723
NO-NAME	0.1166

Table 5.11 – *Mechanical time constant measurement*

The part of the graphs used are shown in figures 5.16, 5.17 and 5.18, using the linearization equation shown inside each one.

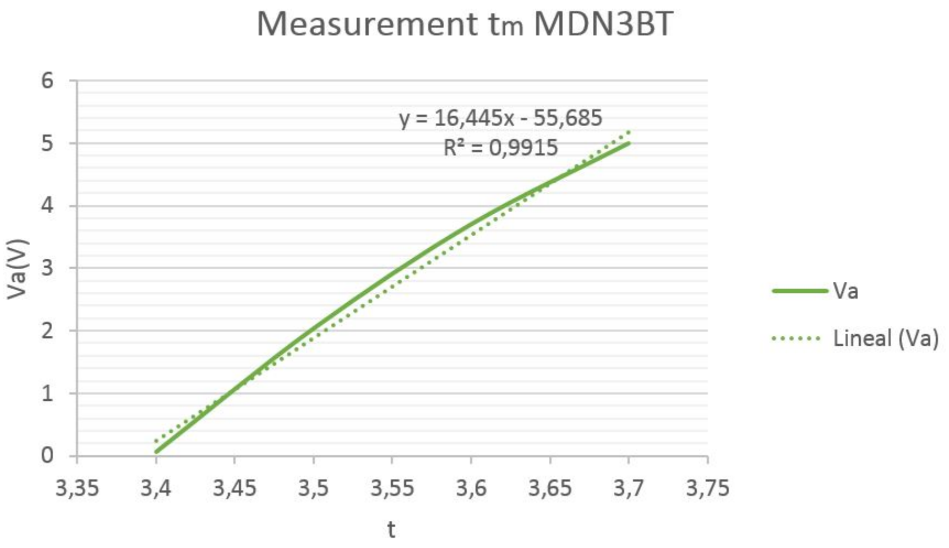


Figure 5.16 – *Mechanical time constant graph for MDN3BT*

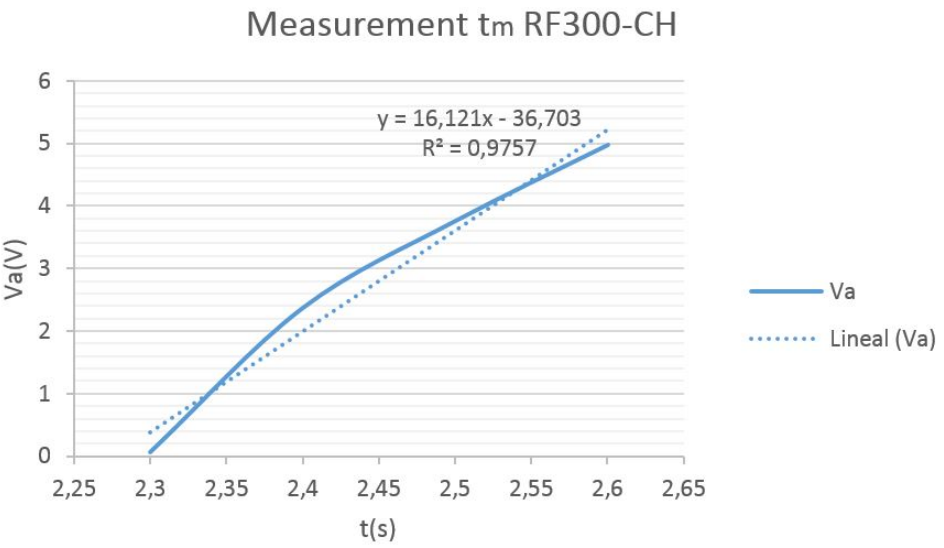


Figure 5.17 – *Mechanical time constant graph for RF300-CH*

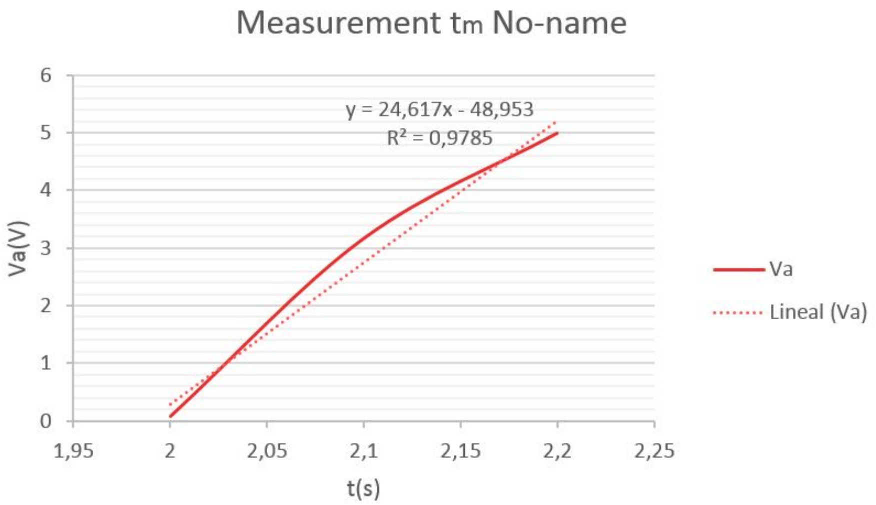


Figure 5.18 – *Mechanical time constant graph for no-name motor*

Inertia momentum (Jm)

It is used a parametric method, because that calculation depends on others parameters calculated or measured before with the following expressions 5.1.8 and 5.1.9 . The figure 2.11 shows an example where the Jm can be seen as L in the figure.

$$tm(s) = \frac{Jm \times Ra}{Kt \times Ke} \tag{5.1.8}$$

$$Jm = \frac{Kt \times Ke \times tm}{Ra} (Kg * m^2) \tag{5.1.9}$$

And finally, table 5.12, which shows the value of Jm for each motor.

Motor	**Jm** $(Kg * m^2)$
MDN3BT	$1.122 * 10^{-6}$
RF-300CH	$1.1776 * 10^{-6}$
NO-NAME	$8.6386 * 10^{-6}$

Table 5.12 – *Inertia momentum calculation*

Friction torque(Tf)

That parameter is calculated with the measurements taken of the start current, in table 5.3, and the Kt value, from table 5.10, following the equation 5.1.10.

$$Tf = Kt \times Istart \tag{5.1.10}$$

Motor	**Tf** (μNm)
MDN3BT	445.33
RF-300CH	551.63
NO-NAME	507.57

Table 5.13 – *Friction torque calculation*

Damping Ratio(B)

The constant B determines when the system is stabilized, with a constant speed, where the acceleration $\dot{\omega}$ is the derived of that speed, in this case, zero:

$$Tm = Kt \times Ia = Jm\dot{\omega} + B\omega + Tf = B\omega + Tf \tag{5.1.11}$$

The table 5.14 have the necessary parameters to calculate the damping ratio.

VCC(V)	I(mA)	ω (RPM)	ω (rad/s)	Tm=Kt*Ia (Nm)	Tm-Tf=B*w (Nm)
1.019	17.66	1007.33	105.4678	0.00015667	0.0001162
2.014	20.42	2200.333	230.3749	0,00018115	0.000140659
3.004	25.36	3336.333	349.3141	0,000224974	0.00018448
4.04	34.12	4484.33	469.5097	0,000302686	0.000262195
5.01	40.5	5407.33	566.1478	0,000359285	0.000318793
6	47.4	6495.667	680.0963	0,000420496	0.000380005

Table 5.14 – *Parameters necessary for the B calculation (MDN3BT)*

VCC(V)	I(mA)	ω (RPM)	ω (rad/s)	Tm=Kt*Ia (Nm)	Tm-Tf=B*w (Nm)
1.028	12.69	937.466	98.1527	0.000119718	0.0000676422
2.025	15.49	2032.33	212.7853	0.000146134	0.0000940576
3.007	18.97	3103.33	324.919	0.000178964	0.000126888
4.04	24.04	4251.66	445.1495	0.000226795	0.000174719
5.01	29.95	5259.66	550.6871	0.00028255	0.000230474
6	38.14	6231	652.3857	0.000359815	0.000307739

Table 5.15 – *Parameters necessary for the B calculation (no name motor)*

VCC(V)	I(mA)	ω (RPM)	ω (rad/s)	Tm=Kt*Ia (Nm)	Tm-Tf=B*w (Nm)
1.019	17.66	1007.33	105.4678	0.00015667	0.0001162
2.014	20.42	2200.333	230.3749	0,00018115	0.000140659
3.004	25.36	3336.333	349.3141	0,000224974	0.00018448
4.04	34.12	4484.33	469.5097	0,000302686	0.000262195
5.01	40.5	5407.33	566.1478	0,000359285	0.000318793
6	47.4	6495.667	680.0963	0,000420496	0.000380005

Table 5.16 – *Parameters necessary for the B calculation (RF300)*

Now, to calculate the damping ratio, the lineal regression will give us the lineal equation, and the slope of that graph will be the B parameter.

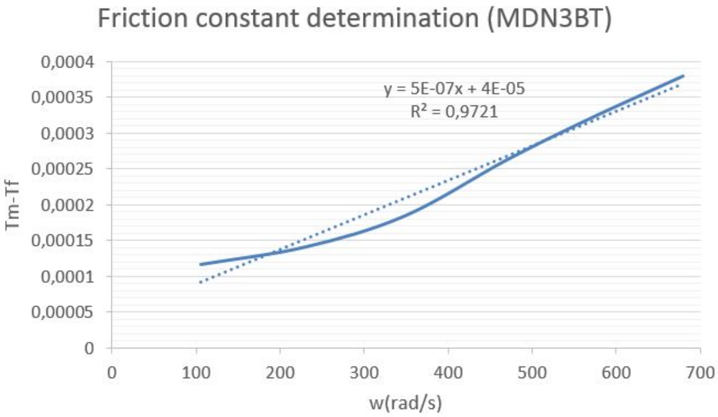


Figure 5.19 – *Graph for the regression determination of damping ratio (MDN3BT)*

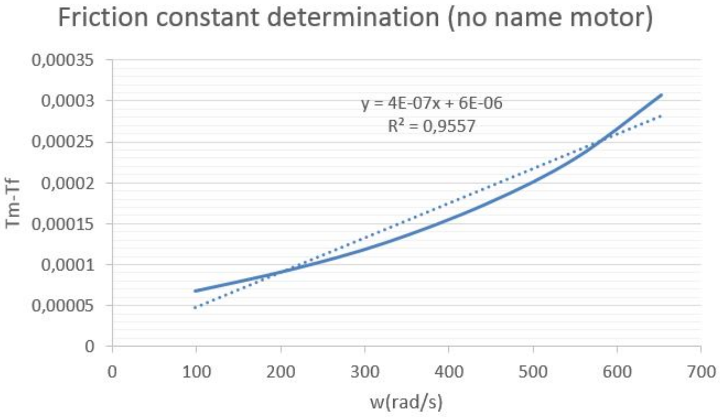


Figure 5.20 – *Graph for the regression determination of damping ratio (no name motor)*

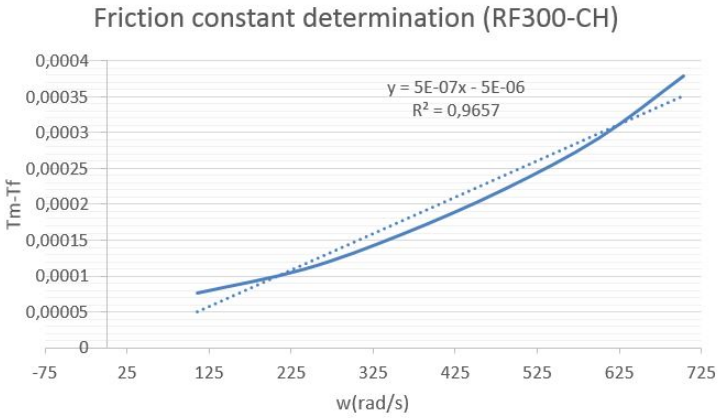


Figure 5.21 – *Graph for the regression determination of damping ratio (RF300-CH)*

Therefore, as the equation of the graph shows that the regression slope is $5 * 10^{-7}$:

$$B = 5 * 10^{-7} Nms \tag{5.1.12}$$

Summary of every parameter for the motors

Now, to conclude that experimental characterization of the motors, the table 5.17 represents a summary of the obtained values:

Motor	Ra(Ω)	La(mH)	te(μs)	Ke$\left(\frac{V}{rad/s}\right)$ = Kt$\left(\frac{Nm}{A}\right)$	tm(s)	Jm($Kg * m^2$)
MDN3BT	12.47	9.54	765.2	0.008713489	0.1777	$100.219 * 10^{-09}$
RF-300CH	10.83	8.31	767.1	0.008422059	0.1723	$82.748 * 10^{-09}$
NO-NAME	11.71	9.00	768.5	0.009226577	0.1166	$108.259 * 10^{-09}$

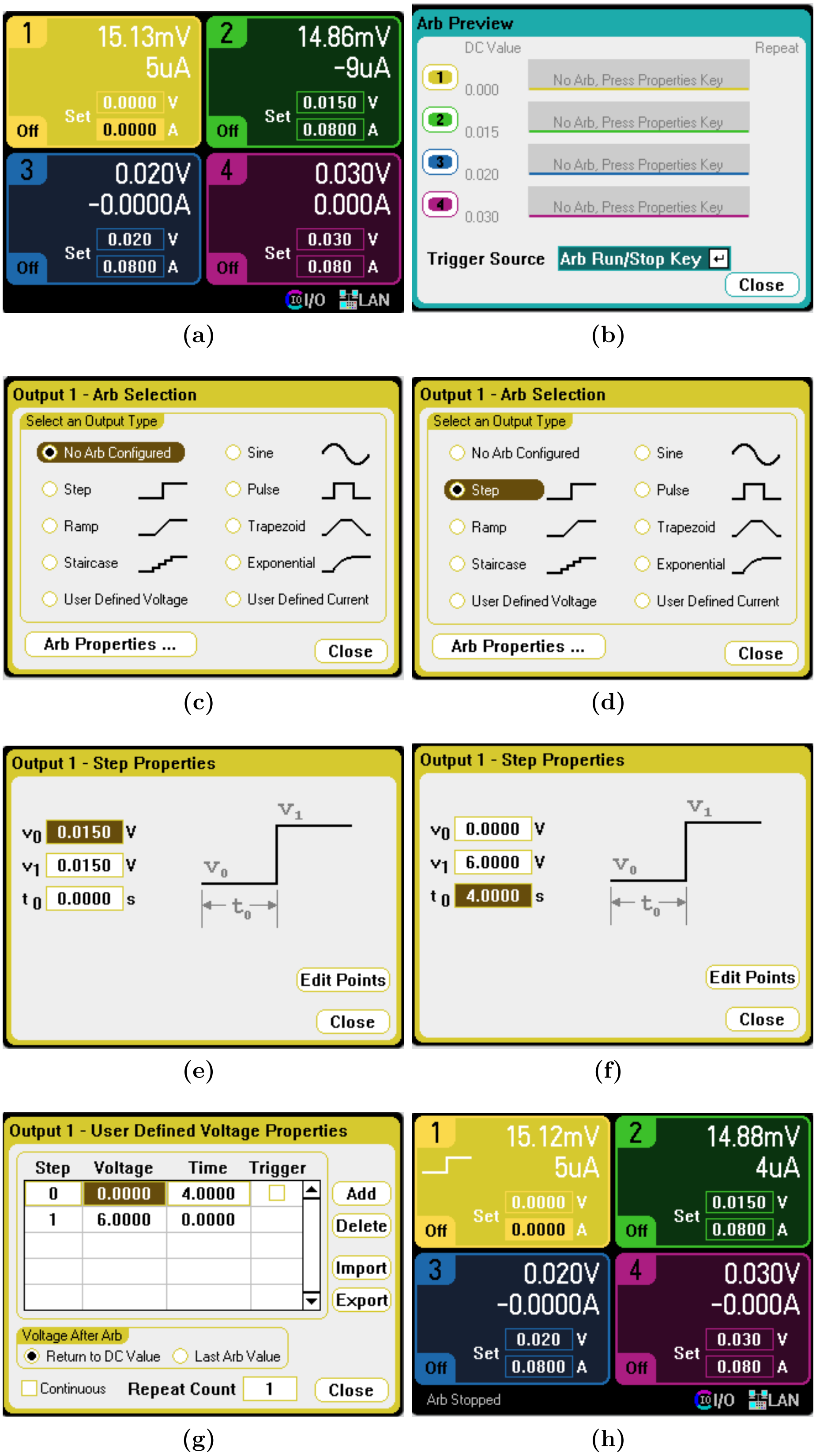
Table 5.17 – *Summary motors parameters*

5.1.3 Motors step responses

In order to know the future responses for the motors and estimate its final current consumption, the motors step response has been measured with the power supply Keysight N6705A (see figure 5.22).

Figure 5.22 – *Key-sight N6705A Source [54]*

The motor response is shown in figures 5.23 and measured with the data-logger of the N6705A. Moreover, the step functions have been pre-configured with the Arbitrary functions option that N6705A has. The figure 5.23 show how that outputs have been configured. Once the N6705A is configured, the data-logger started and the motors prepared, the measurements started and some captures of the results were taken (see figures 5.24a, 5.24b and 5.24c). Furthermore, that results have been data-logged and represented in Excel graphs, in order to see a more accurate representation in figures 5.25a, 5.25a and 5.25a.



(a)

1 15.13mV 5uA Off Set 0.0000 V 0.0000 A

2 14.86mV -9uA Off Set 0.0150 V 0.0800 A

3 0.020V -0.0000A Off Set 0.020 V 0.0800 A

4 0.030V 0.000A Off Set 0.030 V 0.080 A

(b)

Arb Preview

	DC Value	Repeat
1	0.000	No Arb, Press Properties Key
2	0.015	No Arb, Press Properties Key
3	0.020	No Arb, Press Properties Key
4	0.030	No Arb, Press Properties Key

Trigger Source: Arb Run/Stop Key

(c)

Output 1 - Arb Selection

Select an Output Type: ● No Arb Configured ○ Step ○ Ramp ○ Staircase ○ User Defined Voltage ○ Sine ○ Pulse ○ Trapezoid ○ Exponential ○ User Defined Current

(d)

Output 1 - Arb Selection

Select an Output Type: ○ No Arb Configured ● Step ○ Ramp ○ Staircase ○ User Defined Voltage ○ Sine ○ Pulse ○ Trapezoid ○ Exponential ○ User Defined Current

(e)

Output 1 - Step Properties

v_0 0.0150 V

v_1 0.0150 V

t_0 0.0000 s

(f)

Output 1 - Step Properties

v_0 0.0000 V

v_1 6.0000 V

t_0 4.0000 s

(g)

Output 1 - User Defined Voltage Properties

Step	Voltage	Time	Trigger
0	0.0000	4.0000	☐
1	6.0000	0.0000	

Voltage After Arb: ● Return to DC Value ○ Last Arb Value

☐ Continuous Repeat Count 1

(h)

1 15.12mV 5uA Off Set 0.0000 V 0.0000 A

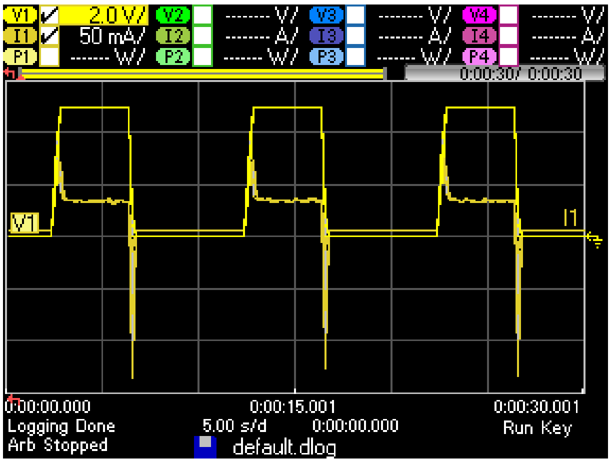
2 14.88mV 4uA Off Set 0.0150 V 0.0800 A

3 0.020V -0.0000A Off Set 0.020 V 0.0800 A

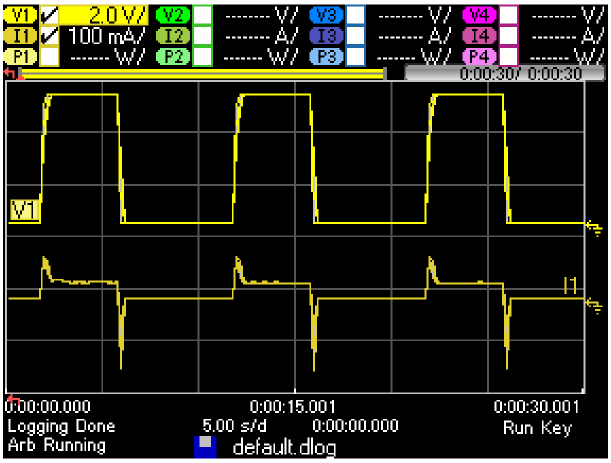
4 0.030V -0.000A Off Set 0.030 V 0.080 A

Arb Stopped

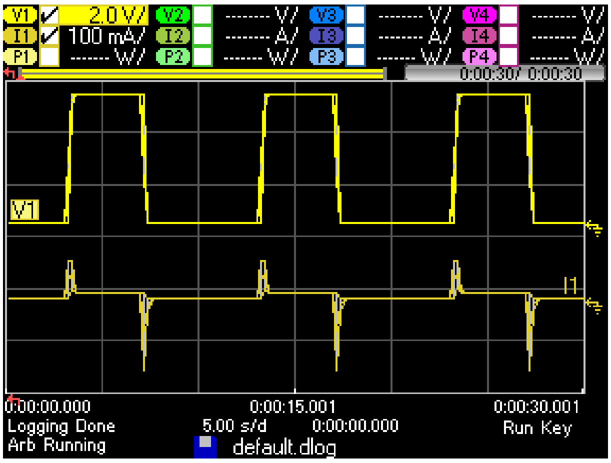
Figure 5.23 – *Output configuration of the N6705A to supply the motors*



(a) *Motor MDN3BT*

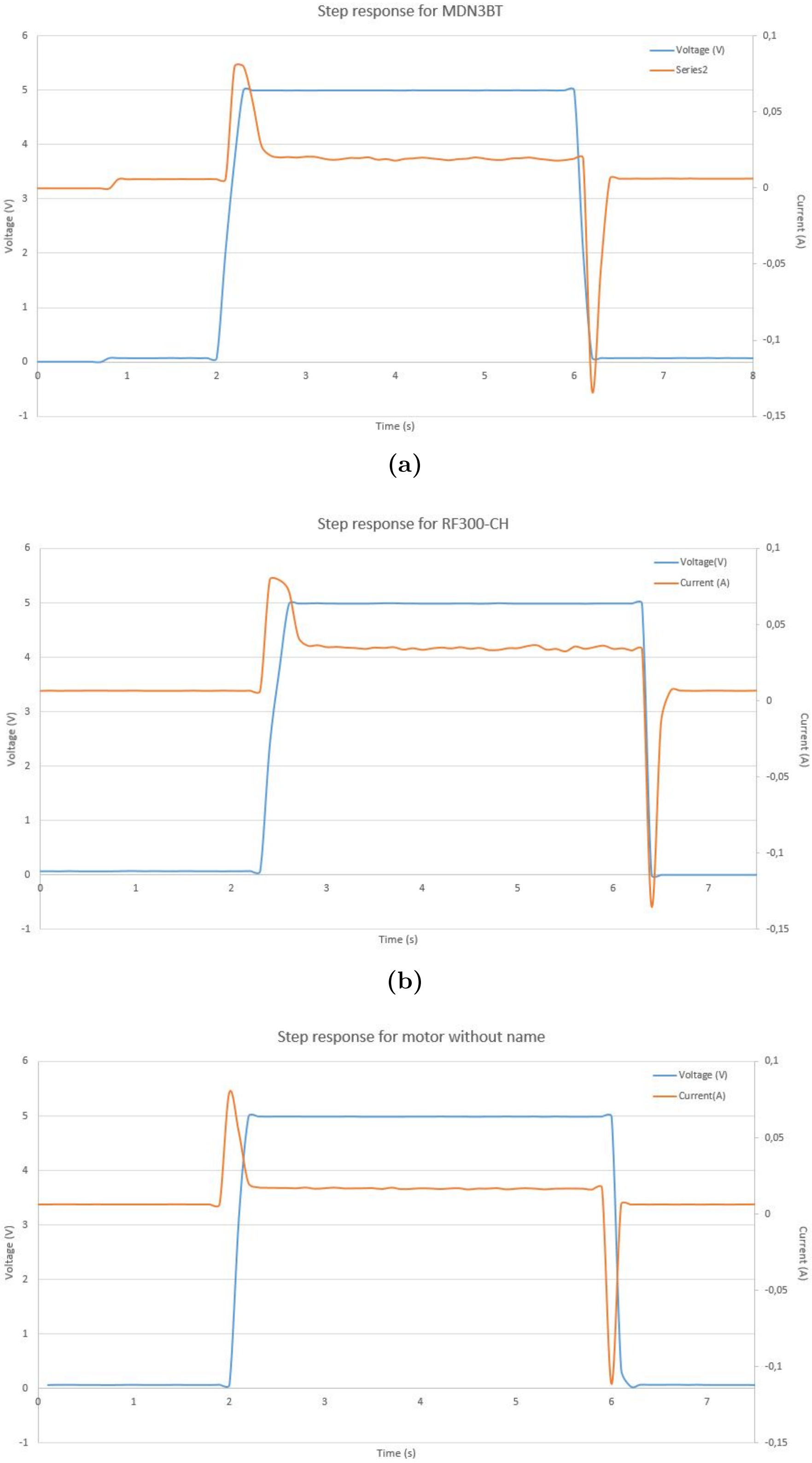


(b) *Motor RF300-CH*



(c) *Motor without name*

Figure 5.24 – *Step response captures from N6705A*



(a)

(b)

(c)

Figure 5.25 – *Step response graphs from the data-logger of N6705A*

5.2 Motors simulation block

For the simulation of the motor, MATLAB Simulink tool has been used. The figure 5.26 shows the whole simulation, with the signal generator, the motor simulation block, and the scope tool to print the graphs. Inside the motor simulation block, it is the diagram which simulate the DC motors that we are using. That diagram is shown in figure 5.27.

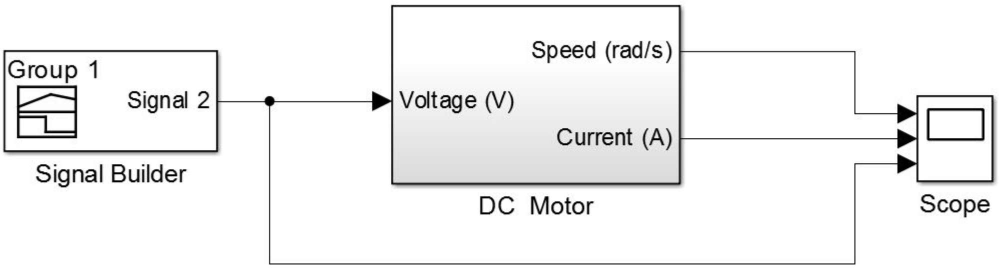


Figure 5.26 – *Simulink simulation - Main diagram*

In order to model the motor, we should study how it works:

First, the torque generated by a DC motor is proportional to the Ia (armature current) and the strength of the magnetic field. If the magnetic field is constant (Kt), the equation 5.2.1 shows that evidence.

$$T = Kt \times Ia \tag{5.2.1}$$

Now, the back EMF, is proportional to the derivative of $\omega$by a product with a constant factor Ke (Kt=Ke=K), equal to Kt:

$$Eb = Ke \times \dot{\omega} \tag{5.2.2}$$

Following the 2nd Newton's law:

$$Jm\dot{\dot{\omega}} + Eb\dot{\omega} = K \times Ia \tag{5.2.3}$$

$$La\frac{di}{dt} + Ra * Ia = V - K \times \dot{\omega} \tag{5.2.4}$$

Finally, with the Laplace transform, it can be expressed as the following form:

$$(s * La + Ra) * Ia(s) = V(s) - K * s * \dot{\omega} \tag{5.2.5}$$

$$P(s) = \frac{\dot{\omega}(s)}{V(s)} = \frac{(Js + Eb)(Las + Ra)}{K}[\frac{rad/s}{v}]. \tag{5.2.6}$$

From that equation appears the model shown in figure 5.27.

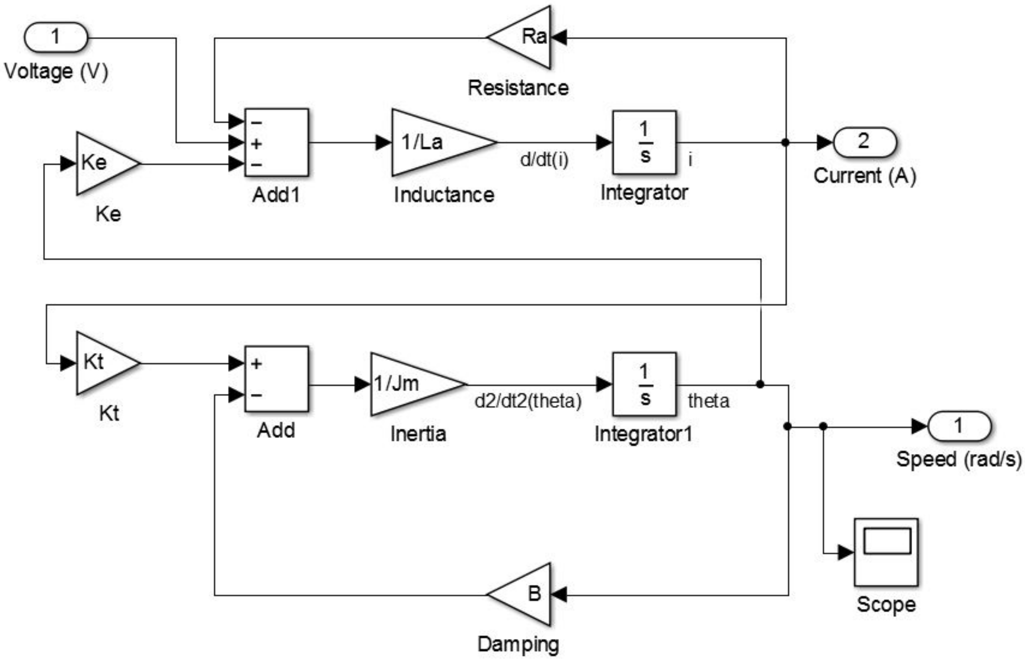


Figure 5.27 – *Simulink simulation - DC Motor block*

First, the motors output of that simulation is shown in figures 5.28, 5.29 and 5.30. The shape of each graph is very similar to the measured from the motors, but the amplitude of the current peaks is bigger. Maybe that difference is due to the measurements errors. The comparison between both graphs is shown in figures 5.31a, 5.31b and 5.31c.

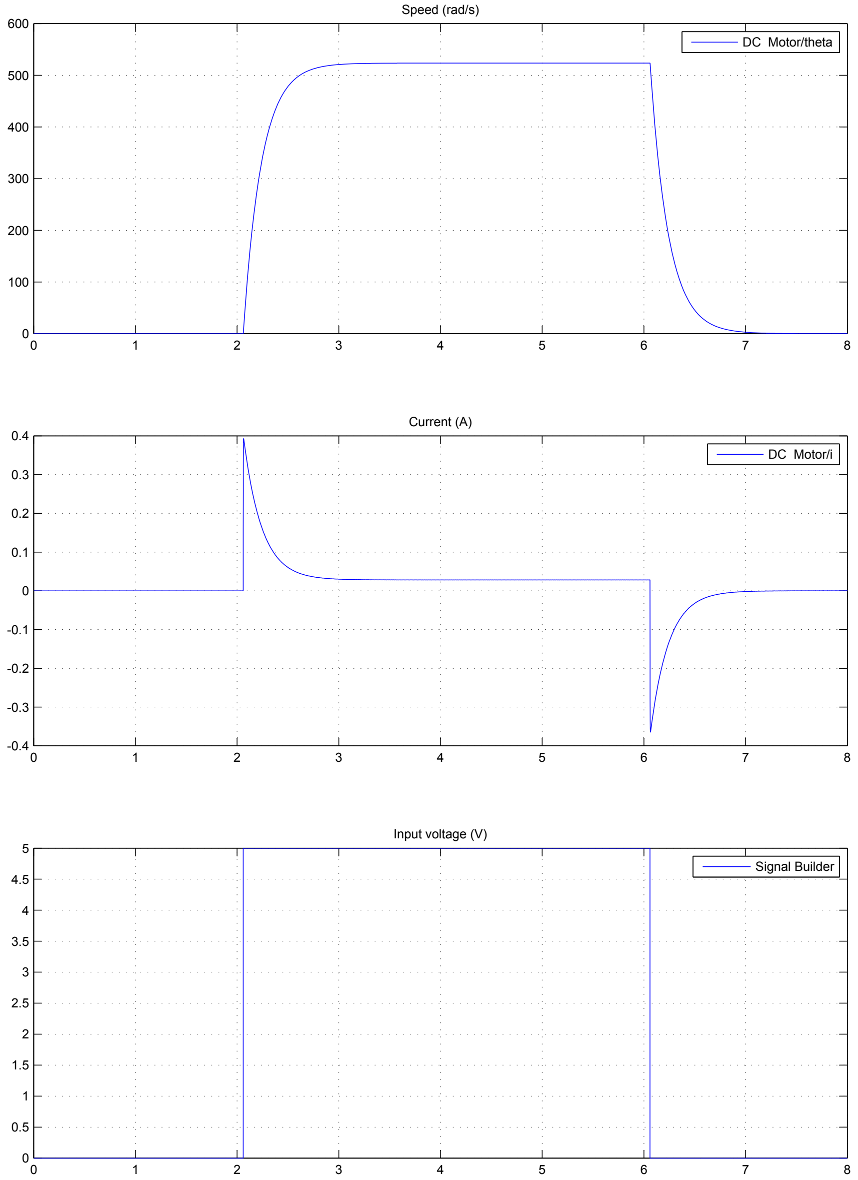


Figure 5.28 – *Simulink simulation - Output for MDN3BT*

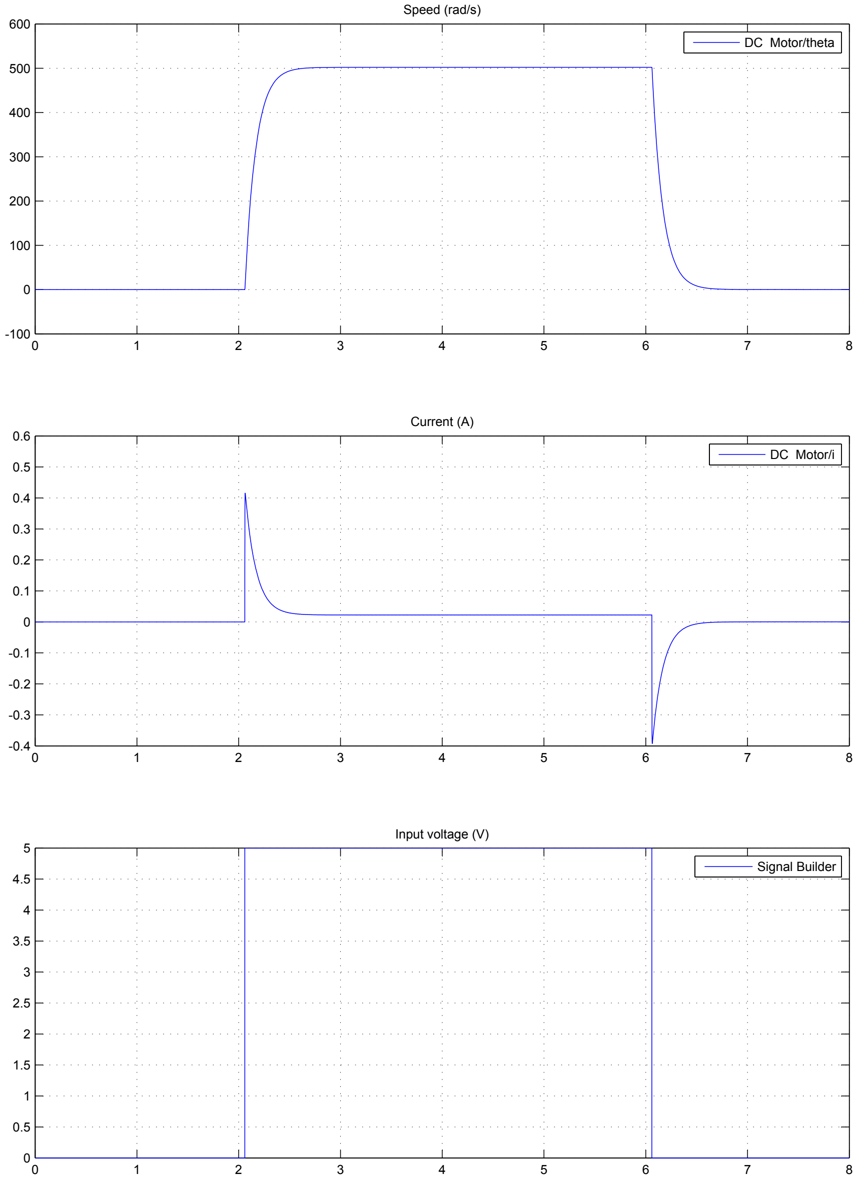


Figure 5.29 – *Simulink simulation - Output for no name motor*

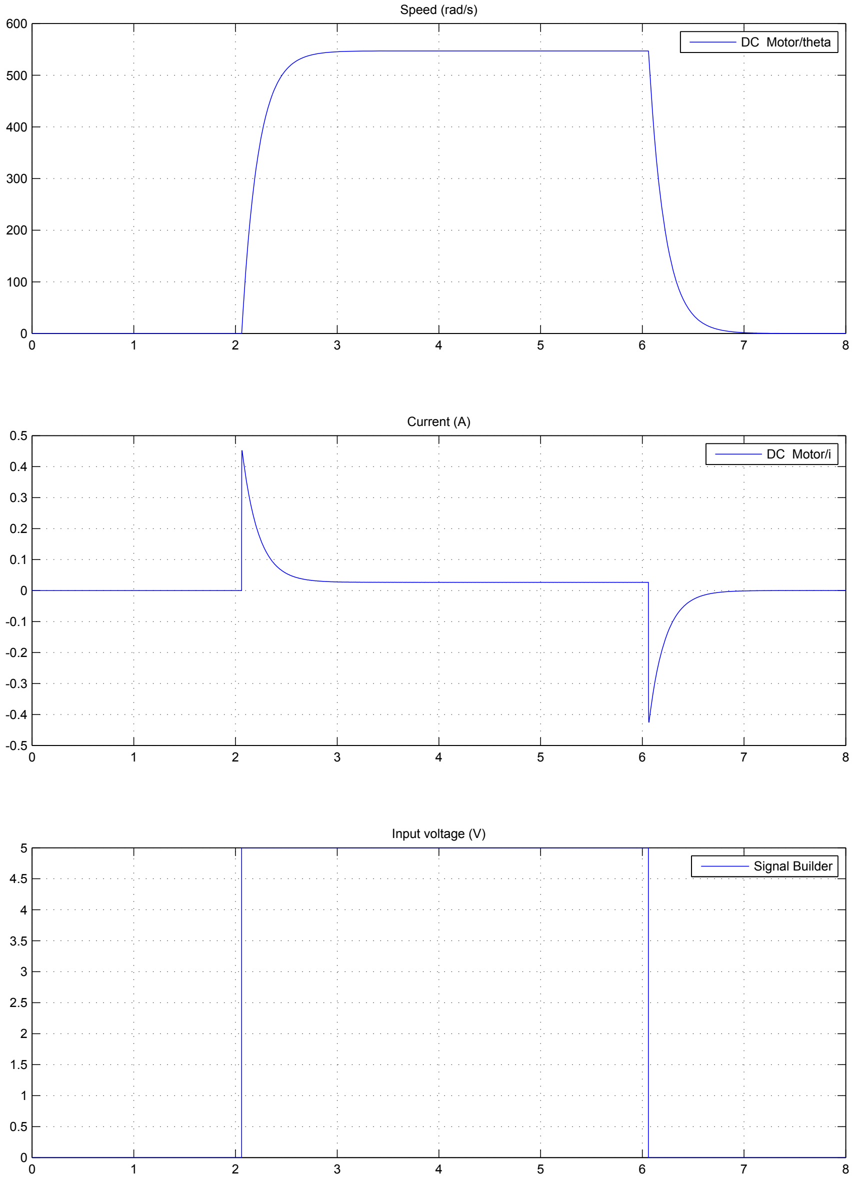
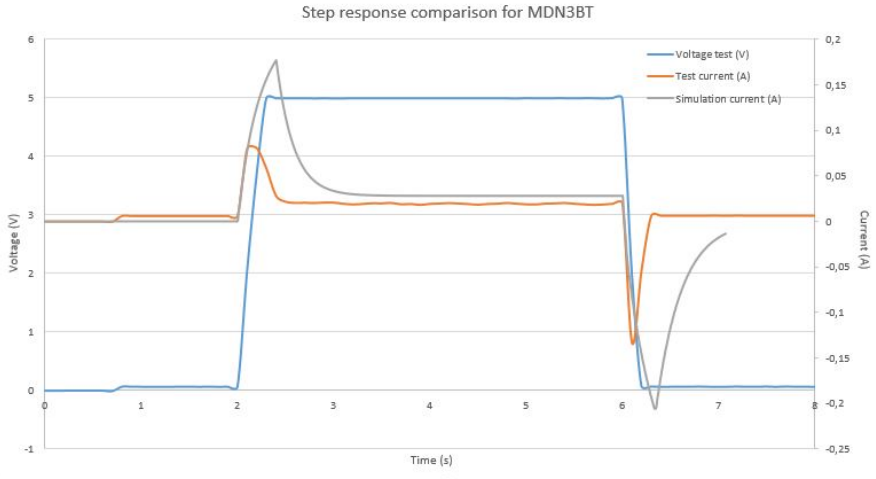
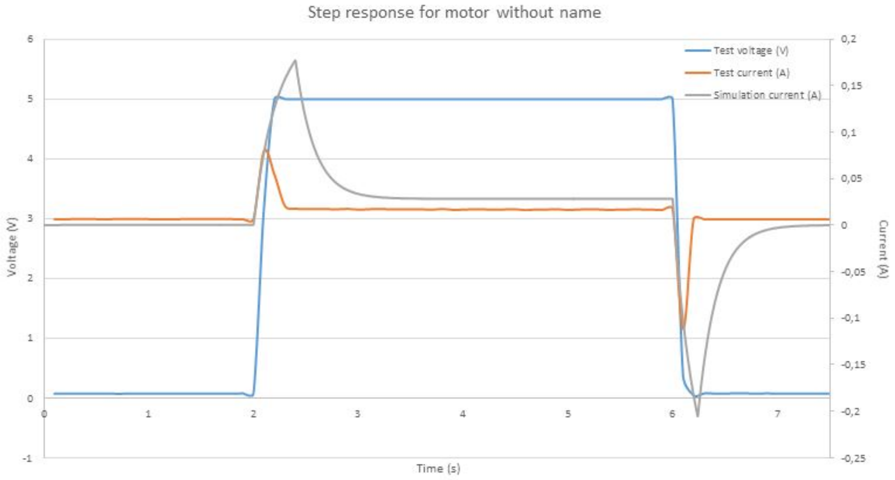


Figure 5.30 – *Simulink simulation - Output for RF300-CH*

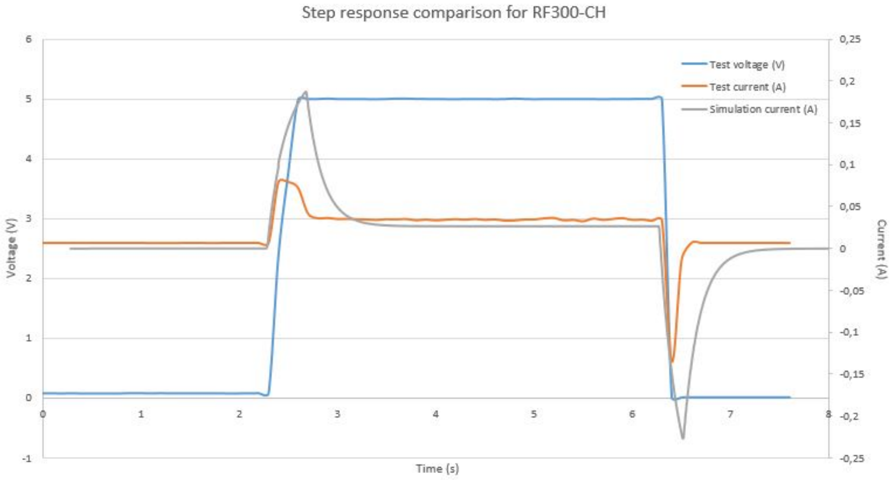
In order to know if the results of the measurements are correct, the graphs from the simulation are compared with the test results in figures 5.31a, 5.31b and 5.31c.



(a) *Simulation - Test comparison for MDN3BT*



(b) *Simulation - Test comparison for motor without name*



(c) *Simulation - Test comparison for RF300-CH*

Figure 5.31 – *Step response graphs from the data-logger of N6705A*

5.3 Final implementation

In that section, the implementation of the whole system will be described. Firstly, the electronics PCB, one of the most relevant parts of the system's implementation. Later, the mechanical implementation, which will be described with more detail in the section 5.3.2.

5.3.1 PCB implementation

The PCB was built PCB the LPFK tool, as section 4.1.2 describes. That building process is in the video of the following link: `https://www.youtube.com/watch?v=5p0AeuMRF3I`.

When the PCB fabrication was finished, a protection layer of a mix of rosin and acetone. That material will protect the PCB of the copper oxidation. The difference between the PCB with and without the mixture is shown in figure 5.32, when the brightness of the copper shows where is the protector and where it is not.

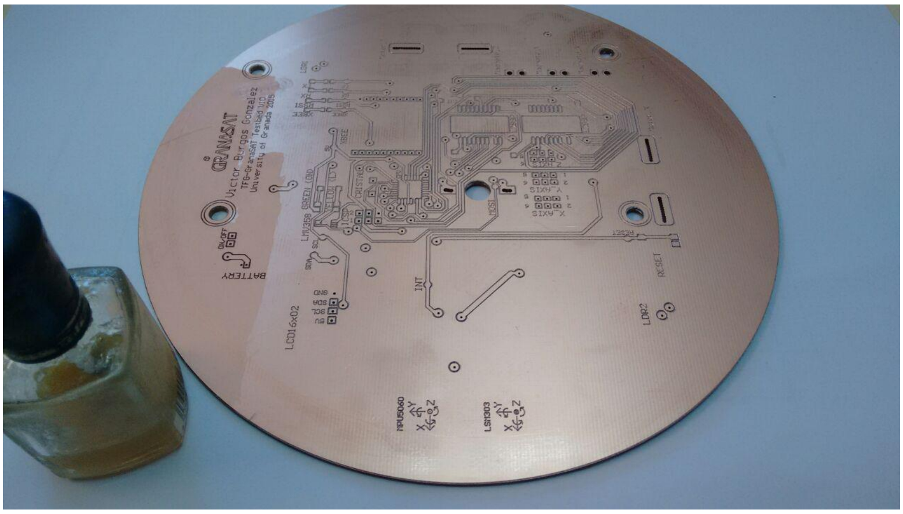


Figure 5.32 – *Mixture application on the PCB*

Once the protection was applied, it is the moment to start soldering the PCB. In figure 5.33 it is shown the first devices that were welded on the copper plate, in bottom layer.

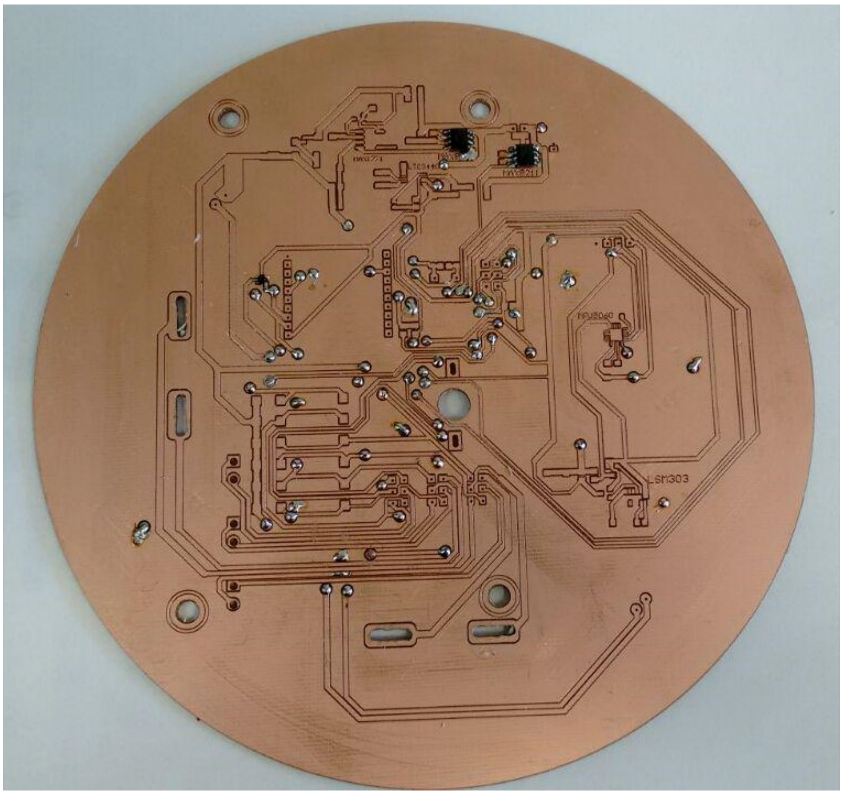
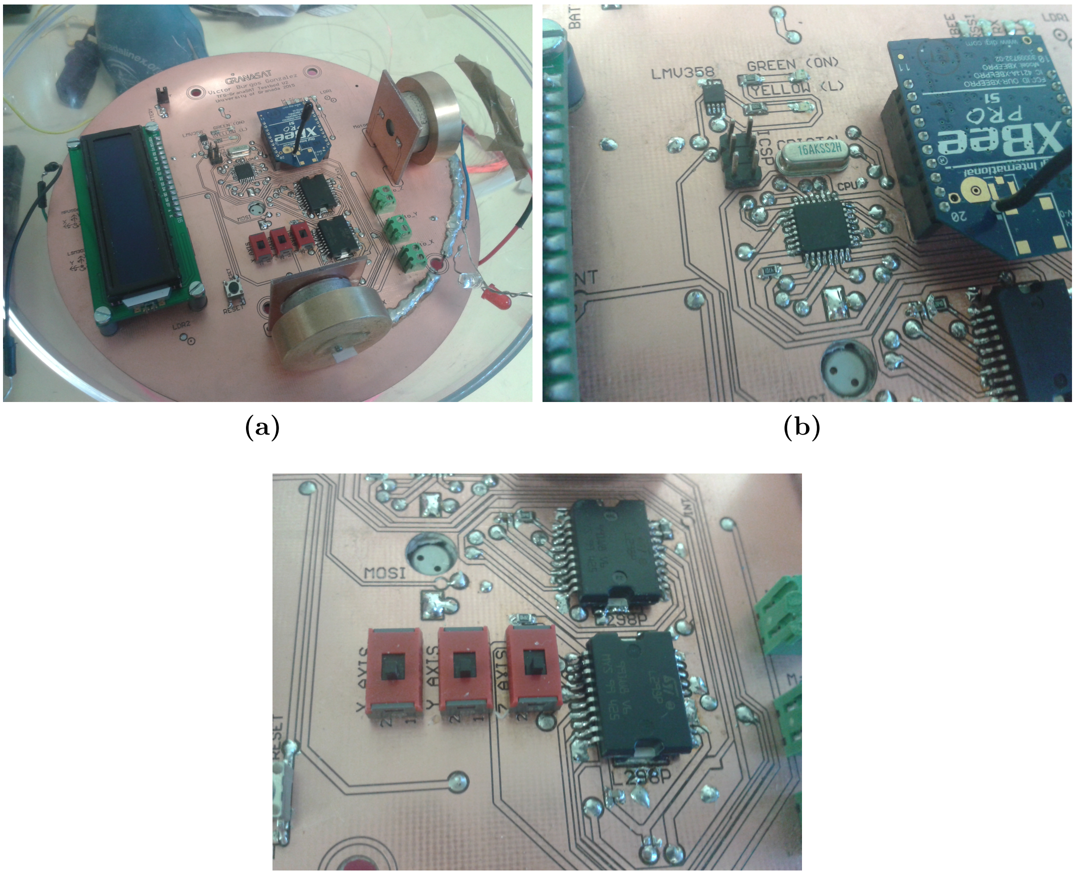


Figure 5.33 – *First soldered devices and vias*

And finally, the PCB with every component welded on its top layer, is shown in figure 5.34.


(a) (b)

(c)

Figure 5.34 – *PCB Implementation*

In figure 5.34a can be seen an error of the fabrication method, changing the layer distribution in one of the software that controls the system, and the attachment with the hemisphere, using only the friction between both surfaces.

Furthermore, the motor with the reaction wheel assembly is shown in figure 5.35.

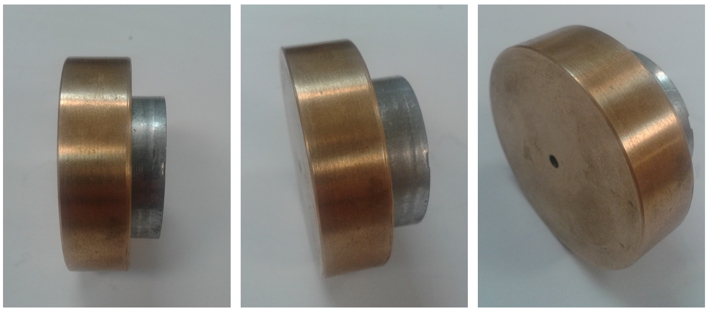

(a) (b) (c)

Figure 5.35 – *Assembly reaction wheel with motor*

5.3.2 3D Printing process

In this section, it will be take in account the printing process of the mechanical parts for the Testbed. Once we know the most important things about the printers that we are using, we are going to show how this printer works.

First at all, the printer has to be calibrated. In order to have a reference for the calibration, a paper is used in that way (see figure 5.36), and you have to note when you pull out the paper that it touch with the nozzle of the printer.

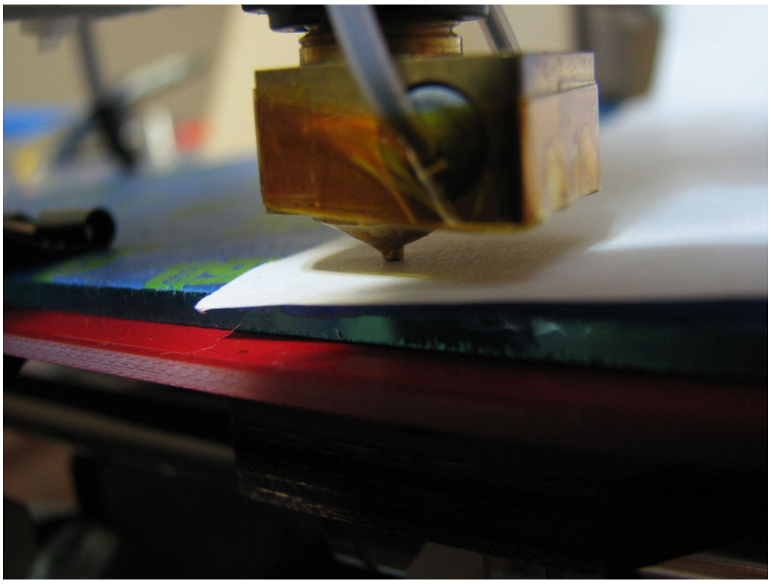


Figure 5.36 – *Calibration of the printer*

Once the heatbed is calibrated in relation to the nozzle (or to the Z axis), it is recommended to print a model to prove that the calibration have been done perfectly.

An example of a incorrect calibration is shown in figure 5.37, because the nozzle and the heatbed do not have the adequate separation (less than expected), and the ABS could not be extruded as well as it ought to.

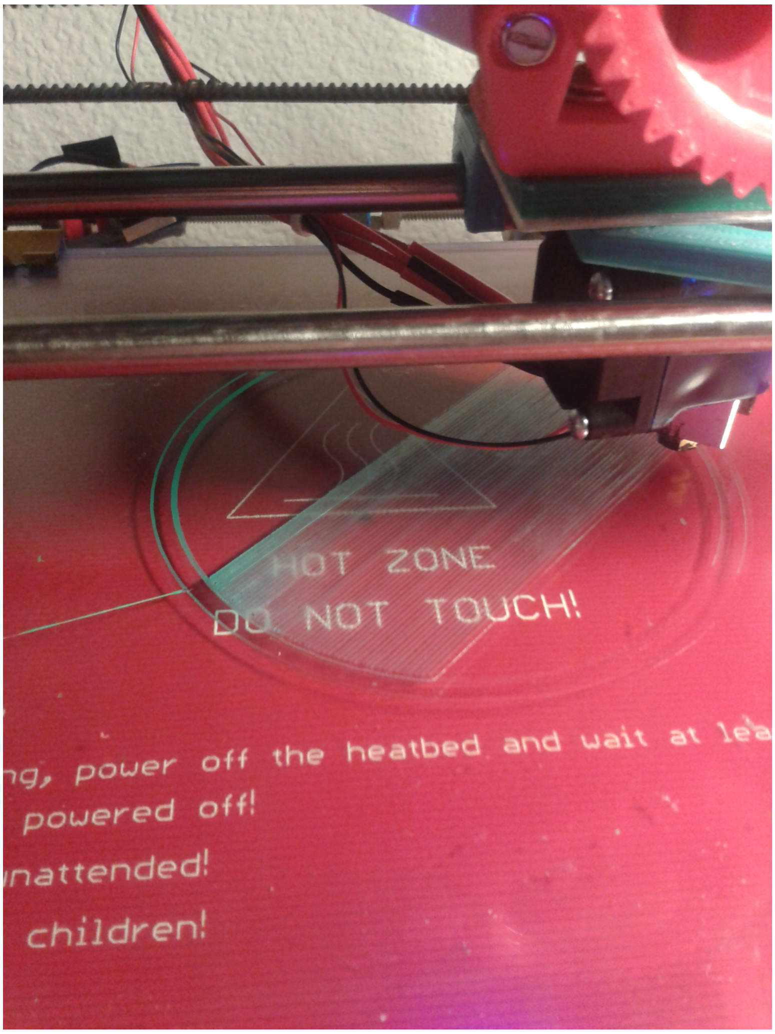
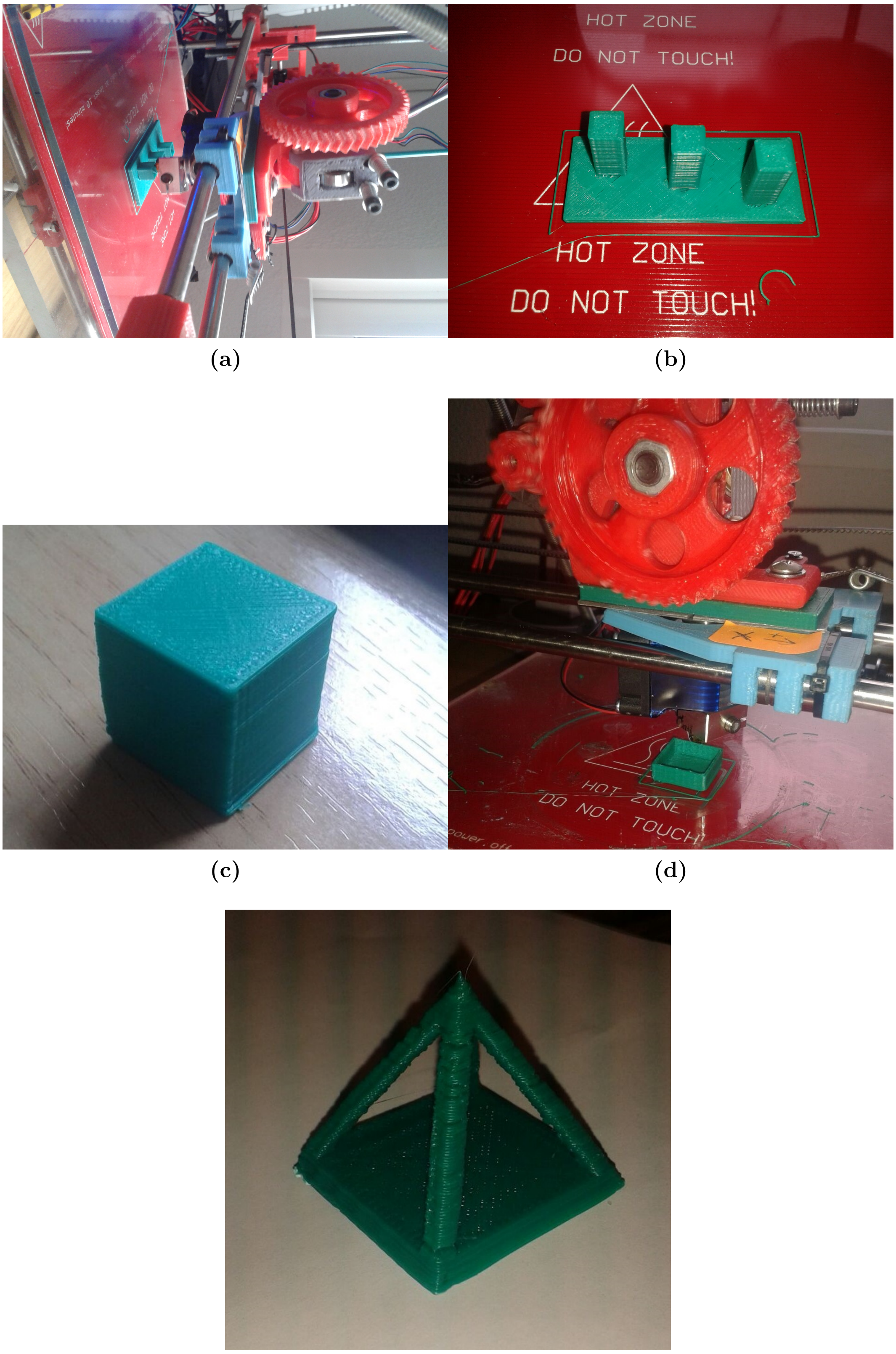


Figure 5.37 – *Result of a bad calibration*

The models that I used for this calibration were the shown in Figure 5.38. Seeing the quality of the corners, the layers, and the finish of these models, you can recalibrate the printer to enhance the features for the printing process.



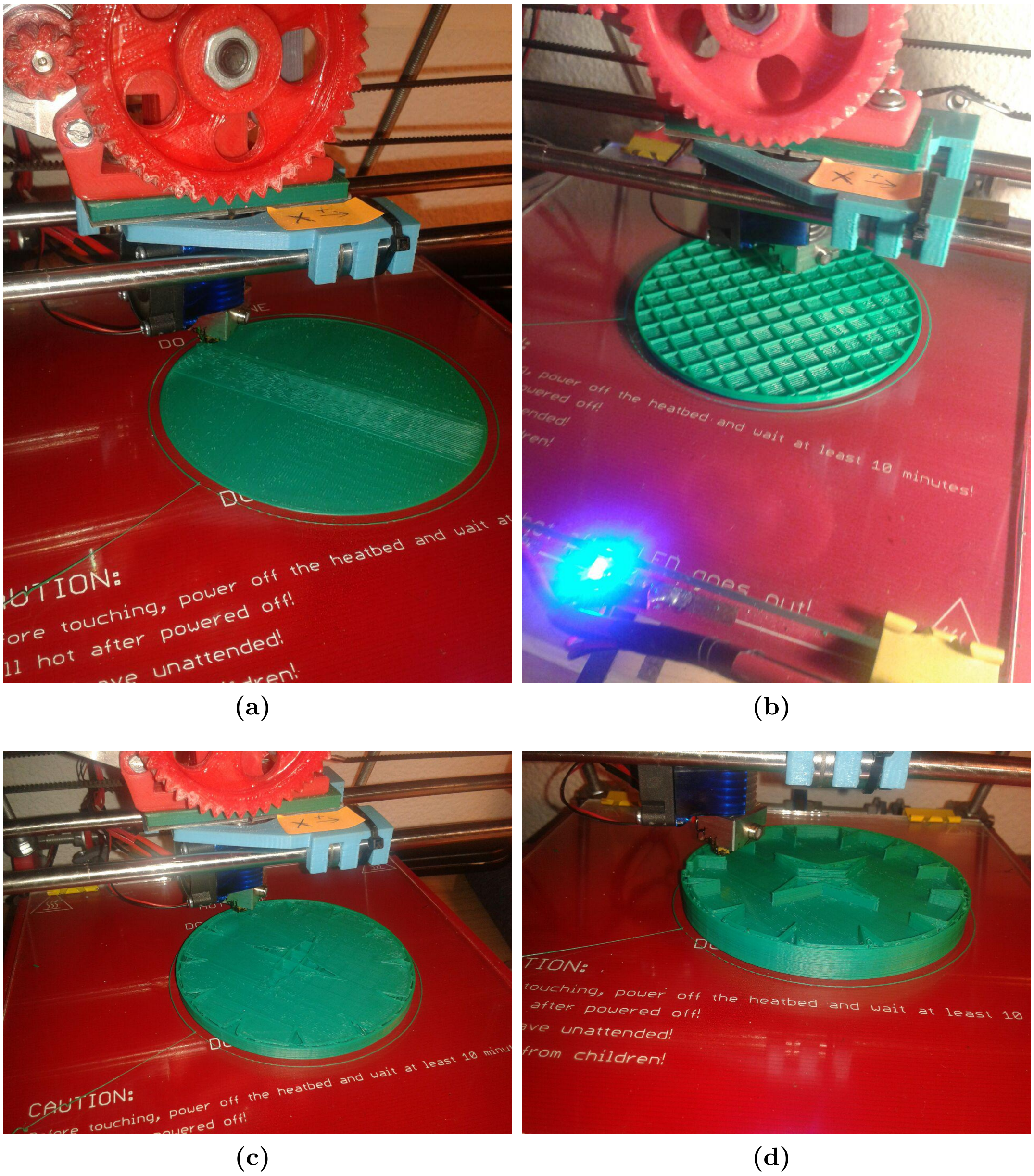
(a) **(b)**

(c) **(d)**

(e)

Figure 5.38 – *Models for the calibration*

When the calibration is ended, you can start to print the 3D model of your design. The first model that was designed was adapted to the dimensions of our Prusa Mendel i2, because the Prusa Mendel i3 was not available yet. So, for the first iteration, the model had a diameter of 10 centimeters. The design of this model can be seen in the Chapter 3.



(a) (b)

(c) (d)

Figure 5.39 – *Printing Testbed V1*

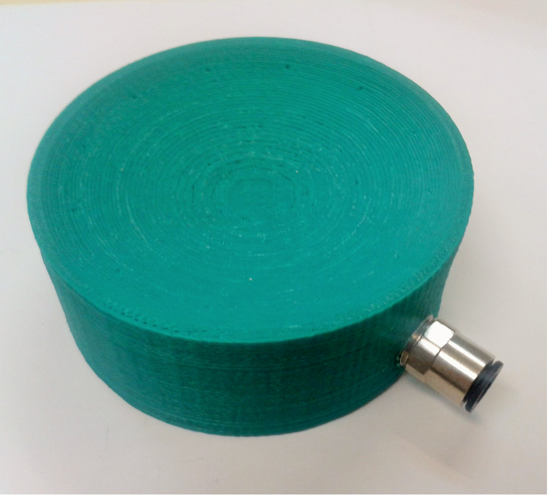


Figure 5.40 – *Testbed V1 printed*

The other 3D model, in this case the second iteration, was printed in the Prusa Mendel i3. Before the printing process, it was needed a calibration as described before.

With this version we had some problems with the drivers of the motors, because the drivers reached a high temperature and they start to fail skipping some steps in the Y axis. The result was the misalignment of the layers, shown in figure 5.41.

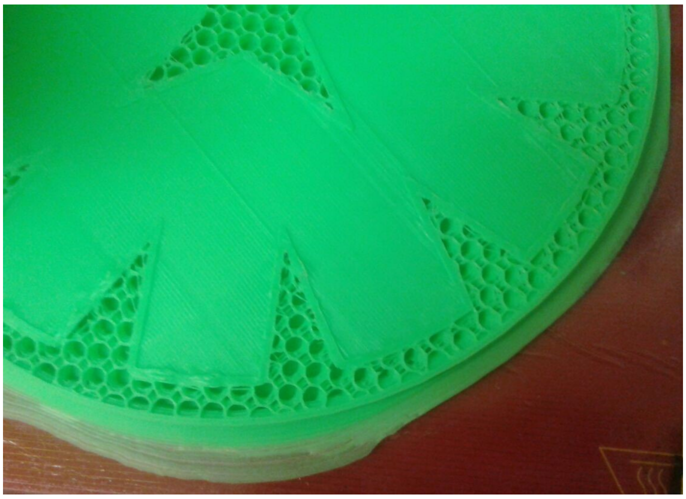
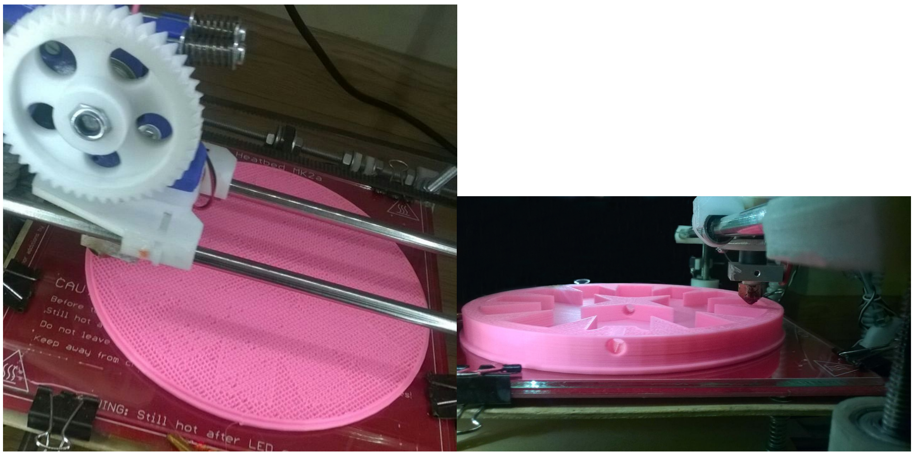


Figure 5.41 – *Failed printed part*

To solve this problem, we used the solution of the fan for the electronics (see chapter B.3.3), which allow us print the model correctly.

When the printer was perfectly prepared for the printing process, we could print the final Testbed part, shown in figure 5.42.


(a) (b)

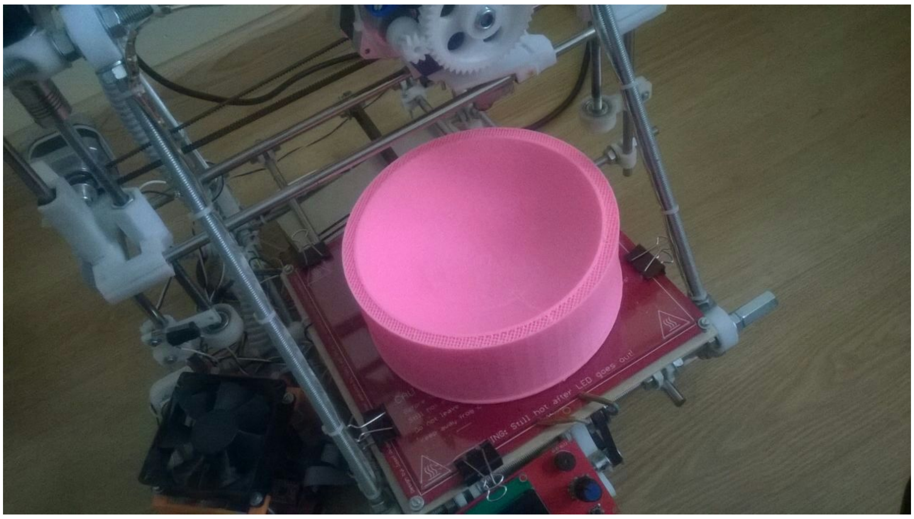

(c)

Figure 5.42 – *Printing the final version of the Testbed*

5.4 Air-bearing tests

To prove if the mechanical design of the Air-bearing was correct, was proven before the final implementation. That tests were made by degree of complexity, in order to organize what is working well or not.

5.4.1 First model of Air-bearing

The first model of the Air-bearing was smaller that the last one, because the printer that we had available in the laboratory when it was done was the Prusa i2, with a smaller printing volume than Prusa i3 (see section 5.3.2).

The Air-bearing had to fill-in the requirements described in the chapter 2, section 2.5.3.

One of the most important was the first one, fits the Air-bearing and the clear hemisphere in order to enhance the low-friction environment. To prove that requirement, first we introduce an air-compressed gun (see figure 5.43), which is supplied by the same compressor that we will use in the project. We adapt the gun's nozzle to the Air-bearing with a piece of gum, to avoid the air leak in that connection. The video of that test is in the following link: `https://www.youtube.com/watch?v=02u2mazmyNM`.

That video proves the no-friction movement that the hemisphere of the Testbed has with a mass in its inside of 453 g.

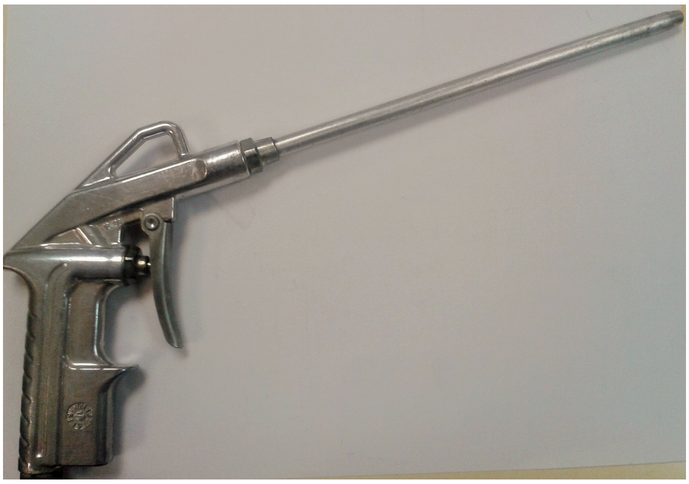


Figure 5.43 – *Airgun for air-compressor*

The second test that the first model of Air-bearing had was with a connector and a tube that fits perfectly with the Air-bearing, thanks to some adjustments hand-made with a sandpaper. In that case, the hemisphere was charged with more mass, because we had better no-friction surface, because of the improvement in the air input. That second test is shown in the video: `https://www.youtube.com/watch?v=pMs7rdHY1VA`.

Proving the air-compressor, we have realized that the compressor has not the sufficient capacity for our application. The air-pressure is changing in every second, and the compressed air run out in less than 1 minute. So that is a problem, because the tests for the control of the actuators have been really difficult to do. In chapter 6 we will explained the future lines of that projects and the possible improvements.

5.4.2 Second model of Air-bearing

Once the first model was working well, was the time to prove the second one. That model (final model for the air-bearing) is bigger than the first one (see schematics in section 4.2.2), so the constraints with the no-friction surface were harder.

In this case, the mass inside the hemisphere was the final PCB (386 g), and the test was succesfull. In order to have an idea of the result a video is uploaded to the link: https://www.youtube.com/watch?v=LL3_oYRGo68.

5.4.3 Mass of the main objects

The table 5.18 shows the values of the weight of each object mounted in the Air-bearing.

Part	Mass (g)
Clear hemisphere	144
Magnetorquers PCB	71
Testbed PCB	67
Motor PCB	4
Motor + Motor PCB + Glue	78
Reaction wheel (bronze)	57
DC Motor	19
Xbee Module	4
Battery	159
Screws	9
Washers and nuts	9
PCB with components welded	386

Table 5.18 – *Summary object mass*

5.5 Actuators tests

Now, once the Air-bearing is tested, the next step is see if the actuators will work correctly. The first test of that issue was with a failed PCB used to test, but not for the correct operation of the whole system. That PCB had some errors in the building process, so it can not be used for the final Testbed.

The test consisted of attaching the motor of Zaxis to the PCB and connect it to a power supply source, which will supply the motors with the voltage necessary for its operation and prove if the hemisphere rotated with respect to the first model of the Air-bearing. The result was successful. To prove it, a video have been uploaded to the following link: https://www.youtube.com/watch?v=2n80lCZw7ss.

And finally, the same test, but with the final model of the Air-bearing: https://www.youtube.com/watch?v=5vqpN39FNoQ.

Both results were well, but, when the weight of the test PCB was increased with the component's weight, the need of more air-compressed was the constraint, and here started the compressor capacity problem mentioned in section 5.4.1.

5.6 MPU6050 calibration

The library of that sensor has three offset values which are default designed, but that values can be wrong if the sensor is not well calibrated. The calibration code belongs to Jeff Rowberg, who programmed that calibration [32].

How it works [33]:

First, the code uses an initial value for each offset (supplied at the top of the sketch), takes a determined number of readings from the sensor (that number of readings can be defined on the top of the code, with variable countMax) and it use the average of that readings. Now we have the condition with three cases:

- If the average calculated is less than the value of the variable errorCheck, it assumes that axis has converged.
- If the average is greater than errorCheck, it will detract a certain value from the initial guess.
- If the errorCheck you set is too small, or the initial value is smaller than what the solution is, it will not converge - in this case either choose a larger value for errorCheck or a bigger initial value will be chosen.

One the code was compiled and the sensor calibrated the final values were really differents as default, so the sensor was not too well calibrated, the calibration test was necessary to confirm it and refine it. The table 5.19 shosh that results, scaled for minimum sensitivity.

Motor	Before calibration	After calibration
xGyroOffset	220	47
yGyroOffset	76	-38
zGyroOffset	-85	35
xAccelOffset	-	-1145
yAccelOffset	-	1166
zAccelOffset	1788	-60121

Table 5.19 – *MPU6050 calibration*

NOTES: yAccelOffset and xAccelOffset were not defined by default. To know if the sensor was well calibrated, an angle measuring stick was used.

5.7 Final implementation

Now, in this chapter, we will see the photos of the final implementation of this Testbed, with every parts mounted in their places.

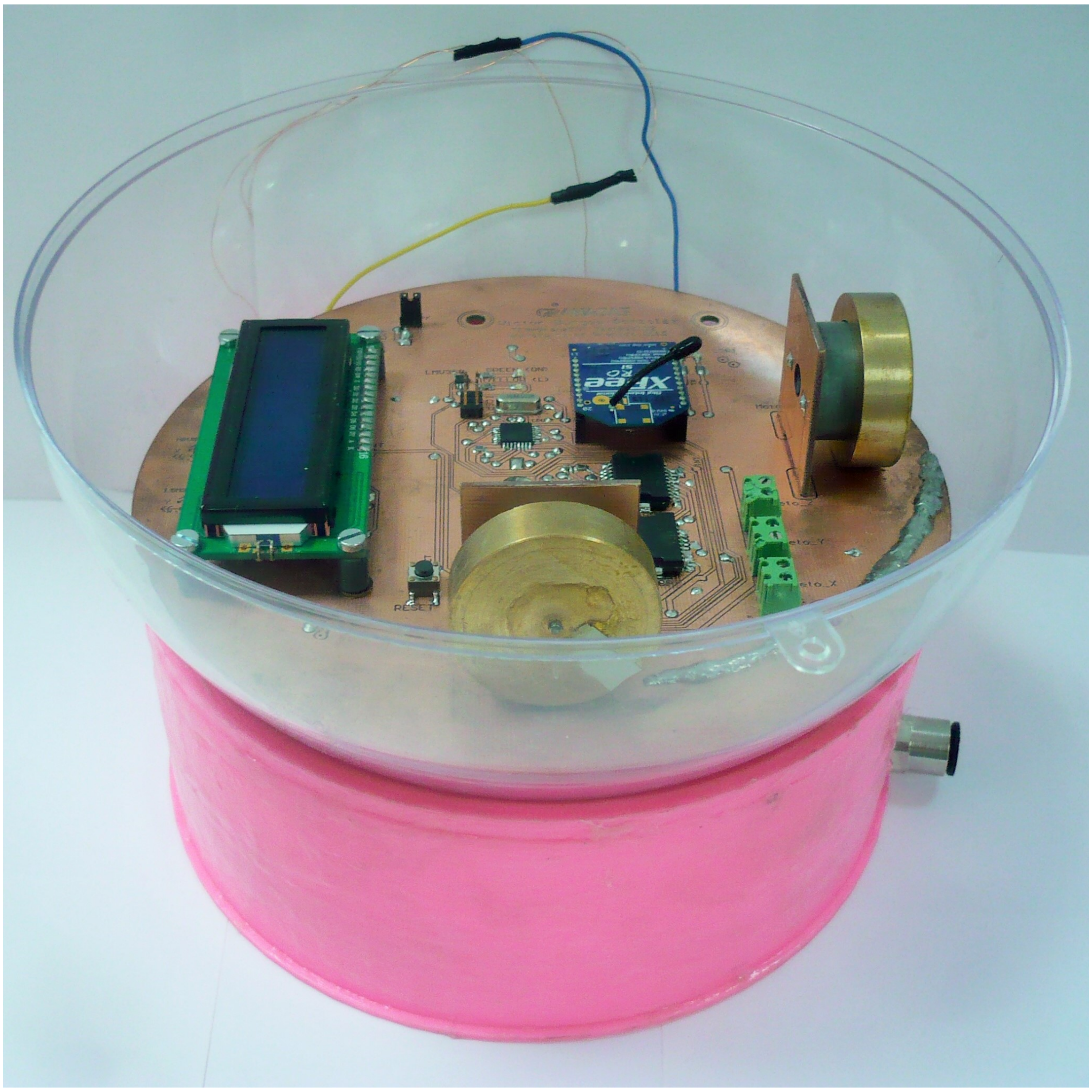


Figure 5.44 – *Final assembly with only Testbed PCB*

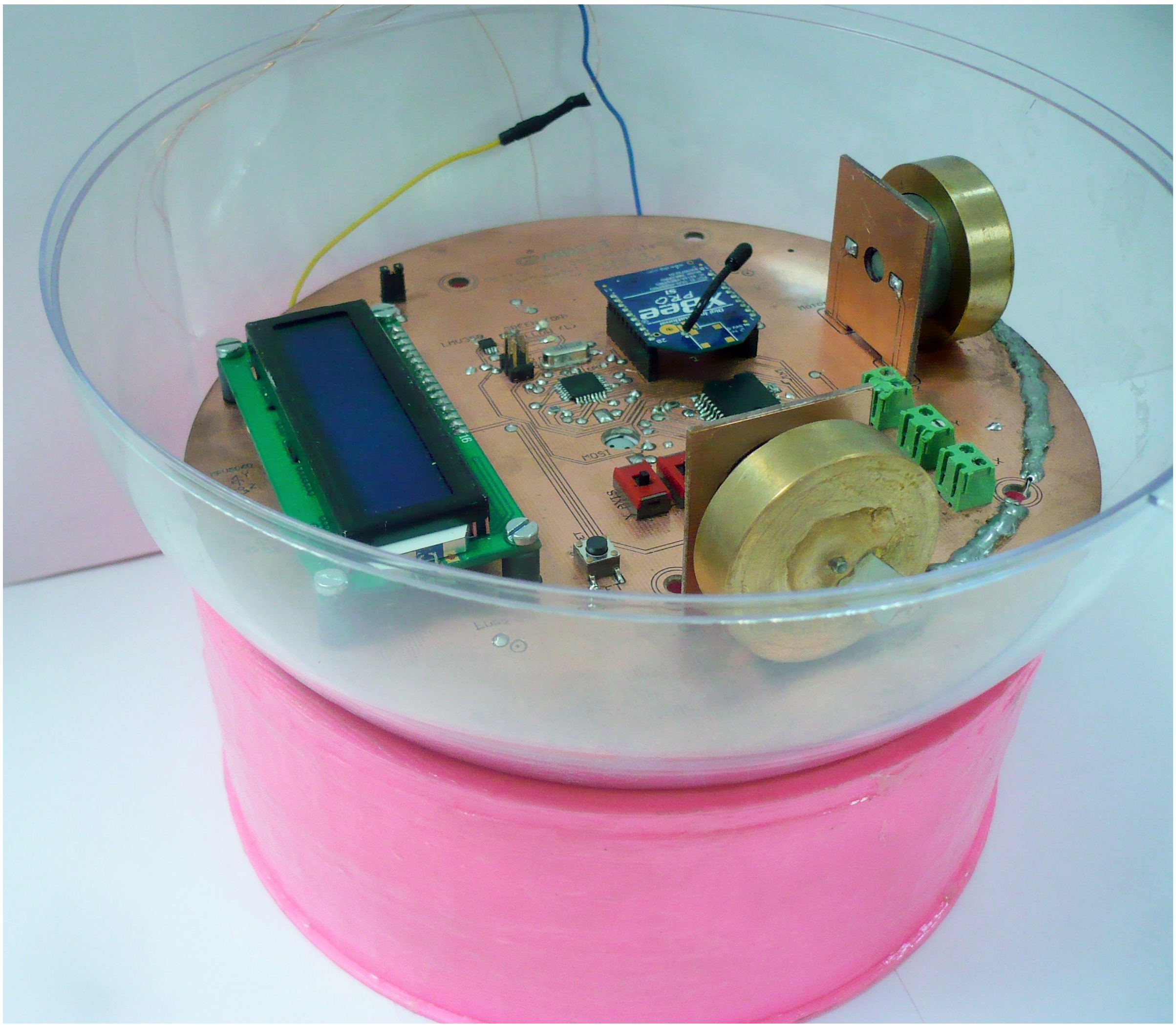


Figure 5.45 – *Final assembly with only Testbed PCB*

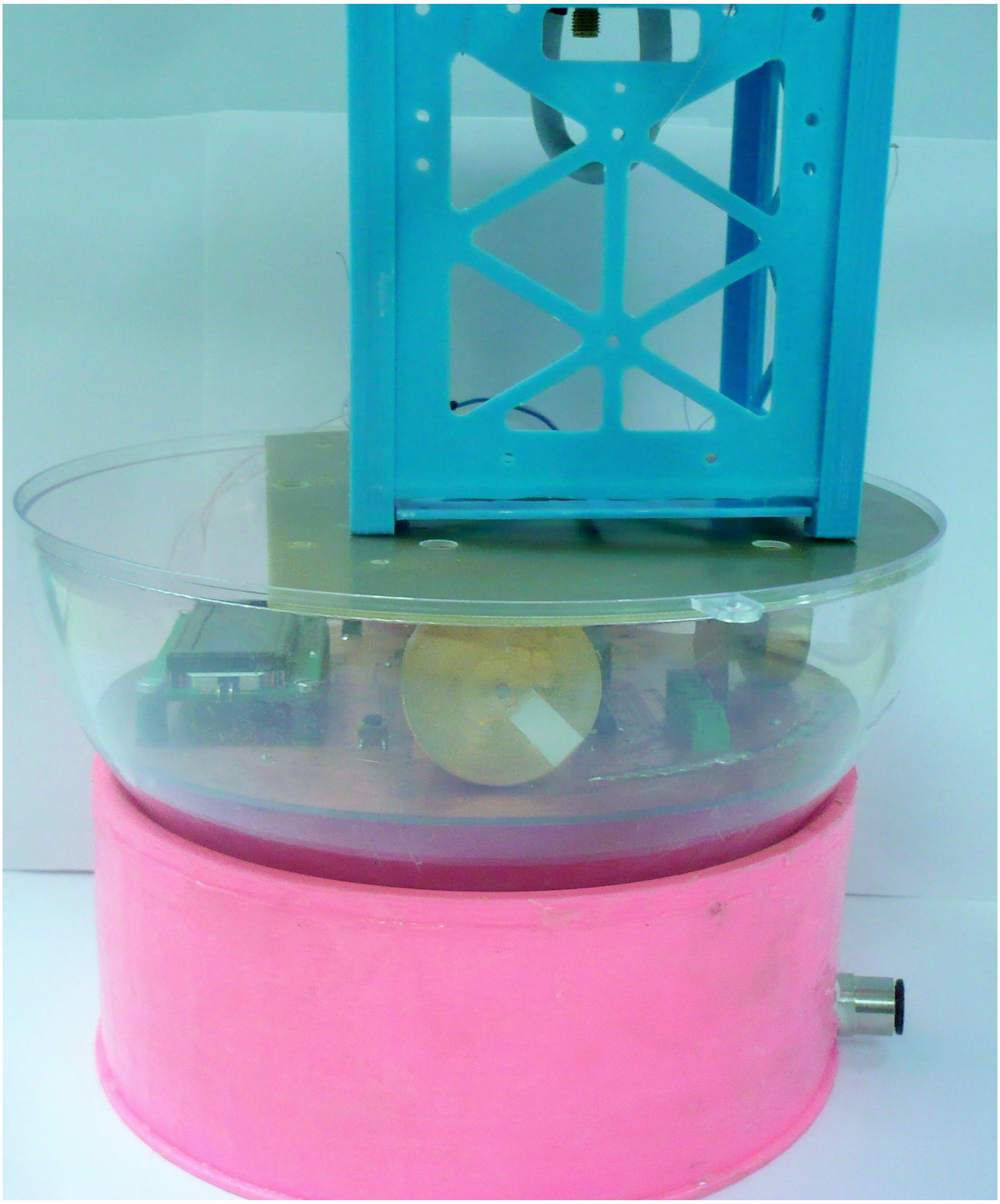


Figure 5.46 – *Final assembly Testbed*

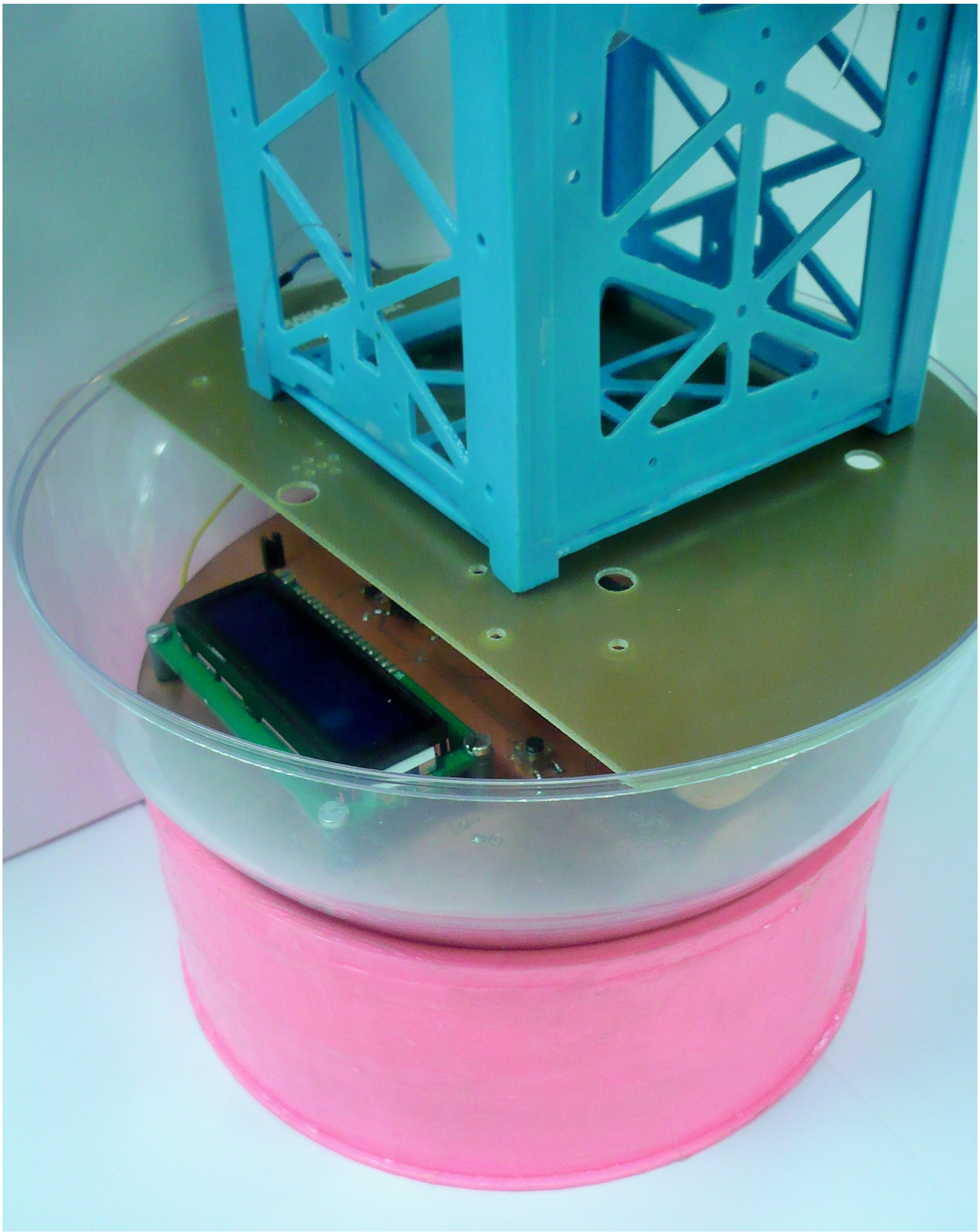


Figure 5.47 – *Final assembly Testbed*

CHAPTER

6

CONCLUSIONS AND FUTURE LINES

In this document, we have presented the design and development of the Testbed which will be used in the GranaSAT Cubesat. This system will be controlled with two sensors, and two types of actuators, one of them (reaction wheels) it has been studied in this Bachelor Thesis. That kind of project was a personal challenge, because it was the first time that I should deal with a complete new field as it is the aerospace one, on my own. For a Telecommunication Engineering student, as the author of this document, it has been really difficult to manage the attitude concepts and the mechanical background necessary for the reaction wheels, because it is a really unknown field.

One of the most important challenges in this project was the limited budget that we had for the development of that project. And the best conclusion that we have proven is that the greatest hits can be achieved with effort rather than a huge economical budget. Anyway, the ECTD have contributed to the project in every that they can.

Start practically from scratch, and achieve the project that we handle now, is one of the best experiences that a Junior (future) engineer could have, because that student have to think how to develop every subsystem, even with the tutor tips, it is the perfect practical experience, where you can apply the knowledge learned in the past four years, and have a little specialization in a field that maybe will be your future job.

One of the most important tasks that the future Testbed will have to take into account is the use of a better air-compressor, maybe a bigger one, with more capacity, which will be able to enhance the productivity of the tests, and support more weight in order to increase the dimensions of the subsystems that were mounted on it.

Moreover, a better 3D printer will enhance the quality of the low-friction surface, because of the quality of the layers extruded, and with less errors in the printing process.

We had some problems with the power system designed for the responsible of the Cubesat EPS. The needed components were no correctly bought, and once the Testbed PCB was designed, built and welded, it did not work properly, so we decided not use it for that prototype and use some wires to supply the system. That wires disturbed to the no-friction movement, so the PID of the actuators could not be calculated (in figure 5.44 that wires can be seen). The most relevant improvement that the Testbed should have is change the power system, in order to have a wireless Testbed to allow us measure the angular speed correctly and calculate the PID values.

Personally, I hope that this Testbed will be useful for future generations of the GranaSAT development team, and every student who need it. It will be the biggest satisfaction after so much time dealing with it.

We would like to conclude this discussion by remarking the support of the ECTD, which allow us the fabrication of the PCB without costs, and bought the majority of the electronic devices, as well as the tools used for this project, because without their support, this project would have not been possible.

APPENDIX

A

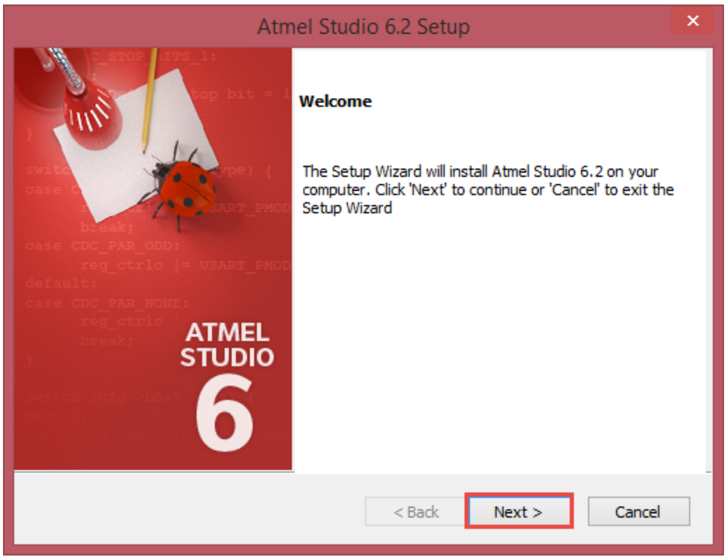
USER GUIDE: USING IN ATMEL STUDIO

Following the instructions of my advisor, that User Guide has been written with the objective to teach how to install and use the Atmel Studio 6 with Arduino to the future students in GranaSAT, because that is the best option to program it, thanks to the applications that Atmel Studio 6 has and the debugging option which is really useful.

A.1 How to compile Arduino's code in Atmel Studio 6

In order to use the Arduino's code in Atmel Studio 6, the following steps must be taken:

- First at all you must download Atmel Studio Software where you will be asked for create an *Atmel Account* or *Download as a Guest* giving your contact details.
- Once the software was downloaded, start the installation following the steps shown in the figures A.1 until A.6.
  - In the first window you will be asked to click in the *Next* button to start the installation (Figure A.1)



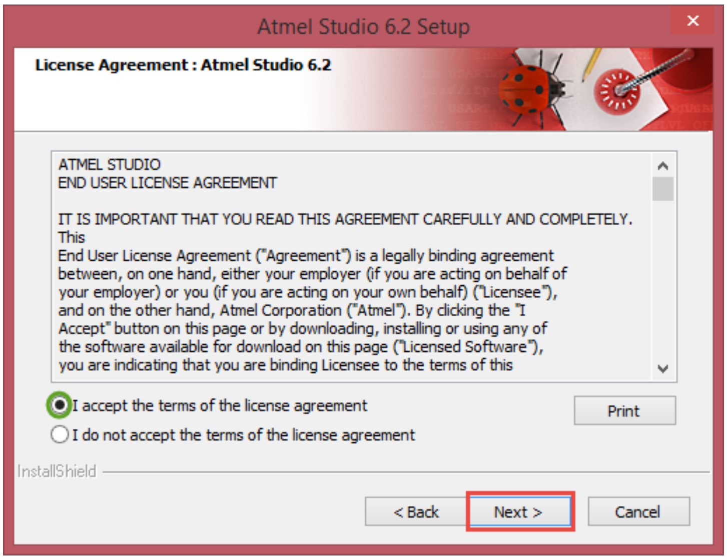
Atmel Studio 6.2 Setup

Welcome

The Setup Wizard will install Atmel Studio 6.2 on your computer. Click 'Next' to continue or 'Cancel' to exit the Setup Wizard

Figure A.1 – *Installation Atmel Studio (1)*

– The figure A.2 shows the Terms of the License, one you read them,accept them and click on *Next* button again:



Atmel Studio 6.2 Setup

License Agreement : Atmel Studio 6.2

ATMEL STUDIO
END USER LICENSE AGREEMENT

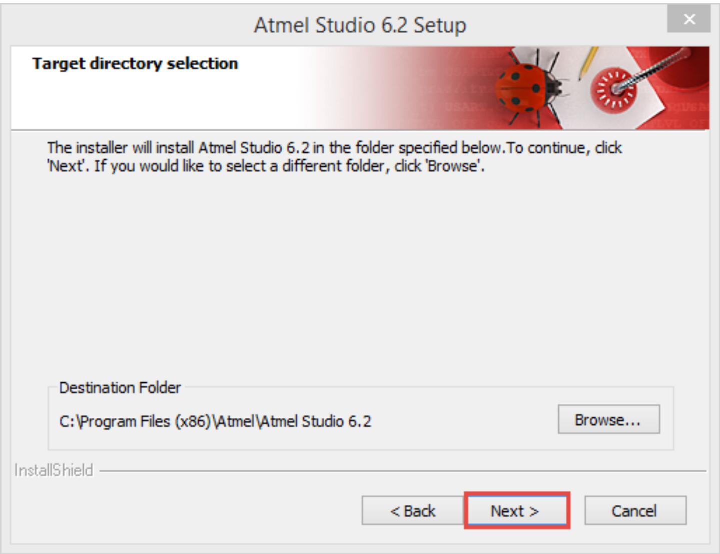
IT IS IMPORTANT THAT YOU READ THIS AGREEMENT CAREFULLY AND COMPLETELY.
This
End User License Agreement ("Agreement") is a legally binding agreement
between, on one hand, either your employer (if you are acting on behalf of
your employer) or you (if you are acting on your own behalf) ("Licensee"),
and on the other hand, Atmel Corporation ("Atmel"). By clicking the "I
Accept" button on this page or by downloading, installing or using any of
the software available for download on this page ("Licensed Software"),
you are indicating that you are binding Licensee to the terms of this

● I accept the terms of the license agreement

○ I do not accept the terms of the license agreement

Figure A.2 – *Installation Atmel Studio (2)*

– The following window, figure A.3, is used to select the destination folder of the installation, in my case is *C:\Program Files (x86)\Atmel\Atmel Studio 6.2*.



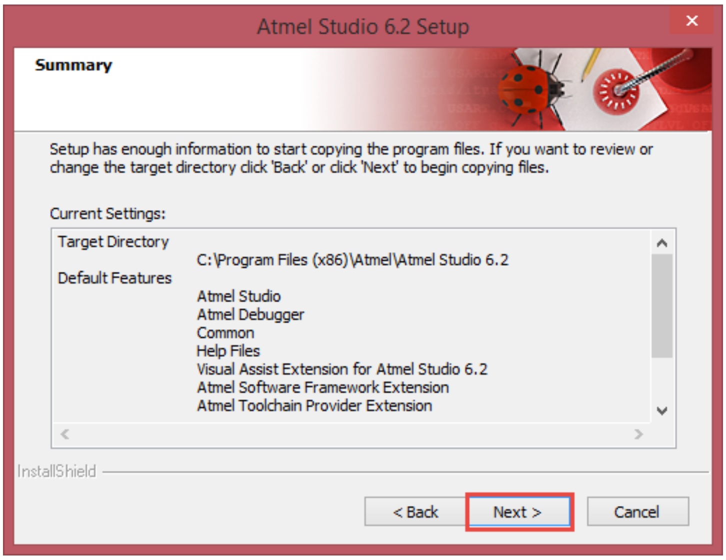
The installer will install Atmel Studio 6.2 in the folder specified below.To continue, click 'Next'. If you would like to select a different folder, click 'Browse'.

Destination Folder

C:\Program Files (x86)\Atmel\Atmel Studio 6.2

Figure A.3 – *Installation Atmel Studio (3)*

- Finally, as you can see in the figure A.4, you will have a resume of the programs that are going to be installed. [14]



Setup has enough information to start copying the program files. If you want to review or change the target directory click 'Back' or click 'Next' to begin copying files.

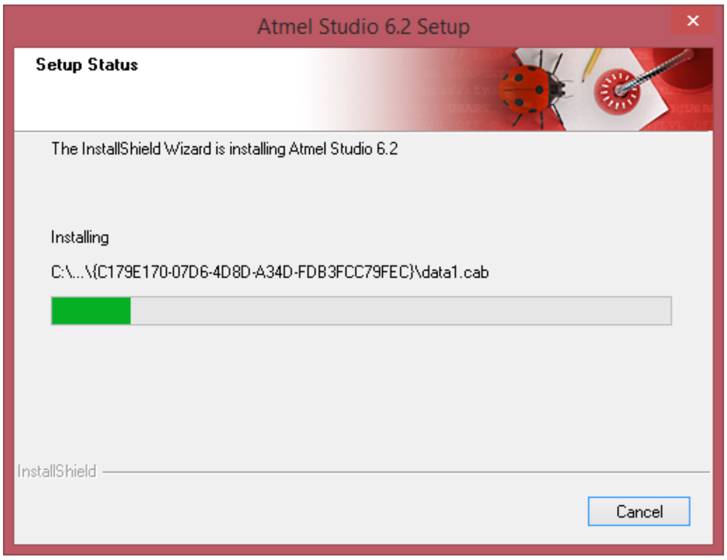
Current Settings:

Target Directory
C:\Program Files (x86)\Atmel\Atmel Studio 6.2

Default Features
Atmel Studio
Atmel Debugger
Common
Help Files
Visual Assist Extension for Atmel Studio 6.2
Atmel Software Framework Extension
Atmel Toolchain Provider Extension

Figure A.4 – *Installation Atmel Studio (4)*

- The next window (figure A.5) shows how the progress bar of the installation becomes filled, it will take a few minutes.


Atmel Studio 6.2 Setup

Setup Status

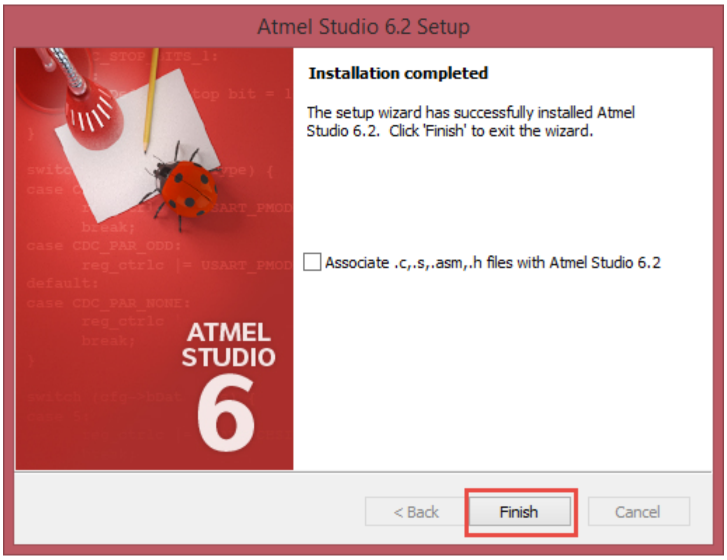
The InstallShield Wizard is installing Atmel Studio 6.2

Installing

C:\...\{C179E170-07D6-4D8D-A34D-FDB3FCC79FEC}\data1.cab

Figure A.5 – *Installation Atmel Studio (5)*

- To finalize, click on the button *Finish* in the following window (figure A.6. If you want, you can select the files association option.


Atmel Studio 6.2 Setup

Installation completed

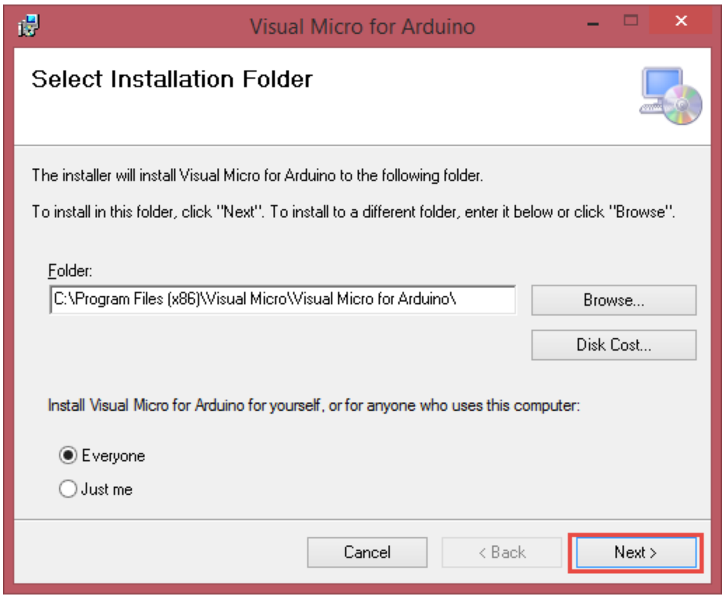
The setup wizard has successfully installed Atmel Studio 6.2. Click 'Finish' to exit the wizard.

☐ Associate .c,.s,.asm,.h files with Atmel Studio 6.2

Figure A.6 – *Installation Atmel Studio (6)*

- Now, you have the Atmel Studio 6.2 installed and prepared to compile and debug your programs, but in this User's Guide, we want to teach you how to compile Arduino's Integrated Development Environment (IDE) programs (.ino). To do this, you will have to download the plugin called *Visual Micro* downloadable here clicking on the button *Download Latest Visual Micro*. Important note: It is necessary to have the Arduino's IDE software installed, because when you execute the plugin, it will ask you the route where the Arduino's compiler is installed, so the plugin will be able to take the libraries and files required.

- After downloading the plugin, you will have to follow the next steps:
  - First, in the first window (figure A.7)it is necessary to browse the folder where the plugin are going to install, and select the option if you want to instal for *Everyone* who uses the computer or *Just me*, only for your user account. Click on next to continue the installation.



Select Installation Folder

The installer will install Visual Micro for Arduino to the following folder.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

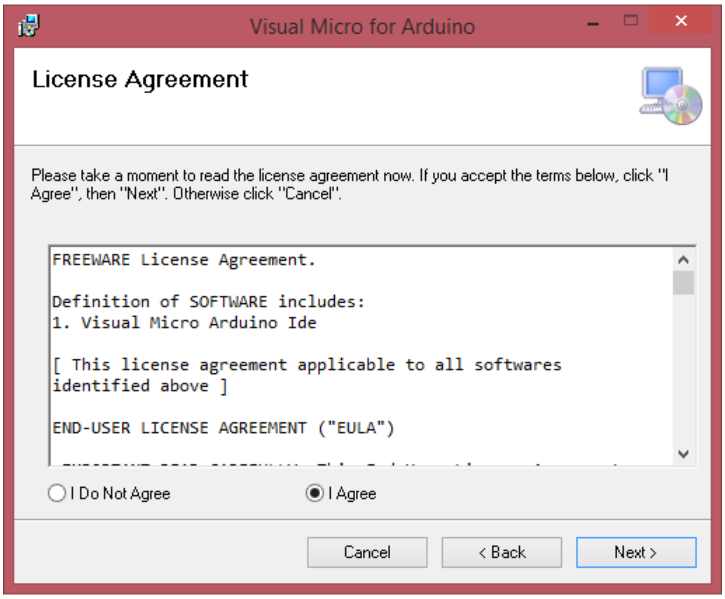
Folder:
C:\Program Files (x86)\Visual Micro\Visual Micro for Arduino\

Install Visual Micro for Arduino for yourself, or for anyone who uses this computer:

● Everyone
○ Just me

Figure A.7 – *Installation Visual Micro (1)*

  - The figure A.8 shows the next window in the instllation, the *License Agreement*, you should read them carefully and agree them.



License Agreement

Please take a moment to read the license agreement now. If you accept the terms below, click "I Agree", then "Next". Otherwise click "Cancel".

FREEWARE License Agreement.

Definition of SOFTWARE includes:
1. Visual Micro Arduino Ide

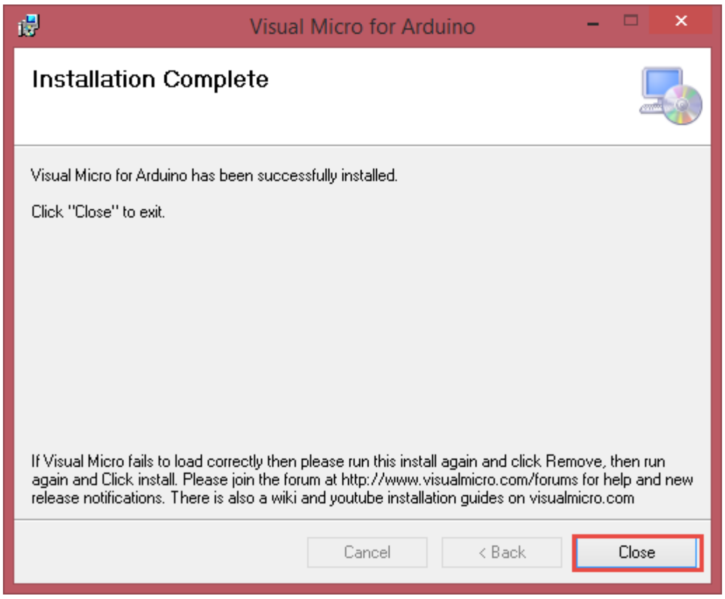
[This license agreement applicable to all softwares identified above]

END-USER LICENSE AGREEMENT ("EULA")

○ I Do Not Agree ● I Agree

Figure A.8 – *Installation Visual Micro (2)*

- The installation will be completed on the next window and you will have to click on *Close* button (see Figure A.9) to close it.



Installation Complete

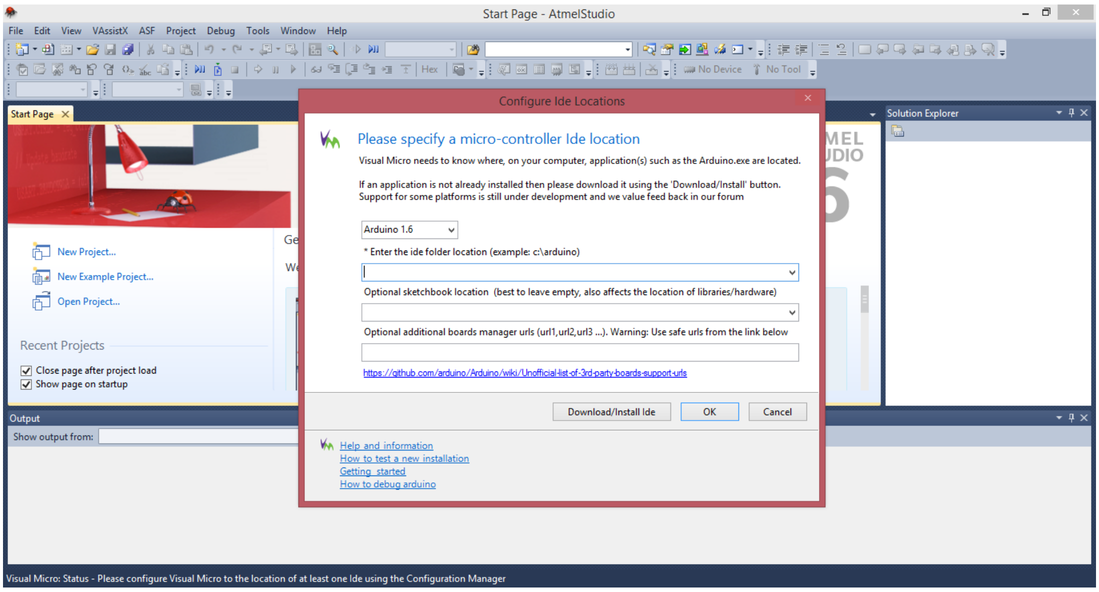
Visual Micro for Arduino has been successfully installed.

Click "Close" to exit.

If Visual Micro fails to load correctly then please run this install again and click Remove, then run again and Click install. Please join the forum at http://www.visualmicro.com/forums for help and new release notifications. There is also a wiki and youtube installation guides on visualmicro.com

Figure A.9 – *Installation Visual Micro (3)*

- After the installation, when you open the Atmel Studio, will appear the path selection window that you can see in the Figure A.10.



Configure Ide Locations

Please specify a micro-controller Ide location

Visual Micro needs to know where, on your computer, application(s) such as the Arduino.exe are located.

If an application is not already installed then please download it using the 'Download/Install' button.
Support for some platforms is still under development and we value feed back in our forum

Arduino 1.6

* Enter the ide folder location (example: c:\arduino)

Optional sketchbook location (best to leave empty, also affects the location of libraries/hardware)

Optional additional boards manager urls (url1,url2,url3 ...). Warning: Use safe urls from the link below

https://github.com/arduino/Arduino/wiki/Unofficial-list-of-3rd-party-boards-support-urls

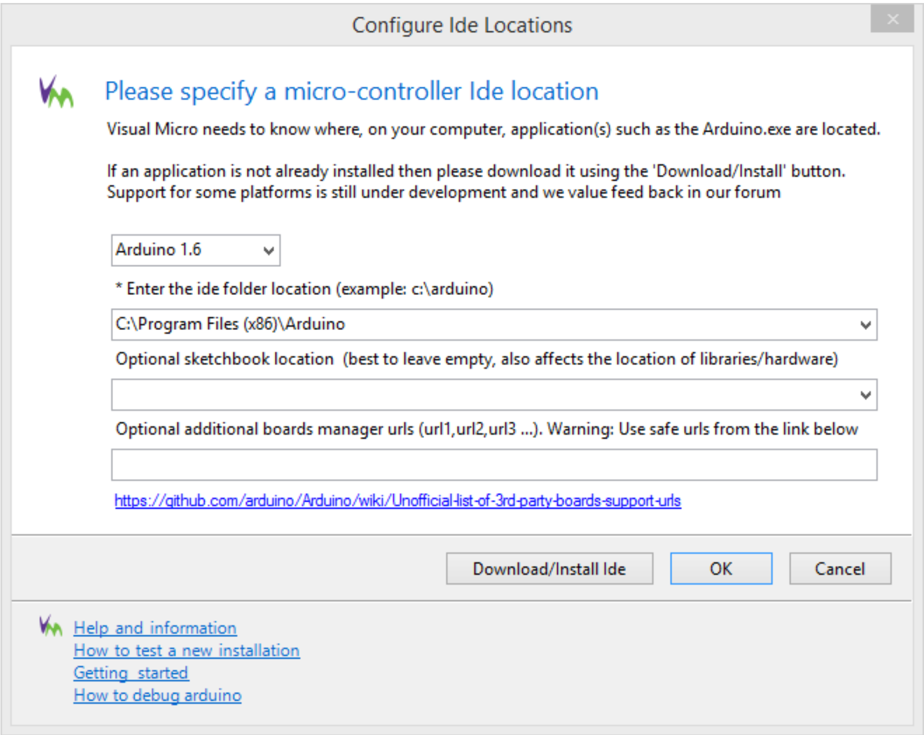
Help and information
How to test a new installation
Getting started
How to debug arduino

☑ Close page after project load
☑ Show page on startup

Visual Micro: Status - Please configure Visual Micro to the location of at least one Ide using the Configuration Manager

Figure A.10 – *Installation Visual Micro (4)*

– And you should enter the Arduino's IDE folder location in the second editable text gap, and the version of your Arduino's IDE in the first one:



Configure Ide Locations

Please specify a micro-controller Ide location

Visual Micro needs to know where, on your computer, application(s) such as the Arduino.exe are located.

If an application is not already installed then please download it using the 'Download/Install' button.
Support for some platforms is still under development and we value feed back in our forum

Arduino 1.6

* Enter the ide folder location (example: c:\arduino)

C:\Program Files (x86)\Arduino

Optional sketchbook location (best to leave empty, also affects the location of libraries/hardware)

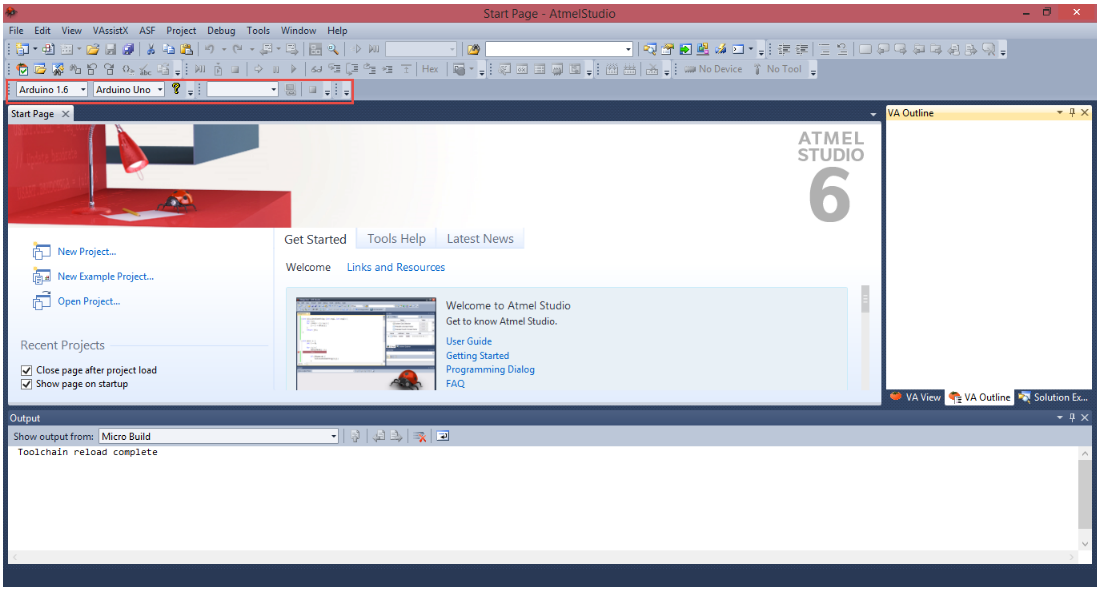
Optional additional boards manager urls (url1,url2,url3 ...). Warning: Use safe urls from the link below

https://github.com/arduino/Arduino/wiki/Unofficial-list-of-3rd-party-boards-support-urls

Help and information
How to test a new installation
Getting started
How to debug arduino

Figure A.11 – *Installation Visual Micro (5)*

– To prove that the installation have been correct, you should see the options remarked with the red box:



Welcome to Atmel Studio

Get to know Atmel Studio.

☑ Close page after project load

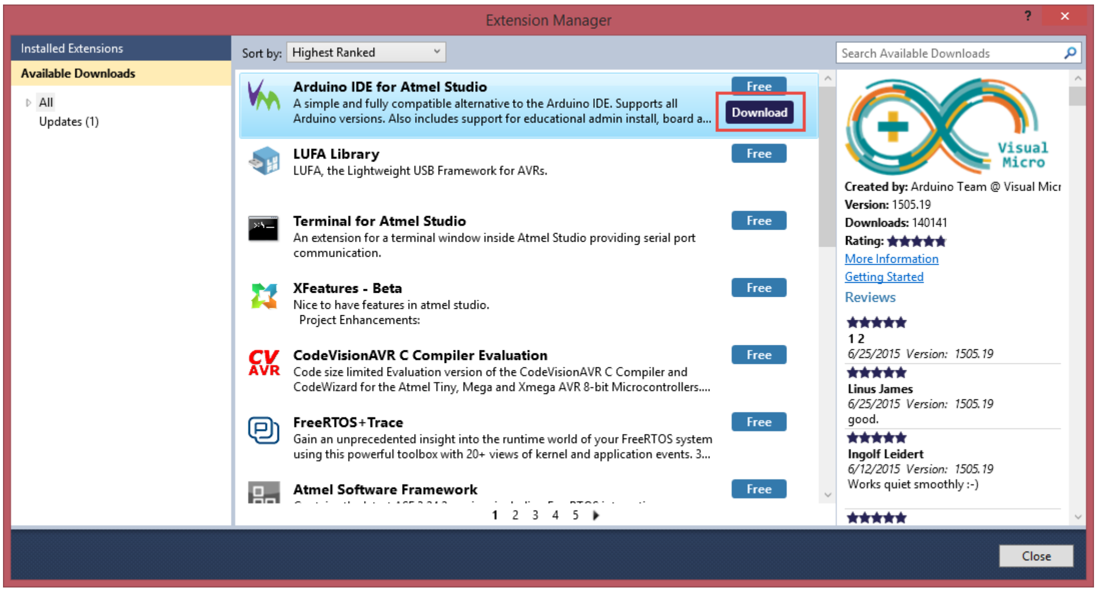
☑ Show page on startup

Show output from: Micro Build

```
Toolchain reload complete
```

Figure A.12 – *Installation Visual Micro (6)*

- Finally, open te menu *Tools* and select Extension Manager, download the plugin of the Arduino IDE (see Figure A.13). Maybe, you will have to enter your user name and password of your Atmel account.



Sort by: Highest Ranked

Arduino IDE for Atmel Studio
A simple and fully compatible alternative to the Arduino IDE. Supports all Arduino versions. Also includes support for educational admin install, board a...

LUFA Library
LUFA, the Lightweight USB Framework for AVRs.

Terminal for Atmel Studio
An extension for a terminal window inside Atmel Studio providing serial port communication.

XFeatures - Beta
Nice to have features in atmel studio.
Project Enhancements:

CodeVisionAVR C Compiler Evaluation
Code size limited Evaluation version of the CodeVisionAVR C Compiler and CodeWizard for the Atmel Tiny, Mega and Xmega AVR 8-bit Microcontrollers....

FreeRTOS+Trace
Gain an unprecedented insight into the runtime world of your FreeRTOS system using this powerful toolbox with 20+ views of kernel and application events. 3...

Atmel Software Framework

Created by: Arduino Team @ Visual Micr
Version: 1505.19
Downloads: 140141
Rating: ★★★★★

★★★★★
1 2
6/25/2015 Version: 1505.19

★★★★★
Linus James
6/25/2015 Version: 1505.19
good.

★★★★★
Ingolf Leidert
6/12/2015 Version: 1505.19
Works quiet smoothly :-)

★★★★★

Figure A.13 – *Installation Visual Micro (7)*

- Now, we have the necessary software to compile the Arduino's Code in Atmel Studio.
  - First, we have to create the Arduino Proyect, in the menu *File/New/Arduino Project*, as you can see in the Figure A.14.

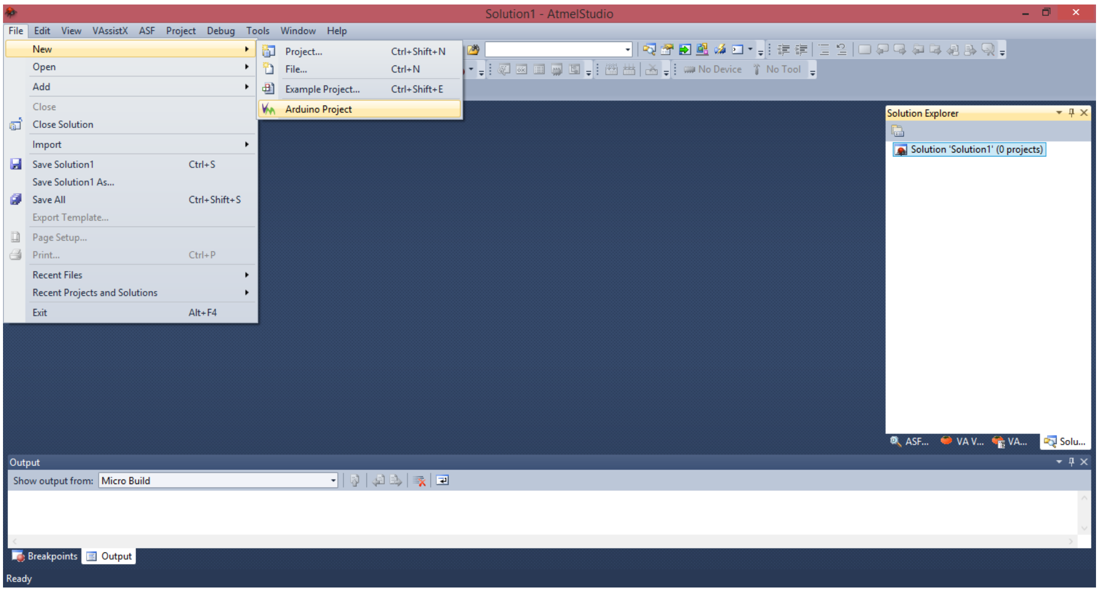
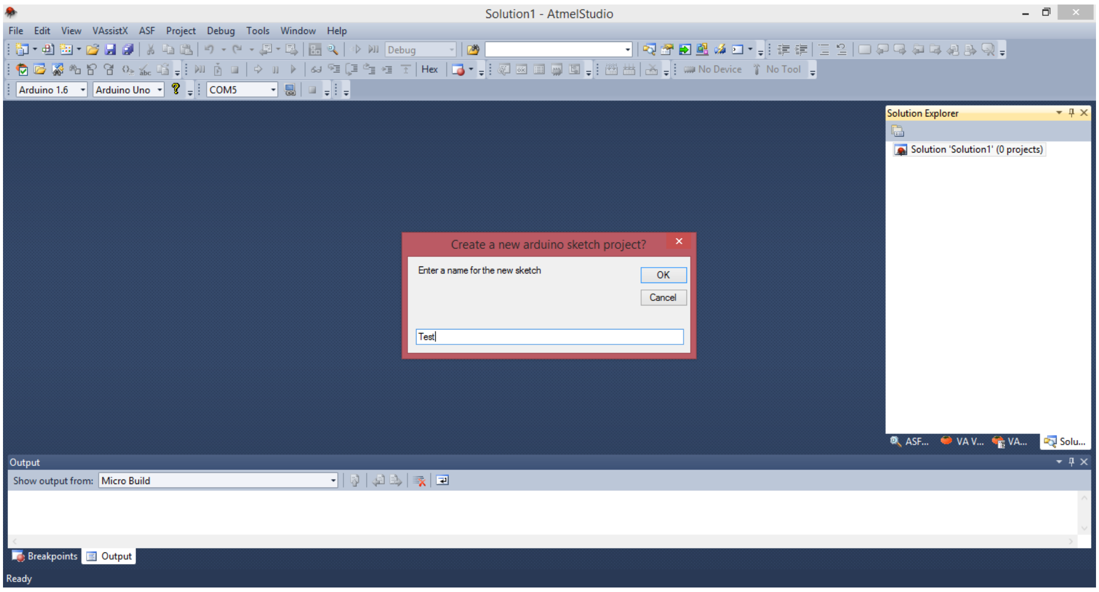


Figure A.14 – *Compiling the Arduino Code in Atmel Studio (1)*

– Now, a window will appear (Figure A.15) asking us for a name for the Sketch. In my case, I have choosen *Test*. Then click on OK to create it.

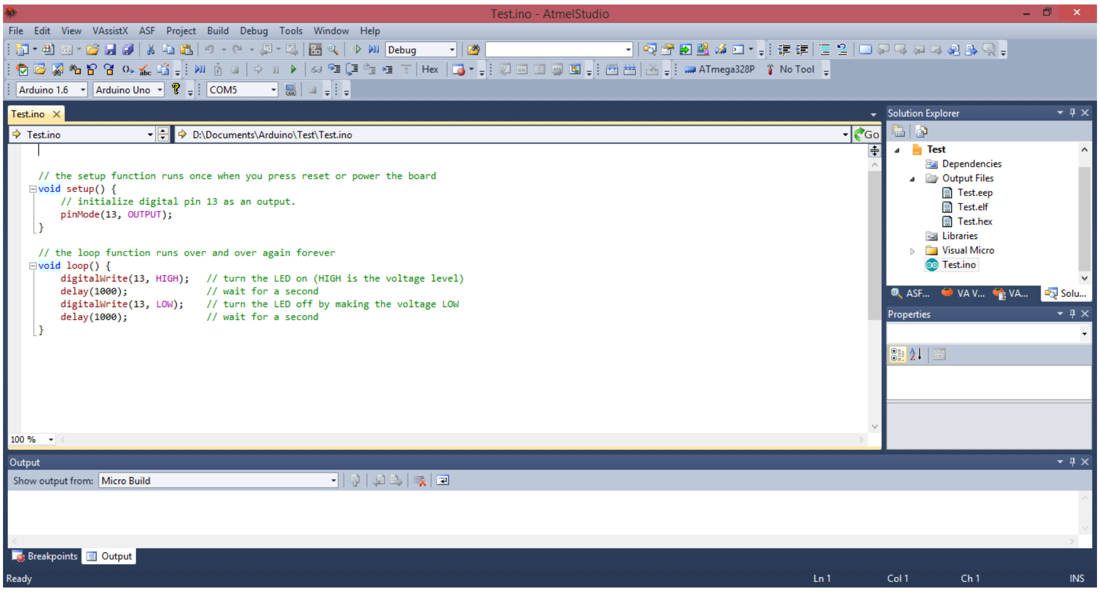

Create a new arduino sketch project?

Enter a name for the new sketch

Test

Figure A.15 – *Compiling the Arduino Code in Atmel Studio (2)*

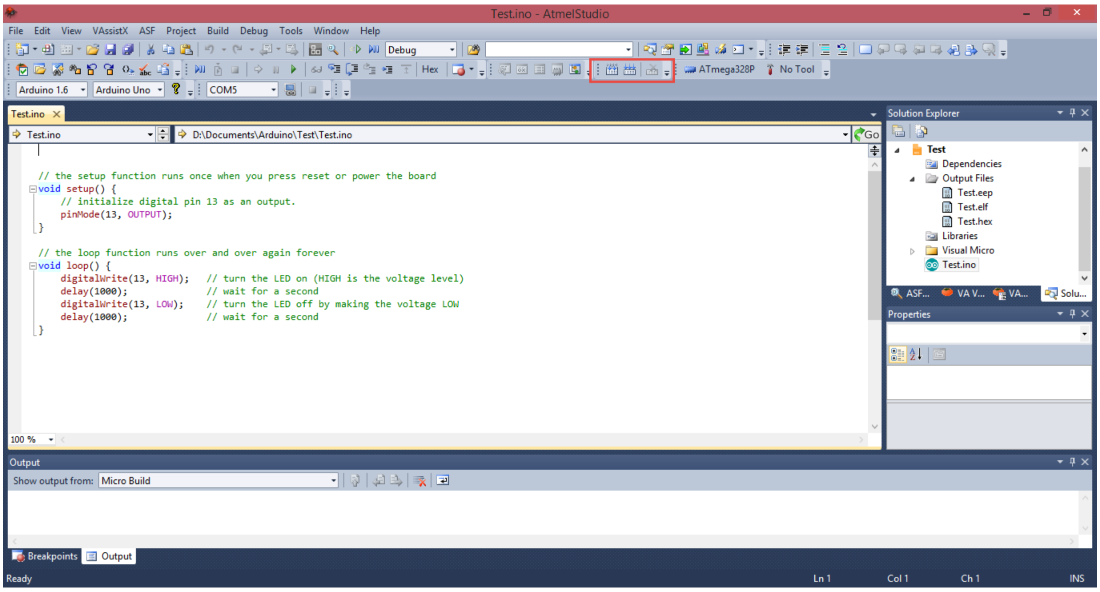
– The next step will be write the Arduino Code. I have write a common example for test the Arduino, the Blink test. The Figure A.16 shows how you will see it in the Atmel Studio Environment.



```
// the setup function runs once when you press reset or power the board
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
    delay(1000);              // wait for a second
    digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
    delay(1000);              // wait for a second
}
```

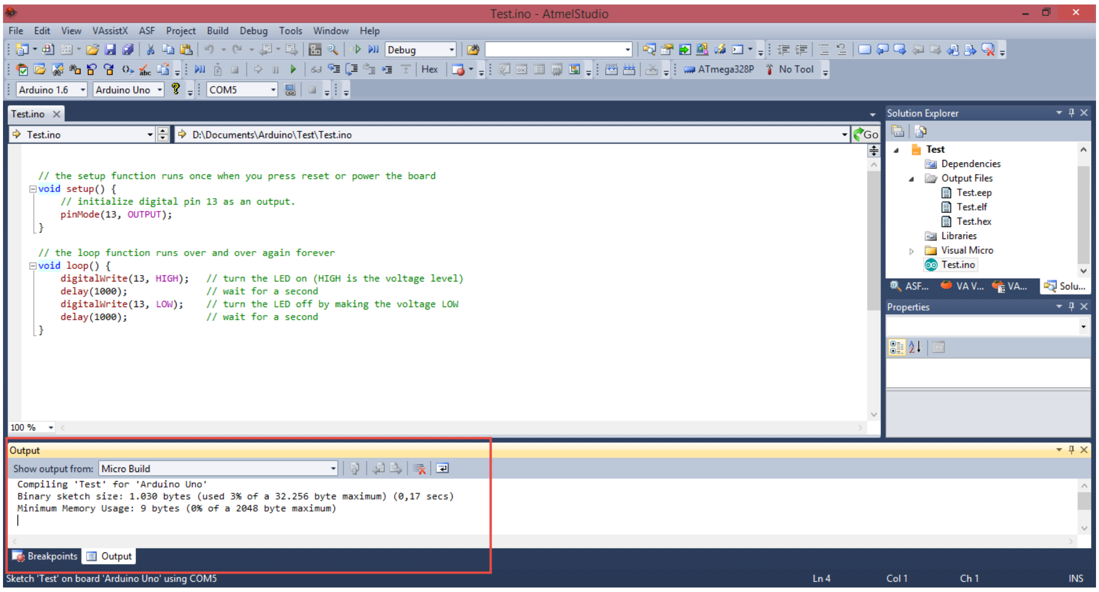
Figure A.16 – *Compiling the Arduino Code in Atmel Studio (3)*

– Once the code was finished, it is the moment of compile it. In the Figure A.17, the red square mark the compilation (build) buttons. The first one is used to *Build* the .ino file, and the second one is used to build the whole project. In this case, I will click on the first one and when the compilation have finished correctly, it will be shown in the *output* window the information that you can see in the Figure A.18



```
// the setup function runs once when you press reset or power the board
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
    delay(1000);              // wait for a second
    digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
    delay(1000);              // wait for a second
}
```

Figure A.17 – *Compiling the Arduino Code in Atmel Studio (4)*



```
// the setup function runs once when you press reset or power the board
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
    delay(1000);              // wait for a second
    digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
    delay(1000);              // wait for a second
}
```

Show output from: Micro Build

```
Compiling 'Test' for 'Arduino Uno'
Binary sketch size: 1.030 bytes (used 3% of a 32.256 byte maximum) (0,17 secs)
Minimum Memory Usage: 9 bytes (0% of a 2048 byte maximum)
```

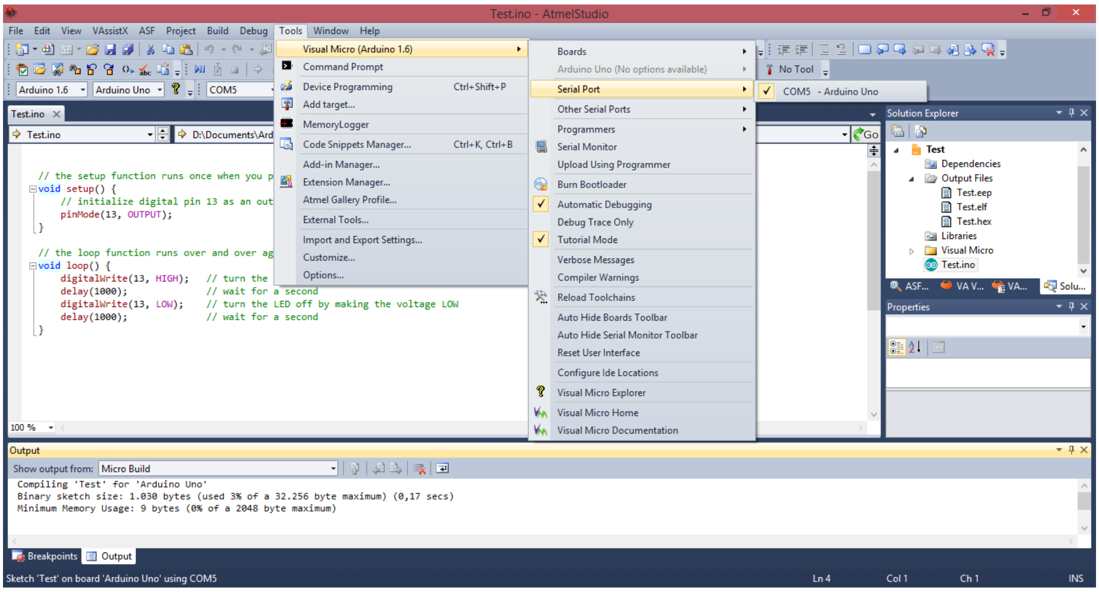
Sketch 'Test' on board 'Arduino Uno' using COM5

Figure A.18 – *Compiling the Arduino Code in Atmel Studio (5)*

A.2 Uploading the Arduino Code in Atmel Studio

If you want to upload the compiled code to an Arduino directly with an USB port (*Arduino UNO* in this example) the steps that you should to take are the following:

- First, it is necessary to select the *Board* that you are using and the COM Port where the Arduino is located (see Figure A.19). In this case, the COM port selected is COM5.



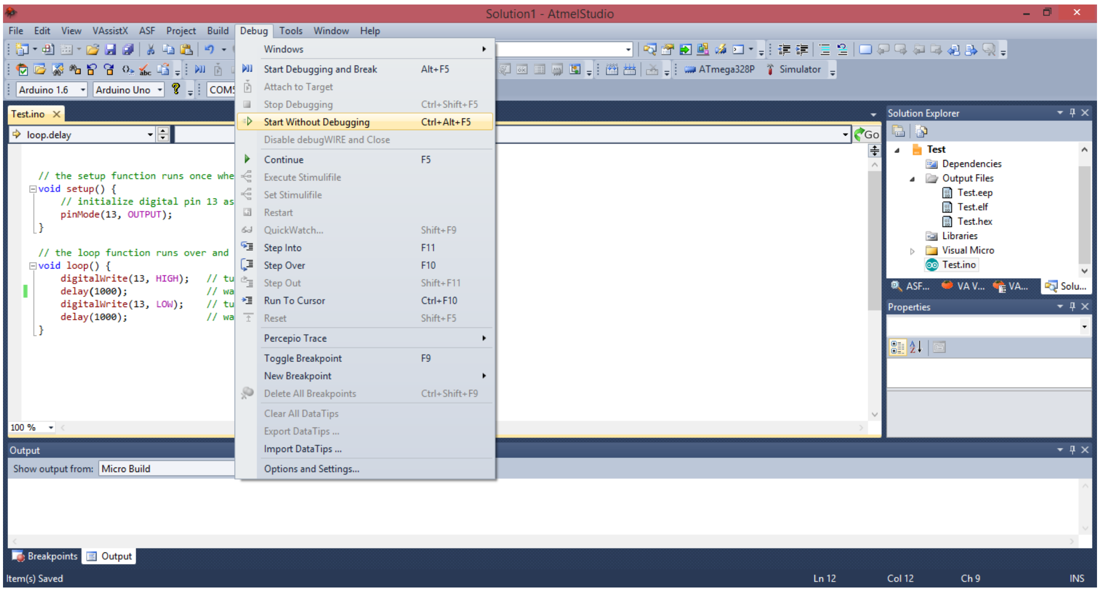
Show output from: Micro Build

```
Compiling 'Test' for 'Arduino Uno'
Binary sketch size: 1.030 bytes (used 3% of a 32.256 byte maximum) (0,17 secs)
Minimum Memory Usage: 9 bytes (0% of a 2048 byte maximum)
```

Sketch 'Test' on board 'Arduino Uno' using COM5

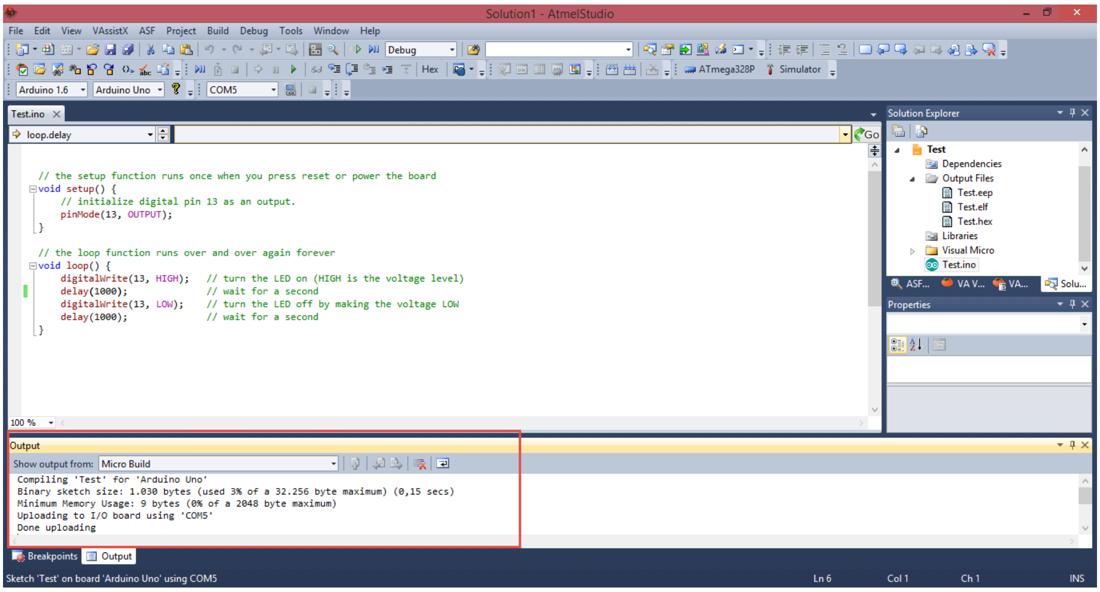
Figure A.19 – *Uploading the Arduino Code in Atmel Studio (6)*

- As we know, the Arduino has not the Debug option, so we will use the Option *Start Without Debugging* in the Menu *Debug* (see Figure A.20), an the code will be uploaded automatically. You will know that the uploading was correct if you see the *Output* window and see something like the figure A.21.



```
 // the setup function runs once whe
void setup() {
    // initialize digital pin 13 as
    pinMode(13, OUTPUT);
}

 // the loop function runs over and
void loop() {
    digitalWrite(13, HIGH);   // tu
    delay(1000);              // wa
    digitalWrite(13, LOW);    // tu
    delay(1000);              // wa
}
```

Figure A.20 – *Uploading the Arduino Code in Atmel Studio (7)*



```
 // the setup function runs once when you press reset or power the board
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

 // the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
    delay(1000);              // wait for a second
    digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
    delay(1000);              // wait for a second
}
```

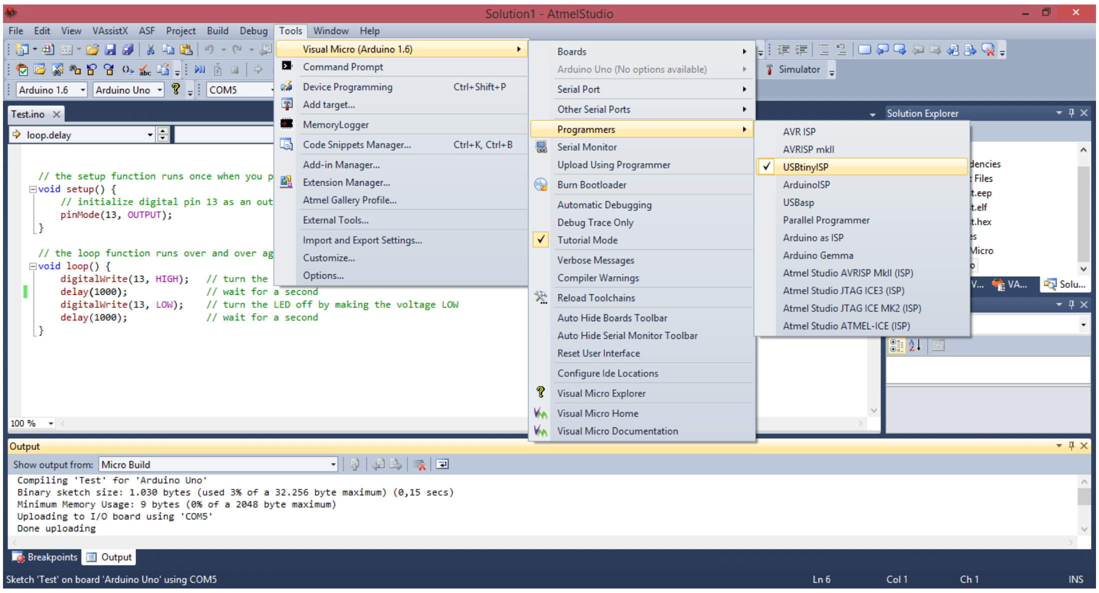
```
Compiling 'Test' for 'Arduino Uno'
Binary sketch size: 1.030 bytes (used 3% of a 32.256 byte maximum) (0,15 secs)
Minimum Memory Usage: 9 bytes (0% of a 2048 byte maximum)
Uploading to I/O board using 'COM5'
Done uploading
```

Sketch 'Test' on board 'Arduino Uno' using COM5

Figure A.21 – *Uploading the Arduino Code in Atmel Studio (8)*

Another way to upload the code is by ICSP programming. To do this, you will need a programmer, in this case, I am using the USBtinyISP, a pen-source USB AVR programmer and SPI interface designed and distributed by Adafruit [3].

- We need to prepare the Atmel Studio for programming by a programmer, on the one hand selecting the programmer used (see Figure A.22) and the other hand activating the option *Upload Using Programmer*, as you can see in the figure A.23. Then will appear the blue message in the bottom of the window remarking the activation of this way to program.

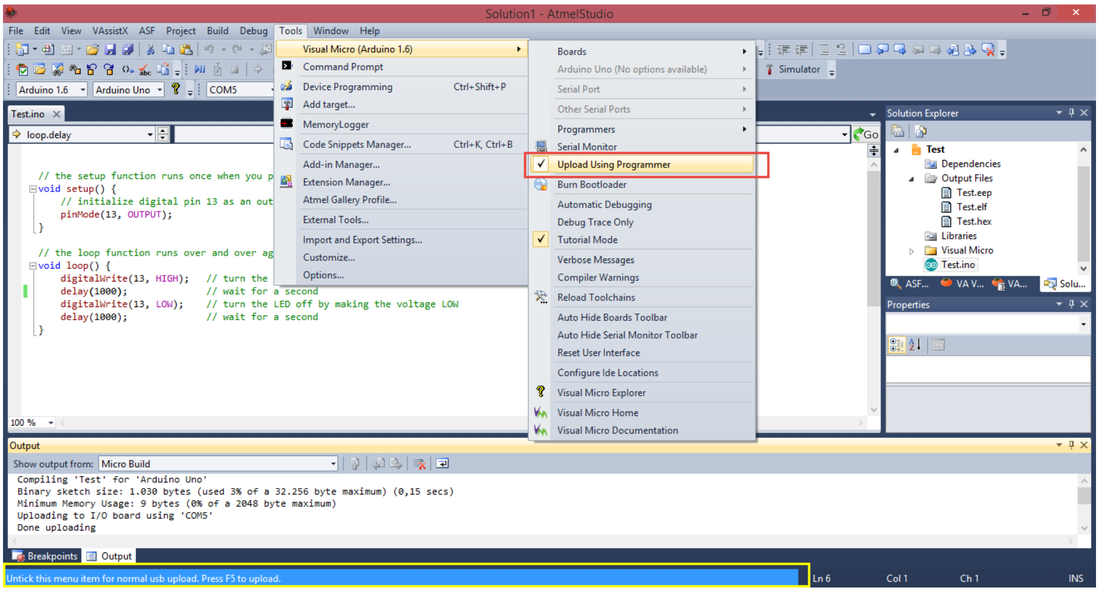


```
// the setup function runs once when you p
void setup() {
    // initialize digital pin 13 as an out
    pinMode(13, OUTPUT);
}

// the loop function runs over and over ag
void loop() {
    digitalWrite(13, HIGH);   // turn the
    delay(1000);              // wait for a second
    digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
    delay(1000);              // wait for a second
}
```

```
Compiling 'Test' for 'Arduino Uno'
Binary sketch size: 1.030 bytes (used 3% of a 32.256 byte maximum) (0,15 secs)
Minimum Memory Usage: 9 bytes (0% of a 2048 byte maximum)
Uploading to I/O board using 'COM5'
Done uploading
```

Sketch 'Test' on board 'Arduino Uno' using COM5

Figure A.22 – *Uploading the Arduino Code in Atmel Studio (9)*



```
// the setup function runs once when you p
void setup() {
    // initialize digital pin 13 as an out
    pinMode(13, OUTPUT);
}

// the loop function runs over and over ag
void loop() {
    digitalWrite(13, HIGH);   // turn the
    delay(1000);              // wait for a second
    digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
    delay(1000);              // wait for a second
}
```

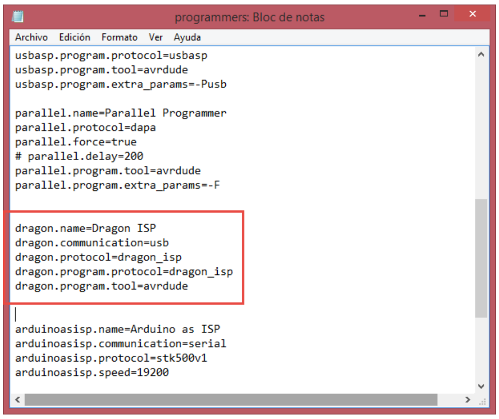
```
Compiling 'Test' for 'Arduino Uno'
Binary sketch size: 1.030 bytes (used 3% of a 32.256 byte maximum) (0,15 secs)
Minimum Memory Usage: 9 bytes (0% of a 2048 byte maximum)
Uploading to I/O board using 'COM5'
Done uploading
```

Untick this menu item for normal usb upload. Press F5 to upload.

Figure A.23 – *Uploading the Arduino Code in Atmel Studio (10)*

- On the other hand, we can upload the code in an Arduino by ICSP is with the AVR Dragon board. Is an Atmel board that allow us upload and debug (if you select the option) the Arduino code while is working on our Arduino UNO.

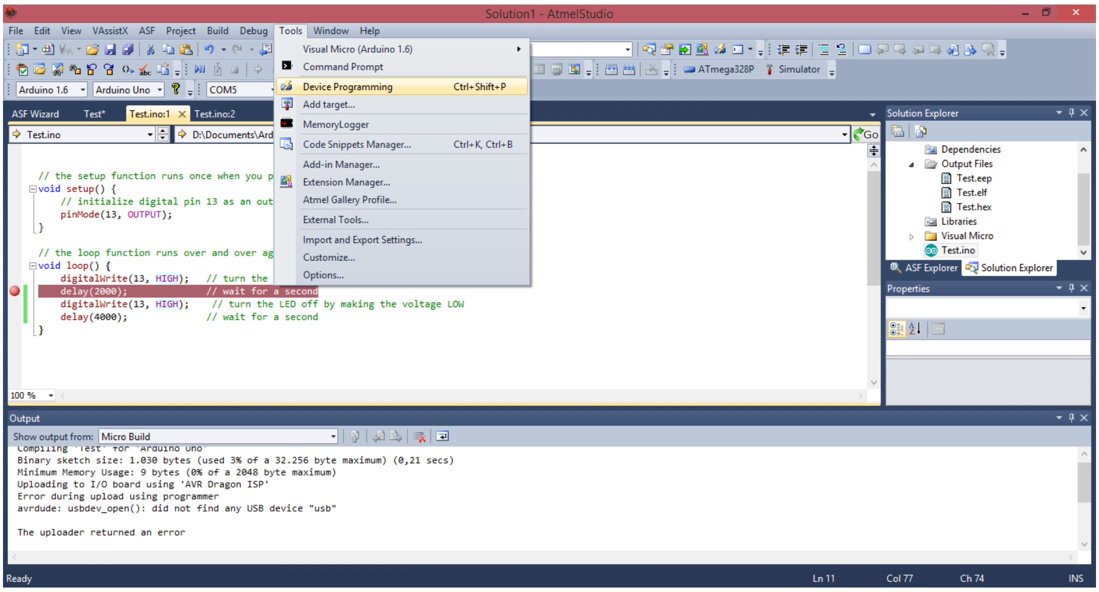
- In this case, we need to select *Upload Using Programmer* again, but now it will be necessary the option for the new programmer (AVR Dragon). Here appears a problem, Arduino IDE is not prepared for the use of this programmer, so we need to change the file *programmers.txt* that you can find in the folder: *C:\Program Files (x86)\Arduino\hardware\Arduino\avr*. In order to change the file, the best option is to copy the file to a folder, for example Desktop, add the lines that you can see in the figure A.24, save the file and copy it again into the default folder. Once you have done this, you will see the textitDragon ISP device as a programmer when you restart Atmel studio.



```
usbasp.program.protocol=usbasp
usbasp.program.tool=avrdude
usbasp.program.extra_params=-Pusb

parallel.name=Parallel Programmer
parallel.protocol=dapa
parallel.force=true
# parallel.delay=200
parallel.program.tool=avrdude
parallel.program.extra_params=-F


dragon.name=Dragon ISP
dragon.communication=usb
dragon.protocol=dragon_isp
dragon.program.protocol=dragon_isp
dragon.program.tool=avrdude


arduinoasisp.name=Arduino as ISP
arduinoasisp.communication=serial
arduinoasisp.protocol=stk500v1
arduinoasisp.speed=19200
```

Figure A.24 – *Uploading the Arduino Code in Atmel Studio (11)*

- In Atmel Studio, the first configuration that you have to do is the AVR Dragon configuration. It can be found in the menu *Tools/Device Programming* (see Figure A.25).



```
// the setup function runs once when you p
void setup() {
    // initialize digital pin 13 as an out
    pinMode(13, OUTPUT);
}

// the loop function runs over and over ag
void loop() {
    digitalWrite(13, HIGH);   // turn the
    delay(2000);              // wait for a second
    digitalWrite(13, HIGH);    // turn the LED off by making the voltage LOW
    delay(4000);              // wait for a second
}
```

```
Compiling 'Test' for 'Arduino Uno'
Binary sketch size: 1.030 bytes (used 3% of a 32.256 byte maximum) (0,21 secs)
Minimum Memory Usage: 9 bytes (0% of a 2048 byte maximum)
Uploading to I/O board using 'AVR Dragon ISP'
Error during upload using programmer
avrdude: usbdev_open(): did not find any USB device "usb"

The uploader returned an error
```

Figure A.25 – *Uploading the Arduino Code in Atmel Studio (12)*

– Once you are in the window to configure it, you need to select the tool (programmer), the device (the board that you are going to program), the Interface (ISP in my case) and press buttom *Apply* (see Figures A.26 and A.27).

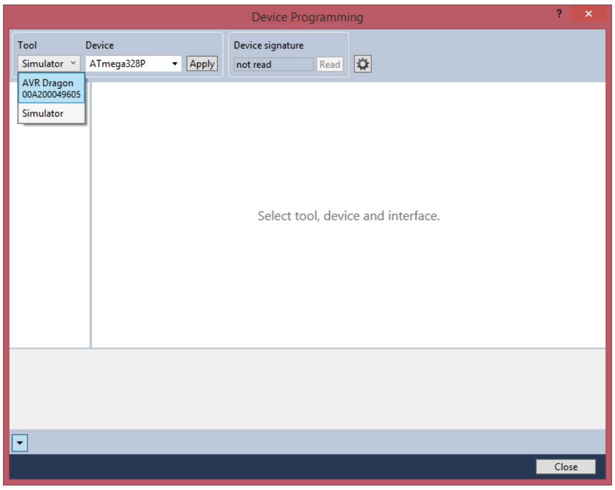
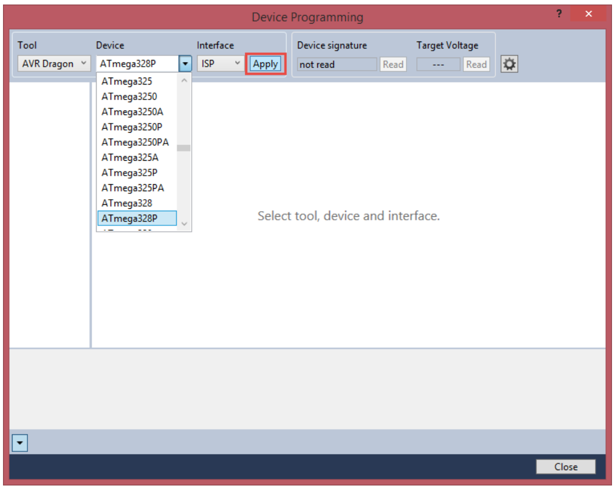


Figure A.26 – *Uploading the Arduino Code in Atmel Studio (13)*



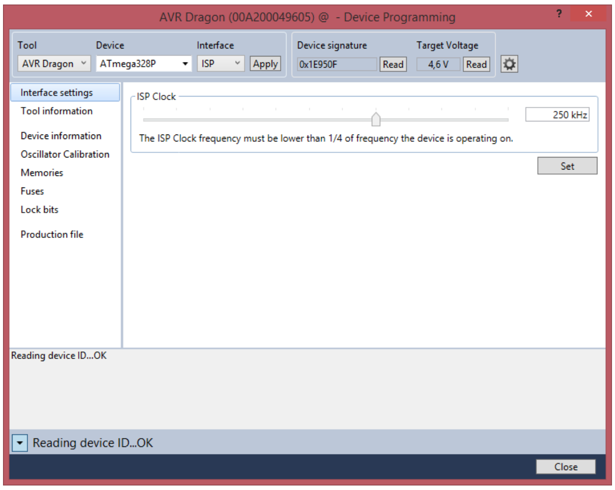
Tool AVR Dragon
Device ATmega328P
Interface ISP
Device signature not read
Target Voltage ---

Select tool, device and interface.

Figure A.27 – *Uploading the Arduino Code in Atmel Studio (14)*

– Secondly, you can see if the connection is correct reading the *Device Signature* and the *Target Voltage*.

– To have a correct operation, copy the configuration shown in figures A.28, A.29, A.30 and A.31.



Tool AVR Dragon
Device ATmega328P
Interface ISP
Device signature 0x1E950F
Target Voltage 4,6 V

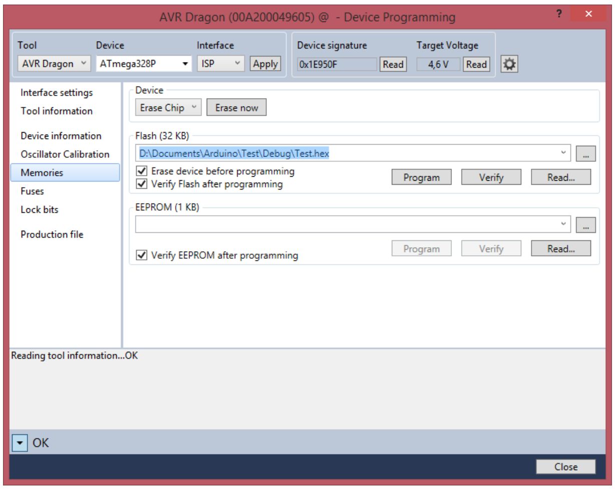
ISP Clock 250 kHz

The ISP Clock frequency must be lower than 1/4 of frequency the device is operating on.

Reading device ID...OK

Reading device ID...OK

Figure A.28 – *Uploading the Arduino Code in Atmel Studio (15)*



AVR Dragon (00A200049605) @ - Device Programming

Tool: AVR Dragon
Device: ATmega328P
Interface: ISP
Device signature: 0x1E950F
Target Voltage: 4,6 V

Device: Erase Chip

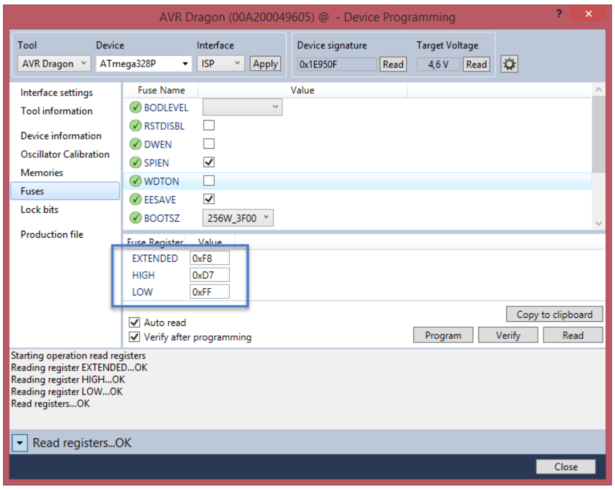
Flash (32 KB): D:\Documents\Arduino\Test\Debug\Test.hex
☑ Erase device before programming
☑ Verify Flash after programming

EEPROM (1 KB)
☑ Verify EEPROM after programming

Reading tool information...OK

OK

Figure A.29 – *Uploading the Arduino Code in Atmel Studio (16)*



AVR Dragon (00A200049605) @ - Device Programming

Tool: AVR Dragon
Device: ATmega328P
Interface: ISP
Device signature: 0x1E950F
Target Voltage: 4,6 V

Fuse Name	Value
BODLEVEL	
RSTDISBL	☐
DWEN	☐
SPIEN	☑
WDTON	☐
EESAVE	☑
BOOTSZ	256W_3F00

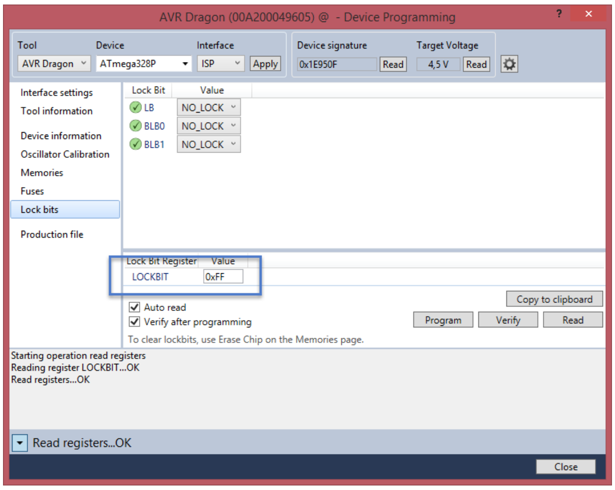
Fuse Register	Value
EXTENDED	0xF8
HIGH	0xD7
LOW	0xFF

☑ Auto read
☑ Verify after programming

```
Starting operation read registers
Reading register EXTENDED...OK
Reading register HIGH...OK
Reading register LOW...OK
Read registers...OK
```

Read registers...OK

Figure A.30 – *Uploading the Arduino Code in Atmel Studio (17)*



Tool: AVR Dragon
Device: ATmega328P
Interface: ISP
Device signature: 0x1E950F
Target Voltage: 4,5 V

Lock Bit	Value
LB	NO_LOCK
BLB0	NO_LOCK
BLB1	NO_LOCK

Lock Bit Register	Value
LOCKBIT	0xFF

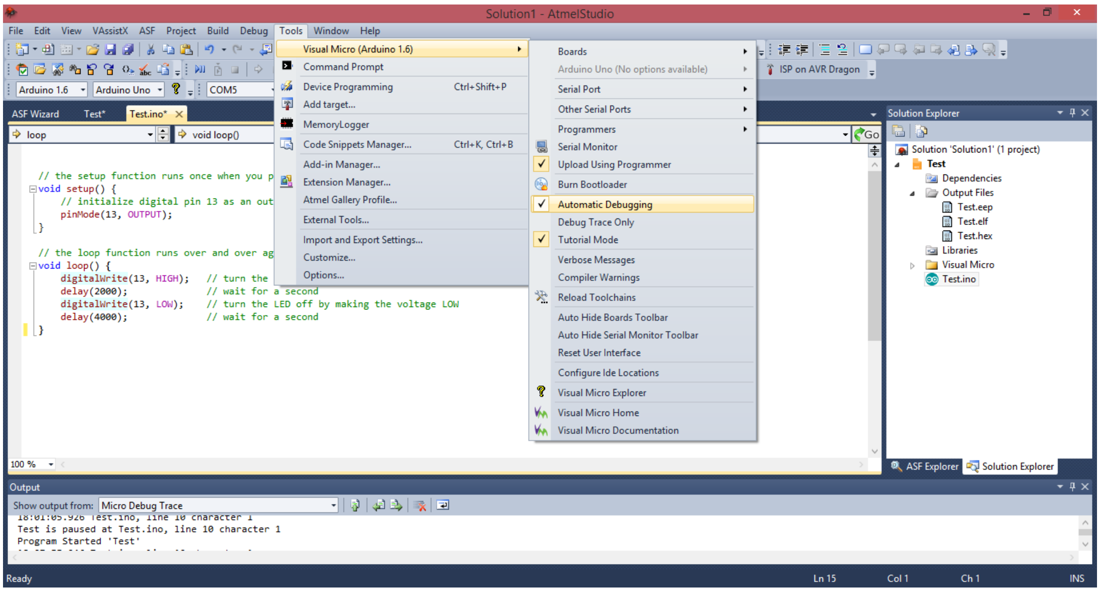
☑ Auto read
☑ Verify after programming
To clear lockbits, use Erase Chip on the Memories page.

```
Starting operation read registers
Reading register LOCKBIT...OK
Read registers...OK
```

Read registers...OK

Figure A.31 – *Uploading the Arduino Code in Atmel Studio (18)*

– When you finish the configuration, you have two options: compile and make a normal update or compile and make a debugging update of your code. If you want to debug it is necessary to select the option *Automatic Debugging*, shown in the figure A.32)



```
// the setup function runs once when you p
void setup() {
    // initialize digital pin 13 as an out
    pinMode(13, OUTPUT);
}

// the loop function runs over and over ag
void loop() {
    digitalWrite(13, HIGH);   // turn the
    delay(2000);              // wait for a second
    digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
    delay(4000);              // wait for a second
}
```

```
18:01:05.926 Test.ino, line 10 character 1
Test is paused at Test.ino, line 10 character 1
Program Started 'Test'
```

Figure A.32 – *Uploading the Arduino Code in Atmel Studio (19)*

APPENDIX

B

3D PRINTING

In this appendix, there will be a summary about the use of the 3D printers on this Final Project Degree. First, an introduction will be taken of the project *RepRap* (Replicating Rapid-prototyper), secondly, we will see the main models that we have used, and finnaly, the improvements hat I have made for the printers of the ECTD and the printing process that we have taken.

B.1 Introduction to ***RepRap***

RepRap is a project which was invented by Adrian Bowyer, a British engineer and mathematician. This project can be described as: open-source self-replicating 3Dprinter [24] [61].

Figure B.1 – *Logo RepRap*

This kind of 3D printer build the models in layers of plastic (ABS or PLA). Depending on the height of this layer, the quality of the 3d model will change, more height of each layer, less quality, and vice versa.

One of the most important aspects of this new way to prototype the 3Dmodels, is the low cost that you will have to face. A 3Dprinter of RepRap, costs less than 500€, while the commercial printers usually costs thousands of euros. So, this issue allow to small companies and individual users, buy one of them.

Furthermore, the RepRap movement follow the ideas of Open Source devices (with the GNU General Public License. For example: "if you have a RepRap machine, you can use it to make another and give that one to a friend". [24]

The first RepRap printer was in february, 2008, and it was called Darwin (because of the *Theory of Evolution*). The printers have been developed and improved further in recent years.

B.2 Printers used in this project

In this project, we have used two models of RepRap machines that the ECTD had: Prusa Mendel i2 and Prusa Mendel i3. In order to have a better quality of printing, we request some quotes to the CITIC from UGR and Oritia Boreas and the prices were:

Company	Price (€)
CITIC (only material)	382
Oritia y Boreas	595
Faculty of Science printers	20 aprox

Table B.1 – *Quotes requested for 3D printing*

So the decision was print the Testbed at the Faculty of Science.

B.2.1 Prusa Mendel i2

The Prusa Mendel i2 (is the second iteration of Prusa Mendel) was launched in 2011, and is preceded by a third iteration (Prusa Mendel i3). The mainly characteristics are shown in the Table B.2.

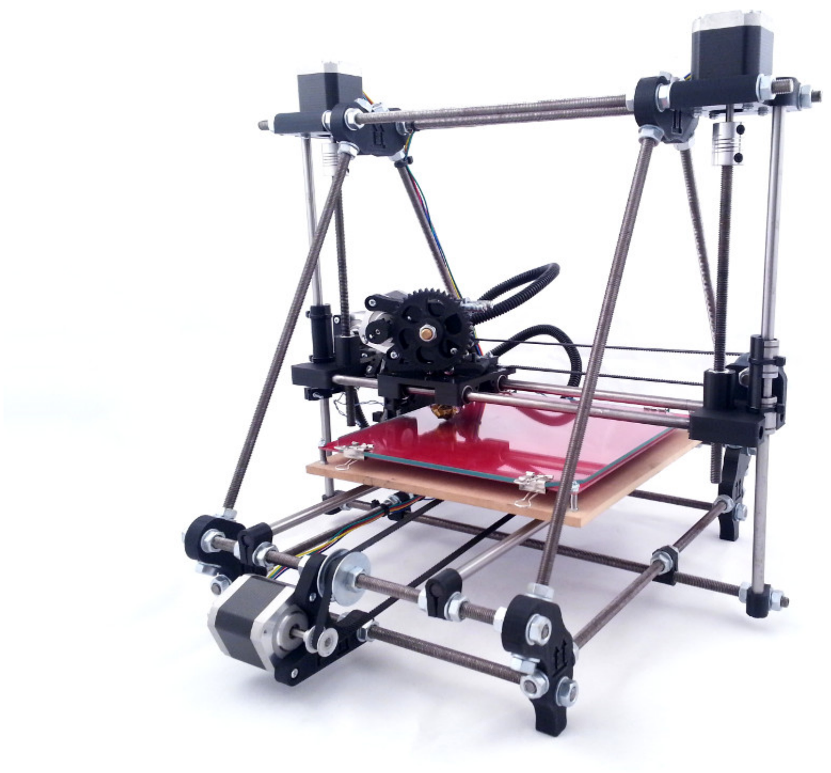


Figure B.2 – *Prusa Mendel i2*

Max printing volume(mm)	120x120x120
Most common materials	PLA,ABS
Fabrication technology	FFF
Extruder	Single
Precision (mm)	0.1
Thickness layer (mm)	0.3-0.5
Communication interface	USB
Machine dimension (mm)	440x470x370
Weight aprox. (kg)	7

Table B.2 – *Prusa Mendel i2 Specifications*

B.2.2 Prusa Mendel i3

The Prusa Mendel i3, is the following model of Prusa Mendel i2, and it has a smaller quantity of components and parts, so it is easier to assembly than Prusa Mendel i2. In addition to its simplicity, it has an enhanced frame rigidity and this also improve the whole machine stability.

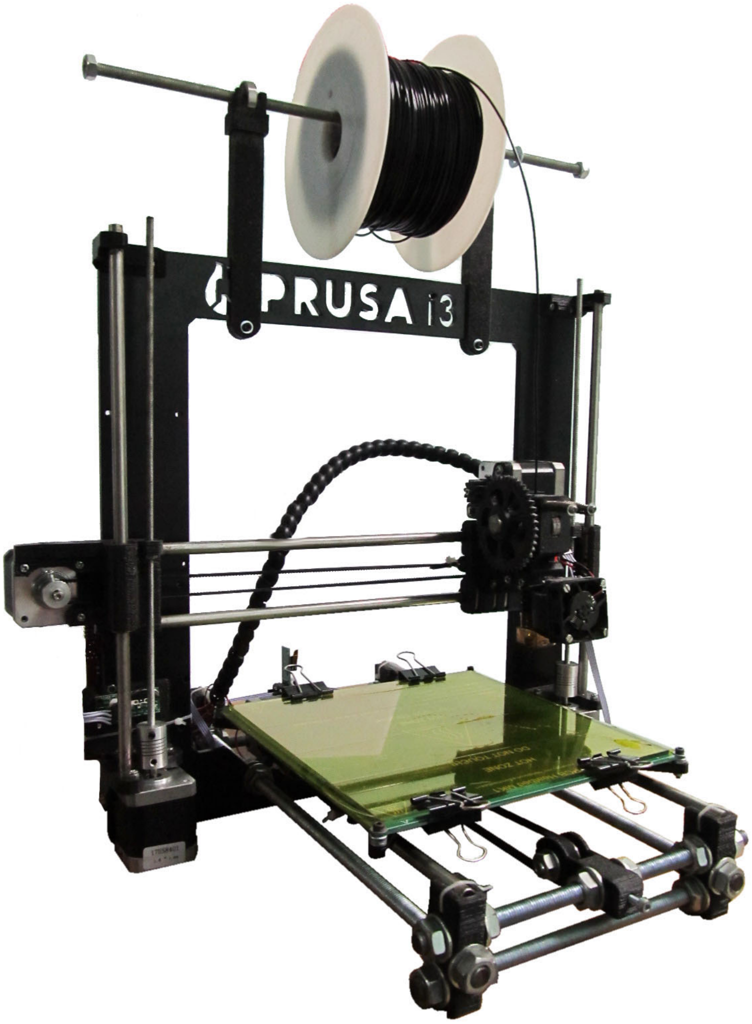


Figure B.3 – *Prusa Mendel i3*

Max printing volume(mm)	200x200x200
Most common materials	PLA,ABS HDPE
Fabrication technology	FFF
Extruder	Single
Precision (mm)	0.1
Thickness layer (mm)	0.3-0.5
Communication interface	USB
Machine dimension (mm)	460x510x370
Weight aprox. (kg)	7

Table B.3 – *Prusa Mendel i3 Specifications*

B.3 3D printing improvements

In order to have a better operation and quality for the parts that we need to build, we have made some improvements. For example, add two relays to control the heatbed, and connect LCDs to control the printer without the PC connection.

B.3.1 Relay configuration

The pieces that we had to build in the printer were huge, so the building process need a special operation because of its size. The PrusaMendel i2 that we had in the ECTD had a problem of overheating of the RAMPS (the Arduino's shield that controls the printer) when we build the Testbed part, because the heatbed was turned on too much time. (see figures B.4 and B.5).

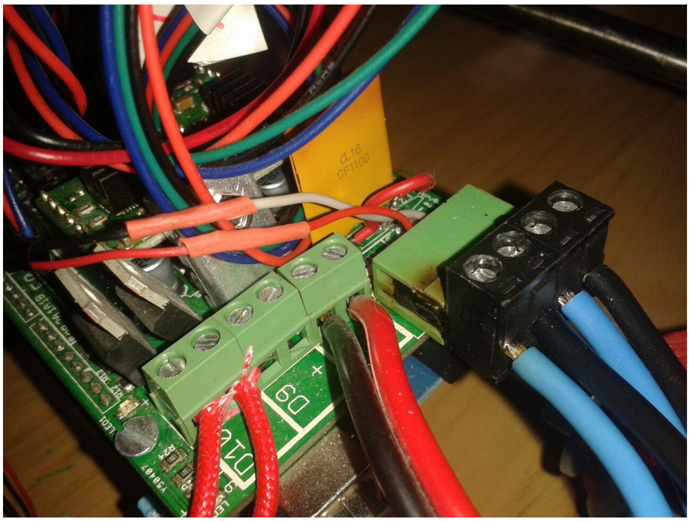


Figure B.4 – *Overheating of the RAMPS*

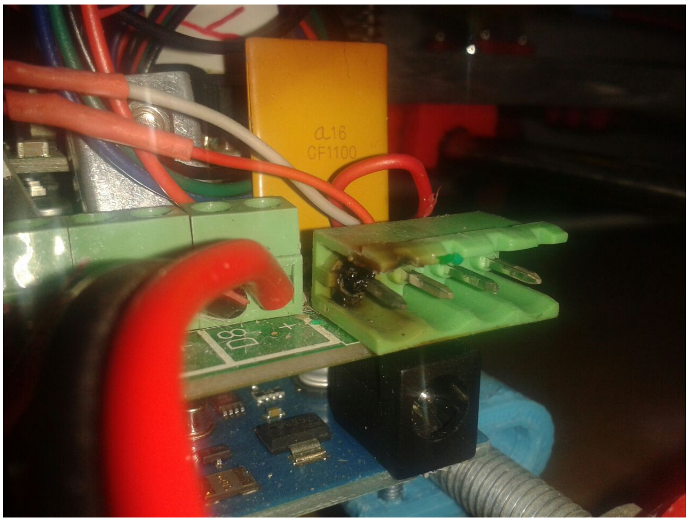


Figure B.5 – *Overheating of the RAMPS (2)*

A good solution was placing a relay between the power source and the heatbed, controlled by the RAMPS, in order to avoid the high current flow through the RAMPS. The relays used were the GOODSKY RWHSH112D (figure B.6), because its availability in the laboratory and the compatibility of the specifications that shows the table B.4.

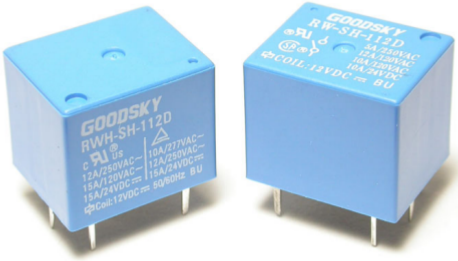


Figure B.6 – *Goodsky Relays used for the solution*

Nominal Coil Voltage (VDC)	12
Contact Capacity	15A to 15VDC
Minimun Switching Load	DC5V, 15mA
Contact Material	AgSnO2

Table B.4 – *GOODSKY RWHSH112D Specifications*

The configuration designed for this solution was the following schematic of the figure B.7:

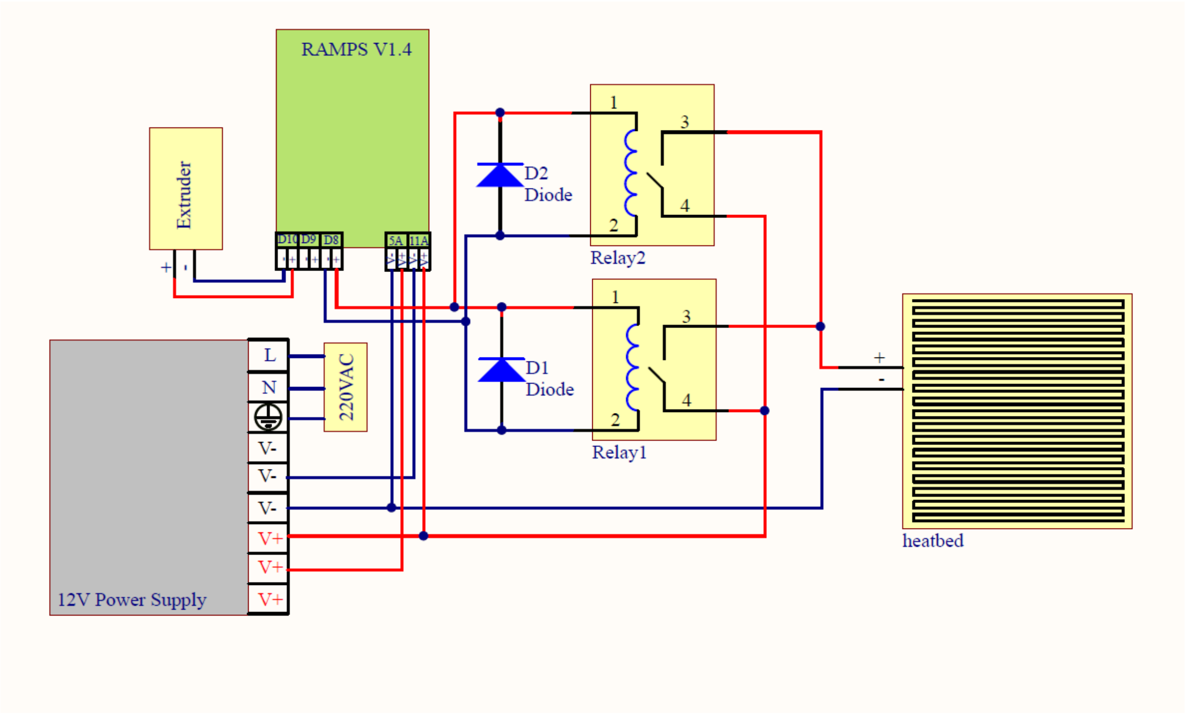


Figure B.7 – *Schematic for configuration Relays in Prusa Mendel i2*

When this improvement was taken, the biggest parts were build in a perfect way and the quality of these pieces was enhanced.

B.3.2 Adding the LCD control to the printers

One of the best operational improvements that you can do in your 3D printer is allow its control, rather than a PC by USB, control it with a LCD to watch and a rotary to select the different options. In other words, with the LCD you will be able to have the printer in any place without the need of a PC. The LCD that we used was the RepRap original version LCD Smart Controller. The figure B.8. This device has a special board and a converter to connect it to the RAMPS (and Arduino Mega) of our Prusa Mendel Printer.

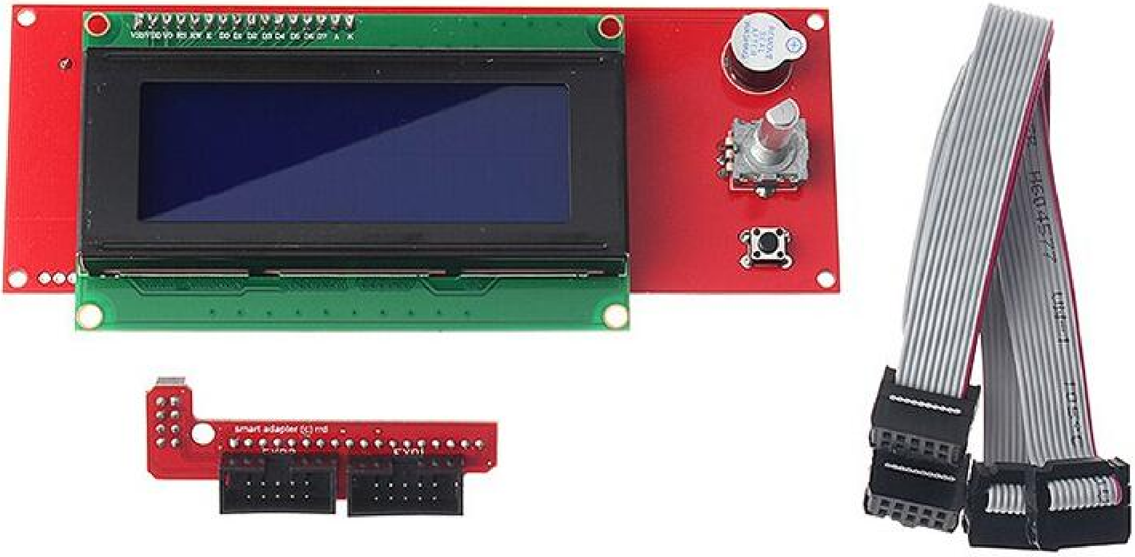


Figure B.8 – *LCD Smart Controller*

This advance can control the following functions:

- Control heatbed and hotend temperature.
- Control the four motors that the printer has (X, y and Z axis, and extruder motor).
- Prepare the printer for print the model.
- Configure the parameters while you are printing and before the printing process.

In the figure B.9 there is the Official LCD Menu Tree (for Marlin firmware). You can find that tree in the Github repository of Marlin Firmware [22].

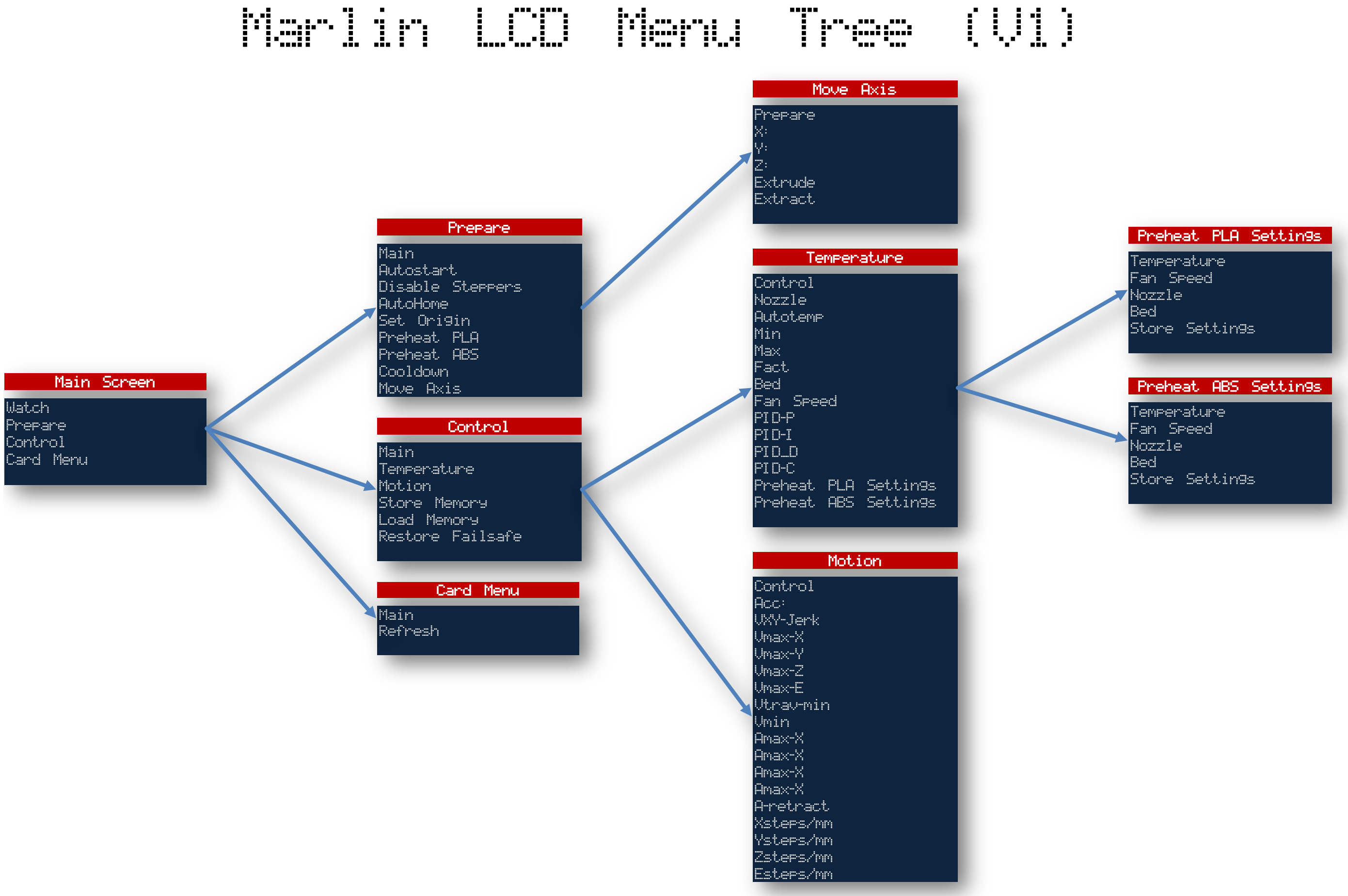


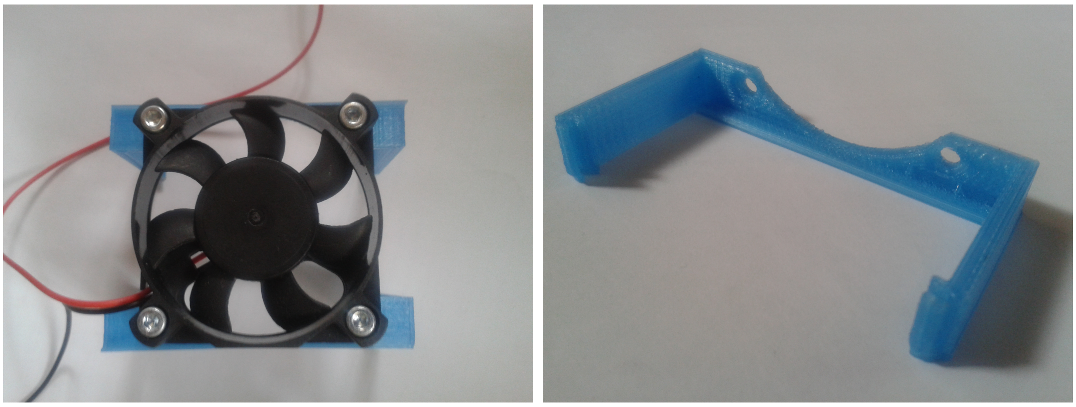
Figure B.9 – *LCD Menu Tree*

B.3.3 Adding the fan for the electronics

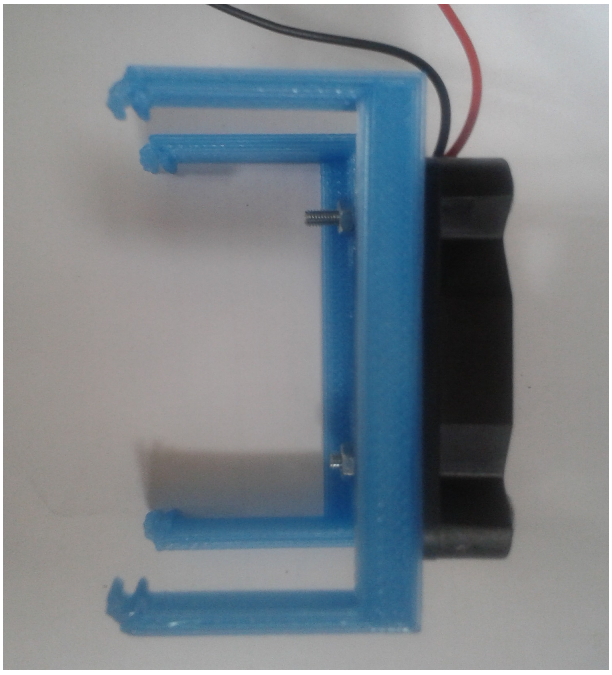
Another very common problem in a 3D printer is the overheating of the driver for the motors. To avoid this problem (cooling this part of the circuit), a fan can be installed in front of the RAMPS and the heat dissipation will be improved.

To install the fan, I designed in SolidWorks the support, and finnally, this parts were printed in our 3Dp printer to place the fan with 4 screws.

This 2 parts were designed conformed to the dimensions of our RAMPS and the fan selected for this use (see figure B.10), recycled from a old personal computer donated by the ECTD.



(a) *Fan and support (1)* **(b)** *Support 3D printed*



(c) *Fan and support (2)*

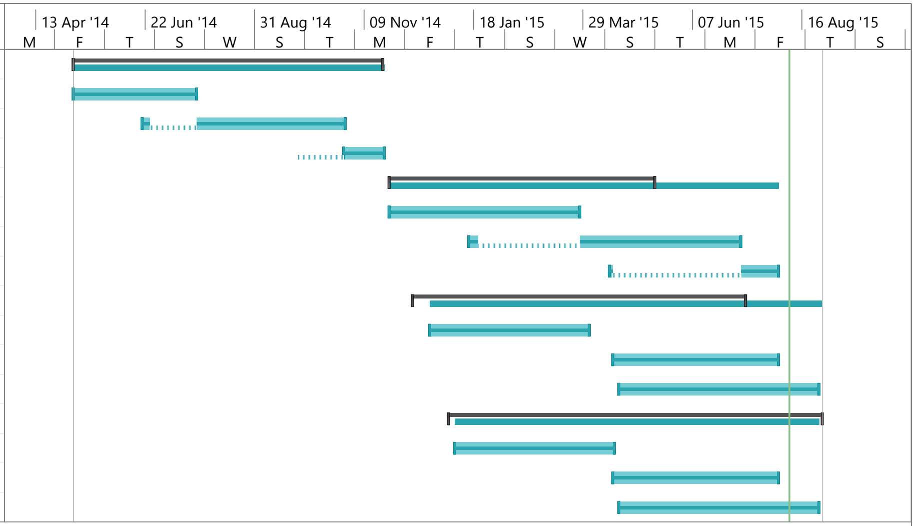
Figure B.10 – *Installed fan and support parts*

APPENDIX

C

GANTT DIAGRAM

ID	ℹ	Task Mode	Task Name	Duration	Start	Finish
1	✓ 📝	📌	**Study and evaluation of the system**	**142 days**	**Wed 07/05/14**	**Thu 20/11/14**
2	✓	📌	Study other equivalent systems	57 days	Wed 07/05/14	Thu 24/07/14
3	✓	📌	Research about this technology	70 days	Fri 20/06/14	Mon 27/10/14
4	✓ 📝	📌	Study the mechanical background	20 days	Sun 28/09/14	Fri 21/11/14
5	✓	📌	**System Design**	**122 days**	**Tue 25/11/14**	**Wed 13/05/15**
6	✓	📌	Mechanical Design	88 days	Tue 25/11/14	Thu 26/03/15
7	✓	📌	Electronic Design	77 days	Thu 15/01/15	Tue 07/07/15
8	✓	📌	Software Design	20 days	Wed 15/04/15	Fri 31/07/15
9	✓	📌	**System Implementation**	**153 days**	**Wed 10/12/14**	**Fri 10/07/15**
10	✓ 📝	📌	Mechanical implementation	74 days	Sun 21/12/14	Wed 01/04/15
11	✓	📌	Electronic Implementation	76 days	Fri 17/04/15	Fri 31/07/15
12	✓	📌	Software Implementation	92 days	Tue 21/04/15	Wed 26/08/15
13	✓	📌	**Integration, Test and Validation of the System**	**171 days**	**Fri 02/01/15**	**Fri 28/08/15**
14	✓	📌	Mechanical implementation	74 days	Tue 06/01/15	Fri 17/04/15
15	✓	📌	Electronic Implementation	76 days	Fri 17/04/15	Fri 31/07/15
16	✓	📌	Software Implementation	92 days	Tue 21/04/15	Wed 26/08/15



Project: Simple project plan
Date: Sat 08/08/15

Task	▬	Project Summary	⊓	Manual Task	▬	Start-only	[	Deadline	⬇
Split	┈	Inactive Task		Duration-only	▬	Finish-only	]	Progress	─
Milestone	◆	Inactive Milestone	◇	Manual Summary Rollup	▬	External Tasks	▬	Manual Progress	─
Summary	⊓	Inactive Summary	⊓	Manual Summary	⊓	External Milestone	◆		

APPENDIX

D

TEST TABLES

D.1 Power Supply - RPM Test Tables

	Vcc(V)	I(mA)	ω_1(RPM)	ω_2(RPM)	ω_3(RPM)	ω_{ave}(RPM)
1	1.054	15.24	1055	1056	1059	1056.66
2	2.066	19.12	2306	2304	2302	2304
3	3.012	24.69	3427	3427	3427	3427
4	4.001	32.00	4621	4620	4623	4621.33
5	5.010	40.30	5721	5726	5729	5725.33
6	6.000	50.40	6718	6713	6713	6714.66

Table D.1 – *VCC-RPM test for Kysan RF-300CH*

	Vcc(V)	I(mA)	ω_1(RPM)	ω_2(RPM)	ω_3(RPM)	ω_{ave}(RPM)
1	1.028	12.69	936.2	940.7	935.5	937.47
2	2.025	15.49	2034	2031	2032	2032.33
3	3.007	18.97	3102	3103	3105	3103.33
4	4.040	24.04	4250	4252	4253	4251.67
5	5.010	29.95	5255	5262	5262	5259.67
6	6.000	38.14	6226	6232	6235	6231

Table D.2 – *VCC-RPM test for no-model motor*

	Vcc(V)	I(mA)	ω_1(RPM)	ω_2(RPM)	ω_3(RPM)	ω_{ave}(RPM)
1	1.019	17.66	1009	1003	1010	1007.33
2	2.014	20.42	2194	2202	2205	2200.33
3	3.004	25.36	3337	3338	3334	3336.33
4	4.020	34.12	4482	4480	4491	4484.33
5	5.010	40.50	5407	5403	5412	5407.33
6	6.001	47.40	6490	6498	6499	6495.67

Table D.3 – *VCC-RPM test for Minebea MDN3BT*

	Vcc(V)	I(mA)	ω_1(RPM)	ω_2(RPM)	ω_3(RPM)	ω_{ave}(RPM)
1	1.015	14.3	1026	1029	1027	1027.33
2	2.014	15.5	2141	2136	2140	2139
3	2.998	17.0	3229	3230	3231	3230
4	4.000	18.6	4367	4368	4371	4368.67
5	5.040	20.6	5462	5454	5457	5457.67
6	6.020	23.6	6553	6548	6546	6549
7	7.050	27.6	7578	7581	7578	7579

Table D.4 – *VCC-RPM test for Mitsumi M25E-4 (not used)*

APPENDIX

E

PROJECT BUDGET

E.1 Electronics costs

For the electronics implementation, we have used the financial contribution of our main sponsor, the ECTD, which paid us the electronics devices shown in the table E.3. Furthermore, the PCB, has been build in the machine LPFK, which belong to the ECTD too. That PCB had a cost of fabrication of 40€, whose breakdown is shown in table E.1.

Item	**Cost(€)**
Copper Plate	10€
Human Resources	25€
Machine working	5€
TOTAL	**40€**

Table E.1 – *PCB building cost*

Therefore, the total cost of the PCB implementation is broken down in the table E.2

Item	**Cost(€)**
PCB Build	40€
Electronic devices	97.46
TOTAL	**137.46€**

Table E.2 – *Total cost of the PCB implementation*

Table E.3 – *Budget for the electronics devices on PCB*

Item	Description	Bought items(Used)	Cost/unit(€)	Bought Cost(€)	Used cost(€)
Microcontroller	ATMEGA328P-AU	2(1)	2.55	5.1	2.55
Buffer driver	SN74LVC1G125	3(1)	0.498	1.494	0.498
Op. Amplifier	LMV358IDGKR	3(1)	0.832	2.496	0.832
Rectifier diode	S1M	14(12)	0.257	3.598	3.084
Push button	B3S-1000P	5(1)	0.353	1.765	0.353
Terminal PCB	3.81mm	5(3)	0.952	4.76	2.856
Switch	SS22SDP2	5(3)	2.05	10.25	6.15
Red LED	LSR976	4(1)	0.271	1.084	0.271
Green LED	2012CGCK	4(3)	0.106	0.424	0.318
Yellow LED	2012SYCK	5(2)	0.199	0.995	0.398
Xbee Module	XB24-AWI-001	1(1)	20.24	20.24	20.24
Crystal	16MHz	1(1)	0.77	0.77	0.77
Driver Motor	L298P	3(2)	6.02	18.06	12.04
Capacitor	22pF 0603 50V	2(2)	0.0071	0.0142	0.0142
Capacitor	4.7uF 0603 16V	5(2)	0.0382	0.191	0.0764
Capacitor	220uF 2917 16V	1(1)	3.24	3.24	3.24
Capacitor	47pF 0603 16V	1(1)	0.0243	0.0243	0.0243
Capacitor	1.5nF 0603 16V	1(1)	0.0133	0.0133	0.0133
Capacitor	2.2uF 0805 6.3V	1(1)	0.0752	0.0752	0.0752
Capacitor	10uF 1206 16V	1(1)	0.504	0.504	0.504
Capacitor	100pF 0805 50V	1(1)	0.0382	0.0382	0.0382
Capacitor	10nF 0805 25V	3(2)	0.237	0.711	0.474
Capacitor	100nF 0805 100V	2(2)	0.128	0.256	0.256
Resistor	0805 4.7kΩ	2(2)	0.0305	0.061	0.061
Resistor	0805 200Ω	2(2)	0.0393	0.0786	0.0786
Resistor	0805 1MΩ	1(1)	0.0948	0.0948	0.0948
Resistor	0805 10kΩ	2(2)	0.021	0.042	0.042
Resistor	0805 2.2Ω	3(3)	0.02	0.06	0.06

Continued on next page

Table E.3 – continued from previous page

Item	Description	Bought items(Used)	Cost/unit(€)	Bought Cost(€)	Used cost(€)
Resistor	0805 267kΩ	2(2)	0.0121	0.0242	0.0242
Resistor	0805 160kΩ	1(1)	0.013	0.013	0.013
Resistor	0805 100kΩ	2(2)	0.0289	0.0578	0.0578
Resistor	0805 200kΩ	4(4)	0.0289	0.1156	0.1156
Resistor	0805 178kΩ	1(1)	0.0719	0.0719	0.0719
Resistor	0805 470kΩ	1(1)	0.0181	0.0181	0.0181
Resistor	0805 340kΩ	1(1)	0.471	0.471	0.471
Resistor	0805 60.4kΩ	1(1)	0.017	0.017	0.017
Resistor	0805 15kΩ	1(1)	0.039	0.039	0.039
Resistor	0805 0.1Ω	1(1)	0.395	0.395	0.395
Resistor	0805 56Ω	1(1)	0.018	0.018	0.018
Resistor	0805 68Ω	2(2)	0.0132	0.0264	0.0264
Resistor	0805 75Ω	1(1)	0.015	0.015	0.015
Inductor	22uH SMD	2(2)	0.26	0.52	0.52
Inductor	10uH SMD	1(1)	0.258	0.258	0.258
Voltage Detector	MAX8211	2(2)	6.16	12.32	12.32
DC-DC Converter	MAX1771	1(1)	7.66	7.66	7.66
DC-DC Converter	LTC3440	1(1)	6.92	6.92	6.92
Diode Zener	BZX84C4V3	2(2)	0.829	1.658	1.658
Sensor Accel+Gyros	MPU6050	1(1)	2.67	2.67	2.67
Sensor Accel+Magn	LSM303DLHC	1(1)	4.76	4.76	4.76
LCD	16x02 I2C	1(1)	4	4	4
				TOTAL	**97.46**

E.2 Mechanics

In the mechanical implementation of this Testbed was the ECTD the main sponsor too, because the department supply us with 2 3D printers, Prusa i2 and i3, and the PLA and ABS was donated by the department too. If we estimate the cost, the breakdown of the mechanical implementation budget is shown in table E.4.

Item	**Cost(€)**
Electrical cost	0.21€/kWh * X kW*10h = Y €
ABS	4€
TOTAL	**Z €**

Table E.4 – *Total cost of the mechanical implementation*

E.3 Software

In the following table is shown how the licenses of the software were adquired:

Software	**Owner of the license**	**Cost(€)**
Altium Designer 14.3	GranaSAT	Free (sponsorship)
SolidWorks 2014	GranaSAT	Free (sponsorship)
KISSlicer	GranaSAT	Free (sponsorship)
Matlab	ECTD	Free
Microsoft Visio 2013	UGR	Free (DreamSpark)
Microsoft Project 2013	UGR	Free (DreamSpark)
Arduino IDE	Víctor Burgos	Free license
AtmelStudio 6.2	Víctor Burgos	Free license
CURA	Víctor Burgos	Free license
TeXnicCenter	Víctor Burgos	Free license
Miktex	Víctor Burgos	Free license
KeySight BenchVue	Víctor Burgos	Free license
SumatraPDF	Víctor Burgos	Free license
	TOTAL	0

Table E.5 – *Software cost*

E.4 Human Resources

In this project, in every stage, a junior engineer was working during a year and a half approximately. If we fix the salary for the junior engineer in Spain (about 10€/h), the total salary related to the human resources in every stage is:

Stage	**Hours**	**Cost(€)**
Study and evaluation of the system	400	4000
System Design	700	7000
System Implementation	400	4000
Integration, test and validation of the system	400	4000
TOTAL	**1900**	**19000**

Table E.6 – *Human resources cost*

ACRONYMS

3D Three dimensions. ix, xi, 11, 30, 32, 142

ABS Acrylonitrile Butadiene Styrene. xi, 20, 22, 23, 41, 83, 127

ADCS Attitude Determination and Control System. xviii, xxiii, xxvi, 3, 5, 7, 10, 11, 18, 20, 25, 27, 29, 31, 33, 35, 47, 95

ADS Attitude Determination System. 2

CD Compact Disc. 35

CIC Centro de Instrumentación Científica. 87

COR Center of Rotation. 25

DC Direct Current. xxvi, xxvii, 35, 102, 103, 106, 107, 118, 119

DLR Deutsches Zentrum für Luft- und Raumfahrt. 2

DOF Degrees of Freedom. 13, 24–26, 31

ECTD Electronics and Computer Technology Department. 19, 20, 24, 30, 35, 37, 41, 63, 77, 141, 142

EMF Electromotive Force. 103, 118

EPS Electrical Power System. 142

ESA European Space Agency. ix, xi, 2

FFF Fused Filament Fabrication. 22, 23, 83

FOV Field of View. 28, 30

FRS Fixed Reference System. 12, 13, 15

GNU GNU is Not Unix. 20

GS Ground Station. xix, 18–20, 29, 45, 55, 89, 90, 92–94

HDPE High Density PolyEthylene. 23

IDE Integrated Development Environment. 4, 7, 8, 14, 41

LCD Liquid Crystal Display. xxi, 13, 27, 45, 92

LED Light-Emitting Diode. xxxi, 39, 44, 45, 55

LEO Low Earth Orbit. 30

MEMS Microelectromechanical Systems. xxiv, 31, 32

MOI Moment of Inertia. 14

MOSFET Metal-Oxide Semiconductor Field-Effect Transistor. 57

PC Personal Computer. 18, 20, 27, 35, 92–94, 98

PCB Printed Circuit Board. ix, xi, xviii, xix, xxvii, 13, 19, 20, 36, 37, 41, 43, 50–52, 56, 58, 60, 62–66, 71, 89, 92, 124–126, 133, 134, 142

PID Proportional Integral Derivative. xviii, 7, 36–40, 93, 94, 142

PLA Polylactic Acid. 20, 22, 23, 41

RepRap Replicating Rapid Prototyper. xi, xx, xxix, 19, 20, 27

SCH Schematic. 64

SMD Surface Mount Devices. 36, 43, 44, 56, 57, 64

SNSB Swedish National Space Board. 2

SRS Sphere Reference System. 12, 13, 15, 17

SSC Swedish Space Corporation. 2

TACT	Triaxial Attitude Control Testbed. 25
THT	Through-Hole Technology. 36, 64, 66
UGR	University of Granada. 3, 41, 87
USB	Universal Serial Bus. 27

GLOSSARY

Air-bearing Air-bearings are bearings that use a thin film of pressurized air to provide an exceedingly low friction load-bearing interface between surfaces [62].. xi, xx, xxiv, 3, 12, 13, 16–20, 23–27, 74, 77, 83, 132–134

Arduino Arduino is an open-source prototyping platform based on easy-to-use hardware and software. Arduino boards are able to read inputs and turn it into an output [11].. ix, xi, xx, xxviii, xxix, 1–18, 24, 27, 36, 41, 43, 44, 54, 90

Cubesat A Cubesat is a miniaturized satellite, which has the aim of a determined space research, with a volume of 10 cm cube, and a mass of less than 1.33 Kg [58].. ix, xi, xviii, xxiii, 1–4, 7, 9–12, 18, 19, 28, 30–34, 60, 66, 84, 85, 90, 141

GranaSAT GranaSAT is an academic project from the University of Granada consisting of the design and development of a picosatellite (Cubesat). Coordinated by the Professor Andrés María Roldán Aranda, GranaSAT is a multidisciplinary project with students from different degrees, where they can acquire and enlarge the necessary knowledge to front a real aerospace project. `http://granasat.ugr.es/`. ix, xi, xxiii, 1–3, 5, 7, 9, 12, 13, 18, 25, 41, 56, 58, 60, 141, 142

MATLAB MATLAB (matrix laboratory) is a multi-paradigm numerical computing environment and fourth-generation programming language [63].. ix, xi, 94, 118

PASS PASS means the success in the tests of each phase. If you do not have the PASS, you will not be allowed to continue with the project.. 3

Testbed A Testbed (also "test bed") is a platform for conducting rigorous, transparent, and replicable testing of scientific theories, computational tools, and new technologies [**?**].. ix, xi, xviii, xxiv–xxvi, 1–5, 7, 9–13, 18–20, 23–26, 30–34, 36, 41, 43–46, 50–52, 55, 56, 58, 59, 66, 75, 89, 90, 92, 94, 133, 134, 141, 142

REFERENCES

[1] Genesat-1. eoPortal Directory. https://directory.eoportal.org/web/eoportal/satellite-missions/g/genesat.

[2] 3D Printers for beginners. *3D Printers - How Do They Work?*, April 2014. :http://3dprintingforbeginners.com/wp-content/uploads/2014/04/3D-Printing-Technology$_{D}$*ownload.pdf*.

[3] Adafruit. Overview author gravatar image - avr programmer & spi interface. Learn Adafruit Webpage. https://learn.adafruit.com/usbtinyisp.

[4] Aerospace, B., and Corp., T. Ct-633 star tracker. http://www.ballaerospace.com/page.jsp?page=104.

[5] Arduino. Gy521 - mpu6050 schematic. Playground Arduino. http://playground.arduino.cc/uploads/Main/MPU6050-V1-SCH.jpg.

[6] Arduino. Mpu-6050 accelerometer + gyro information. Playground Arduino. http://playground.arduino.cc/Main/MPU-6050.

[7] ATmega32-AVR.com. Avr comparison. Website, May 1st 2012. http://atmega32-avr.com/avr-comparison/.

[8] Black, and Decker. Cp2525 air compressor. Products catalogue.

[9] Boynton, R. Using a spherical air bearing to simulate weightlessness. *55th Annual Conference of the Society of Allied Weight Engineers, Inc. at Atlanta, Georgia* (June 1996).

[10] By Lawrence A. Freeman, Michael C. Carpenter, D. O. R. J. P. R. R. U., and McLean, J. S. Use of submersible pressure transducers in water-resources investigations. Techniques of Water-Resources Investigations 8-A3. http://pubs.usgs.gov/twri/twri8a3/.

[11] CC, A. Arduino introduction. https://www.arduino.cc/en/guide/introduction.

[12] Corporation, M. M. M. *Datasheet Minebea MDN3.*

[13] Craig, K. Modeling the mems gyroscope. EDN Blogs - Mechatronics in Design, November 2012. http://www.edn.com/electronics-blogs/mechatronics-in-design/4400475/Modeling-the-MEMS-gyroscope.

[14] Crowell, C. W. Development and analysis of a small satellite attitude determination and control system testbed.

[15] Dachwald, P. D.-I. B. *Spacecraft Attitude Determination And Control*, v1.0 ed. Aerospace Technology Department, Hohenstaufenallee 6, 52064 Aachen, Germany, Winter 2009/2010. http://es.slideshare.net/abi3/bsf08-spacecraft-attitude-determination-and-control-v1-0.

[16] deWeck, O. L. Attitude determination and control (adcs). In *Space Systems Product Development* (Spring 2001), Department of Aeronautics and Astronautics - Massachusetts Institute of Technology.

[17] Digi International Inc. *XBee/XBee-PRO RF Modules Datasheet*, product manual v1.xex ed. https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf.

[18] Edgi. Gy-511 - 3-axis accelerometer and 3-axis magnetometer [lsm303dlhc]. Edgi Foundation Webpage. http://edgi.ru/moduli_datchiki_arduino/.

[19] ELECTRONIC, L. *PHOTO TACHOMETER Model : DT-2234B.* www.hestore.hu/files/DT-2234B.pdf.

[20] Electronics, K. *Datasheet Kysan RF-300FA.*

[21] Energies, E. *Datasheet of MGL2803 - Sealed Lithium Ion Rechargeable Battery*, December 2007. :http://www.enix-energies.fr/media/pdf/MGL2803$_U$*K.pdf*.

[22] EvdZ. Marlin repository. https://github.com/MarlinFirmware/Marlin.

[23] Fairchild. *Datasheet. S1A - S1M . General Purpose Rectifiers.*, May 2015. http://www.farnell.com/datasheets/1931532.pdf.

[24] Foundation, R. Reprap - about, 2014. http://reprap.org/wiki/About.

[25] from UniversitWrzburg, S. Floatsat 2014-2015 technical presentation. Chair of Computer Science VIII - Aerospace Information Technology, 2014/2015.

[26] Hessmer, D. R. Arduino compatible iic / i2c serial 2.5" lcd 1602 display module. Dr Rainer Hessmer Blog, January 2014. `http://www.hessmer.org/blog/2014/01/11/arduino-compatible-iic-i2c-serial-2-5-lcd-1602-display-module/`.

[27] HITACHI. *HD44780U (LCD-II) (Dot Matrix Liquid Crystal Display Controller/-Driver)*, rev. 0.0 ed. :https://www.sparkfun.com/datasheets/LCD/HD44780.pdf.

[28] Ibrahim, D. *Microcontroller Based Temperature Monitoring And Control*. Elsevier Science Technology Books, 2002.

[29] Inc., T. I. *LMV3xx Low-Voltage Rail-to-Rail Output Operational Amplifiers*, October 2014. `http://www.ti.com/lit/ds/symlink/lmv324.pdf`.

[30] Integrated, M. *MAX1771 - 12V or Adjustable, High-Efficiency, Low IQ, Step-Up DC-DC Controller*, 2002. `http://pdfserv.maximintegrated.com/en/ds/MAX1771.pdf`.

[31] InvenSense Inc. *MPU-6000 and MPU-6050 Product Specification*, revision 3.3 ed. 1197 Borregas Ave, Sunnyvale, CA 94089 U.S.A. Tel: +1 (408) 988-7339 Fax: +1 (408) 988-8104, May 2012. .

[32] <jeff@rowberg.net>, J. R. Mpu6050 calibration sketch. GitHub account. `https://github.com/jrowberg/i2cdevlib`.

[33] <jeff@rowberg.net>, J. R. Another auto-offset calibration sketch. i2cdevLib.com, January 31 2014. `http://www.i2cdevlib.com/forums/topic/112-another-auto-offset-calibration-sketch/`.

[34] Liebe, D. C. C. Star trackers for attitude determination. *IEEE AES Systems Magazine* (June 1995), 10–16.

[35] LPKF Laser & Electronics. *Datasheet LPKF ProtoMat S62*. `http://www.lpkfusa.com/datasheets/prototyping/s62.pdf`.

[36] Mabuchi Motor Co., L. *Datasheet Mabuchi motor RF-300FA*.

[37] Microelectronics, S. *L3G4200D MEMS motion sensor: ultra-stable three-axis digital output gyroscope*.

[38] Microelectronics, S. *L3GD20H MEMS motion sensor: three-axis digital output gyroscope*.

[39] Milla, M. Star tracker for bexus19 mission. Msc thesis (dissertation), University of Granada, Granada, January 2015. This MSc Thesis is a contribution to the GranaSAT project.

[40] Mitsumi. *Datasheet DC Mini-Motors M25E-4 Series*.

[41] NASA. 2.2 second drop tower. NASA Glenn Research Facilities. http://facilities.grc.nasa.gov/drop/index.html ; http://facilities.grc.nasa.gov/documents/TOPS/TopZERO.pdf.

[42] of Technology, D. U. Delfi-n3xt attitude determination & control subsystem. http://www.delfispace.nl/delfi-n3xt/attitude-determination-and-control-subsystem.

[43] Ponton's, J. Open loop tuning. The University of Edimburgh webpage. http://www.see.ed.ac.uk/~jwp/control06/controlcourse/course/map/ZN/opennotes.html.

[44] REXUS/BEXUS. Rocket & ballon experiments for university students, 2014. http://www.rexusbexus.net/.

[45] Rob O'Reilly, Kieran Harney, A. D. I. A. K. Sonic nirvana: Mems accelerometers as acoustic pickups in musical instruments. *Sensors Online* (2009). http://www.sensorsmag.com/sensors/acceleration-vibration/sonic-nirvana-mems-accelerometers-acoustic-pickups-musical-i-5852.

[46] Sangbum Cho, J. S., and McClamroch, N. H. Mathematical models for the triaxial attitude control testbed. *Technical report* (2003). University of Michigan, Ann Arbor, Michigan, United States.

[47] Schubert, H., and How, J. Space construction: an experimental testbed to develop enabling technologies. Aerospace Robotics Laboratory Stanford University, Stanford CA.

[48] Schwartz, J. L., and Hall, C. D. The distributed spacecraft attitude control system simulator: Development, progress, plans. *Virginia Polytechnic Institute & State University (Department of Aerospace & Ocean Engineering)* (2003).

[49] Software, R. H. . I2c lcd controller schematic (pcf8574). Radig Hard & Software Shop Webpage. http://shop.ulrichradig.de/Bausaetze/Kits-Boards/I2C-LCD-Controller-PCF8574.html.

[50] ST. *Datasheet of L298 - Dual full-bridge driver.* http://www.st.com/web/en/resource/technical/document/datasheet/CD00000240.pdf.

[51] Standard, I. Ieee standard for local and metropolitan area networks–part 15.4: Low-rate wireless personal area networks (lr-wpans) amendment 1: Mac sublayer. *IEEE* (2012).

[52] STMicroelectronics. *LSM303DLHC:Ultra-compact high-performance eCompass module:3D accelerometer and 3D magnetometer*, November 2013. Available at http://www.st.com/web/en/resource/technical/document/datasheet/DM00027543.pdf.

[53] Team, G. B. *GranaSAT Student Experiment Documentation*. University of Granada, January 2015. v5.

[54] Technologies, K. N6705a dc power analyzer, modular, 600 w, 4 slots. http://www.keysight.com/en/pd-1123271-pn-N6705A/dc-power-analyzer-modular-600-w-4-slots?cc=ES&lc=eng.

[55] Technology, L. *Datasheet LTC3440 - Micropower Synchronous Buck-Boost DC/DC Converter*. http://cds.linear.com/docs/en/datasheet/3440fc.pdf.

[56] Texas Instruments. *Datasheet SN74LVC1G125 Single Bus Buffer Gate With 3-StaDatasheet*, October 2014. http://www.mouser.com/ds/2/405/sn74lvc1g125-464564.pdf.

[57] Toral, A. Marlingnetic simulation anmaimplementation for a testbed for a cubesat. Final degree project, University of Granada, Granada, September 2015.

[58] University, C. S. P. *CubeSat Design Specification (CDS) REV 13*, 2014-07-07. Available at http://cubesat.calpoly.edu/images/developers/cds_rev13_final.pdf.

[59] Valenzuela, C. Desarrollo del sistema de comunicacion rf de la testbed del granasat-i. Msc thesis (dissertation), University of Granada, Granada, May 2015. This MSc Thesis is a contribution to the GranaSAT project.

[60] Wertz, J. R., and Larson, W. J. *Space Mission Analysis and Design*, third ed. Space Technology Series. Microcosm Press, Hawthorne, CA, 1999.

[61] Wikipedia. Adrian Bowyer biography. https://en.wikipedia.org/wiki/Adrian_Bowyer.

[62] Wikipedia. Air bearing. https://en.wikipedia.org/wiki/Air_bearing.

[63] Wikipedia. Matlab. https://en.wikipedia.org/wiki/MATLAB.

[64] Wikipedia. Pid controller. https://en.wikipedia.org/wiki/PID_controller.

[65] Yonatan Winetraub, D. A. B. H. *Attitude Determination Advanced Sun Sensors for Pico-satellites*. Handasaim School, Tel-Aviv University, Israel. https://www.agi.com/downloads/corporate/partners/edu/advancedSunSensorProject.pdf.