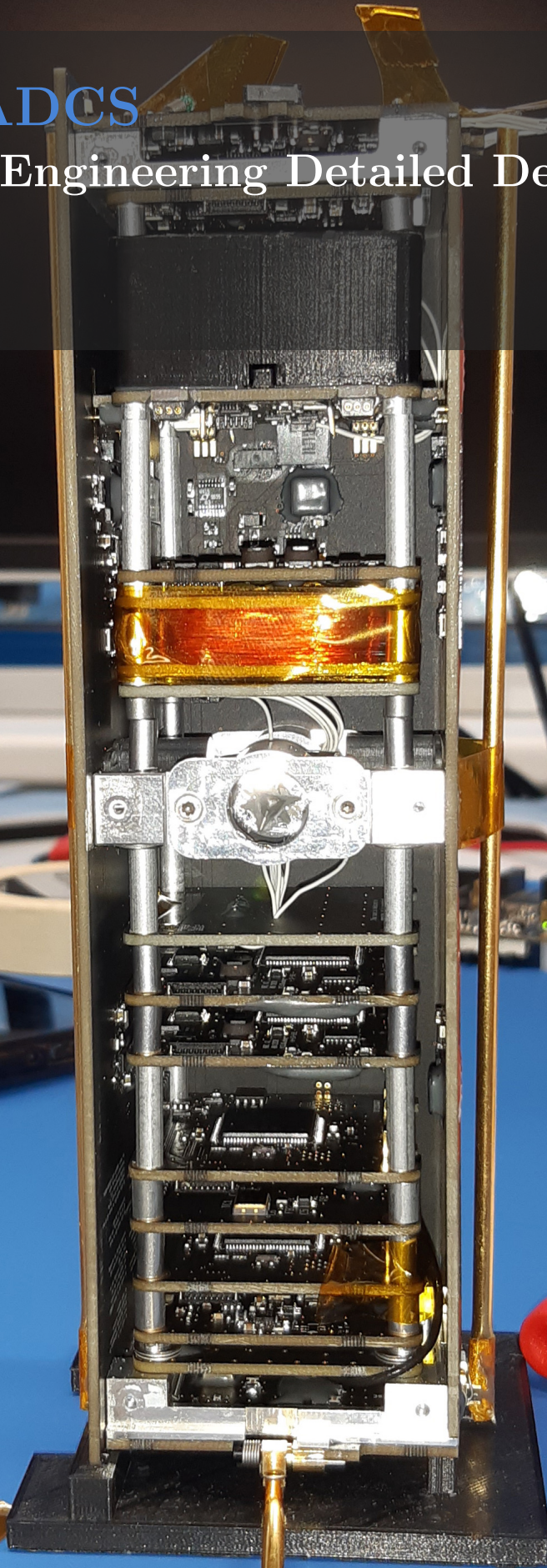


Delfi-Twin ADCS

AE4904 Space Engineering Detailed Design Project

Group B



Delfi-Twin ADCS

by

Group B

Student Names	Student Nr
Fatih Gök	6387276
Abhishek Avadhanam	6435548
Tjalling Huson	5109698
Maurizio Pavel	5797144
Paridhi Agarwal	5766214

Principal Tutor: J. Bouwmeester
Project Duration: Q3 2025/2026
Faculty: Faculty of Aerospace Engineering, Delft
Cover image: Delfi-PQ, nanosats.eu

Contents

Nomenclature	3
1 Introduction	4
2 Literature review	5
2.1 Background Research	5
2.2 Earth Horizon Sensor	5
2.3 Magnetometer	5
2.4 Magnetorquers	6
2.5 Microcontroller	6
2.6 System integration	6
3 Requirements and architecture	7
3.1 Requirement derivation	7
3.1.1 User Requirements	7
3.1.2 Interfaces	9
3.1.3 Functional analysis	10
3.2 Requirements	10
3.2.1 System requirements	10
3.2.2 Component requirements	13
3.3 Software architecture definition	15
3.3.1 Activity flows	16
3.3.2 Main loop timing	16
3.4 Hardware definition	19
3.5 System budgets	21
3.5.1 Power budget	21
3.5.2 Mass budget	22
3.5.3 Memory allocation	24
3.5.4 Communication channel overview	25
4 Components	26
4.1 Earth Horizon Sensor	26
4.2 Magnetorquer	28
4.2.1 Magnetorquer Design and Analysis	28
4.3 Magnetometer	47
4.3.1 Magnetometer Trade-Off Analysis	47
4.4 Microcontroller	50
4.5 Assembly configuration and CAD	51
5 Control and Simulation	55
5.1 Embedded control algorithms	55
5.1.1 Earth Horizon Sensor Algorithm	55
5.1.2 Magnetometer Algorithm	58
5.2 Flight Software Architecture	62
5.2.1 ADCS System Overview	62

5.2.2	Magnetic Detumbling Architecture	63
5.2.3	Nadir Pointing and Magnetic Steering Control	65
5.2.4	Dipole Command Pipeline and Actuator Command	67
5.3	Simulation Architecture and Physical Modelling	69
5.3.1	Integrated Simulation Architecture	69
5.3.2	State Initialization	71
5.3.3	Spacecraft Physical Model	74
5.3.4	Environment Models	78
5.3.5	Disturbance Models	79
5.3.6	Sensor and Actuator Simulation	82
5.4	Verification and Analysis	86
5.4.1	Verification Framework	86
5.4.2	Model Limitations and Assumptions	89
5.4.3	Detumbling Performance Results	92
5.4.4	Nadir-Pointing Performance Results	97
5.4.5	System-Level Robustness Assessment	101
6	Manufacturing	102
6.1	Integration planning	102
6.2	Magnetorquer manufacturing	103
6.2.1	Air-coil magnetorquer manufacturing	103
6.2.2	PCB printed magnetorquers	104
7	Requirement compliance check	105
8	Conclusion	107
9	Inventory & Activities	108
9.1	Hardware & Software Inventory	109
A	Deleted requirements	112

Nomenclature

Abbreviations

Abbreviation	Definition
ADCS	Attitude Determination and Control System
CAD	Computer-Aided Design
CC	Cycle-count
CPU	Central Processing Unit
DRDY	Data Ready
EHS	Earth Horizon Sensor
EPS	Electrical Power System
GNSS	Global Navigation Satellite System
GPIO	General Purpose Input/Output
HAL	Hardware Abstraction Layer
I2C	Inter-Integrated Circuit
IR	Infrared
LEO	Low Earth Orbit
LSB	Least Significant Bit
MEKF	Multiplicative Extended Kalman Filter
MM	Magnetometer
MSB	Most Significant Bit
MTQ	Magnetorquer
OBC	Onboard Computer
PCB	Printed Circuit Board
RAAN	Right Ascension of the Ascending Node
RANSAC	Random Sample Consensus
SMART	Specific, Measurable, Achievable, Reasonable, Time-based
SPI	Serial Peripheral Interface
SRAM	Static Random-Access Memory
SSO	Sun-Synchronous Orbit
SWaP	Size, Weight and Power
TBR	To Be Resolved
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
WIP	Work-In-Progress

1. Introduction

This report aims to design the Attitude Determination and Control Subsystem (ADCS) of the Delfi-Twin project, based on existing hardware and software design. Delfi-Twin is a formation flying mission of two PocketQubes and successor to the Delfi-PQ mission which was launched into space in 2022. It will perform formation flight, using differential drag through the motorized deployment angle of the solar panels.

In the previous years multiple components have already underwent R&D by staff members and students. This project is the culmination that is to bring everything together to a coherent system. It will feature four Commercial Off-The-Shelf (COTS) infrared sensors to be used as earth horizon sensors. This system works in parallel with a magnetometer to provide the attitude sensing. The heart of the system is the STM32L476RG microcontroller, it acts as the master of the system. For actuation a set of magnetorquers is to be designed and/or iterated upon previous work.

The complete system is supposed to adhere to the standards, protocols and dimensions set by the Delfi-Twin team. This report focuses on the design with the knowledge of having to manufacture and test it in the next phase of the project; the Space Engineering Practical.

2. Literature review

This section highlights the preliminary stage where the work that this project builds upon is dissected and new sources are evaluated and explored. As the ADCS has four different hardware components the literature study was conducted in four teams. As well as papers that provided useful information regarding the combination of sensors and background information deemed important to be aware of. The main literature topics were regarding the magnetometer, EHS, magnetorquers, microcontroller and how to make it a coherent system. The relevant papers for each topic are now briefly summarized and cited.

2.1 Background Research

The overall project is designing the subsystem of the Delfi-Twin satellites. The Delfi-Twin mission aims to better map and explain the phenomena called sporadic-E. Sporadic-E is a temporary ionospheric radio propagation reflection phenomenon with a low prediction probability, hence the name sporadic[1]. It is present in the E layer of the atmosphere which lies between 100 and 120 kilometers from the ground. The secondary mission goal was to improve formation flying by use of GNSS[2]. This mission is to be performed by use of PocketQubes, a TU Delft introduced standardized form-factor for satellites smaller than CubeSats[3][4]. They are utilized to have a cost effective way and versatile platform that can also serve as a learning opportunity for students.

2.2 Earth Horizon Sensor

The Earth horizon sensing of the Delfi-Twin was already decided upon in previous years so the documentation and briefing was condensed to a handful of papers as the groundwork was already laid. The Earth's horizon would be detected using four MLX90640 infrared sensors. The algorithm was to be iterated on, which is done in subsection 5.1.1. Apart from last year's report, the sensor itself was explored by studying the data sheet and driver documents. As the algorithm was to be iterated upon, other ways of tackling horizon sensing utilizing large constraints using IR sensors were explored.

The Earth Horizon Sensor system has already been worked on in previous years, it will function by use of four MLX90640BAA[5], one on each of the lateral faces of the Delfi-Twin. These infrared sensors can determine the spacecraft orientation by utilizing the stark thermal contrast between the radiation from the Earth and coldness of space. The previous algorithm works by mapping the Earth and then determining the X and Y vector plus the area of the Earth and searching for a match in a lookup table constructed from around 32000 simulated frames[6].

2.3 Magnetometer

The magnetometer that was used as a baseline in previous background literature was the RM3100[7]. This 3-axis magnetometer is appealing due to its small form factor whilst maintaining accurate readings. It supports I2C and SPI communication which corresponds to those found on the STM32. Lastly, it is very robust in receiving radiation[8].

2.4 Magnetorquers

The magnetorquers (MTQs) also had extensive work done in the last couple of years. The Delfi-PQ already had a magnetorquer configuration on-board consisting of three air-coil wound magnetorquers. This worked but had some shortcomings due to the very constraint space in which the magnetorquers have to fit. Some preliminary calculations were performed that showcased great promise for Printed Circuit-Board (PCB) integrated magnetorquers[9]. These would feature way less windings but would benefit from a way larger enclosed area as it could make use of almost the full length of the 3P pocketcube i.e. 178 mm. The old setup of the aircoiled magnetorquers were refined and tested again in the previous year by another SEP team. They arrived at a way lower magnetic dipole moment, probably due to poor construction[10]. Further reinforcing the idea of PCB integrated magnetorquers. It did provide good information regarding testing, using the DRV8838 as a driver[11].

2.5 Microcontroller

The chosen microcontroller is the STM32L476RG, STM32 for short, from STMicroelectronics. As this trade-off was already performed for us, no extra background research was needed except for the datasheet[12]. section 4.4 contains more information regarding the microcontroller.

2.6 System integration

The integration of magnetometer and Earth Horizon Sensor (EHS) measurements can be achieved using a range of spacecraft attitude determination architectures. Common CubeSat approaches include Kalman-filter-based estimators such as the EKF, MEKF, UKF, and AEKF[13][14][15][16]. Additional deterministic methods, including QuEst-based attitude reconstruction approaches, were also considered during the design trade-space exploration.

The project scope focused on lightweight architectures that could operate without continuous orbit propagation or computationally intensive onboard estimation. As a result, orbit-propagator-dependent and recursive filtering approaches were not pursued further.

The final architecture therefore used the magnetometer for scaled B-dot detumbling and relied on the EHS for nadir-pointing estimation and control.

3. Requirements and architecture

This section will explain how the requirements and architecture of the system was defined and constructed. First, an overview of the requirement derivation is given, containing most importantly the user requirements. From these, a list of system requirements is elaborated that defines what the system will look like. Then, the software architecture is defined, where an activity flow is developed and timing constraints are set for the main loop. Lastly, the resulting hardware architecture is shown and explained in with electrical drawing.

3.1 Requirement derivation

This section will explain the methodology behind the elaboration of the system requirements, shown in the next section. It begins by explaining the user requirements which are used to define the mission, alongside some internal additions to help further constrain the system design. Next, an overview of the interfaces between the ADCS and other systems, its environment and the satellite is given, followed by a brief overview of the functional analysis giving a picture of required system functions.

3.1.1 User Requirements

The first step to defining a system is to understand what the final user needs from the system. The below list contain a shortened overview of all the requirements that were given in the assignment specification. In addition to them, more options for requirements were explored to better define the system performance and characteristics. These are highlighted as "(ADDED)". Furthermore, multiple discussions with the Delfi-Twin team took place to flush out their user requirements. These are formatted in the "shall adhere to budget constraints" requirement.

To note is that all requirements are worded as "user requirements", which means that they were not meant to adhere to the SMART format. These are used to guide the process of defining the system requirements later.

- System shall use 3 Magnetometers
Rationale: Each MM is required to determine one component of the 3D magnetic field vector
- System shall use 3 Magnetorquers
Rationale: Each required for the alignment in one spatial direction
- System shall use 4 Earth Horizon Sensors
Rationale: By installing one EHS on each side of the cuboid satellite it is possible to find the Earth's horizon in any orientation.
- System shall perform detumbling to within 0.5 deg/s
Rationale: Detumbling is the most important task of the ADCS. The number was given by the user but we don't know why.
 - (ADDED) Detumbling shall be done within a few days
Rationale: Fully specifying the detumbling performance requires an upper bound on detumbling time.
 - (ADDED) Max initial rate shall be 180 deg/s
Rationale: Constraining the maximum initial rate of tumbling allows the ADCS's maximum performance to be defined in coordination with detumbling time (measurement frequency, dipole

moment). This value should be picked as the highest tumble rate to be reasonably expected after deployment

- System shall perform Coarse pointing: (simultaneous OR staged)

Rationale: For the spacecraft to perform its mission and communicate, it needs to be able to point in any direction

- Nadir ± 30 deg @ 1σ

Rationale: This is one of the ADCS's main functions. It comes from being able to discern which face is pointed at the earth, which requires more than 45 degree accuracy. On the other hand, we can't do too much accuracy. 30 degrees was chosen as middle ground, but can be increased if necessary.

- velocity vector pointing ± 10 deg @ 1σ

Rationale: One of the key functions of the ADCS is to point keep a prograde orientation when pointing to Nadir. The value is imposed by the user following a similar but stricter rationale as the previous requirement. Can be renegotiated.

- (ADDED) System shall be able to change Nadir-pointing axis to any vector on the satellite's xy-plane based on OBC command

Rationale: Nadir pointing is the other major function of the ADCS, alongside detumbling. While we could always point one axis to the Nadir, more functionality is always desired, and it should be relatively straightforward to implement being able to point to any angle.

- (ADDED) We define the axis system as xyz centered on the satellite with z being along the long axis and the positive z direction as being away from the solar panels

Rationale: Axis definition is crucial. Previous design reports chose a similar axis definition.

- System shall be able to change operational mode on command from OBC

Rationale: The ADCS performs what it is commanded by the OBC. It makes sense to let the OBC decide when it's time to point somewhere and when it isn't.

- System shall use STM32 in real time

Rationale: All the other microcontrollers in the satellite are also STM32s, which makes everything easier.

- System shall adhere to budget constraints

Rationale: ADCS should not be too heavy, power hungry or big.

- (ADDED) Shall not exceed 1 watt during nominal operation with 2 watts for peak performance

Rationale: The whole satellite can perhaps have up to 4 watts of power, and the ADCS should not consume a big chunk of it. The numbers here are guesses of what we should stay under.

- Indivisible volume of 45x45x15 mm

Rationale: A volume constraint to pursue independent development from the DelfiTwin team.

- TBR kg (~ 200 g)

Rationale: the whole satellite is likely to weigh around 700 g, of which the ADCS should not get a big chunk. 200 g is a best guess for what the system should stay under.

- Orbital altitude of 300-600 km SSO, nominal 550 km

Rationale: This is what the Falcon 9 will probably get us to.

- (ADDED) ADCS hardware should work the same regardless of where it is installed on the satellite.

Rationale: We don't know the layout of the rest of the satellite. We might have to be moved anywhere on the satellite to make room for other subsystems.

- (ADDED) Magnetorquers torque of at least TBR ($> \sim 1e-6$ Nm)

Rationale: To overcome the worst of orbital perturbations.

- (ADDED) Magnetometers must feel little to no influence from the Magnetorquers

Rationale: Otherwise, the MMs will read the output of the magnetorquer(s) instead of Earth's Magnetic field

- (ADDED) The EHS should distinguish the sun and the moon from Earth's horizon
Rationale: So the EHS may perform its tasks reliably

3.1.2 Interfaces

The next step is to look at what interfaces the ADCS must maintain. In other words, what other subsystems, items or entities the ADCS works with and must maintain relationships with. These are the following:

- On Board Computer (OBC): The ADCS receives commands and information from the OBC, as well as requests for data which it must then send to the OBC.
- The Electrical Power System (EPS): The ADCS receives power from the EPS according to what it can supply.
- The Environment: The ADCS needs to operate in vacuum and under temperature fluctuations, as well as be resistant to handling loads.
- The satellite itself: The purpose of the ADCS is to point the satellite.

For each interface, relationships were defined with respect to the ADCS. Relationships that go from the entity to the ADCS are noted with a right arrow ($\rightarrow$), and those that go from the ADCS to the entity are noted with a left arrow ($\leftarrow$). Below is the list of identified relationships.

- The Environment
 - $\rightarrow$ Perturbations: Atmospheric/gravity/magnetic disturbance; Micrometeoroids; Solar pressure; Ionizing radiation
 - $\rightarrow$ Sensor readings: Magnetic field; IR irradiance; Noise (Sun, Moon)
 - $\rightarrow$ Deterioration: Oxidation; Thermal stresses
 - $\rightarrow$ Launch & logistics: Vibrations; Humidity/wetness; Human handling; Temperature variation rate
 - $\leftarrow$ Produce Magnetic fields (torque)
 - $\leftarrow$ Produce Heat
- OBC
 - $\rightarrow$ Set Operational mode
 - $\rightarrow$ Receive data
 - $\rightarrow$ Reprogram the ADCS microcontroller
 - $\leftarrow$ Send housekeeping data: Angles; Rotation rates; Estimated state; Voltages; Health status
- EPS
 - $\rightarrow$ Provide power
 - $\leftarrow$ Take power
 - $\rightarrow$ Electrical errors (such as shortcircuits)
- Satellite itself
 - $\leftarrow$ Point towards specified direction
 - $\leftrightarrow$ EM interference
 - $\leftarrow$ MM-MT coupling
 - $\leftrightarrow$ Heat transfer

3.1.3 Functional analysis

Taking into account the relationships and the user requirements, an analysis was performed on what functions the ADCS should fulfill. For reference, below is a summary of the identified functions, graphically shown in Figure 3.1. Note that the F3.4 was deleted. This is because the ADCS is a slave to the OBC and only executing exactly what the OBC says is easier to implement.

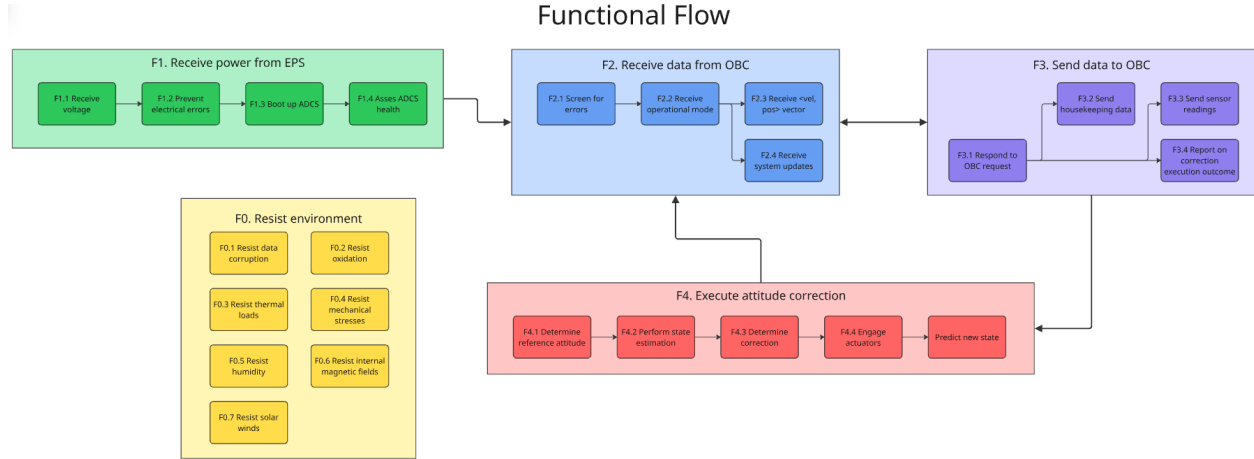


Figure 3.1: Functional flow diagram

3.2 Requirements

This section first shows the system requirements that were defined following the elaboration of the functions of the system. A brief list of component requirements is shown next. This list is meant to be used by the team to design the components, as is kept to minimum length to stay useful.

3.2.1 System requirements

There are 6 branches of the system requirements: ENV, PWR, REC, SND, ACT, CON. These stand for Environment, Power, Receive, Send, Actuate, and Constraint, respectively. Over the course of the project, certain requirements were deleted or edited after initial definition; the original version is still shown, but the modifications are shown and explained in an accompanying note. For reference, Appendix A contains a list of all deleted requirements.

- ENV: The system shall remain operational under its environmental conditions before and during the mission.

Rationale: This category encompasses all requirements that represent the environmental conditions that the ADCS operates in.

- ENV-2: System components shall either be shielded or withstand direct contact with atomic oxygen.

Rationale: Atomic oxygen has the potential to noticeably deteriorate component structures which negatively impact performance. System components should therefore either be shielded or be able to withstand outer space.

Verification method: Inspect integration model (shielding) + factsheet (exposed)

- ENV-3: The system shall operate between -10 and 40 degrees Celsius.

Rationale: Eclipsing leads to thermal cycling. The values used here are common values used for small satellites.

Verification method: Inspection via datasheets.

- ENV-4: The system shall withstand the mechanical loads as specified in the Falcon 9 user guide.

Rationale: Delfi-Twin will be launched on a Falcon 9. This requirement will likely be addressed after the design process has been moved onto PCB design.

- Verification method:** Analysis. Find eigenfrequency of the system and compare with Falcon 9 user manual.
- ENV-7: The components of the system shall be rated to withstand Low Earth Orbit ionizing radiation for at least 5 years.
Rationale: The system will have to operate in the low Earth orbit radiation environment for its entire mission lifetime.
Verification method: Inspection via heritage or data sheets.
 - PWR: The system shall receive power from the EPS.
Rationale: This category contains the requirements that define the interface between the EPS and the ADCS.
 - PWR-6: The system shall consume not more than 1 Watt in normal operation, with a peak consumption of 2 Watts.
Rationale: Since the satellite has not more than 4 Watts available, the system is limited to a small chunk of the total power budget. Values are currently a best-guess for what the system should stay under.
Verification method: Testing via the breadboard model.
 - PWR-1: The system shall operate using a voltage between 2.2 V and 4.5 V.
Rationale: The bus voltage is unregulated between 2.2 V and 4.5 V.
Verification method: Testing via the breadboard model.
 - PWR-7: The system shall draw a peak of no more than 0.5 amperes.
Rationale: Since the whole satellite might be getting 1 A or 1.5 A, we cannot take a big chunk of that. 0.5 is a rough good-enough value for now, might be renegotiated later.
Verification method: Testing via the breadboard model.
 - PWR-3: The system shall remain physically undamaged if power supply is cut off.
Rationale: To prepare for unexpected safe mode initiations.
Verification method: Testing via the breadboard model.
 - PWR-4: The system shall enter ~~systems-check~~ hibernate mode upon every boot-up. *Note: systems check is an obsolete operational mode*
Rationale: The OBC decides what the ADCS should be doing. Every time the ADCS starts receiving power, it waits for a command from the OBC before doing anything.
Verification method: Testing via breadboard model.
 - REC: The system shall receive data from the OBC.
Rationale: This category contains requirements that define how the ADCS receives commands from the OBC.
 - REC-1: The system shall recognize and ~~re-request~~ ignore faulty commands from the OBC. *Note: req edited after learning that the OBC doesn't accept requests*
Rationale: It is important that the ADCS behaves in an expected manner. It may only execute commands that it clearly understands and ignore the others.
Verification method: Testing via breadboard model. Using a laptop USB connection, feed faulty commands into the microcontroller and ensure that the ADCS ignores them.
 - REC-3: The system's operational mode shall be set by command from OBC.
Rationale: The OBC decides what the ADCS is doing, which is determined by its operational mode.
Verification method: Testing via breadboard model. Using a laptop USB connection, check that the operational mode can be changed.
 - REC-6: The system shall change its Nadir-pointing axis on command from the OBC.
Rationale: Although non-critical, more freedom is always desired. Being able to arbitrarily change the roll of the spacecraft opens up opportunities for research.
Verification method: Testing via breadboard model. Using a laptop USB connection, check that the ADCS can change its Nadir pointing axis.

- SND: The system shall send data to the OBC.

Rationale: This category encompasses how the ADCS sends back data to the OBC. Since the ADCS is defined as a slave to the OBC, the only communication in this sense is done as a response to an OBC request.

- SND-4: The system shall send all stored system data to the OBC on request.

Rationale: The OBC will need to know what the status of the ADCS is. By sending all stored system data, the OBC is given full insight into the health and operation of the ADCS. "Stored system data" does not refer to specific calculations. Instead, it refers to state variables (such as operational mode), measurements, and state estimations.

Verification: Testing via breadboard model. Using a laptop USB connection, system data is requested from the ADCS.

- ACT: The system shall control the satellite's attitude.

Rationale: This category describes the performance of the ADCS, including its two main operational modes, detumbling and pointing.

- ACT-1: The system shall detumble the satellite by reducing its rotational rate to at most 0.5 deg/s.

Rationale: Detumbling is one of the ADCS's main tasks. The cut-off value is an educated guess, can be renegotiated.

Verification method: Simulation

- * ACT-1.1 The system shall be able to detumble from at least 180 deg/s.

Rationale: To fully specify system performance, maximum initial spin rate is required to be constrained. The value chosen of 180 deg/s is a best practice value, although the usual tumbling rate following ejection is around 10 deg/s. Can be renegotiated.

Verification method: Simulation: experiment with increasing initial spin rates until the system fails.

- * ACT-1.2: The system shall detumble the satellite in no more than a week, preferably 3 days.

Rationale: Defining system performance requires a time constraint. The value should be chosen as a reasonable time that will neither require too much power nor take too much mission time. The current values were given by the user verbatim with this justification.

Verification method: Simulation: initialize the satellite with the maximum spin rate specified in ACT-1.1 and check that the satellite detumbles in time.

- ACT-2: The system shall point the +z axis of the satellite at the velocity vector within 10 degrees at 1 sigma confidence.

Rationale: One of the key functions of the ADCS is to point keep a prograde orientation when pointing to Nadir. The value is imposed by the user following a similar but stricter rationale as in ACT-3. Can be renegotiated.

Verification method: Simulation.

- * ACT-2.1: The system shall provide x- and y-axis torques at least 2 times stronger than disturbance torques in the respective axes at worst case scenario.

Rationale: A lower constraint on the MTQ torque is enforced by the need to overcome the disturbance torques on the satellite. The number should represent the minimum that would allow the satellite to keep control of itself. A value of 2 was chosen as a safety factor. A lower value was not chosen as the satellite is operating in somewhat uncertain conditions, especially regarding air density fluctuations.

Verification method: Test and analysis. Test the MTQ dipole moment and analyze the disturbance torques.

- ACT-3: The system shall point the $\pm x$ or $\pm y$ axis of the satellite with the Nadir within 30 degrees at 1 sigma confidence.

Rationale: This is one of the ADCS's main functions. It comes from being able to discern which face is pointed at the earth, which requires more than 45 degree accuracy. On the other hand, we can't do too much accuracy. 30 degrees was chosen as middle ground, but can be increased if necessary.

Verification method: Simulation

- * ACT-3.1: The system shall provide a z-axis torque at least 2 times stronger than disturbance torques in that axis at worst case scenario.
Rationale: A lower constraint on the MTQ torque is enforced by the need to overcome the disturbance torques on the satellite. The number should represent the minimum that would allow the satellite to keep control of itself. A value of 2 was chosen as a safety factor. A lower value was not chosen as the satellite is operating in somewhat uncertain conditions, especially regarding air density fluctuations.
Verification method: Test and analysis. Test the MTQ dipole moment and analyze the disturbance torques.
- * ACT-3.2: The system shall be able to turn 180 degrees in 12 hours around the z axis.
Rationale: Once finished with detumbling, the system orients itself towards Nadir. In the worst case scenario, this is a 180 degree turn. To constrain the performance of the MTQ, it is necessary to define a maximum time for this operation.
Verification method: Simulation
- ACT-4: The system shall ignore the presence of the Sun ~~and Moon~~. *Note: Moon's presence is ignored due to its relatively low impact.*
Rationale: The system should only care about the Earth's horizon, and the Sun and Moon are noise that could hinder performance.
Verification method: Analysis, testing.
- ACT-5: The magnetometer shall perform its readings after the magnetorquer's magnetic field is less than 10% of the external field.
Rationale: The MM's readings might be influenced by the MTQ if not enough time is given between measurement and actuation.
Verification method: Analysis. Make sure enough time is left between measuring and actuating to account for the rise and fall time of the magnetic field of the MTQs.
- CON: The system shall adhere to project budget constraints.
 - CON-1: The system shall occupy an indivisible volume not greater than 45 x 45 x 15 mm, except the 80 x 40 mm side panels which may be used.
Rationale: A volume constraint is set in order to pursue development of the ADCS independently from the rest of the Delfi-Twin team and to not intrude in allocated space of other subsystems.
Verification method: Integration model
 - CON-3: The system shall weigh less than 200 g.
Rationale: The satellite must weigh less than 750 g, and the ADCS should not take a big portion of it. The values are a best-guess until more accurate numbers are known.
Verification method: Engineering model
 - CON-4: The system shall be able to operate anywhere on the satellite.
Rationale: Since the layout of the satellite is not currently known, the ADCS needs to be designed to operate regardless of where inside the satellite it is placed. It might need to be moved to make place for other subsystems.
Verification method: Analysis

3.2.2 Component requirements

For the design of each component, component requirements are elaborated. The components are all given a unique identifier. The earth horizon sensor is EHS, magnetorquer is MGT, magnetometer is MGM, microcontroller unit/ADCS PCB is MCU. The list below is subject to change and TBR fields will be resolved once a value is known.

- ESH: Earth Horizon Sensor requirements
 - **ESH-02:** The Earth Horizon Sensors shall provide a pitch and roll measurement with an accuracy of at least 10 degrees.
Rationale: This requirement is meant to make sure that the Earth horizon can be found to a sufficiently high accuracy that ACT-2 and ACT-3 can be fulfilled. 5 degrees was chosen as it is a comfortable value for the system to achieve and it already is half below the required accuracy

from ACT-2.

Verification method: Analysis.

- **ESH-03:** Each Earth Horizon Sensor shall take no longer than 20 ms to perform a full reading.
Rationale: In order for a real-time system to work in a predictable manner, a time constraint must be placed on each task of the system. The current choice of horizon sensor is the MLX90640, which typically takes approximately 19 ms to perform one reading. Since modifications are not expected, 20 ms was gauged to be a reasonable upper limit. However, if modifications are needed, either they shall still satisfy this requirement, or the main loop timing will need to be changed (see subsection 3.3.2).
Verification method: Inspection, testing.

- MTQ: Magnetorquer requirements

- **MTQ-01 Duty cycle:** Each control loop, the MTQs shall actuate for at most 50% of the cycle.
Rationale: In accordance with PWR-06, by setting a duty cycle of 50%, we can get the ADCS to consume a peak of up to 2 Watts without pushing the average power consumption per cycle to more than 1 Watt. Same logic applies to the MTQs.
Verification method: Inspection, testing
- **MTQ-02 Peak power:** The combined peak consumption of all magnetorquers at once shall be no more than 1.8 Watts (instantaneous).
Rationale: In accordance with MTQ-01 and PWR-6, and since most other ADCS components have a power usage on the order of a few milliwatts, it is reasonable to allocate the largest portion of the power budget to the MTQs. Thus, 1.8 Watts was chosen as the upper bound for the power consumption of the MTQs.
Verification method: Analysis, testing
- **MTQ-03 Torque quantum:** The MTQs shall be able to provide a torque quantum each control loop to a step of 5% of the maximum torque in that loop.
Rationale: This requirement is meant to provide the possibility to modulate the strength of the MTQs each cycle, easiest done through time modulation. Given a shortest-possible loop length of 80 ms, 5% of that gives a pulse of 4ms, which the STM32's Hardware Abstraction Layer (HAL) is more than able of accomplishing through GPIO pins (see section 3.4). But if we make it shorter, it might not work as well.
Verification method: Testing
- **MTQ-04 Rise and fall time:** The MTQs shall have a combined rise fall time of no more than 10% of the minimum torquing duration.
Rationale: This means that at least 90% of an actuating cycle will be spent actuating according to nominal torque, which is close to good enough for giving predictable torque kicks. This would lead to a required rise & fall time of at most 0.4 ms, given MTQ-03's torque quantum duration of 4 ms.
Verification method: Analysis, testing.
- **MTQ-05 Detumbling torque:** Each MTQ shall produce a magnetic torque of at least $2.2 \cdot 10^{-7}$ Nm.
Rationale: In order to perform detumbling from 180 deg/s to 0.5 deg/s, a total $\Delta\omega = 3.14072$ rad/s is needed, to be provided in $t = 3 \cdot 24 \cdot 3600 = 259200$ seconds (according to ACT-1.2). This gives a required angular acceleration of $\alpha = \frac{\Delta\omega}{t} = 1.2117 \cdot 10^{-5}$ deg/s². With a worst-case moment of inertia of $I_{detumble} = 6.4 \cdot 10^{-3}$ kg m² (see ??), this result in an average required torque of $0.8 \cdot 10^{-7}$ Nm. Since this is required average output, to get the torque specification for the MTQ we add a $\sqrt{2}$ factor to account for average magnetic field misalignment (RMS of cosine), as well as a $n = 2$ safety factor to account for unforeseens (ACT-2.1, ACT-3.1), resulting in a required torque of $2.2 \cdot 10^{-7}$ Nm
Verification method: Analysis, testing
- **MTQ-06 XY torque:** The x and y magnetorquers shall produce a magnetic torque of at least $4 \cdot 10^{-7}$ Nm.
Rationale: This requirement relates to the need to reorient the satellite according to a new Nadir direction within 12 hours (see ACT-3.2). Simple dynamic problem about rotating a body of a given

moment of inertia I_z in a given time t_z along a certain amount of radians θ . Using kinematic equations, it is possible to express the required torque as $\tau = \frac{4I_z\theta}{t_z^2}$. Using $I_z = 4.895 \cdot 10^{-4}$ (calculated using script described in ?? using nominal values), $t_z = 12 \cdot 3600 = 43200$ s, and $\theta = \pi$ rad (coming from ACT-3.2), the resultant torque is $\tau = 1.4 \cdot 10^{-7}$ Nm. Since this is required average output, to get the torque specification for the MTQ we add a $\sqrt{2}$ factor to account for average magnetic field misalignment (RMS of cosine), as well as a $n = 2$ safety factor to account for unforeseens (ACT-2.1, ACT-3.1), resulting in a required torque of $4 \cdot 10^{-7}$ Nm.

Verification method: Analysis, testing

- **MTQ-07 Aero torque:** Each MTQ shall produce a magnetic torque of at least $9 \cdot 10^{-7}$ Nm.

Rationale: This requirement regards the possibility for the MTQs to overcome aerodynamic torque at the worst case scenario of satellite +z perpendicular to horizon, at 300 km. As shown in section 4.2, the maximum expected aero torque is $3.2 \cdot 10^{-7}$ Nm. Similarly to MTQ-05 and MTQ-06, this value is multiplied by a factor of $2\sqrt{2}$ to obtain the final required torque of $9 \cdot 10^{-7}$ Nm.

Verification method: Analysis, testing

- **MTQ-08 Current:** All MTQs operating simultaneously shall consume no more than 0.5 A in peak loading.

Rationale: Stemming from PWR-7, this requirement ensures that the total consumption of all MTQs at once will not exceed 0.5 A.

Verification method: Analysis.

- **MTQ-09 Voltage:** Each MTQ shall operate at a fixed voltage of 3.3 V.

Rationale: Stemming from PWR-1, all components in the system will be running on a fixed supply of 3.3 V through a voltage regulator.

Verification method: Analysis.

- **MGM: Magnetometer**

- **MGM-01:** The magnetometer shall take no more than 15ms to provide magnetic field data to the STM32.

Rationale: In order for a real-time system to work in a predictable manner, a time constraint must be placed on each task of the system. 200-cycle count readings take approximately 7 ms, with more accurate readings taking longer. 15 ms was chosen to leave a margin of error to ensure that the task is finished in time, and a timeout occurs only in case of error.

Verification method: Testing.

- **MGM-02:** The magnetometer shall measure the Earth's magnetic field with a total uncertainty not exceeding 0.2 %.

- **MGM-03:** The magnetometer shall measure magnetic fields in the range of ± 100 μ T.

- **MCU: Microcontroller unit/ADCS PCB**

- **MCU-01 PCB thickness:** The PCB thickness shall be no less than 1.3mm and no more than 1.8 mm in thickness.

Rationale: This requirement regards the main ADCS PCB that will incorporate all ICs of the system, mainly the microcontroller and sensors. The thickness is constrained for the MLX90640s to be able to slide onto the PCB and be soldered to the tin pads. The values come from the distance between the legs of the sensors (the lower value is the actual distance, the higher value is about as thick as it can be without severely bending the legs). See chapter 6 for more information

3.3 Software architecture definition

This section will describe the architecture of the software of the ADCS. The architecture was built following the steps of the functional analysis, but was adapted for software. First, general activity flow diagram for the ADCS shown. Next, the timing of the main loop is presented, explaining the time constraints of each task within the main loop. Finally, this is followed by pseudocode implementing the activity flows and constraints laid out in the main loop timing section.

Since software design from the ground up is something that the team is not directly experienced with, feedback, comments or directions towards specific guides are more than welcome.

3.3.1 Activity flows

The first step to deciding the software architecture of the system is elaborating its activity flow. An activity flow diagram is a graphical representation of the system's workflow, including actions, decisions and parallel activities. With it, flight software can be further developed based on high-level logic.

Figure 3.2 shows the main activity diagram of the ADCS. It begins with power-on, at which point the system undergoes boot-up, and enters two simultaneous loops. One loop deals with interrupts from the OBC's communications, where requests from the master controller must be dealt with at the highest priority. The other loop is the main control loop of the ADCS.

Upon boot-up, the ADCS enters an inactive state, instantiated by the Hibernation mode. In this mode, the system does nothing until an OBC interrupt is called which sets the operational state to active. The OBC may also set the ADCS back into an inactive state, upon which all its data is erased. Going to an active state again re-initializes all variables, essentially rebooting the system.

There are two active states: De-tumbling and Pointing. In De-tumbling mode, the system reduces the spacecraft angular velocity using a scaled B-dot magnetic damping controller driven by magnetometer measurements. In Pointing mode, the same magnetic damping controller remains active while an additional nadir-pointing controller generates corrective dipole commands from Earth horizon sensor measurements to align the spacecraft with the nadir direction. The outputs of both controllers are combined before magnetorquer actuation, allowing simultaneous rotational damping and nadir pointing. More detail on each state is given in subsection 3.3.2.

Setting the operational state to an active one automatically sets the system into De-tumbling mode. This is the first mode as it is expected that the ADCS will be turned operational right after satellite deployment, which may result in its fast-spin tumbling. The other operational mode is quickly reachable if the system measures a correspondingly low spin rate.

Every loop begins with a system check. This consists of parameters such as continuity checks in crucial connections and measured voltages. If these health parameters are considered acceptable, then system then continues onto its loop. Otherwise, it enters a "Safe Mode" loop where it waits until the OBC takes action.

Each active state has the system do a specific set of readings and calculations, at the end of which the microcontroller can perform an action with the magnetorquer. For maximum output torque, this means setting the MTQ on for the entire allocated duration, but it is not always the case. See next subsection for more details on this matter.

After the magnetorquer actuation is finished and its fall time expires, the loop is finished and the system goes back to the beginning by checking its hibernation state again.

3.3.2 Main loop timing

Since the ADCS is required to work as a real-time system, it is crucial to ensure that each task of the system is done within an allocated time. Thus, it becomes important to establish the precise timings of each task of the microcontroller within its main loop in a way that will ensure that either all tasks are done, or that the system does not get stuck on a task.

The STM32 functions as the master of the system with all the sensors and actuators as slaves. The subsystem itself is a slave to the OBC. This hierarchy defines the logic that is used to construct the software of the main loop.

The main loop has four operational modes, split between two active modes and two inactive modes. Each active mode has their own timing, explained later in this section.

- Detumbling (default active mode)
- Pointing (active)
- Hibernating (inactive)

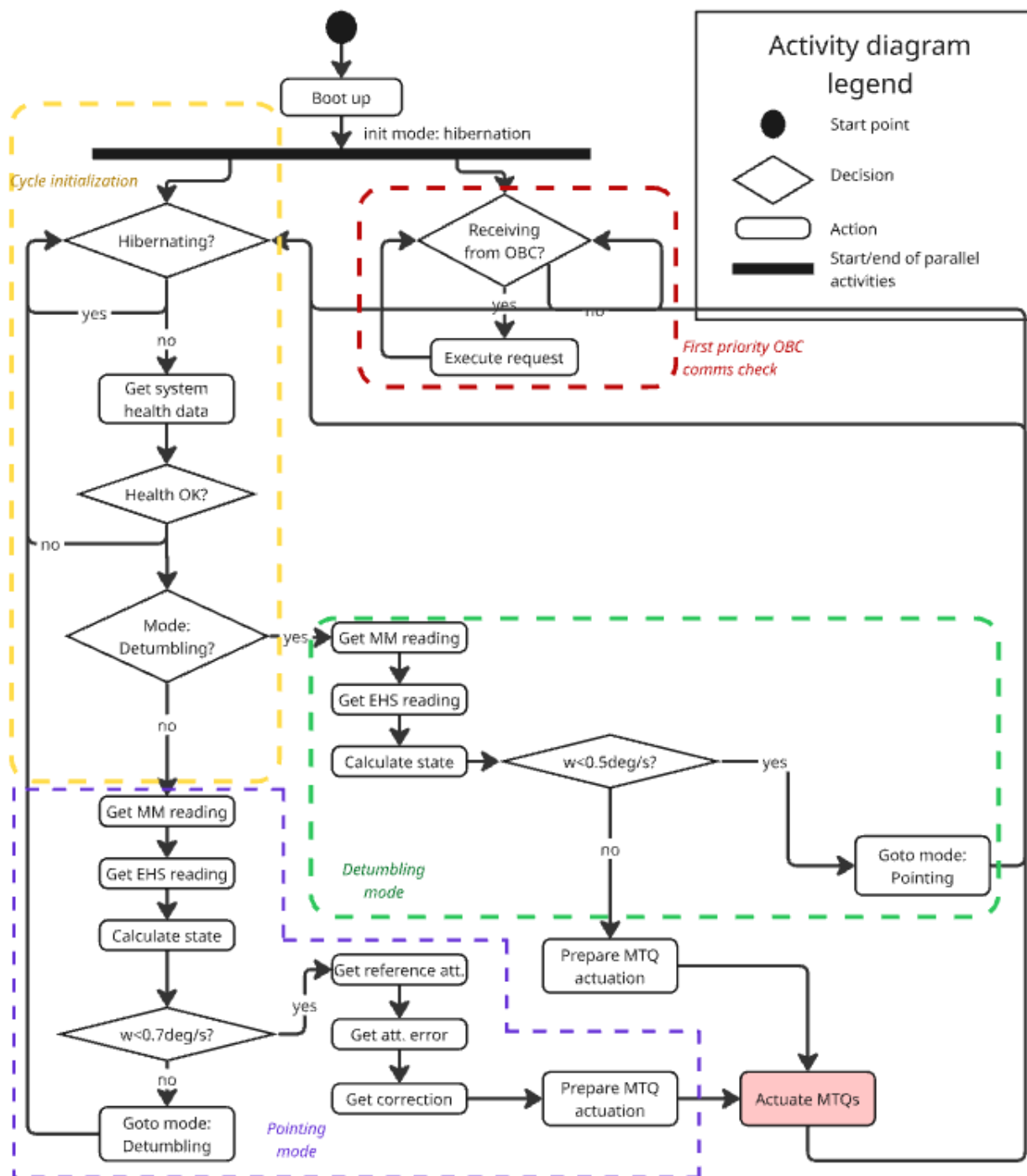


Figure 3.2: ADCS General Activity Flow Diagram

- Safe mode (inactive)

The OBC decides whether the ADCS is in an active or inactive mode, whereas the ADCS decides its own specific active mode. Upon boot-up, the ADCS is by default in Hibernating mode until the OBC moves it into an active mode. The default active mode is Detumbling, which can flow into to Pointing mode according to the situation.

Each active mode features a main loop of a specific length and contains different elements depending on which operational mode the ADCS is in. In any case, according to MTQ-01, the MTQs will always occupy half of the loop. The rest of the loop is split between microcontroller housekeeping, sensor readings, and data processing.

De-tumbling

De-tumbling is the default operational mode, meaning that when the OBC sets the ADCS from inactive to active mode, this is where it gets sent first. De-tumbling is meant to deal with potentially very high tumbling rates resulting from either initial satellite deployment or possibly long periods of time spent without active attitude control.

According to ACT-1.1, the maximum rate expected for this mode is 180 deg/s. At this rotational rate, the horizon sensors are no longer able to function, leaving only the magnetometers as the sole source of rotational rate measurement. As explained in chapter 5, the control loop performs three magnetometer readings per loop before calculating the B-dot vector and actuating the MTQs.

Furthermore, since De-tumbling is meant to be high-speed, the update frequency is maximized, going as fast as each component is able to work. At lower spin rates the MTQs do not have to be used on each cycle. It becomes very important to let measurements take a longer time in order to ensure useful readings. Thus, the MTQs are only used every k cycles, where k is an integer that is above 1 only after the spin rate has gone below 5 degrees/s. k then increases as spin rate decreases to let the satellite spin at least one degree before finishing taking readings to ensure accuracy. The chosen cut-off of one degree is based on a typical performance of the horizon sensors as was discussed in ??=[

, whose accuracy can reach ± 3 degree[6].

Once the tumbling rate decreases below 0.5 degrees/second, the ADCS will switch into Pointing mode, described later.

The below is a list of all tasks with allocated time within the Detumbling mode.

- **Cycle initialization:** The logic and calculations that the ADCS does before doing anything else in the cycle. This also includes health checks. Estimated time: 10 ms.
- **Magnetometer measurement (3x):** Three readings will be taken each cycle, each paired with a horizon sensor reading. According to MGM-01, each measurement will take 15 ms at most.
- **Magnetometer processing (3x):** After each magnetometer measurement, 5 ms is given for processing the data.
- **Horizon sensor measurement (3x):** Three readings will be taken each cycle, each paired with a magnetometer reading. According to EHS-03, each sensor will take at most 20 ms to perform its measurement. Since there are 4 sensors, 80 ms is needed for each full reading.
- **Horizon sensor data processing (3x):** After each 4-sensor reading, 40 ms is allotted for processing, 10 ms for each sensor.
- **Magnetorquer preparation:** Once the three pairs of readings are done and processed, the modified de-tumbling algorithm can be used, resulting in values that the MTQ driver can use. Estimated time: 20 ms.
- **MTQ actuation:** The MTQs are allocated half of the control loop. This does not mean that they will have to be used for half the loop. It only means that the microcontroller is not doing other tasks in this time. It is also possible for the MTQs to not be used at all in certain cycles, in order to let measurements take multiple loops.

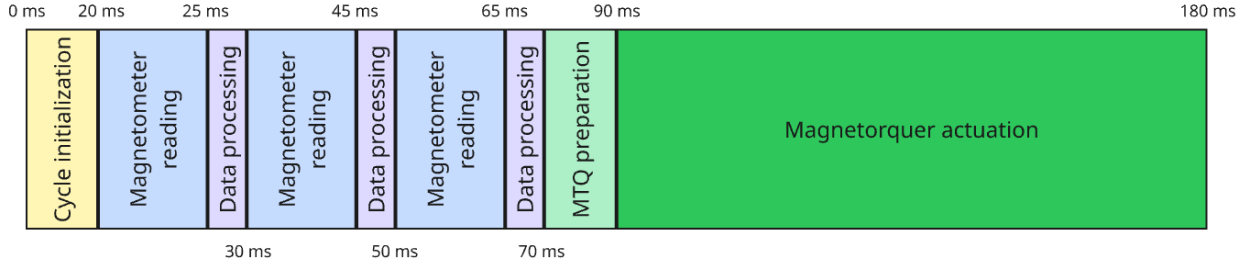


Figure 3.3: Illustration of the timing of the main loop in De-tumbling Mode

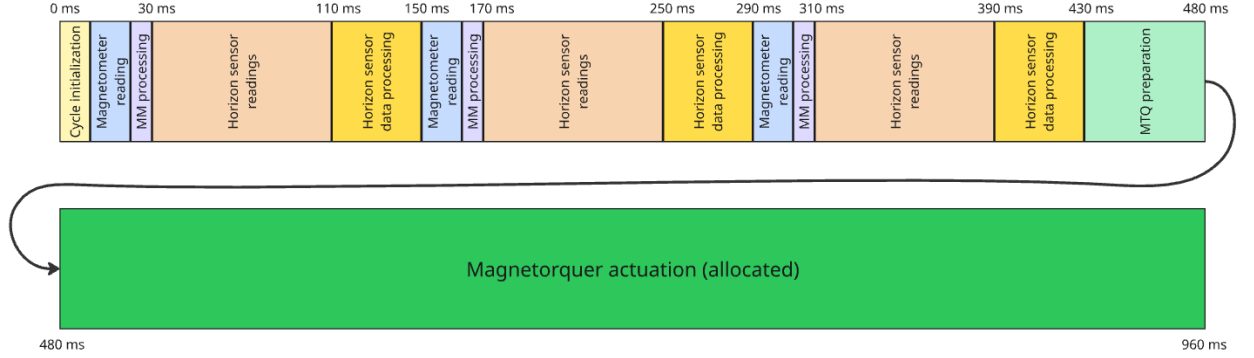


Figure 3.4: Main loop timing in Pointing mode

Adding the elements in the list above, excluding the MTQ actuation, the half-cycle length is 450 ms. Similarly to the De-tumbling mode, this represents an educated guess regarding the typical performance of the ADCS in de-tumbling mode, where all sensors and actuators are able to be used. Including the MTQ actuation, this increases the loop length to 900 ms.

Currently, synchronization with the OBC is not needed, but if the 900 ms loop length becomes a problem, that can be addressed. Increasing the loop length to 1000 ms in order to correspond to an update frequency of 1 Hz can be done by simply adding 100 ms of waiting time at the beginning of the loop.

Figure 3.3 is an illustration of the timings of each task within the main loop in de-tumbling.

Pointing

The Pointing mode is the active operational mode in which the satellite is assumed to not be rotating. The ADCS is tasked with orienting the satellite in the right direction, which is both aligning the $+z$ axis of the satellite with the velocity vector, as well as aligning the chosen Nadir-facing axis ($\pm x/y$) with the Nadir.

The timing of this mode is similar to the timing of the De-tumbling mode. Specifically, the same tasks and their respective timings are allocated, with the exception of the MTQ preparation which is extended by 50 ms to leave time for further data processing, such as algorithm execution and Kalman filtering. The final resulting loop length is 1000 ms, corresponding to an update frequency of 1 Hz.

Figure 3.4 is an illustration of the timings of each task within the main loop in de-tumbling.

3.4 Hardware definition

In parallel to the software architecture development, a hardware architecture was also defined, shown summarized in Figure 3.5. This schematic displays the electric and logic connections between every component, centered around the microcontroller evaluation board. Every component is shown with its full array of connections as specified in their datasheets, and the magnetorquers are represented as simple boxes with one input and one output. The components are as follows:

- **NUCLEO-L476RG:** The evaluation board for the STM32L476RG microcontroller which will be the centerpiece of the ADCS. It is connected to the voltage regulator which supplies it 3.3 V through the corresponding 3V3 pin. A table of all pin connections is given in Table 3.1.
- **U1V10F3 Voltage Regulator:** As mentioned in PWR-1, the supply voltage from the EPS is unregulated. This regulator is able to regulate an input supply voltage between 0.5 V and 5 V and give a fixed output of 3.3 V that all other components can use. It is labeled U1, and it has pins VIN, VOUT and GND.
- **MLX90640 IR Sensors (4x):** These are the sensors used for finding the horizon of the Earth and subsequently finding the Nadir direction. They are numbered M1, M2, M3, M4, and they are connected via two separate I2C connections, I2C1 (M3, M4) and I2C2 (M1, M2).
- **RM3100 Magnetometer:** This is the 3-axis magnetometer that is used to help find tumbling states and yaw direction of the velocity vector. Although the RM3100 supports both I2C and SPI, it was designed for I2C, which is a simpler protocol to implement. For this reason, it is connected via a third I2C connection, I2C3. It is labeled G1.
- **DRV8838EVM DC motor drivers (3x):** These are the evaluation boards for the DRV8838 drivers typically used for DC motors. Their selection is explained in section 4.2. They are connected using three pins: nSLEEP (not sleep), PH (phase) and EN (enable). Since only slow switching of these lines is expected, they will be connected to GPIO ports on the microcontroller. They are labeled D1, D2, D3. [11][17]
- **OBC connection:** The OBC will communicate with the ADCS microcontroller through the CAN1 protocol. This requires two pins, RX and TX.

Pin	Function	Connects to
3V3	Power supply	U1 VOUT
GND	Ground	Various
PC10	GPIO_OUT	D1 nSLEEP
PC11	GPIO_OUT	D1 PH
PC12	GPIO_OUT	D1 EN
PA15	GPIO_OUT	D2 nSLEEP
PC13	GPIO_OUT	D2 PH
PC2	GPIO_OUT	D2 EN
PC3	GPIO_OUT	D3 nSLEEP
PA0	GPIO_OUT	D3 PH
PA1	GPIO_OUT	D3 EN
PC0	I2C3_SCL	G1 SCL
PC1	I2C3_SDA	G1 SDA
PC7	GPIO_IN	G1 DRDY
PC8	I2C1_SCL	M3, M4 SCL
PB9	I2C1_SDA	M3, M4 SDA
PB10	I2C2_SCL	M1, M2 SCL
PB11	I2C2_SDA	M1, M2 SDA
PA11	CAN1_RX	OBC CAN1 TX
PA12	CAN1_TX	OBC CAN1 RX

Table 3.1: Pin connections from the STM32L476RG

The hardware architecture was defined with evaluation boards as most initial testing will be performed with them due to their ease of use. It is expected that the architecture is likely to change once testing is finished, and thus effort to construct a detailed electrical diagram is minimized until design verification through testing.

ADD THE
KICAD
PROJECT
FILE TO
THE AN-
NEX

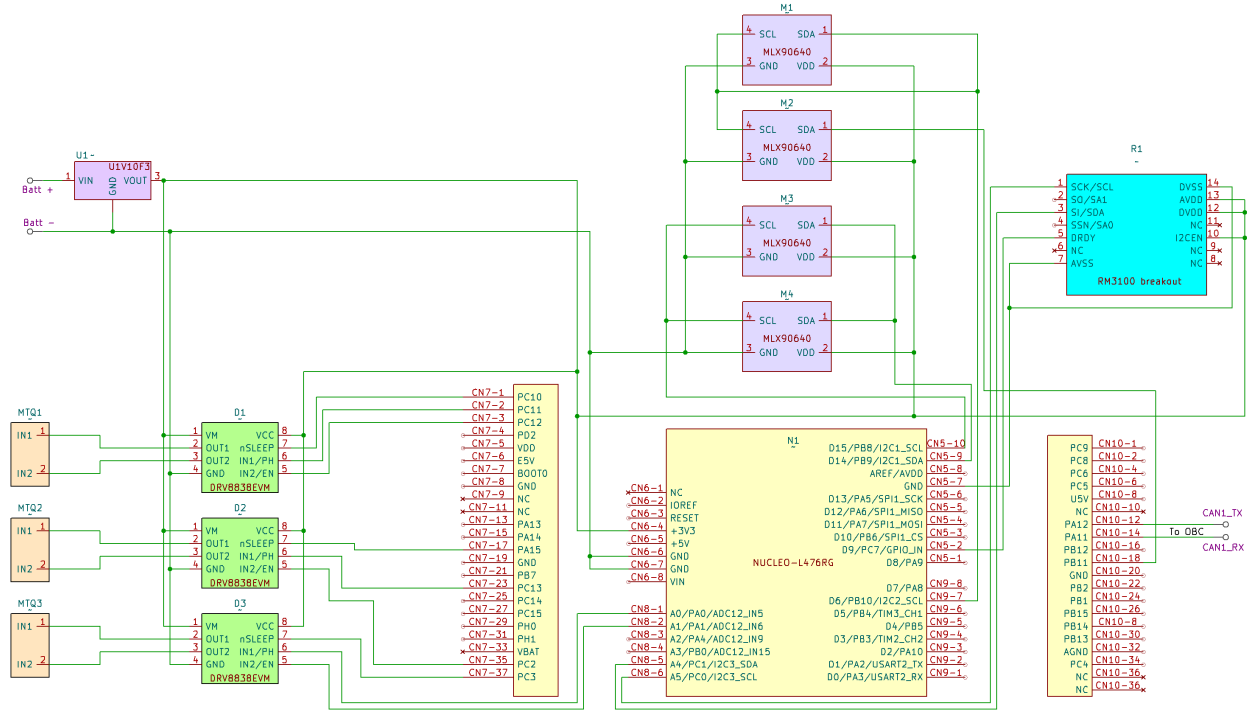


Figure 3.5: Pinout & electric circuit schematic of ADCS hardware

3.5 System budgets

This section will detail the management of the system budgets according to the system and component requirements. First, a power budget is elaborated and shown to fit within PWR-6, followed by the mass budget, which fits within CON-3. Next, all data to be stored into the flash memory of the STM32L476RG is allocated size and location. Finally, a brief overview is given on the each communication channel, with sanity checks to ensure that they can transmit their information in the allotted time.

3.5.1 Power budget

The power budget is shown in Table 3.5. It was constructed as a way to track and verify compliance with PWR-6. In order to do that, it was assumed that each component is supplied the same voltage of 3.3 V. Then, using their respective datasheets, the maximum typical current draw is found, which is multiplied by the voltage to give the peak power consumption.

A column is added to the table labeled "Timing group". When considering the peak power consumption of the ADCS, it matters to know when each component is being used within the main loop update cycle (see subsection 3.3.2). Each cycle is divided into timing groups, in which one or more components are assumed to be working at peak power. With this, the peak power consumption for each timing group can be found by summing the powers of each component in that timing group. Additionally, by multiplying the peak power consumption of each component by the fraction of the cycle that it is in use, it is possible to calculate the equivalent average power consumption of that component.

For the MTQs, their power consumption is shown in section 4.2 as 1.24 W in total. This value is the amount consumed in continuous operation, and is used for calculating the peak power. Since the MTQs are in operation for a duty cycle of a maximum of 50% (see MTQ-01), this gives an average power of 0.62 W.

For the STM32, the assumed case is high-frequency at the upper end of the temperature range (ENV-3). The datasheet states that at high frequency (80 MHz), at 55 degrees C, the microcontroller draws 10.3 mA typically, with a max of 11.8 mA. This leads to 34 mW and 39 mW respectively.

For the RM3100, there are 4 components to consider. First, the MagI2C has a typical current draw of 1 mA. Second, the typical current draws of the Sen-XY-f/Sen-Z-f are 0.26 mA each. This leads to a total current

draw of 1.8 mA.

The voltage regulator has an efficiency dependent on both the current drawn and the voltage supplied. Depending on the efficiency, its power consumption also changes. As a result, the power consumed by the regulator depends on the power consumed by the other components. With an average power draw of 1 W (CON-3), an average output current of ~ 300 mA is drawn. At 3.3 V, this leads to an efficiency of 90%, meaning that the voltage regulator consumes $1/0.9 \approx 11\%$ times the output power. This means that, in order for CON-3 to be satisfied, all other components must have a combined average power of at most 0.9 W.

Table 3.2: Typical current draws of components

Label(s)	Component(s)	Current draw	Timing group
MTQ1-3	Magnetorquers (combined)	376 mA	A
M1-4	4x MLX90640	20 mA	B
D1-3	3x DR883x	0.8 mA source	A
N1	STM32	10.3 mA	A, B, C
R1	RM3100	1.8 mA	C
U1	Volt. regulator	<i>see text</i>	all

For De-tumbling Mode (Table 3.3), group C is assigned to the first half of the main loop, and group A is assigned to the second half (for the magnetorquer actuation; see Figure 3.3). Group A has a current draw of 388.7 mA. From the voltage regulator datasheet, for this current, at an output voltage of 3.3V, the efficiency hovers around 80%. The power is calculated by multiplying the current draw by the voltage and then dividing it by the regulator efficiency. To note is that this operational mode does not use the MLX90640 sensors which is why group B does not exist in the table.

Table 3.3: Power consumption by timing group, De-tumbling mode

Timing group	Components	Duty cycle	Current	U1 eff.	Power incl. U1
A	N1, D1-3, MTQ1-3	50%	388.7 mA	80%	1.60 W
B	N/A	-	-	-	-
C	N1, R1	50%	12.1 mA	65%	0.06 W
Cycle average	all	100%	-	-	0.830 W

For Pointing mode (Table 3.4), groups are assigned similarly to Detumbling Mode. The difference is that group B is assigned 36% of the cycle, and group C is assigned 14%.

Table 3.4: Power consumption by timing group, Pointing mode

Timing group	Components	Duty cycle	Current	U1 eff.	Power incl. U1
A	N1, D1-3, MTQ1-3	50%	388.7 mA	80%	1.60 W
B	N1, M1-4	36%	90.3 mA	65%	0.46 W
C	N1, R1	14%	12.1 mA	65%	0.06 W
Cycle average	all	100%	-	-	0.974 W

Table 3.5: Summary of power budget by operating mode

Operating mode	Peak power	Average power
De-tumbling	1.6 W	0.83 W
Pointing	1.6 W	0.974 W

3.5.2 Mass budget

Table 3.6 shows the mass allocation of each component. The mass budget was created as a way to track and verify compliance with CON-3. To note is that the allocation of ICs includes their supporting electronic

Table 3.6: Mass budget for the Delfi-Twin ADCS

Label	Component	Mass (g)
MTQ1	MTQ X	25
MTQ2	MTQ Y	25
MTQ3	MTQ Z	52
D1	DRV8838	1
D2	DRV8838	1
D3	DRV8838	1
U1	Voltage regulator	1
M1	MLX90640	3
M2	MLX90640	3
M3	MLX90640	3
M4	MLX90640	3
N1	STM32	21
R1	RM3100	1
-	MTQ housing	18
-	20% margin	40
	Total	198

components (capacitors, resistors, leads, etc), except the STM32’s allocation which includes an estimation for the entire ADCS PCB. A significant margin of 20% was chosen because this mass budget does not take into account wires and unforeseen mass contributors, which may become substantial.

For the MLX90640 sensors, the official datasheet does not provide individual sensor weights[5]. However, a Product Overview for the same products by Mouser shows that a 110-degree FOV sensor, together with its breakout board, would equate to 3 grams. Thus, in this mass budget, a mass of 3 grams is assigned to each sensor M1-M4 under the assumption that the mass presented in the Mouser file includes both the sensor mass and a reasonable approximation for the mass of the connectors and ADCS PCB components.

Source for DRV883x mass can be found [here](#). Source for the voltage regulator can be found [here](#).

For the RM3100, the datasheet states a very low mass of about 0.04 grams for the sensors. Since this doesn’t include all other components, including MagI2C, it is assumed that the mass is similar to the DRV883x, which is similarly sized. Thus, 1 gram is chosen as budgeted for the RM3100 mass, so as to include a margin for ADCS PCB area usage and other components such as capacitors.

Documentation for the mass of the STM32 is scarce, and it depends on the model. However, since the IC is very small, it likely has a negligible mass, similar to the previous components. Thus, its weight entry in Table 3.6 will be used for estimating the mass of the ADCS PCB. As shown in section 4.5, there is currently a placeholder for the PCB, whose weight is estimated at 5 grams, using a density 2 g/cm^3 (the density of the fiberglass usually used in PCBs). Therefore, it is prudent to assign 5 grams to the STM32 for in order to account for the weight of the ADCS PCB, whose mass dominates that of the STM32’s itself.

Similarly, the weight of the X and Y MTQs can be estimated using their CAD in Fusion, as shown in section 4.5. With a density of 5 g/cm^3 and a volume of $\sim 5 \text{ cm}^3$, their mass is close to ~ 25 grams. The reason for choosing 5 g/cm^3 instead of fiberglass’s 2 g/cm^3 is because the MTQ PCBs are expected to contain a large quantity of copper, whose density is close to 9 g/cm^3 . Using the rough assumption that half of the mass of the MTQ PCBs will be copper, the density then becomes the rounded average of fiberglass and copper, hence the use of 5 g/cm^3 .

The mass of the MTQ housing, which also includes the mounts for the MLX90640 sensors, can be calculated from section 4.5 similarly. Since it is to be 3D-printed, a conservative density of 2 g/cm^3 will be used (comp. $\sim 1.8 \text{ g/cm}^3$ for ABS). With a Fusion-calculated volume of 9.6 cm^3 , this leads to a mass of 17.3 grams, rounded up to 18 grams.

The mass of the Z aircoil can also be estimated using its model in Fusion (see section 4.5. With a copper density of 9 g/cm^3 and a volume of 5.7 cm^3 (assuming perfect infill, which is impossible), a upper-bound weight estimation is found of 51.3 grams, rounded up to 52 grams.

3.5.3 Memory allocation

The STM32L476RG uses page-based flash memory, each page being 2 Kbyte. In total, it has 512 pages divided into two banks, giving 1024 KByte of usable memory. The bank division allows for read-while-write operations. While this could be an advantageous for systems with very large flash memory usage, shown below, the Delfi-Twin ADCS does not meet this description. To simplify memory usage, banks will be disabled.

Since erasing and writing to flash memory is done at a page level, and given the large amount of available memory, it makes sense to assign one page to each stored group of data. The below is a list naming and describing all data to be stored on the STM32's flash memory.

- **Time and date:** HAL has a Real-Time Clock (RTC) that can store a full timestamp in 20 bytes.
source
Total: 160 bits / 20 bytes.
- **Operational state:** There are 5 operational states, requiring only a small sized variable. The format used will be the smallest standard integer available, which is 16 bits.
Total: 16 bits / 2 byte.
- **Nadir axis:** Expressed as 3 numbers along 3 axes. Since only coarse pointing is used, high number precision is not needed. Half-precision floating numbers (16 bits) can be used.
Total: 48 bits / 6 bytes
- **State estimation:** Three-axis full-precision (32-bit) angle, angular velocity, angular acceleration, and B-dot.
Total: 384 bits / 48 bytes.
- **PID Controller:** Three sets of full-precision (32-bit) PID gains, three numbers per axis.
Total: 288 bits / 36 bytes.
- **B-dot controller:** One full-precision float.
Total: 32 bits / 4 bytes.
- **MMOI:** A 3x3 matrix of full-precision floats.
Total: 288 bits / 36 bytes.
- **MTQ dipole:** One half-precision float per MTQ, stating how much magnetic dipole it can produce.
Total: 48 bits / 6 bytes.
- **EHS Readings:** All the data that one MLX90640 gives, including frame data, amounts to 834 words, or 1668 bytes (see section 4.1). This fits inside one 2 KB page. Since there are four sensors, this amounts to 4 pages.
Total: 53376 bits / 6672 bytes.
- **EHS bias values:** Two half-precision floats, one per sensor.
Total: 128 bits / 16 bytes.
- **Magnetometer:** The RM3100 outputs three 32-bit numbers (one for each axis); additionally, it is possible to read all memory bytes of the sensor, equaling 9 bytes, or 72 bits.
Total: 168 bits / 21 bytes.
- **Magnetometer bias corrections:** Three half-precision floats, one per axis.
Total: 48 bits / 6 bytes.

As shown in Table 3.7, this amounts to: 6873 bytes. Since the STM32L476RG has 1 Mbyte of available flash memory, this amount of used memory falls comfortably within the available budget. No further analysis is required. However, if more flash space is needed in the future for any reason, the size of state estimation, PID gains, and MTQ inputs can be halved by lowering their precision. This is because only coarse pointing is required, and high precision is not strictly needed. However, since higher precision is likely to improve performance, and memory space is not an issue, they are left on full precision for now.

Table 3.7: Summary of flash memory allocation

Stored item	Size (bytes)	Page	Address	Channel
Timestamp	20	0	0x0800 0000	Internal
Opmode	2	1	0x0800 0800	Internal
Nadir axis	6	2	0x0800 1000	Internal
State estimation	48	3	0x0800 1800	Internal
PID Controller	36	4	0x0800 2000	Internal
B-dot controller	4	5	0x0800 2800	Internal
MMOI	36	6	0x0800 3000	Internal
MTQ Dipoles	6	7	0x0800 3800	Internal
M1 data	1668	8	0x0800 4000	I2C2
M2 data	1668	9	0x0800 4800	I2C2
M3 data	1668	10	0x0800 5000	I2C1
M4 data	1668	11	0x0800 5800	I2C1
EHS biases	16	12	0x0800 6000	Internal
MM output	21	13	0x0800 6800	I2C3
MM bias	6	14	0x0800 7000	Internal
Total	6873	-	-	-

3.5.4 Communication channel overview

For each communication protocol, a certain amount of data has to be sent in an allocated time. This subsection will perform a sanity check of the speeds and capacities of each channel. This is summarized in Table 3.8, where each channel is evaluated for its maximum performance (capacity) compared against its required bitrate, which in turn comes from its worst-case maximum packet and its allocated time. See subsection 3.3.2 for more details on the timing.

The I2C1 and I2C2 channels are used for MLX90640 communication. They are set into Fast Mode Plus, giving 1 Mbit/s channel capacity. In contrast, since I2C3 channel is used for communication with the RM3100 with a relatively little needed bitrate, it is set to Standard Mode (100 kbit/s).

The CAN1 channel is used for communication with the OBC. The data packet definition is outside the scope of this report, but in the worst case scenario, the OBC asks for the ADCS to send it all of its stored flash data; as shown in Table 3.7, this amounts to 6.873 Kbytes. Its speed is highly dependent on the design of the rest of the satellite, which is not yet known. Assuming a modest speed of 100kbit/s (equivalent to standard-mode I2C), that gives a transmit speed of 0.55 seconds. In effect, this would mean that the ADCS would be frozen for what would be an update cycle (or a significant chunk of it) while transmitting all of its flash memory to the OBC. Therefore, this operation should not be used during detumbling, unless CAN1 speed is vastly increased.

However, communication through CAN1 can be greatly improved if the EHS data is not used (6672 bytes). From a total of 6873 bytes, only 201 bytes would be transmitted, which a 100 kbit/s channel capacity would transmit in 16 ms. This number is much more manageable and would not significantly disturb the operations of the ADCS during detumbling.

Table 3.8: Channel capacity and needed bitrate

Channel	Max packet (kbit)	Time (ms)	Bitrate (kbps)	Capacity (kbps)	OK?
I2C1	13.344	20	667.2	1000	Yes
I2C2	13.344	20	667.2	1000	Yes
I2C3	0.0205	15	11.2	100	Yes
CAN1	6873 (0.201)	550 (16)	-	100?	see text

4. Components

This section outlines the variety of components the ADCS is composed of, starting with the four earth horizon sensors in section 4.1. Then the magnetorquers and magnetometer are elaborated on. Lastly the micro-controller that directs and reads out from the subsystem is explained, together with its communication protocol with the OBC. To give a good overview of the system and visualisation, a CAD model was made and configuration tree was constructed in section 4.5 to outline all the hard- and software used or created in this project as well as give a conceptual visualisation of the final subsystem.

4.1 Earth Horizon Sensor

The Earth Horizon system was already well into its development as it was a standalone project last year. Thus, a trade-off is already performed. The ADCS has as a primary means of attitude determination a set of four infrared, IR, sensors that detect in the $1.4\mu\text{m}$ to $7\mu\text{m}$ range. The IR sensors selected are the MLX90640BAA from Melexis[5]. This is the wide-angle variant of the MLX90640 family with a FOV of 110° by 75° as to negate any possible spacecraft orientation in which no sensor can sense the horizon. This so called deadzone of sensors[6] is almost eliminated. Only the Z-surfaces are free of EHSs which causes a deadzones. However, these surfaces are designed to face the direction of travel and the retrograde direction.

The refresh rate and resolution are configurable and were already determined to be 8 Hz and 19 bits respectively. This the sweet-spot of data capture and noise. Lastly, 8 Hz is the highest refresh rate possible without congesting the I2C communication channel as two sensors have to share one channel.

The sensor functions by having 32 by 24 pixels resulting in a total of 768 pixels that are to be read out. They are structured as shown in Figure 4.1. These pixels combined with some auxiliary data and housekeeping results in a total of 834 words to be read out each frame[18]. Each word consists of 16 bits thus when `MLX90640.GetFrameData` is invoked 834 times 2 bytes are created.

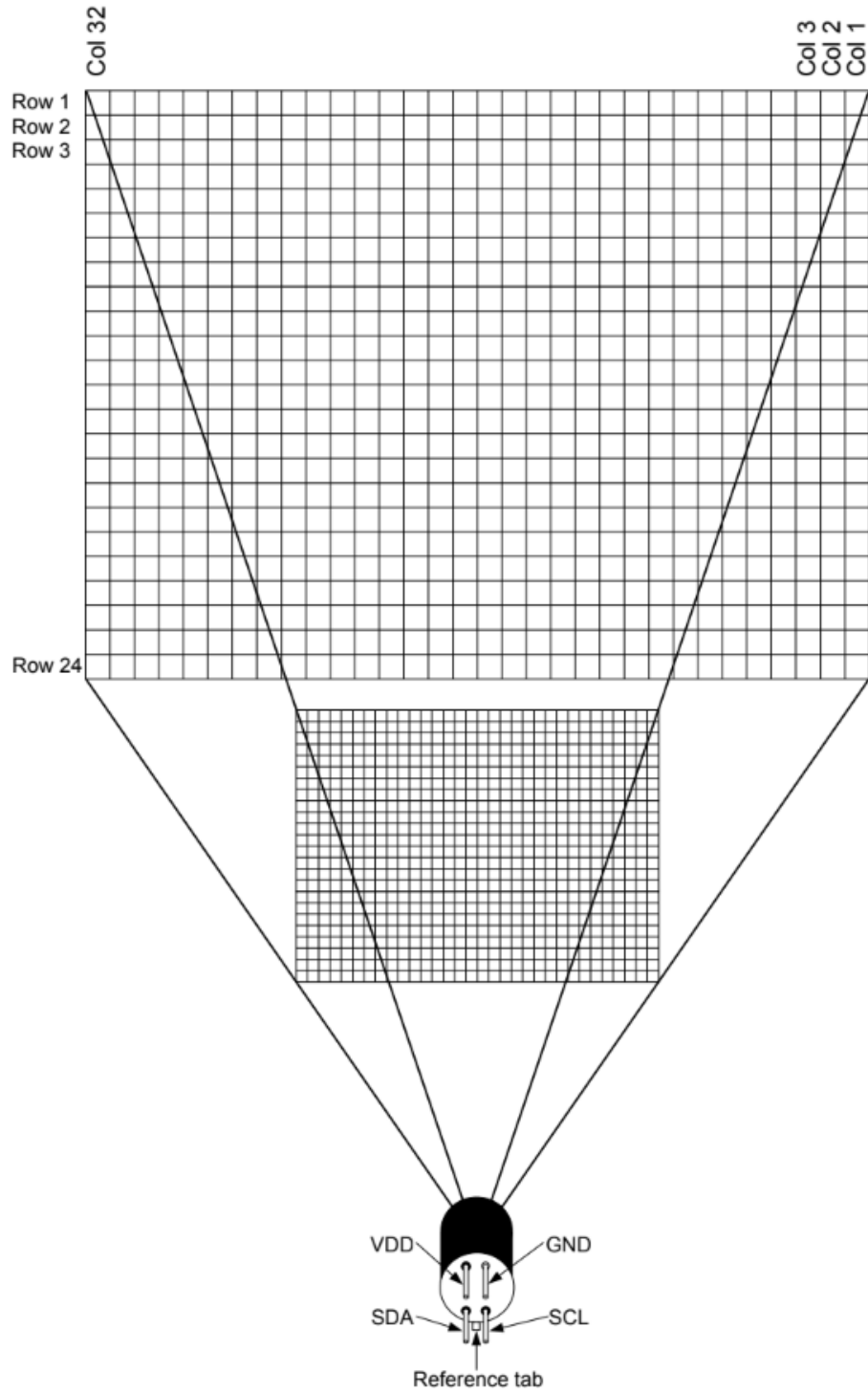


Figure 4.1: Pixel structure definition of the MLX90640BAA

The communication protocol of the MLX90640 is I2C which supports up to Fast Mode Plus (1 MHz)[18]. Frame reading consists of two subpages, each containing information regarding the 768 pixels plus housekeeping and address data. Adding this up, each subpage generates 834 words of 16 bits/2 bytes[5]. The I2C channel is set at Fast Mode Plus which is 1 MHz. To arrive at the time per frame Equation 4.1 is used:

$$\frac{16 \text{ bits} \cdot 834 \text{ words}}{10^6 \text{ bits/second}} = \frac{13344 \text{ bits}}{10^6 \text{ bits/second}} = 0.013344 \text{ seconds} = 13.344 \text{ ms} \quad (4.1)$$

So a frame would take just above 13 ms. However, a margin is taken to account for overhead processing time, protocol cycles and bus capacitance so the time given per frame is 19 ms. Therefore, when two sensors are connected on the same I2C port it sits well under the 125 ms time window before a new frame is generated as to allow the algorithm to have enough time to process each frame.

4.2 Magnetorquer

In the design process of the magnetorquer, the previous Delfi-PQ design was reviewed and two areas for improvement were identified. The first concerns the magnetorquer design itself, while the second concerns its production. The previous Delfi-PQ magnetorquer was limited in scope partly due to the lack of meaningful production capabilities. With an entire quarter available for the Delfi-Twin project, it is expected that a more refined magnetorquer design can be developed given sufficient preparation and consideration of manufacturability from the beginning.

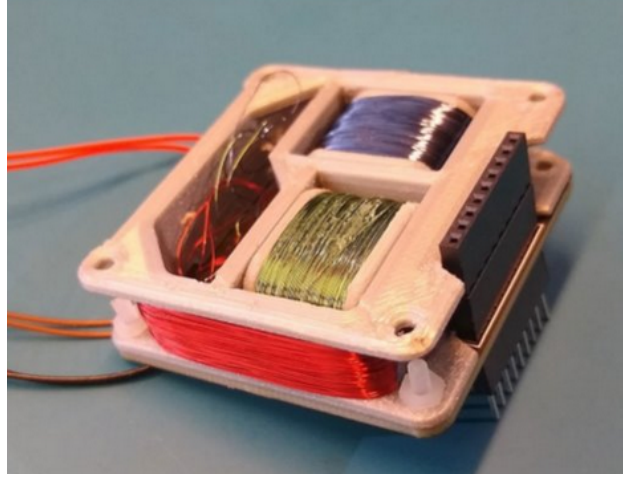


Figure 4.2: Delfi-PQ Magnetorquer (3x aircoils)

4.2.1 Magnetorquer Design and Analysis

Overall, three types of magnetorquers were considered for actuation. For a pocket satellite configuration, air-core coils, solenoids (with core), and PCB-based magnetorquers were analyzed. Each type exhibits distinct characteristics, such as performance and speed, which strongly depend on the available power and volume. As each type comes with its own volume based trade-offs, a matlab script was written to plot all theoretically feasible designs at a fixed voltage to then allow the selection of a specific design of interest.

Aircoil Magnetorquer

In short, an aircoil magnetorquer is just a conductive cable that loops around an open area in the center. It is wrapped around a non-magnetic structure to be held in place within the satellite.

Characteristics: Performance

The magnetic dipole m is defined as:

$$m = NIA \quad (4.2)$$

In this equation, the magnetic dipole m results from a current I flowing through N loops enclosing an area A . Under a fixed maximum volume constraint, this introduces a trade-off: increasing the number of loops

reduces the available core area. This leads to two limiting cases in which the magnetic dipole approaches zero: either no loops are present while the area A is maximal, or the number of loops is maximized such that no free area remains.

In practice, the analysis aims to maximize the magnetic flux over the largest possible area. To aid understanding, the regions between adjacent loops inside the coil effectively cancel out due to opposing magnetic fields, as the magnetic field circulates around the conductor according to the right-hand rule. To tie the magnetic dipole into the design, it is divided by the power consumption of the chosen design, which yields the performance used for this analysis.

Constraints: Electrical based on PWR-1, PWR-7, CON-2 and MTQ-2:

$$I_{AC} = \frac{U_{supply}}{R_{coil}} \quad (4.3)$$

$$U_{supply} = 3.3 \text{ V} \quad (4.4)$$

$$0 < I_{AC} < \frac{I_{max}}{3} \quad (4.5)$$

$$I_{max} = 0.5 \text{ A} \quad (4.6)$$

(4.5) Defines the limits of how much current is allowed to flow in the AC.

$$P_{AC} = U_{supply} I_{AC} \quad (4.7)$$

$$0 < P_{AC} < P_{max}/3 = 0.6[W] \quad (4.8)$$

(4.8) Defines how much power the AC is allowed to draw

Constraints: Volume based on CON-1

Working with a rectangular volume with side length W_{max} and height H_{max} , the free area A is maximized by enclosing a rectangular surface. As a consequence, the air-core coil is assumed to have a rectangular shape, where the side length of the free area is denoted by x_1 . The coil stack is characterized by the coil thickness x_2 , which is constrained by:

$$x_1 = \sqrt{A} \quad (4.9)$$

$$x_2 = \frac{W_{max} - x_1}{2} \quad (4.10)$$

$$0 < x_1 < W_{max} \quad (4.11)$$

$$0 < x_2 < \frac{W_{max}}{2} \quad (4.12)$$

For the estimation of the number of loops, the coil thickness x_2 is multiplied by the available coil height H_{max} and divided by D_{cable}^2 , assuming loose packing. In this approximation, the cables are arranged on a cubic grid, which reduces the packing density but provides a conservative lower-bound estimate for a first-order analysis. Lastly, since this serves for an analytical approach, fractional loops are tolerated.

$$N = \frac{H_{max} x_2}{D_{cable}^2} \quad (4.13)$$

Design Dependent Parameter: Resistance

As evident in (4.3), the resistance R_{coil} is not fixed. Considering 4.7, and the fixed voltage, the current can be freely chosen for any design as long as it respects the given constraints. Based on the cable resistivity ρ , the cable length L_{cable} , and its cross-sectional area A_{cable} , the coil resistance R_{coil} is defined as:

$$R_{coil} = \rho \frac{L_{cable}}{A_{cable}} \quad (4.14)$$

$$L_{cable} = 4N(x_1 + x_2) \quad (4.15)$$

$$A_{cable} = \pi \left(\frac{D_{cable}}{2} \right)^2 \quad (4.16)$$

Where (4.15) approximates the total cable length by multiplying the number of loops with the perimeter of the midsection inside the coil area.

Characteristics: Speed

For this analysis, the speed is defined as the sum of the rise time from 10% to 90% of the steady-state current and the fall time from 90% to 10% after switch-off. To determine the rise time, the instants at which the current reaches 10% and 90% of its steady-state value are evaluated by the dynamic response $I(t)$ which depends on a time constant τ that is a function of inductance L and resistance R_{coil} :

$$I(t) = I_{\infty} \left(1 - e^{-t/\tau}\right) \quad (4.17)$$

$$\tau = \frac{L}{R_{\text{coil}}} \quad (4.18)$$

$$L = \mu_0 \frac{N^2 A}{l_{\text{eff}}} \quad (4.19)$$

$$l_{\text{eff}} = 4(x_1 + x_2) \quad (4.20)$$

(4.20) Defined as a rough approximation for the magnetic path length around the AC using the midsection.

This time constant determines how quickly the current, and therefore the magnetic dipole, can build up or decay. The current response is defined by the steady-state current I_{∞} , time t and characteristic time τ :

$$f \quad (4.21)$$

At 10% of the steady-state current:

$$I(t_{10}) = 0.1I_{\infty} \quad (4.22)$$

$$I_{\infty} \left(1 - e^{-t_{10}/\tau}\right) = 0.1I_{\infty} \quad (4.23)$$

$$1 - e^{-t_{10}/\tau} = 0.1 \quad (4.24)$$

$$e^{-t_{10}/\tau} = 0.9 \quad (4.25)$$

$$-\frac{t_{10}}{\tau} = \ln(0.9) \quad (4.26)$$

$$t_{10} = -\tau \ln(0.9) \quad (4.27)$$

At 90% of the steady-state current:

$$I(t_{90}) = 0.9I_{\infty} \quad (4.28)$$

$$I_{\infty} \left(1 - e^{-t_{90}/\tau}\right) = 0.9I_{\infty} \quad (4.29)$$

$$1 - e^{-t_{90}/\tau} = 0.9 \quad (4.30)$$

$$e^{-t_{90}/\tau} = 0.1 \quad (4.31)$$

$$-\frac{t_{90}}{\tau} = \ln(0.1) \quad (4.32)$$

$$t_{90} = -\tau \ln(0.1) \quad (4.33)$$

The rise time from 10% to 90% is therefore:

$$t_{\text{rise}} = t_{90} - t_{10} \quad (4.34)$$

$$= -\tau \ln(0.1) - (-\tau \ln(0.9)) \quad (4.35)$$

$$= \tau [\ln(0.9) - \ln(0.1)] \quad (4.36)$$

$$= \tau \ln\left(\frac{0.9}{0.1}\right) \quad (4.37)$$

$$= \tau \ln(9) \quad (4.38)$$

Since $\ln(9) \approx 2.197$, this yields:

$$t_{\text{rise}} \approx 2.2\tau \quad (4.39)$$

After switch-off, the current decays according to:

$$I(t) = I_0 e^{-t/\tau} \quad (4.40)$$

Assuming the current starts at I_0 , the times at which the current reaches 90% and 10% of its initial value are obtained in the same way.

At 90% of the initial current:

$$I(t_{90}) = 0.9I_0 \quad (4.41)$$

$$I_0 e^{-t_{90}/\tau} = 0.9I_0 \quad (4.42)$$

$$e^{-t_{90}/\tau} = 0.9 \quad (4.43)$$

$$-\frac{t_{90}}{\tau} = \ln(0.9) \quad (4.44)$$

$$t_{90} = -\tau \ln(0.9) \quad (4.45)$$

At 10% of the initial current:

$$I(t_{10}) = 0.1I_0 \quad (4.46)$$

$$I_0 e^{-t_{10}/\tau} = 0.1I_0 \quad (4.47)$$

$$e^{-t_{10}/\tau} = 0.1 \quad (4.48)$$

$$-\frac{t_{10}}{\tau} = \ln(0.1) \quad (4.49)$$

$$t_{10} = -\tau \ln(0.1) \quad (4.50)$$

Thus, the fall time from 90% to 10% is:

$$t_{\text{fall}} = t_{10} - t_{90} \quad (4.51)$$

$$= -\tau \ln(0.1) - (-\tau \ln(0.9)) \quad (4.52)$$

$$= \tau \ln\left(\frac{0.9}{0.1}\right) \quad (4.53)$$

$$= \tau \ln(9) \quad (4.54)$$

and therefore:

$$t_{\text{fall}} \approx 2.2\tau \quad (4.55)$$

The total speed metric used in this analysis is then defined as:

$$t_{\text{speed}} = t_{\text{rise}} + t_{\text{fall}} \quad (4.56)$$

$$= \tau \ln(9) + \tau \ln(9) \quad (4.57)$$

$$= 2\tau \ln(9) \quad (4.58)$$

which simplifies numerically to:

$$t_{\text{speed}} \approx 4.4\tau \quad (4.59)$$

Characteristics: Mass

For each design, a specific mass emerges depending on the amount of coil material used. Since copper is the best conductor after silver, and the most economic, it is used and defined as:

$$m_{\text{coil}} = \rho_{\text{copper}} L_{\text{cable}} A_{\text{cable}} \quad (4.60)$$

where ρ_{density} is the density of copper, L_{cable} is the total cable length, and A_{cable} is the cable cross-sectional area.

Solenoid Magnetorquer (with core)

In short, a solenoid magnetorquer is a long coil wrapped around a ferromagnetic core to enhance the magnetic flux. For a solenoid's core to contribute to the existing magnetic flux, the following ratio needs to hold: $L_{\text{core}}/W_{\text{max}} = [5 - 10]$. Otherwise, the alignment of elementary dipoles inside the core's material is too weak and loses its amplifying properties.

Characteristics: Performance

For this analysis, the magnetic dipole of the solenoid is defined as:

$$m = \mu_{\text{eff}} N I A_{\text{core}} \quad (4.61)$$

In this equation, the magnetic dipole m results from a current I flowing through N loops around a magnetic core with cross-sectional area A_{core} . Compared to the aircoil, the additional factor μ_{eff} accounts for the amplification introduced by the core material.

Under a fixed maximum width constraint, this again introduces a trade-off. Increasing the core radius enlarges the core area A_{core} , which is beneficial for the magnetic dipole. At the same time, however, it reduces the space available for the coil winding, which lowers the number of loops N . Conversely, reducing the core radius leaves more room for the winding and increases N , but then the enclosed core area becomes very small. This leads to two limiting cases in which the magnetic dipole again approaches zero: either the core becomes vanishingly small while the number of loops is large, or the core occupies almost the full width such that no meaningful winding space remains.

Constraints: Electrical based on PWR-1, PWR-7, CON-2 and MTQ-2:

$$I_{\text{SOL}} = \frac{U_{\text{supply}}}{R_{\text{coil}}} \quad (4.62)$$

$$U_{\text{supply}} = 3.3 \text{ V} \quad (4.63)$$

$$0 < I_{\text{SOL}} < I_{\text{max}} \quad (4.64)$$

$$I_{\text{max}} = 0.5 \text{ A} \quad (4.65)$$

(4.64) defines the limits of how much current is allowed to flow through the solenoid.

$$P_{\text{SOL}} = U_{\text{supply}} I_{\text{SOL}} \quad (4.66)$$

$$0 < P_{\text{SOL}} < P_{\text{max}} \quad (4.67)$$

(4.67) defines how much power the solenoid is allowed to draw.

Constraints: Volume based on CON-1

The solenoid is approximated as a cylindrical core of radius r_{core} and length L_{core} , surrounded by a coil of thickness t_{coil} . The total radial build-up must fit within the available width W_{max} . Since the full outer width consists of the core diameter plus twice the coil thickness on both sides, the geometry is constrained by:

$$A_{\text{core}} = \pi r_{\text{core}}^2 \quad (4.68)$$

$$t_{\text{coil}} = \frac{W_{\text{max}} - 2r_{\text{core}}}{2} \quad (4.69)$$

$$0 < r_{\text{core}} < \frac{W_{\text{max}}}{2} \quad (4.70)$$

$$0 < t_{\text{coil}} < \frac{W_{\text{max}}}{2} \quad (4.71)$$

For the estimation of the number of loops, the available winding cross-section is approximated by the product of the coil thickness t_{coil} and the core length L_{core} , divided by the cable diameter squared. As for the aircoil, loose packing on a cubic grid is assumed, which provides a conservative lower-bound estimate for a first-order analytical model. Fractional loops are again tolerated in this approximation.

$$N = \frac{t_{\text{coil}} L_{\text{core}}}{D_{\text{cable}}^2} \quad (4.72)$$

Design Dependent Parameter: Resistance

As evident in (4.62), the resistance R_{coil} is not fixed. Because the voltage is fixed, the current follows directly from the resistance of the chosen design and is only limited by the imposed electrical constraints. Based on the cable resistivity ρ , the cable length L_{cable} , and its cross-sectional area A_{cable} , the coil resistance is defined as:

$$R_{\text{coil}} = \rho \frac{L_{\text{cable}}}{A_{\text{cable}}} \quad (4.73)$$

$$L_{\text{cable}} = 2\pi \left(r_{\text{core}} + \frac{t_{\text{coil}}}{2} \right) N \quad (4.74)$$

$$A_{\text{cable}} = \pi \left(\frac{D_{\text{cable}}}{2} \right)^2 \quad (4.75)$$

Here, (4.74) approximates the total cable length by multiplying the number of loops with the circumference at the mid-radius of the winding region.

Characteristics: Speed

For this analysis, the speed is again defined as the sum of the rise time from 10% to 90% of the steady-state current and the fall time from 90% to 10% after switch-off. The transient response of the solenoid is governed by the inductance L and resistance R_{coil} , which together define the electrical time constant:

$$I(t) = I_{\infty} \left(1 - e^{-t/\tau}\right) \quad (4.76)$$

$$\tau = \frac{L}{R_{\text{coil}}} \quad (4.77)$$

$$L = \mu_0 \mu_{\text{eff}} \frac{N^2 A_{\text{core}}}{L_{\text{core}}} \quad (4.78)$$

In contrast to the aircoil, the inductance of the solenoid depends explicitly on the magnetic permeability of the core material through μ_{eff} , which can significantly increase the inductance and therefore slow down the current response.

This time constant determines how quickly the current, and therefore the magnetic dipole, can build up or decay.

At 10% of the steady-state current:

$$I(t_{10}) = 0.1 I_{\infty} \quad (4.79)$$

$$I_{\infty} \left(1 - e^{-t_{10}/\tau}\right) = 0.1 I_{\infty} \quad (4.80)$$

$$1 - e^{-t_{10}/\tau} = 0.1 \quad (4.81)$$

$$e^{-t_{10}/\tau} = 0.9 \quad (4.82)$$

$$-\frac{t_{10}}{\tau} = \ln(0.9) \quad (4.83)$$

$$t_{10} = -\tau \ln(0.9) \quad (4.84)$$

At 90% of the steady-state current:

$$I(t_{90}) = 0.9 I_{\infty} \quad (4.85)$$

$$I_{\infty} \left(1 - e^{-t_{90}/\tau}\right) = 0.9 I_{\infty} \quad (4.86)$$

$$1 - e^{-t_{90}/\tau} = 0.9 \quad (4.87)$$

$$e^{-t_{90}/\tau} = 0.1 \quad (4.88)$$

$$-\frac{t_{90}}{\tau} = \ln(0.1) \quad (4.89)$$

$$t_{90} = -\tau \ln(0.1) \quad (4.90)$$

The rise time from 10% to 90% is therefore:

$$t_{\text{rise}} = t_{90} - t_{10} \quad (4.91)$$

$$= -\tau \ln(0.1) - (-\tau \ln(0.9)) \quad (4.92)$$

$$= \tau \ln\left(\frac{0.9}{0.1}\right) \quad (4.93)$$

$$= \tau \ln(9) \quad (4.94)$$

Since $\ln(9) \approx 2.197$, this yields:

$$t_{\text{rise}} \approx 2.2\tau \quad (4.95)$$

After switch-off, the current decays according to:

$$I(t) = I_0 e^{-t/\tau} \quad (4.96)$$

At 90% of the initial current:

$$I(t_{90}) = 0.9I_0 \quad (4.97)$$

$$I_0 e^{-t_{90}/\tau} = 0.9I_0 \quad (4.98)$$

$$e^{-t_{90}/\tau} = 0.9 \quad (4.99)$$

$$-\frac{t_{90}}{\tau} = \ln(0.9) \quad (4.100)$$

$$t_{90} = -\tau \ln(0.9) \quad (4.101)$$

At 10% of the initial current:

$$I(t_{10}) = 0.1I_0 \quad (4.102)$$

$$I_0 e^{-t_{10}/\tau} = 0.1I_0 \quad (4.103)$$

$$e^{-t_{10}/\tau} = 0.1 \quad (4.104)$$

$$-\frac{t_{10}}{\tau} = \ln(0.1) \quad (4.105)$$

$$t_{10} = -\tau \ln(0.1) \quad (4.106)$$

Thus, the fall time from 90% to 10% is:

$$t_{\text{fall}} = t_{10} - t_{90} \quad (4.107)$$

$$= -\tau \ln(0.1) - (-\tau \ln(0.9)) \quad (4.108)$$

$$= \tau \ln\left(\frac{0.9}{0.1}\right) \quad (4.109)$$

$$= \tau \ln(9) \quad (4.110)$$

and therefore:

$$t_{\text{fall}} \approx 2.2\tau \quad (4.111)$$

The total speed metric used in this analysis is then defined as:

$$t_{\text{speed}} = t_{\text{rise}} + t_{\text{fall}} \quad (4.112)$$

$$= \tau \ln(9) + \tau \ln(9) \quad (4.113)$$

$$= 2\tau \ln(9) \quad (4.114)$$

which simplifies numerically to:

$$t_{\text{speed}} \approx 4.4\tau \quad (4.115)$$

Characteristics: Mass

For each design, a specific mass emerges depending on both the amount of copper used for the winding and the amount of magnetic core material used. The total solenoid mass is therefore defined as the sum of both contributions:

$$m_{\text{SOL}} = m_{\text{copper}} + m_{\text{core}} \quad (4.116)$$

$$m_{\text{copper}} = \rho_{\text{density,copper}} L_{\text{cable}} A_{\text{cable}} \quad (4.117)$$

$$m_{\text{core}} = \rho_{\text{density,core}} L_{\text{core}} A_{\text{core}} \quad (4.118)$$

where $\rho_{\text{density,copper}}$ is the density of copper, $\rho_{\text{density,core}}$ is the density of the magnetic core material, L_{cable} is the total cable length, A_{cable} is the cable cross-sectional area, and A_{core} is the cross-sectional area of the cylindrical core.

Compared to the aircoil, the solenoid therefore carries an additional mass penalty due to the core. This is the price paid for the gain in magnetic amplification and has to be considered in the overall trade-off.

PCB Magnetorquer

In short, a PCB magnetorquer is a planar spiral coil manufactured directly in the copper layers of a printed circuit board. In this configuration, multiple spiral layers are stacked through the PCB thickness and electrically connected in series.

Characteristics: Performance

For the PCB magnetorquer, the magnetic dipole is defined as:

$$m = N_{\text{total}} I_{\text{PCB}} A_{\text{free}} \quad (4.119)$$

In this equation, the magnetic dipole m results from a current I_{PCB} flowing through a total number of turns N_{total} , enclosing the inner free area A_{free} . Since the layers are connected in series, the total number of loops becomes:

$$N_{\text{total}} = N_{\text{layer}} N_{\text{LPL}} \quad (4.120)$$

where N_{layer} is the number of PCB layers and N_{LPL} is the number of loops per layer.

As for the other magnetorquer concepts, a geometric trade-off appears. Increasing the number of loops per layer increases the total number of turns, which is beneficial for the magnetic dipole. At the same time, however, the free inner area enclosed by the spiral becomes smaller. This again leads to two limiting cases in which the magnetic dipole approaches zero: either only very few loops are present while the free area is large, or the spiral is wound so far inward that the free area vanishes.

Constraints: Electrical based on PWR-1, PWR-7, CON-2 and MTQ-2:

$$I_{\text{PCB}} = \frac{U_{\text{supply}}}{R_{\text{eq}}} \quad (4.121)$$

$$U_{\text{supply}} = 3.3 \text{ V} \quad (4.122)$$

$$0 < I_{\text{PCB}} < I_{\text{max}} \quad (4.123)$$

$$I_{\text{max}} = 0.5 \text{ A} \quad (4.124)$$

(4.123) defines the limit of how much current is allowed to flow through the series-connected PCB spiral.

$$P_{\text{PCB}} = U_{\text{supply}} I_{\text{PCB}} \quad (4.125)$$

$$0 < P_{\text{PCB}} < P_{\text{max}} \quad (4.126)$$

(4.126) defines how much power the PCB magnetorquer is allowed to draw.

Constraints: Volume and Manufacturing based on CON-1

The PCB spiral is confined to a rectangular board area with outer dimensions $L_{\max}$ and $W_{\max}$, while a free edge margin d_{free} is maintained on all sides. The copper spiral is defined by the trace width w_t , the trace spacing w_s , and the number of loops per layer N_{LPL} .

The total radial build-up of one side of the spiral is defined as:

$$d_{t+s} = N_{\text{LPL}}(w_t + w_s) \quad (4.127)$$

such that the remaining free inner dimensions become:

$$W_{\text{free}} = W_{\max} - 2d_{\text{free}} - 2d_{t+s} \quad (4.128)$$

$$L_{\text{free}} = L_{\max} - 2d_{\text{free}} - 2d_{t+s} \quad (4.129)$$

and therefore the enclosed free inner area is:

$$A_{\text{free}} = W_{\text{free}}L_{\text{free}} \quad (4.130)$$

To remain physically meaningful, the free inner dimensions must remain positive:

$$W_{\text{free}} > 0 \quad (4.131)$$

$$L_{\text{free}} > 0 \quad (4.132)$$

$$A_{\text{free}} > 0 \quad (4.133)$$

In addition, the innermost spiral segments must remain positive. This prevents the analytical spiral from winding further inward than the available rectangular area permits.

Unlike the aircoil and solenoid models, the geometric design variable is here chosen directly as N_{LPL} . Since this serves as a first-order analytical sweep, fractional values of N_{LPL} are tolerated. This allows the design space to be explored continuously rather than only at discrete integer turn counts, which makes the trend behavior in the optimization clearer.

Design Dependent Parameter: Trace Length and Resistance

As evident in (4.121), the equivalent resistance R_{eq} is not fixed, but emerges from the chosen PCB geometry. It depends on the trace length, the trace cross-section, and the number of series-connected layers.

For one single rectangular loop pair, the available horizontal and vertical trace lengths shrink by twice the pitch ($w_t + w_s$) each time the spiral steps inward. Therefore, the total copper trace length of one layer can be written as the sum over all loop segments:

$$L_t = \sum_{i=1}^{2N_{\text{LPL}}} [(L_{\max} - 2d_{\text{free}} - 2(i-1)(w_t + w_s)) + (W_{\max} - 2d_{\text{free}} - 2(i-1)(w_t + w_s))] \quad (4.134)$$

This expression sums all horizontal and vertical segment lengths of the spiral layer. Since it is a finite arithmetic sum, it can be rewritten analytically. Expanding the summation gives:

$$L_t = \sum_{i=1}^{2N_{\text{LPL}}} (L_{\max} + W_{\max} - 4d_{\text{free}} - 4(i-1)(w_t + w_s)) \quad (4.135)$$

$$= 2N_{\text{LPL}}(L_{\max} + W_{\max} - 4d_{\text{free}}) - 4(w_t + w_s) \sum_{i=1}^{2N_{\text{LPL}}} (i-1) \quad (4.136)$$

Using the standard result

$$\sum_{i=1}^n (i-1) = \frac{n(n-1)}{2} \quad (4.137)$$

with $n = 2N_{\text{LPL}}$, this becomes:

$$L_t = 2N_{\text{LPL}}(L_{\text{max}} + W_{\text{max}} - 4d_{\text{free}}) - 4(w_t + w_s) \frac{(2N_{\text{LPL}})(2N_{\text{LPL}} - 1)}{2} \quad (4.138)$$

$$= 2N_{\text{LPL}}(L_{\text{max}} + W_{\text{max}} - 4d_{\text{free}}) - 2(w_t + w_s)(2N_{\text{LPL}})(2N_{\text{LPL}} - 1) \quad (4.139)$$

Thus, the analytical expression for the trace length of one PCB layer is:

$$L_t = 2N_{\text{LPL}}(L_{\text{max}} + W_{\text{max}} - 4d_{\text{free}}) - 2(w_t + w_s)(2N_{\text{LPL}})(2N_{\text{LPL}} - 1) \quad (4.140)$$

This closed-form expression is particularly useful because it allows N_{LPL} to be swept continuously in the analysis, without forcing the model into a purely discrete formulation. This is why the optimization script can evaluate fractional values of N_{LPL} in a smooth design sweep.

The trace cross-sectional area is defined as:

$$A_t = w_t t_t \quad (4.141)$$

where t_t is the copper trace thickness.

The resistance of one layer then follows as:

$$R_{\text{layer}} = \rho \frac{L_t}{A_t} \quad (4.142)$$

Since all layers are connected in series, the equivalent resistance becomes:

$$R_{\text{eq}} = N_{\text{layer}} R_{\text{layer}} \quad (4.143)$$

It follows that increasing the number of layers increases both the total number of turns and the total resistance. The current is therefore not independently chosen, but emerges directly from the fixed supply voltage and the resulting equivalent resistance.

Characteristics: Speed

For this analysis, the speed is again defined as the sum of the rise time from 10% to 90% of the steady-state current and the fall time from 90% to 10% after switch-off. The transient response depends on the equivalent inductance L_{eq} and equivalent resistance R_{eq} , which define the electrical time constant:

$$I(t) = I_{\infty} \left(1 - e^{-t/\tau}\right) \quad (4.144)$$

$$\tau = \frac{L_{\text{eq}}}{R_{\text{eq}}} \quad (4.145)$$

$$L_{\text{eq}} = \mu_0 \frac{N_{\text{total}}^2 A_{\text{eff}}}{L_{\text{eff}}} \quad (4.146)$$

In this first-order approximation, the effective enclosed area is taken as the free inner area:

$$A_{\text{eff}} = A_{\text{free}} \quad (4.147)$$

and the effective magnetic path length is approximated by the perimeter of the free inner window:

$$L_{\text{eff}} = 2(W_{\text{free}} + L_{\text{free}}) \quad (4.148)$$

This time constant determines how quickly the current, and therefore the magnetic dipole, can build up or decay.

At 10% of the steady-state current:

$$I(t_{10}) = 0.1 I_{\infty} \quad (4.149)$$

$$I_{\infty} \left(1 - e^{-t_{10}/\tau}\right) = 0.1 I_{\infty} \quad (4.150)$$

$$1 - e^{-t_{10}/\tau} = 0.1 \quad (4.151)$$

$$e^{-t_{10}/\tau} = 0.9 \quad (4.152)$$

$$-\frac{t_{10}}{\tau} = \ln(0.9) \quad (4.153)$$

$$t_{10} = -\tau \ln(0.9) \quad (4.154)$$

At 90% of the steady-state current:

$$I(t_{90}) = 0.9 I_{\infty} \quad (4.155)$$

$$I_{\infty} \left(1 - e^{-t_{90}/\tau}\right) = 0.9 I_{\infty} \quad (4.156)$$

$$1 - e^{-t_{90}/\tau} = 0.9 \quad (4.157)$$

$$e^{-t_{90}/\tau} = 0.1 \quad (4.158)$$

$$-\frac{t_{90}}{\tau} = \ln(0.1) \quad (4.159)$$

$$t_{90} = -\tau \ln(0.1) \quad (4.160)$$

The rise time from 10% to 90% is therefore:

$$t_{\text{rise}} = t_{90} - t_{10} \quad (4.161)$$

$$= \tau \ln \left(\frac{0.9}{0.1} \right) \quad (4.162)$$

$$= \tau \ln(9) \quad (4.163)$$

Thus:

$$t_{\text{rise}} \approx 2.2\tau \quad (4.164)$$

After switch-off, the current decays according to:

$$I(t) = I_0 e^{-t/\tau} \quad (4.165)$$

Repeating the same derivation yields:

$$t_{\text{fall}} \approx 2.2\tau \quad (4.166)$$

The total speed metric used in this analysis is then:

$$t_{\text{speed}} = t_{\text{rise}} + t_{\text{fall}} \quad (4.167)$$

$$= 2\tau \ln(9) \quad (4.168)$$

which simplifies numerically to:

$$t_{\text{speed}} \approx 4.4\tau \quad (4.169)$$

Characteristics: Mass

For each design, the total mass consists of both the copper trace mass and the remaining FR4 board mass. The copper volume of all series-connected layers is defined as:

$$V_{\text{Cu,total}} = N_{\text{layer}} L_t A_t \quad (4.170)$$

The total PCB volume is approximated as:

$$V_{\text{PCB,total}} = W_{\text{max}} L_{\text{max}} H_{\text{max}} \quad (4.171)$$

and therefore the remaining FR4 volume becomes:

$$V_{\text{FR4}} = V_{\text{PCB,total}} - V_{\text{Cu,total}} \quad (4.172)$$

The corresponding masses are then:

$$M_{\text{trace}} = \rho_{\text{density,copper}} V_{\text{Cu,total}} \quad (4.173)$$

$$M_{\text{PCB}} = \rho_{\text{density,FR4}} V_{\text{FR4}} \quad (4.174)$$

$$M_{\text{total}} = M_{\text{trace}} + M_{\text{PCB}} \quad (4.175)$$

Compared to the cable-wound concepts, the PCB magnetorquer therefore includes both conductor mass and structural substrate mass. Increasing the number of layers or the trace width generally increases the copper volume and can improve the magnetic dipole, but this also increases the total mass and changes the electrical resistance.

Magnetorquer Matlab Analysis

Based on the defined constraints and variables, three MATLAB scripts were written to find the ideal design parameters of different cable diameters D_{cable} (trace widths w_t for PCB). For each design, the supply voltage was fixed at $U_{supply} = 3.3V$, and the free inner dimension x_1 for the aircoil, t_{coil} for the solenoid (With core), and N_{LPL} for the PCB was swept continuously over its entire feasible range. For each value, a singular, corresponding design emerged for the different input sizes.

For each design, the resulting electrical and geometric constraints were checked, and disregarded if they were unfeasible. Only feasible, realistic designs were kept and plotted on an $m(P)$ diagram, which generated the achievable magnetic dipole for each input variable, cable diameter D_{cable} or trace length w_{trace} . Each point on $m(P)$ ended up representing a unique feasible design geometry.

To aid with the design selection, an interactive element was introduced in which a desired power level P_{sel} could be selected. Within this selection, only feasible plots were further investigated by generating additional diagrams regarding characteristic parameters and design sizes.

For this selected design, detailed plots were generated showing the variation of core area A and cable length L_{cable} , response time τ_{R+F} and total mass M , as well as number of loops N and coil thickness x_2 as a function of cable diameter.

Furthermore, torque plots were produced for the available designs at P_{sel} , showing the achievable torque over the angle θ for representative magnetic field strengths at 300 km and 600 km altitude. These plots include both minimum and maximum expected magnetic field conditions, allowing an assessment of the actuation capability across the orbit.

Magnetorquer Trade-Off analysis

For the trade-off, six categories of magnetorquers were considered:

- z-Torquer - Aircoil
- z-Torquer - Solenoid (with core)¹
- z-Torquer - PCB
- xy-Torquer - Aircoil
- xy-Torquer - Solenoid (with core)
- xy-Torquer - PCB

In the first step, each category was sized to meet the aerodynamic disturbance torque of $3.15 \cdot 10^{-7} [Nm] = \frac{\rho_{air@300km} \cdot C_d \cdot A_a \cdot V^2 \cdot (c_{pa} - c_{pm})}{2} = \frac{3.2 \cdot 10^{-11} [kg/m^3] \cdot 1.5 \cdot 10^{-2} [m^2] \cdot 5.97 \cdot 10^7 \cdot 0.01}{2}$ by a multiplicative margin of $2\sqrt{2}$ (based on average of sin-wave and an additional safety factor of 2), resulting in a targeted torque of: $\approx 9 \cdot 10^{-7} [Nm]$. If multiple designs achieved the same magnetic dipole at different power levels, the less power hungry design was chosen as the category candidate design.

In a second step, it was determined whether the chosen candidates were respecting rise and fall time (MTQ-04), designs that violated either were rejected.

Lastly, each z-category candidate was paired with two x-category candidates to meet mass (CON-3) and peak power (MTQ-02) requirements. With only one combination surviving all constraints and requirements, the z-torquer aircoil was paired with the xy-torquer PCBs as the final magnetorquer design configuration.

¹Excluded: the z-axis height constraint of 5 [mm] does not allow a feasible solenoid design with a realistic length-to-diameter ratio ($L/D \approx 5-10$).

z-Torquer: Aircoil

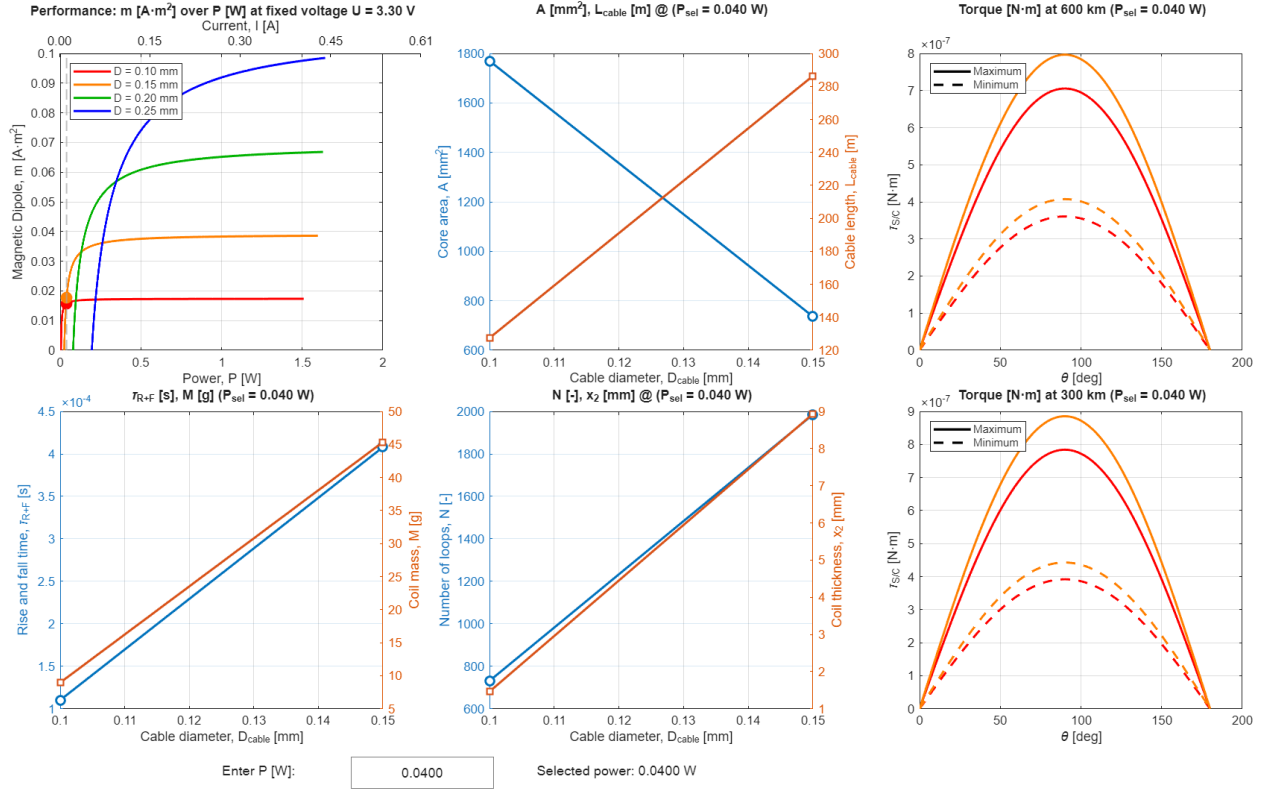


Figure 4.3: Aircoil magnetorquer design for $W_{max} = 45[mm]$ and $H_{max} = 5[mm]$, achieving $9 \cdot 10^{-7}[Nm]$ @ $P_{sel} = 0.04[W]$ @ $300[km]$ for $D_{cable} = 0.15[mm]$ with a total mass of $M = 45[g]$ and rise and fall time $\tau_{R+F} = 4 \cdot 10^{-4}[s]$

In conclusion, this design is feasible and well within the constraints and requirements imposed on this component.

z-Torquer: PCB

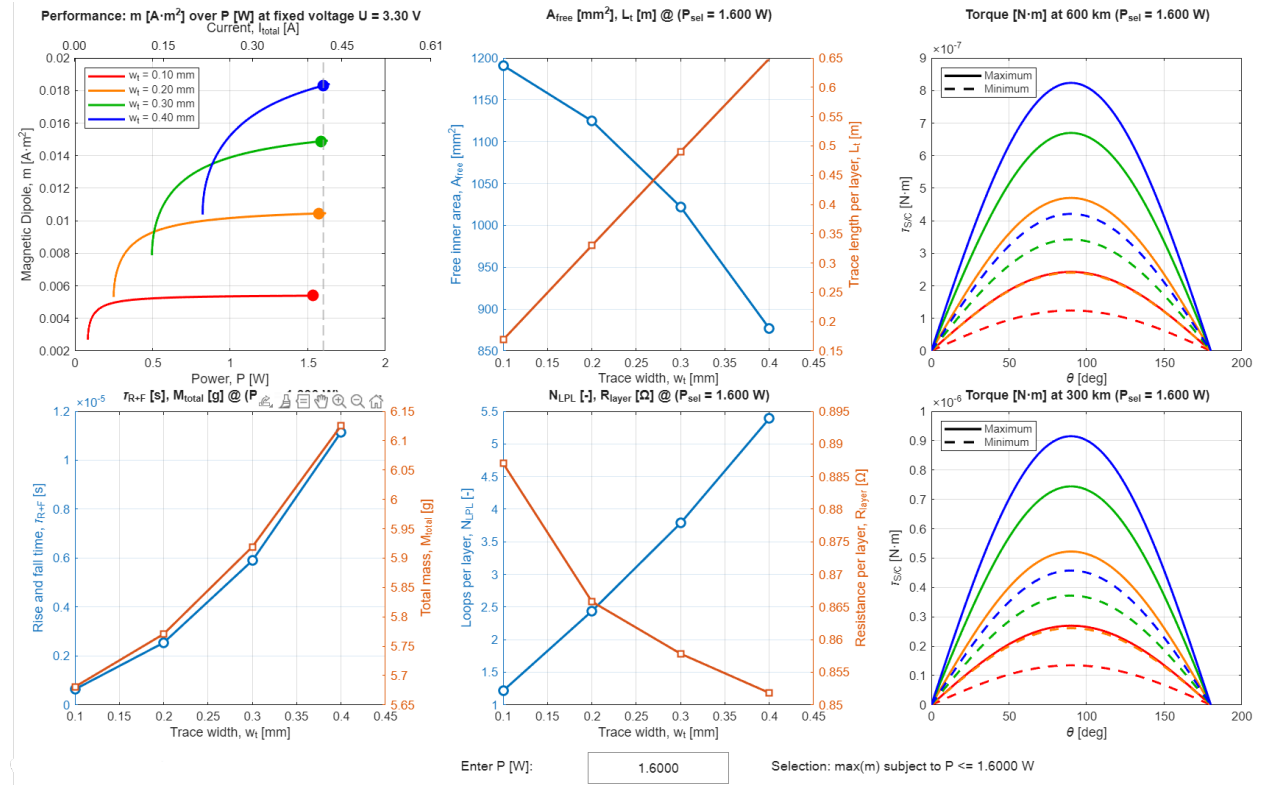


Figure 4.4: PCB magnetorquer design for $W_{max} = L_{max} = 45[mm]$, $N_{Layer} = 8$ and $H_{max} = 1.55[mm]$, achieving $9 \cdot 10^{-7}[Nm]$ @ $P_{sel} = 1.5[W]$ @ $300[km]$ for $w_{trace} = 0.4[mm]$

In conclusion, this design is unfeasible due to high power consumption. The overall lack in output in PCBs originate from the low looping count. This may be improved upon by adding more layers in series, however this may become unaffordable as each would require hidden via's within the PCB layers which is expensive.

xy-Torquer: Aircoil

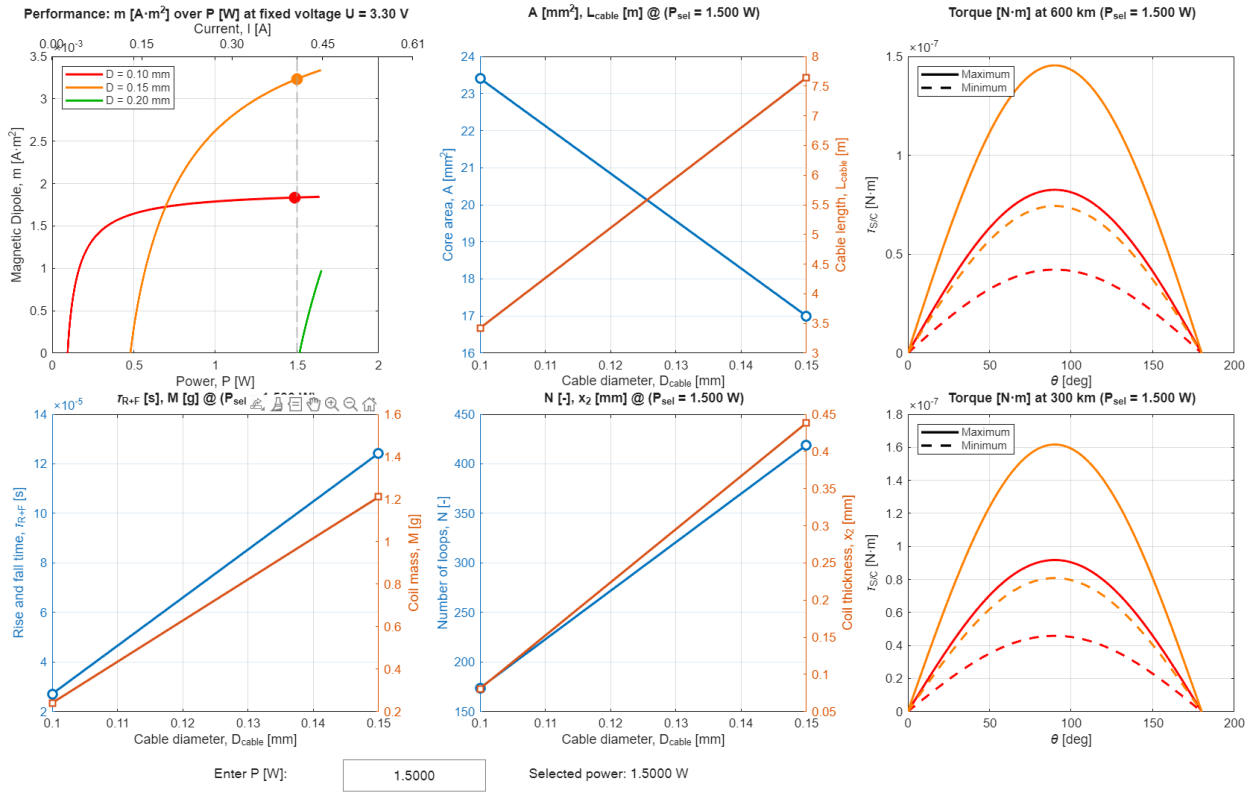


Figure 4.5: Aircoil magnetorquer design for $W_{max} = 5[mm]$ and $H_{max} = 21.5[mm]$, achieving $1.6 \cdot 10^{-7}[Nm]$ @ $P_{sel} = 1.5[W]$ @ $300[km]$ for $D_{cable} = 0.15[mm]$

In conclusion, these aircoils inside the z-torque aircoil magnetorquers are too weak as they are many times lower than the required torque @ $300[km]$

xy-Torquer: Solenoid (with core)

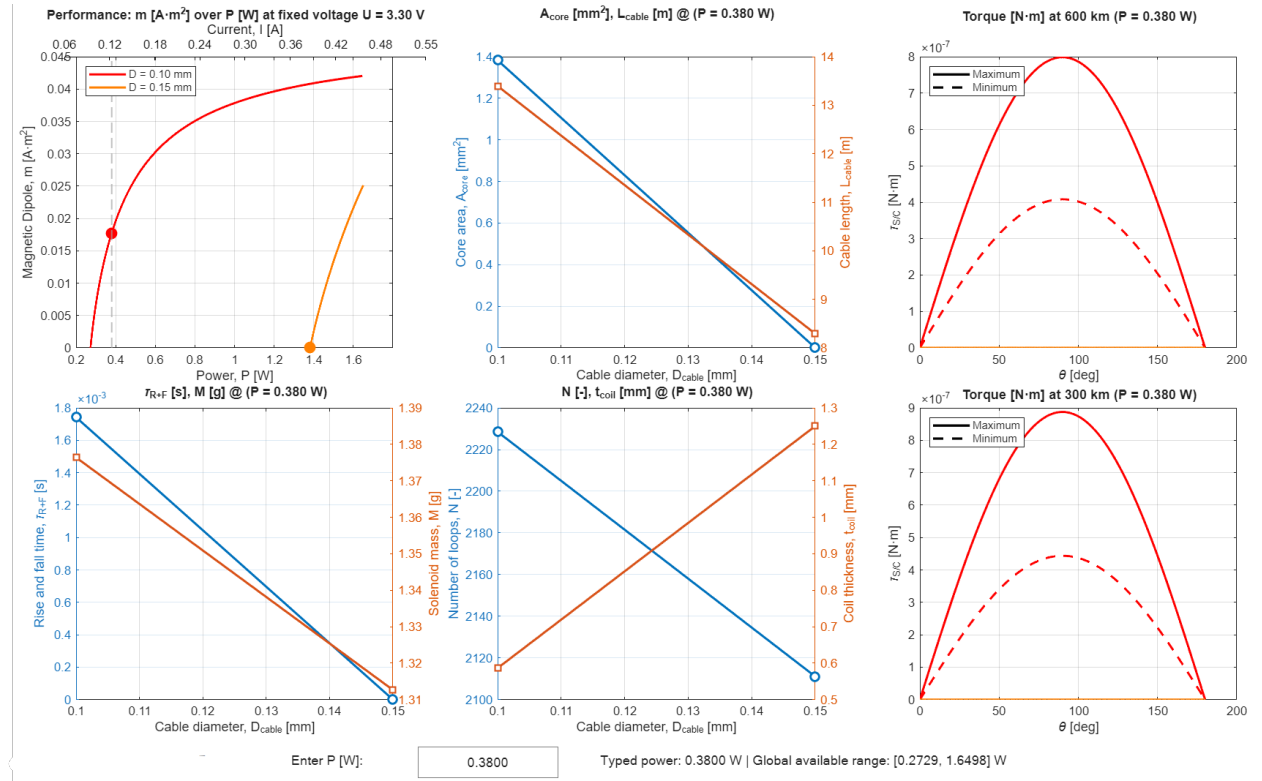


Figure 4.6: Solenoid (with core) magnetorquer design for $W_{max} = 5[mW]$, $\mu_r = 50$ (conservative estimation) and $L_{core} = 38[mm]$, achieving $9 \cdot 10^{-7}[Nm]$ @ $P_{sel} = 0.04[W]$ @ $300[km]$ for $D_{cable} = 0.1[mm]$ with a rise and fall time $\tau_{R+F} = 1.8 \cdot 10^{-3}[s]$

In conclusion, the solenoid's rise and fall time duration is too long and in violation with MTQ-04 and can not be considered as a category.

xy-Torquer: PCB

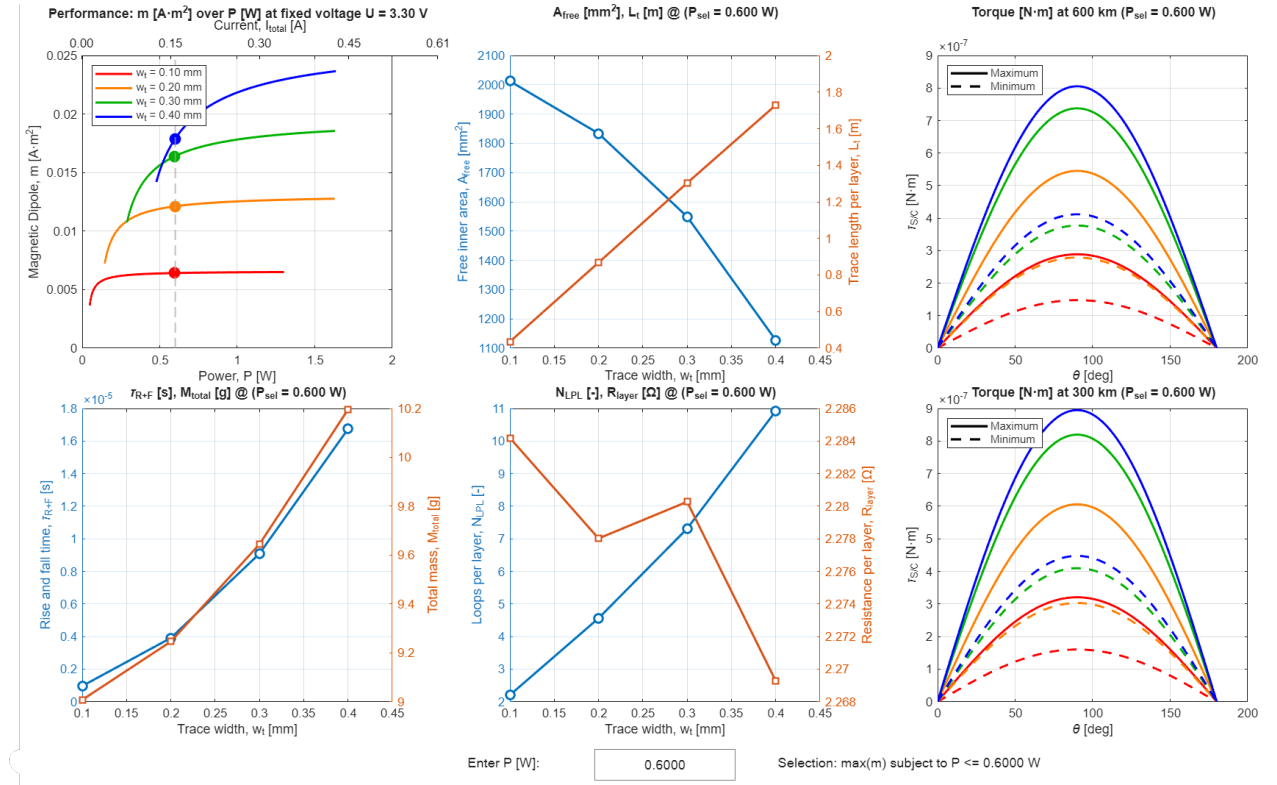


Figure 4.7: Aircoil design for $W_{max} = 40[mm]$, $L_{max} = 80[mm]$, $N_{Layer} = 8$ and $H_{max} = 1.55[mm]$, achieving $9 \cdot 10^{-7}[Nm]$ @ $P_{sel} = 0.04[W]$ @ $300[km]$ for $w_{trace} = 0.4[mm]$ with a total mass of $M = 10[g]$ and rise and fall time $\tau_{R+F} = 1.65 \cdot 10^{-5}[s]$

In conclusion, this design is feasible and well within the constraints and requirements imposed on this component.

Final Conclusion: Configuration

Following the elimination process, only one available configuration remains: z-torquer aircoil paired with two xy-torquer PCBs. Therefore no detailed trade-off analysis regarding the relative performance, cost and complexity between options is possible. Looking into the z-AC-xy-PCB configuration, they consume 1.24 [W] in total with their peak currents adding up to no more than 0.5 [A]. With the rise and fall time of the highest magnetorquer at $4 \cdot 10^{-4}[s]$ it is also well below the required limits of 0.0005[s].

4.3 Magnetometer

From the assignment requirements and predecessor project: 3 sensors required in orthogonal arrangement (one per axis) The RM3100 chip integrates all 3 axes on a single board. Hence, one RM3100 breakout board provides all 3 axes. The RM3100 Magneto-Inductive Magnetometer Sensor stands out as it delivers accuracy, stability, and resilience in diverse environments. Unlike other low SWaP magnetometers, it offers exceptional signal-to-noise performance, zero bias drift, and superior environmental resistance.

4.3.1 Magnetometer Trade-Off Analysis

Methodology

A two-stage trade-off analysis was conducted to select a suitable magnetometer for the Delfi-Twin ADCS. Before potential magnetometers were analysed, they were selected based on a few preliminary conditions.

Table 4.1: RM3100 technical specifications (source: <https://www.pnisenor.com/rm3100/>).

Parameter	Value
Magnetometer	RM3100
Sensitivity	13 nT
Noise	15 nT
Linearity (over $\pm 200 \mu\text{T}$)	0.50%
Field measurement range	$\pm 1100 \mu\text{T}$
Current (@ 8 Hz, 3 axes)	260 μA
Interface	SPI and I ² C
Operating temperature	-40°C to $+85^\circ\text{C}$

These make them a valid choice for this project.

All candidate magnetometers were required to:

- Provide three-axis magnetic field measurements
- Be compatible with the STM32 microcontroller (SPI/I²C²).
- Commercially available (COTS)
- Satisfy PocketQube size, mass, and power constraints

Once potential candidates were selected, the analysis began. The first stage consisted of an initial requirements-based screening to eliminate non-compliant sensors. The second stage evaluated the remaining candidates using weighted performance criteria.

Requirements Screening

The following component-level requirements were imposed:

- MGM-01: Measurement time ≤ 15 ms
- MGM-02: Measurement uncertainty $\leq 0.2\%$ of the Earth magnetic field
- MGM-03: Measurement range $\geq \pm 100 \mu\text{T}$

Given an Earth magnetic field magnitude of approximately 25–65 μT , requirement MGM-02 corresponds to a maximum allowable noise of approximately 50–130 nT per axis.

Table 4.2: Initial selection and Requirements screening results

Sensor	Noise (nT)	Result	Reason
IIS2MDC	~ 460	Fail	Exceeds uncertainty requirement by $> 3\times$
LIS2MDL	~ 450	Fail	Noise above allowable threshold
LIS3MDL	~ 350	Fail	Insufficient accuracy for ADCS
BMM150	300-600	Fail	High noise prevents reliable estimation
BMM350	~ 190	Fail	Above maximum allowable uncertainty
AK8963	–	Fail	Limited standalone availability
MAG3110	–	Fail	Obsolete component
HMC5883L	–	Fail	Discontinued, unreliable sourcing
RM3100	13-30	Pass	Meets all requirements
MMC5983MA	~ 40	Pass	Meets all requirements

The RM3100 and MMC5983MA satisfy all imposed requirements and are therefore retained for further evaluation. They were the magnetometers that were analysed further.

Trade-Off Criteria

The criteria for the second stage of the trade-off analysis were weighed out of a 5 point rank-based scoring system, 1 being the lowest weight in terms of importance and 5 being the highest. After requirements screening, only two parameters showed significant differentiation between the remaining candidates: Noise and Flight heritage.

Table 4.3: Trade-off criteria and weights

Parameter	Weight	Justification
Noise / resolution	5	The magnetometers are placed in close proximity to the magnetorquers due to volume constraints. Having a lower noise sensitivity is relevant and important for the scope of this project.
Flight heritage	4	Reduces mission risk through proven in-orbit performance without requirement for initial screening tests and leaves lesser room for error.

Other parameters such as power consumption, interface compatibility, size, and cost were excluded from the trade-off as both candidates perform similarly and satisfy system constraints. A better justification for the removal of other parameters are mentioned below:

Table 4.4: Comparison of non-discriminating parameters

Parameter	RM3100	MMC5983MA	Justification for exclusion
Power consumption	$\sim 135 \mu A$ @ 8 Hz	$\sim 100 \mu A$	Difference is marginal and both satisfy ADCS power constraints
Measurement range	$\pm 800 \mu T$	$\pm 800 \mu T$	Both significantly exceed Earth magnetic field ($\sim 50 \mu T$)
Interface	SPI + I ² C	SPI + I ² C	Both fully compatible with STM32 microcontroller
Size / Mass	Small module	Compact chip	Both fit within PocketQube volume and mass constraints
EMI immunity	Moderate to high	Moderate	Limited reliable quantitative data available for comparison
Availability / Cost	Readily available	Readily available	Both accessible and within project budget

- Power consumption: Both magnetometers suffice power consumption constraints and do not pose any significant implications in terms of differences for the scope of this project.
- Field Range: Both magnetometers exceed the $100 \mu T$ measurement range for the Earth's magnetic field.
- EMI (Electromagnetic Interference) Immunity: Testing for this criteria for any magnetometer considered wouldn't be feasible. Along with that, reliable data for this criteria is not available.
- Interface: Both magnetometers have sufficient capability to connect and provide data to the STM32 via relevant communication channels.
- Size/Mass: These factors are not very comparable for the magnetometers but they sufficiently pass mass and volume requirements for the project.
- Availability and cost: It is not a trade-off because both are available and do not pose a considerable change in the budget.

Scoring Method

Each parameter is scored on a scale from 1 to 5. The final score is computed using a weighted sum:

$$\text{Score} = w_n \cdot S_n + w_h \cdot S_h$$

where:

- w_n and w_h are weight allotted for given criteria (1-5)
- S_n and S_h are the respective scores (1-5)

Scoring Results

Table 4.5: Weighted trade-off results

Sensor	Noise Score	Heritage Score	Total Score
RM3100	5	5	45
MMC5983MA	4	4	36

Detailed Score Calculation

For completeness, the weighted scores are calculated as follows:

RM3100

$$\text{Score}_{\text{RM3100}} = (5 \times 5) + (4 \times 5) = 25 + 20 = 45$$

MMC5983MA

$$\text{Score}_{\text{MMC5983MA}} = (5 \times 4) + (4 \times 4) = 20 + 16 = 36$$

Maximum score possible = 45. Hence, RM3100 wins the trade-off analysis with the highest score.

Discussion

The RM3100 achieves the highest score due to its superior noise performance and extensive flight heritage. Its noise floor of 13–30 nT is significantly lower than that of the MMC5983MA (~ 40 nT), providing higher accuracy in magnetic field measurements.

This directly improves the performance of magnetic-field-based attitude determination, which is critical for meeting the pointing requirements defined in Section 3.4.1. Additionally, the RM3100 has demonstrated reliability in multiple CubeSat missions, reducing integration and operational risk.

The MMC5983MA remains a viable alternative, offering acceptable performance within all requirements. However, its higher noise level and limited flight heritage introduce additional uncertainty in mission performance.

Final Selection

Based on the trade-off analysis, the RM3100 is selected as the baseline magnetometer for the Delfi-Twin ADCS.

4.4 Microcontroller

The microcontroller is the center of the ADCS. As written in the user requirements, since Delfi-Twin's OBC was chosen to be an STM32 controller, the ADCS will also use an STM32 for all its logic and calculations. The exact model is the STM32L476RG[12], and will be used in prototyping through the NUCLEO-L476RG evaluation board[19].

The STM32L476RG has the following main features, as shown in its datasheet[12]:

- 3.3 V supply
- Arm 32-bit Cortex-M4 CPU
- Up to 80 Mhz instruction frequency
- Up to 1 MB flash memory
- Up to 128 KB of SRAM
- Support for 3 I2C channels, as well as CAN, 3x SPI
- Serial wire debug

While the ADCS microcontroller acts as the master for all other components within the subsystem, it is a slave to the OBC microcontroller through the CAN communication protocol. As such, it can only take commands and reply to questions, but it cannot initiate communication with the OBC. Messages from the OBC are treated as requests which have priority second only to crucial tasks such as magnetorquer on/off setting. Requests are assumed to be only of the following types:

- **Type 0: Write.** In this request, the OBC attempts to change some value in the ADCS's system data, either as an update or as a command. Most importantly, this can include:
 - Operational state: sets the ADCS into an active or inactive state.
 - Nadir axis: changes which face the ADCS will point towards Nadir.
 - Orbital elements (if applicable): updates the ADCS on where it is around the Earth and its orbital elements. This may only be implemented if a GNSS module is brought on board.
- **Type 1: Read.** In this request, the OBC attempts to read some value from the ADCS. The ADCS fetches that value and transmits it back to the OBC. This request serves as the way that the OBC can find if the ADCS is in Safe mode, or simply for monitoring or housekeeping. Most important readable values are:
 - Operational mode: gives the ADCS's current operational mode
 - State estimation: gives the attitude and angular velocity estimation

Defining the exact types of messages that can be communicated between the OBC and the ADCS microcontrollers is crucial for avoiding miscommunication. For example, if, for any reason, the OBC sends a faulty command to the ADCS, the slave will automatically ignore it as it does not recognize it.

4.5 Assembly configuration and CAD

This section will describe the configuration of the ADCS assembly, its representation in CAD, and the reasoning behind all important choices. The most important requirement driving the design of the system was CON-1, which specifies a maximum volume of 45 x 45 x 15 mm. The subsystem assembly consists of the following components:

- Main ADCS PCB (shown as a placeholder)
- 4x IR sensors (MLX90640), placed onto the PCB, facing outwards
- 2x PCB magnetorquers, one for each axis
- One z-axis aircoil magnetorquer, placed below the PCB
- One aircoil magnetorquer housing, around which the magnetorquer is wound
- Wires and connectors (not shown in the CAD).

The assembly is centered around the ADCS PCB, which has not yet been designed, but its placeholder is shown in Figure 4.8. The cutouts are meant for the IR sensors to fit in, the explanation for which is later in this section. This PCB will fit at least the STM32, three RM3100s, three DRV8838s, and their respective required electric components, as well as the necessary electrical connectors and a voltage regulator. With dimensions 45 x 45 x 1.5 mm, the PCB will fit within the required size.

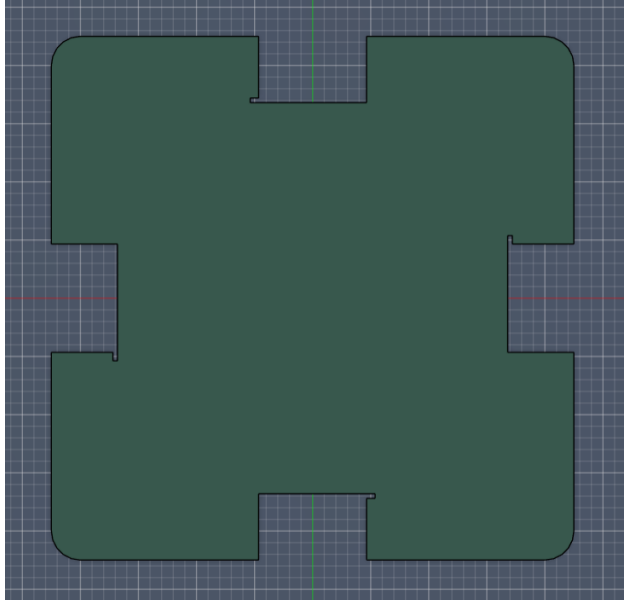


Figure 4.8: Placeholder for the ADCS PCB

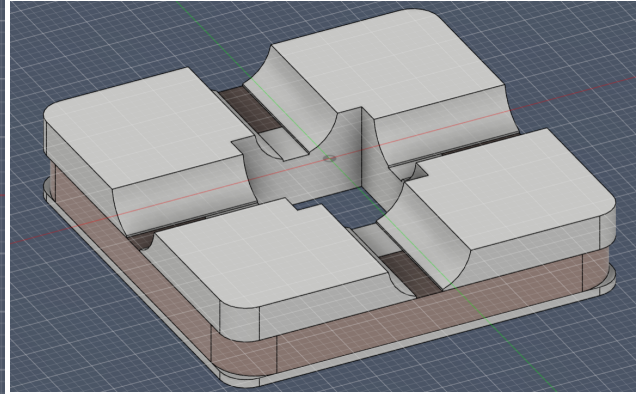


Figure 4.9: MTQ Housing (white) and aircoil (copper)

The IR sensors are bulky, with a diameter of 9.6 mm. In order for them to work, they are required to be oriented towards the exterior of the satellite. This means that they lie horizontally with their pins inwards. There were three options for placing the IR sensors:

- **Place them in the inside empty space of the z-torquer.** This option would have saved space, but it is impossible to make room for the IR sensors to image the outside of the spacecraft because the wires of the aircoil are in the way.
- **Place them on top of or below some of the components.** While this would have been the simplest option, it would not have been feasible to include all components. With a PCB thickness of 2 mm, an IR sensor diameter of 10 mm, the z MTQ is left with only 3 mm in z-wise dimension, which is not enough for a reasonable design.
- **Incorporate them into the PCB.** This compromise has the IR sensors hold space inside the PCB, hence the cut-outs in Figure 4.8. This option not only solves the space requirement, but also offers easy access to the pins of the IR sensors, which can be soldered directly to the PCB, greatly facilitating the integration process. However, this does induce a thickness requirement on the PCB, which is reflected in MCU-01.

The z torquer requires a large volume and enclosed area for optimal performance. The only two ways to integrate the MTQ would be either above or below the existing assembly of the PCB and IR sensors. Placing it above would have prohibited access from important ICs and connectors on the PCB, so a location beneath the PCB and IR sensors was chosen. The aircoil is held in place with the help of a housing, which also interfaces with the IR sensors. The MTQ aircoil and housign are shown in Figure 4.9.

Figure 4.12 shows the configuration of the PCB, IR sensors and magnetorquer. As shown in Figure 4.10, it occupies a z-wise dimension of 14.95 mm, which, in combination with the lengths of the components being no more than 45 mm, satisfies CON-1.

As explained in section 4.2, the x and y magnetorquer were chosen as their PCB variants, placed on the side panels of the satellite. Talks with the user have resulted in the possibility of using the side panels for this purpose. The ADCS assembly, including the PCB MTQs, is shown in Figure 4.11.

To give an overview of all the hard- and software of this project a configuration tree was constructed, shown in Figure 4.13. It highlights all the hardware components that were chosen on the left and all the software in place for it to function. On the software branch, the algorithms and main loop were all developed in-house or iterated upon whereas the drivers were provided by their respective manufactures.

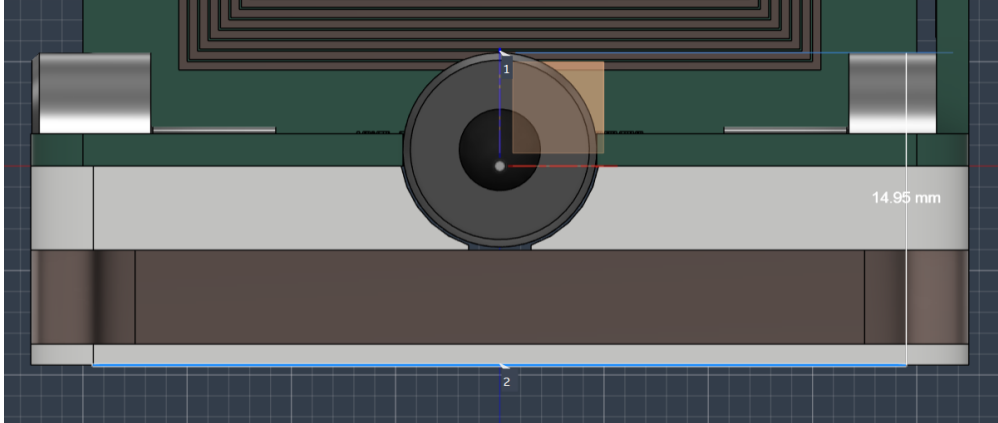


Figure 4.10: Side-on view of the assembly showing its thickness of 14.95 mm

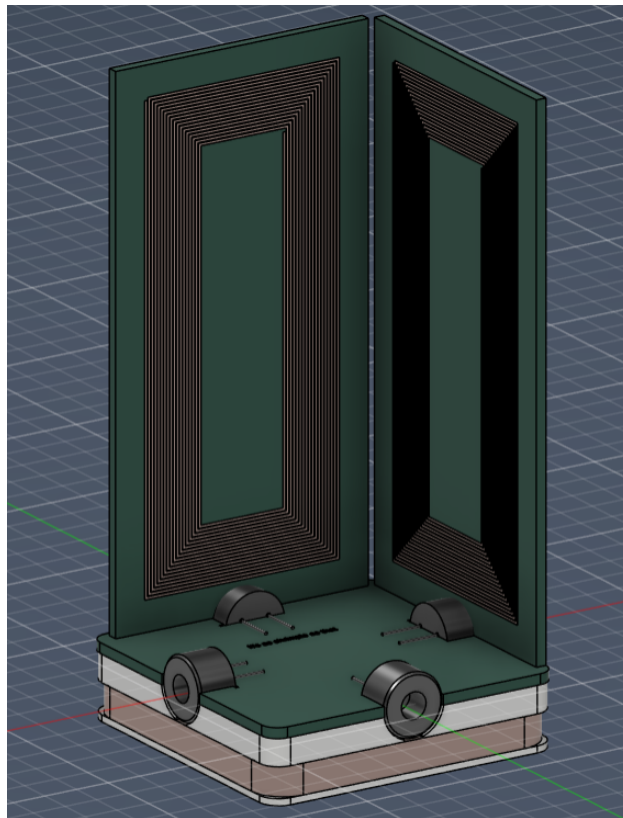


Figure 4.11: Image of the full CAD, including the PCB side panels

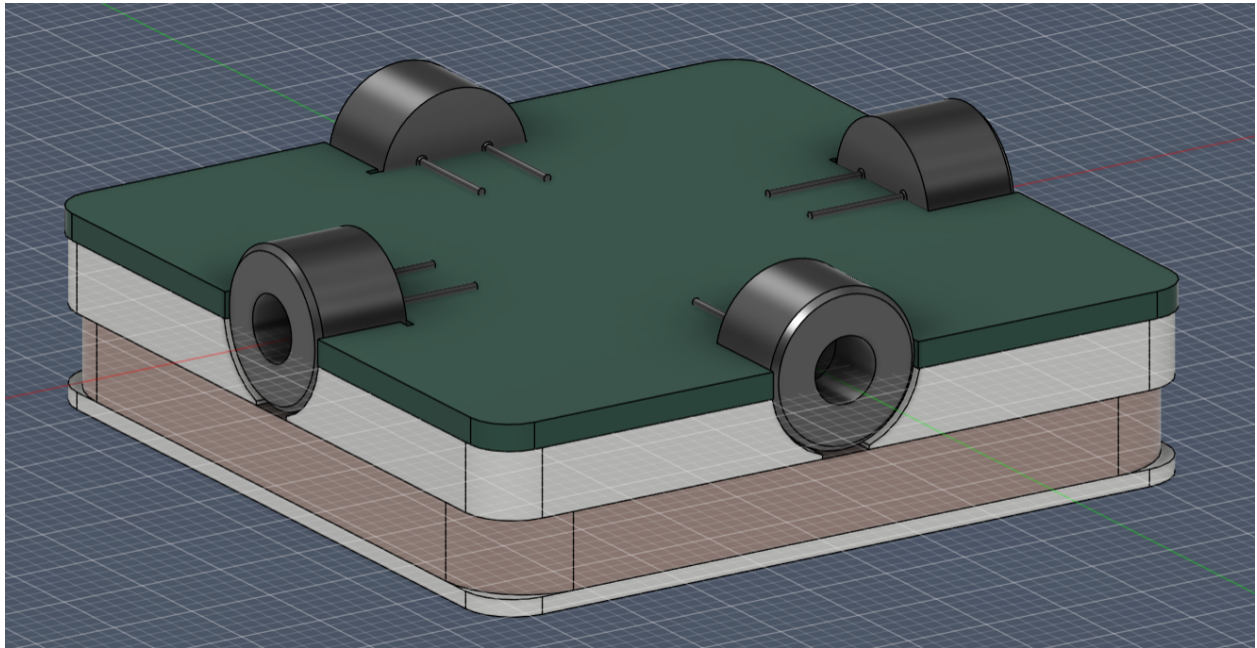


Figure 4.12: Image of the CAD of the main part of the ADCS, excluding the PCB side panels. Shown: PCB placeholder (green), IR sensors (black), aircoil (copper), and MTQ housing (white)

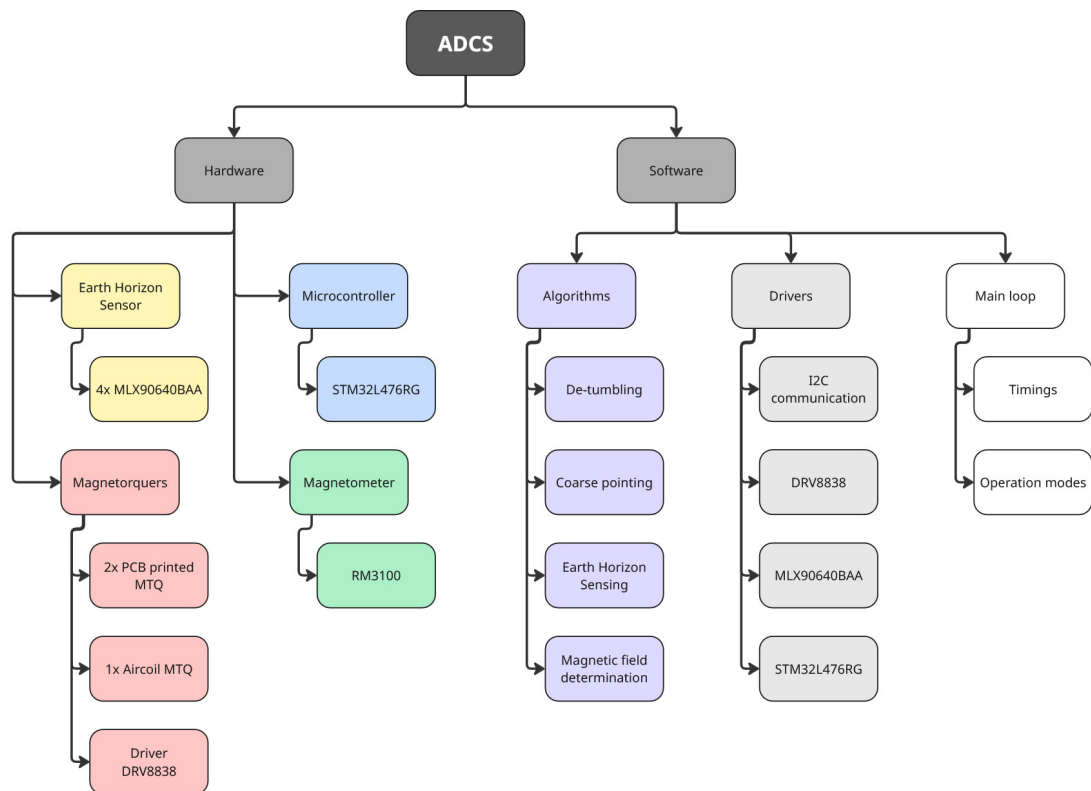


Figure 4.13: Configuration Tree of the ADCS

5. Control and Simulation

In this section the control and simulation required for the ADCS are developed. It can be split up into two major parts; the embedded ADCS software and the simulation software.

5.1 Embedded control algorithms

To determine the attitude there are two types of sensors included in the ADCS. The primary sensors are infrared, IR, sensors whilst the secondary were chosen to be magnetometers. The magnetometers are included to combat the bias present in the IR sensors. The primary algorithm for the IR sensors is explained first, then the magnetometer's algorithm and lastly the way the two interact and correct one another.

5.1.1 Earth Horizon Sensor Algorithm

Single Sensor Algorithm

The algorithm utilized in this study builds upon development initiated during the 2025 project cycle. Because the predecessor group focused exclusively on algorithmic architecture, a significant portion of the foundational framework was already established. For a comprehensive analysis of, for example, the image generation used for testing a reference is made to the 2025 project report [6].

The sensing system employs infrared (IR) sensors to facilitate Earth horizon detection. These sensors detect the IR radiation emitted by the Earth, leveraging the sharp thermal contrast between the planetary disk and the cold vacuum of space to identify the horizon interface. However, the hardware implementation is constrained by the processing limitations of the STM32 microcontroller. To maintain the required 8 Hz update frequency[6], the selected algorithm must prioritize computational efficiency.

The main focus was to comply with **REQ-ESH-03**: *"The Earth Horizon Sensors shall provide a pitch and roll measurement with an accuracy of at least 10 degrees."*. This was already tackled by last years project nevertheless a literature review was conducted to evaluate existing solutions for horizon sensing. A prominent candidate identified was an algorithm specifically optimized for large field-of-view IR sensors. While technically robust, this method relies on spherical trigonometry and a Random Sample Consensus (RANSAC) framework [20]. These requirements were deemed too computationally intensive for the STM32 architecture and too complex for the current implementation timeline. Consequently, due to these time constraints, the algorithm proposed during the previous year's research was chosen. Although previously documented, this method had not yet been ported to C++, making its implementation and optimization the primary focus of the current development phase [6] as it already complied with **REQ-ESH-03**. As mentioned, mainly improvements had to be made to make the algorithm compatible with the hardware and main loop timing. To show the compliance, the error in pitch and roll can be seen in Figure 5.1.

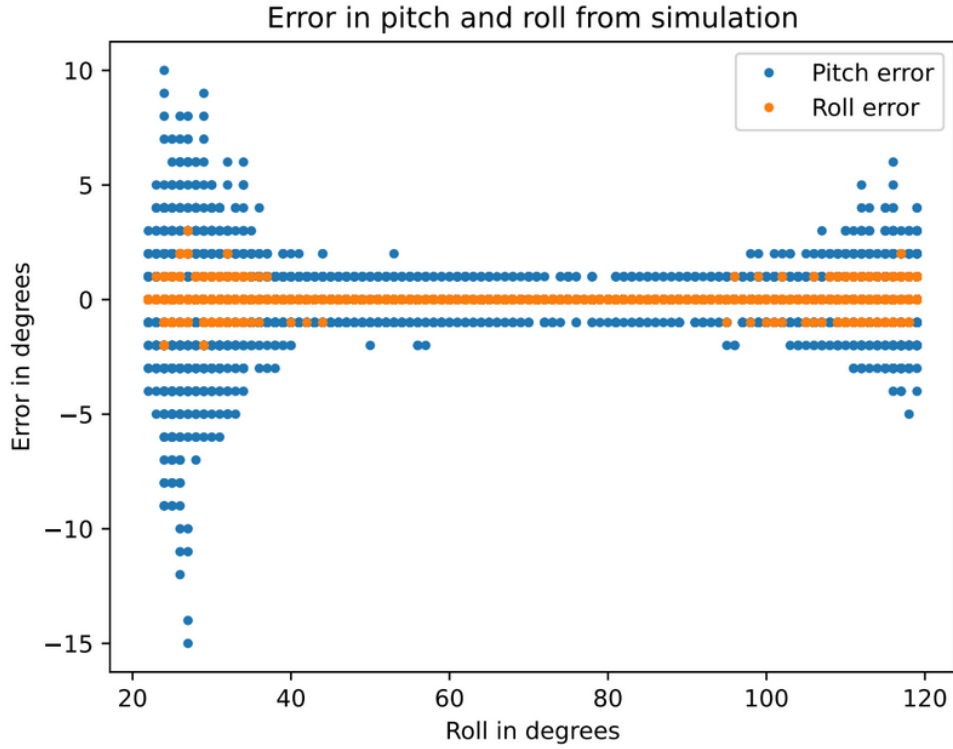


Figure 5.1: Error in pitch and roll when varying the roll for one sensor

Here a large error can be observed in the high and low roll regimes. However, due to there being a sensor present on each of the four lateral faces of the satellite there is always a sensor with the horizon in view between 35° and 110° . This is showcased in Figure 5.2.

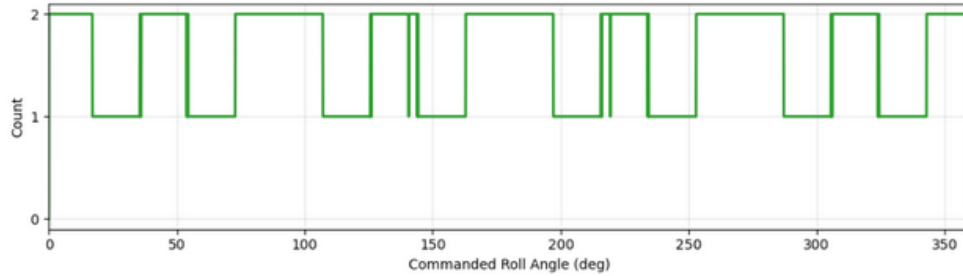


Figure 5.2: Number of Sensors with $\|\text{Roll error}\| < 5^\circ$ and $\|\text{Pitch error}\| < 5^\circ$ (0.1 deg/sec)

The main problem with the algorithm was the degeneracy due to a frame having multiple pitch and roll combinations, this is mitigated by the implementation of a banded search and weighting different sensors depending on the amount of horizon visible which will be elaborated on later on.

The IR sensor works as an array of 32 by 24 pixels so a total of 768 pixels. Each of these give a certain value according to the amount of infrared light hitting it. All the pixels with a value higher than the threshold temperature are mapped onto a mask and should provide the Earth's outline on the sensor field. This specific mask is then compared against roughly 32000 simulated entries in a lookup table, it then picks the pitch and roll of the state that closest align to what the sensor is providing. This way no complex geometry or expensive calculations are done on the STM32 and only a compressed lookup table for the vertical and horizontal angles is used as well as a table to compensate for the geometry of the lens.

The algorithm was further expanded in a couple of ways. The first addition was that of a banded search. A

rudimentary vector would be produced that allows a preliminary search of the large table. The algorithm then searches within this band which is approximately ten percent of the table. This search was compared to the original one with a script and a speed up of around 8 times is achieved. The band is dynamically widened or tightened depending on the amount of trusted frames the microcontroller receives. This is introduced as a way to still allow quick horizon sensing when the rotational rate of the satellite is high so it is ensured that the new frame falls within the band.

The second addition is to try to mitigate the disturbance of the Sun. It was not possible to get entirely rid of its effect as it saturates exposed pixels that could create ghosting effects and spill over into neighboring pixels. This would create a second horizon which would trip up the algorithm. To mitigate this a flood fill was performed to always ensure that the largest blob of hot pixels is chosen. The sun would at most effect 4 pixels due to its minute solid angle. This could create erroneous results if it overlaps with the horizon. However, sunset and sunrise take 7.5 seconds which is negligible compared to its orbital period of 5431.18 seconds.

The last addition was the use of floating point arithmetic and 32-bit integers to allow the compression of the lookup table so that it would fit in the limited storage of the STM32. The scaler was chosen to be twelve as to preserve the data accurately enough to comply with the requirements set and not lose information.

The last effect that was not addressed in the previous report was the effect of atomic oxygen damaging the lens. This was briefly explored but was deemed insignificant as the main flight direction of the Delfi-Twin would be with the small face prograde and the deployable solar panels at the back like a shuttlecock. Thus, the sensors would be facing perpendicular to the flight direction for almost the complete mission thus the amount of oxygen hitting the sensors would be greatly reduced compared to the previously assumed values.

Four Sensor Algorithm

To combine the individual sensors into one coherent system a couple of functions are to be put in place. As to not blindly trust each sensors output a confirmation threshold is applied which at 8 Hz would mean that the horizon needs to be in view for 625 ms. This sets an upper bound to the rotational rate at which the earth horizon algorithm still functions. If the horizon crosses over the field of view of the sensor before five frames can be read out, it would never surpass the confirmation threshold. The FOV of the MLX90640BAA was already mentioned to be 110° by 75° in section 4.1. Thus, the highest rotational rate that would guarantee at least one trusted output is 120 degrees per second. This upper limit would not be reached as most of the de-tumbling would be done by the magnetometer readings exclusively at such high rates. In addition, the expected rotational rate after rocket separation is expected to be far below this value as well.

`Combining_sensors.py` first sets the global parameters; the origin of the sensor, the amount of pixels of each sensor, the threshold detection temperature and the sensor offset that was mentioned previously.

Afterwards, the angles to account for the fish-eye effect of the sensor due to its large field of view are loaded in together with the lookup table of the pitch and roll combinations. It should be noted that compared to last year a sixth column was created to enable roll-band detection.

The sensor switching logic for the earth horizon sensors has pitch gating to save time by having a smaller region to search in. It searches in a narrow band of the complete look up table from a simplified calculation. Two modes are implemented, one for higher rotational rates and a second for nadir pointing. This hysteresis only switches after 40 frames of confirmed horizon sensing. It can switch back when enough frames that should see the earth but do not fall within the band are encountered. Each sensor has to prove to the micro controller that it reliably sees the earth by having a threshold of five frames of giving a correct reading. Once it is trusted, it has 2 frames of grace, to not immediately discard the sensor when it glitches.

The main function of `Switching_logic.py` obtains the data stream of all the four sensors. Initially it starts with zero available candidates which sets the already defined gate at wide. If it has one sensor that is deemed valid it checks if it already acquired a previous reading to apply the pitch gating. If that is the case, the reading can be given a valid or invalid flag depending if it falls in the gate. If the sensor has no history and makes a valid reading it adds to a confirmation counter and sets the grace counter to zero. If the confirmation counter surpasses the threshold it is assigned as a candidate.

If it is acquired as a candidate but has an invalid reading it starts counting up the grace frames and if it surpassed the grace limit it sets the confirmation counter to zero, thus letting the STM32 know not to use this reading for, at least, the next five frames.

After it has checked the validity of the sensors it determines if it wants to switch its gate from wide to narrow or vice versa by counting if it can successfully have 40 frames of good readings.

Lastly it combines the readings into the output. If no candidates are available, which is rare but can happen when the long axis, z , is pointing towards earth. It returns the last known values. Otherwise, it averages out the pitch and roll of the sensors. The views are already converted as to account for the 90 degree offset from one another. it returns the average pitch, roll

The code is fully deterministic achieved by having simple for loops as well as fixed allocated sizes for the variables so no heap memory is used since dynamically allocating memory can be unpredictable.

Lastly, if two or more sensors are believed to have a valid reading, a weighting system is introduced to have a favorable sensor, which is the one is closest to having an Earth area consisting of half the pixels. This would guarantee that this is the sensor that has the most centered horizon which would give the least distorted geometry thus the highest confidence.

The singular algorithm was tested by feeding it frames from the image generation script made by previous years. The algorithm as a whole was tested by feeding the four digital sensors (arrays) ten frames over a some time to check if the grace and conformation systems were working correctly.

5.1.2 Magnetometer Algorithm

Overview

The PNI RM3100 three-axis magnetometer is used as the primary magnetic field sensor in the Attitude and Orbit Control System (ADCS). It interfaces with the STM32L476RG microcontroller via the **I2C3** peripheral operating in Fast Mode at 400 kHz. The driver is responsible for sensor initialisation, continuous magnetic field acquisition, and delivery of calibrated field vectors to the B-dot detumbling controller.

Two top-level requirements govern the design:

- **MGM-01:** Data delivery latency ≤ 15 ms
- **MGM-02:** Measurement uncertainty $\leq 0.2\%$

Sensor Operating Principle

The RM3100 uses PNI's proprietary MagI²C technology, in which a drive coil excites a magneto-inductive core and a sense coil measures the response. The number of drive cycles — the *Cycle Count* (CC) — directly controls both the measurement sensitivity and the sample rate.

Gain

The sensor output is a 24-bit two's complement integer. The gain, which converts raw counts to physical units, is given by:

$$G = 0.3671 \cdot CC + 1.5 \quad [\text{LSB}/\mu\text{T}] \quad (5.1)$$

The magnetic field along each axis is then:

$$B_a = \frac{\text{raw}_a}{G} \quad [\mu\text{T}], \quad a \in \{X, Y, Z\} \quad (5.2)$$

Noise

The RMS noise decreases with increasing CC. For three candidate configurations:

The three-axis rate is one third of the single-axis rate because the sensor measures each axis sequentially:

$$f_{3\text{-axis}} = \frac{f_{1\text{-axis}}}{3} \quad (5.3)$$

Table 5.1: RM3100 configuration candidates

CC	Noise [nT]	$f_{1\text{-axis}}$ [Hz]	$f_{3\text{-axis}}$ [Hz]
50	30	1600	533
100	20	850	283
200	15	440	147

MATLAB Parameter Optimisation

A MATLAB script was used to determine the optimal CC and update rate for each ADCS operating mode. The objective is to select the *highest* CC (lowest noise) that still satisfies the minimum measurement rate required by each mode.

Required Measurement Rate

For each mode, the minimum required sample rate is:

$$f_{\text{needed}} = m \cdot \frac{\omega_{\text{max}}}{360} \quad (5.4)$$

where ω_{max} is the maximum satellite spin rate in deg/s and m is a safety multiplier (analogous to a Nyquist factor). A floor frequency f_{floor} is imposed regardless of spin rate:

$$f_{\text{chosen}} = \max(f_{\text{needed}}, f_{\text{floor}}) \quad (5.5)$$

CC Selection

The valid CC configurations are those for which the three-axis rate is sufficient:

$$\mathcal{S} = \{i \mid f_{3\text{-axis},i} \geq f_{\text{chosen}}\} \quad (5.6)$$

From the valid set $\mathcal{S}$, the configuration with the *highest* index (highest CC, lowest noise) is selected:

$$i^* = \max(\mathcal{S}) \quad (5.7)$$

Pointing Error

The expected pointing error due to sensor noise is estimated as:

$$\epsilon = \frac{180}{\pi} \cdot \sqrt{2} \cdot \frac{\sigma_n}{B} \quad [\text{deg}] \quad (5.8)$$

where σ_n is the RMS noise in nT and B is the local field magnitude. The $\sqrt{2}$ factor accounts for the RMS combination of noise from two successive measurements used in the B-dot finite difference.

Optimisation Results

Table 5.2: MATLAB optimisation results — CC=200 selected for all modes

Mode	ω_{max} [deg/s]	f_{chosen} [Hz]	Period [ms]	CC
Detumbling	180	50	20	200
Pointing/Nominal	0.07	10	100	200

At CC=200, $f_{3\text{-axis}} = 146.7$ Hz, which satisfies all mode rate requirements with significant margin. The resulting gain and measurement time are:

$$G = 0.3671 \times 200 + 1.5 = 74.92 \text{ LSB}/\mu\text{T} \quad (5.9)$$

$$t_{\text{meas}} = \frac{1}{f_{3\text{-axis}}} = \frac{1}{146.7} \approx 6.8 \text{ ms} \quad (5.10)$$

Software Architecture

The driver is structured into four layers, as shown in Figure 5.3.

ADCS Control Loop (<code>main.c</code>) B-dot controller, mode scheduling
RM3100 Driver API (<code>rm3100.c/.h</code>) <code>RM3100_Init()</code> , <code>RM3100_ReadData()</code>
STM32 HAL I2C <code>HAL_I2C_Mem_Write()</code> , <code>HAL_I2C_Mem_Read()</code>
Hardware I2C3: PC0 = SCL, PC1 = SDA, PC7 = DRDY

Figure 5.3: Software layer diagram

I2C Configuration

Table 5.3: I2C3 peripheral configuration

Parameter	Value
Peripheral	I2C3
SCL pin	PC0 (AF4, open-drain)
SDA pin	PC1 (AF4, open-drain)
DRDY pin	PC7 (GPIO input)
Mode	Fast Mode, 400 kHz
TIMINGR	0x10909CEC (80 MHz PCLK1)
Address	0x20 (7-bit), 0x40 (HAL 8-bit)
Pull-ups	4.7 k Ω to 3.3 V (external)

The HAL 8-bit address is the 7-bit address left-shifted by one bit:

$$A_{\text{HAL}} = A_{7\text{-bit}} \ll 1 = 0x20 \ll 1 = 0x40 \quad (5.11)$$

Algorithm Description

Initialisation

Algorithm 1 RM3100 Initialisation

```

1: procedure RM3100_INIT(hi2c)
2:   Read REVID register (0x36) via I2C
3:   if REVID  $\neq$  0x22 then
4:     return ERR_REVID ▷ Wrong device or wiring fault
5:    $CC \leftarrow 200$ 
6:    $cc\_buf \leftarrow [CC_{\text{MSB}}, CC_{\text{LSB}}] \times 3$  ▷ Same CC for X, Y, Z
7:   Burst write  $cc\_buf$  to CCX1 (0x04), 6 bytes
8:   Write 0x79 to CMM (0x01) ▷ All axes | DRDM | Start
9:   return OK

```

The CMM register value 0x79 = 0b01111001 encodes:

Table 5.4: CMM register bit field (0x79)

Bit(s)	Field	Value
0	Start CMM	1 — begin continuous measurement
1	X enable	1
2	Y enable	1
3	Z enable	1
4	DRDM	1 — assert DRDY when all axes complete
7:5	Reserved	0

Measurement Acquisition

Algorithm 2 RM3100 Measurement Acquisition

```

1: procedure RM3100_READDATA(hi2c, data)
2:    $t_{\text{start}} \leftarrow \text{HAL\_GETTICK}$ 
3:   while DRDY (PC7) = LOW do                                      $\triangleright$  Wait for sensor to complete measurement
4:     if HAL_GetTick() -  $t_{\text{start}} \geq 20$  ms then
5:       return ERR_TIMEOUT
6:   Burst read 9 bytes from MX2 (0x24)
7:   for each axis  $a \in \{X, Y, Z\}$  do
8:      $\text{raw}_a \leftarrow (b_2 \ll 16) \mid (b_1 \ll 8) \mid b_0$ 
9:     if  $\text{raw}_a \ \& \ 0x800000 \neq 0$  then
10:       $\text{raw}_a \leftarrow \text{raw}_a \mid 0xFF000000$                                  $\triangleright$  Sign-extend to 32-bit
11:       $B_a \leftarrow \text{raw}_a / G$                                             $\triangleright G = 74.92 \text{ LSB}/\mu\text{T}$ 
12:    $|\mathbf{B}| \leftarrow \sqrt{B_X^2 + B_Y^2 + B_Z^2}$ 
13:   return  $B_X, B_Y, B_Z, |\mathbf{B}|$ 

```

Sign extension. The RM3100 outputs a 24-bit two’s complement value packed in three bytes (big-endian). To convert to a signed 32-bit integer, the sign bit (bit 23) is replicated across the upper byte:

$$\text{raw}_{32} = \begin{cases} \text{raw}_{24} & \text{if bit 23} = 0 \\ \text{raw}_{24} \mid 0xFF000000 & \text{if bit 23} = 1 \end{cases} \quad (5.12)$$

Burst read. Registers MX2–MZ0 (0x24–0x2C) are contiguous. A single I2C transaction reads all 9 bytes:

1. START condition
2. Write device address + WRITE bit (0x40)
3. Write register address 0x24
4. Repeated START
5. Write device address + READ bit (0x41)
6. Clock in 9 data bytes with ACK; final byte with NACK
7. STOP condition

Latency Budget

The total data delivery latency for a single measurement at $CC = 200$ is:

The latency budget is satisfied with 53% margin in every operating mode, since the measurement time depends only on CC and not on the control loop period.

Table 5.5: Latency budget at CC = 200, I2C at 400 kHz

Component	Duration
Sensor measurement (DRDY wait)	6.82 ms
I2C burst read (1 + 9 bytes)	≈ 0.20 ms
Sign extension + float divide	< 0.01 ms
Total	≈ 7.0 ms
MGM-01 requirement	≤ 15 ms
Margin	8.0 ms

Data Flow Summary

The complete data path from hardware to ADCS is:

$$\underbrace{\text{RM3100}}_{\text{hardware}} \xrightarrow{\text{I2C, 9B}} \underbrace{[MX_2 \dots MZ_0]}_{\text{raw bytes}} \xrightarrow{\times 3} \underbrace{[\text{raw}_X, \text{raw}_Y, \text{raw}_Z]}_{\text{int32}} \xrightarrow{\div G} \underbrace{[B_X, B_Y, B_Z]}_{\mu\text{T}} \xrightarrow{\|\cdot\|} |\mathbf{B}| \rightarrow \dot{\mathbf{B}} \quad (5.13)$$

Requirements Compliance

Table 5.6: Requirements compliance summary

Requirement	Threshold	Achieved	Status
MGM-01: latency	≤ 15 ms	≈ 7.0 ms	✓
MGM-02: uncertainty	$\leq 0.2\%$	15 nT at CC = 200	✓

5.2 Flight Software Architecture

5.2.1 ADCS System Overview

The Delfi-Twin attitude determination and control system (ADCS) Flight Software implements a magnetic-only control architecture for rate damping and coarse nadir-pointing control. The system schedules magnetometer measurements, Earth-horizon sensing, and magnetorquer actuation for closed-loop control.

The implemented control architecture operates in two modes: detumbling and coarse nadir-pointing.

During the detumbling operation, the spacecraft reduces angular velocity using a scaled B-dot magnetic damping controller derived from measured magnetic-field variation. During the nadir-pointing operation, the magnetic damping controller remains active while an additional nadir-pointing controller generates corrective dipole commands that steer the spacecraft toward nadir.

The sensing architecture combines RM3100 magnetometer measurements for magnetic-field sensing and Earth-horizon to determine the nadir vector in body-frame.

Actuation is performed exclusively using magnetorquers. Magnetorquers can only generate torque perpendicular to the local geomagnetic field, which prevents arbitrary torque generation about all spacecraft axes simultaneously. The flight software separates sensing and actuation into discrete time windows to prevent magnetorquer interference with magnetic-field measurements.

Integrated ADCS Architecture

The ADCS flight software is divided into modular subsystems that handle sensor measurements and processing, control evaluation, dipole-command calculation, and actuator-command generation.

The controllers generate magnetic dipole commands using sensor measurements. These commands are then combined and conditioned before magnetorquer actuation. Command conditioning refers to the enforcement

of actuator limits and finite command resolution before dipole realizations. Please refer to figure 3.3.1 and for more information.

The modular structure simplifies controller integration, allowing the sensing, control, and actuation sub-systems to remain independently testable within the simulation environment. For more information on the flight software modules, please refer to Section 5.2.2, Section 5.2.3, and Section 5.2.4.

Control and Simulation

The ADCS was implemented within a modular Basilisk simulation environment that included spacecraft dynamics propagation, environmental modeling, sensor simulation, flight software execution, and magnetorquer actuation.

The Basilisk environment reproduces the logical execution order and message flow of the implemented control architecture, but does not model cycle-exact STM32 runtime behavior due to the absence of processing time for sensor readings. However, the flight-software closed loop behavior remains representative of the real system.

Refer to Section 5.3.1 for more information on the simulation environment.

Verification Strategy Overview

System verification was performed using closed-loop software-in-the-loop simulation campaigns executed within the Basilisk environment. The verification process evaluates detumbling convergence, nadir-pointing stability, actuator behavior, and disturbance sensitivity. Refer to Section 5.4.1 for more information on verification.

5.2.2 Magnetic Detumbling Architecture

This section defines the magnetic detumbling controller used to reduce spacecraft angular velocity before and during nadir-pointing.

The implemented controller uses body-frame magnetic-field measurements to generate damping dipole commands.

Detumbling Objectives

The magnetic detumbling controller reduces post-deployment angular velocity, prepares the spacecraft, and then aids it in nadir-pointing operation.

Magnetorquer actuation generates control torque according to:

$$\boldsymbol{\tau}_{MTQ} = \mathbf{m} \times \mathbf{B} \quad (5.14)$$

with the magnetic dipole vector $\mathbf{m}$ [A/m^2] and the Earth magnetic field strength vector $\mathbf{B}$ [T].

Because magnetic torque is generated through interaction with the geomagnetic field, instantaneous control torque cannot be generated along the magnetic field direction. The spacecraft, therefore, remains consistently underactuated during magnetic-only control.

Magnetic-Field Derivative Estimation and Prediction

To minimize discretization noise and transient spikes caused by noisy magnetometer measurements, the flight software estimates the body-frame magnetic-field derivative $\dot{\mathbf{B}}$ using a linear least-squares regression over a finite measurement buffer.

The spacecraft, while sensing the magnetic field, stores the measurements along with their time history into a sliding window buffer of fixed length:

$$N = \text{ESTIMATION_BUFFER_SIZE} \quad (5.15)$$

Once the buffer is fully populated, the time array is centered about its mean:

$$\bar{t} = \frac{1}{N} \sum_{i=1}^N t_i \quad (5.16)$$

$$t_{\text{centered},i} = t_i - \bar{t} \quad (5.17)$$

The best-fit linear slope representing the magnetic-field derivative estimate, $\dot{\mathbf{B}}_{\text{est}}$, is calculated by minimizing the sum of squared residuals:

$$\dot{\mathbf{B}}_{\text{est}} = \frac{\sum_{i=1}^N t_{\text{centered},i} \mathbf{B}_i}{\sum_{i=1}^N t_{\text{centered},i}^2} \quad (5.18)$$

Because there is an inherent delay between sensing magnetic fields and executing actuator commands, the software predicts the magnetic field vector forward to the midpoint of the upcoming actuation interval. The total prediction time horizon $\Delta t_{\text{predict}}$ is given by:

$$\Delta t_{\text{predict}} = \frac{1}{2} \Delta t_{\text{actuation}} + \frac{1}{2} N \Delta t_{\text{fsw}} \quad (5.19)$$

where $\Delta t_{\text{actuation}} = \text{TOTAL_ACTUATION_TIME}$ and Δt_{fsw} is the flight software step time, i.e., (FSW_STEP_TIME_S) .

Using the mean sensed magnetic field vector: $\bar{\mathbf{B}} = \frac{1}{N} \sum_{i=1}^N \mathbf{B}_i$, an intermediate predicted field $\mathbf{B}_{\text{out,raw}}$ is generated:

$$\mathbf{B}_{\text{out,raw}} = \bar{\mathbf{B}} + \dot{\mathbf{B}}_{\text{est}} \Delta t_{\text{predict}} \quad (5.20)$$

To make reliable and accurate predictions concerning the magnetic vector, the final output field $\mathbf{B}_{\text{out}}$ is normalized and scaled to preserve the original physical magnitude of the sampled field environment:

$$\mathbf{B}_{\text{out}} = \|\bar{\mathbf{B}}\| \frac{\mathbf{B}_{\text{out,raw}}}{\|\mathbf{B}_{\text{out,raw}}\|} \quad (5.21)$$

Skew-Symmetric $\mathbf{B}$ -dot Damping Law

Assuming that the short-term rotational variation of the geomagnetic field in the body frame is dominated by spacecraft rotation rather than orbital motion, the field derivative satisfies:

$$\dot{\mathbf{B}} = -\boldsymbol{\omega} \times \mathbf{B} \quad (5.22)$$

From this relationship, the component of the angular velocity orthogonal to the magnetic field vector—representing the controllable angular-rate subspace $\boldsymbol{\omega}_{\perp}$ —can be isolated geometrically:

$$\boldsymbol{\omega}_{\perp} = -\frac{\mathbf{B} \times \dot{\mathbf{B}}}{\|\mathbf{B}\|^2} \quad (5.23)$$

The flight software implements this cross-product relationship using matrix algebra. By mapping the normalized magnetic field vector $\hat{\mathbf{B}} = \mathbf{B}/\|\mathbf{B}\|$ into a skew-symmetric matrix $S(\hat{\mathbf{B}})$, the cross product is evaluated linearly:

$$S(\hat{\mathbf{B}}) = \begin{bmatrix} 0.0 & -\hat{B}_z & \hat{B}_y \\ \hat{B}_z & 0.0 & -\hat{B}_x \\ -\hat{B}_y & \hat{B}_x & 0.0 \end{bmatrix} \quad (5.24)$$

Using this framework, an intermediate vector $\mathbf{v}_1$ is computed in the code via matrix multiplication. Mathematically, this operation is equivalent to extracting a scaled version of the controllable angular velocity vector ($\mathbf{v}_1 \propto \boldsymbol{\omega}_{\perp}$):

$$\mathbf{v}_1 = S(\hat{\mathbf{B}}) \dot{\mathbf{B}} \equiv \hat{\mathbf{B}} \times \dot{\mathbf{B}} \quad (5.25)$$

The controllable angular momentum components are then mapped into the control loop by scaling this reconstructed rate vector by the spacecraft inertia matrix $\mathbf{I}$:

$$\mathbf{v}_2 = \mathbf{I}\mathbf{v}_1 \quad (5.26)$$

To generate the final damping command, a second cross-projection back onto the magnetic field geometry is performed. This project-and-scale law incorporates the base controller gain K and normalizes by the squared magnitude of the local field vector:

$$\mathbf{m}_{\text{raw}} = K \frac{S(\hat{\mathbf{B}})\mathbf{v}_2}{\|\mathbf{B}\|^2} \quad (5.27)$$

To preserve consistent energy dissipation and uniform damping behavior across discrete sensing-actuation cycles, the controller normalizes this dipole calculation by the scheduled actuation window duration: $\Delta t_{\text{actuation}} = \text{TOTAL_ACTUATION_TIME}$:

$$\mathbf{m} = \frac{\mathbf{m}_{\text{raw}}}{\Delta t_{\text{actuation}}} \quad (5.28)$$

If the squared norm of the magnetic field drops below a minimum threshold ($\|\mathbf{B}\|^2 < 10^{-12} \text{ T}^2$), the software bypasses computation and outputs a zero-dipole command ($\mathbf{m} = \mathbf{0}$) to prevent division-by-zero singularities. The resulting scaled magnetic torque directly opposes the local angular momentum variations, progressively reducing the spacecraft's angular velocity under noisy magnetic measurements.

5.2.3 Nadir Pointing and Magnetic Steering Control

This section defines the control logic used after magnetic detumbling to achieve coarse nadir pointing.

During nadir-pointing operations, the magnetic damping loop remains active to suppress residual angular rates, while the nadir-pointing controller concurrently generates steering dipoles. The primary objective of the mode is to continuously steer and align the spacecraft body-frame axis:

$$+\hat{\mathbf{x}}_B \rightarrow \hat{\mathbf{r}}_{\text{nadir},B} \quad (5.29)$$

where $\hat{\mathbf{r}}_{\text{nadir},B}$ represents the body-frame nadir-reference vector. Pointing performance remains instantaneously constrained by magnetic underactuation and orbit-dependent variations in the geomagnetic field.

Nadir-Reference Conditioning and Prediction

The nadir-reference prediction module filters incoming sensor data and projects the reference vector forward to synchronize with the midpoint of the actuator command window.

When a raw body-frame reference vector $\mathbf{r}_{\text{nadir},B}$ is received, it is normalized to enforce unit-vector geometry before normalized vectors $\mathbf{r}_{B,i}$ and their respective simulation timestamps t_i are cached into a sliding buffer until it reaches capacity:

$$N = \text{ESTIMATION_BUFFER_SIZE} \quad (5.30)$$

Once full, a linear least-squares regression is performed across the buffer window to compute the direction-rate derivative vector, $\dot{\mathbf{r}}_{\text{est}}$. To guarantee numerical conditioning of the matrix operations, the time axis is zero-centered around its mean:

$$\bar{t} = \frac{1}{N} \sum_{i=1}^N t_i \quad (5.31)$$

$$t_{\text{centered},i} = t_i - \bar{t} \quad (5.32)$$

The best-fit direction-rate derivative is evaluated by minimizing the sum of squared residuals across all three coordinate channels simultaneously:

$$\dot{\mathbf{r}}_{\text{est}} = \frac{\sum_{i=1}^N t_{\text{centered},i} \mathbf{r}_{B,i}}{\sum_{i=1}^N t_{\text{centered},i}^2} \quad (5.33)$$

To compensate for processing delays and the finite hold of the actuator cycle, the flight software extrapolates the mean buffered reference vector $\bar{\mathbf{r}} = \frac{1}{N} \sum_{i=1}^N \mathbf{r}_{B,i}$ over a predictive time horizon $\Delta t_{\text{predict}}$ defined by:

$$\Delta t_{\text{predict}} = \frac{1}{2} \Delta t_{\text{actuation}} + \frac{1}{2} N \Delta t_{\text{fsw}} \quad (5.34)$$

where $\Delta t_{\text{actuation}} = \text{TOTAL_ACTUATION_TIME}$ and $\Delta t_{\text{fsw}} = \text{FSW_STEP_TIME_S}$. This yields an intermediate unconstrained vector:

$$\mathbf{r}_{\text{out,raw}} = \bar{\mathbf{r}} + \dot{\mathbf{r}}_{\text{est}} \Delta t_{\text{predict}} \quad (5.35)$$

Because unconstrained linear extrapolation alters the magnitude of a directional reference, the vector is re-normalized to map back onto a valid unit sphere, yielding the final output reference vector $\hat{\mathbf{r}}_{\text{nadir},B}$:

$$\hat{\mathbf{r}}_{\text{nadir},B} = \frac{\mathbf{r}_{\text{out,raw}}}{\|\mathbf{r}_{\text{out,raw}}\|} \quad (5.36)$$

If the prediction yields a singularity ($\|\mathbf{r}_{\text{out,raw}}\| < 10^{-12}$), the flight software falls back safely to the most recent valid historical sample in the buffer.

Recovery-Mode Hysteresis Logic

Large-angle deviations are managed via a dedicated recovery steering routine. The alignment metric of the spacecraft is evaluated directly by projecting the target vector onto the primary body axis, which simplifies to the first element of the conditioned reference:

$$x_{\text{alignment}} = \hat{\mathbf{r}}_{\text{nadir},B}[0] \equiv \hat{\mathbf{x}}_B \cdot \hat{\mathbf{r}}_{\text{nadir},B} \quad (5.37)$$

where $x_{\text{alignment}}$ is $[-1.0, 1.0]$.

To prevent rapid, noise-induced mode switching near the operational boundaries, a hysteresis scheme regulates the operational state. The system transitions into Recovery Mode if the alignment metric falls below a critical threshold:

$$x_{\text{alignment}} < \cos(\theta_{\text{enter}}) \quad (5.38)$$

where $\theta_{\text{enter}} = \text{RECOVERY_ENTER_ANGLE_DEG}$. The system remains locked in recovery until the spacecraft completes its reorientation and satisfies the exit condition:

$$x_{\text{alignment}} > \cos(\theta_{\text{exit}}) \quad (5.39)$$

where $\theta_{\text{exit}} = \text{RECOVERY_EXIT_ANGLE_DEG}$.

Geometric Steering Law

When the vehicle operates within "Nadir Recovery Mode", steering is strictly restricted to the body yaw axis (Z -axis) to execute a clean, shortest-rotation "roll" maneuver. The scalar heading error is extracted using a two-argument arctangent of the in-plane components:

$$\theta_{\text{heading}} = \text{atan2}(\hat{\mathbf{r}}_{\text{nadir},B}[1], \hat{\mathbf{r}}_{\text{nadir},B}[0]) \quad (5.40)$$

In the unique singular case of a perfect 180° inversion ($\hat{\mathbf{r}}_{\text{nadir},B} = [-1 \ 0 \ 0]^T$), where the arctangent becomes mathematically undefined, the software forces an explicit escape value of $\theta_{\text{heading}} = \pi$. The resulting recovery torque vector becomes:

$$\boldsymbol{\tau}_{\text{cmd}} = [0.0 \ 0.0 \ K_P I_{zz} \theta_{\text{heading}}]^T \quad (5.41)$$

where $K_P = \text{KP_NADIR}$ and I_{zz} is the principal yaw moment of inertia.

When the vehicle operates in "Normal Nadir Pointing Mode" (outside recovery), the steering law uses an axis-angle geometric-error formulation to align the body $+X$ axis with the predicted nadir vector. The absolute angular tracking error θ is determined via:

$$\theta = \arccos(\hat{\mathbf{x}}_B \cdot \hat{\mathbf{r}}_{\text{nadir},B}) \quad (5.42)$$

To avoid actuator chatter, small-angle limit cycles, and unnecessary electrical activity, a deadband threshold is enforced:

$$\theta_{\text{deadband}} = \text{NADIR_POINTING_DEADBAND_DEG} \quad (5.43)$$

If $\theta < \theta_{\text{deadband}}$, the desired control torque is set to zero ($\boldsymbol{\tau}_{\text{cmd}} = \mathbf{0}$). For errors exceeding the deadband threshold, the required unnormalized rotation axis $\mathbf{e}$ is isolated via a vector cross product:

$$\mathbf{e} = \hat{\mathbf{x}}_B \times \hat{\mathbf{r}}_{\text{nadir},B} = \begin{bmatrix} 0.0 & -\hat{\mathbf{r}}_{\text{nadir},B}[2] & \hat{\mathbf{r}}_{\text{nadir},B}[1] \end{bmatrix}^T \quad (5.44)$$

Conditioned upon a non-singular axis norm ($\|\mathbf{e}\| > 10^{-12}$), the unit direction vector $\hat{\mathbf{e}} = \mathbf{e}/\|\mathbf{e}\|$ is constructed to form the full three-dimensional rotation vector:

$$\boldsymbol{\theta}_{\text{vec}} = \theta \hat{\mathbf{e}} \quad (5.45)$$

The final desired three-axis steering torque vector is then mapped through the entire spacecraft inertia tensor $\mathbf{I}$:

$$\boldsymbol{\tau}_{\text{cmd}} = K_P \mathbf{I} \boldsymbol{\theta}_{\text{vec}} \quad (5.46)$$

Because the cross product of $\hat{\mathbf{x}}_B$ and $\hat{\mathbf{r}}_{\text{nadir},B}$ naturally yields a zero component along the body X -axis, no direct "yaw" torque is generated.

Magnetic Moment Calculation

The desired control torque $\boldsymbol{\tau}_{\text{cmd}}$ must be projected into the available actuation space since magnetorquers can only apply torques orthogonal to the local geomagnetic field vector $\mathbf{B}$ via the relation $\boldsymbol{\tau} = \mathbf{m} \times \mathbf{B}$.

Using a vector cross product using the predicted local magnetic field vector:

$$\mathbf{m} = \frac{\mathbf{B} \times \boldsymbol{\tau}_{\text{cmd}}}{\|\mathbf{B}\|^2} \quad (5.47)$$

The implemented operational logic is summarized in Table 5.7.

Table 5.7: Nadir-controller operational logic summaries

Operational State	Parameter Criterion	Software Execution Action
Recovery Entry	$\hat{\mathbf{r}}_{\text{nadir},B}[0] < \cos(\theta_{\text{enter}})$	Enable single-axis Z -body recovery steering
Recovery Exit	$\hat{\mathbf{r}}_{\text{nadir},B}[0] > \cos(\theta_{\text{exit}})$	Disable recovery; restore full cross-axis steering
Deadband Suppression	$\theta < \theta_{\text{deadband}}$	Force $\boldsymbol{\tau}_{\text{cmd}} = \mathbf{0}$ to eliminate steady-state actuator chatter

5.2.4 Dipole Command Pipeline and Actuator Command

This section defines the command pipeline that converts controller outputs into magnetorquer dipole commands that account for finite resolution and saturation.

The implemented command chain performs:

$$\text{Controllers} \rightarrow \text{Arbitration} \rightarrow \text{Conditioning} \rightarrow \text{MTB Execution}$$

The resulting architecture centralizes controller arbitration, conditioning, and moment application.

Dipole conditioning refers to enforcing actuator limits and realization constraints before magnetorquer actuation.

Controller Arbitration

Controller outputs are combined into a single unconstrained dipole command before actuator conditioning based on the active flight software mode.

The implemented arbitration modes are summarized in Table 5.8.

Table 5.8: Dipole-command arbitration modes

Mode	Command structure	Purpose
BDOT	$\mathbf{m}_{total} = \mathbf{m}_{BDOT}$	Angular-rate damping
NADIR_POINTING	$\mathbf{m}_{total} = \mathbf{m}_{BDOT} + \mathbf{m}_{nadir}$	Combined damping and nadir steering

Continuous magnetic damping remains active during nadir-pointing operation to suppress residual angular velocity and improve transient closed-loop stability.

The implemented conditioning mechanisms are summarized in Table 5.9.

Table 5.9: Dipole-command conditioning mechanisms

Conditioning mechanism	Purpose
Uniform saturation	Preserve dipole direction during clipping
Finite-resolution quantization	Model discrete MTB command resolution
Magnitude clipping	Enforce hard actuator limits
Zero-dipole fallback	Provide deterministic actuator suppression

Saturation Scaling

Following arbitration, actuator saturation limits are applied using the maximum allowable dipole capability:

$$\mathbf{m}_{max} = [m_{x,max} \quad m_{y,max} \quad m_{z,max}]^T \quad (5.48)$$

If any axis exceeds its allowable dipole magnitude, the commanded vector is uniformly scaled to ensure the physical limits are not violated:

$$\mathbf{m}_{sat} = \frac{\mathbf{m}}{\max\left(1.0, \max_i \left(\frac{|m_i|}{m_{i,max}}\right)\right)} \quad (5.49)$$

Uniform saturation scaling preserves the commanded dipole direction during actuator clipping, preventing cross-coupling of control authority into unintended directions.

The resulting conditioning pipeline preserves realizable actuator behavior while enforcing finite magnetorquer authority limits.

Finite-Resolution Quantization

Continuous dipole commands are converted into discrete actuator states that represent finite-resolution magnetorquer behavior using a floor-based step resolution mapping.

The dipole resolution step size is defined as Δm_i . The downward-directed quantization is implemented through:

$$m_{quant,i} = \text{sign}(m_{cont,i}) \cdot \Delta m_i \cdot \left\lfloor \frac{|m_{cont,i}|}{\Delta m_i} \right\rfloor \quad (5.50)$$

with the hard maximum ceiling enforced via numerical clipping immediately following discretization:

$$|m_{quant,i}| \leq m_{i,max} \quad (5.51)$$

The realized actuator, therefore, behaves as a finite-resolution digital dipole source rather than an ideal continuous actuator.

Hardware Implementation Distinction: In this flight software simulation model, the dipole command $\mathbf{m}_{quant}$ varies in magnitude and is assumed to be applied continuously at that fixed level throughout the actuation window. On the physical spacecraft hardware, magnetorquers will operate at a constant, maximum physical magnetic dipole magnitude for varying lengths of time instead. By adjusting the actuation window duration, the hardware achieves the same net average torque. This hardware-level approach, in fact, minimizes the already low geometric misalignment error caused by the vehicle rotating relative to the geomagnetic field vector during the actuation window, rendering the physical application slightly more robust than an idealized constant-magnitude simulation block.

5.3 Simulation Architecture and Physical Modelling

5.3.1 Integrated Simulation Architecture

This section defines the closed-loop Basilisk simulation architecture used for integrated software-in-the-loop verification of the ADCS.

The simulation environment couples spacecraft dynamics propagation, environmental modeling, sensor simulation, flight-software execution, and actuator physics within a unified message-based simulation framework. The two distinct architectures are for the de-tumbling scenarios and nadir pointing scenarios, where the latter has a larger timestep as the angular velocities are lower.

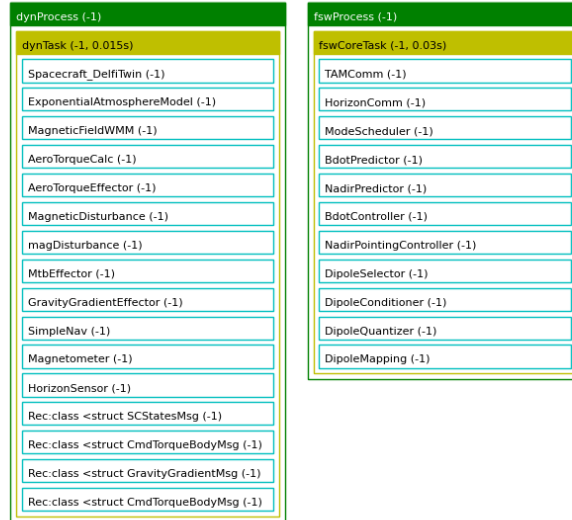


Figure 5.4: Integrated closed-loop Basilisk simulation architecture for de-tumbling mode



Figure 5.5: Integrated closed-loop Basilisk simulation architecture for nadir pointing mode

The simulation environment was designed as a physically representative closed-loop verification environment rather than a flight-qualified digital twin.

Integrated Simulation Components

The primary simulation subsystems are summarized in Table 5.10.

Table 5.10: Integrated simulation subsystems

Subsystem	Function
Dynamics propagation	Spacecraft translational and rotational propagation
Environmental modeling	Gravity, geomagnetic field, atmospheric, and disturbance propagation
Sensor simulation	Navigation and magnetic-field measurement generation
Flight software	Scheduling, estimation, and controller execution
Actuator realization	Magnetic dipole torque realization

Process Partitioning

The Basilisk simulation architecture is partitioned into a spacecraft dynamics process and a flight-software process. The responsibilities and features of each are summarized in Table 5.11.

Table 5.11: Basilisk process partitioning and attributes

Process	Responsibilities and Capabilities
Dynamics process	Rigid-body propagation, orbital propagation, environmental modeling, disturbance generation, sensor-truth generation, and actuator realization. Executes deterministically at $\Delta t_{dyn} = 0.5 * \Delta t_{fsw}$.
Flight-software process	Scheduler execution, reference and measurement processing, controller execution, and magnetic command generation. Executes modularly every $\Delta t_{fsw}[s]$.

This timing preserves the separation between continuous spacecraft propagation and discrete onboard software execution. While the Basilisk environment executes this architecture through scheduled module updates and message passing, the embedded implementation follows the same logical layout but is intended for execution on the STM32 microcontroller using real-time scheduling.

Message-Based Coupling

All subsystem interaction is managed through Basilisk message interfaces. The primary message sets are summarized in Table 5.12.

Table 5.12: Primary Basilisk message categories

Message Category	Function
Spacecraft-state messages	Distribute spacecraft translational and rotational states
Environmental-state messages	Provide geomagnetic-field and disturbance information
Sensor-output messages	Transmit simulated measurements to flight software
Reference-state messages	Distribute pointing-reference information
Actuator-command messages	Transmit actuator commands into the spacecraft dynamics process

This message-based SIL (software-in-loop) architecture separates flight software from spacecraft dynamics, supports modular subsystem integration, preserves deterministic execution ordering, and maintains compatibility with asynchronous process execution.

Closed-Loop Execution Flow

The dynamics process propagates translational motion, rotational motion, environmental disturbances, and true sensor states, distributing them to the flight software process via Basilisk message interfaces.

The flight software performs sensor processing, control algorithms, and magnetic command generation. These generated commands are transmitted back into the dynamics process through the magnetorquer effector model, completing the loop while preserving message-level isolation and repeatable software-dynamics interaction.

Spacecraft Dynamics

Spacecraft propagation is implemented using the Basilisk rigid-body model initialized from inertial position, velocity, attitude, body angular velocity, and mass properties. The rotational dynamics satisfy:

$$\mathbf{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\mathbf{I}\boldsymbol{\omega}) = \boldsymbol{\tau}_{ext} \quad (5.52)$$

where the external torque environment includes:

$$\boldsymbol{\tau}_{ext} = \boldsymbol{\tau}_{mtb} + \boldsymbol{\tau}_{dist} \quad (5.53)$$

with $\boldsymbol{\tau}_{mtb}$ representing the magnetic control torque and $\boldsymbol{\tau}_{dist}$ representing combined environmental disturbance torques defined in the environment-model section. This enabling layout allows direct evaluation of spacecraft response under simultaneous actuator and disturbance loading.

5.3.2 State Initialization

This section defines orbit and attitude initialization, as well as the reference-frame conventions used throughout the integrated ADCS simulation.

The resulting system establishes the relationships between:

- orbital motion,
- spacecraft body axes,

- nadir direction,
- velocity direction,

These definitions are used consistently throughout the simulation and verification architecture.

Orbital Initialization

The spacecraft orbit is initialized using the Keplerian-element set:

$$\mathcal{O} = \{a, e, i, \Omega, \omega, f\} \quad (5.54)$$

The orbital configuration parameters are summarized in Table 5.13.

Table 5.13: Nominal orbital configuration

Orbital element	Symbol	Description
Semi-major axis	a	Orbital radius definition
Eccentricity	e	Orbit circularity parameter
Inclination	i	Orbital-plane inclination
RAAN	Ω	Ascending-node orientation
Argument of periapsis	ω	In-plane periapsis rotation
True anomaly	f	Initial orbital position

The inertial Cartesian spacecraft state is generated through:

$$(\mathbf{r}_N, \mathbf{v}_N) = \text{orbitalMotion.elem2rv}(\mu, \mathcal{O}) \quad (5.55)$$

where:

- $\mathbf{r}_N$ is the inertial position vector,
- and $\mathbf{v}_N$ is the inertial velocity vector.

These vectors define the orbital geometry used throughout the integrated spacecraft simulation.

Reference-Frame Definitions

The simulation uses the reference frames summarized in Table 5.14.

Table 5.14: Reference-frame definitions

Frame	Description
$\mathcal{F}_N$	Earth-centered inertial reference frame used for orbital propagation
$\mathcal{F}_B$	Spacecraft body-fixed reference frame used for spacecraft dynamics and control
$\mathcal{F}_{orbit}$	Orbital geometry frame constructed from nadir and velocity directions

The spacecraft body frame is defined as:

$$\mathcal{F}_B = \{\hat{\mathbf{x}}_B, \hat{\mathbf{y}}_B, \hat{\mathbf{z}}_B\} \quad (5.56)$$

with:

- origin at the center of the geometry of the main bus,
- $+\hat{\mathbf{x}}_B$ and $+\hat{\mathbf{y}}_B$ along sides perpendicular to the long axis, aligned with the horizon sensors.
- $+\hat{\mathbf{z}}_B$ aligned with the long axis of the satellite, pointed away from solar panel deployment direction.

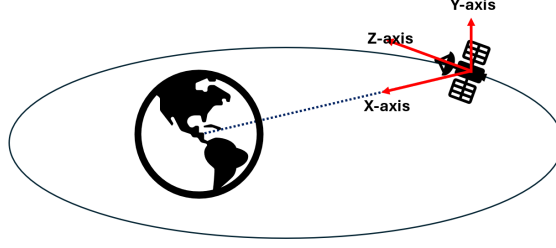


Figure 5.6: Spacecraft body reference frame

The inertial-to-body transformation is represented through:

$$\mathbf{C}_{BN} \quad (5.57)$$

which maps vectors from:

$$\mathcal{F}_N \rightarrow \mathcal{F}_B \quad (5.58)$$

These frame definitions are used consistently throughout spacecraft propagation, environmental modeling, sensor simulation, control algorithm, actuator use, and verification post-processing. It is visualized in ??

Orbital Reference Frame

The normalized radial and velocity directions are defined.

These directions define the orbital geometry used throughout:

- nadir-reference generation,
- environmental modeling,
- disturbance evaluation,
- and spacecraft pointing assessment.

The nominal spacecraft alignment is initialized such that:

$$+\hat{x}_B \rightarrow \text{nadir} \quad (5.59)$$

$$+\hat{z}_B \rightarrow \text{velocity direction} \quad (5.60)$$

The body-frame basis vectors are constructed through:

$$\hat{\mathbf{x}}_B = -\frac{\mathbf{r}_N}{\|\mathbf{r}_N\|} \quad (5.61)$$

$$\hat{\mathbf{z}}_B = \frac{\mathbf{v}_N}{\|\mathbf{v}_N\|} \quad (5.62)$$

$$\hat{\mathbf{y}}_B = \hat{\mathbf{z}}_B \times \hat{\mathbf{x}}_B \quad (5.63)$$

forming:

$$\mathbf{C}_{BN} = \begin{bmatrix} \hat{\mathbf{x}}_B \\ \hat{\mathbf{y}}_B \\ \hat{\mathbf{z}}_B \end{bmatrix} \quad (5.64)$$

This establishes the reference while preserving a right-handed body-frame basis.

Initial Attitude Construction

To avoid equilibrium initialization, an initial attitude perturbation is applied through:

$$\mathbf{C}_{BN,0} = \mathbf{C}_{perturb} \mathbf{C}_{BN} \quad (5.65)$$

The perturbed attitude is subsequently converted into Modified Rodrigues Parameters (MRPs) for spacecraft initialization.

5.3.3 Spacecraft Physical Model

This section defines the spacecraft's physical model used by the integrated ADCS simulation environment.

The model provides solar-panel dependent mass properties, inertia calculation, and aerodynamic surface geometry definition.

The resulting representation preserves geometric consistency between spacecraft geometry, disturbance-torque application, actuator sizing, and rotational dynamics propagation.

Spacecraft Geometry

The spacecraft geometry is parameterized using:

$$L_x, \quad L_y, \quad L_z, \quad L_p, \quad \theta_{panel} \quad (5.66)$$

where:

- L_x , L_y , and L_z define the spacecraft bus dimensions,
- L_p defines the solar-panel span length,
- and θ_{panel} defines the solar-panel deployment angle wrt $+\hat{\mathbf{z}}_B$.

The implemented geometric representation is summarized in Table 5.15.

Table 5.15: Spacecraft geometry representation

Component	Representation
Spacecraft bus	Rigid body-frame geometry
Solar panels	Deployment-dependent appendages
Aerodynamic surfaces	Facet-based planar surface representation
Surface normals	Outward-facing body-frame facet vectors
Surface centers	Body-frame disturbance reference locations

The spacecraft bus geometry is constructed from body-frame corner points, which define the rigid spacecraft surfaces used throughout the simulation environment.

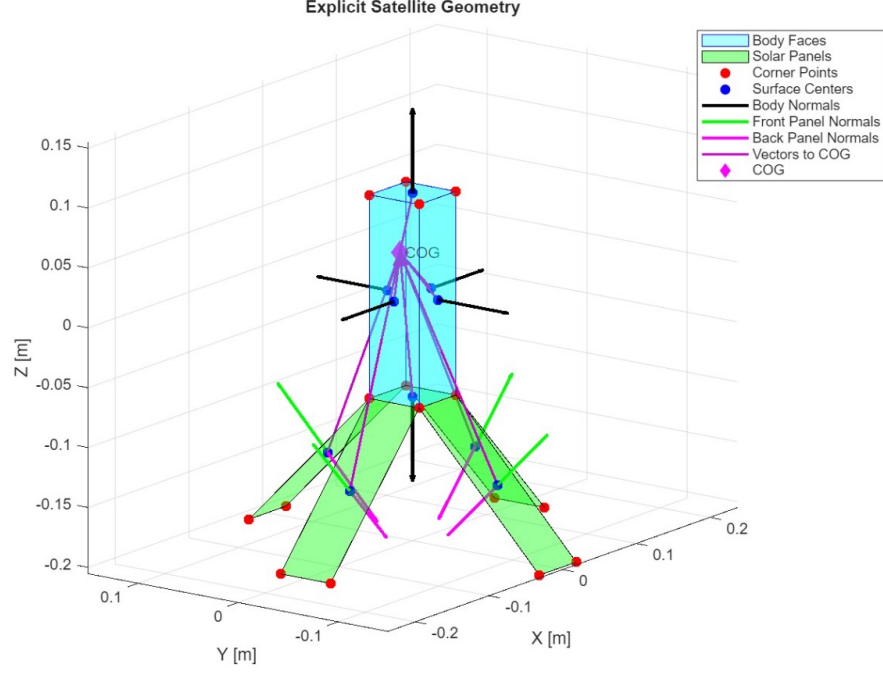


Figure 5.7: Spacecraft geometry and aerodynamic surface representation.

The resulting geometric model defines the spacecraft surfaces used for aerodynamic disturbance evaluation and inertia assembly.

Deployable Solar-Panel Representation

The solar panels are modeled as deployment-dependent rectangular appendages attached to the spacecraft body frame.

The deployment geometry is generated through:

$$\mathbf{p}_{panel} = \begin{bmatrix} x \pm L_p \sin \theta_{panel} \\ y \\ -\frac{L_z}{2} - L_p \cos(180^\circ - \theta_{panel}) \end{bmatrix} \quad (5.67)$$

Each spacecraft surface is represented using a four-corner planar facet:

$$S_i = \{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4\} \quad (5.68)$$

Independent front and rear panel surfaces are retained to support bidirectional aerodynamic interaction during arbitrary spacecraft attitudes.

The deployment geometry, therefore, modifies:

- aerodynamic surface exposure,
- disturbance-torque moment arms,
- spacecraft center-of-gravity location,
- and spacecraft inertia properties.

This representation preserves geometric consistency between appendage deployment, disturbance modeling, and rotational dynamics.

Aerodynamic Surface Representation

Aerodynamic disturbances are evaluated using the facet-based surface representation summarized in Table 5.16.

Table 5.16: Aerodynamic surface representation

Quantity	Definition
Surface center	Geometric centroid of the aerodynamic surface
Surface normal	Normalized outward-facing facet vector
Surface area	Planar aerodynamic area
Drag coefficient	Assigned flat-plate aerodynamic coefficient
Body-frame location	Disturbance-torque reference location

Facet normals are computed from the surface corner geometry and exported together with aerodynamic surface areas, aerodynamic coefficients, and body-frame disturbance locations for downstream disturbance evaluation. Shadowing of aerodynamic surfaces was not implemented because the existing model is representative, in that the applied aerodynamic disturbance would be higher than the actual values, especially under worst-case conditions.

These quantities are subsequently consumed by the environmental disturbance models introduced later.

Mass Properties and Inertia Representation

The spacecraft mass-property architecture is summarized in Table 5.17.

Table 5.17: Spacecraft mass-property inputs

Quantity	Description
m_{bus}	Spacecraft bus mass
r_{bus}	Bus center-of-mass location
$\mathbf{I}_{bus}$	Spacecraft bus inertia tensor
m_{panel}	Solar-panel mass
θ_{panel}	Solar-panel deployment angle

The implemented formulation:

- treats spacecraft bus inertia as fixed engineering input data,
- models deployable appendages explicitly,
- and updates spacecraft inertia using deployment geometry.

All quantities are expressed in the spacecraft body frame:

$$\mathcal{F}_B \tag{5.69}$$

The spacecraft bus inertia tensor is treated as authoritative engineering input data to preserve consistency between:

- CAD-derived spacecraft properties,
- disturbance-torque evaluation,
- actuator sizing,
- and closed-loop spacecraft dynamics.

Solar-Panel Inertia Modelling

Solar-panel dimensions are extracted directly from the geometric model through:

$$L = \|\mathbf{p}_2 - \mathbf{p}_1\| \quad (5.70)$$

$$W = \|\mathbf{p}_4 - \mathbf{p}_1\| \quad (5.71)$$

A local orthonormal panel frame is constructed as:

$$\mathbf{R}_{panel} = [\hat{\mathbf{x}} \quad \hat{\mathbf{y}} \quad \hat{\mathbf{z}}] \quad (5.72)$$

Each panel is modeled as a thin rectangular plate with a local inertia tensor:

$$\mathbf{I}_{panel,local} = \frac{m}{12} \begin{bmatrix} W^2 + t^2 & 0 & 0 \\ 0 & L^2 + t^2 & 0 \\ 0 & 0 & L^2 + W^2 \end{bmatrix} \quad (5.73)$$

The corresponding body-frame panel inertia becomes:

$$\mathbf{I}_{panel,B} = \mathbf{R}_{panel} \mathbf{I}_{panel,local} \mathbf{R}_{panel}^T \quad (5.74)$$

This representation captures deployment-dependent inertia variation introduced by rotating appendage geometry.

Center-of-Gravity and Inertia Assembly

The spacecraft's center of mass is computed through:

$$\mathbf{r}_{CM} = \frac{\sum_i m_i \mathbf{r}_i}{\sum_i m_i} \quad (5.75)$$

The parallel-axis theorem is applied through:

$$\mathbf{I}_{shifted} = \mathbf{I}_{CM} + m \left(\|\mathbf{r}\|^2 \mathbf{I} - \mathbf{r} \mathbf{r}^T \right) \quad (5.76)$$

The final spacecraft inertia tensor becomes:

$$\mathbf{I}_{B_c/B} = \mathbf{I}_{bus} + \sum_i \mathbf{I}_{panel,i} \quad (5.77)$$

The resulting spacecraft model captures:

- deployment-dependent inertia variation,
- center-of-mass shifts,
- asymmetric appendage loading,
- geometry-dependent disturbance moment arms,
- and deployment-dependent rotational response.

This preserves consistent rigid-body dynamics throughout spacecraft propagation, disturbance evaluation, and closed-loop ADCS simulation.

5.3.4 Environment Models

This section defines the external environmental conditions for a spacecraft in Low Earth Orbit.

Environmental Model Architecture

The primary environmental models are summarized in Table 5.18.

The simulation environment uses prebuilt Basilisk environmental modules to provide coupled orbital, magnetic, and atmospheric interactions throughout closed-loop propagation.

Table 5.18: Environmental model architecture

Environment	Model	Purpose
Gravity	<code>GGM03S_J2_only</code>	Orbital propagation with Earth-oblateness perturbations
Geomagnetic field	<code>magneticFieldWMM.MagneticFieldWMM()</code>	Time-varying geomagnetic-field propagation
Atmosphere	<code>exponentialAtmosphere.ExponentialAtmosphere()</code>	Exponential atmospheric-density propagation

The implemented environment preserves repeatable spacecraft-environment coupling while maintaining representative low-Earth-orbit disturbance behaviour.

Gravity Environment

The gravity environment uses the Basilisk spherical-harmonics gravity framework with a reduced `GGM03S_J2_only` configuration containing central gravity and J2 secular perturbations.

The gravitational acceleration is dominated by the inverse-square central-body term, while the J2 contribution captures first-order Earth-oblateness effects responsible for secular orbital-element drift and orbital-plane precession.

The resulting orbital propagation captures orbital-frame precession, low-Earth-orbit secular drift, and dominant Earth-oblateness effects.

J2 perturbations are particularly important for preserving realistic orbital-frame evolution and nadir-reference geometry throughout long-duration closed-loop simulations.

Geomagnetic Environment

The geomagnetic environment uses the Basilisk `magneticFieldWMM` module, which evaluates the Earth magnetic field using the World Magnetic Model (WMM). The model computes the geomagnetic field as a function of spacecraft position relative to the Earth-fixed frame.

The inertial geomagnetic field:

$$\mathbf{B}_N \quad (5.78)$$

varies continuously with spacecraft orbital position and geomagnetic latitude.

The resulting field therefore changes both spatially along the orbit and through spacecraft attitude-dependent frame transformations.

The corresponding body-frame magnetic field becomes:

$$\mathbf{B}_B = \mathbf{C}_{BN} \mathbf{B}_N \quad (5.79)$$

The resulting geomagnetic field is distributed throughout the simulation environment for:

- magnetic-field sensing,
- scaled B-dot damping,
- magnetic steering,
- and residual magnetic disturbance evaluation.

The geomagnetic field epoch remained fixed throughout the simulation campaign to maintain deterministic, repeatable conditions for controller evaluation.

Because the ADCS uses magnetorquer-only actuation, the geomagnetic environment directly determines both available control authority and magnetic disturbance loading throughout orbit propagation.

Atmospheric Environment

The atmospheric environment uses the Basilisk `ExponentialAtmosphere` module, which implements an isothermal exponential-density atmosphere model for low-Earth-orbit density estimation.

The atmospheric density is modeled as:

$$\rho(h) = \rho_0 \exp\left(-\frac{h}{H}\right) \quad (5.80)$$

where ρ_0 is the reference atmospheric density and H is the atmospheric scale height.

This formulation follows the standard exponential atmosphere assumption derived from hydrostatic equilibrium under an isothermal approximation.

The resulting atmospheric density is used to compute aerodynamic forces and disturbance torques.

Although simplified, the exponential model preserves the dominant aerodynamic-loading trends required for low-Earth-orbit disturbance evaluation.

5.3.5 Disturbance Models

This section defines the disturbance torques acting on the spacecraft throughout low-Earth-orbit operation. These models provide the external forcing inputs coupled into the spacecraft rotational dynamics simulation.

Disturbance Torque Architecture

The total environmental disturbance torque becomes:

$$\boldsymbol{\tau}_{dist} = \boldsymbol{\tau}_{aero} + \boldsymbol{\tau}_{gg} + \boldsymbol{\tau}_{mag} \quad (5.81)$$

The implemented disturbance models are summarized in Table 5.19.

Table 5.19: Environmental disturbance models

Disturbance source	Physical mechanism
Aerodynamic torque	Atmospheric drag acting through aerodynamic-center offsets
Gravity-gradient torque	Orbital gravity differential acting on distributed spacecraft inertia
Residual magnetic torque	Static body-frame magnetic dipole interacting with the geomagnetic field

The resulting disturbance architecture couples environmental forcing directly into the spacecraft’s rotational dynamics propagation.

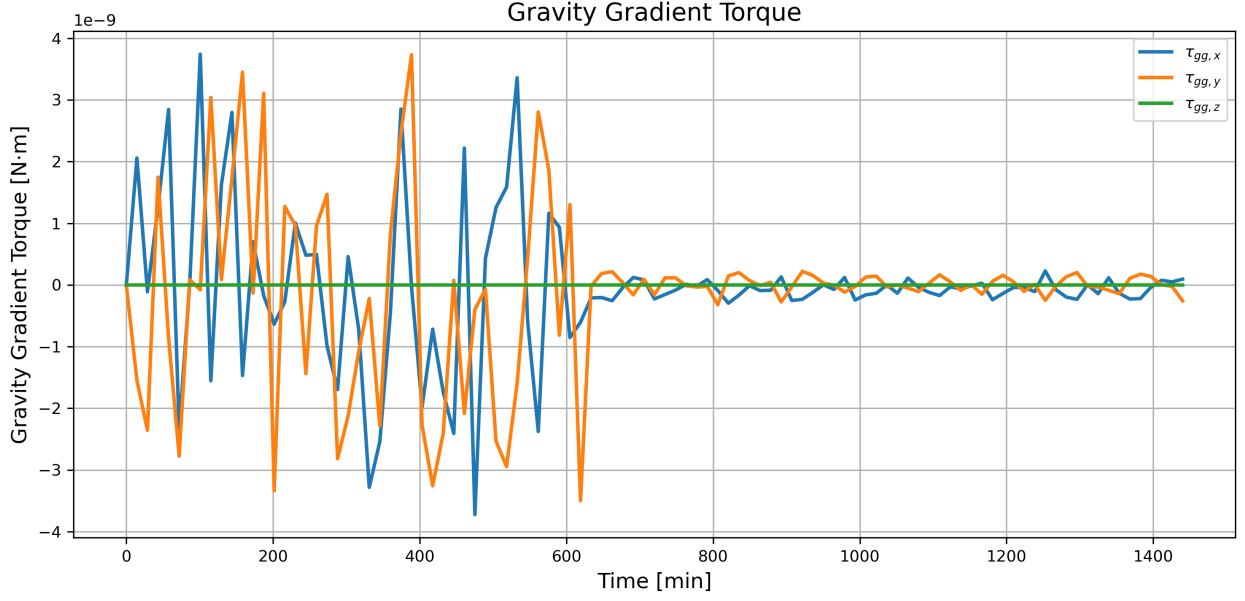


Figure 5.8: Gravity gradient torque for 300 km SSO with 90 deg solar panel deployment angle

Gravity-Gradient Torque

The gravity-gradient disturbance torque is computed using the Basilisk `GravityGradientEffector` module, which implements the first-order gravity-gradient formulation for rotational dynamics coupling.

The resulting disturbance torque follows:

$$\boldsymbol{\tau}_{gg} = 3 \frac{\mu}{r^3} \hat{\mathbf{r}} \times (\mathbf{I} \hat{\mathbf{r}}) \quad (5.82)$$

where:

- μ is the Earth's gravitational parameter,
- r is the spacecraft radial distance from the Earth centre,
- $\hat{\mathbf{r}}$ is the spacecraft radial direction,
- and $\mathbf{I}$ is the spacecraft inertia tensor.

This first-order formulation captures the coupling between the spacecraft mass distribution and the local gravitational field gradient.

The disturbance varies continuously with orbital position, spacecraft attitude, and deployment-dependent inertia asymmetry.

The resulting torque couples the spacecraft mass distribution directly into the closed-loop rotational dynamics.

Aerodynamic Disturbance Torque

Aerodynamic disturbance torques are computed using a custom Basilisk disturbance module implementing a facet-based flat-plate drag formulation over the explicit spacecraft geometry defined previously.

The aerodynamic geometry is constructed using individual spacecraft body and solar-panel facets, each containing:

- facet area,
- drag coefficient,

- surface normal,
- and facet center location.

The body-frame aerodynamic velocity becomes:

$$\mathbf{v}_B = \mathbf{C}_{BN}\mathbf{v}_N \quad (5.83)$$

The corresponding dynamic pressure is:

$$q = \frac{1}{2}\rho \|\mathbf{v}_B\|^2 \quad (5.84)$$

For facet i , the projected-area visibility term becomes:

$$\cos \theta_i = \hat{\mathbf{n}}_i \cdot \hat{\mathbf{v}}_B \quad (5.85)$$

where $\hat{\mathbf{n}}_i$ is the facet surface normal.

Back-facing surfaces satisfying:

$$\cos \theta_i \leq 0 \quad (5.86)$$

are excluded from aerodynamic loading calculations.

The aerodynamic drag force magnitude acting on facet i becomes:

$$F_{D,i} = qC_{D,i}A_i \cos \theta_i \quad (5.87)$$

The resulting aerodynamic force vector becomes:

$$\mathbf{F}_{D,i} = -F_{D,i}\hat{\mathbf{v}}_B \quad (5.88)$$

The aerodynamic torque arm is computed relative to the total spacecraft centre of gravity:

$$\mathbf{r}_i = \mathbf{r}_{facet,i} - \mathbf{r}_{CM} \quad (5.89)$$

producing aerodynamic disturbance torque:

$$\boldsymbol{\tau}_{D,i} = \mathbf{r}_i \times \mathbf{F}_{D,i} \quad (5.90)$$

The total aerodynamic disturbance torque becomes:

$$\boldsymbol{\tau}_{aero} = \sum_i \boldsymbol{\tau}_{D,i} \quad (5.91)$$

This formulation captures deployment-dependent aerodynamic loading, asymmetric drag behavior, centre-of-mass shifts, and geometry-dependent disturbance-torque generation throughout spacecraft deployment configurations.

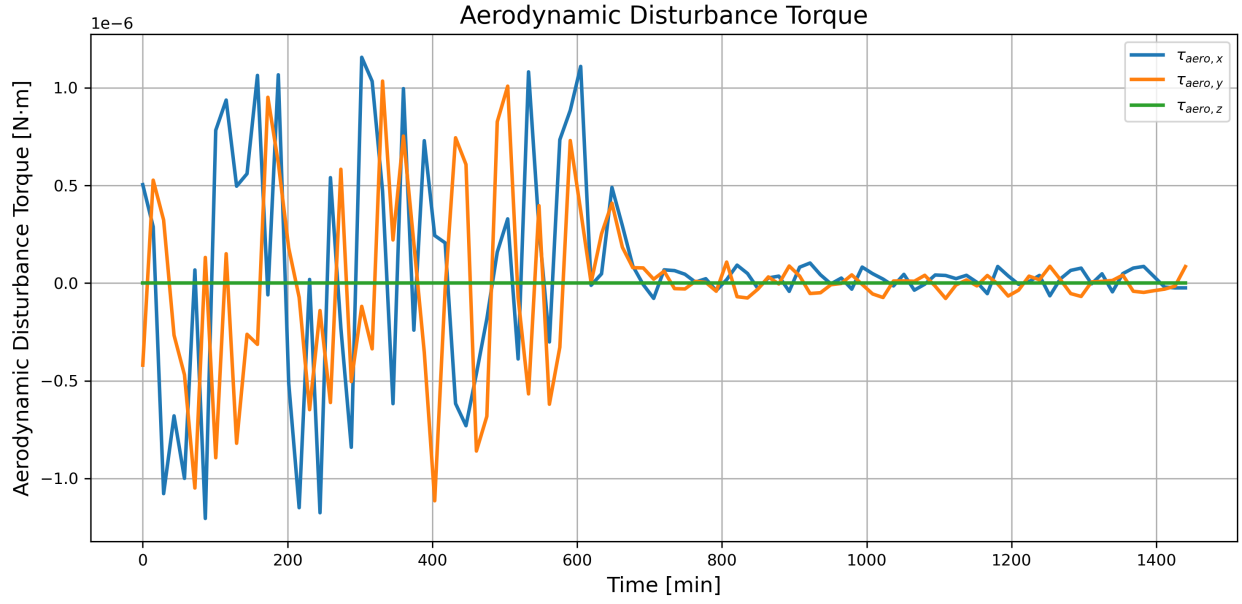


Figure 5.9: Aerodynamic Disturbance torque for 300 km SSO with 90 deg solar panel deployment angle

Residual Magnetic Disturbance Torque

Residual magnetic disturbance torque is modeled using a fixed body-frame magnetic dipole:

$$\mathbf{m}_{res,B} = [m_x \ m_y \ m_z]^T \quad (5.92)$$

interacting with the local body-frame geomagnetic field.

The resulting disturbance torque becomes:

$$\boldsymbol{\tau}_{mag} = \mathbf{m}_{res,B} \times \mathbf{B}_B \quad (5.93)$$

The disturbance, therefore, varies continuously with:

- orbital position,
- spacecraft attitude,
- and geomagnetic-field orientation.

This model reproduces the dominant time-varying magnetic disturbance loading experienced by magnetically actuated spacecraft throughout orbit propagation.

Environment Coupling Summary

The implemented environment models reproduce the dominant external forcing conditions acting on the spacecraft throughout the simulation campaign.

These disturbances are directly coupled into the spacecraft's rotational dynamics, enabling a physically representative closed-loop ADCS evaluation under time-varying orbital environmental conditions.

5.3.6 Sensor and Actuator Simulation

This section defines the sensing and actuation models used to couple spacecraft dynamics with closed-loop flight-software execution.

The implemented models provide:

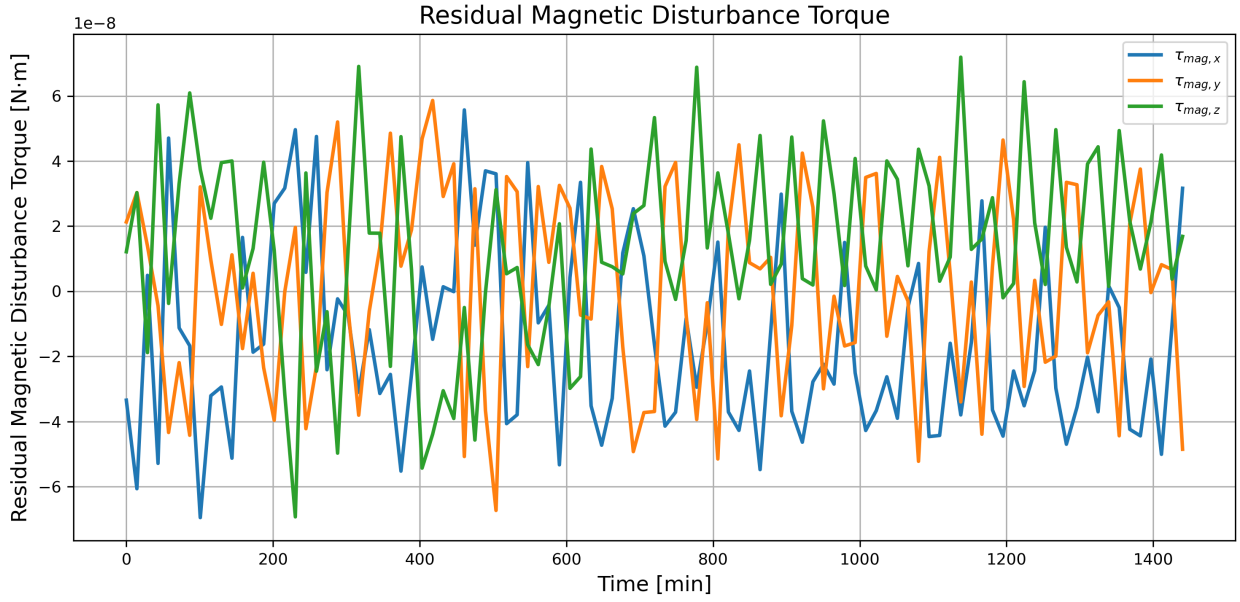


Figure 5.10: Magnetic disturbance torque for 300 km SSO with 90 deg solar panel deployment angle

- body-frame magnetic-field sensing using magnetometer model,
- body-frame nadir-reference sensing using horizon sensor model,
- and magnetic torque realization using magnetorquer model.

Integrated Sensor and Actuator Architecture

The implemented sensing and actuation chain is summarized in Table 5.20.

Table 5.20: Sensor and actuator simulation models

Subsystem	Model	Function
Navigation reconstruction	<code>simpleNav.SimpleNav()</code>	Idealized navigation-state reconstruction
Magnetometer	<code>magnetometer.Magnetometer()</code>	Body-frame magnetic-field sensing
Earth horizon sensor	Custom Basilisk module	Body-frame nadir-reference reconstruction
Magnetorquer effector	<code>MtbEffector.MtbEffector()</code>	Magnetic torque realization

The resulting execution chain becomes:

Dynamics → Sensors → Flight Software → Dipole Commands → MTB Torque

Sensor measurements are acquired during scheduled sensing intervals while commanded magnetorquer actuation remains disabled. This preserves ACT-5 scheduler separation between magnetic sensing and magnetic actuation.

Magnetometer Model

Body-frame magnetic-field sensing used the Basilisk three-axis magnetometer module:

$$\text{magnetometer.Magnetometer()} \quad (5.94)$$

The module receives the inertial geomagnetic field from the environmental magnetic-field model and transforms it into the spacecraft body frame using the spacecraft attitude solution.

The measured body-frame magnetic field becomes:

$$\mathbf{B}_B \quad (5.95)$$

The magnetometer is used to sense the magnetic field and its local time history.

The implemented model includes:

- white measurement noise,
- bounded Gauss–Markov drift,
- constant sensor bias,
- and finite sensor saturation.

The sensor measurement model follows:

$$\mathbf{B}_{measured} = (\mathbf{B}_{truth} + \mathbf{e}_{noise} + \mathbf{e}_{bias}) f_{scale} \quad (5.96)$$

where $\mathbf{e}_{noise}$ is the measurement-noise component, $\mathbf{e}_{bias}$ is the sensor bias, and f_{scale} is the sensor scale factor.

The primary magnetometer parameters found via the datasheet and implemented in the simulation are summarized in Table 5.21.

Table 5.21: Magnetometer configuration

Parameter	Value	Description
σ_B	15 nT	White-noise standard deviation
B_{walk}	45 nT	Bounded Gauss–Markov drift magnitude
$ B_{max} $	800 μ T	Magnetometer saturation limit

The sensor output is distributed through:

$$\text{TAMSensorBodyMsg} \quad (5.97)$$

to downstream controller modules.

The resulting model captures bounded magnetic-field sensing uncertainty while preserving deterministic closed-loop measurement behavior throughout the simulation campaign.

Earth Horizon Sensor Model

The Earth horizon sensor generates body-frame nadir-reference vectors for geometric nadir-pointing control.

The implemented sensor uses propagated spacecraft position and attitude states to reconstruct the nadir direction geometrically in the spacecraft body frame.

The inertial-frame nadir direction becomes:

$$\hat{\mathbf{r}}_{nadir,N} = -\frac{\mathbf{r}_{BN,N}}{\|\mathbf{r}_{BN,N}\|} \quad (5.98)$$

The corresponding body-frame nadir-reference vector becomes:

$$\hat{\mathbf{r}}_{nadir,B} = \mathbf{C}_{BN} \hat{\mathbf{r}}_{nadir,N} \quad (5.99)$$

where $\mathbf{C}_{BN}$ is generated from the spacecraft Modified Rodrigues Parameters.

The primary horizon-sensor inputs are summarized in Table 5.22.

Table 5.22: Earth horizon sensor inputs

Input quantity	Description
$\mathbf{r}_{BN,N}$	Inertial spacecraft position vector
$\boldsymbol{\sigma}_{BN}$	Spacecraft attitude represented using Modified Rodrigues Parameters
$\mathbf{C}_{BN}$	Body-to-inertial attitude transformation matrix

The reconstructed nadir-reference vector is distributed through:

$$\text{BodyHeadingMsg} \quad (5.100)$$

to downstream geometric steering and nadir-pointing control modules.

This architecture provides repeatable body-frame nadir-reference reconstruction compatible with coarse magnetic geometric steering.

Angular Noise Modeling Optional bounded angular uncertainty is introduced using small-angle rotational perturbations:

$$\hat{\mathbf{r}}_{nadir,B}^{noise} = \mathbf{C}_{err} \hat{\mathbf{r}}_{nadir,B} \quad (5.101)$$

where $\mathbf{C}_{err}$ is constructed from bounded roll, pitch, and yaw perturbations.

The perturbed nadir-reference vector is subsequently renormalized before downstream controller execution.

The implemented angular-noise model is summarized in Table 5.23.

Table 5.23: Earth horizon sensor angular-noise model

Noise component	Description
ϕ	Roll-axis angular perturbation
θ	Pitch-axis angular perturbation
ψ	Yaw-axis angular perturbation
Bounded sampling	Uniform worst-case angular-error generation
Vector renormalization	Preservation of unit-vector magnitude after perturbation

Magnetorquer Actuator Model

Magnetic actuation used the Basilisk dynamic-effector module:

$$\text{MtbEffector.MtbEffector()} \quad (5.102)$$

which converts commanded magnetic dipoles into body-frame control torques.

The actuator receives body-frame magnetic dipole commands:

$$\mathbf{m}_B \quad (5.103)$$

and generates magnetic torque:

$$\boldsymbol{\tau}_{mtb} = \mathbf{m}_B \times \mathbf{B}_B \quad (5.104)$$

within the spacecraft dynamics process.

The magnetorquer configuration is summarized in Table 5.24.

Table 5.24: Magnetorquer configuration

Quantity	Value	Description
N_{MTB}	3	Number of orthogonal magnetorquer bars
$\mathbf{G}_{TB}$	$\mathbf{I}_{3 \times 3}$	Body-axis steering matrix
m_{max}	0.0177 A·m ²	Maximum realizable dipole magnitude
Δm	8.5×10^{-4} A·m ²	Dipole-command resolution

Individual torque-bar dipole commands are generated through body-frame dipole commands.

Because magnetic torque generation satisfies:

$$\boldsymbol{\tau}_{mtb} \perp \mathbf{B}_B \quad (5.105)$$

the instantaneous control torque remains underactuated and varies continuously with geomagnetic-field orientation throughout orbit propagation.

The implemented actuator representation does not model detailed electrical dynamics, thermal behavior, hysteresis, or cycle-exact embedded timing behavior. Instead, it provides a deterministic magnetic torque realization suitable for physically representative closed-loop ADCS verification.

5.4 Verification and Analysis

5.4.1 Verification Framework

System-level verification was performed using Basilisk simulation telemetry and automated post-processing tools to evaluate ADCS requirement compliance directly from closed-loop simulation outputs.

The framework provides:

- repeatable closed-loop performance evaluation,
- automated requirement assessment,
- deterministic telemetry processing,
- and direct traceability between simulation outputs and ADCS requirements.

Verification Objectives

The verification campaign evaluates:

- magnetic detumbling performance,
- coarse nadir-pointing behaviour,
- velocity-direction alignment,
- disturbance response,
- and long-duration closed-loop stability.

Performance assessment uses:

- spacecraft angular-rate histories,
- geometric pointing metrics,
- and disturbance-torque telemetry.

Verification Architecture

The implemented verification pipeline follows:

Simulation → Telemetry Export → Post Processing → Requirement Evaluation

The verification architecture is summarized in Table 5.25.

Table 5.25: Verification architecture

Verification stage	Function
Closed-loop simulation	Integrated spacecraft dynamics and flight-software execution
Telemetry export	Recording of spacecraft states, disturbance torques, and controller outputs
Post-processing	Metric extraction, statistical evaluation, and performance visualization
Requirement evaluation	Automated acquisition and threshold-compliance assessment

Requirement Traceability

Verification requirements, evaluation methods, and corresponding analysis sections are summarized in Table 5.26.

Table 5.26: Requirement traceability matrix

Requirement	Verification method	Section
ACT-1	Detumbling statistics and persistent-threshold evaluation	Section 5.4.3
ACT-1.2	Acquisition-time and settling analysis	Section 5.4.3
ACT-2	Velocity-alignment statistical evaluation	Section 5.4.4
ACT-3	Nadir-pointing statistical evaluation	Section 5.4.4
ACT-5	Scheduler sequencing and timing analysis	Section 5.3.5

Requirement Thresholds

The implemented verification thresholds are summarized in Table 5.27.

Table 5.27: ADCS verification requirements

Requirement	Threshold	Evaluated quantity
Nadir-pointing requirement	$\theta_{x,nadir} < 30^\circ$	Nadir-pointing angle
Velocity-alignment requirement	$\theta_{z,v} < 10^\circ$	Velocity-alignment angle
Detumbling requirement	$\ \boldsymbol{\omega}\ < \omega_{req}$	Angular-rate magnitude

Telemetry Architecture

Verification telemetry is exported during simulation execution into structured datasets for automated post-processing and requirement evaluation.

The exported telemetry vector becomes:

$$\mathbf{y} = [t \quad \boldsymbol{\omega} \quad \|\boldsymbol{\omega}\| \quad \boldsymbol{\tau}_{mag} \quad \boldsymbol{\tau}_{gg} \quad \boldsymbol{\tau}_{aero} \quad \theta_{z,v} \quad \theta_{x,nadir}]^T \quad (5.106)$$

The exported telemetry quantities are summarized in Table 5.28.

Table 5.28: Exported verification telemetry quantities

Telemetry quantity	Description
t	Simulation time history
$\omega_x, \omega_y, \omega_z$	Body-frame angular-velocity components
$\ \boldsymbol{\omega}\ $	Angular-rate magnitude
$\boldsymbol{\tau}_{mag}$	Residual magnetic disturbance torque
$\boldsymbol{\tau}_{gg}$	Gravity-gradient disturbance torque
$\boldsymbol{\tau}_{aero}$	Aerodynamic disturbance torque
$\theta_{z,v}$	Body-frame velocity-alignment angle
$\theta_{x,nadir}$	Body-frame nadir-pointing angle

Additional exported metadata includes:

- active controller mode,
- simulation timestep configuration,
- telemetry cadence,
- and logging configuration.

Primary Verification Metrics

The primary verification metrics are summarized in Table 5.29.

Table 5.29: Primary ADCS verification metrics

Metric	Evaluation objective
Body angular velocity magnitude	Rotational-energy dissipation and detumbling compliance
Nadir-pointing angle	Earth-pointing acquisition and maintenance
Velocity-alignment angle	Orbital-frame orientation verification
Disturbance torques	Environmental forcing characterization

For each evaluated requirement, the automated scorecard extracts:

- acquisition time,
- post-acquisition mean,
- post-acquisition standard deviation,
- post-acquisition median,
- post-acquisition maximum,
- in-specification percentage,
- escape count and escape rate,
- and a conservative 1σ envelope metric.

Geometric Pointing Metrics

The velocity-alignment metric becomes:

$$\theta_{z,v} = \cos^{-1}(\hat{\mathbf{z}}_B \cdot \hat{\mathbf{v}}) \quad (5.107)$$

The nadir-pointing metric becomes:

$$\theta_{x,nadir} = \cos^{-1}(\hat{\mathbf{x}}_B \cdot \hat{\mathbf{r}}_{nadir}) \quad (5.108)$$

Successful closed-loop pointing therefore satisfies:

$$\theta_{x,nadir} < 30^\circ \quad (5.109)$$

while velocity alignment satisfies:

$$\theta_{z,v} < 10^\circ \quad (5.110)$$

Detumbling Metrics

Detumbling performance is evaluated using:

$$\|\boldsymbol{\omega}\| = \sqrt{\omega_x^2 + \omega_y^2 + \omega_z^2} \quad (5.111)$$

Requirement acquisition occurs only when:

$$\|\boldsymbol{\omega}\| < \omega_{req} \quad (5.112)$$

remains continuously satisfied throughout a persistent verification window.

The detumbling acquisition time:

$$t_{acq} \quad (5.113)$$

therefore corresponds to the first sustained compliant interval.

Post-acquisition statistical analysis is then performed exclusively on the compliant telemetry segment.

If a requirement threshold is never persistently achieved:

$$x(t) \geq x_{req} \quad \forall t \quad (5.114)$$

the corresponding verification case is reported as failed.

Verification Outputs

The implemented framework produces:

- automated requirement-compliance summaries,
- acquisition-time measurements,
- persistent-threshold compliance statistics,
- disturbance-response visualizations,

5.4.2 Model Limitations and Assumptions

The implemented ADCS simulation environment captures the dominant coupled dynamics associated with magnetorquer-only spacecraft attitude control in low-Earth orbit while introducing several simplifying assumptions to preserve deterministic closed-loop verification behaviour and maintain tractable simulation complexity.

The resulting environment should therefore be interpreted as a physically representative engineering verification environment rather than a flight-qualified spacecraft digital twin.

Primary Modelling Assumptions

The primary modelling assumptions are summarized in Table 5.30.

Table 5.30: Primary modelling assumptions

Subsystem	Simplifying assumption
Atmospheric environment	Static exponential atmosphere without atmospheric co-rotation
Space weather	No solar-cycle or geomagnetic-storm variability
Solar radiation pressure	Neglected entirely
Residual magnetic dipole	Constant body-frame magnetic moment
Magnetorquer realization	Ideal dipole realization without electrical dynamics
Aerodynamics	Constant drag coefficient approximation
Simulation timing	No STM32 runtime jitter or hardware-execution latency
Structural dynamics	Rigid-body spacecraft assumption
Fault modelling	No radiation-induced faults or communication failures

These assumptions define the validity boundaries of the presented closed-loop verification results.

Environmental Simplifications

The environmental propagation models neglect:

- atmospheric co-rotation,
- thermospheric winds,
- local density transport,
- solar-cycle variation,
- geomagnetic storm effects,
- transient space-weather disturbances,
- and solar radiation pressure.

Atmospheric density therefore depends only on orbital altitude, while geomagnetic propagation remains deterministic throughout the simulation campaign.

The disturbance environment consequently includes only aerodynamic disturbance torque, gravity-gradient torque, residual magnetic disturbance torque, and commanded magnetic-control torque.

These assumptions preserve repeatable disturbance conditions while retaining the dominant low-Earth-orbit disturbance mechanisms relevant to magnetic attitude control.

Residual Magnetic Dipole and Actuator Simplifications

The spacecraft residual magnetic moment was assumed constant in the spacecraft body frame:

$$\mathbf{m}_{res,B} = \text{constant} \quad (5.115)$$

The implemented model therefore excludes:

- temperature-dependent magnetization,
- hysteresis drift,
- time-varying onboard current loops,
- payload electrical activity,
- and long-term magnetic aging effects.

The magnetorquer realization model additionally assumes ideal dipole execution:

$$\mathbf{m}_{realized} = \mathbf{m}_{commanded} \quad (5.116)$$

throughout all operating conditions.

The simulation therefore neglects:

- coil inductance,
- current transients,
- driver nonlinearities,
- voltage limitations,
- actuator latency,
- thermal degradation,
- current-control error,
- and detailed electrical power behaviour.

Finite-resolution quantization and actuator saturation remain included within the actuator-realization architecture to preserve realizable magnetic-control behaviour.

Sensor Simplifications

The Earth-horizon sensing architecture provides direct body-frame nadir-reference vectors rather than full infrared horizon reconstruction.

Sensor modelling excludes:

- detailed hardware timing behaviour,
- thermal sensor drift,
- radiation-induced corruption,
- communication latency,
- and sensor synchronization error.

These assumptions prioritize deterministic controller evaluation and repeatable geometric pointing analysis rather than flight-grade navigation realism.

Verification Scope and Interpretation

The implemented simulation environment was developed primarily for ADCS architecture verification, controller development, disturbance analysis, actuator sizing evaluation, and closed-loop performance assessment.

Verification results should therefore be interpreted as closed-loop engineering validation results within the evaluated simulation envelope rather than flight-certification performance guarantees.

The presented results additionally depend on the selected:

- orbital configuration,
- disturbance environment,
- actuator sizing,
- controller gains,
- and initial spacecraft conditions.

Despite the stated assumptions, the integrated environment preserves coupled spacecraft rotational dynamics, realistic magnetic underactuation, disturbance-driven spacecraft behaviour, finite-resolution actuator realization, persistent-threshold verification logic, and deterministic closed-loop flight-software execution.

5.4.3 Detumbling Performance Results

Closed-loop magnetic detumbling performance was evaluated via a one day simulation across multiple simulation cases by varying:

- Sun-synchronous orbit altitude,
- solar-panel deployment angle

Performance metrics and convergence criteria follow the verification framework defined previously.

The detumbling campaign evaluates:

- angular-rate convergence,
- settling-time behaviour,
- velocity-alignment,
- persistent post-acquisition compliance,

Verification Matrix

The detumbling verification campaign used the automated scorecard framework presented previously to evaluate persistent closed-loop convergence behaviour using exported Basilisk telemetry.

Angular-rate acquisition was declared only after the spacecraft angular-rate magnitude remained continuously below the specified requirement threshold for a sustained hold duration of:

$$t_{hold} = 5400s \tag{5.117}$$

This persistent-threshold requirement prevents transient threshold crossings from being incorrectly classified as successful convergence.

The resulting detumbling verification matrix is summarized in Table 5.31.

Table 5.31: Detumbling mode-only verification results

Alt. [km]	Panel [°]	K_{BDOT}	1σ $\ \boldsymbol{\omega}\ $ [deg/s]	Mean $\ \boldsymbol{\omega}\ $ [deg/s]	Settling time [days]	Vel. 1σ [°]	Vel. mean [°]
300	90	0.04	0.1367	0.1020	0.44	2.06	1.38
300	105	0.04	0.1399	0.1042	0.47	1.62	1.10
300	120	0.04	0.1429	0.1058	0.54	1.50	1.00
300	135	0.04	0.1528	0.1073	0.57	1.51	1.07
300	150	0.04	0.1638	0.1083	0.61	1.95	1.33
300	165	0.04	0.1866	0.1150	0.66	3.36	2.28
300	180	0.04	0.2700	0.1566	0.66	FAIL	FAIL
550	90	0.04	0.2073	0.1645	0.49	FAIL	FAIL
550	105	0.04	0.2256	0.1744	0.53	FAIL	FAIL
550	120	0.04	0.2171	0.1670	0.59	FAIL	FAIL
550	135	0.04	0.2216	0.1689	0.63	FAIL	FAIL
550	150	0.04	0.2231	0.1693	0.67	FAIL	FAIL
550	165	0.04	0.2370	0.1756	0.70	FAIL	FAIL
550	180	0.04	0.2511	0.1770	0.70	FAIL	FAIL
600	90	0.04	0.1983	0.1568	0.50	FAIL	FAIL
600	105	0.04	0.1942	0.1588	0.54	FAIL	FAIL
600	120	0.04	0.2034	0.1559	0.61	FAIL	FAIL
600	135	0.04	0.2032	0.1631	0.65	FAIL	FAIL
600	150	0.04	0.1887	0.1506	0.69	FAIL	FAIL
600	165	0.04	0.2160	0.1649	0.71	FAIL	FAIL
600	180	0.04	0.2100	0.1657	0.71	FAIL	FAIL

The reported quantities correspond to:

- settling time required to achieve sustained angular-rate convergence,
- percentage of post-acquisition simulation time satisfying the velocity-alignment requirement,
- mean post-lock angular-rate magnitude,
- and the conservative 1σ residual angular-rate envelope.

All performance values were extracted automatically from exported closed-loop telemetry using the verification scorecard framework. The verification reports additionally included, escape-event statistics, compliance percentages, and post-lock statistical margins for each evaluated simulation case.

Representative Convergence Behaviour

Representative detumbling convergence behaviour is shown for the three tested altitudes (300, 550, 600 kilometers) in Figure 5.11, Figure 5.12 and Figure 5.13 respectively.

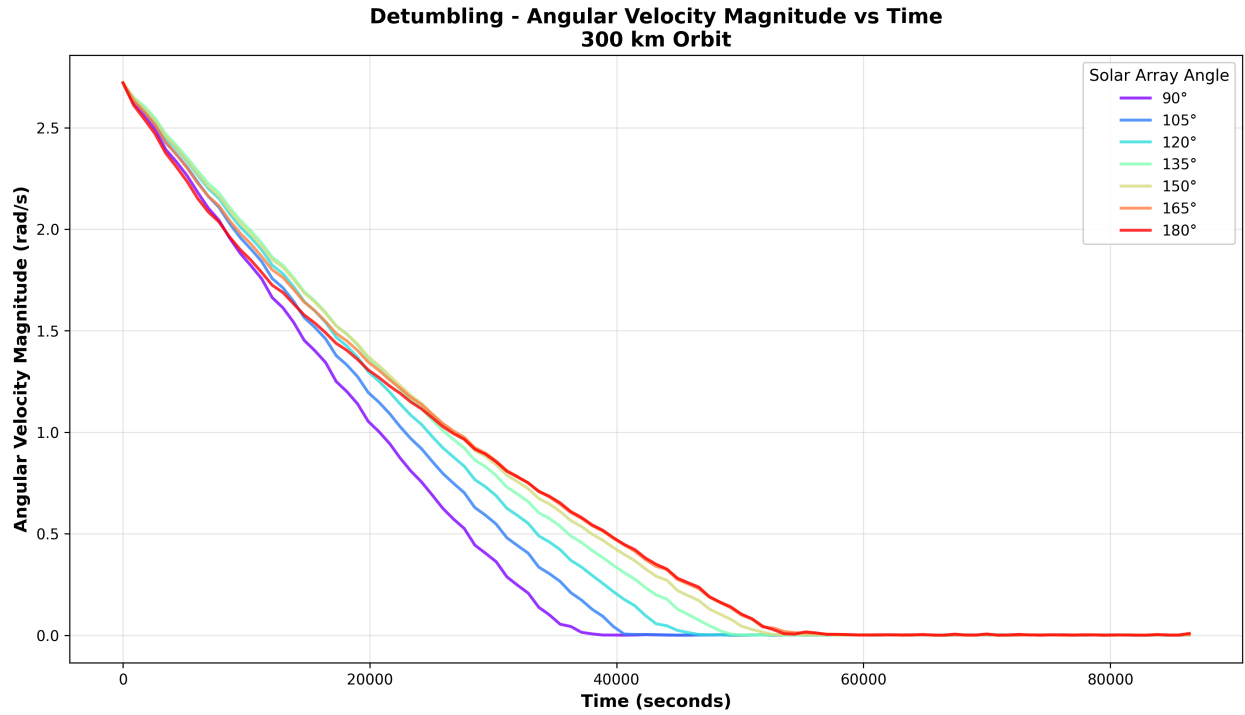


Figure 5.11: Angular-rate convergence behavior during magnetic detumbling for an altitude of 300 kilometers

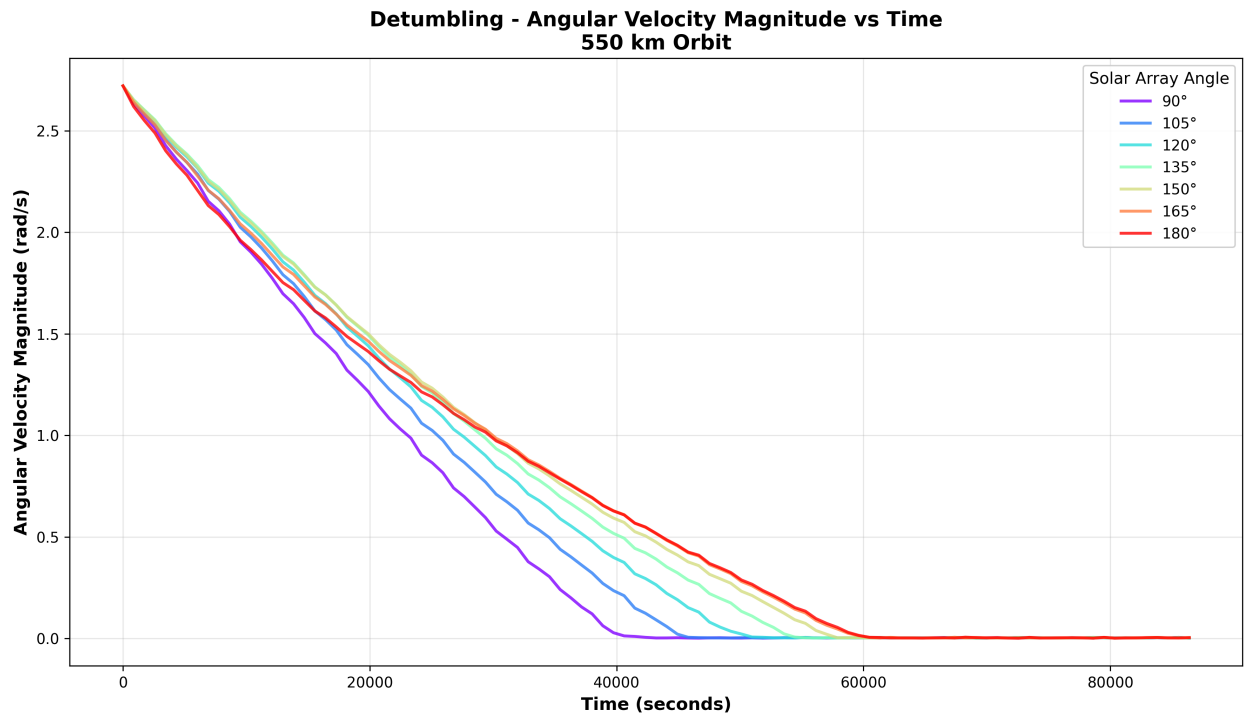


Figure 5.12: Angular-rate convergence behaviour during magnetic detumbling for an altitude of 550 kilometers

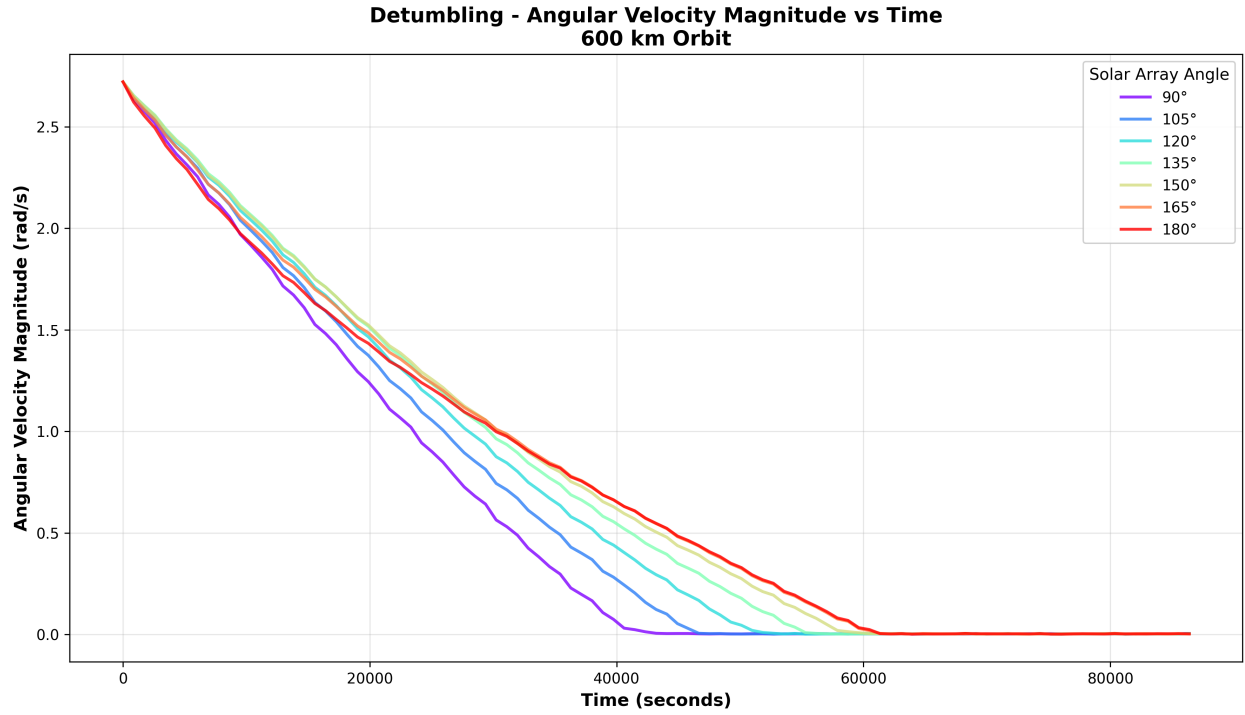


Figure 5.13: Angular-rate convergence behaviour during magnetic detumbling for an altitude of 600 kilometers

Across the evaluated campaign, the controller produced stable angular-rate reduction without sustained divergence or loss of controllability. T

Post-lock mean angular-rate magnitudes remained approximately within:

$$0.10 \leq \|\omega\|_{mean} \leq 0.20 \text{ deg/s} \quad (5.118)$$

with conservative 1σ envelopes remaining below the detumbling requirement threshold for all converged cases.

Altitude Sensitivity

Increasing orbital altitude reduced available magnetic control authority and increased detumbling settling time. At equivalent deployment configurations, the 550 km and 600 km cases consistently converged more slowly than the corresponding 300 km cases.

For the 90° deployment configuration, settling time increased from:

$$38016 \text{ s} \quad (5.119)$$

at 300 km altitude to:

$$42336 \text{ s} \quad (5.120)$$

at 550 km altitude.

The corresponding mean post-lock angular-rate magnitude increased from:

$$0.1035 \text{ deg/s} \quad (5.121)$$

to:

$$0.1623 \text{ deg/s} \quad (5.122)$$

Velocity-pointing performance also degraded with altitude. All 300 km configurations achieved measurable velocity-alignment compliance, while all evaluated 550 km and 600 km configurations failed the corresponding velocity-pointing verification requirement despite maintaining stable detumbling convergence.

Reduced atmospheric disturbance loading partially compensated for the weaker geomagnetic control environment, and all evaluated altitude cases maintained bounded closed-loop behaviour throughout the tested operating envelope.

Solar-Panel Deployment Sensitivity

Increasing solar-panel deployment angle increased inertia asymmetry and aerodynamic disturbance sensitivity, producing slower convergence and degraded velocity-alignment performance.

At 300 km altitude, settling time increased from:

$$38016 \text{ s} \quad (5.123)$$

at 90° deployment to:

$$57024 \text{ s} \quad (5.124)$$

at 165° deployment, while velocity-pointing compliance decreased from:

$$100\% \quad (5.125)$$

to:

$$93.94\% \quad (5.126)$$

The worst-performing configuration occurred for the 180° deployment case, which produced the largest residual angular velocity and the lowest velocity-pointing compliance of:

$$61.29\% \quad (5.127)$$

Despite the degraded performance, all deployment configurations maintained stable detumbling convergence without uncontrolled rotational growth.

Closed-Loop Detumbling Assessment

Across the evaluated simulation campaign, the magnetic detumbling architecture achieved stable angular-rate convergence for all tested cases. Lower deployment-angle configurations produced the strongest overall performance, including faster convergence, lower residual angular velocity, and improved velocity-pointing compliance.

Performance degraded with increasing deployment angle and orbital altitude due to reduced magnetic control authority and increased disturbance sensitivity. The 180° deployment configuration produced the weakest velocity-pointing performance, although detumbling convergence remained stable.

The evaluated results indicate that the implemented B-dot architecture provides sufficient magnetic-control authority to dissipate initial spacecraft rotational energy and transition the spacecraft into coarse nadir-pointing operating conditions throughout the tested simulation envelope.

5.4.4 Nadir-Pointing Performance Results

Closed-loop nadir-pointing performance was evaluated across a multi-case simulation campaign using the automated verification framework introduced previously. The campaign included half day simulations with varied:

- Sun-synchronous orbit altitude,
- solar-panel deployment angle,
- nadir-pointing gain,
- and B-dot damping gain.

The evaluated cases included:

- a baseline gain sweep using ($K_{BDOT} = 0.004$, $K_P = 0.00192$),
- and an aggressive-gain sweep using ($K_{BDOT} = K_P = 0.015$).

Performance was evaluated using:

- nadir acquisition,
- velocity-direction alignment,
- persistent post-lock compliance,
- steady-state rotational stability,
- escape-event frequency,
- and long-duration closed-loop maintenance.

The nadir-pointing verification campaign used the automated scorecard framework to evaluate persistent closed-loop pointing behaviour using exported Basilisk telemetry.

Nadir acquisition was declared only after the nadir-pointing error remained continuously below the specified threshold for a sustained hold duration of

$$t_{hold} = 5400 \text{ s} \tag{5.128}$$

This persistent-threshold requirement prevents transient geometric alignment events from being incorrectly classified as successful pointing acquisition.

The verification framework subsequently evaluated:

- nadir acquisition time,
- velocity-alignment acquisition time,
- post-lock compliance percentage,
- residual steady-state angular velocity,
- statistical steady-state stability margins,
- and post-lock escape-event frequency.

Representative verification results are summarized in Table 5.32.

Table 5.32: Representative nadir-pointing verification results

Alt. [km]	Panel [°]	B_G [$\times 10^{-3}$]	$K_{p,N}$ [$\times 10^{-3}$]	Nadir 1σ [°]	Nadir mean [°]	Nadir sett. [days]	Vel. 1σ [°]	Vel. mean [°]
300	90	4	1.92	4.64	2.76	0.13	13.47	8.30
300	105	4	1.92	4.08	2.58	0.06	8.44	4.79
300	120	4	1.92	4.48	2.84	0.22	8.20	4.88
300	135	4	1.92	8.79	4.40	0.12	10.55	6.97
300	150	4	1.92	14.77	6.71	0.04	12.93	8.30
300	165	4	1.92	38.08	16.13	0.22	FAIL	FAIL
300	180	4	1.92	42.00	17.94	0.25	FAIL	FAIL
300	90	15	15	FAIL	FAIL	FAIL	FAIL	FAIL
300	105	15	15	14.63	6.39	0.73	17.13	8.80
300	120	15	15	7.46	3.62	0.16	14.72	8.78
300	135	15	15	9.78	4.45	0.26	6.98	4.38
300	150	15	15	48.77	24.03	0.63	FAIL	FAIL
300	165	15	15	30.09	12.14	0.46	FAIL	FAIL
300	180	15	15	11.56	4.61	0.19	FAIL	FAIL
550	90	4	1.92	3.84	2.69	0.27	FAIL	FAIL
550	105	4	1.92	4.40	2.96	0.11	FAIL	FAIL
550	120	4	1.92	4.72	2.97	0.20	FAIL	FAIL
550	135	4	1.92	5.44	3.14	0.26	FAIL	FAIL
550	150	4	1.92	54.64	19.69	0.11	FAIL	FAIL
550	165	4	1.92	62.99	26.29	0.05	FAIL	FAIL
550	180	4	1.92	49.49	19.63	0.50	FAIL	FAIL
600	90	4	1.92	3.23	2.52	0.21	FAIL	FAIL
600	105	4	1.92	3.70	2.66	0.41	FAIL	FAIL
600	120	4	1.92	4.58	2.88	0.07	FAIL	FAIL
600	135	4	1.92	4.01	2.84	0.20	FAIL	FAIL
600	150	4	1.92	7.01	3.87	0.44	FAIL	FAIL
600	165	4	1.92	78.30	34.09	0.35	FAIL	FAIL
600	180	4	1.92	59.92	29.79	0.57	FAIL	FAIL

The reported quantities correspond to:

- the percentage of post-acquisition simulation time satisfying the nadir-pointing requirement,

- the percentage of post-acquisition simulation time satisfying the velocity-alignment requirement,
- the elapsed simulation time required to satisfy the persistent hold criterion,
- the post-acquisition escape-event frequency,
- and the residual steady-state angular-rate statistics.

Pointing Acquisition and Recovery Behaviour

Representative pointing behaviour showed that the controller can achieve nadir acquisition without persistent divergence throughout the baseline campaign. The strongest steering activity occurred during the initial recovery transient, when the spacecraft contained the largest geometric error and residual rotational energy.

As the spacecraft approached the nadir-reference attitude, commanded dipole activity reduced progressively and the post-lock behaviour became smoother. Lower-altitude baseline cases exhibited the cleanest steady-state response and the smallest statistical pointing envelopes.

Large-angle initial attitude perturbations exercised the recovery-mode steering architecture introduced previously. The evaluated recovery behaviour demonstrated:

- successful inversion recovery,
- bounded transient response,
- stable large-angle convergence,
- deterministic post-recovery acquisition,
- and sustained bounded rotational behaviour.

Increasing initial rotational energy increased:

- transient dipole-command activity,
- recovery-mode engagement,
- convergence duration,
- and transient pointing excursions.

Despite the stronger transient response, the baseline gain sweep maintained stable closed-loop convergence throughout the evaluated operating envelope. The aggressive-gain sweep amplified transient oscillation and reduced post-lock stability margins in the more demanding configurations.

Steady-State Sensitivity Analysis

Increasing orbital altitude reduced available magnetic steering authority and progressively weakened closed-loop pointing performance. The 300 km baseline cases produced the strongest overall behaviour, with:

- rapid nadir acquisition,
- low escape-event frequency,
- smaller residual angular-rate envelopes,
- and the strongest velocity-alignment performance.

At 550 km and 600 km, velocity-alignment compliance degraded substantially and several cases failed to establish persistent velocity hold windows despite maintaining acceptable nadir-pointing behaviour. The results therefore indicate that increasing altitude primarily reduces the ability to maintain persistent velocity-direction alignment rather than coarse Earth-pointing behaviour itself.

Increasing solar-panel deployment angle increased spacecraft inertia asymmetry and aerodynamic disturbance sensitivity. Larger deployment angles increased:

- residual oscillation amplitude,
- rotational coupling behaviour,

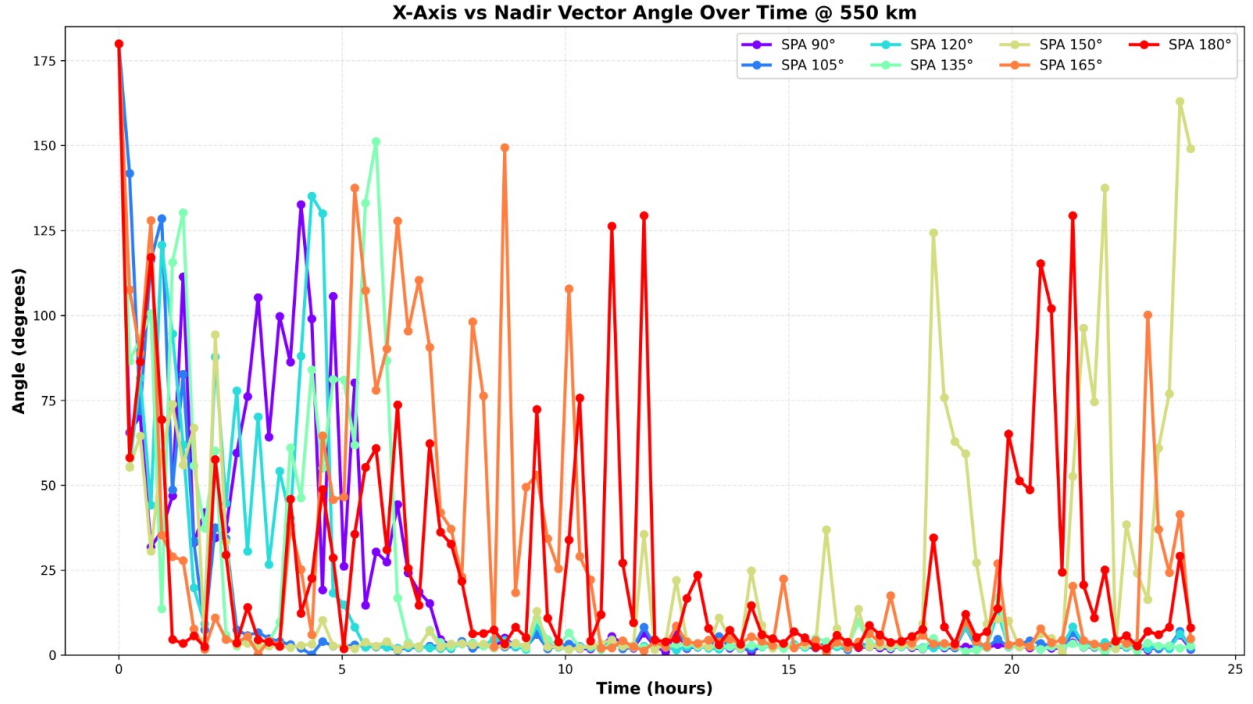


Figure 5.14: Angle between nadir vector and Body X axis wrt time for different solar panel deployment angles.

- escape-event activity,
- and velocity-alignment degradation.

The effect became substantially stronger at higher altitude, where reduced magnetic steering authority combined with the larger rotational asymmetry introduced by the deployed panels. The 150° and larger deployment configurations consistently produced the weakest velocity-alignment behaviour and the widest steady-state rotational envelopes.

The gain sweep demonstrated a strong trade-off between transient aggressiveness and long-duration robustness. The baseline gain set produced:

- smoother steady-state behaviour,
- lower residual angular velocity,
- improved escape suppression,
- and more reliable persistent acquisition.

Increasing both K_P and K_{BDOT} increased:

- transient overshoot,
- residual oscillation,
- actuator activity,
- escape-event frequency,
- and steady-state statistical envelopes.

The aggressive gain sweep demonstrated that larger gains do not eliminate the controllability limitations of magnetic steering. Instead, they amplified oscillatory behaviour and reduced overall closed-loop robustness in the most demanding configurations.

Closed-Loop Pointing Assessment

Across the evaluated simulation campaign, the nadir-pointing architecture demonstrated robust coarse Earth-pointing capability throughout much of the tested operating envelope.

The controller consistently achieved:

- stable nadir acquisition,
- bounded rotational behaviour,
- deterministic recovery behaviour,
- disturbance-tolerant operation,
- and sustained long-duration pointing maintenance.

Velocity-direction alignment remained the dominant limiting mode throughout the campaign. Several higher-altitude and larger-deployment configurations failed to achieve persistent velocity hold windows despite maintaining bounded nadir-pointing behaviour. This demonstrates the fundamental controllability limitations on yaw imposed by depending on atmospheric drag for stabilization. It is recommended to include an on-board orbit propagator and use a QuEst-based 3-axis pointing logic instead of just pitch and roll.

The evaluated campaign, therefore, indicates that the implemented magnetic steering architecture provides sufficient authority for stable coarse nadir-pointing operation across the evaluated low-Earth-orbit operating envelope, while persistent velocity-direction alignment cannot be achieved via atmospheric drag alone.

5.4.5 System-Level Robustness Assessment

System-Level Robustness Assessment

Across the evaluated simulation campaign, the integrated ADCS architecture demonstrated:

- stable disturbance rejection,
- bounded actuator utilization,
- stable coarse nadir-pointing maintenance,
- and sustained post-acquisition controllability under environmental loading.

The evaluated results further indicate robustness against:

- deployment-dependent inertia variation,
- altitude-dependent magnetic-authority variation,
- environmental disturbance torques,
- transient actuator saturation,
- and finite actuator resolution.

Within the evaluated low-Earth-orbit operating envelope, the selected actuator configuration maintained sufficient magnetic-control authority to preserve closed-loop stability and bounded rotational behaviour throughout nominal operation.

6. Manufacturing

This chapter outlines how the whole ADCS is to be constructed. Starting with the assembly of the sensors and the microcontroller and then the procedure of constructing the magnetorquer assembly.

6.1 Integration planning

The ADCS system was constrained to a space of $15\text{ mm} \times 15\text{ mm} \times 45\text{ mm}$ as to fit in the allocated space in the spacecraft. It was not required to violate these dimensions so the EHSs have to be flush with the rest of the spacecraft bus. Thus, they intrude inwards. The X- and Y-axis magnetorquers are the only components that are not in this volume as they are integrated in the walls of the satellite. The Z-axis magnetorquer, being of the air-coil variant, does have to adhere to the volume constraint. Due to the intrusion of the EHSs the magnetorquer is squeezed into a smaller space. An example of what the bracket and PCB walls will likely look like is given in Figure 6.1.

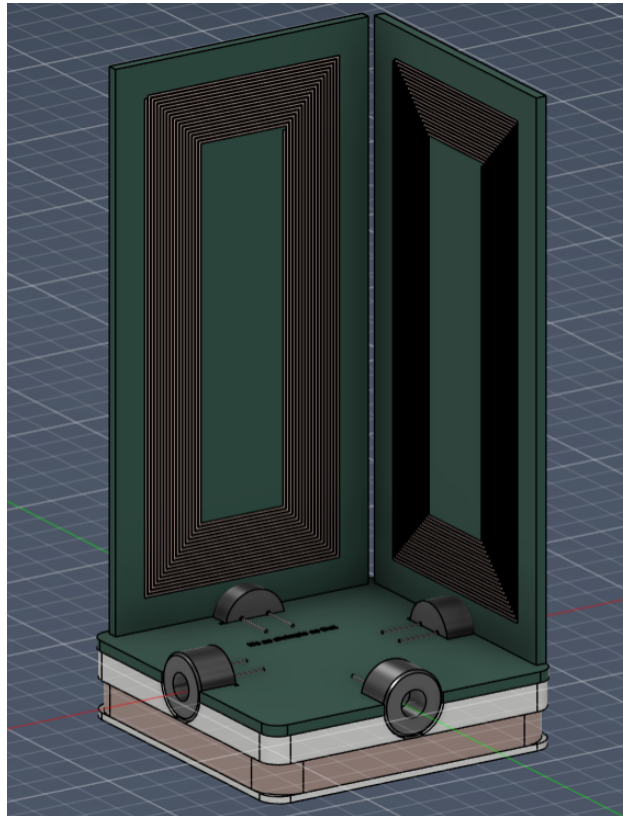


Figure 6.1: Image of the full CAD, including the PCB side panels

6.2 Magnetorquer manufacturing

The magnetorquer manufacturing can be split up into two parts; the X- and Y-axis and the Z-axis as they are different types of magnetorquers. The air-coil magnetorquer is first elaborated on as it requires the most work. The PCB magnetorquers can not be made in-house and thus are outsourced.

6.2.1 Air-coil magnetorquer manufacturing

The manufacturing process of the air-coil magnetorquer is based on the one used by M.F van den Bos in his MSc thesis[9]. The bracket housing will also be 3D printed as described in section 6.1. The enameled copper that is to be used (accessible here) has a cross-section of 0.15 mm which will prove to be a very difficult and delicate process. Thus, the designed winding apparatus will also be used to ensure proper winding that is tight enough without breaking[9]. A 3D model of this component is shown in Figure 6.2. As the manufacturing is to mimic M.F. van den Bos' process, it is best described in his thesis:

"The wire is fed from the spool it arrived on from the wire manufacturer. The stepper motor counts the number of revolutions and can be controlled with a foot pedal. This leaves the operators hands free to guide the wire using a pulley. Using pair of tweezers winding defects were reduced. Tension in the wire was created by friction resistance on the feed spool and the action of the stepper motor. The wire on the feed spool was very loosely wound, hence creating uneven tension in the winding process. Tension was increased or decreased by adjusting a screw to create more or less friction."

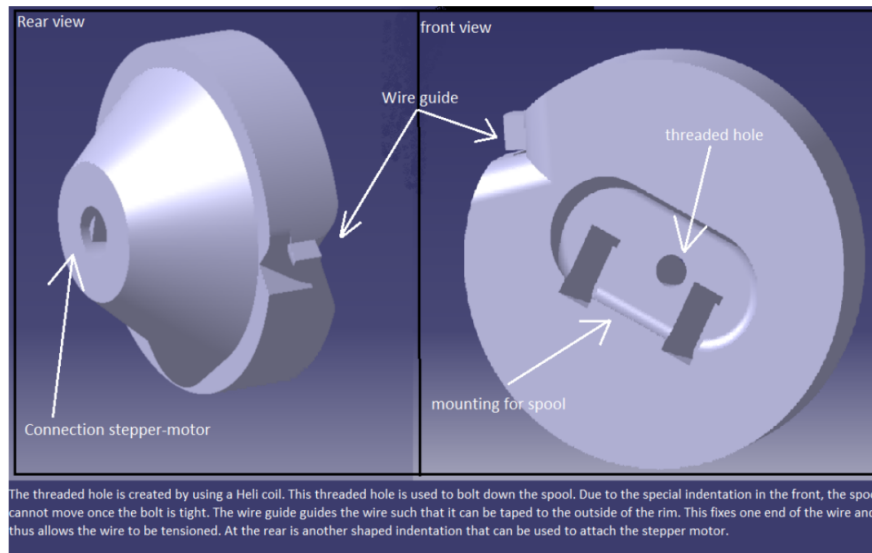


Figure 6.2: This piece is to be connected to the spool and stepper motor and hold one end of the wire

There were some future recommendations mentioned. These additions are the following and are to be integrated in the next phase of the project; the Space Engineering Practical.

- The rewinding of the factory spool onto another spool before winding around the bracket. This is done as the tension around the factory spool is too loose.
- The inclusion of a uniform tensioning mechanism which will result in a more compact packing of wire.
- As a result of the two additions above a more powerful stepper motor is to be included as to overcome this extra tension force.
- The wire guidance is also to be improved, either through computer guidance or with a manually operated mechanism.

6.2.2 PCB printed magnetorquers

As mentioned, the PCB printed magnetorquers are not able to be produced at the TU Delft due to the absence of the complex machinery needed. After exploring a multitude of manufactures, Eurocircuits was chosen (be.eurocircuits.com). It was chosen due to its heritage and being a well known PCB manufacturing that allows for a wide range of configuration options. The specific configuration can be found with this link: <https://be.eurocircuits.com//shop/assembly/configurator.aspx?sessionId=c7d2ce15-eb3d-4a00-a556-07b70120b8db&quantity=1&deliveryTerm=3&layers=16&dimY=80&dimX=40>. The specific configuration that was chosen is as follows.

The dimensions of the PCB were set by the Delfi-Twin team to be $80\text{ mm} \times 40\text{ mm} \times 1.5\text{ mm}$. To achieve the required magnetic torque for de-tumbling eight layers of copper tracing in series are required. They need to be in series to increase the resistance so the current can stay within reasonable limits. To add to this, the inner and outer layer of copper foil were set at 35 mm . Furthermore, the trace-width and trace-spacing are 0.4 mm and 0.1 mm respectively. A very costly sacrifice had to be made to keep the wiring in series, that is the option to go for hidden via's. This almost quintuples the cost of the PCBs.

These options should result in a PCB printed magnetorquer design that fully complies with the requirements set.

7. Requirement compliance check

This chapter is meant to provide a quick reference for checking the compliance check of the system and component requirements. Note that not all requirements have been fulfilled by the design presented in this report. Notably, ENV-4, ENV-7, REC-4, ACT-2, and MCU-01 have not been met. For each of these, a justification is given below for why they cannot be met at this time, as well as an explanation for why ACT-2 failed to be met.

Table 7.1: Compliance check table

Req ID	Fulfilled?	Compliance check method
ENV-2	Yes	The only exposed parts of the ADCS are the IR sensor lenses and the outside of the side panel PCBs which cannot be further oxidized. See section 4.5 for more details.
ENV-3	Yes	The datasheets of the MLX90640[5], RM3100[7], DRV8838[11], and STM32L476RG[12] state their component's thermal range, which are all within the requirement.
ENV-4	No	The final ADCS assembly does not exist yet. This requirement can be fulfilled only once the ADCS PCB has been designed and the full system is integrated in CAD.
ENV-7	No	Since the datasheets of the components do not include radiation specifications, this requirement cannot be proven as met without a doubt. However, all hardware used is flight proven, which is a satisfactory compromise.
PWR-6	Yes	See section 3.5
PWR-1	Yes	All components work on 3.3V, through a voltage regulator.
PWR-7	Yes	See section 3.5
PWR-3	Yes	All components are COTS items meant to be arbitrarily turned on or off. As the MTQs are simple circuits with no logic, they can be easily turned off without damage.
PWR-4	Yes	This requirement is implemented through design. See subsection 3.3.1
REC-1	Yes	This requirement is implemented through design. See section 4.4
REC-3	Yes	This requirement is implemented through design. See section 3.3
REC-6	Yes	This requirement is implemented through design. See section 4.4
SND-4	Yes	This requirement is implemented through design. See section 4.4
ACT-1	Yes	As shown in chapter 5, the ADCS is able to halt the rotation of the satellite in all simulations.
ACT-1.1	Yes	As shown in chapter 5, the ADCS is able to perform detumbling from an initial angular velocity of 180 deg/s.
ACT-1.2	Yes	As shown in chapter 5, all scenarios show detumbling finished within 3 days.
ACT-2	No	Although it was possible to perform prograde pointing within 10 degrees at lower altitudes, the lower atmospheric density at 500 km proves the differential torque-based prograde pointing in yaw unstable. The requirement is not met for higher altitudes, although it becomes fulfilled once the satellite decays to a lower altitude. A potential solution could be including an orbital propagator onboard to execute yaw pointing. See chapter 5 for more details

ACT-2.1	Yes	The MTQs are able to produce $9 \cdot 10^{-7}$ Nm. With an aerodynamic disturbance torque of $3.2 \cdot 10^{-7}$ Nm, they fulfill the requirement of at least $2 \cdot 3.2 \cdot 10^{-7} = 6.4 \cdot 10^{-7}$ Nm. See section 4.2 for more details.
ACT-3	No	Although nadir pointing is possible conceptually with the designed hardware, as shown in chapter 5, the current software is unable to perform Nadir pointing in certain edge cases due to non-optimal tuning of the PD controlled gains.
ACT-3.1	Yes	The MTQs are able to produce $9 \cdot 10^{-7}$ Nm. With an aerodynamic disturbance torque of $3.2 \cdot 10^{-7}$ Nm, they fulfill the requirement of at least $2 \cdot 3.2 \cdot 10^{-7} = 6.4 \cdot 10^{-7}$ Nm. See section 4.2 for more details.
ACT-3.2	Yes	MTQ-06 was formulated as a component-level translation of ACT-3.2 Since MTQ-06 is met (see subsection 4.2.1), ACT-3.2 becomes met automatically.
ACT-4	Yes	Presence of the Sun accounted for in the design of the horizon detection algorithm. See ?? for more details
ACT-5	Yes	The magnetic field can be calculated using $B = \mu_0/2\pi \cdot 2m/r^3$ [21]. Assuming the magnetometer is placed 3 cm away from the magnetorquers, and with a MTQ magnetic dipole moment of 0.0177 A m^2 (see section 4.2), this gives a maximum MTQ-induced magnetic field strength of $262 \mu\text{T}$, which is 4.8 times higher than Earth's $5.4 \mu\text{T}$. The fall time of the MTQ is calculated as the time when the field strength is 10% of its peak value, giving a value of $26.2 \mu\text{T}$ or 50% of Earth's. This is not enough to fulfill the requirement, but due to how the main loop is designed, there is always at least 10 ms between the end of the fall time of the MTQ and the magnetometer measurement (subsection 3.3.2). Since the fall time is on the order of 0.5 ms, this is deemed enough to satisfy the requirement.
CON-1	Yes	See chapter 6
CON-3	Yes	See section 3.5
CON-4	Yes	The system was designed to be self-contained and does not depend on its z-wise placement in the satellite. See chapter 6.
ESH-02	Yes	There is always a sensor in view of the Earth's horizon that accomplishes this accuracy as shown in ??.
ESH-03	Yes	The time it takes for the frame to travel via I2C was determined to be 13.344 ms which increases to 19 ms with margins as shown in section 4.1.
MTQ-01	Yes	Requirement fulfilled by design. See subsection 3.3.2.
MTQ-02	Yes	See section 3.5
MTQ-03	Yes	STM32's HAL is able to switch the MTQ on and off within 4 ms using regular GPIO pins.
MTQ-04	Yes	$\max(\tau_{R+F,z-aircoil}, \tau_{R+F,xy-PCB}) = 4 \cdot 10^{-4} \text{ s} \leq 4 \cdot 10^{-4} \text{ s}$
MTQ-05	Yes	See subsection 4.2.1
MTQ-06	Yes	See subsection 4.2.1
MTQ-07	Yes	See subsection 4.2.1
MTQ-08	Yes	See Figure 4.2.1
MTQ-09	Yes	See Equation 4.4
MGM-01	Yes	See section 4.3
MGM-02	Yes	See section 4.3
MGM-03	Yes	See section 4.3
MCU-01	No	Requirement cannot be met because PCB is not yet designed. As a compromise, placeholder of required thickness was used in chapter 6

8. Conclusion

This report aimed to design a stand-alone ADCS for the Delfi-Twin mission. The final design consists of four MLX90640BAA infrared sensors mounted on each of the lateral faces of the satellite that functions via an Earth Horizon Sensing algorithm. They switch based on which sensor is sensing the horizon. For the first phase of the mission i.e. detumbling will be led by the magnetometer, RM3100. The readings from the magnetometer are fed into the STM32L476RG microcontroller, the heart of the ADCS. It acts as the master of the system and can decide on a variety of modes. It directs magnetorquers in three axis, two PCB printed and one aircoil, via a B-dot algorithm in the de-tumbling stage. When it notices a stable slow rotation it switches over to the EHSs. The main loop has a detumbling and pointing mode of 1.111 Hz and 1 Hz respectively.

It can be concluded that the goal to achieve a stand-alone ADCS system with the given user requirements is achieved. There are a few caveats, the volume constraints are violated due to the introduction of the PCB printed magnetorquers. However, this was agreed upon by the Delfi-Twin team but it required a very expensive manufacturing method, hidden via's.

The design in this project will be used in the next phase of the design; the Space Engineering Practical. The team will actually develop and test the set-up to validate and verify if the system can function nominally.

9. Inventory & Activities

This chapter showcases all the hardware and software that is to be used in the next step of this project; the Space Engineering Practical. It highlights what components are still missing and which tests are to be performed. The latter can always be expanded upon or changed.

To make it easier for the reader, the pages have been rotated in landscape format. They can be found below.

9.1 Hardware & Software Inventory

Table 9.1: ADCS Hardware Inventory Status

Item	Description	Status	Interfaces	Subcomponents	Responsibility
Magnetorquer	3x Actuators	WIP (1x Design)	Power supply, Physical Interface	Ferromagnetic core, coil	Fatih
DRV8838	Magnetorquer Driver Electronics	Available	Power Supply, Physical Interface	none	Fatih
RM3100	3x Magnetometers	Available	Cables, Power supply, Physical interface	none	Pari
NUCLEO-L476RG (STM32)	Evaluation Board, Micro-controller	Available	Cables, USB, Power supply, Physical interface	none	Maurizio
MLX90640BAA	4x Horizon Sensor	2x Available, 2x WIP	Cables, Power supply, Physical interface	none	Tjalling
MLX90640-41_evb_2	Horizon Sensor Driver	Available	Cables, Power supply, Physical interface	none	Tjalling

Table 9.2: ADCS Software Inventory Status

Item	Description	Status	Import/Export format	Dependencies	Installed-@
Ansys	Simulation Environment	Available	.step	none	Fatih
Fusion	CAD, PCB design	Available	.step	none	Fatih
Granta Edupack	Material Catalogue	Available	.ces, .cet	none	Fatih
KiCAD	PCB and electrical design	Available	.pro	none	Maurizio
VS Code	IDE	Available	.txt, .py, .c, .cpp	none	Tjalling, Maurizio, Pari
Arduino	IDE	Available	.txt, .ino, .pde	none	Tjalling, Maurizio, Pari
Basilisk	Coding + Simulation Environment (Python, C++)	Available	.pip	none	Abhi
Github	Code Repository	Available	All	None	Abhi, Maurizio, Pari, Tjalling

Table 9.3: ADCS Dedicated Testing Setup Inventory Status

Item	Description	Status	Interfaces	Subcomponents	Responsibility
Helmholtz Cage	Inducing magnetic fields	WIP	N/A	All components	Abhi, Fatih, Pari
Earth Horizon Sensor Test 2025	Physical Test Setup	WIP	VDD, GND, SDA, SCL	Microcontroller, Servo Motor, Stepper Motor, Temperature Sensor, Heating elements, USB Cables, Resistors, Capacitors, Power Source, Earth Sensors, Motor Driver	Tjalling
STM32 Testing Environment	Breadboard model	Available	STM32CubeMX	STM32 + breadboard, Cables, Oscilloscope, Power supply, Multimeter	Maurizio
Magnetometer Algorithm Testing	Physical Test Setup	Available	TBR	1x STM32 Nucleo-L476RG board, 1x RM3100 breakout/module, 1x USB data cable for Nucleo ST-LINK port (micro-USB or USB-C adapter as needed), 7x Jumper wires minimum (SPI + CS + DRDY + power + ground)	Pari

Bibliography

- [1] J. Vanhamel, S. Speretta, and Uludag M.S. “Concept of Sporadic E Monitoring Using Space-Based Low Power Multiple Beacon Systems”. In: Atmosphere (2024).
- [2] M. Centrella et al. “Advancing In-Space Precise Tracking: A Formation-Flying Picosatellites Mission”. In: 75th International Astronautical Conference. 2024.
- [3] AlbaOrbital, TU Delft, and AUSS. The Pocket Standard. 2018.
- [4] J. Bouwmeester et al. “Utility and constraints of PocketQubes”. In: CEAS Space Journal (2020).
- [5] Melexis. MLX90640 32x24 IR array. 2019.
- [6] J. Sampedro et al. Earth Sensor for PocketQube. Assignment brief. Unpublished document.
- [7] PNI Sensor. User Manual RM3100 & RM2100 Magneto-Inductive Magnetometer. 2022.
- [8] M.B. Moldwin et al. “Single-event effect testing of the PNI RM3100 magnetometer for space applications”. In: Geoscientific Instrumentation Methods and Data Systems (2022).
- [9] M.F. van den Bos. “Design and Testing of Magnetic Torquers for Pico Satellite Attitude Control”. MA thesis. Delft University of Technology, 2019.
- [10] S. Dixit, S. Van Hulle, and V. Veldhuizen. Detumbling the PQ. Tech. rep. Delft University of Technology, 2024.
- [11] Texas Instruments. DRV883x Low-Voltage H-Bridge Driver datasheet (Rev. F). 2021. URL: <https://www.ti.com/lit/ds/symlink/drv8838.pdf>. (accessed: 24.03.2026).
- [12] STMicroelectronics. Datasheet STM32L476xx. 2024. URL: <https://www.st.com/resource/en/datasheet/stm32l476je.pdf>. (accessed: 24.03.2026).
- [13] A. Finance et al. “In-Orbit Attitude Determination of the UVSQ-SAT CubeSat Using TRIAD and MEKF Methods”. In: Sensors (2021).
- [14] S. Kim et al. Unscented KALMAN Filtering for Spacecraft Attitude and Rate Determination Using Magnetometer. 2009.
- [15] Mehmet Ipek. “Satellite Orbit Estimation Using Kalman Filters”. MA thesis. Middle East Technical University, 2017.
- [16] G. Zhang et al. “Real-Time Attitude Estimation for Spinning Projectiles by Magnetometer Based on an Adaptive Extended Kalman Filter”. In: Micromachines (2023).
- [17] Texas Instruments. DRV8838EVM User’s Guide. 2014. URL: <https://www.ti.com/lit/ug/slvua43/slvua43.pdf?ts=1772540314739>. (accessed: 24.03.2026).
- [18] Melexis. MLX90640 32x24 IR array Driver. 2022.
- [19] STMicroelectronics. STM32 Nucleo-64 boards (MB1136) - User manual. 2025. URL: https://www.st.com/resource/en/user_manual/um1724-stm32-nucleo64-boards-mb1136-stmicroelectronics.pdf. (accessed: 24.03.2026).
- [20] Bedong Wang, Hao Wang, and Zhong Jin. “An Efficient Algorithm for Infrared Earth Sensor with a Large Field of View”. In: Sensors (2022).
- [21] Konstantin K. Likharev. 5.4: Magnetic Dipole Moment and Magnetic Dipole Media - Physics LibreTexts. URL: [https://phys.libretexts.org/Bookshelves/Electricity_and_Magnetism/Essential_Graduate_Physics_-_Classical_Electrodynamics_\(Likharev\)/05%3A_Magnetism/5.04%3A_Magnetic_Dipole_Moment_and_Magnetic_Dipole_Media](https://phys.libretexts.org/Bookshelves/Electricity_and_Magnetism/Essential_Graduate_Physics_-_Classical_Electrodynamics_(Likharev)/05%3A_Magnetism/5.04%3A_Magnetic_Dipole_Moment_and_Magnetic_Dipole_Media). (accessed: 25.03.2026).

A. Deleted requirements

- ENV-1: The STM32 microcontroller shall correct corrupted data and re-request corrupted commands. *Note: requirement moved to REC-1 and REC-2*
- ENV-5: The system shall be protected from condensating water and sweat. *Note: req deleted because it is not adequately verifiable.*
- ENV-6: The system shall resist internal magnetic fields from other subsystems. *Note: req is poorly formulated and inaccurate.*
- PWR-2: The system shall be protected from current surges up to TBR (5) amperes. *Note: req deleted because it is the EPS's responsibility to prevent a short-circuit.*
- PWR-5: The system shall spend at least 10,000 seconds before attempting any attitude corrections. *(this is to make sure we're not deploying things or doing something stupid while still nearby other satellites) Note: req deleted because it was unnecessary. PWR-4 now fulfills it by letting the OBC decide when it's time to start moving*
- REC-2: The system shall correct corrupted data. *Note: req deleted after learning that MCUs already do this*
- REC-4: The system shall receive positional data from the OBC. *Note: req deleted after lack of orbit propagator*
- REC-5: The system shall be able to be reprogrammed by the OBC. *Note: req deleted because it is unrealistic given time constraints and resources*
- SND-1: The system shall send a response to each OBC request. *Note: req deleted as it was decided that the ADCS will be the slave and will not send anything unprompted.*
- SND-2: The system shall send at least the following info as system health indicators to the OBC on request:
 - SND-2.1. All measured voltages
 - SND-2.2. Connection status of all ADCS components
 - SND-2.3. Current attitude state estimation
 - SND-2.4. Sensor data
 - SND-2.5. System update rate
 - SND-2.6. Operational Mode (including Safe Mode)

Note: SND-2 was poorly written. Reworded into SND-4 by sending all stored system data instead of specifying which items to send.

- SND-3: The system shall relay sensor data to the OBC in real time. *Note: requirement moved into SND-2.5*
- ACT-2.2: The system shall be able to turn 180 degrees in 12 hours around the x and y axes at worst case scenario.

Rationale: Once finished with detumbling, the system orients itself towards prograde. In the worst case scenario, this is a 180 degree turn. To constrain the performance of the MTQ, it is necessary to

define a maximum time for this operation. A value of 12 hours was chosen as this is the axis with the minimum moment of inertia, and this does not pose a driving constraint on the actuators.

Verification method: Simulation

Note: requirement deleted as it was no longer relevant for the mission profile. Aerodynamic torque will automatically point the satellite in the desired prograde orientation (except the Nadir pointing). It is not necessary to impose a requirement on it.

- CON-2: The system shall consume at most 1 Watt during nominal operation, with a peak consumption lower than 2 Watts.

Rationale: The satellite has about 4 Watts available. The ADCS should not consume a big chunk of it, and the numbers used here are TBC guesses until clearer values are known.

Verification method: Breadboard model