Delfi-n3Xt's Attitude Determination and Control Subsystem

Implementation and verification of the hardware and software

M. Vos

CE-MS-2013-04

Master of Science Thesis

TU Delft Delft University of Technology

Computer Engineering laboratory

Delfi-n3Xt's Attitude Determiniation and Control Subsystem

Implementation and verification of the hardware and software

Master of Science Thesis

For the degree of Master of Science in Embedded Systems at Delft University of Technology

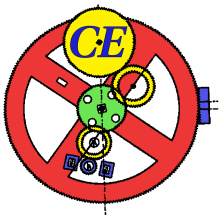
M. Vos

mvos100@gmail.com

May 16, 2013

Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS) · Delft University of Technology

The work in this thesis is done for the Delfi programme



Abstract

Attitude determination and control is the estimation of the current orientation of a satellite and controlling or changing it in order to reach a desired orientation. In the current generation of nano satellites an Attitude Determination and Control Subsytem (ADCS) plays an important roll and sometimes even critical roll. An ADCS is necessary in order to achieve certain mission objectives and to prevent the satellite from tumbling to fast. A failure of this subsystem can have serious consequences and can even lead to a failing mission.
Because of the importance, this subsystem must be reliable and robust in order to reduce the risk of a failing system. To achieve reliability and robustness, a software framework must be designed and implemented that minimizes the risk of software failures. Apart from software failures the risk of hardware failures should also be reduced as much as possible. Therefore all hardware should be thoroughly tested, before it can be marked as a flight candidate. Besides reliability and robustness the software framework should also be user friendly. Because the Delfi-n3Xt's ADCS incorporates two different micro controllers the software framework should be easy to adapt for different micro controllers. For future purposes the software framework should have a modular structure in order to make re-usability and extendibility possible.
In this thesis the design and implementation of such a software framework is described. Furthermore, the process and results of testing the hardware thoroughly are presented.

Laboratory : Computer Engineering
Codenumber : CE-MS-2013-04

Committee Members :

Advisor : Dr. Ir. A.J. van Genderen, CE, EEMCS TU Delft

Advisor : Ir. J. Bouwmeester, SSE, AE TU Delft

Advisor : Dr. D. Choukroun, SSE, AE TU Delft

Chairperson : Dr. S.D. Cotofana, CE, EEMCS TU Delft

Table of Contents

List of Figures

List of Tables

List of Acronyms

ADC	Analogue to Digital Converter
ADCS	Attitude Determination and Control Subsytem
AFSK	Audio Frequency Shift Keying
AL	Application Layer
AWWS	Autonomous Wireless Sun Sensors
BOP	Bottom Panel
BPSK	Binary Phase-Shift Keying
CDHS	Command and Data Handling Subsystem
CGGs	Cold Gas Generators
CSP	Coarse Sun Pointing
COMMS	Communication Subsystem
DAB	Deployment Antenna Board
DBGU	Debug Unit
DCM	Direction Cosine Matrix
DSSB	Delfi Standard System Bus
ECEF	Earth Centered Earth Fixed
ECI	Earth Centered Inertial
EKF	Extended Kalman Filter
EPS	Electrical Power Subsystem

ETC	Elapsed Time Counter
FSP	Fine Sun Pointing
G-EPS	Global Electrical Power Subsystem
GPS	Global Positioning System
GST	Ground Station Tracking
HAL	Hardware Abstraction Layer
HDRM	Hold Down and Release Mechanism
IGRF	International Geomagnetic Reference Field
I/O	Input/Output
ITRX	ISIS Transceiver
KF	Kalman Filter
L-EPS	Local Electrical Power Subsystem
MAB	Modular Antenna Box
MCC	Master Clock Controller
MCK	Master Clock
MCU	Micro Controller Unit
MechS	Mechanical Subsystem
MPPT	Maximum Power Point Tracker
NVM	Non Volatile Memory
OBC	OnBoard Computer
ORF	Orbital Reference Frame
PCB	Printed Circuit Board
PCC	Processor Clock Controller
PCK	Processor Clock
PIO	Parallel Input Output
PRCC	Peripheral Clock Controller
PMC	Power Management Controller
PTRX	Primary Transceiver
QSS	Quadrant Sun Sensor

RTT	Real Time Timer
SBF	Satellites Body Fixed
SDM	Silicon Solar Cell Degradation Measurement
SGP	Simplified General Perturbations
SL	Service Layer
SLCK	Slow Clock
STS	Structural Subsystem
STX	S-band Transmitter
T3uPS	T3 Micro Propulsion System
TC	Timer Counter
TCS	Thermal Control Subsystem
TFSC	Thin Film Solar Cells
TLE's	Two Line Elements
TP	Thruster Pointing
TOP	Top Panel
TWI	Two Wire Interface
UHF	Ultra High Frequency
VHF	Very High Frequency
VVB	Variable Voltage Bus
WDT	Watchodog Timer

Acknowledgements

First of all I would like to thank a couple of staff members from the Delft University of Technology. My special gratitude goes out to Arjan van Genderen for being the supervisor for this Master thesis. I am thankful for the feedback and the time and effort he has spend on reading and reviewing parts of my thesis.
Special gratitude also goes out to Jasper Bouwmeester, the project leader, for giving me the opportunity to work in such a great and educational project like the Delfi-n3Xt. Moreover, I am thankful for his time and effort that he has put into leading the project and making sure everything went according to plan. Although it was a bumpy road at the end, everything ended up well. Also his time and effort he spend on reading my thesis and providing we with feedback is really appreciated.
I am also thankful to Daniel Choukroun who helped me with the complicated theory behind Attitude Determination and Control. The time and effort he has put into helping me and providing feedback on my thesis are really appreciated.
Secondly, I want to thank the entire Delfi-n3Xt team, with in particular the guys whom with a worked together most of the time. I really enjoyed being part of a team that together achieved some great work. Besides that I learned a lot from various team members about the field of Space Systems Engineering. This knowledge will sure become useful in my future career. Furthermore my appreciation also goes out to the people who were not directly part of the Delfi-n3Xt team, but provided very useful and important support to the project.
At last I would of course like to thank my parents for their love, support and patience during my entire study. They have given me the opportunity to have a carefree period of being a student. I also want to thank my friends and all the people who I forgot to mention for their support and the great time.

Michel Vos
Delft, The Netherlands
April 10th, 2013

Chapter 1

Introduction

In the current generation of nano satellites attitude determination and control plays an important roll in order to achieve certain mission objectives. Therefore these satellites are increasingly equipped with an Attitude Determination and Control Subsytem (ADCS). The Delfi-n3Xt is one of those satellites that is equipped with an ADCS.
The ADCS on board the Delfi-n3Xt is designed in [3] and is one of the critical subsystems. A failure of this subsystem can be the end of the Delfi-n3Xt mission. The ADCS presented in [3] still is a theoretical representation of this subsystem. The actual implementation still has to be done. This thesis focusses on turning the theoretical representation into a practical implementation in the form of a software framework that can run on the ADCS hardware. Furthermore this thesis also focusses on the verification of the software framework in combination with the ADCS hardware. Therefore this thesis is entitled:

> *"Delfi-Next's Attitude Determiniation and Control Subsystem: Implementation and verification of the hardware and software."*

1-1 Problem statement

Because the ADCS is one of the critical subsystems on board the Delfi-n3Xt it should function correctly under all circumstances. A complete failure of the ADCS or one of its hardware components can lead to an increase in rotational rate of the satellite. In a worst case scenario the rotational rate of the satellite continues to increase beyond the operational limit, which means the end of the Delfi-n3Xt mission.
In order to prevent the ADCS from failing, a reliable and robust software framework must be designed and implemented. Besides reliability and robustness the software framework should also provide all functionality to make the following five operational modes possible, which were also described in [3].

- **Detumble** Detumble mode is a critical mode of the Delfi-n3Xt which must reduce its tumble rate, that is caused due to ejection from the launch container and external

disturbances acting on the satellite. Using magnetometers and magnetorquers only the Detumble mode should be able to reduce the tumble rate to below 1 degree/s. Once this tumble rate is achieved it can switch from Detumble mode to one of the sun pointing modes.

- **Coarse Sun Pointing** Coarse Sun Pointing (CSP) mode must point the solar panels of the Delfi-n3Xt towards the sun, with a maximum absolute pointing error of 25 degrees, in order to maximize the generated power. In order to achieve this it utilizes the sun sensor measurements for attitude determination and the magnetorquers and reaction wheels for attitude control.
- **Fine Sun Pointing** Fine Sun Pointing (FSP) mode is different from CSP mode in the sense that it uses a more complex algorithm to point Delfi-n3Xt's solar panels to the sun. Besides the sun sensor measurements this algorithm also utilizes the magnetometer measurements, Earth magnetic field reference data and orbit information for attitude determination. Furthermore Fine Sun Pointing mode is the starting pointing for switching to one of the other two advanced modes.
- **Ground Station Tracking** Ground Station Tracking (GST) mode should track the ground station pointing the antenna of the S-band Transmitter (STX), with a maximum absolute pointing error of 10 degrees, to the ground station. Furthermore the solar panels should be pointed as close as possible to the sun direction. To achieve this GST mode uses the same resources and complex algorithms as FSP mode.
- **Thruster Pointing** Thruster Pointing (TP) mode should point the T3 Micro Propulsion System (T3uPS) in the velocity direction with a maximum absolute pointing error of 25 degrees. Furthermore the solar panels should be pointed as close as possible to the sun direction. To achieve this TP mode uses the same resources and complex algorithms as FSP mode.

Apart from the requirements on the software framework the hardware is also a important point of attention. The hardware should be tested extensively in order to make sure the hardware functions correctly and prevent it from failing.

1-2 Previous work

The Delft University of Technology is with the Delfi-n3Xt not the first university that has developed a nano/pico satellite equipped with an ADCS. Several other universities around the world have already successfully developed and launched a nano satellite that is equipped with an ADCS. An example of such a satellite is the Swiss cube [4]. The Swiss cube is the first pico satellite developed entirely by several universities in Switzerland. It is equipped with an ADCS consisting of 3-axis magnetometer, a 3-axis gyroscope and six sun sensors to provide attitude information, a magnetorquer system to control the attitude and a micro controller on which determination and control algorithms are implemented. Furthermore the Swiss cube has an experimental inertia wheel on board. In the beginning of the Swiss Cube mission its ADCS did not function correctly. Due to the high initial rotational rate of the satellite, after it was ejected from the launch container, and the low sampling frequency of

the detumbling control algorithm, the ADCS was not able to detumble the satellite. Instead the rotational rate of the satellite increased. With research and an analysis of A.E. Overlack, presented in [5], they managed to decrease the rotational rate of the satellite up to a level in which its ADCS could function correctly. Currently the Swiss cube is in space for more than 3 years.
Another example of a nano satellite is the Illinois Observing Nanosatellite (ION) [6] built by students from the University of Illinois. The ION satellite was also equipped with an ADCS consisting of 3-axis magnetometer and six sun sensors to provide attitude information and a magnetorquer system to control the attitude. Furthermore the ION was also equipped a micro-vacuum arc thrusters which can rotate the satellite around the x and y-axis and translation in the z-axis. The ION satellite was launched in July 2006 on board a Dnepr rocket from Kazakhstan. Unfortunately the launch of this rocket failed and thus the ION nano satellite has never been operational.
In addition to the just-mentioned, there are many more nano/pico satellite developed that are equipped with an ADCS (e.g. The NCUBE [7] and the AAUSAT3 [8]).
The ADCS of the Delfi-n3Xt is different from the aforementioned Swiss cube and ION, because besides the magnetorquers to control the attitude, the Delfi-n3Xt also has three reaction wheels on board for attitude control. These reaction wheels should provide extra possibilities and more precise control for the Delfi-n3Xt. The downside of the reaction wheels is that the introduce extra complexity to the ADCS and more risk of failures. Furthermore, another difference between the ADCS of the Delfi-n3Xt and the Swiss cube and the ION is that the ADCS of the Delfi-n3Xt has two micro controllers on board, while both Swiss cube and ION only have one micro controller on board their ADCS. Having two micro controllers provides redundancy, because their is a backup when one micro controller fails, but it also introduces more complexity for the ADCS and the OnBoard Computer (OBC).

1-3 Thesis objectives

The main objective of this thesis is to develop a software framework for Delfi-n3Xt's Attitude Determination and Control Subsytem (ADCS) and to verify the correct working with the ADCS hardware. The main objective can be broken down in several sub-objectives, which are listed below.

Design of a software framework for the ADCS The software framework should provide all the functionality to ensure a correct, reliable and robust working of the ADCS. Furthermore, the software framework should be modular to allow extending and re-use of the framework. Moreover, it should provide an easy way to adapt the complete software framework for a different micro controller/processor.

Implementation of a software framework for the ADCS The software framework should be implemented in a layered structure making distinction between hardware, service and application related functionality. This distinction should ensure modularity and make it easy to adapt for different micro controllers/processors. The hardware layer should cover the hardware specific functionality. The service layer should provide services to the application

layer using functions from the hardware layer. The application layer should define the control and data flow of the application, such that the behaviour of the ADCS is correct.

Verifying the ADCS software Before testing the complete software framework directly on the ADCS hardware, different parts and functions should be tested separately. This testing is done in the form of unit tests, which should verify the correct working of a part or function. Using unit tests to verify parts or functions of the software can safe a lot of time in finding the source of an error.

Verifying the ADCS hardware The hardware (i.e. sensors, actuators and micro controller-s/processors) should be tested individually whether they function correctly, before testing the ADCS as a whole. Testing the hardware parts individually can safe a lot of time solving errors that may occur in a later stadium. Because each part is tested individually it is easier to trace down the source of the error.

Verifying the combination ofADCS software and hardware When both ADCS hardware and software are verified individually they should be tested together. Testing the complete ADCS should verify whether the determination and control algorithms work correctly and the behaviour of the ADCS corresponds to the operational mode it is currently in.

1-4 Personal contribution

Since the software ADCS is a complicated and large system to design, implement and verify, work on the ADCS was done in cooperation with L. Rotthier. The work packages were divided in such a way that both had their own piece for which they were responsible. Where Leon Rotthier's focus was mostly on the hardware specific part for the secondary micro controller, a basis for the service layer and algorithmic parts of the software, the focus of the author of this thesis was mostly on the hardware specific part for the primary micro controller, additional parts of the service layer, implementation of the sun vector determination algorithm and verification of the individual hardware, such as the magnetometers, magnetorquers, sun sensors and reaction wheels. Verification of the whole software framework in combination with all the hardware, and analysis of the results was done together with L. Rotthier.
Furthermore the author of this thesis was involved in several general tasks during the development of the Delfi-n3Xt and he provided support to other team members when necessary.

1-5 Thesis structure

In this chapter a short introduction about the subject of this thesis was given together with a problem statement that is inherent to the subject. Furthermore some previous work on the subject was presented, the objectives for this thesis were listed and the personal contribution of the author of this thesis was described.
The structure of the thesis is as follows. In the following chapter, Chapter 2, an overview of the Delfi program is given and the Delfi-n3Xt and its sub-systems are discussed in more detail.

Chapter 3 explains some of the theory behind Attitude Determination and Control, such as attitude reference frames, representations, kinematics and dynamics, sensors, actuators and various determination and control algorithms. A more detailed description of the ADCS hardware of the Delfi-n3Xt is given in Chapter 6. Chapter 6 gives an overview of the ADCS software, describing the structure and various modules. In Chapter 5 a more detail description of the primary micro controller unit of the ADCS is given. Chapter 7 and Chapter 8 describe the verification of the ADCS hardware and software, beginning in Chapter 7 with the hardware and software for the detumble mode followed by the hardware and software for the sun sensors and reaction wheels in Chapter 8. This thesis concludes, in Chapter 9, with a short summary of this thesis followed by some conclusions and further recommendations.

Chapter 2

The Delfi program

The Delfi program is a nano satellite development line at the faculty of Aerospace Engineering, department of Space Systems Engineering, at Delft University of Technology. The program has an educational objective for students as well as an technological objective for the Dutch space industry. Currently there are two satellites in the development line, the Delfi C^3 and its successor the Delfi-n3Xt, and a third one, code name DelFFi, is in the planning.

In this chapter a short overview of Delfi-C^3 is given, see Section 2-1, and a more in depth overview of the Delfi-n3Xt, its mission, its objectives, its subsystems and its payloads are given, see Section 2-2 and its subsections. Readers that are familiar with the Delfi-C^3 and/or Delfi-n3Xt and its subsystems can continue reading the next chapter.

2-1 Delfi-C3 satellite

The Delfi-C^3 was the first nano satellite in the development line in the Delfi program and also the first Dutch nano satellite. In November 2004 the first students, with support of staff members, started working on the Delfi-C^3 and they completed the satellite in 2008. In the same year, on 28th of April, it was launched from India onboard a PSLV-C9 rocket.
The major subsystems of the Delfi-C^3 are the Communication Subsystem, the Electrical Power Subsystem, the Attitude Control & Determination Subsystem and the Structural,Mechanical and Thermal Subsystem. Further more it has two payloads on board, four arrays of Thin Film Solar Cells (TFSC) from Dutch Space and two Autonomous Wireless Sun Sensors (AWWS) from TNO. At time of writing this thesis, the Delfi-C^3 is still operational.

2-2 Delfi-n3Xt satellite

In October 2007 a decision was made to continue with the developement line of nano satellites. Short after this decision four students and staff-members started working on the successor of

the Delfi-C^3. The name given to this successor is 'Delfi-n3Xt', where the '3X' stands for 3-axis stabilised and 'n3Xt' (pronounced 'next) indicates that it is the successor of the Delfi-C^3.
Just like its predecessor the Delfi-n3Xt is a 3-unit CubeSat and has the same major subsystems, however most of the subsystems will be more advanced. Apart from these subsystems the Delfi-n3Xt also has three payloads on board which were developed in cooperation with TNO, Dimes and ISIS. The different subsystems, payloads and advancements will be described in more detail in one of the following subsections.
The Delfi-n3Xt will be launched from Russia onboard a Dnepr LV rocket together with CubeSat's from other universities and a main satellite named DubaiSat-2 [9]. Initialy the launch was planned in September 2012, but due to problems with the main satellite the launch is rescheduled two times. Currently the exact launch date is still unknown.

2-2-1 Objectives and Mission statement

The Delfi-n3Xt mission has three general objectives. The educational objective, the technological objective and the development objective [10].

- **Education**
  The first objective is education. Since the Delfi-n3Xt is mainly developed by students, this is an important objective of the Delfi-n3Xt mission. It gives an opportunity to students from TU Delft, as well as from other universities, to work on a large engineering project which goal is to develop a real satellite. Working on such a project will improve their skills on various kinds of aspects and give the students an optimal preparation for working in the space industry.
- **Technology demonstration & qualification**
  The second objective is technology demonstration and qualification. The Delfi-n3Xt provides a way to demonstrate and/or qualify micro technologies for space applications.
- **Development**
  The last objective is development. With the Delfi-N3Xt mission Delfi Space is aiming to bring the nano satellite platform to a higher level and enable new applications for clusters of nano satellites which are not yet feasible because the high development costs and lack of the current space technology.

The Delfi-n3Xt mission statement [10] can be summarized as follows.

> *Delfi-n3Xt shall be a reliable triple-unit CubeSat of the TU Delft which implements substantial advances in one subsystem with respect to Delfi-C3 and allows technology demonstration of two payloads from external partners from 2012 onwards.*

2-2-2 Advancements

The Delfi-n3Xt has several advancements compared to its predecessor the Delfi-C^3. These advancements are key points to make the mission a success. Listed below are the advancements of a couple of subsystems.

- **Attitude Determination and Control Subsystem**
  One of the key points for the Delfi-n3Xt is to provide a full 3-axis active Attitude Determination and Control Subsytem (ADCS). A full 3-axis ADCS provides more advanced capabilities, such as pointing the satellite's solar panels to the sun, aligning the micro thrusters with the velocity vector or tracking the ground station. Compared to the Delfi-C^3, which only has passive magnetic attitude control and none of these advanced capabilities, this is a great advancement.

- **Electrical Power Subsystem**
  The Electrical Power Subsystem (EPS) of the Delfi-n3Xt provide batteries to store all extra energy. Together with a Battery Management System, which control the charge and discharge of the batteries, they make it possible for the satellite to operate while it is in eclipse (i.e. no sun light on the solar panels) and handle a temporary high power consumption. On the Delfi-C^3 there are no batteries and all extra power is dissipated on the solar panels [11]

- **Command and Data Handling Subsystem**
  On the Delfi-n3Xt the OnBoard Computer (OBC) [12], part of the Command and Data Handling Subsystem (CDHS), will be redundant, making it Single Point of Failure free. With the redundant design and the aim to have a lower bit error rate, the OBC should be more robust and fault tolerant. On the Delfi-C^3 there is no redundant OBC making it less robust and more vulnerable for errors.

- **S-band transmitter**
  The S-band Transmitter (STX) is an experimental radio which should provide much higher downlink data rates. In future missions where higher data rates are needed this might be useful. Because the STX is experimental it will not perform mission critical functions. To make the STX not only experimental the data sent to the other two radios is also send to the STX and stored on a SD card. If the satellite is in eclipse or not in view of the ground station (i.e. the radios can not transmit data to the ground station) all telemetry data is stored on the SD card. When the satellites comes in view of the ground station again, data stored on the SD card can be requested through a telecommand.

2-2-3 Subsystem and payload overview

The Delfi-n3Xt consists of various subsystems and payloads. In this subsection each subsystem and payload is shortly described.

ISIS Transceiver The ISIS Transceiver (ITRX) developed by ISIS BV is a further development of the ISISś UHF/VHF Transceiver, which is again an update of the radios used in the Delfi-C^3. The ITRX is an experimental radio which can be configured while the satellite is in orbit. Characteristics, such as data rate, data protocol, modulation and output power, can be changed, therefore making it very flexible for use in missions with different requirements on these characteristics. For the Delfi-n3Xt mission the ITRX will be configured with a 2400 bps data rate, AFSK modulation for the UHF uplink, BPSK modulation for the VHF downlink

and the protocol will be AX.25. Figure 2-1 and Figure 2-2 show the ITRX and the Linear Transponder.

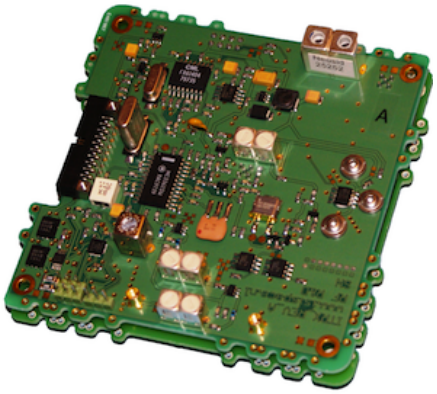


Figure 2-1: Top view of the ITRX.

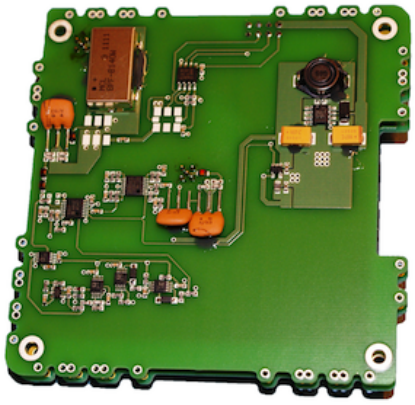


Figure 2-2: The Linear Transponder on top of the ITRX.

T3 Micro propulsion system The T3 Micro Propulsion System (T3uPS) is a payload developed by TNO in cooperation with the TU Delft and the University of Twente. The T3uPS is based on 8 Cold Gas Generators (CGGs). Stored solidified nitrogen can be heated up, thereby releasing some nitrogen gas. The released nitrogen is stored in a plenum chamber. Thrusting can then be performed by opening the valve of the plenum chamber thereby releasing the pressurized nitrogen gas [13]. In Figure 2-3 one can see the T3uPS.

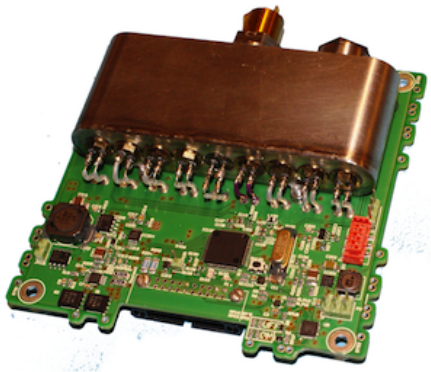


Figure 2-3: The T$^3\mu$PS mounted on a standard Delfi-n3Xt PCB.

Silicon Solar Cell Degradation Measurement The Silicon Solar Cell Degradation Measurement (SDM) is a payload developed by DIMES in cooperation with TU Delft. It consists of 14 small amorphous silicon solar cells which are characterized by measurements from 8 points on the current-voltage curve (IV-curve) of each solar cell and the temperature measurement

of the solar cells. The SDM is mounted on a standard Delfi-n3Xt PCB together with the read-out electronics which reads out the measurement data. The measurement data from the SDM will be send to the ground station in the telemetry frame.Figure 2-4 shows the SDM on top of a standard PCB with the read-out electronics.

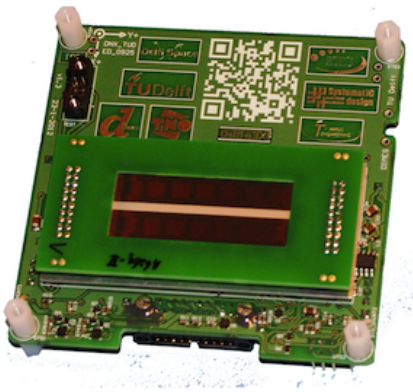


Figure 2-4: The SDM mounted on a standard Delfi-n3Xt PCB with read-out electronics

Electrical Power Subsystem The Electrical Power Subsystem (EPS) will deliver power to the whole satellite. It consists of an energy source, an energy storage, a Global Electrical Power Subsystem (G-EPS) and a Local Electrical Power Subsystem (L-EPS). The energy source are four double-sided solar panels placed in a omni-directional configuration, which consists of TECTAR solar cells, provided by Dutchspace, mounted on PCB substrate. A model of a solar panel can be seen in Figure 2-5
The energy storage consist of four Li-Ion batteries from Samsung with a energy storage capacity of 1.2 MJ and 4 temperature sensors to monitor the temperature of the batteries. Both are placed inside a customized aluminium bracket, which is shown in Figure 2-6.

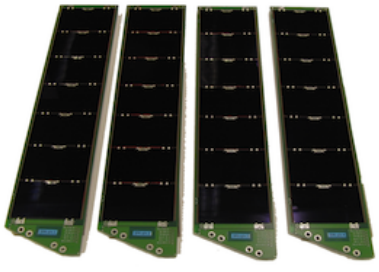


Figure 2-5: A model of one of the Delfi-n3Xt solar panels.

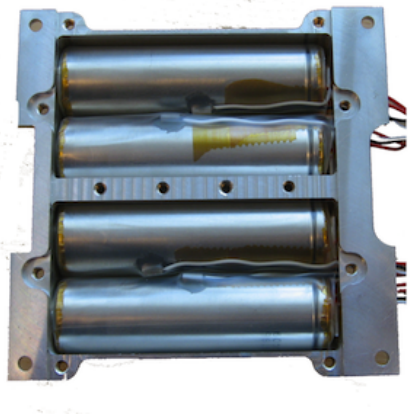


Figure 2-6: Four batteries place in their aluminium bracket.

The G-EPS converts and manages the power delivered to the whole satellite. Itself consists of three subsystems. The Maximum Power Point Tracker (MPPT) board, the Control & Regulation board and the Battery Management board. The MPPT board has eight MPPT's which will retrieve the maximum power for each solar panel side and supply this to the Variable Voltage Bus (VVB) with upper limit of 30V. The Control & Regulation board converts the voltage on the VVB to the regulated 12V bus and retrieves data from the MPPT's and supplies this to the OBC on request.
The Battery Management board consists of four circuits for (dis)charging and thermal protection of the batteries and two micro controllers for control. When the VVB exceeds 22V, the batteries are charged if possible, while reducing the VVB to 22V. If the VVB drops below 18V the batteries are discharged if possible, keeping the VVB at 18V tops.
The L-EPS covers all circuits from local subsystems which convert the 12V from the regulated bus to a 3.3V or 5V logic level used by the local micro controllers. An schematic overview of the EPS can be seen in Figure 2-7.

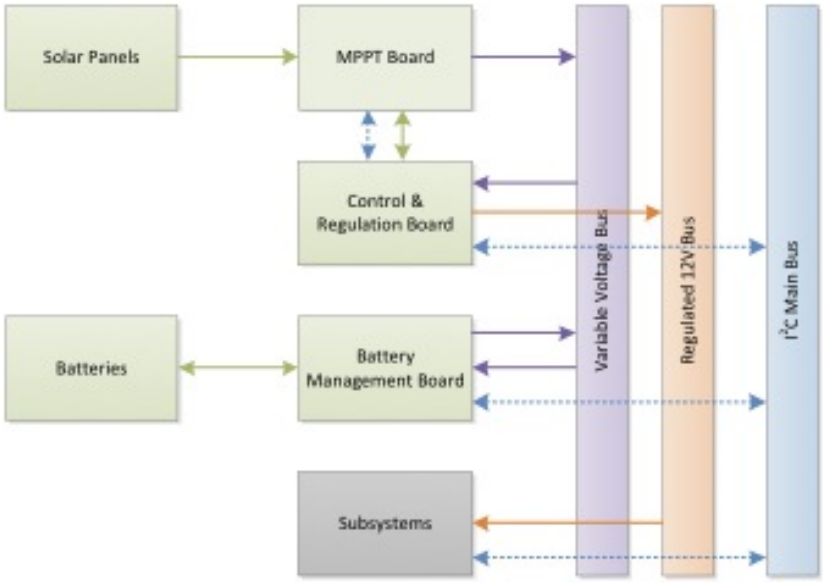


Figure 2-7: Schematic overview of the EPS.

Attitude Determination and Control Subsystem The Attitude Determination and Control Subsytem (ADCS) of the Delfi-n3Xt is a full 3-axis attitude determination and control system. It consists out of two magnetometers and six sun sensors, for determining the attitude, three magnetorquers and three reaction wheels, to control the attitude, and a main controller and back up controller for running the determination and control algorithms [3]. Figure 2-8 shows the bottom of ADCS assembly with the reaction wheels and the magnetorquers, part of the magnetorquer system, mounted on top of the Printed Circuit Board (PCB). In Figure 2-9 one can see a top view of the ADCS, showing the main controller and the magnetorquer drive electronics to drive the magnetorquers. The sun sensors, which are not shown in these figures, are placed on the outer structure and are connected to the ADCS. More details on the ADCS hardware and software can be found in Chapter 4 and Chapter 6.

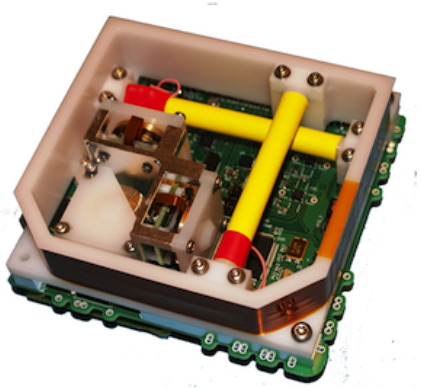


Figure 2-8: Bottom view of the ADCS assembly.

Figure 2-9: Top view of the ADCS assembly.

Communication Subsystem The Communication Subsystem (COMMS) of the Delfi-n3Xt consists out of three radios, phasing circuitry, UHF and VHF antennas and a S-band patch. In Figure 2-10 one can see an overview of the COMMS.

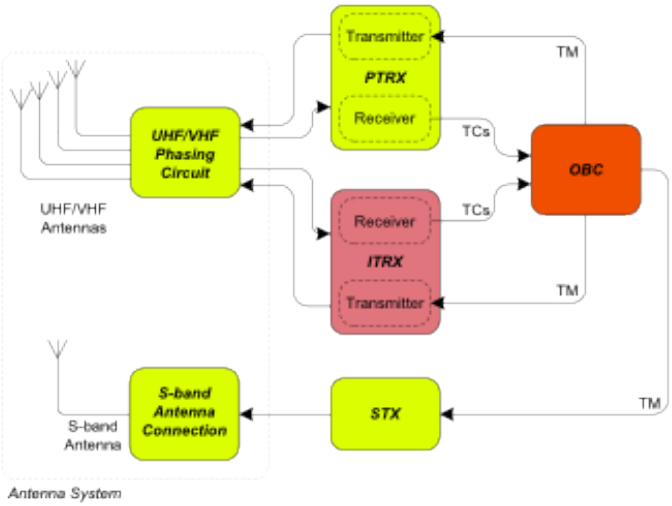


Figure 2-10: Overview of the Delfi-n3Xt communication subsystems

The Primary Transceiver (PTRX) is the primary radio of the Delfi-n3Xt and acts as a transmitter, for sending telemetry frames to the ground station, and as a receiver, for receiving telecommands from the ground station. The ISIS Transceiver (ITRX) will act as a backup radio for the PTRX as well as an experimental radio. Therefore it has all the functionalities of the PTRX and some extra functionalities commonly named 'ITRX tests' [14]. The third radio is the S-band Transmitter (STX), which should provide the satellite with much higher downlink data rates than the other radios. The STX has also a capability of storing teleme-

try frames and returning them when requested. All the UHF and VHF antennas are on the bottom of the Delfi-n3Xt and are connected to each of the radios with two coaxial cables, one for uplink and one for downlink. The antenna of the STX is integrated in the hardware itself.

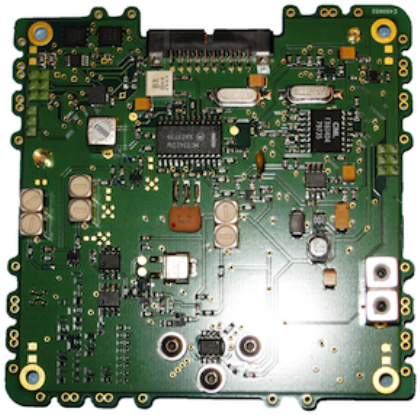


Figure 2-11: Top view of the PTRX.

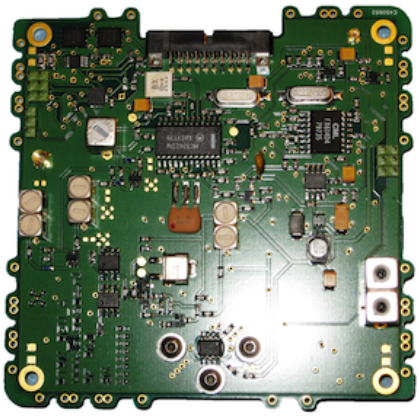


Figure 2-12: The STX (small PCB) connected to its interface board (large PCB)

Structural Subsystem The Structural Subsystem (STS) of the Delfi-n3Xt consists of a outer structure, a Bottom Panel (BOP), a Top Panel (TOP) and an inner structure. The outer structure, the BOP and the TOP are protecting the components inside, adding stiffness to the satellite and are providing fastening for outside components. The dimensions of the structure are compliant with the standard for triple unit CubeSat's [15]. The inner structure is a stack of Printed Circuit Boards (PCBs) fastened on threaded rods. In Figure 2-13 one can see both inner and outer structure.

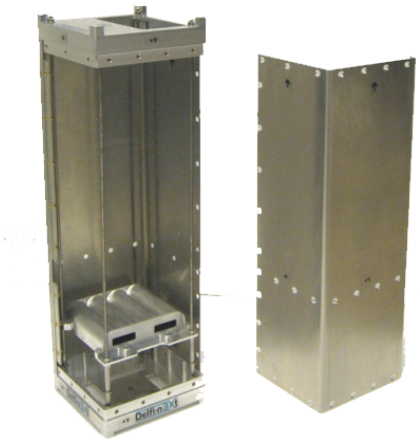


Figure 2-13: The outer and inner structure of the Delfi-n3Xt [1]

Thermal Control Subsystem The Thermal Control Subsystem (TCS) of Delfi-n3Xt is made passive by the use of radiators and thermal tapes. For small components, such as the power amplifiers of the radios, that produce relatively large amount of heat, radiators are placed on top. On the outside and the inside of the outer structure, thermal tapes are applied which can absorb or reflect heat and thereby shifting the temperature range within the satellite.

Command and Data Handling Subsystem The Command and Data Handling Subsystem (CDHS) is all the hardware and software that takes care of the operations and data flow within the satellite. It consists of a redundant OBC and the Delfi Standard System Bus (DSSB). The OBC is the 'brain' of the satellite which can command all the subsystems and requests housekeeping data from all the subsystems and send it through to the radios. The DSSB forms the data and power interfaces between all the subsystems. It consist of a flex-rigid wiring harness with 20 pins Harwin Datamate M80 connectors, a protection circuit and software on each PCB to protect data and power interface and an interface for external monitoring and commanding from the ground.

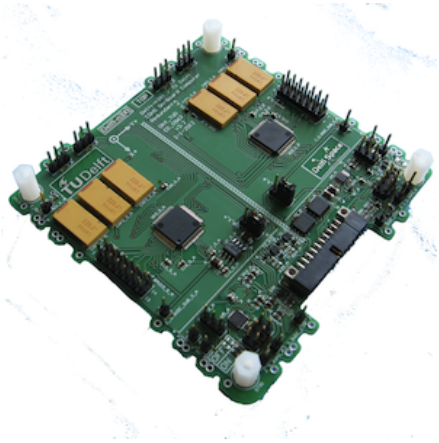


Figure 2-14: PCB with the redundant OnBoard Computers and DSSB protection circuits

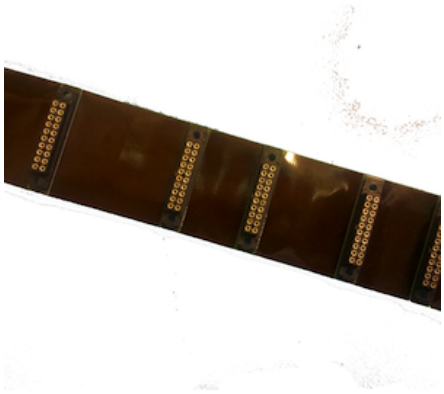


Figure 2-15: Flex-rigid wiring harness with the 20 pins Harwin Datamate M80 connectors

Mechanical Subsystem The Mechanical Subsystem (MechS) consist of the deployment mechanisms for the solar panels and the antenna's. During launch the solar panels are held down by a Hold Down and Release Mechanism (HDRM) [16]. When the satellite is safely in orbit a current is sent through a resistor, heating up the resistor and burning the restraining wire of the HDRM.
The antennas are stored in Modular Antenna Box (MAB) during launch. These MAB's are the same as with the Delfi-C^3, only a small adjustment has been made [17]. Similarly to the deployment of the solar panels, the antennas are also deployed by heating up a resistor which burns through a wire. Both the deployment mechanisms are made redundant in case one fails.

Initiating the deployment is done by the OBC which sends a command to the Deployment Antenna Board (DAB). In Figure 2-16 and Figure 2-17 one can see a MAB and the DAB with deployed antennas.

Figure 2-16: A Modular Antenna Box with an antenna rolled up inside

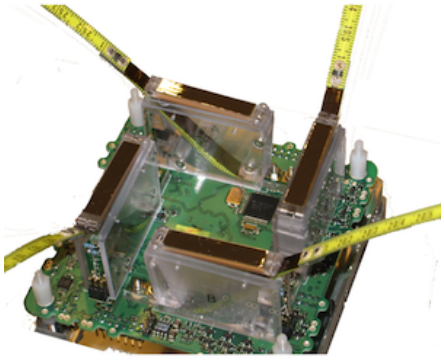


Figure 2-17: The Deployment Antenna Board with all antennas deployed

Chapter 3

Attitude Determination and Control

Attitude determination is the process of determining the orientation of the satellite relative to some reference frame or object. Attitude control is the process of orienting the satellite in a desired direction, maintaining it in its current orientation or controlling its spin. In each satellite some form of attitude determination and control is needed. A satellite's attitude determination and control system typically uses a combination of sensors to provide attitude information and a combination of actuators to control its attitude. Furthermore determination and control algorithms are used to determine the attitude and calculate the control needed to orient or maintain the satellite in a direction.

This chapter gives a more in depth overview of the theory behind attitude determination and control. First, in Section 3-1 different reference frames, in which attitude can be expressed, shall be discussed. Secondly, Section 3-2 describes various ways to represent the attitude of a satellite. Following, in Section 3-4, attitude determination is described in more detail, discussing several sensors and algorithms used in attitude determination. As last, in Section 3-5, attitude control is described, discussing actuators and algorithms to control the attitude.

3-1 Attitude reference frames

To describe a physical vector or attitude of a satellite a reference frame is needed. In attitude determination and control various reference frames are used to express the attitude of a satellite. A reference frame is a set of three orthogonal unit vectors typically noted as the same letter with subscripts 1,2 and 3. For example in a inertial reference frame we would write them as $\{\hat{\boldsymbol{i}}\} = \{\hat{\boldsymbol{i}}_\mathbf{1}, \hat{\boldsymbol{i}}_\mathbf{2}, \hat{\boldsymbol{i}}_\mathbf{3}\}$ and in a body reference frame it would become $\{\hat{\boldsymbol{b}}\} = \{\hat{\boldsymbol{b}}_\mathbf{1}, \hat{\boldsymbol{b}}_\mathbf{2}, \hat{\boldsymbol{b}}_\mathbf{3}\}$. Moreover the shorter notation $\mathcal{F}_i, \mathcal{F}_b$ is used to indicate the inertial or body reference frame. In this subsection several often used reference frames are discussed. Apart from the reference frame discussed in this section, the Delfi-n3Xt uses several others, such as the Sun, Thruster and Ground station pointing reference frame, which are discussed in [3].

3-1-1 Earth Centered Inertial reference frame

The Earth Centered Inertial (ECI) reference frame, with unit vectors $\mathcal{F}_i = \{\hat{\boldsymbol{i}}\} = \{\hat{\boldsymbol{i}}_1, \hat{\boldsymbol{i}}_2, \hat{\boldsymbol{i}}_3\}$, has its origin at the center of mass of the earth. While the earth is rotating with respect to the celestial sphere, the ECI reference frame does not. The x-axis of the ECI reference frame lies in the equatorial plane and points in a fixed direction on the celestial sphere. The z-axis is perpendicular to the x-axis and extends through the North pole. As last the y-axis is perpendicular to both x- and z-axis and completes the right handed reference frame. Figure 3-1 shows the ECI reference frame.

3-1-2 Earth Centered Earth Fixed reference frame

The Earth Centered Earth Fixed (ECEF) reference frame, with unit vectors $\mathcal{F}_e = \{\hat{\boldsymbol{e}}\} = \{\hat{\boldsymbol{e}}_1, \hat{\boldsymbol{e}}_2, \hat{\boldsymbol{e}}_3\}$, also has its origin at the center of mass of the earth. The x-axis of the ECEF reference frame is pointing towards 0 degrees latitude (i.e. the equator) and 0 degree longitude (i.e. Greenwich). The z-axis is pointing north parallel to the rotation axis of the earth. The y-axis is perpendicular to both x- and z-axis and completes the right handed reference frame. The main difference with the ECI reference frame is that the ECEF reference frame rotates along with the earth, therefore the coordinates of a point fixed on the earth surface do not change with respect to the ECEF reference frame. Figure 3-2 shows the ECEF reference frame

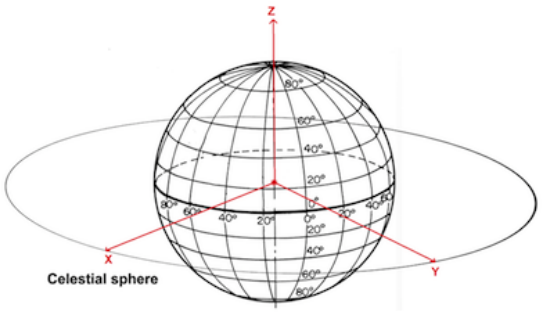


Figure 3-1: The Earth Centered Inertial reference frame.

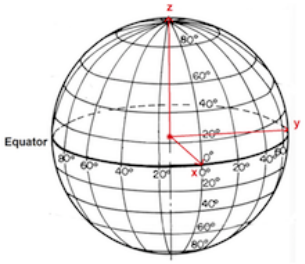


Figure 3-2: The Earth Centered Earth Fixed reference frame.

3-1-3 Orbit Reference Frame

The Orbital Reference Frame (ORF), with unit vectors $\mathcal{F}_o = \{\hat{\boldsymbol{o}}\} = \{\hat{\boldsymbol{o}}_1, \hat{\boldsymbol{o}}_2, \hat{\boldsymbol{o}}_3\}$, has its origin at the center of mass of the satellite. The z-axis of the ORF is pointing towards the Nadir, which is always in the direction of the center of the earth. The y-axis is perpendicular to the z-axis and the orbital plane. The x-axis completes the right handed reference frame

and is the tangent line of the orbit (i.e. the velocity vector). This is only true if the orbit is circular.

3-1-4 Satellite Body Fixed reference frame

The Satellites Body Fixed (SBF) reference frame, with unit vectors $\mathcal{F}_b = \{\hat{\boldsymbol{b}}\} = \{\hat{\boldsymbol{b}}_\mathbf{1}, \hat{\boldsymbol{b}}_\mathbf{2}, \hat{\boldsymbol{b}}_\mathbf{3}\}$, has its origin at the center of mass of the satellite. All three axes, x,y and z are aligned with the three principle axes of inertia of the satellite.

3-2 Attitude representation

There are various ways to represent the attitude of a satellite. In this section an overview of the three most common ways are given, beginning with the Direction Cosine Matrix (DCM) representation, followed by the Euler angles representation and as last the Quaternion representation. The three representations discussed in this section are all used in some way to represent the attitude of the Delfi-n3Xt.

3-2-1 Direction Cosine Matrix

The DCM is a rotation matrix which transforms a vector expressed in an initial reference frame to a target reference frame by expressing the axes of the initial reference frames in terms of the target reference frame. A DCM is composed of the direction cosine values between the initial reference frame and the target reference frame.
Let us consider the transformation of a vector $\boldsymbol{v}_a$ expressed in reference frame $\mathcal{F}_a$ to a vector $\boldsymbol{v}_b$ expressed in reference frame $\mathcal{F}_b$. We can write this rotation as Eq. (3-1).

$$\boldsymbol{v}_b = \boldsymbol{R}_a^b \boldsymbol{v}_b \tag{3-1}$$

Where $\boldsymbol{R}_a^b$ is the rotation matrix from reference frame $\mathcal{F}_a$ to reference frame $\mathcal{F}_b$. If we write this equation in full form we get Eq. (3-2).

$$\begin{bmatrix} v_{b1} \\ v_{b2} \\ v_{b3} \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \begin{bmatrix} v_{a1} \\ v_{a2} \\ v_{a3} \end{bmatrix} \tag{3-2}$$

To determine the matrix $\boldsymbol{R}_a^b$ we write the linear equations for the unit vectors of reference frame $\mathcal{F}_b$ down in terms of the components of the rotation matrix and the unit vectors of reference frame $\mathcal{F}_a$, we get equations (3-3), (3-4) and (3-5).

$$\hat{\boldsymbol{b}}_1 = R_{11}\hat{\boldsymbol{a}}_1 + R_{12}\hat{\boldsymbol{a}}_2 + R_{13}\hat{\boldsymbol{a}}_3 \tag{3-3}$$

$$\hat{\boldsymbol{b}}_2 = R_{21}\hat{\boldsymbol{a}}_1 + R_{22}\hat{\boldsymbol{a}}_2 + R_{23}\hat{\boldsymbol{a}}_3 \tag{3-4}$$

$$\hat{\boldsymbol{b}}_3 = R_{31}\hat{\boldsymbol{a}}_1 + R_{32}\hat{\boldsymbol{a}}_2 + R_{33}\hat{\boldsymbol{a}}_3 \tag{3-5}$$

From Eq. (3-3) we can see that $\hat{\boldsymbol{b}}_1$ is a weighted sum of the unit vectors of reference frame $\mathcal{F}_a$ (i.e $\hat{\boldsymbol{a}}_1, \hat{\boldsymbol{a}}_2, \hat{\boldsymbol{a}}_3$). Thus for example factor R_{11} basically is the amount of $\hat{\boldsymbol{b}}_1$ that extends along the $\hat{\boldsymbol{a}}_1$ axis, it is a projection of $\hat{\boldsymbol{b}}_1$ onto the $\hat{\boldsymbol{a}}_1$ axis. In other words $R_{11} = \hat{\boldsymbol{b}}_1 \cdot \hat{\boldsymbol{a}}_1$ and by the definition of the dot project we can write it as Eq. (3-6) [18].

$$\begin{aligned} R_{11} = \hat{\boldsymbol{b}}_1 \cdot \hat{\boldsymbol{a}}_1 &= |\hat{\boldsymbol{b}}_1||\hat{\boldsymbol{a}}_1| \cos\alpha_{11} \\ &= \cos\alpha_{11} \end{aligned} \tag{3-6}$$

If we do this for every element R_{ij} with $i, j = (1..3)$ we get the following matrix $\boldsymbol{R}_a^b$.

$$\boldsymbol{R}_a^b = \begin{bmatrix} \cos\alpha_{11} & \cos\alpha_{12} & \cos\alpha_{13} \\ \cos\alpha_{21} & \cos\alpha_{22} & \cos\alpha_{23} \\ \cos\alpha_{31} & \cos\alpha_{32} & \cos\alpha_{33} \end{bmatrix} \tag{3-7}$$

Matrix $\boldsymbol{R}_a^b$ is called the Direction Cosine Matrix (DCM) and can transform any vector expressed in reference frame $\mathcal{F}_a$ to reference frame $\mathcal{F}_b$. The matrix $\boldsymbol{R}_a^b$ is a unique DCM representation. There is one and only one DCM that transforms a vector expressed in reference frame $\mathcal{F}_a$ to reference frame $\mathcal{F}_b$. The DCM also has the important property that the rotation matrix $\boldsymbol{R}_a^b$ is equal to the transpose and inverse of the rotation matrix $\boldsymbol{R}_b^a$ [19].

$$\boldsymbol{R}_a^b = \boldsymbol{R}_b^{aT} = \boldsymbol{R}_b^{a-1} \tag{3-8}$$

Thus if we have the DCM from reference frame $\mathcal{F}_a$ to reference frame $\mathcal{F}_a$ we can easily find the DCM to transform a vector from reference frame $\mathcal{F}_a$ back to reference frame $\mathcal{F}_a$, by simply transposing or inverting the DCM.
Another useful property of the DCM is that if we want to transform a vector from a reference frame $\mathcal{F}_a$ to a reference frame $\mathcal{F}_c$, but we only have the rotation matrices $\boldsymbol{R}_a^b$ and $\boldsymbol{R}_b^c$. Then we can perform two successive rotations in order to get the rotation matrix that transforms a vector from reference frame $\mathcal{F}_a$ to $\mathcal{F}_c$ [20]. In Eq. (3-9) these successive rotations are depicted.

$$\boldsymbol{R}_a^c = \boldsymbol{R}_a^b \boldsymbol{R}_b^c \tag{3-9}$$

The above mentioned property can also be extended to more than two successive rotation.
Within the Delfi-n3Xt the DCM is used in several occasions to transform from one reference frame to an other. Furthermore the DCM is also used to derive the attitude quaternion, which is an other representation of the attitude discussed in Subsection 3-2-3.

3-2-2 Euler angles

The DCM described in the previous subsection uses nine parameters to describe the attitude of a satellite, from which six are redundant. Using nine parameters to describe the attitude requires a lot of computational power. Another way of representing the attitude, which only uses three parameters, are the Euler angles.
According to Euler's rotation theorem any rotation can be described using three successive rotations around the principle axes with angles ψ, θ and ϕ. These angles are called the Euler angles. The three successive rotations can be chosen in any order and multiple rotations around the same axis may be chosen. However, successive rotations should be around a different axis. For example a rotation sequence around the X axis, then the Y axis and again around the X axis is valid, but a rotation sequence around the X axis, then again around the X axis and finally around the Y axis is not a valid rotation sequence.
Let us consider a reference frame $\mathcal{F}_a$, with unit vectors $\{\hat{\boldsymbol{a}}_1, \hat{\boldsymbol{a}}_2, \hat{\boldsymbol{a}}_3\}$ as frame axes, and another reference frame $\mathcal{F}_b$, with unit vectors $\{\hat{\boldsymbol{b}}_1, \hat{\boldsymbol{b}}_2, \hat{\boldsymbol{b}}_3\}$ as frame axes. Suppose we want to transform the axes of reference frame $\mathcal{F}_a$ to reference frame $\mathcal{F}_b$. Euler's rotation theorem says that we can do this by three successive rotations around the axes of reference frame $\mathcal{F}_a$ with angles ψ, θ and ϕ. If we take the first rotation to be around the axis $\hat{\boldsymbol{a}}_3$ with ϕ, the second rotation around $\hat{\boldsymbol{a}}_2$ with θ and the last rotation around $\hat{\boldsymbol{a}}_1$ with ψ, we can then write this successive set of rotations as Eq. (3-10).

$$\boldsymbol{R}_a^b = \boldsymbol{R}_{a_3}(\phi)\boldsymbol{R}_{a_2}(\theta)\boldsymbol{R}_{a_1}(\psi) \tag{3-10}$$

The rotation matrices for the rotations about the axes $\hat{\boldsymbol{a}}_1$, $\hat{\boldsymbol{a}}_2$, $\hat{\boldsymbol{a}}_3$ can be written as Eq. (3-11),Eq. (3-12) and Eq. (3-13) [21].

$$\boldsymbol{R}_{\hat{a}_1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{bmatrix} \tag{3-11}$$

$$\boldsymbol{R}_{\hat{a}_2} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3-12}$$

$$\boldsymbol{R}_{\hat{a}_3} = \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3-13}$$

With the three rotation matrices from Eq. (3-11), Eq. (3-12) and Eq. (3-13), we can write Eq. (3-10) as Eq. (3-14).

$$\begin{aligned}\boldsymbol{R}_a^b &= \begin{bmatrix}1 & 0 & 0\\ 0 & \cos\psi & \sin\psi\\ 0 & -\sin\psi & \cos\psi\end{bmatrix}\begin{bmatrix}\cos\theta & 0 & -\sin\theta\\ 0 & 1 & 0\\ \sin\theta & 0 & \cos\theta\end{bmatrix}\begin{bmatrix}\cos\phi & \sin\phi & 0\\ sin\phi & \cos\phi & 0\\ 0 & 0 & 1\end{bmatrix}\\ &= \begin{bmatrix}\cos\theta\cos\phi & \cos\theta\sin\phi & -\sin\theta\\ \begin{pmatrix}\sin\psi\sin\theta\cos\phi\\ -\cos\psi\sin\phi\end{pmatrix} & \begin{pmatrix}\sin\psi\sin\theta\sin\phi\\ +\cos\psi\cos\phi\end{pmatrix} & \sin\psi\cos\theta\\ \begin{pmatrix}\cos\psi\sin\theta\cos\phi\\ +\sin\psi\sin\phi\end{pmatrix} & \begin{pmatrix}\cos\psi\sin\theta\sin\phi\\ -\sin\psi\cos\phi\end{pmatrix} & \cos\psi\cos\theta\end{bmatrix}\end{aligned} \tag{3-14}$$

Given the rotation matrix $\boldsymbol{R}_a^b$ from Eq. (3-14), the transformation of the axes of reference frame $\mathcal{F}_a$ to reference frame $\mathcal{F}_b$ becomes Eq. (3-15).

$$\begin{bmatrix}\hat{\boldsymbol{b}}_1\\ \hat{\boldsymbol{b}}_2\\ \hat{\boldsymbol{b}}_3\end{bmatrix} = \begin{bmatrix}\cos\theta\cos\phi & \cos\theta\sin\phi & -\sin\theta\\ \begin{pmatrix}\sin\psi\sin\theta\cos\phi\\ -\cos\psi\sin\phi\end{pmatrix} & \begin{pmatrix}\sin\psi\sin\theta\sin\phi\\ +\cos\psi\cos\phi\end{pmatrix} & \sin\psi\cos\theta\\ \begin{pmatrix}\cos\psi\sin\theta\cos\phi\\ +\sin\psi\sin\phi\end{pmatrix} & \begin{pmatrix}\cos\psi\sin\theta\sin\phi\\ -\sin\psi\cos\phi\end{pmatrix} & \cos\psi\cos\theta\end{bmatrix}\begin{bmatrix}\hat{\boldsymbol{a}}_1\\ \hat{\boldsymbol{a}}_2\\ \hat{\boldsymbol{a}}_3\end{bmatrix} \tag{3-15}$$

The rotation sequence we used to rotate reference frame $\mathcal{F}_a$ to reference frame $\mathcal{F}_b$ can be denoted by three numbers, corresponding to the axis around which we rotated. Therefore the sequence used in the equations above is the 321-sequence. In total there are twelve valid rotation sequences (i.e. 123, 121, 131, 132, 213, 212, 231, 232, 312, 313, 321, and 323) with each a unique rotation matrix [21].
Although it is very convenient and much less computational intensive to use only three angles to describe the attitude of a satellite, the Euler angles suffers from a singularity problem which is said to arise from the "gimbal lock" [21].
Just as with the DCM the Euler angles are used within the Delfi-n3Xt to describe the transformation from one reference frame to an other. As an example the Q1-method described in [3] and is used for Coarse Sun Pointing (CSP) mode, uses Euler angles to describe the relation between the SBF and the Sun Pointing reference frame.

3-2-3 Quaternions

Quaternions, also referred to as the Euler symmetric parameters, is another way and one of the most useful ways to describe the attitude of a satellite. The quaternion representation is based on the Euler's rotational theorem which states that the relative orientation between two reference frames can be described by only one rotation around a fixed axis.
A quaternion is a column vector consisting of a scalar part q_4 and a vector part $\mathbf{q}$, as depicted in Eq. (3-16).

$$\boldsymbol{q} = \begin{bmatrix}\mathbf{q}\\ q_4\end{bmatrix} = \begin{bmatrix}q_1\\ q_2\\ q_3\\ q_4\end{bmatrix} \tag{3-16}$$

The components q_1, q_2 and q_3 are the vector part and the component q_4 is the scalar part of the quaternion. These components can be written in terms of the Euler axis $\hat{\boldsymbol{e}} = \{e_1, e_2, e_3\}$ and an angle θ according to Eq. (3-17).

$$\begin{aligned} q_1 &= \hat{e}_1 \sin\left(\frac{\theta}{2}\right) \\ q_2 &= \hat{e}_2 \sin\left(\frac{\theta}{2}\right) \\ q_3 &= \hat{e}_3 \sin\left(\frac{\theta}{2}\right) \\ q_4 &= \cos\left(\frac{\theta}{2}\right) \end{aligned} \tag{3-17}$$

The four components also satisfy the constraint from Eq. (3-18) [20].

$$\sqrt{q_1^2 + q_2^2 + q_3^2 + q_4^2} = \| \boldsymbol{q} \| = 1 \tag{3-18}$$

There are a number of similarities between quaternions and the DCM. Just as with the DCM matrix $\boldsymbol{R}_a^b$, the quaternion $\boldsymbol{q}_a^b$ describes the attitude of a satellite in reference frame $\mathcal{F}_a$ with respect to reference frame $\mathcal{F}_a$. Since they are both ways to represent the attitude of a satellite the DCM can be expressed in terms of quaternions and vice versa. Eq. (3-19) shows how the DCM is expressed in terms of a quaternion [20].

$$\boldsymbol{R}(\boldsymbol{q}) = \begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1q_2 + q_3q4) & 2(q_1q_3 - q_2q_4 \\ 2(q_1q_2 - q_3q4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 + q_1q4) \\ 2(q_1q_3 + q_2q_4 & 2(q_2q_3 - q_1q4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix} \tag{3-19}$$

$$= (q_4^2 - \| \mathbf{q} \|^2) I_3 + 2\mathbf{q}\mathbf{q}^T - 2q_4[\mathbf{q}\times] \tag{3-20}$$

where $\mathbf{q}$ is the vector part of the quaternion $\boldsymbol{q}$ and $[\mathbf{q}\times]$ the skew symmetric defined as Eq. (3-21).

$$[\mathbf{q}\times] = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 0 & \end{bmatrix} \tag{3-21}$$

Vice versa the quaternion components are expressed in terms of the matrix components of a DCM as shown in Eq. (3-22) [20].

$$\begin{aligned} q_1 &= \frac{1}{4q_4}(R_{23} - R_{32}) \\ q_2 &= \frac{1}{4q_4}(R_{31} - R_{13}) \\ q_3 &= \frac{1}{4q_4}(R_{12} - R_{21}) \\ q_4 &= \pm\frac{1}{2}(1 + R_{11} + R_{22} + R_{33})^{\frac{1}{2}} \end{aligned} \tag{3-22}$$

A second similarity between the DCM and the quaternion representation is that if the quaternion $\boldsymbol{q}_a^b$ is known it is easy to find the quaternion $\boldsymbol{q}_b^a$ that describes the attitude of satellite in reference frame $\mathcal{F}_b$ with respect to $\mathcal{F}_a$. This is done by taking the conjugate of $\boldsymbol{q}_a^b$ as shown in Eq. (3-23) [22].

$$\boldsymbol{q}_b^a = \overline{\boldsymbol{q}_a^b} \tag{3-23}$$

Where the conjugate of a quaternion is defined by Eq. (3-24) [20].

$$\overline{\boldsymbol{q}} = \begin{bmatrix} \mathbf{-q} \\ q_4 \end{bmatrix} = \begin{bmatrix} -q_1 \\ -q_2 \\ -q_3 \\ q_4 \end{bmatrix} \tag{3-24}$$

Furthermore another similarity is that a rotation such as in Eq. (3-1) can also be expressed by a multiplication of quaternions, as shown in Eq. (3-25).

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ 0 \end{bmatrix} = \overline{\boldsymbol{q}}_a^b \otimes \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ 0 \end{bmatrix} \otimes \boldsymbol{q}_a^b \tag{3-25}$$

Where the $\otimes$ operation is a quaternion multiplication as defined in [20]. Quaternion multiplication is almost equal to multiplication of complex numbers, but with quaternions the order in which they are multiplied has to be taken into account.
One final similarity is that we can perform two successive rotations, as shown in Eq. (3-9), to get the right quaternion that transforms a vector to another frame [20]. Eq. (3-26) shows two successive rotations for quaternions.

$$\boldsymbol{q}_a^c = \boldsymbol{q}_b^c \otimes \boldsymbol{q}_a^b \tag{3-26}$$

The advantage of using quaternions instead of the DCM, is that relative to the DCM they are very compact. Instead of nine parameters in the DCM, the quaternions uses only four parameters to describe the attitude. Another advantage of the quaternion representation is that they do not have the singularity problem that the Euler angle representation has.
Within the Delfi-n3Xt the quaternion representation is used in the advanced modes to represent its attitude. Using this representation also makes it possible to use the Quaternion Feedback Regulator, described in [3] and Subsection 3-5-2-2, for controlling the attitude of the Delfi-n3Xt.

3-3 Attitude kinematics and dynamics

3-3-1 Angular velocity

In the previous section it was explained how the attitude could be described as static representation between the different reference frames, from Section 3-1, using various representations.

This static representation is however not sufficient, because the orientation of different reference frames often change with respect to each other. This change can be expressed as the angular velocity. The angular velocity is the speed at which one reference frame rotates with respect to an other reference frame. A short notation for the rotation between reference frames is $\boldsymbol{\omega}_a^b$, which describes the angular velocity of reference frame $\mathcal{F}_b$ with respect to reference frame $\mathcal{F}_a$. In this subsection a few important properties of angular velocities are described which can also be found in different texts [20], [22], [23].
A first important property is that if reference frame $\mathcal{F}_b$ is rotating with angular velocities of $\boldsymbol{\omega}_a^b$ with respect to $\mathcal{F}_a$, then reference frame $\mathcal{F}_a$ is rotating with an angular velocity of $-\boldsymbol{\omega}_b^a$ with respect to $\mathcal{F}_b$. This can be denoted as Eq. (3-27).

$$\boldsymbol{\omega}_a^b = -\boldsymbol{\omega}_b^a \tag{3-27}$$

Another important property of angular velocity is the additive property. Angular velocities of different reference frame can be added to obtain an angular velocity in terms of angular velocities from intermediate frames.

$$\boldsymbol{\omega}_a^c = \boldsymbol{\omega}_a^b + \boldsymbol{\omega}_b^c \tag{3-28}$$

Eq. (3-27) says that if we have three reference frames $\mathcal{F}_a$, $\mathcal{F}_b$, and $\mathcal{F}_c$, where reference frame $\mathcal{F}_b$ is rotating with angular velocity $\boldsymbol{\omega}_b^a$ with respect to $\mathcal{F}_a$ and reference frame $\mathcal{F}_c$ is rotating with angular velocity $\boldsymbol{\omega}_c^b$ with respect to $\mathcal{F}_b$, then reference frames $\mathcal{F}_c$ is rotating with respect to $\mathcal{F}_a$ by the sum of these two angular velocities.
A final important property is taking the derivative of vectors in various reference frames. Angular velocities are relative to the frame in which they are computed. The derivative of a vector is also relative to its reference frame. Therefore taking a derivative of a vector in one reference frame relates to the derivative of the same vector in another reference frame by the angular velocity that these two frames have with respect to each other.

$$\left\{\frac{d\boldsymbol{v}}{dt}\right\}_b = \left\{\frac{d\boldsymbol{v}}{dt}\right\}_a + \boldsymbol{\omega} \times \boldsymbol{v} \tag{3-29}$$

Eq. (3-29) states that the derivative of a vector in reference frame $\mathcal{F}_a$ equals the derivative of the same vector in reference frame $\mathcal{F}_a$ plus the outer product of the angular velocities of reference frame $\mathcal{F}_b$ with respect to $\mathcal{F}_a$ and the vector itself.

3-3-2 Angular momentum

The angular momentum is closely related to the angular velocity described in the previous subsection. The angular momentum is the dot product of a body's moment of inertia and its angular velocity. If a rigid body, such as a satellite, with reference frame $\mathcal{F}_b$ rotates with an angular velocity ω_i^b with respect to an inertial reference frame, then the angular momentum is defined as Eq. (3-30).

$$\boldsymbol{h}_b = \boldsymbol{J}\boldsymbol{\omega} \tag{3-30}$$

Where $\boldsymbol{\omega}$ is the angular velocity between the reference frames and $\boldsymbol{J}$ is the tensor of inertia, which is a measure of resistance to changes in rotational velocity and is defined by Eq. (3-31).

$$\boldsymbol{J} = \begin{bmatrix} J_x & -J_{xy} & -J_{xz} \\ -J_{xy} & J_y & -J_{yz} \\ -J_{xz} & -J_{yz} & J_z \end{bmatrix} \tag{3-31}$$

The components of the matrix $\boldsymbol{J}$ are integrals which represent the mass distribution of the rigid body.

3-3-3 Equations of motion

The equations of motion can be subdivided into the dynamic equations of motion and the kinematic equations of motion. Dynamics is the study of the behaviour of a system or body in time. Kinematics is the study that describe the motion of a point or body without considering the forces that are causing that motion. In this subsection both dynamic and kinematic equations of motion are discussed.

Dynamic equation The dynamic equation describes the development of angular velocities of rigid body under influence of external torques. As stated in Subsection 3-3-2 the angular momentum of a rigid body is defined by Eq. (3-30). We also know from mechanics that the time derivative of the angular momentum at a certain time is equal to the total external torque that is acting on the rigid body at a certain time. So suppose we have a total external torque τ_{ext} acting on a rigid body. The angular momentum then corresponds to Eq. (3-32).

$$\left\{ \frac{d\boldsymbol{h}}{dt} \right\}_i = \boldsymbol{\tau}_{ext} \tag{3-32}$$

From Eq. (3-29) we know that the time derivative of a vector in one reference frame equals the derivative of the same vector in an other reference frame plus the outer product of the angular velocities between both reference frames and the vector itself. If we use this derivative property of a vector, described in Eq. (3-29), and apply it to Eq. (3-32) we get Eq. (3-33).

$$\begin{aligned} \left\{ \frac{d\boldsymbol{h}}{dt} \right\}_b + \boldsymbol{\omega} \times \boldsymbol{h} &= \boldsymbol{\tau}_{ext} \\ \left\{ \frac{d\boldsymbol{h}}{dt} \right\}_b &= \boldsymbol{\tau}_{ext} - \boldsymbol{\omega} \times \boldsymbol{h} \end{aligned} \tag{3-33}$$

Eq. (3-33) relates the angular velocity to the total external torque, where $\boldsymbol{\omega}$ is the angular velocity between both reference frames and $\boldsymbol{h}$ the angular momentum. From [20] we also know that the matrix $\boldsymbol{J}$ from Eq. (3-30) has an inverse. Multiplying both sides of Eq. (3-30) with the $\boldsymbol{J}^{-1}$, then taking the derivative on both sides with respect to reference frame $\mathcal{F}_b$ and combining it with Eq. (3-33) we get Eq. (3-34).

$$\frac{d\boldsymbol{\omega}}{dt} = \boldsymbol{J}^{-1} \left(\boldsymbol{\tau}_{ext} - \boldsymbol{\omega} \times (\boldsymbol{J}\boldsymbol{\omega}) \right) \tag{3-34}$$

Eq. (3-34) forms the dynamic equation and describes the change of the angular velocity as a function of the current angular velocity, its moment of inertia matrix and the external torque acting on the rigid body.

Kinematic equations The kinematic equations are a set of differential equations which describe the time derivative (i.e. the evolution in time) of the parameters in which the attitude is represented. In Section 3-2 three different attitude representation have been presented, each with their own set of kinematic equations. We only consider the kinematic equations for the quaternion representation in this paragraph, because this representation has proven to be the most useful for satellite kinematics.
Suppose we have a quaternion, $\boldsymbol{q}(t)$, which represents the attitude of the satellite at time t and a quaternion, $\boldsymbol{q}(t+\Delta t)$, which represents the attitude of the satellite at time $t+\Delta t$. We can write $\boldsymbol{q}(t+\Delta t)$ as Eq. (3-35) [20].

$$\boldsymbol{q}(t+\Delta t) \approx [\boldsymbol{I}_4 + \frac{1}{2}\Omega]\boldsymbol{q}(t) \tag{3-35}$$

Where Ω is the skew symmetric matrix as shown in Eq. (3-36) and $\boldsymbol{I}_4$ a four by four identity matrix.

$$\boldsymbol{\Omega} = \begin{bmatrix} 0 & \omega_3 & -\omega_2 & \omega_1 \\ -\omega_3 & 0 & -\omega_1 & \omega_2 \\ \omega_2 & -\omega_1 & 0 & \omega_3 \\ -\omega_1 & -\omega_2 & -\omega_3 & 0 \end{bmatrix} \tag{3-36}$$

Then using the definition of the derivative we can write it as Eq. (3-37) [20].

$$\frac{d\boldsymbol{q}(t)}{dt} = \lim_{\Delta t \to 0} \frac{\boldsymbol{q}(t+\Delta t) - \boldsymbol{q}(t)}{\Delta t} \tag{3-37}$$

$$= \lim_{\Delta t \to 0} \frac{[\boldsymbol{I}_4 + \frac{1}{2}\Omega]\boldsymbol{q}(t) - \boldsymbol{q}(t)}{\Delta t} \tag{3-38}$$

$$= \frac{1}{2}\boldsymbol{\Omega}\boldsymbol{q} \tag{3-39}$$

Eq. (3-37) is the kinematic differential equation for the quaternion representation and describes the evolution of the attitude as a function of the rotational rates.
For discrete time controlled systems the solution to the kinematic differential equation from Eq. (3-37) is denoted as Eq. (3-40).

$$\boldsymbol{q}_{k+1} = \boldsymbol{\Phi}_k \boldsymbol{q}_k \tag{3-40}$$

Where Φ_k is given as Eq. (3-41).

$$\boldsymbol{\Phi}_k = e^{\frac{1}{2}\boldsymbol{\Omega}_k \Delta t} \tag{3-41}$$

Given that Δt is small and $\boldsymbol{\Omega}$ is constant.

3-4 Attitude determination

Determining the attitude of a satellite is equivalent to determining the rotation matrix that describes the orientation of the SBF reference frame with respect to a known reference frame (e.g the ECI reference frame). The basic idea behind determining this rotation matrix is to use a variety of sensors and mathematical models to collect vector components in the different reference frames, described in Section 3-1. These vector components are then used in one of the determination algorithms to determine the rotation matrix, usually in the form of a DCM, Euler angles or quaternions, which where described in Subsection 3-2-1. In the following subsections a couple of sensors and determination algorithms are discussed.

3-4-1 Sensors

To determine a satellites attitude a variety of sensors are used. These sensors can be divided into two main categories, reference sensors and inertial sensors. Often these two classes of sensors are used to complement each other in a measurement system. Reference sensors measure the direction of a known vector e.g. the Sun or Earth's magnetic field. Inertial sensors measure rotation and/or translational acceleration with respect to an inertial reference frame. In this subsection a couple of these reference sensors and inertial sensors are briefly described.

Magnetometer Magnetometers are widely used sensors to determine the attitude of a satellite. This is because they provide information about the direction as well as the magnitude, are lightweight, reliable and are low power devices. Three sensor elements inside the magnetometer measure the earth's magnetic field in all three axes. Data of the magnetometer is processed, usually using a Kalman filter, and combined with a model of the earth's magnetic field. A frequently used earth magnetic field model is the International Geomagnetic Reference Field (IGRF) which is developed by several institutes. Although magnetometer give reliable information some things have to be taken into account when using the magnetometer data. Electronics on the satellite can cause a disturbance field, causing the magnetometer to not only measure the Earth's magnetic field but also the disturbance field. In order to minimize this influence it is important to consider the placement of different electronic components that can cause magnetic disturbance and/or use shielding. Another concern is the IGRF model. Since it is only a model of the earth magnetic field it will never be totally accurate, therefore inducing errors when estimating the attitude. Depending on the requirements for the accuracy this can or can not be a problem. Also should be noted that magnetometers are only useful when the satellite is in a low orbit where the earth magnetic field is strong enough and well modelled.

Sun Sensor Sun sensors are one of the most common used sensors for attitude determination. They can determine the orientation of the satellite, relatively to the sun, by measuring the amount of light incidence. There are different types of sun sensors. There are the simple sun presence detectors that can detect whether the sun is in the field of view of the detector. Furthermore, there are the analogue sun sensors with lesser accuracy, ranging between 1 and 6 degrees, on one end and the digital ones with higher accuracy, up to 0.5 degree [20], on the other end. Although sun sensors can determine the orientation of a satellite accurately,

they need a light source to do so. Since most low-earth orbits include eclipse periods (i.e. absence of a light source) there must be a way of handling this loss of reference, for example by making use of other type sensors.

Star Tracker Star trackers are the most accurate sensors to determine the attitude of a satellite. An accuracy down to the arc second range can be achieved [20]. To determine the attitude one or more star tracker sensors use their camera to make a picture of the sky and compare this image with an on board catalogue with images of the starry sky. Although this sensor can achieve a high attitude accuracy, it has the disadvantage that it is expensive, large and heavy and uses more power than other sensors. Therefore this sensor is not feasible for most CubeSat's.

Earth Horizon Sensor Earth horizon sensors provide the satellite with attitude knowledge relative to the Earth. It measures infra-red radiation coming from the direction of Earth's horizon so that the angle between the horizon and the sensor can be determined.

Global Positioning System A Global Positioning System (GPS) can be used to determine the attitude of a satellite to a fraction of a degree. To determine the attitude of a satellite in three axes at least three antennas are needed. These antennas receive a GPS signal from a GPS satellite. With the phase difference between the signals, that are received on the three antennas, and the known distance between these antennas the attitude of the satellite can be determined.

Gyroscope In contrast to the other sensors the gyroscope is not a reference sensor but an inertial sensor. The gyroscope measures angular acceleration in three axes. Integrating these accelerations over time gives you a certain displacement in each axis. Together with an inertial reference this gives an estimate of the current attitude of the satellite. A disadvantage of the gyroscope is that it is drifting over time, inducing displacement errors. The gyroscope is therefore more suitable in combination with other sensors or for measuring rapid changes in attitude.

3-4-2 Determination algorithms

In general the attitude determination algorithms can be divided into two groups; the deterministic algorithms where the attitude is determined based on two or more vector observations from a single point in time and the recursive algorithms where filters or stochastic estimators use measurements from several sensors often combined with dynamic and kinematic models in order to give an estimate of the attitude. In this subsection some of the well known algorithms of both groups are discussed.

3-4-2-1 Deterministic algorithms

TRIAD algorithm The TRIAD algorithm, introduced by Black in 1964 [24], is one of the earliest algorithms used for attitude determination. It is an attitude determination algorithm that uses two pairs of linearly independent corresponding vector observations, in different reference frames to find a rotation matrix between these frames. In attitude determination often used pairs of vector observations are the Earth magnetic field and the sun vector in the ECI reference frame and in the SBF reference frame measured by the on board magnetometer and the sun sensors respectively. The TRIAD algorithm consist of three steps:

Step 1: Obtain the two pair of vector observations.
Step 2: Create an intermediate reference frame, which is common to both reference frames.
Step 3: Solving the complete rotation matrix between the two reference frames.

For our example we will be using the following two pair of vector observations. The magnetic field vector and the sun vector in the SBF reference frame, $\boldsymbol{B}_b$ and $\boldsymbol{S}_b$, and the known magnetic field vector and the sun vector in the ECI reference frame, $\boldsymbol{B}_i$ and $\boldsymbol{S}_i$.
To create the base vectors for an intermediate reference frame $\mathcal{F}_t$ we use the relative orientation between the two vector observations. The relative orientation between the two vector observations is the same in every reference frame and thus it does not matter which pair of vector observations from one of the reference frames we use to create the base vectors for the intermediate reference frame. Equations (3-42) and (3-43) show the base vectors for the intermediate reference frame created from the vector pairs in the SBF and the ECI reference frame.

$$\boldsymbol{t}_{1b} = \frac{\boldsymbol{B}_b}{\parallel \boldsymbol{B}_b \parallel} \qquad \boldsymbol{t}_{2b} = \frac{\boldsymbol{B}_b \times \boldsymbol{S}_b}{\parallel \boldsymbol{B}_b \times \boldsymbol{S}_b \parallel} \qquad \boldsymbol{t}_{3b} = \boldsymbol{t}_{1b} \times \boldsymbol{t}_{2b} \tag{3-42}$$

$$\boldsymbol{t}_{1i} = \frac{\boldsymbol{B}_i}{\parallel \boldsymbol{B}_i \parallel} \qquad \boldsymbol{t}_{2i} = \frac{\boldsymbol{B}_i \times \boldsymbol{S}_i}{\parallel \boldsymbol{B}_i \times \boldsymbol{S}_i \parallel} \qquad \boldsymbol{t}_{3i} = \boldsymbol{t}_{1i} \times \boldsymbol{t}_{2i} \tag{3-43}$$

The three base vectors from Equations (3-42) and (3-43) form together the rotation matrices $\boldsymbol{R}_b^t$ and $\boldsymbol{R}_i^t$, which transform any vector observation from the SBF reference frame to the intermediate reference frame $\mathcal{F}_t$ and any vector observation from the ECI reference frame to the intermediate reference frame $\mathcal{F}_t$.
With the two rotation matrices to the intermediate reference frame we can now find the rotation matrix from the SBF directly to the ECI reference frame by simply applying two successive rotations as shown in Eq. (3-44).

$$\boldsymbol{R}_b^i = \boldsymbol{R}_b^t \boldsymbol{R}_t^i \tag{3-44}$$

To get the matrix $\boldsymbol{R}_t^i$ we apply Eq. (3-8), which states that $\boldsymbol{R}_t^i = (\boldsymbol{R}_i^t)^T$. The resulting matrix from Eq. (3-44) can transform any vector observation from the SBF reference frame directly to the ECI reference frame.
In [23] an example is shown in which the TRAID method, described above, is used to derive the rotation matrices which rotate a vector in one reference frame to an other.

Davenport's q-Method Davenport's q-method is one of the first widely applied solutions for Wahba's problem [25]. Wahba's problem tries to find a rotation matrix $\boldsymbol{A}$ between two reference frames from a set of weighted vector observations. To find this rotation matrix Wahba's problem seeks to minimize the cost function stated in Eq. (3-45).

$$J(A) = \frac{1}{2}\sum_{i=1}^{N} a_i \| \boldsymbol{b}_i - \boldsymbol{A}\boldsymbol{r}_i \|^2 \tag{3-45}$$

Where $\vec{\boldsymbol{b}}_i$ is a set of N unit vectors measured in the body reference frame of the satellite, $\vec{\boldsymbol{r}}_i$ the N corresponding unit vectors in a reference frame (e.g. the ECI reference frame), a_i the non negative weights and $\boldsymbol{A}$ the rotation matrix we try to find.
Earlier solutions [26] to Wahba's problem already showed that the cost function from Eq. (3-45) can also be written as Eq. (3-46).

$$J(A) = \frac{1}{2}\sum_{i=1}^{N} a_i - trace[\boldsymbol{A}\boldsymbol{B}^{\boldsymbol{T}}] \tag{3-46}$$

Where $\boldsymbol{B}$ is generally referred to as the attitude profile matrix and is defined by Eq. (3-47).

$$\boldsymbol{B} = \sum_{i=1}^{N} a_i \boldsymbol{b}_i \boldsymbol{r}_i^T \tag{3-47}$$

From Eq. (3-46) it is easy to see that when the $trace[\boldsymbol{A}\boldsymbol{B}^{\boldsymbol{T}}]$ is maximized the cost function is minimized.
Davenport's q-method tries to maximize $trace[\boldsymbol{A}\boldsymbol{B}^{\boldsymbol{T}}]$ by parametrizing the rotation matrix $\boldsymbol{A}$ as a unit quaternion, given in Eq. (3-19). Since Eq. (3-19) is a homogeneous quadratic function it can be rewritten as Eq. (3-48) [27].

$$trace[\boldsymbol{A}\boldsymbol{B}^{\boldsymbol{T}}] = \boldsymbol{q}\boldsymbol{K}\boldsymbol{q}^T \tag{3-48}$$

Where the 4 by 4 matrix $\boldsymbol{K}$ is given by Eq. (3-49)

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{S} - \boldsymbol{I}_3 trace(\boldsymbol{B}) & \boldsymbol{z} \\ \boldsymbol{z}^T & trace(\boldsymbol{B}) \end{bmatrix} \tag{3-49}$$

with $\boldsymbol{S}$ and $\boldsymbol{z}$ defined as Eq. (3-50).

$$\boldsymbol{S} = \boldsymbol{B} + \boldsymbol{B}^T \quad \text{and} \quad \boldsymbol{z} = \begin{bmatrix} B_{23} - B_{32} \\ B_{31} - B_{13} \\ B_{12} - B_{21} \end{bmatrix} \tag{3-50}$$

From Eq. (3-48) it was shown that the optimal quaternion representing the attitude of the satellite is the quaternion that maximizes the right side of this equation. In [26] it is stated that solving Eq. (3-48) for the optimal quaternion is equal to determining the normalized eigenvector of the matrix $\boldsymbol{K}$ with the largest eigenvalue. Thus, determining the attitude of the satellite now reduces to calculating the eigenvector of a 4 by 4 matrix, for which very robust algorithms exist [28].

3-4-2-2 Kalman filtering

Discrete Kalman filter The Kalman Filter (KF), introduced by R. E. Kalman in 1960, is the most widely used method to determine the attitude of a satellite from multiple sensor measurements and their measurement history. The discrete KF addresses the problem of trying to estimate the state $\boldsymbol{x} \in \mathbb{R}^n$ of a discrete-time controlled process given by a linear stochastic difference equation as shown in Eq. (3-51)

$$\boldsymbol{x}_k = \boldsymbol{A}_k \boldsymbol{x}_{k-1} + \boldsymbol{B}_k \boldsymbol{u}_{k-1} + \boldsymbol{w}_{k-1} \tag{3-51}$$

And with a measurement $\boldsymbol{z} \in \mathbb{R}^m$ as shown in Eq. (3-52)

$$\boldsymbol{z}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k \tag{3-52}$$

The matrix $\boldsymbol{A}_k$ is a n by n matrix which relates the state of the process at a previous time step $k-1$ to the state at the current step k. The n by l matrix $\boldsymbol{B}_k$ relates the optional control input $\boldsymbol{u} \in \mathbb{R}^n$ at time step $k-1$ to the state at time step k. The m by n matrix $\boldsymbol{H}_k$ relates the current state at time step k to the measurement at time step k. The random variables $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ represent the process noise and are assumed to be independent from each other, normally distributed and have a zero mean.

$$p(\boldsymbol{w}) \sim N(0, \boldsymbol{Q}) \tag{3-53}$$

$$p(\boldsymbol{v}) \sim N(0, \boldsymbol{R}) \tag{3-54}$$

They are also often referred to as the process noise covariance $\boldsymbol{Q}$ and the measurement noise covariance $\boldsymbol{R}$ matrices.
The KF estimates the process, given by Equations (3-51) and (3-52) with a form of feedback control. It estimates the state $\boldsymbol{x}_k$ of the process at some time k and then gets feedback in the form of noisy measurements. The equations for the KF can be divided into two groups. The 'time update' equations give an a priori estimate of the state $\hat{\boldsymbol{x}}_k^-$ and the error covariance $\boldsymbol{P}_k^-$, which definition is shown in Eq. (3-55).

$$\boldsymbol{P}_k^- \triangleq E[\boldsymbol{e}_k^- \boldsymbol{e}_k^{-T}] \quad \text{with} \quad \boldsymbol{e}_k^- = \boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^- \tag{3-55}$$

The 'measurement update' equations incorporate a new measurement in the a priori estimate to give a posteriori estimate $\hat{\boldsymbol{x}}_k$. Eq. (3-56) and Eq. (3-57) show the equations for the 'time update' stage. The derivation of these equations are not further discussed here, but can be found in [2].

$$\hat{\boldsymbol{x}}_k^- = \boldsymbol{A}_k \hat{\boldsymbol{x}}_{k-1} + \boldsymbol{B}_k \boldsymbol{u}_{k-1} \tag{3-56}$$

$$\boldsymbol{P}_k^- = \boldsymbol{A}_k \boldsymbol{P}_{k-1} \boldsymbol{A}_k^T + \boldsymbol{Q} \tag{3-57}$$

The first 'time update' equation shows how the a priori estimate, $\hat{\boldsymbol{x}}_k^-$ is related to the a posteriori estimate, $\hat{\boldsymbol{x}}_{k-1}$, from the previous time step through the matrix $\boldsymbol{A}_k$. The matrix $\boldsymbol{A}_k$

incorporates the the linear dynamics/kinematics of the process the KF tries to estimate. The matrix $\boldsymbol{B}_{k-1}$ and the vector $\boldsymbol{u}$ are the optional control input of the process. The other equation shows how the a priori error covariance, $\boldsymbol{P}_k^-$, is related the a posteriori error covariance, $\boldsymbol{P}_{k-1}$, from the previous time step through the matrix $\boldsymbol{A}_k$. Where $\boldsymbol{P}_{k-1}$ is defined as shown in Eq. (3-58). The matrix $\boldsymbol{Q}$ is the process noise covariance matrix from Eq. (3-53).

$$\boldsymbol{P}_{k-1} \triangleq E[\boldsymbol{e}_{k-1}\boldsymbol{e}_{k-1}{}^T] \quad \text{with} \quad \boldsymbol{e}_{k-1} = \boldsymbol{x}_{k-1} - \hat{\boldsymbol{x}}_{k-1} \tag{3-58}$$

The 'measurement update' equations are given by Equations (3-59), (3-60) and (3-61). Derivations of these equations can be found in [2].

$$\boldsymbol{K}_k = \boldsymbol{P}_k^- \boldsymbol{H}_k^T (\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{R})^{-1} \tag{3-59}$$

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k(\boldsymbol{z}_k - \boldsymbol{H}_k \hat{\boldsymbol{x}}_k^-) \tag{3-60}$$

$$\boldsymbol{P}_k = (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k)\boldsymbol{P}_k^- (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k)^T + \boldsymbol{K}_k \boldsymbol{R} \boldsymbol{K}_k^T \tag{3-61}$$

The first equation shows the calculation of the gain or blending factor $\boldsymbol{K}_k$. This gain basically determines if the actual measurement $\boldsymbol{z}_k$ or the predicted measurement $\boldsymbol{H}_k \hat{\boldsymbol{x}}_k^-$ is trusted more. In this equation the matrix $\boldsymbol{R}$ is the measurement noise covariance matrix from Eq. (3-54). The second equation shows how the current state $\hat{\boldsymbol{x}}_k$ at time step k is updated using the a priori estimate $\hat{\boldsymbol{x}}_k^-$ and the 'weighted' measurement $\boldsymbol{z}_k$. The last equation shows the update of the error covariance matrix $\boldsymbol{P}_k$.
The 'time update' and 'measurement update' process is repeated continuously, using the previous a posteriori estimate and error covariance to predict a new a priori estimate and error covariance. Figure 3-3 shows the 'time update' and 'measurement update' process.

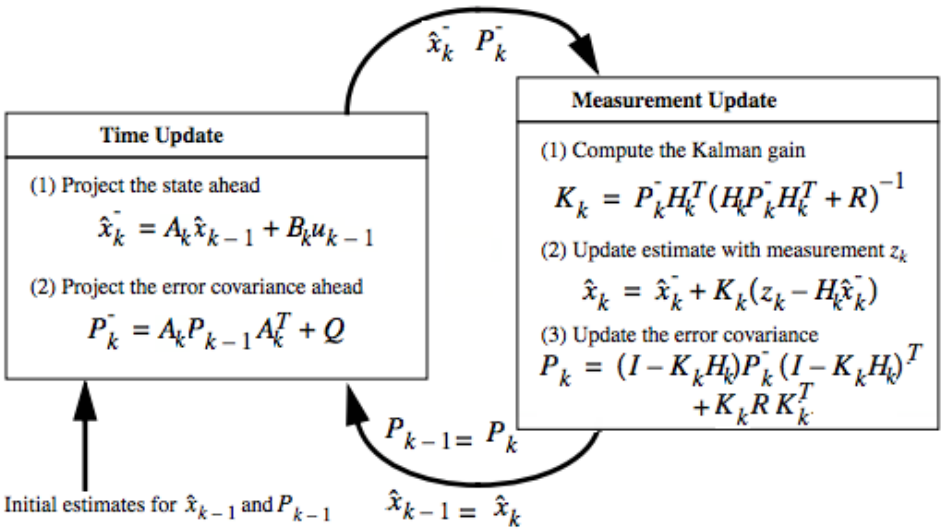


Figure 3-3: The 'time update' and 'measurement update' process of the Kalman filter [2].

Discrete Extended Kalman filter The equations given in the previous section describe a method to estimate the state of discrete-time controlled process represented by a set of linear stochastic differential equations. However these equations do not apply when we need to estimate a non linear discrete-time controlled process, such as depicted in Eq. (3-62) and Eq. (3-63).

$$\boldsymbol{x}_k = f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}, \boldsymbol{w}_{k-1}) \tag{3-62}$$

$$\boldsymbol{z}_k = h(\boldsymbol{x}_k, \boldsymbol{v}_k) \tag{3-63}$$

Where $\boldsymbol{x}_k \in \mathbb{R}^n$ represents the non linear process as a function of the state at a previous time step $k-1$, the control input $\boldsymbol{u}_k \in \mathbb{R}^n$ and the process noise $\boldsymbol{w}_{k-1}$. The function $\boldsymbol{z}_k$ represents the non linear measurement as a function of the state $\boldsymbol{x}_k$ and the measurement noise $\boldsymbol{v}_k$.
A non linear discrete-time controlled process, such as described by Eq. (3-62) and Eq. (3-63), should be estimated using an Extended Kalman Filter (EKF). The EKF linearises the process and measurement functions by taking the partial derivatives of these functions. These partial derivatives are defined by the equations Eq. (3-64).

$$\boldsymbol{A}_{[i,j]} = \frac{\partial f_{[i]}}{\partial \boldsymbol{x}_{[j]}}(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k, 0) \qquad \boldsymbol{W}_{[i,j]} = \frac{\partial f_{[i]}}{\partial \boldsymbol{w}_{[j]}}(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k, 0) \tag{3-64}$$

$$\boldsymbol{H}_{[i,j]} = \frac{\partial h_{[i]}}{\partial \boldsymbol{x}_{[j]}}(\hat{\boldsymbol{x}}_k, 0) \qquad \boldsymbol{V}_{[i,j]} = \frac{\partial h_{[i]}}{\partial \boldsymbol{v}_{[j]}}(\hat{\boldsymbol{x}}_k, 0) \tag{3-65}$$

Where $\boldsymbol{A}$ is the Jacobian matrix [18] with the partial derivatives of $f(..)$ with respect to $\boldsymbol{x}$, $\boldsymbol{W}$ the Jacobian matrix with the partial derivatives of $f(..)$ with respect to $\boldsymbol{w}$, $\boldsymbol{H}$ the Jacobian matrix with the partial derivatives of $h(..)$ with respect to $\boldsymbol{x}$ and $\boldsymbol{V}$ the Jacobian matrix with the partial derivatives of $h(..)$ with respect to $\boldsymbol{v}$.
With these four Jacobian matrices a same set of 'time update' and 'measurement update' equations can be constructed. Equations 3-66 and 3-67 show the 'time update' equations for the EKF and Equations 3-68, 3-69 and 3-70 the 'measurement update' equations for the EKF.

$$\hat{\boldsymbol{x}}_k^- = f(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_{k-1}, 0) \tag{3-66}$$

$$\boldsymbol{P}_k^- = \boldsymbol{A}_k \boldsymbol{P}_{k-1} \boldsymbol{A}_k^T + \boldsymbol{W}_k \boldsymbol{Q}_{k-1} \boldsymbol{W}_k^T \tag{3-67}$$

$$\boldsymbol{K}_k = \boldsymbol{P}_k^- \boldsymbol{H}_k^T (\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{V}_k \boldsymbol{R}_k \boldsymbol{V}_k^T)^{-1} \tag{3-68}$$

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k (\boldsymbol{z}_k - h(\hat{\boldsymbol{x}}_k, 0)) \tag{3-69}$$

$$\boldsymbol{P}_k = (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \boldsymbol{P}_k^- (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k)^T + \boldsymbol{K}_k \boldsymbol{R}_k \boldsymbol{K}_k^T \tag{3-70}$$

These 'time update' and 'measurement update' equations for the EKF can also be depicted as the same schematic shown Figure 3-3.
Besides the EKF there are a number of other variants of the KF, each having their own set of 'time update' and 'measurement update' equations. A couple examples are the Additive Extended Kalman Filter (AEKF), Multiplicative Extende Kalman Filter (MEKF), the Quaternion Kalman Filter (QKF) and the Quaternion Extende Kalman Filter (QEKF) [29].

3-5 Attitude control

The attitude of a satellite can be controlled passively or actively. Passive control is much simpler and inexpensive but gives you lesser accuracy and control over the satellite than active control. Passive control can for example be achieved by placing a permanent magnet inside the satellite. The permanent magnet orients the satellite along the earth magnetic field. This is a very robust and simple way, but gives most of the time not the desired attitude of the satellite and the permanent magnet can disturb the magnetometer measurements. Another example of passive control is Gravity-gradient stabilization. Gravity gradient stabilization uses the distribution of the satellite's body mass and the earth gravitational field to get a desired attitude. Due to the earth gravitational field the point in the satellite with more mass will be attracted harder to earth then the point with less mass. Clever use of the mass distribution makes it therefore possible to orient the satellite in a desired attitude.
For more control and higher accuracy active control is necessary. Active control is more expensive and harder to implement, but most of the time needed to achieve certain mission objectives. Active control can be done with a variety of actuators and algorithms. In this section a few actuators and control algorithms are discussed.

3-5-1 Actuators

Magnetic torquers Magnetic torquers or short magnetorquers are often used in satellites for three-axes stabilization and angular momentum control. Magnetorquers are essentially winded electromagnetic coils with or without a permeable metal core. Sending a current through the coil will create a magnetic field perpendicular to the current loop. This magnetic field will interact with the earth magnetic field. Due to this interaction a force, called torque, is created which acts on the satellite and will make it rotate. The advantages of magnetorquers are that they are relatively cheap, easy to make and have no moving parts, which fail more often. The disadvantage however is that they are rather coarse and that always one axis is uncontrollable.

Reaction wheels Reaction wheels are essentially just wheels connected to an electro motor. The wheel has a relatively large mass, so when the wheel accelerates it will apply a force to the whole satellite making the satellite rotate slightly. Reaction wheels can make very accurate adjustments, but may build up a significant momentum during the course of operation. Therefore they are often used with other actuators, such as described above, to dump momentum from the reaction wheels. Disadvantages of reaction wheels are that they have moving parts which can easily fail and have a high power consumption.

Momentum wheels Momentum wheels are basically a special kind of reaction wheel. Spinning a reaction wheel up to a relatively high rate and leaving it at that speed will provide gyric stiffness in the axis of the reaction wheel. Due to the gyric stiffness the satellite is able to resist small external torques and maintain the pointing direction of the satellite.

Thrusters Thrusters are propulsive devices that shoot out mass in one direction, thus creating a force in the opposite direction. With thrusters it is possible to keep the satellite

stationary or to change the orbit of the satellite. Thrusters can also be used for attitude control if six thrusters are used for the postive and negative x,y and z directions. The main disadvantages of thrusters are the limitations on fuel and the cycles of the control valves.

3-5-2 Control algorithms

In this subsection two different control algorithms are described. The first algorithm is solely based on changes in magnetic field and pure magnetic control to detumble the satellite, while the second one is based on errors in rotational rate and attitude of the satellite and one or more actuators to orient the satellite.

3-5-2-1 Bdot controller

When the satellite is ejected from its launch container into orbit, the first and most important task for attitude control is to stabilize the rotational rate (i.e. detumbling). Detumbling should be done by a robust and fail safe system, which is not dependable on very complex systems. A robust and fail safe way to detumble the satellite is by using a Bdot controller. The Bdot controller uses only magnetic field measurements and magnetic actuation, in the form of magnetorquers, to detumble the satellite.
The principle behind a Bdot controller is actually very simple. A magnetometer on board the satellite measures the magnetic field vector in the body frame of the satellite. While the satellite orbits the Earth, the measured magnetic field vector changes, due to variations in magnetic field strength of the earth. However the magnetic field changes caused by tumbling of the satellite is much more dominant, because the tumbling rate of the satellite can be much larger than the orbital rate. These changes in magnetic field, caused by tumbling of the satellite, are called the Bdot and is basically the time derivative magnetic field measurements. Minimizing the Bdot is minimizing the changes in measured magnetic field and thus minimizing the tumbling rate of the satellite.
A control law that minimizes the Bdot is the Bdot control law as proposed in [30] and stated in Eq. (3-71).

$$\boldsymbol{m}_{Bdot} = -k_B \boldsymbol{Bdot} \tag{3-71}$$

Where $\boldsymbol{m}_{Bdot}$ is the magnetic dipole vector that must be generated by the magnetorquers in order to counteract the tumbling and k_B a controller gain dependent on physical properties of the magnetorquers and expected rotational rates of the satellite [3]. The controller gain is negative in order to actuate in the opposite direction of the rotation and thus taking kinetic energy from the system.
The torque acting on the satellite because of the applied magnetic dipole can be determined with Eq. (3-72).

$$\boldsymbol{T}_{control} = \boldsymbol{m}_{Bdot} \times \boldsymbol{B}_b \tag{3-72}$$

Where $\boldsymbol{B}_b$ is the magnetic field vector in the body reference frame. In [3] a more detailed description is given on how this control torque influences the rotation of the satellite.

3-5-2-2 Quaternion feedback regulator

The Quaternion feedback regulator, as described in [3] and [31], is another control algorithm to calculate the required torque to control the satellite. The control law of the Quaternion feedback regulator is formulated as follows.

$$\boldsymbol{T}_{control} = -d\boldsymbol{J}\boldsymbol{\omega}_e - k\boldsymbol{J}\boldsymbol{q}_e + \hat{\boldsymbol{\Omega}}\boldsymbol{I}\hat{\boldsymbol{\omega}} \tag{3-73}$$

Where d and k are gain parameters, $\boldsymbol{J}$ the moment of inertia matrix of the satellite, $\vec{\boldsymbol{\omega}}_e$ the error between the desired rotational and the estimated rotational rate vector, $\vec{\boldsymbol{q}}_e$ the vector part of the quaternion that describes the error between the desired and the estimated attitude quaternion, $\tilde{\boldsymbol{\omega}}$ the estimated rotational rates of the satellite and $\hat{\boldsymbol{\Omega}}$ a skew symmetric matrix [32] constructed from the estimated rotational rates. As stated in [3] and [33] the term $\hat{\boldsymbol{\Omega}}\boldsymbol{I}\tilde{\boldsymbol{\omega}}$, which is for counter acting gyroscopic coupling torque, can be discarded because this only adds computational complexity without providing much more control accuracy. Leaving this term out simplifies Eq. (3-73) to Eq. (3-74).

$$\boldsymbol{T}_{control} = -d\boldsymbol{J}\boldsymbol{\omega}_e - k\boldsymbol{J}\boldsymbol{q}_e \tag{3-74}$$

The gains d and k determine the settling time and the damping of the control algorithm. More on the calculations of the values for these gain parameters can be found in [3].
If more than one type of actuator is present on the satellite, the total amount of control torque needed can be divided over the different actuators. This is called load dividing. For Delfi-n3Xt the 'geometric division' load dividing scheme is chosen [3],[34], which divides the control torque between the magnetorquers and the reaction wheels, according to Eq. (3-75).

$$\boldsymbol{T}_{control} = \boldsymbol{T}_{rw} + \boldsymbol{T}_{mtq} = -\left\{\frac{d\boldsymbol{h}}{dt}\right\} + \boldsymbol{m}_{Bdot} \times \boldsymbol{B}_b \tag{3-75}$$

Controlling the satellite with reaction wheels also requires a method for unloading the reaction wheels (i.e. dumping angular momentum stored in the reaction wheels). When the reaction wheels are accelerated or decelerated, to deliver the required torque, angular momentum is stored in the reaction wheels. Since each reaction wheel can only store a maximum amount of angular momentum the reaction wheels should be unloaded on a regular basis. In [3] several unloading algorithms have been described and investigated.

Chapter 4

Delfi-n3Xt's Attitude Determination and Control Subsystem: Hardware

The Attitude Determination and Control Subsytem (ADCS) of the Delfi-n3Xt consists out of two magnetometers, six sun sensors, three magnetorquers, three reaction wheels and two Micro Controller Units (MCUs). These components are interconnected to each other as shown in the block diagram in Figure 4-1. From the block diagram one can see that one MCU, the primary MCU, is connected to the full set of sensors and actuators, while the secondary MCU, is only connected to a magnetometer and the magnetorquers. In normal operation the primary MCU will be switched on and perform detumbling, sun pointing, thruster pointing or ground station tracking, depending on which mode its in. The different modes and details of the software will be described in more detail in Chapter 6. If a sensor or an actuator connected to the primary MCU fails and causes the satellite to behave incorrectly, the primary MCU can be shut down and the secondary can be switched on and serve as a backup, only with reduced functionality. Switching the MCUs on or off is done by the Delfi Standard System Bus (DSSB) circuitry, which is part of de CDHS.

In this chapter a description of all the hardware components of the ADCS is given. Section 4-1 and Section 4-2 describes the used sensors and actuators in more detail and Section 4-3 gives a short overview of the two MCUs.

4-1 Sensors

For the attitude determination of the Delfi-n3Xt two types of sensors are used. Magnetometers to measure the earth magnetic field and sun sensors to determine the direction of the sun. A description of both is given below. Initially an angular rate sensor was also part of the ADCS design. However, tests with the angular rate sensor showed unacceptable performance degradation, due to micro vibrations produced by the reaction wheels [35].

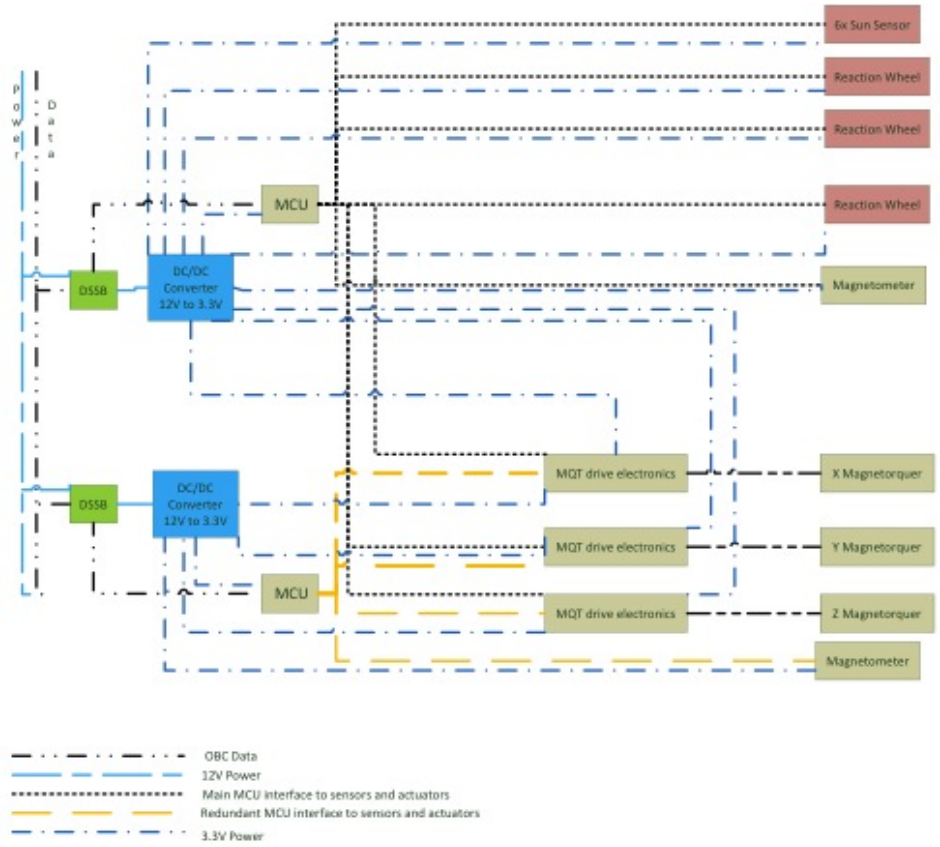


Figure 4-1: Overview of the ADCS subsystem.

4-1-1 Magnetometers

The magnetometers used are the HMC5883L from Honeywell [36]. Its a three axis magnetometer which can measure magnetic fields from +/- 1 milli Gauss to +/- 8 Gauss. Communication with the magnetometer is done through the Two Wire Interface (TWI). The measurement range can be set by adjusting the gain setting of the magnetometer.
Furthermore the magnetometer can be set in a continuous measurement mode, single measurement mode or in idle mode. In continuous measurement mode the magnetometer measures the magnetic field with a configurable data output rate up to 75Hz. In single measurement mode the magnetometer only measures the magnetic field when requested. When put idle mode the major power sources are disabled, but the device remains accessible through TWI. The measurement data for each axis is stored in two registers and is a 16 bits 2's complement value ranging from 0xF800 to 0x07FF (-2048 to 2047). If the measured field exceeds this range the ADC will over or underflow and the two registers will contain the value -4096. This value clears when a next valid measurement is done. The value in the data register are in counts and can be converted to Gauss with Eq. (4-1).

$$Magnetic\ field\ in\ Gauss = \frac{Counts}{Gain} \tag{4-1}$$

The magnetometer also features a scale factor calibration, which can be used to scale the axis

sensitivities. More details on that can be found in [36].
The operating temperature of the magnetometer, as stated in the datasheet, is -30 degrees up to 85 degrees, which falls in the calculated operating temperature range of 1.0 to 27.9 degree Celsius of the ADCS electronics [37].

4-1-2 Sun sensors

None of the commercially available sun sensors were suitable for the Delfi-n3Xt mission, because they were either to large or to expensive [38]. Therefore the sun sensors used for the Delfi-n3Xt are developed in-house. In [38] a trade off has been made between different types of sun sensors and the type of sun sensor chosen is a Quadrant Sun Sensor (QSS). The QSS consists of Hamamatsu S5981 quadrant photo diode [39]. A housing with a well-defined square window, the size of one quadrant, is placed over the quadrant photodiode. The housing is made out of graphite filled PEEK to reduce effects of internal reflections.

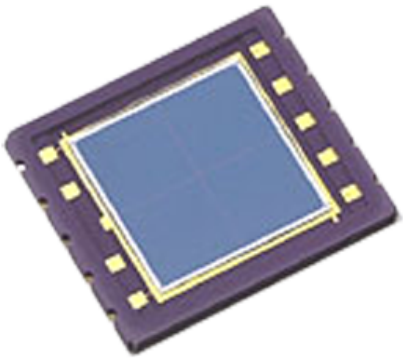


Figure 4-2: The Hamamatsu S5981 quadrant photodiode.

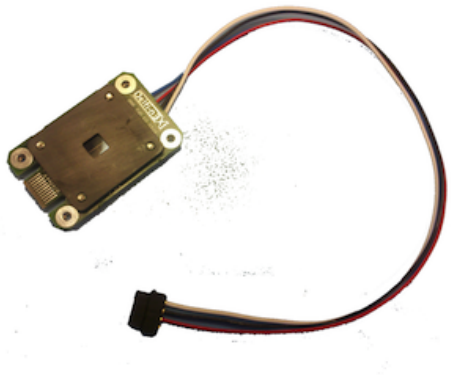


Figure 4-3: The complete sun sensor PCB.

The Printed Circuit Board (PCB) with the electronic read-out circuitry consists of an transimpedance amplifier which amplifies the outputs of the quadrant photo diode to a voltage range between 0V and 1.8V. The output of the amplifier is connected to the Analogue to Digital Converter (ADC) of an Atmega88PA micro controller. The ADC converts the input voltage with a range between 0V and 2,048V to a 16 bits value, which is stored in a 'register'. All communication with the sun sensor is done via the TWI interface.
Furthermore the sun sensor PCB also incorporates a LM75B temperature sensor, from NXP Semiconductors, to monitor the temperature of the sun sensor. The temperature sun sensor can also be addressed via the TWI interface.
For a more detailed description of the sun sensor design one is referred to [38].

4-2 Actuators

Attitude control on the Delfi-n3Xt is done with two types of actuators, namely three magnetorquers and three reaction wheels. A description of both is given below.

4-2-1 Magnetorquers

For each axis (X,Y and Z) of the satellite there is one magnetorquer. The magnetorquer for the X- and Y axis are respectively a mu-metal rod of 74mm and 82mm with a diameter of 4mm. Six layers of copper wire, with a diameter of 0,15mm, are wounded around the rods. The magnetorquer for the Z-axis is different from the X- and Y-axis. Its a so called 'air-coil' type. Copper wire is wounded around the structure of the magnetorquer assembly, as can be seen in Figure 4-4.

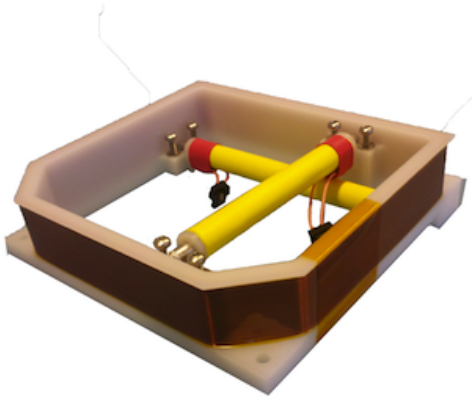


Figure 4-4: The complete magnetorquer assembly.

All three magnetorquers are driven by the magnetorquer drive. The magnetorquer drive has six output pins, two for each magnetorquer, to drive the magnetorquers. The output pins, when set high, output a voltage of 1.2V, making a current flow through the magnetorquer. Depending on which of the two pins is set high, current flows in the positive or negative direction through the magnetorquer creating a positive or negative magnetic dipole in that axis. Inputs for the magnetorquer drive comes from one of the two Micro Controller Unit (MCU)s. The inputs are processed by three AT90PWM3B micro controllers, one for each axis, and determine which output pin should be set high.

4-2-2 Reaction wheels

For every axis of the satellite there is also one reaction wheel. Each reaction wheel consist out of a PCB with a MCU and some other electrical components, a Faulhaber 1202 004 BH BLDC motor, and a flywheel. The MCU chosen for the reaction wheel is a ATmega164P. The ATmega164P is used to control the motors, do speed measurements and for communication with the ADCS main board. Communication with the ADCS is done via the TWI interface.

Controlling the motor of the reaction wheel is done with the electric coils in the motor and a set of MOSFET's to apply power to the electric coils. With information from the Hall sensors in the motor and input from the ATmega164P switching between different MOSFET configurations is done, such that power is applied to the electric coils in the right order to make the motor turn in the right direction. Information from the HAL sensors is also used to determine the rotation speed of the flywheel.
The flywheel is made from solid bronze and is disk-shaped. It has a diameter of 20mm and has a total weight of 6 grams. The motor is able to accelerate the flywheel to around 25000 rpm clockwise and counter-clockwise.
All three reaction wheels are placed in a specially designed aluminium bracket. Figure 4-5 shows the three reaction wheels placed in this aluminium bracket.

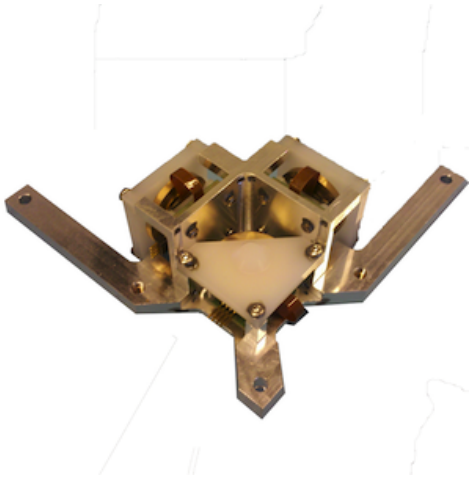


Figure 4-5: The thee reaction wheels in their complete assembly.

4-3 Micro Controller Units

As already mentioned at the beginning of this chapter, the ADCS has a primary and a secondary controller. These controllers are the 'brain' of the ADCS. A brief description of both controllers is given below. The rest of this thesis will apply to the ICnova SAM9G45 SO-DIMM, because this MCU will run the complete ADCS software, while the ATxmega128A1 will only run a subset.

4-3-1 ICnova SAM9G45 SO-DIMM

The primary controller of the ADCS is the ICnova SAM9G45 SO-DIMM module. The module has the form factor of a 200 pins SO-DIMM memory module with dimensions 32.6mm by 72.2mm. Due the form factor the module can be easily plugged into a connector on the ADCS main board and is easily interchangeable if a defect occurs during the development and testing. The ICnova SAM9G45 SO-DIMM module is shown in Figure 4-6.

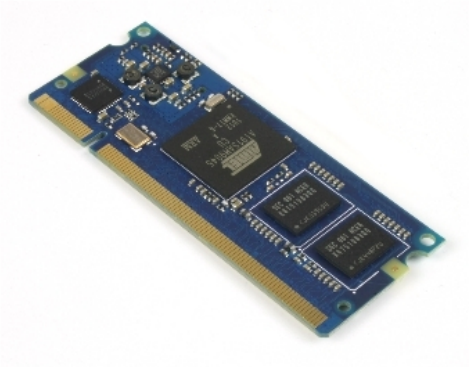


Figure 4-6: The ICnova SAM9G45 SO-DIMM module.

The ICnova SAM9G45 SO-DIMM module houses an AT91SAM9G45 micro processor, 128MB of DDR2 RAM, 8MB of NOR-Flash and 256 of NAND-Flash. The AT91SAM9G45 is an ARM9 based micro processor from Atmel with a maximum clock frequency of 400Mhz. Furthermore it offers 64KB of internal SRAM memory and a wide range of connectivity and user peripherals. More details on the ICnova SAM9G45 SO-DIMM module is given in Chapter 5.

4-3-2 ATxmega128A1

The ATxmega128A1 is a low power and high performance micro controller from Atmel with a maximum clock frequency of 32MHz. The ATxmega128A1 features 128KB of flash program memory, 8KB boot section memory, 8KB of SRAM and 2KB of EEPROM. Furthermore the micro controller also offers a wide range of user peripherals and therefore can be used for a large variety of applications. For more details on the ATxmega128A1 one is referred to the datasheet [40].

Chapter 5

ICnova SAM9G45 SO-DIMM module

As already described in Subsection 4-3-1 the ICnova SAM9G45 SO-DIMM module consist of a micro processor, the AT91SAM9G45, and three types of external memories: DDR2 RAM, NAND-Flash and NOR-Flash. In this chapter a more detailed description of the micro processor with its peripherals and the various external memories is given. It is assumed that the reader has some basic knowledge about micro controllers or processors and some other related subjects.

5-1 AT91SAM9G45

The AT91SAM9G45 is a 400Mhz ARM9 based micro processor that offers a large variety of connectivity and user peripherals. In this section a more detailed description of a couple of these peripherals is given.

5-1-1 Peripherals

5-1-1-1 Power Management Controller

The Power Management Controller (PMC) provides all the clock signals to the system. To generate these clock signals the PMC has three main clock controllers, the Master Clock Controller (MCC), the Processor Clock Controller (PCC) and the Peripheral Clock Controller (PRCC). This paragraph describes these three clock controllers in more detail. In Figure 5-1 one can see a schematic overview of the clock controllers. The schematic overview also shows a fourth clock controller, the Programmable Clock Controller, but this one is not discussed, because it is not used in the Attitude Determination and Control Subsytem (ADCS) software.

Master Clock Controller The Master Clock Controller (MCC) generates the Master Clock (MCK), the clock signal for the memory controller, the input clock signal for the PCC and the

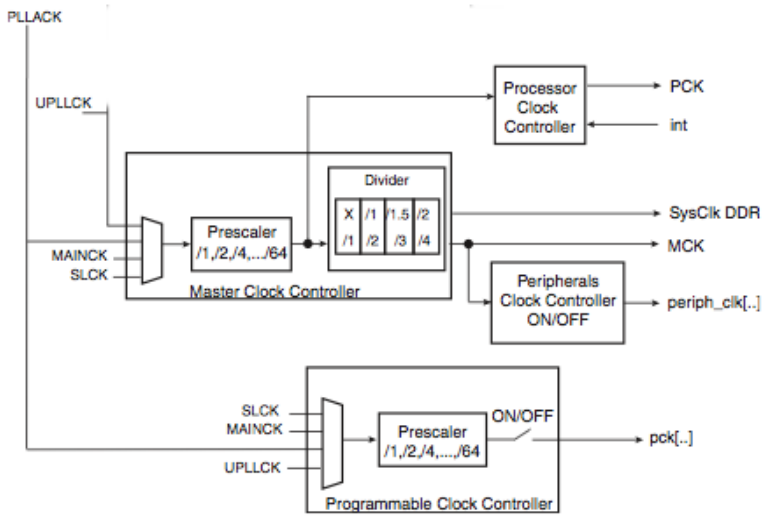


Figure 5-1: Schematic overview of the four clock controllers.

input clock signal for the PRCC. It consists of a clock selector, a pre-scaler and a divider to generate different clock signals. The MCK is selected from one of the four input clock signals, which can be seen in Figure 5-2. Selection of the input clock signals is done by writing the Clock Source Selection (CSS) bits in the PMC Master Clock Register.

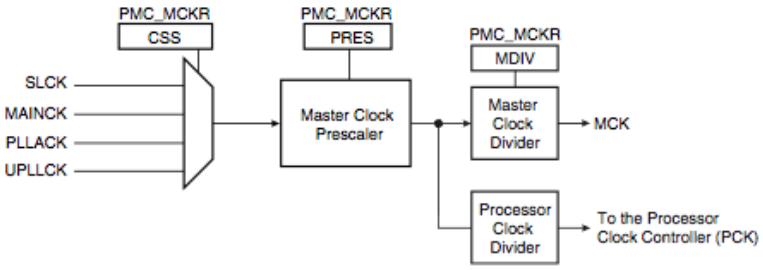


Figure 5-2: Schematic overview of the Master Clock Controller.

After an input clock signal for the MCK is selected it can be pre-scaled. Pre-scaling is done by writing the Pre-scaler (PRES) bits in the PMC Master Clock Register. The input clock signal can be pre-scaled with a power of two between 1 and 64. The pre-scaled clock signal serves as an input for the PCC.
After pre-scaling the MCK can be divided to create a MCK that is lower than the Processor Clock (PCK). Division of the MCK can be done by writing the Division (DIV) bits in the PMC Master Clock Register. A maximum division of 4 can be done, which results in a MCK of PCK/4. The resulting MCK serves as an input to the PRCC and other parts of the system. The input for the memory controller is two times the resulting MCK.

Processor Clock Controller The Processor Clock Controller (PCC) provides the Processor Clock (PCK) to the processor. The PCC can put the processor in idle mode by disabling the PCK. Disabling the PCK is done by writing the PMC System Clock Disable Register. If the PCK is disabled it can be re-enabled manually or automatically by any enabled interrupt. After a reset the PCK is enabled by default.

Peripheral Clock Controller The Peripheral Clock Controller (PRCC) controls all the clock signals to the peripherals. Each clock signal to a peripheral can be enabled and disabled separately by setting the bit, corresponding to that peripheral, in the PMC Peripheral Clock Control Register to 1 or 0. Setting a bit in this register will automatically set the same bit in the PMC Peripheral Clock Status Register to 1 or 0. This way one can check whether the peripheral clock is already enabled by reading the PMC Peripheral Clock Status Register.

5-1-1-2 Two Wire Interface

The Two Wire Interface (TWI) peripheral is a unique two-wire bus made up of one clock line (TWCK) and one data line (TWD) and can be used to communicate with any Atmel TWI interface or I2C compatible device. Figure 5-3 shows a simple schematic of the two-wire bus.

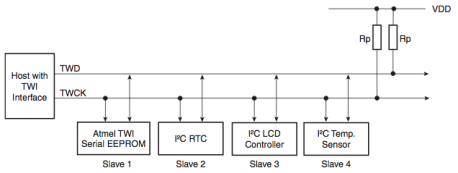


Figure 5-3: Simple schematic overview of the two-wire bus.

The two-wire bus supports a data output rate up to 400Kbits per second on the data line and due to the configurable baud generator the data output rate can be adapted to a wide range of different clock frequencies of the processor. The TWI peripheral clock is controlled by the PRCC and thus the peripheral clock must first be enabled before the TWI peripheral can be used.
On the AT91SAM9G45 there are two TWI peripherals available which can either be configured in Master or in Slave mode. The TWI peripheral also supports Multi-Master mode, but this mode is not used in the ADCS software, so not further discussed. The following paragraphs gives a more detailed description of these two operational modes.

Master mode In master mode the device (i.e. the Master) always starts the transfer, generates the clock on the clock line and stops the transfer. To configure a TWI peripheral in master mode one has to disable slave mode and enable master mode by setting the SVDIS and the MSEN bits in the TWI Control Register. Further more the data output rate has to be set by writing the CLDIV, CHDIV and CKDIV fields with the correct values. The Master can either be in master transmitter mode (i.e transmitting bytes to a slave device) or master receiver mode (i.e requesting and receiving bytes from a slave device).

- **Master transmitter mode** To start a transfer in transmitter mode, first one has to configure the address of the slave device where the bytes should be transferred to. This is done by writing the 7-bit slave address into the DADR field in the TWI Master Mode Register. Secondly the MREAD bit in the TWI Master Mode Register has to be cleared, indicating that we want to transmit a number of bytes to a slave device. To generate a start condition, indicating the start of a transfer, the first byte that one wants to transmit should be written in the TWI Transmit Holding Register. This will automatically generate the start condition and puts the 7-bit slave address and the MREAD bit on the bus. The slave device should acknowledge the slave address by sending an Acknowledge (ACK). If the address is acknowledged the first byte from the TWI Transmit Holding Register is sent. Every byte sent should be acknowledged by the slave device. If the Master receives an ACK the TXRDY bit in the TWI Status Register is set and cleared again when writing the next byte to the TWI Transmit Holding Register. When the slave device does not acknowledge a sent byte, the Not Acknowledge (NACK) bit in the TWI Status Register is set and the master should terminate the transfer by sending a stop condition. Sending a stop condition is done by setting the STOP bit in the TWI Control Register. When all bytes are sent and acknowledged by the slave device the master should also terminate the transfer by sending a STOP condition and setting the TXCOMP in the TWI Status Register.
- **Master receiver mode** In master receiver mode the DADR field and the MREAD bit in the TWI Master Mode Register should also be set first. The MREAD bit should be set, indicating that we want to receive bytes from a slave device. To start a transfer the START bit in the TWI Control Register must be set, which automatically generates a START condition on the bus. After the START condition is sent the 7-bit slave address and the MREAD bit are sent. The slave device should acknowledge the slave address by sending an ACK. As soon as the Master receives the acknowledge from the slave device it is ready to receive a number of data bytes. If no acknowledge is received the NACK bit in the TWI Status Register is set. With every byte received the Master should send an acknowledge to the slave device indicating that the byte has been received and another should be sent. If a byte is received by the Master also the RXRDY bit in the TWI Status Register is set which indicates the byte can be read from the TWI Receive Holding Register. If the TWI Receive Holding Register is read the RXRDY bit is reset. As long as the Master keeps acknowledging bytes the Slave devices keeps sending bytes. If the Master has received its last byte it should send a STOP condition by setting the STOP bit in the TWI Control Register. This will automatically send a NACK and a STOP condition to the slave device indicating the end of transfer. After the NACK and STOP condition have been sent the RXCOMP bit in the TWI Status Register could be set to generate an interrupt if the interrupt is enabled.

Slave mode In Slave mode the device (i.e. the Slave) never starts or stops a transfer. Also the clock on the clock line is generated by an other device that acts as the Master. In order the configure a TWI peripheral in slave mode one first has to set the 7-bit slave address on which the Master can access the device. Setting the slave address is done by writing the 7-bit address into the SADR field in the TWI Slave Mode Register. Furthermore one has to disable the master mode and enable the slave mode by setting the MSDIS and the SVEN bits in the TWI Control Register. The Slave can either be in slave transmitter mode (i.e

transmitting bytes to the master device) or slave receiver mode (i.e receiving bytes from the master device).

- **Slave transmitter mode** When the Slave detects a START condition on the bus and the 7-bits slave address that is sent by the Master matches the 7-bits slave address in the SADR field, the Slave sends an ACK to acknowledge his address. Furthermore the SVACC bit and the SVREAD bit in the TWI Status Register are set to indicate that a Master is accessing the Slave and wants to receive a number of bytes from the Slave. The SVACC bit remains high until a STOP condition is detected on which the bit is cleared and the EOSVACC and the TXCOMP bit in the TWI Status Register is set. Every byte written in the TWI Transmit Holding Register is sent to the Master. As soon as a byte is written in the TWI Transmit Holding Register the TXRDY bit in the TWI Status Register is cleared and set again as soon the byte is transferred. If the transferred byte is acknowledged the next byte is written in the TWI Transmit Holding Register. If the transferred byte is not acknowledged by the Master the NACK bit in the TWI Status Register is set.
- **Slave receiver mode** Equal to the slave transmitter mode the slave waits for a START condition on the bus and a match with the 7-bit slave address in the SADR field. When the 7-bit slave address is matched the Slave sends an ACK, sets the SVACC bit in the TWI Status Register and clears the SVREAD bit to indicate the slave needs to receive a number of bytes. Until a STOP condition is detected on the bus the Slave stores the received bytes in the TWI Receive Holding Register, clears the RXRDY bit in the TWI Status Register and sends an ACK. When the received byte is read from the TWI Receive Holding Register the RXRDY bit is set again. If a STOP condition is detected the SVACC bit is cleared and the EOSVACC and the TXCOMP bit in the TWI Status Register is set.
- **General Call** A general call is a special case of slave access. If a general call is detected (i.e a START condition followed by a 7-bits slave address of 0x00) the GACC bit in the TWI Status Register is set. Its up to the programmer what kind of action needs to be taken when a general call is detected.

TWI Interrupts The TWI interface has an interrupt line connected to interrupt controller to handle interrupts. Every time a bit in the TWI Status Register is set an interrupt is fired. The interrupt will only fire if the interrupt corresponding to that bit is enabled in the TWI Interrupt Enable Register. If the interrupt is not enabled nothing will happen. It is up to the programmer to take necessary actions when the interrupt fires.

5-1-1-3 Parallel Input/Ouput Controller

The AT91SAM9G45 features up to five Parallel Input Output (PIO) controllers, with each 32 fully programmable Input/Output (I/O) lines. Each individual line can be configured as a general purpose I/O line or may be assigned to a function of one of on-chip the peripherals. The PIO controllers are clocked via the PRCC and can thus be enabled and disabled. Configuring the PIO controllers do not require the clock to be enabled, but specific functions

of the PIO controller do. Every PIO controller also has an interrupt line connected to it to serve interrupts occurring on the I/O lines.

On-chip peripheral functions The peripheral functions that can be assigned to an I/O line is hardware defined, thus not every peripheral function can be assigned to every I/O line. For each I/O line there are only two peripheral functions to select from. To assign a peripheral function to an I/O line one has to clear the corresponding bit in the PIO Enable Register and set the corresponding bit in the PIO Disable register. Whether a I/O line is assigned to a peripheral function or configured as a general purpose I/O line can be read from the PIO Status Register. If an I/O line is assigned to a on-chip peripheral selecting between the two peripheral functions can be done by setting and clearing the the corresponding bits in the PIO A Select Register and PIO B Select Register. From the PIO AB Status Register can be read which of the two peripheral functions the line is currently assigned to.

General purpose Input/Output When an I/O line is configured as a general purpose I/O line, the line is controlled by the PIO controller. The I/O line can be either configured as an output or as a pure input. Configuring the I/O line as an output is done by setting and clearing the corresponding bits in the PIO Output Enable Register and PIO Output Disable Register. If the I/O line is configured as an output, the level (high or low) can be controlled by setting and clearing the corresponding bits in the PIO Set Output Data Register and PIO Clear Output Data Register. To configure an I/O line as a pure input the corresponding bits in the PIO Output Enable Register and the PIO Output Disable Register should be respectively cleared and set. When an I/Oline is pure input the current level on that line can be read from the PIO Data Status Register. This however requires the PIO controller clock to be enabled. If this is not the case the PIO Data Status Register will contain the level of the I/O line at the time the clock was disabled.

Input change interrupt Each I/O line can also be configured to generate an interrupt when it detects an input change. This is done by setting and clearing the PIO Interrupt Enable Register and PIO Interrupt Disable Register. Because an input change can only be detected by comparing to successive level samples of the line, the PIO controller clock must be enabled. The input change interrupt is available regardless of the configuration of the I/O line.

5-1-1-4 Timer Counter

The Timer Counter (TC) peripheral consists of three identical 16-bit timer counter channels, which can be programmed individually to perform a wide range different functions, such as interval measurement, delay timing, event counting, frequency measurement and several more. Each channel can be configured with one of the three external clock inputs or one of the five internal clock inputs. Furthermore an interrupt line is connected to each timer channel which can generated a processor interrupt. The TC also offers the feature to start/stop the TC three channels simultaneously or chain the three TC channels to create a 32-bit or 48-bit timer. In the paragraphs below some features used in the ADCS software are discussed in more detail.

Clock selection The TC peripheral is clocked through the PMC, so the user must first enable the TC peripheral via the PMC. Each TC channel has three external clock inputs and five internal clock inputs, which can be selected by writing the clock selection bits (TCCLKS) in the TC channel mode register. In Table 5-1 one can see the definition of the five internal input clocks.

Input clock	Definition
TIMER_CLOCK1	MCK/2
TIMER_CLOCK2	MCK/8
TIMER_CLOCK3	MCK/32
TIMER_CLOCK4	MCK/128
TIMER_CLOCK5	SLCK

Table 5-1: Input clocks for the three timer channels

For chaining the timer channels one can also select an interrupt output from one of the timer channels as an input clock. This is done by writing the right bits in the TC block mode register. For more detailed information about this one is referred to [41].

Clock control The clock of each timer channel can be controlled in two different ways. The clock can be enabled or disabled. Enabling and disabling the clock can be done by writing the CLKEN or CLKDIS in the TC Control register of that timer channel. When the clock is enabled the CLKSTA status bit in the TC Status Register corresponding to that timer channel is set. Disabling the clock can be done manually or by an external event.
The clock can also be started or stopped. Starting de clock is always done by a trigger, that can be generated by software or an external event. Stopping the clock can be done manually or by an external event. The start and stop actions only have effect when the clock is enabled.

Counting The TC counts from 0x0000 to 0xFFFF (i.e. from 0 to 65535). On each positive edge of the clock the counter is incremented with 1. When the counter value 0xFFFF is reached the counter overflows, the COVFS bit in the TC Status Register is set and the counter passes to 0x0000 again. Counting can also be done on the opposite clock edge by writing CLKI bit in the TC Channel Mode register, which inverts the clock. The counter value is accessible in real time by reading the TC Counter Value register. Resetting the counter value can be done by a trigger, which sets the counter back to 0x0000 on the next valid clock edge of the selected clock.

Compare interrupt The TC channels offer the feature to generate an interrupt when the timer counter reaches a certain value. To enable the compare interrupt one has to set the WAVESEL and the CPCTRG bit in the TC Channel Mode register. The value on which the compare interrupt is generated should be written in the TC Register C. When the compare interrupt occurs also a trigger can be generated, which stops, start or resets the current TC channel or one of the other two TC channels. If this trigger is not generated the timer counter continues counting. The value in TC Register C is not automatically cleared when a compare interrupt occurs, thus the compare interrupt keeps firing at regular intervals until the value

in TC Register C is cleared or the compare interrupt is disabled by clearing the CPCTRG bit in the TC Channel Mode register.

5-1-1-5 Real Time Timer

The Real Time Timer (RTT) peripheral is a 32-bit counter which has as an input clock the Slow Clock (SLCK) (32.768Khz) divided by a configurable 16-bit value. Usually the Real Time Timer is used to keep track of the elapsed seconds, but it can also be used as a timer with a lower time base. Setting the RTPRES value to 32768 in the RTT Mode Register will generated an input clock of 1Hz and thus a period of 1 second. Any value lower than 32768 will generate an input clock with a higher frequency and thus a shorter period. For higher values it is the other way around.
The value of the RTT can be read at any moment by reading the RTT Value Register.
The RTT also offers an 'Alarm' feature which generates an interrupt if the counter matches the value written in the RTT Alarm Register. After reset the value of the RTT Alarm Register is set to the maximum 32-bit value.

5-1-1-6 Watchdog Timer

The Watchodog Timer (WDT) peripheral is a 12-bit down counter to prevent the system from lock-up if the software is in a deadlock. The WDT is clocked by the SLCK (32.768Khz) divided by 128, which makes a maximum period of 16 seconds possible. By default de WDT is enabled, disabling can be done by setting de WDTDIS bit in the WDT Mode Register. This register can only be written once after each reset/power-up. When the WDT Mode Register is written the WDT is loaded with the new parameters. Setting the WDT period can be done by writing the WDV bits in the WDT Mode Register with a value which can be calculated with Eq. (5-1).

$$WDT\ Value = \left(\frac{Period\ in\ ms}{1000}\right) * \left(\frac{32768}{128}\right) \tag{5-1}$$

In normal operation the WDT should be reset before an underflow occurs. This is done by setting the WDRSTT bit in the WDT Control Register. Setting this bit immediately reloads the WDT with the parameters from the WDT Mode Register. If no reset is performed the processor is reset when an underflow occurs.

5-1-1-7 Debug Unit

The Debug Unit (DBGU) is a two-pin UART consisting of independently operating transmitter and receiver. Both transmitter and receiver share a common baud rate generator. To use the DBGU, the baud rate first has to be set by writing a value to the CD field in the DBGU Baud Rate Generator Register. The value that should be written in the CD field to set a certain baud rate can be determined by Eq. (5-2).

$$DBGU\ Baudrate = \frac{MCK}{16 \times CD} \tag{5-2}$$

With a maximum value of 65535 that can be written in de CD field, the minimum baud rate is depended on the clock frequency of the MCK. Both receiver and transmitter should be enabled, because by default they are disabled after a reset. Setting the RXEN and TXEN bits in the DBGU Control Register will enable the receiver and transmitter.
When the baud rate is configured and the receiver and transmitter are enabled, data can be received and transmitted over the UART by reading from the DBGU Receive Holding Register or writing to the DBGU Transmit Holding Register.

5-2 External memories

On the ICnova SAM9G45 SO-DIMM module there are three different external memories: 128MB of DDR2 RAM, 256MB of NAND-Flash and 8MB of NOR-Flash. In this section a short description of all three will be given.

5-2-1 DDR2 RAM

The DDR2 RAM is the fastest memory of the three and can be used as execution and working memory for an application. For storing data the DDR2 RAM is not suitable, because it is a volatile memory (i.e. data is not preserved during a reset or power loss). Of the three memories the DDR2 RAM is the most power consuming. Thus, when using DDR2 RAM the power consumption must be taken into account.

5-2-2 NAND-Flash

The NAND-Flash is a non volatile memory (i.e. data is preserved during a reset or power loss) and thus can be used to store data or an application. The application can not be executed directly from NAND-Flash, but should be copied to suitable execution memory first. It is possible to boot from NAND-Flash, after which any valid code found in NAND-Flash will be copied to SRAM and executed there (e.g. a boot loader which copies any application stored in NAND-Flash or NOR-Flash to DDR2 RAM for execution)

5-2-3 NOR-Flash

The NOR-Flash is also a non volatile memory and thus can also be used to store data or an application. In contrast to NAND-Flash, it is possible to execute an application directly in NOR-Flash. Executing an application directly from NOR-Flash is known as eXecute in Place (XiP). The application is stored, booted and executed in NOR-Flash. The advantage of executing an application in NOR-Flash is the power consumption. NOR-Flash has a much lower power consumption compared DDR2 RAM and is thus suitable for applications with limited availability of power. The disadvantage however, is that NOR-Flash is also slower than DDR2-RAM. Therefore, computationally intensive applications with strict timing deadlines are less suitable for execution in NOR-Flash.

Chapter 6

Delfi-n3Xt's Attitude Determination and Control Subsystem: Software

The Attitude Determination and Control Subsytem (ADCS) software can be subdivided into three different layers. The lowest layer is the Hardware Abstraction Layer (HAL), which consists of all the hardware specific functions. On top of the HAL there is the Service Layer (SL), which uses the HAL functions to provide services to the rest of the software. The top layer is the Application Layer (AL) which contains all the algorithms and the software which determines the control and data flow. Figure 6-1 shows an overview of the three layers. For the ICnova SAM9G45 there is an additional piece of software called the 'Bootstrap'. In this section an overview of the three different layers and the bootstrap is given.

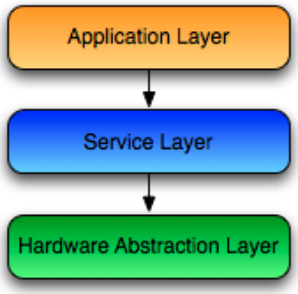


Figure 6-1: Overview of the three software layers.

6-1 Bootstrap

The Bootstrap is a standard piece of software created by Atmel which provides a set of algorithms to manage the initialization of the hardware (e.g. memories, clocks, interrupt handlers) and copy the main application from a specified memory into execution memory

and start it. In our case the specified memory is the NAND-Flash and the the execution memory is the DDR2 RAM. When the processor boots the bootstrap is the first thing that starts executing. The bootstrap copies itself to SRAM, does all the necessary hardware initialization and copies the main application from NAND-Flash to DDR2 RAM. After the main application has been copied into the DDR2 RAM, at a certain memory address, the bootstrap jumps to that memory address and starts executing the main application.
Initially the goal was to execute the main application directly from NOR-Flash (XiP), which would have made the Bootstrap unnecessary. However, attempts to run directly from NOR-Flash were not successful. First attempts failed, because we were unable to boot from NOR-Flash. Although, we succeeded to boot from NOR-Flash in later attempts, the execution speed did not meet our expectations. Time did not permitted us to investigate whether it was possible to gain some speed and therefore the initial goal was discarded.

6-2 Hardware Abstraction Layer

The Hardware Abstraction Layer (HAL) is a software layer that consists of all the hardware specific functions. All the functions in this layer have a predefined interface. Because the interfaces are predefined, and thus the same for both Micro Controller Unit (MCU)s, it is much less work to develop the software for different types of MCUs. The implementation of the functions itself however differs per MCU, because of the physical differences between the hardware. In our case we have two different MCUs, thus two versions of the HAL.
The functions in the HAL can be roughly divided into a couple of different modules according to their hardware peripheral. More details on the different modules are described in this subsection. An overview of the modules can be seen in Figure 6-2.

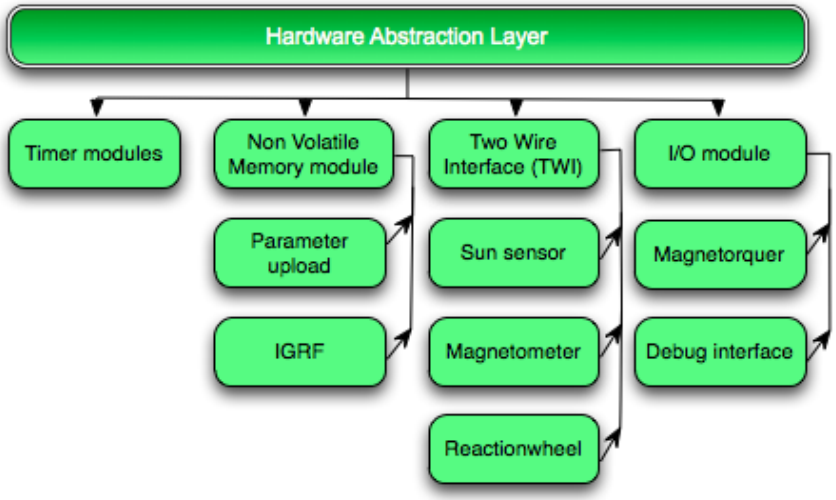


Figure 6-2: Overview of the Hardware Abstraction Layer.

6-2-1 Timer modules

All the timing related functions can be grouped together in the timer module. The ADCS software uses three different timers. The first timer is used is as an internal timer, the second timer, the so called 'elapsed time counter' is used to keep track of the total elapsed time since the start of the application and the third timer is the watchdog timer. A short description of all three timers is given below.

Internal timing module The internal timer uses the Timer Counter (TC) peripheral, discussed in Subsection 5-1-1-4 and is used for internal timing. The TC peripheral is configured with an input clock of 32.768KHz (i.e TIMER_CLOCK5 from Table 5-1), which corresponds to a timing resolution of 30,5μS. Since the TC is a 16 bits timer the maximum timing interval, with the selected clock, is two seconds. Any period between 30,5μS and 2 seconds can be timed by setting a timed callback using the compare interrupt feature of the TC peripheral.

Elapsed Time Counter module The Elapsed Time Counter (ETC) module uses the Real Time Timer peripheral, discussed in Subsection 5-1-1-5. The ETC is used to keep track of the elapsed time since start of the application. Because the Real Time Timer (RTT) can drift over time the module offers the functionality to set an 'alarm'-interrupt at a certain time interval. Each time this interrupt fires a value is added to the total elapsed time to correct for the drift. Furthermore the elapsed time can be requested at any time.

Watchdog timer module The Watchodog Timer (WDT) module uses the watchdog timer peripheral, discussed in Subsection 5-1-1-6. The WDT is configured with the value 4750 (i.e. a 4,75 second period) and is automatically started when the processors boots. The watchdog timer can be reset at any moment. If no reset is done and the WDT reaches the value 4750, the processor is resets.

6-2-2 Two Wire Interface module

The Two Wire Interface (TWI) module uses both the TWI peripherals, discussed in Subsection 5-1-1-2, available on the AT91SAM9G45. One TWI peripheral, (TWI1), is used in master mode and the other TWI peripheral, (TWI0), is used in slave mode. A description of the master mode and slave mode is given in the next two sub-paragraphs.

TWI1: Master mode The TWI1 peripheral is configured in master mode with a TWI frequency of 100Khz and is used to communicate with a device, such as the sensors and actuators. The module offers a write functionality, to write a number of bytes to a device, a read functionality, to read a number of bytes from a device, and a write-read functionality, to write a number of bytes to a device and directly read a number of bytes from the same device. Both read and write functions can be done sequential or on an interrupt basis.

TWI0: Slave mode The TWI0 peripheral is used for communication with the OnBoard Computer (OBC) (i.e. the master) and is therefore configured in slave mode. The 7-bit slave address, is set to 0x76 and the general call is enabled in order to receive sync signals from the OBC. Unlike the master mode, the slave mode is completely interrupt based and every time a byte is requested or received an interrupt is fired and handled. If a byte request interrupt occurs, necessary actions are taken and the byte is transmitted. If a byte receive interrupt occurs, the byte is read and depending on the contents of the received byte the appropriate action is taken.

6-2-3 Non Volatile Memory module

The Non Volatile Memory (NVM) module contains the functions to read and write to NVM, which in our case is the NOR-Flash. The functions from the NVM module are directly used by two other sub-modules, namely the parameter upload and the IGRF. The parameter upload sub-module offers the functionality to write and read specific parameters from a certain address and to a certain address in NVM. The IGRF sub-module only offers the functionality to read a set of IGRF values from NVM. Before flight all these IGRF values are stored at a certain address in NVM.

6-2-4 Input/Output Module

The Input/Output (I/O) module uses the Parallel Input Output (PIO) peripheral described in Subsection 5-1-1-3. The I/O module consist of the functions to control the magnetorquers. An initialize function initializes six pins, two pins for each magnetorquer, as general purpose output pins. For each magnetorquer separately there is a function to switch the magnetorquer on or off and set the direction to positive or negeative. This is done by setting or clearing one or both pins. The I/O module is also used to detect the data ready interrupt from the magnetometer. The data ready interrupt is connected to a pin on the AT91SAM9G45, which, during initialization of the magnetometer, is configured as an input pin with the input change interrupt enabled. Furthermore the module is used for the debug interface. Two pins are configured with the on-chip debug peripheral function described in Subsection 5-1-1-7.

6-3 Service Layer

The Service Layer (SL) is the software layer on top of the HAL that provide services to the rest of the software. Since the abstraction between MCUs is made in the HAL the service layer can be shared between MCUs. The service layer consist of several modules with their own sub-modules. In this section different modules and their sub-modules are described in more detail. An overview of the SL can be seen in Figure 6-3.

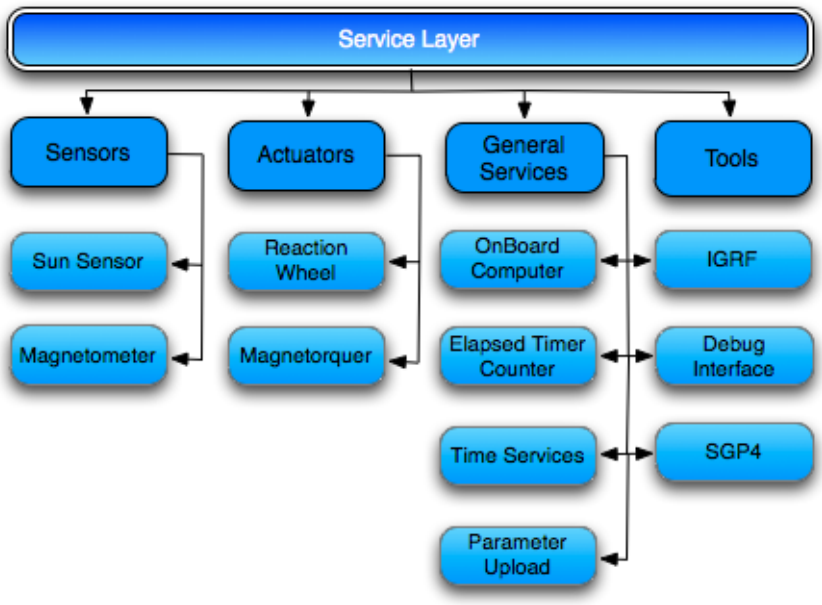


Figure 6-3: Overview of the Service Layer.

6-3-1 Sensors

The sensors module consists of the magnetometer and the sun sensor sub-module. In this subsection a brief description of the functionalities of both sub-modules is given.

6-3-1-1 Magnetometer

The magnetometer sub-module provides functionality to perform a magnetometer measurement, process the measurement and store the relevant data. It consists of three parts: the magnetometer 'core', 'service' and 'extended'.

Magnetometer core The magnetometer 'core' consist of all the functions that take care of writing configurations to the magnetometer and retrieving measurements from the magnetometer, using the functions from the HAL. Three different configurations can be written to the magnetometer. Depending on which configuration (i.e. normal, positive bias, negative bias) is written to the magnetometer, the magnetometer performs a normal, a positive bias or a negative bias measurement. Directly after a configuration is written the magnetometer does a measurement and when a data ready interrupt occurs the measurement data is retrieved from the magnetometer and stored in a temporary storage buffer for further processing.

Magnetometer services The magnetometer 'services' consist of functions to initialize ring buffer storage space to store the measurements, Bdot values and Tumble parameter values, functions that process the raw measurement data and check them for validity and functions

to retrieve previously stored measurements from the ring buffers. The function that processes the raw data takes the measurement from the temporary storage buffer and checks whether it is in the valid range and converts the raw data to Tesla's. If a measurement is invalid, it is marked as invalid and stored in the measurement ring buffer, but will not be used in further calculations. A valid measurement will be marked as valid, stored in its measurement ring buffer and used in further calculations.

Magnetometer extended The third part is the 'magnetometer extended', which consist of functions that apply the bias corrections and transformations to the magnetometer measurements, functions that do post processing of the measurements (i.e. calculate the Bdot and the Tumble parameter) and functions to retrieve previously stored Bdot an Tumble parameter values from the ring buffers. Before post processing of the measurements takes place, a bias correction and a transformation needs to be applied to correct for the hard and soft iron distortions. The bias correction and transformations can be summarized with Eq. (6-1).

$$\boldsymbol{B}_{actual_i} = \boldsymbol{A}(\boldsymbol{B}_{meas_i} - \boldsymbol{B}_{bias}) \tag{6-1}$$

Why this bias corrections and transformation are needed is explained in more detail later in Chapter 7. After the measurements are bias corrected and transformed they are post processed. During post processing the Bdot and Tumble parameter are determined. To determine the Bdot, a time derivative of the magnetic field, the previously stored measurement is retrieved. Together with the current measurement the Bdot is calculated according to Eq. (6-2).

$$\boldsymbol{Bdot}_i = \frac{\boldsymbol{B}_i - \boldsymbol{B}_{i-1}}{\Delta t} \tag{6-2}$$

Where Δt is the time difference between the two measurements. If no previous measurement or valid measurement is available, no Bdot calculation is done. As later will be explained in Chapter 7, an extra low-pass filter is applied to reduce noise in the Bdot. The low-pass filtered Bdot is stored in the Bdot ring buffer. The Tumble parameter [3], which is a indication of the tumbling rate of the satellite, is calculated from the magnitude of the current Bdot and the previous Tumble parameter according to Eq. (6-3).

$$P_{tumble_i} = \alpha_{tumble} \parallel \boldsymbol{Bdot}_i \parallel + (1 - \alpha_{tumble}) P_{tumble_{i-1}} \tag{6-3}$$

Where α_{tumble} is a design parameter determined in [3] and has a value of 1/1200. The calculated Tumble parameter is stored in the tumble parameter ring buffer.

6-3-1-2 Sun sensors

The sun sensor sub-module provides functionality to initialize ring buffer storage space for storing the measurements, to perform a sun sensor measurement and to process this measurement, to determine a sun vector and to store the relevant data. It consists of a function to initialize ring buffer storage space to store the measurement data from the sun sensor, the

temperature sensor and the determined sun vector, a function that retrieves measurements from the sun sensor and temperature sensor using functions from the HAL, checks them for validity and processes the raw data, a function that determines a sun vector from the processed data and functions to retrieve previously stored sun sensor measurements, temperature values or determined sun vectors.

Sun sensor measurements The 'measure' function retrieves raw measurement data from the six sun sensors and temperature sensors. Both sensor measurements are checked for validity and converted to Volts and degree Celsius. Valid and invalid measurement are stored in its measurement ring buffer, but the invalid ones are marked invalid and will not be used in further calculations.

Sun vector determination The 'process' function takes the current measurement from its ring buffer and determines a sun vector and a sun presence flag. The sun presence flag is determined by checking if the sum of the quadrant voltages of a sun sensor exceeds a threshold. If one sun sensor exceeds this threshold the sun presence flag is set to true and else to false. When the sun presence flag is true a sun vector is determined with the sun vector determination algorithm designed in [38]. If there is no valid measurement or if the sun presence flag is false, the three sun vector components are set to zero. The determined sun vectors and sun presence flags are stored in the sun vector ring buffer.

6-3-2 Actuators

The actuators module consists of the magnetorquer and the reaction wheel sub-module. In this subsection a brief description of the functionalities of both sub-modules is given.

6-3-2-1 Magnetorquers

The magnetorquer sub-module provides functionality to activate and shut down the magnetorquers using functions from the HAL. Activation of the magnetorquers is done by calling the 'activate' function. The 'activate' function determines for each magnetorquer the period $\Delta t_{mtq_{on}}$ they should be switched on. This time is dependent on the magnetic dipole needed, the maximum dipole the magnetorquers can deliver and the time available. With the determined $\Delta t_{mtq_{on}}$ a timed callback is scheduled $\Delta t_{mtq_{on}}$ seconds in the future. When a timed callback triggers the corresponding magnetorquer is switched off automatically. Each magnetorquer can also be switch off manually.

6-3-2-2 Reaction wheels

The reaction wheel sub-module provides functionality to initialize ring buffer storage space for storing the measurements and commanded accelerations, to switch each reaction wheel on and off, to retrieve a RPM measurement from each reaction wheel and to command all three reaction wheels.

Reaction wheel measurements The 'measure' function retrieves RPM measurement data from each reaction wheel and calculates the momentum stored in the reaction wheels. Both RPM measurement data and stored momentum are stored in its ring buffer for further use in the algorithms.

Reaction wheel commanding Commanding of the reaction wheels is done by calling the 'activate' function. The 'activate' functions first determines the amount of acceleration, in RPM/s, that needs to be sent to each reaction wheel. The amount of acceleration depends on the needed torque computed by one of the algorithms and on the physical properties of each reaction wheel. The computed accelerations are sent to the reaction wheels and stored in the commanded accelerations ring buffer. In order to command the reaction wheels they must be switched on, sending an acceleration when they are switched off will not have any effect.

6-3-3 General services

The general services module contains all the sub-modules that are not related to any sensor or actuator. It consists of an OBC, Parameter upload, ETC, Time services, IGRF, SGP4 and Debug interface sub-module. This subsection gives a brief description of these sub-modules with their functionalities.

6-3-3-1 OnBoard Computer

The OnBoard Computer (OBC) sub-module provides functionality to handle all communication with the OBC and to gather and prepare housekeeping data for transmitting it to the ground station. Communication with the OBC is always handled by the 'OBC interrupt handler', which is called from the TWI slave module if the OBC tries to communicate with the ADCS. The 'OBC interrupt handler' determines first whether the OBC tries to write something (i.e. a read for the ADCS) or tries to read something (i.e. a write for the ADCS). If a read is detected the read byte handler is called and if write is detected the write byte handler is called.

Read byte handler The first things that the read byte handler does is verifying whether the byte received is the first data byte of the current transfer. If so, the byte is checked for its contents. Depending on the contents of the byte different actions are taken. If it equals '0xFF' then a sync signal was received and a sync received flag is set to true. Else, if the first byte equals '0x00' a reset signal is received and the processor is reset. If the first byte is neither '0xFF' or '0x00', the 4 most significant bits of the first byte are checked. When these bits are equal to '0x2-' then a mode switch is requested by the OBC. The 4 least significant bits of that byte then indicate to which mode the ADCS should switch in the next loop. If the 4 most significant bits are equal to '0x8-' it indicates that the OBC is about to send a parameter upload to the ADCS. If a parameter upload is requested it is first verified that there are no other pending requests. If this is not the case then the byte is stored in the parameter upload buffer and a parameter pending flag is set to true. Until the communication is terminated, any following bytes are also stored in the parameter upload buffer.If the contents of the first

byte is different than the aforementioned options, the byte is ignored and no action is taken. Any received bytes following this byte are also ignored.

Write byte handler The write byte handler is only called when the OBC is requesting the ADCS for housekeeping data. The write byte handler takes the N'th byte from the housekeeping data buffer and places it in the data buffer of the TWI slave module, from which it is transmitted to the OBC. After a byte is transmitted N is increased. If the communication with the OBC is terminated, N is reset to zero. The housekeeping data buffer contains a total of 122 bytes of housekeeping data, thus every time the OBC requests for housekeeping data there are 122 bytes transmitted to the OBC. In chap::hkdata some more information about the housekeeping data is given and Table A-2 gives an overview of the data that is part of the housekeeping data.

6-3-3-2 Parameter upload

The parameter upload sub-module provides functionality to update various parameters in during flight. It consist of functions to store the parameters in storage format, to read the parameters back from NVM and convert them to execution format and to update parameters.

Paremeter storage format The parameters are stored in NVM in the so called storage format. In this storage format the parameters only take up the nescessary memory space that is needed to represent the parameter value range with a certain resolution. All the parameters are stored three times in NVM on a different memory location to provide redundancy. If a parameter set at one memory location becomes corrupt it can be updated with one of the others. The first time the application starts the NVM is checked whether there are valid parameters available in NVM. This is done by checking the 'parameter data valid code' byte that should be the first byte of each parameter set. If a parameter set is missing the 'parameter data valid code', the parameters are (re)written to NVM.

Parameter execution format For use in the application the parameters are converted from storage format to the execution format. The execution format are basically the standard data types such as integers and doubles. Every time the application starts the parameters are read from NVM and converted to execution format. Also when parameters upload occurs the relevant parameters are updated in the execution format.

Parameter update Updating of the parameters is done by sending a parameter upload command as described in the previous paragraph. At the end of each main control loop a check is performed whether a parameter upload request is pending. If a parameter upload request is pending it is processed by the 'process' function. The 'process' function takes the received bytes, which are in storage format, from the parameter upload buffer and deciphers the first two bytes, which contain information about the amount of parameters that need to be updated and a start ID indicating from which on parameters should be updated. The rest of the bytes in the parameter upload buffer are the data bytes, which contain the values of the relevant parameters. A parameter upload request consist of at least four bytes. (i.e.

I2C address, command/amount, start ID and value). With information about the amount of parameters to be updated and the start ID of the parameter it is verified whether the amount of data bytes that are supplied correspond with the total bit size of the parameters that need to be updated (Information about the bit size of each parameter is hard coded into the software). If this is not the case processing is aborted and the ADCS status flag is set to PARAMETER UPLOAD FAILED. When enough data bytes are supplied the parameters are updated in NVM and in the execution format and the ADCS status flag is set to PARAMETER UPLOAD SUCCESS if the update succeeds and else to PARAMETER UPLOAD FAILED.
More information and an example on how to update parameters using parameter upload is given in Appendix B.

6-3-3-3 Elapsed Time Counter

The Elapsed Time Counter (ETC), as described in the HAL, is used to keep track of the elapsed time since the start of the application. The ETC sub-module provides the functionality to return the current elapsed time and to enable and set a value and period for the drift correction. The period and the value for the drift correction can be uploaded via the parameter upload.

6-3-3-4 Time services

The Time services sub-module uses the internal timer module functions from the HAL to provide all timing related functionality. A couple of this timing related functions is listed below.

- Reset the timer.
- Wait/Sleep for a certain period.
- Get elapsed time since timer reset
- Schedule/Unschedule a timed callback using the compare interrupt feature of the timer

6-3-3-5 IGRF

The International Geomagnetic Reference Field (IGRF) is standard mathematical description of the Earth's magnetic field. With this mathematical description and the 11th generation of Gauss coefficients released by the International Association of Geomagnetism and Aeronomy (IAGA), the earth magnetic field strength in X,Y and Z direction at every latitude/longitude combination can be determined. Since the model is very computational intensive, all the values for each latitude/longitude combination are pre-calculated before flight and stored in a look-up-table in NVM. To reduce the amount of memory needed for storage, all the values are stored as 16 bit values [3]. The amount of memory needed to store all the values is given by Eq. (6-4).

$$Amount\ of\ memory = Longitude * Latitude * 3 * 16bit = 360 * 181 * 3 * 16 = 2,98MB \quad (6\text{-}4)$$

The IGRF sub-module provides the functionality to read a set of X,Y and Z values from NVM given a latitude and longitude value as input.

6-3-3-6 SGP4

The SGP4 model, where SGP stands for Simplified General Perturbations, is one of the simplified perturbation models to calculate the orbital state vector of a satellite at a particular time. The orbital state vector is a set vectors of position and velocity that together with their time defines the orbit in space. To determine the orbital state vector at a particular time, the SGP4 takes as an input a set of Two Line Elements (TLE's), which describe the orbits of objects orbiting the earth, and predicts the effects of perturbations, for near earth objects, caused by the Earth's shape, drag, radiation and gravitational effects of the sun and the moon.
The SGP4 sub-module, written by Prof. Dr. E.K.A Gill and integrated into the software framework by M. Vos, consists of several functions which describe the SGP4 model. One function extracts the TLE's from the parameter upload and two other functions calculates the orbital state vector from these TLE's and the elapsed time provided by the elapsed time counter.

6-3-3-7 Debug interface

The Debug interface sub-module is used during development to log useful information, such as measurement data from the sensors, error/warning messages or control flow of the application. It uses the debug functions from the HAL layer to sent debug output via the debug unit, to a computer running a terminal application. The Debug interface features some macros to distinct between different groups of debug output, such as errors, warnings, data and control flow. Each group can be enabled and disabled individually, therefore making it easy to get for example only error messages or only measurement data as debug output. For the flight version of the software the Debug interface can entirely be disabled.

6-3-4 Tools

The Tools service layer consist of collection of functions and macros that are often used through the whole code. In the algorithms and for processing measurement data a lot of matrix arithmetic is needed. Therefore, a matrix arithmetic library is written, which contain macros such as matrix multiplication, addition and subtraction, but also simple macros for initialization of matrices. Furthermore the Tools service layer also incorporates a ring buffer functionality, to store measurements and retrieve previous measurements upto N steps back, where N is the length of the ring buffer. For debug purposes there are also some functions to do type conversion. For example, if we want to print an I2C address in hexadecimal, the integer value must be converted to a hexadecimal value.

6-4 Application Layer

The Application Layer (AL) is the most upper layer. It implements the main control loop and the different operational modes. Figure 6-4 shows an overview of the AL. In this section the implementations of the main control loop and the different operational modes are described.

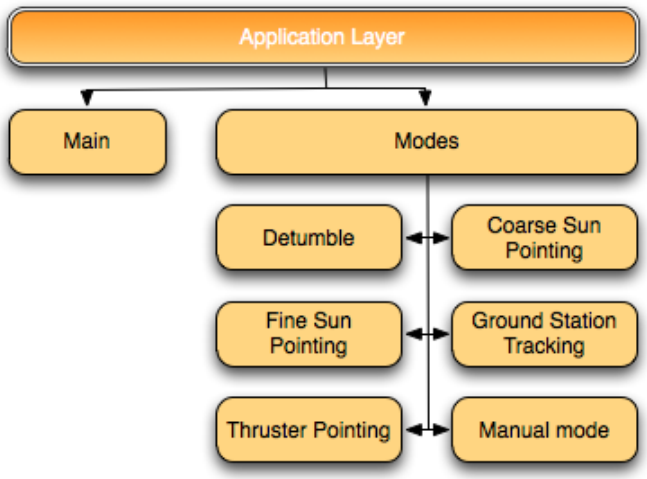


Figure 6-4: Overview of the Application Layer.

6-4-1 Main control loop

Once the ADCS the is turned on by the OBC and the Bootstrap has copied the main application to execution memory the main control loop starts running. From the main control loop the flow through the rest of the application is determined. In Figure C-1 a control flow graph of the main control loop is given.
From Figure C-1 it can be seen that before the actual main control loop is entered first some initializations are performed. After all initialization procedures are completed the application enters the main control loop. At the beginning of the main control loop the application waits for a received sync signal, which sets the sync received flag to true. As soon as this flag is set to true the main control loop continues executing beginning with resetting the internal timer and the watchdog timer. After that a magnetometer measurement is performed and processed. If this magnetometer measurement is invalid a status flag indicating the status of the ADCS is set to MTM_ERROR. Following the magnetometer measurement a sun sensor and a reaction wheel measurement are performed and processed. When all the sensor measurements are completed the current Tumble parameter is compared with a threshold $L_{tumbling}$. If it exceeds this threshold and not already in Detumble mode, a mode switch back to Detumble mode is requested. After this, if a mode switch is requested, the mode switch is performed. If it is a mode switch back to Detumble mode an initial Tumble parameter $L_{tumbkle_0}$ is added to the Tumble parameter ring buffer. The value of $L_{tumbkle_0}$ is determined in [3] to be $9.4 \cdot 10e^{-7}$. When it is a mode switch from Detumble mode to an other mode then a Tumble parameter exit value $P_{tumble_{exit}}$ is added to the Tumble parameter ring buffer. The value for $P_{tumble_{exit}}$ is determined in [3] to be $1.0 \cdot 10e^{-7}$. This value is much lower than $L_{detumbled}$ in order to prevent from switching directly back to Detumble mode.
After any mode switches the current selected mode is executed. When finishing execution of this mode all housekeeping data is gathered and prepared to be transmitted to the OBC if it sends a request. At the end of the main control loop the ADCS status flag is reset and any pending parameter upload requests are processed. Then the main control loop starts all over again, waiting for a sync signal from the OBC

6-4-2 Detumble mode

The Detumble mode is the most basic mode of the ADCS application and takes care of detumbling of the satellite. It is based on the Bdot controller described in Subsection 3-5-2 and also explained [3]. The control flow of the mode is shown in Figures C-2 and C-3 and is an implementation of the steps described in Appendix C in [3].

As can be seen from Figure C-2 the Detumble mode consist of 'detumble main', which is called every 2 seconds from the main control loop, and a 'detumble control step' which is executed N times every time 'detumble main' is executed. The value for N thus directly determines the time for one detumble control step. Currently the value for N has been set to N=4, which corresponds with a detumble control step of 0.5 seconds. In [3] it has been determined that with a detumble control step of 0.5 seconds, the Detumble mode has a theoretical upper limit of 180 degree/s from which it can still detumble the satellite. If detumbling from higher rotation rates is needed, a shorter detumble control step can be chosen. This however reduces the usage efficiency of the magnetorquers.
The first time 'detumble main' is entered the detumble counter is initialized to a predetermined value and the reaction wheels are switched off. After initialization N detumble control steps are performed. A detumble control step consist of several steps. First a magnetometer measurement is performed and processed to determine the Bdot and Tumble parameter. Secondly, with the Bdot control law from Eq. (3-71) the needed magnetic dipole is calculated. Then the magnetorquers are activated and another magnetometer measurement is performed and processed. At the end of the detumble control step it waits for the magnetorquers to shut down. If this does not happen within the 0.5 seconds (minus some buffer time) of this detumble control step, the magnetorquer shut down is forced. The first detumble control step differs from the other N-1, because it uses the magnetometer measurement done in the main control loop to determine the Bdot and Tumble parameter.
When the N detumble control steps are completed it is checked whether the Tumble parameter P_{tumble_i} is below a certain threshold $L_{detumbled}$. If this is the case, then the detumble counter is decreased, else the detumble counter is reset. When the detumble counter reaches zero, a mode switch is requested, to either Coarse Sun Pointing (CSP) or Fine Sun Pointing (FSP). In [3] the value for $L_{detumbled}$ is determined to be $L_{detumbled} = 9.4 \cdot 10e^{-7}$, which should correspond to tumbling rates below 1 degree/s.
This completes the execution of a 'detumble main' call and the application returns to the main control loop.

6-4-3 Coarse Sun Pointing mode

After the satellite is detumbled the application can switch to either Coarse Sun Pointing (CSP) mode or Fine Sun Pointing (FSP) mode. The CSP mode tries to point the solar panels of the satellite in the direction of the sun using the Q2-method described in [3]. The control flow of a CSP control loop is based on the steps described in Appendix D in [3].
In the first step of CSP mode the magnetometer, sun sensor and reaction wheel measurement data, from the measurements performed in the main control loop, are used to determine the Bdot and Tumble parameter, a sun vector and the stored angular momentum in the reaction wheels. After that the total amount of control torque, $\boldsymbol{T}_{conrol}$, needed to orient the satellite, is

calculated using the simplified Quaternion feedback regulator law discussed in Subsection 3-5-2-2 and shown in Eq. (3-74). The attitude and rotational rate errors used in this control law are those from the previous CSP control loop.
When the total control torque is calculated it is verified whether the reaction wheels should be unloaded, by checking if the stored angular momentum exceeds a threshold. If this is the case the reaction wheels are unloaded using one of the unloading algorithms described in [3]. If this is not the case load dividing is performed, which divides the control torque between the magnetorquers and the reaction wheels. With the determined control torques, for both magnetorquers and reaction wheels, the torque time for the magnetorquers and acceleration for the reaction wheels are calculated and both are activated.
After both actuators are activated the attitude and the rotational rates of the satellite are estimated using the methods described in [3]. From the estimated attitude, $\hat{\boldsymbol{q}}_i$, and the estimated rotational rates, $\hat{\boldsymbol{\omega}}_i$ the attitude error, $\boldsymbol{q}_{e_i}$, and the rotational rate errors, $\boldsymbol{\omega}_{e_i}$, are calculated. These errors are the differences between the estimated $\hat{\boldsymbol{q}}_i$, $\hat{\boldsymbol{\omega}}_i$ and the desired ones. In CSP mode the desired attitude and rotational rates are $\boldsymbol{q}_{d_i} = [0, 0, 1, 0]^T$ and $\boldsymbol{\omega}_{d_i} = [0, 0, 0]^\circ/s$. The errors calculated in the current CSP control step are used in the next CSP control step to calculate the control torque.
At the end of each CSP control loop it is determined whether a mode switch from CSP to FSP should be requested. This depends on the availability of TLE parameters.

6-4-4 Advanced modes

The three advanced modes (i.e. Fine Sun Pointing (FSP), Thruster Pointing (TP) and Ground Station Tracking (GST)) try to point the solar panels of the satellite to the sun (FSP), the thruster to the velocity direction (TP) or the S-band Transmitter (STX) to the ground station (GST). All three modes use the same attitude control algorithm as used in CSP mode, but use a different attitude determination algorithm, namely a variant of the Kalman filter discussed in Paragraph 3-4-2-2. The control flow of the advanced mode control loops are based on the steps described in Appendix E in [3].
Until the activation of the actuators the three advanced modes are exactly the same as CSP mode. After activation of the actuators the desired attitude and rotational rates for the current mode are calculated. In FSP mode the desired attitude, $\boldsymbol{q}_{d_i}$, is calculated using only sun vector data. The desired rotational rates in FSP are $\boldsymbol{\omega}_{d_i} = [0, 0, 0]^\circ/s$. In TP and GST mode the desired attitude is calculated using sun vector data and orbit information from the SGP4 orbit propagator. The desired rotational rates in both modes are determined by first calculating the attitude for these modes one time step in the future and from that the desired rotational rates can be determined to keep the satellite pointing in the right direction.
Using the the Kalman filter variant described in [3] the current attitude is then estimated by combining the sensor measurements, data from the SGP4 orbit propagator, data from IGRF magnetic field look-up table and the attitude and rotational rate errors from the previous control loop. From the estimated attitude, $\hat{\boldsymbol{q}}_i$, and the estimated rotational rates, $\hat{\boldsymbol{\omega}}_i$ the attitude error, $\boldsymbol{q}_{e_i}$, and the rotational rate errors, $\boldsymbol{\omega}_{e_i}$, are calculated. These errors are the difference between the estimated $\hat{\boldsymbol{q}}_i$, $\hat{\boldsymbol{\omega}}_i$ and the desired ones.
At the end of each advanced mode control loop it is determined whether a switch from FSP to either TP or GST should be requested. A mode switch to either TP or GST is only requested when the ground station becomes visible within a certain period. This period depends on the

time the satellite needs to reorient itself. When the ADCS is currently in either TP or GST it is verified whether the ground station is still visible. If so, no mode switch is requested, else the ADCS automatically requests a mode switch back to FSP mode.

6-4-5 Manual mode

The manual mode is an extra introduced mode, where can be switched to if the satellite is detumbled. Switching to this manual mode should be done with a command from the ground station. The manual mode was implemented as a backup mode in case the CSP mode and none of the advanced modes were completed before the deadline and is only used to test the reaction wheels.
In manual mode a maximum positive acceleration is sent to the reaction wheels for a period of 30 sync periods (i.e. 60 seconds). This will accelerate the reaction wheels to around +9000 RPM. Then for 10 sync periods a zero acceleration is sent, which keeps the reaction wheels at a constant RPM. Finally, for another 30 sync periods a maximum negative acceleration is sent to the reaction wheels, which should decrease the RPM of the reaction wheels to around 0 RPM. After in total 70 sync periods the reaction wheels are switched off and a mode switch back to detumble mode is requested. If during manual mode the satellite starts tumbling too fast the reaction wheels are also switched off and a mode switch back to detumble mode is requested.

Chapter 7

Verification of the detumble hardware and software

For the Delfi-n3Xt mission it is crucial that the detumble mode of the Attitude Determination and Control Subsytem (ADCS) works correctly. If this is not the case the tumble rate of the satellite will continue to increase until the point where the satellite becomes completely uncontrollable. Therefore, it is important that a good verification of the detumble hardware and software is done. With this verification we must verify that all hardware is working correctly and the software does what it is supposed to do. After verification we should guarantee a correct working of the detumble mode if no hardware failures occur during launch or during operations.

In this chapter an overview of the verification procedure for the detumble hardware and software is given. Section 7-1 starts with a description of the test facility. The two following sections describe the verification of the magnetorquer and magnetometer hardware and software. The last section concludes this chapter with a complete detumble test.

7-1 Test facility

To verify the correct working of the detumble hardware (i.e. the magnetometer and the magnetorquers) and software a good test facility is required. This test facility should meet the following requirements.

- The test facility must be able to generate a reliable homogeneous static magnetic field in three directions.
- The magnetic field strength in each direction should be adjustable to a user specified magnetic field strength.
- The test facility must be able to generate a rotating magnetic field around each axis, with a user adjustable rotation speed and field strength.

- The test facility must have a magnetometer, which is known to give reliable measurement data, to measure the generated magnetic field and can be used as a reference.

The test facility meeting all this requirements is located in the clean room of the Faculty of Aerospace Engineering and consist out of a Helmholtz cage, a reference magnetometer and a computer to control the Helmholtz cage and process measurement data. This section gives a more detailed overview of the Helmholtz cage, the reference magnetometer and the software to control the Helmholtz cage and process the measurement data.

7-1-1 Helmholtz cage

The Helmholtz cage, designed by F.M. Poppenk [42], was originally built for the Delfi-C^3, but is also a good testing environment to test the magnetometer and the magnetorquers for the Delfi-n3Xt. The Helmholtz cage is built from non magnetic materials and consist of an outside frame and three pairs of square electric coils inside. With these electric coils a homogeneous magnetic field in X, Y and Z direction can be generated inside the Helmholtz cage. Figure 7-1 shows the Helmholtz cage.

Figure 7-1: The Helmholtz cage located in the clean room.

Each pair of coils can be shifted individually in the cage to create a larger or smaller volume. A larger volume will allow testing of larger objects, but has the disadvantage that it suffers from difference in magnetic field strength and direction when comparing the center of the Helmholtz cage with the boundaries [42]. A smaller volume on the other hand creates a more homogeneous field, but limits the size of the test object. Therefore a trade off should be made between the field homogeneity and the test volume. For testing the magnetometer and the magnetorquers it was chosen to place each pair of electric coils 60cm apart from each other, creating a cube of 60x60x60cm.
The electric coils are powered by three power supplies, for each pair one, which can generate up to 30 volts and 10 amperes. The bench supplies are connected to a computer which controls the voltage output of the power supplies with software written in LabView.
Inside the cage a table is placed to put the ADCS and the reference magnetometer on. The

height of the table is such that the ADCS and reference magnetometer are exactly in the middle of the 60x60x60 cm cube if they are placed in the middle of the table. The table itself is also built from non magnetic materials.

7-1-2 Reference magnetometer

On the table inside the cage a reference magnetometer is placed, which measures the field generated by the cage. The measurements from the reference magnetometer serve as a reference for verifying the correct working of the magnetometer on the ADCS. The reference magnetometer is a 2,5cm cube with three sensor elements inside that measure the magnetic field in X,Y and Z direction. The reference magnetometer has a resolution of 0.01 milligauss and a range of +/- 2000 milli Gauss. A long flat cable connects the reference magnetometer to a device to display the measured data. This device itself is again connected to the computer, through an USB cable, making it possible to process the measurement data in real-time. Figure 7-2 and Figure 7-3 shows the reference magnetometer and the display device.

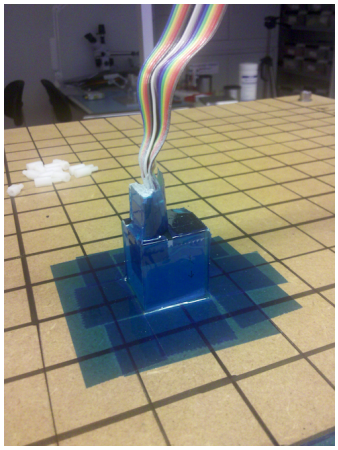


Figure 7-2: he reference magnetometer with the three sensor elements inside.

Figure 7-3: The gauss meter which displays data from the reference magnetometer.

7-1-3 Computer hardware and software

The computer used to control the Helmholtz cage an process the measurement data is equipped with a PCI card with six extra COM ports. Three of these COM ports are connected via a serial cable to the three power supplies. The power supplies are controlled with an LabView application, originally written by K. de Ridder but modified to our needs by L. Rotthier. With this application we are able to generate a static or rotating magnetic field of variable strength inside the Helmholtz cage. The application also shows in real-time the magnetic field measured by the reference magnetometer. These measurements can all be stored in a tab separated file for later processing with Matlab. Figure 7-4 shows the graphical user interface of the LabView application.

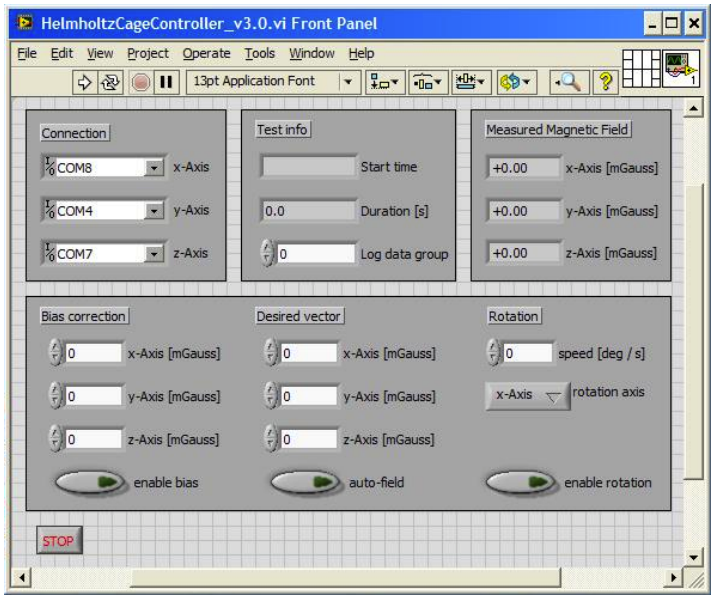


Figure 7-4: The graphical user interface of the LabView application.

For logging all the UART data of the ADCS, the ADCS is connected via a Serial-to-USB converter to the computer. A terminal application logs and stores all this data in a log file for later processing.
To analyse all the logged data an application written in Matlab is used. This application is written by L. Rotthier and slightly adjusted by the author of this thesis. With this application it is possible to plot the magnetic field data from the reference magnetometer and from the magnetometer of the ADCS, separately for each direction (i.e. X, Y and Z) or in 3D. Furthermore the application has the feature to do a transformation and/or bias correction on the magnetometer data of the ADCS. In Figure 7-5 the graphical user interface of the Matlab application is depicted.

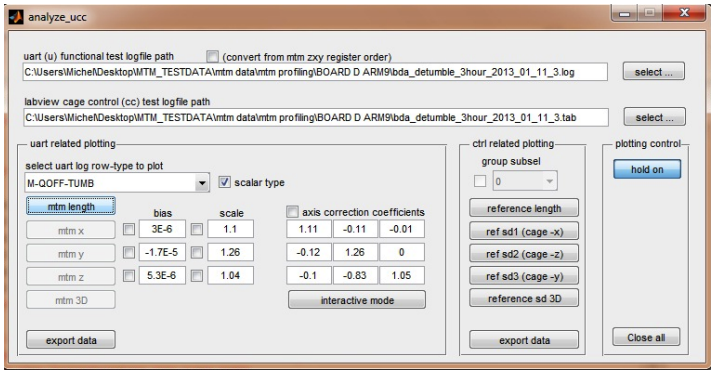


Figure 7-5: The graphical user interface of the Matlab application.

7-2 Verification of the magnetorquers

Verifying whether the magnetorquers work correctly involves testing several things. First it is important that the magnetorquers switch on and off when commanded to. For good performance it is important that the magnetorquers are only torquing the time that is required. If they continuously torque too short the performance will decrease, if they continuously torque too long, especially with low tumbling rates, the satellite can start tumbling again. Furthermore, for correct working the maqnetorquers should always switch off when commanded to. If they switch off too late and are still torquing when a magnetometer off measurement is done, the measurement is faulty, which can lead to incorrect actuation causing the satellite to start tumbling. In worst case scenario the magnetorquers do not switch off at all, in which the satellite keeps accelerating and eventually becomes uncontrollable. This scenario would be mission killer. Secondly, it is important that the magnetorquers are torquing in the right direction. If they torque in the opposite direction in which it is commanded, the satellite's tumble rate will be increasing instead of decreasing.
Another property that should be tested is the remanance of the magnetorquers. When the magnetorquers are switched on the mu-metal cores in the magnetorquers are magnetized. If they are switched off again they should demagnetize completely. Any magnetization left behind is called the remanence or residual magnetization and can influence the magnetometer measurements, which again can lead to incorrect actuation.
Finally the magnetic dipole moment that each magnetorquer delivers, when switched on, must be determined. This is important to in order to convert the calculated amount of torque needed to the time each magnetorquer should be torquing.
The following subsections will give a more detailed description and results of the above four mentioned tests.

7-2-1 On/off switching

To verify whether the maqnetorquers switch on and off when they are commanded to, a small test set-up was created. An schematic overview of this test set-up is depicted in Figure 7-6.

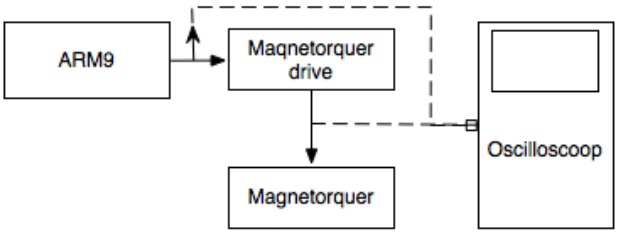


Figure 7-6: Schematic overview of the magnetorquer test set-up

In Figure 7-6 can be seen that the output of the ARM9 is directly connected to an input channel on the oscilloscope. Furthermore, the output from the magnetorquer-drive to the magnetorquer itself is connected to another input channel on the oscilloscope. This way we can see on the oscilloscope when we give the 'switch on'-command on the ARM9 and when the magnetorquer drive actually switches on the magnetorquer. A small test application was written, that uses functions from the magnetorquer service layer, to switch the magnetorquers

on and off with a constant frequency. In Figure 7-7 we can see the first test result with an on and off switching frequency of 10KHz (i.e. 50μs on and 50μs off).

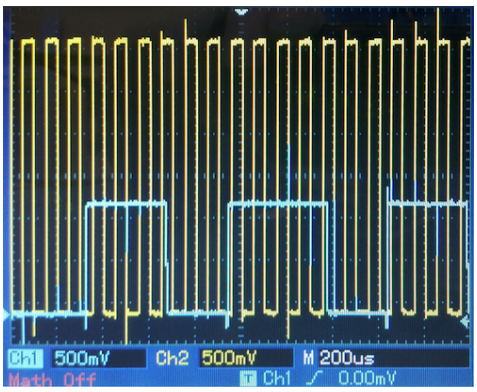


Figure 7-7: On/off switching of the magnetorquer with a 10KHz frequency.

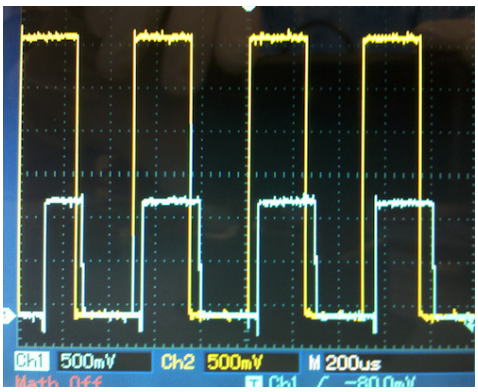


Figure 7-8: On/off switching of the magnetorquer with a 2Khz frequency.

From Figure 7-7 it is clear that a switching frequency of 10Khz is to fast. The blue line (i.e the output to the magnetorquers) does not follow the yellow line (i.e. the output of the ARM9) as supposed to. Lowering the frequency to 2KHz gives already a much better result, as can be seen in Figure 7-8. There is still a small delay between the 'switch on'-command and the actual switching on of the magnetorquer, but this inevitable and has no impact on the performance. Since there was no strict requirement on the switching frequency we have set the minimum to 2KHz. From Figure 7-8 it can also be seen that the rise and fall time of the magnetorquers is far below the maximum of 5ms that was determined in [43].
With the same test set-up another test was performed. In this test the actual ADCS main application, running the detumble mode, was used to see if the magnetorquers are correctly switched on after a magnetometer measurement and switched off again before a next magnetometer measurement. The result of this test is shown in Figure 7-9 and Figure 7-10.

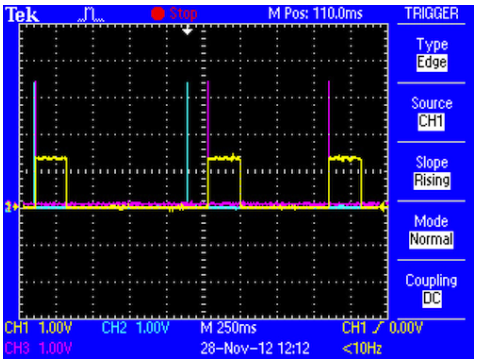


Figure 7-9: On/off switching of the magnetorquers in detumble mode

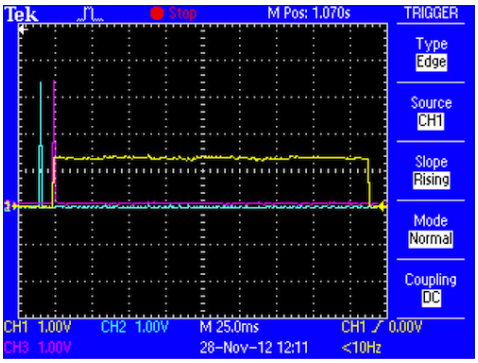


Figure 7-10: Close-up of Figure 7-10
.

The blue spike and the purple spike both indicate the moment when a magnetometer measurement is done. However the blue spike indicates a magnetometer measurement when the magnetorquer should be switched off and the purple spike indicates a magnetometer measurement when the magnetorquer should be switched on. The yellow line indicates whether the magnetorquer is switched on or off (i.e. high or low). We can see, from Figures 7-9 and 7-10, a correct behaviour of the magnetorquer. When the blue spike occurs the magnetorquer is switched off and when the purple spike occurs the magnetorquer is switched on.

7-2-2 Torque direction

As mentioned at the beginning of this section it is very important that all three magnetoquers are torquing in the same direction (i.e. positive or negative) as commanded. When producing the magnetorquers the copper wire is wounded around the mu-metal core and around the assembly for the Z axis. The beginning and end of this copper wire are both put in a 2 pins connector. For the torque direction it is important that for each magnetorquer the wires are put in the same order into the connector. If this is not the case one magnetorquer could for example torque in the negative direction when a positive direction is commanded.
To verify the torque direction the ADCS is placed on the table inside the Helmholtz cage parallel to the reference magnetometer. One by one the magnetorquers are commanded to first torque in the negative direction and then in the positive direction. At the same time the reference magnetometer measures the present magnetic field. The results of these measurements are shown in figures 7-12, 7-13, 7-11.

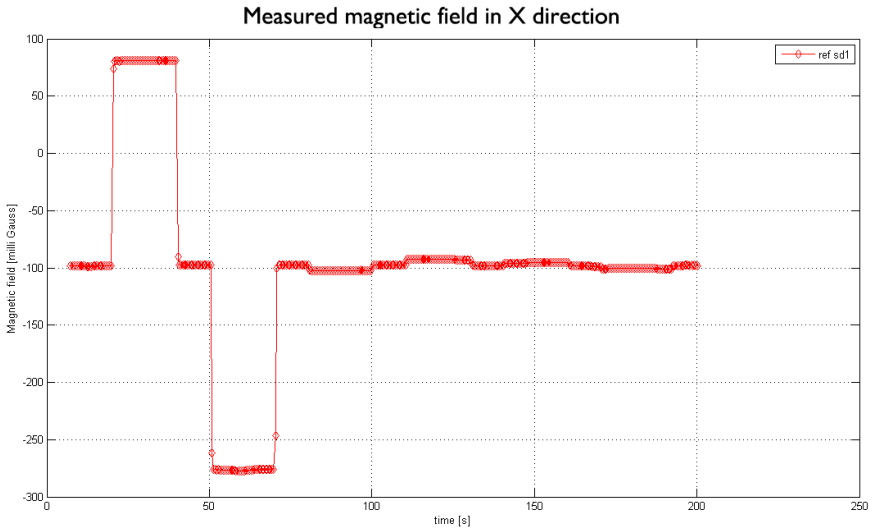


Figure 7-11: Measured magnetic field in X direction during magnetorquer direction test.

From these figures we can see that in the X direction the magnetic field first increases and then decreases while in the Y and Z direction the magnetic field first decreases and then increases. Since we commanded the magnetorquers to torque in the negative direction first and then in the positive direction the results for the Y and Z direction are correct and thus the magnetorquer in the X direction is torquing in the opposite direction of what is commanded. This could be solved by swapping the pins in the connector, but it was chosen to change

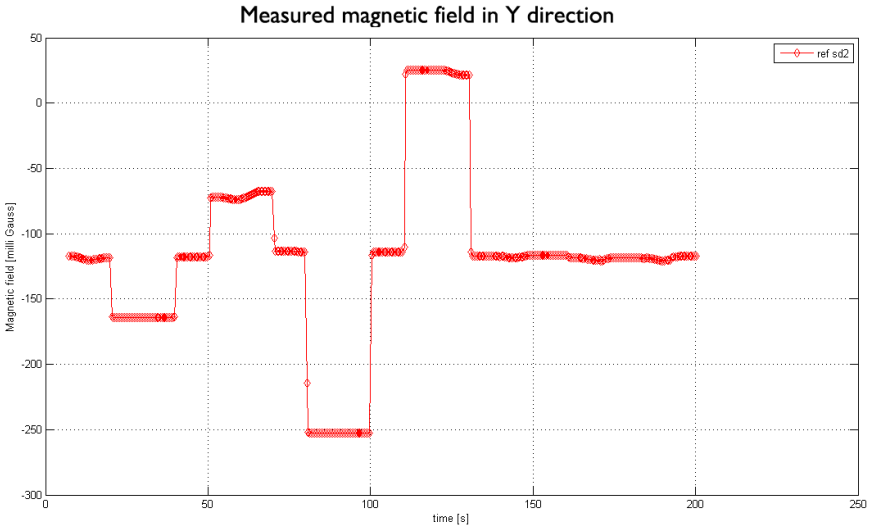


Figure 7-12: Measured magnetic field in Y direction during magnetorquer direction test.

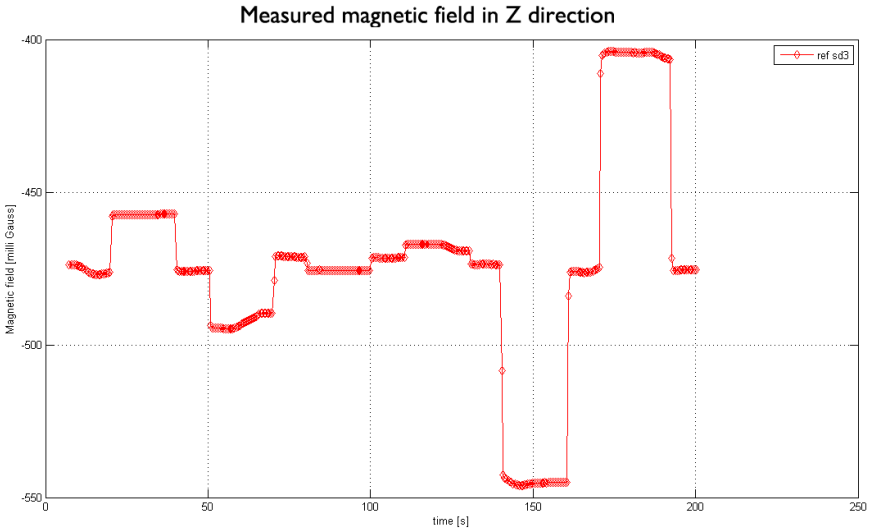


Figure 7-13: Measured magnetic field in Z direction during magnetorquer direction test.

it in the ADCS software. Thus when the algorithm says we need to torque in the positive direction, we command the magnetorquer in the negative direction, and vice versa.

7-2-3 Remanence

The remanence of the magnetorquers can influence the magnetometer measurements and should therefore be zero or at least very low. To see if the magnetorquers have any remanence we use the test results from the previous test, but instead of measurement data from the reference magnetometer we use the measurement data from the magnetometer on the ADCS, since this is the magnetometer that should not be influenced.

What we expect to see, if the remanence is zero, is that the magnetic field measurements before switching on the magnetorquers and after switching them off again are more or less equal. Figures 7-14, 7-15, 7-16 show the test results.

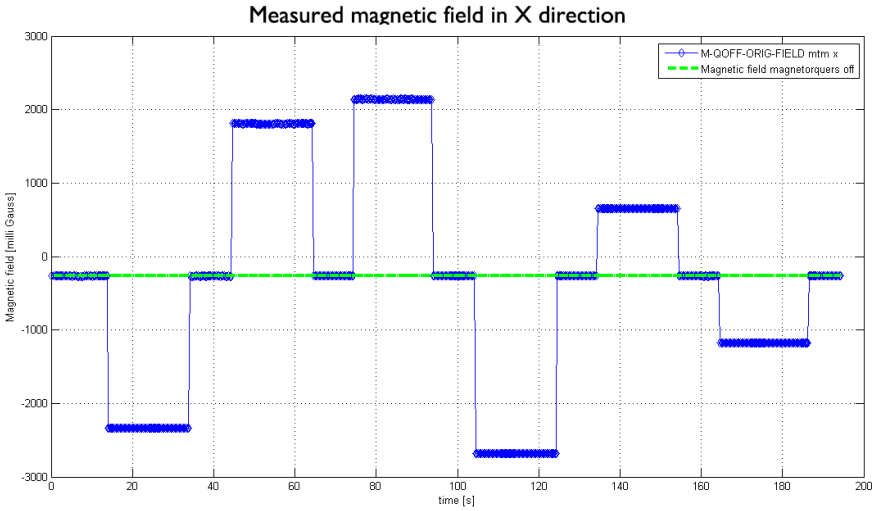


Figure 7-14: Measured magnetic field in X direction during the remanence test and the field error due to magnetization effects of the magnetorquers.

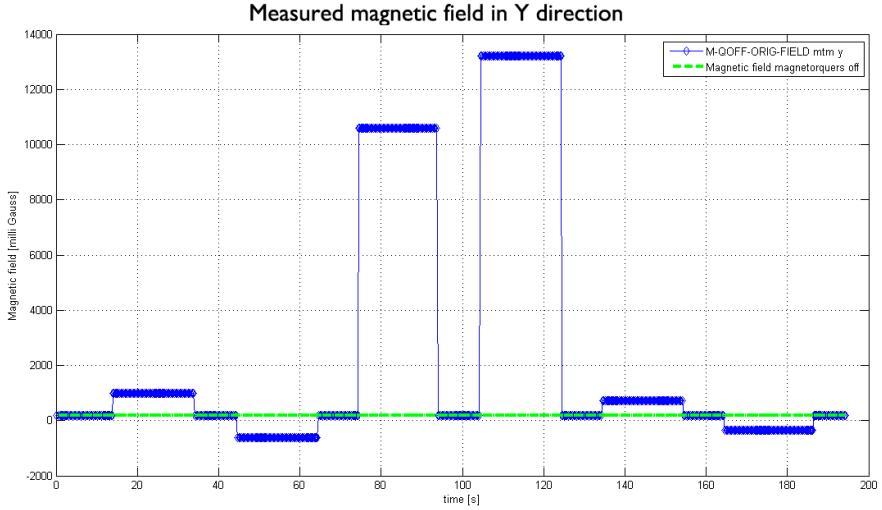


Figure 7-15: Measured magnetic field in Y direction during the remanence test and the field error due to magnetization effects of the magnetorquers.

In these figures the blue lines are the magnetometer measurements, the green lines are the reference magnetic field when the magnetorquers are off. We can see that the magnetometer measurements, before switching on the magnetorquers and after switching them off, are more or less equal. From these test results we can conclude that the remanence is zero, or at least close enough to zero, to have no significant influence on the magnetometer measurements. The magnetorquers therefore satisfy the requirement of a maximum residual magnetic dipole of $0.005Am^2$, which was stated in [43].

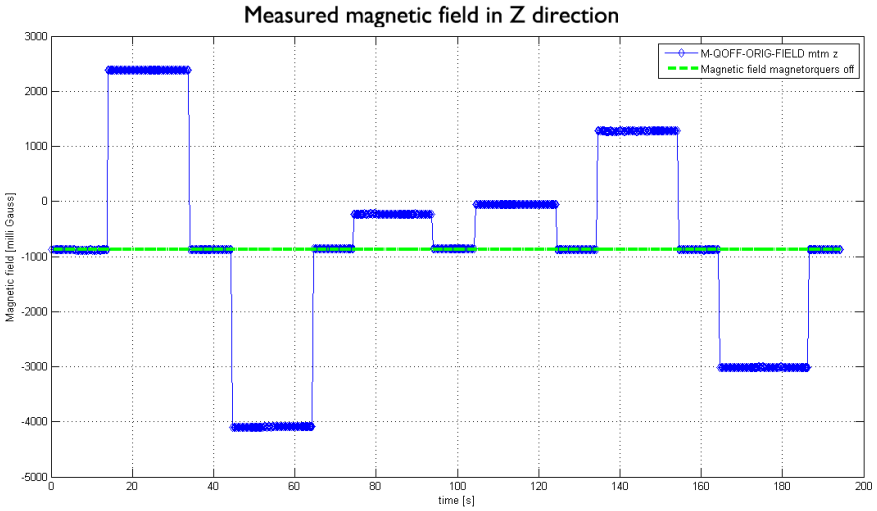


Figure 7-16: Measured magnetic field in Z direction during the remanence test and the field error due to magnetization effects of the magnetorquers.

7-2-4 Magnetic dipole moment

To determine the magnetic dipole of the magnetorquers, each magnetorquer is placed on three different distances from the reference magnetometer in the Helmholtz cage. At each distance, two measurements are done. The first measurement is a period of 2 minutes where the magnetic field is measured when the magnetorquer is switched off. The second measurement is period of 2 minutes where the magnetic field is measured when the magnetorquer is switched on. For both measurements the average magnetic field is determined. The averages are subtracted from each other and the length is determined to get the difference in magnetic field strength. From the difference in magnetic field strength we can determine the magnetic dipole with Eq. (7-1), which is derived from equations in [42].

$$B_{rad} = \mu_0 \frac{2m_{mtq}}{4\pi r^3} \tag{7-1}$$

Here is B_{rad} the difference in magnetic field strength, μ_0 the permeability of free space, r the distance from magnetorquer to the reference magnetometer and m_{mtq} is the magnetic dipole moment we want to determine. If we rewrite Eq. (7-1) for the magnetic dipole moment we get Eq. (7-2).

$$m_{mtq} = \frac{4\pi r^3 B_{rad}}{2\mu_0} \tag{7-2}$$

For each magnetorquer the measurement data and the calculated magnetic dipole moments are shown in Tables 7-1, 7-2 and 7-3. From these tables it is clear that all three magnetorquers meet the requirement, stated in [43], that the minimum delivered magnetic dipole should be $0.06 Am^2$.

	Magnetorquer off			Magnetorquer on							
Distance	$\boldsymbol{B_x}$	$\boldsymbol{B_y}$	$\boldsymbol{B_z}$	$\boldsymbol{B_x}$	$\boldsymbol{B_y}$	$\boldsymbol{B_z}$	$\boldsymbol{Bdiff_x}$	$\boldsymbol{Bdiff_y}$	$\boldsymbol{Bdiff_z}$	$\boldsymbol{B}$ **magnitude**	m
90,5	91	466	106	489	537	98	398	71	8	404	0.149
140,5	91	470	110	187	477	107	96	7	3	96	0.134
240,5	90	466	108	108	468	109	18	2	1	18	0.130

Table 7-1: Magnetic dipole moment measurements for the X axis magnetorquer. Units: mm, mG and A/m^2

	Magnetorquer off			Magnetorquer on							
Distance	$\boldsymbol{B_x}$	$\boldsymbol{B_y}$	$\boldsymbol{B_z}$	$\boldsymbol{B_x}$	$\boldsymbol{B_y}$	$\boldsymbol{B_z}$	$\boldsymbol{Bdiff_x}$	$\boldsymbol{Bdiff_y}$	$\boldsymbol{Bdiff_z}$	$\boldsymbol{B}$ **magnitude**	m
94,5	90	467	107	548	429	50	458	38	57	463	0.195
144,5	90	471	109	197	466	98	107	5	11	107	0.161
244,5	90	471	109	111	470	107	21	1	2	20	0.153

Table 7-2: Magnetic dipole moment measurements for the Y axis magnetorquer. Units: mm, mG and A/m^2

	Magnetorquer off			Magnetorquer on							
Distance	$\boldsymbol{B_x}$	$\boldsymbol{B_y}$	$\boldsymbol{B_z}$	$\boldsymbol{B_x}$	$\boldsymbol{B_y}$	$\boldsymbol{B_z}$	$\boldsymbol{Bdiff_x}$	$\boldsymbol{Bdiff_y}$	$\boldsymbol{Bdiff_z}$	$\boldsymbol{B}$ **magnitude**	m
59,5	81	461	102	401	255	47	320	206	55	384	0.040
109,5	89	468	108	179	431	97	90	37	11	97	0.064
209,5	90	470	109	107	464	106	17	6	3	17	0.081

Table 7-3: Magnetic dipole moment measurements for the Z axis magnetorquer. Units: mm, mG and A/m^2

7-3 Verification of the magnetometer

Verifying that the magnetometer is working correctly is crucial. Without a magnetometer, which gives reliable measurement data, the performance of detumble can decrease drastically or in worst case do the opposite. Verifying the magnetometer is done in three steps. The first step is the built in self test of the magnetometer. If this gives the correct results, the second step is to generate an increasing magnetic field, up to 500 milli Gauss, in X,Y and Z direction. Finally, if the two previous two steps gave correct results, a rotating magnetic field, covering a whole sphere, is generated. If all the three steps give correct results we can draw the conclusion that the magnetometer is working correctly and gives reliable measurement data. The following subsections give a more detailed description and the results of the three steps described above.

7-3-1 Built-in self test

The magnetometer used on the ADCS, described in Subsection 4-1-1, features a built-in self test. This built-in self test does two measurement cycles. In the first measurement cycle the magnetometer measures the external magnetic field currently present. In the second measurement cycle the magnetometer itself induces a magnetic field with a strength around 1.1 Gauss in each direction and again measures the magnetic field. The first measurement is subtracted from the second, which leaves us with the measured magnetic field value induced by the magnetometer itself. According to the data sheet, with a gain setting of 660 LSB/Gauss, this should yield to values of 766 in the X/Y data registers and a value of 713 in the Z data register. These values correspond, according to equation (4-1), with 1.16 Gauss for the X/Y direction and 1.13 Gauss for the Z direction. In the graph shown in Figure 7-17 we can see the measured magnetic field in the X direction due to a number of consecutive self tests. This graph and more graphs of the self test can be found in Appendix D.

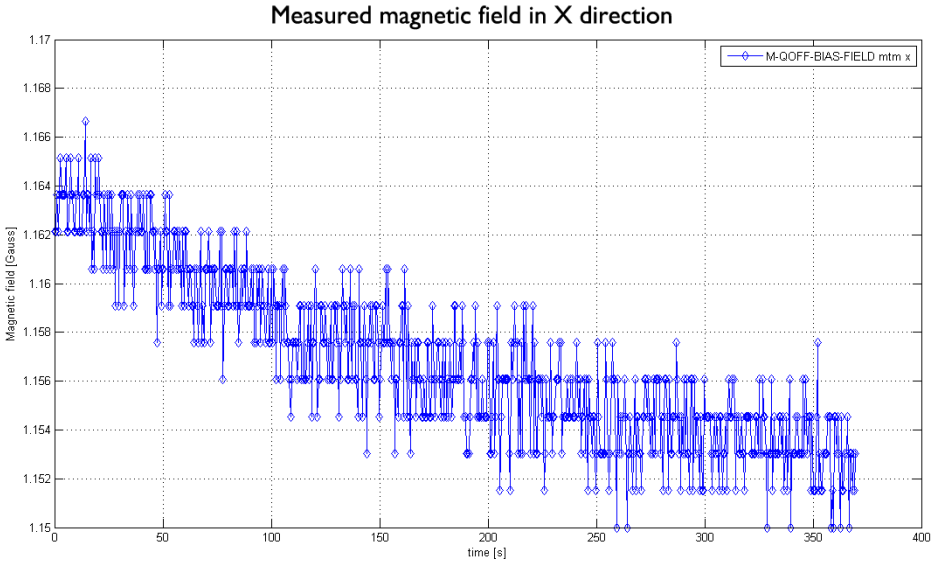


Figure 7-17: Magnetic field in the X direction due to the built-in self test.

From Figure 7-17 we can see that in the first set of self tests the magnetic field strength in the X direction is indeed around 1.16 Gauss as specified in the data sheet. More to the end we can see that the measured field slowly drops to around 1.15 Gauss. This can be explained by the fact that the temperature of the magnetometer is increasing a little and therefore influencing the measurements. In normal operation differences in temperature will also influence the measurements. Therefore we need to compensate for this effect by simply doing a self test for before each normal measurement. We know that during a self test the measured magnetic field in the X direction should be around 1.16 Gauss. If the actual measured magnetic field during a self test is 1.15 Gauss, then we need to scale the normal measurement according to Eq. (7-3).

$$\boldsymbol{B}_{actual_i} = \left(\frac{\boldsymbol{B}^{ST}_{theory}}{\boldsymbol{B}^{ST}_{meas_i}}\right)\boldsymbol{B}_{meas_i} = \left(\frac{1.16}{1.15}\right)\boldsymbol{B}_{meas_i} \tag{7-3}$$

Where $\boldsymbol{B}_{actual_i}$ is the actual magnetic field after compensating for temperature differences, $\boldsymbol{B}^{ST}_{theory}$ and $\boldsymbol{B}^{ST}_{meas_i}$ the theoretical and measured magnetic field value during a self test and $\boldsymbol{B}_{meas_i}$ the measured magnetic field value in normal operation.

7-3-2 Increasing magnetic field

Step two in verifying the correct working of the magnetometer is doing an incremental test. In this test the magnetic field is first slowly decreased down to -500 milli Gauss, then directly set back to 0 and then slowly increased up to 500 milli Gauss and again directly set back to 0. This is done for the three principle directions, beginning with the X direction, then the Y direction and as last the Z direction. What we would expect is that the magnetometer measurements would more or less correspond to the magnetic field generated by the Helmholtz cage. Figures 7-18, 7-19 and 7-20 show the results of the first incremental test.

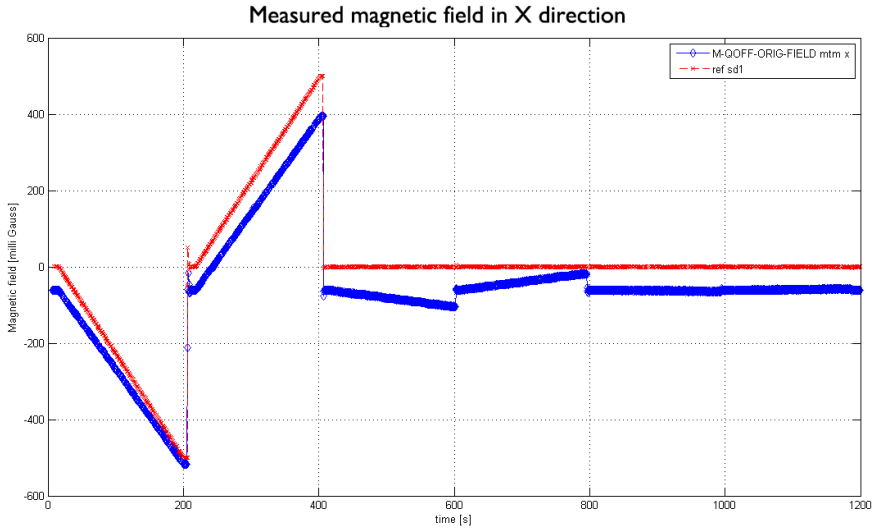


Figure 7-18: Magnetic field in the X direction.

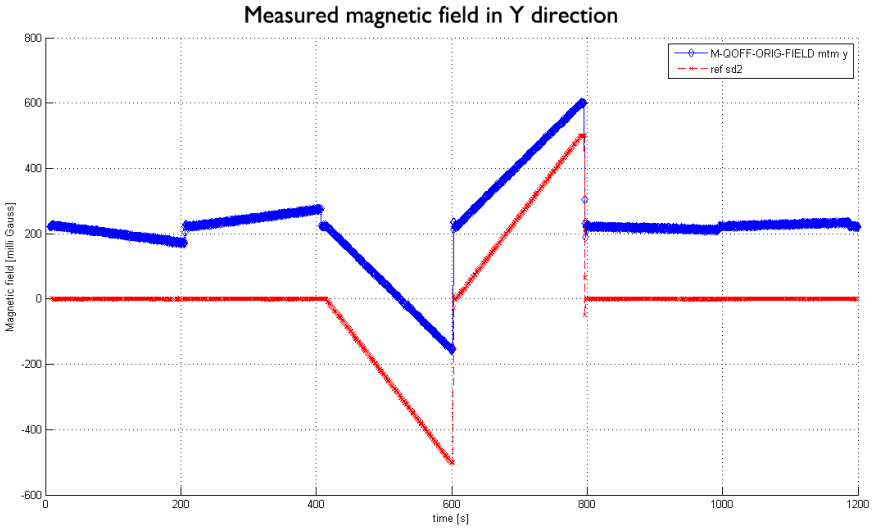


Figure 7-19: Magnetic field in the Y direction.

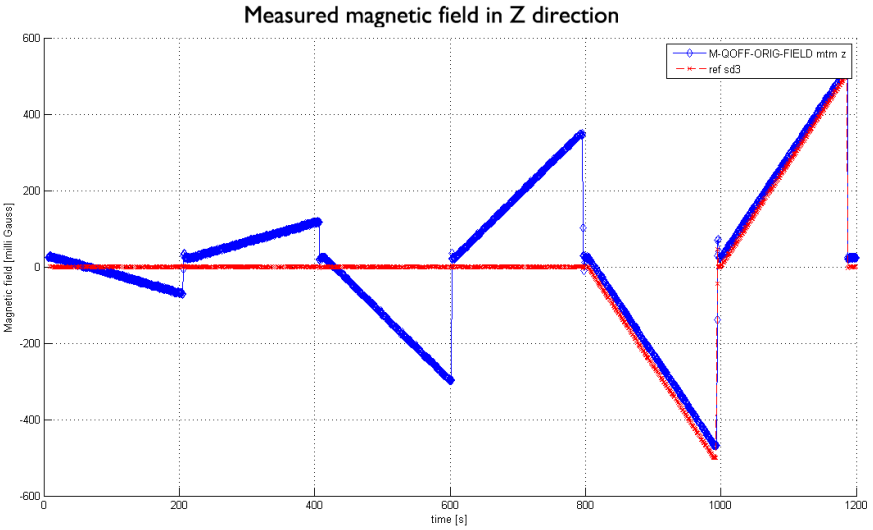


Figure 7-20: Magnetic field in the Z direction.

It is clear that the results from Figures 7-18, 7-19 and 7-20 is not what we expected. The first thing we notice from these figures is the large offset with respect to the reference magnetometer. The large offset can be explained by an electronic bias that is present in almost every magnetometer after production. This electronic bias is constant and can easily be compensated for by adding a constant value to every measurement.

$$\boldsymbol{B}_{actual_i} = \boldsymbol{B}_{meas_i} + \boldsymbol{B}_{bias} \tag{7-4}$$

Another explanation would be the presence of other external magnetic fields caused by materials that add a constant distortion to the magnetic field measured by the magnetometer. For example a constant current through a wire generates a magnetic field. If this wire is located near the magnetometer, it will add a constant distortion to the measured magnetic field.

These kinds of distortions are generally known as 'hard iron distortions' and are discussed in several articles [44],[45], [46] and [47]. As long as this distortion caused by external sources is constant it can also easily be compensated with Eq. (7-4). In Figure 7-22 an example of hard iron distortion is depicted.

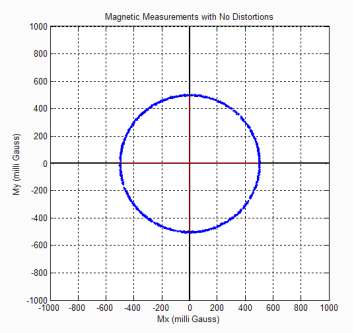


Figure 7-21: A magnetic field with no distortion.

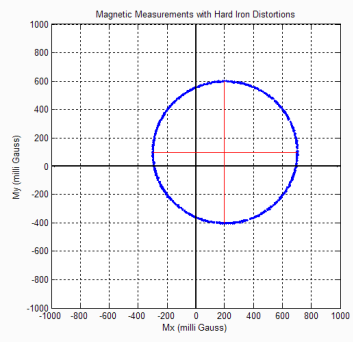


Figure 7-22: A magnetic field with hard iron distortion.

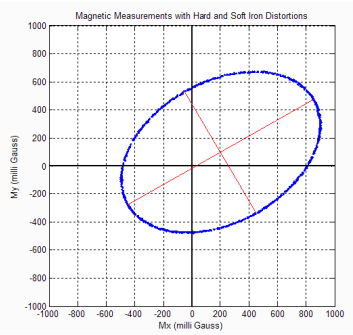


Figure 7-23: A magnetic field with soft iron distortion.

Furthermore Figures 7-18, 7-19 and 7-20 show that if in the Helmholtz cage a magnetic field in a direction of a principle axis is generated, the magnetometer not only measures a magnetic field component on that axis, but also on one of the other principle axes. This behaviour is also different from what was expected. Some additional tests, to exclude possible causes, it became clear that the mu-metal core in the magnetorquers was distorting the magnetic field. The distortion changed the direction of the magnetic field, which explains why there are magnetic field components on axes different than the direction of the magnetic field generated by the Helmholtz cage.
These kinds of distortions are generally known as 'soft iron distortions' and are discussed in several articles [44],[45], [46], and [47]. Soft iron distortions are similar to hard iron distortions except that the distortion added to the present magnetic field is not constant, but is dependent on the direction of the soft iron material (i.e. the mu-metal core in the magnetorquers) relative to the present magnetic field. In Figure 7-23 an example of soft iron distortion is depicted. Soft iron distortions can not be compensated by adding a simple constant to each measurement, but require a somewhat more complicated method. Eq. (7-5) shows the method to compensate for soft iron distortions.

$$\boldsymbol{B}_{actual_i} = \boldsymbol{A}\boldsymbol{B}_{meas_i} \tag{7-5}$$

Where $\boldsymbol{A}$ is a 3x3 matrix with coefficients C_{ij}, as depicted in Eq. (7-6).

$$\boldsymbol{A} = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \tag{7-6}$$

The coefficients of the matrix $\boldsymbol{A}$ can be determined mathematically or by trial and error. Mathematically will probably give a more precise value for the coefficients, but with the

Matlab application, shown in Figure 7-5, it is relatively easy to determine the coefficients by trial and error. Therefore we have chosen this way of determining the coefficients.
Combining the compensations for the hard iron and soft iron distortions (i.e. Eq. (7-4) and Eq. (7-5)) will result in the following transformation.

$$\boldsymbol{B}_{actual_i} = \boldsymbol{A}(\boldsymbol{B}_{meas_i} - \boldsymbol{B}_{bias}) \tag{7-7}$$

If we apply the transformation from Eq. (7-7) to the measurements from Figures 7-18, 7-18 and 7-18 we get the graphs shown in Figures 7-24, 7-25 and 7-26. From these graphs we can see that applying a transformation like Eq. (7-7), with the right bias and C_{ij} values, can compensate for the hard and soft iron distortions.

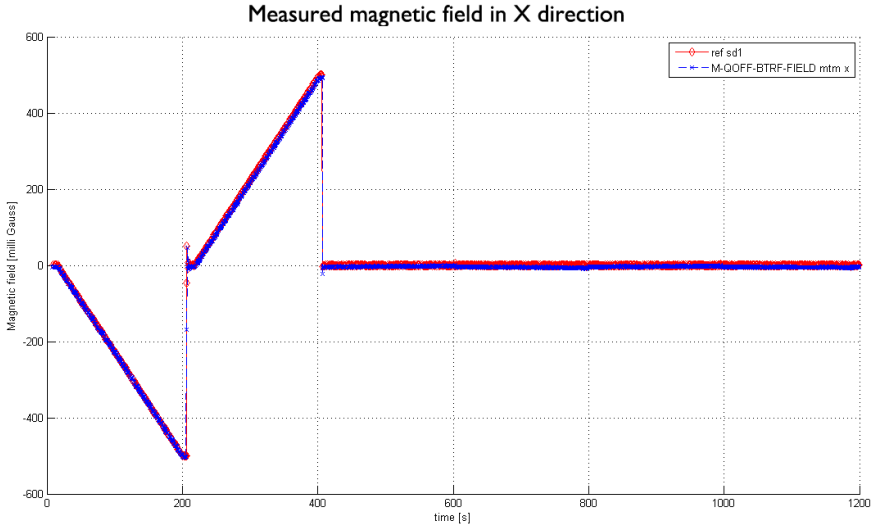


Figure 7-24: Magnetic field in the X direction transformed with Eq. (7-7).

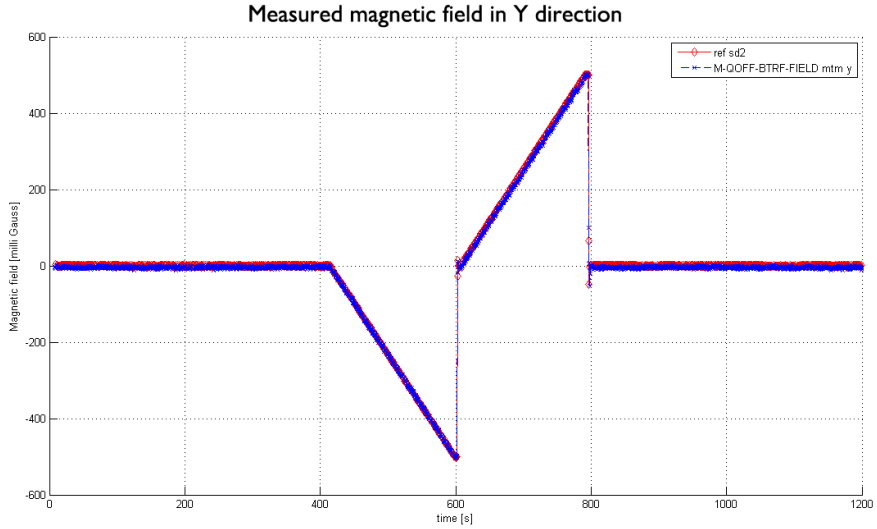


Figure 7-25: Magnetic field in the Y direction transformed with Eq. (7-7).

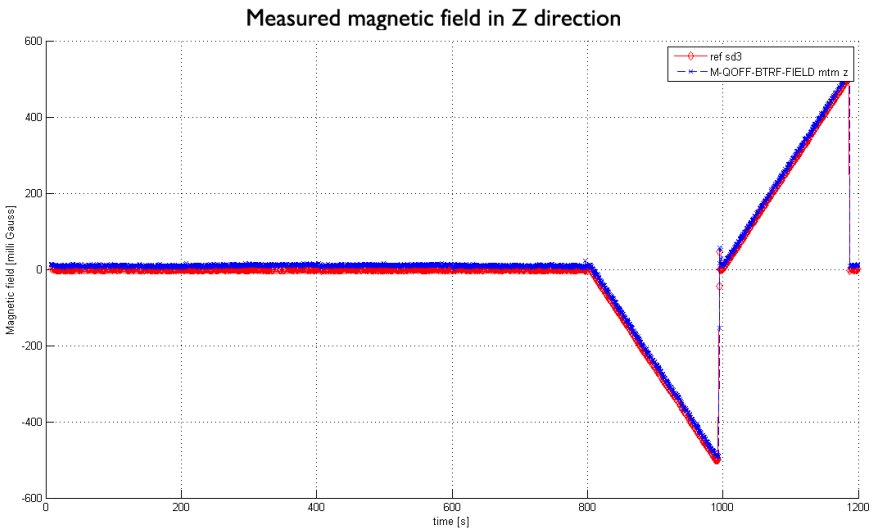


Figure 7-26: Magnetic field in the Z direction transformed with Eq. (7-7).

7-3-3 Rotating magnetic field

In the previous step we have verified that we can compensate for the distortion induced by the magnetorquers, by applying the transformation from Eq. (7-7), and is valid for magnetic fields of different strengths in X,Y and Z direction. The third and final step is verifying whether applying this transformation is also valid for a magnetic field with a random direction. To verify this we will generate several magnetic fields in the Helmholtz cage, each with a different direction, and rotate them around one of the principle axis. This will create sphere that covers many magnetic field directions. Figure 7-27 shows the resulting sphere before applying the transformation.

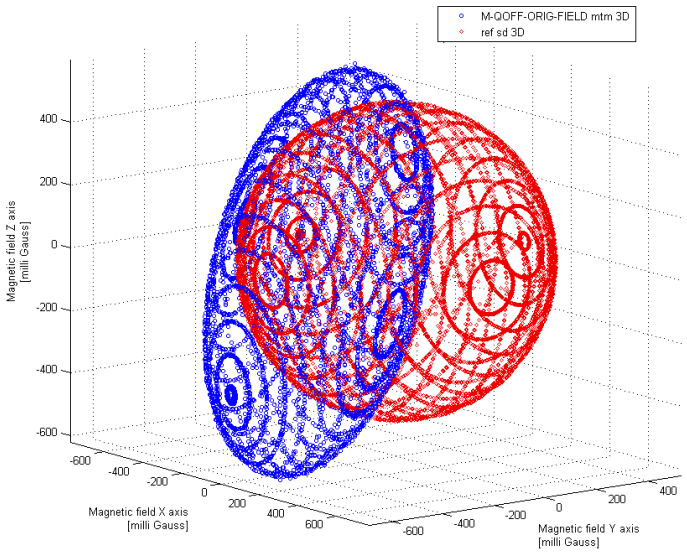


Figure 7-27: The magnetic field before transformation.

In Figure 7-27 we can clearly see that the magnetic field is distorted by the magnetorquers. If we then apply the transformation from Eq. (7-7) we get the resulting graph shown in Figure 7-28. From this graph we can conclude that the applied transformation correctly compensates the hard and soft iron distortions for a random magnetic field direction.

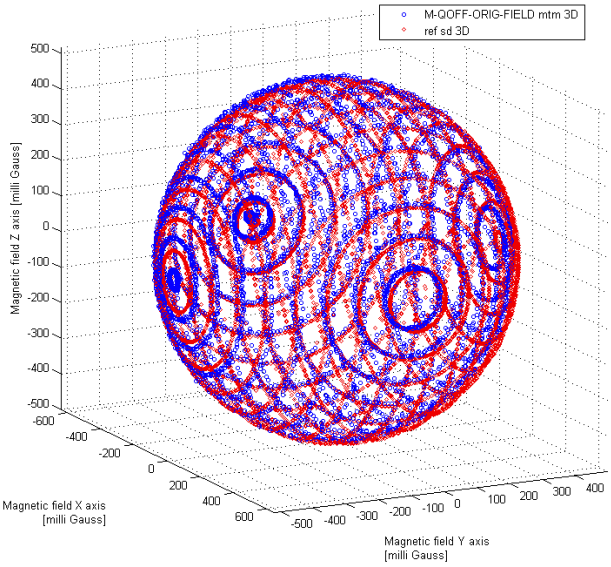


Figure 7-28: The magnetic field after transformation.

7-4 Verification of the detumble mode

Now that we have verified that the magnetorquers and magnetometer are working correctly we can start testing the detumble mode. Testing detumble mode involves several steps. First needs to be verified that the the Bdot is calculated correctly and that the behaviour of the tumble parameter is correct. Secondly, if the Bdot and tumble parameter are both verified, the calculated dipole needs to be verified and whether the magnetorquers are torquing corresponding to this dipole. Finally if all results seem correct a final duration test is performed to see whether the satellite detumbles and switches autonomously to Coarse Sun Pointing (CSP) mode and back to detumble mode if the tumble rate exceeds the tumble threshold.
The following subsections describe and present the results of the above mentioned steps.

7-4-1 Bdot

Verifying the Bdot is done with a simple rotation test. In the Helmholtz cage a rotating magnetic field around the X axis is generated with a strength of 500 milli Gauss in the Y direction. This should induce a changing magnetic field, following a sine wave, in the Y and Z direction and in the X direction it should ideally remain zero. The rotational rate of the magnetic field starts at 10 degrees/s and is decreased in several steps downto 1 degree/s (i.e. 10,5,2 and 1 degree/s). Since the Bdot is basically the time derivative of the magnetic field we expect to see a sine wave with a decreasing amplitude and period equal to that of the

rotating magnetic field. Figures 7-29, 7-30 and 7-31 show the measured magnetic field and the Bdot in the X, Y and Z direction.

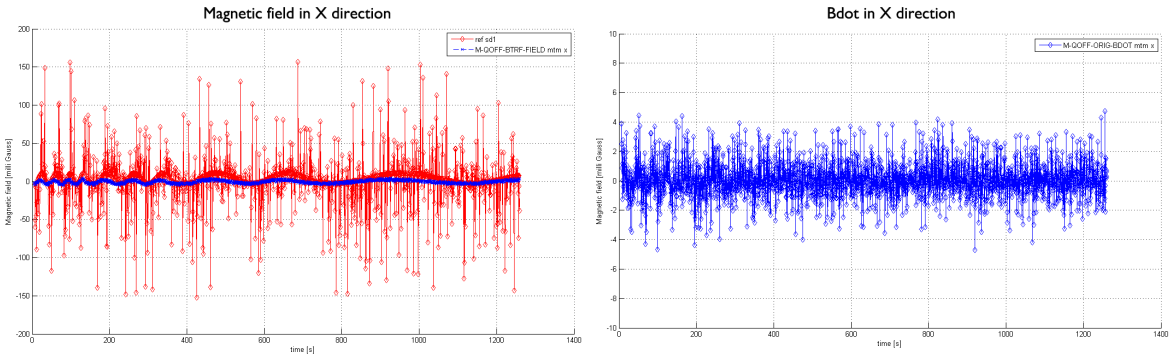


Figure 7-29: LEFT: Magnetic field in the X direction measured by the magnetometer on the ADCS and by the reference magnetometer, RIGHT: the Bdot in the X direction

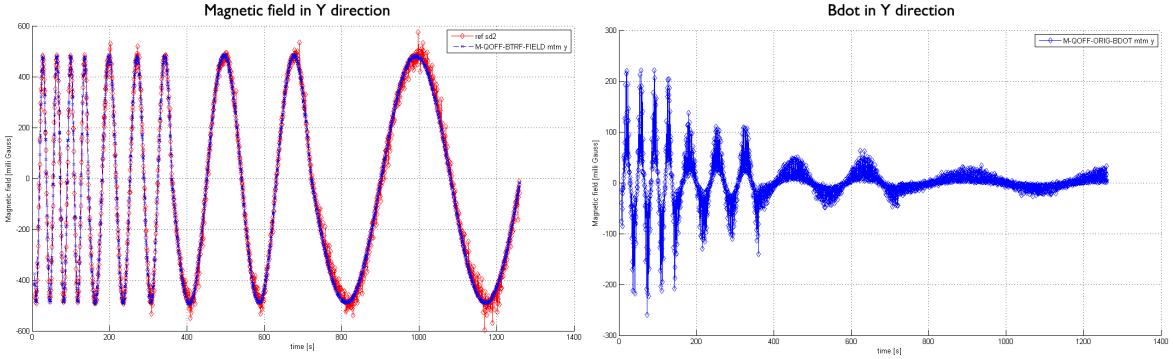


Figure 7-30: LEFT: Magnetic field in the Y direction measured by the magnetometer on the ADCS and by the reference magnetometer, RIGHT: the Bdot in the Y direction

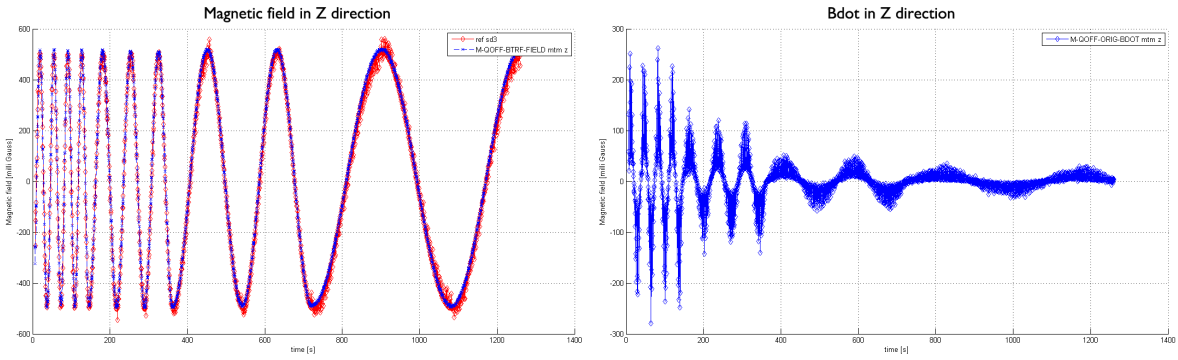


Figure 7-31: LEFT: Magnetic field in the Y direction measured by the magnetometer on the ADCS and by the reference magnetometer, RIGHT: the Bdot in the Y direction

From the figures 7-29, 7-30 and 7-31 we can see that the measured magnetic field in Y and Z direction from the magnetometer on the ADCS closely corresponds to that of the reference magnetometer and the magnetic field in X direction fluctuates a bit around zero. The heavy fluctuations that are visible on the reference magnetometer data in the X direction and a

little in the Y and Z direction, are the effects of the magnetorquers switching on and off. This will be discussed in more detail at the end of next section. If we look at the Bdot in X direction we only see small fluctuations of around +/-4 milli Gauss, which can be considered measurement noise and is below the maximum noise of 20 milli Gauss (i.e $2\mu T$) determined in [43]. The Bdot in Y and Z direction, as expected, looks like a sine wave with decreasing amplitude and a period equal to that of the rotating magnetic field. However, there is also quite some noise visible, in the order of 30-40 milli Gauss. This amount of noise is higher then the maximum of 20 milli Gauss and is not desirable, because it will cause unnecessary and maybe even actuation of the magnetorquers in the wrong direction. To reduce this noise a low-pass filter, given in Eq. (7-8), is applied to the Bdot.

$$\boldsymbol{Bdot}_{lpf_i} = \alpha_{bdot}\boldsymbol{Bdot}_i + (1 - \alpha_{bdot})\boldsymbol{Bdot}_{i-1} \tag{7-8}$$

In Eq. (7-8) the parameter α_{bdot} is the filter constant and is chosen to be between 0.4 and 0.5. Any larger or smaller value for α_{bdot} will inherit to much noise from the previous or from the current Bdot. Figure 7-32 shows the result of applying the low-pass filter, with $\alpha_{bdot} = 0.4$, to the Bdot in Y direction. This graph and graphs of the other directions are given in Appendix D.

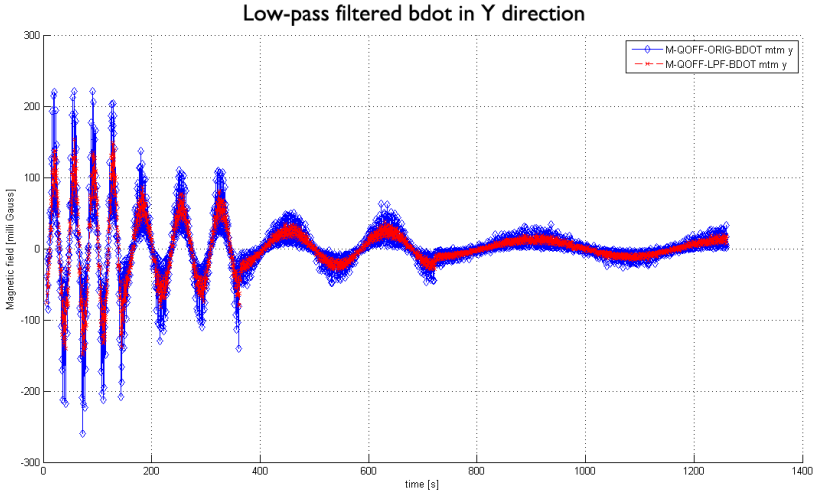


Figure 7-32: The low-pass filtered Bdot in the Y direction.

It is clear from Figure 7-32 that the noise is drastically reduced to below the maximum of 20 milli Gauss. This noise reduction should minimize the unnecessary or wrong actuation.

7-4-2 Tumble parameter

Since the tumble parameter is an indication of the tumbling rate of the satellite, it should increase or decrease depending on how fast the satellite is rotating. Every time the detumble mode is entered, at start up or via a mode switch, the tumble parameter is set to the value of P_{tumble_0}, which is determined in [3] to be the value where the satellite has a rotational rate below 1 degree/s. To verify whether the behaviour of tumble parameter is correct we use the same rotating magnetic field, described in the previous section. The tumble parameter

should first increase for a certain period and then start to decrease when the rotational rate of the magnetic field is decreased down to 1 degree/s. Depending on the noise level of the magnetometer, it will keep decreasing until a certain value where the noise on the magnetometer is larger than the Bdot length. Figure 7-33 shows the behaviour of the the tumble parameter when a same magnetic field as in figures Figure 7-29, Figure 7-30 and Figure 7-31 is generated.

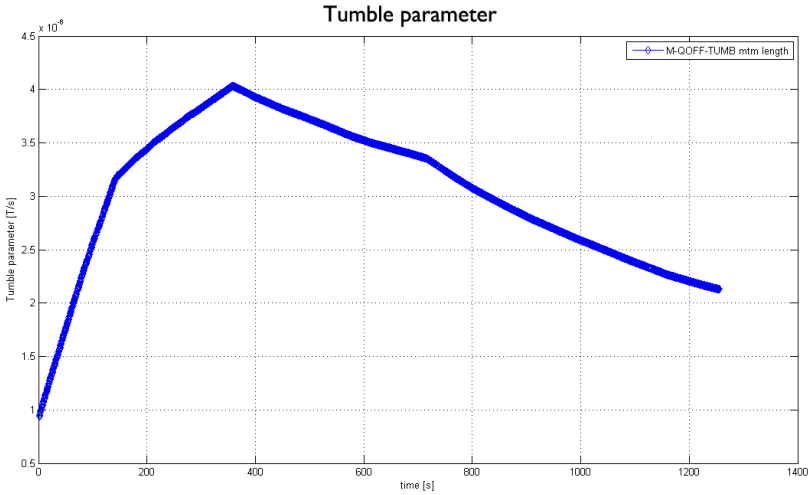


Figure 7-33: The tumble parameter calculated from the magnetic field and Bdot from figures Figure 7-29, Figure 7-30 and Figure 7-31.

From Figure 7-33 it can be seen that the behaviour of tumble parameter is correct. When the rotational rate of the magnetic field is 10 degrees or 5 degrees/s the tumble parameter increases. With a rotational rate of 2 degrees/s the tumble parameter is already decreasing slowly and at 1 degree/s is starts to decrease more rapidly. If a longer test was performed then the tumble parameter should at least have decreased to somewhere below P_{tumble_0}. This will be later verified in the detumble duration test.

7-4-3 Magnetic dipoles, commanded dipoles and actuation

The magnetic dipoles that are needed to counteract any rotation of the satellite and decrease its tumble rate is calculated with the Bdot control law described in [3] and shown in Eq. (3-71). According to this law the magnetic dipoles m, needed to counteract rotation of the satellite, is the Bdot multiplied by a negative gain constant k_B. Therefore, the magnetic dipoles are a scaled version of the Bdot with opposite sign. The constant k_B is depended on the maximum magnetic dipole, m_{max}, a magnetorquer can deliver. In our tests the constant k_B was set to the value 60000, which was determined in [3], taking into account a m_{max} for a magnetorquer of $0.06 A/m^2$. From the calculated magnetic dipoles, the actual commanded dipole, m_{cmd}, for each magnetorquer is calculated with Eq. (7-9).

$$m_{cmd} = \begin{cases} m_{max} t_{frac} & \text{if } \frac{m}{m_{max}} \geq t_{frac} \\ \frac{m}{m_{max}} t_{frac} & \text{if } \frac{m}{m_{max}} < t_{frac} \end{cases} \tag{7-9}$$

Where t_{frac} indicates the maximum fraction of time, from a sync period, the magnetorquers can be switched on. In our tests t_{frac} was approximately 0.2. In Figures 7-34 and 7-35 the calculated magnetic and commanded dipole for the Y direction are shown. This graph and graphs for the X and Z direction are also shown in Appendix D.

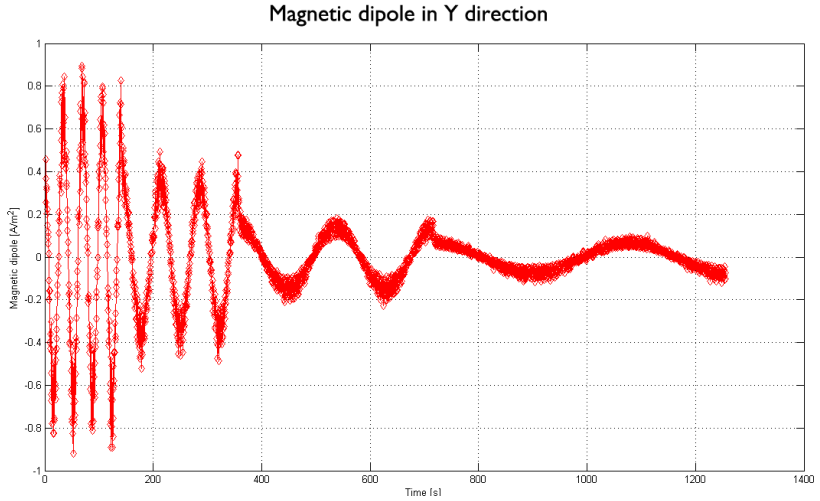


Figure 7-34: The calculated magnetic dipole in the Y direction

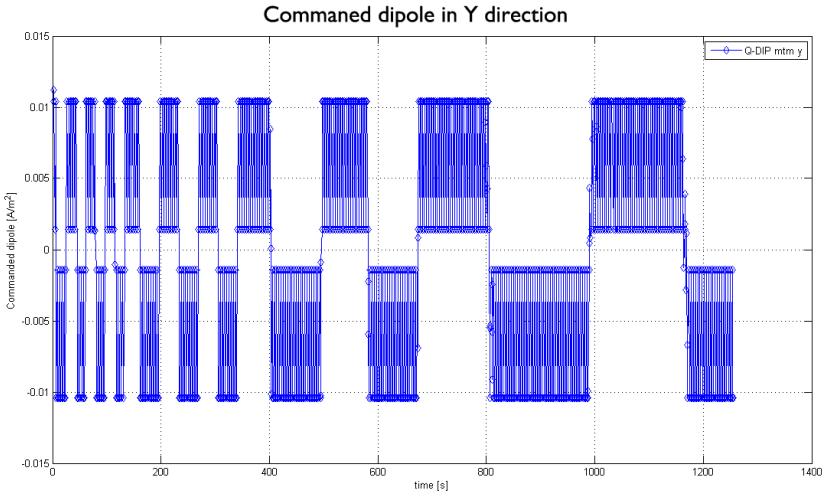


Figure 7-35: The calculated commanded dipole in the Y direction

From this two graphs, together with Figure 7-32, we can see that the magnetic dipole is indeed a scaled version of the Bdot and has an opposite sign (i.e. where the Bdot is positive the magnetic dipole is negative and vice versa). If we take some magnetic dipole values from the graph in Figure 7-34 and plug them into Eq. (7-9) we can also verify that the calculated commanded dipole is correct.
To verify whether the magnetorquers are torquing corresponding to the calculated commanded dipole, a closer look is taken at the fluctuations on the reference magnetometer data mentioned in the previous section.Figure 7-36 shows a close up of the left graph from Figure 7-30.

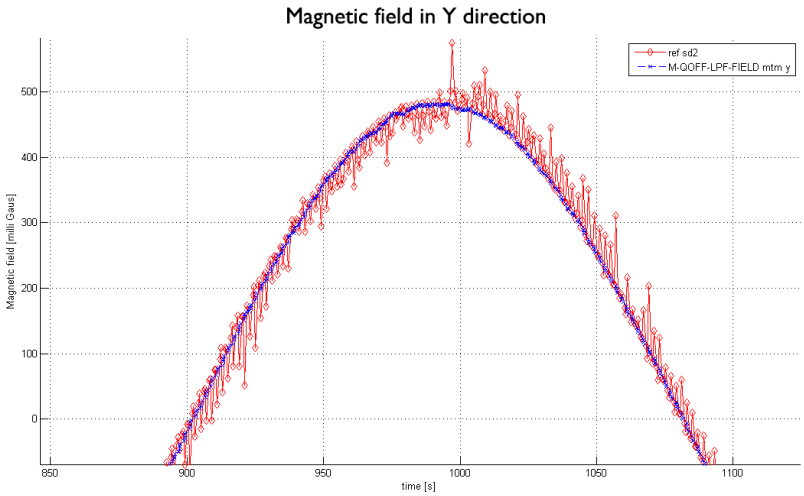


Figure 7-36: Close-up of the magnetic field in the Y direction measured by the magnetometer on the ADCS and by the reference magnetometer.

We can see from Figure 7-36 that if the magnetic field increases (i.e. the Bdot is positive) the fluctuations are in the negative direction. According to the Bdot control law we know that the magnetic dipole is the opposite of the Bdot and therefore negative. This will also result in a negative commanded dipole. Hence, the negative fluctuations we see on the reference magnetometer data correspond with the commanded dipole and thus can be concluded that the magnetorquer is torquing correctly. The same reasoning can be done for a decreasing magnetic field.

7-4-4 Detumble duration test

In the previous sections we have verified that all parts (i.e. Bdot, tumble parameter, dipoles and actuation) of the detumble mode work correctly. The final test that needs to be performed is a duration test. With this test we want to verify that the detumble mode is able to fully detumble the satellite and that the ADCS switches autonomously to an advanced mode (i.e. CSP mode). Once in CSP mode the ADCS must also switch back autonomously, when the tumbling rate exceeds a certain threshold $L_{tumbling}$.
For this test we generated a same rotating magnetic field as with the previous tests, but at we let it run for a longer period at 1 degree/s until it switches to CSP mode. Once it has switched to CSP mode we start to increase the rotational rate of the magnetic field again. What we expect to see, is a tumble parameter that is decreasing to below the $L_{detumbled}$ threshold and after a period of $t_{detumbled}$ staying below this threshold it should switch to CSP mode. Once we start to increase the rotational rate of the magnetic field again, the tumble parameter should increase and when it exceeds the threshold $L_{tumbling}$ it should switch back to detumble mode. In Table 7-4 the values for the most important detumble variables are listed and Figure 7-37 shows the result of the test.

Parameter	Value
$\alpha_{tumble}[-]$	$1/2000$
$P_{tumble_0}[Ts^{-1}]$	$9,4 \cdot 10e^{-7}$
$P_{tumble_{exit}}[Ts^{-1}]$	$1.0 \cdot 10e^{-7}$
$L_{detumbled}[Ts^{-1}]$	$9,4 \cdot 10e^{-7}$
$L_{tumbling}[Ts^{-1}]$	$6 \cdot 10e^{-7}$
$t_{detumbled}[s]$	1000

Table 7-4: Tumbling variables and threshold used in the detumble duration test

From Figure 7-37 we can see that the tumble parameter shows indeed the expected behaviour. However, the tumble parameter does not get below the threshold $L_{detumbled}$ within reasonable time and it did not look like it would have gotten below this threshold any time soon. To make this test still useful and verify whether it switches autonomously, we increased the threshold $L_{detumbled}$, during the test to $L_{detumbled_{NEW}} = 1.3 \cdot 10e^{-6}$. We can then see that after a period below this new threshold it switches to CSP mode (indicated by the pink line). Once in CSP mode the tumble parameter starts to increase again, because the rotational rate of the magnetic field was increased. As soon as it reaches the threshold $L_{tumbling}$ it switches back to detumble mode.

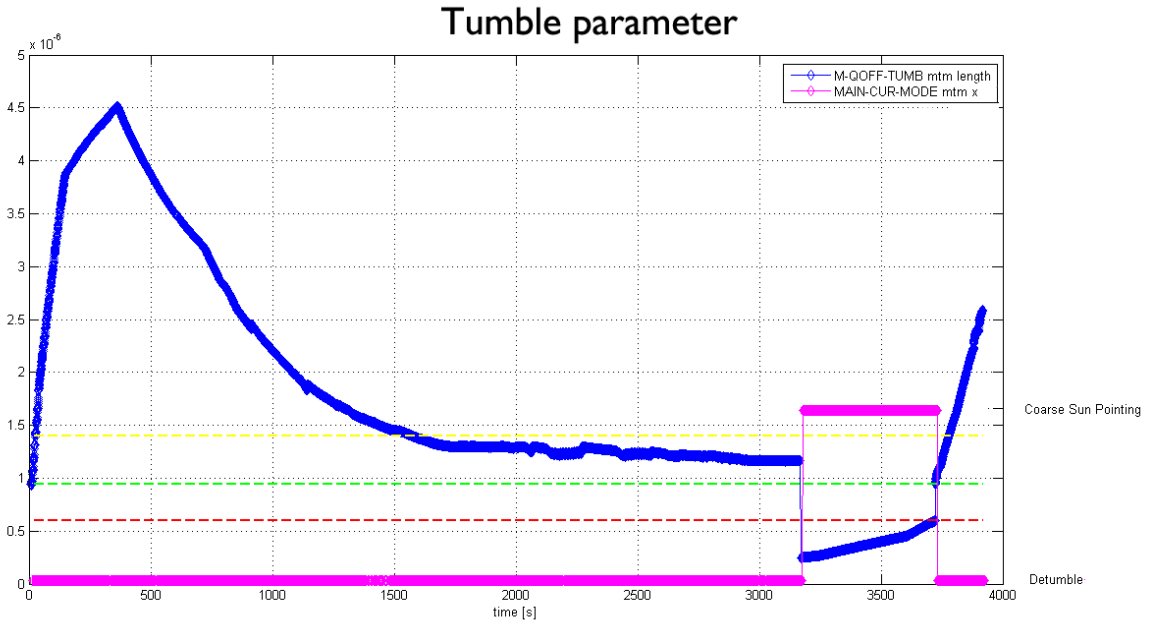


Figure 7-37: Course of the tumble parameter during the detumble duration test.

In the previous test we have seen that detumble mode works and that ADCS is able to switch autonomously back and forth. However, we also saw that the tumble parameter did not get below the threshold $L_{detumbled}$. To verify whether it can reach a value below the threshold $L_{detumbled}$, another test was performed, but this time the rotational rate of the magnetic field was further decreased to 0.1 degree/s. Figure 7-38 shows the course of the tumble parameter during this test.

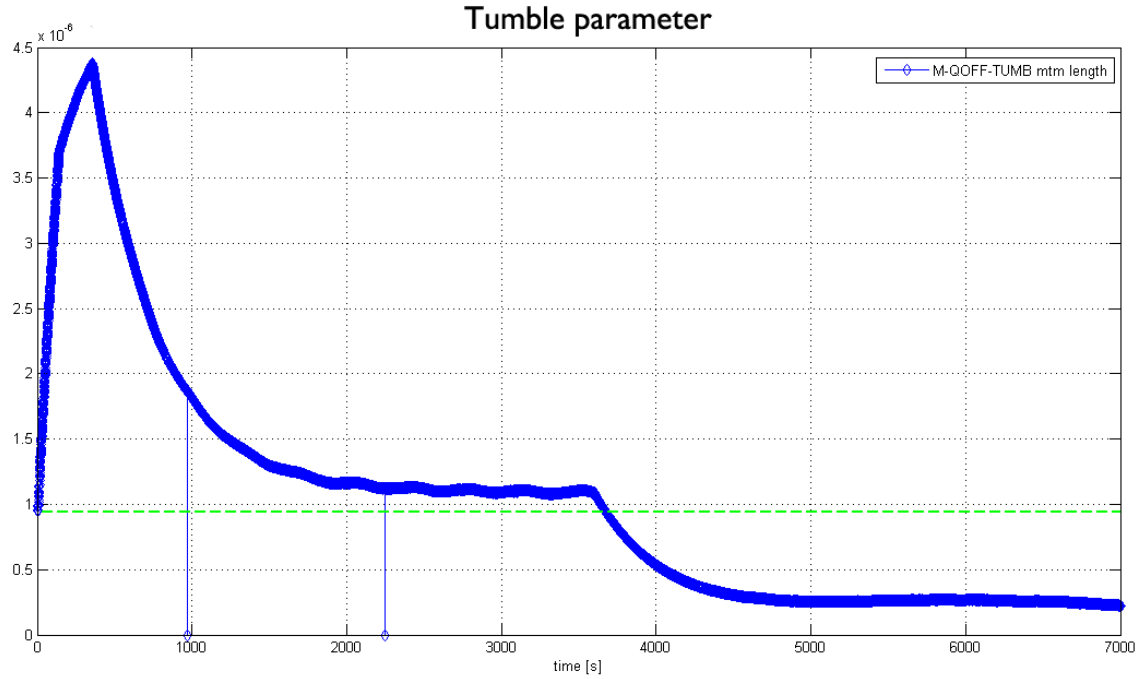


Figure 7-38: Course of the tumble parameter during the second detumble duration test.

Figure 7-38 shows that the tumble parameter does not drop below the threshold $L_{detumbled}$ when the rotation rate of the magnetic field is kept at 1 degree/s, but fluctuates around $1.2 \cdot 10e^{-6}$. When the rotational rate of the magnetic field is further decreased to 0.1 degree/s the tumble parameter drops below the threshold $L_{detumbled}$. The value that is chosen for $L_{detumbled}$ is thus depended on the rotational rate at which we want to switch from detumble mode to an advanced mode. If a value of $1.3 \cdot 10e^{-6}$ is chosen the satellite will still have a tumble rate of around 1 degree/s when it switches to an advanced mode. With a lower value the satellites tumble rate will be below 1 degree/s. For the current baseline of the flight software the value of $L_{detumbled}$ is set to $1.3 \cdot 10e^{-6}$.
Based on the results of these tests and the selected value for $L_{detumbled}$ it can be concluded that the ADCS meets the requirement, stated in [43], to be able to fully detumble the satellite within a day. Estimated is that the ADCS will even perform slightly better than these results indicate, because the noise levels on the magnetometer are expected to be lower.

7-5 Summary

In this chapter the verification procedure for the detumble hardware and software was described, and the results were presented. In the first part of this chapter the magnetorquer hardware and software was verified. It was shown that the magnetorquers switch on and off correctly and that the magnetorquers in Y and Z direction torque in the same direction as commanded, but the magnetorquer in the X direction in the opposite direction. From the results it was also clear that the magnetoruers do not show any remanence that can influence the magnetometer measurements. Furthermore we determined the magnetic dipole each

magnetorquer produces when switched on.
In the second part of this chapter we have verified the magnetometer hardware and software. It was found that hard and soft iron distortions of the magnetic field influenced the magnetometer measurements. After correction for these distortions, the magnetometer measurements showed correct results for static as well as rotation magnetic fields.
The final part of this chapter described verification of complete detumble mode. First the correct behaviour of Bdot and tumble parameter was verified. The Bdot showed some heavy noise, which was reduced by applying a low-pass filter. The tumble parameter showed the correct behaviour. Secondly it was verified that the magnetorquers were actuated correctly corresponding to the calculated magnetic and commanded dipole. Finally two duration tests were done to see whether the ADCS switched autonomously between the detumble mode and an advanced mode. These tests showed that the ADCS switches autonomously back and forth, but it is depended on the thresholds $L_{detumbled}$ and $L_{tumbling}$ at which rotation rate of the satellite this switching is performed.

Chapter 8

Verification of the sun sensors, sun vector determination and reaction wheels

For the advanced modes of the Attitude Determination and Control Subsytem (ADCS), such as Coarse Sun Pointing (CSP), Fine Sun Pointing (FSP), Thruster Pointing (TP) and Ground Station Tracking (GST) it is important that the sun sensors and reaction wheels function correctly. The sun sensors must give a correct output to determine a correct sun vector, which is used in determining the attitude, and the reaction wheels must work correctly in order to control the satellite. Incorrect working of one of these two can decrease the performance of the advanced modes or even make them completely unusable.

This chapter describes the verification of the sun sensors and reaction wheels. First, in Section 8-1 a description of the test facility for the sun sensors is given. Secondly, in Section 8-2 the verification procedure for the sun sensors is described and the results are given. Finally, Section 8-3 describes the verification procedure of the reaction wheels and shows the results.

8-1 Test facility

To verify whether the sun sensors function correctly a test set-up with a good light source was needed. An ideal test set-up would be a solar simulator with a light source that has a spectrum equal to that of the sun, or at least close to that spectrum, and delivers a power output of one solar constant. Building such a solar simulator was however not feasible with the available time and budget. The closest we got to this was a test set-up with a metal halide lamp, which is often used in solar simulators. Preliminary tests with this light source brought up a problem that was not thought of when designing this test set-up . The metal halide lamp is powered with AC instead of DC, therefore making the light flicker with a 50Hz

frequency. This flickering was clearly visible in the sun sensor measurements, which showed large fluctuations, making the measurements not useful. Figure 8-1 shows an example of these fluctuations for one sun sensor quadrant together with a 50Hz sine wave.

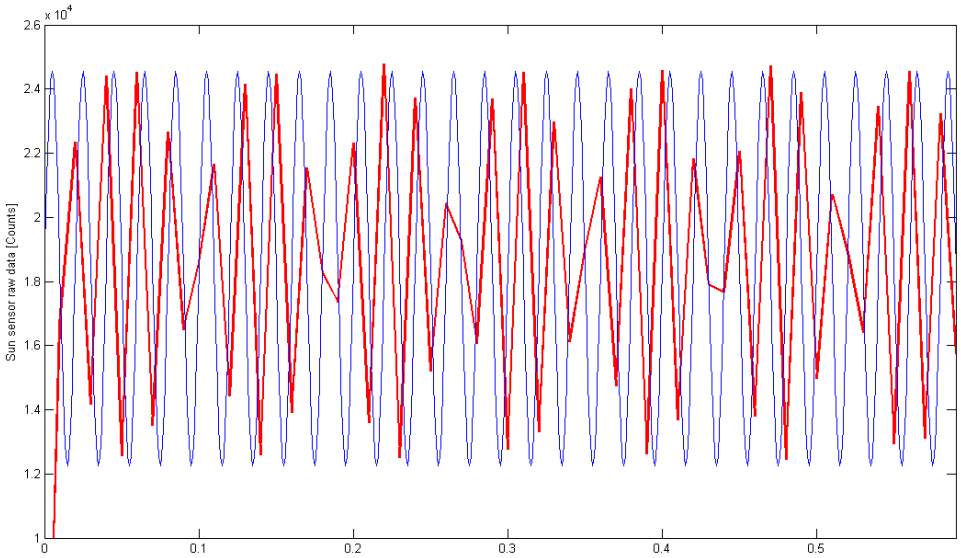


Figure 8-1: Sun sensor measurements from one sun sensor quadrant together with a 50hz sine wave.

To overcome this problem a new, much simpler, test set-up was designed. This test set-up consisted of high intensity LED flash light and a small horizontal levelled platform, adjustable in height, with a round hole in de middle. The sun sensor is placed underneath the platform and the LED light is placed on top of the platform directly over the hole. Both sun sensor and LED light are aligned as precisely as possible to the center of the hole. With this set-up the sun sensor is illuminated perpendicular and homogeneously. To vary the intensity of the light, which illuminates the sun sensor, the height of the platform can be adjusted.

8-2 Sun sensor and sun vector verification

Since a prototype of the sun sensors is already extensively tested on performance in [38] and we lack a good calibrated test set-up we only verify if the functionality of the sun sensors are correct. Verifying whether the sun sensors functions correctly involves doing several tests. First we need to verify if and what the electronic bias of the sun sensors is. Secondly, there must be verified if the four quadrants of each sun sensor respond correctly when illuminated under different angles. If all sun sensors respond correctly we can verify if the correct sun vector is determined if we illuminate the sun sensors under a certain angle. Finally the incorporated temperature sensor on the sun sensor Printed Circuit Board (PCB) needs to be verified.The following subsections will give a more detailed description of the above mentioned tests.

8-2-1 Electronic bias

In [38] has been found that the electronic circuitry of the sun sensor induces a bias in the sun sensor measurements. This bias is caused by dark current of the photo diode and by the imperfections of the op-amp in the electronic circuitry. In order to determine the quantity of this bias sun sensor measurements are done with the sun sensors completely covered. The value that is measured is the electronic bias and can be, if significantly large, subtracted from feature measurements for correction. Figure 8-2 shows the results of the bias measurement for the sun sensor in positive X direction. In Table 8-1 the averages of the biases for all sun sensors are given.

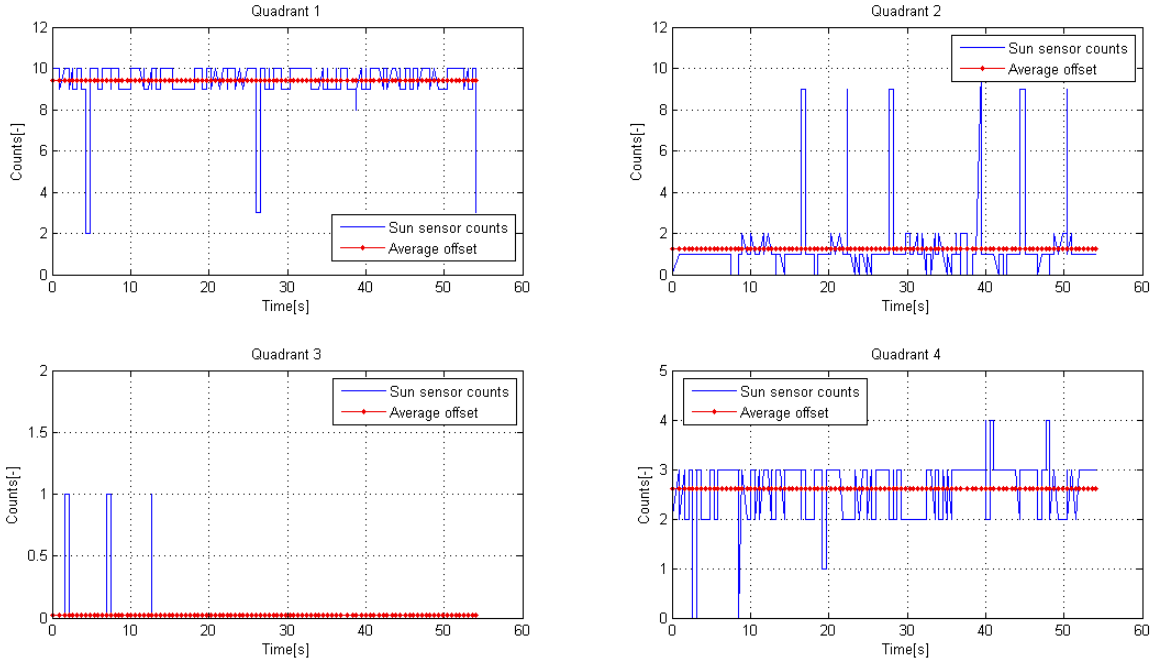


Figure 8-2: The bias of the four quadrants of the sun sensor in positive X direction

From Figure 8-2 we can see that the biases of all four quadrants are non zero. The largest measured output bias has a value of 10 counts, which, according to Eq. (8-1), corresponds to a value of $312\mu V$.

$$Output\ voltage = \frac{Sensor\ counts * V_{ref}}{2^N} \quad where\ N = 16\ and\ V_{ref} = 2.048V \tag{8-1}$$

With a total measurement range from 0V to 1.8V the $312\mu V$ is not large enough to induce a significant error in the determination of the sun vector. Therefore the sun sensor measurements will not be bias corrected. If in the future it turns out that the electronic bias does induces an significant error in the sun vector, a bias correction value can be uploaded from the ground station.

	Average bias			
Sun sensor	**Quadrant 1**	**Quadrant 2**	**Quadrant 3**	**Quadrant 4**
X+	281	31	0	62
X-	31	62	62	0
Y+	62	218	124	31
Y-	124	93	0	0
Z+	31	0	187	31
Z-	124	0	0	124

Table 8-1: Average sun sensor biases in μV

8-2-2 Variation in illumination angle

To verify whether all four quadrants of each sun sensor respond correctly when illuminated under different angles, we mounted the sun sensor on a rotatable platform. The LED light was placed on a certain distance perpendicular to that platform. The platform was rotated a total of 180 degrees in steps of 15 degrees, thus covering 12 different illumination angles. In Figure 8-3 the measurement results for the sun sensor in positive X direction are depicted.

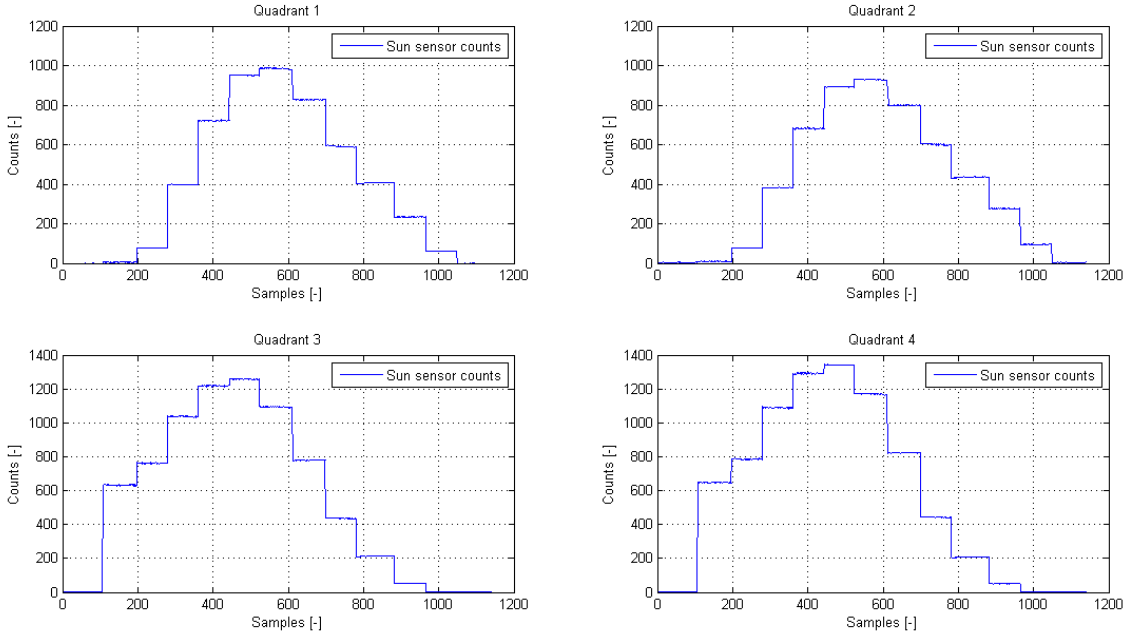


Figure 8-3: The response of the sun sensor in positive X direction under 12 different illumination angles.

Figure 8-3 shows us that the sun sensor quadrants more or less show the right behaviour. From an angle of 180 degrees up to an angle α the values of all the four quadrants start to increase. From angle α to angle an angle β two quadrants start to decrease while the other two are still increasing. And from angle β to 0 degrees the values of all four quadrants decrease. However, due to the lack of a good calibrated test set-up, the angles α and β could not be determined precisely. Therefore, from these measurements, we can only conclude that the four quadrants respond to different illumination angles, but further measurements must

more precisely show if this response is correct.

8-2-3 Sun vector determination

To verify if the correct sun vector is determined from the sun sensor outputs one sun sensor is connected to the ADCS, while the other five are disconnected. The connected sun sensor is illuminated perpendicular and should be the only sun sensor that gives an output value (i.e the other five sun sensors should give zero output values). This process is repeated for all six sun sensors. Because every time only one sun sensor gives output values for the four quadrants, a sun vector should be determined in the direction corresponding to that sun sensor. For example if the sun sensor for the positive X direction is connected, then the calculated sun vector should also lie in the X direction. In an ideal case the determined sun vectors for each sun sensor should correspond the unit vectors in the six principle directions (i.e. X+, X-, Y+, Y-, Z+ and Z-). Figure 8-4 shows a 3D representation of the six determined sun vectors.

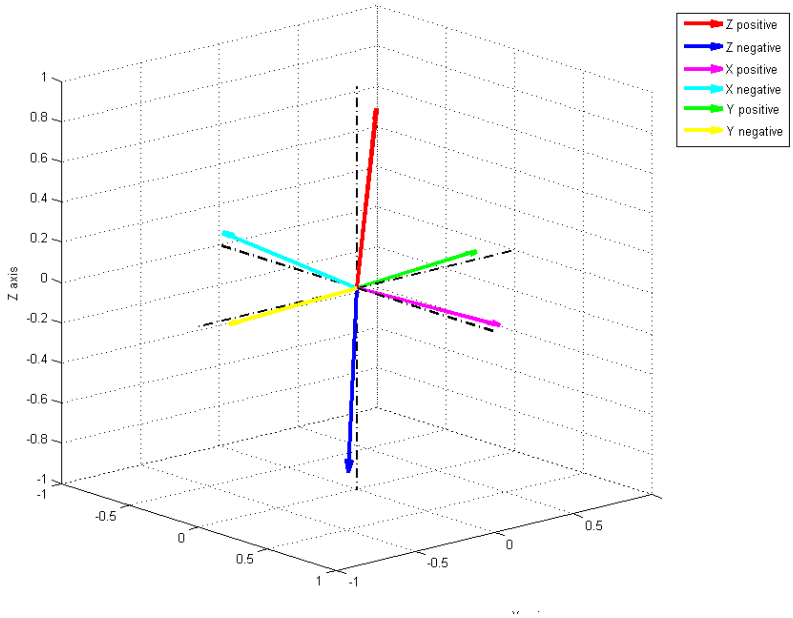


Figure 8-4: The six sun vectors determined from the sun sensor measurements

From Figure 8-4 we can clearly see the sun vectors in the six directions. The black dotted lines indicate the unit vectors in the six principle directions. If it was a perfect measurement the sun vectors would correspond with these unit vectors. This however is not the case. All six sun vectors slightly deviate from their corresponding unit vector. This deviation can maybe slightly be reduced by incorporating correction factors or values for the quadrant of each sun sensor, which can be determined from the sensitivity measurements. This however has not been investigated or verified and could be an option for future work.
For testing other directions than the principle directions we used a different test set-up, because the LED light was not powerful enough for this purpose. In this test set-up we fastened two sun sensors together under an angle of 90 degrees. With a beamer, projecting a white image, the two sensors were illuminated under an angle of roughly 45 degrees. This

way we should get a sun vector with an angle of roughly 45 degrees from one of the principle directions. The test results however were not as satisfying as expected. As with the metal halide lamp the light of the beamer also flickers with a frequency of 50Hz. Due to this flickering the sun vector showed great fluctuations in the direction, making it hard to verify whether the direction was correct. Also the light beam of the beamer diverges inducing an error in the illumination angle of the sun sensors. In order to still test other directions than the principle directions, we used simulated output of the sun sensors, corresponding to different directions, and used these as an input for the sun vector determination algorithm. Figure 8-5 shows the resulting sun vectors for the XY directions and Figures 8-6 and 8-6 shows the resulting sun vectors for the directions XZ+/YZ+ and XZ-/YZ-.

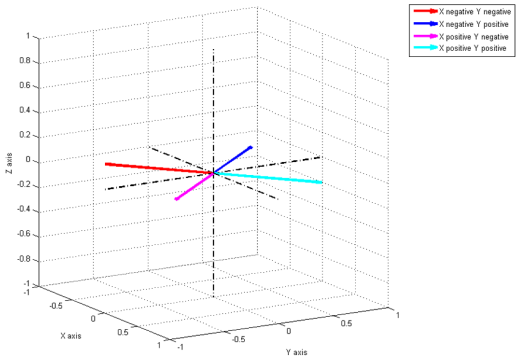


Figure 8-5: Sun vectors in de XY directions

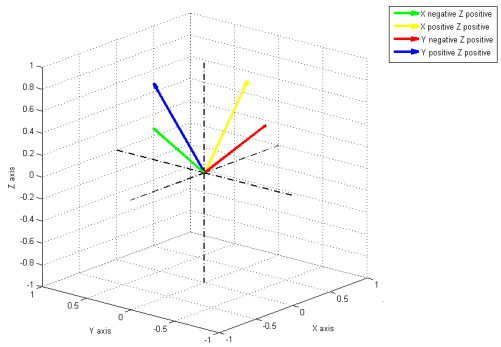


Figure 8-6: Sun vectors in de XZ+ and YZ+ directions

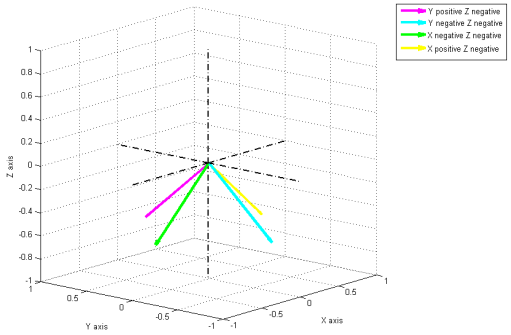


Figure 8-7: Sun vectors in de XZ- and YZ- directions

8-2-4 Temperature sensor

Although the temperature data from the sun sensor is not needed to determine the sun vector, it is useful information to verify if the determined sun vector is correct, because the side pointing to the sun has the highest temperature. Verification of the temperature sensor is done with the vacuum oven set-up inside the clean room, constructed by M. Boerci. With this set-up we can create a near vacuum and cool down and heat up the sun sensor. The sun sensor

is placed inside the vacuum oven and a thermocouple is attached to the sun sensor PCB. Data from the thermocouple is used as a reference temperature to verify the temperature data from the sun sensor. The result of the temperature measurement is depicted in Figure 8-8.

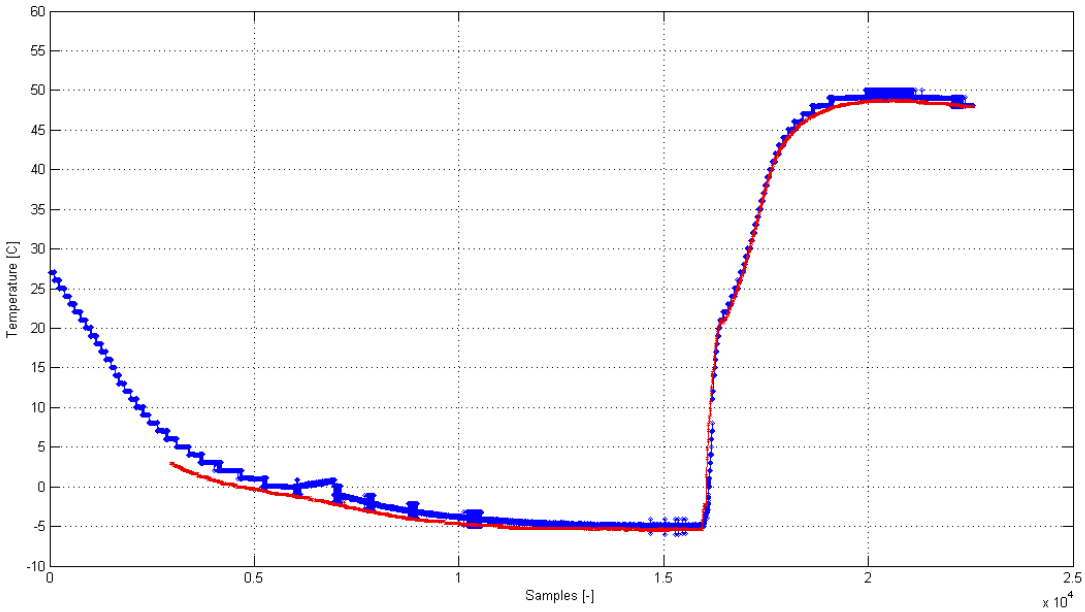


Figure 8-8: Temperature measurements from the temperature sensor on the sun sensor PCB and the thermocouple

In Figure 8-8 the red line indicates the temperature as measured by the thermocouple and the blue line the temperature measured by the temperature sensor on the sun sensor PCB. The first thing we notice from Figure 8-8 is that the measurements of the thermocouple start later than those of the temperature sensor. This is due to the fact that in the beginning of the test the thermocouple was not placed in measuring mode, therefore not storing temperature measurements. Despite the missing data from thermocouple in the beginning, we can see that in the rest of the plot the temperature sensor closely follows the thermocouple. Therefore we can conclude that the temperature sensor is working correctly in does not need further calibration.

8-3 Reaction wheel verification

The reaction wheels have been already extensively tested in [48], therefore we will only verify the functionality. To test the functionality we first sent every two seconds the maximum positive acceleration of 150 rpm/s to all three reaction wheels. This will accelerate the reaction wheels in positive direction until it reaches its maximum speed of around +25000 RPM. If the maximum speed is reached zero acceleration is sent every two seconds, therefore keeping the reaction wheels spinning at a constant speed. After some period a maximum negative acceleration of -150 RPM/s is sent to the reaction wheels every two seconds. This will first slow down the reaction wheels to 0 RPM and then starts accelerating in the negative

direction until it reaches its maximum speed of around -25000 RPM. Again for a certain period zero acceleration is sent to the reaction wheels, after which again a maximum positive acceleration is sent the reaction wheels until it reaches zero RPM. During this process the RPM of the reaction wheels is measured with time intervals of 0.1 second. The results of these measurements are depicted in Figure 8-9.

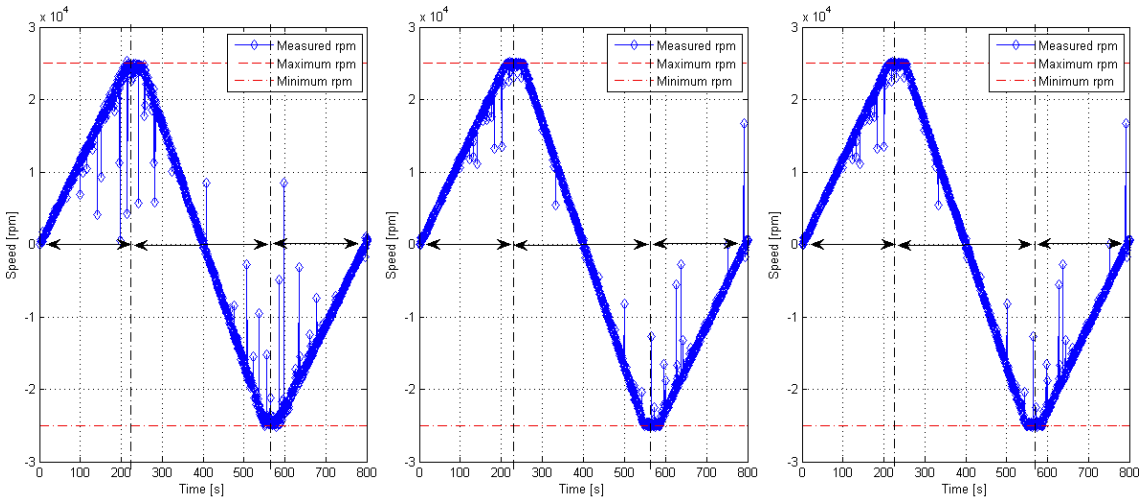


Figure 8-9: Acceleration to maximum speed, in positive and negative direction, of the three reaction wheels.

Figure 8-9 shows that all three reaction wheels accelerate in the right direction and reach the maximum positive and negative speed of +/- 25000 RPM. According to the maximum positive and negative accelerations we sent to the reaction wheels the slope of graphs should be +150 or -150, depending on which direction we are accelerating. In Table 8-2 the calculated slopes of the three parts, indicated by the double sided arrows, in each graph are given.

	Slope		
Reactionwheel	**Part 1**	**Part 2**	**Part 3**
X	118	-137	109
Y	114	-147	102
Z	92	-142	103

Table 8-2: Average sun sensor biases in micro Volts

From Table 8-2 it is clear that none of the reaction wheels reach the maximum acceleration of 150 RPM/s. One explanation for this could be that because of gravity the flywheel encounters more friction. It also is possible that switching between MOSFET configurations to power the electric coils and make the motor turn, is not done in an optimal way.

8-4 Summary

In this chapter the verification procedures of the sun sensors and reaction wheels were described and the results were presented. For the sun sensors we have found that the electronic

bias does not induce a significant error in the determination of the sun vector. From a first test could not be concluded if the four quadrants of each sun sensor responded correctly to different illumination angles. A second test showed that the direction of the sun vector is determined correctly if one or two sun sensors are illuminated perpendicular or under an angle of roughly 45 degrees. Hence, we could also conclude that the four quadrants of each sun sensor respond correctly to different illumination angles. Furthermore it has been verified that the temperature sensor on the sun sensor PCB works correctly and no further calibration is needed.
For the reaction wheels we have seen that all three reaction wheels function correctly and are able to reach the maximum positive and negative speed of +/-25000 RPM. The results also showed us that the reaction wheels accelerate slower than the commanded RPM/s. This could be explained by friction due to gravity or an non optimal control of the motor.

Chapter 9

Conclusions and recommendations

In this thesis the design, implementation and the verification of the Attitude Determination and Control Subsytem (ADCS) software was presented. Furthermore the hardware of the ADCS was individually tested and the results were presented. Finally both hardware and software were tested together to see if the ADCS showed correct behaviour in various operational modes.

In this chapter we draw some conclusions based on the thesis objectives, mentioned in the introduction, and the results from the verification of the ADCS software and hardware. Furthermore we provide some recommendations and work for the future on how to finalize and improve the ADCS software.

9-1 Conclusions

In order to draw conclusions from the work and results presented in this thesis we do a review of the main objective and the sub-objectives mentioned in the introduction. We begin with discussing the sub-objectives and conclude with the main objective.

Design and implementation of a software framework for the ADCS These two sub-objectives stated that the software framework should provide all the functionality to ensure correct, reliable and robust working and should be easy to extend and adapt for different micro controllers/processors. To achieve this, the framework should be implemented using different software layers.
In Chapter 6 I described the design and implementation of the software framework. The design and implementation described, provides all the functionality that ensures a correct, reliable and robust working of the ADCS. The partitioning in different software layers and modules make it easy to extend and adapt the software framework for different micro controllers/processors.

Verification of the ADCS hardware and software individually These two sub-objectives stated that parts of the ADCS software and hardware should be tested individually using several unit tests. Although not explicitly mentioned in one of the chapters in this thesis, different parts of the software were tested individually using various unit tests. Testing of the individual hardware was described in the Chapters 7 and 8. The results that were presented in these chapters showed us that all the hardware functioned correctly. The magnetorquers switched on and off correctly and did not show any remanance that could influence the magnetometer. The magnetometer gave correct results after corrections were done for the hard and soft iron distortions induced by the magnetorquers. The six sun sensors showed correct behaviour when illuminated under different angles. The performance however could not be determined precisely due to a lack of good calibrated test set-up. All three reaction wheels also showed correct behaviour. However, they accelerated slower than they theoretically should have.

Verifying the combination ofADCS software and hardware This sub-objective states that when both hardware and software are test individually for their functionality, they must be tested in combination with each other in order to verify whether the determination and control algorithms work correctly. In Chapter 7 we have verified the detumble mode for its functionality. From the results presented in this chapter we could that the detumble mode does what it is supposed to do. The Bdot, tumble parameter and magnetic dipole are calculated correctly and the magnetorquers are actuated accordingly.

9-2 Future work

Despite the postponement of the final deadline for the Delfi-n3Xt we were not able to deliver an ADCS with all the functionality as described in this thesis and [3]. For future missions it maybe useful to complete the remaining parts of the ADCS that were not yet finished. Therefore the following future work packages are proposed.

Verifying the functionality of CSP mode Despite all the effort and extra time that has been put into verifying the functionality of the CSP mode, we did not have sufficient time to fully verify the correct working of this mode. The individual modules from the service layer that are used in the Coarse Sun Pointing (CSP) mode have been verified, but the actual determination and control algorithm in the application layer has not. Therefore a future work package would be the verification of the determination and control algorithm in the application layer of the ADCS software.

Verifying the functionality of the advanced modes Just as with CSP mode there was insufficient time left to verify the functionality of the advanced modes. All the individual components from the service layer that are used by these modes have been verified. What is remaining is the verification of the actual determination and control algorithms for these modes. A future work package would therefore be the verification of the determination and control algorithms for the advanced modes.

Implementation of a more advanced sun vector determination algorithm Currently the simplest sun vector determination algorithm, described in [38], is implemented in the ADCS software. This algorithm does not take the rotational rate of the satellite and any albedo effects into account. Therefore a future work package would be implementing one of the more advanced sun vector determination algorithms described in [38].

Reduce power usage of the ADCS hardware Currently the ADCS uses much more power than what was accounted for. Therefore the ADCS much be switched off when the satellite is in eclipse, which has serious consequences for the advanced operational modes. The large power consumption is due the fact that the software is running from DDR-RAM, which consumes a lot of power. Originally it was supposed to run from the available NOR-Flash on the ICnova SO-DIMM module, which use much less power. Unfortunately we did not get the software working correctly and fast enough from NOR-Flash. Therefore a future work package would be reducing the power usage by running the software from NOR-Flash or by other power saving features.

9-3 Recommendations

During the development and testing of the ADCS software framework different problems were encountered. Some of these problems are of a more general nature while the others are more related to the ADCS itself. This section a couple of these problems are discussed and a recommendation is given.

Software availability Not only with the development of the ADCS, but also with other subsystems, we sometimes encountered the problem that not on every computer in the clean room software was available that was needed to perform certain tests or tasks. With installing the right software every time it is missing precious time is lost. Therefore my recommendation is to make sure all used software is installed on every computer in the clean room. If this is not feasible then at least make sure that installation files for the software is available from a central storage space.

Availability of flight (representable) hardware It is important that the ADCS software is tested on flight or at least flight representable hardware. Also all flight hardware must be thoroughly tested before it can be used as a flight candidate. Therefore it is important that flight and/or flight representable hardware becomes available as soon as possible. My recommendation for the future is therefore to start producing flight hardware in an earlier stadium of the project.

Magnetorquer induced distortions Although the distortion of the magnetic field caused by the magnetorquer proved not to be a critical problem, it could have been prevented during the design of the ADCS. Luckily for us the solution for this problem did not have any impact on the performance of the ADCS, but we did lost precious development time. My recommendation for a future version of the ADCS is to place the magnetorquer assembly and the magnetometer far away from each other.

Bibliography

[1] J.P. de Jong. Dnx-tud-tn-0169 [3.4] sts - top level design of structural subsystem. Technical report, Tu Delft, September 2012.

[2] Greg Welch and Gary Bishop. An introduction to the kalman filter. 2001.

[3] Johannes P.J. Reijneveld. Design of the attitude determination and control algorithms for the delfi-n3xt. Master's thesis, TU Delft, January 2012.

[4] Swiss cube. Available from: `http://swisscube.epfl.ch/`.

[5] H. Peter-Contesse A.E. Overlack, J.M. Kuiper and M. Noca. Analysis of the attitude control stability of the swisscube nano-satellite. 2011.

[6] Illinois observing nanosatellite. Available from: `http://cubesat.ece.illinois.edu/`.

[7] K. Svartveit. Attitude determination of the ncube satellite. Master's thesis, Norwegian University of Science and Technology, June 2003.

[8] Aausat3. Available from: `http://www.space.aau.dk/aausat3/`.

[9] Goverment of dubai - dubaisat2, October 2012. Available from: `http://www.eiast.ae/default.aspx?options=%7Ba93e7034-0baa-4e2b-be21-721a4b6feb8e%7D&view=Article&layout=Article&itemId=160&id=257` [cited 27th].

[10] Delfispace website, October 2012. Available from: `http://www.delfispace.nl` [cited 18th].

[11] J.H. Doorn. Delfi-c3s electrical power system and its thin film solar cell payload. Master's thesis, TU Delft, June 2006.

[12] A.F.C van den Berg. Fault-tolerant on-board computer software for the del fault-tolerant on-board computer software for the delfi-n3xt nanosatellite. Master's thesis, TU Delft, September 2012.

[13] B. Zandbergen G. Brouwer R.Amini D. Kajon C. Mller, L. Perez Lebbink and B. Sanders. Implementation of the t3ups in the delfi-n3xt satellite. volume 7 th IAA sym- posium on Small Satellites for Earth Observation, page 8. TU Delft, 2009.

[14] Lieuwe S. Boersma. Verification and testing of the delfi-n3xt communications subsystem. Master's thesis, TU Delft, March 2012.

[15] Mr. J. Bouwmeester Ms. S.Y. Go and Mr. G.F. Brouwer. Optimized three-unit cubesat structure for delfi-n3xt. *59th International Astronautical Congress*, September 2008.

[16] Y. Awchi. Design of mechanical systems. Master's thesis, June 2009.

[17] J.P. de Jong. Dnx-tud-tn-0572 [2.3] mechs - design of mechanical systems. Technical report, TU Delft, June 2012.

[18] D. Poole. *Linear Algebra: A Modern Introduction*. Available Titles CengageNOW Series. Brooks/Cole, 2006.

[19] Dr. Ing Zizung Yoon Dipl. Ing. Karsten Großekatthofer. Introduction into quaternions for spacecraft attitude representeation. 2012.

[20] Members of the Technical Staff Attitude Systems Operation Computer Sciences Corporation. *Spacecraft Attitude Determination and Control*. Kluwer academic publisher, 1978.

[21] James Diebel. Representing attitude: Euler angles, unit quaternions, and rotation vectors. October 2006.

[22] Bryan Scott Gregory. Attitude control system design for ion, the illinois observing nanosatellite. Master's thesis, University of Illinois, 2004.

[23] Christopher D. Hall. *Spacecraft Attitude Dynamics and Control*, chapter 4. January 2003.

[24] H.D. Black. A passive system for determining the attitude of a satellite. *AIAA Journal*, 2:1350–1351, July 1964.

[25] G. Wahba. *Problem 65–1: A Least Squares Estimate of Spacecraft Attitude*, volume 7. SIAM Review, 1965.

[26] F. Landis Markley and Daniele Mortari. *Quaternion attitude estimation using vector observations*, volume 48. The Journal of the Astronautical Sciences, September 2000.

[27] Quadratic form. Available from: `http://en.wikipedia.org/wiki/Quadratic_form`.

[28] Roger A. Horn and Charles R. Johnson. *Matrix analysis*. Cambridge University Press, 1985.

[29] Javier Arenas Rodriguez. Estimation: Kalman filters for attitude and orbit determination. Master's thesis, Delft University of Technology, September 2011.

[30] A.C. Stickler and K.T. Alfriend. Elementary magnetic attitude control system. *Journal of Spacecraft and Rockets*, pages 282–287, 1975.

[31] H. Weiss B. Wie and A. Arapostathis. Quaternion feedback regulators for spacecraft eigenaxis rotation. Control and Dynamics. AIAA - Journal of Guidance, 1989.

[32] David Lerner. Lecture notes on linear algebra. October 2007.

[33] B. Wie. *Space Vehicle Dynamics and Control*. Reston, 2008.

[34] J.R. Forbes and C.J. Damaren. A geometric approach to spacecraft attitude control using magnetic and mechanical actuation. Control and Dynamics. Submitted to Journal of Guidance, 2009.

[35] V. van Gelder. Dpg-tud-iw-1019 [1.0] gelder, vera van - internship report on gyroscopes and dude. Technical report, TU Delft, 2011.

[36] Three-axis digital compass ic hmc5883l, November 2012. Available from: `http://www51.honeywell.com/aero/common/documents/myaerospacecatalog-documents/Defense_Brochures-documents/HMC5883L_3-Axis_Digital_Compass_IC.pdf` [cited 2nd].

[37] C. Macco. Dnx-tud-bu-0872 [4.0] delfi-n3xt thermal budget. Technical report, TU Delft, 2012.

[38] Alexander S.B.B. Maas. The development of the sun vector determination system of delfi-n3xt. Master's thesis, TU Delft, October 2011.

[39] Hamamatsu si pin photodiode, November 2012. Available from: `http://jp.hamamatsu.com/resources/products/ssd/pdf/s5980_etc_kpin1012e04.pdf` [cited 3th].

[40] Atmel xmega128a1 datasheet. Available from: `http://www.atmel.com/Images/doc8067.pdf`.

[41] Atmel sam9g45 datasheet. Available from: `http://www.atmel.com/Images/doc6438.pdf`.

[42] F.M. Poppenk. Design and testing of an attitude control system for a nano satellite. Master's thesis, Delft University of Technology, February 2009.

[43] Various authors. Dnx-tud-tn-0169 [3.4] sts - top level design of attitude determination and control subsystem. Technical report, TU Delft, February 2012.

[44] Michael J. Caruso. Applications of magnetic sensors for low cost compass systems. page 8.

[45] Anthony Wutka Michael Garton and Andrew Leuzinger. Local magnetic distortion effects on 3-axis compassing. page 7, July 2009.

[46] Christopher Konvalin. Compensating for tilt, hard iron and soft iron effects. Technical report, MEMSense, August 2008.

[47] Vectornav magnetometer model. Available from: `http://www.vectornav.com/support/library?id=83`.

[48] T. Hoevenaars. Dnx-tud-tn-0284 [2.3] adcs - reaction wheel system design. Technical report, TU Delft, February 2012.

Appendix A

Housekeeping data

In this appendix some more information about the housekeeping data of the ADCS is given. At the end of each main control loop the housekeeping data of the ADCS is gathered from the storage ring buffer. From the gathered data a housekeeping data frame is prepared for transmission, by applying the inverse transfer functions, which are depicted in Table A-2, to the data and dividing or concatenate it into byte sized pieces. Every two seconds the housekeeping data is requested and the prepared frame is sent to the OBC. In Figure A-1 an example of such a frame, taken from an I2C monitor, is depicted.

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x0000	01	15	DF	5D	75	22	03	DB	A9	98	94	56	C2	5C	84	7F
0x0010	F7	FF	7F	F0	00	16	00	11	00	1B	00	1C	00	14	00	15
0x0020	00	12	00	1F	00	11	00	18	00	16	00	0E	00	19	00	0C
0x0030	00	1F	00	0F	00	00	00	00	00	00	00	00	00	16	00	17
0x0040	00	18	00	17	80	00	80	00	80	00	80	2D	2B	7F	E8	00
0x0050	00	00	F0	0F	A0	FA	0F	A0	FA	7F	7F	7F	00	00	00	00
0x0060	08	05	80	58	05	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
0x0070	FF	FF	FF	00	9B	9B	9A	9B	3C	98						

Figure A-1: An example of a housekeeping data frame.

The OBC gathers housekeeping data frames from every subsystem and puts it into a larger frame which is sent to one of the radios for transmission to the ground station. At the ground station the data from the received frames can be converted back to their original format with their corresponding transfer function. In the next section a brief description of all the housekeeping data parameters is given and the transfer functions for converting the ADCS housekeeping data back to its original format are given in Table A-2

A-1 Parameter descriptions

In this section a brief description of all the housekeeping data parameters is given. The value ranges and transfer functions of these parameters are depicted in Table A-2.

ADCS Last Parameter Upload Storage: Parameter that indicates whether the last parameter that was uploaded is correctly stored in Non Volatile Memory (NVM).

ADCS Last Received Command: Parameter that indicates whether the last received command from the OnBoard Computer (OBC) was a valid command.

ADCS Last Parameter Upload: Parameter that indicates whether the last parameter upload was successful (i.e the parameter was correctly stored in NVM and correctly updated in the execution format).

ADCS status and Magnetometer Status: These two parameters indicate the current status of the Attitude Determination and Control Subsytem (ADCS). If the magnetometer status indicates an error the ADCS status is automatically 'NOT OK' and if the magnetometer status indicates no error the ADCS status is automatically 'OK'.

ADCS Mode Indicator: Parameter that indicates which operational mode the ADCS is currently in.

Magnetic Field X,Y and Z (MTQ Off): These three parameters contain the raw magnetometer data, for X,Y and Z direction, when the magnetorquers are switched OFF. The corresponding transfer functions from Table A-2 converts the raw data only to magnetic field values in Gauss, but does not do any of the bias corrections or transformations described in this thesis. (i.e. bias correction and transformations have to be done at the ground station).

Magnetic Field X,Y and Z (MTQ On): These three parameters contain the raw magnetometer data, for X,Y and Z direction, when the magnetorquers are switched ON. The corresponding transfer functions from Table A-2 converts the raw data only to magnetic field values in Gauss, but does not do any of the bias corrections or transformations described in this thesis. (i.e. bias correction and transformations have to be done at the ground station).

Magnetic Bias X,Y and Z: These three parameters are raw magnetometer data for X,Y and Z direction, which contain the bias values measured during a positive or negative bias measurement. The corresponding transfer functions from Table A-2 converts the raw data to magnetic field values in Gauss.

Magnetorquer X,Y and Z Commanded Dipole: These three parameters are the dipoles commanded by each magnetorquer.

Sun sensor X+,X-,Y+,Y-,Z+ and Z- Qn: These six parameters contain the measurement data from each sun sensor.

Reaction Wheel X,Y and Z Actual Rate: These three parameters contain the measured RPMs for each reaction wheel.

Reaction Wheel X,Y and Z Commanded Torque: These three parameters contain the commanded acceleration in RPM/s for each reaction wheel.

Reaction Wheel Unloading Status: Parameter that indicates whether the reaction wheels are being loaded or unloaded.

Sun Presence: Parameter that indicates whether the sun is present or the satellite is in eclipse (i.e. the sun is not present). If the sun is present the values for the sun vector components should be non zero.

ADCS Time Since 01-01-2010 00:00:00: Parameter that contains the elapsed time since the start of the ADCS application. If the ADCS is reset or switched off, the elapsed time is also reset.

Pointing quaternion 1,2,3 and 4: These four parameters contain the quaternion values that represent that actual attitude of the satellite.

Sun Vector X,Y and Z Components: These three parameters contain the X,Y and Z component of the sun vector. Together the form the sun vector which indicate the direction of the sun.

Angle EIRF-SAT 1,2 and 3: These three parameter that contain the angles of the satellite with the Earth Centered Inertial reference frame, which are determined in the advanced modes.

Rotation Rate X,Y and Z: These three parameters contain the rotational rate of the satellite, in X,Y and Z direction, estimated by the algorithm in one of the advanced modes.

Covariance Quaternion 1,2,3 and 4: These four parameters contain the quaternion covariances from the covariance matrix P, which is used by the Kalman filter in the advanced modes.

Covariance Rotational Rate X,Y and Z: These three parameters contain the rotational rate covariances from the covariance matrix P, which is used by the Kalman filter in the advanced modes.

Sun Sensor X+,X-,Y+,Y-,Z+ and Z- Temperature: These six parameters are contain the measurement data of the temperature sensors from each sun sensor.

Content	Bitsize	Unit	Min. Value	Max. Value	Transfer function
Empty	2				=[00]
ADCS Last Parameter Upload Storage	1				[0]=no error, [1]=failed
ADCS Last Received Command	1				[0]=no error, [1]=invalid
ADCS Last Parameter Upload	2				[00]=no param, [01]=failed,[10]=success
Magnetometer Status	1				[0]=no error, [1]=error
ADCS Status	1				[0]=not OK, [1]=OK
ADCS Mode Indicator	4				see parameter description
Magnetic Field X (MTQ Off)	12	μT	-310.3	310.2	[value] · 0.1515 - 310.3
Magnetic Field Y (MTQ Off)	12	μT	-310.3	310.2	[value] · 0.1515 - 310.3
Magnetic Field Z (MTQ Off)	12	μT	-310.3	310.2	[value] · 0.1515 - 310.3
Magnetic Field X (MTQ On)	12	μT	-310.3	310.2	[value] · 0.1515 - 310.3
Magnetic Field Y (MTQ On)	12	μT	-310.3	310.2	[value] · 0.1515 - 310.3
Magnetic Field Z (MTQ On)	12	μT	-310.3	310.2	[value] · 0.1515 - 310.3
Magnetic Bias	12	μT	-310.3	310.2	[value] · 0.1515 - 310.3
Magnetic Bias	12	μT	-310.3	310.2	[value] · 0.1515 - 310.3
Magnetic Bias	12	μT	-310.3	310.2	[value] · 0.1515 - 310.3
Magnetorquer X Commanded Dipole	12	mAm^2	-60	60	[value] · 0.0293 - 60
agnetorquer Y Commanded Dipole	12	mAm^2	-60	60	[value] · 0.0293 - 60
Magnetorquer Z Commanded Dipole	12	mAm^2	-60	60	[value] · 0.0293 - 60
Empty	4				=[0000]
Sun sensor X+ Q1	16	mV	0	2048	[value] · 0.0313
Sun sensor X+ Q2	16	mV	0	2048	[value] · 0.0313
Sun sensor X+ Q3	16	mV	0	2048	[value] · 0.0313
Sun sensor X+ Q4	16	mV	0	2048	[value] · 0.0313
Sun sensor X- Q1	16	mV	0	2048	[value] · 0.0313
Sun sensor X- Q2	16	mV	0	2048	[value] · 0.0313
Sun sensor X- Q3	16	mV	0	2048	[value] · 0.0313
Sun sensor X- Q4	16	mV	0	2048	[value] · 0.0313
Sun sensor Y+ Q1	16	mV	0	2048	[value] · 0.0313
Sun sensor Y+ Q2	16	mV	0	2048	[value] · 0.0313
Sun sensor Y+ Q3	16	mV	0	2048	[value] · 0.0313
Sun sensor Y+ Q4	16	mV	0	2048	[value] · 0.0313
Sun sensor Y- Q1	16	mV	0	2048	[value] · 0.0313

Continued on next page

Table A-2 – *Continued from previous page*

Content	Bitsize	Unit	Min. Value	Max. Value	Transfer function
Sun sensor Y- Q2	16	mV	0	2048	[value] · 0.0313
Sun sensor Y- Q3	16	mV	0	2048	[value] · 0.0313
Sun sensor Y- Q4	16	mV	0	2048	[value] · 0.0313
Sun sensor Z+ Q1	16	mV	0	2048	[value] · 0.0313
Sun sensor Z+ Q2	16	mV	0	2048	[value] · 0.0313
Sun sensor Z+ Q3	16	mV	0	2048	[value] · 0.0313
Sun sensor Z+ Q4	16	mV	0	2048	[value] · 0.0313
Sun sensor Z- Q1	16	mV	0	2048	[value] · 0.0313
Sun sensor Z- Q2	16	mV	0	2048	[value] · 0.0313
Sun sensor Z- Q3	16	mV	0	2048	[value] · 0.0313
Sun sensor Z- Q4	16	mV	0	2048	[value] · 0.0313
Reaction Wheel X Actual Rate	16	RPM	-32768	32767	[value] · 1 - 32768
Reaction Wheel Y Actual Rate	16	RPM	-32768	32767	[value] · 1 - 32768
Reaction Wheel Z Actual Rate	16	RPM	-32768	32767	[value] · 1 - 32768
Reaction Wheel X Commanded Torque	12	RPM/s	-2048	2047	[value] · 1 - 2048
Reaction Wheel Y Commanded Torque	12	RPM/s	-2048	2047	[value] · 1 - 2048
Reaction Wheel Z Commanded Torque	12	RPM/s	-2048	2047	[value] · 1 - 2048
Reaction Wheel Unloading Status	1				[0]=loading, [1]=unloading
Sun Presence	1				[0]=eclipse, [1]=sun
Empty	2				=[00]
ADCS Time Since 01-01-2010 00:00:00	32	s	0	4294967295	[value] · 1
Pointing quaternion 1	12	-	-1	1	[value] · 0.00049 - 1
Pointing quaternion 2	12	-	-1	1	[value] · 0.00049 - 1
Pointing quaternion 3	12	-	-1	1	[value] · 0.00049 - 1
Pointing quaternion 4	12	-	-1	1	[value] · 0.00049 - 1
Sun Vector X Component	8	-	-1	1	[value] · 0.00784 - 1
Sun Vector Y Component	8	-	-1	1	[value] · 0.00784 - 1
Sun Vector Z Component	8	-	-1	1	[value] · 0.00784 - 1
Angle EIRF-SAT 1	12	°	0	360	[value] · 0.08791
Angle EIRF-SAT 2	12	°	0	360	[value] · 0.08791
Angle EIRF-SAT 3	12	°	0	360	[value] · 0.08791
Rotation Rate X	12	$^\circ/s$	-180	180	[value] · 0.08791 - 180
Rotation Rate Y	12	$^\circ/s$	-180	180	[value] · 0.08791 - 180

Continued on next page

Table A-2 – *Continued from previous page*

Content	Bitsize	Unit	Min. Value	Max. Value	Transfer function
Rotation Rate Z	12	$^\circ/s$	-180	180	[value] · 0.08791 - 180
Covariance Quaternion 1	16	-	0	1.00E-02	[value] · 1.526E-7
Covariance Quaternion 2	16	-	0	1.00E-02	[value] · 1.526E-7
Covariance Quaternion 3	16	-	0	1.00E-02	[value] · 1.526E-7
Covariance Quaternion 4	16	-	0	1.00E-02	[value] · 1.526E-7
Covariance Rotational Rate X	16	-	0	1.00E-06	[value] · 1.526E-11
Covariance Rotational Rate Y	16	-	0	1.00E-06	[value] · 1.526E-11
Covariance Rotational Rate Z	16	-	0	1.00E-06	[value] · 1.526E-11
Empty	8				=[00000000]
Sun Sensor X+ Temperature	8	°C	-15	48.75	[value] · 0.25 - 15
Sun Sensor X- Temperature	8	°C	-15	48.75	[value] · 0.25 - 15
Sun Sensor Y+ Temperature	8	°C	-15	48.75	[value] · 0.25 - 15
Sun Sensor Y- Temperature	8	°C	-15	48.75	[value] · 0.25 - 15
Sun Sensor Z+ Temperature	8	°C	-15	48.75	[value] · 0.25 - 15
Sun Sensor Z- Temperature	8	°C	-15	48.75	[value] · 0.25 - 15

Table A-2: Housekeeping data of the ADCS

Appendix B

Parameter upload example

In this appendix some more information about the construction of a parameter upload command is given and an example on how to update parameters is given. Furthermore, Table B-7 shows an overview of all the parameters that can be changed with a parameter upload.
To update parameters during flight we need to send a parameter upload command to the ADCS. A parameter upload command consist of at least 4 bytes beginning with the 7-bit slave address of the ADCS. The second byte is divided into two parts of 4 bits. The four most significant bits are the 'command', which tells the ADCS we are dealing with a parameter upload command, and the 4 least significant bits is the amount of parameters we want to update. Since we only have 4 bits for the amount of parameters, we can update a maximum of 15 parameters with one parameter upload command. The third byte contains the start ID of the parameter from where we want to start updating parameters. The N bytes that follow the third byte are the data bytes, where N depends on the amount of parameters and the bit size of those parameters. Table B-1 shows the construction of a parameter upload command

Byte 1	**Byte 2**		**Byte 3**	**Byte 4 to N**		
I2C address	Command	Amount of parameters	Start ID	Data byte 1	...	Data byte N

Table B-1: Construction of parameter upload command

Suppose we want to update the following three parameters: MTQ_DIPOLE_X, MTQ_DIPOLE_Y and MTQ_DIPOLE_Z to the values 0.121, 0.123 and -0.081 respectively. From Table B-7 we can see that these are three consecutive parameters with decimal upload ID's: 37, 38 and 39. Therefore, the first three bytes of the parameter upload command become 0x76, 0x83 and 0x25 (i.e. 37 in hexadecimal) as shown in Table B-2.

Byte 1	**Byte 2**	**Byte 3**	**Byte 4 to 6**		
0x76	0x83	0x25	Data byte 1	Data byte 2	Data byte 3

Table B-2: First three bytes of a parameter upload command

To determine the values of the data bytes we must take a look at the units, the minimum value, the maximum value and the resolution of the parameters. From Table B-7 we can see

that the unit is $mA \cdot m^2$, the minimum and maximum value are -128 and 127 respectively and the resolution is 1. In other words, the value -128 and 127 correspond to a MTQ_DIPOLE value of -0.128 and 0.127. The values for the data bytes however must be an unsigned number. Therefore the range from -128 to 127 must be shifted such that a value of 0 represents -128 and a value of 255 represent 127. If we do this for the aforementioned MTQ_DIPOLE_X, MTQ_DIPOLE_Y and MTQ_DIPOLE_Z values, then we get the value 249, 251 and 81, which correspond with 0xF9, 0xFB and 0x51 hexadecimal. In Table B-3 the complete parameter upload command is shown.

Byte 1	**Byte 2**	**Byte 3**	**Byte 4 to 6**		
0x76	0x83	0x25	0xF9	0xFB	0x51

Table B-3: Complete parameter upload command to update the three parameters: MTQ_DIPOLE_X, MTQ_DIPOLE_Y and MTQ_DIPOLE_Z

In the example above the three parameters were each exactly one byte. It can also happen that we want to update a parameter that is only a few bits long. Take for example GEN_TWI_RETRIES parameter, which is only 3 bits. These 3 bits are then left aligned in a byte as shown in Table B-4.

Data byte							
GEN_TWI_RETRIES			"00000"				
B7	B6	B5	B4	B3	B2	B1	B0

Table B-4: Bit alignment of parameters shorter than a byte

If we also want to update the parameter after GEN_TWI_RETRIES, which is 9 bits long, then a part of this parameter is left aligned in the first data byte and the remaining part left aligned in a second data byte, as shown in Table B-5.

Data byte 1								**Data byte 2**							
GEN_TWI_RETRIES			GEN_..._TICKS					GEN_..._TICKS				"0000"			
B7	B6	B5	B4	B3	B2	B1	B0	B7	B6	B5	B4	B3	B2	B1	B0

Table B-5: Bit alignment of two parameters

B-1 Parameter descriptions

In this section a brief description of all the parameter upload parameters is given. Table B-7 gives a more detailed overview of the bit size, the maximum value, the minimum value, the resolution and the default value of these parameters.

ETC_DELTA_T: Time correction in seconds which is applied to the ETC. The parameter resolution is affected by the value of ETC_CLOCKDIV.

GEN_TWI_RETRIES: The number of attempts the application retries a TWI connection in case errors.

GEN_MODE_END_BUFFER_TICKS: The amount of buffer time at the end of any operational mode execution. The resolution of this parameter depends on the micro processor/controller (i.e. AT91SAM9G45 or Atmega128A1).

GEN_AUTOSW_TUMBLE_LIMITER_ENABLE: Enable auto switch back to Detumble mode when exceeding the GEN_TUMBLE_LIMIT threshold (i.e. $L_{tumbling}$).

GEN_TUMBLING_LIMIT: The threshold value at which a auto switch back to Detumble mode is performed.

GEN_TUMBLE_PARAM_LPFALPHA: The low-pass filter constant, α_{tumble} used in the tumble parameter calculation.

MTM_GAIN_gauss: Hardware gain for magnetometer which is related to the MTM_GAINSET. See the HMC588RL Datasheet for more information.

MTM_GAINSET: Hardware gain configuration setting for the magnetometer. See the HMC588RL Datasheet for more information. If this parameter is changed the MTM_GAIN_gauss should also be changed accordingly.

MTM_BIASCORR_X/Y/Z: Static magnetometer bias correction value. This bias correction is applied before transformation to correct for the hard iron distortions

MTM_TRANSFORM_XX/XY/XZ/YX/YY/YZ/ZX/ZY/ZZ: The 9 components of the transformation matrix that is applied to the magnetometer measurements to correct for the soft iron distortions. The transformation is applied after static bias correction.

MTM_COUNTS_VALID_MIN_X/Y/Z: Minimum amount of counts for a valid normal magnetometer measurement. Measurements with lower counts are marked invalid. This parameter is dependent on the gain setting.

MTM_COUNTS_VALID_MAX_X/Y/Z: Maximum amount of counts for a valid normal magnetometer measurement. Measurements with higher counts are marked invalid. This parameter is dependent on the gain setting.

MTM_BM_CONFIG: Configuration setting to switch between a positive or negative bias magnetometer measurement or none at all

MTM_BM_COUNTS_VALID_MIN_X/Y/Z: Minimum amount of counts for a valid positive bias magnetometer measurement. Measurements with lower counts are marked invalid. This parameter is dependent on the gain setting.

MTM_BM_COUNTS_VALID_MAX_X/Y/Z: Maximum amount of counts for a valid positive bias magnetometer measurement. Measurements with higher counts are marked invalid. This parameter is dependent on the gain setting.

MTM_BM_FIELD_EXPECTED_OS_X/Y/Z: Expected magnetic field for a positive bias measurement.

MTQ_DIPOLE_X/Y/Z: The dipoles produced by each magnetorquer.

MTQ_ACTUATION_TIME_MIN_X/Y/Z: The minimal time for which a magnetorquer is switched on. For torque durations lower than this no torquing is performed.

MDET_SYNC_DS_COUNT: The Amount of detumble steps to conduct within a single sync period (i.e. 2 seconds)

MDET_MEAS_LPFALPHA: The low-pass filter constant, α_{meas}, for the low-pass filter that filters the measurements. If set to 1 no low-pass filtering is performed.

MDET_BDOT_LPFALPHA: The low-pass filter constant, α_{bdot}, for the low-pass filter that filters the Bdot. If set to 1 no low-pass filtering is performed.

MDET_BDOT_DIPOLE_GAIN: Design parameter k that is used in the Bdot control law to calculated the magnetic dipole needed to counter act tumbling of the satellite.

MDET_TUMBLE_PARAM_ENTER: The initial value, P_{tumble_0} of the tumble parameter. Also with every switch to Detumble mode the tumble parameter is set to this value.

MDET_TUMBLE_PARAM_EXIT: The tumble parameter exit value. Every time a switch from Detumble mode to an other mode is made, the tumble parameter is set to this value.

GEN_AUTOSW_FROM_MDET_ENABLE: Configuration setting to enable or disable auto switching from Detumble mode to CSP/FSP mode when the satellite is detumbled.

GEN_AUTOSW_FROM_MCSP_ENABLE: Configuration setting to enable or disable auto switching from CSP to FSP when TLE's are available.

GEN_AUTOSW_MFSP_MGST_ENABLE: Configuration setting to enable or disable auto switching between FSP and GST using on-board ground station pass prediction

GEN_AUTOSW_MTHP_ENABLE: Configuration setting to enable or disable auto switching from FSP/GST to THP on the next ground station pass, which is determined by the on-board ground station pass prediction.

GEN_DISABLE_ACTUATION_IN_ECLIPSE: Configuration setting to enable or disable actuation of the magnetorquers and reaction wheels when sun sensors detect eclipse.

ETC_CLOCKDIV: The clock divider used to divide the 32.768 kHz clock of the ETC. This parameter determines the timing resolution of the ETC.

ETC_DRIFTCORR_INTERVAL: The time interval at which a correction is applied to the ETC. The resolution of this parameter is determined by the ETC_CLOCKDIV.

ETC_DRIFTCORR_CORRECTION: THe correction value that is applied to the ETC each ETC_DRIFTCORR_INTERVAL. The resolution of this parameter is determined by the ETC_CLOCKDIV.

VE_EPOCH: Vernal equinox epoch, relative to Delfi-n3Xt base epoch (i.e. 01-01-2010 00:00:00).

I_EF_ALIGNED_EPOCH: Epoch, relative to Delfi-n3Xt base epoch, at which the ECI and ECEF frames are aligned.

SGP4_TLE_EPOCH: The TLE epoch, relative to Delfi-n3Xt base epoch.

SGP4_TLE_INCLINATION: TLE parameter: Inclination of the orbit.

SGP4_TLE_RAAN: TLE parameter: RAAN.

SGP4_TLE_ECCENTR: TLE parameter: Eccentricity.

SGP4_TLE_ARGPERIG: TLE parameter: Argument of Perigee.

SGP4_TLE_MANOM: TLE parameter: Mean Anomaly.

SGP4_TLE_MMOT: TLE parameter: Mean Motion.

SGP4_TLE_BSTAR_SIGNIF: TLE parameter: Bstar drag significant part ($bstar = signif * 2^{-expo}$).

SGP4_TLE_BSTAR_EXPO: Bstar drag base 2 exponent part ($bstar = signif * 2^{-expo}$).

ACON_SAT_ROT_INERTIA_X/Y/Z: The rotational inertia values of the satellite, used in the moment of inertia matrix.

ACON_RW_ROT_INERTIA_X/Y/Z: The rotational inertia values of the reaction wheels, used to calculate commanded accelerations and stored angular momentum.

ACON_RW_RPM_DES_X/Y/Z: The desired rotational rate of each reaction wheel when unloaded.

ACON_RW_MAX_ACCEL_X/Y/Z: The maximum allowed commanded acceleration sent to each reaction wheel.

ACON_d: The gain factor used in the Quaternion Feedback Regulator control law to calculated the total control torque.

ACON_k: Another gain factor used in the Quaternion Feedback Regulator control law to calculated the total control torque.

ACON_THRESHOLD_UL_ON: The threshold value at which the reaction wheels should be unloaded.

ACON_THRESHOLD_UL_OFF: The threshold value at which unloading of the reaction wheels should be stopped.

ADET_LPFALPHA_m: A low-pass filter constant for the low pas filter in CSP mode that filters magnetometer and sun sensor measurements.

ADET_LPFALPHA_w: A low-pass filter constant for the low pass filter in CSP mode that filters the rotational rates.

ADET_OMEGA_OUTLIER: The threshold value in CSP for the rotational rate outlier.

ADET_MAT_EXP_TERMS: The amount of terms to use for a Taylor series matrix exponential approximation used in the Kalman filter.

ADET_COVMAT_PROCNOISE_SIGMA_Q: The quaternion part of the propagation noise covariance matrix used in the Kalman filter.

ADET_COVMAT_PROCNOISE_SIGMA_O: The omega part of the propagation noise covariance matrix used in the Kalman filter.

ADET_COVMAT_INIT_QC1/C2/C3/C4: The first four diagonal entries of the covariance matrix used in the Kalman filter.

ADET_COVMAT_INIT_OC1/C2/C3: The last three diagonal entries of the covariance matrix used in the Kalman filter.

ADET_SIGMA_MTM: The value for the magnetometer measurement noise, used in the Kalman filter.

ADET_SIGMA_SS: The value for the sun sensor measurement noise, used in the Kalman filter.

ADET_THRESHOLD_ALTERNATIVE_PIVOT: Alternative pivot for DCM to attitude quaternion conversion is used when quaternion component q_4 smaller than or equal to this threshold.

MDET_TUMBLE_PARAM_DETUMBLED_LIM: The threshold value $L_{detumbled}$ for the tumble parameter at which the satellite is considered detumbled.

MDET_TUMBLE_PARAM_DETUMBLED_TIME: The time that the tumble parameter must remain below MDET_TUMBLE_PARAM_DETUMBLED_LIM before switching from Detumble mode to an other mode.

MCSP_ATTITUDE_DES_QC1/C2/C3/C4: The four quaternion components for the desired attitude of the satellite in CSP and FSP mode.

MCSP_OMEGA_DES_X/Y/Z: The desired rotational rates of the satellite in CSP mode.

MFSP_EARTH_AXIS_TILT: The tilt of the Earth's axis.
MFSP_DURATION_YEAR: The duration of a year in seconds.
MFSP_RA_OS: An offset value added to the right ascension angle.
MFSP_DECL_OS: An offset value added to the declination angle.
MFSP_OMEGA_DES_X/Y/Z: The desired rotational rates of the satellite in FSP mode.
MGST_REORIENTATION_PERIOD: The maximum time duration for reorienting the satellite when switched from FSP to GST.
MGST_THRESHOLD_GST_IN_VIEW: The threshold value for the angle below which the GST is considered in view.
MGST_STATION_COUNT: The number of ground stations.
MGST_STATION_A/B/C_POS_X/Y/Z: The positions (i.e. the X,Y and Z coordinates) of the three ground stations in the ECEF reference frame.
MGST_MIN_ELEVATION: The elevation value that is subtracted from the MGST_THRESHOLD_GST_IN_VIEW angle that defines the. GST_IN_VIEW cone
MGST_GS_TO_GS_SWITCH_DIFFERENCE: On closest ground station determination the closest new ground station must be at least this much closer.
MGST_EARTH_RADIUS: The radius of the Earth in kilometers.
MGST_DURATION_SIDEREAL_DAY: The duration of a sidereal day in seconds.
SUNS_BIASCORR_XP/XN/YP/YN/ZP/ZN_Qn: The bias correction values for the sun sensor quadrants. This value is added to the determined output voltage of each quadrant.

Upload ID	Parameter name	Unit	Bitsize	Min. value	Max. value	Resolution	Default value
1	ETC_DELTA_T	$[s]$	32	-214761472	214761471.9	0	0
2	GEN_TWI_RETRIES	[-]	3	0	7	1	2
3	GEN_MODE_END_BUFFER_TICKS	$[ms]$	9	0	15594.48242	30.51757813	5004.882813
4	GEN_AUTOSW_TUMBLE_LIMITER_ENABLE	[-]	1	0	1	1	1
5	GEN_TUMBLING_LIMIT	$[mT/s]$	16	0	65.535	0.001	1
6	GEN_TUMBLE_PARAM_LPFALPHA	[-]	16	0	1.3107	0.00002	0.0008
7	MTM_GAIN_gauss	$[LSB/Gauss]$	12	0	4095	1	660
8	MTM_GAINSET	[-]	3	0	7	1	3
9	MTM_BIASCORR_X	$[mT]$	12	-204.8	204.7	0.1	2.2
10	MTM_BIASCORR_Y	$[mT]$	12	-204.8	204.7	0.1	-17.3
11	MTM_BIASCORR_Z	$[mT]$	12	-204.8	204.7	0.1	5.3
12	MTM_TRANSFORM_XX	[-]	12	-2.048	2.047	0.001	1.106
13	MTM_TRANSFORM_XY	[-]	12	-2.048	2.047	0.001	-0.135
14	MTM_TRANSFORM_XZ	[-]	12	-2.048	2.047	0.001	-0.01
15	MTM_TRANSFORM_YX	[-]	12	-2.048	2.047	0.001	-0.139
16	MTM_TRANSFORM_YY	[-]	12	-2.048	2.047	0.001	1.31
17	MTM_TRANSFORM_YZ	[-]	12	-2.048	2.047	0.001	-0.02
18	MTM_TRANSFORM_ZX	[-]	12	-2.048	2.047	0.001	-0.11
19	MTM_TRANSFORM_ZY	[-]	12	-2.048	2.047	0.001	-0.84
20	MTM_TRANSFORM_ZZ	[-]	12	-2.048	2.047	0.001	1.05
21	MTM_COUNTS_VALID_MIN_X	[-]	12	-2048	2047	1	-500
22	MTM_COUNTS_VALID_MIN_Y	[-]	12	-2048	2047	1	-500
23	MTM_COUNTS_VALID_MIN_Z	[-]	12	-2048	2047	1	-500
24	MTM_COUNTS_VALID_MAX_X	[-]	12	-2048	2047	1	500
25	MTM_COUNTS_VALID_MAX_Y	[-]	12	-2048	2047	1	500
26	MTM_COUNTS_VALID_MAX_Z	[-]	12	-2048	2047	1	500
27	MTM_BM_CONFIG	[-]	2	0	2	1	1
28	MTM_BM_COUNTS_VALID_MIN_X	[-]	11	0	2047	1	512
29	MTM_BM_COUNTS_VALID_MIN_Y	[-]	11	0	2047	1	512
30	MTM_BM_COUNTS_VALID_MIN_Z	[-]	11	0	2047	1	512
31	MTM_BM_COUNTS_VALID_MAX_X	[-]	11	0	2047	1	1024
32	MTM_BM_COUNTS_VALID_MAX_Y	[-]	11	0	2047	1	1024
33	MTM_BM_COUNTS_VALID_MAX_Z	[-]	11	0	2047	1	1024
34	MTM_BM_FIELD_EXPECTED_OS_X	$[mT]$	8	-12.8	12.7	0.1	-0.2
35	MTM_BM_FIELD_EXPECTED_OS_Y	$[mT]$	8	-12.8	12.7	0.1	-4.2
36	MTM_BM_FIELD_EXPECTED_OS_Z	$[mT]$	8	-12.8	12.7	0.1	0.3
37	MTQ_DIPOLE_X	$[mA \cdot m^2]$	8	-128	127	1	60
38	MTQ_DIPOLE_Y	$[mA \cdot m^2]$	8	-128	127	1	60
39	MTQ_DIPOLE_Z	$[mA \cdot m^2]$	8	-128	127	1	-60
40	MTQ_ACTUATION_TIME_MIN_X	$[ms]$	7	0	3875.73	30.51	213.62
41	MTQ_ACTUATION_TIME_MIN_Y	$[ms]$	7	0	3875.73	30.51	213.62
42	MTQ_ACTUATION_TIME_MIN_Z	$[ms]$	7	0	3875.73	30.51	213.62

Continued on next page

Table B-7 – *Continued from previous page*

Upload ID	Parameter name	Unit	Bitsize	Min. value	Max. value	Resolution	Default value
43	MDET_SYNC_DS_COUNT	[-]	3	1	8	1	6
44	MDET_MEAS_LPFALPHA	[-]	16	0	1.3107	0.00002	0.4
45	MDET_BDOT_LPFALPHA	[-]	16	0	1.3107	0.00002	0.4
46	MDET_BDOT_DIPOLE_GAIN	$[mAm^2/(mT/s)]$	8	-128	127	1	60
47	MDET_TUMBLE_PARAM_ENTER	$[mT/s]$	16	0	65.535	0.001	1.2
48	MDET_TUMBLE_PARAM_EXIT	$[mT/s]$	16	0	65.535	0.001	0.2
49	GEN_AUTOSW_FROM_MDET_ENABLE	[-]	1	0	1	1	1
50	GEN_AUTOSW_FROM_MCSP_ENABLE	[-]	1	0	1	1	1
51	GEN_AUTOSW_MFSP_MGST_ENABLE	[-]	1	0	1	1	1
52	GEN_AUTOSW_MTHP_ENABLE	[-]	1	0	1	1	1
53	GEN_DISABLE_ACTUATION_IN_ECLIPSE	[-]	1	0	1	1	1
54	ETC_CLOCKDIV	[-]	16	3	65538	1	3277
55	ETC_DRIFTCORR_INTERVAL	$[s]$	16	0	0	0	0
56	ETC_DRIFTCORR_CORRECTION	$[s]$	9	0	0	0	0
57	VE_EPOCH	$[s]$	32	0	429496729.5	0.1	0
58	I_EF_ALIGNED_EPOCH	[s]	32	0	429496729.5	0.1	0
59	SGP4_TLE_EPOCH	$[s]$	32	0	429496729.5	0.1	0
60	SGP4_TLE_INCLINATION	[°]	22	0	419.4303	1.0E-04	0
61	SGP4_TLE_RAAN	[°]	22	0	419.4303	1.0E-04	0
62	SGP4_TLE_ECCENTR	[-]	24	0	1.6777215	1.0E-07	0
63	SGP4_TLE_ARGPERIG	[°]	22	0	419.4303	1.0E-04	0
64	SGP4_TLE_MANOM	[°]	22	0	419.4303	1.0E-04	0
65	SGP4_TLE_MMOT	$[rev/day]$	24	0	16.777215	1.0E-06	0
66	SGP4_TLE_BSTAR_SIGNIF	[-]	18	0	262143	1	0
67	SGP4_TLE_BSTAR_EXPO	[-]	8	0	255	1	0
68	ACON_SAT_ROT_INERTIA_X	$[kg \cdot m^2]$	12	0	0.4095	0.0001	0.055
69	ACON_SAT_ROT_INERTIA_Y	$[kg \cdot m^2]$	12	0	0.4095	0.0001	0.055
70	ACON_SAT_ROT_INERTIA_Z	$[kg \cdot m^2]$	12	0	0.4095	0.0001	0.017
71	ACON_RW_ROT_INERTIA_X	$[kg \cdot m^2]$	12	-2.05E-06	2.05E-06	1.00E-09	2.95E-07
72	ACON_RW_ROT_INERTIA_Y	$[kg \cdot m^2]$	12	-2.05E-06	2.05E-06	1.00E-09	2.95E-07
73	ACON_RW_ROT_INERTIA_Z	$[kg \cdot m^2]$	12	-2.05E-06	2.05E-06	1.00E-09	2.95E-07
74	ACON_RW_RPM_DES_X	$[RPM]$	8	-25600	25400	200	0
75	ACON_RW_RPM_DES_Y	$[RPM]$	8	-25600	25400	200	0
76	ACON_RW_RPM_DES_Z	$[RPM]$	8	-25600	25400	200	0
77	ACON_RW_MAX_ACCEL_X	$[RPM/s]$	8	0	255	1	150
78	ACON_RW_MAX_ACCEL_Y	$[RPM/s]$	8	0	255	1	150
79	ACON_RW_MAX_ACCEL_Z	$[RPM/s]$	8	0	255	1	150
80	ACON_d	$[Hz]$	16	0	0.65535	0.00001	0.0154
81	ACON_k	$[Hz^2]$	16	0	0.65535	0.00001	0.00023
82	ACON_THRESHOLD_UL_ON	$[mN \cdot m \cdot s]$	8	0	19125	75	225
83	ACON_THRESHOLD_UL_OFF	$[mN \cdot m \cdot s]$	8	0	19125	75	225
84	ADET_LPFALPHA_m	[-]	16	0	1.3107	0.00002	0.1

Continued on next page

Table B-7 – *Continued from previous page*

Upload ID	Parameter name	Unit	Bitsize	Min. value	Max. value	Resolution	Default value
85	ADET_LPFALPHA_w	[-]	16	0	1.3107	0.00002	0.25
86	ADET_OMEGA_OUTLIER	$[°/s]$	8	0	2.55	0.01	1
87	ADET_MAT_EXP_TERMS	[-]	8	0	255	1	20
88	ADET_COVMAT_PROCNOISE_SIGMA_Q	[-]	32	0	4.294967295	1.00E-09	5.00E-03
89	ADET_COVMAT_PROCNOISE_SIGMA_O	$[°^2/s^2]$	32	0	4.294967295	1.00E-09	3.00E-05
90	ADET_COVMAT_INIT_QC1	[-]	32	0	4.294967295	1.00E-09	1.00E-02
91	ADET_COVMAT_INIT_QC2	[-]	32	0	4.294967295	1.00E-09	1.00E-02
92	ADET_COVMAT_INIT_QC3	[-]	32	0	4.294967295	1.00E-09	1.00E-02
93	ADET_COVMAT_INIT_QC4	[-]	32	0	4.294967295	1.00E-09	1.00E-02
94	ADET_COVMAT_INIT_OC1	$[(°)^2/s^2]$	32	0	4.294967295	1.00E-09	1.00E-06
95	ADET_COVMAT_INIT_OC2	$[(°)^2/s^2]$	32	0	4.294967295	1.00E-09	1.00E-06
96	ADET_COVMAT_INIT_OC3	$[(°)^2/s^2]$	32	0	4.294967295	1.00E-09	1.00E-06
97	ADET_SIGMA_MTM	[-]	32	0	4.294967295	1.00E-09	5.00E-03
98	ADET_SIGMA_SS	[-]	32	0	4.294967295	1.00E-09	3.00E-05
99	ADET_THRESHOLD_ALTERNATIVE_PIVOT	[-]	32	0	0.42949673	1.00E-10	1.00E-10
100	MDET_TUMBLE_PARAM_DETUMBLED_LIM	$[mT/s]$	8	0	2.55	0.01	0.2
101	MDET_TUMBLE_PARAM_DETUMBLED_TIME	$[s]$	8	0	25500	100	900
102	MCSP_ATTITUDE_DES_QC1	[-]	12	-1	1.0475	0.0005	0
103	MCSP_ATTITUDE_DES_QC2	[-]	12	-1	1.0475	0.0005	0
104	MCSP_ATTITUDE_DES_QC3	[-]	12	-1	1.0475	0.0005	0
105	MCSP_ATTITUDE_DES_QC4	[-]	12	-1	1.0475	0.0005	1
106	MCSP_OMEGA_DES_X	$[°/s]$	8	-6.4	6.35	0.05	0
107	MCSP_OMEGA_DES_Y	$[°/s]$	8	-6.4	6.35	0.05	0
108	MCSP_OMEGA_DES_Z	$[°/s]$	8	-6.4	6.35	0.05	0
109	MFSP_EARTH_AXIS_TILT	$[°]$	24	0	167.77215	1.00E-05	23.44
110	MFSP_DURATION_YEAR	$[s]$	32	0	429496729.5	0.1	31557600
111	MFSP_RA_OS	$[°]$	12	-204.8	204.7	0.1	0
112	MFSP_DECL_OS	$[°]$	12	-204.8	204.7	0.1	0
113	MFSP_OMEGA_DES_X	$[°/s]$	8	-8	7.9375	0.0625	0
114	MFSP_OMEGA_DES_Y	$[°/s]$	8	-8	7.9375	0.0625	0
115	MFSP_OMEGA_DES_Z	$[°/s]$	8	-8	7.9375	0.0625	0
116	MGST_REORIENTATION_PERIOD	$[s]$	14	0	1638.3	0.1	480
117	MGST_THRESHOLD_GST_IN_VIEW	$[°]$	13	0	163.82	0.02	89.9
118	MGST_STATION_COUNT	[-]	2	0	3	1	1
119	MGST_STATION_A_POS_X	$[km]$	13	0	8191	1	3556
120	MGST_STATION_A_POS_Y	$[km]$	13	0	8191	1	5272
121	MGST_STATION_A_POS_Z	$[km]$	13	0	8191	1	486
122	MGST_STATION_B_POS_X	$[km]$	13	0	8191	1	0
123	MGST_STATION_B_POS_Y	$[km]$	13	0	8191	1	0
124	MGST_STATION_B_POS_Z	$[km]$	13	0	8191	1	0
125	MGST_STATION_C_POS_X	$[km]$	13	0	8191	1	0

Continued on next page

Table B-7 – *Continued from previous page*

Upload ID	Parameter name	Unit	Bitsize	Min. value	Max. value	Resolution	Default value
126	MGST_STATION_C_POS_Y	$[km]$	13	0	8191	1	0
127	MGST_STATION_C_POS_Z	$[km]$	13	0	8191	1	0
128	MGST_MIN_ELEVATION	$[^\circ]$	12	-204.8	204.7	0.1	5
129	MGST_GS_TO_GS_SWITCH_DIFFERENCE	$[km]$	13	0	8191	1	10
130	MGST_EARTH_RADIUS	$[km]$	13	0	8191	1	6371
131	MGST_DURATION_SIDEREAL_DAY	$[s]$	32	0	429496729.5	0.1	86163.9
132	SUNS_BIASCORR_XP_Q1	$[mV]$	16	-1024	1023.96875	0.03125	0
133	SUNS_BIASCORR_XP_Q2	$[mV]$	16	-1024	1023.96875	0.03125	0
134	SUNS_BIASCORR_XP_Q3	$[mV]$	16	-1024	1023.96875	0.03125	0
135	SUNS_BIASCORR_XP_Q4	$[mV]$	16	-1024	1023.96875	0.03125	0
136	SUNS_BIASCORR_XN_Q1	$[mV]$	16	-1024	1023.96875	0.03125	0
137	SUNS_BIASCORR_XN_Q2	$[mV]$	16	-1024	1023.96875	0.03125	0
138	SUNS_BIASCORR_XN_Q3	$[mV]$	16	-1024	1023.96875	0.03125	0
139	SUNS_BIASCORR_XN_Q4	$[mV]$	16	-1024	1023.96875	0.03125	0
140	SUNS_BIASCORR_YP_Q1	$[mV]$	16	-1024	1023.96875	0.03125	0
141	SUNS_BIASCORR_YP_Q2	$[mV]$	16	-1024	1023.96875	0.03125	0
142	SUNS_BIASCORR_YP_Q3	$[mV]$	16	-1024	1023.96875	0.03125	0
143	SUNS_BIASCORR_YP_Q4	$[mV]$	16	-1024	1023.96875	0.03125	0
144	SUNS_BIASCORR_YN_Q1	$[mV]$	16	-1024	1023.96875	0.03125	0
145	SUNS_BIASCORR_YN_Q2	$[mV]$	16	-1024	1023.96875	0.03125	0
146	SUNS_BIASCORR_YN_Q3	$[mV]$	16	-1024	1023.96875	0.03125	0
147	SUNS_BIASCORR_YN_Q4	$[mV]$	16	-1024	1023.96875	0.03125	0
148	SUNS_BIASCORR_ZP_Q1	$[mV]$	16	-1024	1023.96875	0.03125	0
149	SUNS_BIASCORR_ZP_Q2	$[mV]$	16	-1024	1023.96875	0.03125	0
150	SUNS_BIASCORR_ZP_Q3	$[mV]$	16	-1024	1023.96875	0.03125	0
151	SUNS_BIASCORR_ZP_Q4	$[mV]$	16	-1024	1023.96875	0.03125	0
152	SUNS_BIASCORR_ZN_Q1	$[mV]$	16	-1024	1023.96875	0.03125	0
153	SUNS_BIASCORR_ZN_Q2	$[mV]$	16	-1024	1023.96875	0.03125	0
154	SUNS_BIASCORR_ZN_Q3	$[mV]$	16	-1024	1023.96875	0.03125	0
155	SUNS_BIASCORR_ZN_Q4	$[mV]$	16	-1024	1023.96875	0.03125	0

Table B-7: Parameter upload parameters

Appendix C

Control Flow Diagrams

In this appendix the control flow graphs of the main control loop and the Detumble mode are given. Section C-1 shows the control flow graph of the main control loop and in Section C-2 the control flow graphs of the Detumble mode are shown, with in Subsection C-2-1 the Detumble main loop and in Subsection C-2-2 the Detumble control step.

C-1 Main control loop

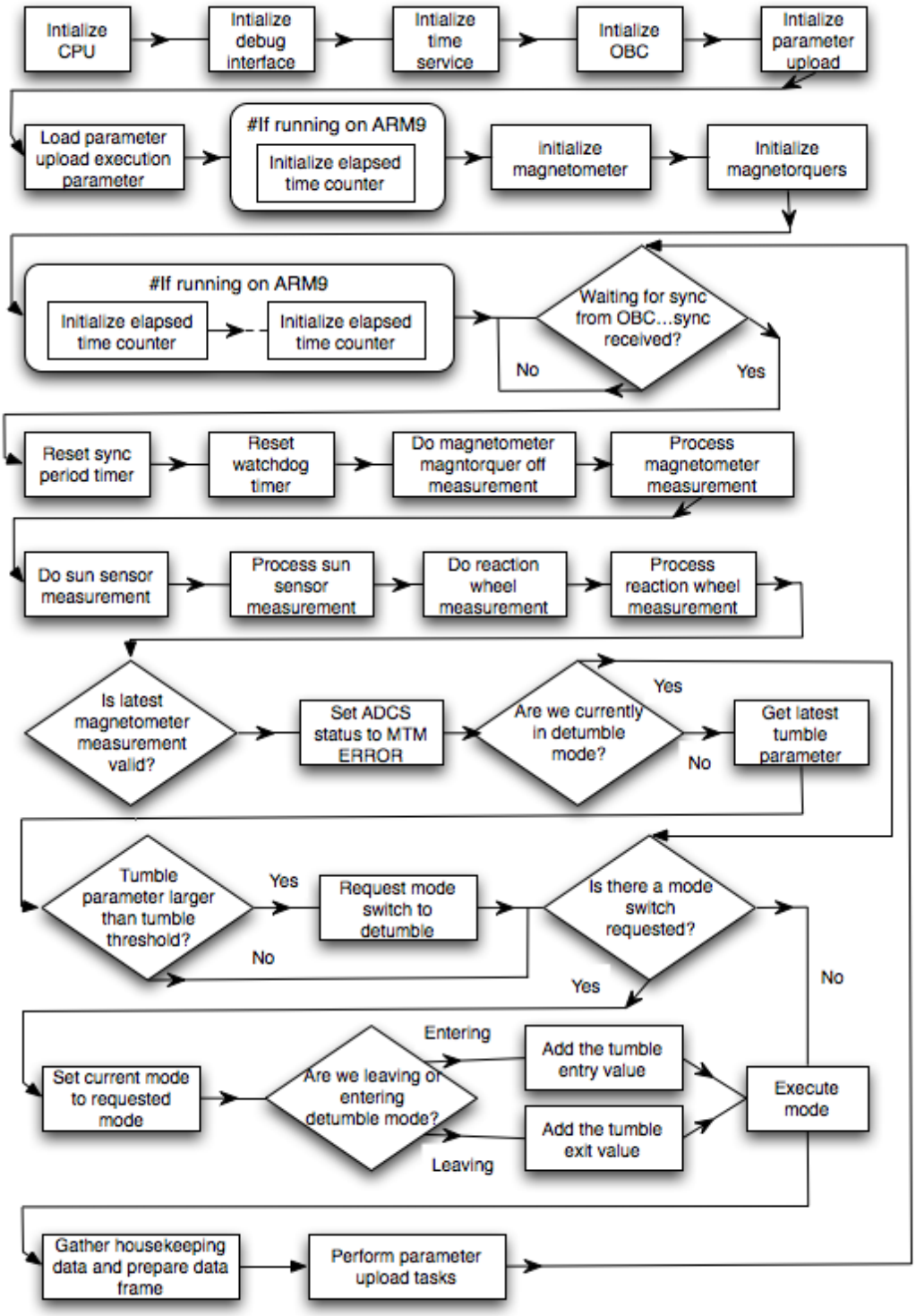


Figure C-1: Control flow graph of the main control loop.

C-2 Detumble mode

C-2-1 Detumble main

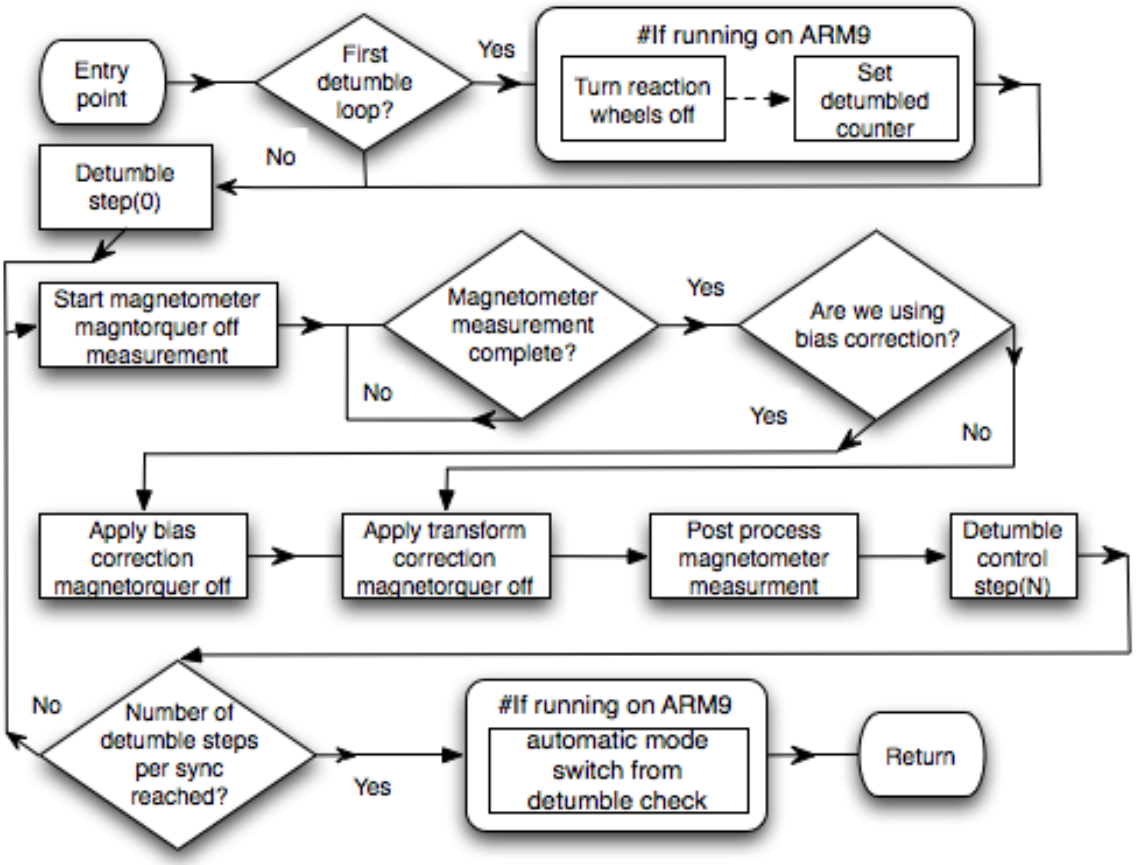


Figure C-2: Control flow graph of the Detumble main loop.

C-2-2 Detumble control step

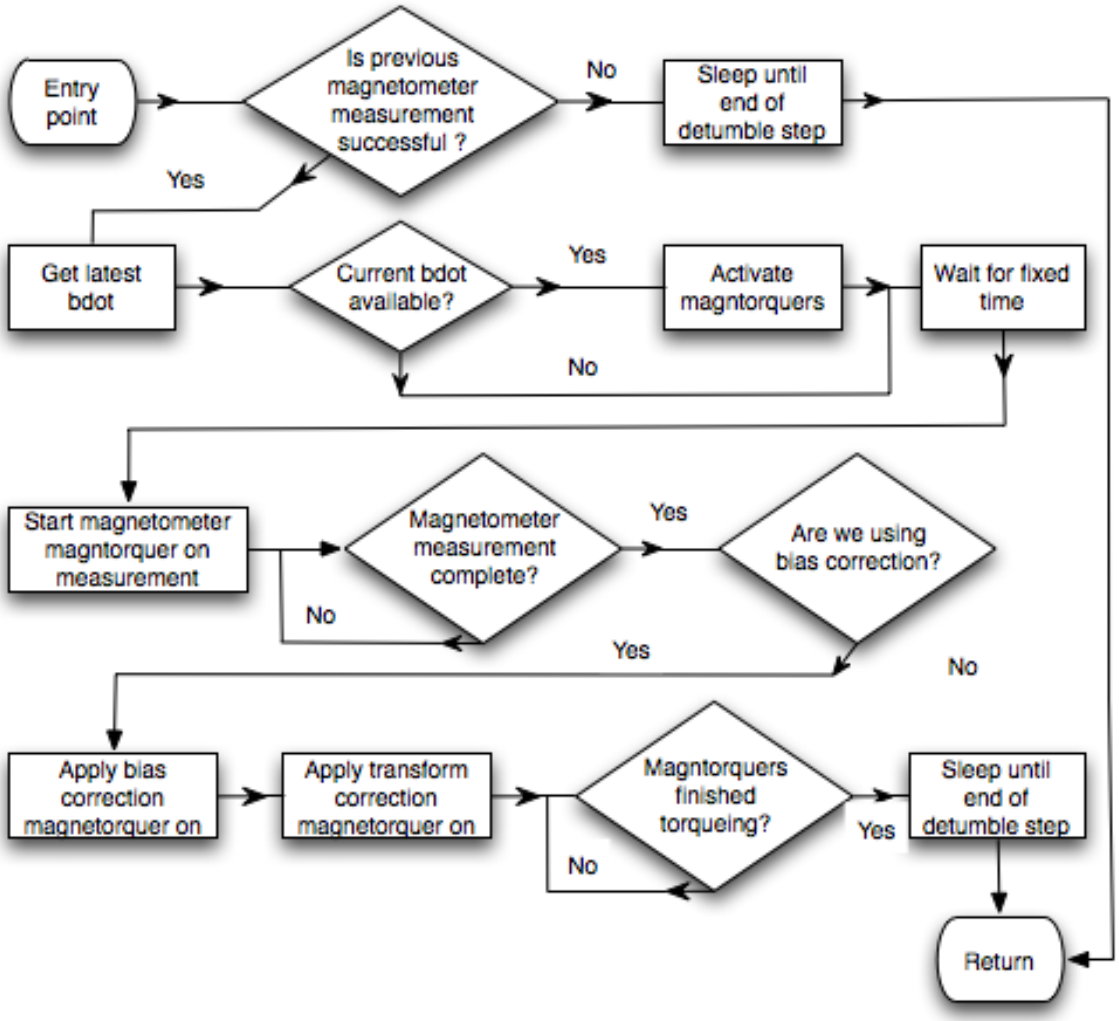


Figure C-3: Control flow graph of a Detumble control step.

Appendix D

Magnetometer verification plots

In this appendix some additional figures are shown that have to do with the verification of the magnetometer. In Section D-1 some extra figures from the built-in self test are shown from which one figure was already shown in Subsection 7-3-1. Section D-2 shows the low-pass filtered bdot for the X, Y and Z direction, from which the Y direction was already shown in Subsection 7-4-1. Furthermore in Section D-3 shows some extra figures related to the calculated magnetic and commanded dipoles, from which one figure was already shown in subsec::magneticcommandeddipole.

D-1 Built-in self test

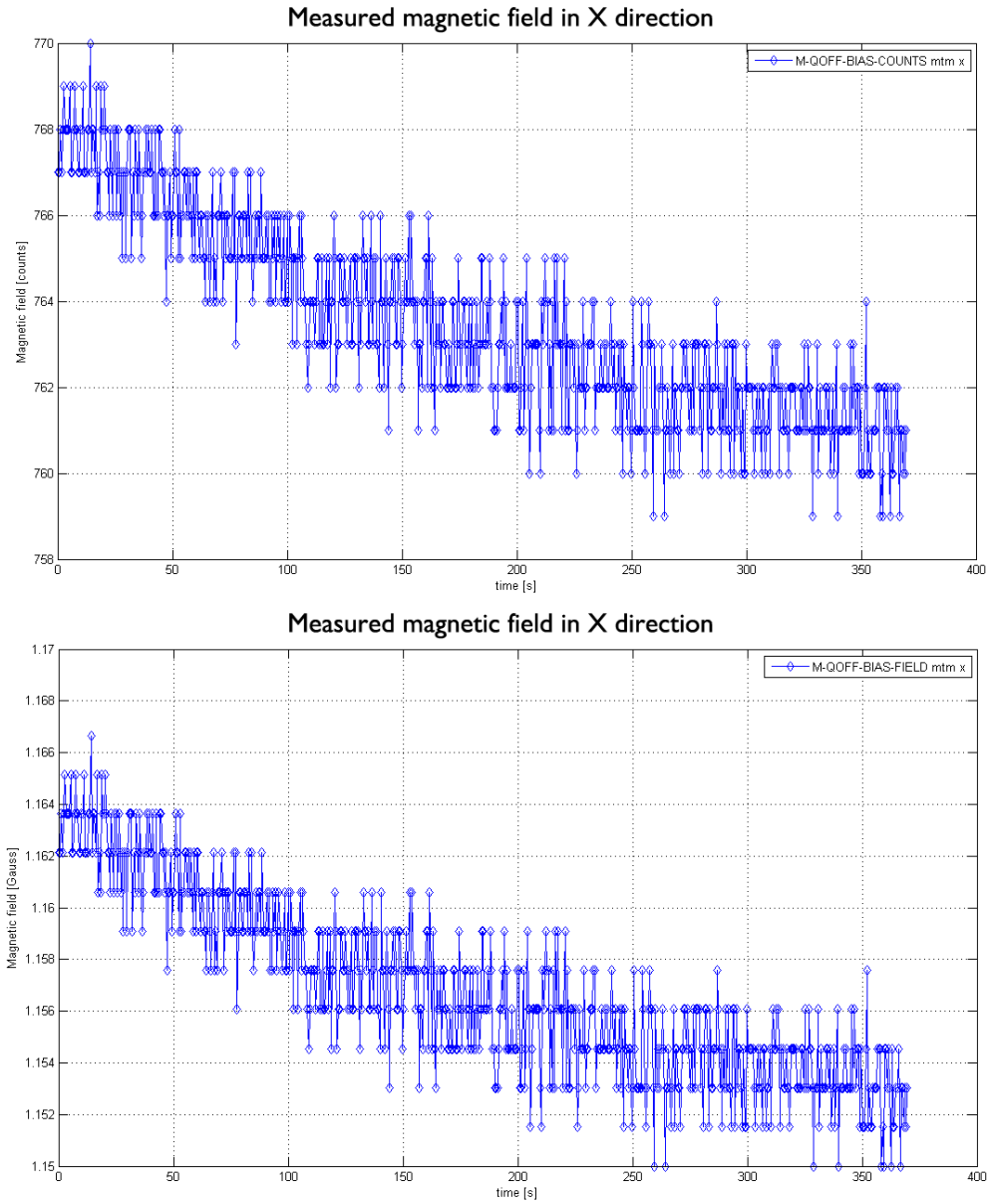


Figure D-1: Magnetic field in the X direction due to the built-in self test. The upper figure showing the raw data and the lower figure showing the magnetic field in Gauss.

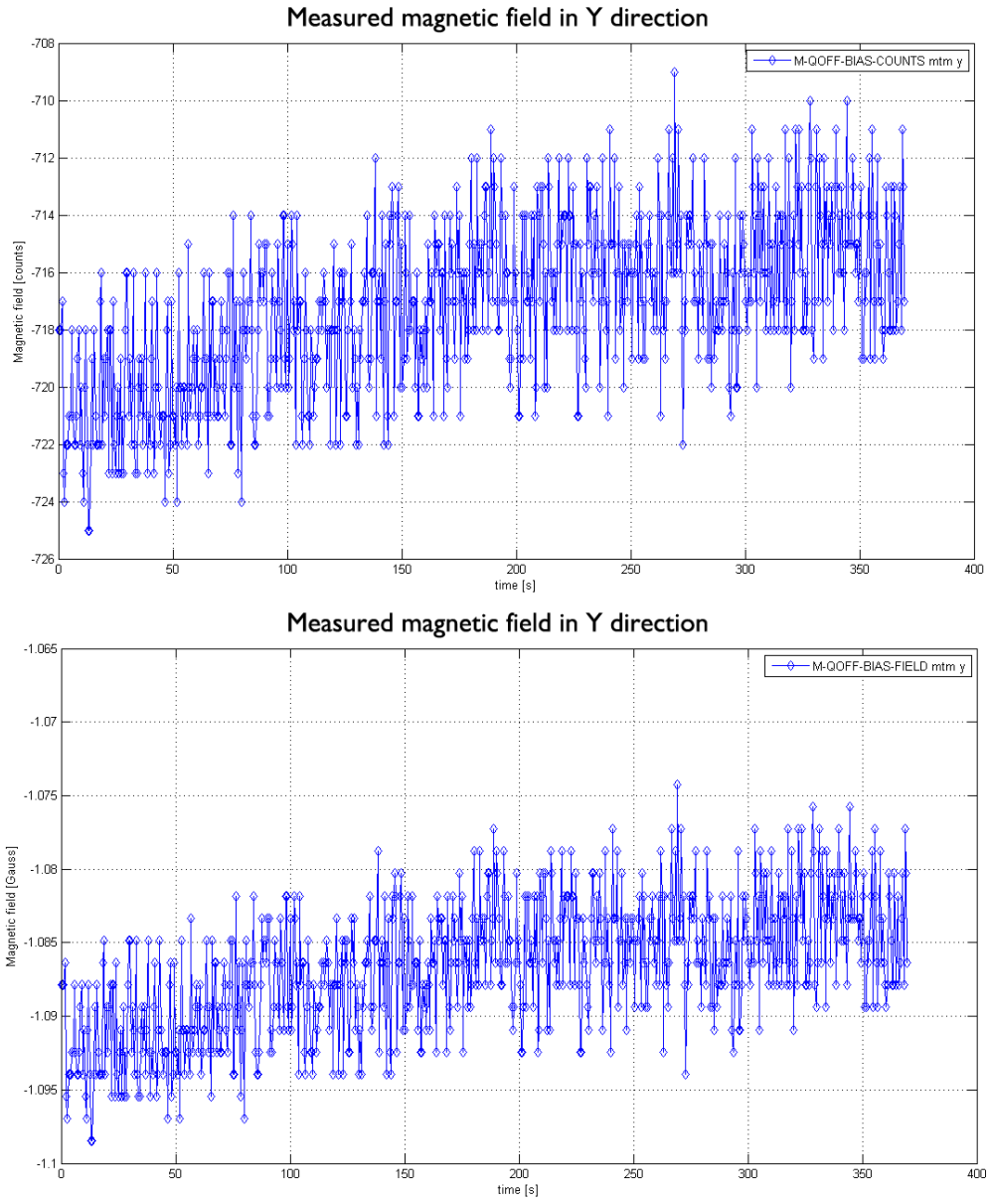


Figure D-2: Magnetic field in the Y direction due to the built-in self test. The upper figure showing the raw data and the lower figure showing the magnetic field in Gauss.

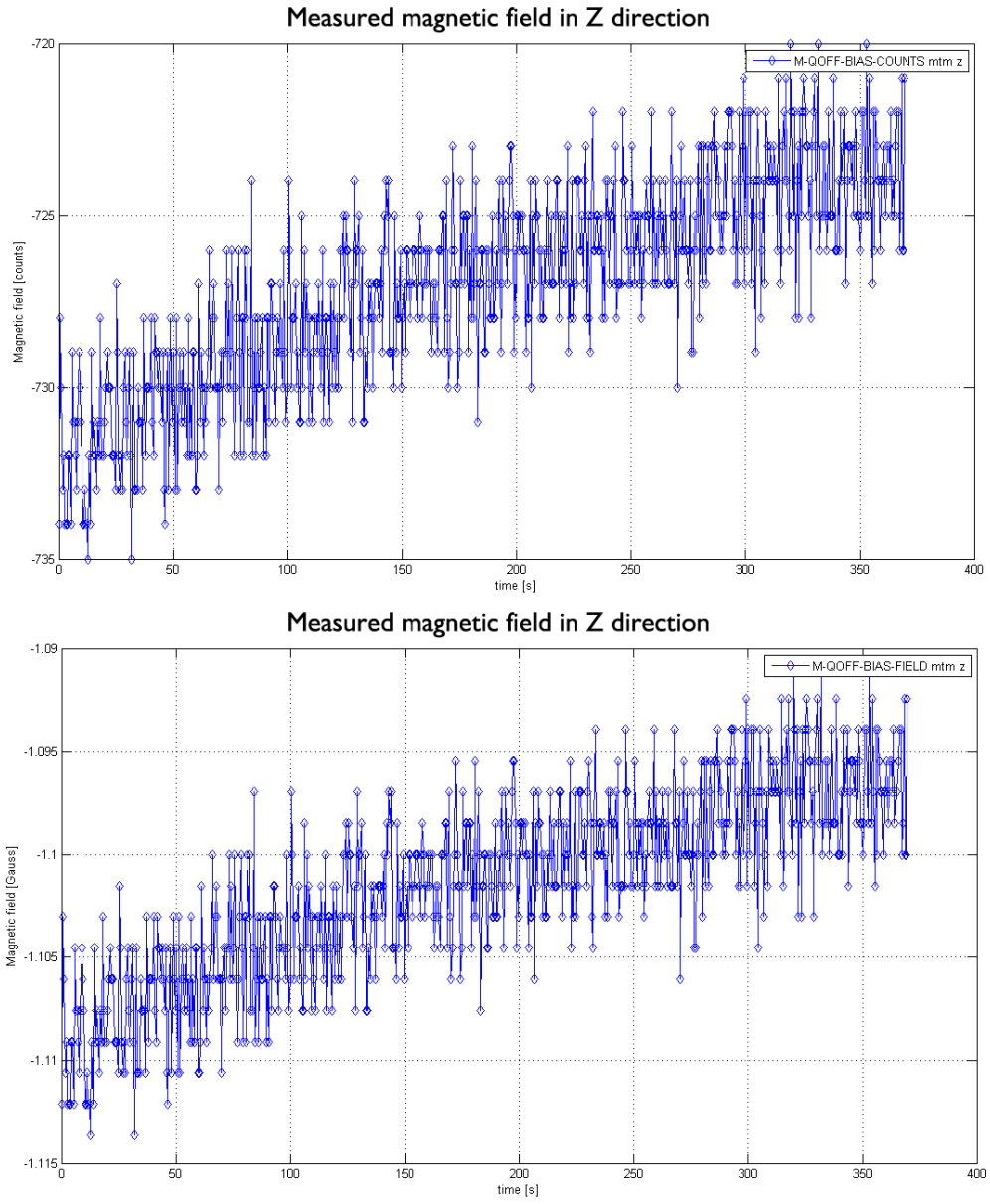


Figure D-3: Magnetic field in the Z direction due to the built-in self test. The upper figure showing the raw data and the lower figure showing the magnetic field in Gauss.

D-2 Low-pass filtered Bdot's

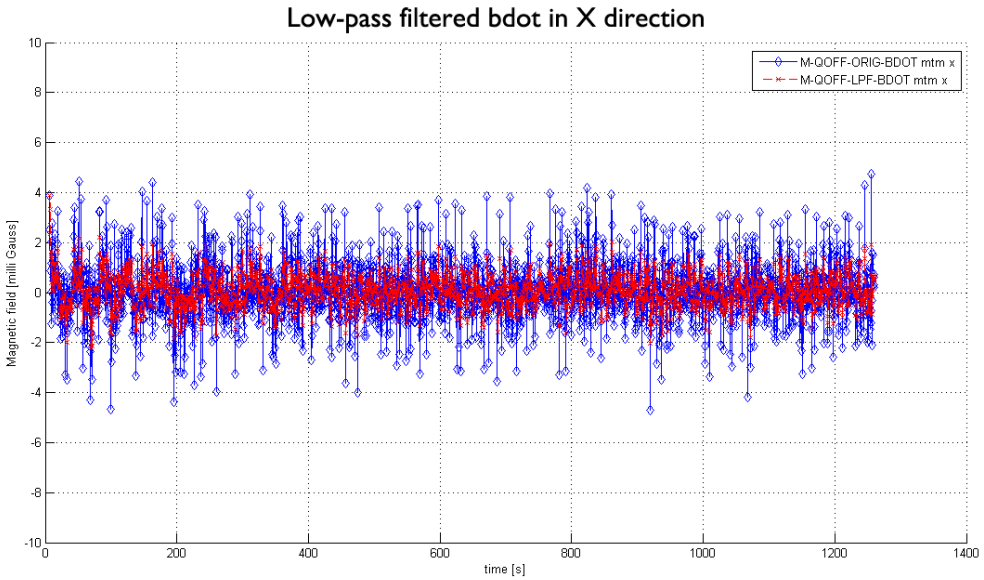


Figure D-4: The low-pass filtered Bdot in the X direction.

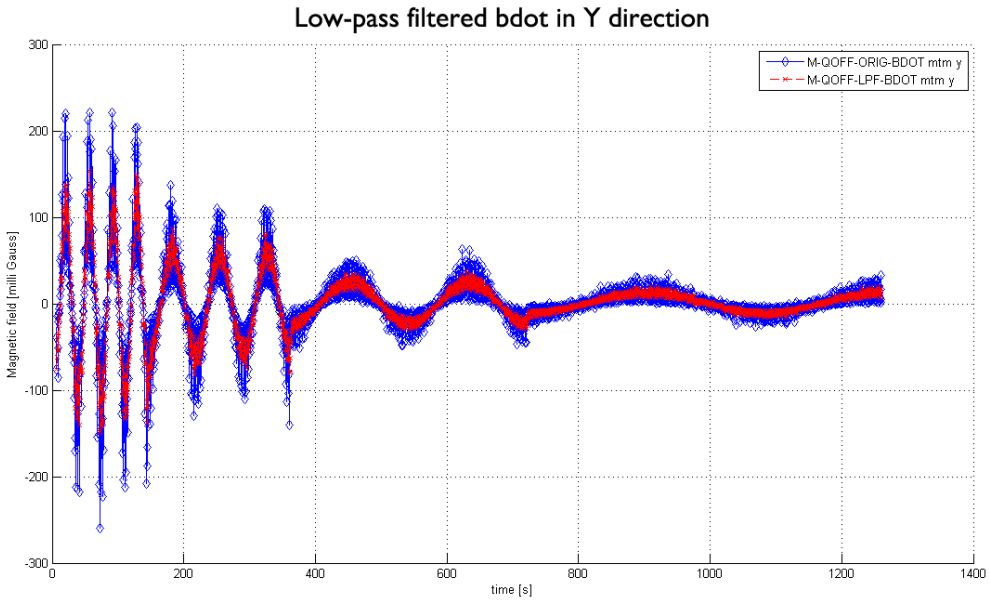


Figure D-5: The low-pass filtered Bdot in the Y direction.

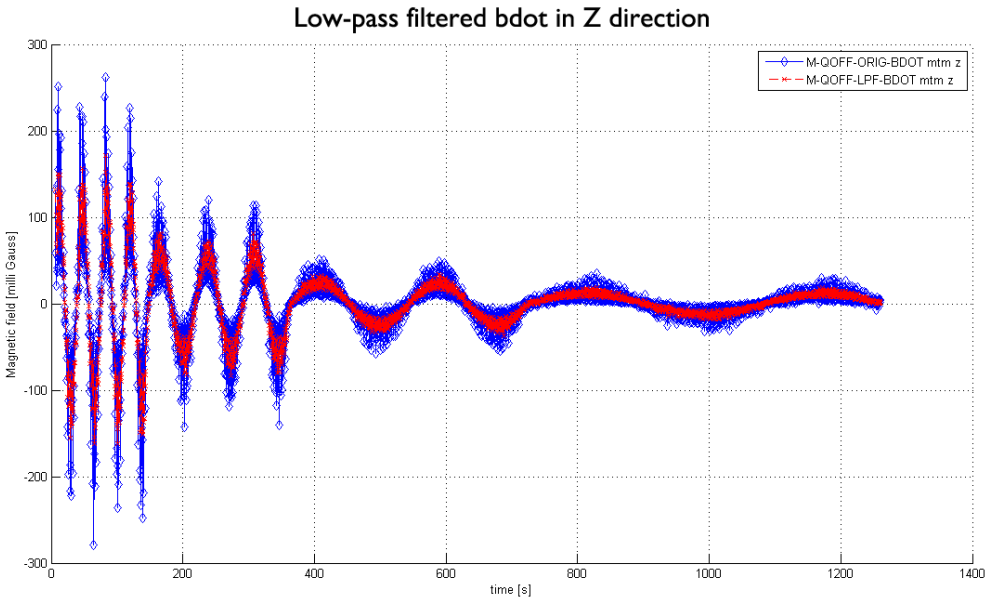


Figure D-6: The low-pass filtered Bdot in the Z direction.

D-3 Magnetic dipoles and commanded dipoles

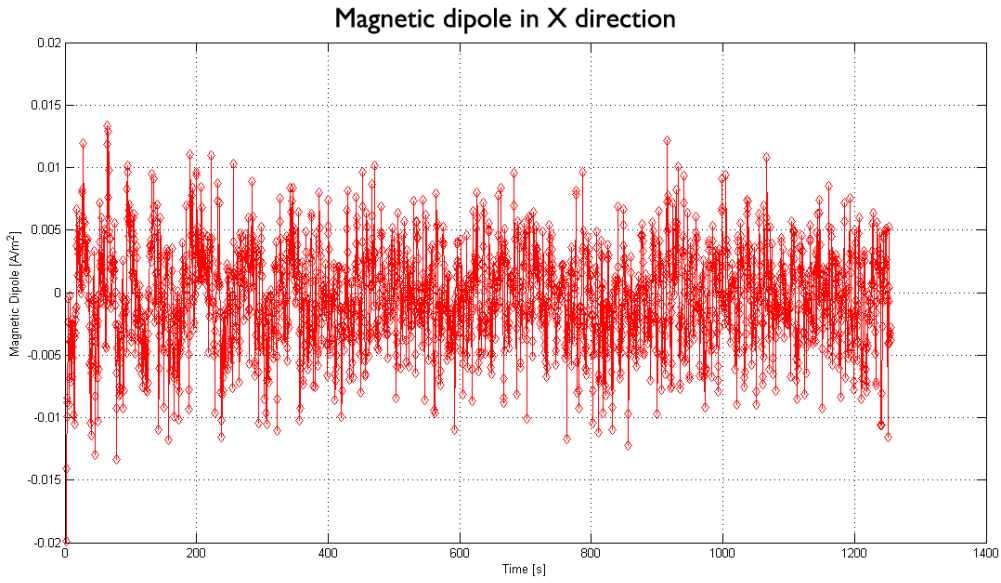


Figure D-7: The calculated magnetic dipole in the X direction

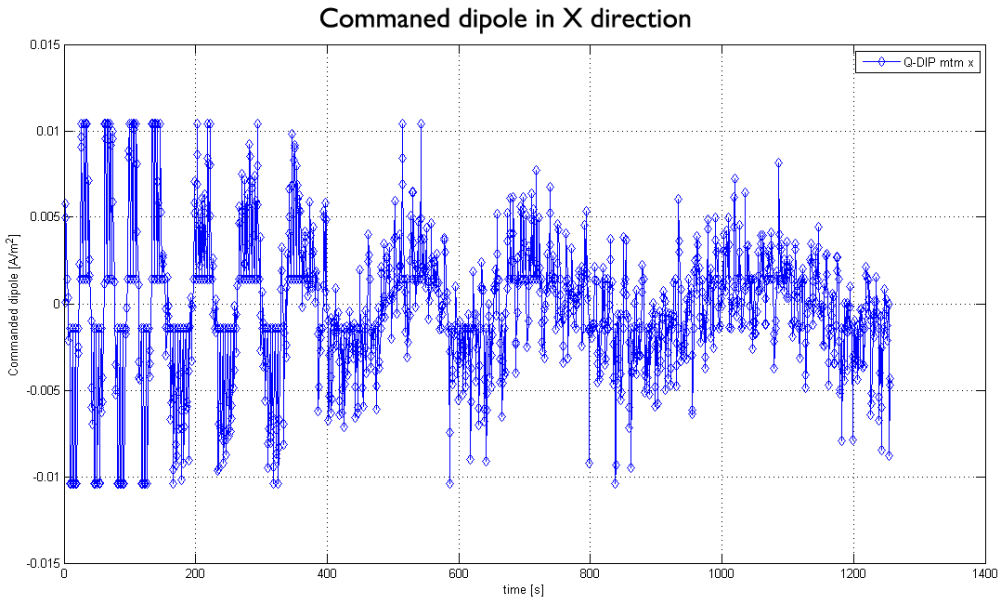


Figure D-8: The calculated commanded dipole in the X direction

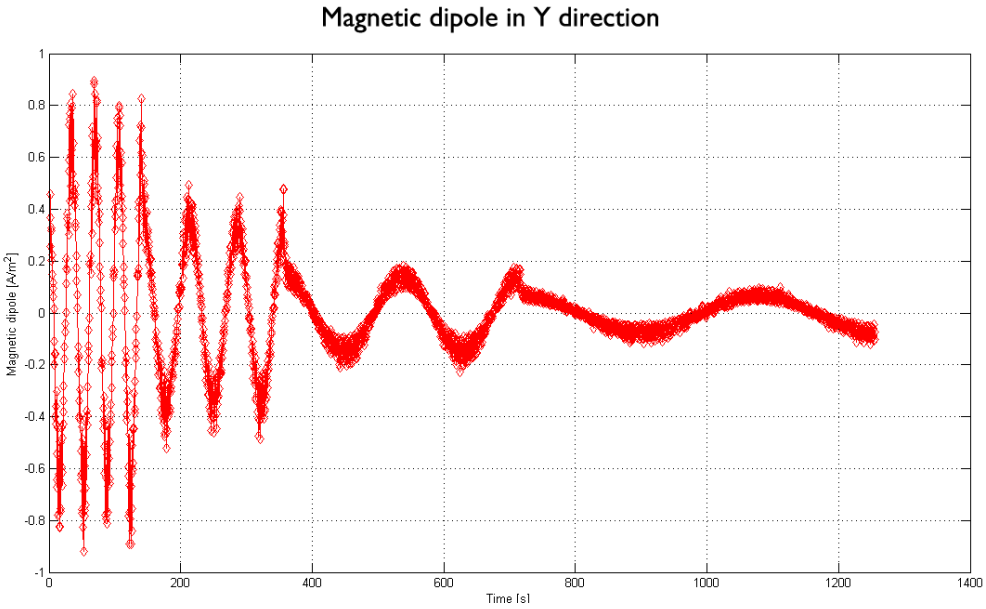


Figure D-9: The calculated magnetic dipole in the Y direction

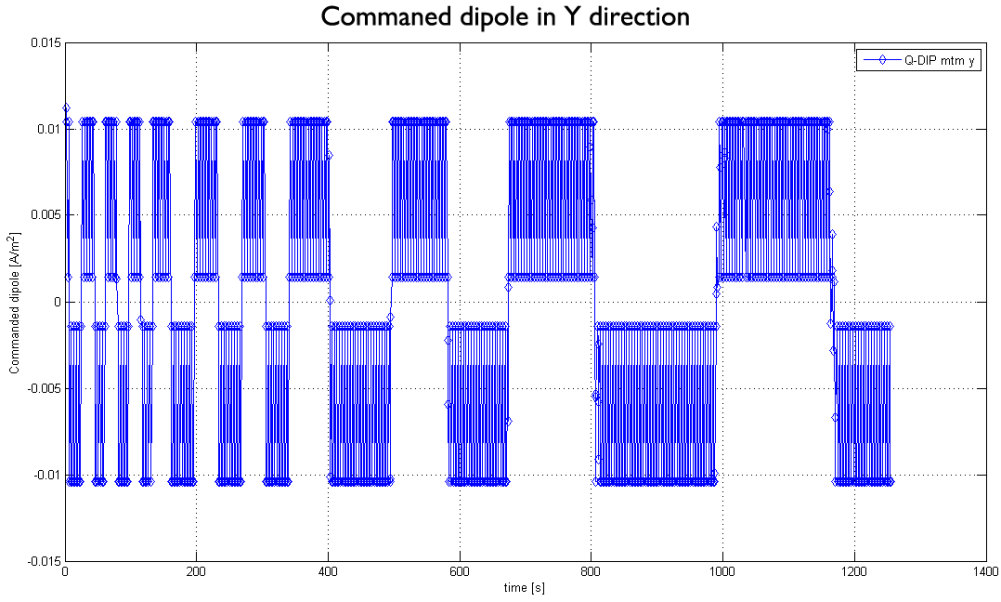


Figure D-10: The calculated commanded dipole in the Y direction

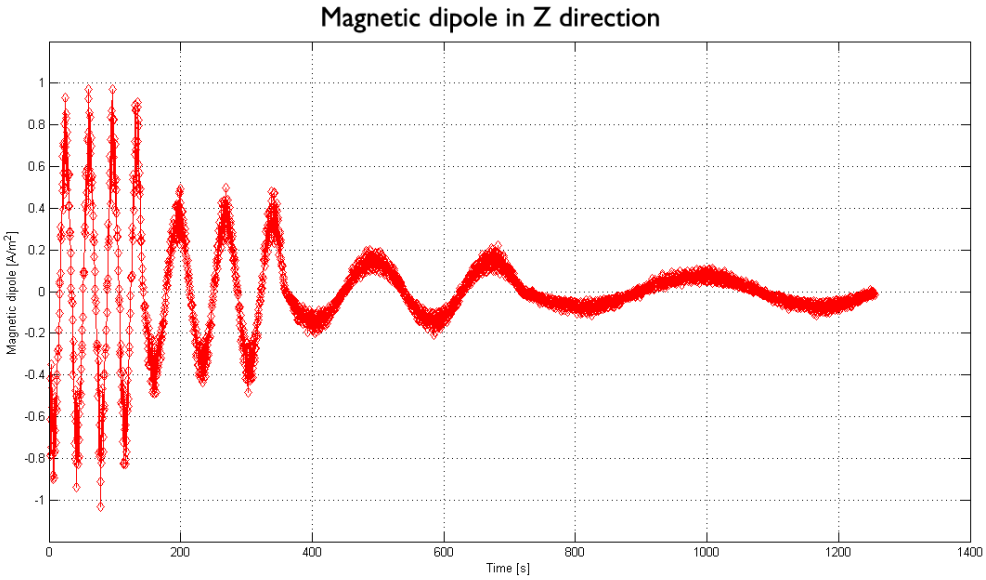


Figure D-11: The calculated magnetic dipole in the Z direction

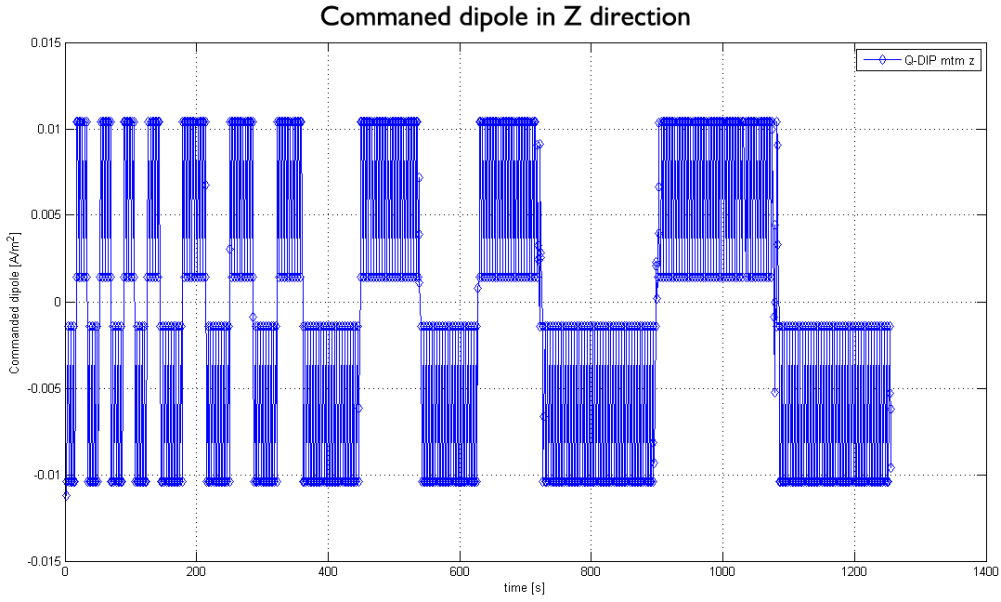


Figure D-12: The calculated commanded dipole in the Z direction

Appendix E

Sun sensor verification plots

In this appendix some additional figures are depicted, which were not shown in Subsection 8-2-2. These figures show the response of the four quadrants from each sun sensor under a varying illumination angle.

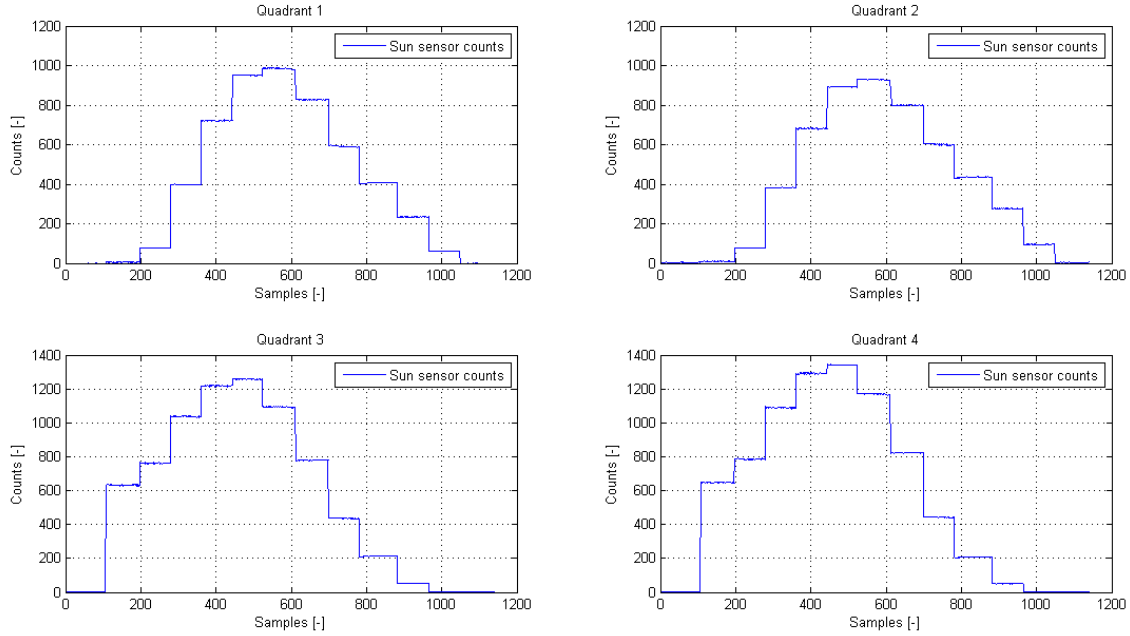


Figure E-1: The response of the sun sensor in positive X direction under 12 different illumination angles.

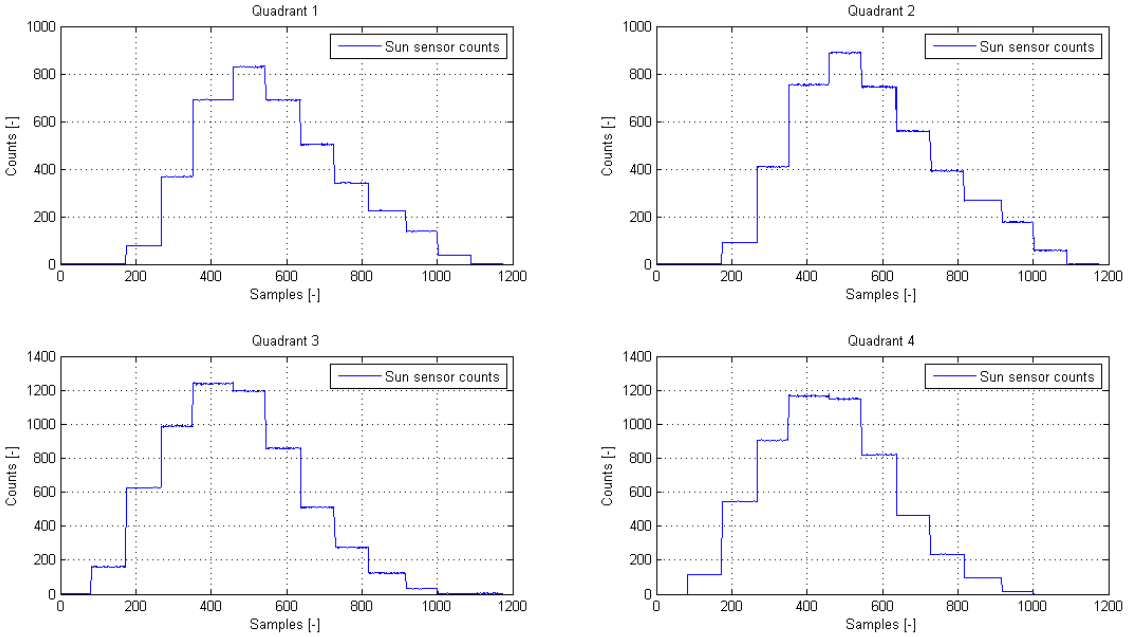


Figure E-2: The response of the sun sensor in negative X direction under 12 different illumination angles.

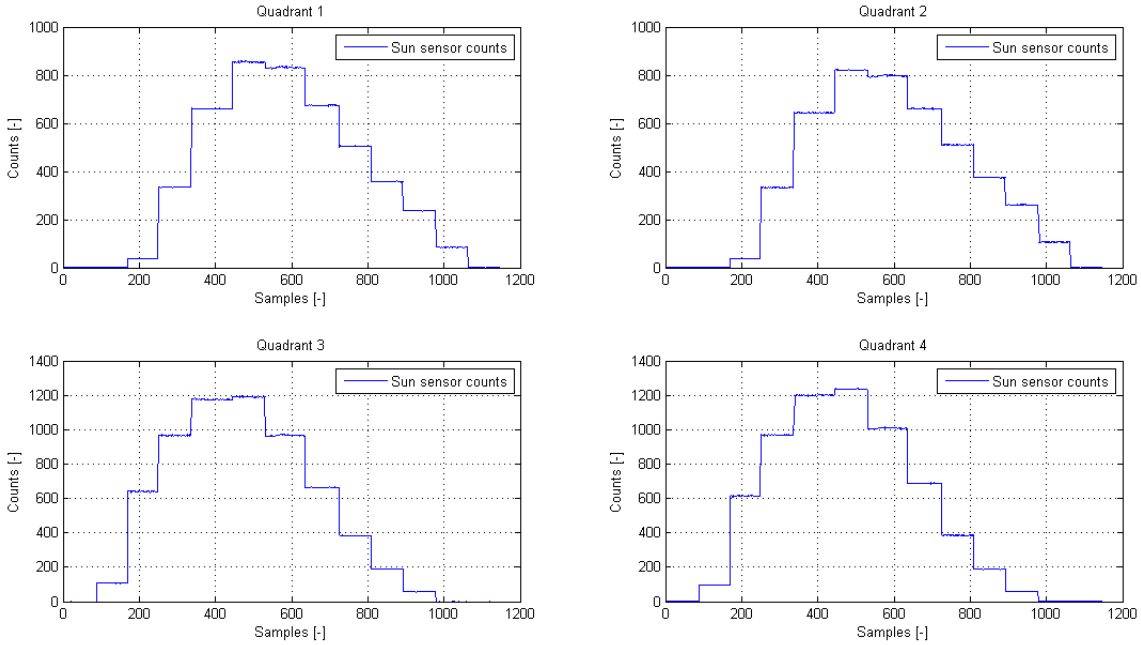


Figure E-3: The response of the sun sensor in positive Y direction under 12 different illumination angles.

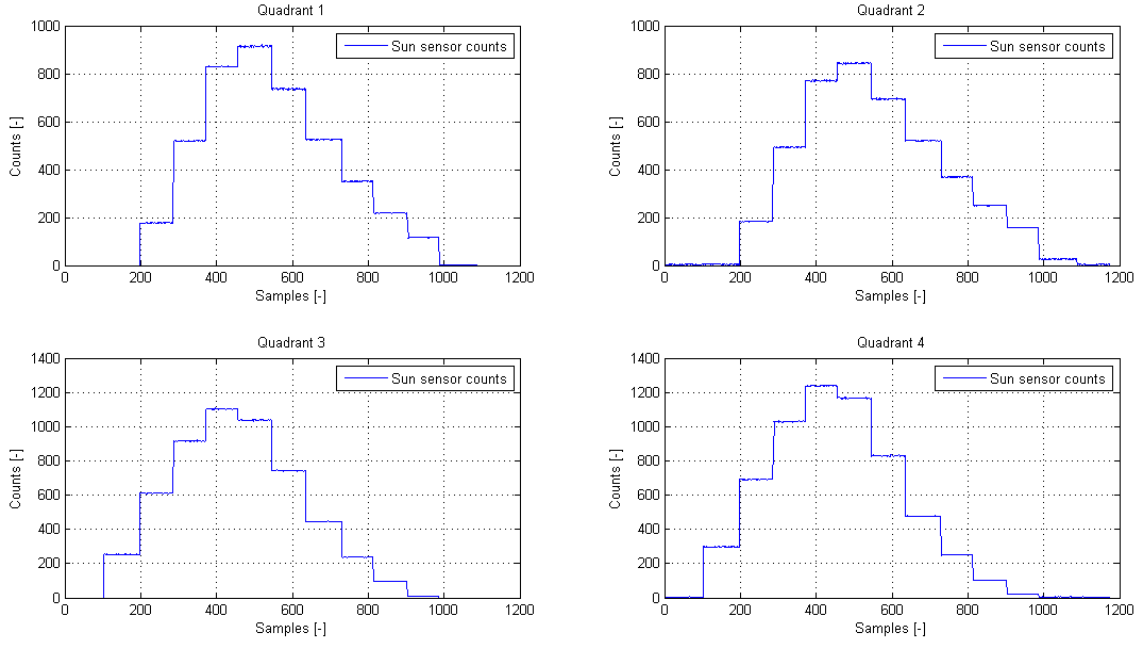


Figure E-4: The response of the sun sensor in negative Y direction under 12 different illumination angles.

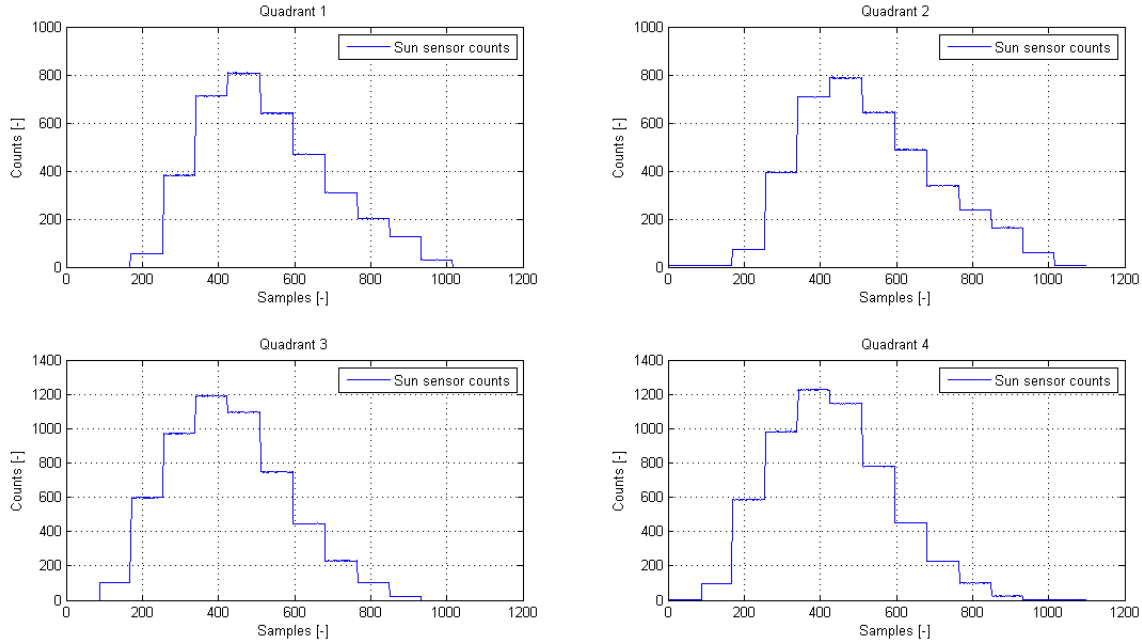


Figure E-5: The response of the sun sensor in positive Z direction under 12 different illumination angles.

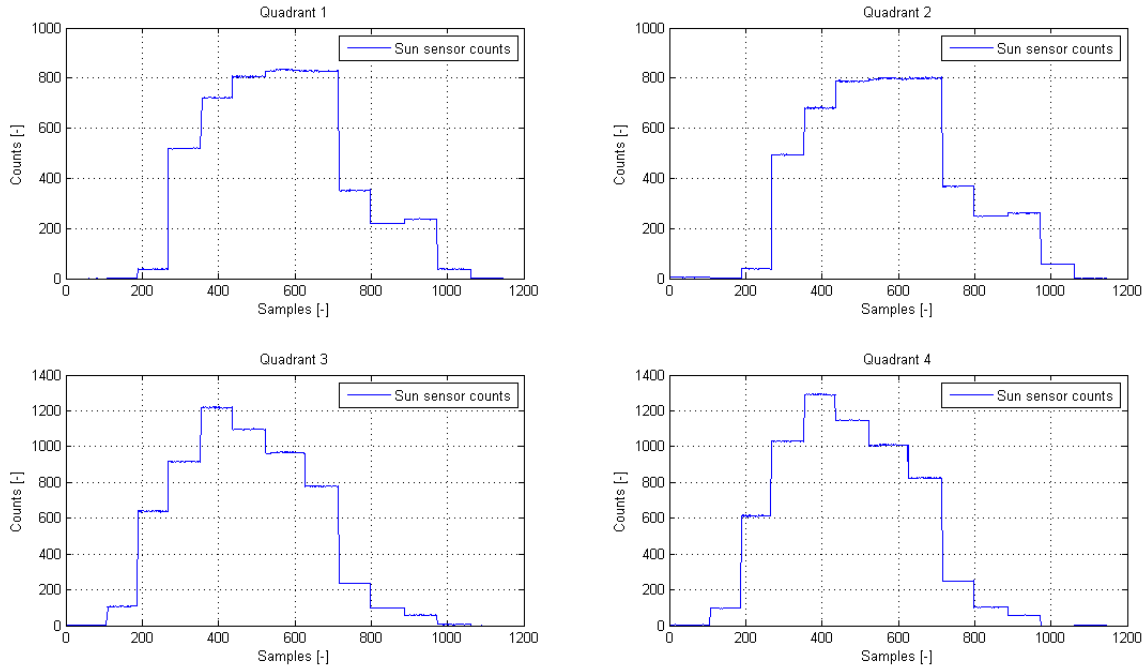


Figure E-6: The response of the sun sensor in negative Z direction under 12 different illumination angles.