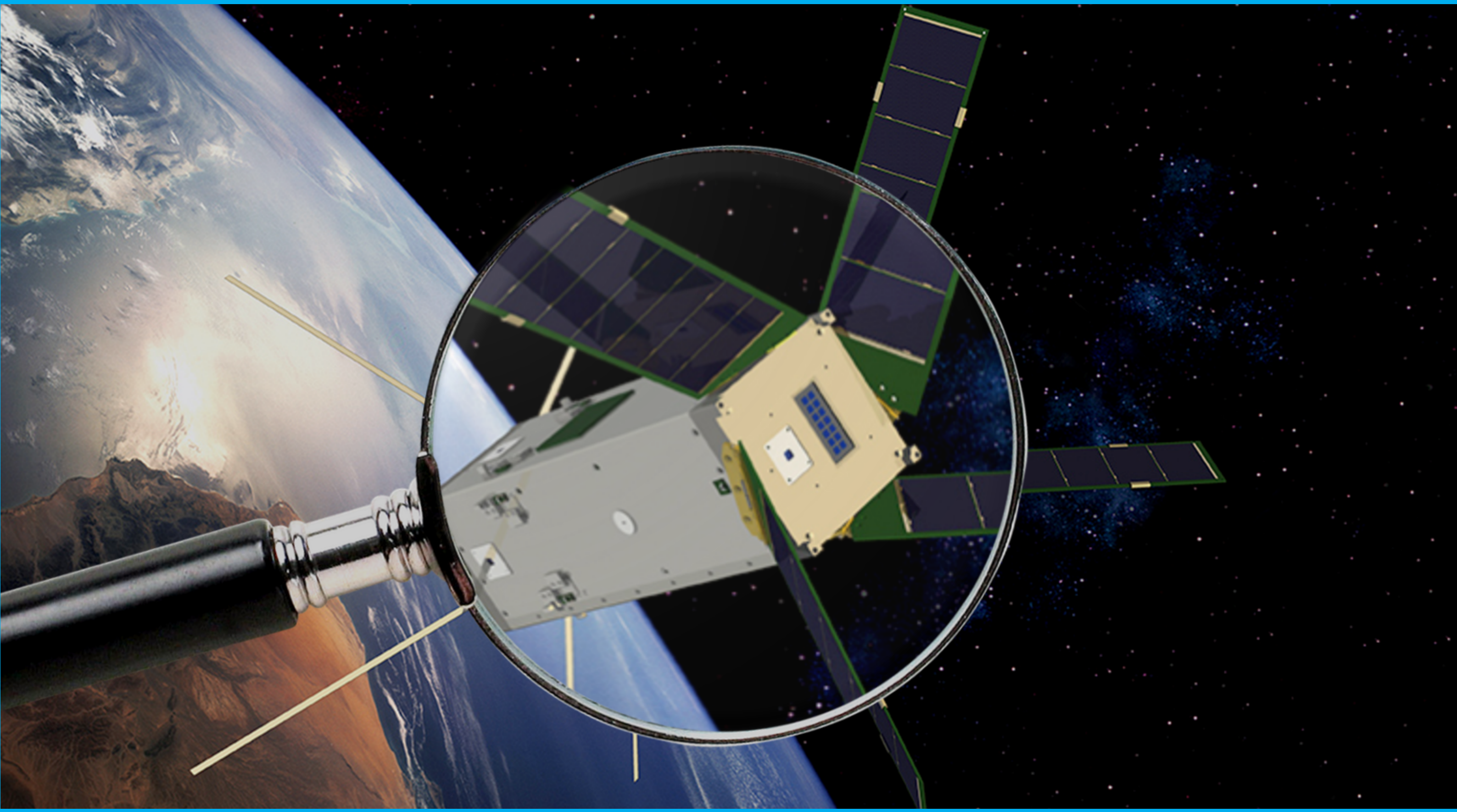
Delfi-n3Xt forensics

A hybrid methodology

S. van Kuijk

Master of Science Thesis, 2016



TU Delft

Delfi-n3Xt forensics

A hybrid methodology

by

S. van Kuijk

to obtain the degree of Master of Science
at the Delft University of Technology.
to be defended publicly on Friday December 16, 2016 at 02:00 PM.

Student number:	1507273	
Project duration:	October 1, 2015 – November 25, 2016	
Thesis supervisors:	ir. P.P Sundaramoorthy,	TU Delft
	T. Watts,	TU Delft

TU Delft

Abstract

In this master thesis the reader will find the application of a hybrid forensic investigation on a CubeSat project. This CubeSat, known as Delfi-n3Xt, was developed by the Delft University of Technology and it was launched in November 2013. The satellite was successfully operated for three months but contact was lost after execution of an experiment. The reader will find that the investigation is greatly affected by a lack of facts. There is no accessible physical evidence, the CubeSat wasn't formally verified, and internal documents are lacking in many aspects. Therefore, the research methodology itself is the topic of interest and Delfi-n3Xt is used as a case-study.

Due to the lack of facts, a combination of a forensic approach and a more general empirical approach is used. In this thesis, the satellite operations are reviewed extensively by inspection of orbital elements, by an extensive telemetry analysis, by a reconstruction of the mission and by a detailed analysis of the last moments of contact. Beyond the operations, the design of the ISIS Transceiver (ITRX), including the Linear Transponder (LT), is reviewed to identify hypotheses. During this thesis over 30 hypotheses were identified. Some hypotheses are directly related to the loss of contact and some are not. Since many hypotheses were identified, a selection was made for verification. This selection procedure was based on a methodology similar to a risk assessment, where the hypotheses with highest risk are selected.

Eventually five verification campaigns were performed. One campaign was based on analysis and the other four were based on tests using spare hardware. Unfortunately, no root cause was found during this thesis, but two hypotheses were rejected, one hypothesis was verified, and one hypothesis was considered plausible and highly likely. Based on the achieved results it is concluded that the used methodology is effective for cases like Delfi-n3Xt, were evidence is lacking. It is acknowledged however, that full confirmation of the root cause of failure can't, and will never be achieved, because the only way to determine the root cause is by inspection of the satellite itself. Many lessons were learned however, regarding forensics, regarding Delfi-n3Xt and regarding CubeSat projects in general.

Preface

During this thesis I had lot's of fun, but also many frustrations. I have seen nearly all aspects of a CubeSat project; from mission goals to operations, from telemetry processing to stakeholder politics. This thesis was therefore an excellent learning experience which I will definitely use in my future career. Many people supported me through this thesis and I would like to thank them all.

Firstly I would like to thank my supervisors P.P. Sundaramoorthy and T.G. Watts for their supervision and advice. I also would like to thank the project manager of Delfi-n3Xt, J. Bouwmeester, for his inputs. Other people that supported me are: C.J.M. Verhoeven, S. Speretta, W.J. Weggelaar, T. Perez Soriano, N. von Storch, E.P. Smit, P. Beckers and all students that worked on the 8th floor of the faculty of Aerospace Engineering. Beyond that I would like to thank the thesis assessment committee for their assessment.

Not surprisingly, I would also like to thank my parents, my girlfriend, the rest of my family and friends for their love, passion, support, dedication and patience.

S. van Kuijk
Delft, November 2016

List of Acronyms

μC	Micro controller
ADCS	Attitude Determination and Control System
AHP	Analytic Hierarchy Process
AI	Action Item
BOP	Bottom Panel
BMS	Battery Management System
BPSK	Binary Phase-Shift Keying
CDHS	Command and Data Handling Subsystem
CDR	Critical Design Review
CGG	Cold Gas Generator
COG	Center of Gravity
COMMS	Communications Subsystem
CW	Continuous Wave
DAB	Deployment and Antenna Board
DAQ	Data Acquisition System
DGS	Delft Ground Station
DSSB	Delfi Standard System Bus
ECSS	European Cooperation on Space Standardization
EOL	End-Of-Life
EPS	Electrical Power Subsystem
FRR	Flight Readiness Review
GS	Ground Segment
GSE	Ground Support Equipment
HDRS	Hold Down and Release System
HK	House Keeping
INIT	Intitalization
ITC	ISIS transceiver Transmitter Controller
ITRX	ISIS Transceiver
JSpOC	Joint Space Operations Center
LEOP	Launch and Early Orbit Phase
LT	Linear Transponder
LOS	Launch & Orbit Segment
MAB	Modular Antenna Box
MechS	Mechanical Subsystem
MPPT	Maximum Power Point Tracker

OBC	On-Board Computer
PCB	Printed Circuit Board
PDR	Preliminary Design Review
PTRX	Primary Transceiver
RF	Radio Frequency
ROD	Review-of-Design
SAA	South Atlantic Anomaly
SAT	Satellite
SBF	Satellite Body Frame
SDM	Silicon Solar Cell Degradation Measurement
SE	Systems Engineering
STS	Structural Subsystem
STX	S-band transmitter
$\mathrm{T}^3\mu PS$	Micro Propulsion System
TCS	Thermal Control Subsystem
TLE	Two-Line Element
TRL	Technology Readiness Level
UHF	Ultra High Frequency
UI	User Interface
UTC	Coordinated Universal Time
VHF	Very High Frequency

List of Figures

List of Tables

Contents

1 Introduction

The Delft University of Technology initiated the Delfi program in 2004. The program provides hands-on education and training for students in CubeSat development and research. The main objectives of this program are to educate students, demonstrate new technology and to develop new technology. Up to now two CubeSats have been launched; the Delfi-C3 in April 2008 and Delfi-n3Xt in November 2013. Delfi-C3 is still operational and considered as a full mission success. Delfi-n3Xt also achieved mission success[9], but unfortunately, contact was lost after three months of operation.

It is well known that university CubeSats have a high failure rate and that the failures are often not acknoledged nor explained[24]. A plausible explanation for this silence is simple: bad publicity. Swartwout[25] suggests that many of these failures could have been prevented by thorough verification through system level testing. This however increases unwanted costs and thus certain risks are accepted. This raises the question however, whether the risks were critically assessed before launch or not.

Unfortunately, Delfi-n3Xt is currently one of those CubeSats that dissappeared in silence. To this day, no official statements were provided by the Delft University of Technology, except for the fact that the project team currently experiences a (temporary) loss of contact and that the situation is under investigation[9][20]. Almost three years later, it is still unknown why contact with Delfi-n3Xt was lost. Not knowing the root cause might form a barrier to publicly acknowledge the situation. Therefore, the goal of this master thesis is to determine the source of failure using a systematic approach.

To determine the loss of contact a combination of a forensic approach and a more general empirical approach is used. This combination is chosen since fact-finding is very complex as there is a lack of physical evidence. In addition, a literature review performed by the author, regarding the verification status of Delfi-n3Xt before launch[32], showed that the satellite wasn't formally verified and that the internal documentation was lacking in many aspects. Therefore, many facts are missing which greatly increase the difficulty of the application of a forensic methodology on the Delfi-n3Xt project. The research methodology itself is thus a topic of interest. For this project a hybrid solution is used where hypotheses are generated which are based on various observations.

The research question is defined below:

How should one perform forensics on a launched and not-responding satellite?

To answer this question sub questions are generated:

1. *How can one identify failure scenario's?*
   (a) *Can one identify failure scenario's by reviewing the design?*
   (b) *Can one identify failure scenario's by reviewing the implementation of the design?*
   (c) *Can one identify failure scenario's by reviewing the performed operations?*
   (d) *Can one identify failure scenario's by reviewing the external factors?*
   (e) *Can one identify failure scenario's by reviewing the interviewing stakeholders?*
2. *Can one verify the failure scenario's using hardware spare parts/left overs?*

These research questions lead to the research methodology described in Chapter 2. This chapter is followed by Chapter 3 which provides an overview to Delfi-n3Xt and it provides an overview of the TRL for each subsystem. The analysis starts in Chapter 4 which reviews the performed operations. This chapter includes an orbit analysis, a telemetry anomaly analysis, a reconstruction of the mission time-line and briefly describes some observations on the audio-recordings. Chapter 5 describes a review of the design to support the previously made observations and conclusions and Chapter 6 provides an overview and selection of the defined hypotheses. This chapter is followed by Chapter 7. This chapter states the verification campaigns of the selected hypotheses and finally, the conclusions and recommendations are described in Chapters 8 and 9.

2
Research methodology

This chapter describes the research methodology that is used to reach the goal and to answer the research questions which are stated in Chapter 1.

As the goal of this thesis is to determine why Delfi-n3Xt was lost, a forensic methodology, displayed in Figure 2.1 seems very promising. Unfortunately, the application of a forensic investigation methodology on a launched spacecraft which doesn't respond is much harder than an Earth-based event. Fact-finding, the foundation of a forensic methodology, has an increased difficulty since there is a lack of evidence as the spacecraft is inaccessible. For this project the following sources of information are available:

- Telemetry[18].
- Internal design and verification documents on the Delfi disk[26].
- Flight software on the Delfi disk[26].
- Hardware logs on the Delfi disk[26].
- Delfi-n3Xt operational logs which include transmitted commands (Delfi-mailbox)[29].
- Audio recordings of Delft Ground Station (DGS)-passes[28].
- Experience report regarding operations by R.Schoemaker[22] (part of thesis).
- Preliminary mission results analysis by J.Bouwmeester[9].
- Delfi-n3Xt orbital data, TLEs[23].
- Stakeholder(s) opinions/experience.
- Spare hardware (flight models and engineering models).

As shown in Figure 2.1 it is important to categorise the sources to identify facts from opinion or questionable/outdated data. Unfortunately it was found during the literature review[32] and from personal communication with with project team members, that the documentation regarding Delfi-n3Xt, located on the Delfi disk[26], is incomplete, outdated, inconsistent or conflicting. It is also known that the satellite wasn't formally verified and reviewed before launch and therefore it is unclear whether the telemetry provides the correct data. The combination of lacking documentation and an unverified satellite has major consequences regarding this investigation. Design interpretation is a challenge and it is unclear whether Delfi-n3Xt meets the design and is free of defects. Therefore, identification of facts possibly leading to the source of loss of contact becomes even more challenging. Inspection of the operational logs also resulted in the conclusion that a careful approach must be taken when interpreting this information. The operational logs are incomplete, prone to human error and questionable, since there was no automatic recording system of the transmitted commands.

Currently the only source providing facts are the audio recordings of DGS-passes located on the Delfi server[28]. All other sources should be used carefully. Throughout this report it is mentioned when certain information is considered as a fact, or not.

Figure 2.1 provides an overview of a forensic research methodology. A forensic investigation starts when an event has occurred which should be avoided in the future to prevent catastrophic consequences. The event is observed and facts are determined by examining what happened and how it happened. These facts are used to reconstruct the event and plausible scenario's are identified. The

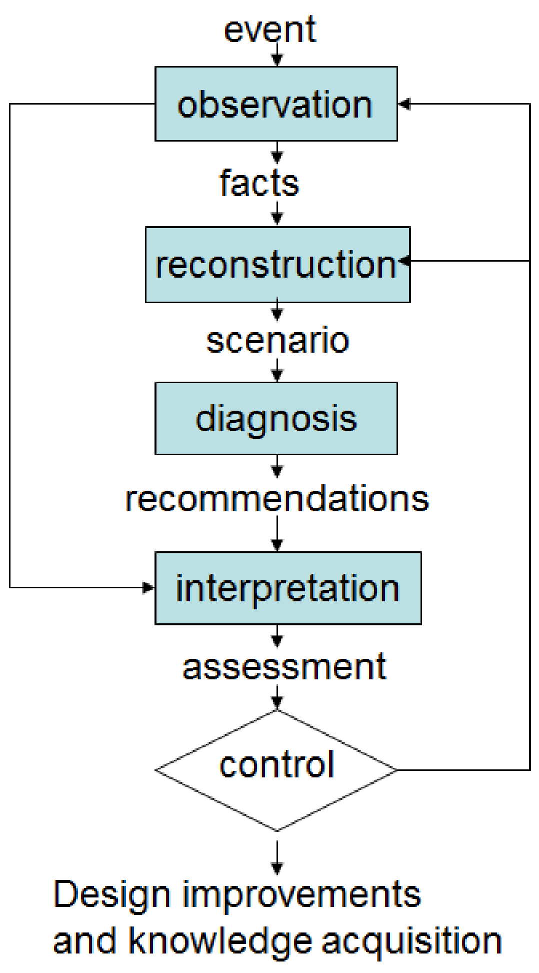


Figure 2.1: The basic cycle of forensic scientific research[7]

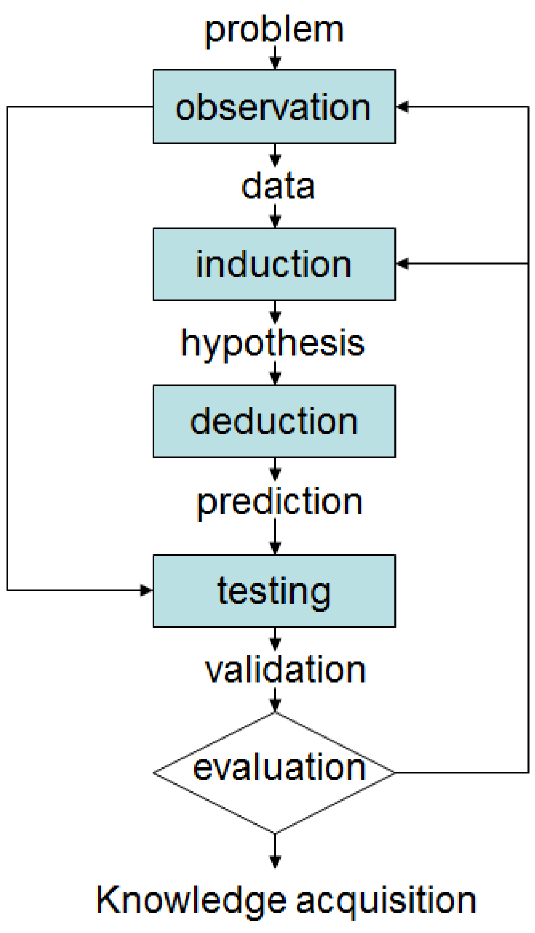


Figure 2.2: The basic cycle of empirical scientific research[7]

scenario's are thoroughly diagnosed and recommendations are made to avoid the scenario's. The recommendations are interpreted and solutions are found using an iterative process which ends until all the scenario's are avoided.

This process strictly relies on fact finding while it is already known that the facts are lacking for this investigation. A forensic investigation can be seen as a subset of an empirical investigation, shown in Figure 2.2. In an empirical research methodology a problem is observed regarding its causes. This is followed by induction, where hypotheses are formed explaining the observed phenomenon. Experiments are deduced to test the formulated hypotheses and test is performed to verify or reject the hypothesis. Again this is an iterative process that ends until a reasonable theory is found which is based upon a verified hypothesis.

Since the there are many questionable informative sources, a hybrid research methodology shall be used which consists of a combination of an empirical and forensic methodology. Facts and questionable data are used to reconstruct a mission time-line leading to hypotheses/failure scenario's. It is important to use a different data set to verify a hypothesis as otherwise the verification method is fallacious[1], due to circular reasoning. Therefore, verification shall be performed using tests on spare hardware. Verification shall be performed carefully as, depending on the performed test, the hardware should be identical to the flight hardware of Delfi-n3Xt.

To answer the research question, including sub questions, a solution strategy will be used which is stated in Figure 2.3. The figure clearly indicates that the proposed research uses an iterative process that ends when a failure scenario/hypothesis has been verified. The research starts with identifying various plausible failure scenario's. The flow chart indicates that there are five Action Items (AIs) that shall be investigated to identify failure scenario's. It is expected that *AI 2.1.2, AI 2.1.3* and *AI 2.1.5* will identify most scenario's. The other AIs are mentioned for completeness. These AIs are expected to answer research questions *2.X*.

Upon acquiring several failure scenario's/hypotheses, the most promising one's are selected in *AI 3*. The selection process consists of a methodology similar to a risk assessment where the impact/importance and the likelihood are examined.

Verification of the failure scenario's/hypotheses will be performed in *AI 4*. Verification shall generally be performed using testing on hardware that is available at the aerospace faculty of Delft. It might be the case that the test rejects the hypothesis. In that case a new failure scenario is selected or identified (it might occur that during testing new scenario's/hypothesises are identified).

Due to the iterative nature of this research it might take a long time to verify the failure scenario. It must be noted however, that the duration of this research project shall be constraint to 7 months (unless when the verification process of last selected hypothesis is not completely verified/rejected). It might thus be the case that the root cause is not found at the end of the research project. Valuable knowledge is gained however, as some failure scenarios are excluded from the list of possible scenario's.

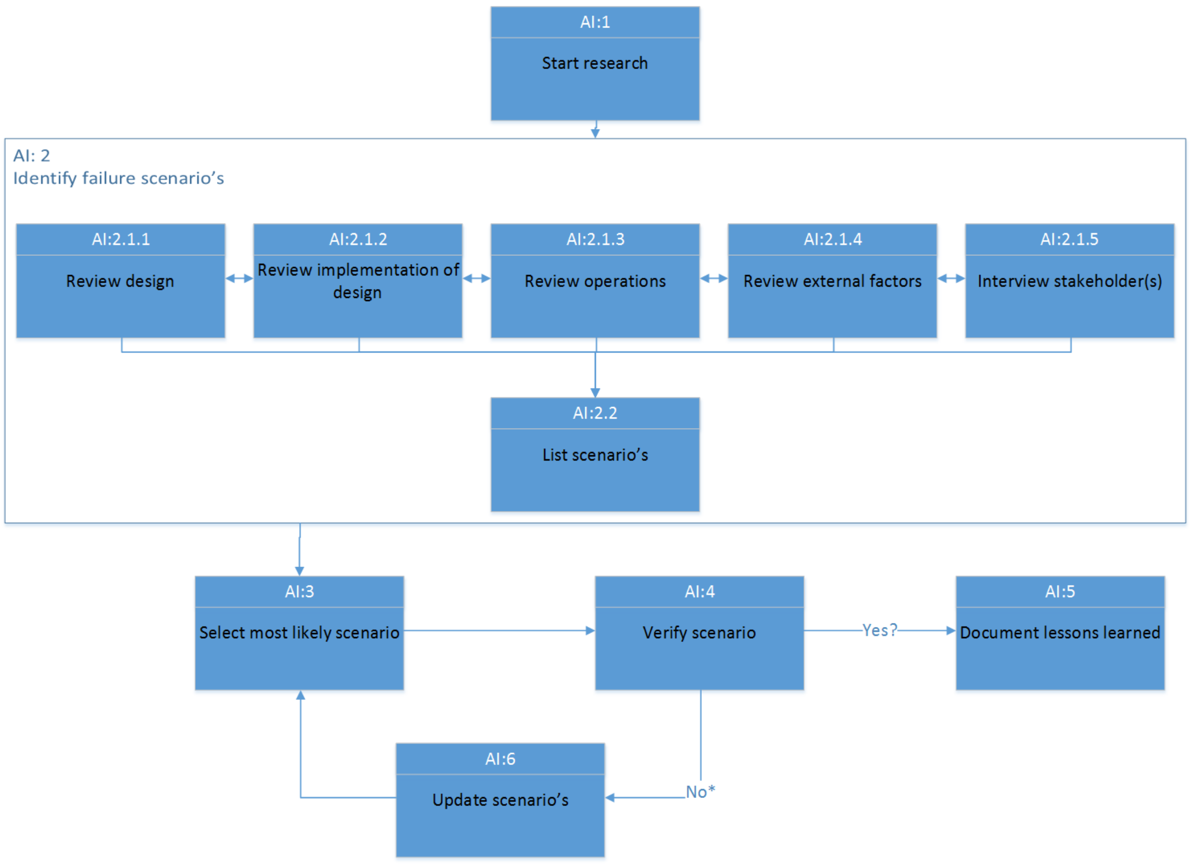


Figure 2.3: Flow chart of proposed research methodology

Comparing Figure 2.3 with 2.2 and 2.1 one finds that the used methodology mostly represents the empirical research methodology. It must be noted that during the review of the performed operations hypothesis identification is based on reconstruction, prediction and data analysis.

The next chapter, Chapter 3 provides an overview of Delfi-n3Xt to introduce the various (sub)-systems. It is recommended to read this chapter when one has no- or limited knowledge regarding the Delfi-n3Xt mission.

3 Overview Delfi-n3Xt

This chapter gives an overview of Delfi-n3Xt. Section 3.1 provides the mission definition, Section 3.2 describes the top level overview and in Section 3.3 an overview is provided of the CubeSat design. These sections are a reduced and combined version of [4] and [3, p. 9-17]. The last section, Section 3.4 briefly describes the opinion of the author regarding the design of Delfi-n3Xt and the TRLs are defined for each subsystem.

3.1. Mission

Delfi-n3Xt is the second satellite project within the Delfi program. In this section the mission goals of this project are described. In Section 3.1.1 the objectives are provided and in Section 3.1.2 the mission statement is defined.

3.1.1. Mission objectives

The Delfi program is divided in three objectives:

- *Education:* The Delfi program shall provide students optimal preparation for careers in space industry.
- *Technology demonstration:* Delfi satellites will perform technology demonstration of micro-technologies for space applications, emerging from various developments within the Dutch space sector. Stand alone experiments are considered as 'payloads' and are explicitly mentioned as mission objectives.
- *Nano-satellite bus development objective:* The Delfi program will advance the nano-satellite platform gradually with the aim to make very small satellites more capable for advanced technology demonstration, scientific or commercial purposes.

3.1.2. Mission statement

"Delfi-n3Xt shall be a reliable triple-unit CubeSat of TU Delft which implements substantial advances in 1 subsystem with respect to Delfi-C3 and allows technology demonstration of 2 payloads from external partners from 2012 onwards."

3.2. Top level overview

Figure 3.1 shows the top level configuration items of Delfi-n3Xt. The most important element is the satellite itself, but can't perform its mission without the other items. The three items are: the Launch & Orbit Segment (LOS), the Ground Segment (GS) and the Satellite (SAT). The LOS consists of the launcher and the payload adapter, which includes interfaces between the satellite and the launcher. The GS consist of the ground station network and the ground support equipment and the SAT can be split up in payload and a spacecraft bus.

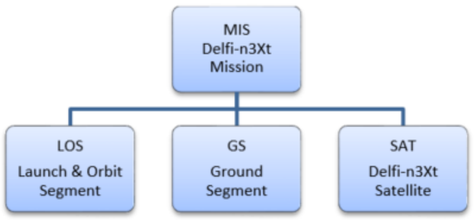


Figure 3.1: Top level configuration items[4, p. 8]

Figure 3.2 provides an overview of the interfaces between the ground segment and the space segment. From the figure one can see that the space segment provides telemetry tot the ground station network, and that only the DGS is able to command the satellite. On ground, all telemetry is transferred tot the telemetry server which collects and processes all telemetry.

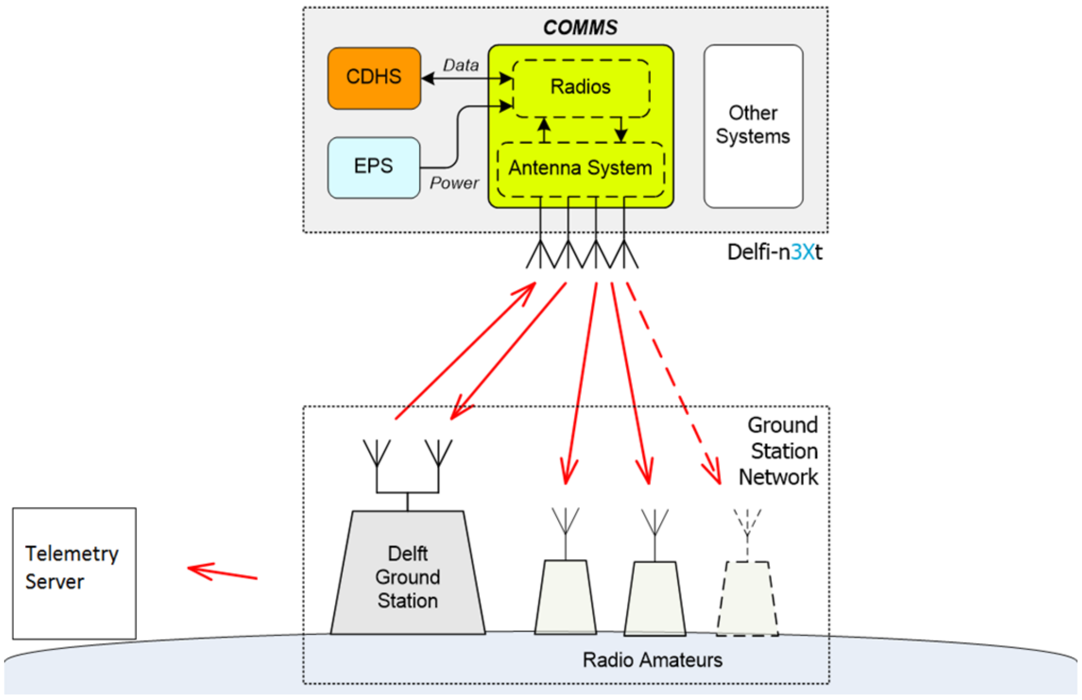


Figure 3.2: System overview[30]

3.3. Design overview

Figure 3.3 shows the satellite breakdown of Delfi-n3Xt. Figure 3.4 provides a render of the satellite design, excluding side panels.

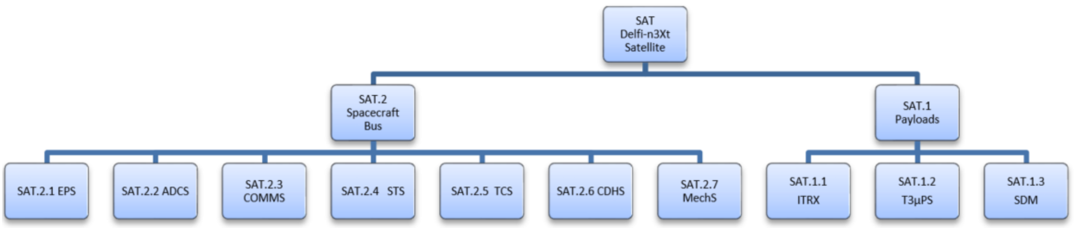


Figure 3.3: Delfi-n3Xt breakdown [4, p. 11]

Nomenclature:

- Electrical Power Subsystem (EPS)
- Attitude Determination and Control System (ADCS)
- Micro Propulsion System ($\mathrm{T}^3\mu PS$)
- On-Board Computer (OBC)
- Primary Transceiver (PTRX)
- S-band transmitter (STX)
- ISIS Transceiver (ITRX)
- Deployment and Antenna Board (DAB)
- Bottom Panel (BOP)
- Command and Data Handling Subsystem (CDHS)
- Communications Subsystem (COMMS)
- Mechanical Subsystem (MechS)
- Structural Subsystem (STS)
- Silicon Solar Cell Degradation Measurement (SDM)
- Thermal Control Subsystem (TCS)

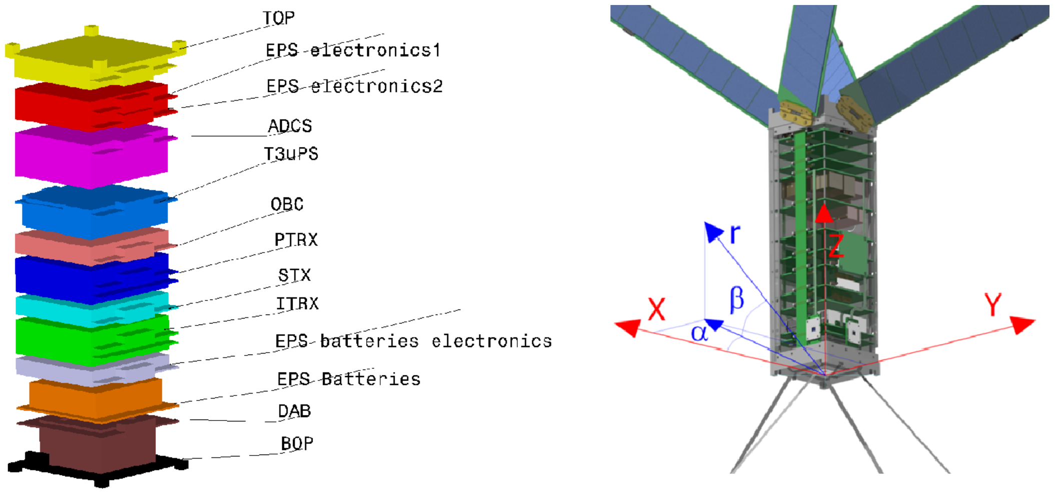


Figure 3.4: *Left* Satellite bus, *right* Axis definition Delfi-n3Xt[4, p. 12]

3.3.1. Attitude Determination and Control System (ADCS)

Delfi-n3Xt includes an 3-axis active ADCS. Figure 3.5 provides the configuration items of this subsystem.

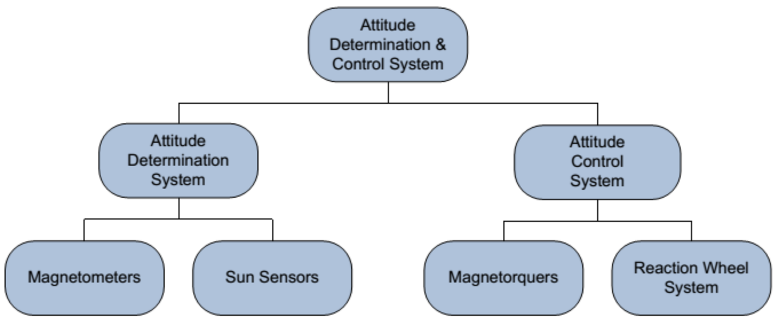


Figure 3.5: Configuration items ADCS Delfi-n3Xt[3, p. 11]

As one can see in Figure 3.5, the ADCS actuators are three orthogonal magnetorquers and three orthogonal reaction wheels. The attitude is determined using six sun sensors and one three-axis magnetometer. Figure 3.6 shows the hardware components of the ADCS system.

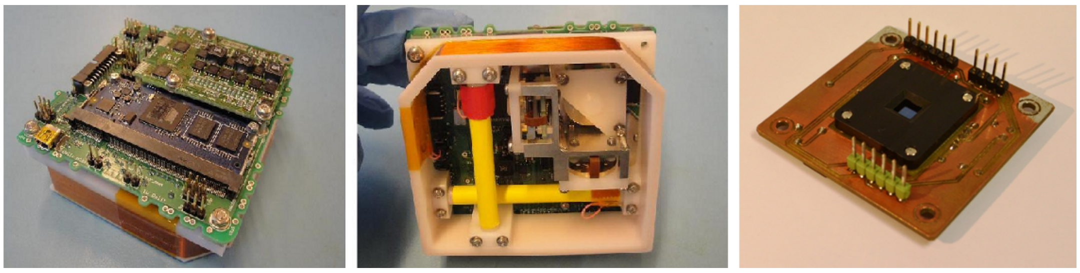


Figure 3.6: *Left* microcontrollers ADCS, *middle* magnetorquers and reaction wheels [3, p. 12], *right* sun sensor [4, p. 17]

The ADCS includes a fast micro controller (ARM 9 based) for various modes and a micro controller (XMega) for backup. The XMega controller is also used for detumbling. Table 3.1 provides the various ADCS modes:

Table 3.1: Overview of ADCS modes of Delfi-n3Xt [4, p. 19]

Mode	**Determination algorithm**	**Sensors**	**Control algorithm**	**Actuators**
Detumble	B-dot	MM	B-dot	MTQ
Coarse sun pointing	Leas-Squares	SS	PD	MTQ
Fine sun pointing	Additive Extended Kalman	SS & MM	PD	MTQ & RW
Ground station Tracking	Additive Extended Kalman	SS & MM	PD	MTQ & RW
Thruster pointing	Additive Extended Kalman	SS & MM	PD	MTQ & RW

3.3.2. Communications Subsystem (COMMS)

The communication system of Delfi-n3Xt consits of two tranceivers: ITRX and PTRX. Beyond that the STX is carried that can achieve high data rates. Figure 3.7 shows the configuration items of the communication subsystem and Table 3.2 provides the corresponding characteristics.

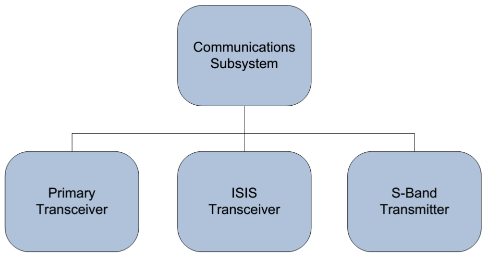


Figure 3.7: Configuration items COMMS [3, p. 12]

Table 3.2: Communication characteristics Delfi-n3Xt[4, p. 19]

Parameter	**PTRX, ITRX**	**PTRX, ITRX**	**STX**
Frequency [MHz]	*Transmitter* ITRX:145.870 PTRX:145.870	*Receiver* 435	*Transmitter* 2405
Data Rate [bit/s]	2400	1200	2.4-500 x 10^{-3}
Transmitter Output power [W]	0.16	-	0.13
Modulation	BPSK	AFSK	MFSK

The PTRX is used as the primary transceiver and is identical to the ITRX. The ITRX however, includes a linear transponder and is considered as payload. The used communication protocol used is AX.25, as it is known by radio amateurs. The antenna system is part of the DAB and and comprises of four shared Ultra High Frequency (UHF)/Very High Frequency (VHF) antennae which are placed in a canted turnstile configuration, which can be seen in Figure 3.4. A phasing circuit induces a omnidirectional gain pattern and an isolation circuit is used to isolate receiving and transmitting signals.

3.3.3. Structural Subsystem (STS) and Thermal Control Subsystem (TCS)

The structure of Delfi-n3Xt consists of:

- a bottom panel, including openings for the antenna boxes, a sun sensor and kill switches.
- a top panel, including openings for the SDM system and a sun sensor.
- an outer skin consisting of who L-profiles with openings the sun sensors, the $\mathrm{T}^3\mu PS$ thruster and the patch antenna of the STX.
- a stack consisting of four rods, fasteners and distance tubes to hold the Printed Circuit Boards (PCBs) in place.

The TCS is passive. Radiators are placed on small components with relatively high heat dissipation, such as the power amplifiers of the radios. On the structure, thermal tapes will be applied which can shift the thermal range within the satellite. Figure 3.8 shows images of the used structure. One can clearly see the thermal tape inside the structure.

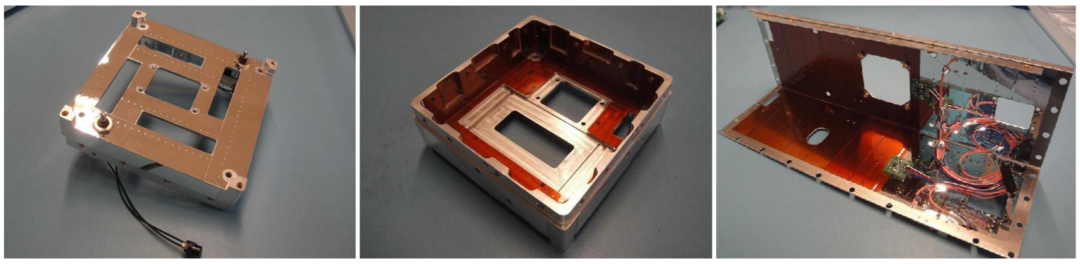


Figure 3.8: *Left:* Bottom panel, *Middle:* Top panel, *Right:* Outer skin[3, p. 9-10]

3.3.4. Mechanical Subsystem (MechS)

The MechS consists of two deployment mechanisms: A mechanism to deploy the UHF/VHF antennae, called the MAB, and a mechanism to deploy the solar panels, known as the HDRS. Both systems are controlled by the DAB. Figure 3.9 provides both mechanical systems.

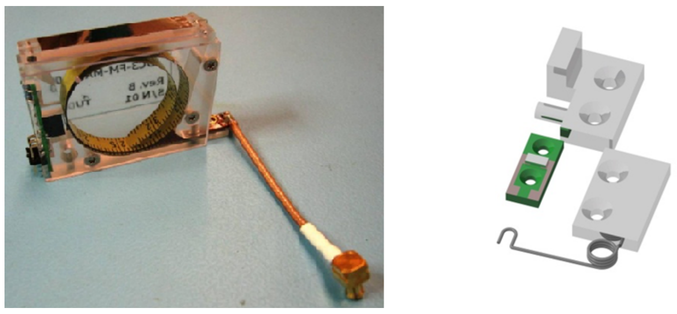


Figure 3.9: *Left:* MAB, *Right:* HDRS[4, p. 24]

3.3.5. Electrical Power Subsystem (EPS)

The EPS consists of four configuration items, shown in Figure 3.10. The purpose of the EPS is to provide power to all the subsystems while in orbit. Delfi-n3Xt is designed to remain powered during eclipse. Therefore energy storage is required as well. The EPS is developed by the company SystematIC Design and the solar panels, the power sources, were provided by Dutch Space.

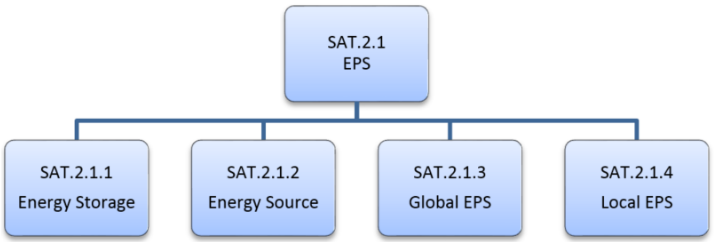


Figure 3.10: EPS configuration items [4, p. 14]

Energy storage

The energy is stored on four Li-ion batteries with a total capacity of 34 Wh. The battery temperature is measured by four temperature sensors and the batteries are placed in a customized aluminium bracket. The batteries can be seen in Figure 3.11.

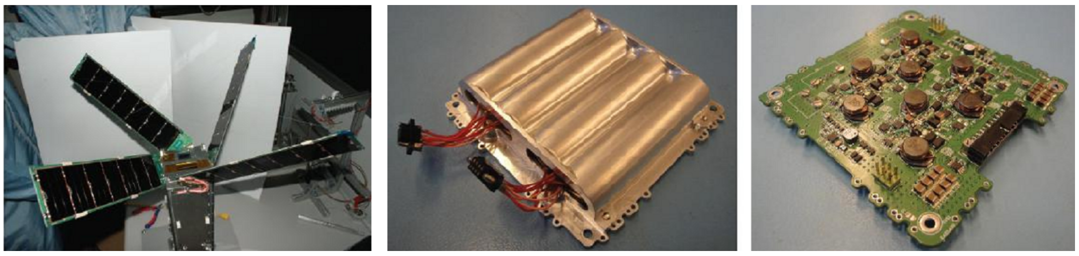


Figure 3.11: *Left:* Solar panel deployed configuration *Middle:* Battery bracket *Right:* BMS board [3, p. 11-12]

Energy source

The power source of Delfi-n3Xt consist of four double sided deployable solar panels which are mounted on a PCB substrate. The deployed configuration can be seen in Figure 3.11.

Global EPS

- *Maximum Power Point Tracker (MPPT) board:* The purpose of this board is to increase the efficiency of the solar panels depending on the IV-curve at a certain point in time. This board includes eight MPPTs and the power is supplied to the variable voltage bus up to a limit of 30V. Upon reaching the limit, the power point shifts towards the open circuit point.
- *Control & Regulation board:* This board is responsible for converting the incoming variable voltage to the regulated 12V bus. Also redundant micro controllers are located which retrieve data from the MPPT controllers and provide them to the OBC on request.
- *BMS board:* The BMS manages the charging and discharging of the batteries. When the variable voltage exceeds 22V the batteries are charged, and when the variable voltage drops below 18V the batteries are discharged if possible. The board can be seen in Figure 3.11.

Figure 3.12 shows the interfaces of the EPS. One can clearly see that the subsystems are only powered by the regulated 12V bus.

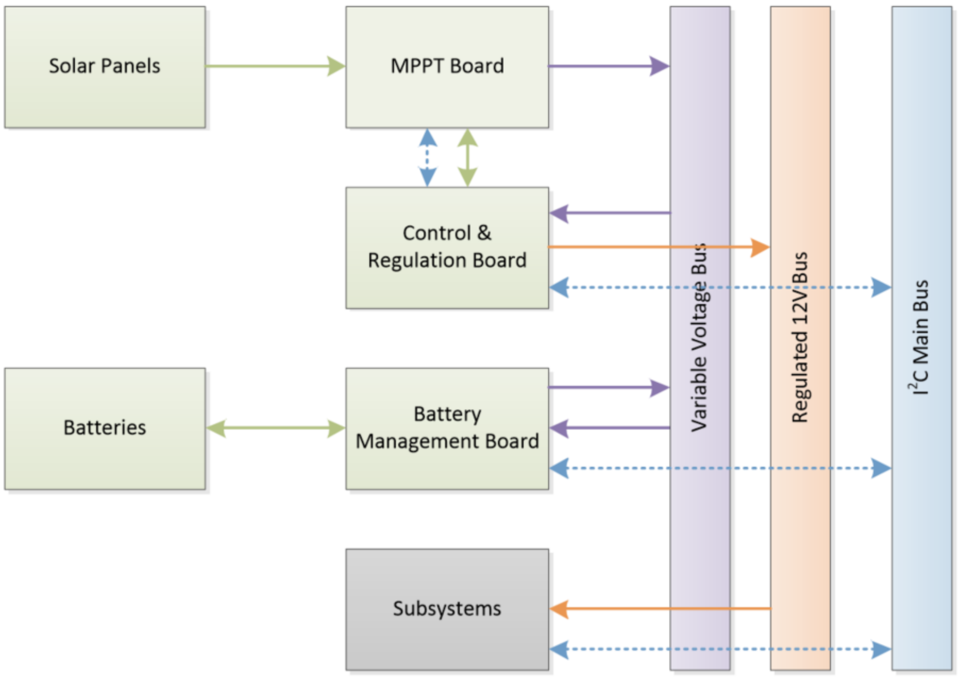


Figure 3.12: EPS interfaces[4, p. 15]

Local EPS

The purpose of the local EPS is to convert the global 12V bus to usable voltages like 5V and 3.3V. These voltages are used by the electronics used on the PCBs of various subsystems. These local converters are not considered to be part of the EPS but of the CDHS-DSSB. It is mentioned here to complete the power supply chain.

3.3.6. Command and Data Handling Subsystem (CDHS)

The CDHS can be simplified into two systems as shown in Figure 3.13. The OBC serves as the brain of the satellite and performs fully autonomously while the DSSB can be considered as the nerve system of the satellite.

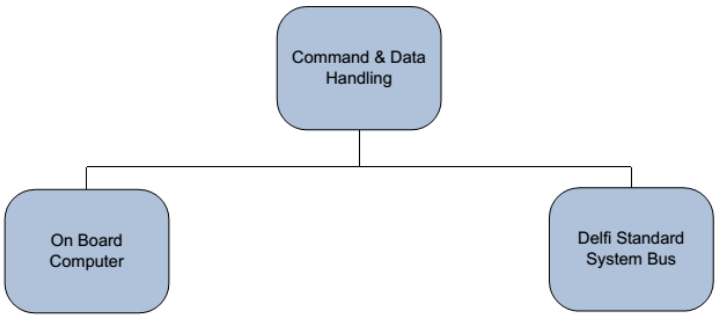


Figure 3.13: EPS interfaces[4, p. 15]

On-Board Computer (OBC)

The OBC consists of two micro controllers (one redundant) and time counters with backup power. The software consists of multiple operational modes which can be seen in Figure 3.14 and the main activity flow can be seen in Figure 3.15. Once the OBC is powered by the regulated bus, various parameters are loaded from flash memory. The INIT vector is loaded, indicating the state of each subsystem and which Micro controller (μC) acts as the master and which one acts as the slave. After this initialization process, the main loop is executed. This main loop has a frequency of 0.5 Hz and performs a repetitive sequential series of actions.

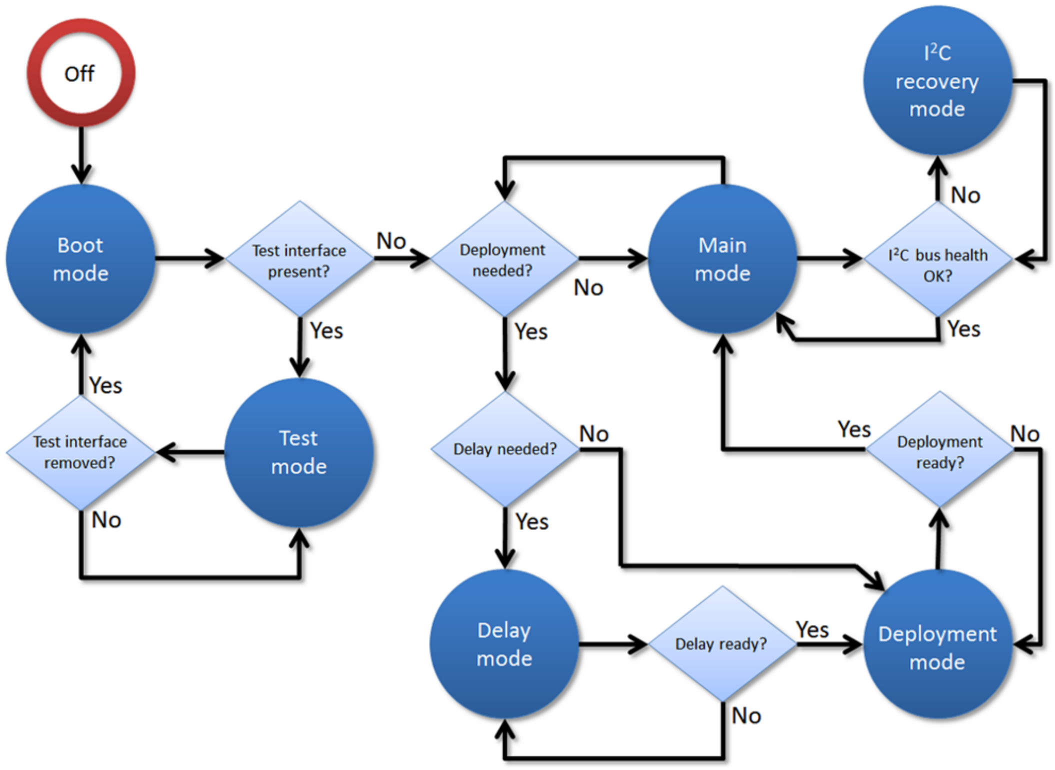


Figure 3.14: OBC system modes[31]

The main mode consists of two sub-modes: the sun-lit part of the orbit and the eclipse part of the orbit. Both sub-modes are similar; only a different INIT vector is loaded from flash memory.

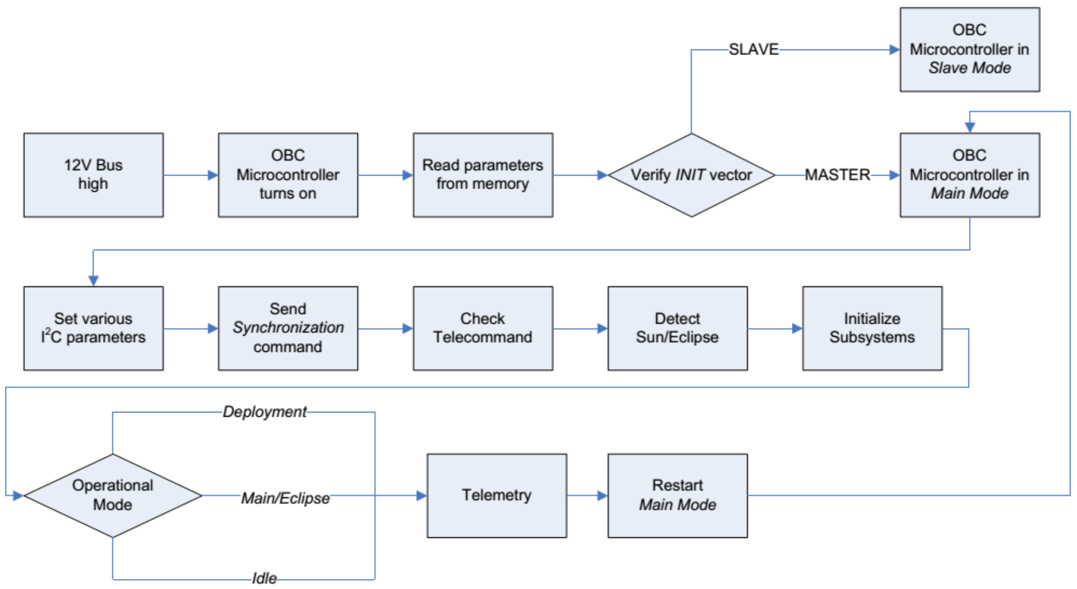

12V Bus high
OBC Microcontroller turns on
Read parameters from memory
Verify INIT vector
SLAVE
OBC Microcontroller in Slave Mode
MASTER
OBC Microcontroller in Main Mode
Set various I²C parameters
Send Synchronization command
Check Telecommand
Detect Sun/Eclipse
Initialize Subsystems
Operational Mode
Deployment
Main/Eclipse
Idle
Telemetry
Restart Main Mode

Figure 3.15: OBC activity flow[3, p. 65]

Delfi Standard System Bus (DSSB)

The DSSB is an standardized data and power interface which connects all subsystems used in Delfi-n3Xt. It consist of:

- Flex-rigid wiring harness
    - Standard 20-pins Harwin Datamate M80 connectors
    - Flexible parts providing thermo-mechanical stress relief
    - I^2C Data interface (including redundant wires)
    - 12V power interface (including redundant wires)
- Protection circuit and software placed on each connected PCB
- Interface for external monitoring and commanding

Figure 3.16 provides a schematic representation of the DSSB protection circuit. The circuit provides protection to failures of subsystems affecting the I^2C bus. The circuit resets the subsystem when synchronization commands, transmitted by the master OBC, are not being received.

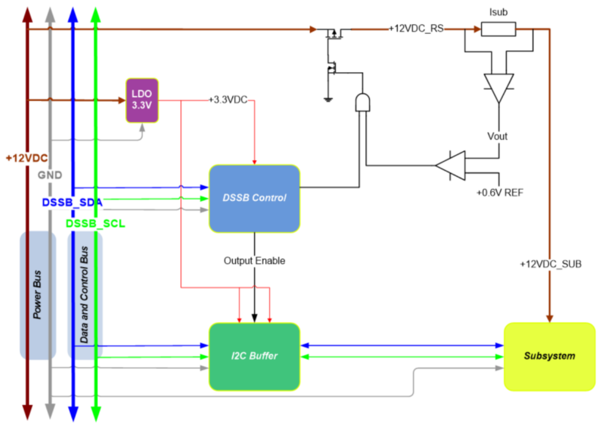


Figure 3.16: DSSB protection circuit[4, p. 23]

The OBC and flex rigid interface can be seen in Figure 3.17.

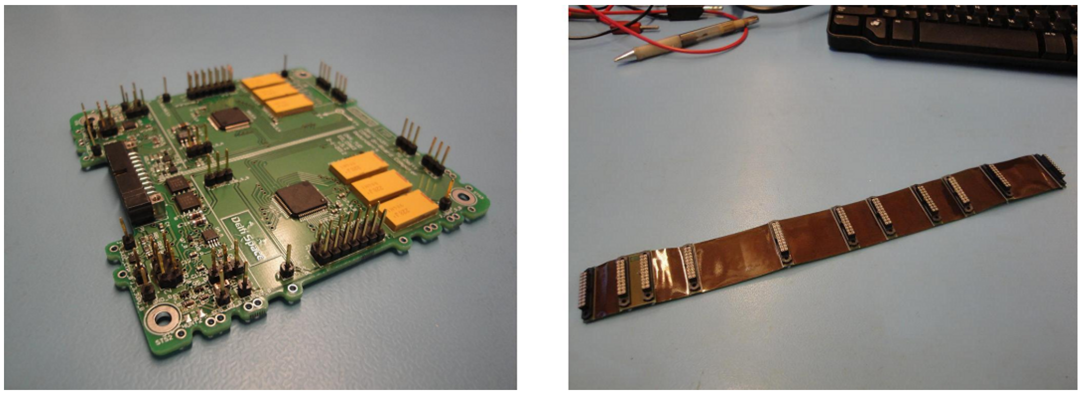


Figure 3.17: *Left:* OBC *Right:* DSSB flex rigid[3, p. 15]

3.3.7. Payload

As can be seen in Figure 3.3 Delfi-n3Xt carries three payloads: ITRX $\mathrm{T}^3\mu PS$, and SDM. ITRX has already been discussed in Section 3.3.2. The other payloads are explained here and displayed in Figure 3.18.

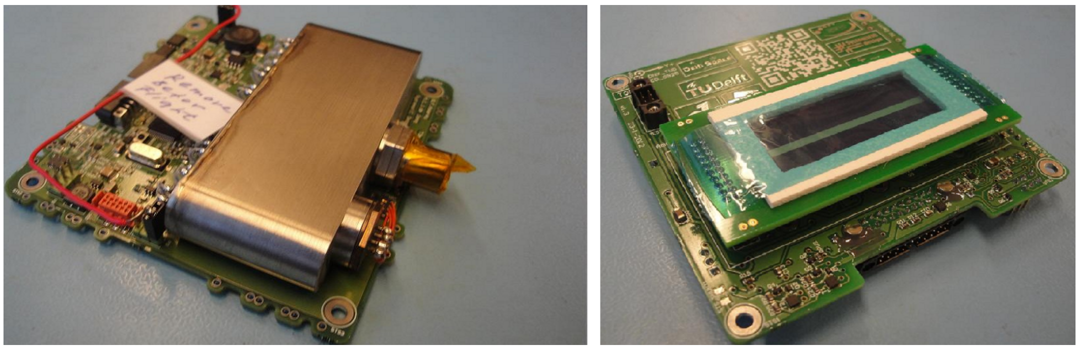


Figure 3.18: *Left:* $\mathrm{T}^3\mu PS$ *Right:* SDM [3, p. 17]

Micro Propulsion System ($\mathrm{T}^3\mu PS$)

This payload has been developed by TNO, TU Delft and the university of Twente. The propulsion system consists of eight cold gas generators. Nitrogen is stored in solidified form and transforms towards the gas upon heating the element. Due to the state transformation pressure builds up in the chamber. Thrusting is then performed by opening the valve and expelling the nitrogen.

Silicon Solar Cell Degradation Measurement (SDM)

DIMES developed an experiment to measure the degradation of silicon solar cells. 14 Small solar cells are characterised by measuring eight points on the IV-curve. Beyond that the temperature is registered and the update frequency is synchronised with the OBC. By measuring these variables the degradation of the solar cells can be analysed. The PCB also contains a test connector interface, so the satellite power and data lines can be accessed externally through an opening in the top panel (which can be seen in Figure 3.8.

3.4. Authors opinion and TRL

Delfi-n3Xt contains some significant improvements with respect to Delfi-C3. The addition of an active ADCS system and the $\mathrm{T}^3\mu PS$ provide the satellite many more capabilities. Also the use of a single antenna system for the UHF and VHF band is considered advantageous since this significantly reduces the volume and mass of the satellite. Beyond that, the implementation of the DSSB should increase reliability of the I2C bus as most subsystems can be isolated from the system. This is therefore considered as a welcome addition.

To get a feeling of the risks Delfi-n3Xt was facing after launch, the TRL are determined for the various subsystems or PCBs of Delfi-n3Xt. The TRL describes the maturity of a system and is expressed on a scale from 0 to 9. A high value describes a high maturity. In this report the level definition of ISO standard 16290 Space systems[10] are used which can be found in Appendix D.

The TRLs are based upon internal documentation found on the Delfi webdrive and the values are provided in Table 3.3. It is observed from Table 3.3 that the LT and STX have a low TRL as these systems were not tested or reported. Also the OBC has a fairly low TRL. These values are considered throughout the rest of this thesis. It must be noted that one of the mission goals is to demonstrate and test new technologies. Therefore, low TRLs are found, and are not considered major issues. The use of barely tested systems however, is not recommended.

Table 3.3: TRL table

System	TRL	Argument
EPS	5	System efficiency test performed[17], but not in relevant environment.
ADCS	4	System functionality tests performed [33], but not in relevant environment, advanced modes not tested on system level. Components tested in relevant environment.
$\mathrm{T}^3\mu PS$	5	Test performed by TNO, no test results available. In-house test planned, but not documented or performed.
OBC	4	System level test performed, but not all commands were used[3]. Relevant environment test was planned, but not reported[16].
DSSB	5	Functional tests performed[3] and no environmental tests reported.
PTRX	8	Design based on Delfi-C3(2008), Flight heritage PWSAT(2012)
STX	3	Based on CC2500 from Texas Instruments, no tests reported.
ITRX	7	Same as PTRX with slight adjustments.
LT	3	No test documents available but simple functional test reported by project manager.
DAB	6	Solar panel and antenna deployment test performed in relevant environment [13][14][3]. Phasing circuit tested in laboratory[12].
SDM	5	Functional tests performed[3] and no environmental tests reported.

4 Review of operations

This chapter provides a review of operations, which is identified by *AI 2.1.3*. By reviewing the operations various hypotheses are generated based on the performed analysis and observations. Some hypotheses are directly linked to the loss of the satellite and some are not. All hypotheses are however considered important as all of them might be related to the unfortunate loss of contact with Delfi-n3Xt. The operations are reviewed using the various sources stated below:

- Telemetry[18].
- Delfi-n3Xt operational logs which include transmitted commands (Delfi-mailbox)[29].
- Delfi-n3Xt TLEs[23].
- Audio recordings of DGS-passes[28].
- Previous work performed by R.Schoemaker[22].

The chapter starts with a brief analysis of the TLEs in Section 4.1. The following section, section 4.2 introduces telemetry and telecommands. In Section 4.3 an overview of the mission time-line is provided and it describes various performed operations. To generate hypotheses, an anomaly identification and analysis is performed in Section 4.4. The last section , Section 4.5 discusses the last events before loss of contact.

4.1. Orbit analysis

During the mission, and after, various collision warnings were provided by the Orbital Protection Team of Joint Space Operations Center (JSpOC). Unfortunately, these warnings were not stored on the mailbox[29]. It does raise the question however, whether Delfi-n3Xt was lost due to a collision with space debris or other objects. This is therefore stated as H1.

Hypothesis H1:

Observation: Collision warnings.
Question: What caused the loss of contact with Delfi-n3Xt?
Hypothesis: Delfi-n3Xt was lost due to an in-orbit collision with an object.

This section provides an overview of the orbital parameters of Delfi-n3Xt. These parameters extracted from the TLEs provided by Space-Track[23]. Delfi-n3Xt has a designated NORAD ID of 39428 or with the international designator 2013-066N. Figures 4.1 to 4.7 state the various orbital parameters. These figures are used to determine whether Delfi-n3Xt is still in orbit and whether the orbital parameters changed upon loss of the satellite. A sudden change in one or more of these parameters could indicate an impact.

The figures do not show any sudden change in orbital elements at the time that Delfi-n3Xt was lost. Also Delfi-n3Xt is still being tracked and therefore hypothesis H1 doen't appear to be likely at all. Interestingly, there are some outliers in eccentricity and inclination from 9-6-2014 to 17-9-2014. Since this is not part of this thesis this will not be further discussed in this report.

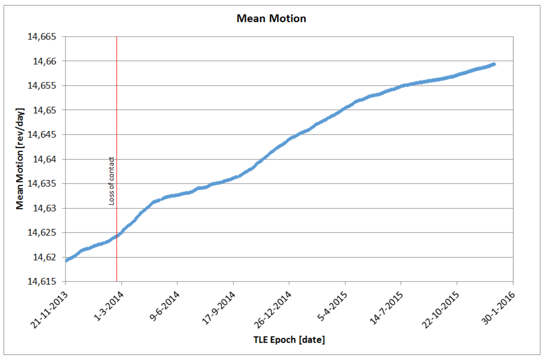


Figure 4.1: Mean motion

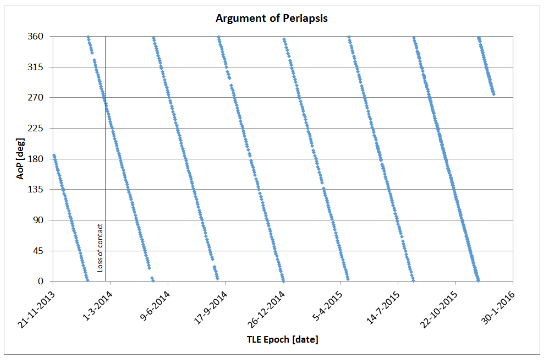


Figure 4.2: Argument of Periapsis

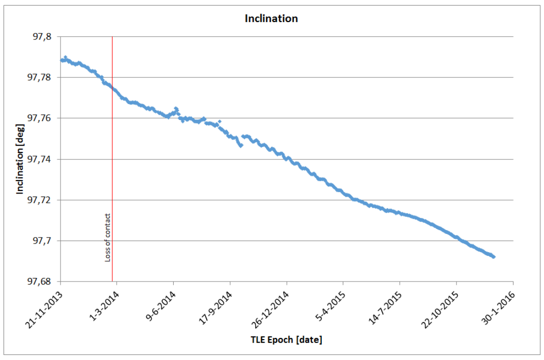


Figure 4.3: Orbit inclination

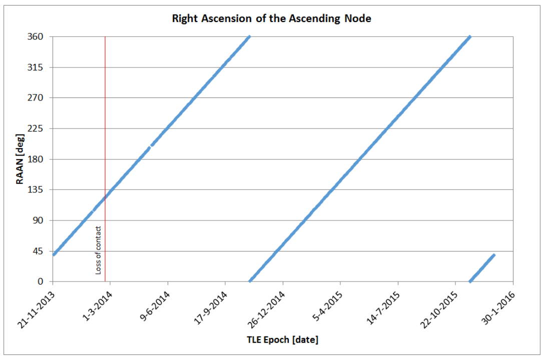


Figure 4.4: Right ascension of ascending node

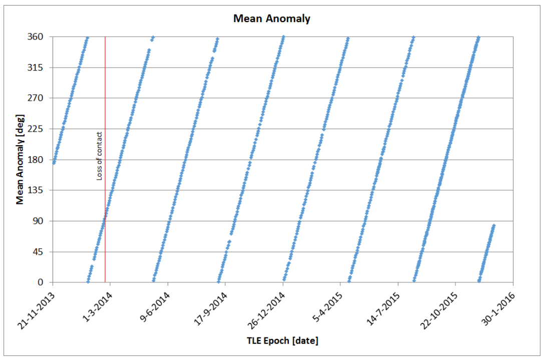


Figure 4.5: Mean anomaly

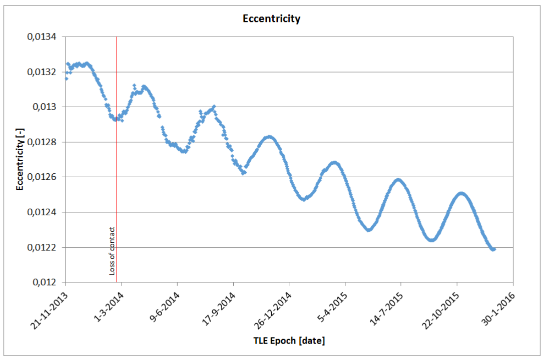


Figure 4.6: Orbit eccentricity

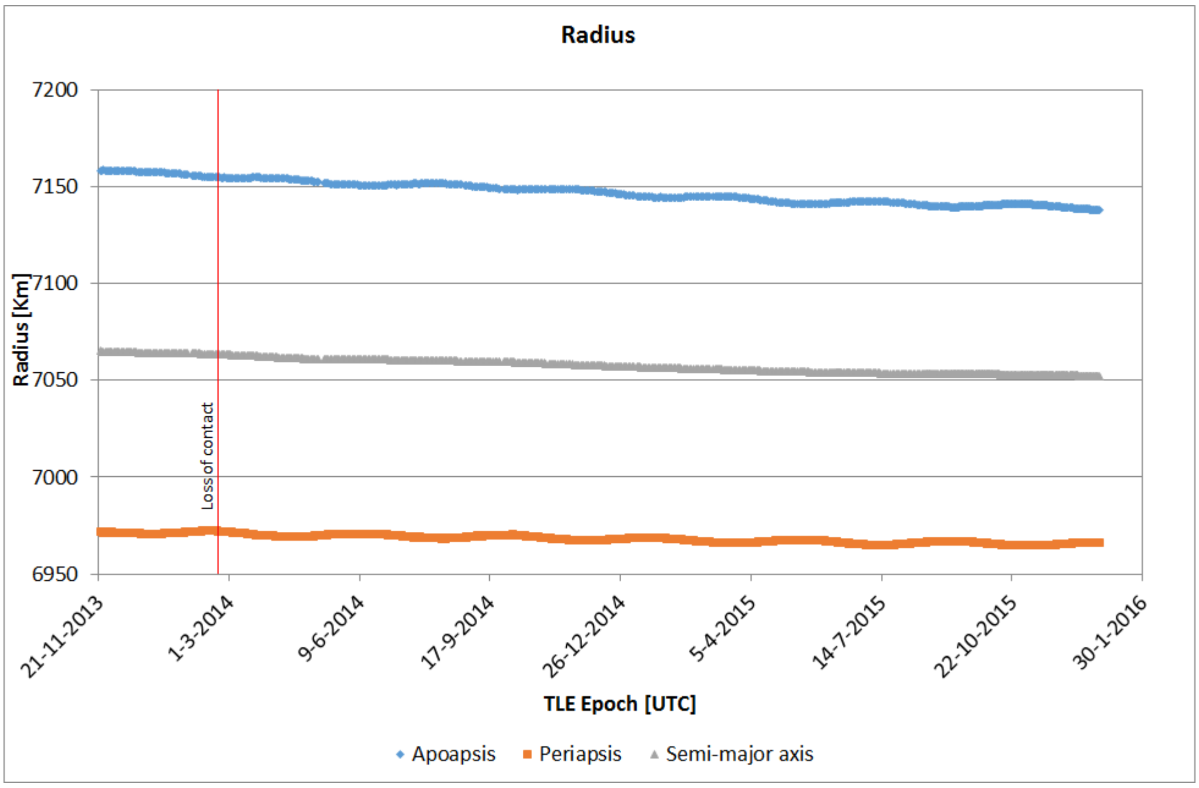


Figure 4.7: Orbit radius

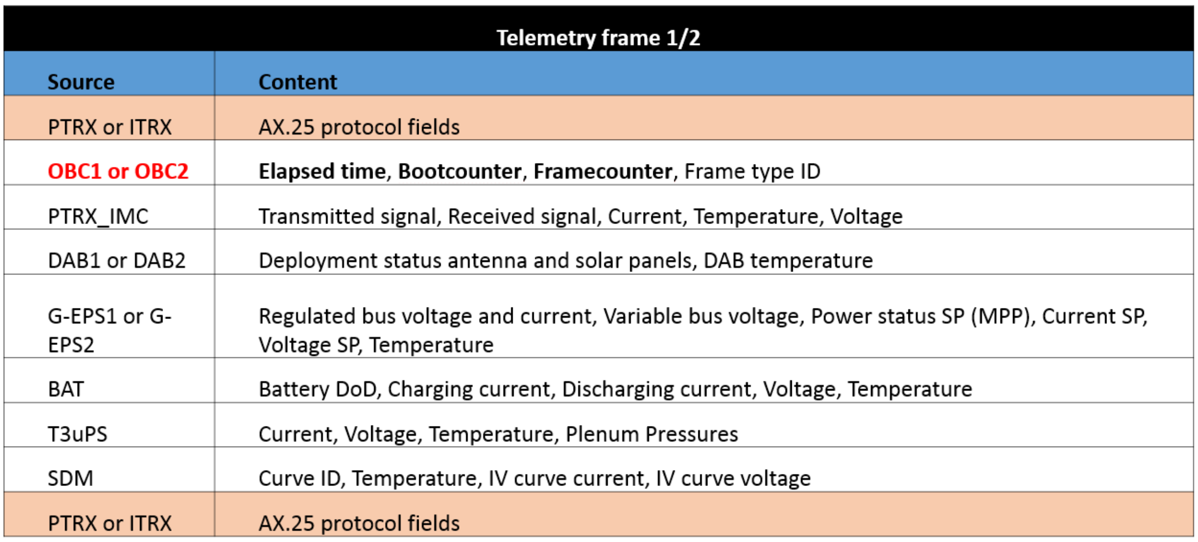
4.2. Introduction to telecommands and telemetry

This section provides an introduction into the telemetry and telecommanding of Delfi-n3Xt. The telemetry is discussed in subsection 4.2.1 and the telecommands are discussed in 4.2.2. The last section, section 4.2.3 briefly describes a process that was required to obtain a full telemetry dataset for further analysis.

4.2.1. Telemetry

The telemetry transmitted by Delfi-n3Xt using the PTRX or ITRX consists of two separate frames which are combined in a single package. Telemetry frame 1 is transmitted in the first second of the OBC main loop and telemetry frame 2 is transmitted in the second second. Delfi-n3Xt therefore continuously transmits telemetry frames while being in 'main mode'. The telemetry is received by the ground stations and demodulated using the DuDe client. This software package then transmits the raw frames, consisting of hexadecimals, towards the telemetry server which stores the raw frames in a database. On the telemetry server a continuously running software script processes the frames into readable data (decimal numbers and strings), which can then be interpreted by operators or analysts. Figure 3.2 provides an overview of the interfaces.

A single telemetry package provides over 400 parameters. Figure 4.8 provides an overview of the parameters contained in telemetry frame 1 and Figure 4.9 provides an overview of telemetry frame 2. The entire telemetry layout can be found in [19].



Telemetry frame 1/2	
Source	**Content**
PTRX or ITRX	AX.25 protocol fields
OBC1 or OBC2	**Elapsed time**, **Bootcounter**, **Framecounter**, Frame type ID
PTRX_IMC	Transmitted signal, Received signal, Current, Temperature, Voltage
DAB1 or DAB2	Deployment status antenna and solar panels, DAB temperature
G-EPS1 or G-EPS2	Regulated bus voltage and current, Variable bus voltage, Power status SP (MPP), Current SP, Voltage SP, Temperature
BAT	Battery DoD, Charging current, Discharging current, Voltage, Temperature
T3uPS	Current, Voltage, Temperature, Plenum Pressures
SDM	Curve ID, Temperature, IV curve current, IV curve voltage
PTRX or ITRX	AX.25 protocol fields

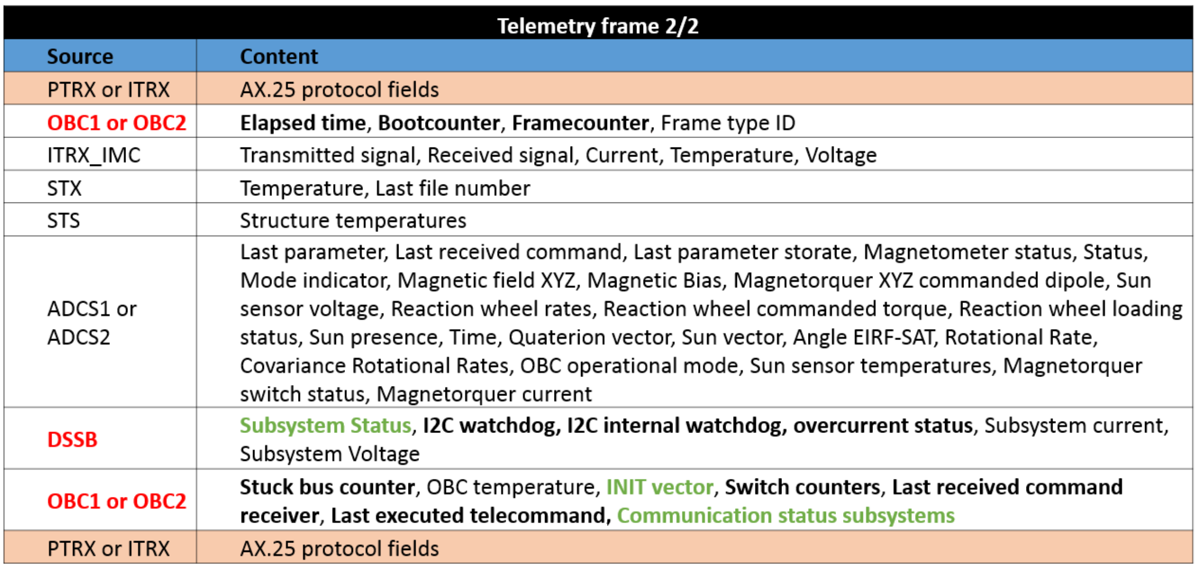
Figure 4.8: Overview telemetry frame 1[19]

In Figures 4.8 and 4.9 some parameters are highlighted. These parameters are used to identify anomalies, which are stated in Section 4.4.

4.2.2. Telecommands

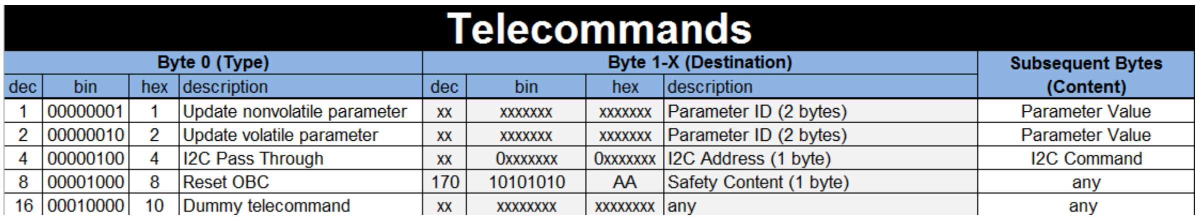
A telecommand generally consists of three parts: the command type, the destination, and the content/value. An overview is provided in Figure 4.10. There are 5 different types of commands:

- *Non-volatile parameter update:* By updating a non-volatile parameter the flash memory and RAM memory are overwritten with the provided values, at the provided destination.
- *Volatile parameter update:* By updating an volatile parameter the only the RAM memory is overwritten with the provided values at the provided destination.
- *I2C pass through:* An I2C pass through command is directly inserted on the I2C bus. This can be used to directly access connected subsystems.
- *OBC reset:* Resets the OBC
- *Dummy:* A dummy command has no influence on Delfi-n3Xt, except that the parameter 'last executed command' changes value (can be used to verify proper command reception).
- Any other data is ignored.



Telemetry frame 2/2	
Source	**Content**
PTRX or ITRX	AX.25 protocol fields
OBC1 or OBC2	**Elapsed time**, **Bootcounter**, **Framecounter**, Frame type ID
ITRX_IMC	Transmitted signal, Received signal, Current, Temperature, Voltage
STX	Temperature, Last file number
STS	Structure temperatures
ADCS1 or ADCS2	Last parameter, Last received command, Last parameter storate, Magnetometer status, Status, Mode indicator, Magnetic field XYZ, Magnetic Bias, Magnetorquer XYZ commanded dipole, Sun sensor voltage, Reaction wheel rates, Reaction wheel commanded torque, Reaction wheel loading status, Sun presence, Time, Quaterion vector, Sun vector, Angle EIRF-SAT, Rotational Rate, Covariance Rotational Rates, OBC operational mode, Sun sensor temperatures, Magnetorquer switch status, Magnetorquer current
DSSB	**Subsystem Status**, **I2C watchdog, I2C internal watchdog, overcurrent status**, Subsystem current, Subsystem Voltage
OBC1 or OBC2	**Stuck bus counter**, OBC temperature, **INIT vector**, **Switch counters**, **Last received command receiver**, **Last executed telecommand**, **Communication status subsystems**
PTRX or ITRX	AX.25 protocol fields

Figure 4.9: Overview telemetry frame 2 [19]



Telecommands								
Byte 0 (Type)				**Byte 1-X (Destination)**				**Subsequent Bytes (Content)**
dec	bin	hex	description	dec	bin	hex	description	
1	00000001	1	Update nonvolatile parameter	xx	xxxxxxx	xxxxxxx	Parameter ID (2 bytes)	Parameter Value
2	00000010	2	Update volatile parameter	xx	xxxxxxx	xxxxxxx	Parameter ID (2 bytes)	Parameter Value
4	00000100	4	I2C Pass Through	xx	0xxxxxxx	0xxxxxxx	I2C Address (1 byte)	I2C Command
8	00001000	8	Reset OBC	170	10101010	AA	Safety Content (1 byte)	any
16	00010000	10	Dummy telecommand	xx	xxxxxxxx	xxxxxxxx	any	any

Figure 4.10: Telecommand overview [19]

4.2.3. Database re-processing

The raw, binary telemetry is stored in a database. When telemetry is received it is stored in raw form and it is processed using a script to readable data which can be easily interpreted by humans. Inspection of the processed telemetry stored on the Delfi-drive[18] resulted in the following observations:

- The parameter *communication status* of the $T^3\mu PS$ is missing.
- The *last telecommand ID* is not registered in the first 360 frames.
- The DSSB parameters of the $T^3\mu PS$ contained the wrong name tags, leading to confusion for telemetry analysis.

It was thus concluded that the processing script did not meet the software interface document [19], and that the processing script was changed during operations. To obtain properly processed telemetry, the processing script was changed and the whole telemetry database was re-processed using a lengthy process, which resulted in a complete data set[21]. Throughout this report, it is assumed that the complete telemetry data set contains valid values unless it is specifically mentioned.

4.3. Mission time-line

During the mission, 46701 unique telemetry packages were received. Figure 4.11 provides cumulative plot of the received telemetry packages. The figure displays that the telemetry reception was not continuous. This is due to the fact that there was no global coverage of the ground stations and due to the fact that during eclipse no telemetry was transmitted to reduce power consumption. There are thus time-gaps between individual packages.

Figure 4.12 shows the mission time-line of Delfi-n3Xt, which is based on the telemetry[21], previous work of R.Schoemaker[22] and the Delfi mailbox[29]. The received commands by Delfi-n3Xt were identified by the *Last executed command* telemetry parameter. This parameter only states the command

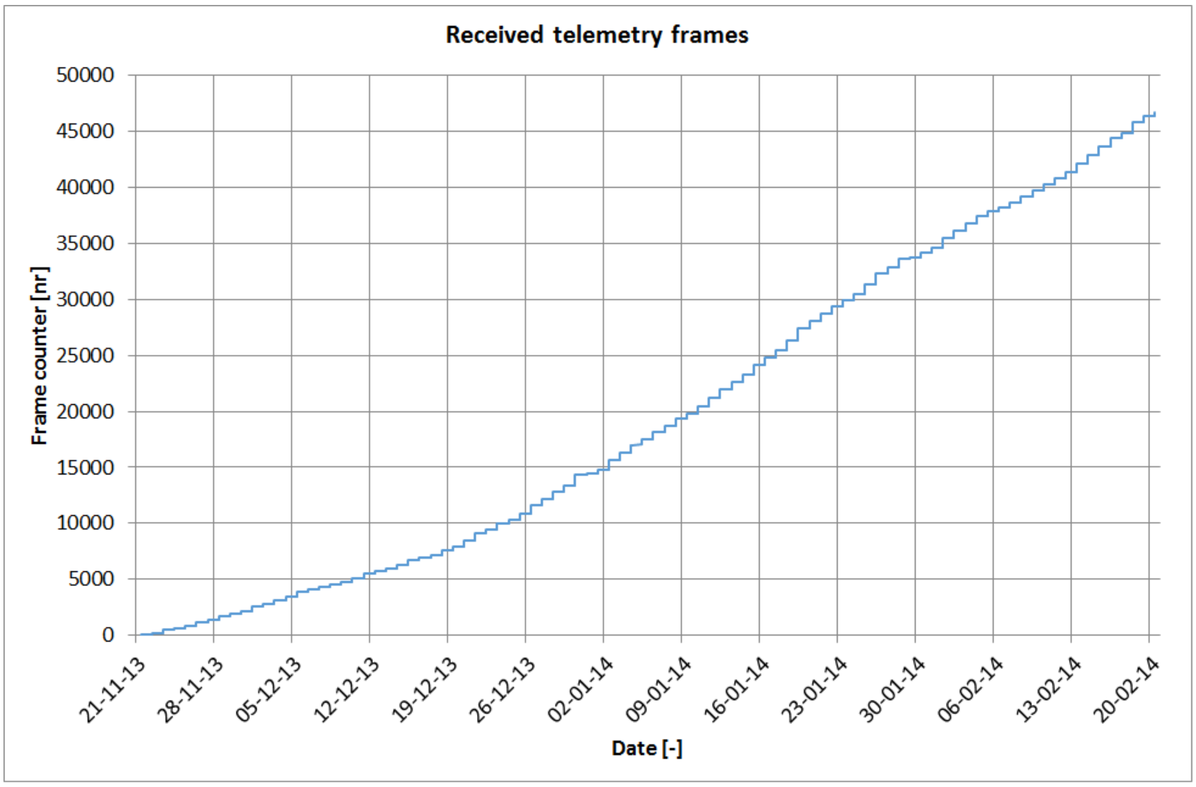


Figure 4.11: Telemetry package counter

type and destination and is therefore incomplete. The Delfi mailbox was used to obtain the complete command. As the mailbox was lacking, some command are unknown.

The time-line indicates some observations and most performed operations. It can be seen that Delfi-n3Xt was launched on 21-11-2013 and immediately after launch telemetry was received, indicating successful deployment. After deployment it was noticed that the tumble rate of Delfi-n3Xt was increasing instead of decreasing, which indicated a sign error in the ADCS software. Therefore, a command was uploaded to fix this anomaly. Beyond that the INIT vectors were updated to decrease power consumption as the batteries were discharging. This change resulted in fully charged batteries on 25-11-2013.

On 28-11-2013 the ADCS was switched from 'detumble mode' to 'manual mode' by a command, to determine the performance of the reaction wheels. Unfortunately, noise affected the magnetometers and therefore the tumble rate never achieved the required value of 1 deg/sec or lower. Therefore, the ADCS did to switch to other advanced modes [9], and thus the 'manual mode' was used. After this experiment the ADCS was switched back to 'detumble mode'.

The experiments using the $\mathrm{T}^3\mu PS$ started on 25-11-2013. Various thrust manoeuvres and Cold Gas Generator (CGG) ignitions were performed by uploading various I2C pass through commands. Experiments were successful until 11-12-2013. The CGG did not ignite upon successful reception of the command (the pressure did not increase). Afterwards various attempts were performed to ignite CGGs, but were all unsuccessful. During the thrust experiments, on 02-01-2013, an anomaly occurred after a thrust command of 4 seconds with a duty cycle of 50%. After command execution the the STX turned off and did not switch on any more. On 30-12-2013 an unknown I2C pass through command was transmitted to the STX. This command was probably to reset the subsystem, but nothing happened. Only after 1.5 months, on 23-01-2014, the STX switched on again after changing the switch counter value to 0. One day later however, the STX switched off again. Explanation of the switch counters and this anomaly can be found in Section 4.4.4.

During operations various INIT vectors were transmitted to Delfi-n3Xt to (de-)activate subsystems in OBC 'main-mode', 'eclipse-mode' or the 'current-mode' using volatile and non-volatile commands. On 05-12-2013 various anomalies occurred after transmission of an invalid INIT vector. This will be discussed in more detail in Section 4.4. From launch until 19-12-2013 the ITRX was used to transmit telemetry. Afterwards the PTRX was used, which provided a better signal. Eventually, on 20-2-2014 a transponder test was performed by uploading an INIT vector. The first test was unsuccessful and the satellite was restored to regular telemetry mode. A second attempt was performed and this was however the last moment of contact which is described detail in Section 4.5.

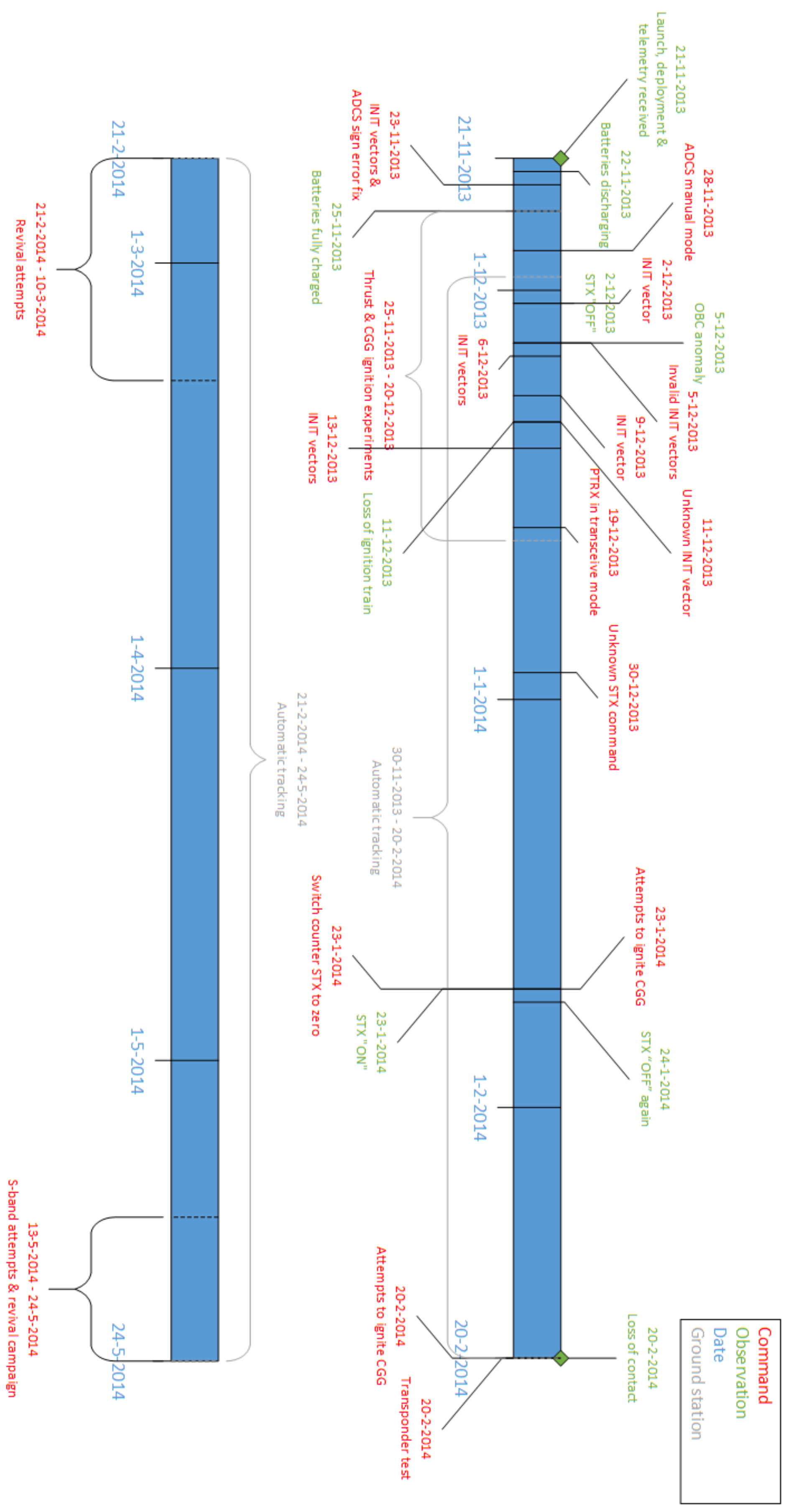

Command
Observation
Date
Ground station
21-11-2013 Launch, deployment & telemetry received
22-11-2013 Batteries discharging
23-11-2013 INIT vectors & ADCS sign error fix
25-11-2013 Batteries fully charged
28-11-2013 ADCS manual mode
2-12-2013 INIT vector
2-12-2013 STX "OFF"
5-12-2013 OBC anomaly
5-12-2013 Invalid INIT vectors
6-12-2013 INIT vectors
9-12-2013 INIT vector
11-12-2013 Unknown INIT vector
11-12-2013 Loss of ignition train
13-12-2013 INIT vectors
19-12-2013 PTRX in transceive mode
25-11-2013 - 20-12-2013 Thrust & CGG ignition experiments
30-12-2013 Unknown STX command
30-11-2013 - 20-2-2014 Automatic tracking
23-1-2014 Attempts to ignite CGG
23-1-2014 Switch counter STX to zero
23-1-2014 STX "ON"
24-1-2014 STX "OFF" again
20-2-2014 Loss of contact
20-2-2014 Attempts to ignite CGG
20-2-2014 Transponder test
21-11-2013
1-12-2013
1-1-2014
1-2-2014
20-2-2014
21-2-2014
1-3-2014
1-4-2014
1-5-2014
24-5-2014
21-2-2014 - 10-3-2014 Revival attempts
21-2-2014 - 24-5-2014 Automatic tracking
13-5-2014 - 24-5-2014 S-band attempts & revival campaign

Figure 4.12: Mission time-line based on [22], [21] and [29]

4.4. Anomaly analysis

This section describes an anomaly analysis based on the reprocessed telemetry packages[21]. An anomaly is defined as an event that should not have occurred during the mission. The anomalies are identified an analysed to identify various hypotheses.

This section starts with Section 4.4.1 where the various anomalies are introduced. Sections 4.4.2 to 4.4.10 elaborate on the various anomalies and Section 4.4.11 includes an overview of all identified anomalies.

4.4.1. Introduction

A telemetry package consists of two frames which provide over 400 parameters [19]. To find a possible explanation regarding the loss of Delfi-n3Xt, anomalies should be identified. There are various parameters, or relations between them, which might reveal interesting information. The identified relations and important parameters are stated below, and are highlighted in Figures 4.8 and 4.9:

- Counters:
  - Reboot counter
  - Stuck bus counter
  - Switch counter
- Status indicators:
  - INIT vector vs. DSSB status
  - DSSB status vs. Communication status
  - I2C watchdog
  - I2C internal watchdog
  - Over-current status
  - Simultaneous transmission

It must be noted that other parameters like temperature, voltage or current are also parameters that might reveal anomalies. These are however not considered separately since those parameters are reviewed upon identification of one of the previously mentioned anomalies.

For the following subsections it is important to realise that there was *no* continuous flow of telemetry packages during operations, as described in Section 4.3. It is therefore reasonable to assume that not all anomalies are detected.

4.4.2. Reboot counter

The reboot counter registers the number reboots of the satellite. This is provided by the OBC. The reboot counter might reveal information regarding the workings of the OBC and is therefore investigated. In Figure 4.13 the counter value over time is plotted.

In theory, a reboot should not have occurred during the mission. As can be seen from Figure 4.13 the counter is increasing with time. It is therefore interesting to determine why those reboots occurred, as they might be related to the loss of Delfi-n3Xt. Within the three months of operation 198 reboots occurred, which results in a rough average of 2 reboots per day. The relation seems to be linear and therefore immediately raises questions: Why did Delfi-n3xt reboot?; Why does it appear to be linear?; Is it related to the loss of contact? Possible answers are found in the next subsections.

Previous work by R. Schoemaker[22]

One of the operators of Delfi-n3Xt, R. Schoemaker, already performed an analysis on the various reboots. His hypothesis was that the reboots occurred due to radiation. Since the radiation levels are higher at the South Atlantic Anomaly (SAA) he tried to find a relation between the location of Delfi-n3Xt at the time of reboot. Unfortunately he was unable to find a statistical relation. This does however, not reject the hypothesis and therefore a similar hypothesis is used in this report.

Hypothesis H2

Observation: The telemetry indicates that 198 reboots occurred in 3 months of operation.
Question: What caused those reboots?
Hypothesis: The reboots experienced by Delfi-n3Xt were caused by radiation.

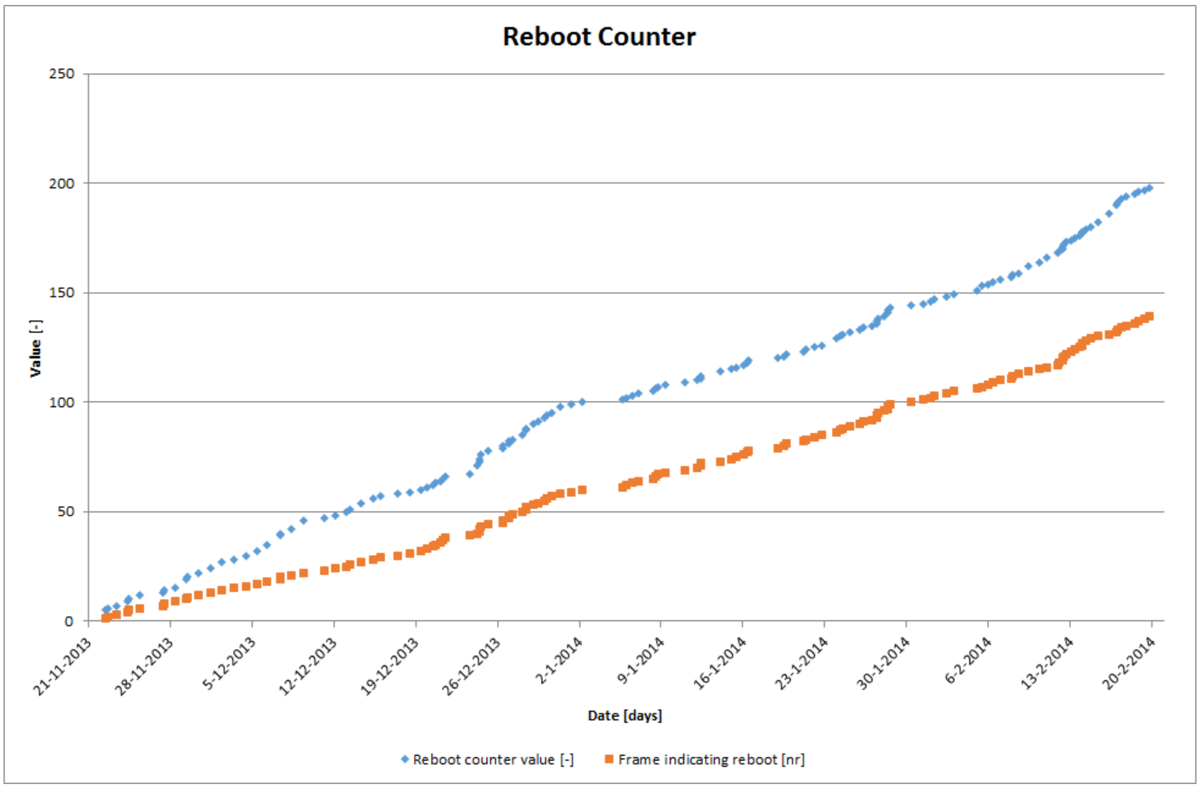


Figure 4.13: Reboot counter value during mission.

R. Schoemaker suggests another hypothesis. He suggests that the reboots occurred due to I2C lock-ups as the OBC is one of the last subsystems to perform a power reset during the recovery procedure. Also this hypothesis is noted. In that case however, the stuck bus counter, which is also provided by the telemetry, should increase in value. This however did not occur as stated in Section 4.4.3.

Hypothesis H3

Observation: The telemetry indicates that 198 reboots occurred in 3 months of operation.
Question: What caused those reboots?
Hypothesis: The reboots experienced by Delfi-n3Xt were caused by I2C lock-ups.

Current work

A possible explanation of the quasi-linear behaviour is that some sort of parameter overflow occurred which results in 2 reboots each day. This is therefore Hypothesis H4:

Hypothesis H4

Observation: The telemetry indicates a Quasi-linear reboot behaviour.
Question: Why does the reboot behaviour appear to be linear?
Hypothesis: The quasi-linear reboot behaviour of Delfi-n3xt was caused by a parameter overflow.

To further investigate this behaviour, the telemetry frames are examined in more detail. To acquire more information the *change* in counter value is investigated. Comparing the last frame *before* a reboot, and the first frame *after* a reboot might reveal the actual cause of the reboot. As stated before, the stream of telemetry frames was not continuous as there was no global coverage of Delfi-n3Xt ground stations. Therefore, the change of the reboot counter over time is plotted in Figure 4.14. A higher number of reboots/sec is likely to contain more useful information regarding the cause of the reboot. To be able to find the last frame before a reboot and the first frame after a reboot, a value of 0.5

reboots/sec should be found which should be reported by a single frame, as the time between two transmitted telemetry packages is 2 seconds. From Figure 4.14 it is observed that only 9 frames might be worth comparing, since the time between two consecutively received frames is small. In Table 4.1 the frames are compared.

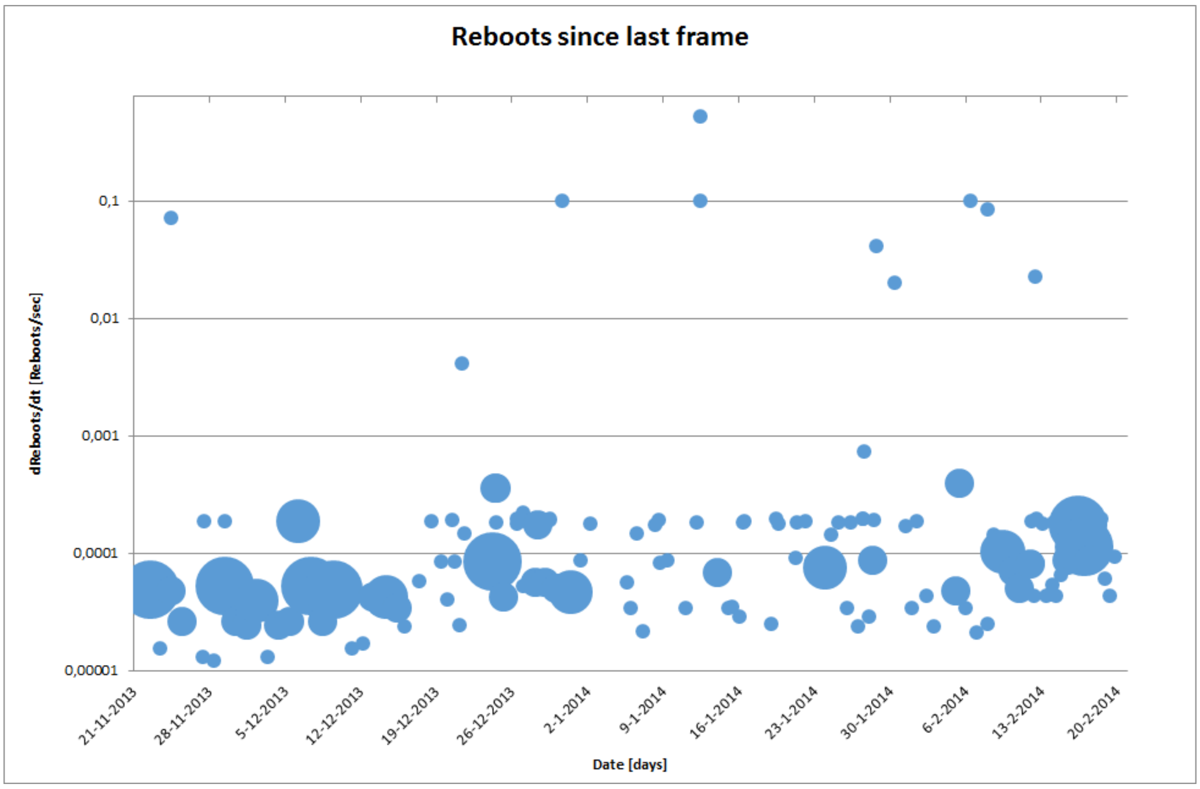


Figure 4.14: Reboots per second since last frame. The size of the markers indicates the number of reboots since last received frame.

Table 4.1: Observations reboot behaviour

Corrected time (UTC)	Observations frame before reboot	Observations frame reporting reboot	Frame counter since reboot	Frame counter before reboot
24-11-2013 10:42	Frame counter set 0, OBC Temperature sensor at minimum value, Switch events(ADCS1, DAB1, DAB2, ITRX), Communication failure(ADCS1)	None	6	11727
30-12-2013 15:39	Jump in current over solar panel(high to low), Switch events(ADCS1, DAB1)	Switch events(ADCS1)	1	26348
12-1-2014 09:05	Frame counter set 0, Switch events(ADCS1, DAB1, DAB2, ITRX), INIT events(MTQx, MTQy, MTQz), Communication failure(ADCS1)	SDM not registering all parameters	1	12829
12-1-2014 10:50	None	Simultaneous transmission PTRX, ITRX	3	3128
28-1-2014 16:02	None	None	9	10425
30-1-2014 09:02	None	Simultaneous transmission PTRX, ITRX	23	73783
6-2-2014 10:15	None	Simultaneous transmission PTRX, ITRX	2	22705
8-2-2014 01:41	None	None	3	4262
12-2-2014 09:08	None	None	6	5804

In Table 4.1 it can be seen that the frame counter before rebooting varies significantly. Therefore hypothesis H4 seems less likely. Comparing the various observations of the 9 frames does, unfortunately, not lead to relations between the frames nor a clear cause. Some anomalies are identified however. The anomalies are further discussed in this chapter.

4.4.3. Stuck bus counter

The stuck bus counter increases value when the the OBC does not receive an acknowledgement after transmission of a synchronization command over the I2C bus. In that case, the I2C bus is stuck and OBC changes modes towards the I2C recovery mode[31]. A stuck bus prevents internal communication which might lead to loss of the satellite and is therefore a point of interest. The telemetry of Delfi-n3Xt[21] indicates a constant counter value of 239 since the first received frame, and is therefore not plotted. This raises two questions: Why is the stuck bus counter constant?; And why is the value of the stuck bus counter set at 239? Possible answers are stated in hypotheses H5, H6, and H7.

Hypothesis H5

Observation: The telemetry indicates that the stuck bus counter has a constant value of 239.
Question: Why is the stuck bus counter constant?
Hypothesis: The I2C bus did not get locked since deployment mode.

Hypothesis H6

Observation: The telemetry indicates that the stuck bus counter has a constant value of 239.
Observation: The stuck bus counter was not formally verified.
Question: Why is the stuck bus counter constant?
Hypothesis: The stuck bus counter is constant because the counter is not working as intended.

Hypothesis H7

Observation: The telemetry indicates that the stuck bus counter has a constant value of 239.
Question: Why is the stuck bus counter value fixed at 239?
Hypothesis: 239 stuck bus events occurred during the delay or deployment mode.

4.4.4. Switch counter

Switch counters register the number of reboot attempts for each subsystem to prevent infinite loops. When more than four attempts have been performed the INIT vector is changed as such that the subsystem won't reboot any more. Upon reboot success the switch counter is set to 0 again. This functionality is provided by the OBC. An increasing counter therefore indicates failure to initialise a subsystem and is therefore a point of interest. In total there are 16 switch counters.

Previous work by R. Schoemaker[22]

It was questioned before whether the switch counters were implemented correctly. As reported by R.Schoemaker there were troubles switching the STX back on-line. The telemetry indicated a switch counter value of zero, but the system did not switch on. Only after manually setting the counter to zero (23-01-2013) the STX successfully booted. Therefore, two hypotheses are suggested: H8 and H9.

Hypothesis H8

Observation: The telemetry indicates that the switch counter value of STX was zero.
Observation: A parameter update was required to turn on the STX again.
Question: What prevented the STX from booting?
Hypothesis: A software bug prevented the STX from rebooting.

Hypothesis H9

Observation: The telemetry indicates that the switch counter value of STX was zero.
Question: Why was the value of switch counter of the STX zero?
Hypothesis: A bit flip caused a switch counter value outside the telemetry range.

Current work

The troubles with the STX raises the question whether similar events occurred during the mission, which have not been detected. Possibly it is related to the loss of the satellite. By comparing the switch counter values of the various telemetry packages one can identify 'switch events'. Figure 4.15 provides the various switch events reported by the telemetry, including the affected subsystems. Each marker represents a package where the value of one or more counters has increased. Some switch events occurred in a small time frame and therefore the affected subsystems are less readable. Figure 4.15 still provides a good overview of the switch events with the affected subsystems.

In total there are 39 telemetry packages identified which indicate 106 switch events. Those events are spread over 16 day's and 17 passes [21]. It must be noted that some cases have been identified where the counter did not reset back to zero, although the subsystem has successfully rebooted. This plot does not show these telemetry packages. Interestingly, this supports both hypothesis H8 and H9, as the counter *should* reset to zero, but sometimes doesn't (indicating a bug) and therefore the counter value could rise outside the telemetry range (Telemetry: 2 bits per counter, OBC parameter: 40 bits per counter[19]).

The switch counter of the STX remains a topic of interest, as the counter value reported by the telemetry was not increasing, while the subsystem *did* shut down and the INIT vector was changed. This indicates that the mechanism that relies on the switch counter is correctly working, but the value reported by the telemetry is not. This however does not explain why the switch counter needed an external reset, as reported in Section 4.4.4. Therefore, it is expected that the STX switch counter increased in value upon boot failure, did not reset, and was not correctly transmitted by telemetry.

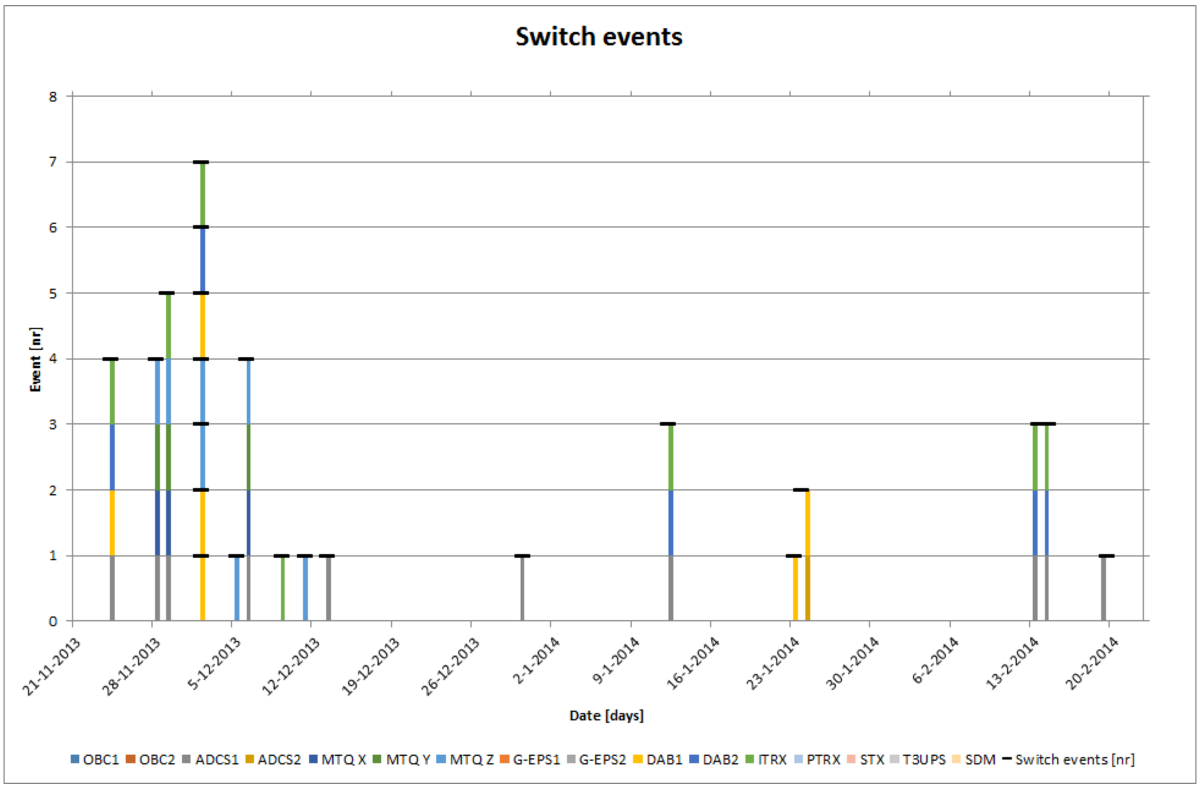


Figure 4.15: Switch events

Hypothesis H10:

Observation: The telemetry indicates that the switch counter of the STX did not increase in value.
Observation: The telemetry indicates that the INIT value of the STX changed to off while the switch counter was 0.
Observation: The telemetry indicates that the STX did not switch-on while the counter value was 0.
Observation: The telemetry indicates that the STX only switched-on after a parameter update of the switch counter.
Observation: The telemetry indicates that the other switch counters appear to work as intended.
Question: Why is the switch counter of the STX behaving differently to the other counters?
Hypothesis: The switch counter of the STX did not reset and was reported incorrectly by telemetry.

4.4.5. Status indicator: INIT vector vs. subsystem status

The INIT vector states which subsystem *should* be active and each DSSB μC states which subsystem *is* active. If the INIT vector doesn't correspond to the DSSB status indicator there is clearly something wrong. This is therefore defined as a 'INIT mismatch event.' It is therefore interesting to investigate. The INIT vector controls 16 subsystems. In Figure 4.16 the various INIT mismatch events are shown, including the affected subsystems.

In total there are 1754 telemetry packages indicating 5282 INIT mismatch events. The various events are unfortunately hard to identify in Figure 4.16, as the resolution is too high for proper display. Further analysis is performed in Section 4.4.11.

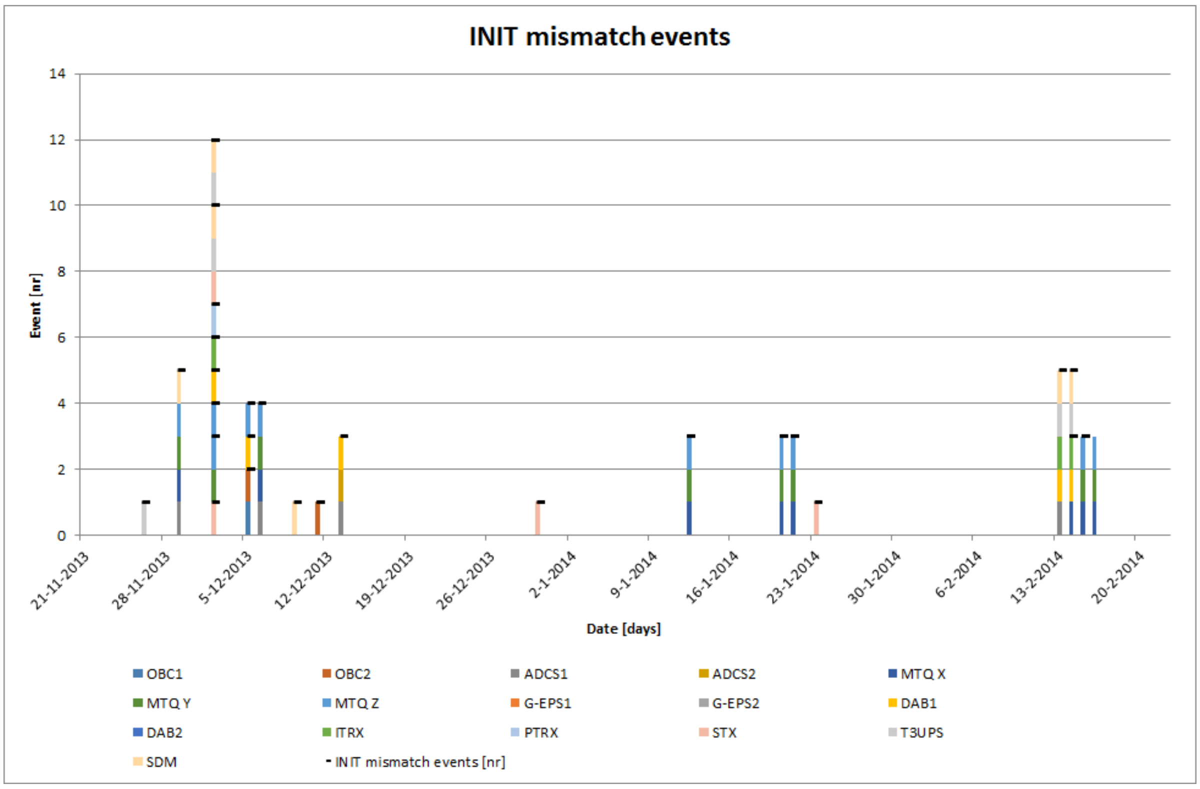


Figure 4.16: Frames indicating an INIT mismatch event

4.4.6. Status indicator: Subsystem status vs. communications status

The DSSB status indicator provides whether a subsystem is active, in that case the subsystem should be able to provide a data package to the OBC and the telemetry should indicate communication success. Therefore, if the communication status does not correspond to the DSSB status something is wrong. This is therefore defined as 'a communication failure'. In total there are 32 μC connected to the OBC using the I2C bus.

In total there are 206 telemetry packages indicating 1107 communication failures. As previously mentioned, the resolution for proper display is too high and the amount of connected μCs is too high. The plot still identifies the problem cases however. Further analysis is performed in Section 4.4.11.

4.4.7. Status indicator: I2C subsystem watchdog

The OBC transmits a SYNC pulse every 2 seconds. This SYNC pulse is received by all connected μC. Each DSSB μC contains a timer which resets upon receiving a SYNC pulse. When no SYNC pulse has been received before the timer reaches around 5000ms (different for each subsystem) the I2C subsystem watchdog is triggered, which switches the connected subsystem off[15, p.3]. This watchdog indicates therefore a communication failure between the OBC and the DSSB μC and is therefore a point of interest. There are 11 I2C subsystem watchdog timers located on Delfi-n3Xt. Figure 4.18 provides the reported events.

In total there are 19 telemetry packages indicating 39 subsystem watchdog events. Further analysis is performed in Section 4.4.11.

4.4.8. Status indicator: I2C DSSB watchdog

The I2C DSSB watchdog is similar to the I2C subsystem watchdog. The only difference is the timer value (in this case around 8000ms [15, p.3]). When the watchdog is triggered, the DSSB μC resets *itself*. There are 12 I2C DSSB watchdog timers. Figure 4.19 provides the reported events.

In total there are 19 telemetry packages indicating 36 subsystem watchdog events. These telemetry packages correspond to the I2C subsystem events reported in the previous section. This is peculiar as the subsystem watchdog *should* be triggered before the DSSB watchdog, as the timer trigger values are

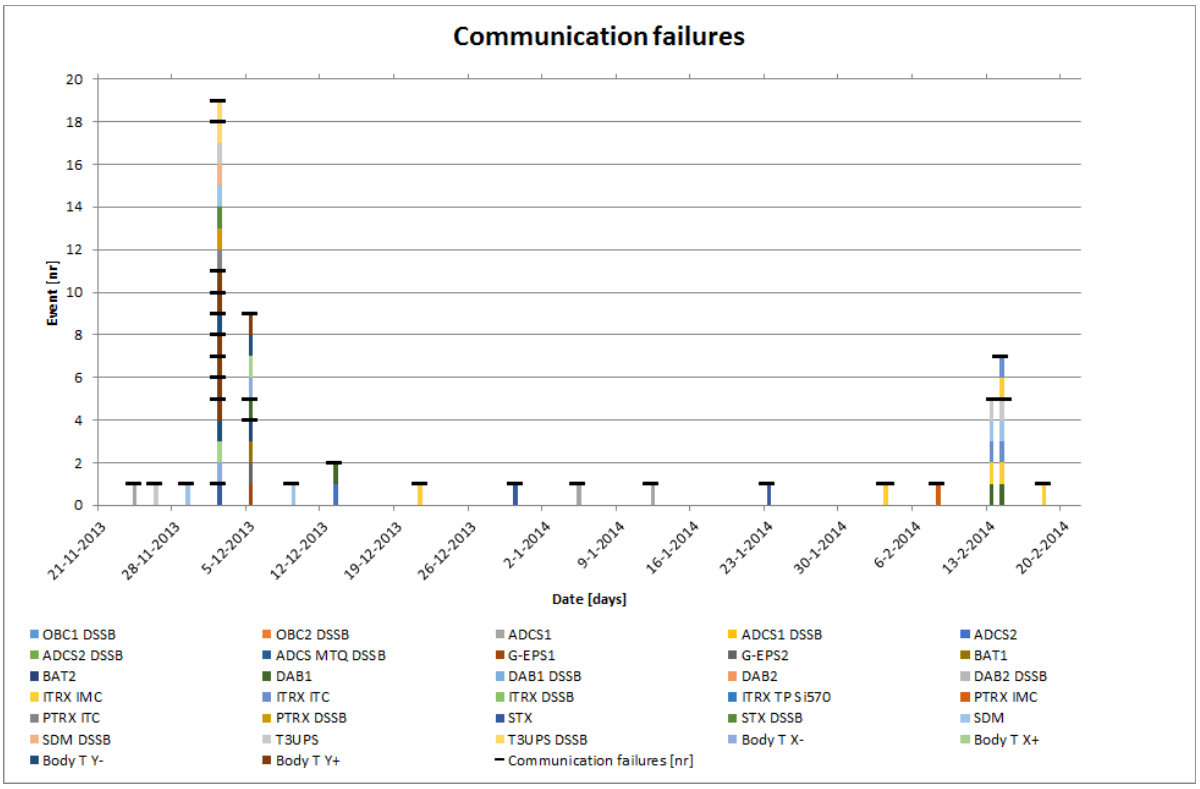


Figure 4.17: Telemetry packages indicating communication failures

different[19]. One would therefore expect that a telemetry package indicating a *subsystem* watchdog event is followed by a telemetry package indicating a *DSSB* watchdog event. A possible reason is stated in hypothesis H11.

Hypothesis H11

Observation: The telemetry indicates that I2C watchdog events correspond with each other.
Observation: The design documents[19] indicate that I2C watchdog events are triggered at different time intervals.
Question: Why are the I2C watchdogs triggered simultaneously?
Hypothesis: I2C watchdog events are triggered simultaneously because the software wasn't correctly implemented.

4.4.9. Status indicator: Over-current

Each DSSB μC provides a single bit indicating an over-current event. The current is determined by measuring a voltage over a resistor. If the voltage is higher than 0.6V, this event is triggered and the subsystem is rebooted. The telemetry does not include packages indicating an over-current event, which therefore results in hypothesis H12.

Hypothesis H12

Observation: The telemetry indicates zero over-current events.
Question: Why does the telemetry state zero over-current events?
Hypothesis: During the operations of Delfi-n3Xt no over-current events occurred.

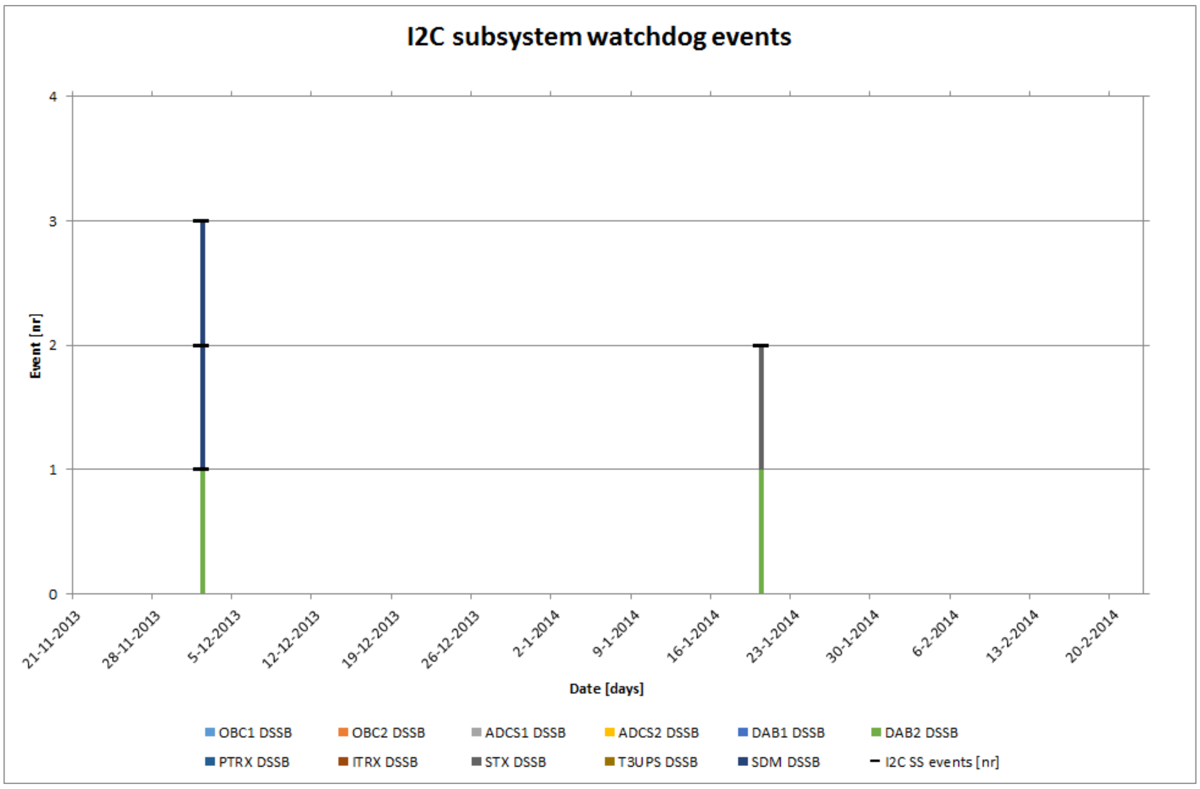


Figure 4.18: Frames indicating I2C subsystem watchdog event

4.4.10. Status indicator: Simultaneous transmission transceivers

Simultaneous transmission of the ITRX and PTRX can occur when both radios are set in transceiving mode. During development of Delfi-n3Xt, it was not verified what happens when both transceivers are transmitting. Since both radios are connected to the same phasing-circuit and antenna's, this is a large unknown factor, which could potentially prevent external communications. Therefore, simultaneous transmission of the transceivers is investigated.

Previous work by J.Bouwmeester[9]

The project manager of Delfi-n3Xt, J.Bouwmeester[9], performed an experiment (a functional test, not reported) with the ITRX after loss of satellite with spare hardware. He expected that the LT was causing problems leading to loss of contact. During this experiment he found that the ITRX does not switch off the LT when commanded to. It is thus expected that the LT did not switch-off upon entering eclipse, and that upon leaving eclipse simultaneous transmission occurred (PTRX in transceiving mode after leaving eclipse, see Section 4.5), which possibly leads to loss of the satellite since this situation was never tested or reported. Therefore, there are three hypotheses: H13, H14, H15.

Hypothesis H13:

Observation: J. Bouwmeester reported that the LT doesn't switch of when commanded to.
Observation: LT was not formally verified.
Question: Why doesn't the LT switch off upon receiving the shut-down command?
Hypothesis: A design or implementation flaw prevents the LT from shutting down.

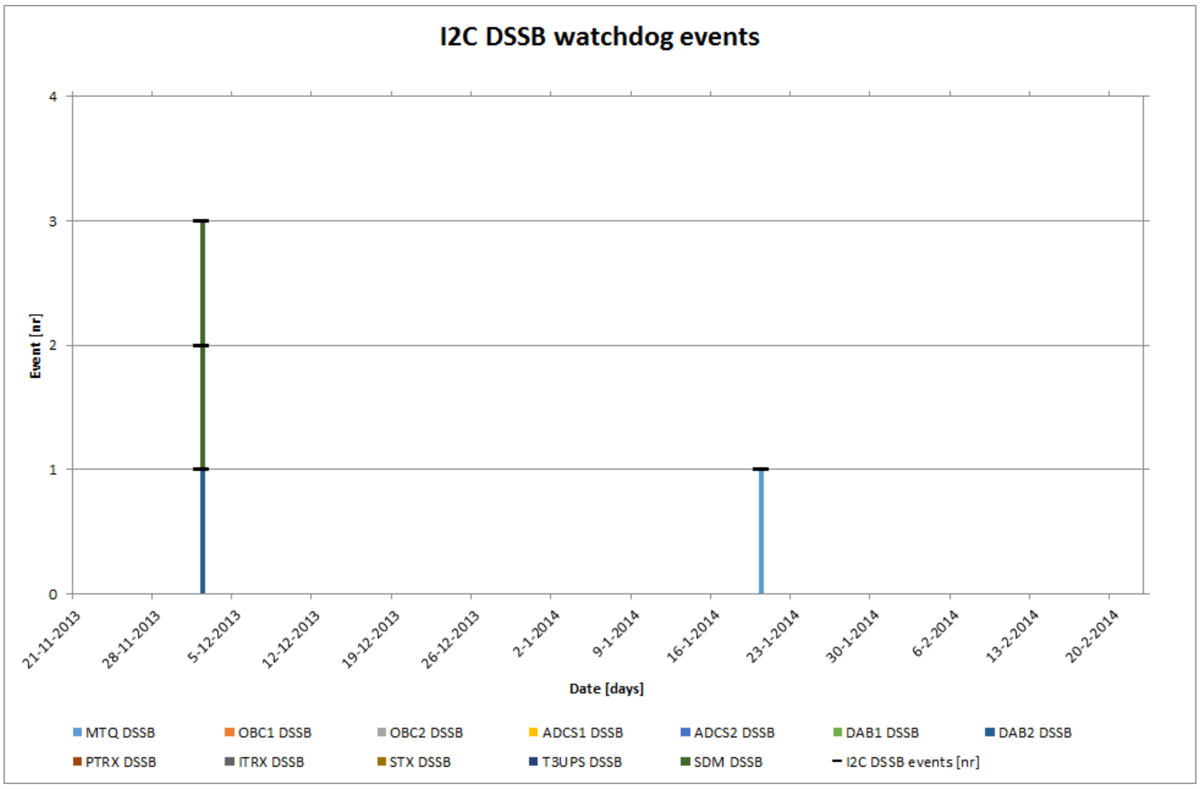


Figure 4.19: Frames indicating I2C internal watchdog event

Hypothesis H14:

Observation: J. Bouwmeester reported that the LT doesn't switch of when commanded to.
Observation: PTRX in transceiving mode after eclipse.
Condition: Delfi-n3Xt still functional upon leaving eclipse.
Question: What happened upon leaving eclipse?
Hypothesis: Simultaneous transmission occurred after leaving eclipse.

Hypothesis H15:

Observation: Loss of contact with Delfi-n3Xt.
Observation: Simultaneous transmission of PTRX and ITRX never verified/documented.
Condition: Delfi-n3Xt still functional upon leaving eclipse.
Condition: Simultaneous transmission of PTRX and ITRX.
Question: Why was contact lost with Delfi-n3Xt?
Hypothesis: Simultaneous transmission of the PTRX and ITRX leads to loss of external communications.

To determine whether simultaneous transmission has occurred during the mission, the transceiver telemetry parameters can be examined. The telemetry includes a parameter which provides the transmitter current. This current is zero when the transmitter is switched-off and above zero (around 140mA) when transmitting. Simultaneous transmission therefore occurs when both transceivers indicate a transmitter current. Figure 4.20 provides all telemetry packages indicating simultaneous transmission.

In total there are 270 frames indicating simultaneous transmission. Interestingly, this should not have occurred, since the INIT vector does not state that both transceivers should be transmitting. Figure 4.20 shows several singular events and repeated patterns. The repeated patterns are the most interesting as the simultaneous transmission occurs for a longer time interval. Table 4.2 provides the observations of the first and last frame indicating a repeated pattern.

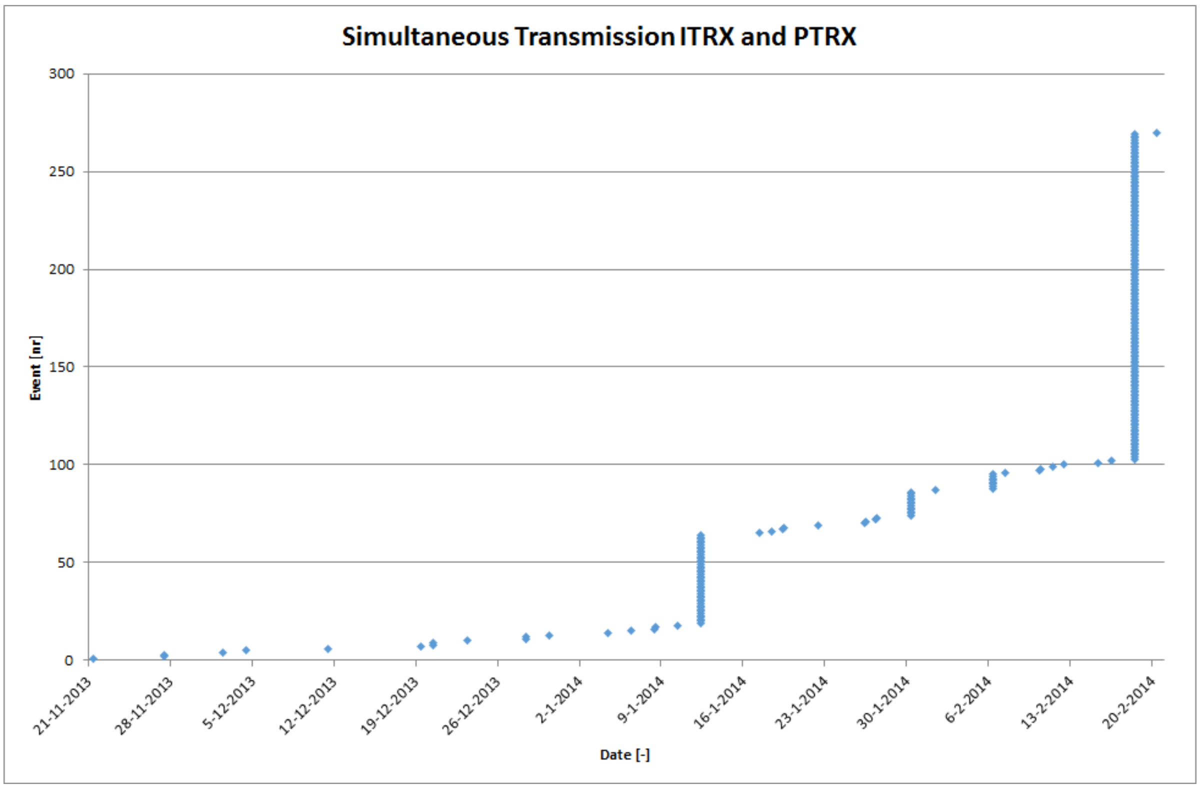


Figure 4.20: Frames indicating simultaneous transmission of PTRX and ITRX

Table 4.2: Observations simultaneous transmission

Anomaly start (UTC)	Anomaly end (UTC)	Observations first package	Observations last package
12-1-2014 10:50	12-1-2014 10:52	First package since reboot (10 sec ago)	Last received package of orbital period
30-1-2014 09:02	30-1-2014 09:08	First package since reboot (10 sec ago)	Last received package of orbital period
6-2-2014 10:15	6-2-2014 10:16	First package since reboot (50 sec ago)	Last received package of orbital period
18-2-2014 12:44	18-2-2014 12:54	First package since reboot and first package since eclipse	Last received package of orbital period

From Table 4.2, it is clear that simultaneous transmission occurs after a reboot (H16), and there is good reason to assume that simultaneous transmission is continued until eclipse (hypothesis H17), as a new INIT vector is loaded. This therefore decreases the likeliness of H15.

Hypothesis H16:

Observation: The first package indicating simultaneous transmission is also the first package since a OBC reboot.
Observation: The last package indicating simultaneous transmission is also the last received frame of the orbital period.
Question: What causes simultaneous transmission of ITRX and PTRX?
Hypothesis: Simultaneous transmission of the PTRX and ITRX can occur after a reboot.

Hypothesis H17:

Observation: The first package indicating simultaneous transmission is also the first package since a OBC reboot.
Observation: The last package indicating simultaneous transmission is also the last received package of the orbital period.
Question: What solves simultaneous transmission?
Hypothesis: Simultaneous transmission continues until a mode switch or reboot occurs.

4.4.11. Anomaly overview

Figure 4.21 provides an overview of all investigated anomalies. It provides the number of anomalies, when they occurred, and which kind of anomaly occurred. Comparing Figure 4.21 with figures 4.17, 4.19, 4.18, 4.16, 4.14, 4.13, 4.20, 4.15, 4.12 one can determine *when* an anomaly occurred, *the number of anomalies* reported by a single package, *which type* of anomaly occurred, and *which subsystem* was affected.

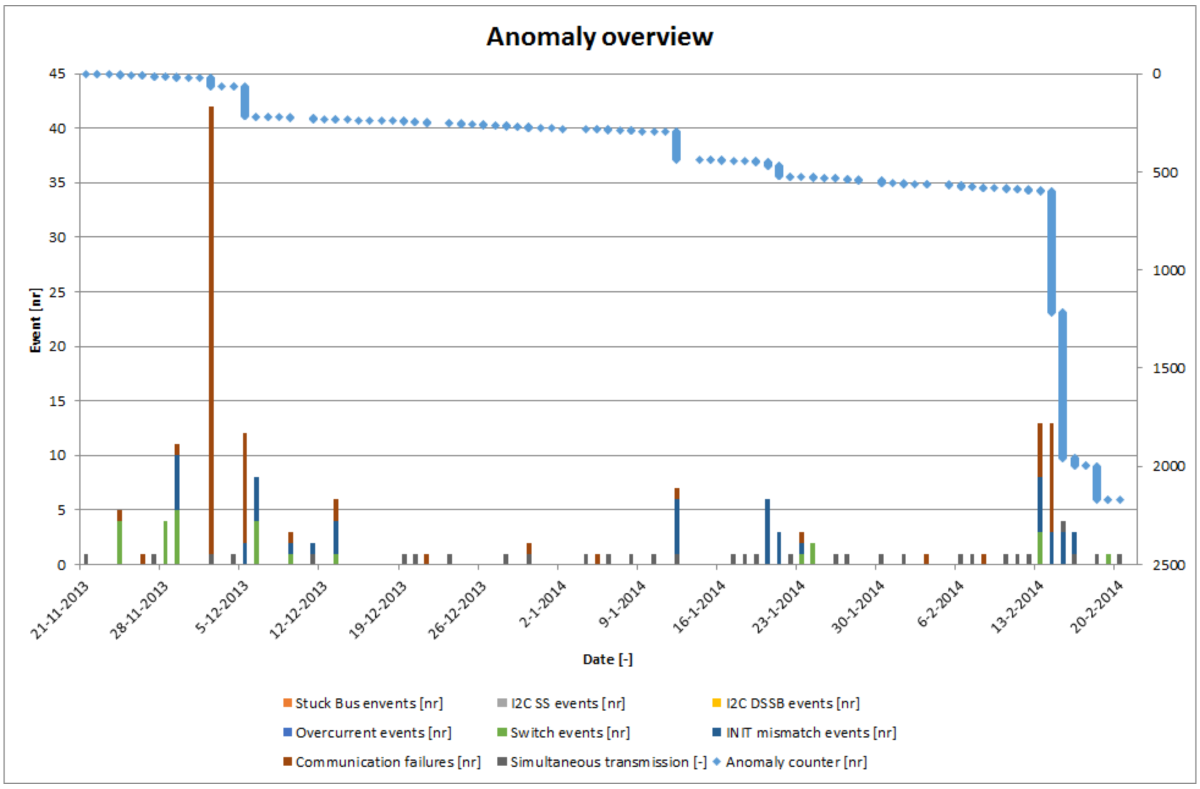


Figure 4.21: Anomaly overview

In total there are 2170 telemetry packages indicating an anomaly, which is displayed by the anomaly frame counter in Figure 4.21. Since all anomaly data is available, one should look for events which might be related to the loss of the satellite. Each anomaly might reveal information regarding the loss of the satellite, but the dataset is still very large. Therefore, repeating or consecutive patterns are examined in more detail, as the satellite is not able to recover autonomously. Note: by only looking for similarities in the anomalies and neglecting the differences the Texas sharpshooter fallacy[2] applies. This is however not a problem, since this method is only used to *identify* a hypothesis, and not for verification.

In Table 4.3 the various repeating patterns are provided. From Table 4.3 it is clear that most repeating anomalies occur after a command, reboot, or mode switch, and is solved by the switch logic, a reboot, or a mode switch. The various anomalies occurring at 2-12-2013 indicate that the switch counters perform as intended, except for the STX.

Table 4.3: Repeating anomalies/patterns

First package (UTC)	Last package (UTC)	Observations first package	Observations last package	Other remarks
2-12-2013 10:55	2-12-2013 10:57	Various anomalies occurring after reception of thrust command(0x04 0x3E 0x12 0x28 0x64 [29]*), started by increasing switch counters, Batteries DoD(65%, 50%, 61%, 49%)	Solved by switch logic (INIT vector changed: ADCS1 from ON to OFF, ADCS2 from OFF to ON, STX from beacon to OFF). Remaining switch counter value: DAB1 = 1, ADCS1 = 2.	Thrust campaign, switch counter STX is not increasing while INIT vector is changed. ADCS2 does not provide HK data. After CGG 04 ignition (0x04 0x3E 0x11 0x78 0x04) ADCS1 switches ON again and the switch counter resets.
5-12-2013 11:13	5-12-2013 11:19	Various anomalies occurring after reception of the wrong Command(0x01 0xEC 0x00 ~~0x00~~ 0x72 0x08 0x92 [29]). INIT vector changes (OBC1 from ON to OFF, OBC2 from BACKUP to UNKNOWN)	OBC2 switched all subsystems OFF, except for G-EPS1 and PTRX. Eventually OBC1 took over again.	The INIT vector changed many times while the switch counters did not increase.
12-1-2014 07:23	12-1-2014 09:05	First package since eclipse indicates $DSSB(MTQs)_{ON}$ while $INIT(MTQs)_{OFF}$.	Solved by reboot.	
12-1-2014 10:50	12-1-2014 10:52	Simultaneous transmission occurs after reboot	Last received package of orbital period	
20-1-2014 15:53	20-1-2014 15:56	First package since eclipse indicates $DSSB(MTQs)_{ON}$ while $INIT(MTQs)_{OFF}$.	Last received package of orbital period.	
21-1-2014 01:42	21-1-2014 01:48	First package since eclipse indicates $DSSB(MTQs)_{ON}$ while $INIT(MTQs)_{OFF}$	Last received package of orbital period.	
14-2-2014 10:12	14-2-2014 10:26	First package since eclipse indicates $DSSB(MTQs)_{ON}$ while $INIT(MTQs)_{OFF}$.	Last received package of orbital period.	
14-2-2014 11:50	14-2-2014 12:03	First package since eclipse indicates $DSSB(MTQs)_{ON}$ while $INIT(MTQs)_{OFF}$.	Last received package of orbital period.	
15-2-2014 09:12	15-2-2014 09:26	First package since eclipse indicates $DSSB(MTQs)_{ON}$ while $INIT(MTQs)_{OFF}$, $INIT(G-EPS2)_{ON}$ and $INIT(G-EPS1)_{OFF}$ seems unusual.	Last received package of orbital period	Should be $INIT(G-EPS1)_{ON}$
15-2-2014 10:52	15-2-2014 11:05	First package since eclipse indicates $DSSB(MTQs)_{ON}$ while $INIT(MTQs)_{OFF}$, Last executed telecommand parameter provides impossible value of 620000	Last received package of orbital period	Value should consist of 4 hexadecimals.
15-2-2014 12:30	15-2-2014 12:36	First package since eclipse indicates $DSSB(MTQs)_{ON}$ while $INIT(MTQs)_{OFF}$.	Last received package of orbital period.	
16-2-2014 00:03	16-2-2014 00:13	First package since eclipse indicates $DSSB(MTQs)_{ON}$ while $INIT(MTQs)_{OFF}$.	Last received package of orbital period.	
18-2-2014 12:44	18-2-2014 12:54	Simultaneous transmission occurs after reboot	Last received package of orbital period.	

The various anomalies that occurred at 5-12-2013 were caused by a invalid command. The transmitted command did not contain the leading hexadecimals (0x00) in the destination part of the command. The pass report can be found in Section C.1. Upon reception OBC2 took over control and OBC1 switched off completely. Interestingly, the INIT vector changed rapidly, switching many subsystems off, including the ITRX. G-EPS1 and G-EPS2 kept exchanging control until the end of the pass. It appears that the anomaly was solved upon entering or leaving eclipse, where a new INIT vector was loaded from the memory.

Table 4.3 provides a few repeating anomalies which are similar. In these cases the magnetorquers where switched *ON* while they should have been switched *OFF*. In all cases the first package indicating this anomaly is also the first received package of a pass and the last package indicating the anomaly is also the last package received before eclipse. It therefore appears that the anomaly *occurred due to* a mode switch, and was also *solved by* a mode switch. Apparently, the magnetometers were not correctly initialised.

Hypothesis H18:

Observation: Magnetorquers are switched ON while they should be switched OFF.
Observation: First package since eclipse indicates anomaly.
Question: Why were the magnetorquers switched ON?
Hypothesis: The magnetorquers are not correctly initialised due to a software bug.

Unfortunately, the detected anomalies do not directly lead to a hypothesis that might explain the loss of the satellite. Therefore the last moment of contact is analysed in Section 4.5.

4.5. Last-contact

On the last day of contact, 20-02-2014 UTC, there were two passes over the Delft ground station which were actively monitored/commanded (based on telemetry and mailbox[21][29]). A time-line can be found in Figure 4.22.

- Pass 1: 09:07 to 09:18 UTC
  - Various attempts to fire the igniter of the T3UPS thruster (unsuccessful)
  - No anomalies detected
  - Active subsystems: OBC1, OBC2(backup), G-EPS1, DAB1, ITRX(receive), PTRX(transceive), $\mathrm{T}^3\mu PS$, SDM
- Pass 2: 10:43 to 10:53 UTC
  - 10:44: Activation of LT using volatile command(0x02 0xEC 0x00 0x00 0x72 0x08 0xD0). This command changes the INIT vector where all subsystems are switched OFF except OBC1, OBC2(backup), G-EPS1, ITRX(LT), PTRX(receiving).
  - Beacon detected, but no reception of relay data.
  - 10:52: De-activation of LT using volatile command(0x02 0xEC 0x00 0x00 0x73 0x08 0x90). This command changes the INIT vector where all subsystems are switched OFF except OBC1, OBC2(backup), G-EPS1, ITRX(transcieving), PTRX(receiving).
  - 10:53: Activation of LT using volatile Command(0x02 0xEC 0x0 00x00 0x72 0x08 0xD0).[1]
  - Beacon detected, but no reception of relay data. Beacon detected until end of pass, which is also the last contact with Delfi-n3Xt.

The INIT vector should have changed upon entering and exiting eclipse. To determine the change in INIT vector upon a mode switch, the last parameter uploads are determined. The last time the eclipse mode INIT vector was changed using a non-volatile command was on 05-12-2013 at 12:51 UTC[21]. The last non-volatile parameter upload of the main mode INIT vector occurred at 23-01-2014 at 09:35 UTC[21]. Therefore the next mode changes *should* have occurred:

- Main mode: Volatile INIT vector **(0x02 0xEC 0x00 0x00 0x72 0x08 0xD0)**: All subsystems OFF except: OBC1, OBC2(backup), G-EPS1, ITRX(LT), PTRX(receiving)

[1]From the midterm meeting at 09-02-2016 it became clear that it was unsure whether the command was volatile or non-volatile.

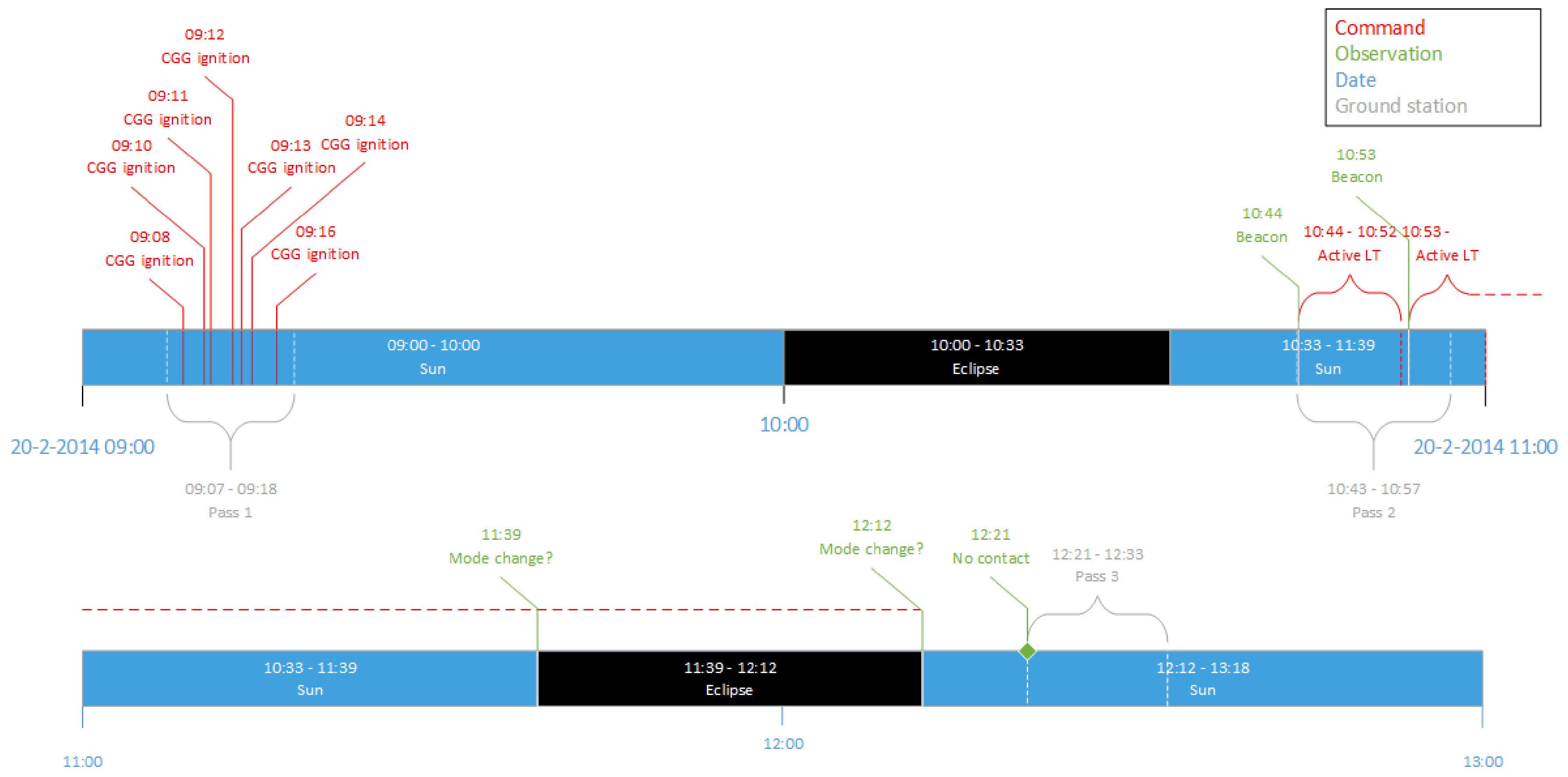


Figure 4.22: Time-line 20-02-2014: Last contact, based on TLE epoch 20-02-2014 18:27:42

- Eclipse: Non-volatile INIT vector **(0x01 0xEE 0x00 ~~0x00~~ 0x72 0x08 0x10)**[29]: All subsystems OFF except: OBC1, OBC2(backup), G-EPS1, PTRX(receiving)
- Main mode: Non-volatile INIT vector **(0x01 0xEC 0x00 0x00 0x73 0x0A 0x63)**[21] All subsystems OFF except: OBC1, OBC2(backup), ADCS1, G-EPS1, DAB1, ITRX(receiving), PTRX(transceiving), $\mathrm{T}^3\mu PS$, SDM

Interestingly, the eclipse mode vector was changed using a invalid command similar to the command causing the anomalies reported on 05-12-2013 (destination parameter is missing leading hexadecimals 0x00)[29]. The actual pass report can be found in Section C.2. Therefore, similar behaviour is expected to have occurred during each eclipse since that day. On 09-02-2016 however, during the midterm review, it was argued that the e-mail stating this error possibly contains a typo.

Hypothesis H19:

Observation: The operational logs state that the last non-volatile parameter update of the eclipse mode, transmitted on 05-12-2013, lacked leading hexadecimals 0x00 in the value/content parameter.
Observation: Many anomalies occurred after transmission of a volatile parameter update of the current mode which lacked leading hexadecimals 0x00 in the value/content parameter.
Question: What happened in eclipse since the last non-volatile parameter update of the eclipse mode?
Hypothesis: Since 05-12-2013 many anomalies occurred during eclipse.

As previously mentioned in Section 4.4.10 it was found by J. Bouwmeester that the LT didn't switch off when commanded to. This *did* occur in orbit however. Since it is unknown what happened in eclipse due to the possible invalid command, in combination with a possibly not responding LT, it is basically impossible to reveal what occurred during eclipse, and whether the combination of both is related to the loss of the satellite.

Hypothesis H20:

Observation: Loss of contact.
Condition: The last non-volatile parameter update of the eclipse mode, transmitted on 05-12-2013, lacked leading hexadecimals 0x00 in the value/content parameter.
Condition: The LT did not shut down upon entering eclipse.
Question: What caused the loss of contact with Delfi-n3Xt?
Hypothesis: The combination of the incomplete eclipse mode INIT vector and and active LT lead to loss of contact.

From the mode changes that *should* have occurred, and the fact that contact was lost at the next ground station pass, several hypotheses regarding the time of failure can be created. It might be that one of active subsystems initiated a catastrophic event during the sun-lit part of the orbit. Otherwise the failure occurred upon mode switch towards eclipse mode, during eclipse, or upon the last mode switch.

Hypothesis H21:

Observation: Last contact at 20-02-2014 10:57 UTC.
Observation: No contact at 20-02-2014 12:21 UTC.
Question: When did Delfi-n3Xt fail?
Hypothesis: Delfi-n3Xt failed in the sun-lit part of the orbit, between 20-02-2014 10:57 and 11:39 UTC.

Hypothesis H22:

Observation: Last contact at 20-02-2014 10:57 UTC.
Observation: No contact at 20-02-2014 12:21 UTC.
Question: When did Delfi-n3Xt fail?
Hypothesis: Delfi-n3Xt failed upon mode switch towards eclipse mode, at 20-02-2014 11:39 UTC.

Hypothesis H23:

Observation: Last contact at 20-02-2014 10:57 UTC.
Observation: No contact at 20-02-2014 12:21 UTC.
Question: When did Delfi-n3Xt fail?
Hypothesis: Delfi-n3Xt failed during eclipse, between 20-02-2014 11:39 and 12:12 UTC.

Hypothesis H24:

Observation: Last contact at 20-02-2014 10:57 UTC.
Observation: No contact at 20-02-2014 12:21 UTC.
Question: When did Delfi-n3Xt fail?
Hypothesis: Delfi-n3Xt failed upon mode switch towards main mode, at 20-02-2014 12:12 UTC.

4.5.1. Rotational rate

A telemetry package provides over 400 parameters, as introduced in Section 4.2.1, including attitude parameters. Examining the rotational rate during the last pass might reveal an anomaly. Unfortunately, the attitude was never computed on-board Delfi-n3Xt, because the ADCS never entered an advanced mode. Therefore the attitude parameters are empty. The telemetry does however provide the magnetic- and sun vector determined by the 3-axis magnetometer and the six sun sensors. According to M. Vos[33], the vectors are expressed in the Satellite Body Frame (SBF)-frame, which has its origin located at the Center of Gravity (COG) and the axes aligned with the principle axes of inertia. The normalized vectors during the last day of contact can be seen in Figure 4.23 and 4.24. Figure 4.23 indicates the vectors during the first pass as described in Figure 4.22, and Figure 4.24 describes the vectors during the second pass. In both figures '$T = 0$ indicates the begin of telemetry reception at the DGS. In the second pass only 20 telemetry packages were received as the linear transponder was activated twice.

From both figures it can be seen that the vectors are not rapidly changing and therefore computation of the actual attitude and rotational rate are not of interest. The likeliness of a high rotational rate leading to excessive internal forces, defined as hypothesis H25, is therefore low.

Hypothesis H25:

Observation: Attitude and rotational rate of Delfi-n3Xt are unknown because Delfi-n3Xt didn't enter advanced attitude modes.
Question: Why was contact lost?
Hypothesis: Contact was lost as the rotational rate was too high causing excessive internal forces.

4.5.2. Audio recordings

On 09-02-2016 the midterm meeting took place at the faculty of Aerospace Engineering. The purpose of this meeting was to review the project progress and to determine the heading for continuation of the forensic investigation. The project status was presented to the group and during this meeting various opinions and experiences were shared and discussed. The attendees were:

- **J. Bouwmeester:** Delft University of Technology - Project manager Delfi-n3Xt

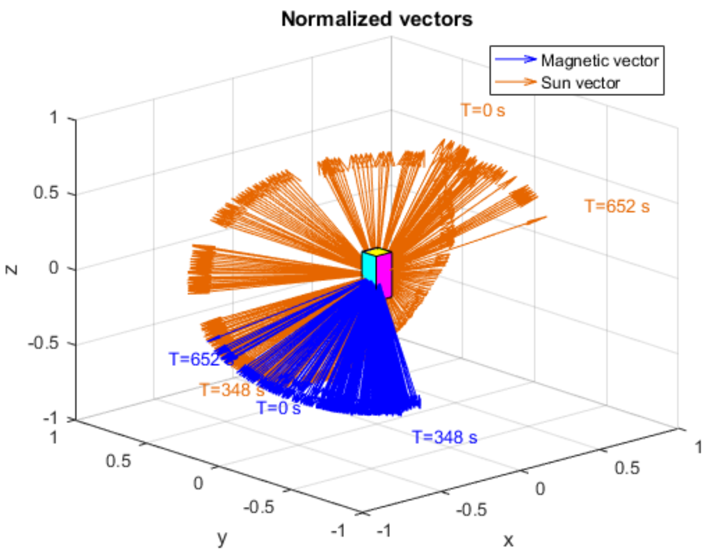


Figure 4.23: Attitude vectors pass 1

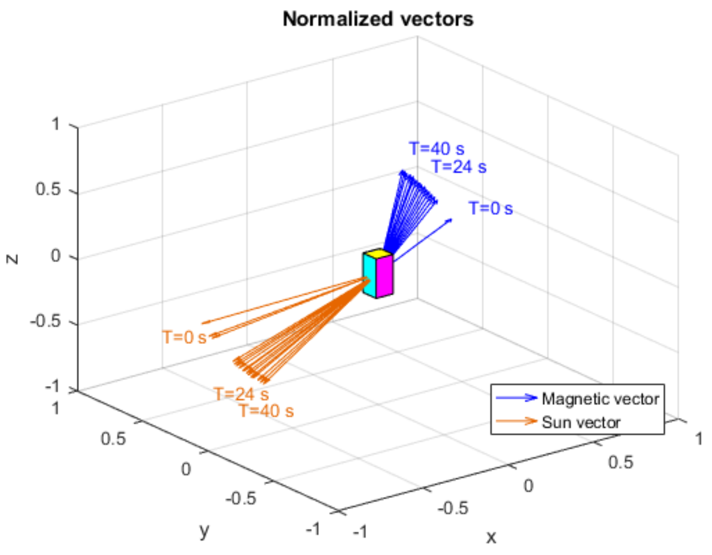


Figure 4.24: Attitude vectors pass 2

- **W.J. Weggelaar:** Innovative Solutions In Space - Radio Frequency (RF) expert
- **S. Speretta:** Delft University of Technology - RF expert
- **N. von Storch:** Delft University of Technology - Operator Delfi-n3Xt
- **E.P. Smit:** T-minus engineering - Implementation electronics Delfi-n3Xt
- **P. Beckers:** Delft University of Technology - DSSB design Delfi-n3Xt
- **C.J.M Verhoeven:** Delft University of Technology - Electrical engineering expert
- **T.G.Watts:** Delft University of Technology - Thesis supervisor
- **P.P. Sundaramoorthy:** Delft University of Technology - Thesis supervisor
- **S. van Kuijk:** Delft University of Technology - Author

During the midterm meeting the audio recordings of the last pass were presented and it became clear that the ITRX showed strange behaviour between the transponder tests. It was therefore suggested by S. Speretta and W.J. Weggelaar to analyse the signal in more detail.

Introduction to audio recordings
The ITRX has several modes:

- Idle mode, flags off (default): Transmits frames when buffer is full, flags are inactive. (ITRX can't be detected when not transmitting frames)
- Idle mode, flags on: Transmits frames when the frame buffer is full, flags are active. (ITRX can be detected when not transmitting frames)
- Transponder mode: Continuous Wave (CW) beacon is active while relaying telemetry.

Since the ITRX and PTRX don't distinguish between transmitting or receiving mode, the OBC determines which transceiver is used to transmit frames. When the OBC selects the ITRX for frame transmission, the flags are switched on for the ITRX and switched off for the PTRX.

To transmit a telemetry frame the ITRX uses BPSK modulation and the AX.25 protocol for package definition. A BPSK modulated signal uses phase-shifts to distinct between a 1 and a 0. The Ax.25 protocol states that telemetry frames are separated by 'flags'. The ITRX uses a minimum of 10 flags between transmitted frames and the bitrate is 2400 bits/sec. A single flag consists of the following Byte: 01111110. This flag can be easily recognised in the BPSK signal, as bit stuffing is used, where the maximum number of consecutive ones is five. Therefore a flag is a unique Byte in the signal.

Analysis
In Figure 4.25 the signal is visualised as a waterfall of the amplitude spectrum using HDSDR.

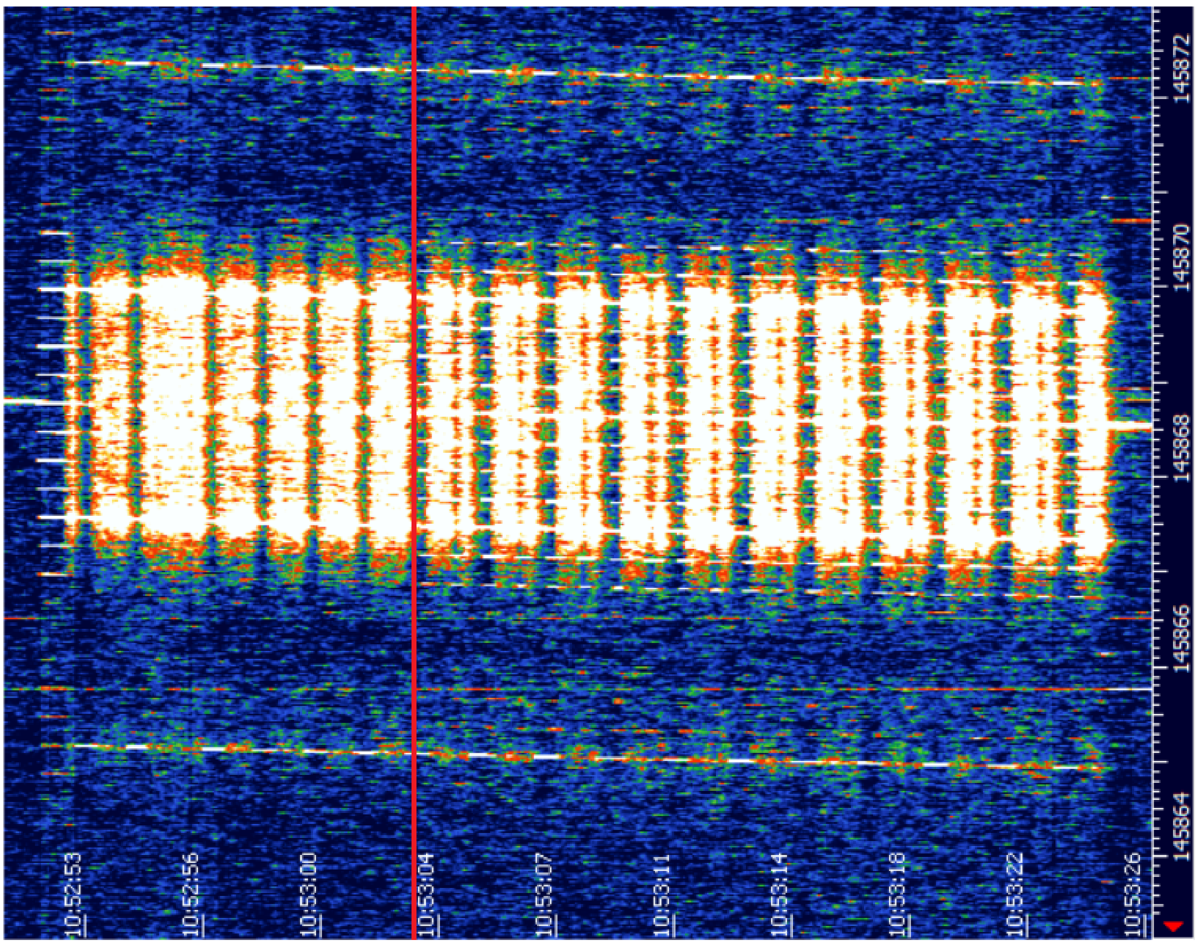


Figure 4.25: Waterfall of ITRX between transponder tests.

Using Figure 4.25 several observations can be made. Firstly a shift in frequency can be seen. This frequency shift is a result of the Doppler effect as the satellite is moving with respect to the DGS. One can also observe images of the transmitted signal. At the beginning and at the end of the time interval, the CW beacon can be observed, which has a very small bandwidth. At 10:52:53 the flags can be observed, followed by the telemetry frames. It appears that something changed in transmission around the red line indicator, as from then one can observe flags between the transmitted packages.

As the waterfall indicates that flags appear to be missing in the first transmitted frames, identification of the flags in the BPSK signal might reveal useful information. The flag intervals are manually determined by locating the first and last flag. An example is provided in Figure 4.26. The flag intervals are plotted in Figure 4.27. The leading- and trailing flag indicators are also used to plot the transmission time of a single frame. This can be found in Figure 4.28. By comparing either the leading flag indicators, or the trailing flag indicators, the transmission frequency/period can be determined. These periods can be found in Figure 4.29. In these figures theoretical values are indicated. These values are computed using Equation 4.1. Both transmitted telemetry frames consist of 224 Bytes with a bitrate of 2400 bits/sec. Beyond that Delfi-n3Xt transmits frame every second.

$$Interval = \frac{bits}{bitrate} \tag{4.1}$$

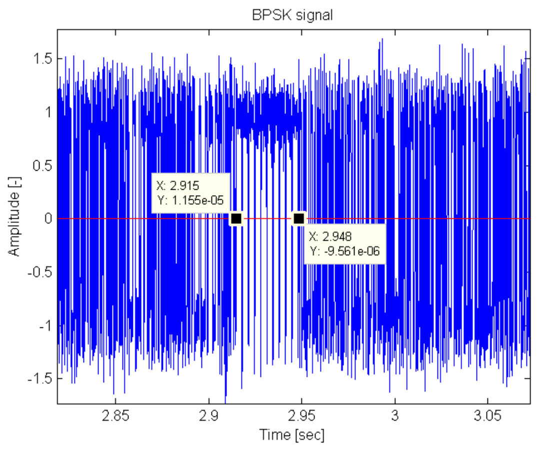


Figure 4.26: BPSK: Example of identification of AX.25 flags.

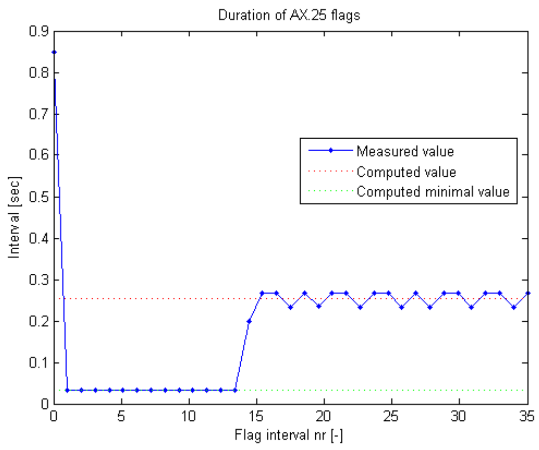


Figure 4.27: Flag intervals between transponder tests.

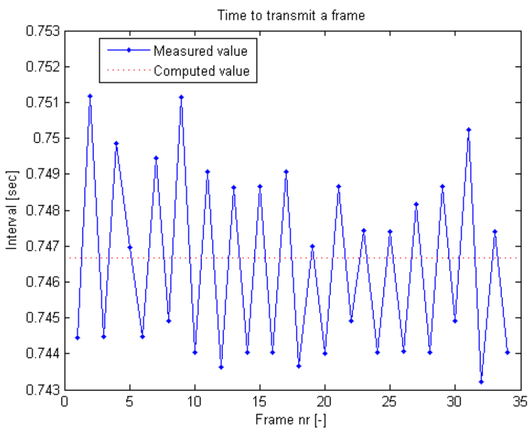


Figure 4.28: Time to transmit a frame.

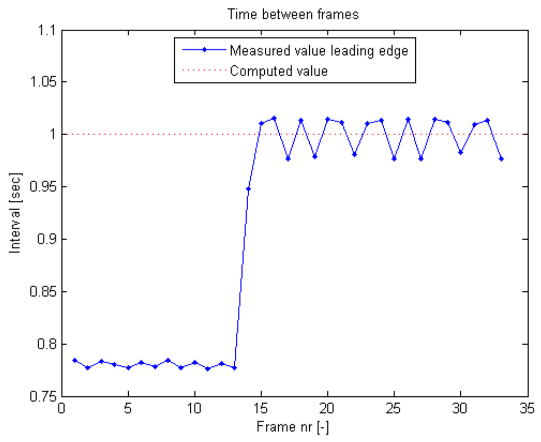


Figure 4.29: Time between frames.

Figure 4.27 indicates that flags were detected between the transmitted frames. The first flag-interval has a long duration (0.85 sec). This interval is followed by 13 intervals of similar duration (0.033 sec). Then the flag-interval increases in duration and jumps between 0.23 sec and 0.26 sec. The figure includes the theoretical value and the minimum flag interval. This is computed with Equation 4.1.

The first interval corresponds with Figure 4.25, where the transceiver switches from transponder mode to idle mode. Afterwards the waterfall is less readable since the flag intervals are very short, and therefore appear to be missing. In Figure 4.25 the red line indicator corresponds to the jump in flag interval, as the time between frames increases and flags can be observed.

It is clear from the figures that the timing sequence has changed. 13 Frames were transmitted at a higher frequency, by decreasing the flag interval duration. Possible explanations are stated in hypotheses H26 and H27.

Hypothesis H26

Observation: Change in timing sequence of telemetry transmission in-between the LT-experiments.
Observation: Minimal interval duration of flag transmission matches the theoretical minimal interval duration.
Question: What caused the change in timing sequence?
Hypothesis: The OBC clock frequency changed.

Hypothesis H27

Observation: Change in timing sequence of telemetry transmission in-between the LT-experiments.
Observation: Minimal interval duration of flag transmission matches the theoretical minimal interval duration.
Question: What caused the change in timing sequence?
Hypothesis: The buffer of the ITRX was full.

Another interesting observation is the fact that the time to transmit a single frame indicates jumps in duration. As stated in 4.2.1, two different telemetry frames are transmitted. These telemetry frames have the same message length in Bytes and therefore the time-to-transmit *should* be equal. Possible hypotheses are stated in H28 and H29,

Hypothesis H28

Observation: Time to transmit a frame jumps.
Question: What caused these jumps?
Hypothesis: The various interval jumps are observed due to the difference between the two telemetry frames.

Hypothesis H29

Observation: Time to transmit a frame jumps.
Question: What caused these jumps?
Hypothesis: The various interval jumps are observed as wrong measurement points were selected.

The last observation is an anomaly in 4.28. At frame number five one would expect a lower value which corresponds with the 'bouncing trend'. It appears to be a phase change, but the reason is unclear. It is not further investigated as it is not considered to be important/related to the loss of the satellite.

5 Review of design of the ITRX and LT

In Section 4.5 several hypotheses are stated regarding the time of failure. Since the LT was not used before, and the LT has a low TRL, it is likely that the LT or ITRX caused a problem. Therefore, this chapter examines the design of the LT and it's interfaces.

5.1. ITRX and LT

An overview of the design of the ITRX and LT can be seen in Figure 5.1 and the I2C and power interfaces are shown in Figure 5.2. From Figure 5.1 it is observed that the LT connects the receiver with the transmitter and that it is enabled by the ISIS transceiver Transmitter Controller (ITC). The LT is therefore able to relay incoming signals when activated.

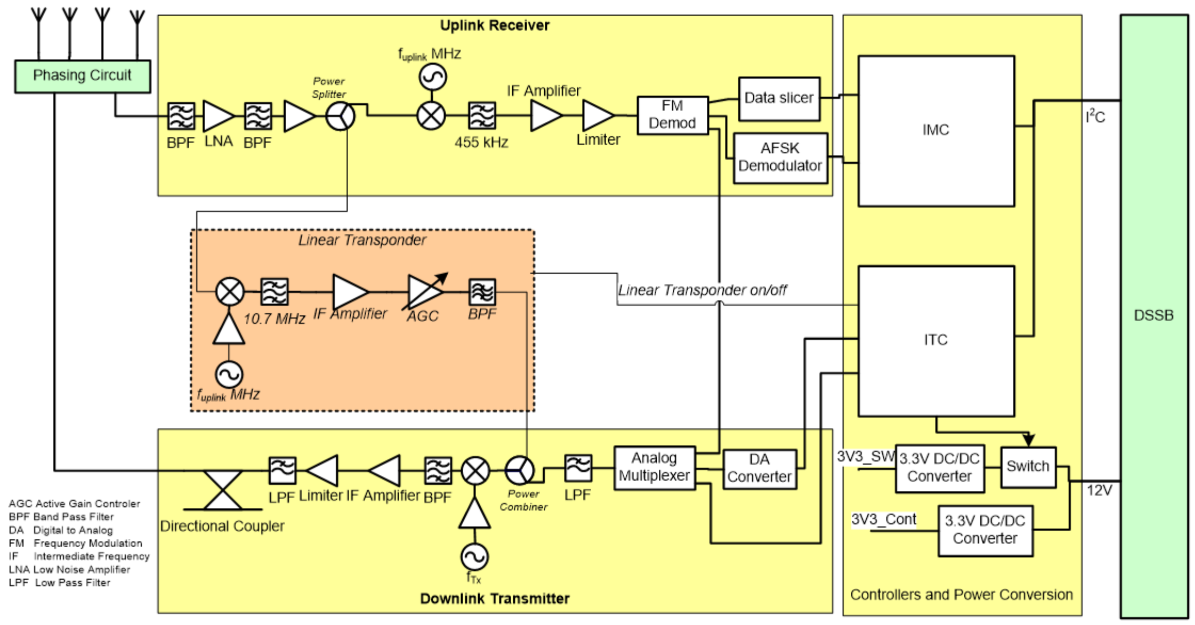


Figure 5.1: Simplified electrical diagram ITRX and LT

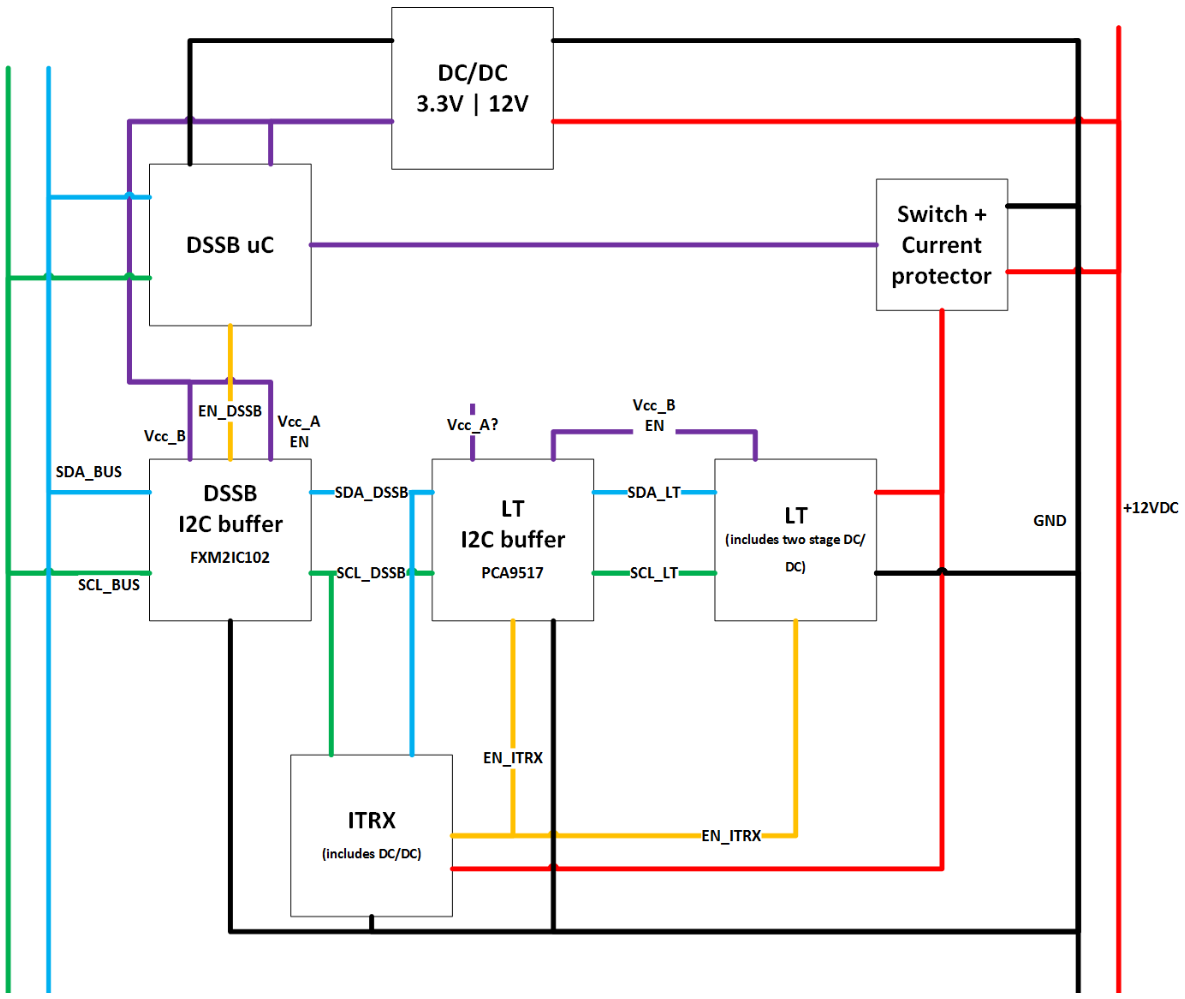


Figure 5.2: I2C interfaces between DSSB,ITRX and LT. V_{ccB} EN indicates the pull-up resistor to the $Enable$ signal.

From the Figure 5.2 , one can clearly see that multiple I2C buffers are used and that one reference voltage source is unclear as the electrical diagram of the ITRX is unavailable at the time of writing. More importantly is the fact that the ITRX controls the $Enable$ signal to the LT and it's buffer.

The $Enable$ signal is connected tot he V_{ccB} side using a pull-up resistor, while it should be connected to the V_{ccA} side of the buffer. In the current design, the V_{ccB} side is able to pull-up the $Enable$ signal which connects the LT to the I2C bus, creating a dangerous situation in case the LT fails. This $Enable$ is also connected to the DC/DC converters located on the LT, which include capacitors with high capacity. This could therefore lead to an infinite loop where the ITC of the ITRX is not able to shut down the LT. This would therefore support the observations performed by J.Bouwmeester and the hypothesis H13, discussed in Section 4.4.10.

As explained in Sections 4.4.7 and 4.4.8 the watchdog is triggered in case there are communication problems over the internal I2C bus and Figure 3.14 shows that the OBC enters recovery mode. One would therefore expect that the ITRX is disconnected from the I2C bus in case the I2C bus gets stuck. By default however, the ITRX is switched on, which therefore could create an infinite loop while locking the I2C bus, leading to loss of the satellite.

Hypothesis H30:

Observation: LT not formally verified.
Observation: Active LT.
Question: What could be the consequence of using unverified hardware?
Hypothesis: I2C bus locked due to failure at the LT.

Hypothesis H31:

Observation: The PCA9517 located on the LT is oriented incorrectly.
Observation: High capacity capacitors located on the DC/DC converters at the LT.
Condition: Hypothesis H13: A design or implementation flaw prevents the LT from shutting down.
Condition: Hypothesis H25: I2C bus stuck due to failure of the LT.
Question: Why was Delfi-n3Xt lost?
Hypothesis: Delfi-n3Xt was lost as the I2C bus was locked.

6 Hypothesis overview and selection

This Chapter provides an overview of all hypotheses previously identified. It is expected that verification of all hypotheses will take longer than the schedule allows, since there are many hypotheses, and therefore a selection is made. Section 6.1 describes the selection method and Section 6.2 provides the application of that method. The last section, Section 6.3 the hypotheses are grouped per subject.

6.1. Selection method

There are several trade-off methods available which can be used to select hypotheses for verification. The classical trade-off table and the Analytic Hierarchy Process (AHP) method are commonly used. In this report however, a method similar to a risk assessment is used. The highest risks are identified and selected for verification. This method is preferred over the AHP method for three reasons:

- A hypothesis has many similarities with respect to risk (likeliness/impact).
- A hypothesis map provides a clear overview.
- AHP requires pairwise comparison of all hypotheses. Since there are many hypotheses (31) this requires a lot of work.

The downside of the selected method is the fact that selection can be biased. The method is similar to the application of a classical trade-off where the criteria are likeliness and consequence with equal weigths, since risk is defined as the product of likeliness of occurrence and severity of consequence [5]. Another downside is the fact that 'complexity of verification campaign' (the availability of Ground Support Equipment (GSE), documentation, expertise) is not taken into account and therefore the verification campaign might consume more time than initially planned.

The selection method consists of selecting the hypothesis with the highest risk, or priority. In this report the following definitions are used:

- "Priority is the product of likeliness and impact or likeliness and importance."
- "Importance is a measure of the potential to gain new knowledge upon verification of the hypothesis."
- "Impact is a measure for the relation between the hypothesis and the loss of contact."
- "Likeliness is a measure for the chance of verifying the hypothesis."

In Table 6.1 the criteria scale is explained. As can be seen, a scale from 1 to 5 is used. In Section 6.2 the selection is performed.

Table 6.1: Hypothesis criteria

Score	Likeliness	Importance	Impact
0	Rejected	x	x
1	Very unlikely	Not important: Doesn't reveal useful formation	Negligible: Not related to loss of contact
2	Unlikely	Not so important: Doesn't reveal much information	Significant: Barely related to loss of contact
3	Moderately likely	Moderately important: Reveals moderately important information	Major: Indirectly related to loss of contact
4	Likely	Important: Reveals important information	Critical: Related to loss of contact
5	Very likely	Very important: Reveals critical information	Catastrophic: Directly related to loss of contact
6	Verified	x	x

6.2. Hypothesis assessment

Table 6.2 shows the hypothesis ID, the hypothesis description, the importance/impact and the likeliness. The table includes a short elaboration of the assigned values. For the likeliness and importance, the scale goes from 1 to 5, where high value is considered a high likeliness or importance. The values are based on experience from the author and the work described in previous chapters. The assigned values are possibly biased. The author was however, not involved in the Delfi-n3Xt project and therefore has no personal attachments to any of the hypotheses or subsystems.

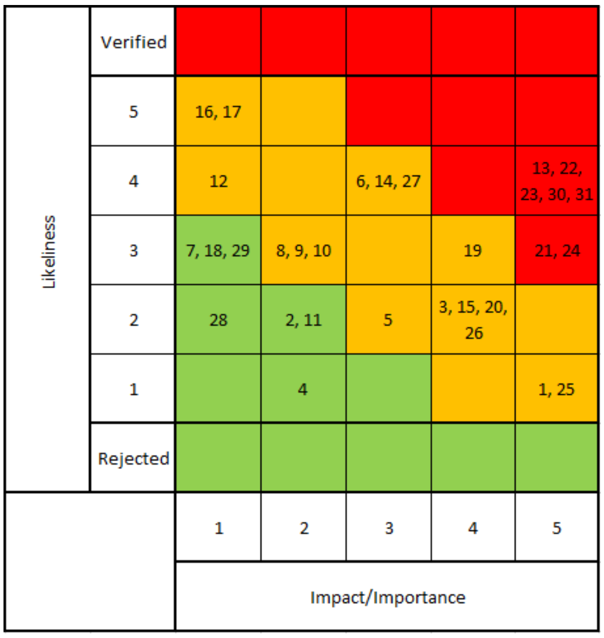
Table 6.2: Hypothesis overview

H1	Description	x	Delfi-n3Xt was lost due to an in-orbit collision with an object.
	Likeliness	1	Orbital elements unaffected.
	Impact	5	Impact is expected to destroy satellite
H2	Description		The reboots experienced by Delfi-n3xt were caused by radiation.
	Likeliness	2	No relation found between SAA and location of reboot, see Section 4.4.2.
	Importance	2	No direct connection to loss of satellite found.
H3	Description		The reboots experienced by Delfi-n3xt were caused by I2C lock-ups.
	Likeliness	2	Stuck bus counter constant, see Sections 4.4.2 and 4.4.3.
	Impact	4	I2C lock-ups might result in infinite loops.
H4	Description		The quasi-linear reboot behaviour of Delfi-n3xt was caused by a parameter overflow.
	Likeliness	1	Frame counter varies significantly between reboots, see Section 4.4.2
	Impact	2	No direct connection to loss of satellite found.
H5	Description		The I2C bus did not get stuck since deployment mode.
	Likeliness	2	Many reboots occurred, with unknown reasons, see Sections 4.4.2 and 4.4.3.
	Importance	3	Upon verification knowledge is gained regarding the performance of the I2C bus and source of reboots.
H6	Description		The stuck bus counter is constant because the counter is not working as intended.
	Likeliness	4	Stuck bus counter never verified.
	Importance	3	Upon verification knowledge is gained regarding the performance of the I2C bus and possibly the source of reboots.
H7	Description		239 stuck bus events occurred during the delay or deployment mode.
	Likeliness	3	First telemetry received indicates this value.
	Importance	1	Satellite was correctly deployed, so it is not considered important.
H8	Description		A software bug prevented the STX from rebooting.
	Likeliness	3	Several cases were found where the switch counters didn't reset to 0, see Section 4.4.4

	Impact	2	No mission critical subsystem.
H9	Description		A bit flip caused a switch counter value outside the telemetry range.
	Likeliness	3	Only the STX was affected, see Section 4.4.4
	Impact	2	No mission critical subsystem.
H10	Description		The switch counter of the STX did not reset and was reported incorrectly by telemetry.
	Likeliness	3	Considered equally likely as H8 and H9.
	Impact	2	No mission critical subsystem.
H11	Description		I2C watchdog events are triggered simultaneously because the software wasn't correctly implemented.
	Likeliness	2	Design specification describes that there should be a time delay. [19]
	Importance	2	Not considered to be important as the subsystems did shut-down as intended.
H12	Description		During the operations of Delfi-n3Xt no over-current events occurred.
	Likeliness	4	No over current events reported, although telemetry reception was discontinuous.
	Importance	1	Not considered important to verify as there are no reasons to believe otherwise.
H13	Description		A design or implementation flaw prevents the LT from shutting down.
	Likeliness	4	J. Bouwmeester performed a test (not reported) where the LT did not shut down, see Section 4.4.10.
	Impact	5	The LT was active when contact was lost and thus impact is high.
H14	Description		Simultaneous transmission occurred after leaving eclipse.
	Likeliness	4	J. Bouwmeester performed a test (not reported) where the LT did not shut down, see Section 4.4.10.
	Importance	3	Considered important since it is expected that simultaneous transmission leads to loss of contact (H15).
H15	Description		Simultaneous transmission of the PTRX and ITRX leads to loss of external communications.
	Likeliness	2	Simultaneous transmission occurred during operations (unintentionally), see Section 4.4.10. However, the combination of ITRX(transponder) and PTRX(transceiver) was never tested/reported.
	Impact	4	Possibly leads to loss of contact since both transceivers use the same phasing circuit/antenna's.
H16	Description		Simultaneous transmission of the PTRX and ITRX can occur after a reboot.
	Likeliness	5	First package indicating simultaneous transmission is also the first frame indicating a reboot.
	Importance	1	Not considered to be important as only little knowledge is gained upon verification.
H17	Description		Simultaneous transmission continues until a mode switch or reboot occurs.
	Likeliness	5	Last frame indicating simultaneous transmission is also the last received frame before eclipse.
	Importance	1	Not considered to be important as only little knowledge is gained upon verification.
H18	Description		The magnetorquers are not correctly initialised due to a software bug.
	Likeliness	3	DSSB circuit is unique for the magnetorquers.
	Impact	1	No link to loss of contact found.
H19	Description		Since 05-12-2013 many anomalies occurred during eclipse.
	Likeliness	3	Last NV parameter update was performed using an incomplete destination parameter, although this could be a typo. See Section 4.5
	Impact	4	Many anomalies occurred after reception of invalid command.
H20	Description		The combination of the invalid command and and active LT lead to loss of contact.
	Likeliness	2	Unlikely since both conditions should apply.
	Impact	4	Considered high since this scenario didn't occur before.
H21	Description		Delfi-n3Xt failed in the sun-lit part of the orbit, between 20-02-2014 10:57 and 11:39 UTC.

	Likeliness	3	Satellite was lost between 10:57 and 12:21 UTC and no reboots occurred yet that day (avg 2 reboots/day), see Section 4.4.2.
	Importance	5	Directly related to loss of contact.
H22	Description		Delfi-n3Xt failed upon mode switch towards eclipse mode, at 20-02-2014 11:39 UTC.
	Likeliness	4	At mode switch subsystems are shut-down and possibly an invalid command is executed.
	Importance	5	Directly related to loss of contact.
H23	Description		Delfi-n3Xt failed during eclipse, between 20-02-2014 11:39 and 12:12 UTC.
	Likeliness	4	Satellite was lost between 10:57 and 12:21 UTC and no reboots occurred yet that day (avg 2 reboots/day). Possibly many anomalies occurring due to invalid command.
	Importance	5	Directly related to loss of contact.
H24	Description		Delfi-n3Xt failed upon mode switch towards main mode, at 20-02-2014 12:12 UTC.
	Likeliness	3	Satellite was lost between 10:57 and 12:21 UTC.
	Importance	5	Directly related to loss of contact.
H25	Description		Contact was lost as the rotational rate was too high causing excessive internal forces.
	Likeliness	1	The magnetic- and sun vectors were not rapidly changing.
	Impact	5	Directly related to loss of contact.
H26	Description		The OBC clock frequency changed.
	Likeliness	2	No sudden change in temperature or bus voltage found, so it's not considered likely.
	Impact	4	OBC clock frequency affects all subsystems through I2C communication.
H27	Description		The buffer of the ITRX was full.
	Likeliness	4	OBC still provides telemetry frames to buffer while in LT mode.
	Importance	3	Rejects H25 and therefore considered important.
H28	Description		The various interval jumps are observed due to the difference between the two telemetry frames.
	Likeliness	2	Both telemetry frames are equal in size. [19]
	Importance	1	Not considered important
H29	Description		The various interval jumps are observed as wrong measurement points were selected.
	Likeliness	3	Selecting the wrong measurements points induces a systematic error.
	Importance	1	Not considered important as it has very little impact on conclusions.
H30	Description		I2C bus locked due to failure at the LT.
	Likeliness	4	Unverified subsystem and incorrect implementation of I2C buffer.
	Impact	5	Directly related to loss of contact.
H31	Description		Delfi-n3Xt was lost as the I2C bus was locked.
	Likeliness	4	Likely because all subsystems are affected and infinite loops can occur due to I2C recovery procedure.
	Impact	5	Directly related to loss of contact.

All hypotheses are displayed in a hypothesis map, which is similar to a risk map. This hypothesis map can be found in Figure 6.1. The figure shows that seven hypotheses have a high priority; H13, H21, H22, H23, H24, H30 and H31. Four of these hypotheses, H21, H22, H23 and H24 consider the time of failure which is impossible to determine without knowing what happened. Therefore these hypotheses are not selected for verification. Hypotheses H13, H30 and H31 however are selected for verification, as well as H6, H27, H19 and H15. H15 is preferred over H14, as H14 is directly related to H13, which is already selected, and H15 describes the consequence of H14. The hypotheses are grouped in section 6.3.



Likeliness	1	2	3	4	5
Verified	🟥	🟥	🟥	🟥	🟥
5	🟨 16, 17	🟨	🟥	🟥	🟥
4	🟨 12	🟨	🟨 6, 14, 27	🟥	🟥 13, 22, 23, 30, 31
3	🟩 7, 18, 29	🟨 8, 9, 10	🟨	🟨 19	🟥 21, 24
2	🟩 28	🟩 2, 11	🟨 5	🟨 3, 15, 20, 26	🟨
1	🟩	🟩 4	🟩	🟨	🟨 1, 25
Rejected	🟩	🟩	🟩	🟩	🟩
	Impact/Importance				

Figure 6.1: Hypothesis map before verification

6.3. Hypothesis grouping

Many hypotheses are similar or are directly related or exclusive to each other. Therefore hypotheses are grouped to separate verification activities.

1. Telemetry timing sequence: **H26**, **H27**, H28, H29.
2. I2C and LT: **H13**, H14, H20, **H30**, **H31**.
3. Simultaneous transmission: H14, **H15**, H16, H17.
4. Stuck bus counter: H5, **H6**, H7, H30, H31.
5. Switch counters: H8, H9, H10.
6. I2C watchdog: H11.
7. Rotational rate: H25.
8. Collision in space: H1.
9. Reboots: H2, H3, H4.
10. Over current: H12.
11. Magnetorquers: H18.
12. Invalid command: **H19**, H20, H22, H23.

The highlighted hypotheses are selected for verification, which is described in Chapter 7.

7 Verification of hypotheses

This chapter elaborates on the various verification activities performed during this thesis. The various hypotheses selected in 6 are grouped and verified. According to European Cooperation on Space Standardization (ECSS) there are four verification methods[6]:

- Inspection
- Review-of-Design (ROD)
- Analysis
- Testing

In this thesis, verification by testing, is preferred, since spare hardware is available at the faculty and circular reasoning is avoided. Beyond that testing on hardware is the only way to recreate the scenario's, and ECSS states that verification by testing is required for the pre-launch phase[6]. It must be noted however that the hardware logs[27] were outdated as well, and therefore it cannot be guaranteed that the available spare hardware fully represents the hardware used on Delfi-n3Xt.

7.1. Verification campaign 1: Telemetry timing sequence

In this section the verification activities related to hypotheses H26 and H27 are stated, which were identified in Section 4.5.2. These hypotheses are considered to exclude each other as no other possibility was identified. In order to fully verify one hypothesis the other should therefore be rejected.

7.1.1. Plan

The verification plan of campaign 1 consists of first verifying H27 and followed by verification of H26. This order is chosen since verification of the OBC stability is a complex and lengthy procedure as the OBC should be tested under various environmental conditions. A flow chart of the verification order is shown in Figure 7.1.

Since it is already noted that the hypotheses are considered exclusive, there are two scenario's that should not occur, as can be seen in the Flowchart. The verification method chosen for verification of H27 is verification by analysis. This method is used since the GSE is currently lacking the right equipment to record telemetry. Verification of H26 shall be performed using testing of actual hardware. Since testing the stability of the OBC clock is a lengthy process, this shall only be performed when H27 is rejected.

To verify H27, it has to be shown that 13 frames should be transmitted before the buffer of the ITRX becomes empty, as this was observed in Figure 4.29. In the figure it can be seen that the interval between frames increases after 13 transmitted frames. From design specifications [11] the following information is available:

- Buffer capacity: 4 frames.
- Telemetry frame: 224 Bytes.
- OBC frequency: 1 frame/sec.
- Minimum of 10 flags between frames.

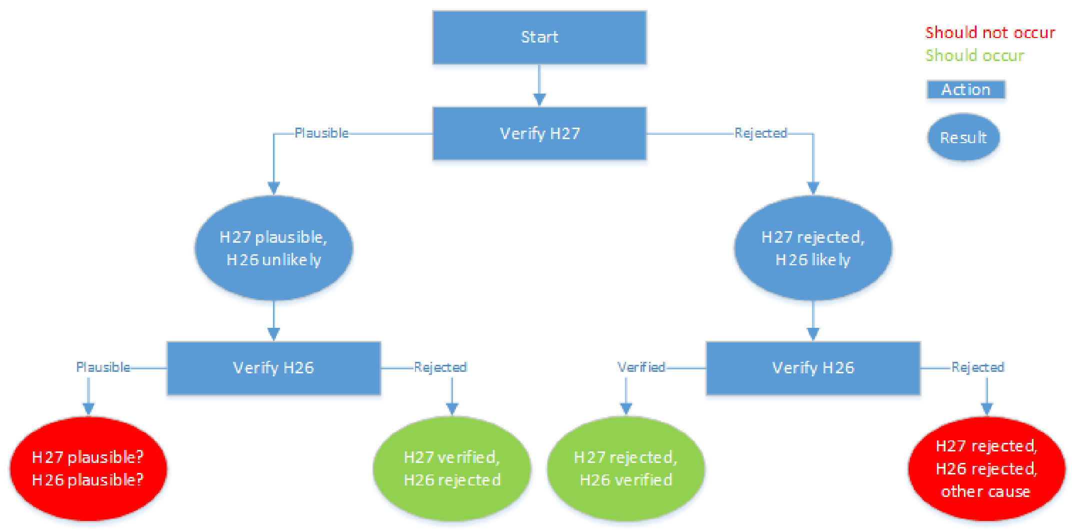


Figure 7.1: Flowchart verification campaign 1.

- 1 Flag is 1 Byte: 01111110.
- Actual frame transmission frequency known from audio recordings, see Figure 4.29.

By simulating the buffer size of the ITRX with an initial value of 4, using the OBC feeding the buffer every second, and by using the actual frame transmission frequency, the number of frames required to empty the buffer can be determined. It must be noted that the OBC could have an initial offset in time (phase) of maximum 1 second. Therefore, if an OBC offset between 0 and 1 s could be found, which corresponds to 13 transmitted required frames to empty the buffer, hypothesis H27 is plausible.

7.1.2. Results

The results of the simulation are shown in the Figures 7.2, 7.3 and 7.4.

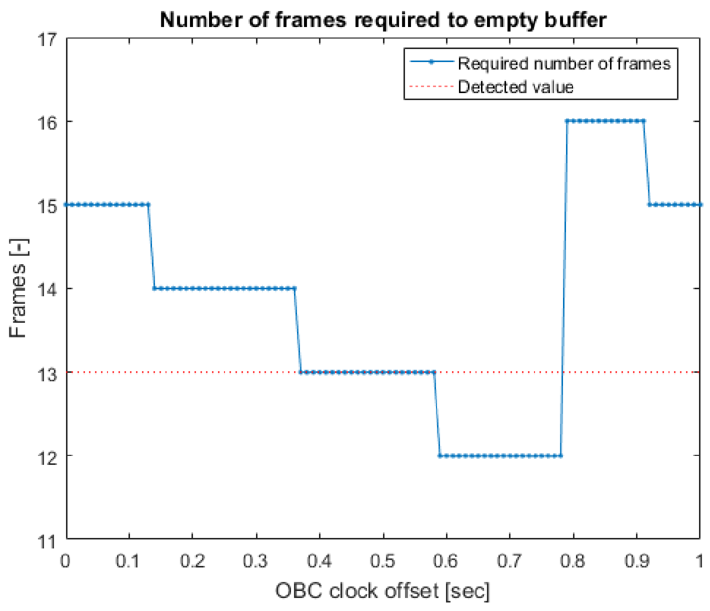


Figure 7.2: OBC offset

Figure 7.2 shows the OBC offset that is required to empty the buffer by transmitting the number frames shown on the y-axis. It must be noted that the actual frame transmission intervals are used, which were known from the audio recordings. These are however not available for 14 or more required frames. For these values an average is used.

It is found that an offset between 0.37 and 0.58 seconds matches with 13 frames which are required to empty the buffer. Therefore the hypothesis H27 is already considered plausible since the number of 13 frames is found on the figure. Figures 7.3 and 7.4 indicate the buffer size over time using the OBC offset found in Figure 7.2.

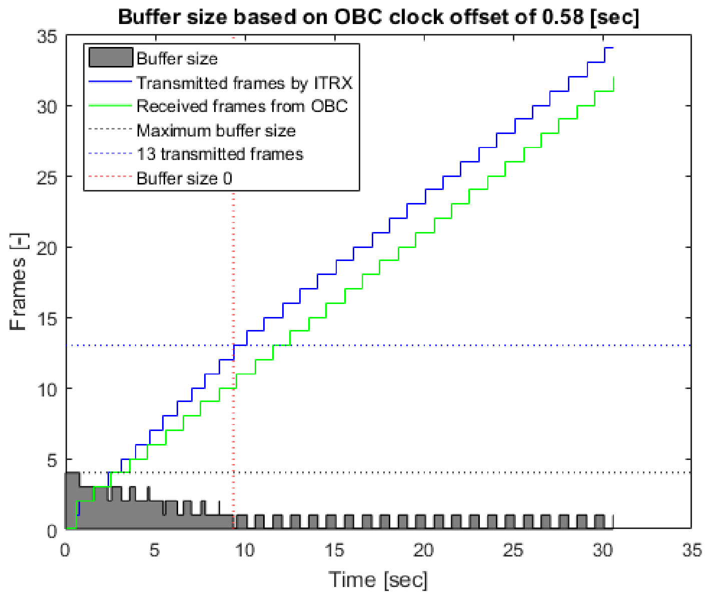


Figure 7.3: Buffer capacity.

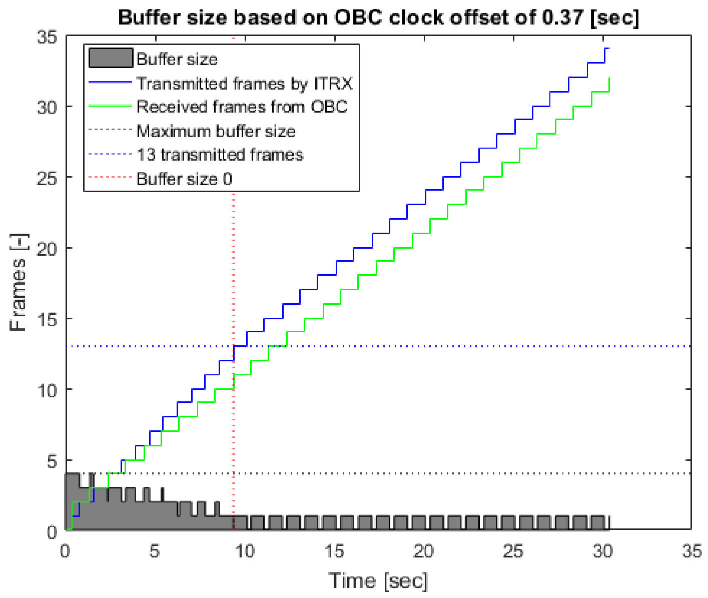


Figure 7.4: Buffer capacity.

As can be seen from Figure 7.3 the buffer size decreases over time, due to the fact that the transmission frequency of the ITRX is higher than the frequency of the OBC. It is also noticed that after 13 transmitted frames the buffer size is indeed zero and that the ITRX and OBC frame counters run parallel afterwards. The same observations are found in Figure 7.4. The influence of the OBC offset can be seen by comparing the figures. It is clear that a lower offset value (0.37 s) barely erases the buffer after 13 frames, while the higher offset value (0.58 s) nearly requires only 12 frames to empty the buffer. This can also be seen in Figure 7.2.

7.1.3. Conclusion

It has been found that H27 is plausible by simulating the ITRX buffer. For this reason the likeliness of H26 is extremely low as the hypotheses were considered exclusive. It must be noted however, that neither H26 or H27 are fully rejected or verified. Full verification can be achieved by rejecting H26. This is not performed during this thesis however, since rejection of that hypothesis is estimated to be a very time-consuming task while there are other hypotheses with higher priorities.

7.2. Verification campaign 2: Reboot and stuck bus counter

In Section 4.4.2 the reboot counter was examined and it was expected that the reboot behaviour was caused by radiation (H2), I2C lock-ups/stuck bus events (H3) or by a parameter overflow (H4). It was argued that H3 was the most likely hypothesis, but in that case the stuck bus counter should match the reboot counter. This is however not the case as shown in Section 4.4.3. Therefore, in this section H6 is tested.

7.2.1. Plan

Verification of H6 would influence the likeliness of various hypotheses. A flowchart can be found in Figure 7.5.

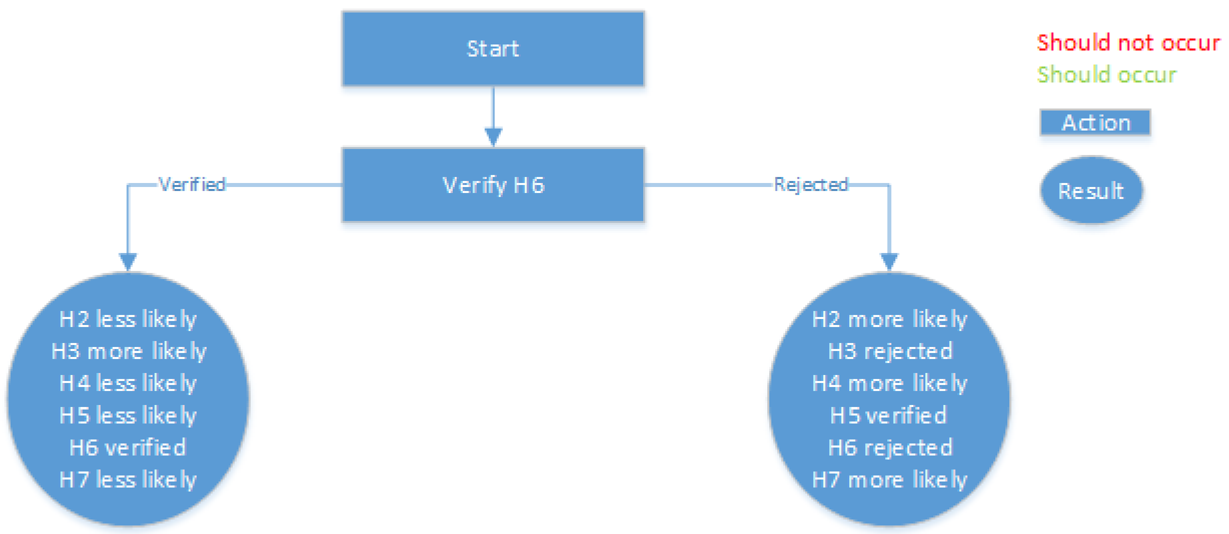


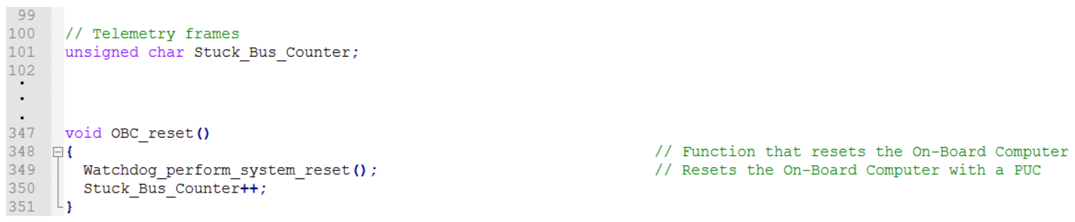
Figure 7.5: Flowchart verification campaign 2.

To verify H6 the I2C bus should be locked manually and the counter value should be examined. The reboot counter shall also be examined to investigate the relation between the reboot and stuck bus counter.

7.2.2. Results

Verification campaign 2 was combined with verification campaign 3 since both campaigns use similar hardware. The results can therefore be found in Sections 7.3.8 to 7.3.11. From these test results it appears that the stuck bus counter wasn't correctly implemented. Therefore the OBC software is briefly examined in this section. Figure 7.6 highlights the implementation of the stuck bus counter in version 5.0 (20130111) of the OBC software.

As is observed from the Figure 7.6, the stuck bus counter is initialised as an unsigned char without an assigned value. Therefore the initial value should be zero. This value is only changed in the function *OBC reset()*. Unfortunately, the reset is triggered *before* the value is increased in size. Therefore the value will never change. Deeper inspection of the software (.map file of the OBC), reveals that the stuck bus counter is stored on volatile memory. Therefore, the counter value is lost after a reboot or power



```

// Telemetry frames
unsigned char Stuck_Bus_Counter;

.
.
.

void OBC_reset()
{                                                           // Function that resets the On-Board Computer
  Watchdog_perform_system_reset();                          // Resets the On-Board Computer with a PUC
  Stuck_Bus_Counter++;
}
```

Figure 7.6: Implementation of stuck bus counter in OBC software. (obc.c version 5.0 (20130111) [26])

cycle. Unfortunately, no explanation was found for the values that were detected during the performed tests.

7.2.3. Conclusion

It is clear that the stuck bus counter wasn't correctly implemented, or implementation wasn't finished before launch. Therefore H6 is verified, which possibly explains the reboot behaviour of the satellite as well. The stuck bus counter should have been stored on flash memory and the counter should have been increased before the actual reset is executed. It is however unclear why the telemetry indicates that the stuck bus counter has a value assigned and why this value, in some cases, changes, as by default, variables are zero initialised. The value appears to be bound between 228 and 238 and the reason for this is unclear.

This phenomenon raises new questions as it might indicate larger problems on the OBC software. Does this happen to other variables which are similarly initialised? Does this phenomenon influence the operations of Delfi-n3Xt? Further investigation is required to answer these questions and therefore no new hypotheses are generated.

7.3. Verification campaign 3: ITRX and LT

From Section 4.5 it is clear that the LT was activated before loss of contact and the LT is therefore a point of interest. It was also found that the LT has operational problems (H13) and that the design of the LT, see Chapter 5, contains a flaw which possibly leads to I2C problems (H30) and (H31). Therefore, in this section functionality test are performed using the ITRX and LT.

7.3.1. Plan

This verification plan consist of multiple functional tests. The plan is to verify H13, by activating and deactivating the LT with varying intervals, using a bottom-up approach, while logging the 5 interfaces between the ITRX and LT. A flowchart is provided in Figure 7.8. During this campaign special attention is paid the the $Enable$ signal to examine pull-up behaviour as discussed in Section 5.1 (H30/H31). Also stuck bus events are introduced to examine it's influence on the performance of the ITRX and LT. For all tests, including the tests proposed in Section 7.2, a similar test set-up is used. The complex form of the test set-up is shown in Figure 7.7. The test set-up provides the following capabilities:

- Transmission of I2C commands from a PC using an Arduino interface that functions as an I2C master when OBC is not connected.
- Transmission of I2C commands from a PC using an Arduino interface that functions as an I2C slave (PTRX) when OBC is connected.
- Continuous recording of all I2C traffic by using a Beagle I2C protocol analyser.
- Switching of supply voltage.
- Introduction of short circuits on the I2C lines using the switch box.
- Connection of multiple subsystems to the I2C bus using the DSSB flex-rigid backbone and break-out box.
- Recording of the interfaces between the ITRX and LT using a Data Acquisition System (DAQ), LabView and a breakout box. The interfaces are (see Figures 5.1 and 5.2):
  - SDA: Data line I2C (3.3V).
  - SCL: Clock line I2C (3.3V).
  - $Supply$: 12 V supply controlled by ITRX.

  - $Enable$: Enable line connected to I2C buffer located on LT (3.3V).
  - $VREF$: Reference voltage of I2C signal to I2C buffer located on LT (3.3V).

- Detection of RF signals using the transceiver.
- Readout of supply and I2C bus voltage.
- Simultaneous transmission of ITRX and PTRX.

7.3.2. Test preparations

To achieve the test set-up for verification campaign 3, two breakout boxes were manufactured during this thesis as interfaces were missing; the I2C breakout box and the LT breakout box. The LT breakout box includes a voltage divider to provide the capability of measuring the voltage supply of the LT. A voltage divider was introduced because the $Supply$ voltage (12 V) exceeds the specifications of the used DAQ (10 V Max). The implemented voltage divider divides the $Supply$ voltage by two.

For the DAQ a program was made in LabView that reduces the sample rate when the input doesn't change. Therefore the output only provides a high sample rate when inputs are changing which are the points of interest. The main advantage is the reduction of file size, which was especially useful for long duration tests.

The Arduino I2C master software was readily available on the Delfi-drive. Unfortunately, no I2C slave device was found. An I2C slave was required since the OBC functions as the I2C master. To transmit a command to the OBC there are two possibilities; use the ITRX or PTRX as a receiver and sent an telecommand while using an external transceiver, or use an Arduino interface that pretends to be a transceiver. Due to a missing interface between PC and transceiver an Arduino script was written that pretends to be the PTRX. For proper functionality of the Arduino interface the commands had to be encrypted before transmission.

The advantage of the used test set-up is the fact that the I2C traffic, $Enable$ signal and voltage $Supply$ are recorded, without adjusting the actual hardware. Beyond that the set-up is easily expandable with multiple subsystems as they could be connected to the DSSB backbone. The downside is the fact that the radio signals(telemetry) are not recorded but only observed due to a missing interface between PC and transceiver. Figures 7.9 provide pictures of the actual test set-up.

For this verification capaign various spare models were selected for use. The list below states which model is selected with some important remarks from the hardware logs [27]:

- **ITRX model C**:
  This model is classified as a flight model and fully represents the model used on Delfi-n3Xt.
- **LT model B**:
  This model is classified as an engineering model since the Si570 chip was replaced by a chip with a lower accuracy. Also input capacitor C33 was removed to solve inrush currents. The board was tested to be stable and represents the flight model used on Delfi-n3Xt.
- **DSSB flex-rigid backbone**:
  No document exists.
- **OBC model H**:
  No remarks, flight model.
- **Phasing circuit**:
  Engineering model which functionally represents the flight model.
- **PTRX model C**:
  No remarks, flight model.

Table 7.1 indicates the various tests performed and their differences. It must be noted that during each test new information was gathered which influenced the plan and procedures of following tests.

Using the selected test set-up the following values were expected:

- SDA @ 3.3 V when ITRX is ON, SDA @ 0 V when ITRX is OFF.
- SCL @ 3.3 V when ITRX is ON, SCL @ 0 V when ITRX is OFF.
- $Supply$ 6 V (due to voltage divider) when LT is ON, $Supply$ 0V when LT is OFF.
- $VREF$ @ 3.3 V when ITRX is ON, $VREF$ @ 0 V when ITRX is OFF.
- $Enable$ @ 3.3 V when LT is ON, $Enable$ @ 0 V when LT is OFF.

Table 7.1: Performed tests

Test	Procedure	Used hardware	Purpose
1	Section A.1	ITRX + LT + Arduino I2C master	Familiarize with operations, and examine ITRX mode switches
2	Section A.2	ITRX + LT + Arduino I2C master	Long duration test of active LT based on orbital period
3	Section A.3	ITRX + LT + Arduino I2C master	Introduce I2C lock-up when ITRX in idle mode or transponder mode and activate flags and transponder mode simultaneously.
4	Section A.4	ITRX + LT + Arduino I2C master	Examine behaviour when flags and transponder mode are on, including introduction of I2C lock-ups and a long duration test.
5	Section A.5	ITRX + LT + Arduino I2C master	Introduction of I2C lock-ups during anomaly.
6	Section A.6	OBC + ITRX + LT + Arduino I2C slave	Familiarize with OBC operations, determine whether simultaneous transmission can occur.
7	Section A.7	OBC + ITRX + LT + Arduino I2C slave	Similar to test 6, examine stuck bus counter.
8	Section A.8	OBC + ITRX + LT + Arduino I2C slave	Introduction of power cycles and short circuits on I2C bus.
9	Section A.9	OBC + ITRX + LT + Arduino I2C slave	Similar to test 8.

7.3.3. Results: Test 1

Figure 7.11 shows the results of Test 1. The figure provides the five interfaces recorded by the DAQ and indicates I2C commands transmitted through the Arduino interface.

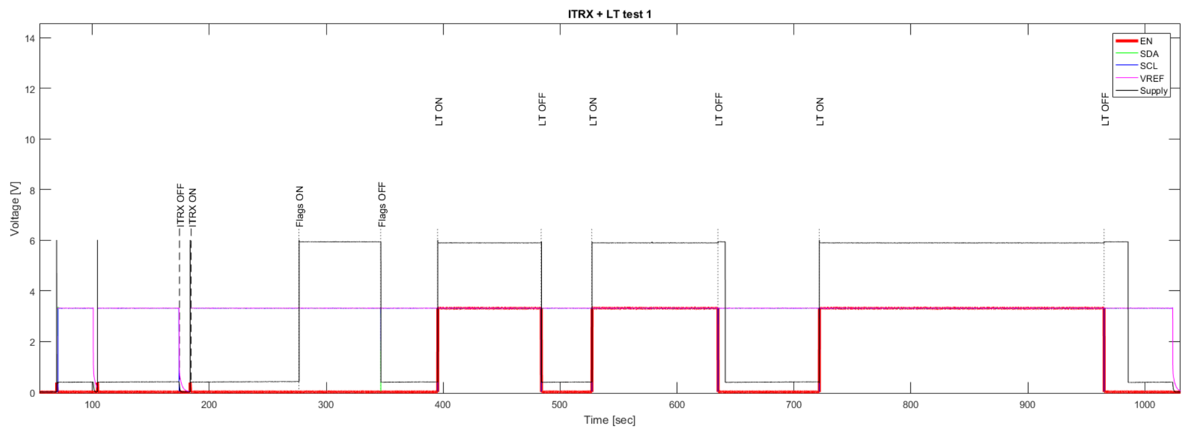


Figure 7.11: Test results Test 1

It can be seen from the figure that the measured values correspond with their expected values, except for the $Supply$ line. The $Supply$ voltage doesn't reach the 0 V (instead 0.8 V) when expected and it reaches 6 V when the flags are ON. It is also observed that I2C data was lost after execution of a device scan (around $t = 100s$). This is normal behaviour since the $SYNC$ messages are not transmitted during the device scan by the I2C master as stated in Section A.1. Therefore the watchdog of the DSSB is triggered, which disables the DSSB I2C buffer. Also when the ITRX is switched OFF I2C data is lost since the system is disconnected. At $t = 280s$ the flags of the ITRX were activated and observed using the transceiver. Afterwards the flags were deactivated and the transponder mode was activated three times. During all three intervals the CW beacon was observed. Analysing the figure it is observed that the $Enable$ line indeed enables the I2C buffer located on the LT and also deactivates when commanded to. Interestingly, the $supply$ voltage has a delay with respect to the $Enable$ signal when the LT is switched off. It was observed that the CW beacon matched the supply line timings. It is also observed that the time to shut-down the supply line/beacon varies for the three intervals.

It is clear from Test 1 that the LT is able to shut-down using the shut-down command. This therefore counteracts H13. Beyond that no I2C problems were found. Since a difference in timing between the supply line and $Enable$ signal upon LT shut-down were observed, a long duration test, Test 2 was performed.

7.3.4. Results: Test 2

The duration of this test was based on the time-line stated in Figure 4.22. According to the time-line the LT should have been active for 46 minutes, until eclipse. This interval was multiplied by two to account for manufacturing differences between flight model used on Delfi-n3Xt and the models used during this test. The results are shown in Figures 7.12, 7.13 and 7.14.

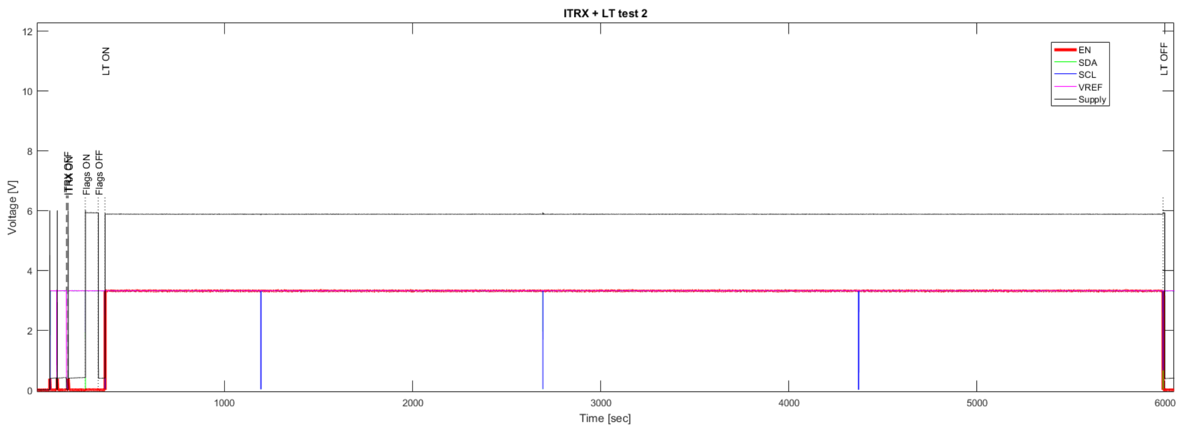


Figure 7.12: Test results Test 2

As was observed during Test 1, the CW beacon and ITRX flags matched the supply line. From Figures 7.12, 7.13 and 7.14 it is observed that there are no differences with respect to Test 1. Some spikes are observed in the I2C signals. These correspond with the $SYNC$ messages transmitted by the I2C master. These were observed in LabView but not recorded continuously due to a mismatch between DAQ sampling and the I2C master. It is also observed that it takes 10 seconds to shut-down the supply line after transmission of the command.

7.3.5. Results: Test 3

Since no direct problems were identified during Test 2, I2C lock-ups were introduced in Test 3 by introducing a short circuit on the SDA or SCL line. These tests were followed by a long duration test were the flags *and* transponder mode were both active. This scenario is possible since these modes are activated using different registers. The results are shown in Figures 7.15 and 7.16.

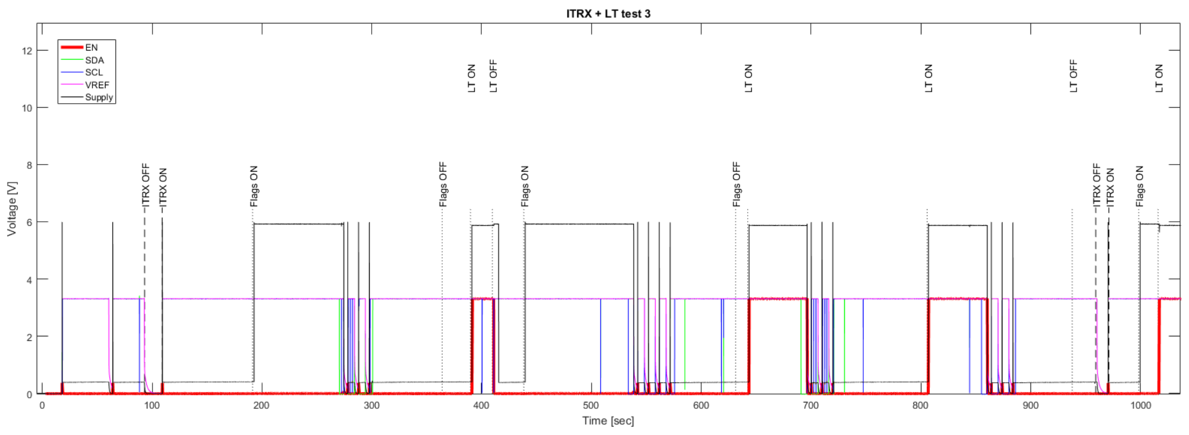


Figure 7.15: Test results Test 3 part 1

Figure 7.15 shows the first part of Test 3. During this part of the test I2C lock-ups were introduced while the ITRX was in idle mode flags ON or in transponder mode. It is observed that in all four scenario's all interfaces drop to zero. This indicates that the I2C watchdogs located on the DSSB μC are functioning as intended, as they shut-down the subsystem. When the short circuit is removed the subsystem reboots in receiving mode (flags OFF), which is the default mode of the ITRX). Therefore the system appears to be working correctly.

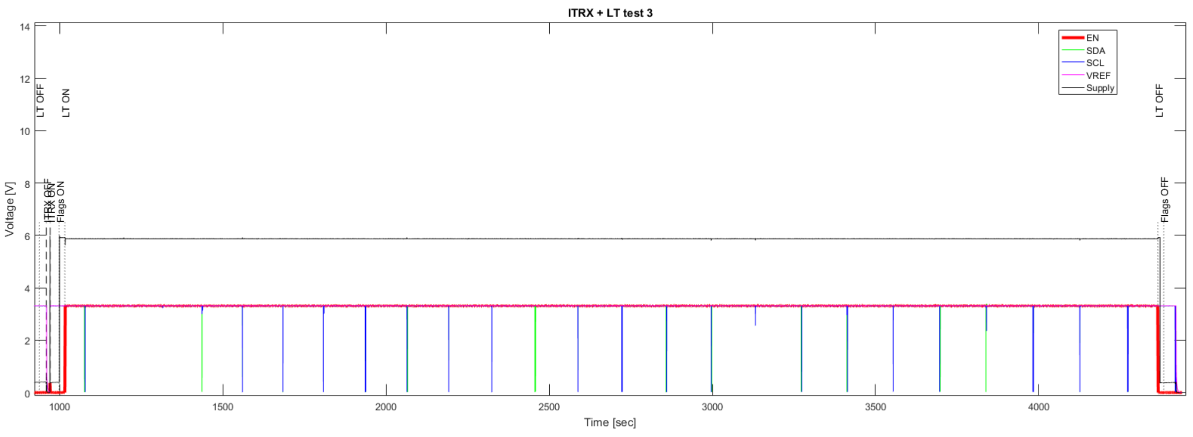


Figure 7.16: Test results Test 3 part 2

Figure 7.15 provides the second part of Test 3. It is observed that the flags were activated, since the supply line reaches 6 V and the flags were observed. Afterwards the transponder mode was activated and both the CW beacon and flags were observed simultaneously. It is noticed that the supply line slightly drops upon activation of the transponder mode. Beyond that it is observed that the transponder mode and flags were deactivated as intended by transmission of the shut-down commands.

7.3.6. Results: Test 4

Also during Test 3 no major anomalies were found. It was however discovered that the transponder mode and flags could be activated simultaneously. Therefore, Test 4 focuses on this phenomenon. Since the I2C commander was used to transmit commands to the ITRX, the same procedures used by the OBC were replicated. Appendix B includes a flowchart of the OBC of the initialization sequence of the ITRX. It is observed that if the current state is *transceiver mode (TX)*, when the flags are ON, that switching to transponder mode doesn't actually switch OFF the flags (highlighted in red) and therefore this scenario is plausible. During Test 4 other scenario's (order or commands) were tested as well and some steps performed in Test 3 were also repeated. The results are shown in Figures 7.17, 7.18, 7.19, 7.20 and 7.21. The last part is not shown here (the long duration test) since the results are similar to Test 3.

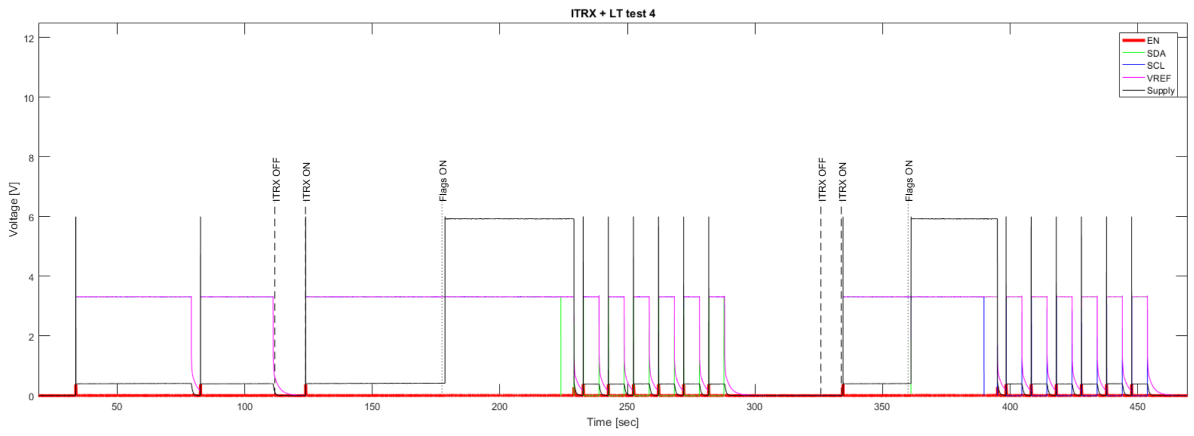


Figure 7.17: Test results Test 4 part 1

Figure 7.17 shows the first part of Test 4, which show a repetition of Test 3 where an I2C lock-up was introduced while the flags were ON. The difference between the tests is the duration of I2C lock. It can be seen that the ITRX completely shuts down after 7 cycles. There are differences to be found in reaction speed of the DSSB μC. This makes sense as the $SYNC$ message possibly has an offset with the introduced lock-up.

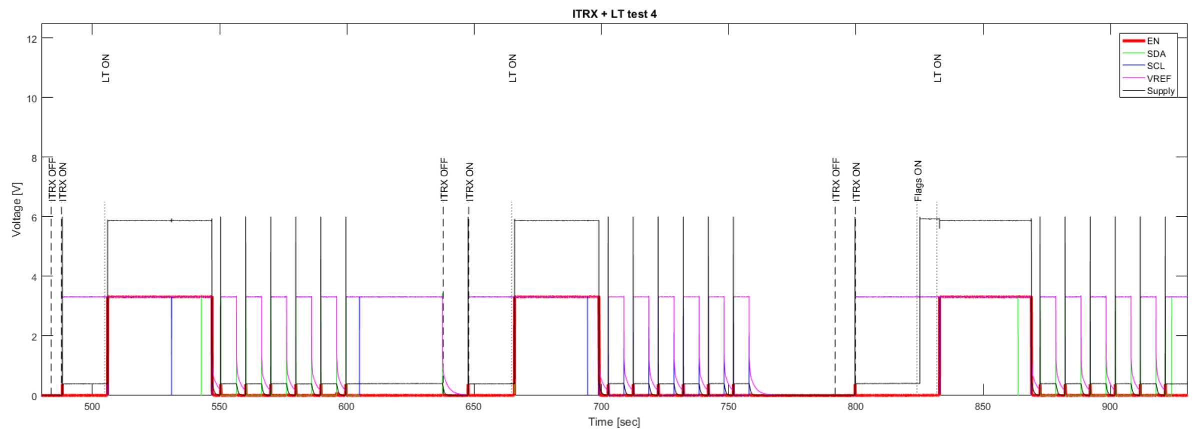


Figure 7.18: Test results Test 4 part 2

Figure 7.18 shows the second part of Test 4 which is very similar to the first part. Again 7 cycles are required to shut-down the ITRX completely. Also Figure 7.19 provides similar results. There are however some interesting observations when the order of commands is changed. When the flags are turned OFF while the transponder mode is active, the $Enable$ line bounces between 1.7 V and 3.3 V and the $Supply$ line bounces between 12 V and 0.8 V. Also the beacon was lost when the $Supply$ was low. Interestingly, the I2C bus was unaffected by this phenomenon, as the Beagle I2C protocol analyser did not indicate communication failures. Also the I2C lines were both at 3.3 V. It is also observed that the anomaly disappears when the LT is switched OFF.

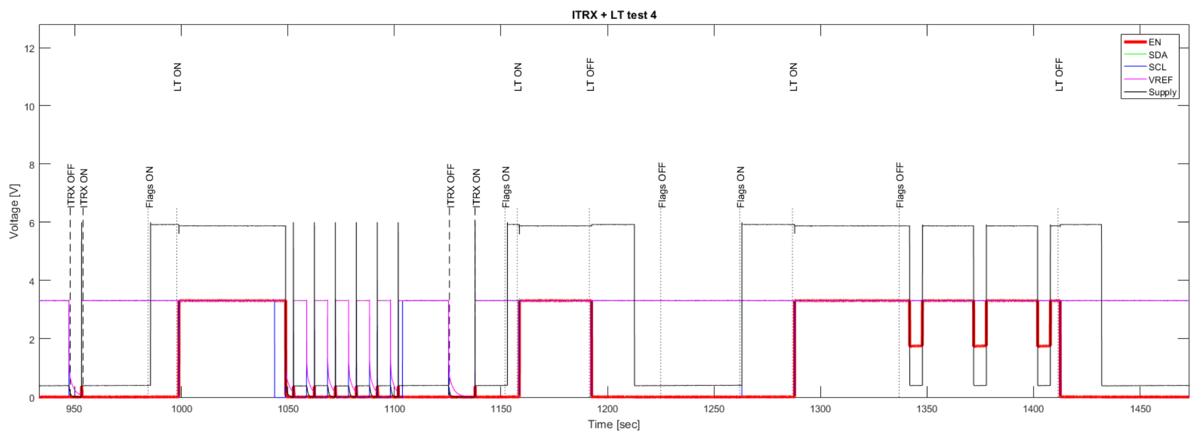


Figure 7.19: Test results Test 4 part 3

Figures 7.20 and 7.21 display the last parts of the performed experiment. These figures display similar behaviour as the previous figures.

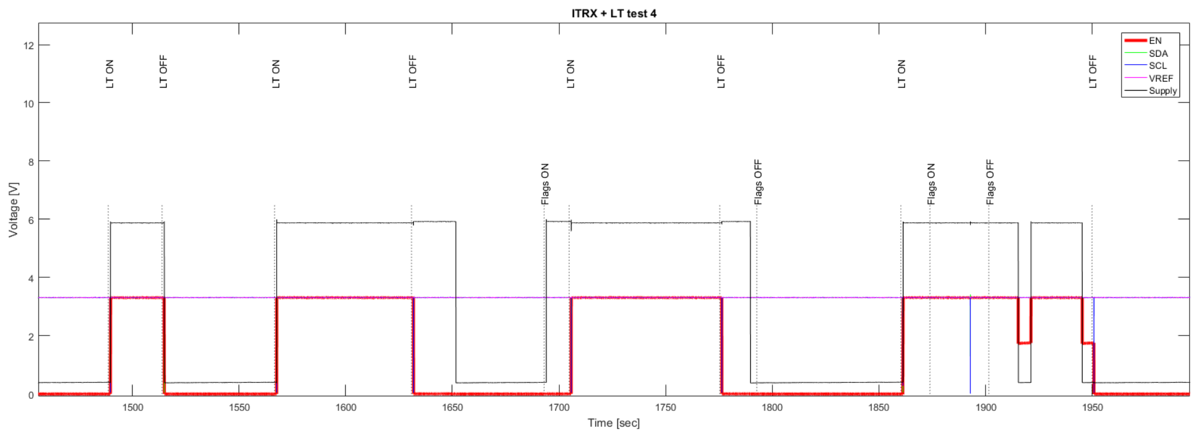


Figure 7.20: Test results Test 4 part 4

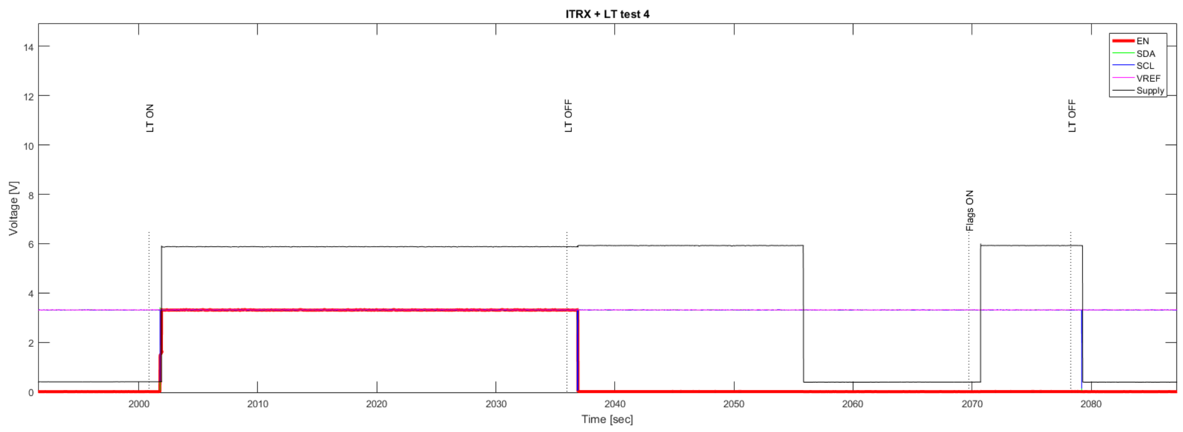


Figure 7.21: Test results Test 4 part 5

7.3.7. Results: Test 5

During Test 4 an anomaly was found when the flags are switched OFF while the transponder mode was active. In Test 5 an I2C lock-up was introduced during the anomaly to examine what happens if such a situation would occur. The results are shown in Figure 7.22. It must be noted that not all data is provided as it was repetition of previous tests.

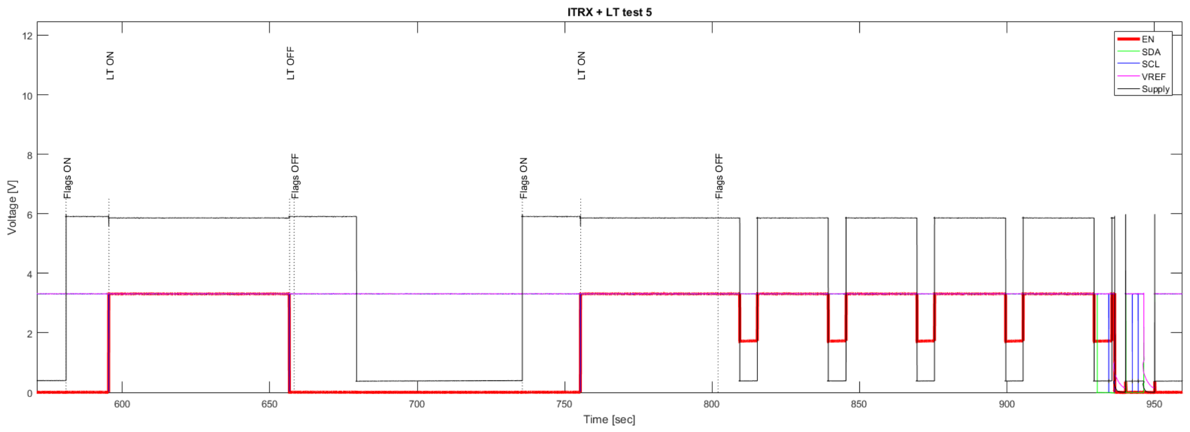


Figure 7.22: Test results Test 5

In Figure 7.22 it is observed that the SDA line was short-circuited when the $Enable$ line was at 1.7 V. It is observed that the anomaly did not affect the DSSB μC since the ITRX was shut-down as intended.

7.3.8. Results: Test 6

Test 6 is the first test were the OBC, ITRX and LT were used while commands were transmitted by the I2C Arduino interface. The purpose of this test was to determine whether the order of I2C commands could lead to the simultaneous activation of the flags and beacon. Also I2C lock-ups were introduced during this test. The results are shown in Figure 7.23.

Figure 7.23 is slightly different from the previously performed tests. The figure indicates the executed (decrypted) commands by OBC and the commands executed by the ITRX. Beyond that the reboot counter and stuck bus counter values are plotted in a separate plot. These counter values are only updated upon transmission of telemetry. Therefore these values are not updated while the transponder is active, when the ITRX is switched off or when the ITRX is set to receiving mode. After activation of the LT only the beacon was observed, no flags or telemetry were transmitted. Thus, changing the $INIT$ vector from transceiving mode to transponder mode doesn't results in simultaneously transmitting the beacon and flags. Interestingly, at $t = 216s$, 71 seconds after the applied command, the ITRX switched to receiving mode while no commands were transmitted. Inspection of the I2C data reveals that the OBC stopped providing telemetry frames towards the ITRX and that the flags and transponder mode were switched off. Since no reboots occurred it appears that $INIT$ vector was changed for unknown reasons. Unfortunately, there is no way to prove this as telemetry is not transmitted after the event occurred. Around $t = 260s$ the SDA line was pulled low and at $t = 328s$ three reboots were reported due to this event, while the stuck bus counter remains constant. This already indicates that the stuck bus counter is not working as intended.

7.3.9. Results: Test 7

Test 7 is mainly a repetition of test 6 with a little variation in transmission of commands. The results can be found in Figure 7.24.

From Figure 7.24 it is observed that there were some anomalies between $t = 0s$ and $t = 100s$. This occurred due to a bad connection at the LT breakout box. The issues were solved and the test was continued. To confirm the observations performed during test 6, the ITRX was set to transceiver mode, followed by transponder mode. Again the beacon was observed, but no flags were detected on the transceiver. At $t = 256s$, 27s after command execution, the OBC stopped providing telemetry and the ITRX was set to receiving mode, which is similar to the anomaly found during test 6. A similar event occurs at $t = 402s$, while the ITRX was set to transponder mode. During the last part of the test stuck bus events were introduced. Again the OBC rebooted several times while the stuck bus counter did not increase in value. Interestingly, when comparing the value of the stuck bus counter with the value detected during test 6, it is increased by one. Clearly, the counter is not working as intended.

7.3.10. Results: Test 8

During test 7, it was found that the stuck bus counter wasn't functioning as intended while it did increase in value between two tests. During test 8 this anomaly is further investigated by introducing I2C lock-ups and power cycles. The results can be seen in Figure 7.25.

From Figure 7.25 it can be seen that initially the LT breakout box was again causing problems. The problems were solved and the test was continued. The SDA line and SCL line were pulled low using the switchboard and similar behaviour was found as during test 7. The reboot counter increased in value, while the stuck bus counter remained constant. It must be noted that it took a very long time to restore the introduced SCL failure. The reason for this is unclear. Afterwards power cycles were introduced. It is observed that the stuck bus counter value is affected. After the first power cycle the value increases, while after the second power cycle the value decreased.

7.3.11. Results: Test 9

During test 8, power cycles were introduced to examine the behaviour of the stuck bus counter. In test 9 multiple power cycles were introduced to further investigate the anomalies. The results are shown in Figure 7.26.

Figure 7.26 indicates the various performed power cycles as the reboot counter increases in value. The stuck bus counter changes value as well but there doesn't appear to be a logic pattern as the value both increases and decreases. It is clear that there is an error in implementation of the stuck bus counter.

7.3.12. Conclusions

During verification campaign 3 various functional tests were performed using the OBC, ITRX and LT. The purpose was to verify H13. It was found that the order of I2C commands towards the ITRX influences the behaviour of the $Enable$ signal and that the ITRX can transmit the beacon and flags simultaneously. It was however also shown that the OBC uses the right order of commands and that therefore this doesn't actually occur during operations. Therefore H13 is rejected. During testing it was found however, that the OBC apparently automatically changes the $INIT$ vector for unknown reasons. Beyond that no direct consequences were found due to the incorrect placement of the I2C buffer and the DSSB μC appeared to be correctly working. Therefore, the likeliness of H30 and H31 should be decreased in value.

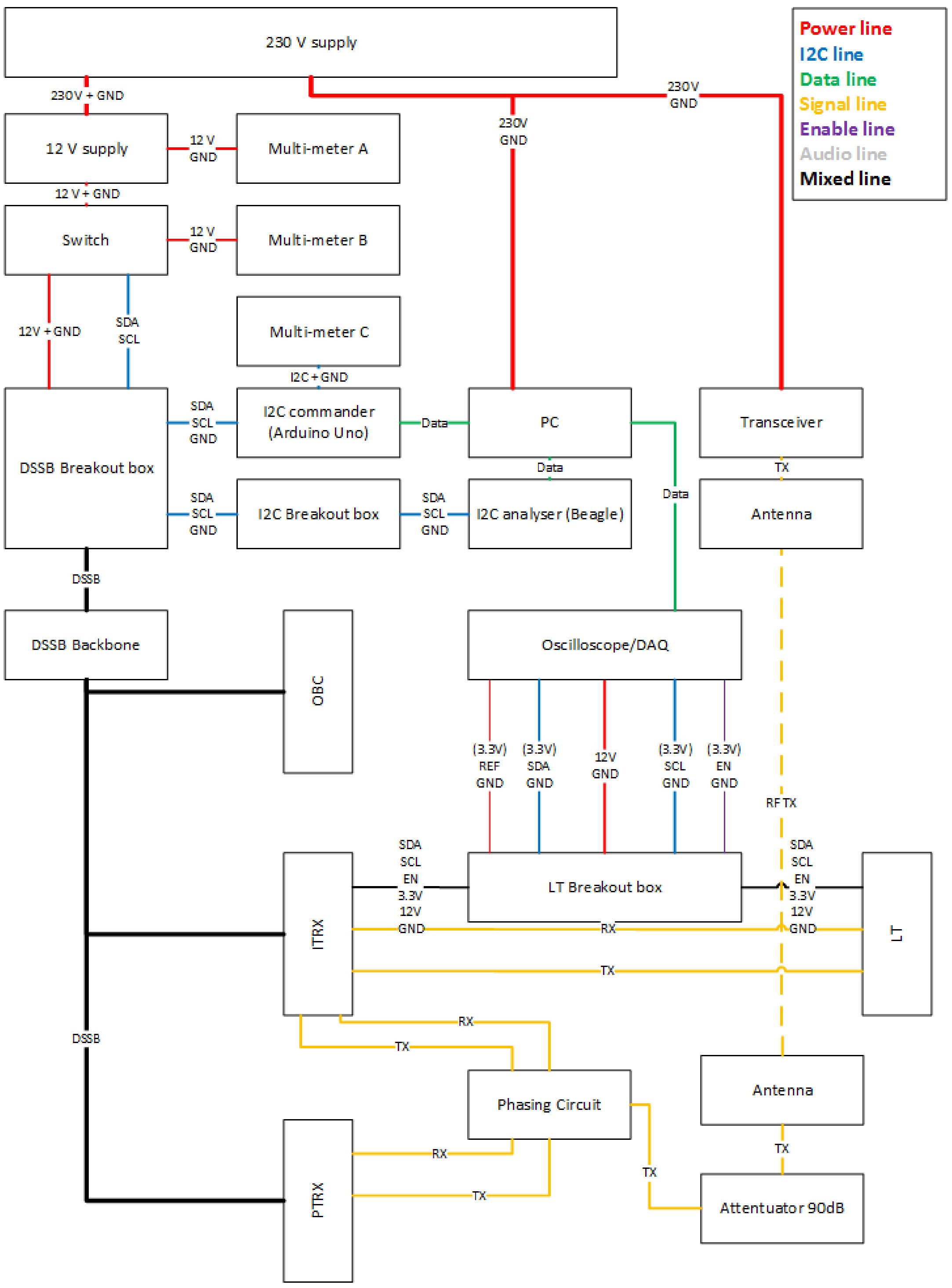


Figure 7.7: Test set-up verification campaign 3, most complex form.

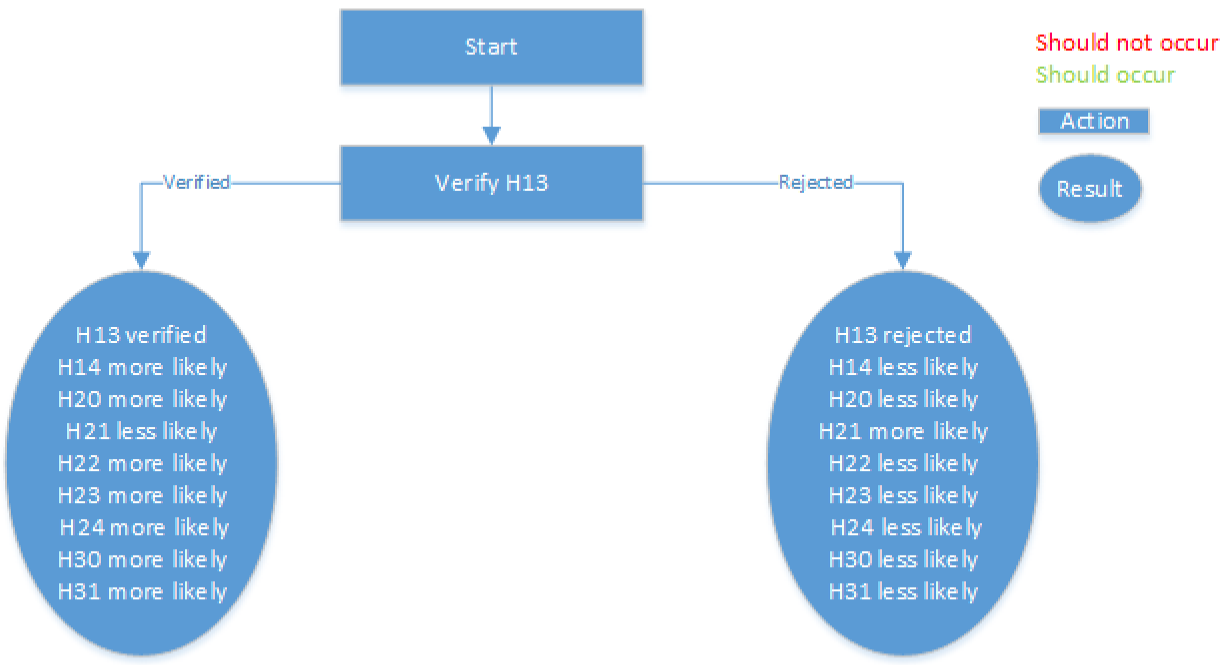


Figure 7.8: Flowchart verification campaign 3.

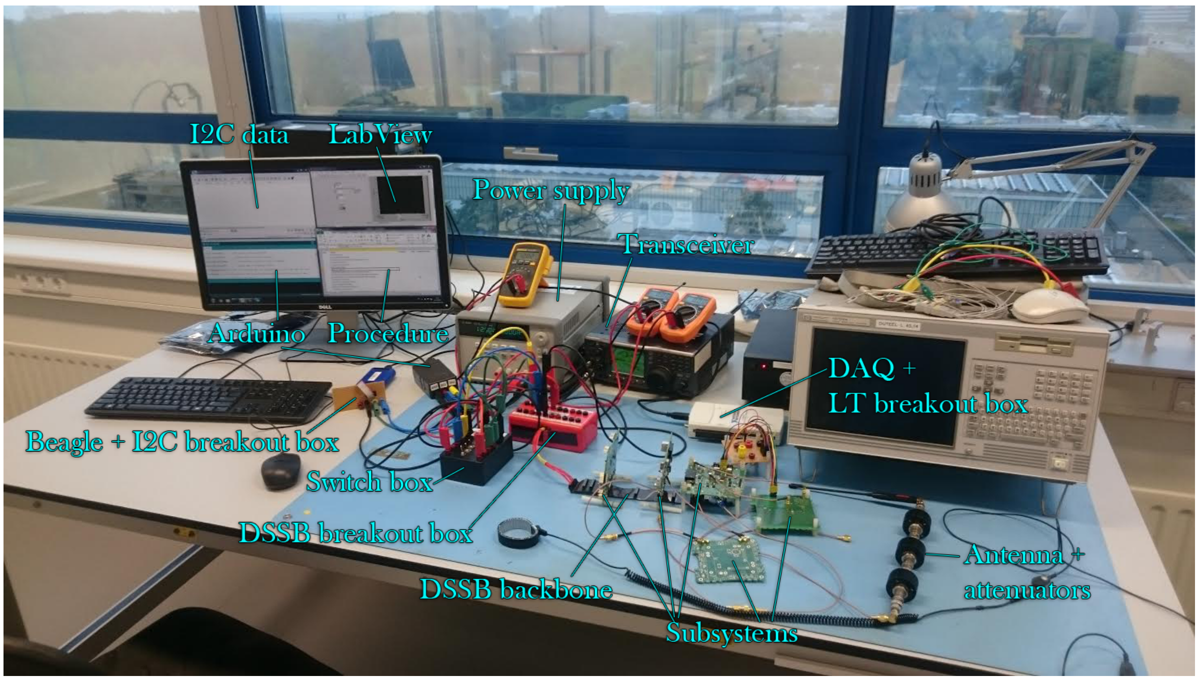


Figure 7.9: Actual test set-up verification campaign 3.

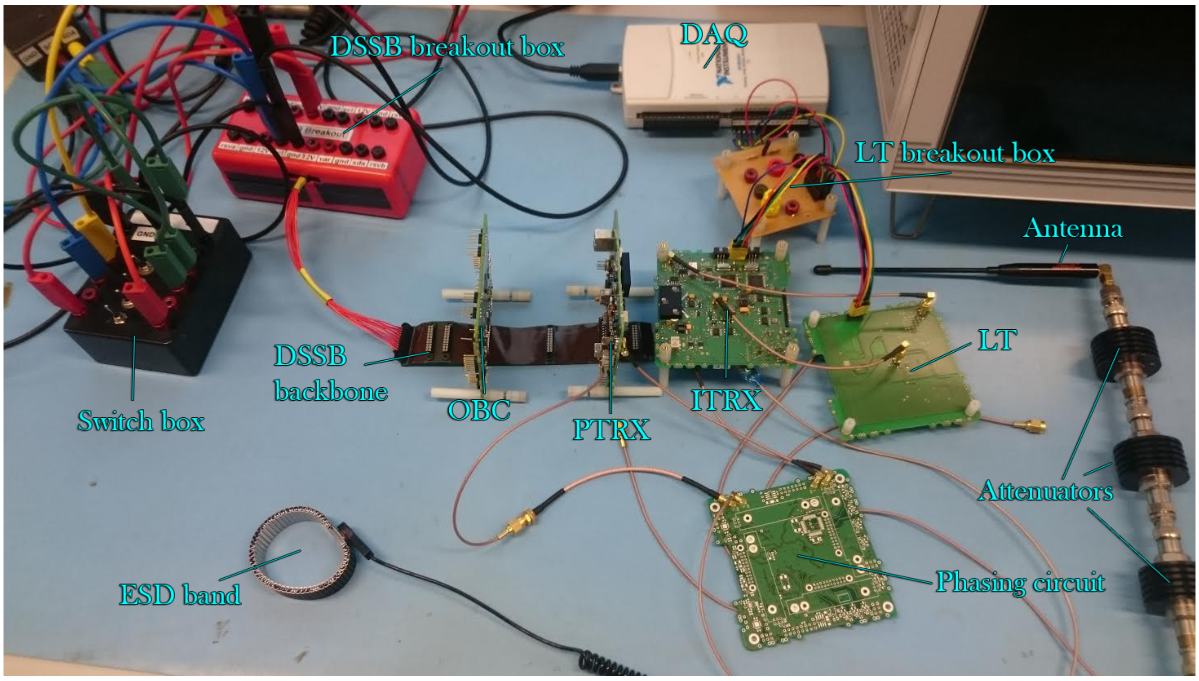


Figure 7.10: Actual test set-up verification campaign 3.

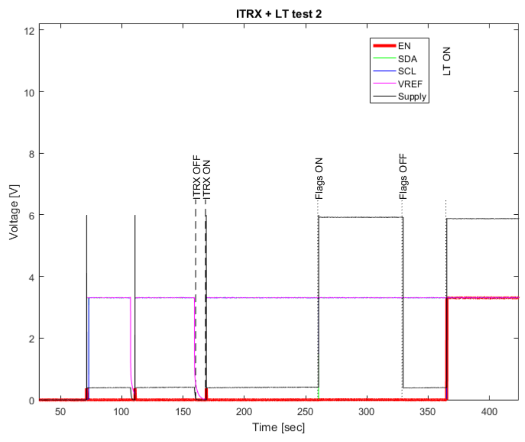


Figure 7.13: Test results Test 2 first part

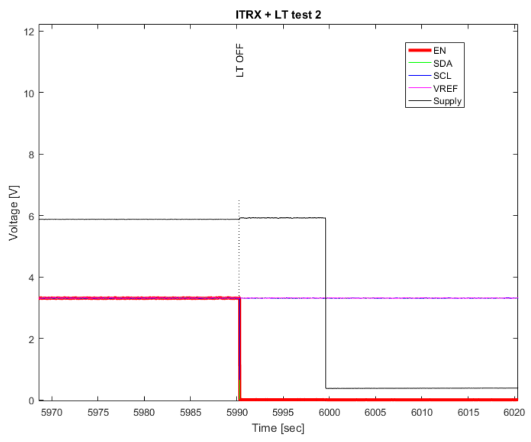


Figure 7.14: Test results Test 2 last part

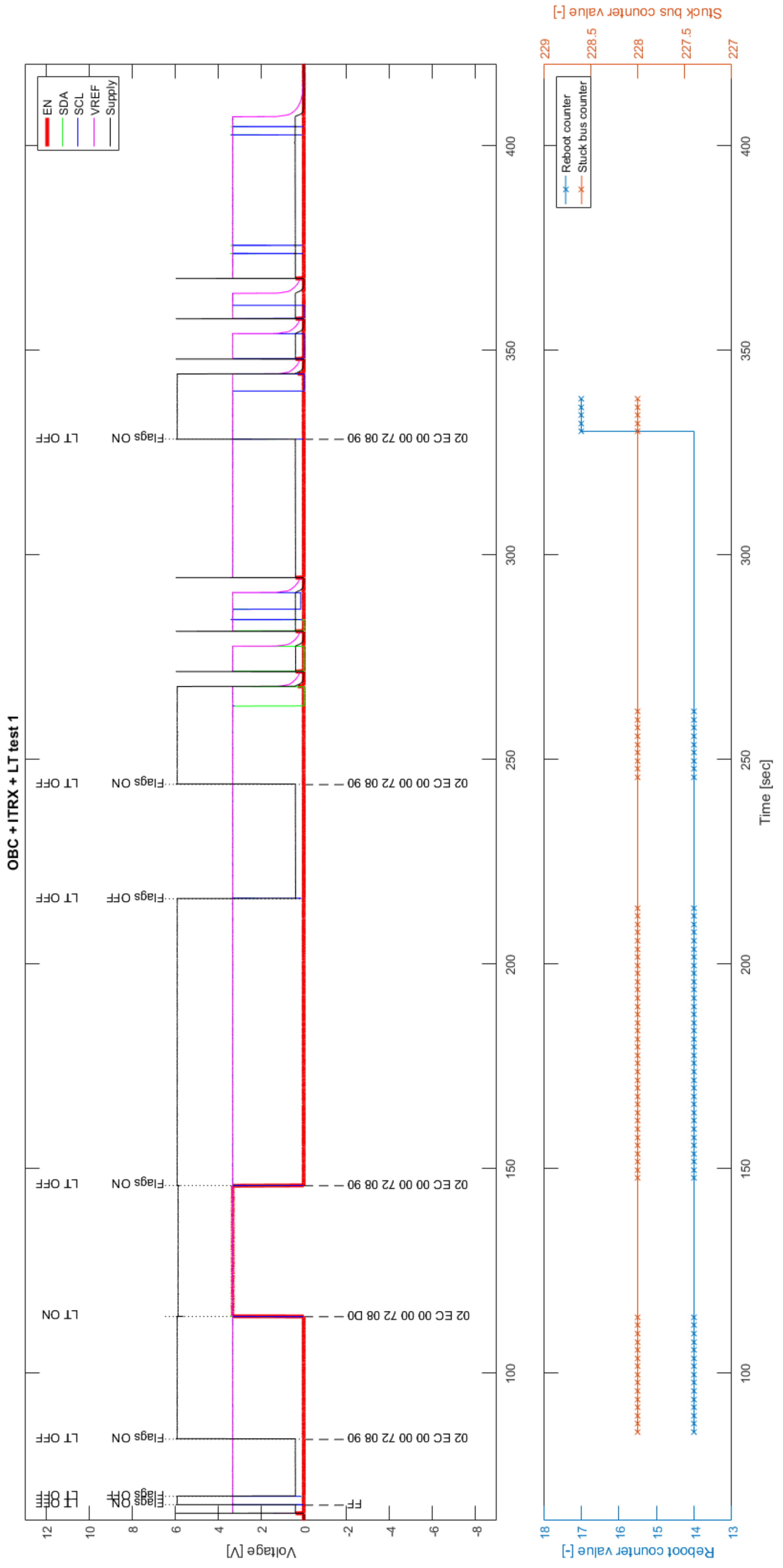


Figure 7.23: OBC + ITRX + LT Test 1

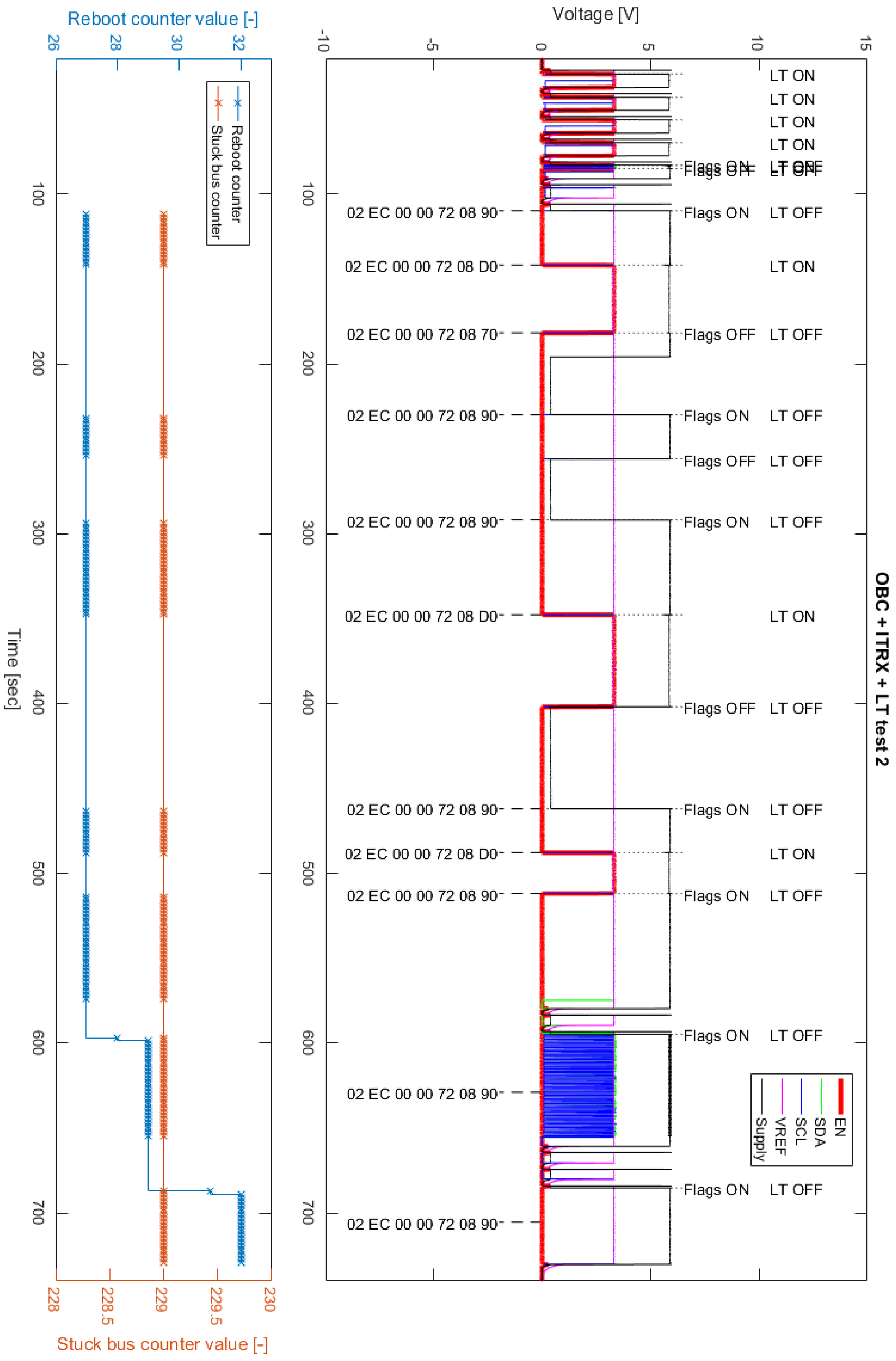


Figure 7.24: OBC + ITRX + LT Test 2

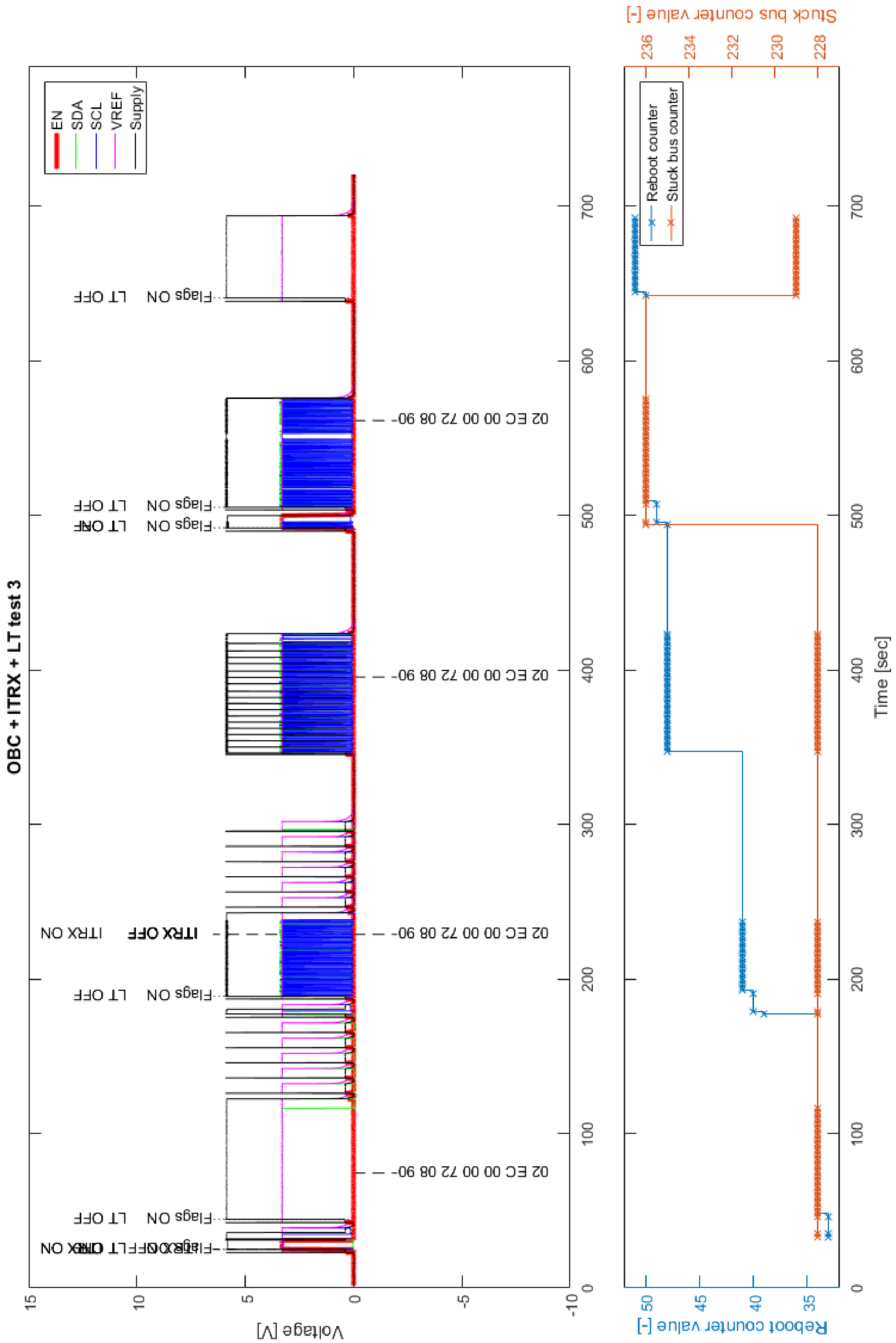


Figure 7.25: OBC + ITRX + LT Test 3

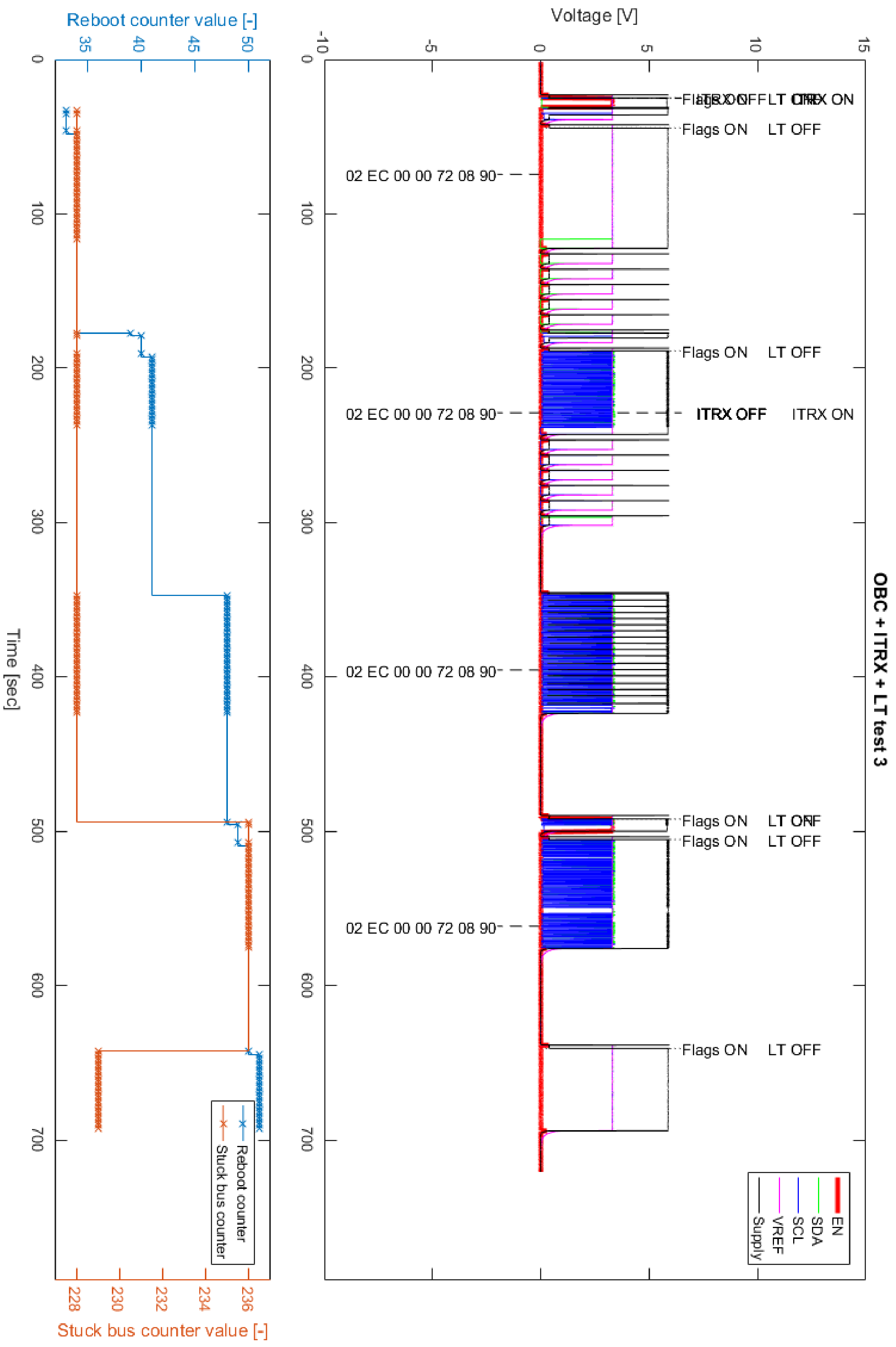


Figure 7.26: OBC + ITRX + LT Test 4

7.4. Verification campaign 4: Simultaneous transmission

During the telemetry analysis, see Section 4.4.10, it was found that there is a possibility that the ITRX and PTRX were simultaneously transmitting after last pass. Although the telemetry analysis has already shown that simultaneous transmission already occurred during the mission while not leading to failure. It is still useful to verify that simultaneous transmission does not lead to failure. The telemetry analysis did not reveal the scenario that the ITRX was in transponder mode, while the PTRX was in transceiver mode. Therefore this is a point of interest.

7.4.1. Plan

The plan is to verify hypothesis H15, by testing, using a similar test set-up as indicated in Figure 7.7. For this test set-up, the PTRX, the ITRX, the LT and the phasing circuit are used. Commands are transmitted to the subsystems using the Arduino interface that functions as the I2C master. During the test, the ITRX is set to transponder mode and the PTRX is set to transceiver mode for 30 minutes to ensure proper functioning of the systems. The relations with other hypotheses are shown in Figure 7.27.

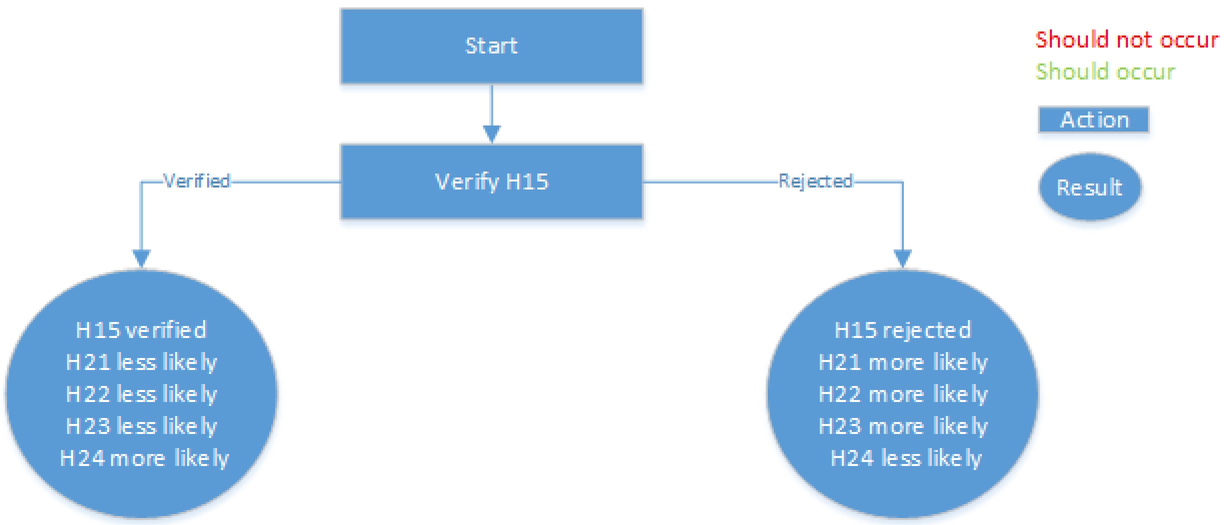


Figure 7.27: Flowchart verification campaign 4.

7.4.2. Results: Test 1

Figure 7.28 indicates that both the ITRX and PTRX were turned ON and were set to planned modes. Both the flags and beacon were observed. No anomalies were found during this test.

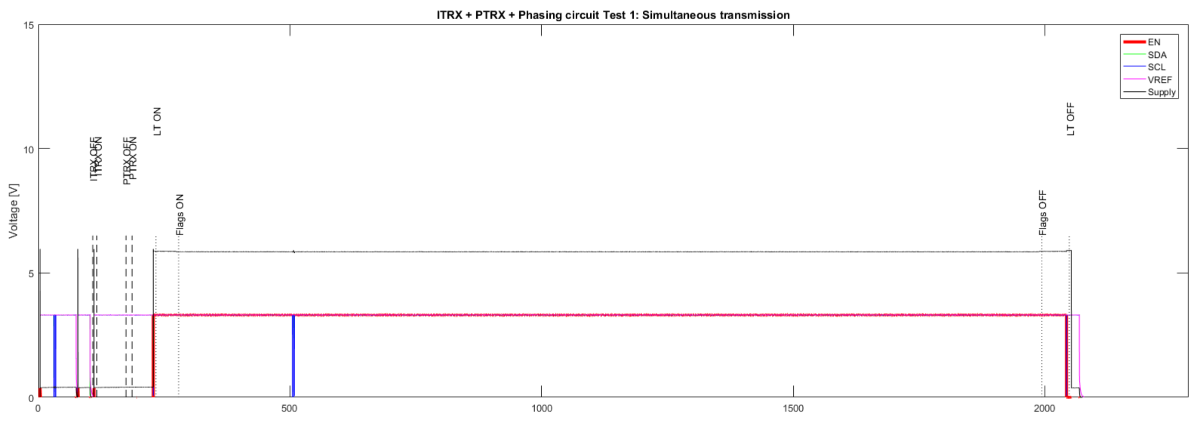


Figure 7.28: Test results simultaneous transmission.

7.4.3. Conclusions

It is concluded that simultaneous transmission of the PTRX and ITRX, for all all operational modes, does not lead to an anomaly and that it doesn't cause a signal loss. Therefore, H15 is rejected. This results in a reduced likeliness of H24 and a slightly increased likeliness of H21, H22, H23.

7.5. Verification campaign 5: Invalid command

This section provides the performed verification activities related to hypotheses H19 and H20. These hypotheses were identified in Section 4.5. During the telemetry analysis, see Section 4.3, it was found that an invalid command was transmitted during operations which resulted in various anomalies. Since this was a non-volatile command, and no other commands were transmitted, this is a point of interest.

7.5.1. Plan

The plan consists of verifying H19. Verification of H19 directly influences the likeliness of H20, H22 and H23 as shown in Figure 7.29.

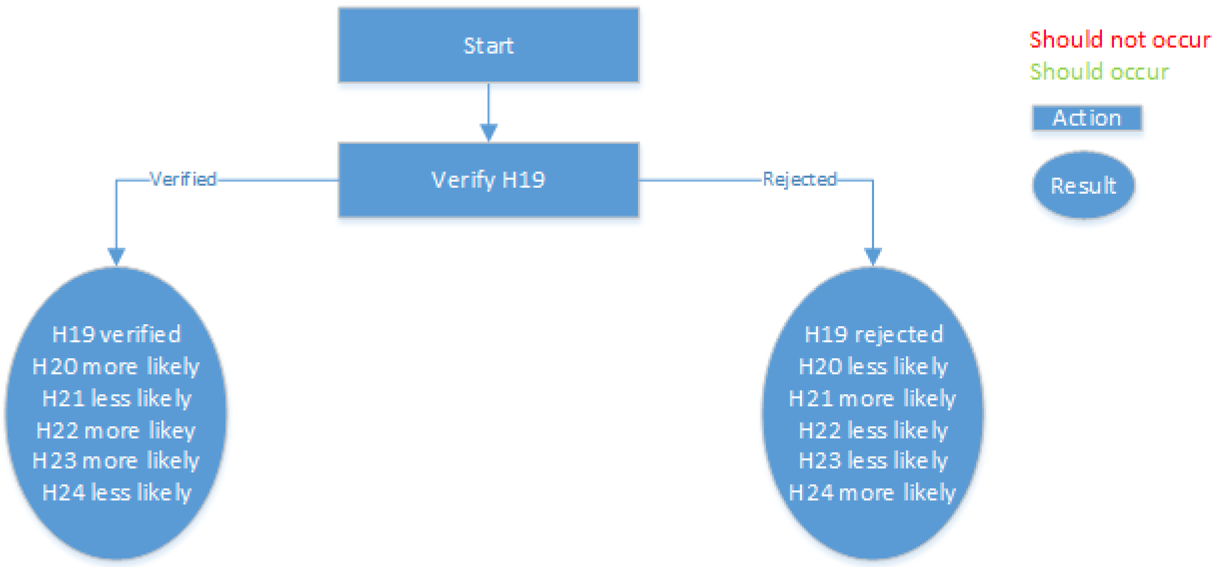


Figure 7.29: Flowchart verification campaign 5.

The verification method used for this campaign is verification by testing. H19 is verified or rejected by transmission of the invalid command to the OBC. Observation of the I2C traffic, reboot counter and ITRX interfaces might reveal possible anomalies. Verification by testing is performed using a similar test set-up as indicated in Figure 7.7. For this test set-up, the OBC, ITRX and LT are used. Commands are transmitted to the OBC by the Adruino interface that functions as a I2C slave device while pretending to be the PTRX. The current test set-up will unfortunately not fully reject H19 when no anomalies are detected. Only a full assembly of Delfi-n3Xt will do. The test however is still useful as it possibly verifies H19.

7.5.2. Results: Test 1

The test results can be seen in Figure 7.30. It can be seen that the invalid command was transmitted at $t = 34s$ and at $t = 132s$, where the ITRX was set to transceiver mode and transponder mode respectively. After reception of the invalid command the reboot counter did not increase in value, and the ITRX remained operational in its current mode. No anomalies were detected during this test. Interestingly, when the ITRX was switched to transceiver mode, at $t = 209s$, it was observed that the telemetry was received at a higher frequency. This seems to match with the results of verification campaign 1.

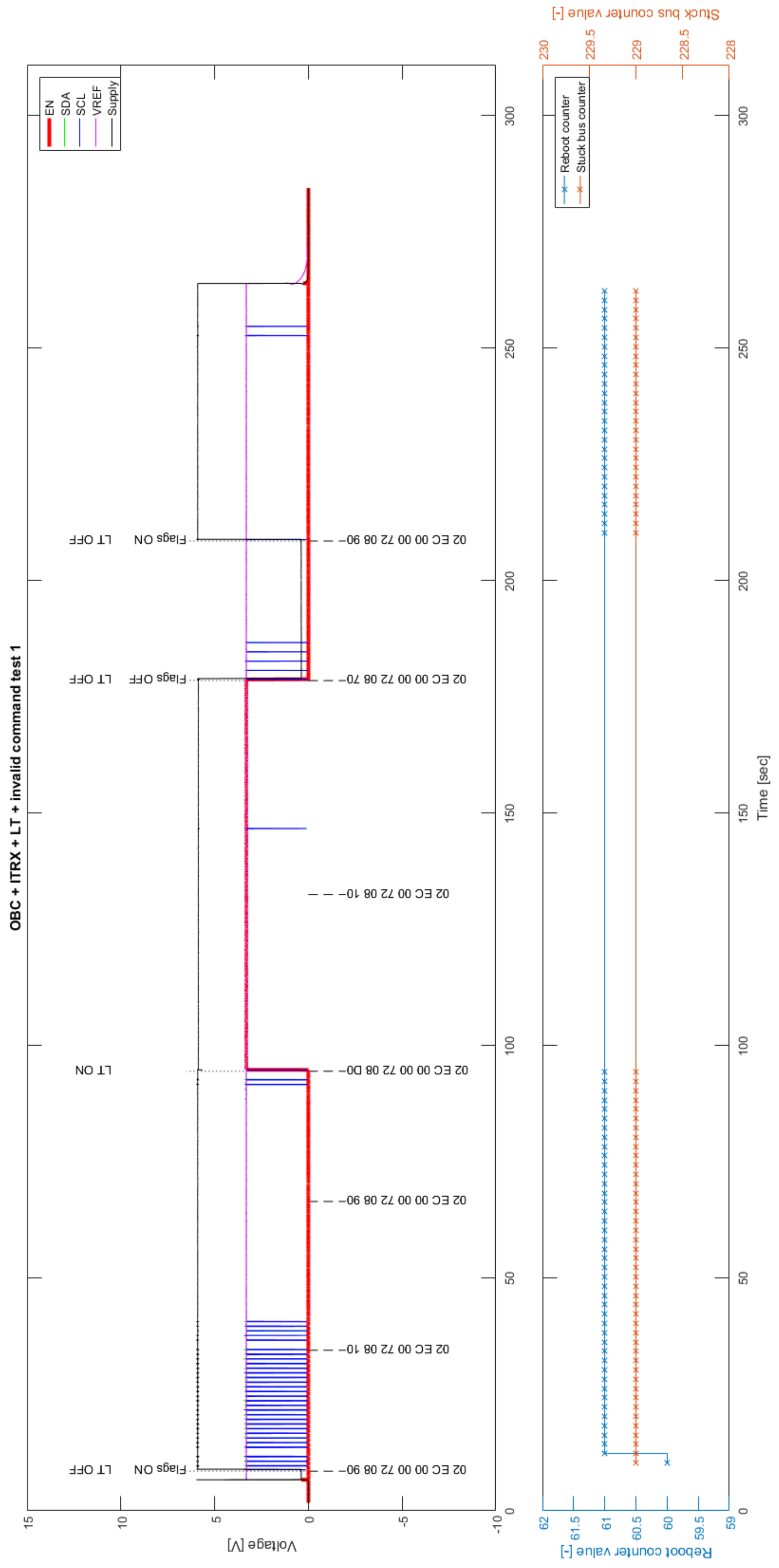


Figure 7.30: Test results invalid command.

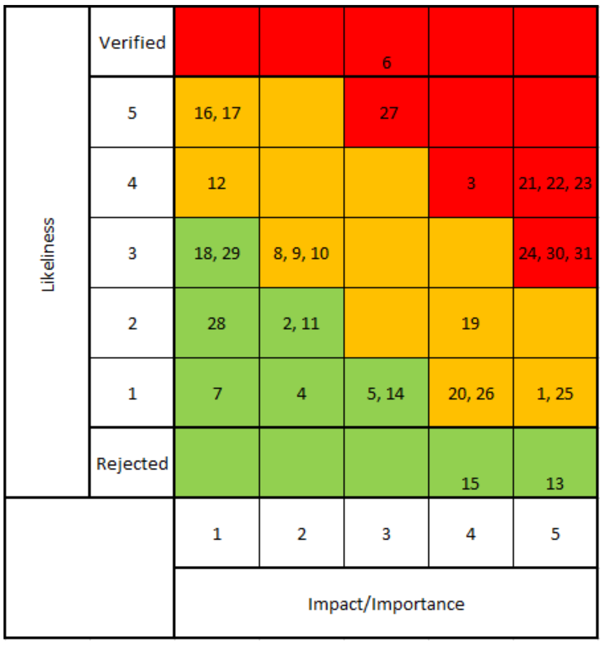
7.5.3. Conclusions

During the test, no direct anomalies were detected. Unfortunately H19 is not rejected since the test set-up did not represent the fully assembled satellite. It can however be concluded that the likeliness of occurrence of H19 is decreased in value and thus also H20, H22 and H23.

7.6. Conclusions

During the various verification campaigns, no anomalies were found that could directly lead to loss of contact. The results of verification campaign 1 indicate a high likeliness that the ITRX was functioning as intended. It is concluded, through analysis of the audio recordings, that H27 is plausible and that, for practical reasons, it is not worth pursuing the rejection of H26. Verification campaign 2 resulted a verified hypothesis (H6). It was proven that the stuck bus counter wasn't correctly implemented through multiple tests and by inspection of the software. It is therefore argued that it is highly likely that the reported reboots were caused by I2C lock-ups. During this verification campaign a new, and rather remarkable anomaly was found. It is unclear why the the stuck bus counter has a value assigned and this might indicate larger problems in the OBC software. Verification campaign 3 was a lengthy campaign were various functional tests were performed using a bottom-up approach. It was found that a specific order or I2C commands could lead to an anomaly, but that this order doesn't actually occur. It was also found that the OBC changes the $INIT$ vector for an unknown reason, but H13 was rejected. Verification campaign 4 resulted in rejection of H15. It was found that simultaneous transmission of the ITRX and PTRX does not lead to failure, although the transceivers share the same phasing circuit. The last verification campaign, campaign 5, resulted in a reduction of likeliness for H19.

Figure 7.31 provides an updated hypothesis map with the results. Only the likeliness values were adjusted with respect to Figure 6.1.



Likeliness						
	Verified			6		
	5	16, 17		27		
	4	12			3	21, 22, 23
	3	18, 29	8, 9, 10			24, 30, 31
	2	28	2, 11		19	
	1	7	4	5, 14	20, 26	1, 25
	Rejected				15	13
		1	2	3	4	5
		Impact/Importance				

Figure 7.31: Updated hypothesis map.

8 Conclusions

In this report a forensic investigation of the loss of contact of Delfi-n3Xt is documented. Delfi-n3Xt was used as a case-study to determine how forensics should be applied to the loss of spacecraft. It was found that fact-finding was a big challenge; the satellite didn't respond to signals, the satellite was inaccessible, formally unverified and the documentation was lacking in many aspects. Therefore, a forensic and empirical methodology were merged into a hybrid approach. All informative sources were used to identify hypotheses and a mission time-line was produced. Hypotheses were selected through a risk assessment and verification was performed by testing on spare hardware and by analysis of audio recordings.

Chapter 1 introduced some questions to support the main research question. These questions are answered below.

1. **How can one identify failure scenario's?**
   (a) **Can one identify failure scenario's by reviewing the design?**
   Documentation was inconsistent and lacking. Therefore design documentation was only consulted to review the electrical design of the LT since the reproduced time-line indicated that the LT was activated before loss of contact. It was found that an I2C buffer was incorrectly oriented but verification did not indicate direct consequences. Beyond that design documentation was consulted to obtain a general idea of the design of Delfi-n3Xt.
   (b) **Can one identify failure scenario's by reviewing the implementation of the design?**
   Before this thesis, a literature survey was performed regarding the verification status of Delfi-n3Xt before launch. It was found that verification wasn't formally performed (lacking documentation) and that it was certainly unclear whether Delfi-n3Xt was free of defects. This did not lead to direct hypotheses, but it was found that the the TRL of the LT, STX were very low indicating a high risk. Also the OBC and and ADCS could have been tested more thoroughly.
   (c) **Can one identify failure scenario's by reviewing the performed operations?**
   A large part of this thesis consisted of reviewing the performed operations. The TLEs were examined for anomalies to identify possible collision events, but non were found at the date of interest. An extensive anomaly investigation was performed using the telemetry and the rotational rate of Delfi-n3Xt was briefly examined. A mission time-line was reproduced and the last passes before loss of contact were analysed in detail by consulting the audio recordings. By reviewing the operations most hypotheses were identified.
   (d) **Can one identify failure scenario's by reviewing the external factors?**
   External factors were not analysed during this thesis, except for impact by other space objects. Beyond that R. Schoemaker [22] attempted to find a relation between the reboot counter and SAA but could not find a direct relation.
   (e) **Can one identify failure scenario's by reviewing the interviewing stakeholders?**
   Throughout this thesis several stakeholders were consulted. A midterm meeting was organised were the telemetry analysis was presented, the audio recordings were analysed and the electrical design of the LT was inspected. The meeting was considered very useful due

to the level of expertise. Some opinions or experiences of stakeholders were directly used as hypotheses while verification resulted in rejection. Remaining objective as a researcher was challenging since some stakeholders attempted to protect their personal interest.

2. **Can one verify the failure scenario's using hardware spare parts/left overs?**
   During this thesis five verification campaigns were executed. During four campaigns testing on spare hardware was used as verification method. Fortunately, all spare hardware parts were functionally similar to the flight models, with some minor differences. It must be noted that the hardware logs were outdated and that important information is therefore missing. For this reason, full confirmation of root cause of failure can never be achieved.

It can be concluded that the used hybrid methodology was effective, but that full confirmation of the root cause of failure is not, can not and will never be achieved since there is no direct access to the satellite. It must be noted however, that using this methodology, an accurate indication can be found by systematically rejecting hypotheses until one is verified.

During this thesis, several recommendations were identified. These are stated in chapter 9.

9 Recommendations

This chapter provides recommendations that were identified during this project and during the literature survey. Throughout this project various issues were found that could be improved by implementation of the recommendations. This chapter is divided in four parts. The first section, Section 9.1 provides some recommendations regarding forensics in general. This section is followed by Section 9.2. This section provides recommendations for the continuation of the forensic investigation regarding Delfi-n3Xt. Section 9.3 states some recommendations regarding the Delfi-program and the last section, Section 9.4 states some final closing words of the author.

9.1. Forensics

In this section recommendations are stated which are related to forensic engineering in general.

1. **Hypothesis identification:** Informative sources
   It is recommended to clearly identify the various resources that could be used to identify hypotheses, to rank them for importance, and to categorise them for facts or questionable data. During this thesis the facts were scarce which made recreation of the mission time-line a time consuming task. Beyond that during this thesis the sources were not ranked for importance and therefore the research process was started by a telemetry analysis instead of a detailed analysis of the last contact. It must be noted however, that the telemetry analysis is considered useful for familiarization with the project and operations (due to a lack of proper documentation).
2. **Hypothesis identification:** Definition
   It is recommended to define an hypothesis as detailed as possible and to avoid general hypotheses. It might be possible to verify a general hypothesis, but it is much more difficult to reject this general hypothesis (only decreases likeliness). Therefore, presenting the results might be disappointing. Beyond that it is recommended to describe the verification method upon identification (similar to requirement identification, stated in Section 9.2).
3. **Hypothesis identification:** Hypothesis discovery tree
   During this thesis hypotheses were identified by observation and by answering the question: Why is this observed? In many cases this question has multiple answers, where some of them are conditional. It is therefore recommended to use a hypothesis discovery tree. This is similar to a requirement discovery tree used in Systems Engineering (SE) which is a very useful tool to identify requirements using a top-down approach.
4. **Hypothesis selection:** Risk assessment, grouping and weights.
   In this thesis hypotheses were selected for verification using a risk assessment. This method is similar to a classical trade-off table as mentioned in Section 6.1, where equal weights for the criteria impact and likeliness are used. It was found that this method was quite a challenge since grouping of these hypotheses should also be considered before selection. Beyond that it is recommended to consider changing the weight ratio between the criteria, since an event with a high probability, but low impact might not be as important as an event with a low probability but high impact.

5. **Workforce:** Motivation and efficiency.
   It is recommended to perform a forensic investigation in a small group with varying expertise. When working alone one could get stuck and therefore possibly lose motivation. When working in a team, team members can motivate each other. Also working in a team increases efficiency due to a wide range of expertise.
6. **Planning:**
   Planning a forensic investigation is challenging since it is unknown what is encountered and how much time it consumes. Especially hypothesis identification is hard to estimate. It is therefore recommended to assign a timeslot to a certain informative source. Verification planning is initially similar to a regular verification campaign. This changes however when anomalies are detected. It is therefore recommended to assume a regular verification campaign with 30 percent extra time for anomalies.

9.2. Delfi-n3Xt forensics

This section states some recommendations for those who would like to continue the forensic investigation of Delfi-n3Xt.

1. **OBC INIT vectors:**
   It is currently unclear what the difference is between a volatile and a non-volatile command intended for the current $INIT$ vector. It is also uncertain whether the last command was volatile or not, see Section 4.5. It is therefore recommended to assemble the EPS and ADCS to simulate the mode change from- and towards eclipse mode and to examine the differences.
2. **OBC variable initialisation:**
   During verification campaign 2 it was found that the stuck bus counter wasn't correctly implemented. More importantly it was found that it's value *should* be zero while the telemetry shows otherwise. It is therefore recommended to analyse the OBC software thoroughly as other variables might be affected as well.
3. **ITRX temperature:**
   It was found (although not reported) that the temperature of the ITRX increased over time. It is thus recommend to investigate this phenomenon.
4. **OBC changing $INIT$ vector:**
   During test campaign 3 it was found that the $INIT$ vector changed suddenly while no commands were transmitted. It is therefore recommended to examine the switch counters, communication status and $INIT$ vector in more detail.
5. **Invalid command:**
   In Section 7.5 an attempt to verify H19 was performed. Unfortunately H19 was neither verified or rejected. It is therefore recommended to transmit the invalid command again for a fully assembled satellite.
6. **DSSB μC:**
   It was found that the μCs of the DSSB were not correctly isolated from the I2C bus. Therefore, if one of those μCs introduces a short circuit on the I2C lines, the problem cannot be solved. It would thus be interesting to investigate the reliability of these μCs. It must be noted, that if this phenomenon actually occurs, it can never be proven.

9.3. Delfi-program

In this section the recommendations regarding the Delfi-program, executed on the University of Technology Delft, are stated.

9.3.1. Project management and Systems engineering

1. **Project reviews:**
   It is strongly recommended to perform various objective reviews during the project on fixed dates. This forces the team members to prepare documentation and to verify the design. During the literature survey [32], it was found that project reviews were planned but not executed. A Preliminary Design Review (PDR) and Critical Design Review (CDR) were performed at a very early stage of the project and an Flight Readiness Review (FRR) was performed during a meeting. Unfortu-

nately, the outcome and decisions were not documented, and therefore it is unclear which risks were accepted.

2. **Review electrical diagrams:**
   It is recommended to review the electrical diagrams in detail by an external expert. During this thesis, in a meeting at December 2015, the electrical diagram of the LT was examined and within five minutes various remarks were made by W. Weggelaar.
3. **Risk assessment and tracking:**
   It is recommended to perform a risk assessment on subsystem level during the project and to track the risks continuously. Risk identification creates awareness among the group on were the problem area's are (resources / verifications status / schedule) and it is a good starting point for a verification plan.
4. **Verification:**
   It is recommended to apply the verification process at the beginning of the design process of a project. Upon identification of a requirement also describe how the requirements should be verified (similar to hypothesis identification, stated in Section 9.1). All requirements should be gathered in a single document which is also used to track the verification status. This way the verification status is continuously and centrally monitored. When combined with a risk assessment unforeseen risks can be mitigated.
5. **Documentation:**
   It is strongly recommended to implement and enforce a documentation standard. Possibly a reduced version of ECSS could be used, were various templates are used. For Delfi-n3Xt and DelFFI, various students only provided their thesis, instead of internal design and verification documents. It is therefore very unclear to determine the final design, since documents are missing, are inconsistent or are ambiguous. This greatly influenced the pace of this thesis, since a lot of time was spent on interpretation of documentation.
6. **Documentation: Operational manuals**
   It is recommended to write operational manuals for each engineering/flight model. The team structure continuously changes due to graduating students. Therefore knowledge is lost. A simple document describing which GSE are needed and how to use it would greatly simplify the steep learning curve for new students.
7. **Team structure:** During the literature review[32], it was argued that for the Delfi-program, the customer is also the main supplier. This possibly has an impact on the number of performed reviews, the verification strategy and verification documents, as the customer is already informed. It is therefore recommended to make a clear distinction between the customer (staff members) and supplier (students).

9.3.2. Technical details

1. **Telemetry server:** Timestamps, location, client side User Interface (UI).
   It is recommended to record as many timestamps as possible, all in Coordinated Universal Time (UTC), and to include the TLEs with epoch closest to the recorded timestamp in the telemetry server upon reception of a telemetry frame. A client side UI could than be used to determine the location of the satellite (using SPG4) for each received telemetry frame. It is also recommended to display each ground station that received the specified telemetry frame in the UI map. This simplifies telemetry analysis greatly as it removes manual labour.
2. **Telemetry server:** Processing
   It is recommended to perform all post-processing of telemetry at each connected client in stead of processing the frames at the telemetry server. During the thesis, it was found that the processing script that was used contained errors and that multiple versions were used during the Delfi-n3Xt mission. This resulted in a bloated database which needed to be formatted and reprocessed before the telemetry could be analysed.
3. **Telemetry package:**
   For Delfi-n3Xt a telemetry package consisted of two telemetry frames containing more than 400 parameters. The first frame was transmitted in the first second of the OBC main loop and the second frame was transmitted in second two. The advantage is that all parameters are downlinked which makes telemetry analysis rather straightforward. The downside is however that information is transmitted which is not of interest and only occupy the frequency bandwidth (for example de-

ployment status of solar panels is only interesting during Launch and Early Orbit Phase (LEOP)). Beyond that for Delfi-n3Xt default values were used for subsystems that were switched off and this might even confuse the operators or telemetry analysts. It is recommended to *only* downlink information that is useful for the operators and customers. It is therefore recommended to design unique telemetry frames for each subsystem. By deactivation of a subsystem the corresponding telemetry frame is not transmitted. This could possibly decrease the time between reception of two consecutive telemetry packages (this also depends on the implementation of the OBC main loop). Beyond that it is recommended to include a dedicated telemetry frame containing debug information regarding the satellite, which should be transmitted as the first frame in the transmission sequence, followed by payload telemetry frames. Beyond that it is recommended to implement an advanced debug frame which can be transmitted upon reception of a specific command.

4. **Telemetry frames:** Status indicators
   The Delfi-n3Xt telemetry frames contain many status indicators. Status indicators are very useful when telemetry is continuously received due to a global ground station network. For Delfi-n3Xt however, this was not the case, and therefore much information wasn't received during the mission. It is therefore recommended to store frames on flash memory when anomalies are detected which can be downlinked while the satellite is in view of a ground station. Unfortunately this makes the system much more complex. Another solution would be or to replace the status indicators by counters. Counters do not register *when* an event occurred, but do at least indicate that *some* event occurred between two consecutively received telemetry frames. A status indicator does neither when the satellite is not in view of a ground station.
5. **Telemetry frames:** Anomaly detection
   During the thesis it was found that many anomalies were ignored or not detected during operations, since the relations within the telemetry frames were not monitored. It is therefore recommended to carefully select the parameters to construct the telemetry frames, and to investigate whether post-processing on ground could be used to detect anomalies during satellite operations. This post-processing script should be finished and verified before launch.
6. **Operations:** Commands.
   It is strongly recommended to implement an UI to transmit commands towards the satellite, and to automatically store the commands in a database. Beyond that it is recommended to include a command identification number within the command frame. The satellite should than include this identification number in the debug frame to indicate that the command was executed. For Delfi-n3Xt the commands, consisting of hexadecimal numbers, were manually transmitted and stored on the Delfi-mailbox. This system is prone to human errors which made reconstruction of the mission time-line very complex. Beyond that, from the telemetry it was unclear which command was executed since the first two most-significant Bytes were used to identify the command. These Bytes are *not* unique.
7. **Operations:** Use of unverified hardware.
   Do not activate unverified hardware before End-Of-Life (EOL) and only after performing a detailed risk assessment. During Delfi-n3Xt the LT was activated while being unverified. Afterwards contact was lost.
8. **Test interface: I2C slave**
   It is strongly recommended to implement an extra slave device on the I2C bus master (OBC). This slave device should behave similar to a transceiver , which is capable of receiving commands. An Arduino interface could than be used to transmit commands to the OBC without use of a transceiver. For Delfi-n3Xt, this was not implemented which resulted various operational problems during this thesis.
9. **DSSB:** Subsystem isolation.
   During testing it was found that the DSSB μC isn't correctly isolated from the I2C bus, see Appendix A. When the I2C bus was active and the 12V supply inactive, the DSSB μC was powered by the I2C signal itself. Therefore, when the DSSB μC itself causes failure on the I2C bus, the I2C bus cannot be recovered. The idea of the DSSB is to be able to isolate subsystems of the I2C bus in case the subsystem fails. With the current implementation the risk of I2C failure is not mitigated but only transferred from subsystem to the DSSB μC. Only when the reliability of the DSSB μC is higher one might speak of risk mitigation. For this reason it is recommended to isolate the the DSSB μC.

9.4. Opinion of the author

During this thesis, the direct root cause was not found, but it is clear that Delfi-n3Xt was not ready for launch, as it wasn't formally verified. The mission was therefore under great risk. It is thus recommended to avoid such situation in future Delfi projects. It must be noted, that Delfi-n3Xt has a small chance of reviving when the batteries are fully depleted. When the batteries deplete a full power cycle occurs after each eclipse which might result in a sign of life.

The Delfi-n3Xt project is considered a very successful project. Many students graduated while working on the project and the satellite was operated for 3 months successfully. It is almost three years after loss of contact and during those three years no signals were detected. During the last two years no revival attempts were performed and therefore it is recommended to officially declare Delfi-n3Xt a lost cause for unknown reasons.

A

Test Procedures

A.1. ITRX + LT Test 1

Table A.1: Procedure ITRX + LT Test 1

Step	Action	x	Time	Comment/result
1	ESD-wrist band	x	11.06	
2	Set switchboard to [off]	x		
3	Connect all ground wires according to test-setup	x		
4	Connect all I2C lines	x		
5	Connect all power lines	x		
6	Verify test-setup	x		
7	Start I2C commander	x		
8	Verify I2C signal equal to 3.3V using multi-meter C	x		[2 - 0V]
9	Start measurements (I2C monitor, oscilloscope, transceiver)	x		
10	Verify functioning of I2C commander (SYNCs/Nacks)	x		I2C pulled low
11	Verify frequency transceiver 145,870 MHz and no signals	x		
12	Turn [on] 12 V supply	x		
13	Verify 12 V with multi-meter A	x	11.07	12,0V
14	Set switchboard to [on]	x		
15	Scan for devices: S	x		data low, timeout
16	Set [0x45 0x20] (turn ITRX [OFF])	x		
17	Set [0x45 0x2F] (turn ITRX [ON])	x		
18	Set ITRX in idle mode: [0x32 0x24 0x01]	x	11.10	
19	Verify flags [ON]	x		Flags observed
20	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x		
21	Verify flags [OFF]	x		
22	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		Beacon observed
23	Verify beacon [ON]	x		
24	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x	11.14	
25	Verify flags [OFF]	x		
26	Set ITRX in transponder mode: [0x32 0x18 0x01]	x	11.14	
27	Verify beacon [ON]	x		Beacon observed
28	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x		
29	Verify flags [OFF]	x		Doesn't switch off immediately
30	Set ITRX in transponder mode: [0x32 0x18 0x01]	x	11.18	
31	Verify beacon [ON]	x		Beacon observed
32	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x	11.22	
33	Verify flags [OFF]	x		Doesn't switch off immediately
34	Set switchboard to [OFF]	x		
35	Turn [OFF] 12 V supply	x		
36	Store all data files	x		

A.2. ITRX + LT Test 2

Table A.2: Procedure ITRX + LT Test 2

Step	Action	x	Time	Comment/result
1	ESD-wrist band	x	13.27	
2	Set switchboard to [off]	x		
3	Connect all ground wires according to test-setup	x		
4	Connect all I2C lines	x		
5	Connect all power lines	x		
6	Verify test-setup	x		
7	Start I2C commander	x		
8	Verify I2C signal equal to 3.3V using multi-meter C	x		[2 - 0V]
9	Start measurements (I2C monitor, oscilloscope, transceiver)	x	13.28	
10	Verify functioning of I2C commander (SYNCs/Nacks)	x		I2C pulled low
11	Verify frequency transceiver 145,870 MHz and no signals	x		
12	Turn [on] 12 V supply	x		I2C pulled low
13	Verify 12 V with multi-meter A	x		12,0V
14	Set switchboard to [on]	x		
15	Scan for devices: S	x		data low, timeout
16	Set [0x45 0x20] (turn ITRX [OFF])	x		
17	Set [0x45 0x2F] (turn ITRX [ON])	x		
18	Set ITRX in idle mode: [0x32 0x24 0x01]	x	13.33	
19	Verify flags [ON]	x		Flags heared
20	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x		
21	Verify flags [OFF]	x		
22	Set ITRX in transponder mode: [0x32 0x18 0x01]	x	13.35	planned time to end: 15,05
23	Verify beacon [ON]	x		
24	Set ITRX transponder mode [off] / receiving mode after 90 min: [0x32 0x18 0x00]	x	15.05 +	
25	Verify flags [OFF]	x		
26	Turn [OFF] 12 V supply	x		
27	Store all data files	x		

A.3. ITRX + LT Test 3

Table A.3: Procedure ITRX + LT Test 3

Step	Action	x	Time	Comment/result
1	ESD-wrist band	x	13.24	
2	Set switchboard to [off]	x		
3	Connect all ground wires according to test-setup	x		
4	Connect all I2C lines	x		
5	Connect all power lines	x		
6	Verify test-setup	x		
7	Start I2C commander	x		
8	Verify I2C signal equal to 3.3V using multi-meter C	x		[2 - 0V]
9	Start measurements (I2C monitor, oscilloscope, transceiver)	x	13.27	
10	Verify functioning of I2C commander (SYNCs/Nacks)	x		I2C pulled low
11	Verify frequency transceiver 145,870 MHz and no signals	x		
12	Turn [on] 12 V supply	x		I2C pulled low
13	Verify 12 V with multi-meter A	x		12,0V
14	Set switchboard to [on]	x		
15	Scan for devices: S	x		data low, timeout
16	Set [0x45 0x20] (turn ITRX [OFF])	x	13.28	
17	Set [0x45 0x2F] (turn ITRX [ON])	x		
18	Set ITRX in idle mode: [0x32 0x24 0x01]	x	13.30	
19	Verify flags [ON]	x		Flags observed
20	Pull I2C SDA low for 30 seconds	x	13.30	
21	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x		
22	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
23	Verify beacon [ON]	x		Beacon observed
24	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x		
25	Set ITRX in idle mode: [0x32 0x24 0x01]	x	13.33	
26	Verify flags [ON]	x		Flags observed
27	Pull I2C SCL low for 30 seconds	x		
28	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x		
29	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
30	Verify beacon [ON]	x		Beacon observed
31	Pull I2C SDA low for 30 seconds	x		
32	Set ITRX in transponder mode: [0x32 0x18 0x01]	x	13.40	
33	Verify beacon [ON]	x		Beacon observed
34	Pull I2C SCL low for 30 seconds	x	13.41	
35	Set [0x45 0x20] (turn ITRX [OFF])	x		
36	Set [0x45 0x2F] (turn ITRX [ON])	x		
37	Set ITRX in idle mode: [0x32 0x24 0x01]	x		
38	Verify flags [ON]	x		Flags observed
39	Set ITRX in transponder mode: [0x32 0x18 0x01] for 90 min	x	13.44	
40	Verify beacon [ON]	x		Beacon + Flags observed
41	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x		
42	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x		

A.4. ITRX + LT Test 4

Table A.4: Procedure ITRX + LT Test 4

Step	Action	x	Time	Comment/result
1	ESD-wrist band	x	15.03	
2	Set switchboard to [off]	x		
3	Connect all ground wires according to test-setup	x		
4	Connect all I2C lines	x		
5	Connect all power lines	x		
6	Verify test-setup	x		
7	Start I2C commander	x	15.03	
8	Verify I2C signal equal to 3.3V using multi-meter C	x		[2 - 0V]
9	Start measurements (I2C monitor, oscillo-scope, transceiver)	x		
10	Verify functioning of I2C commander (SYNCs/Nacks)	x		I2C pulled low
11	Verify frequency transceiver 145,870 MHz and no signals	x		
12	Turn [on] 12 V supply	x	15.04	I2C pulled low
13	Verify 12 V with multi-meter A	x		12,0V
14	Set switchboard to [on]	x		
15	Scan for devices: S	x	15.05	data low, timeout
16	Set ITRX in idle mode: [0x32 0x24 0x01]	x	15.06	
17	Verify flags [ON]	x		
18	Pull I2C SDA low for 1 minute	x	15.07	Shutdown signal
19	Set [0x45 0x20] (turn ITRX [OFF])	x		
20	Set [0x45 0x2F] (turn ITRX [ON])	x		
21	Set ITRX in idle mode: [0x32 0x24 0x01]	x	15.09	
22	Verify flags [ON]	x		
23	Pull I2C SLC low for 1 minute	x	15.10	loss of signal
24	Set [0x45 0x20] (turn ITRX [OFF])	x		
25	Set [0x45 0x2F] (turn ITRX [ON])	x		
26	Set ITRX in transponder mode: [0x32 0x18 0x01]	x	15.12	
27	Verify beacon [ON]	x		
28	Pull I2C SDA low for 1 minute	x		
29	Set [0x45 0x20] (turn ITRX [OFF])	x		
30	Set [0x45 0x2F] (turn ITRX [ON])	x		
31	Set ITRX in transponder mode: [0x32 0x18 0x01]	x	15.14	
32	Pull I2C SLC low for 1 minute	x	15.15	
33	Set [0x45 0x20] (turn ITRX [OFF])	x		
34	Set [0x45 0x2F] (turn ITRX [ON])	x		
35	Set ITRX in idle mode: [0x32 0x24 0x01]	x		
36	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
37	Verify beacon [ON]	x		
38	Pull I2C SDA low for 1 minute	x	15.18	
39	Set [0x45 0x20] (turn ITRX [OFF])	x		
40	Set [0x45 0x2F] (turn ITRX [ON])	x		
41	Set ITRX in idle mode: [0x32 0x24 0x01]	x	15.20	

42	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
43	Verify beacon [ON]	x		
44	Pull I2C SCL low for 1 minute	x	15.21	
45	Set [0x45 0x20] (turn ITRX [OFF])	x		
46	Set [0x45 0x2F] (turn ITRX [ON])	x		
47	Set ITRX in idle mode: [0x32 0x24 0x01]	x	15.22	
48	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
49	Verify beacon [ON]	x		
50	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x	15.23	long time to switch off
51	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x		
52	Set ITRX in idle mode: [0x32 0x24 0x01]	x	15.24	
53	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
54	Verify beacon [ON]	x		
55	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x	15.26	Beacon still on
56	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x		Beacon still on for a while (stops when 12v goes down)
57	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
58	Verify beacon [ON]	x		
59	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x		Beacon switches off fast
60	Set ITRX in transponder mode: [0x32 0x18 0x01]	x	15.29	
61	Verify beacon [ON]	x		
62	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x		beacon switches off after a long time (with 12V)
63	Set ITRX in idle mode: [0x32 0x24 0x01]	x	15.31	
64	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
65	Verify beacon [ON]	x		
66	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x	15.33	
67	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x		
68	Set ITRX in transponder mode: [0x32 0x18 0x01]	x	15.34	
69	Set ITRX in idle mode: [0x32 0x24 0x01]	x		
70	Verify beacon [ON]	x		
71	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x		Enable signal halved?
72	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x		Switched when enable signal was 50%
73	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
74	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x		Long time to switch off
75	Set ITRX in idle mode: [0x32 0x24 0x01]	x		

76	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x		
77	Set ITRX in idle mode: [0x32 0x24 0x01] for 90 min	x	15.38	
78	Set ITRX in transponder mode: [0x32 0x18 0x01]	x	15.39	
79	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x		
80	Set ITRX transponder mode [off] / receiving mode: [0x32 0x18 0x00]	x		
81	Verify flags [OFF]	x		
82	Turn [OFF] 12 V supply	x		
83	Store all data files	x		

A.5. ITRX + LT Test 5

Table A.5: Procedure ITRX + LT Test 5

Step	Action	x	Time	Comment/result
1	ESD-wrist band	x	10.36	
2	Set switchboard to [off]	x		
3	Connect all ground wires according to test-setup	x		
4	Connect all I2C lines	x		
5	Connect all power lines	x		
6	Verify test-setup	x		
7	Start I2C commander	x	10.37	
8	Verify I2C signal equal to 3.3V using multi-meter C	x		[2 - 0V]
9	Start measurements (I2C monitor, oscilloscope, transceiver)	x		
10	Verify functioning of I2C commander (SYNCs/Nacks)	x		I2C pulled low
11	Verify frequency transceiver 145,870 MHz and no signals	x		
12	Turn [on] 12 V supply	x		I2C pulled low
13	Verify 12 V with multi-meter A	x		12,0V
14	Set switchboard to [on]	x	10.37	
15	Scan for devices: S	x		data low, timeout
16	Set [0x45 0x20] (turn ITRX [OFF])	x	10.38	Some interface issues at DAQ
17	Set [0x45 0x2F] (turn ITRX [ON])	x		
18	Set ITRX in idle mode: [0x32 0x24 0x01]	x	10.43	
19	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
20	Verify beacon [ON]	x		
21	Set ITRX transponder mode & idle [off] / receiving mode: [0x32 0x18 0x00][0x32 0x24 0x00]	x		
22	Set ITRX in idle mode: [0x32 0x24 0x01]	x	10.45	
23	Set ITRX in transponder mode: [0x32 0x18 0x01]	x	10.46	
24	Verify beacon [ON]	x		
25	Set ITRX transponder mode & idle [off] / receiving mode: [0x32 0x18 0x00][0x32 0x24 0x00]	x		beacon hearred until 12V drops
26	Set ITRX in idle mode: [0x32 0x24 0x01]	x	10.48	
27	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
28	Verify beacon [ON]	x		
29	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x		Beacon off when enable signal is 50%
30	Turn SDA off when enable 50%	x		
31	Set [0x45 0x20] (turn ITRX [OFF])	x		
32	Set [0x45 0x2F] (turn ITRX [ON])	x		
33	Set ITRX in idle mode: [0x32 0x24 0x01]	x		
34	Set ITRX in transponder mode: [0x32 0x18 0x01]	x		
35	Verify beacon [ON]	x		
36	Set ITRX idle mode [off] / receiving mode: [0x32 0x24 0x00]	x		
37	Set [0x45 0x20] (turn ITRX [OFF])	x		
38	Set [0x45 0x2F] (turn ITRX [ON])	x		
39	Turn [OFF] 12 V supply	x	10.55	
40	Store all data files	x		

A.6. OBC + ITRX + LT Test 6

Table A.6: Procedure OBC+ ITRX + LT Test 6

Step	Action	x	Time	Comment/result
1	ESD-wrist band	x	11.55	
2	Set switchboard to [off]	x		
3	Connect all ground wires according to test-setup	x		
4	Connect all I2C lines	x		
5	Connect all power lines	x		
6	Verify test-setup	x		
7	Start I2C receiver	x		
8	Verify I2C signal equal to 3.3V using multi-meter C	x		
9	Start measurements (I2C monitor, oscilloscope, transceiver)	x		
10	Verify frequency transceiver 145,870 MHz and no signals	x		
11	Turn [on] 12 V supply	x		
12	Verify 12 V with multi-meter A	x		
13	Set switchboard to [on]	x	11.59	
14	Verify 12 V with multi-meter B	x		
15	Verify I2C (Ack, data FF)	x		
16	Verify transceiver receives noise (thus ITRX in receiving mode)	x		
17	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x		ITRX in transceive/PTRX in receive
18	Listen for telemetry (4 frames at least)	x		No telemetry received, INIT vector retransmitted with success
19	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0xd0:	x		ITRX in transponder/PTRX in receive
20	Verify beacon	x		Beacon detected, no flags.
21	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x		ITRX in transceive/PTRX in receive, telemetry received
22	Listen for telemetry (4 frames at least)	x		Telemetry received, but lost after a while. INIT vector retransmitted
23	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x		ITRX in receive/PTRX in transmit
24	Listen for telemetry (4 frames at least)	x		Telemetry received
25	Pull low I2C SDA for 16 sec	x		Telemetry lost until transmission of new INIT vector
26	Listen for telemetry (4 frames at least)	x		Telemetry received
27	Pull low I2C SCL for 16 sec	x		Telemetry lost
28	Listen for telemetry (4 frames at least)	x		
29	Set switchboard to [off]	x		
30	Record all data	x		

A.7. OBC + ITRX + LT Test 7

Table A.7: Procedure OBC+ ITRX + LT Test 7

Step	Action	x	Time	Comment/result
1	ESD-wrist band	x	16.37	
2	Set switchboard to [off]	x		
3	Connect all ground wires according to test-setup	x		
4	Connect all I2C lines	x		
5	Connect all power lines	x		
6	Verify test-setup	x		
7	Start I2C receiver	x		
8	Verify I2C signal equal to 3.3V using multi-meter C	x		
9	Start measurements (I2C monitor, oscilloscope, transceiver)	x		
10	Verify frequency transceiver 145,870 MHz and no signals	x		
11	Turn [on] 12 V supply	x		
12	Verify 12 V with multi-meter A	x		
13	Set switchboard to [on]	x	16.38	SLC pulled low?
14	Verify 12 V with multi-meter B	x		
15	Verify I2C (Ack, data FF)	x		
16	Verify transceiver receives noise (thus ITRX in receiving mode)	x		
17	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x		ITRX in tranceive/PTRX in receive
18	Listen for telemetry (4 frames at least)	x		Telemetry received
19	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0xd0:	x		ITRX in transponder/PTRX in receive
20	Verify beacon	x		Beacon observed, no flags
21	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x70:	x	16.41	ITRX in receive/PTRX in transmit
22	Verify loss of beacon	x		Takes a long time to shutdown
23	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x		ITRX in tranceive/PTRX in receive
24	Listen for telemetry (4 frames at least)	x		Idle flags ITRX goes off after a while? INIT vector retransmitted
25	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0xd0:	x	16.43	ITRX in transponder/PTRX in receive
26	Verify beacon	x		Transponder mode switches off?
27	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x		ITRX in tranceive/PTRX in receive
28	Listen for telemetry (4 frames at least)	x		Telemetry received
29	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0xd0:	x		ITRX in transponder/PTRX in receive
30	Verify beacon	x		Beacon observed
31	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x		ITRX in tranceive/PTRX in receive
32	Listen for telemetry (4 frames at least)	x		Telemetry received
33	Pull low I2C SDA for 16 sec	x		
34	Listen for telemetry (4 frames at least)	x	16.47	Telemetry received
35	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x		ITRX in tranceive/PTRX in receive
36	Listen for telemetry (4 frames at least)	x		Telemetry received
37	Pull low I2C SCL for 16 sec	x		
38	Listen for telemetry (4 frames at least)	x	16.48	Telemetry received
39	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x		ITRX in tranceive/PTRX in receive
40	Listen for telemetry (4 frames at least)	x		Telemetry received
41	Set switchboard to [off]	x		
42	Store all files	x		

A.8. OBC + ITRX + LT Test 8

Table A.8: Procedure OBC+ ITRX + LT Test 8

Step	**Action**	**x**	**Time**	**Comment/result**
1	ESD-wrist band	x	11.26	
2	Set switchboard to [off]	x		
3	Connect all ground wires according to test-setup	x		
4	Connect all I2C lines	x		
5	Connect all power lines	x		
6	Verify test-setup	x		
7	Start I2C receiver	x		
8	Verify I2C signal equal to 3.3V using multi-meter C	x		
9	Start measurements (I2C monitor, oscilloscope, transceiver)	x	11.31	
10	Verify frequency transceiver 145,870 MHz and no signals	x		
11	Turn [on] 12 V supply	x		
12	Verify 12 V with multi-meter A	x		
13	Set switchboard to [on]	x	11.32	SDA and SCL pulled low due to interface issues at LT breakout box.
14	Verify 12 V with multi-meter B	x		
15	Verify I2C (Ack, data FF)	x		
16	Verify transceiver receives noise (thus ITRX in receiving mode)	x		
17	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x		
18	Listen for telemetry (4 frames at least)	x		Telemetry received
19	Pull low SDA for at least 1 min.	x		
20	Listen for telemetry (4 frames at least)	x		Telemetry received
21	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x	11.35	
22	Listen for telemetry (4 frames at least)	x		
23	Pull low SCL for at least 1 min.	x		Accidentally pulled SDA low just before cutoff SCL
24	Listen for telemetry (4 frames at least)	x		Takes a long time to receive telemetry again, 12v drops detected.
25	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x	11.38	Telemetry sounds odd
26	Set switchboard to [OFF]	x		
27	Wait 1 min	x		
28	Set switchboard to [ON]	x	11.4	After on switch beacon is observed, shortly! Telemetry sounds normal again.
29	Listen for telemetry (4 frames at least)	x		Telemetry received
30	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x		
31	Listen for telemetry (4 frames at least)	x		
32	Wait 1 min	x		
33	Set switchboard to [OFF]	x	11.41	
34	Wait 1 min	x		
35	Set switchboard to [ON]	x		
36	Listen for telemetry (4 frames at least)	x		No beacon detected this time, telemetry received.
37	Set switchboard to [OFF]	x		
38	Store all data	x		

A.9. OBC + ITRX + LT Test 9

Table A.9: Procedure OBC+ ITRX + LT Test 9

Step	Action	x	Time	Comment/result
1	ESD-wrist band	x	11.26	
2	Set switchboard to [off]	x		
3	Connect all ground wires according to test-setup	x		
4	Connect all I2C lines	x		
5	Connect all power lines	x		
6	Verify test-setup	x		
7	Start I2C receiver	x		
8	Verify I2C signal equal to 3.3V using multi-meter C	x		
9	Start measurements (I2C monitor, oscillo-scope, transceiver)	x	11.31	
10	Verify frequency transceiver 145,870 MHz and no signals	x		
11	Turn [on] 12 V supply	x		
12	Verify 12 V with multi-meter A	x		
13	Set switchboard to [on]	x	11.32	Interface issues at LT breakout box detected.
14	Verify 12 V with multi-meter B	x		
15	Verify I2C (Ack, data FF)	x		
16	Verify transceiver receives noise (thus ITRX in receiving mode)	x		
17	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x	11.33	
18	Listen for telemetry (4 frames at least)	x		Telemetry received
19	Set switchboard to [OFF]	x		
20	Wait 1 min	x		
21	Set switchboard to [ON]	x		
22	Listen for telemetry (4 frames at least)	x		Telemetry received
23	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x	11.38	
24	Listen for telemetry (4 frames at least)	x		Telemetry received
25	Set switchboard to [OFF]	x		
26	Wait 1 min			
27	Set switchboard to [ON]	x		
28	Listen for telemetry (4 frames at least)	x		Telemetry received
29	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x	11.40	
30	Listen for telemetry (4 frames at least)	x		Telemetry received
31	Set switchboard to [OFF]	x		
32	Wait 1 min	x		
33	Set switchboard to [ON]	x		
34	Listen for telemetry (4 frames at least)	x		Telemetry received
35	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x	11.41	
36	Listen for telemetry (4 frames at least)	x		Telemetry received
37	Set switchboard to [OFF]	x		
38	Wait 1 min	x		
39	Set switchboard to [ON]	x		
40	Listen for telemetry (4 frames at least)	x		Telemetry received

41	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90:	x	11.42	INIT vector transmitted twice
42	Listen for telemetry (4 frames at least)	x		Telemetry received
43	Set switchboard to [OFF]	x		
44	Store all files	x		

A.10. OBC + ITRX + LT: Invalid command test 1

Table A.10: Procedure OBC + ITRX + LT: invalid command test 1

Step	Action	x	Time	Comment/result
1	ESD-wrist band	x		
2	Set switchboard to [off]	x		
3	Connect all ground wires according to test-setup	x		
4	Connect all I2C lines	x		
5	Connect all power lines	x		
6	Verify test-setup	x		
7	Start I2C receiver	x		
8	Verify I2C signal equal to 3.3V using multi-meter C	x		
9	Start measurements (I2C monitor, oscilloscope, transceiver)	x		
10	Verify frequency transceiver 145,870 MHz and no signals	x		
11	Turn [on] 12 V supply	x		
12	Verify 12 V with multi-meter A	x		
13	Set switchboard to [on]	x	13.22	
14	Verify 12 V with multi-meter B	x		
15	Verify I2C (Ack, data FF)	x		
16	Verify transceiver receives noise (thus ITRX in receiving mode)	x		
17	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90: (ITRX telemetry)	x		
18	Listen for telemetry (4 frames at least)	x		Telemetry received
19	Set non-volatile init vector: 0x01 0xec 0x00 0x72 0x08 0x10: (invalid command)	x		
20	Observe 2 min	x		No anomalies detected
21	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90: (ITRX telemetry)	x		
22	Listen for telemetry (4 frames at least)	x		Telemetry received
23	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0xd0: (ITRX LT)	x	13.24	
24	Verify beacon	x		Beacon observed
25	Set non-volatile init vector: 0x01 0xec 0x00 0x72 0x08 0x10: (invalid command)	x		
26	Observe 2 min	x		No anomalies detected
27	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x70: (PTRX transceive)	x		
28	Observe 1 min	x		No telemetry received
29	Set volatile init vector: 0x02 0xec 0x00 0x00 0x72 0x08 0x90: (ITRX telemetry)	x		
30	Observe 1 min	x		Telemetry received, initially high frequency
31	Set switchboard to [off]	x		
32	Store all data	x	13.27	

A.11. ITRX + LT + PTRX: Simultaneous transmission test 1

Table A.11: Procedure ITRX + LT + PTRX: Simultaneous transmission test 1

Step	Action	x	Time	Comment/result
1	ESD-wrist band	x	14.32	
2	Set switchboard to [off]	x		
3	Connect all ground wires according to test-setup	x		
4	Connect all I2C lines	x		
5	Connect all power lines	x		
6	Verify test-setup	x		
7	Start I2C receiver	x		
8	Verify I2C signal equal to 3.3V using multi-meter C	x		
9	Start measurements (I2C monitor, oscilloscope, transceiver)	x		
10	Verify frequency transceiver 145,870 MHz and no signals	x		
11	Turn [on] 12 V supply	x		
12	Verify 12 V with multi-meter A	x		
13	Set switchboard to [on]	x		
14	Verify 12 V with multi-meter B	x		
15	Verify I2C (Ack, data FF)	x		
16	Verify transceiver receives noise (thus ITRX in receiving mode)	x		
17	Set ITRX in transponder mode: [0x32 0x18 0x01]	x	14.37	
18	Verify beacon	x		Beacon detected
19	Set PTRX in idle mode: [0x2C 0x24 0x01]	x		
20	Verify beacon + flags	x		Beacon + flags detected
21	Set PTRX in idle mode: [0x2C 0x24 0x00]	x	15.07	
22	Verify flags off	x		Beacon only
23	Set ITRX in transponder mode: [0x32 0x18 0x00]	x		
24	Set switchboard to [off]	x		
25	store all data	x		

B
OBC flowchart initialization ITRX

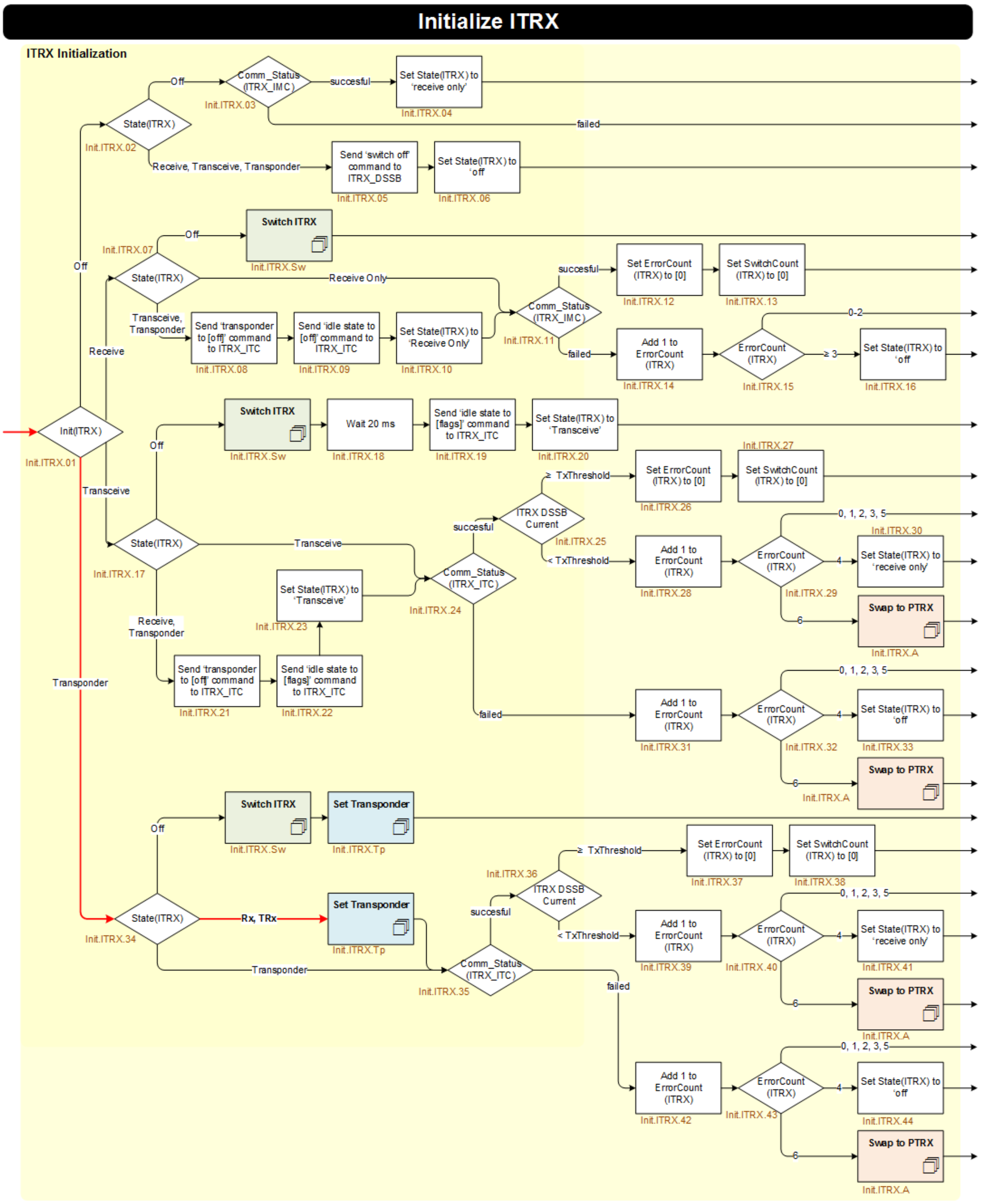


Figure B.1: OBC flowchart initialization ITRX

C
Pass reports

This appendix contains two pass reports which were generated on 05-12-2013[29]. Names are removed for privacy reasons.

C.1. Pass report 1

Delfi-n3Xt Pass Report
Date: < 05-12-2013>
Time:< 11:08 UTC>
Delfi-n3Xt Reception: Good
Observations: Batteries depleted, switched to PTRX due to wrong command interpretation. See notes below
Telecommands:
Main mode: ADCSs off, T3 measuring, SDM off, DABs off, ITRX in transceiver and PTRX off Non-volatile: 0x01 Parameter: 0xF6 0x00 Value: ~~0x00~~ 0x72 0x08 0x82
Eclipse mode: ADCSs off, T3 off, SDM off, DABs off, ITRX off and PTRX in receive Non-volatile: 0x01 Parameter: 0xEE 0x00 Value: ~~0x00~~ 0x72 0x08 0x10
Current Mode: ADCSs off, T3 measuring, SDM off, DABs off, ITRX in transceiver and PTRX off Non-volatile: 0x01 Parameter: 0xEC 0x00 Value: ~~0x00~~ 0x72 0x08 0x82
Because of the missing 0x00 in front of the value, the OBC totally 'screwed' up the init vector and turned off many subsystems and turned on the PTRX instead. This should be solved next pass.

C.2. Pass report 2

Delfi-n3Xt Pass Report
Date: < 05-12-2013>
Time:< 12:30 UTC>
Delfi-n3Xt Reception: Good
Observations: Restored the satellite to low correct low power mode to save the batteries.
Telecommands:
Main mode: ADCSs off, T3 measuring, SDM off, DABs off, ITRX in transceiver and PTRX receiver Non-volatile: 0x01 Parameter: 0xF6 0x00 Value: ~~0x00~~ 0x72 0x08 0x92
Eclipse mode: ADCSs off, T3 off, SDM off, DABs off, ITRX off and PTRX in receive Non-volatile: 0x01 Parameter: 0xEE 0x00 Value: ~~0x00~~ 0x72 0x08 0x10
Current Mode: ADCSs off, T3 measuring, SDM off, DABs off, ITRX in transceiver and PTRX eceiver Non-volatile: 0x01 Parameter: 0xEC 0x00 Value: ~~0x00~~ 0x72 0x08 0x92
Everything was correctly received.

D

Technology Readiness Level (TRL)

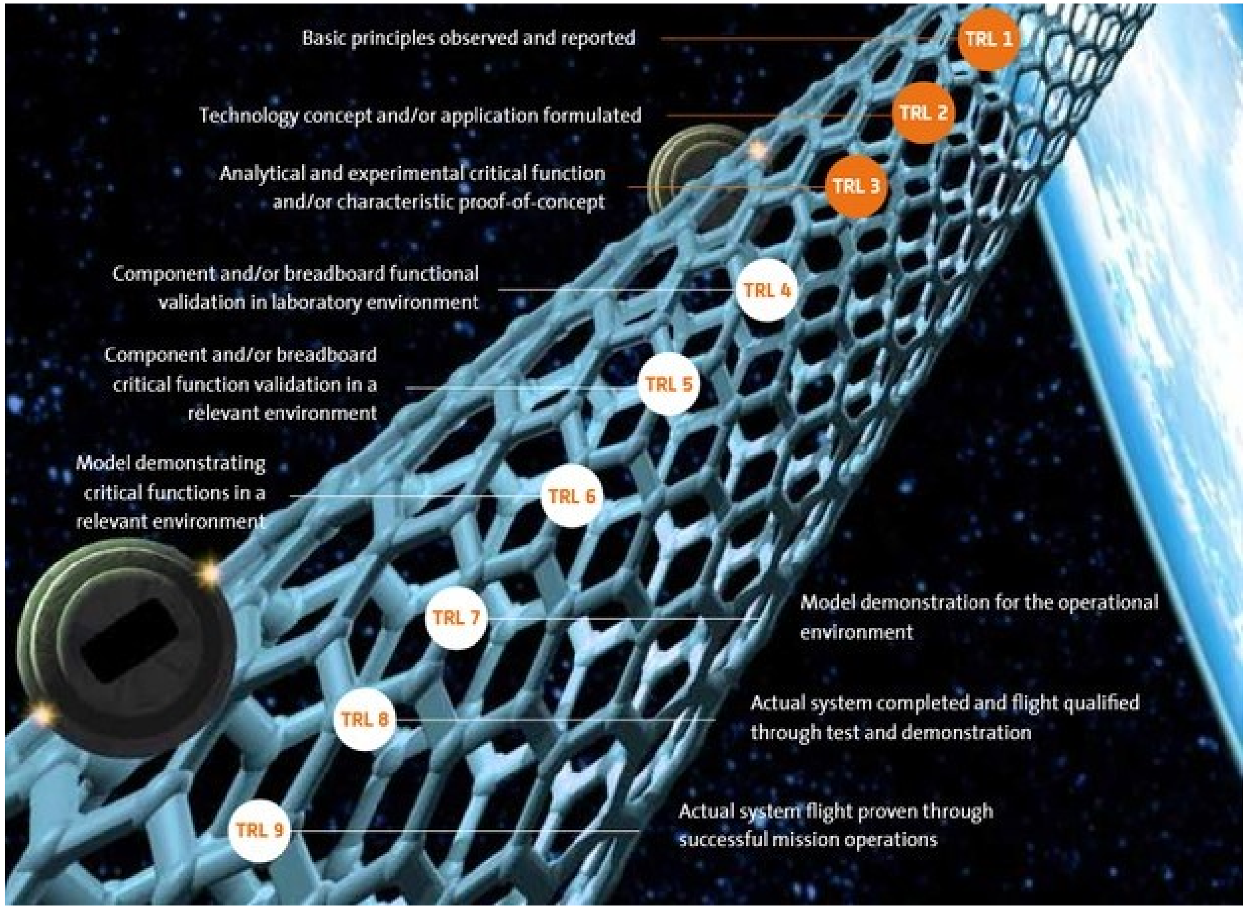

Basic principles observed and reported
TRL 1
Technology concept and/or application formulated
TRL 2
Analytical and experimental critical function and/or characteristic proof-of-concept
TRL 3
Component and/or breadboard functional validation in laboratory environment
TRL 4
Component and/or breadboard critical function validation in a relevant environment
TRL 5
Model demonstrating critical functions in a relevant environment
TRL 6
Model demonstration for the operational environment
TRL 7
Actual system completed and flight qualified through test and demonstration
TRL 8
Actual system flight proven through successful mission operations
TRL 9

Figure D.1: TRL definition[10][8]

Bibliography

[1] Testing hypotheses suggested by the data. https://en.wikipedia.org/wiki/Testing_hypotheses_suggested_by_the_data, Nov 2016.

[2] Texas sharpshooter fallacy. https://en.wikipedia.org/wiki/Texas_sharpshooter_fallacy, Nov 2016.

[3] M.J. Boerci. Assembly, integration and verification of delfi-n3xt. *Master of Science Thesis*, 2013.

[4] J. Bouwmeester. Delfi-n3xt mission definition, systems overview and general design approach. *Internal document: DNX-TUD-TN-0263 V2.3*, 2011.

[5] ECSS. Risk management. *ECSS-M-ST-80C*, 2008.

[6] ECSS. Space engineering: Verification. *ECSS-E-ST-10-02C*, 2009.

[7] Roozenburg en Eeckels. Produktontwerpen, structuur en methoden. *ISBN 9051897065, 9789051897067*, 1991.

[8] ESA. Technology readiness level. http://sci.esa.int/sci-ft/50124-technology-readiness-level/, 2016.

[9] J.Bouwmeester et al. Preliminary results of the delfi-n3xt mission. In *Proceedings of the 4S symposium 2014*, 2014.

[10] International Organization for Standardization. Space systems. *ISO standard 16290*, 2013.

[11] ISIS. Trxuv uhf-vhf transceiver manual. *Internal document: DNX-ISS-MA-0938 V1.2.3*, 2011.

[12] Delfi n3Xt team. Comms - deployment and antenna board (dab) design. *Internal document: DNX-TUD-TN-0715 V1.2*, 2012.

[13] Delfi n3Xt team. Mechs - solar panel deployment test report. *Internal document: DNX-TUD-TR-1009 V1.2*, 2012.

[14] Delfi n3Xt team. Mechs - mab deployment resistor hot spot location report. *Internal document: DNX-TUD-TR-1054 V1.0*, 2012.

[15] Delfi n3Xt team. Cdhs dssb software design. *Internal document: DNX-TUD-TN-1059 V0.1*, 2012.

[16] Delfi n3Xt team. Cdhs - obc thermal vacuum test plan and procedure. *Internal document: DNX-TUD-TP-1070 V1.2*, 2012.

[17] Delfi n3Xt team. Eps - basic consumption and efficiency test results. *Internal document: DNX-TUD-TR-1090 V1.0*, 2012.

[18] Delfi n3Xt team. Delfi-n3xt telemetry. https://webdata.tudelft.nl/staff-umbrella/Delfi/[DNX]Delfi-n3Xt/[DNX-TUD-TM]Telemetry, 2014.

[19] Delfi n3Xt team. Cdhs - software icd. *Internal document: DNX-TUD-IC-0861V4.38*, 2015.

[20] Delfi n3Xt Team. Delfi program. http://www.delfispace.nl/general/delfi-space-program, 2015.

[21] Delfi n3Xt team. Delfi-n3xt telemetry. https://webdata.tudelft.nl/staff-umbrella/Delfi/[DNX]Delfi-n3Xt/[DNX-TUD-TM]Telemetry, 2016.

[22] R. Schoemaker. Robust and flexible command and data handling onboard the delffi formation flying mission. *Master of Science Thesis*, 2014.

[23] SpaceTrack. Two-line elements. `https://www.space-track.org/`, 2016.

[24] M. A. Swartwout. The first one hundred cubesats: A statistical look. In *Journal of small satellites, Vol. 2, No. 2, pp. 213-233*. A. Deepak Publishing, 2013.

[25] M. A. Swartwout. Cubesats and mission success. Presentation CubeSat Developers Workshop, Saint Louis University, 2016.

[26] Delfi team. Delfi disk. `https://webdata.tudelft.nl/staff-umbrella/Delfi/`, 2015.

[27] Delfi team. Delfi-n3xt hardware logs. `https://webdata.tudelft.nl/staff-umbrella/Delfi/2.WorkingDirectory/#HardwareLogs`, 2015.

[28] Delfi team. Delfi server, 2016.

[29] Delfi team. Delfi mail box. `Delfi@tudelft.nl`, 2016.

[30] A. Tindermans. End-toend analysis and design of the satellite communication links. *Master of Science Thesis*, 2014.

[31] S. van den Berg. Fault-tolerant on-board computer software for the delfi-n3xt nanosatellite. *Master of Science Thesis*, 2012.

[32] S. van Kuijk. Application of aiv on cubesats. *Literature review*, 2015.

[33] M. Vos. Delfi-n3xt's attitude determination and control subsystem. *Master of Science Thesis*, 2013.