Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Aerospaziale

A.a. 2020/2021

Sessione di Laurea Luglio 2021

Development of a fault-tolerant software for a CubeSat Test Platform

Politecnico di Torino

Supervisors:

Prof. Sabrina Corpino

Eng. Fabrizio Stesina

Candidate:

Giovanni Cena

Abstract

This thesis deals with the work on ESA-uProp program sponsored by the European Space Agency (ESA) that aims at developing a facility for the verification and qualification of miniaturized electric propulsion systems using a CubeSat Test Platform (CTP).

One of the main objectives of the project is to assess the mutual interactions between the current CubeSat technologies and the new propulsion systems, in order to improve the Technological Readiness Level (TRL).

The thesis is focused on the first phase of design of ESA-uProp 3 project: the heritage of the previous project ESA-uProp 2, the analysis of the objectives and the drivers that define the product, the preliminary design, development tests and the beginning of integration of the prototype. In particular, this thesis analyses the aspects of hardware and software redundancy and functions distribution in order to improve the reliability and safety of the product. Different fault tolerant techniques are investigated and implemented to support the memory storage and the transmission of the data gathered by the onboard systems. Solutions on the hardware include passive, active and hybrid redundancies applied at component and part levels. The verification of these solutions is made on through a step-by-step approach: it means verification of the single modules one-by-one and then integration of all parts up to the final product assembly.

In the first chapter the motivations of this thesis are explained. The particular economic and technological environment of these years is emphasizing the importance of focusing the research on small satellites. There is a rapid overview of the main adopted reliability philosophies (i.e. fault avoidance and fault tolerant techniques) and the reasons that led to follow a Fault Tolerant Design as promising solution.

The second chapter is focused on the Fault Tolerant Design techniques. Main ones are classified, discussed individually and compared according with their possible application in the project. Different techniques of hardware, passive and active and hybrid, and information redundancy are detailed in order to select the most effective techniques for the ESA-uProp Project.

The third chapter is the core of the thesis because Fault Tolerant techniques are applied to two case studies, ESA-uProp 2 and ESA-uProp 3, that are presented in terms of general description and architecture. The focus is posed on the testing that includes the definition of the plan, the procedures, the setup, the execution and the discussion of the results for the conducted test campaign. It is demonstrated that the solutions applied to improve the reliability work properly because a reduced loss of information occurs both for storage in memory and the transmission of data, an improvement of the acquisition accuracy of the information from the onboard sensor and sensing circuits is observed, and the critical parameters are correctly managed. Finally, the integrated software is able to manage all the operative modes of the platform.

The thesis results allow to consider the onboard computer system ready for the final integration with the other subsystem and with the ground support system. In the next future, pre-qualification tests of the integrated CTP are planned and the final test campaign at ESA/ESTEC is planned for the end of 2021.

Contents

List of Figures

List of tables

Abbreviations

ADC: Analogue to Digital Converter

AIV: Assembly, Integration and Validation

CDH: Command and Data Handling

CDS: CubeSat Design Specification

COTS: Commercial Off-The-Shelf

CRC: Cyclic Redundancy Check

CTP: CubeSat Test Platform

DL: Data Logger

ECSS: European Cooperation for Space Standardization

ESA: European Space Agency

EP: Electric Propulsion

EPIS: Electric Propulsion Interface System

EPS: Electric Power System

GPS: Global Positioning System

GSE: Ground Support Equipment

HL: Hard Line

LEO: Low Earth Orbit

PPU: Power Processing Unit

PS: Propulsion System

RF: Radio Frequency

SEC/DED: Single Error Correction/Double Error Detection

SD: Secure Digital

SPI: Serial Peripheral Interface

TMR: Triple Modular Redundancy

TRL: Technological Readiness Level

UART: Universal Asynchronous Receiver-Transmitter

1. Introduction

1.1 Importance of small satellites in the modern contest

The space sector is gradually increasing its importance in last decades. From a purely scientific purpose it is opening a huge number of commercial opportunities. Already today, some common technologies from space sector shape the modern society, like the GPS or satellite communications or the weather forecasting service, but also some technologies derived from space sector, like thermal blankets or digital cameras.

The modern trend of investing on small satellites is increasing the number of commercial, government, and academic entities which invest on space projects in the world. This is due to different reasons: the miniaturization of technologies and payloads, the increasing feasibility of new commercial opportunities in space, the reduction of development time that leads to a faster investment response and the economic possibility of having a personal spacecraft in orbit.

Small satellites demonstrate their advantages in costs and development time. Less weight and volume lead to a minor amount of materials and parts, less labour for development and assembly and less amount of fuel to insert them into orbit. Typically, the launch represents a remarkable slice of cost in a space mission that can be pulled down with small satellites: it's possible adopting the ride sharing technique, that means to exploit a large vector that is launching a big satellite like a secondary payload, or directly using minor and cheaper vectors. The adoption of small satellites can also improve the same quality of the mission: for example, using a constellation of them despite of a large satellite can increase, at the same cost, the coverage, the reliability and in general is more flexible, in viewpoint of a possible reconfiguration.

Generally, a small satellite is a spacecraft whose wet mass is minor than 500 Kg. The wet mass is referred to the spacecraft and the propellant. Exist different categories of small satellites according to their wet mass and classified in this list.

Class	Wet Mass (kg)
Mini-Satellites	100 - 500
Micro-Satellites	10 - 100
Nano-Satellites	1 - 10
Pico-Satellites	0,1 - 1
Femto-Satellites	< 0,1

Table 1: Small satellite classification

A particular typology of small satellites, that falls into the categories of Micro and Nano satellites, are the CubeSats. These are characterised by a volume multiple of 1U, a standard volume unit of 10 cm x 10 cm x 10 cm. So a CubeSat can have a volume from 1U up to 12U or 24U. Typically the mass is less than 1,33 Kg per unit. The adoption of standard volumes allows to standardise also the launchers interfaces with them, or dispensers, reducing costs.

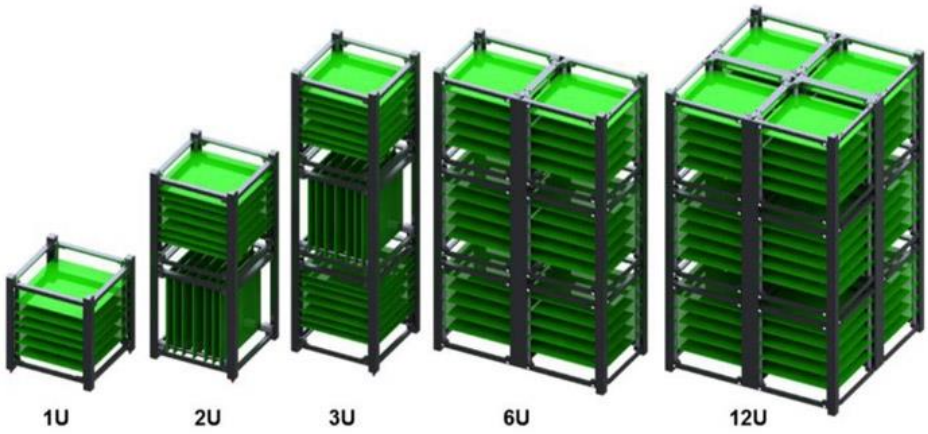

Figure 1: CubeSat volume classification

The term "CubeSat" was born in 1999, in California Polytechnic State University and Stanford University, that developed its specifications. Initially they are thought to be used for academic use and test platforms, but

because of their low cost, due to also by the wide use of cots, they called the attention of commercial agencies.

The use of small satellites, and in particular CubeSats, involves important technological challenges in miniaturizing different subsystems. The propulsion system is one of the most critical one. Applications range from orbital changes and maintenance, attitude control and desaturation of reaction wheels to drag compensation and de-orbit at spacecraft end-of-life.

Space propulsion can be enabled by chemical or electric means, each having different performance and scalability properties. Electric models are very interesting, in the viewpoint of low volume and mass, because of their high specific impulse that allows to better exploit the propellant, and reaching the same goal carrying on less mass. The micro-propulsion is one of the least developed aspect, but one of the most interesting one in a viewpoint of future goals: an increasing number of entities are working to improve the TRL of this fundamental aspect. The TRL, Technological Readiness Level, is a parameter that identifies the maturity of a technology.

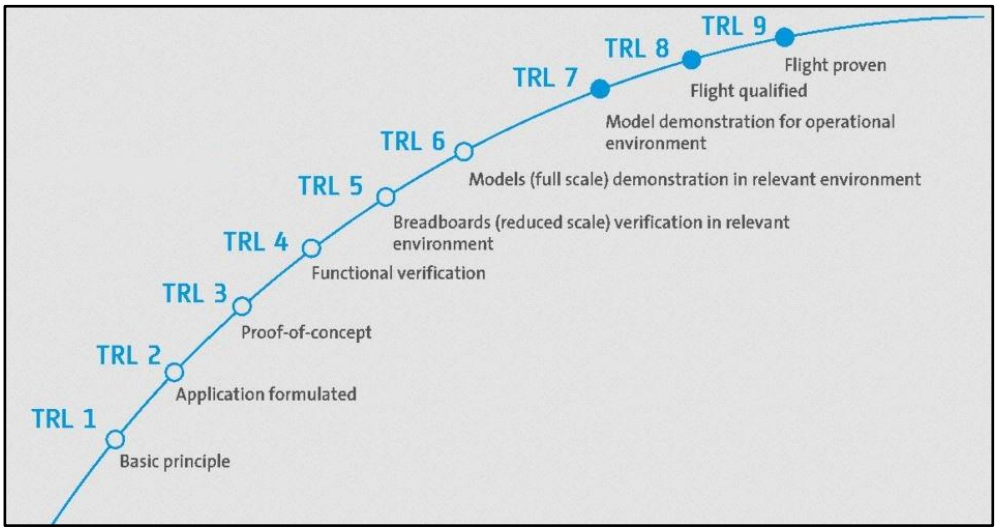

Figure 2: TRL step progression

Miniaturized electric propulsion technology has at the moment low TRL and little to no actual launches to space. To reach a satisfying level it is necessary to operate tests on ground and in space. One of the most fundamental aspect

and at the same time one the least investigated is the mutual interaction between these emerging technologies and small satellites.

1.2 ESA-uProp and thesis motivation

In this context is born the ESA-uProp program, a collaboration between ESA and Politecnico di Torino, that aims to design, produce and test the CTP, the CubeSat Test Platform, in order to test new technologies and prototypes of electric micro propulsors and evaluate the mutual impact trough different measurements.

My thesis, according to my personal contribution to the project, focuses on definition and verification of different techniques of fault tolerant design applied to small satellites in order to improve their reliability, in particular applied to software.

1.3 Fault Tolerant design choice

The reliability of a product, and in general its dependability that's composed by reliability, availability, maintainability, safety and testability, can be affected by the adoption of different techniques. These concern each phase of cycle of life of the product, from its design to its production and operating life.

These techniques can be classified in 4 classes, that are often complementary and the adoption of one often means only a larger use than others, that are always partially present supporting the main one.

- **Fault prevention:** it aims at minimizing the probability that a fault occurs. It depends to the technological level of design and production process from each single component up to system level. The high quality often means high cost. It minimizes the risk but doesn't completely eliminate it.

- **Fault removal:** it aims at detecting, isolating and correcting the faults that could emerge from each phase, from the formal verification in the design phase to testing in productive or operative phase. Dynamic or static procedures exist according to the need of execution of the system or not. During the operation phase they can be distinguished on-line or off-line test, according to the possibility of the system to work during the test or not. These techniques are strictly connected to the maintainability, that sometimes couldn't be possible, especially in space sector.

Techniques of fault prevention and removal are often collected under the name of fault avoidance.

- **Fault forecasting:** It aims at estimating the probability that faults occur and evolve into failures, in order to individuate critical elements and to identify the most suitable fault prevention, removal and tolerance techniques.

- **Fault tolerance:** It aims at assuming that faults could happen but, because expected, mitigating their effects. These are based on redundancy, which means duplicated some critical parts of the system.

Techniques of fault tolerance allow to use components of minor quality than a fault-avoidance approach. In particular in space sector the production is limited to one or two pieces and elements specifically designed and tested are very expensive. The major resilience to possible faults allows to use commercial components, called COATS, that leads to a consistent cost reduction, according to the point of view of small satellites.

As in many engineering problems, there is an optimal trade-off between these different techniques. In fact, sometimes redundancies produce less, not more reliability. Redundant safety devices result in a more complex system, more prone to errors and accidents. Redundancy may lead to underestimate responsibility among workers. Moreover, the increased number of components affects the volume, the mass and the energy consumption, which are all critical parameters for small satellites.

2. Fault Tolerant Design

2.1 Fault tolerant design definition

It is fundamental to understand some concepts of system reliability.

- **Fault** A fault is a defect in a system. The presence of a fault in a system may or may not lead to a failure. For instance, a system may contain a fault, but its inputs and state conditions may never cause this fault to be activated, so that an error never occurs and never exhibits as a failure.
- **Error** An error is a discrepancy between the intended behaviour of a system and its actual behaviour inside the system boundary. Errors occur at runtime when some part of the system enters an unexpected state due to the activation of a fault.
- **Failure** A failure (or misbehaviour) is an instance in time when a system displays a behaviour that is different with respect to its specifications. An error may not necessarily cause a failure. For instance, an exception may be triggered by a system due to an error, but this may be handled using fault tolerance techniques so that the overall operation of the system still conforms to the specifications.

The fault tolerance design is an approach that aims reducing the effects of possible faults. It means that the system continues working avoiding that a fault could become a failure. Sometimes, following a fault, the system could reduce its performance but in a predicted way, in order that mission could be completed: this is called “graceful degradation”. On an extreme way the system could also be required to fail-safe or fail-gracefully, that means if its function fails completely, this doesn’t have to provide damages at persons, properties or data.

There are three typologies of fault tolerance techniques.

- Some techniques aim at preventing misbehaviours: when a fault occurs, it is masked directly and doesn’t become a failure.
- Some techniques aim at detecting, isolating and correcting the errors. Typically, they are more flexible and efficient, but have also differences

from the first techniques, for instance they are often characterised by latency.
- Techniques that aim at making the system fail-safe.

In general, all techniques of fault tolerance are based on the concept of redundancy.

Redundancy is the insertion of elements that copies some functions of a system, in order to increase its reliability. It can be applied at different level: component, part or subsystem.

There are four major classes of redundancy:

- Hardware redundancy: multiplication of physical parts.
- Information redundancy: methods to detect and correct errors in data.
- Time redundancy: repetition of processes in case of fails.
- Software redundancy: multiplication of processes whose results should be the same.

2.2 Hardware redundancy

There are conceptually three kinds of hardware redundancy:

- **2.2.1 Passive techniques**: the faults don't need to be detected and no correction actions are needed, because when a fault occurs, its effects are automatically ignored through the technique of masking. It's based on the concept of voting. Some repeated modules, for example sensors, or memory devices, provide a value to a voter, that can compare them. The voter, basically, can evaluate the intermediate value, but excluding values too discordant, that correspond to failed modules.
  This is called NMR (N modular redundancy), whose most common version is the triple one, the TMR.

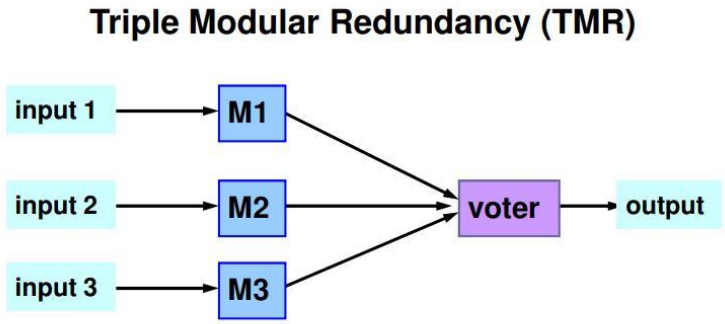

Figure 3: TMR schematic

If a higher level of tolerance is needed, more redundancy can be used: no more than one faulty module accepted for TMR or 2 faulty modules for 5MR. It's important to consider that a faulty module means that can have different faults inside and this method can be applied at different levels (gate, register, element, subsystem, etc.): if multiple fails can affect a subsystem, it's possible to avoid a failure if the entire subsystem is redundant.
The voter can also be implemented by a software.
This method is also characterised by different critical elements:

- Common mode fault: the TMR architecture fails if the same fault exists in all the modules that could be caused, for instance, by an error in design of the modules.
- The voter could fail: the solution is multiplying also the voter through a restoring organ.

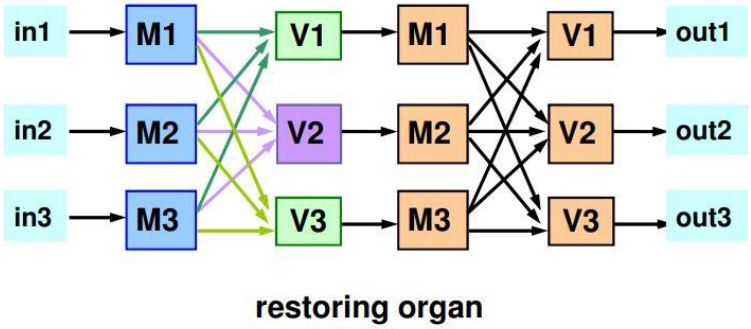

Figure 4: restoring organ schematic

- Timing issue: it's fundamental that voter compares values that refer to the same time.
- In some cases it is not possible to execute a voting: an alternative is known as flux summing.

- **2.2.2 Active or Dynamic techniques**: errors are detected and reconfiguration is applied. It means that after detection, the system identifies the location of the faulty module, recovers it to a correct state, replaces the faulty component and restarts the system (sometimes with a reduced capability). It's characterized by a period of latency between error detection and correction that leads to possible temporary misbehaviours. Compared with the passive redundancy, active techniques imply a lower cost, generally.
  It's based on the technique of "duplication with comparison", or "duplex": two modules provide their outputs to a comparator, that verifies their discordance, and in case it detects a fault.

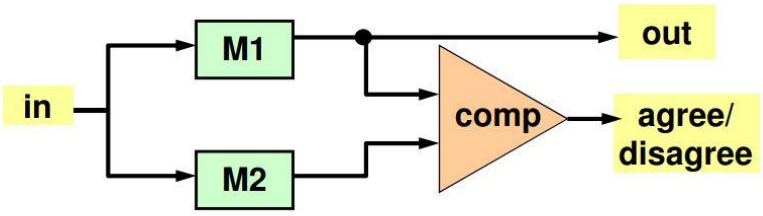

Figure 5: duplication with comparison

The most critical element is the comparative technique, that could be very complex. This architecture has also other weaknesses: comparator could fail, or common mode fault in modules could be not detected.
It's possible adding spares that replace those faulty modules to increase the accepted number of faults and so the general reliability of

the system. There are two possible ways of sparing: hot sparing means the replacing modules are still active and in a stand-by mode, while the cold sparing means they are turned off and must be activated before the use. The hot sparing has the advantage of reducing the time of latency, but involves a consume of energy or other sources, and aging, reducing the reliability of the stand-by modules when they are used.

- **2.2.3 Hybrid redundancy techniques:** they combine active and passive redundancy. They adopt the technique of masking while reconfiguration: on this way reliability is maximized avoiding temporary misbehaviours, but the hardware cost is quite elevated.
  These are the principal techniques:

  - NMR with spares: the voter allows the system working continuously, while a comparator identifies the faulty module and replace it with a spare. On this way future faults are accepted without becoming failures.

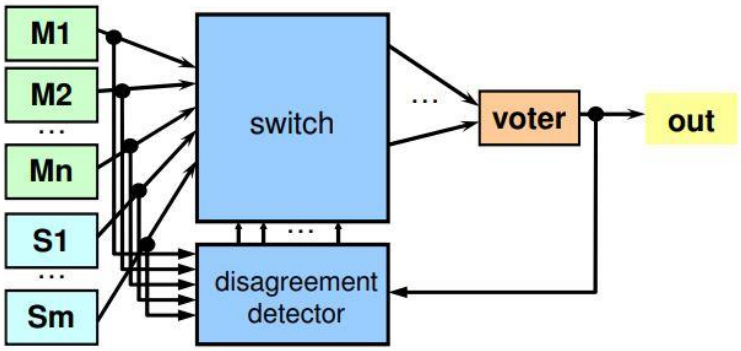

Figure 6: NMR with spares schematic

  - Pair-and-a-spare: it's composed by two duplex modules linked by a switch. When a fault involves a module of the first duplex the switch is set on the second duplex, and in the first can be acted

the error detection and correction avoiding a latency time in the system.

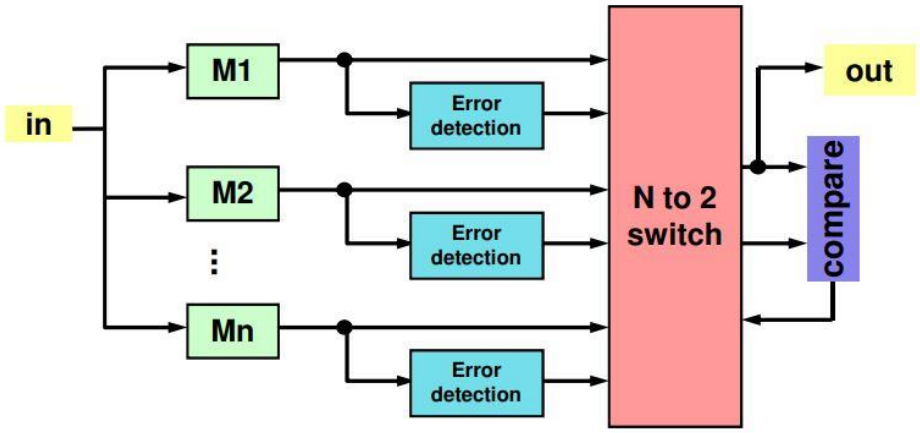

Figure 7: pair and spares schematic

- Triple-duplex: The TMR architecture avoids any misbehaviour while duplication with comparison allows the substitution of the faulty modules: in such a way that further faults can still be tolerated.

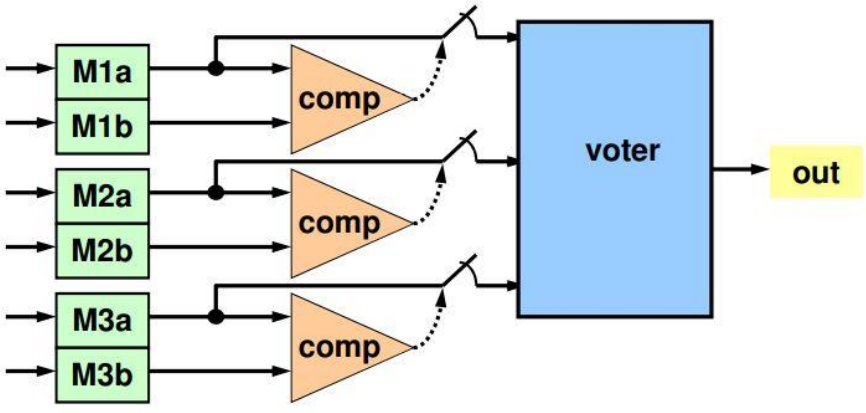

Figure 8: triple-duplex schematic

2.3 Information Redundancy

It's based on techniques to enrich an information with more bits in order to detect some errors, that can be generated in a hardware component, and in some cases to correct them. In general, a code, a set of rules, elaborates the data encoding it, that means transforming the data in a codeword. Then the codeword passes in the module that has to be hardened, for instance a memory or a communication device. At this point the codeword is decoded: the code recognises the data and verify if the rest of the codeword is the same as after the encoding, as expected. If the codeword is the same the data is correct, otherwise it has been corrupted, and in some cases, according to which are the differences between the codeword after the decoding and the expected one, the corrupted data can be corrected.

The choosing of a code is characterised by different parameters. A code could detect, and possibly correct, only some types of faults. Every code has a cost in term of additional information, hardware for encoding/decoding and computational request.

There are some common rules that regulate these codes:

- First Hamming theorem: If a code has distance k, it can detect all errors affecting less than k bits.

- Second Hamming theorem: If a code has distance k, it can correct all errors affecting less than k/2 bits.

  The distance of a code is k if any couple of words belonging to the code has a Hamming distance not lower than k.
  The Hamming distance, given two words, is the number of bits in the corresponding position having a different value in the two words.

The separability of a code is the property that a relative codeword has the added bits distinguished from the data bits.

Exist many kinds of codes for information redundancy. The main ones are the following:

- ### 2.3.1 Parity code

  It is one of the simplest codes. It consists in considering a certain number of bits, counting how many bits of value 1 there are, and adding a parity bit, whose value is according with the previous counter, at the end of the string of data bits. The parity bit can be set arbitrarily on even or odd parity.

  This code is separable and has a distance of 2: this means that it can detect only one faulty bit, including the parity bit, and correction is not possible. In case of error detection, the entire string is corrupted, and if possible it must be retransmitted.

  The hardening of the method depends by the length of string that it is applied on: if data strings are shorter the probability of multiple faulty bits in the same string is lower, but at the same time the cost in term of number of added bits is higher. The major cost results in more memory required, or longer packets to be transmitted or processed.

  The most critical element of this method is that a double fault, especially of two adjacent bits, is not detected. More complex versions of parity bit codes can be adopted to improve this weakness, for instance interlaced parity or overlapped parity.

- ### 2.3.2 m-out-of-n code

  This method is based on a codeword composed by n bits, and m of them have the value 1: a data word is analysed and are added a number of bits to reach the value of n, the firsts of them have the value 1 to reach a total number of 1s equal to n.

  This method is characterised by a quite elevated cost in terms of hardware and computation for encoding and decoding and in terms of memory because it duplicates the length of the data packet.

  Its distance is equal to 2, that means it can detect a single error without the possibility of correcting it but, differently from the parity code, it can also detect unidirectional multiple errors (all 0 become 1 and vice versa).

- ### 2.3.3 Duplication code

  As suggested by the name, it consists in using the data bits as control bits, that means a high cost in term of memory required. It's very easy to be implemented and all errors can be detected: it's critical only the

rare combination both the bits in the data part and control part are affected by an error simultaneously.
The two-rail code is a variant of the duplication code where the code bits correspond to the complemented data bits.

- **2.3.4 Checksum**

The string of data is divided into blocks. These blocks are added up and the result is placed at the end of the string. This method can only detect errors and correction is not possible: if the packet is corrupted it must be retransmitted, if possible.
The length of the control part is equal to the block. The most critical element is the possible overflow. It's so possible adopt some variant of the code. The residual checksum adds up the rest of the sum to control part. Otherwise it's possible to dedicate more memory to control bits with a doble-precision checksum.
The honeywell checksum is an ulterior variant of the code: It is the same as the previous, but the computation is performed in double precision adding couples of adjacent data in double precision. This has the advantage of reducing the risk of overflow and increasing the capacity to detect multiple faults.

Single Precision	Double Precision	Residue	Honeywell
0000	0000	0000	00000101
0101	0101	0101	11110010
1111	1111	1111	
0010	0010	0010	
0110	**00010110**	**0111**	**11110111**

Figure 9: checksum techniques classification

- **2.3.5 Cyclic Redundancy Check (CRC)**

This method is based on properties of polynomials and Galois field. In particular the binary code is a finite field of two element, a GF(2), where 0 and 1 are respectively additive and multiplicative identities. It means that addition is represented by the logical operator XOR: each

element equals its opposite, and subtraction is thus the same operation as addition. Multiplication is represented by the logical operator AND. A string of m+1 bits can be seen as a polynomial of degree m, called D(x).

$$101101 = x^5 + x^3 + x^2 + 1$$

Each CRC code is founded on a generator polynomial G(x) of degree r. The main operation in the encoding phase is the following division.

$$\frac{D(x) \cdot x^r}{G(x)}$$

The remainder of this division R(x) represents the control bits which follow the data string in the codeword.
When the codeword is received, it is divided by the same generator polynomial G(x): if the remainder is 0 the codeword has not been affected by error, otherwise it is considered as corrupted and it must be retransmitted, if possible.

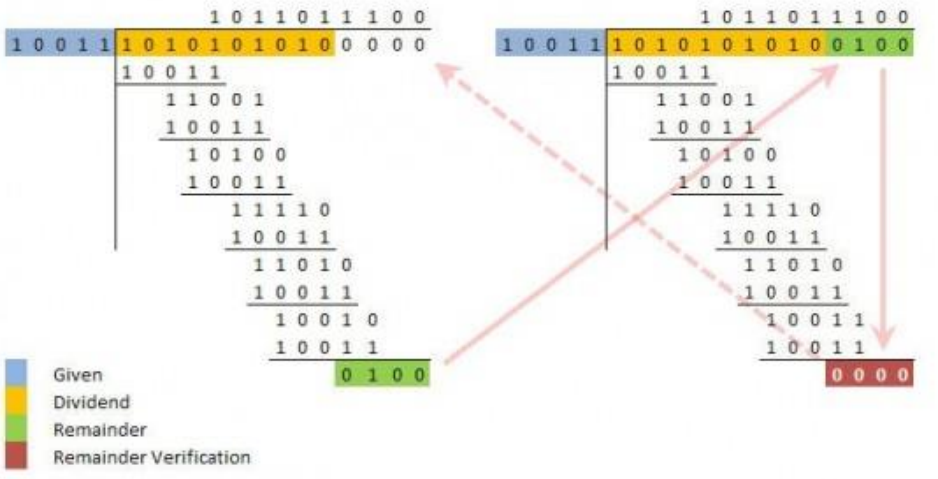

Figure 10: CRC visual explanation

Obviously the encoding and decoding unit must use the same generator polynomial G(x), which is typical of the code, and affects the error detection capabilities of the code.

An interesting class of generator polynomials G(x) are the primitive ones: the probability that a fault during the transmission produces a sequence whose remainder is the same than for the original sequence (aliasing) is proportional to 2^{-r}.
A primitive polynomial is a polynomial of degree r which divides the polynomial x^T+1 (with $T = 2r - 1$) and does not divide any other polynomial x^S+1, $0 \leq S < T$.

2.3.6 Modified Hamming Codes

This code can be seen as a complex variant of a parity bit code, in particular the overlapping one.
The codeword is formed inserting some parity bits among the data bits: numerating the bits from the left, the control bits are in the positions which are multiple of 2 (1, 2, 4, 8…). Each parity bit refers to alternated blocks of bits of the same length of their position: for instance the parity bit in position 4 (p4) refers to bits 4-5-6-7, a block of 4 bits isn't considered, then are considered the bits 12-13-14-15, and so on…

Bit position		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8	d9	d10	d11
Parity bit coverage	p1	✗		✗		✗		✗		✗		✗		✗		✗
	p2		✗	✗			✗	✗			✗	✗			✗	✗
	p4				✗	✗	✗	✗					✗	✗	✗	✗
	p8								✗	✗	✗	✗	✗	✗	✗	✗
	p16															

Figure 11: Hamming Code bits

The number of control bits required k depends by the number of data bits n, following this relation:

$$2^k \geq n + k + 1$$

When the codeword is received, the parity bits are recalculated on the data bits. The comparison operation between the read code and the

expected code is called “syndrome”. If there are not differences, no error is occurred.
The Hamming code has a distance equal to 3: this means it can correct a single error or detect up to double error. There is a critical element: if the code is built to correct a single error could sometimes act to correct a double error but in an erroneous way, that leads to consider then correct a packet that is actually corrupted. It’s necessary to implement an ulterior control bit, placed at the top or the end of the codeword, that verifies the parity on all the data bits. On this way it’s possible to distinguish between a single or double error, because the distance of the code is increased to 4. This modified Hamming code is a SEC/DED code (Single Error Correction/Double Error Detection).
After the syndrome there are five possible situations:

- The parity bit and the control bits are all correct, so no error occurred.
- Only one control bit (parity bit included) is not correct, so the error is the control bit itself. Detection and correction are possible, and data bits are correct anyway.
- The parity bit and some control bits are not correct, so a single bit error occurred. It can be detected and corrected. The sum of the position of the erroneous check bits gives the position of the faulty bit.
- The parity bit is correct, while some control bits are not, so either a double error or a control bit error occurred. It cannot be corrected.

The real strength of this code is that an autonomous action of correction is possible, without an operator’s action is needed and without interrupting the flux of information.
It’s possible to mark packets that has been corrected for a future check and evaluate if a retransmission is needed, if possible.

3 Case Study: ESA-uProp program

3.1 ESA-uProp program

3.1.1 general description

ESA-uProp is a program developed by ESA in order to improve the application of electric micro-propulsors on small satellites. It's composed by a series of projects assigned to Politecnico di Torino: the purpose is the design of platforms based on small satellites of different sizes according to CubeSat standards in order to fit with a wide range of micro-propulsors. As shown in the introduction, CubeSat is a miniaturized satellite which volume is a multiple of a Unit, and a Unit is a volume of 10 cm x 10 cm x10 cm. For this reason, the final products are called CubeSat Test Platforms or CTP [7].

The test platform will be used for testing miniaturised propulsion systems integrated in the test platform, with the final goal of assessing the effects of operations and interactions between the propulsion system and the platform. Interactions between the propulsion systems and other onboard subsystems are hard to be modelled and analysed through simulation, but the effects of operations of electric propulsion systems within a small platform must be assessed in order to validate the spacecraft design and mission operations. Moreover, the TRL of miniaturised electric propulsion systems is still low and need to be raised to enable future nanosatellite missions. These two objectives can be pursued through the definition and implementation of a test campaign, based on a platform able to host different electric propulsors without major modifications.

3.1.2 Development

It's possible to define different phases that lead to different goals. The Phase 1 is oriented to define the feasibility and the design and can be assigned two main objectives:

- To demonstrate feasibility of a Test Platform to host Electric Propulsion (EP) systems.

- To design and manufacture a low-cost prototype/EM of a platform (CTP) [6] to host EP system.

The Phase 2 is oriented to the operational part and is characterised by three main objectives:

- To assess the mutual effects of EP system and CTP [9]
- To integrate and verify in vacuum CTP with a selected EP system
- To identify of a set of procedures for the AIV of an all-electric CubeSat

The design of a complex engineering system is a challenge. The best approach is to follow the V model.

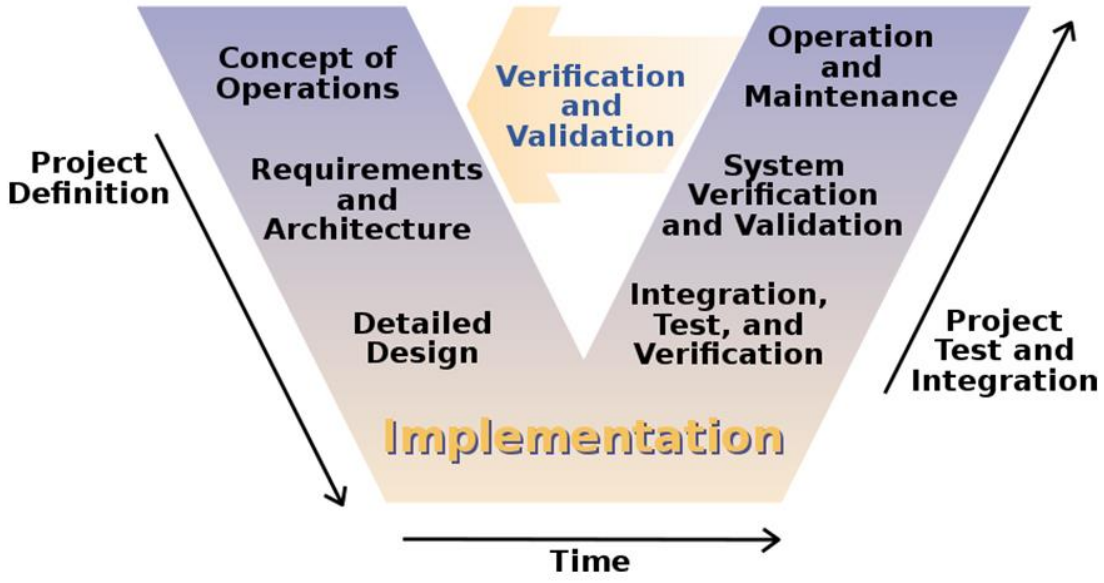

Figure 12: V model

The V-model is the graphical representation of the sequence of steps to be taken in systems development lifecycle. It shows the activities to be performed and the results that must be produced during product development. The left side of the V represents the definition of requirements, allocation of functions, and creation of system design. The base of the graph represents the actual implementation of the system (production). The right

side of the V represents the integration of parts and their validation and verification and the operative phase. During the phase Top-Down there is a conceptual passage from the customer to the supplier and vice versa during the phase Bottom-Up, in a sort of cycle that starts from the customer need towards the customer satisfaction.

3.1.3 Drivers

As in many engineering problems, the real challenge is finding the best solution as possible through a process called optimisation. The system engineer focuses efforts on assessments to optimize the overall design, and not favour one system/subsystem at the expense of another. In particular the entire design is influenced by a large number of elements like parameters, constraints, and limits that sometimes are in contrast, such as the quality of components and the cost of them. The most significant parameters are called "drivers" and have a great influence in the decision-making processes.

In ESA-uProp contest the drivers are the following [6], [7]:

- **Safety.** The CTP shall be compliant with safety requirements. In particular, the following aspects shall be considered, at platform level:
    - Leakage and contamination protection: protection of the CTP from chemical residuals and plume released by the thruster.
    - Over-voltage and over-current protection: protection of the electrical loads from undesired changes of voltage and current.
    - Thermal protection: isolation of the parts that may provoke changes of the temperatures (overheating/under-heating) inside the platform.
    - Electro-magnetic interferences protection: protection of CTP and propulsion system from mutual and external electromagnetic interference.

  Fault avoidance (high-quality parts and appropriate design margins), and fault tolerance (redundancy) approaches shall be implemented at least for safety-critical functions, in particular the system should guarantee a fail-safe approach in case of multiple failures.

Notwithstanding the intrinsic safety level of the CTP, the test operators shall be able to take control of the test at any time.

- **Reliability.** The CTP shall guarantee the test execution with a certain level of reliability. In particular the system should be tolerant to one failure, and at least should assure graceful degradation. For this reason, no single failure points shall exist.
  An onboard autonomous Failure Detection Isolation and Recovery (FDIR) system would well serve the purpose of enhancing reliability (and safety) of the CTP.

- **Autonomy.** The CTP is intended for use in a test facility with adequate GSE and skilled operators to conduct the test. However, a high degree of autonomy of the CTP can be foreseen in order to increase the value of the project in terms of fidelity, and to reduce the effort and workload of operators, thus pursuing cost reduction of CubeSat test campaign. Regarding autonomy, two main extreme options exist: 1) the CTP is fully autonomous and has full authority over the test run, or 2) operators have total control over the test, and the CTP only performs commanded functions and returns data as required. Hybrid solutions between these extreme options can be implemented in order to optimise test execution and to comply with safety requirements.

- **Flexibility.** The CTP is intended to host a variety of miniaturised propulsion systems and to run a wide range of tests. With this regard, the design of the CTP shall consider the capability of:
  - handling (wide) ranges of electrical powers and operative voltages.
  - managing different communication protocols.
  - managing different data and commands.
  - providing mechanical interfaces (between the platform and the propulsion system) able to adapt to the specific system of interest.

  A modular architecture can well serve the purpose of flexibility and shall be pursued throughout the design process.

- **Accessibility.** The CTP shall guarantee easy physical access to onboard systems during all test phases (setup/preparation, run, post-test), and for maintenance activities between subsequent test sessions. From the mechanical point of view, it is required that the structure of the CTP features access ports, and/or it is an "open" structure with easy mounting panels. The CTP shall also be interfaced with GSE through hard-lines for data and commands exchange for all test phases.

- **Cost.** According to the CubeSat philosophy, the CTP shall make use of low-cost technology and processes. It's largely suggested the use of cots and commercial parts to better understand the interaction between propulsion system and components that are usually on small satellites. Cost reduction represents a stakeholder constraint and influence customer satisfaction.

3.1.4 ECSS standards

ESA promoted the development of common standards for space projects and according with several international organisation founded the European Cooperation for Space Standardization (ECSS). This initiative developed a coherent, single set of user-friendly standards for all European space activities in order to minimise life-cycle cost, while continually improving the quality, functional integrity, and compatibility of all elements of a space project. This goal is achieved by applying common standards for project management and for the development and testing of hardware and software.

Thanks to their generality, they are quite flexible to permit customers and suppliers, small and big company as well as universities or agencies to use their own means and resources to accomplish with the standards. On the other hand, they provide a controlled and unified method, valid for all the European countries, to tailor all the projects in a precise sequence of standardized processes.

3.1.5 AIV – Assembly Integration Verification

One of the main process to guarantee a certain quality level and reliability degree is the definition and performance of the AIV programme for the CubeSats Test Platform (CTP). This is based on requirements and associated verification matrix [7]. The verification plan [10]:

1. states the objectives and the scope of verifications.
2. describes and sequences all verifications.
3. specifies the verification means/test facilities.
4. manages the execution of all verifications.

The definition of the verification sequences follows two main considerations:

1. the detection of potential failures and defects as early as possible in the development of the project.
2. the procurement sequence of the hardware.

The verification process of a CubeSat cannot disregard the peculiarities of this class of satellites:

- simplicity, about both the adopted process and the technology solutions.
- use of COTS equipment.
- low-cost project.
- fast delivery and reduced time consuming for testing.

Notwithstanding this context, CTP verification plan tries to follow the actual verification plan methods applied to industrial/commercial satellites. The ECSS standards have been considered whenever possible.

The verification is executed by one or more of the following verification methods:

- **Analysis**. A verification method which entails performing a theoretical or empirical evaluation by accepted analytical techniques. The selected techniques may typically include systematics, statistics, qualitative design analysis, modelling and computer simulation.
- **Inspection**. A verification method that determines conformance to requirements for constructional features, document and drawing

conformance, workmanship and physical conditions without the use of special laboratory equipment, procedures or services.

- **Testing**. A verification method wherein requirements are verified by measurement of product performance and functions under various simulated environments.
- **Review of design**. A verification method using validation of previous records or evidence of validated design documents, when approved design reports, technical descriptions, and engineering drawings unambiguously show that the requirement is satisfied.

Testing has been the preferred verification method with the lowest risk, but the tailoring of the test program should also assure that a cost-effective AIV is achieved.

The verification campaign is performed through a step-by-step approach at different levels of product decomposition. ESA-µProp product requires verification at equipment, subsystem, and system level.

During the verification campaign two main stages can be defined:

- **Development verification**. The objective of the first stage of development verification is to support the design feasibility and to assist in the evolution of the design. Development verifications are used to validate the design concepts using appropriate models, usually virtual (digital) models and mock-ups/dummy. Most verifications are performed via analysis and review of design rather than testing and inspection. In the second phase of the development verification, the aim is the formal demonstration of the design through representative hardware and software tests that confirm the expected functionalities and performance are reached. Testing has been preferred when hardware and software are available and, in case, they can be substituted by simulation made through virtual models.

- **Pre-Qualification verification**. The purpose of pre-qualification verification is to demonstrate that the items perform satisfactorily in the laboratory environment and prepare it for future verification in the

intended environment. The pre-qualification testing shall be conducted on dedicated engineering to qualification models (EQM) that will be used for the whole test campaign. Hardware and software are representative of the end item configuration in terms of design, materials, tools, and methods. Most part of the pre-qualification activity is done by testing through a step-by-step verification campaign that confirms the capability of the single equipment, the subsystem up to the CTP and the CTP integrated into laboratories.

3.2 ESA-uProp 2

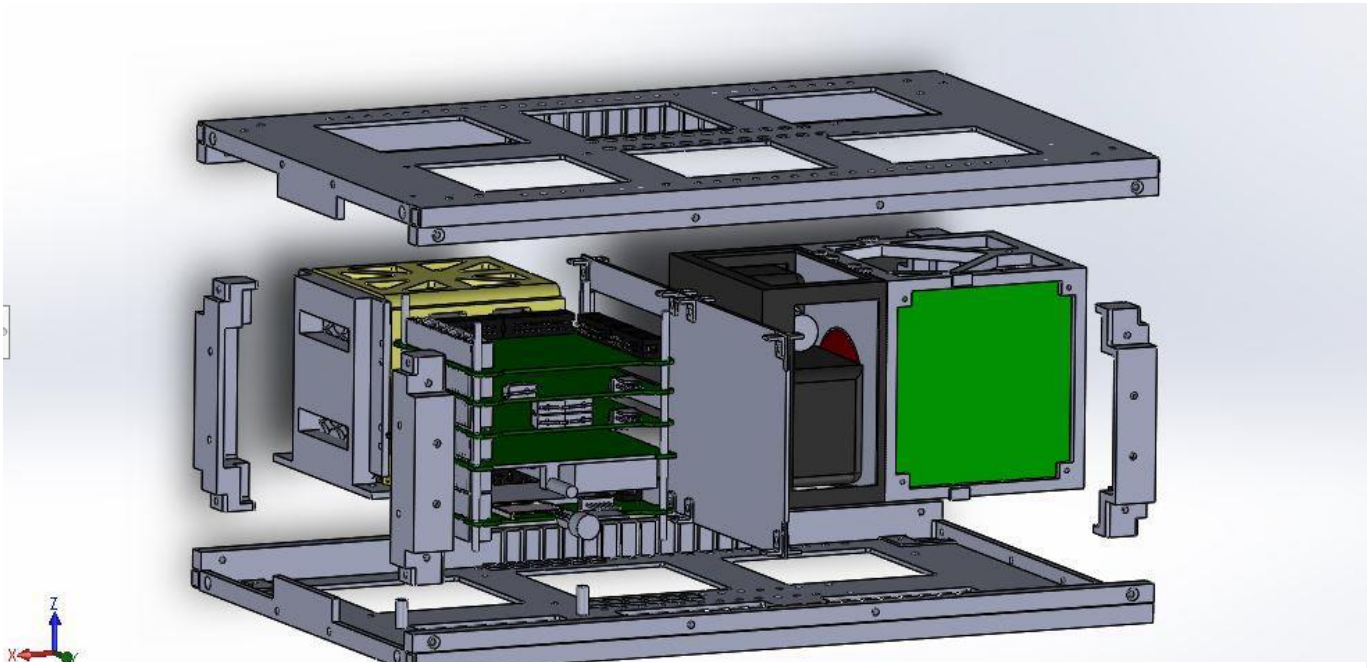

Figure 13: ESA-uProp 2 architecture

3.2.1 Architecture

The platform features an Al-alloy 6U structure, which a Propulsion Box hosts the propulsion system (up to 4U), and a Service Module contains the on-board avionics (1U), and battery packs(1U) [6].

The Propulsion System includes the Thruster, the Power Processing Unit (PPU) and the Propellant Feed System (PFS), and the propellant tank.

The avionics is constituted by the on-board computer for command and data handling functions, the electrical power system (PCDU and battery, no solar panels), and the communication module (UHF for housekeeping and experiment data) and two boards that constitutes the Electric Propulsion Interface System (EPIS). EPIS provides the interfaces of CTP towards the EP system and the instruments and devices to measure the parameters for assessing the mutual interactions between CTP and EP system. Two main parts constitute EPIS: the Data Logger (DL) and the High-Power Management System (HPMS). The avionics boards are in-house developed electronic boards resulting representative of the basic CubeSat technology. Data Logger gathers all the information about the radiation and thermal environment and the power consumption of the EP system. The HPMS supplies electrical power at a regulated voltage to the EP system using the energy of two dedicated battery packs; battery packs are recharged thanks to an external source and recharging control circuits.

Two lines guarantee communications between the platform and operators: a RF link in UHF band [8] and a wired serial line that directly connect the on-board computer with the Ground Support System. Command & Data Handling is based on ARM-9 microcontroller that manages data and commands time, operations and on-board failures. Sensors and acquisition circuits provide the information (e.g. voltages, currents, temperatures, magnetic fields, and electrical fields).

Electrical Power System is constituted by a board that controls and distributes power to the other subsystems and manages the avionics battery packs recharging. Batteries (both the Avionic Battery packs and the Propulsion Battery packs) are installed in the second unit of the Service module and can be recharged during the test thanks to an external line connected to GSE through umbilicals.

The structure is fully compliant with the CDS in terms of external geometrical interface and material (apart from surface coatings and treatments). The structure is constituted by two truss-like parts joined together through four brackets and closed by panels. The internal layout can be adapted depending on the specific test. For the first application, a bulkhead is fixed to separate the propulsion box from the rest of the platform. The PS thruster is mounted on this bulkhead, with the thrust axis along the X geometrical axis of the satellite. The PPU and PFS (including propellant tank) are also located in the propulsion box. The avionic system is enclosed in an avionic box and interfaced with other onboard systems and GSE through connectors.

3.2.2 Redounded memories

The CTP saves all data that acquires (CTP telemetry and PS telemetry) in their internal memories. It has been evaluated to equip the CDH board with two different memories: the integrated memory of the processor, a flash memory, and a second external removable memory, a SD card. Motivations are multiple. It represents an easy and fast way to transfer collected data for the post analysis process, moreover this technique improves the general reliability of the project.

First of all, it represents a technique of hardware redundancy, in particular a passive technique. If one of the memories is affected by a failure, the other one is still working and no misbehaviour occurs.

On the other hand, it's an example of information redundancy. Same data are saved both in flash and in SD memories, so they can be compared during a post process analysis.

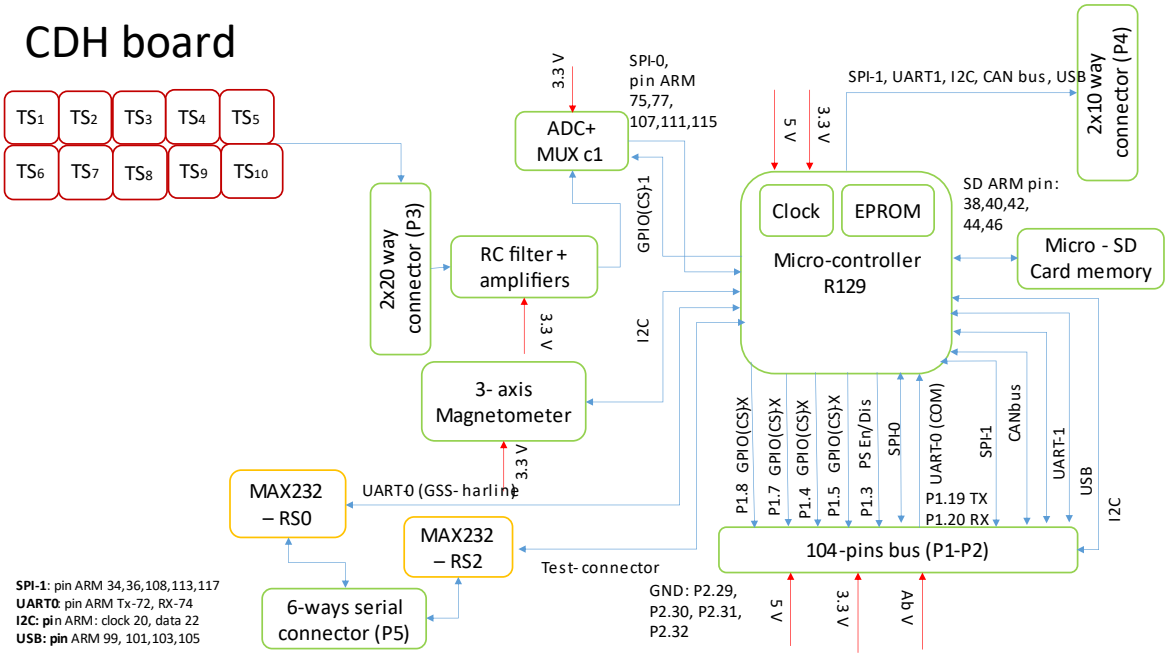

Figure 14: ESA-uProp 2 CDH board electric schematic

3.2.3 Watchdog

This driver allows to detect software anomalies and reboot the system if any occur. It has two main functions:

- Automatically rebooting the system if the main loop period is bigger than a determined value that would mean a failure occurred during software operations.
- Forcing a reboot in specific circumstances (operator command, critical parameters management).

The software set the watchdog to with an established time period. The system during his loop pings the system. The loop lasts roughly a certain period of time that is largely lower than the watchdog one. If an error occurs during software operations and watchdog is not pinged in time, it automatically reboots the system.

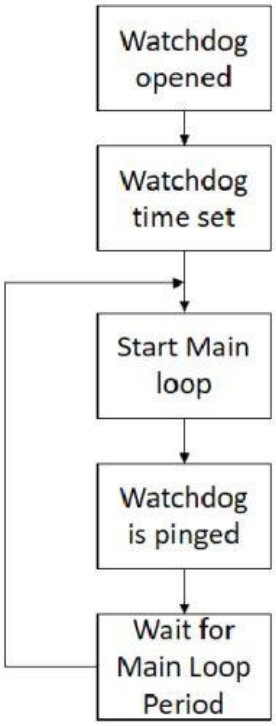

Figure 15:ESA-uProp 2 Watchdog management

This device has been tested for several different time periods of both the watchdog and loop. The watchdog works as expected. This means that:

- if the Main loop period is shorter than the Watchdog time, the system is never rebooted.
- if the Main loop period is longer than the Watchdog time, the system is always rebooted.
- if the two times are equal, the system is rebooted, this is likely because the loop period is actually a slightly higher than expected and because the watchdog timer is set prior of entering the Main loop.

This is an application of time redundancy. In particular it exploits the time to detect possible anomalies in software operations that could lead to a lack of control on the system and a consequent risk for the system itself. The reboot is the corrective actions that system automatically attempts to correct the failure.

3.2.4 HL and RF

CTP exchanges the information (data and commands) with GSS through the COM SYS. Two main links are available: a wired link that directly connects C&DH micro-controller GSS computer and a RF link with the RF equipment of GSS.

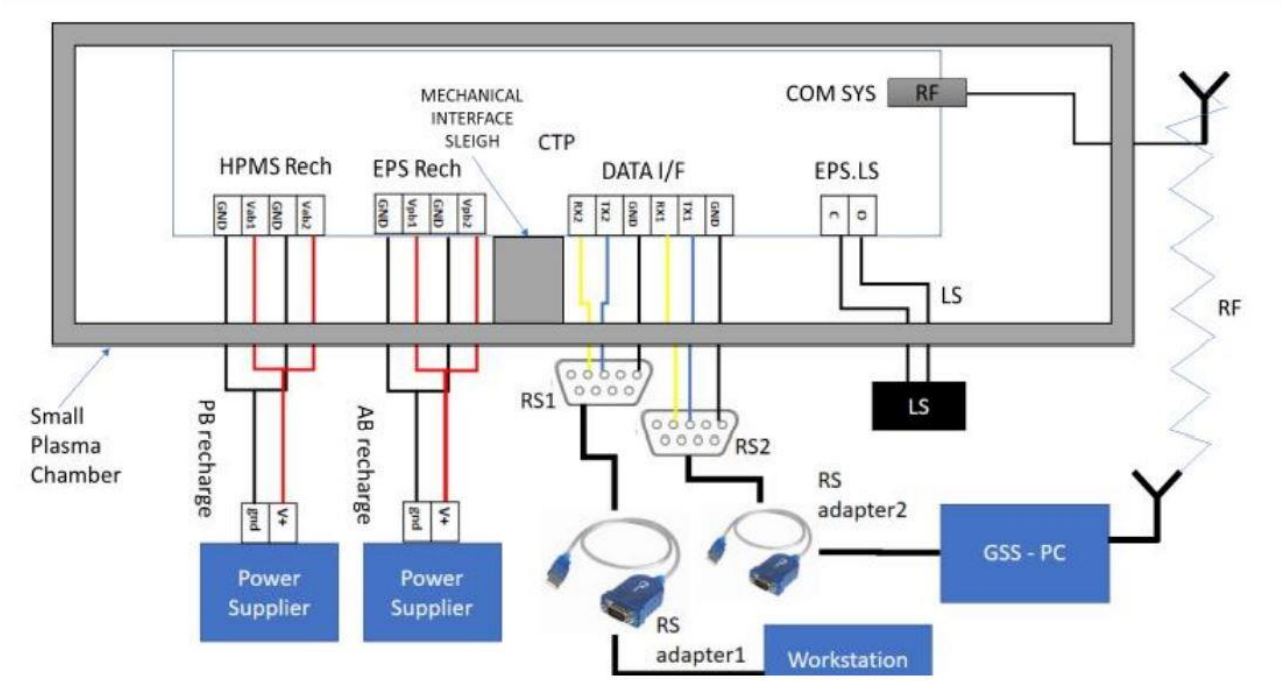

Figure 16: ESA-uProp 2 external interfaces

The RF links allows to test this technology in the unique environments that the CTP is supposed to operate. It is possible to evaluate the interaction between an electric micro-propulsor and the CommSys, that is the only way to communicate during a normal space mission. On the other hand, the CTP is designed to operate not in space, but in laboratories: this guarantee the possibility to adopt a physical wire to communicate with it. This feature provides a relevant increase in reliability level. This form of hardware redundancy allows to test RF communications but always guarantee a link with the CTP, avoiding lack of control in every moment. Both the ways have

been tested both in downlink (sending data packets) and in uplink (sending commands towards the CTP).

The HL coincide with the UART2 debug line. UART (Universal Asynchronous Receiver-Transmitter) is a serial asynchronous communication protocol generally used to allow exchange of information (bytes) between dedicated peripheral and/or computers. This makes communications easily performed by means of "printf" functions. Thus, this unit simply consists of several "printf" to show in a direct graphic way all the required information. The frequency of transmission can be changed.

The board communicates with the RF through the UART0 line. The structure of the test is similar to the one of the HL but, instead of the "printf" function, it makes use of the "write" function to send the string of data to CommSys. Compared to HL it is possible to notice that the header and closer bits of the KISS.AX25 protocol have been added, as well as the 4 control bytes computed with the check sum algorithm.

3.2.5 Check Sum

For a successful data transfer, it's required that the target system receives the same data sent by the source system. However, data may get corrupted while being transferred from one node to another.

One of the most common method to verify the correct transmission of a packet is the Check Sum, which is a validation technique. Each byte of the string to be sent is summed in a 32 bit variable. The resulting 4 bytes are sent at the end of the string.

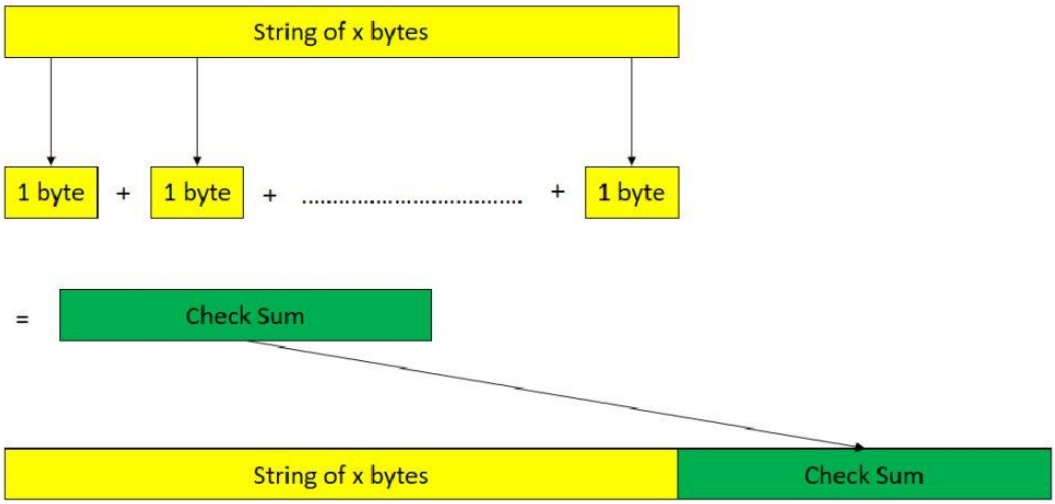

Figure 17: ESA-uProp 2 Checksum

This technique is applied only to RF communications because they are more probable to be affected by disturbances, while HL is considered safe enough. When string is received, it is simply validated performing the same algorithm on the received data bytes and comparing the result with the received 4-bytes check sum. This method requires only 4 bytes, independently the size of the string it is applied on: for this reason, it has been applied on communications, where it is crucial to avoid wasting space. On the other hand, it allows only to detect the presence of multiple errors, but without the ability to identify and eventually correct them. In case of detected corrupted packets they must be discarded and they must be sent again.

3.3 ESA-uProp 3

3.3.1 Architecture

The 12U CubeSat test platform (CTP) design is updated with the objective of testing at system-level (i.e. CubeSat) a variety of electric propulsion systems. The test platform will be used for testing miniaturised propulsion systems integrated in a test platform, with the final goal of assessing the effects of operations and interactions between the propulsion system and the platform.

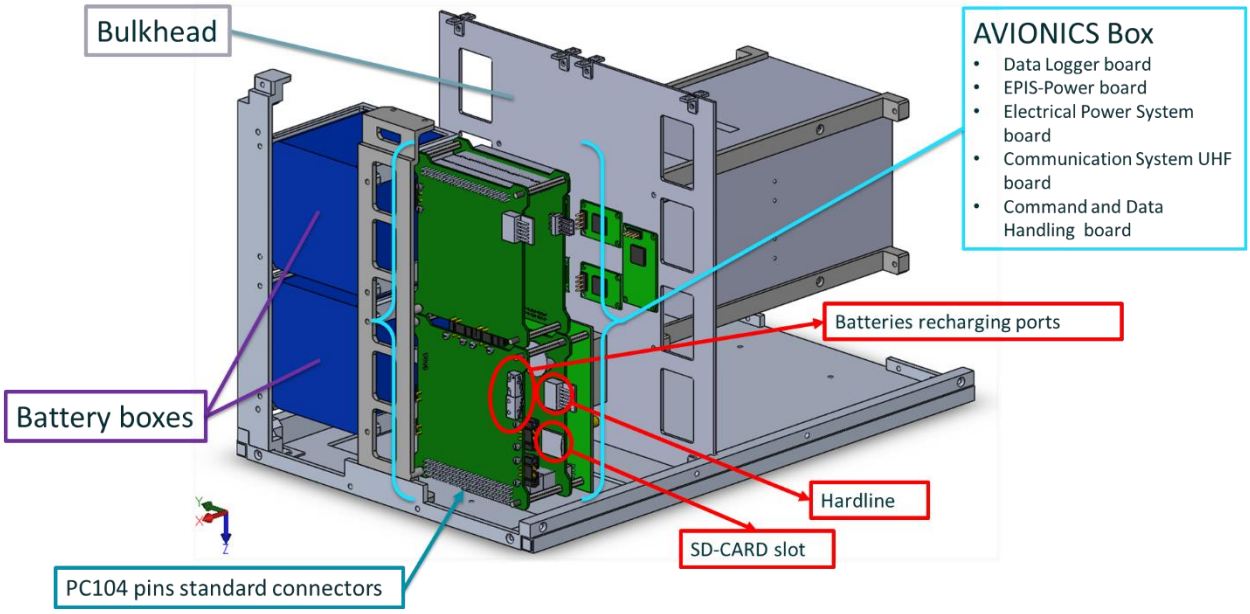

Figure 18: ESA-uProp 3 architecture

Among the enabling technologies for the enhancement of nanosatellite capabilities (e.g. high data rate communication, high accuracy attitude and orbit determination and control, steerable solar arrays), propulsion represents a key technology towards innovative applications and unprecedented missions based on small platforms. To date, very few CubeSats have flown in space featuring propulsion systems, thus very few data are available on propulsion systems performance in the operative conditions. Electric Propulsion (EP) systems are gaining interest for application in nanosatellite, especially for beyond LEO missions, and many

developments are ongoing on this technology. Consequently, a growing need exists for mission and satellite designers and engineers to understand the interactions between EP systems and the host spacecraft.

Interactions between Propulsion Systems (PS) and other onboard subsystems are hard to be modelled and analysed through simulation, but the effects of operations of EP systems within a small platform must be assessed in order to validate the spacecraft design and mission operations. Moreover, the TRL of miniaturised EP systems is still low and need to be raised to enable future nanosatellite missions. These two objectives can be pursued through the definition and implementation of a test campaign, based on a platform able to host different EP systems without major modifications. Main drivers for the platform design are: flexibility/adaptability of interfaces (mechanical, electrical, and data) towards propulsion system, accessibility, simple manufacturing and assembly, low-cost development. Platform requirements have been drawn through functional analysis developed from system-level up to component-level functions, considering all constraints imposed by applicable documents. Product tree has been developed though application of Quality Function Deployment method and N2 diagrams in turn, up the required level of system decomposition.

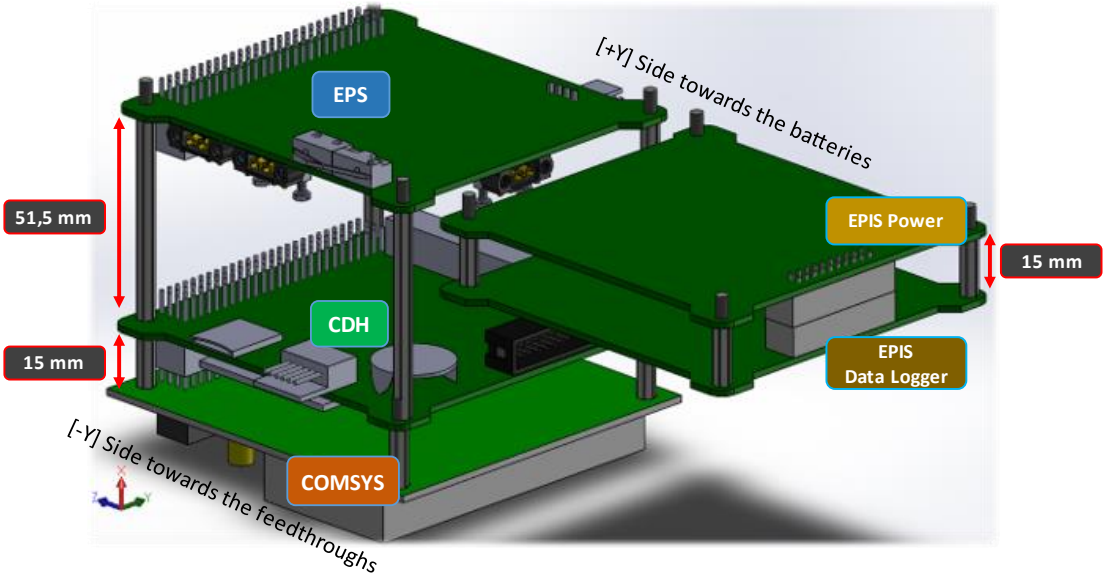

Figure 19: ESA-uProp 3 avionics

The platform features an Al-alloy 12U structure, which a Propulsion Box hosts the propulsion system (up to 8U), and a Service Module contains the on-board avionics (2U), and battery packs(2U). The Propulsion System includes the Thruster, the Power Processing Unit (PPU) and the Propellant Feed System (PFS), and the propellant tank. The avionics is constituted by the on-board computer for command and data handling functions, the electrical power system (PCDU and battery, no solar panels), and the communication module (UHF for housekeeping and experiment data) and two boards that constitutes the Electric Propulsion Interface System (EPIS). EPIS provides the interfaces of CTP towards the EP system and the instruments and devices to measure the parameters for assessing the mutual interactions between CTP and EP system. Two main parts constitute EPIS: the Data Logger (DL) and the EPIS Power. The avionics boards are in-house developed electronic boards resulting representative of the basic CubeSat technology. Data Logger gathers all the information about the radiation and thermal environment and the power consumption of the EP system. The EPIS and the EPS supply electrical power at a regulated voltage to the EP system using the energy of two dedicated battery packs; battery packs are recharged thanks to an external source and recharging control circuits. Two lines guarantee communications between the platform and operators: a RF link in UHF band and a wired serial line that directly connect the on-board computer with the Ground Support System. Command & Data Handling is based on ARM-9 microcontroller that manages data and commands time, operations and on-board failures. Sensors and acquisition circuits provide the information (e.g. voltages, currents, temperatures, magnetic fields, and electrical fields). Electrical Power System is constituted by a board that controls and distributes power to the other subsystems and manages the avionics battery packs recharging. Batteries are installed in 2 units of the Service module and can be recharged during the test thanks to an external line connected to GSE through umbilicals.

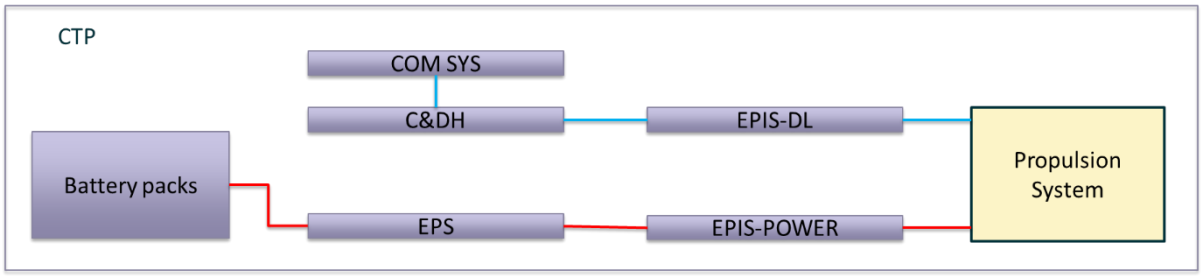

Figure 20: ESA-uProp 3 internal interfaces

The structure is fully compliant with the CDS in terms of external geometrical interface and material (apart from surface coatings and treatments). The structure is constituted by two truss-like parts joined together through four brackets and closed by panels. The internal layout can be adapted depending on the specific test. For the first application, a bulkhead is fixed to separate the propulsion box from the rest of the platform. The PS thruster is mounted on this bulkhead, with the thrust axis along the X geometrical axis of the satellite. The PPU and PFS (including propellant tank) are also located in the propulsion box. The avionic system is enclosed in an avionic box and interfaced with other onboard systems and GSE through connectors.

3.3.2 CRC

Reasons

A modification on the CTP software is the substitution of the checksum validation method with a stronger Cyclic Redundancy Check (CRC) code. The main reason is the major reliability of the CRC code.

Data corruption errors are classified based on the number of bits altered during transmission. An alteration can happen due to interference, which can change the shape of the signal. These errors are classified into two categories.

A Single-Bit error occurs when there's a change in one bit in a data byte:

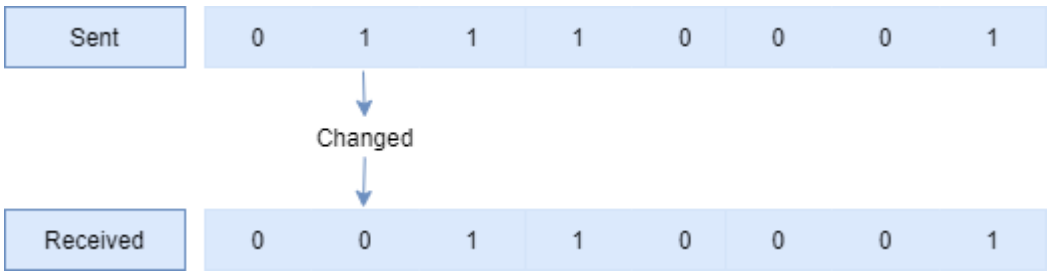

Figure 21: single bit error

A Burst error occurs when there's a change in more than one bit in a data byte:

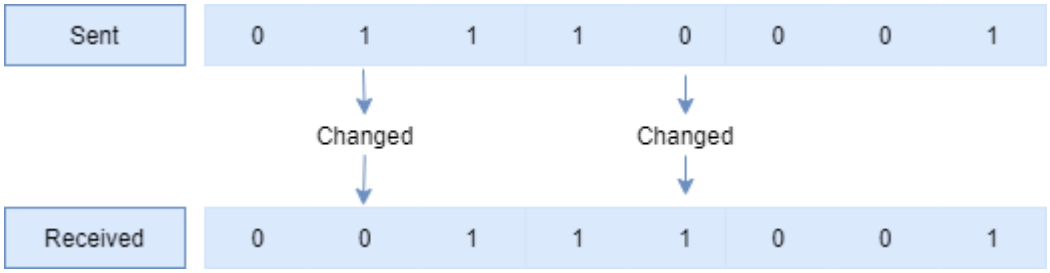

Figure 22: burst error

According to checksum, any 2- bit errors in the same bit position of a block are undetected, and error detection performance is insensitive to data values. If any bit position has an odd number of bit errors, the checksum will detect that there is an error, improving average case performance compared to a single parity bit. A checksum can be thought of as computing a separate parity bit for each bit position of the block and saving those parity bits as an FCS. An error is detected unless every one of those parity bits fails to detect all errors in that bit position. The probability of an undetected 2-bit error for checksum depends on the computation chunk size. Conceptually, it is the probability that the 2-bit errors happen to align in the same bit position of the chunk (e.g., for an 8-bit chunk with one error in bit 3 of the chunk, the second error must be in bit 3 of some other chunk for the FCS to be undetected).

The CRCs are mathematically based on polynomial division over Galois field (2). This means that the data word is considered to be a polynomial in which each bit has an ascending power of variable "x" with a "1" or "0" coefficient, depending upon the binary value of the data word at that bit position. A "CRC polynomial" is used to divide the data word and the remainder is the control value. This division process can be performed via a shift-and-XOR procedure for each bit of the data word.

While a CRC computation may be slower than a checksum computation, it can produce considerably better error-detection results. The error-detection properties are largely determined by the particular CRC polynomial used.

Area	CRC	Checksum
Purpose	To detect data transmission errors between source and target machines.	Software applications use the checksum to calculate data integrity
Error Detection	Capable of detecting double-digit errors	Can't detect many double-bit errors
Complexity	Uses complex functions to detect errors	Uses relatively less complex functions
Reliability	More reliable than a checksum due to the mathematical formula it employs to calculate the CRC	Less reliable than CRC

Table 2: CRC vs Checksum comparison

In particular the CRC control method is applied to data packets that are transmitted between CDH board and DL board, in order to validate these transmissions, both commands and data.

Both the packets towards the CDH and towards the DL are identified by a header and a closer. The CRC algorithm is applied to all the packet except the header and closer. Because of the kind of the algorithm, the length of the control data is of 4 bytes and equal to both the packets, independently with the dataword length.

C2D packet						
Header	Time data	status	Active operative mode	command	CRC32	Closer
3	4	2	1	200	4	3
217						

D2C packet									
Header	Time data	status	Saved data packet number	Active operative mode	Last command	PS telemetry	DL telemetry	CRC32	Closer
3	4	2	4	1	1	70	68	4	3
160									

Table 3: ESA-uProp 3 communication packets between CDH and DL

Implementation

Many versions of CRC implementations are possible, but the algorithm is always the same and it is based on the division between polynomials.

Another important choice is the length of the generator polynomial, in fact if n bits is the length of the generator polynomial, n-1 bits will be the length of the CRC code. For ESA-uProp 3 application a CRC-32 implementation is chosen, so the length of the generator polynomial is 33 bits.

The algorithm can be resumed, considering W as the length of the generator polynomial and R a shift left register, as follows:

1. Add at the end of the message W bits with value 0;
2. Load the R shift register with all 0s;
3. If the most significant bit in the register is 1 do the XOR operation between the message and the polynomial, the result is the new value of the register;
4. If the most significant bit is a 0, shift left the register of one bit;
5. If there are other bits to process, go to step 3, otherwise step 6;
6. The message is processed, and the result is inside the shift register;

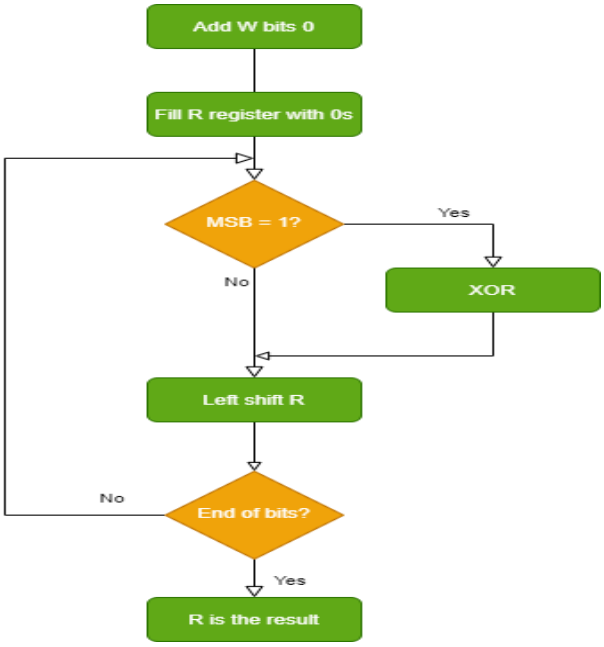

Figure 23: ESA-uProp 3 CRC flow chart

From the receiver side the CRC is computed again on the packet excluding the control field, the result is compared to the one stored into the control field, if the two are equal the packet received is correct, if they are different a retransmission is requested.

Test

Objective

Three kinds of test have been performed:

- TEST 1. This test demonstrates that the CRC function is solid, performing always the same results on the same dataword: there are no errors in software implementation.

- TEST 2. The algorithm is able to detect single errors of different kinds in different position.

- TEST 3. The algorithm is able to detect double errors of different kinds in different position.

Setup

The CDH processor is located on a developing board. A debug line is set towards a workstation trough a UART connection. The workstation uses a minicom terminal in order to communicate with the processor.

Execution

The test has been developed around the function CRC32bitwise() that elaborates the CRC code having as input the string to be encoded. The CRC calculated is consequently added to the packet that must be transmitted.

A random 250 byte string dataword has been created for the test. The length has been chosen as major than all packets that the CRC algorithm is applied on. The CRC has been calculated on the original message and saved. Three kinds of test have been performed:

- TEST 1: the original message has not been modified. The CRC is calculated again and compared with the original CRC: if the new and old CRC are the same the message is not corrupted and this is what we expected from this test.

- TEST 2: a single data error bit is introduced in the simulated received message. The CRC is calculated again and compared with the original CRC. The result expected is a different CRC whose meaning is that the message has been corrupted and the algorithm is able to detect single errors.

- TEST 3: a double data error bit is introduced in the simulated received message. The CRC is calculated again and compared with the original CRC. The result expected is a different CRC whose meaning is that the message has been corrupted and the algorithm is able to detect double errors.

Result

The following image shows the results of one of the tests with all three kinds of test.

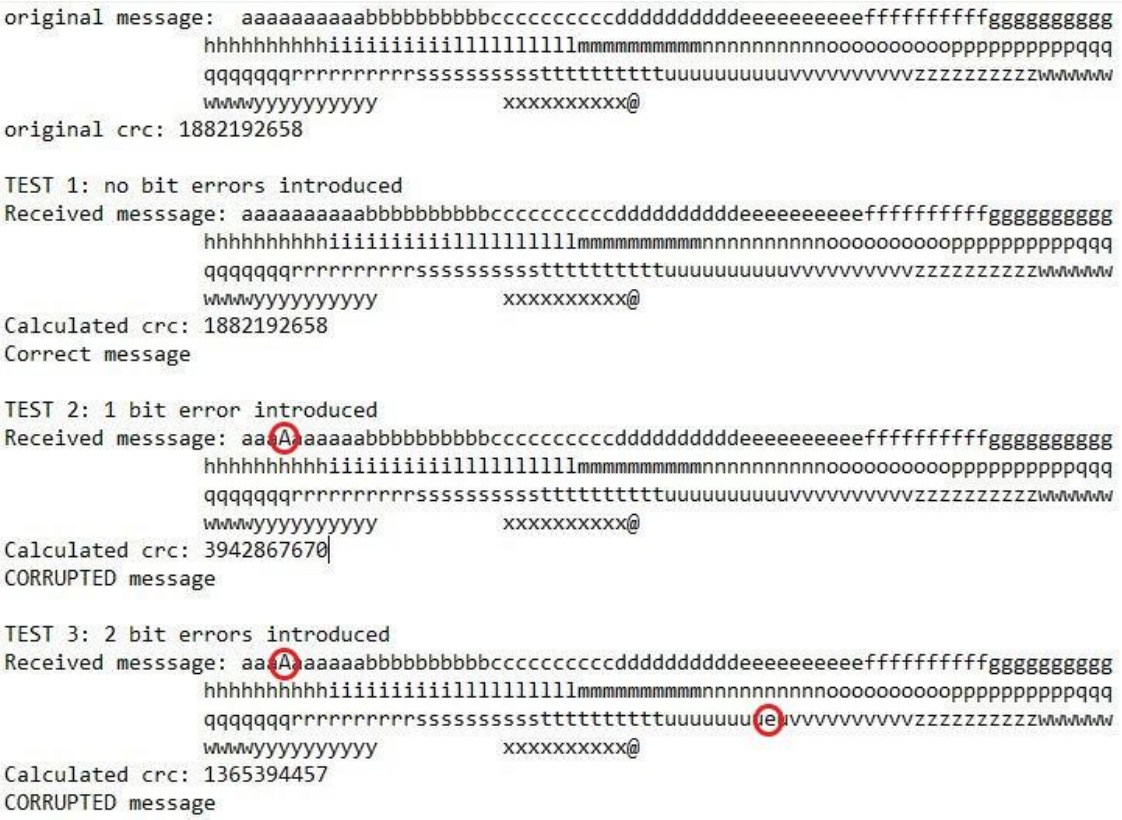

Figure 24: ESA-uProp 3 CRC test result

In the results the original message string can be noticed in form of characters. The modifications on single bits can be noticed as different representations of ASCII symbols respect the original message.

When the received message is equal to the original, the CRC is equal and the packets results as uncorrupted, while in both cases where a single or two bits have been modified, the CRC is different than the original and the packet results as corrupted.

Discussion

Different tests with different datawords have been performed in order to guarantee the independence between the kind of data to be encoded and the algorithm reliability and to guarantee the software reliability with different kind of single and double errors in different positions and different combinations.

3.3.3 Hamming Codes

Reasons

Hamming codes is a SEC/DED code (Single Error Correction/Double Error Detection). This means that ensure the function of a verification code, detecting if any corruption has occurred in a data packet, detecting both single than multiple error bits. Moreover it is able to correct single errors, isolating which bit is changed. This feature allows to keep corrupted packets and correct them, avoiding discarding them or retransmitting them. Moreover the specific algorithm that is applied works on 8 bytes chunks: this feature deals to a slightly more expensive use of memory, but it allows to loose only a single chunk of data in case of corruption instead of the entire packet. Obviously if correction is not possible due to multiple errors, the corrupted chunk of the packet must be retransmitted if possible, or it is lost. In the last case a post process analysis can be performed to isolate the smallest part as possible as corrupted, and sometimes correction is possible.

On the other hand, this method dramatically increases the length of the packets respect the checksum or the CRC. In particular, according with the application on ESA-uProp 3, 1 control byte is added every 8 data bytes. For instance a 250 bytes data string would increase its length of 32 bytes. This value is markedly higher than the 4 control bytes of the CRC method.

This method requires more computational resources to be performed. In fact it requires more complex operations respect CRC and checksum, where just a series of simple operations are performed.

Another disadvantage of this method is that is non-separable. Control bits are located among data bits. This means that encoded packets are not readable by a simple text editor and must be decode before their use. The CRC and checksum methods are separable and the encoded packet is the original packet with control bits located at the end of this and separated from data bits: encoded packets could be simply read also if not decoded.

This modified version of Hamming Code presents some disadvantages, but the pros completely justify cons: complexity rise is completely accepted because of the reliability growth.

For these reasons this encoding protocol is applied to data packets that are saved in memories, because if definitely corrupted would be lost, and the memory space increase is completely acceptable. In fact a 207 bytes string would increase its length of 26 bytes.

Encoded Data packet					
Time data	CTP status	CDH telemetry	DL telemetry	PS telemetry	Hemming Code Bytes
4	11	54	68	70	26
233					

Table 4: ESA-uProp 3 encoded data packet

Because of its high reliability and the feature of real time correction, we evaluated to apply this method on communications with ground. It's quite crucial the dimension size in respect of communications, but in this case it's completely acceptable. The RF communication was the critical element, because of its hardware limitations of physical buffers, but the data budget shows the respect of size limits despite the Hamming code algorithm application. Both downlink and downlink packets of both RF and HL are encoded.

C2G CommSys Data packet							
Kiss AX.25 protocol	Time data	CTP status	CDH telemetry	DL telemetry	PS telemetry	Hemming Code Bytes	Kiss AX.25 protocol
18	4	11	54	68	70	26	1
252							

C2G HL Data packet							
Header	Time data	CTP status	CDH telemetry	DL telemetry	PS telemetry	Hemming Code Bytes	Closer
3	4	11	54	68	70	26	3
239							

G2C HL command			
Header	command	Hemming Code Bytes	Closer
3	200	25	3
231			

G2C RF command			
Kiss AX.25 protocol	command	Hemming Code Bytes	Kiss AX.25 protocol
18	200	25	1
244			

Table 5: ESA-uProp 3 communication packets between CDH and GSS via RF and HL

Implementation

Given n information bits, k check bits are added following this relation:

$$2^k \geq n + k + 1$$

In this case the data string to be encoded is split in blocks of 64 bits, encoded with the Modified Hamming Codes algorithm and written in memory. If the last block is shorter than 64 bits, a bunch of zeros is added such to properly apply the algorithm. For n = 64, a minimum of k = 7 check bits and the additional parity bit are required.

Then, if the 71 positions are numbered starting from 1, all positions that are power of two are filled with a check bit, the other with data bits.

Then while the algorithm reads the string in decreasing order:

- Check bit in position 1 is computed doing a XOR operation of all information bits in a position having the least significant bit set (according with the binary form of the position number) this means: 1,3,5,7,9,11, etc…
- check bit 2 with information bits having the second bit set, this means: 3, 6, 7, 10, 11, etc...
- This procedure is the same for all check bits.
- The parity bit is calculated considering all data bits.

The final encoded string is now formed by singles 9 bytes encoded chunks.

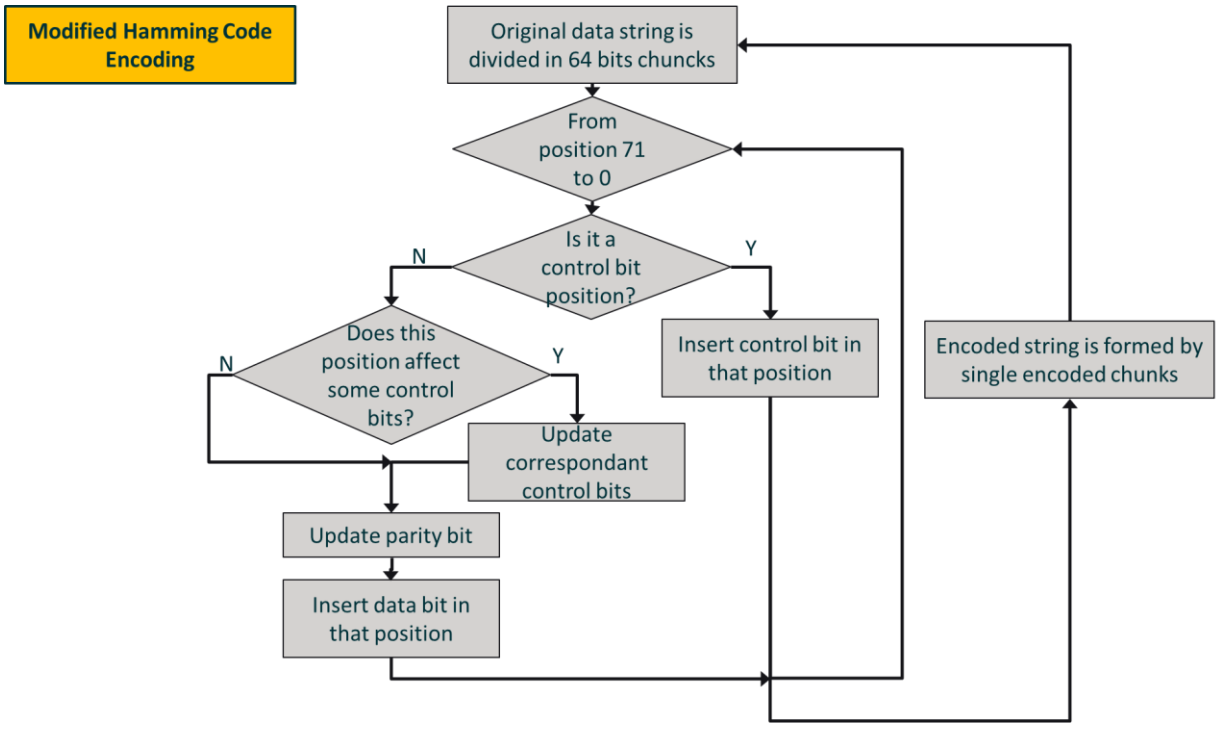

Figure 25: ESA-uProp 3 Modified Hamming Code encoding flow chart

The decoding algorithm is similar to the encoding one. The receiver calculates by his own the control bits by knowing the positions of data bits and control

bits in the encoded word. In this case the encoded chunks are of 9 bytes and the decoding algorithm is applied to each one. The new control bits are now compared with the ones located in the received message.

- If all the calculated control bits, both the parity bit and the check bits, are all equal to the received ones, no error occurred during the transmission or storage.
- If only 1 check bit is different, then the check bit itself is the erroneous one.
- If the parity bit and some of the check bits are different, a single error occurred. It can be identified and corrected: the sum of the positions of the erroneous check bits gives the position of the erroneous information bit.
- If the parity bit is correct, while the check bits are not, either a double error or a check bit error occurred: in any case, it cannot be corrected, and the entire chunk must be considered as corrupted.

The decoding function also assembly the decoded string from the different decoded chunks.

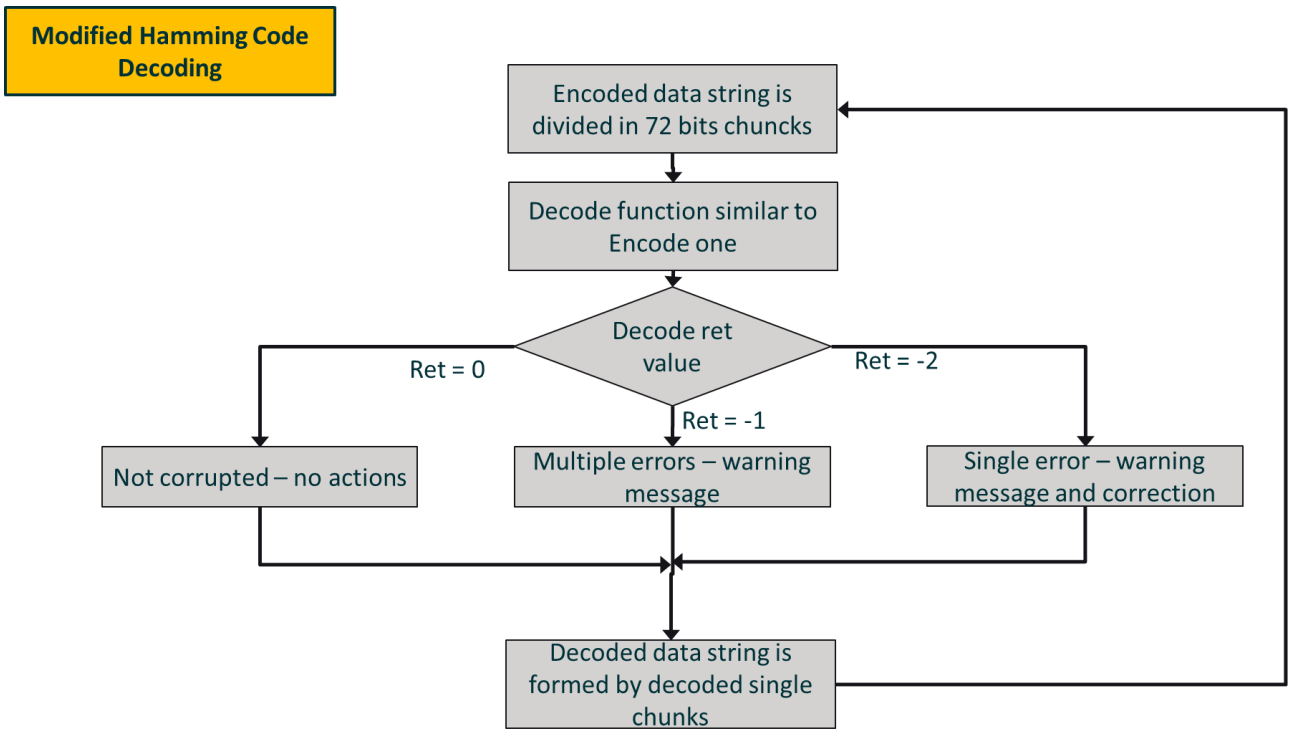

Figure 26: ESA-uProp 3 Modified Hamming Code decoding flow chart

Test

Objective

Different aspects of the algorithm have been tested:

- TEST 1. The objective is testing the correct encoding and decoding processes of this code. No errors shall be introduced by the algorithm, independently the kind of datawords that it's applied on. The aim of this test is to verify the correct encoding and decoding procedures on uncorrupted packets.

- TEST 2 and 3. The algorithm must detect, isolate and correct single errors.

- TEST 4 and 5. The algorithm must recognise different kinds of double errors and report them. This kind of test shows limits of this method.

Setup

The CDH processor is located on a developing board electrically supplied. A debug line is set towards a workstation trough a UART connection. The workstation uses a minicom terminal in order to communicate with the processor.

Execution

A 250 bytes random string dataword has been created for the test. The length has been chosen as major than all packets that the modified Hamming Code algorithm is applied on.

This string has been encoded and saved in a file.

- TEST 1: no errors introduced in the encoded string. Then the decoding function is launched on the new string and results are printed. Different tests have been performed changing data king and length of the string.

- TEST 2: single error of a data bit introduced in the encoded message. Then the decoding function is launched on the new string and results

are printed. Different tests have been performed with different positions of the erroneous data bit and in different chunks, and in multiple chunks on the same time.

- TEST 3: single error of a control bit introduced in the encoded message. Then the decoding function is launched on the new string and results are printed. Different tests have been performed with different positions of the erroneous control bit and in different chunks, and in multiple chunks on the same time.

- TEST 4: double error of data bits introduced in the encoded message. Then the decoding function is launched on the new string and results are printed. Different tests have been performed with different positions of the erroneous control bit and in different chunks, and in multiple chunks on the same time.

- TEST 5: a particular error combination of a control and a data bit. Then the decoding function is launched on the new string and results are printed.

Results

Figure 27: ESA-uProp 3 Modified Hamming Code test results

- TEST 1. The decoding algorithm is applied and doesn't detect any errors as expected. The decoded dataword is printed in the file and results identical to the original one, once again as expected.

- TEST 2. In this the algorithm works in the way as expected: the erroneous bit is detected, isolated and corrected. In fact the decoded word doesn't shows the error and is identical to the original one. In particular the decoding algorithm supplies a warning message indicating the anomalous chunk.

- TEST 3. In this the algorithm works in the way as expected: the erroneous bit is detected. In this case correction is not necessary because in the decoding phase control bits are discarded: the strength of the algorithm is that it understands the difference between the case a control bit is different because it is erroneous and the case it is

different because a data bit is erroneous and affects the control bit. In fact the decoded word doesn't shows the error and is identical to the original one. In particular the decoding algorithm supplies a warning message indicating the anomalous chunk.

- TEST 4. This test aims to check the limit of the correction functionality of the method. In fact when 2 errors occur in the same chunks the system isn't able to correct them but can detect them, and the correspondent chunk is reported. In this case correction is not possible, as demonstrated by the errors in the decoded message. The advantage is that the method report which chunk is corrupted and not the entire string must be discarded.

- TEST 5. This test highlines a weakness of this method. If a particular combination of errors afflicts a chunk, the algorithm could risk to badly understand the origin of the corruption. In this particular case this error combination is confused with a single data bit error (different from the erroneous one) and the system tries to correct it. Obviously the decoding operation has an erroneous result and the decoded string presents errors and differences with the original one.

Discussion

This method demonstrates to be very reliable and useful.

Test 5 results could seem to be a real problem. Despite all, this aspect should not be worrying. In fact the system always report chunks where a correction has been performed, and the operator can verify the quality of data. Moreover the probability that this combination happens is very low and completely justify this problematic aspect.

3.3.4 Operative modes

There are four different operative modes implemented on the platform.

- **Dormant mode**: CTP subsystems are switched off. No operations are executed in this mode. During this mode the CTP could be externally alimented to allow battery recharge.

- **Basic mode**: CTP avionics is active, while PS if off. In this mode the CTP collects data from ADCs and control them. Communications between CDH and DL are enabled and towards ground both via HL and via RF, if enabled. The PS is not alimented from EPIS, so PS operations are not performed, nor communications.

- **PS mode**: CTP avionics is active, PS is active, while thruster is off. This operative mode is very similar to the Basic mode. Differently, the PS is alimented and PS avionic is active and can perform operations, but thruster is not active.

- **Burst mode**: CTP avionics, PS and thruster are active. This mode is similar to the PS mode, but also thruster is active and can perform a burst. This phase is dramatically more critical than others because of the Burst: temperature can markedly rise, electrical consumption is very high and so on voltages and currents, this means a big stress for many electronic devices, from batteries to power managements boards and electrical buses, and finally the electromagnetic field could become very intense.

Transition between these operative modes could be both automatic and via commands. The operator can send specific commands, both via HL and RF, to allows operative modes transitions.

The system can also automatically change its operative mode for safety reasons. In fact, a check is performed on critical parameters in every operative mode, except for dormant, and according with the kind of critical parameters

outside the safety range and the period of this misbehaviour, the system can perform a series of corrective actions to avoid material damages.

The third way to change operative mode is the manual transition. In particular the CTP has a load switch that cuts off the energy supply from batteries to every platform load, interrupting all the processes and transitioning to the dormant mode. The operator can use this device if notice some misbehaviours that the system can't automatically recognise. When the CTP is located inside laboratory's vacuum chamber a physical interface allows the switch activation from the outside of the chamber. This is a crucial safety feature. The switch is also open when platform is not used or during transportation. Obviously operator must manually close the switches before the use of the CTP.

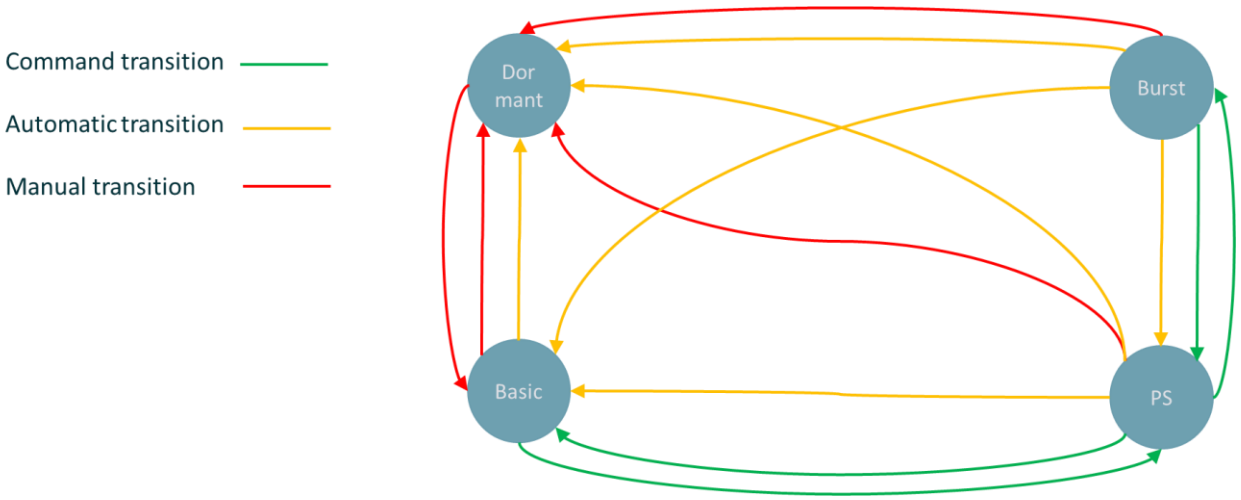

Figure 28: ESA-uProp 3 Operative mode transitions

Implementation

The CDH manages operative modes, but DL is continuously updated to the current operative mode.

At the beginning of each loop the system automatically manages the different operative modes and the transitions among them. For each of them different actions are performed.

- NOMINAL: The watchdog is pinged and the Main loop starts.
- PS_ON: The watchdog is pinged and the Main loop starts. (electrical supply towards PS is provided, but this function is in DL software's competence).

- BURST: The watchdog is pinged and the Main loop starts. DL communicates with PS and also PS critical parameters are checked.

- POWEROFF (transition): The watchdog is disabled, all open interfaces are closed, the system exits from the Main loop and terminates execution.

- REBOOT (transition): All open interfaces are closed, the system stops pinging the watchdog and then waits until the watchdog reboots the system

- Other (transition): if the value in "opmode" variable can't be recognised it means that the software had a failure and the system is automatically rebooted.

Test

Objective

The objective of the test is to check the correct execution of the specific actions in different operative modes. In particular the algorithm shall:

- Correctly ping the watchdog in NOMINAL/PS_ON/BURST modes (TEST 1). The watchdog shall recognise a time over event (TEST 2).

- Correct program shut down in POWEROFF mode (TEST 3).

- Correct reboot of the system in REBOOT mode (TEST 4) or when the value in "opmode" is not recognised (TEST 5).

Setup

The CDH processor is located on a developing board electrically supplied. A debug line is set towards a workstation trough a UART connection. The workstation uses a minicom terminal in order to communicate with the processor.

Execution

The “opmode” value is set according with which mode has to be tested. The program is then launched.

In case of TEST 2 a sleep pause major than watchdog period, previously set, is forced.

Visual verify on Workstation screen is acted for each test.

Results

In all tests results are compliant with expected ones.

Discussion

Some tests have been repeated to confirm previous results.

Future tests will be performed to guarantee the correct integration of operative modes in other software functions during the AIV campaign tests.

3.3.5 Critical Parameters Check

Reasons

The system aims to guarantee the highest level of safety and reliability as possible. The best way to perform this feature is to improve the automatization degree in order to decrease the amount of work of the operator and to avoid human errors. In particular there are some parameters that can

reveal the status of the platform. The human operator is not able to continuously check all these parameters, and the best solution is to make this function automatic. Obviously an algorithm can't have evaluation skills compared to a human. The crucial aspect is evaluating all possible failures and cases in order to permit the software to recognise it and to implement the bests solutions in any case.

The first aspect is to evaluate which parameters must be checked. Anyone of them, if out of an expected range, are a symptom of a failure in the system.

Critical parameters	CDH	DL
battery 1 voltage	x	
battery 1 current	x	
battery 1 temperature	x	x
battery 2 voltage	x	
battery 2 current	x	
battery 2 temperature	x	x
BCR temperature	x	x
Transmitter temperature	x	x
C&DH temperature	x	x
EPS temperature (specify - HUB)	x	x
EPS 5V bus current	x	
EPS 5V bus voltage	x	
EPS 3,3V bus current	x	
EPS 3,3V bus voltage	x	
EPS 9V bus current	x	
EPS 9V bus voltage	x	
EPS-EPIS bus current (CDH side)	x	
EPS-EPIS bus voltage (CDH side)	x	
EPS-EPIS bus current (DL side)		x
EPS-EPIS bus voltage (DL side)		x
EPIS temperature (specify - DCDC, step-up)	x	x
EPIS 5V bus current		x
EPIS 5V bus voltage		x
EPIS 3,3V bus current		x
EPIS 3,3V bus voltage		x
EPIS-PS bus current		x
EPIS-PS bus voltage		x
DL temperature	x	x

Table 6: ESA-uProp 3 critical parameters list

Acquisition of all parameters, among that the critical ones, is performed by CDH and DL that can communicate with different ADCs via SPI.

Failure could afflict a component, or a series of them, or a software operation. A critical situation could be also caused by a bad evaluation of the environment where the CTP must operate, especially because the low TRL of the performing technologies that obviously are present during test operations.

It is crucial to consider that the same acquisition process could fail, for example a sensor could break or could fail. For instance, a very educational heritage of the previous project ESA-uProp 2 is that some thermal sensors detached from their assigned components due to bad fastenings and high vibrations, and they weren't able to measure the correct temperature. This is also very dangerous because if this failure isn't noticed, the real temperature of some components would be underestimated and if not monitored would be a risk for the platform itself, the PS system, the test laboratory and the operators.

The main aim of the project is to finish the mission, also in a degraded way, if necessary. It means that the platform must avoid useless interruptions due to false alarms. For this reason, the critical parameters check function has been improved in terms of reliability of the algorithm itself in order to reach the maximum performance in terms of safety and availability of the system.

Two main aspects have been implemented:

- **Hardware Redundancy**. Temperature sensors have been redounded in order to increase their reliability and safety and to minimise the probability of false measurements. Other kinds of critical measurements are performed by more reliable sensors and don't have the necessity to be redounded. Each couple of thermal measure is performed by two different sensors linked by two different ADCs and acquisition circuits operated by two different processors, the CDH and the DL. The two measures are provided to the CDH that can evaluate and comparing them.

- **Time Redundancy**. The system performs the control on critical parameters every second, and when one of them is evaluated as out of safety range the system doesn't actuate immediately corrective actions but continues to monitor all parameters. If a certain parameter continues to be out of range for an established period of time the system acts corrective actions. This feature allows to maximize the reliability

of perceived critical parameters and to avoid useless mission interruption due to false measurements.

Implementation

Different parameters acquisition is performed by CDH and DL. Some parameters are under the competence of the first element, others the second and temperature parameters are measured by both. Acquisition is performed by sensors and ADCs interrogated via SPI.

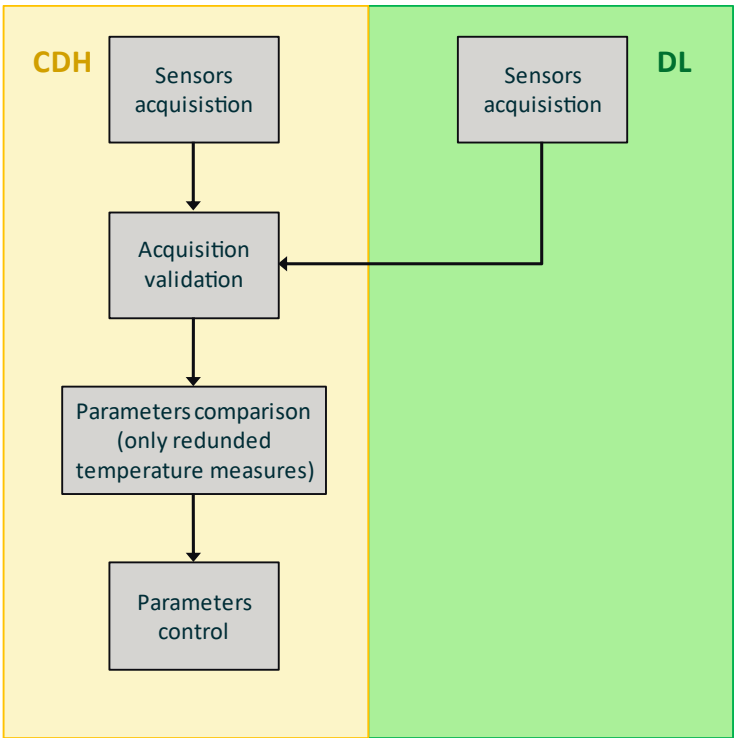

Figure 29: ESA-uProp 3 critical parameters management

Then all these information are collected in predisposed structures by CDH that performs their check. All measurements are expressed in forms of 16 bits data in a scale between 0 and 65535 (or 2^16-1).

All parameter checks can be individually enabled or disabled before the test. This feature guarantees a marked flexibility in kind of test and it allows an easier conduction of development tests.

Two different kinds of management can be considered:

- **Temperature measurement.** The system controls if both the measures of the same component are validated in order to check that no errors occurred during acquisition. Three cases are possible:

    - No one of them are validated. Both acquisition processes have failed. It's impossible to estimate the real value of the component. For this reason, a conservative process is applied and the critical parameter is considered as out of range: the critical counter for that specific parameter is increased.

    - Only one of them is validated. In this case the unvalidated value is discarded and the validated one is considered to apply on the check process.

    - Both measures are validated. In this case the system performs an ulterior control. The system checks the coherence between the two measures, comparing the absolute value of their difference with an established delta: if the difference is major a warning message is sent to the operator. In any case the major value between them is chosen for the control according with a conservative approach.

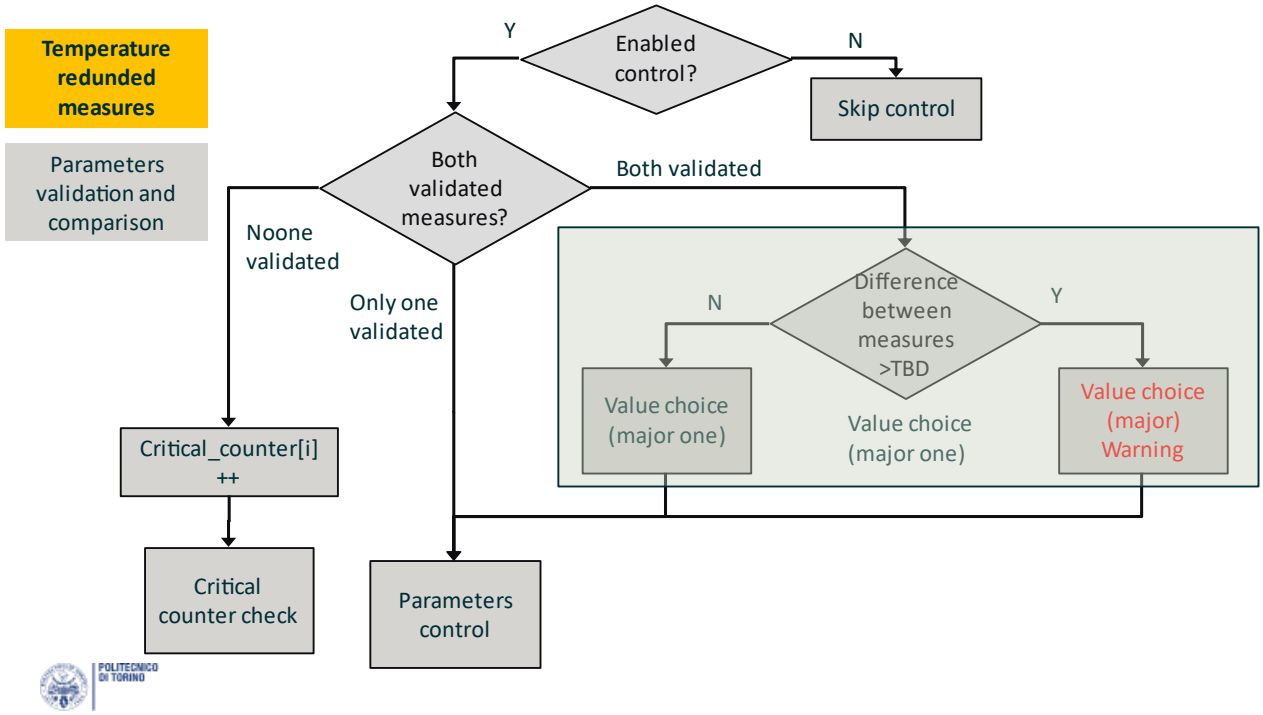

Figure 30: ESA-uProp 3 temperature redundant measures validation and comparison flow chart

The following step is the evaluation if the parameter is inside an established safety range. In particular for critical temperatures only upper limits have been considered. Two different steps of upper limits have been considered, in order to improve availability of the system and to prevent mission interruptions that could be avoided. The two steps are of increasing severity: the overpassing of the first limit leads to a series of corrective actions in order to correct the failure, but if these don't demonstrate positive effects and the parameter overpasses the second limit more severe corrective actions are performed in order to guarantee the avoidance of material damages to CTP.

Each time a certain parameter doesn't respect a certain limit the system reports it in a global variable, a counter that considers the time since the parameter is not respecting the limit in a continuative way. Then the system recognises if the counter is over the established time limit. In case of affirmative result some corrective actions are performed according with the kind of off-nominal parameter and the

severity of the not respected limit. While if the parameter respects the considered limit, the respective counter is set to 0.

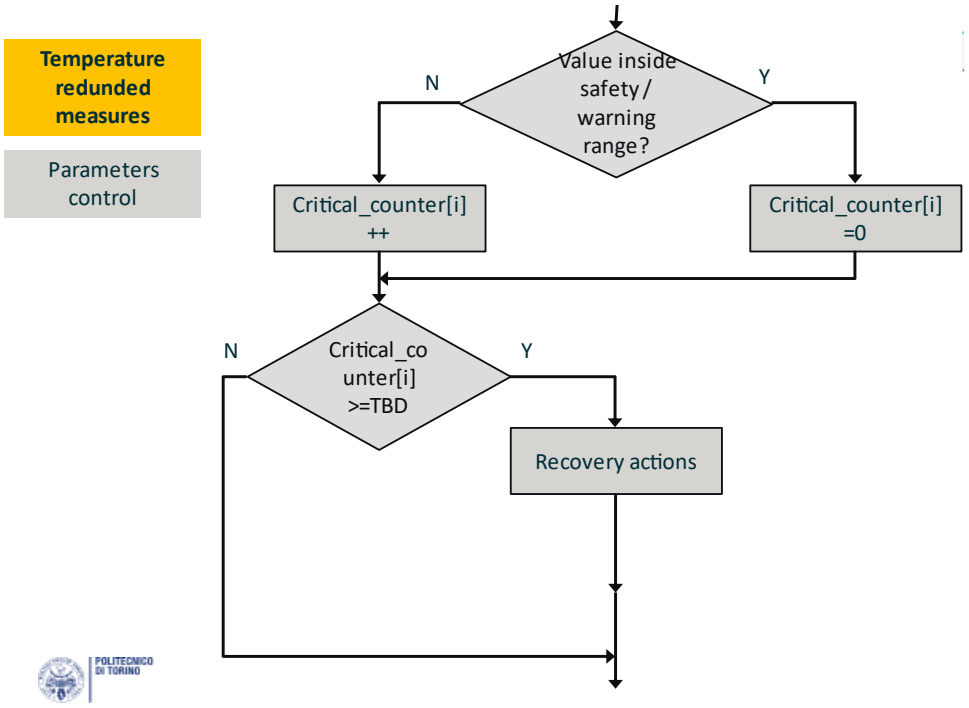

Figure 31: ESA-uProp 3 temperature redundant measures control flow chart

- **Other parameters measurement**. All other parameters (except 12V bus parameters) present a single measure of them, because their acquisition reliability is considered as high enough. The validation phase presents only two cases:

    - The value is not validated. It's impossible to estimate the real value of the component. For this reason, a conservative process is applied and the critical parameter is considered as out of range: the critical counters for all the limits of that specific parameter are increased.

    - The value is validated. The system can correctly proceed to the parameter control phase.

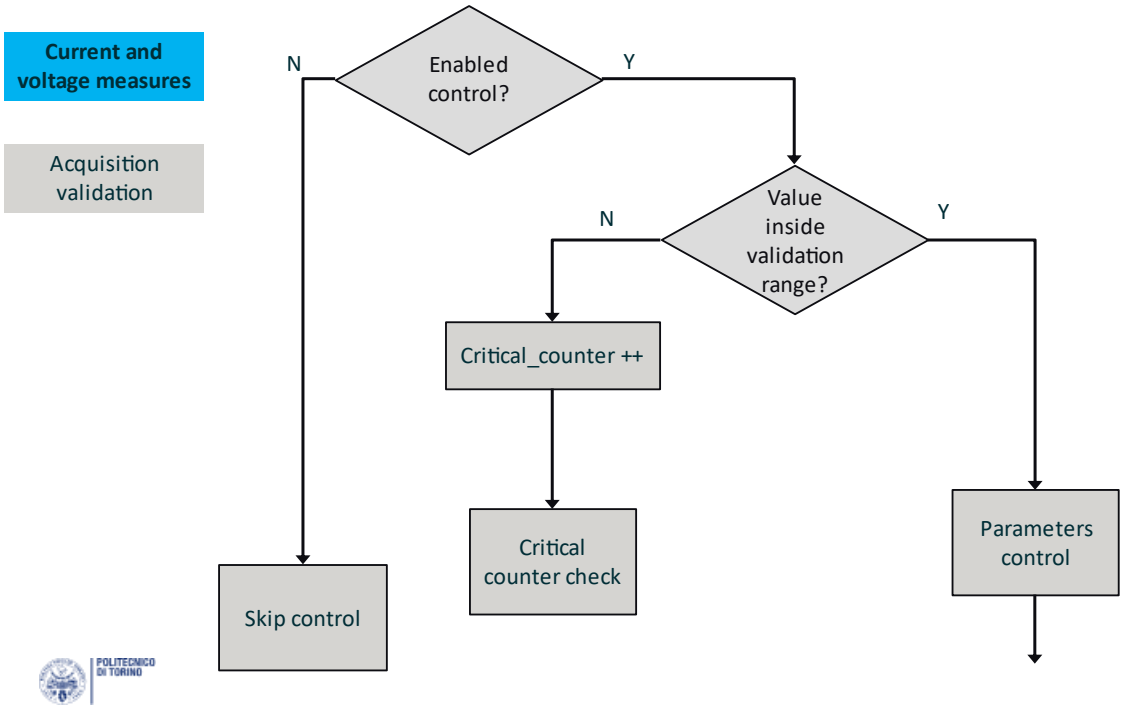

Figure 32: ESA-uProp 3 critical parameters validation flow chart

The following step is the evaluation if the parameter is inside an established safety range. For the same parameter it's evaluated upper limits or lower limits or both. Sometimes different steps for upper or lower limits have been considered, in order to improve availability of the system and to prevent mission interruptions that could be avoided. The two steps are of increasing severity: the overpassing of the first limit leads to a series of corrective actions in order to correct the failure, but if these don't demonstrate positive effects and the parameter overpasses the second limit more severe corrective actions are performed in order to guarantee the avoidance of material damages to CTP.

Each time a certain parameter doesn't respect a certain limit the system reports it in a global variable, a counter that considers the time since the parameter is not respecting the limit in a continuative way. Then the system recognises if the counter is over the established time limit. In case of affirmative result some corrective actions are performed according with the kind of off-nominal parameter and the

severity of the not respected limit. While if the parameter respects the considered limit, the respective counter is set to 0.

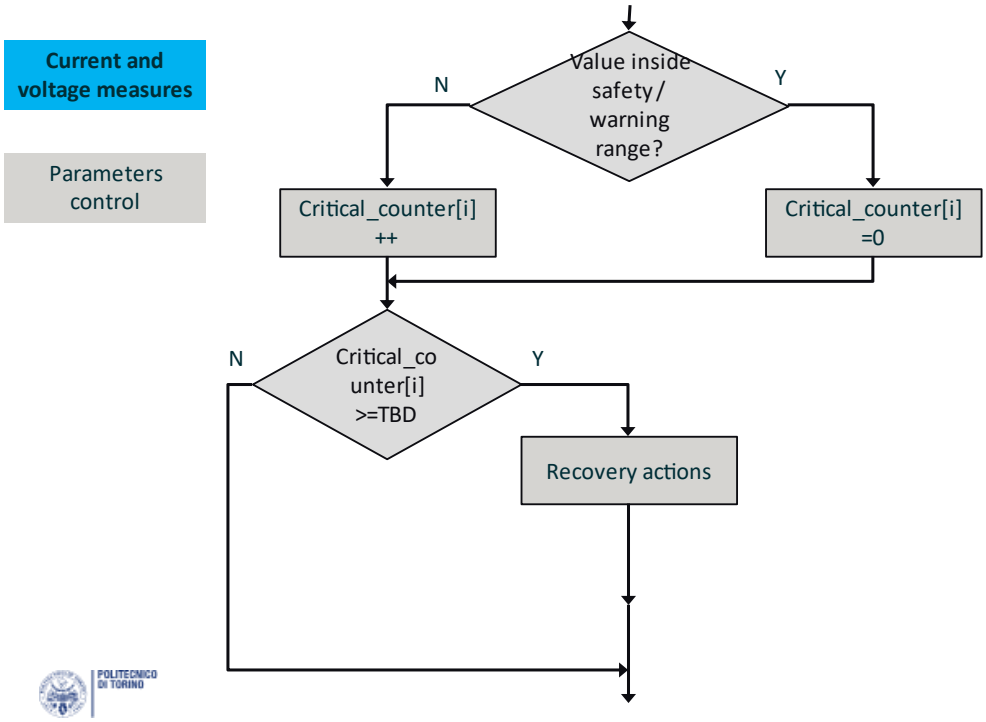

Figure 33: ESA-uProp 3 critical parameters control flow chart

- **12V bus current and voltage.** These two parameters represent an exception in the precedent model. In fact this electrical bus is monitored by both the EPS and EPIS boards. On each of them an ADC can monitor its current and voltage. Moreover the automatic check, if enabled by the operator, is performed only when the PS is active, that means just in case the operative mode is PS mode or BURST mode, to avoid false reporting of misbehaviours when the component is off as expected.
  In the same way of temperature parameters management, the system controls if both the measures of the same component are validated in order to check that no errors occurred during acquisition. Three cases are possible:

  - No one of them are validated. Both acquisition processes have failed. It's impossible to estimate the real value of the

parameter. For this reason, a conservative process is applied and the critical parameter is considered as out of range: the critical counter for that specific parameter is increased.

- Only one of them is validated. In this case the unvalidated value is discarded and the validated one is considered to apply on the check process.

- Both measures are validated. In this case the system performs an ulterior control. The system checks the coherence between the two measures, comparing the absolute value of their difference with an established delta: if the difference is major a warning message is sent to the operator. In any case the major value between them is chosen for the control according with a conservative approach.

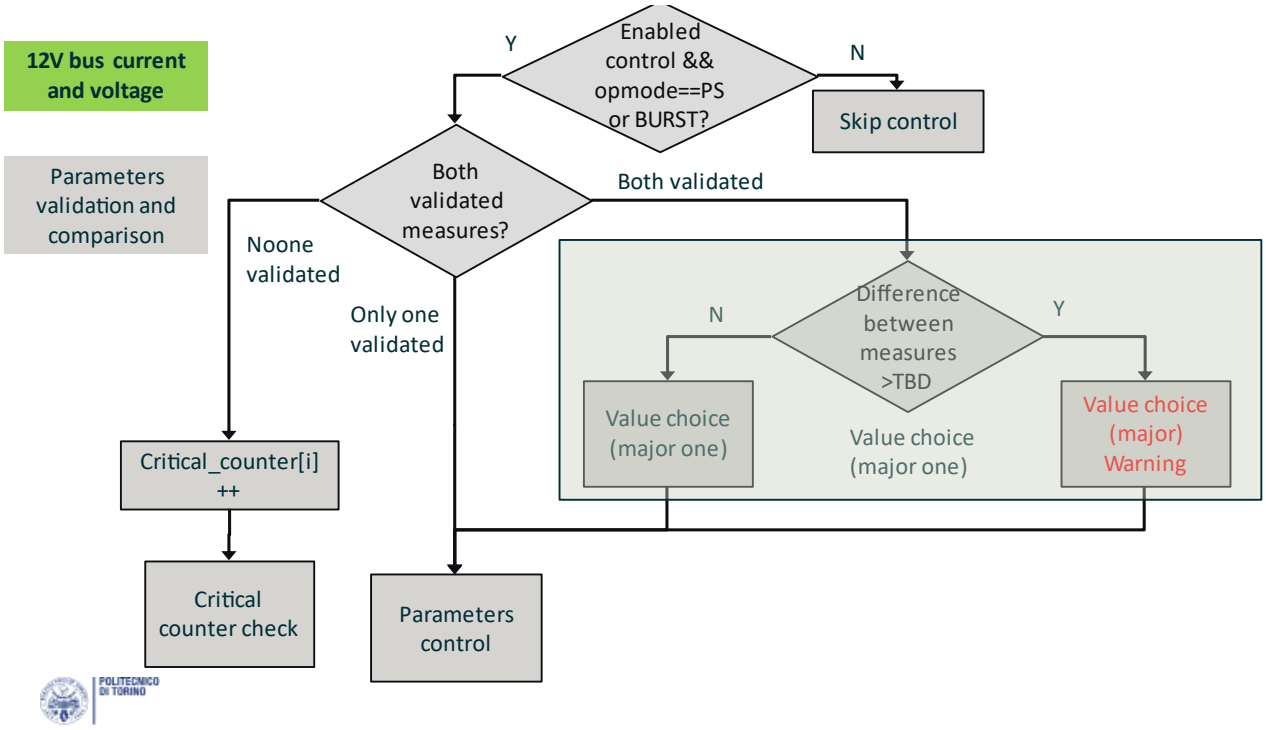

Figure 34: ESA-uProp 3 critical parameters 12V bus validation and comparison flow chart

The control of 12 V bus current represents an ulterior exception. There is a single upper limit, but the system evaluates the counter above two different time limits. The overpassing of the first limit leads to a series of corrective actions in order to correct the failure, but if these don't demonstrate positive effects and the parameter overpasses the second limit more severe corrective actions are performed in order to guarantee the avoidance of material damages to CTP.
The control of 12V bus voltage is performed between a lower and an upper limit.

Test

Development test has been organised in two phases: the main difference is the practical execution of corrective actions, that is testable only when the function is inserted in the general software and this is generally tested.

Objective

The first phase experienced the testing of validation and control algorithm, excluding the application of corrective actions. The algorithm shall be able to validate data in data structures, to compare redundant measures if present, to check the respect of parameters limit, and the increase of the critical counter.

Setup

The CDH processor is located on a developing board electrically supplied. A debug line is set towards a workstation trough a UART connection. The workstation uses a minicom terminal in order to communicate with the processor.

Execution

Many tests have been conducted in order to test all different parameters individually and combination of them in a second moment. During the tests only interested parameters checks have been enabled. Some values about values limits and time limits have been used for test, but can be easily changed, the aim of test is evaluating the correct functionality of the algorithm.

For each test necessary fields of data structures, both from CDH and DL, have been filled with arbitrary values that could change in each loop during the test execution, in order to test all cases that could happen, and that CTP must be able to face up to. The test is launched and results are printed.

Results

An example of test conducted is reported.

```
loop 1
ATTENTION: bcr temperature both not validated
loop 2
ATTENTION: bcr temperature both not validated
loop 3
ATTENTION: bcr temperature both not validated
loop 4
ATTENTION: bcr temperature both not validated
 !!!! bcr_temp1 !!!!
loop 5
ATTENTION: bat1 temperature over the limit 1
ATTENTION: bcr temperature both not validated
 !!!! bcr_temp1 !!!!
loop 6
ATTENTION: bat1 temperature over the limit 1
ATTENTION: bat1 temperature over the limit 2
ATTENTION: bcr temperature both not validated
 !!!! bcr_temp1 !!!!
loop 7
ATTENTION: big delta bat1 temperatures
ATTENTION: bat1 temperature over the limit 1
ATTENTION: bat1 temperature over the limit 2
ATTENTION: bcr temperature both not validated
 !!!! bcr_temp1 !!!!
loop 8
ATTENTION: big delta bat1 temperatures
ATTENTION: bat1 temperature over the limit 1
ATTENTION: bat1 temperature over the limit 2
ATTENTION: bcr temperature both not validated
 !!!! bat1_temp1 !!!!
 !!!! bcr_temp1 !!!!
loop 9
ATTENTION: big delta bat1 temperatures
ATTENTION: bat1 temperature over the limit 1
ATTENTION: bat1 temperature over the limit 2
ATTENTION: bcr temperature both not validated
 !!!! bat1_temp1 !!!!
 !!!! bat1_temp2 !!!!
 !!!! bcr_temp1 !!!!
loop 10
ATTENTION: bat1 temperature both not validated
ATTENTION: bcr temperature both not validated
 !!!! bat1_temp1 !!!!
 !!!! bat1_temp2 !!!!
 !!!! bcr_temp1 !!!!
```

Figure 35: ESA-uProp 3 critical parameters test results

In this test results many features of the check algorithm can be noticed.

- BCR temperature measures have been set as out of validation range in all loops. In every loop the system can detect it and the respective critical counter is increased. In particular BCR temperature has only one upper limit. Another example of not validated parameter is in loop 10.

- In loop 4 and 5 the temperature of bat 1 is increased and the system detects it. In fourth loop the system detects the overpassing if the first limit, while in the second also the second.

- In loop 7, 8 and 9 both measures of bat1 temperature are validated, but the system can automatically detect their incoherence, reporting that there is a big delta between them.

- The system reports every limit overpassing after 4 loops that this condition is continually confirmed. In loop 4 BCR temp limit 1 is reported, because of its overpassing since loop 1. While bat 1 temp limits are reported in loops 8 and 9. The management of different step limits for the same parameter is so tested.

Discussion

Future tests during AIV campaign of tests will be conducted in order to test the management of numerous multiple parameters and the correct execution of corrective actions that involve other functionalities of the software.

3.3.6 Electrical boards hardware redundancies.

During the development of electronic boards, in collaboration with an external company, some solutions to improve reliability has been purposed. These boards are the EPS and the EPIS. Their function is to manage electrical energy in CTP and supply it to any component who needs.

The EPS manages the electricity provided directly by two dedicated battery packs and regulates voltages towards avionics boards (CDH and CommSys). Then a bus supplies the EPIS from EPS. The EPIS regulates voltages to supply DL board and PS.

Two main examples of redundancies are applied:

- **EPIS DC-DC boost component signal redundancy**
  On the EPIS board there is a component whose function is to set the correct tension of the electrical bus towards the PS. This component has also a switch function: the DL can regulate the activation of this switch with the set of a pin. In order to avoid accidental errors in management of this critical component, the signal that activates it is redundant: two pins and a logical port AND allow this feature.

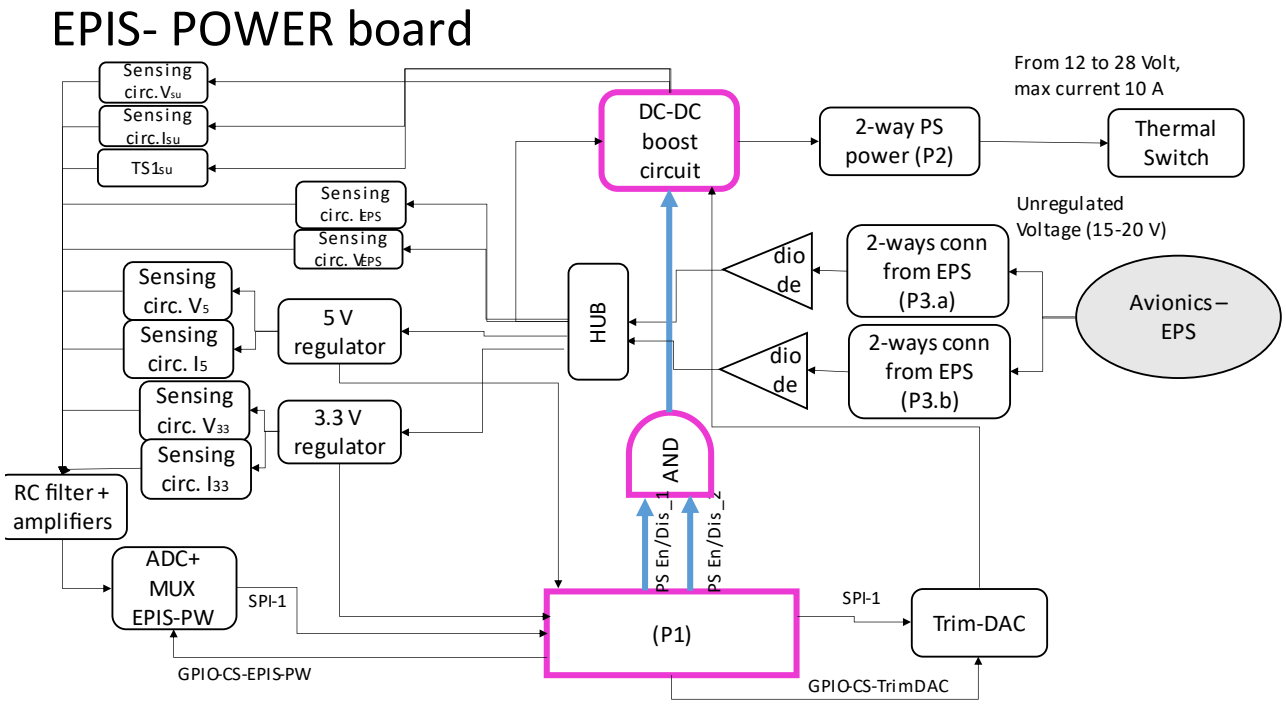

Figure 36: ESA-uProp 3 EPIS-Power board electrical schematic DC-DC boost signal redundancy

- **Electrical supply bus EPS-EPIS redundancy.**
  Between the EPS board and the EPIS one there is an electrical bus. Energy supply passes from the first board towards the second one. The EPIS board provides regulated electricity to PS and DL board. This last

element is crucial for the correct functionality of CTP. For this reason this bus is made as redundant, in order to avoid every misbehaviours in case of failures of this electrical link.
As can be seen in electrical schematics, on EPIS board there are two twin ports. Two separates electrical lines are linked by a hub. Diodes are precautional elements to avoid possible reflux currents.

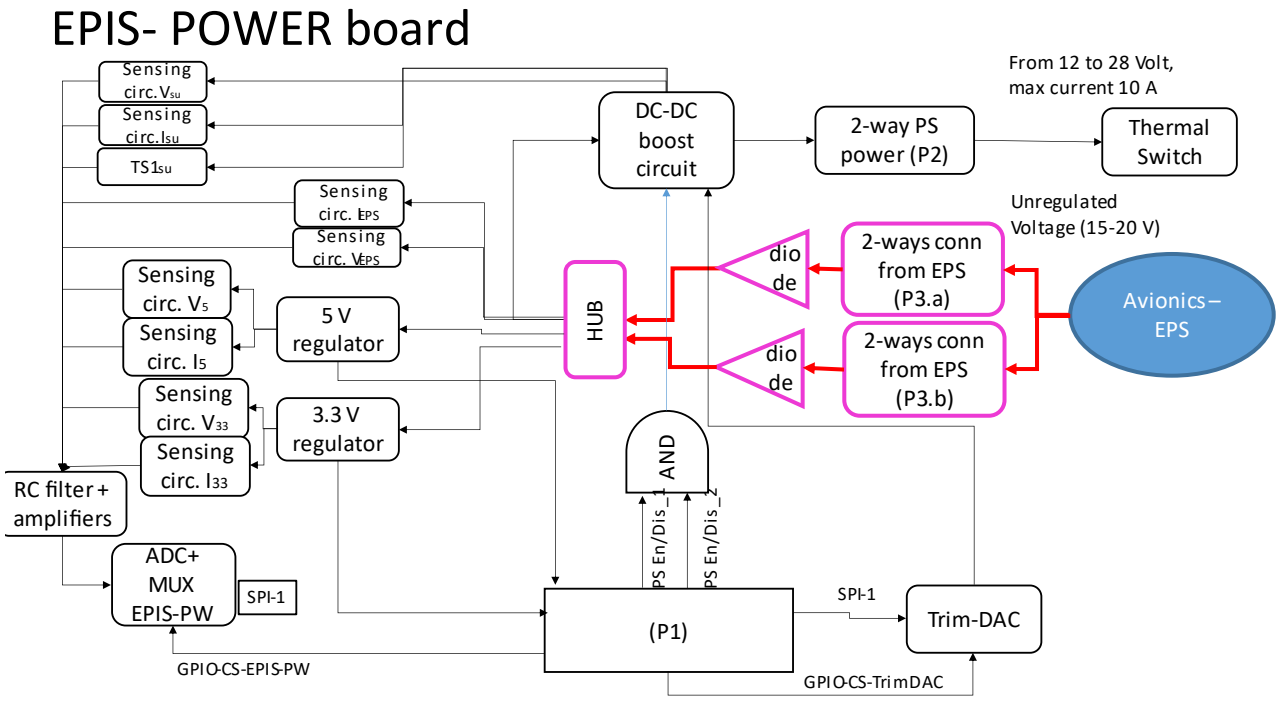

Figure 37: ESA-uProp 3 EPIS-Power board electrical schematic electrical bus redundancy

3.3.7 CDH – DL communication test

Reasons

In the previous ESA-uProp project (ESA-uProp 2) the CDH board used to communicate directly with PS through different transmission protocols. In ESA-uProp 3 CDH and communicates with DL: for this reason the

verification of a correct communication between these boards is crucial, also because these boards must be coordinated because their functions are mutually connected.

The communication between CDH and DL is based on a device called UART (universal asynchronous receiver-transmitter). This device uses an asynchronous serial protocol. The term serial means that data are transmitted along time, one bit per time, and in particular the buffer is transmitted from the least significant bit to the most significant one. The term asynchronous means that there isn't a time signal transmission, but when the serial is set a previous baud rate is shared between two communicating devices: special bits indicate the begin and the end of the transmission.

During this test campaign 5 different test have been conducted.

Test 1

Objective

The objective of this test is to verify if some raw data can be sent and received through the UART line. This test verifies that there aren't hardware problems on the line and the communication protocol works.

Setup

CDH processor is located on the development board that is electrically supplied. A debug line is connected to the CDH Workstation, in order to control the processor and monitor its activity.

The DL processor also is located on a development board externally power supplied. This processor is controlled and monitored by a DL Workstation through a debug line between the Workstation and the development board.

The communication line between CDH board and DL board is provided by small cables called jumpers that links the correct pins. In particular one jumper links CDH TX to DL RX, one jumper links CDH RX to DL TX and the third jumper is used to link grounds.

Execution

Each workstation commands to its processor to execute a program that does the following actions:

- A string of raw data is created.
- The UART serial communication is set.
- A loop is periodically executed:
    - The string is sent through the transmission serial.
    - The received buffer is read and all received data are printed on the workstation's screen.

Results

On both CDH and DL workstations' screens the strings sent is shown in most of cases, as expected.

Discussion

Sometimes the receiver device reads more bytes than the string's ones. This problem has been highlighted and solved during the Test 5 execution.

Test 2

Objective

The objective of this test is to verify if some data inserted in a packet can be sent and received through the UART line. This test verifies that there aren't hardware problems on the line and the communication protocol works.

Setup

CDH processor is located on the development board that is electrically supplied. A debug line is connected to the CDH Workstation, in order to control the processor and monitor its activity.

The DL processor also is located on a development board externally power supplied. This processor is controlled and monitored by a DL Workstation through a debug line between the Workstation and the development board.

The communication line between CDH board and DL board is provided by small cables called jumpers that links the correct pins. In particular one jumper links CDH TX to DL RX, one jumper links CDH RX to DL TX and the third jumper is used to link grounds.

Execution

Each workstation commands to its processor to execute a program that does the following actions:

- A packet of established data is created.
- The UART serial communication is set.
- A loop is periodically executed:
    - The packet is sent through the transmission serial.
    - The received buffer is read and all received data are printed on the workstation's screen.

Results

On both CDH and DL workstations' screens the packets sent are shown in most of cases, as expected.

Discussion

Sometimes the receiver device reads more bytes than the packet's ones. This problem has been highlighted and solved during the Test 5 execution.

Test 3

Objective

The objective of this test is to verify if some data inserted in a packet can be sent and received through the UART line. The software must be able to recognise all fields of that packet, decomposing it. In this test a header and a closer have been inserted in order to identify the packet.

Setup

CDH processor is located on the development board that is electrically supplied. A debug line is connected to the CDH Workstation, in order to control the processor and monitor its activity.

The DL processor also is located on a development board externally power supplied. This processor is controlled and monitored by a DL Workstation through a debug line between the Workstation and the development board.

The communication line between CDH board and DL board is provided by small cables called jumpers that links the correct pins. In particular one

jumper links CDH TX to DL RX, one jumper links CDH RX to DL TX and the third jumper is used to link grounds.

Execution

Each workstation commands to its processor to execute a program that does the following actions:

- A packet of established data is created.
- The UART serial communication is set.
- A loop is periodically executed:
    - The packet is sent through the transmission serial.
    - The received buffer is read and the packet is decomposed. The software verifies if the packet is recognised searching for the specific bytes of header and closer.
    - If the packet is recognised, the data contained in it are saved in a specific structure. Each field of the structure is printed on workstation's screen.

Results

Both CDH and DL workstations' screens shows the packets sent in most of cases, as expected. In this case they have been correctly send, received and recognised.

Discussion

Sometimes the receiver device reads more bytes than the packet's ones. This problem has been highlighted and solved during the Test 5 execution.

Test 4

Objective

The objective of this test is to verify if some data inserted in a packet can be sent and received through the UART line. The software must be able to recognise all fields of that packet, decomposing it. In this test a header and a closer have been inserted in order to identify the packet. Moreover the CRC method is added as a verification technique on the entire packet (except header and closer). The software must be able to correctly calculate, receive and compare CRC codes.

Setup

CDH processor is located on the development board that is electrically supplied. A debug line is connected to the CDH Workstation, in order to control the processor and monitor its activity.

The DL processor also is located on a development board externally power supplied. This processor is controlled and monitored by a DL Workstation through a debug line between the Workstation and the development board.

The communication line between CDH board and DL board is provided by small cables called jumpers that links the correct pins. In particular one jumper links CDH TX to DL RX, one jumper links CDH RX to DL TX and the third jumper is used to link grounds.

Execution

Each workstation commands to its processor to execute a program that does the following actions:

- A packet of established data is created. One field of this packet is filled with the CRC code (32 bits).
- The UART serial communication is set.
- A loop is periodically executed:
    - The packet is sent through the transmission serial.
    - The received buffer is read and the packet is decomposed. The software verifies if the packet is recognised searching for the specific bytes of header and closer.
    - If the packet is recognised, the software verifies if it's validated or corrupted. The algorithm calculates the CRC code and compares it with the received one.
    - If the packet is validated, the data contained in it are saved in a specific structure. Each field of the structure is printed on workstation's screen.

Results

Both CDH and DL workstations' screens shows the packets sent in most of cases, as expected. In this case they have been correctly send, received, recognised, and validated.

Discussion

Sometimes the receiver device reads more bytes than the packet's ones. This problem has been highlighted and solved during the Test 5 execution.

Test 5

Objective

The objective of this test is to verify the number of packets correctly transmitted between CDH and DL boards. This durability tests allows to verify the reliability of communications.

Setup

CDH processor is located on the development board that is electrically supplied. A debug line is connected to the CDH Workstation, in order to control the processor and monitor its activity.

The DL processor also is located on a development board externally power supplied. This processor is controlled and monitored by a DL Workstation through a debug line between the Workstation and the development board.

The communication line between CDH board and DL board is provided by small cables called jumpers that links the correct pins. In particular one jumper links CDH TX to DL RX, one jumper links CDH RX to DL TX and the third jumper is used to link grounds.

Execution

Each workstation commands to its processor to execute a program that does the following actions:

- A packet of established data is created. One field of this packet is filled with the CRC code (32 bits).
- The UART serial communication is set.
- A loop is periodically executed:
    - The packet is sent through the transmission serial.
    - The received buffer is read and the packet is decomposed. The software verifies if the packet is recognised searching for the specific bytes of header and closer.

- If the packet is recognised, the software verifies if it's validated or corrupted. The algorithm calculates the CRC code and compares it with the received one.
- If the packet is validated, the data contained in it are saved in a specific structure. Each field of the structure is printed on workstation's screen.

The software has some counters that memorise the number of received packets, of which ones of them whose header and closer are recognised, of which of these last ones are validated with CRC. The software shows on screen them on each loop with also the throughput value, that is the ratio between the number of uncorrupted packets and the received ones.

Results

The test has been performed for more than 16 minutes. On every loop each packet has been received, recognised, and validated on both CDH and DL sides.

A representative loop example of CDH side is shown below.

```
time: 0 days 0 hours 16 min 54 sec
Bytes inviati = 217
Bytes letti = 160
Header detected in position 0
Closer detected in position 160
header: D2C
time: 0 days 0 hours 16 min 56 sec
status: 0       0
pack num: 0
operative mode: 0
last comm: 0
PS telemetry:
DL telemetry:
crc: 3719841611
closer: END
packets received: 916   recognised: 916  uncorrupted: 916        throughput: 1.000000
```

Figure 38: ESA-uProp 3 CDH-DL communication throughput test results

Discussion

Many tests have been conducted and the software has been checked many times before obtaining a satisfactory result. In fact it has been noticed that a wrong setting in serial configuration leaded to a wrong reading of received buffer: transmitted data bytes with decimal value of 10 were read as a double byte with values of decimal 10 and 13. The reason is that UART protocol uses special control bytes according with imposed settings that must be the same for receiver and transmitter. This issue used to lead a low throughput and lack of communications for many consecutive rounds, an unacceptable condition.

3.3.8 CDH acquisition test

One of the main functions of the CTP is to acquire parameters to evaluate the interaction between the PS and the platform. The CTP must correctly acquire these data.

On avionic boards there are particular acquisition circuits called ADC (analogue to digital converter) that allow to read signals of sensors that are linked to them. These sensors provide a certain voltage value that is proportional to the parameter they are measuring. These ADC are linked to others electronic components called MUX (multiplexer) in order to serialize these data.

The ADC are connected via SPI to a micro-processor. The SPI is a serial communication protocol where a device called master can communicate with other devices called slaves. The master coordinates the communication interrogating one slave per time and waiting for its answer.

In particular the CDH must be able to acquire data from the sensors that are linked to the ADC on the CDH board and on the EPS board. The sensors regarding the CDH ADC are all temperature sensors, while the ones regarding the EPS ADC are of different kinds: temperature, current and voltage sensors.

Test

Objective

The CDH microprocessor must be able to correctly set the SPI line with the two ADCs.

The microprocessor must be able to dialogue with ADCs and obtaining information from sensors.

Setup

The CDH processor is mounted on the CDH board. The CDH board is mounted on a Flat Board, a special device that allows to connect different boards through the 104 pins connector.

This board is electrically supplied by an external electrical source distributing power on electrical buses implemented on the 104 pins connector.

The EPS board is mounted on the Flat Board and is alimented. The Flat Board also provides a data connection (SPI) between CDH board and EPS board through the 104 pins connector.

A debug line is connected to the CDH Workstation from CDH board, in order to control the processor and monitor its activity.

Execution

The executive program on CDH is launched by the workstation. The program executes the following test operations:

- Set the SPI line.
- Periodically (in loop):
    - Acquires data from ADCs.
    - Acquired data are saved in a structure.
    - Saved data are shown on the workstation's screen.

A test sensor is linked to every ADC connection (one per time) and the respective acquired measure is checked on the Workstation's screen.

Results

The SPI line is correctly set, no error messages appear during this part of the test.

Every ADC's measure is correctly acquired and its value changes when the test sensor is linked.

Discussion

The software tested, in particular the parts used for parameters acquisition, is similar to the one of the previous project (ESA-uProp 2). This test has been particularly useful in testing the modifies that have been progressively applied.

3.3.9 storage test

Objective

The goal of the test is to verify that data are correctly encoded and stored in both memories of CDH.

The expected result is the reading of correct data on both memories after the decoding of files log.txt.

Setup

The CDH processor is located on a developing board electrically supplied. A debug line is set towards a workstation trough a UART connection. The workstation uses a minicom terminal in order to communicate with the processor.

Execution

The software is launched and runs for ten loops, saving encoded data on both memories in files log.txt.

Then a specific program (log_dec.exe) is launched in order to read data from both memories, decoding and printing obtained data on screen.

Results

No error messages appeared on the screen during the execution of test.

The decoded data are the same of the saved. This demonstrates the correct execution of the whole procedure of packaging, encoding and saving.

```
MISSION TIME:  0 d 7 h 55 m 47 s
NO acquisition of ADC on EPS board
NO acquisition of ADC on CDH board
Can't find the magnetometer device
NO acquisition of Magnetometer

MISSION TIME:  0 d 7 h 55 m 48 s
Enter POWEROFF mode



/ # ./log_dec
SD
0 d     7 h     55 m    47 s
reboot 33
cdh temp 65535
FLASH
0 d     7 h     55 m    47 s
reboot 33
cdh temp 65535
```

Figure 39: ESA-uProp 3 CDH data storage test result

The acquisition of ADCs and Magnetometer is not executed during this test, and the software correctly reports this event.

Discussion

During the pre-qualification test campaign the storge aspect will be again tested and storage data will be verified in post process analysis.

3.3.10 CDH Software assembly and final product

After that all parts of CDH software have been separately tested, it's necessary to integrate all parts in a unique executable file. The correct approach, following a bottom-up approach, is to assembly the entire software adding one part per time:

- Insert the part in the main program.
- Verify that there aren't errors during compilation, such as missed declarations of variables or functions.
- Test the correct execution of the whole program, focusing on the last added part.

These actions are repeated until the whole program is ultimate.

The final result will be the complete integration and verification of the software, whose flow chart is the following.

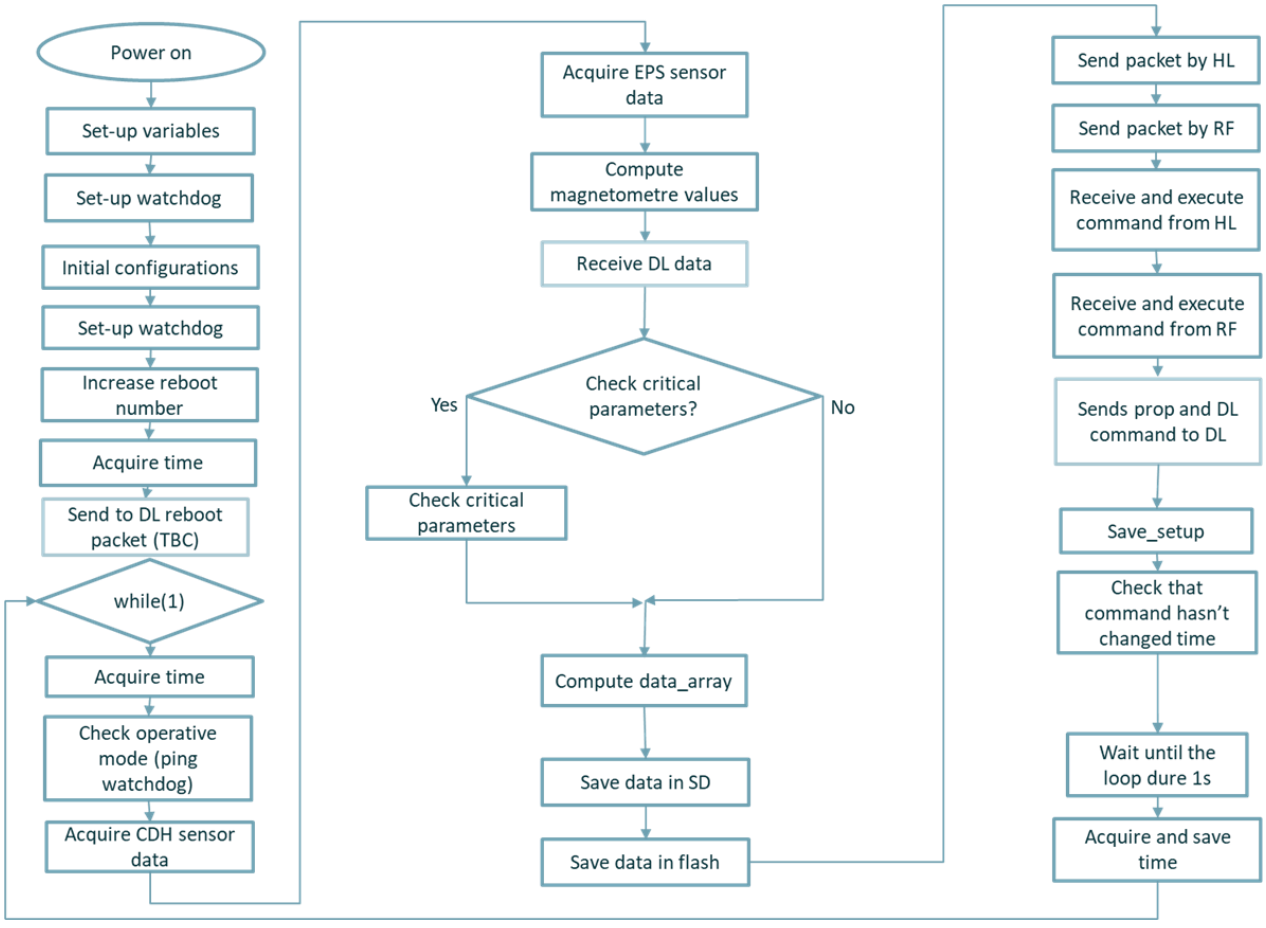

Figure 40: CDH Software flow chart

After the boot of the micro-controller the following tasks are performed:

A. **Initialization, CTP activation**:
   - All libraries, variables, constants and function declarations are set.
   - Load time: last saved time is loaded from files time.txt and time variable is set. Valued from three files time.txt are evaluated and voted.
   - Set-up the interfaces: UART, I2C and SPI communications are initialized, and timer and GPIO lines managed through file system.
   - Load configuration parameters: last saved configuration parameters are loaded from setup.txt file and configuration variables are set.
   - Communication initialization packet: a reboot packet is sent to DL to start the nominal communication between the two boards.

B. **Loop:**
   - Acquire initial time: the time that the loop starts in is acquired.
   - Check operative mode: different actions are actuated according to the operative mode. Watchdog is pinged, or system is shut down or rebooted.
   - CDH telemetry acquisition: CDH acquires data via SPI from its ADC and from the EPS ADC.
   - DL telemetry and PS telemetry reception: the CDH receives a packet from DL containing DL telemetry data end PS telemetry data.
   - Check critical parameters (individually enabled/disabled): all CTP data are checked if they are in the acquisition range for ensuring acquisition faults absence. Then critical parameters are checked if are in nominal range: if not different processes, according to the kind of off nominal parameter, are actuated.
   - Process the data: data are encoded and formatted according to specific protocols (custom for HL data packets and KISS AX.25 for RF data packets), all the configuration parameters and time are updated. All data are inserted in a specific data packet.

- Save data packet in memory: each packet is saved in log files in the E-PROM and SD card memories. The packet is encoded.
- Send data to GSS PC: each packet is sent to GSS PC via RF or HL using specific headers and closers. The packet is encoded.
- Verify the reception of a new command:
    - If Yes, command is managed: Each command is validated, decoded and all the useful information are extracted. Specific sequences are executed according to the received command.
    - If No, flow passes to next stage.
- Packet sent to DL: useful information and, if present, commands towards DL or PS are sent to DL. A specific structure and specific header and closer are used.
- Save configuration parameters and time: all the configuration parameters are saved in configurations (setup.txt) files and time is saved in time1.txt, time2.txt, and time3.txt files. This process is executed for both memories.
- Sleep: final time is acquired and compared to the initial one. The system waits until the loop lasts 1 second.

Conclusions

According with results obtained in development tests, the CTP of ESA-uProp 3 demonstrates to be a reliable product, able to perform many test campaigns with many different kinds of electrical propulsors, expressing a great flexibility.

Compared to the previous project, the platform will have more reliable communication techniques, able to detect and in some cases correct errors. The CTP will be able to better manage off nominal events, such us off nominal critical parameters, or some kinds of hardware failures.

The future of the project provides the completion of DL and GSS software functionalities, and the execution of the AIV plan when all components will be provided by the different collaborating agencies, in order to obtain a working prototype. Then a pre-qualification test campaign will be performed in order to verify the whole system.

The CTP will be used in ESA/ESTEC laboratories that simulate space environment to test different miniaturized electric propulsors and evaluate their performance and their impact on the host platform.

References

1. https://en.wikipedia.org/wiki/CubeSat
2. https://en.wikipedia.org/wiki/Fault_tolerance
3. https://en.wikipedia.org/wiki/Redundancy_(engineering)
4. https://www.baeldung.com/cs/CRC-vs-checksum
5. http://users.ece.cmu.edu/~koopman/pubs/faa15_tc-14-49.pdf
6. Corpino, S.; Stesina, F.; Saccoccia, G.; Calvi, D. Design of a CubeSat test platform for the verification of small electric propulsion systems. Adv. Aircr. Spacecr. Sci. 2019, 6, 427–442, doi:10.12989/aas.2019.6.5.427.
7. Stesina, F. Validation of a test platform to qualify miniaturized electric propulsion systems. Aerospace 2019, 6, 99, doi:10.3390/aerospace6090099
8. Busso, A.; Mascarello, M.; Corpino, S.; Stesina, S.; Mozzillo, R. The communication module on-board E-ST@R-II cubesat. In Proceedings of the 7th ESA International Workshop on Tracking, Telemetry and Command Systems for Space Applications, TTC 2016, ESTEC, Noordwijk, The Netherlands, 13–16 September 2016.
9. Stesina F., Corpino S., Calvi D., A Test Platform to Assess the Impact of Miniaturized Propulsion Systems, Aerospace, Volume 7, Issue 11, Pages 1 – 16 November 2020, DOI 10.3390/aerospace7110163
10. Obiols Rabasa, G.; Corpino, S.; Mozzillo, R.; Stesina, F. Lessons learned of a systematic approach for the E-ST@R-II CUBESAT environmental test campaign. In Proceedings of the 66th International Astronautical Congress, Jerusalem, Israel, 12–16 October 2015.

11. Manente, M.; Trezzolani, F.; Bellomo, N.; Magarotto, M.; Toson, E.; Mantellato, R.; Barato, F.; Pavarin, D. Magnetic Enhanced Plasma Propulsion System for small-satellites IOD development, IAC-18-F1.2.3. In Proceedings of the 69th International Astronautical Congress (IAC), Bremen, Germany, 1–5 October 2018.

12. Stesina, F.; Corpino, S.; Calvi, D.; Pavarin, D.; Trezzolani, F.; Bellomo, N.; Bosch Borras, E.; Gonzalez Del Amo, J. Test campaign of a Cubesat equipped with a helicon plasma thruster. In Proceedings of the 71st International Astronautical Congress (IAC)—The CyberSpace Edition, 12–14 October 2020.

13. Zanola Sergio, ASSESSMENT OF THE IMPACT OF MINIATURIZED ELECTRIC PROPULSION SYSTEMS ON SMALL SATELLITES TECHNOLOGY Master Degree Thesis, Politecnico di Torino, 2019, web.thesis.polito.it

14. Stesina F, Corpino S., Feruglio L., An in-the-loop simulator for the verification of small space platforms, Int. Rev. Aerosp. Eng IREASE, 2017,

15. Garrone Jacopo, Design and verification of the Thermal Control System for a Cubesat equipped with a miniaturized electric propulsion system. Master Degree Thesis, Politecnico di Torino, 2021, web.biblio.polito.it,

16. Visca Marco, Software development of a Cubesat test platform for electric propulsion systems. Development, integration and tests under ECSS guidelines., Master Degree Thesis, Politecnico di Torino, 2018, web.biblio.polito.it,