# Fault-tolerant satellite computing with modern semiconductors

Fuchs, C.M.

# Chapter 1

# Introduction

## Brief Abstract

Modern semiconductor technology has enabled the development of miniaturized satellites, which are cheap to launch, low-cost platforms for a broad variety of scientific and commercial instruments. Especially very small satellites (<100kg) can enable space missions which previously were technically infeasible, impractical or simply uneconomical. However, as discussed in Chapter 2, they suffer from low reliability. Especially the smallest such satellites are typically not considered suitable for critical and complex multi-phased missions, as well as for high-priority science missions for solar-system exploration and astronomical applications [1]. The on-board computer (OBC) and related electronics constitute a significant part of such spacecraft, and in related work, e.g., [2], were responsible for a majority of post-deployment failures, which are further discussed also in Chapter 3.

Indeed, the modern embedded and mobile-market semiconductors used aboard nanosatellites lack the fault tolerance (FT) capabilities of computer-architectures for larger spacecraft. Due to budget, energy, mass, and volume restrictions in miniaturized satellites, existing FT solutions developed for such larger spacecraft can not be adopted. Today, there exist no fault-tolerant computer architectures that could be used aboard nanosatellites powered by embedded and mobile-market semiconductors, without breaking the fundamental concept of a cheap, simple, energy-efficient, and light satellite that can be manufactured en-mass and launched at low cost [3].

To overcome this limitation, in this thesis, we develop a new approach to achieve fault tolerance for miniaturized satellite computers based upon modern semiconductors. The method we use to approach this challenge is to first consider protective measures proposed by science as theoretical concepts, as well as measures that are in use today in the space industry and other industries in Chapters 2, 3, and 4. We consider how these can be utilized to systematically protect each component of a spacecraft's OBC, as well as the software run on it.

A high-level schematic of the components making up a satellite on-board computer is depicted in Figure 1. For each OBC component indicated in this figure, we develop fault tolerance measures that can be used to protect them and describe them in the different chapters of this thesis. To assure that these concepts are effective, we develop them specifically considering the application constraints and requirements of a

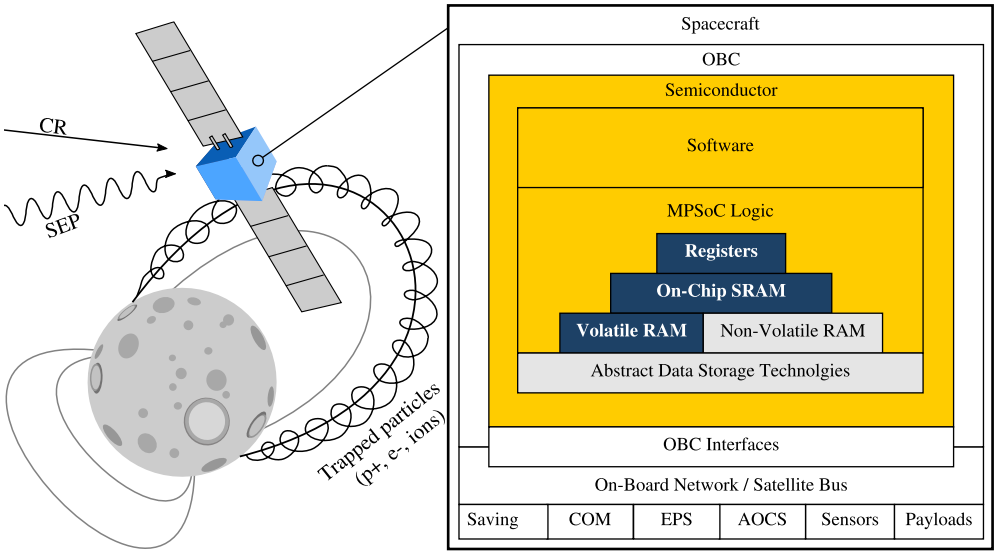


**Figure 1:** A high-level component model of an OBC, and the other subsystems within a satellite interacts with.

satellite operating in the space environment. Based on these concepts, we propose the hypothesis that fault tolerance can be achieved through hardware-software co-design, for which we produce a theoretical design in the form of a three-stage fault tolerance architecture.

We show that by systematically protecting critical key-component of the OBC using software measures, synergies between different fault tolerance measures can be achieved. These synergies enable us to protect the system as a whole more effectively, efficiently and in a way that is economical and feasible even for small-scale professional CubeSat developers and academic teams working on scientific spacecraft and instruments with a limited project budget. We test our hypothesis through fault-injection and provide statistics on the results, and implement a proof-of-concept for this system architecture in a reconfigurable logic device (FPGA).

Our ultimate objective is to allow a suitable miniaturized satellite design to reliably achieve a minimum of 2 years of on-orbit operation. At the time of writing, miniaturized satellite computer components do not include sophisticated fault tolerance capabilities, and may fail at any point in time during a space mission. In contrast to large spacecraft, they therefore can not be designed to achieve a specific mission lifetime, but designs function as long as no critical faults occur. Therefore, these missions are kept brief, as is further discussed in Chapters 2 and 3, thus implying risk acceptance instead of risk mitigation and risk handling.

We realize fault tolerance in software and assure an on-board computer's long-term robustness by exploiting partial FPGA-reconfiguration (see Chapter 5) and mixed criticality aspects (see Chapter 6), and develop a multiprocessor System-on-Chip (MPSoC) architecture through hardware-software co-design (see Chapter 4). Hence, this computer architecture also provides spacecraft designers with the capabilities necessary to achieve a given mission lifetime by adjusting our architecture's parameters, such as the necessary level of replication of software run on the system, provisioning

of spares, scrubbing periods, and error correction coding strength.

The MPSoC requires no custom-written IP-cores (library logic) and can be assembled from well tested commercial-off-the-shelf (COTS) components, and powerful embedded and mobile-market processor cores, yielding a non-proprietary, and open system architecture. The resulting computer architecture consists only of conventional consumer-grade hardware, commodity processor cores, standard parts, and openly available standard library IP.

In the final chapter of this thesis, we provide a proof-of-concept implementation of this MPSoC for three FPGAs, the Xilinx Kintex Ultrascale+ KU3P (the smallest of its class), KU11P, and the Xilinx Kintex Ultrascale KU60. Our implementation for KU3P requires only 1.94W total power consumption, which is well within the power budget range achievable aboard 2U CubeSats. To our understanding, this is the first scalable and COTS-based, widely reproducible OBC solution which can offer strong fault tolerance even for 2U CubeSats.

## 1.1 Problem Statement

Hardware-based fault tolerance measures for large satellites are effective for older, large-feature-size technology nodes which have fallen out of use in the mobile-market and the IT industry decades ago [4]. Modern mobile-market COTS processors depend upon manufacturing in low-feature size technology nodes, and can not be manufactured anymore using old technology nodes. Traditional hardware-implemented fault tolerance techniques diminish in effectiveness and efficiency with shrinking feature size [5]. This has left a protective gap due to a lack of fault-tolerant solutions, and the reliability of such miniaturized satellites is insufficient for critical missions, which is further discussed in Chapter 3.

Countless novel academic fault tolerance concepts have been proposed over the years, which, in theory, could be used to protect modern computer systems. But at the time of writing, there is a significant gap between fault tolerance research, and its applications to spacecraft of all classes, as discussed as part of related work in Chapters 4, 6, and 8. Many of the concepts mentioned there have low technological maturity and do not meet practical application constraints for a use within a real computer system, regardless of the intended operating environment [1]. Software-implemented fault tolerance concepts have thus until today been ignored by the space industry due to lacking maturity, perceived complexity, doubts about their effectiveness and testability [1].

In this thesis we therefore explore how fault tolerance can be achieved for computer systems manufactured in state-of-the-art technology nodes with low power-usage, and small feature-size through scientific means. We do this in collaboration with the European Space Agency, supported by a Networking Partnership Program grant. In this thesis we address the following problem:

RQ0 *Can a fault tolerance computer architecture be achieved with modern embedded and mobile-market technology, without breaking the mass, size, complexity, and budget constraints of miniaturized satellite applications?*

## 1.2 Research Questions

To show that it is indeed possible to address the problem stated in RQ0 in an affirmative way, we develop a fault-tolerant system architecture which can do exactly that. Systematically for each component in a satellite's on-board computer, we develop specific measures to address challenges regarding fault tolerance. These components are also depicted in Figure 1. However, we do not try to apply fault tolerance everywhere in the system as, as this would inflate system complexity and fault potential. Instead, we place fault tolerance measures strategically within the system to handle and cover faults where these can be addressed best at a system level.

In this thesis, we investigate the following research questions throughout the different chapters:

RQ1 Considering the design constraints of nanosatellites, can a fault-tolerant computer architecture be achieved with COTS components?
(Chapter 4)

RQ2 How can the correct functionality of a CubeSat's FPGA-based on-board computer be assured and verified, and its lifetime extended?
(Chapter 5)

RQ3 Can a satellite computer architecture enable novel functionality for a satellite computer, that improves satellite computing beyond just offering better fault tolerance and an increased lifetime?
(Chapter 6)

RQ4 Can commercial memories be retrofitted with error detection and correction in software, to substitute for hardware measures, and to what extent?
(Chapter 7)

RQ5 How can its software-implemented fault tolerance measures of a hardware- software hybrid architecture be tested and validated?
(Chapter 8)

RQ6 Can such a computer architecture be practically implemented within the size, energy, and budget constraints of nanosatellite applications?
(Chapters 9 & 10)

These questions are discussed in this thesis. To do so, we develop a fault-tolerant computer architecture for irradiated environments which can offer protection for on-board computer systems based upon modern semiconductors. Through implementation, testing via fault-injection, and the construction of a proof-of-concept implementation on FPGA, we show that this approach is technically feasible with contemporary technology.

The key contribution of this thesis is a computing concept that can allow future critical commercial and high-priority science missions to be done at low cost, to enable REAL progress in satellite miniaturization to take us as a species to the stars. My hope is that this thesis is the beginning of something new and significant, and in the coming years I plan to advance this technology from its current proof-of-concept state to maturity. To do so, radiation testing, long-term testing, as well as on-orbit demonstration aboard a CubeSat will be necessary.

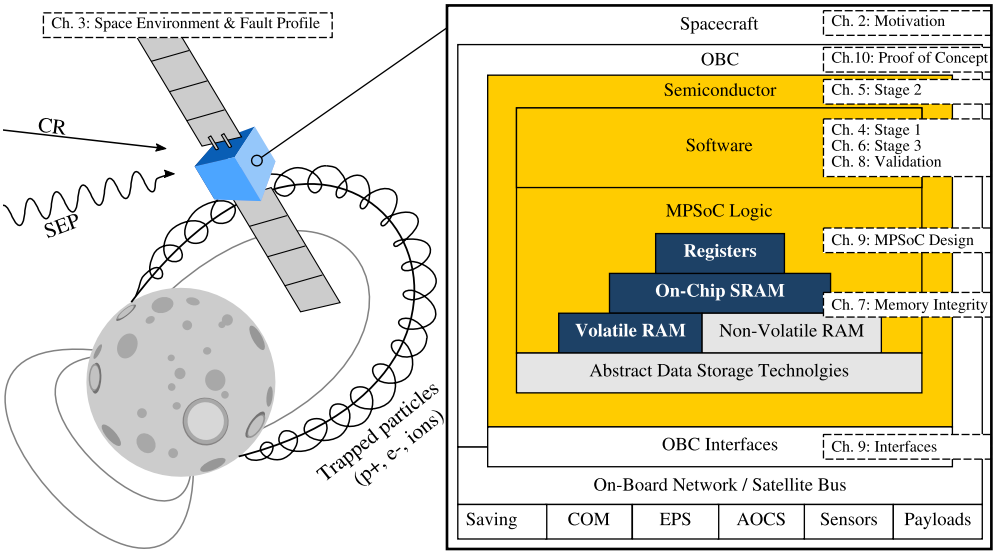


**Figure 2:** Chapter guide for this thesis.

## 1.3 Thesis Organization

A brief outline of the subsequent chapters follows, with a visual chapter guide depicted in Figure 2.

### Chapter 2: A Brief Introduction to Spaceflight and Fault Tolerance

The research upon which this thesis is based is interdisciplinary. It relies upon concepts and results from several different fields, including computer engineering, nuclear science, electrical engineering, physics and astronomy, as well as space engineering. In this chapter, we provide a brief introduction to our application, its design constraints, as well as fault-tolerant computer architecture. We further provide an overview over the current status of small satellite space missions, as well as a review on satellite failures in the past and at the time of writing. This chapter therefore serves also as motivation and legitimization for our research, including mission success and failure statistics, which underline the lack of reliability of very small satellites today.

### Chapter 3: The Space Environment

A satellite's on-board computer has to cope with unique challenges, requiring a general understanding of the physical effects of a spacecraft's operating environment. Hence, for the understanding of the fault profile and application constraints for this thesis, in this chapter we provide an in-depth discussion of the space environment and its effects. We discuss the physical design restrictions aboard spacecraft, and operational considerations. Most importantly we discuss the impact of radiation on semiconductors, and how it can be mitigated.

### Chapter 4: A Fault Tolerance Architecture for Modern Semiconductors

In this chapter, we describe a non-intrusive, integral, flexible, hardware-software-hybrid approach which enable the use of modern MPSoCs for spaceflight meeting real-world constraints. Neither traditional hardware- nor software-based FT solutions can offer the functionality necessary to guarantee fault tolerance for state-of-the-art SoCs used in miniaturized satellite OBCs. We achieve fault-detection, isolation and recovery through the use of a co-designed fault tolerance architecture consisting of multiple interlinked protective measures. In combination, they form a fault tolerance architecture which can guarantee strong fault coverage even during space missions with a long duration, for which we provide an early proof-of-concept implementation. The research in this chapter was published in the proceedings of the *IEEE Asian Test Symposium (ATS)* [Fuchs9].

### Chapter 5: MPSoC Management and Reconfiguration

In this chapter, we present the concept and proof-of-concept implementation of a subsystem for autonomous chip-level debugging within a CubeSat via JTAG [6]. This concept provides all the necessary functionality needed to implement Stage 2 of the fault tolerance architecture described in Chapter 4. In our multi-stage fault tolerance architecture, remote debugging is one of several tasks this subsystem performs: It is now used to control the coarse-grain lockstep implemented within an MPSoC, and referred to as supervisor in remainder of this thesis. It interacts with an on-chip configuration controller to control partial reconfiguration and error scrubbing for the FPGA's fabric via the internal configuration access port (Xilinx's ICAP). An early version of this chapter was presented in the proceedings of the *International Conference on Architecture of Computing Systems (ARCS)* [Fuchs11], and an extended paper [Fuchs10] was published in the proceedings of the *ESA/CNES Small Satellites, System & Services Symposium (4S)*.

### Chapter 6: Mixed Criticality and Resource Pooling

In this chapter, we discuss Stage 3 of our multi-stage fault tolerance architecture, and the advantages it offers not just for miniaturized satellites, but for spacecraft of all weight classes. Our architecture allows a satellite to dynamically adjust the fault tolerance level, compute performance, and energy consumption to meet the varying performance requirements to a satellite computer during long and multi-phased space missions. The operator of a spacecraft can prioritize between processing performance, functionality, fault coverage, and energy consumption. The system can be autonomously adapted to the OBC's thread assignment to retain a functional system core by sacrificing performance or availability of less critical applications. This allows an OBC to to more efficiently handle accumulating permanent faults and to age gracefully. The research in this chapter was published [Fuchs7] in the proceedings of the *NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*.

### Chapter 7: Reliable Data Storage for Miniaturized Satellites

Reliable operation of an OBC can only be guaranteed if the integrity of the OBC's operating system, applications, as well as payload data can be safeguarded. Chapter 7 is therefore dedicated to discussing fault tolerance for the various volatile and non-volatile memories used aboard miniaturized satellites and within our architecture. The research presented in this chapter was published as finalist paper [Fuchs15] in the proceedings of the *AIAA/USU Conference on Small Satellites (SmallSat)*. It was awarded second place and a research grant in the *Annual Frank J. Redd Student Competition*. We describe the implementation of FTRFS, a fault-tolerant radiation-robust filesystem for space use. It was published [Fuchs18] in the proceedings of the *International Conference on Architecture of Computing Systems (ARCS)*. Furthermore, a protective concept for flash memory and phase change memory is described in the second part of this chapter. It was published [Fuchs16] in the proceedings of the *International Space System Engineering Conference Data Systems In Aerospace (DASIA)*.

### Chapter 8: Validating Software-Implemented Fault Tolerance

In this chapter, we test and validate the software-mechanisms that are the foundation of our fault tolerance architecture by injecting faults into an RTEMS implementation of Stage 1. Traditional computer architectures for space applications are validated using system-level testing. This is viable for systems relying on hardware measures, but unsuitable for testing software due to a lack of test coverage and the expanded test-space. For testing software-based FT measures, a realistic test-setup is considered good practice and required to deliver representative fault-injection results. Therefore, a fault-injection campaign was conducted using system emulation through QEMU into a representative ARMv7a-SoC matching our architecture target, ARM's Cortex-A53, and into a RISC-V-based SystemC-model. Our results show that our lockstep implementation is effective and efficient, and we provide a direct comparison to related work. An early version of this chapter was published in the proceedings of the *IEEE Asian Test Symposium (ATS)* [Fuchs5].

### Chapter 9: Combining Hardware and Software Fault Tolerance

As optimal platform for our architecture, we developed a compartmentalized MPSoC design for FPGA, where Stage 2's partial reconfiguration functionality can be utilized to recover defective parts of the MPSoC. This architecture is designed to satisfy the high performance requirements of current and future scientific and commercial space missions at very low cost, while offering the strong fault coverage guarantees necessary for missions with a long duration. We describe the topology of our multiprocessor System-on-Chip (MPSoC), and show how it can be assembled in its entirety from only well tested COTS components with commodity processor cores. The MPSoC can be implemented using only COTS hardware and extensively validated library IP, requiring no custom logic or space-proprietary processor cores. The research in this chapter was published [Fuchs6] in the proceedings of the *IEEE Conference on Radiation and Its Effects on Components and Systems (RADECS)*.

### Chapter 10: On-Board Computer Integration and MPSoC Implementation

In the final research chapter of this thesis, we discuss practical implementation results for our MPSoC design. We provide detailed resource utilization results for this MPSoC for 3 different FPGAs: Xilinx Kintex Ultrascale+ KU3P (the smallest of its class), KU11P, and the Xilinx Kintex Ultrascale KU60, for which we are collaborating within the Xilinx Radiation Testing Consortium to achieve a suitable device-test platform for radiation testing in the future. We provide statistics on power consumption, and show that even between two FPGA generations power consumption can be reduced drastically through the use of more modern and efficient technology nodes. This serves as proof-of-concept for our architecture. This chapter is based on two publications [Fuchs1,Fuchs2] in the proceedings of to the *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)* and the *AIAA/USU Conference on Small Satellites (SmallSat).*