Fault-tolerant satellite computing with modern semiconductors

Fuchs, C.M.

English Summary

Modern semiconductor technology allows the construction of miniaturized satellites, which are cheap to launch, low-cost platforms for a broad variety of scientific and commercial instruments. Especially the smallest and lightest satellites can enable space missions which previously were technically infeasible, impractical or simply uneconomical. In particular satellites constructed as CubeSats can be manufactured rapidly at low cost, with the limited resources available in academic environments. However, today such spacecraft suffers from low reliability. Hence, they have up until now mainly been used for less critical and low-budget missions, where risks can be taken.

Many sophisticated scientific and commercial applications can today also be fit into a miniaturized satellite form factor, which make a much longer mission duration desirable. Theoretically, such spacecraft could also be used in a variety of critical and complex multi-phased missions, as well as for high-priority science missions for solar system exploration and astronomical applications. However, due to their low reliability, these spacecraft have until now been used only as companions to accomplish secondary tasks.

Modern electronics constitute a significant part of such spacecraft, and make up several of their most critical subsystems. Considering their lower weight, these electronics must be lighter, smaller, and offer a better performance-per-watt ratio than traditional space-grade components. Thus, all advanced CubeSats today utilize cutting-edge industrial embedded and mobile-market derived computer designs. At minimal cost, these offer an abundance of performance, require less energy, and are easier to work with than their space-grade counterparts that have a long legacy of use.

However, conventional systems-on-chip-based computers also lack the fault tolerance capabilities of computer-architectures aboard larger spacecraft. In related work, subsystems using these components were determined responsible for a majority of failures after spacecraft were launched and deployed in space. Due to budget, energy, mass, and volume restrictions in miniaturized satellites, existing fault-tolerant computer solutions developed for such larger spacecraft can not be adopted.

As of 2019, there exists 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. Miniaturized satellite developers are, thus, left with the following options:

Upscaling: Resort to utilize traditional space-grade components. This usually requires upscaling of the spacecraft design to a larger form factor, as such components require more energy and offer less functionality, flexibility, and processing performance. In practice, this drastically increases cost, manpower requirements, and satellite development times. Hence, this approach is not constructive for most novel mission concepts centered

around utilizing specifically spacecraft that can be developed rapidly, or which have to be kept small, expendable, or cheap.

SpareSats: Mitigate the risk of early failure by deploying one or multiple SpareSats to replace a CubeSat once it has failed. In practice, this not only increases costs, but also makes failures more likely as the total number of components launched is increased. Hence, this approach only becomes viable after a sufficient level of robustness can be achieved. Today this approach is only viable for constellation missions where satellite generations are replaced continuously at a rapid pace (e.g., Planet Lab), and individual satellites with an exceptionally abundant budget (e.g., MarCo).

Acceptance: Accept the lack of reliability. Keep the mission brief in the hope of achieving all main objectives, before the spacecraft eventually fails by chance. For future miniaturized satellite missions with a longer duration, hope, faith, and luck should not be factors upon which systems engineering is based.

When this thesis was written, developers of most miniaturized satellite missions were forced to follow this third option. For very simple and brief CubeSat missions, this approach resulted in success more often than not, but also in many early failures. However, gambling against time and clinging to hope to not be impacted by environmental effects in the wrong moment is unacceptable, and increasingly less tolerated by governments, space agencies, and investors. To ensure success for advanced long-term CubeSat missions, better, more reliable system architectures are required. Hence, fault-tolerant concepts are needed that are suitable for on-board computers based on modern commercial semiconductors.

This Thesis and its Results

To overcome the technological deficits that impact the use of very small satellites today, in this thesis a new fault-tolerant computer architecture is detailed. It is suitable for integration even into light scientific CubeSats, which are based on modern commercial semiconductors.

To develop the architecture presented in this thesis, results and concepts from a wide range of science and engineering fields are used. The expertise involved in developing this architecture transcends both science and engineering individually. Instead, we combine the best of both of these worlds: we integrate scientific advances, conceptual knowledge, and theoretical notions, with the practical implementation and thorough testing that is standard in the fields of space and electrical engineering.

To make the research contained within this thesis accessible to both scientists and engineers, Chapters 2 and 3 are intended as an informal introduction and definition of the fault-model considered in this thesis. Chapter 2 contains a brief overview over key aspects of spaceflight today, for readers who are unfamiliar with this topic. It serves as motivation for this thesis. The chapter also introduces concepts related to fault-tolerant computer design. In order to design and develop a fault-tolerant on-board computer architecture that is actually effective and efficient, it is crucial to understand the effects of the space environment on a computer. Chapter 3 thus details

these effects, design constraints for space electronics, and operational considerations during space missions, such as communication times, and celestial mechanics.

Based on the preceding chapters, in Chapter 4 we present a fault-tolerant on-board computer architecture which combines software implemented fault tolerance concepts with FPGA reconfiguration and mixed criticality. This is further complemented with several other, more conventional fault tolerance and error correction measures. Fault tolerance in this architecture is implemented as several interlinked stages that allow an on-board computer to age gracefully.

To enable all this functionality, we utilize a software-implemented coarse grain lockstep, which is described in detail in Chapter 4. This functionality alone offers strong fault tolerance capabilities, but would be insufficient for long term missions. Therefore, in Chapter 5, we describe how reconfigurable logic can be used to recover a defective system from a broad variety of faults. We utilize FPGA reconfiguration to assure the integrity of a system-on-chip design, in order to extend the useful lifespan of an on-board computer, and to maximize the fault coverage potential of spare resources. In space missions with a very long duration, defective parts of an FPGA will eventually no longer be recoverable through reconfiguration. Hence, the amount of intact programmable logic available within an on-board computer diminishes overtime. In Chapter 6, we show how mixed criticality can enable a computer to adapt to degradation, instead of failing spontaneously as traditional systems do. We can use this functionality to trade performance for power-saving and robustness autonomously at runtime. This allows the flight software core functionality to be safeguarded as faults occur, achieving graceful aging and pooling spare resources to maximize survivability.

All of this functionality exists as software. It is run on a multi-processor system-on-chip that is implemented within an FPGA. Software, payload information, and the logic programmed into an FPGA are data, the integrity of which must be safeguarded during the entirety of a space mission. In Chapter 7, protective concepts for the different memory technologies present aboard a modern satellite are described.

Previous software-based fault-tolerant concepts applicable to modern semiconductors often sound nice in theory. However, these turn out to be impractical for real-world application. To date no such fault tolerance architecture has been practically implemented and validated, but doing so is a critical step. We take this critical step in Chapters 8 through 10 of this thesis.

The lockstep functionality used in our architecture is validated using Fault Injection in Chapter 8. In Chapter 9, we describe a practical multi-processor system-on-chip design for implementation on an FPGA that serves as an ideal platform for said architecture. We then dedicate Chapter 10 to the practical implementation of the concepts and designs described in the previous chapters. Thereby, we show how an on-board computer with this architecture can look like in the real-world, using a breadboard-based proof-of-concept constructed from development boards. This was done for the following 6 Xilinx FPGAs:

- Kintex UltraScale KU60,
- Kintex UltraScale+ KU11p, KU3p, the KU5p of a Xilinx KCU116 development board, and the
- Virtex UltraScale+ VU9P of a Xilinx VCU118 development board.

For three of these FPGAs, KU60, KU11p, and KU3p, we provide detailed power and utilization data.

Conclusions

At the start of this thesis, we raised the question:

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?

A PhD, many published research papers, and several catastrophes later, it is now possible to answer this question in the following way:

Yes. *A fault-tolerant computer architecture for miniaturized satellites is technically feasible with contemporary consumer- and industrial-grade technology. Once fully implemented as a prototype, it can be used to expand the lifetime of modern day CubeSats drastically, thereby enabling their use in critical and long-term space missions.*

The software-components of the architecture presented in this thesis can be implemented in a non-invasive manner. They provide protection for preexisting applications, without the need to custom-write them to support this architecture. Using real-world software, we show that these mechanisms can detect faults rapidly and with a high probability, and that we can successfully recover from faults at low computational cost in most cases. We demonstrate that the performance cost of this architecture is economical, and remains effective even when operating in exceptionally heavily irradiated regions of space.

With contemporary commercial components, a system-on-chip design that serves as ideal platform for this architecture can be implemented even on the smallest Ultrascale+ FPGA with just 1.94W power consumption. Hence, this on-board computer architecture can be applied to satellites as small as 2U CubeSats.

As the architecture scales with technology, advances in semiconductor manufacturing in the next generation of FPGAs will make this approach even more appealing, and also usable to protect smaller spacecraft. It can improve efficiency and scalability when implemented aboard heavier spacecraft that we use today for high-priority science and solar system exploration. And maybe in the future, hopefully, we can explore even what lies beyond its boundaries.