

DYNAMIC PARTIALLY RECONFIGURABLE COPROCESSOR
FOR
LOW-POWER, RADIATION-TOLERANT COMPUTER

by

Hezekiah Ajax Austin

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

Doctor of Philosophy

in

Electrical Engineering

MONTANA STATE UNIVERSITY
Bozeman, Montana

November 2025

©COPYRIGHT

by

Hezekiah Ajax Austin

2025

All Rights Reserved

TABLE OF CONTENTS

1. INTRODUCTION	1
Satellite Classifications	2
Commercial-Off-the-Shelf	3
Launch Costs	3
Paradigm Shift	4
Small Satellite On-Board Computing	5
Governmental	6
Military	8
Commercial	9
Summary	10
2. BACKGROUND & MOTIVATION	12
Radiation Tolerance in Small Satellites	12
Types of Radiation	12
Types of Radiation Protection	14
Power Consumption in Small Satellites	17
Power Budget	18
Power Generation	19
Energy Storage	20
Computational Performance in Small Satellites	22
Categories of On-Board Computers	22
Sample of On-Board Computing Systems	25
Cost Considerations in Small Satellites	33
Factors Influencing Costs in Small Satellites	34
Small Satellites Trade Space	36
Small Satellite On-Board Computer Examples	38
Flying Laptop	38
GomSpace Express-3	40
PhoneSat 1.0	41
EnduroSat	43
CUAVA-1	44
MarCO	46
On-board Computer for Small Satellite Examples	47
3. RESEARCH AIMS	49
Solution Motivation	49
Problem Statement	51

TABLE OF CONTENTS – CONTINUED

Research Objective	52
Research Questions	52
Coprocessor Research Questions	53
Printed Circuit Board Research Questions	53
Proposed Solution.....	54
4. RESEARCH APPROACH.....	57
Design Process	58
Methodology	59
Problem Identification and Motivation.....	60
Artifact Design.....	60
Rigorous Knowledge Contribution	61
Evaluation Metrics	62
Ranking	64
Computational Performance Scoring.....	64
Radiation Tolerance Scoring	69
Power Consumption Scoring	70
Cost Scoring	72
Communication.....	73
Artifact Development.....	73
Comparison.....	74
5. RESEARCH ACTIVITIES	78
Artifact Definition	78
Coprocessor: Floating-Point Unit.....	79
Initial Coprocessor Design.....	79
Coprocessor Data Storage.....	81
Coprocessor Simulation	83
Coprocessor Integration.....	85
RadPC & Coprocessor Integration.....	89
RadPC & Coprocessor Resource Usage.....	93
RadPC & Floating-Point Coprocessor Limitations.....	95
Hardware: Single Board Computer	97
Prior Work	97
Design	98
Hardware Development Summary.....	102
Coprocessor: First-Difference Edge Detection	103
RGB-to-Greyscale converter.....	106

TABLE OF CONTENTS – CONTINUED

Horizontal Edge Detection	107
Vertical Edge Detection	108
First-Difference Coprocessor.....	109
First-Difference Power Estimation and Monitoring.....	113
First-Difference Resource Usage	117
Coprocessor: First-Derivative Edge Detection	118
Algorithms.....	118
Configuration & Operation Timing.....	120
Image Processing.....	122
Limitations	125
Reliability and Environment Analysis	126
Markov Modeling for Coprocessor	127
Radiation Environment for Coprocessor.....	130
Reliability Analysis for Coprocessor	131
6. SCORING, COMPARISON, & DISSEMINATION.....	134
Scoring	134
Computational Performance Scoring.....	134
Flying Laptop Scoring.....	135
GomSpace Express-3 Scoring	136
Nexus One Scoring.....	137
EnduroSat OBC Scoring.....	138
iOBC Scoring	139
MarCO C&DH Scoring.....	139
RadPC Scoring	140
RAD750 Scoring	141
HPSC Scoring.....	142
Radiation Tolerance Scoring.....	143
Power Scoring	145
Cost Scoring	147
Comparison.....	148
Scoring System Reliability	151
Dissemination.....	158
7. CONCLUSION AND FUTURE WORK.....	159
REFERENCES CITED.....	161

TABLE OF CONTENTS – CONTINUED

APPENDICES	183
APPENDIX A : Average Pairwise Quadratic Weighted Kappa Python	184
APPENDIX B : Kruskal Wallis Python	188

LIST OF TABLES

Table	Page
1.1 Federal Aviation Administration Spacecraft Mass Classes	2
2.1 Sample of On-board Computing Systems	26
2.2 Examples of Small Satellite On-board Computers.....	48
4.1 Computational Performance Scoring Metrics.....	67
4.2 Power to Mass Ratios for Satellite Mass Classes	72
4.3 Large Satellite Processors	76
5.1 Simulated Single Floating-Point Division Results	85
5.2 FPU Timing	93
5.3 Greyscale Conversion Weights.....	106
5.4 First-Difference Configuration and Operation.....	111
5.5 Estimated FPGA Power	114
5.6 Measured Board Power	116
5.7 First-Derivative Configuration and Operation	122
5.8 Current Limitations and Proposed Solutions	125
6.1 On-Board Computer Comparison	149
6.2 Multiple Raters On-Board Computer Rankings	152
6.3 Average Pairwise Quadratic Weighted Kappa	156
6.4 Kruskal-Wallis	156

LIST OF FIGURES

Figure	Page
1.1 Cost of Space Launches to Low Earth Orbit.	4
1.2 Small Satellite Operator and Mission Type Trends	9
2.1 Examples of Radiation Hardening by Process and Design	15
2.2 The general architecture of the subsystems in a CubeSat	18
2.3 Small Satellite Mission Power	19
2.4 Comparison of cell level specific energy and energy densities of various battery systems	21
2.5 FPGA Hardware Architecture.....	24
2.6 Space Launches with Small Satellite Mass	35
2.7 Cost Breakdown Plots	36
2.8 Flying Laptop.....	39
2.9 GOMX-3 Small Satellite.....	40
2.10 PhoneSat 1.0	42
2.11 EuduroSat One	43
2.12 CUAVA-1	45
2.13 MarCO Satellites and Operations.....	46
3.1 Venn Diagram for Research Trade Space.....	49
3.2 Updated Venn Diagram for Research Trade Space.....	51
4.1 Design Science Research Cycles.....	59
4.2 The traditional format for testing processor architectures which use a C code compiler.	65
4.3 Power to Mass Ratios for Four Small Satellite Categories	71
4.4 Design Science Research Cycles.....	74
5.1 RadPC+Coprocessor integrated model	80
5.2 Optimal data storage location	82

LIST OF FIGURES – CONTINUED

Figure	Page
5.3 Simulation of floating-point division operation	84
5.4 Coprocessor placement in computer architecture	86
5.5 RadPC+Coprocessor block diagram	88
5.6 RadPC with floating-point coprocessor implementation	89
5.7 RadPC with FPU implementation.....	91
5.8 RadPC & FPU Resource Utilization Summary.....	94
5.9 Single Board Computer & Softcore Processor with Coprocessor.....	98
5.10 Single Board Computer & Softcore Processor with Coprocessor.....	100
5.11 Experiment Test Setup	104
5.12 First-Difference Hardware Accelerators Operations on MSU logo.....	110
5.13 Configuration Timing	111
5.14 Test images for edge detection coprocessor.....	113
5.15 Hydra on-board FPGA power for first-difference with manual PR	115
5.16 Comparison of Partial Blocks for Coprocessor on 100T FPGA and 200T FPGA	117
5.17 First-Derivative algorithm processing MSU logo	120
5.18 Standard set of test images for first-derivative edge detection coprocessor	123
5.19 Test images for first-derivative edge detection coprocessor.....	124
5.20 Markov Model of Simple System	126
5.21 Markov Model of Coprocessor System	128
5.22 Markov Modeling of Coprocessor Reliability	131
5.23 Markov Modeling of Coprocessor Reliability	132
6.1 RadPC+Coprocessor Patent	158

ABSTRACT

The proliferation of small satellites increases the need for on-board computers balancing computational performance, radiation tolerance, power consumption, and cost. Within this trade space, Field Programmable Gate Arrays (FPGAs) are increasingly used as processors because of their versatility in hardware implementations of custom coprocessing functions. FPGA's partial reconfiguration capability enhances versatility and performance by dynamically swapping functionality of the hardware. The dynamic swappable coprocessor approach was tested using a coprocessor implemented as a peripheral in a reconfigurable region for a 32-bit RISC-V processor, RadPC. During operation, partial reconfiguration swapped functionality of the coprocessor in real-time. Three coprocessors were developed: a floating-point unit, first-difference edge detection, and first-derivative edge detection with multiple hardware functions per coprocessor. These were evaluated on five criteria: 1) functional operation using the dynamic reconfigurable approach; 2) computational performance; 3) power consumption during operation; 4) radiation tolerance of the coprocessor system; and 5) cost of overall system. Functionality was evaluated using simulated results, observed results, and visual examination of outputs. Computational performance was evaluated via logic analyzer measurements, including partial reconfiguration latency. Power consumption was estimated for the FPGA using Vivado Power Analysis. Empirical power measurements were taken from the single board computer's power supply and on-board power monitoring circuitry during operation. Radiation tolerance was evaluated with reliability models of the proposed approach compared with published reliability specifications of small satellite computers. Cost was evaluated by comparing the cost of RadPC+Coprocessor approach with the cost of other small satellite on-board computer options. Testing demonstrated that the proposed method was successful at computing floating point functions and image processing algorithms as a series of dynamically swappable coprocessor functions. Empirical measurements showed the coprocessors were dynamically swapped in 2.5 s via partial reconfiguration. Power consumption for the entire system during operation averaged 2.1 W, peaking at 2.3 W for data intensive operations across all three coprocessors. The trade space analysis showed this approach has potential to optimize computational capability in small satellites within power and cost constraints while achieving acceptable reliability. The comparison shows the RadPC+Coprocessor concept matched or exceeded the capability of an average small satellite on-board computer.

INTRODUCTION

Satellites have undergone a transformation from the era of custom-built, long-lead, and long-duration satellites to a new paradigm of mass-produced, short-duration, small satellites [1]. Large satellites are generally expensive, custom systems with a development time in the ranges of 5 to 15 years, and were designed for specific missions such as telecommunications, weather monitoring, or military [2]. These large, complex satellites were built to function for decades with multiple layers of redundancy to ensure reliability in the harsh space environment. Large satellites are primarily developed by governments due to the complexity, launch capability, expense, and time requirements for production and operation [1]. Advancements in launch technology has driven a shift toward smaller, standardized satellites like CubeSats and nanosatellites, which are cheaper, quicker to develop, and designed for shorter duration missions [3]. Since the introduction of the CubeSat standard in 1999, small satellite numbers have exponentially increased to support NASA science and exploration, expand military space capabilities, and enable the emerging commercial space market [4]. This shift has led to reductions in satellite development time, production cost, and launch complexity for small satellites. Parts used in these small satellites have shifted from expensive custom devices to commercial components that use standardized platforms and modular designs to further lower the cost and development time of small satellites [5]. Rideshare programs have further spread launch costs across multiple payloads, which has helped bolster the commercial launch economy. Advances in reusable launch vehicles with dedicated small satellite launch deployers have resulted in a launch cost reduction of nearly 50x from the beginning of the space race to today [6]. All combined these operational and technological advances are allowing small satellites to be widely produced and rapidly

deployed with designs evolving to meet new mission needs.

Satellite Classifications

First, it is important to define the scope of the small satellite terminology which this paper address. Spacecraft are generally classified based on mass and size [7]. Table 1.1 gives the spacecraft class definitions based on mass with the classification of "small satellites" falling between 0.01 kg and 1200 kg [8]

Table 1.1: Adaptation of Federal Aviation Administration Spacecraft Mass Classes [8]

Terminology	Class	Mass (kg)	Mass (lbs)
SmallSat	Femto	0.01 – 0.1	0.02 – 0.2
SmallSat	Pico	0.09 – 1	0.19 – 2
SmallSat	Nano	1.1 – 10	3 – 22
SmallSat	Micro	11 – 200	23 – 441
SmallSat	Mini	201 – 600	442 – 1323
SmallSat	Small	601 – 1200	1324 – 2646
Sat	Medium	1201 – 2500	2647 – 5512
Sat	Intermediate	2501 – 4200	5513 – 9259
LargeSat	Large	4201 – 5400	9260 – 11905
LargeSat	Heavy	5401 – 7000	11906 – 15432
LargeSat	Extra Heavy	> 7000	> 15433

Small satellites have relatively low mass and have typically taken on limited capability missions. This approach aligns with the NASA class D mission, where schedules are typically capped and lower costs are traded for higher risk during development [9]. A widespread type of small satellite is the CubeSat. The four most common CubeSats are the 1U, a 4"x4"x4"

cube satellite, the 2U, a 4"x4"x8" satellite, the 3U, a 4"x4"x12" satellite, and the 6U, a 4"x8"x12" satellite. These are a variation of the standard nanosatellite and microsatellites that are characterized by size instead of mass.

Commercial-Off-the-Shelf

Small satellites were early adopters of innovative COTS technologies as a solution to overcome the limitations imposed by smaller budgets, lower volumes, and limited mass [3]. For the past two and a half decades, most class A/B missions have relied on the BAE RAD750 radiation-hardened processor as their primary computer system [10]. This ultra-reliable processor achieves 400 MIPS of performance and can consume up to 20 W of full system power. There has been a continued effort for the past decade to replace the RAD750 with a technology that has higher on-board processing capability and lower power consumption. This need has become even more pressing of late as the \$200k price tag of the RAD750 is prohibitively expensive to the growing commercial space industry. By adapting and carefully selecting COTS devices for use in a space environment, small satellites were to achieve high performance at increased functional density and low cost through miniaturization and reduced power consumption [3]

Launch Costs

For decades, the high cost of space launches has been one of the limiting factors on the number, size, and length of space missions [12]. With the rise of SpaceX, RocketLabs, and other commercial launch providers, the cost of access to low earth orbit has greatly decreased [11]. This in turn has led to a reduction in spacecraft development costs due to the design re-use opportunities associated with more missions. As shown in Figure 1.1, launch costs are trending down with SpaceX's Falcon 9 and Falcon Heavy greatly decreasing

Cost of space launches to low Earth orbit

Cost to launch one kilogram of payload mass to low Earth orbit¹ as part of a dedicated launch. This data is adjusted for inflation.



Launch cost per kilogram of payload (constant 2021 US\$ per kilogram)

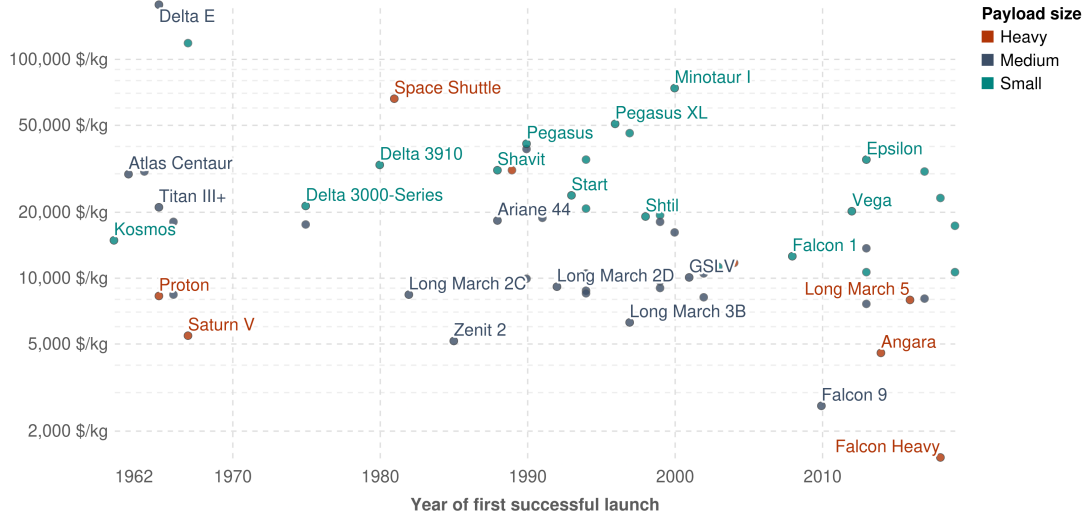


Figure 1.1: Cost of Space Launches to Low Earth Orbit [11]

price per kilogram. From 2000 to 2020, the per kilogram launch costs have decreased at an average annual rate of 5.5% [6]. Coupled with declining satellite mass has resulted in a 10.4% reduction in cost per satellite launch over the same period [6]. The effect of commercial launch providers has been an overall reduction in launch costs for all users [13]

Paradigm Shift

In summary, the satellite paradigm has shifted to reflect the new realities, from custom-designed, long-lead-time, long-mission-duration satellites to mass-produced, short-mission-duration satellites [14]. This transition reflects a move toward rapid development and deployment cycles, enabling more frequent technology upgrades and adaptability to evolving mission requirements. Small satellites, such as CubeSats, are now produced in larger quantities with standardized designs, reducing costs and lead times compared to traditional

bespoke satellites [15]. This shift has democratized access to space, allowing universities, startups, and smaller organizations to participate in space missions [16].

This paradigm shift has also benefitted from the higher performance of commercial off-the-shelf (COTS) components [17]. Advances in COTS electronics, such as high-performance microprocessors and sensors, have enabled small satellites to achieve computational capabilities previously reserved for larger platforms. Coupled with the emergence of rideshare opportunities and dedicated small satellite deployers has lowered the financial barrier of placing small satellites into orbit [18]. These factors have collectively made small satellite missions more accessible and led to an increase in the number of organizations engaged in this area.

Small satellites are now performing increasingly complex tasks, including on-board data processing, intra-constellation communication, and autonomous operation [19]. CubeSats equipped with advanced processors are now taking on the challenge of processing large datasets, such as high-resolution imagery, on-board and reducing the need for extensive ground-based processing [20]. Intra-constellation communication is enabling satellites to coordinate within networks to share data and provide more sophisticated and responsive capability [21]. Autonomous operations, supported by artificial intelligence, further enhance mission efficiency by enabling real-time decision-making without constant ground intervention [22].

Small Satellite On-Board Computing

The paradigm shift toward small satellites has driven their widespread adoption across government, military, and commercial sectors due to their cost-effectiveness, flexibility, and rapid deployment capabilities.

Governmental

NASA has increasingly embraced small satellites to meet mission requirements, leveraging their ability to perform diverse tasks such as Earth observation, communication, and scientific research with reduced costs and shorter development timelines compared to traditional large satellites. This transition is supported by advancements in compact, high-performance avionics systems, enabling small satellites to deliver robust functionality in constrained environments. Reflecting the paradigm shift to small satellites, NASA has increasingly used and encouraged the development of small satellites in science and exploration missions. When it comes to on-board computing within small satellites, existing solutions fall within a spectrum that ranges from high reliability and high cost to unprotected and low cost.

The NASA Space Technology Mission Directorate (STMD) has created a Strategic Framework that details the computing requirements for future missions [23]. The Advanced Avionics Systems outcome of this framework specifically calls for “next generation high performance computing” technology that delivers a “100x increase in computational capacity for the same power utilization of current space-based processors”. It further states that radiation tolerance and fault tolerance are qualities that should be available in any commercial computing solution. Simultaneous to the technical needs from the NASA strategic framework are calls by the 2022 NASA Strategic Plan for STMD to invest in technologies that can reduce cost of future missions to support the growth of the commercial space industry [24]. The RAD750 from BAE Systems has been the primary high performance computer for NASA for the past two decades. The RAD750 is capable of 400 DMIPS at 20 W using a 32-bit PowerPC CPU. Follow-on space qualified processors such as the RAD5545 have achieved 5,200 DMIPS using an SPARC multicore approach. Future HPSC-based systems such as the Moog Cascade Single-Board-Computer (SBC) promises 26,000 DMIPS using a 64-bit RISC-V multicore approach at 25 W [25]. These processors represent the

most reliable, expensive, and power intensive solutions for future missions. At the lower end of performance spectrum are microcontrollers based on the ARM architecture (e.g., the ST Microelectronics STM32 running 250 MHz at 4 W, the Clyde Space Kryten-M3 running 50 MHz at 1 W), the LEON 3/4 FT architecture (e.g., the SWRI Centaur SBC running at 200 DMIPS at 5 W), and the RISC-V architecture (e.g., the Resilient Computing RadPC-SBC-100 running at 50 MIPS at 2 W). These represent solutions for low-power embedded computers used for sensors and in small satellites [26, 27].

The largest class of emerging space computers are System-on-Chip (SoC) solutions that promise slightly higher performance than microcontrollers while still preserving ease-of-programming by including CPU hardware on the same fabric as the FPGA. Companies using FPGA SoC style solutions include Ramon Space (AMD Versal w/ Cortex A72), Xiphos (Xilinx Zync Ultrascale w/ ARM Cortex-A53), Skylabs (PolarFire SoC w/ RISC-V), Unibap (MicroChip SmartFusion w/ RISC-V) and NASA’s Goddard Space Flight Center (SPACECube 3.0 AMD Kintex UltraScale w/ MicroBlaze). These solutions run in the 300 MHz to 1 GHz range with power consumption between 10 – 18 W [26, 27]. FPGAs provide the ability to optimize their hardware for the algorithm they are processing, thus providing heterogeneous compute capability. This optimization has been shown to provide increased energy efficiency and performance for tasks that can exploit massive parallelism and outperform traditional CPUs in both regards. For massively parallel tasks such as convolutional neural networks (CNNs) and machine learning (ML), FPGAs can outperform CPUs by as much as 100x [28–30]. As a result, FPGA-based coprocessors are included within the HPSC computing ecosystem to deliver heterogeneous compute offload to a main rad-hard processor such as the Cascade SBC. Commercial FPGAs have the greatest potential to deliver the 100x compute increase called for by the HPSC subtopic while simultaneously providing the energy efficiency to make spacecraft deployment practical. These solutions are targeted more towards small satellites, where higher risk and lower radiation reliability is

acceptable because of short mission durations.

Military

The United States military relies heavy on satellites to provide communications, reconnaissance, and surveillance to support the warfighter. These satellites operate in harsh environments, including radiation, extreme temperature variation, jamming, and cyberattacks. The number and complexity of satellites is increasing exponentially due to advances in manufacturing and the reduced cost of access to space. This is allowing the military to move towards constellations of satellites, which provide similar or greater capabilities to previous satellites by using small satellite constellations. Historically, the Department of Defense has used radiation hardened processors in its satellites to provide reliability at the expense of performance and cost efficiency. Recently, the computational demand of satellite constellations has focused a shift in focus towards COTS computers instead. However, COTS parts require novel radiation mitigation strategies to keep them operational in space. Additionally, the use of COTS computers opens up the constellations to cyberattacks designed for terrestrial systems.

Towards the goal of putting more computing capability into space, the US Space Force's (USSF) Space Development Agency (SDA) is establishing the Proliferated Warfighter Space Architecture (PWSA) to provide technology and military superiority in support of the warfighter. The PWSA is a low-latency data transport network and military sensing system made up of a multi-layered constellation of satellites with each layer, or tranche, focusing on specific tasks. The PWSA takes a new approach to military space authority by exploiting smaller space vehicles, shorter design cycles (2-year spirals), and commercial parts. This new approach seeks to enable faster and more reliable delivery of data to the tactical edge, including missile warning and tracking. The USSF is continually seeking commercial computing technology that can support the PWSA. Furthermore, the SDA is putting an

emphasis on mitigating “vendor lock” to ensure more suppliers can provide cutting-edge technology for the PWSA.

Commercial

In the commercial sector, small satellites are playing a crucial role in broadband connectivity, Earth observation, and weather monitoring [31]. Broadband conductivity has been a particularly large area of growth as the demand for internet has scaled with both population and worldwide industrialization. SpaceX’s *StarLink* leads in this area with the largest constellation of small satellites. Eutelsat’s *OneWeb* comes in a distant second with roughly a tenth of the number of satellites. Communication satellites represent 79% of all the small satellite launched in 2024 of which Starlink and OneWeb were 72% of total launches [8]. This statistic is reflected in Figure 1.2. The majority of operators of small satellites are commercial companies, with governments coming in a distinct second.

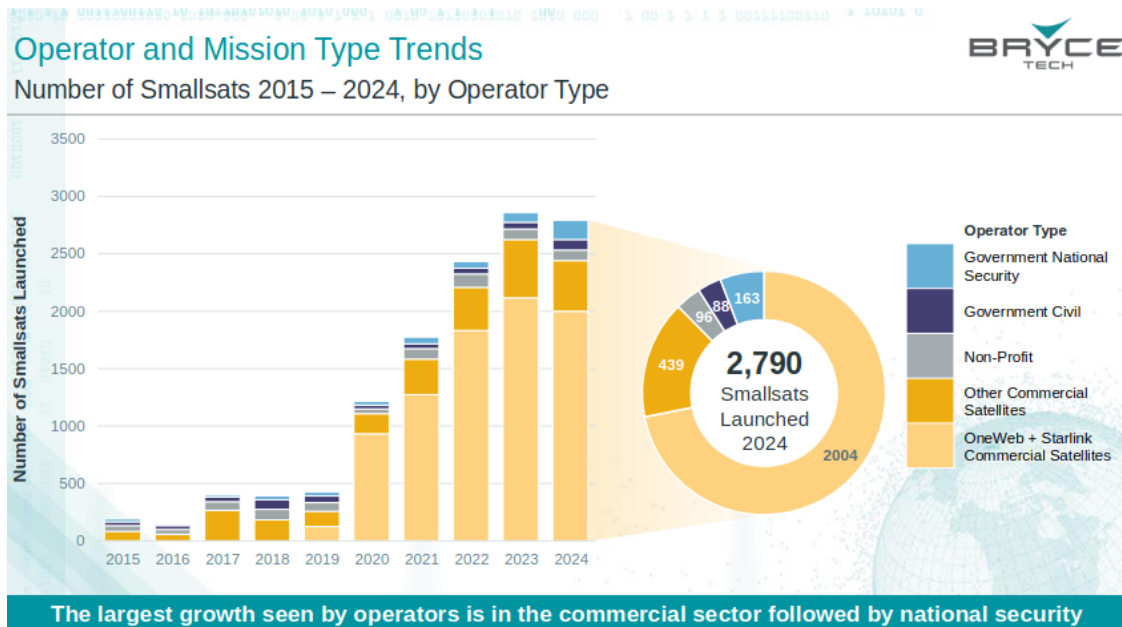


Figure 1.2: Small Satellite Operator and Mission Type Trends for 2024 [8]

For commercial space, the cost of a satellite system has greater impact on the design

than with government or military. A commercial satellite needs to not only perform its mission in space, but the mission also needs to be profitable for the company. A traditional long-duration satellite (10+ year mission) is not ideal when the underlying technology is updating on a 5-year cycle or less. Phone networks and internet connectivity is an excellent example of this trend. Phone networks have updated from the 3G standard to the 5G standard in 18 years [32]. Similarly, internet connectivity has shifted over the course of two decades from the modem dial up with speeds of 14.4 kbps to broadband Ethernet with speeds of over 40 Gbps. This necessitates technology with the capability to support such quick turnover. With internet connectivity, a satellite based provider needs downlink speeds approach 1 Mbps in order to compete effectively. Commercial space has therefore leaned heavily into short-duration (<10 year) missions with cheaper small satellites. Linking the satellites into a constellation provides the greatest capability and widest coverage. Also, the constellation can be slowly upgraded with newer satellite design while maintaining service during upgrading. Commercial satellite orbits also tend to be lower since commercial satellites are primarily focused on Earth observation and communication. These lower orbits are less expensive to reach and have less complex deorbiting procedures.

Summary

Lower launch costs, the adoption of commercial parts, and the move toward satellite constellations has led to a paradigm shift and the widespread use of small satellites across government, military, and commercial space operations. The versatility and low cost of small satellites makes them an affordable option for testing new subsystem designs, instruments, and satellite operations. However, small satellite must operate in the same harsh radiation environment as traditional large satellites, but without the power and budget resources to adopt the custom radiation-hardened processors that have been historically used. There is a pressing demand for techniques to boost computation in small satellite computing while

balancing power consumption, cost, and reliability. This dissertation presents an approach to boost on-board computing using dynamically reconfigurable coprocessors implemented within the framework of a COSTS-based, radiation-tolerant small satellite computer name RadPC. The system is demonstrated using two sets of coprocessor libraries while empirical measurements on performance and power consumption are taken. Finally, a scoring approach is developed to compare this approach to the current state-of-the-art in space computers to illustrate where it contributes within the small satellite computing trade space.

BACKGROUND & MOTIVATION

On-board computers face a number of engineering constraints within the small satellite trade space. Paramount among these constraints is being able to operate in the harsh space environment with exposure to ionizing radiation that degrades electronics. Additionally, there are significant power limitations that must be considered due to the constrained size of batteries and solar panels that can be implemented within a small satellite form factor. Next, computational performance is limited in small satellites by factors including power availability, thermal management, and the impact of radiation hardening techniques. Finally, small satellites are by nature cost constrained, which in turn constrains all other variables within the design including reliability, power availability, and performance.

Radiation Tolerance in Small Satellites

Small satellites generally operate in Low Earth Orbit (LEO) beyond the environmental protection provided by Earth's atmosphere. In these orbits, bombardment by high-energy particles causes a significant number of failures in the computer [33]. While operating inside this harsh radiation environment, computers suffer from two board categories of radiation effects, Total Ionizing Dose (TID) and Single Event Effects (SEE) [34]. Both of these radiation effects are caused by ionizing radiation striking the substrate of an integrated circuit (IC) and depositing energy.

Types of Radiation

TID leads to the gradual degradation of the material within a device. When lower energy protons and electrons (≤ 30 MeV/amu) deposit energy in the substrate, it creates electron/hole pairs in the IC's insulating layers [35]. The two main causes of damage are trapped charge in the gate oxide of a transistor or within the isolation regions between

transistors [36]. Trapped charge in the gate oxide of a transistor will alter the threshold voltage of a device and ultimately put the semiconductor into a state that is either always on or always off. When the trapped charge accumulates inside the isolation region between transistors, it causes leakage current that ultimately to device burn out [37]. Modern IC feature sizes (< 22 nm) are becoming naturally resistant to TID due to the shrinking feature sizes used for the gate oxides and isolation regions. This is due to a lower probability of charge getting trapped in these regions since they spend less time in the thin material [38, 39]. Modern COTS process nodes (< 22 nm) are achieving TID immunity that can surpass 50 krad due to their thin insulation regions [40]. This level of TID immunity is suitable for the majority of LEO missions and is one step towards using COTS parts in small satellites.

An SEE is a rapid spike in charge on a logic line inside a device due to a radiation strike. This is the same charge creation phenomenon as in TID, but occurs in the diffusion, or active, regions of a transistor as opposed to in the insulator materials. In this region of the transistor, the charge creation is due to higher energy particles and heavy ion strikes (≥ 30 MeV/amu). When high energy particles and heavy ions strike the diffusion regions of a device, the energy deposited can be sufficient to be recognized as a logic state change, and are called single event effects. These effects appear as transient pulses in circuitry and logic state changes in memory cells or registers [41]. An SEE does not necessarily cause permanent damage. Destructive SEEs are generally second order effects where the transient pulse can put the device into a lock-up state leading to runaway current draw that damages the device [42]. For non-destructive SEEs, the electron/hole pairs cause unwanted logic-level transitions inside the IC. These logic-level transitions escalate to system faults when the transitions occur on a digital logic line that leads to an error. There are different classes of SEEs. A single event transient (SET) is a temporary state change on the logic line that eventually returns to its original state. If an SET gets stored or latched into a memory device or register, it is called a single event upset (SEU). A single event functional interrupts (SEFI)

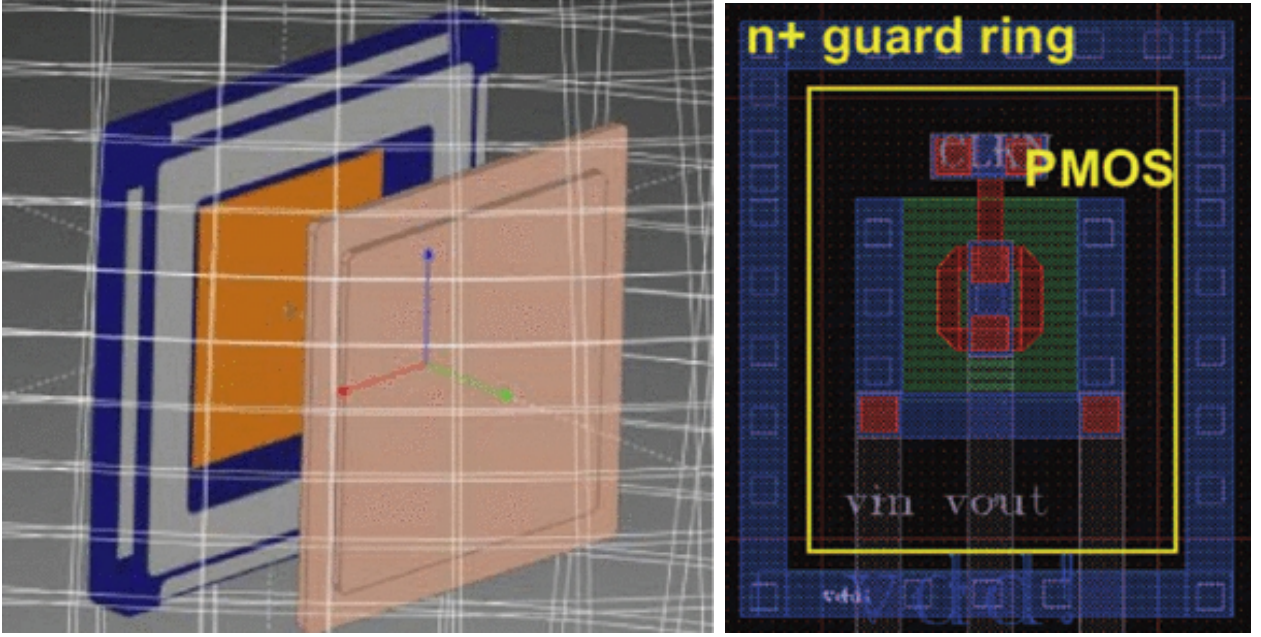
occur when an SET or SEU puts the system into a state that cannot be recovered by resetting the affected circuit and entire system requires a power cycling, full system re-initialization, or reconfiguration.

Overall, radiation protection in small satellites is critical to maintain functionality during exposure to harsh space environments, particularly during space weather effects [41]. To mitigate these challenges, small satellites can employ a combination of strategies to ensure mission success and longevity by enhancing radiation protection, including shielding, radiation hardened components, system redundancy, reconfiguration of programmable devices, and periodic resets. Each method involves tradeoffs with the small satellite designs that is tailored to the specific mission requirements, orbit, risk, and budget.

Types of Radiation Protection

Shielding refers to placing a layer of radiation resistant material over the particularly vulnerable or critical systems to block radiation strikes during operation and generally falls into three categories, structural, component, and spot [43]. *Structural shielding* focuses on integrating shield directly into the space hull, frame, or internal structure of the spacecraft [44]. This method has the highest mass requirement for the shielding in the spacecraft's design. There are design tradeoffs when selecting a shield as radiation strikes on the shield can actually create secondary particles that cause more severe radiation bombardment of the device being protected than if the shield was not present in the first place [45]. As a solution to these issues, composite shielding developed with multiple layers of different materials are being explored [46]. Due to the strict mass and volume limitations of satellites, comprehensive shielding against radiation is often impractical or does more harm than benefit with the case of secondary particles [47].

Therefore, satellite designers prioritize *spot shielding*, or selective shielding, for critical subsystems, such as on-board computers, power systems, or sensors, using lightweight



(a) Radiation Hardened Device Packaging [48]

(b) Guard Ring for Semiconductor [49]

Figure 2.1: Examples of Radiation Hardening by Process and Design

materials like properly-sized aluminum and/or composites. Additional radiation protection can be incorporated into the integrated circuit packaging, an example of *component shielding*, to increase chip radiation tolerance as shown in Figure 2.1a [50]. In many cases, especially for short-duration or low-cost missions, shielding is minimized or omitted, relying instead on fault mitigation procedures for recovery from inevitable faults [51]

Another type of radiation protection is hardening of the components against radiation. These radiation hardened components incorporate a number of different techniques within the physical design and manufacturing process to reduce the impact of radiation strikes. One technique is called *radiation hardening by process* (RHBP) where the semiconductor material is doped with conducting material to reduce the resulting charge from a strike. This greatly increases the energy requirement to produce a logic state change. RHCP can also employ silicon-on-insulator (SOI) substrates to reduce the amount of time a radiation

particle resides within the device, which in turn reduces the amount of charge created. RHBP increases components' radiation protection at the material level, but with the tradeoff of greater expense due to the low volume nature of custom IC manufacturing and reduced performance due to increased capacitance added to the design because of the additional features added to each device [52, 53].

Another hardening technique is called *radiation hardening by design* (RHBD) where charge reduction strategies are implemented at the circuit level and/or at the architectural level of the system. RHBD includes layout alteration techniques such as drain channels to bleed excess charge, guard rings to insert protective electrical barriers around susceptible diffusion regions (shown in Figure 2.1b), and raising the threshold levels of devices so that transients are less likely to cause a state change [49, 54]. RHBD also includes implementing redundancy at the gate level. Redundancy provides identical components or subsystems performing the same function to enhance mission reliability by mitigating failures. Hardware redundancy uses either active or passive redundancy. With active redundancy, the components are duplicated and a voting mechanism selects the correct outputs [55]. Triple modular redundancy uses three complete copies of the functional circuit and is used as it ensures the correct output via majority vote with the lowest amount of hardware duplication [56]. Passive redundancy provides a single duplication of the hardware which switches on when the primary fails [57]. All of these methods use additional hardware to increase reliability in exchange for higher costs, higher power, and lower computational performance. Periodic resets, on a daily or twice daily cycle, are one of the most common methods for adding radiation protection against SEEs to small satellites [58]. The two categories of resets are soft resets and power cycle. Soft resets are software resets of the whole system or the subsystem individually. Soft resets reinitialize the system to the starting point of the software program or state machine [59]. For a power cycle, the system or subsystem is shutdown complete to restart everything from the known off state and clear out persistent

problems [60]. Reconfiguration is particular to FPGAs and use overwrites the configuration memory to reinitialize the hardware to clear out radiation induced faults, specially logic state changes from SEEs [61]. Reconfiguration effectively rebuilds the implemented logic design on the FPGA and places it into a known state. There is a loss of functionality during the configuration process, however no loss of power as required by a power cycle. Radiation hardness and tolerance varies greatly across the range of small satellite missions and depends on the mission class, available budget, available power, and whether the mission requires near 100% uptime.

Power Consumption in Small Satellites

Small satellites missions have grown in length and complexity, driving an increased demand for on-board power collection, storage, and management [62]. Modern small satellites have expanded to more sophisticated objectives, such as Earth observation and global communication network that require sustained power for extended periods. Early small satellite missions were short-term missions relying on solar panels and batteries [63]. Current missions demand robust power collection, storage, and management systems to support frequent data transmissions and high performance payloads, particularly since electrical power system failure is one of the major factors in small satellite failures [64]. This new developing mission set has spurred a shift to compact, high-efficiency solar cells, deployable solar arrays, and high-capacity batteries that enable small satellites to capture and storage more solar energy despite their limited surface area. Combined with standardization of small satellite modular power systems, these developments support the increased number of satellites currently being deployed [65]

Power Budget

A satellite power budget is determined by mission capability requirements and is the deciding factor in small satellite power design. The power budget is calculated through the power utilization and consumption for each subsystem, the power generated, and the power stored [66]. The main subsystems of a small satellite are shown in Figure 2.2 and include an electrical power system (EPS), on-board computer (OBC), attitude determination and control system (ADCS), payload(Payload), and communications transmitter and receiver (COMM) [62]. Each of these subsystems have individual power requirements driven by the satellite's overall mission. Small satellite power systems are shifting to a modular electrical power system of solar panels, a power control unit, and batteries with scalable production [67].

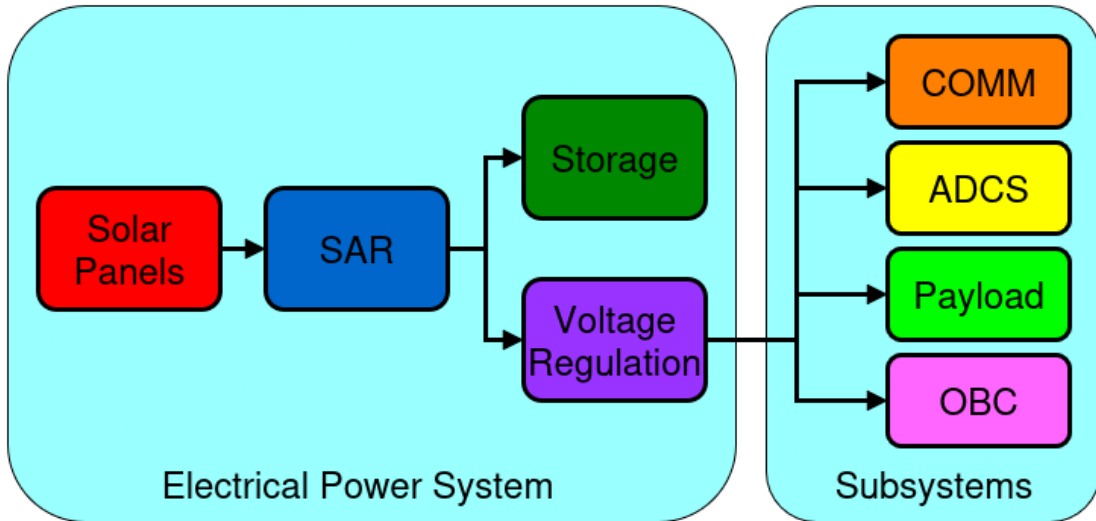


Figure 2.2: Adapted diagram of the general architecture of the subsystems in a CubeSat [68]

The power available for a satellite will fluctuate greatly depending on the orbit, satellite orientation, and energy storage capability. Generally, 1U, 2U, and 3U CubeSats have a maximum power budget ranges of 1 W to 2.5 W, 2 W to 5 W, and 7 W to 20 W respectively [66, 69, 70]. This provides a rough power baseline for CubeSats based off previous satellites

or satellite specifications. This can also be used to size solar panels on CubeSats based on mass of at least 1 W/kg minimum to be able to charge satellite batteries and achieve mission viability [71].

Power Generation

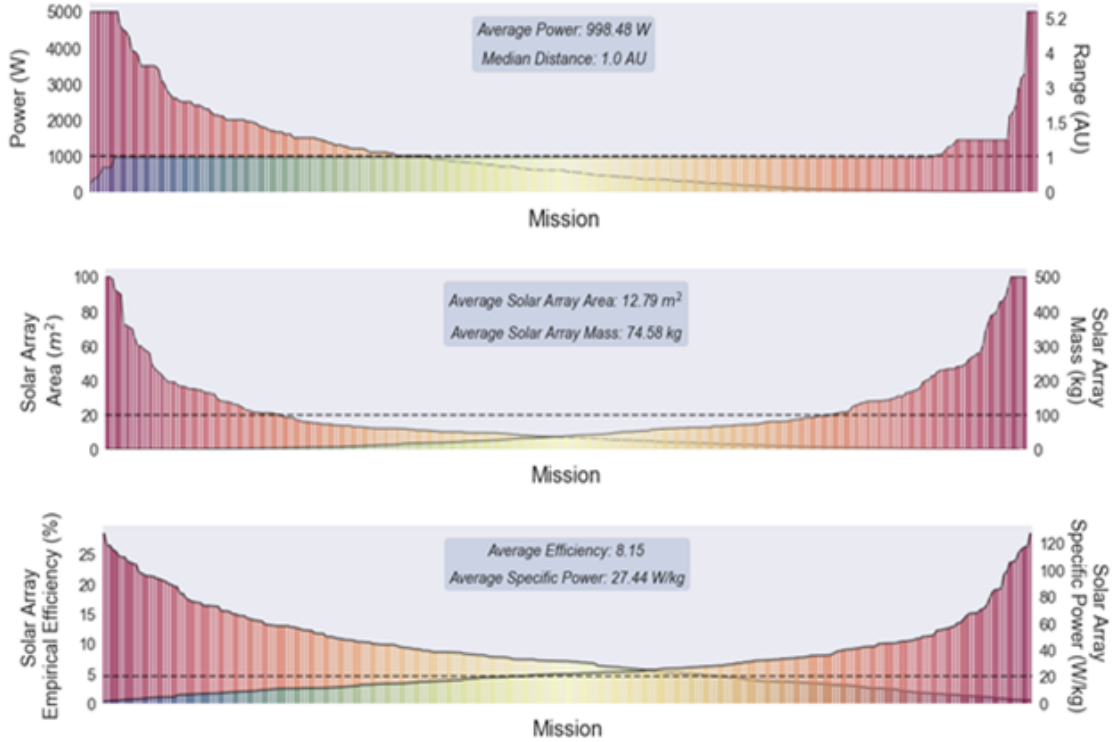


Figure 2.3: (Top) Distribution of mission range, or the furthest point from the sun that the spacecraft reaches, and mission power levels [power capped at 5 KW]. (Middle) Distribution of solar array surface area and solar array mass [mass capped at 500 Kg]. (Bottom) Distribution of solar array empirical efficiency (calculated at Earth) and specific power (for the entire array measured at the destination of the mission). X axis represents 389 missions and color scheme represents highest to lowest values (red to blue) [72].

Power sources for satellites must be small light, rugged, and reliable for a long operational life under the harsh environmental conditions of space. Solar energy converters, mainly solar panels, and nuclear energy devices, primarily radioisotope thermoelectric generators, have proved themselves as practical approaches for satellite power generation

[63]. With small satellites, the low mass requirements have driven designs to a common solar power architecture consisting of solar panels and battery storage for a mission power level cap as shown in Figure 2.3 [72]. The predesigned and qualified solar panel is preferred over designing unique solar panels to reduce the cost and lead times [71]. Solar panels are made from cells of thin semiconductor wafers that produce an electric current when exposed to light [71]. Modern solar panels focus on increasing flexible for resistance to mechanical stresses and radiation protection for survival in harsher radiation environments such as Jupiter and Saturn [73] Degradation from the space environment including deposited heat reduces the efficiency of a solar panel during exposure [74].

Energy Storage

Efficient energy storage is especially critical for handling power demands during prolonged missions that have periods of operation which are eclipsed by the Earth's shadow, particular to support edge computing tasks and maintain battery life [75]. Lithium-ion batteries have been the preferred energy storage option since 2003, following the success of the Mars Exploration Rovers missions where lithium-ion batteries lasted for more than 14 years of continuous operation [76]. Lithium-ion battery technology provided higher specific energy per kilogram than other options at the time as shown in Figure 2.4. Further advancements on this technology include pairing lithium-ion batteries with capacitors to provide for high current draws and faster energy discharge rates [76].

Power management systems have become more sophisticated, incorporating intelligent energy distribution algorithms and fault-tolerant designs to optimize power usage across subsystems [77]. These advancements ensure that small satellites can maintain operational reliability over extended mission durations, even as they handle increasingly complex tasks.

The growing complexity of small satellite missions has driven the need for modular and scalable power architectures to accommodate diverse mission requirements [78]. Modern

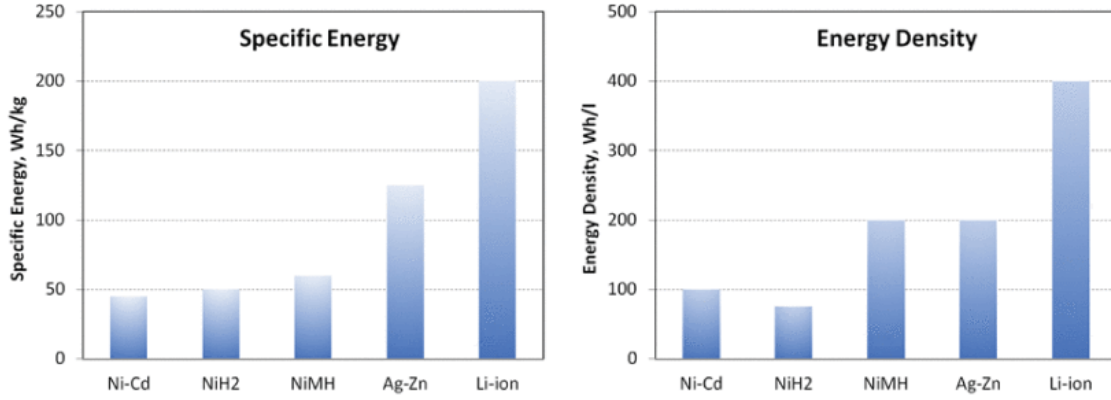


Figure 2.4: Comparison of cell level specific energy and energy densities of various battery systems [76]

power management systems include programmable controllers that dynamically allocate energy to prioritize critical functions, such as propulsion or data processing, based on real-time mission needs. As small satellite constellations expand to provide global coverage for applications like internet connectivity and climate monitoring, the demand for reliable, high-capacity power systems continues to grow, pushing the boundaries of engineering innovation in the small satellite industry. In the future, there are plans to have these constellations share power via a satellite power grid which can provide power to satellites using energy transmission over radio waves or lasers [79].

Small satellite designs have evolved from short-duration missions using a basic solar panel and battery power system to long-duration missions requiring more advanced power system. Small satellites have increased the number of solar panels and have shifted to deployable solar panels on more complex designs. Lithium-ion battery technology has been applied to small satellites to provide a higher specific energy per kilogram. More complex power control and distribution systems have been implemented to provide greater reliability and longer operation.

Computational Performance in Small Satellites

The computational performance of small satellites, such as CubeSats and nanosatellites, is a critical factor in their ability to execute complex missions within the constraints of size, weight, power, and cost. Unlike traditional large satellites, small satellites operate with limited on-board resources, necessitating highly efficient computational systems. Advances in miniaturized electronics, such as low-power microprocessors and FPGAs, have significantly enhanced the processing capabilities of these platforms. For instance, modern CubeSats often employ processors like ARM Cortex-M series or radiation-hardened versions of COTS microcontrollers, which balance performance with energy efficiency. These systems enable small satellites to handle tasks such as data processing, attitude control, and communication with ground stations, despite their compact form factor.

Categories of On-Board Computers

Small satellite on-board computers can be categorized into two main types, commercial and aerospace, with four major subtypes based on processor architecture; 1) traditional processors; 2) radiation hardened processors; 3) FPGA-based processors; and 4) hybrid processors that combine multiple architecture elements from the previous three categories [80]. A new subtype of graphical processing unit (GPU) is also forming as small satellites have become an avenue for implementing machine learning and artificial intelligence in space. Commercial on-board computers are those provided as a finished product from the market. Satellite components are built around these commercial computing solutions using standardized interfaces and flown. These commercial computers have the advantage of access to cutting edge technology, particularly for artificial intelligence and machine learning [81]. Aerospace on-board computers are custom solutions either developed internally by the satellite manufacturer or by an aerospace company for space applications and provide a

number of radiation protection features. Traditional processors are COTS microcontrollers that have been the preferred option for CubeSat on-board computer systems in the Low Earth Orbit [80]. Typically, these microcontrollers are selected for their low power consumption and cost-effectiveness. A common example is the ARM Cortex family of processors which are widely used in CubeSats [82]. However, the limitation of microcontrollers is their processing capability for more complex operation and reliably under higher radiation levels from solar storms or higher orbits [83]. Radiation hardened processors are a subtype of aerospace computer that focus more on radiation protection than computational performance or cost. For a higher cost, these radiation hardened microcontrollers and processors provide increased reliability when compared to commercial microcontrollers by adopting RHBP and/or RHBD techniques. For example, Gaisler GR712RC provides a dual core LEON3 processor which is radiation-tolerant to 300 krad [84]. Another subtype of aerospace computers are FPGA-based on-board processors that apply the benefits of FPGAs to digital signal processing, matrix operations, machine learning, and hardware configurability to small satellites. The FPGA has emerged as a commercial technology that has the promise to increase the computational performance of small satellites [85]. These devices provide programmable interconnected computer elements, such as logic elements, memory, digital signal blocks, and look up tables, in a single integrated circuit as shown in Figure 2.5.

FPGAs offer advantages by allowing the design of customized applications that provide the flexibility and adaptability of software with the speed of custom hardware [87]. Recovery strategies for radiation protection can be implemented into FPGAs to increase the reliability of the design. The final subtype of aerospace computer is a hybrid processor that combines different processing units, such as CPUs+GPUs or CPUs+FPGAs into a SoC or system on module (SoM) [80]. CPU+FPGA is the most widely used SoC for small satellites [88–90]. The hybrid processor provides a tradeoff where the FPGA or GPU acts as a high performance node for computationally demanding tasks and the CPU acts as the

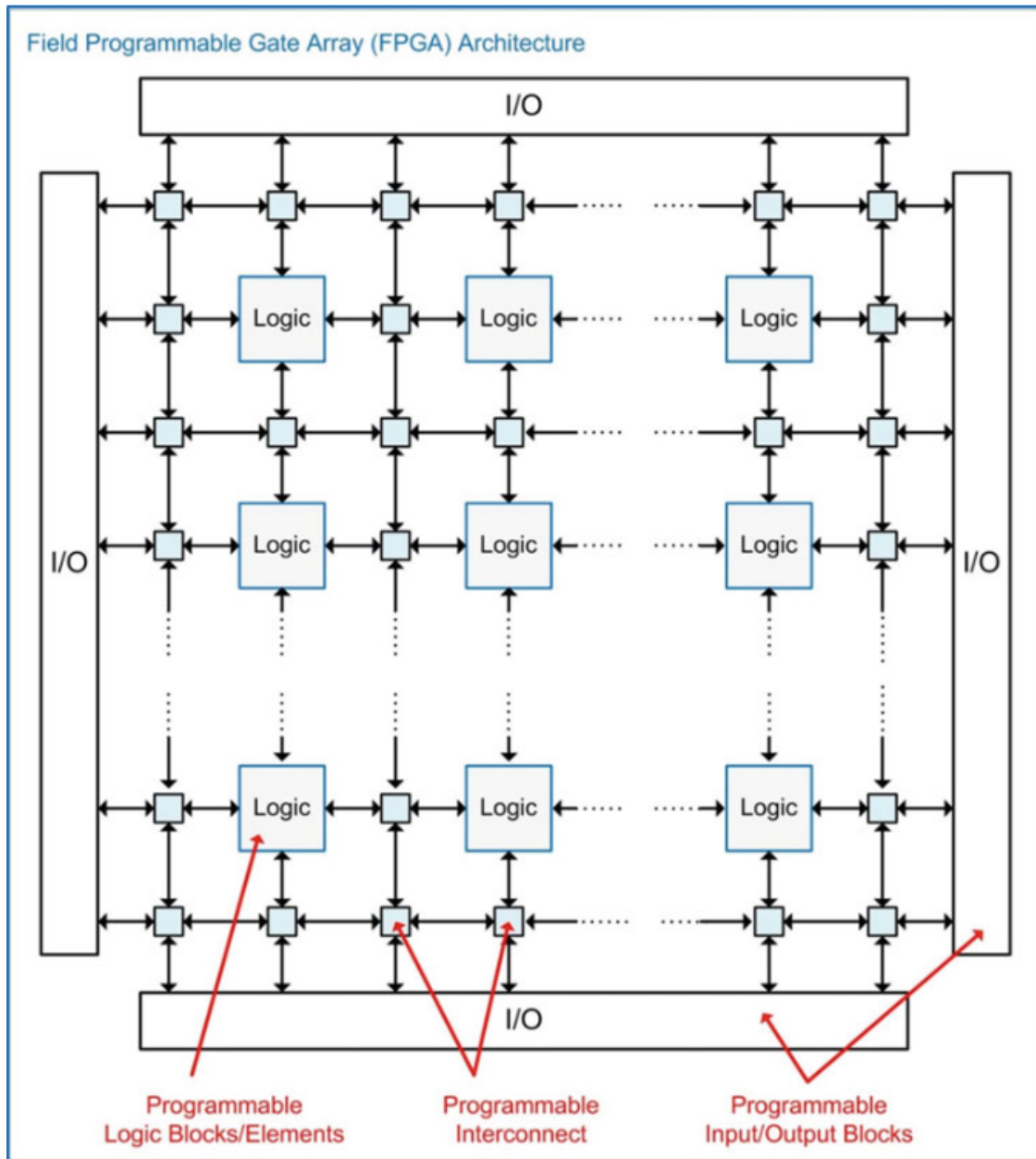


Figure 2.5: FPGA Hardware Architecture [86]

control node and external supervisor. SmallSat missions have a constrained level of funding compared to larger spacecraft missions, therefore purchasing state-of-the-art, radiation-hardened processors is often cost infeasible [91]. A hybrid processor can lower costs by only radiation hardening the monitoring CPU and by providing the high performance computing

nodes with less protection. For example, the NASA SpaceCube processors use several FPGAs paired together to provide both radiation protection and high computational performance [92]. The Kintex UltraScale FPGA functions as the high performance node with radiation tolerance and the Microsemi RT ProASIC3 FPGA provides the radiation hardened monitor for reliability [93]. The second most likely hybrid SoC processor is the CPU+GPU variant, which uses the GPU hardware for machine learning. For example, University of Georgia's Small Satellite Research Laboratory developed a companion board to provide an Nvidia Jetson GPU for rapid image processing in parallel to the on-board computer handling flight operations [94]. Hybrid processors are frequently used for short-duration, high computational operations, such as image processing or machine learning, as required.

Sample of On-Board Computing Systems

To compare the different approaches shown by commercially available boards. A list of small satellite computers with the computational performance is available in Table 2.1. This table is an adaption of NASA's Sample of On-Board Computing Systems from the State-of-the-Art report for 2024.

Table 2.1: Adaptation of NASA's Sample of On-board Computing Systems [26]

Company	Processor	Type	Result	Clock & Memory	Radiation Hardness	Power (W)	Ref
AAC Clyde Space	Smart Fusion 2 SoC including ARM Cortex-M3 Processor	Traditional	62.5 DMIPS	50 MHz, MRAM 16MB	TID 20 krad (system level)	0.4	[26, 95]
AAC Clyde Space	4 Core LEON4FT 32-bit Fault Tolerant Processor	FPGA	~1.4 DMIPS/MHz	250 MHz max/core, SDRAM 4 GBit	TID 20 krad (system level)	1.3	[26, 96]
Alen Space	ARM 32-bit Cortex-M7 with FPU	Traditional	Not Listed	400 MHz or 435 MHz, NAND 1GB, MRAM 8MB, RAM 1.4MB	N/A	0.280	[26, 97]
Aitech Systems	Power PC 1020, Alcatel RTAX	Traditional	~3.0 DMIPS/MHz per core	1.0 GHz, DDR2 1GB, Flash 1GB	100 krad TID, SEL/LET 40 MeV-cm ² /mg (without enclosure)	12	[26, 98]

Company	Processor	Type	Result	Clock & Memory	Radiation Hardness	Power (W)	Ref
Aitech Systems	Intel Tiger Lake UP3 SoC, integrated GPU, Xilinx UltraScale+	Hybrid	Not Listed	4.4 GHz max, SATA SSD 1TB, NVMe SSD 2TB	1 krad TID	CPU Configurable from 12-28	[26]
Aerospace Lab	Xilinx Zynq-7045 with ARM dual-core Cortex A9 and Kintex-7	FPGA	Not Listed	667 MHz, Not Listed	Radiation hardened supervisor MCU and watchdog, Heavy ion (62.5 MeV) with no destructive event	<4	[26, 99]
Andes Aerospace	2x ARM 32-bit Cortex-M7 with FPU	Traditional	~2.14 DMIPS/MHz per core	550 MHz, Not Listed	Estimated TID: 30 krad	~0.1	[26, 100]
Argotec	Dual-Core LEON3FT SPARC V8 Processor + Microchip RTG4	Rad-Hard FPGA	~1.4 DMIPS/MHz per core	250 MHz, 16 GB ECC-corrected mass memory	Rad-hard, 50 krad, SEL immune up to 62.5 MeV cm ² /mg	5	[26, 101, 102]
Berlin Space Technologies	STM32F103	Traditional	90 MIPS, 2x90 + 400 MIPS	72 MHz, 64 or 128 Kbytes	TID 30 kRad	<2	[26, 103, 104]

Company	Processor	Type	Result	Clock & Memory	Radiation Hardness	Power (W)	Ref
BradFord Space	Dual Polarfire SoC	FPGA	1000 DMIPS	Not Listed, Non-volatile memory (1TB+)	30 krad (Si) + functional SEE resilience through SW EDAC	<25 Full Load/ <3 Typical / <1 Idle	[26]
C3S Electronics Development	32-bit ARM Cortex-M7	Traditional	642 DMIPS	300 MHz, RAM 384kByte	Continuous integrity check on the program memory, multi-level watchdog system, MRAM storage	0.46	[26, 105]
CesiumAstro	ARM Cortex	Traditional	Not Listed	1.4 GHz, SDRAM 2 GB, Non-volatile 30GB, FRAM 256KB	Rad Tolerant	8 typ.	[26, 106]
EnduroSat	ARM Cortex M7	Traditional	~2.14 DMIP-S/MHz per core	216 MHz, 2MB RAM and 2MB Flash Memory	40 krad TID	1.5 Peak, 0.2 Idle	[26, 107]

Company	Processor	Type	Result	Clock & Memory	Radiation Hardness	Power (W)	Ref
GomSpace	Xilinx Zynq 7030/7045	FPGA	2.5 DMIP-S/MHz per CPU	800 MHz, 1GB DDR3 RAM	>20 krad	Not Listed	[26, 108, 109]
Ibeos	AMD Ryzen SOC	Traditional	1T FLOPS	3.6 GHz, 16 GB DDR4 w/ ECC, 8 GB NAND Flash, 1Mb FRAM	TID: 30 krad, SEE: > 37 MeV	8-35	[26, 110]
Innoflight	Defense Grade Xilinx UltraScale+ MPSoC	FPGA	Not Listed	Up to 1.33 GHz, 256 KB On-Chip Memory	TID: 30 krad	6-25	[26, 111]
KP Labs	RM57 Herkules and AMD Xilinx Zynq UltraScale+ MPSoC	Hybrid	160 GOPS	1.5 GHz, 8GiB DDR4, 4 GB or 8 GB SLC NAND Flash	COTS with SEE mitigation	<0.5 (DPU is off)- 6 (DPU is on)	[26, 112]
Nara Space Technology	Atmel ARM 9 based Microprocessor Unit	Traditional	Not Listed	400 MHz, 256MB NAND, 256MB DDR2 SDRAM, 32 GB Micro SD Card	<24 krad	0.429 (Idle)	[26, 113]

Company	Processor	Type	Result	Clock & Memory	Radiation Hardness	Power (W)	Ref
NOVI LLC	AMD/Xilinx Versal and Vorago ARM MCU	Hybrid	12 TOPS	Not Listed, 8 GB DDR4	Radiation-hardened Vorago ARM MCU, SEL immune Xilinx processor, Optional radiation-hardened memory, Built in SEU/SEFI	<0.75 (Low Power Mode), 5-50	[26, 114]
Novo Space	Zynq Ultra-scale+	FPGA	~ 1488M FLOPS, ~ 10559M IOPS	1.3 GHz, 4 GB DDR3 SDRAM, 256 Mbytes NOR Flash	TID: 30 krad	max: 15	[26, 115, 116]
Pumpkin Space Systems	BegalBone Black	Traditional	Not Listed	1 GHz, SDRAM 512MB DDR3L, Flash 4 GB	~5 krad	2	[26]
Ramon Space	Space Grade Xilinx Versal ACAP	FPGA	Not Listed	Not Listed, Not Listed	TID: 30 krad, SEL: 43 MeV cm ² /mg	20-90	[26]
Resilient Computing	RISC-V 32-Bit (AMD Artix 7)	FPGA	Not Listed	32 MHz, 4 MB NOR Flash	COTS with SEE mitigation	1.5	[26]

Company	Processor	Type	Result	Clock & Memory	Radiation Hardness	Power (W)	Ref
SkyLabs	4x RISC-V 64-bit / PolarFire / SoC	FPGA	3.125 Core-Marks/MHz, 1.714 DMIPS/MHz	600 MHz, 64kB L1, 2MB L2, 128 GB NAND Flash	COTS with SEE mitigation	~10 Peak	[26, 117]
Space Dynamics Laboratory	GR740 LEON4 + RTG4	Rad-Hard FPGA	Not Listed	250 MHz, SDRAM 256MiB	50 krad	12	[26]
SPiN USA	GR712RC dual-core 32-bit LEON3 fault-tolerant, SPARC V8 processor	Traditional	140 DMIPS/- Core	50 MHz, FLASH 64 MB, RAM 32MB SDRAM 3GB	Processor is 300 krad	1-1.2	[26, 118]
Spiral Blue	Nvidia Jetson Xavier NX	Traditional + GPU	21 TOPS, 6T FLOPS	Not Listed, 16 GB	Processor is 300 krad	7	[26, 119]
Southwest Research Institute	PowerPC based MPC8548e CPU, Microsemi RT-ProASIC FPGA	Hybrid	3056 MIPS @133.3 MHz	1.5 GHz core, 133 MHz local bus, 32 kB, 32 kB	TID: Up to 60 krad (Si), SEL: Immune to >67eV cm ² /mg, SEU: <1 error per 24-hour period	12-20	[26, 120]

Overall, on-board computers are selected based on computational performance, memory, power efficiency, reliability and communication interfaces and are pivotal for executing complex missions within stringent size, weight, power, and cost constraints. Advances in miniaturized electronics, including low-power microprocessors, radiation-hardened processors, FPGAs, and hybrid processors, have significantly enhanced the available processor options.

Cost Considerations in Small Satellites

Cost is a primary driver in the ongoing paradigm shift from large satellites to small satellites. Traditional large satellites demand substantial funding, often costing tens to hundreds of millions of dollars per satellite [124]. Large satellite development cycles are lengthy frequently spanning years or decades due to the complex design requirements, low acceptable risk, and custom components. Comparatively, small satellites leverage standardized designs, such as CubeSat kits, and commercially available components. This approach significantly reduces costs and accelerates development in exchange for lower reliability and performance. It allows small satellites to move from the prototype to operation status in months rather than years [125]. Together the advantages in cost and time advantages have resulted in higher production runs for small satellites with greater than 61.5% of active spacecraft in orbit being considered small satellites [126]. This paradigm shift has turned small satellites into a cornerstone of orbital satellites.

This combination of lower design, development, and launch costs, coupled with the ability to apply mass-manufacturing to small satellites has created a dominant advantage. The emphasis on cost-effective development and deployment has catalyzed a fundamental shift in the space industry. Small satellites are now enabling governments, private companies, and research institutions to access space with unprecedented affordability and efficiency.

Factors Influencing Costs in Small Satellites

The key factors influencing the cost of small satellites are launch costs, design and development costs, and finally acceptable mission risk. The tenets of the small satellite paradigm are low cost, maximum reuse of existing components, lower program oversight, reduced development effort, and wide application of commercial-off-the-shelf technology [127]. This last factor does not directly affect small satellite cost. However, it does indirectly set the design and development, and operational limits for small satellites. Importantly, the higher acceptable risk for class C/D missions, the majority of small satellite missions, allows the use of more affordable commercial-off-the-shelf components.

The reduction of launch costs driven by numerous launch providers, SpaceX, RocketLabs, Blue Origin, and Firefly Aerospace, has further fueled this paradigm shift. Large satellites typically require much larger rockets and dedicated satellite payload housings, which are expensive and logistically complex. Small satellites offer flexibility in launch vehicles, as the lower mass and volume allows them to be deployed as secondary payloads or launched as constellations [128]. These rideshare and mass launch programs allow multiple small satellites to share a single launch and drastically lower the per-unit costs associated with the launch [129]. Small satellite launches have increased dramatically due to this overall launch cost reduction as shown in Figure 2.6. For example, SpaceX launches their StarLink Small Satellites as sets of 21 to 60 using the Falcon 9 rocket [130]. Additionally, advancements in launch technology, particularly reusable rockets has further decreased expenses. Launch costs have substantially dropped twice in the history of the United States space program. The first drop occurred with the development of the Saturn V in support of the Apollo Program. Development of the Saturn V costed approximately \$66 billion in 2020 dollars [131] and lowered the orbital launch costs to 5000 kg [132].

The second drop in launch costs occurred in the last decade and is primary due to SpaceX [11]. SpaceX's Falcon 9 costs \$62 million to launch 22,800 kg to LEO, making the

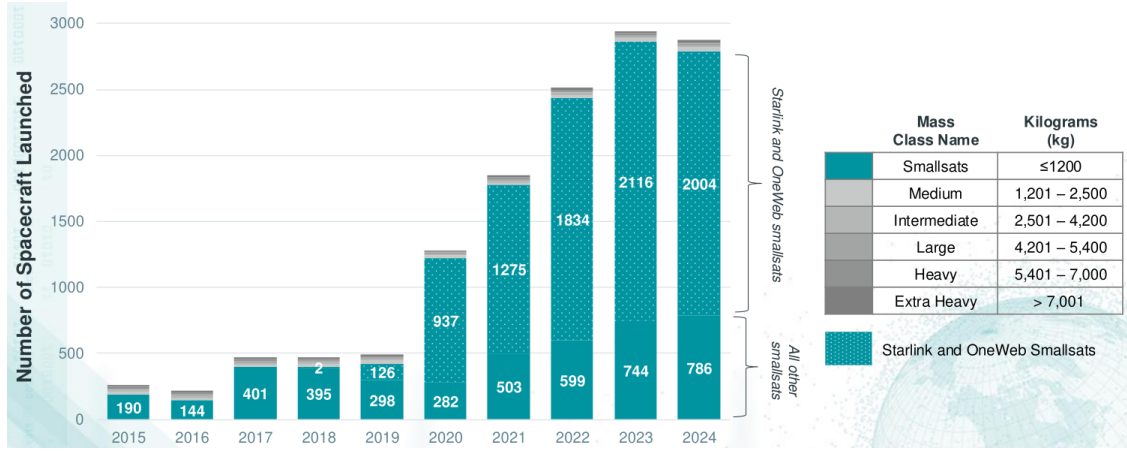


Figure 2.6: Space Launches with Small Satellite Mass for 2024 [8]

cost \$2,720/kg [12]. This specific reduction made small satellites increasingly viable for a wide range of applications in telecommunication and Earth observations.

Cost for the design and development of small satellites breaks into the four categories of administrative, engineering, materials, and labor expenses. The administrative category has seen a number of new cost models from estimating small satellite program costs [133]. The design and development phase are the highest risk point for cost as unforeseen technical difficulties has the greatest potential to cause cost uncertainty than any other single influence [1]. Figure 2.7 shows the subsystem components that majorly influence cost and are comprised of Command & Data Handling (C&DH), Thermal, Power, Structure, ADCS, Propulsion, and Telemetry Tracking & Command (TT&C). Each of these subsystems directly impacts the capability and operation of the satellite.

Cost saving measures for small satellites are standardization and modularity in design. The adoption of the CubeSat standard and modular architecture has allowed individual subsystems to be built out as modules. For example, attitude determination subsystems are now available from multiple vendors as standardized star tracker modules. Major progress in mass, size, and power consumption has reduced the design requirements and enabled the

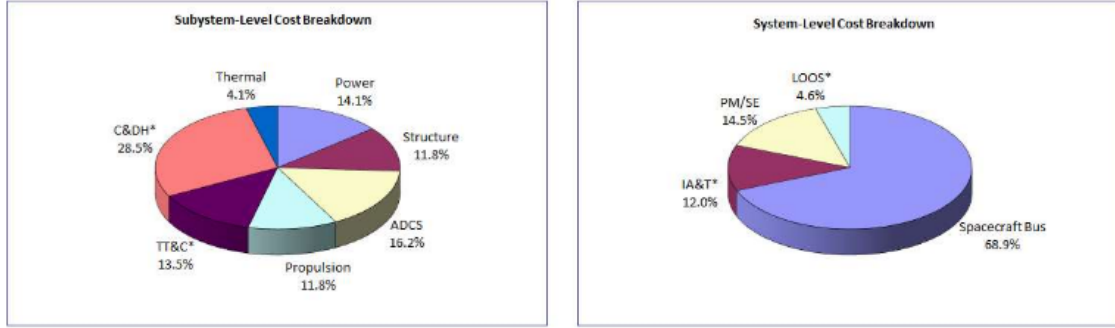


Figure 2.7: Cost Breakdown Plots. The left shows for subsystem-level cost breakdown and the right shows the system-level cost breakdown [1].

use of ADS in small satellites such as PicoSats and NanoSats [134–136]. Another cost saving measure has been the adoption of COTS hardware to reduce development costs.

The final major influence on cost is acceptable mission risk. The majority of small satellites fall under Class C/D missions [137]. Class D is the least costly and lowest priority mission type that are given less stringent requirements. Small satellites have a higher failure rate with 41.3% of all small satellites between 2000 and 2016 failing or partially failing [138]. While small satellite failures have become more reliable, picosatellites and nanosatellites still exhibit high failure rates in either power or communication subsystems following deployment [139]. Overall, small satellites are a tradeoff between cost and reliability. The low cost in development time and resources is exchanged for lower reliability.

Small Satellites Trade Space

Power constraints are a primary challenge in optimizing computational performance for small satellites. With solar panels typically generating limited power, on-board computers must operate within stringent energy budgets. Low-power processors, such as those based on RISC-V architectures, have emerged as viable solutions, offering sufficient computational throughput while minimizing energy consumption. Additionally, power-

efficient algorithms and software optimization techniques, such as selective data compression and task prioritization, are employed to maximize performance. For example, on-board image processing for Earth observation missions often relies on lightweight algorithms that reduce data volume before transmission, thereby conserving both power and bandwidth.

The use of COTS components has revolutionized the computational landscape for small satellites by providing cost-effective, high-performance solutions. However, COTS hardware must be carefully selected and tested to withstand the harsh space environment, including radiation and extreme temperatures. Radiation can cause SEUs that disrupt computational processes, necessitating error-correcting codes and fault-tolerant architectures. Some small satellites incorporate redundant systems or software-based mitigation strategies to enhance reliability without significantly increasing costs. For instance, the use of triple modular redundancy (TMR) in critical computational tasks ensures that errors caused by radiation do not compromise mission objectives, though this approach may slightly reduce available processing power.

Emerging technologies, such as artificial intelligence (AI) and ML, are increasingly being integrated into small satellite systems to enhance computational performance. On-board AI enables autonomous decision-making, such as real-time anomaly detection or dynamic resource allocation, which reduces reliance on ground-based control and improves mission efficiency. For example, satellites like those in Planet Labs' Dove constellation use AI to prioritize image data for transmission, optimizing bandwidth usage. However, implementing AI on resource-constrained platforms requires lightweight models and specialized hardware, such as GPUs tailored for space applications, which are still in early stages of adoption for small satellites.

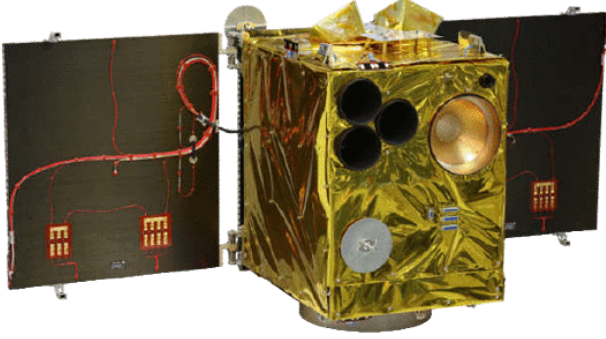
Advancements in computational performance for small satellites focus on scalability and modularity to support diverse mission requirements. The development of standardized computational platforms, such as modular CubeSat avionics, allows for plug-and-play

integration of high-performance processors and sensors. Additionally, the growing trend of satellite constellations, where multiple small satellites work collaboratively, enables distributed computing approaches to share processing loads across the network. This paradigm shift, combined with innovations in low-power, radiation-tolerant hardware and efficient software frameworks, will continue to push the boundaries of what small satellites can achieve, making them increasingly viable for sophisticated scientific, commercial, and defense applications.

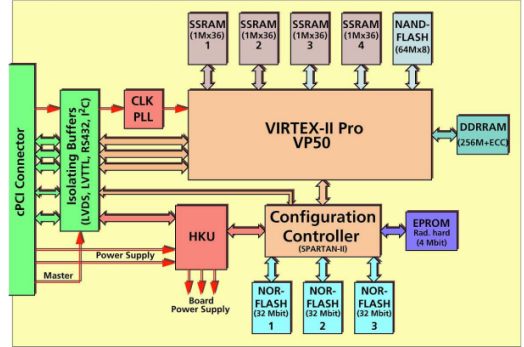
Small Satellite On-Board Computer Examples

Six small satellites were selected as examples from the current design trade space and for comparison of their on-board computer system with the on-board computer with the RadPC+Coprocessor system developed for this research. A variety of radiation protection schemes for the satellites' on-board computers were selected so that they could be evaluated inside the trade space of computational performance, radiation tolerance, power consumption, and cost. A higher level of radiation protection generally correlates with greater power consumption, greater cost, and lower computational performance [140]. The three types of radiation protection are radiation hardening, radiation tolerance, and no radiation protection. This selection provides a fairly wide evaluation of how varying levels of radiation protection impact the functionality and longevity of small satellite systems, paving the way for advancements in space technology and mission reliability.

Flying Laptop Launched in 2017, Flying Laptop is a small satellite developed by the University of Stuttgart as a technology demonstrator to test and verify new technologies, reconfigurable FPGA on-board computer, an unfolding mechanism for the solar panels, a GPS experiment, communication and navigation using an optical data link, and scientific earth observation [143]. Shown in Figure 2.8a, the Flying Laptop is a 110 kg satellite with 270 W of on-on-board power generation through its solar panels [141]. The background



(a) Flying Laptop Small Satellite [141].



(b) Flying Laptop Computer Architecture [142].

Figure 2.8: Flying Laptop: Small Satellite using a radiation tolerance approach for an FPGA-based computer

power consumption was 40 W for satellite operation with an additional 100 W for instrument/payload power. On-board computer system uses four central processing nodes and one command decoder and voter using the Virtex-II Pro XC2VP50 from Xilinx and RTPProASIC3 RT3PE3000L from MicroChip respectively [144]. The central processing nodes used the UT699 processor from the LEON3FT family, designed by ESA/ESTEC as a fault-tolerant softcore processor [143]. LEON3FT is a 32-bit processor core using a 7 stage pipeline computer architecture with separate instructions and data caches [145]. Each central processing node functions as a payload computer [146]. The command decoder and voter functions as the overall satellite computer. In addition to the radiation tolerance of the FPGAs, the command decoder and voter implements a double redundant system using a pair of microblaze soft processors in parallel [142]. This provides an extra layer of radiation protection for the computer system.

This satellite applied the fault recovery method of full configuration of individual FPGAs. The command decoder and voter was implemented on a Flash FPGA using radiation hardened electrically erasable programmable read-only memory (EEPROM) to store the bitstream for a full configuration [144]. This system in turn provides the

configuration controller for the four SRAM FPGAs, each of which has a central processing node implemented on it via full configuration [144]. Configuration is used to flush out SEEs and SEUs during operation. Periodic full reconfiguration is used on individual central processing nodes to preemptively reset the node to a known state and increase reliability [144]. The Flying Laptop is an example of the radiation tolerance approach to small satellite on-board computing with configuration options on the FPGAs to provide task specific processing.

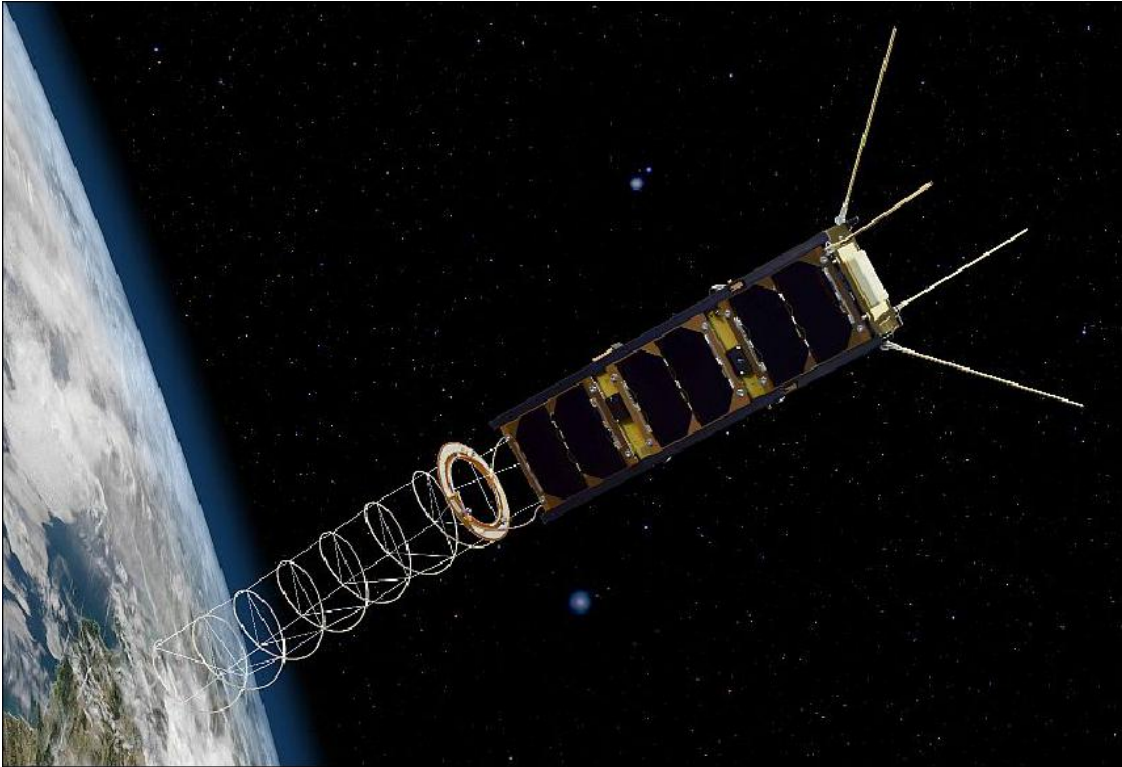
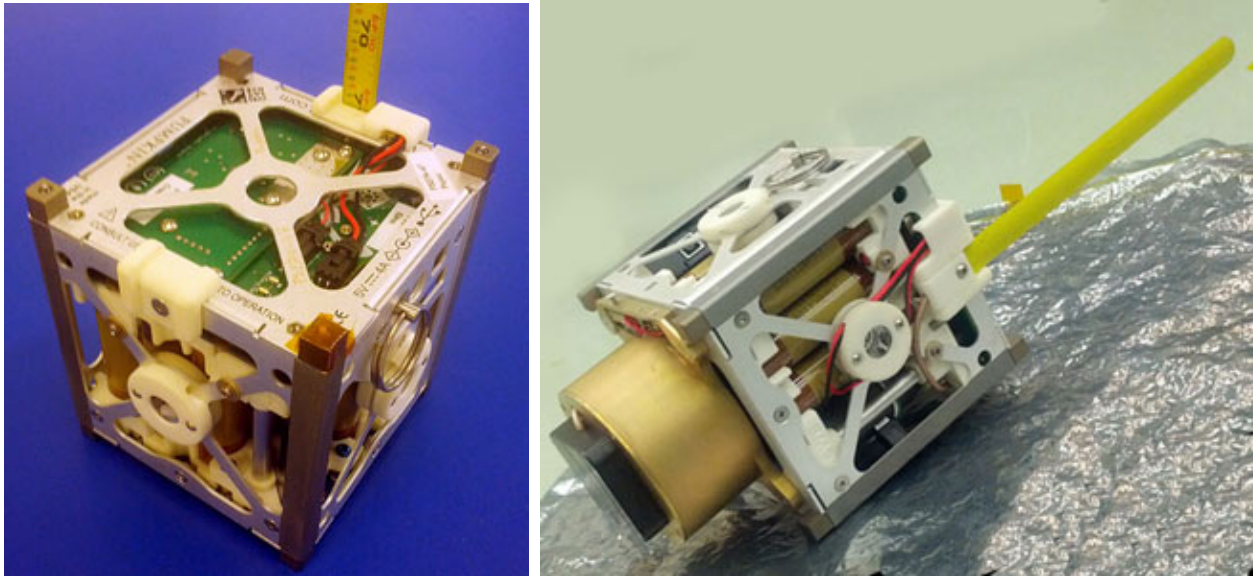


Figure 2.9: GOMX-3 Small Satellite in deployed configuration (Image Credit: GomSpace, NASA) [147].

GomSpace Express-3 The GomSpace Express-3 (GOMX-3) CubeSat was a collaborative mission between the European Space Agency (ESA) and GomSpace which deployed a 3U nanosatellite from the ISS in October 2015 and is shown in Figure 2.9. This CubeSat mission

tested aircraft tracking in remote areas, such as over oceanic regions, using signals from the automatic dependent surveillance broadcasts from commercial aircraft [147]. The satellite used an X-band software defined radio to receive signals from aircraft with information on aircraft identification, position, altitude, and intent, GOMX-3 downlinked the data to provide an update of the aircraft's status in under 5 minutes with a 7 km accuracy. GOMX-3 on-board computer was the NanoMind A3200 designed by GomSpace. This is a combination of an on-board computer with a 3-axis magnetometer, coil drivers, and a gyroscope. The on-board computer is a traditional processor built around the Atmel AT32UC3C microcontroller. Although this microcontroller is not radiation hardened, it does incorporate a memory protection unit that provides a level of protection from radiation strikes in the memory [148]. This microcontroller provides from 8 MHz to 64 MHz processor clock for the 32 bit RISC-V architecture with a floating-point unit (FPU) and 32 MB of SDRAM for memory [149]. The approach used in this on-board computer is to use more industrial grade commercial parts. GOMX-3's robust attitude control extended its lifespan from 6 months to over a year, supporting extended operations like weather data collection for the UK Met Office. The mission's success validated commercial off-the-shelf components and flexible, reconfigurable systems for the ESA and paved the way for additional CubeSat missions like GOMX-4B.

PhoneSat 1.0 The PhoneSat 1.0 mission is the perfect example of the fly it and see approach. Conducted by NASA Ames Research Center, PhoneSat 1.0 was a technology demonstration using two 1U CubeSats, named Graham and Bell. The Graham CubeSat is shown in Figure 2.10a and its partner the Bell CubeSat is shown in Figure 2.10b. Launched on April 21, 2013, the cube satellites were secondary payloads on the maiden flight of the Antares-110 rocket. The mission validated the use of a commercially available smartphone as the on-board computer for the cube satellites. The two 1U cube satellites were built



(a) PhoneSat 1.0 Graham [150].

(b) PhoneSat 1.0 Bell [150].

Figure 2.10: PhoneSat 1.0: A pair of 1U CubeSat built around the HTC Nexus One Phone

around the HTC Nexus One Smartphone and running the Android operating system. The phone functioned as the on-board computer for the satellite with its 5 MP camera being used for earth observation, 3-axis accelerometer and 3-axis magnetometer being used for attitude determination. A watchdog circuit was implemented on an Arduino microcontroller board to monitor the phone and force a reboot if the phone stopped sending radio signals. These cube satellites were solely battery powered without any solar panels and the mission lasted only six days. However, the mission achieved its objective of proving that commercial phone parts were capable of operating in space for a short-duration (<7 days).

The mission was highly successful despite the satellites short lifespan of approximately one week. Both Graham and Bell achieved their primary goals of demonstrating that a smartphone could perform critical spacecraft functions such as command and control, data storage, and imaging. The CubeSats successfully captured and transmitted Earth images using the smartphone camera, and their attitude sensors provided functional orientation

data. The PhoneSat missions were built around the technology industry's "Release early Release often" approach and the focused low costs of <\$3500 per cube satellite [151]. The mission validated the low-cost, COTS-based approach, achieving all objectives within budget and schedule constraints. TID was ignored due to the short-duration of the mission (<1 week) and a reboot watchdog circuit was used to keep the satellites transmitting in the presence of SEEs. Overall, the success of PhoneSat 1.0 paved the way for subsequent missions by proving the viability of smartphones in space applications and contributing to the development of cost-effective nanosatellite technologies.

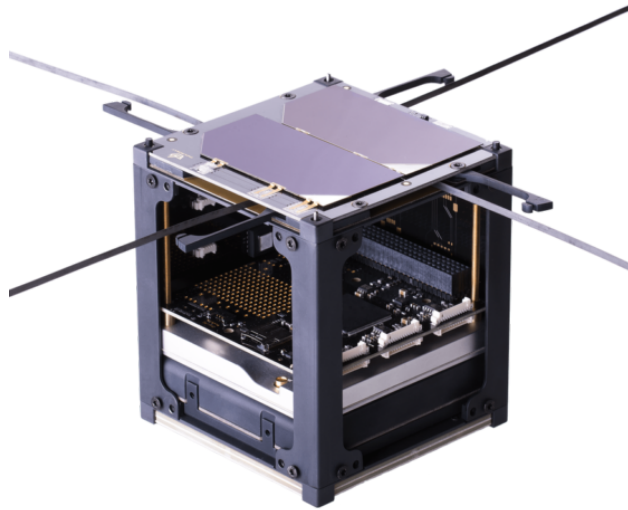


Figure 2.11: EnduroSat One in open configuration displaying antennas and PCB stack [152].

EnduroSat The EnduroSat One mission was a 1U cube satellite launched on May 21, 2018 from the International Space Station. Its development and mission was used to demonstrate cube satellite technologies and support educational outreach via the radio amateur network. This cube satellite was developed by EnduroSat in collaboration with

the Space Challenges Program [152]. EnduroSat One was equipped with an EnduroSat On-board Computer Type I, shown in Figure 2.11, featured a UHF communication system operation at 437.050 MHz with 27 dBm output power, and tested attitude determination and control. During beacon mode operations, the satellite transmitted telemetry and audio, including Beethoven's "Ode to Joy" to the global radio amateur community. The EnduroSat On-board Computer Type I is based on an ARM Cortex M4 with a clock rate up to 180 MHz with 256 kB RAM, 2 MB Flash Memory, and 256Mbit Serial NOR Flash Memory [153]. ARM Cortex M4 is a traditional processor optimized for low power operation and integrates a number of watchdog circuits for fault recovery.

EnduroSat One was a success, achieving nominal performance in beacon mode shortly after deployment, with radio amateurs worldwide successfully receiving telemetry and audio data. After completion of the beacon mode, the satellite transitioned to full system testing of attitude control optimization. EnduroSat One used a compact design with integrated solar panels and a robust communication system. It demonstrated the potential of small CubeSats for educational and technological purposes, fostering global participation and advancing cube satellite technology in Bulgaria. The mission success highlighted the reliability of the EnduroSat hardware and the effectiveness of data collection via an amateur radio network [152].

CUAVA-1 Shown in Figure 2.12, CUAVA-1 3U cube satellite was developed by the ARC Training Centre for CubeSats, UAVs & Their Applications (CUAVA) at the University of Sydney, Australia. This satellite was developed as capability demonstration and payload design validation satellite [155]. Its on-board computer is the ISIS iOBC, which is a 400 MHz, 32-bit ARM9 CPU with 32 MB SDRAM and 2x2 GB SD-card for housekeeping and data handling [156]. iOBC was integrated with a GomSpace NanoPower P31u Electrical Power System with seven solar panels, a CubeSpace ADCS with reaction wheels and magnetorquers,

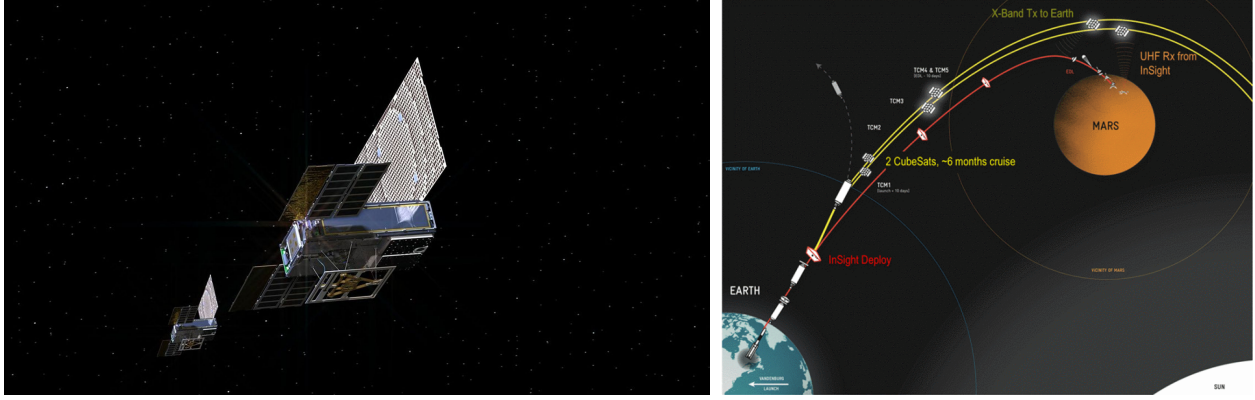


Figure 2.12: CUAVA-1 Mock Picture in Orbit [154].

and an ISIS VHF/UHF transceiver for communications. These mission objectives were primarily educational and technological, as this cube satellite mission focused on training students and engineers in space operations and validating four experiential payloads for Earth observation and imaging.

The mission was a failure with no confirmed signals received CUAVA-1 following deployment from the International Space Station. CUAVA-1 ultimately re-entered Earth's atmosphere on September 2, 2022, without fulfilling its scientific goals. A comprehensive fault root causes analysis identified multiple contributing factors: excessive computational load on the OBC from the operating system and graphical interfaces, vulnerabilities in the SD card and boot sequence, launcher-imposed antenna tie-downs causing deployment issues, EPS compatibility problems with the bus voltage, potential current leakage from payloads activated at boot, insufficient solar panel power margins, and limited pre-launch testing due to COVID-19 disruptions [157]. No definitive single cause was pinpointed, but these issues prevented sustained operations. While not a success, the lessons learned did feed into the

design of the CUAVA-2 cube satellite.



(a) Artist's Rendering of MarCO in deep space [158].

(b) MarCO Flight Paths [159].

Figure 2.13: MarCO Satellites and Operations

MarCO Mars Cube One (MarCO) mission was a pioneering mission that launched two 6U CubeSats, MarCO A and MarCO B, to Mars as the first and second interplanetary cube satellites [160]. A rendering of these CubeSats is shown in Figure 2.13a. Deployed alongside the InSight Lander, these CubeSats are a pair of communication relay satellites designed to monitor the Mars InSight Lander during landing. As the first CubeSats to operate in deep space, their primary objective was to test the feasibility of small spacecraft for interplanetary missions by serving as communication relays during the entry, descent, and landing on Mars on November 26, 2018. Their flight path to Mars with operations is shown in Figure 2.13b. The MarCO pair carried out their own communication and navigation experiments in route to Mars before the Mars flyby and relay operations [161]. Each CubeSat, weighing 13.5 kg, was equipped with an AstroDev MSP430F2618-based C&DH system, UHF radio receivers to capture InSight's entry, descent, and landing data, X-band transponders for high-rate data relay to Earth, and flat-panel X-band antennas. The spacecraft also featured cold-gas thrusters for attitude control, solar panels for power, and cameras for imaging, with no

on-board propulsion for orbit insertion. Both MarCO-A and MarCO-B successfully relayed real-time EDL data from InSight to Earth, providing critical telemetry during the landing phase when direct communication was not possible, thus serving as a proof-of-concept for a CubeSat in support of interplanetary missions. Contact was lost with MarCO-B on December 29th 2018 and with MarCO-A on January 4th 2019 [158]. The MarCO on-board C&DH computer was the MSP430F268 microcontroller that is a traditional processor without any special radiation tolerance. This C&DH has on-board non-volatile storage, a real-time clock, and cascaded watchdog system, and interfaces (SPI, I2C, UART, GPIO and RS-422) to all on-board subsystems [162]. The design focused on taking commercial low-power parts and implementing software resets as SEE protection to achieve the mission.

On-board Computer for Small Satellite Examples As evident in Table 2.2, cost and power were the driving factors for these on-board computers. The majority of these small satellites focused on continuous operation and low mass which naturally limited the maximum available power from solar panels. There is also a trend of ignoring TID in these satellites along with accepting data loss due to the short duration of the missions and the nature of the data being processed. This allows the on-board computer to be designed with a focus on low-power, low-cost, relatively high computational performance, and SEE recovery using resets.

Table 2.2: Six Examples of Small Satellite On-board Computers

Board Name	Computational Performance	Radiation Tolerance	Power Consumption	Cost
Flying Laptop [144, 163]	UT 699 LEON 3 FT microprocessor,	25 krad to 30 krad for RT3PE3000L	40 W for satellite	\$10,082.13 OBC
GomSpace Express-3 [148, 149]	Upto 64 MHz Clock, FPU, 512 KB Flash, 128 MB NOR Flash, 32 kB FRAM, 32 MB SDRAM	20 kRad (Si)	0.17 W (Typ.) to 0.9 W (Max)	\$6,200
HTC Nexus One [151]	1 GHz Qualcomm Snapdragon S1 processor, 512 MB of RAM, 512 MB of internal storage, 4 GB SD	Not Listed	68.6 mW min power without screen	\$95-250 OBC, <\$3,500 CubeSat
EnduroSat OBC Type I [164]	ARM Cortex M4, 180 MHz Clock, 256 kB RAM, 2MB Flash Memory, 256 Mbit Serial NOR Flash	Tested at 40 krad TID	0.9 W Ave, 1.5 W Peak, 0.2 W Idle	\$16,000 to \$22,000
ISIS iOBC [156, 165]	400 MHz, 32-bit ARM9 CPU with 32 MB SDRAM and 2x2 GB SD-card	Not Listed	0.40 W	\$5,000 OBC
AstroDev MarCO C&DH [161, 166]	12 MHz MSP430F2618 Microcontroller, 128 kB Flash, 8 kB RAM	5 krad (Si) based on family test results [167]	Estimated 14.454 mW at 12 MHz	\$18.23 OBC
End of table				

RESEARCH AIMS

Solution Motivation

Dynamic PR takes advantage of hardware flexibility inherent in an FPGA to swap hardware accelerators during real-time operation. As FPGA-based computing systems are increasingly being used as sensor data processors, swappable hardware accelerators provide a solution to expand capability for on-board processing of flight data. A hardware accelerator is designed for rapid data processing and swapping these in real-time would permit reuse of hardware and allow the main processor to maintain operation during data processing. This line of research balances radiation tolerance, computational performance, power consumption, and cost as shown in Figure 3.1.

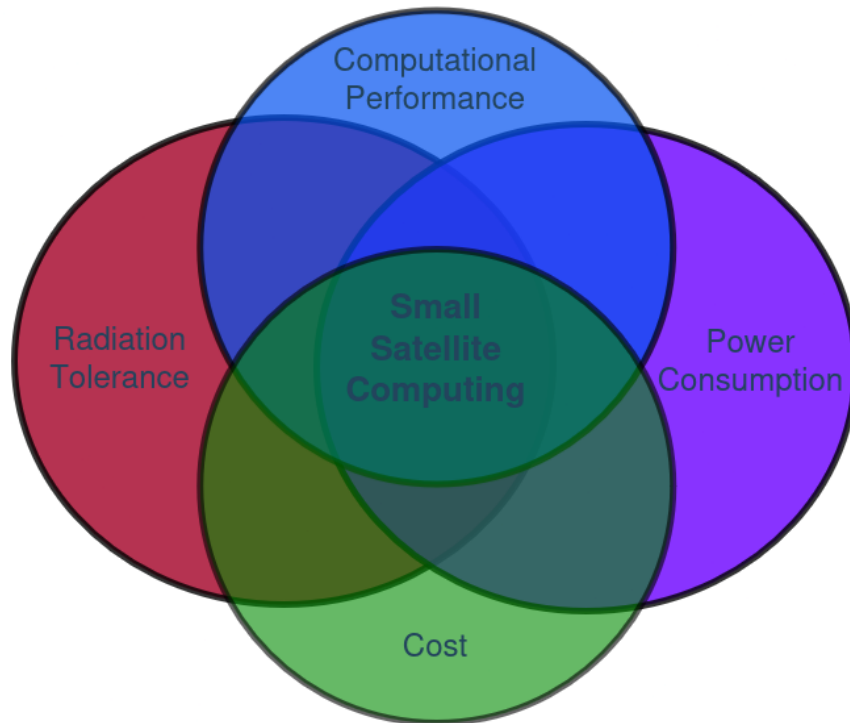


Figure 3.1: Venn Diagram for Research Trade Space

On FPGAs, swapping between different logic designs is achieved via writing to the device's configuration memory. Swapping accelerators during operation clears SEEs in the FPGA's configuration memory by overwriting that section of memory. Inside the FPGA logic elements, it clears SEEs by reinitiating the hardware within the configuration region. Through this method, radiation tolerance is increased by specifically targeting the FPGA with full and PR. Low-power spacecraft would benefit as dynamic PR greatly increases versatility without dramatically increasing power consumption or monopolizing the main processor. Mass, size, and power are all constrained resources for small satellites, creating obvious limitations for the OBCs. Small satellite computers must balance reliability, performance, power, and cost as design constraints[168]. The dynamically reconfigurable coprocessor functions as a hardware accelerator for data intensive operations that are unsupported or time-consuming for the main processor. Dynamic PR enhances an FPGA-based system with in-flight reprogrammable hardware and real-time adaptive functionality [169]. This approach has several advantages for low-power, FPGA-based computer systems. First, hot-swapping a coprocessor does not interrupt the main processor's operation as parallel implementation of a soft processor and coprocessor allows the coprocessor to be swapped out in the background. Next, the coprocessor reuses a set amount of FPGA resources limiting the power consumption for the coprocessor and allows the total usage of FPGA resources to be minimized. Additionally, coprocessors provide increased flexibility in functionality. By standardizing the processor/coprocessor interface, a variety of coprocessors are integrable into the computer architecture and accessible to the user on the software level. Finally, the PR process clears out any SEE faults in the FPGA configuration memory or related logic components. This provides a built-in degree of radiation tolerance for the coprocessor design. With the increased number of small satellites in orbit and the projected increases with the decrease in launch costs, a new OBC that boosts performance using novel processing techniques while still accounting for radiation tolerance, power efficiency, and cost

is a need for many satellite developers.

Problem Statement

Small spacecraft OBCs are generally an edge computer operating within the trade space of computational performance, radiation tolerance, power consumption, and cost feasibility. Current space-capable edge computers focus on one or two variables with this trade space and are not adaptable for the changing needs for future missions. This limits the reusability of designs to the original or similar mission sets. Future space missions are continuously seeking a solution that balance computational capability, radiation tolerance, power efficiency, and cost feasibility while also providing a level of reusability for different missions types.

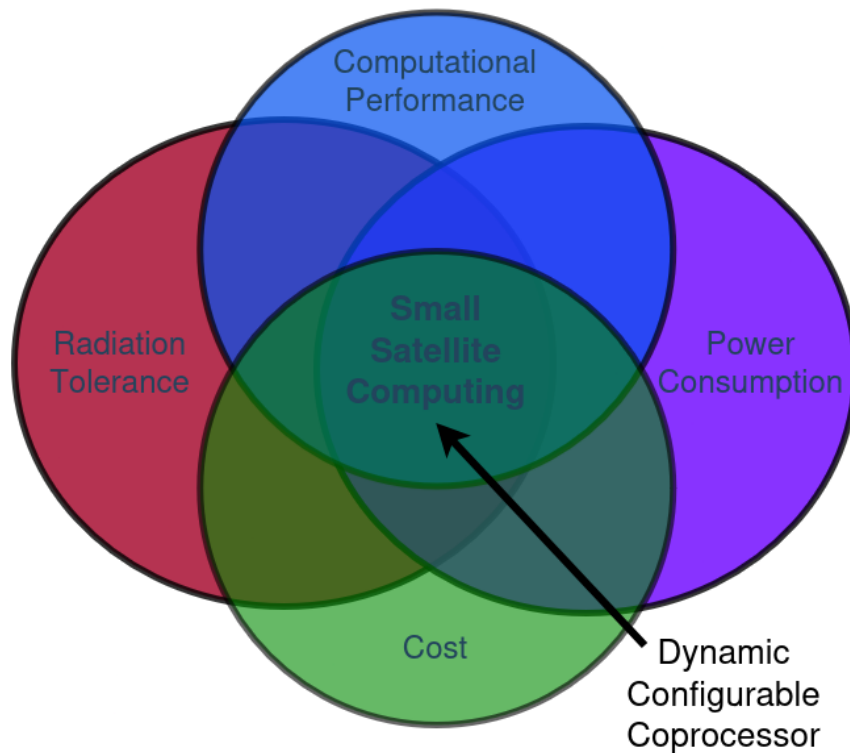


Figure 3.2: Updated Venn Diagram for Research Trade Space

The novelty of this work is to create an OBC capable of using real-time PR to achieve the needed computational performance and radiation tolerance while meeting the low-power and affordability requirements of small satellites. This solution has applicability across the entire spectrum of small satellite developers including government, military, and commercial organizations. Within this context, an FPGA-based OBC using swappable coprocessors was developed with the physical, computational, radiation tolerance, power, and cost requirements necessary to provide a solution to this problem as shown in Figure 3.2. These coprocessors are dedicated hardware accelerators that provide higher speed processing for specific tasks. The proposed solution is a balanced approach within the trade space that optimizes along all four axes by limiting the cost via reusable hardware and COTS parts, limiting the power consumption with low power parts, increasing the radiation tolerance with redundancy, scrubbing, and reconfiguration, and finally increasing computational performance with dedicated hardware accelerator functions.

Research Objective

This research evaluates the effectiveness of using reconfigurable hardware accelerators to enhance the capability of OBCs for small satellites. The contribution of this research is the development of a novel architectural solution to optimize resource utilization inside the trade space of computational performance, radiation tolerance, power consumption, and cost feasibility for small satellites and the development of the hardware to support an OBC capable of PR of the FPGA during operation.

Research Questions

With respect to the problem statement and the research objective, four research questions set the direction of development for the hardware accelerator coprocessors. These

questions frame the research performed on hardware accelerators within the defined trade space and provide focus for the research evaluation. An additional four research questions were developed following the need for new PCB hardware to support on-board PR for the FPGA. The single board computer developed was an essential artifact to enable the PR of coprocessors within the RadPC computer system and was developed in parallel with the hardware accelerators implementation. The research questions guiding this work are as follows:

Coprocessor Research Questions

1. How can reconfigurable hardware accelerators optimize computational performance for an on-board computer within the trade space of small satellites?
2. How can reconfigurable hardware accelerators optimize radiation tolerance for an on-board computer within the trade space of small satellites?
3. How can reconfigurable hardware accelerators optimize power consumption for an on-board computer within the trade space of small satellites?
4. How can reconfigurable hardware accelerators optimize cost feasibility for an on-board computer within the trade space of small satellites?

Printed Circuit Board Research Questions

1. How can the hardware be designed to support on-board PR for a single board computer and provide computational performance with the trade space of small satellites?
2. How can the hardware be designed to support on-board PR for a single board computer and provide radiation tolerance with the trade space of small satellites?
3. How can the hardware be designed to support on-board PR for a single board computer and provide power consumption with the trade space of small satellites?

4. How can the hardware be designed to support on-board PR for a single board computer and provide cost feasibility with the trade space of small satellites?

Proposed Solution

The proposed solution to address the coprocessor and printed circuit board research questions is a design leveraging PR on an FPGA to dynamically swap a coprocessor functions that are coupled with an aerospace soft processor (e.g., RadPC) to function as a complete OBC for small satellites. This approach exploits dynamic PR capability to adapt hardware modules in real-time, optimizing computational resource usage within the small satellite constraints of size, mass, power, cost, radiation exposure, and bandwidth limitations. A number of different architectures will be evaluated to find the ideal placement of the coprocessor within the soft processor's computer architecture. For coprocessor power consumption, the coprocessor architecture will be cleared when not in use to deactivate idle logic circuits. This has the double benefit of protecting the logic circuits from radiation strikes as the circuit will not be implemented and reduce power consumption. The unimplemented circuit will not be drawing current and all inputs/outputs (I/O) into coprocessor will be disabled to minimize power draw. For computational performance, hardware accelerators will boost performance for tasks like floating-point mathematical operations or image processing. These coprocessors will increase capability and performance for the RadPC soft processor by providing a system capable of rapidly performing tasks, achieving a significant speed up over software-based processing. For coprocessor radiation tolerance, the reconfigurable coprocessor will be implemented in parallel to a radiation tolerance soft processor that implements quad modular redundancy (QMR) and memory scrubbing. This coprocessor will be regularly partially reconfigured, both to swap out hardware accelerators for different functionality and to mitigate SEUs in the implemented hardware accelerator caused by radiation strikes. For coprocessor cost feasibility, the design will reuse the same hardware

for multiple tasks to reduce costs by using a smaller FPGA than if the functions were implemented at the same times. Furthermore, a single board computer will be developed to support the on-board PR and will be designed with COTS parts to minimize cost and make this solution affordable for universities. The use of COTS parts has been demonstrated in prior RadPC missions that are kept healthy in the presence of radiation using a set of fault recovery procedures.

To address the printed circuit board research questions, a design supporting on-board dynamic PR will be developed centered on a Xilinx FPGA. This FPGA was selected because the RadPC soft processor had been extensively implemented on Xilinx devices and was compatible with the FPGA at the hardware level. For radiation tolerance, the design will incorporate an option for the housekeeping microcontroller to be radiation hardened, isolated power islands, control over cycling FPGA I/O banks' power, monitoring for power rails, NAND flash for critical configuration and executable files, QMR external memory for the RadPC soft processor on the FPGA, and an interface for debugging RadPC by the housekeeping microcontroller. These SEU mitigation strategies are in combination with COTS parts inherently have ~ 50 krad of TID tolerance. These design features will permit the board to provide radiation tolerance for relatively low cost. For power consumption, the board is designed to operate at a 3 W peak during maximum usage and ~ 2.5 W during regular usage. Power isolation is implemented to allow for disabling sections of the board to achieve even further power reduction. For computational performance, the FPGA for the design will move to the XC7A200T version, which is an improvement compared to the XC7A35T currently in use at the start of this research project. A larger FPGA will provide more computing resources for the coprocessor. For cost feasibility, the single board computer's components will be COTS parts. Using entirely commercially available parts will reduce the overall cost of the board. The printed circuit board design will target a six-layer design to minimize cost per board. Target cost will be $< \$5000$ as this is the spending cutoff

limit for single order purchase by publicly funded universities. This integrated solution of hardware, a printed circuit board, and firmware, logic circuit coprocessors, will be validated through simulation and laboratory testing and promises resilient, efficient solution for on-board computing for future small satellites.

RESEARCH APPROACH

The design science research methodology is used to present this research and evaluate its results. Engineering research focuses on creating an explicitly applicable solution to a problem within a specific trade space [170]. Design science research allows the results of engineering research to be evaluated as artifacts. These artifacts are the solutions designed and developed during this research and design science is used to properly frame and evaluate the artifacts within a research context. The design science process includes six steps, problem identification and motivation, definition of objectives for solutions, design and development of artifacts, demonstration, evaluation, and communication [170]. Artifact definition includes constructs, models, methods, instantiations, or implementations constructed during the research [171]. In this case, the artifacts of the research are the printed circuit board (PCB) and the coprocessors that were developed to implement the dynamically swappable hardware accelerators functions. Evaluation is performed at two levels. Artifacts will be evaluated on the component level where the operation and capabilities of the coprocessors and the PCB will be observed and measured. Following successful operation on the individual level, the coprocessors will be integrated into RadPC, a RISC-V 32I QMD softcore processor, and implemented on the Hydra SBC-100, PCB. This integrated solution will be evaluated as an OBC for small satellites and will be compared against OBC examples from Table 2.2. The comparison will rank the results of this evaluation inside the small satellite OBC trade space of computational performance, radiation tolerance, power consumption, and cost.

The developed PCB is designed as a single board computer, centered on an FPGA with a soft processor. Design work was split into four main phases; 1) development of a coprocessor logic design for a proof of concept on a Digilent Nexys A7 FPGA Development board; 2) development of a PCB with the capabilities needed for the research; 3) integration of the coprocessor with RadPC and implementation on the new PCB; and 4) development

of two image processing coprocessors to showcase the system’s capabilities. PCB design and development were performed in parallel to the coprocessor research. This new board was required as commercially available FPGA development boards did not support on-board partial reconfiguration (PR). Therefore, a new PCB with built-in PR support was designed for this research. FPGA development is an inherently iterative process with the whole logic design being systematically developed from individual components. Each of these individual components was designed and tested in isolation before being integrated into the system as a whole. Given the highly iterative nature of the research, the design research science approach was used to evaluate the results of the research.

Design Process

Design science is a research approach focused on creating and evaluating solutions (artifacts) to solve identified problems. The identified problem for this research is balancing an OBC’s computational performance, radiation tolerance, power consumption, and cost. The development of an FPGA-based OBC for small satellites focuses on addressing specific technical and operational challenges by exploiting the reprogrammable hardware of modern FPGAs to implement reconfigurable hardware accelerators. The development process for the coprocessors and PCB is iterative and follows the design cycle shown in Figure 4.1.

The aerospace engineering performed during this research is of a highly practical nature and the design science research is essentially pragmatic in nature with an emphasis on making a relevant and clear contribution to the application environment [172]. The design science approach fits well with the actual development cycle for FPGA logic circuits with the research primarily focusing on the Design Cycle shown in Figure 4.1.

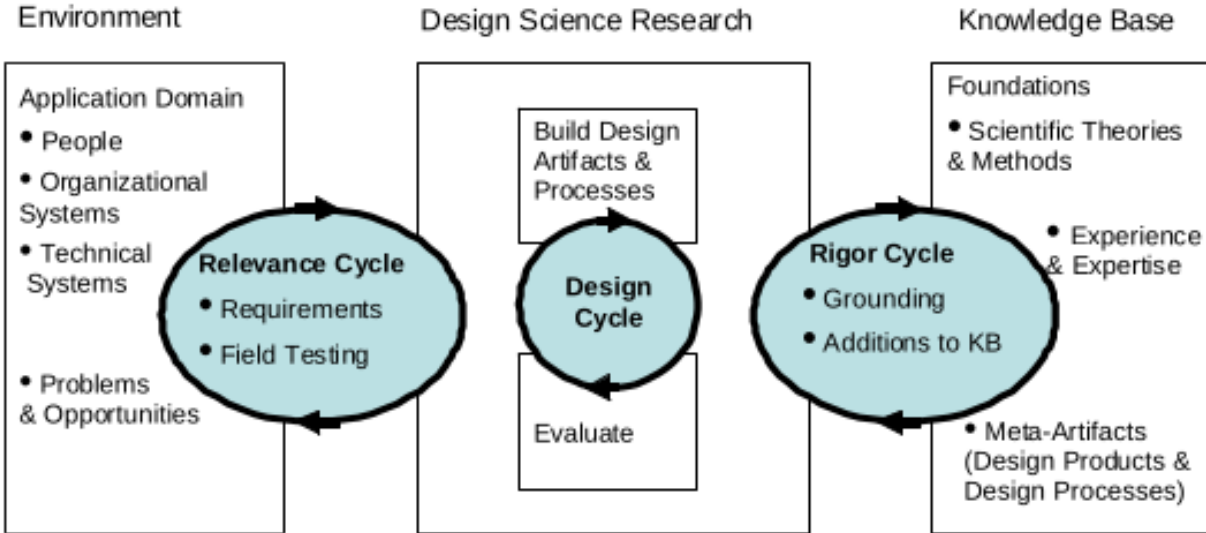


Figure 4.1: Design Science Research Cycles [172]

Methodology

The activities performed in this work adhered to the design science research methodology, completing the six activities in the framework for a systematic and rigorous approach to address engineering challenges and design as research. These six activities provide a structured process for conducting design as research [170]. The first activity, problem identification and motivation, is used to define the research problem, justify the significance of the problem, and motivate the value of a solution to said problem [170]. The second activity, defining the objectives for a solution, entails inferring clear, feasible objectives based on the identified problem and existing knowledge, and creating a foundation for the development of an artifact [170]. The third activity, design and development, focuses on creating artifacts. These artifacts are the constructs, models, methods, or instantiations that provide a solution to the specific problem defined in the previous activity by determining their functionality and architecture using relevant theoretical knowledge [170]. The fourth activity, demonstration, is the application of the artifact to solve one or more instances of the problem through methods such as experimentation, simulation, or case studies, to show its

practical applicability [170]. The fifth activity, evaluation, is the rigorous assessment of the effectiveness of the artifact by comparing its performance against the defined objectives, using metrics, empirical evidence, or logical proof. There is the option to iterate back to design for improvements during this activity. The sixth activity, communication, ensures that the problem, artifact, its utility, novelty, and rigorous evaluation were effectively disseminated to relevant audiences, including researchers and practitioners, through structured publications or presentations. This comprehensive execution of the six activities ensures the research produces a valuable, rigorously developed, and well-communicated artifact, contributing to academic research and practical application.

Problem Identification and Motivation

There is a need for a reliable, power efficient, affordable, and compact OBC to perform data processing, communication, and power management in small satellites. The solution developed for this research focuses on optimizing along four different axes; 1) radiation tolerance; 2) computational performance; 3) power consumption; and 4) cost within the small satellite OBC trade space. The constraints are meeting the requirements for a small satellite OBC capable of functioning in the LEO radiation environment for the duration of a small satellite mission (~ 2 years), operating within a cube satellite (1U-6U) power budget, providing the necessary computational performance needed for on-board processing of instrument data and flight operations, and being affordable for a small satellite financial budget ($< \$50,000$). Within this context, an FPGA-based OBC using swappable coprocessors was developed as a novel strategy to meet the increasing demand for small satellite computing with a balanced design within the trade space that optimizes along all four axes

Artifact Design

The design of research artifacts is approached within the trade space of radiation tolerance, computational performance, low power consumption, and affordability. The design

science research definition of an artifact is a solution created to address a problem [171]. For this research, the problem is a radiation tolerance, low power, affordable, small satellite computing solution with sufficient computational performance. The overarching system developed during this research is the OBC as an integrated system of four artifacts. Hydra, a single board computer, is designed to meet requirements of real-time processing, fault tolerance, and low power consumption. The suite of coprocessors implemented on this single board computer will be evaluated as artifacts. These coprocessors address computation performance via dedicated combinational logic circuitry, radiation tolerance via PR to clean out faults, and affordability via a multipurpose reusable hardware model. For the hardware artifact, the PCB was developed to support FPGA PR with multiple full and partial configurations. The combination of various radiation mitigation techniques addresses radiation tolerance. Furthermore, Hydra uses linear voltage regulators and power switches to minimize the power requirements. The creation of these artifacts used an iterative design process, including multiple rounds of prototyping of the Hydra PCB and logic simulation, FPGA implementation, and FPGA integration to refine the coprocessor architecture.

Rigorous Knowledge Contribution

This research contributes to knowledge within the aerospace small satellite field with prescriptive knowledge, the OBC itself as a solution for small satellite missions, and descriptive knowledge, the insights into the tradeoffs within the power, performance, reliability, and affordability trade space. For the prescriptive knowledge portion, RadPC with coprocessors and implemented on a Hydra PCB is a practical solution for the problem set faced by small satellites within the radiation tolerance, power, and cost constraints of the OBC system and the need for higher performance than is currently available. All the techniques used in the development of the Hydra with coprocessors have been previously developed in different contexts. PR is a standard feature of FPGAs. Task focused

hardware acceleration is a classic computer design approach commonly seen in GPUs. Low power board design has been widely implemented using microcontrollers. Affordability in computer systems has been a general trend following along with Moore’s Law and production optimization. The descriptive knowledge section focuses on the insights into the tradeoff between power, performance, reliability and costs. These insights are generalizable to other small satellite OBCs and subsystems.

Evaluation Metrics

The design science methodology provides a structured approach to developing a small satellite OBC by creating and evaluating an artifact to address specific operational and technical challenges in the cost-constrained environment of small satellites. Individual artifacts are evaluated on their successful performance. However, a problem lies in evaluating the solution developed from the artifacts, the OBC, within the complex trade space of small satellites. This is particularly difficult as partial reconfiguration is not commonly used by small satellite OBCs. While partially reconfigurable FPGA-based computers have been flown, these computers are generally used as instrument data processing nodes as opposed to the flight operation computing node. For example, an FPGA-based architecture for runtime parallel processing on space solar telescopes was developed that used PR to recover from single partial upset errors (SEUs) [173]. PR is the configuration of a subsection of the FPGA. This requires the new logic design to be either isolated from the rest of the logic design on the FPGA or be integrated in a way that accounts for swapping logic designs. PR allows for efficient use of FPGA resources. Comparatively, a full reconfiguration or static configuration will configure the entirety of the FPGA with a new logic design. Full configuration makes the development process and satellite operation simpler compared to PR [174]. A hybrid approach is generally implemented with an entire FPGA functioning as the coprocessor and swapping occurring using a full reconfiguration of the FPGA [142]. Furthermore, the

developed dynamically swappable coprocessor is not functional without the single board computer developed to support PR. Therefore, a board-to-board evaluation strategy will be used. In this work, RadPC is the computer system that will be enhanced with coprocessors that, in turn, will be implemented on a single FPGA that runs on the Hydra PCB, forming a complete OBC solution. Example small satellite OBCs will be directly compared to each other and the OBC developed in this result.

The evaluation strategy is a critical component of the design science methodology, focusing on the development of evaluation criteria and a scoring metric within a trade space to assess the artifact's effectiveness. The developed small satellite OBC with coprocessors is evaluated using a weighted scoring in four categories, with scores derived from the literature review of OBCs. These key criteria include radiation tolerance (required to ensure reliability in space), computation performance (processing speed to handle real-time data, memory, operations per second), power efficiency (low operating power to stay within satellite power budget), and cost (a dominant factor due to lower budgets of small satellites). The trade space analysis allows for comparing multiple OBC configurations by assigning scores based on these criteria, with the equation balancing trade-offs to identify the optimal design. Evaluation will be performed on a bracketing system. This allows grouping of examples with different approaches. For RadPC with the developed coprocessors, evaluation will include simulations to test functionality and computational performance, ground-based testing to validate hardware reliability, and power measurements. Coprocessors will be evaluated individually, and the combined RadPC & Coprocessor system will be scored based on the literature review for each of the sections. Cost is a strong factor as the main motivation behind small satellites and has a major influence on OBC selection for small satellite developers.

Ranking Scores will be ranked poor, fair, good, and excellent in ascending order. Poor represents a score that is only acceptable for the OBC if the category, computational performance, radiation tolerance, power consumption, or cost, is not a priority for the small satellite design. Fair is a score that would be generally acceptable for the small satellite design. Good is a score where the OBC would well within that category. And excellent is a score for OBC where the performance in that category is exceptional.

Computational Performance Scoring Computational performance will be evaluated based on the specifications of the processor, clock speeds, integer math capability, floating-point math capability, vector processing capability, and memory size with respect to power and physical size of the OBC system. Unfortunately, a direct comparison between processor benchmark results is not possible as OBCs are tested for application specific benchmarks. In addition to the specifications previously listed, the file system and compiler efficiencies, cache memory size and organization, main memory size and organization, I/O and file transfer speeds, user loads, and application effect computational performance [175]. Processors also vary greatly in versatility, capacity, timing sensitivity, scalability, and interfacing [176]. The best benchmark for computational performance is to directly perform the target application and measure the system's performance [175]. This matches with the general approach of testing OBCs in their target application. Since OBCs are generally designed for a particular mission profile, the testing these systems undergo is individualized to the system itself and the mission being performed. However, this approach creates the problem that test results are not directly comparable between systems. This problem is further compounded by the different computer architectures being evaluated differently. Traditional processors, radiation hardened processors, FPGAs, and hybrid processors all have different methods of evaluations and use different metrics for benchmarking the system. Traditional processors generally use a benchmark method for integer operations per second (IOPS), floating-point operations per

second (FLOPS), or a Dhrystone benchmark (DMIPS) to characterize a system. Radiation hardened processors use similar metrics while enduring either radiation or injected faults. FPGAs are evaluated based on latency, throughput, and usage rates of FPGA resources. This evaluation is further complicated by the reconfiguration capability of FPGAs that allows the latency, throughput, and resource usage to change during operation. Thus, FPGA benchmarking focuses on functional benchmarks that measure end-to-end execution time for specific applications as shown in Figure 4.2. Finally, the hybrid processor combines all the complexities of a traditional processor and an FPGA or GPU and difficulties of benchmarking both systems.

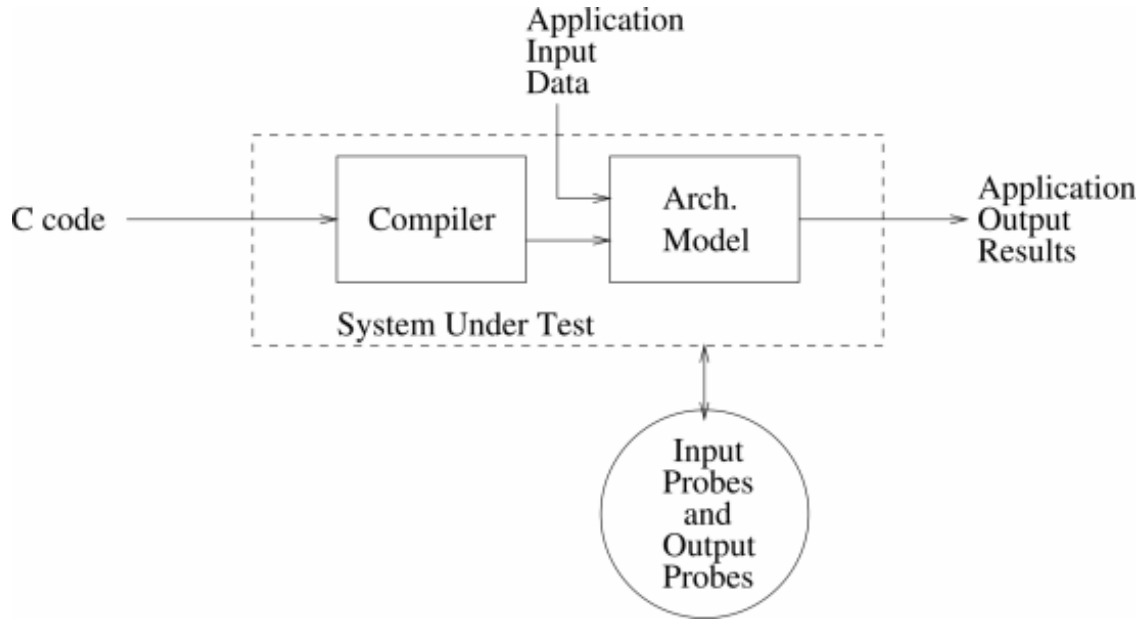


Figure 4.2: The traditional format for testing processor architectures which use a C code compiler. [177]

Given these issues in the comparison between different architectures, the computational performance will be evaluated on specifications available from the OBC's datasheet and the system's capability to performance its specific application of functioning as the flight computer and/or instrument data processor for a small satellite. The score for computational

performance will be calculated using Equation 4.1. This method of evaluating the computational performance is rough approximation for the system's actual computational performance since it focuses primarily on the capabilities commonly focused on for small satellite design.

$$C_{comp} = K_{clk} + K_{int} + K_{flt} + K_{vec} + K_{mem} \quad (4.1)$$

Clock frequency provides an overall metric for the processing speed of the system. Integer processing is used extensively for flight control software that requires rapid responses. Floating-point processing is used in small satellites for navigation and location calculations from star trackers. This type of processing with floating-point processing is generally flight critical processing. Therefore, the floating-point scoring will reflect the inherent capability to perform float-point processing on an integer-based CPU at the software level including the CPU's capabilities for systems that lack a dedicated floating-point processing unit. Vector data processing is used extensively in image processing applications and with the boom of Earth observation satellites. Since this processing is generally not flight critical, the

Finally, the memory size of the OBC will decide the small satellite's capability to store payload data. These measurements will not provide results relatable to absolute computational performance. However, they will allow for direct comparison between OBCs using different computer architectures. Table 4.1 gives the specifications that will be used to generate C_{comp} for the calculation of the score for computational performance.

To convert from a clock frequency specification into a dimensionless score, OBC's clock frequency will be ranked on a logarithmic scale with 1MHz corresponding to a score of 0 and 1 GHz corresponding to a score of 100. These rankings will be rescaled by 0.2 and incorporated in Equation 4.1 for the variable K_{clk} using the Equation 4.2.

Table 4.1: Computational Performance Scoring Metrics

Variables	Specifications	Metrics	Scoring
K_{clk}	Clock Speed	Logarithmic scale (1 MHz - 1 GHz)	0-0.20
K_{int}	Integer Operations	Capable of integer operations	0-0.20
K_{flt}	Floating Operations	Capable of floating-point operations	0-0.20
K_{vec}	Vector Processing Operations	Capable of vector processing	0-0.20
K_{mem}	Memory Size	Size in brackets by factor of 10	0-0.20

$$K_{clk} = 0.2 * \frac{\log_{10}(Frequency) - \log_{10}(MinFrequency)}{\log_{10}(MaxFrequency) - \log_{10}(MinFrequency)} \quad (4.2)$$

An example using a CPU clock speed of 32 MHZ scored on the logarithmic scale from 1 MHz to 1 GHz is shown in Equation 4.3.

$$K_{clk} = 0.2 * \frac{\log_{10}(32) - \log_{10}(1)}{\log_{10}(1000) - \log_{10}(1)} = 0.100343332 \approx 0.100 \quad (4.3)$$

For integer operations, the OBCs will be evaluated on the range of operations supported by the processor or coprocessor. Mathematical integer operations including addition, subtraction, and multiplication need to be supported for a full score of 0.2. If the CPU does not support integrated integer mathematical operations in the processor or with a coprocessor, then 0.02 will be deducted from the full score per operation which is not supported.

For floating-point operations, the OBCs will be evaluated on the range of float-point operations supported by the processor or coprocessor. An FPU integrated into the processor will receive a higher score because of the inherently faster processing speed from minimizing data transfer. The deduction for lack of support for a floating-point mathematical operation

(addition, subtraction, multiplication, division) will result in a deduction of 0.2 from the full score of 0.2. Coprocessors will have 0.025 deducted from the score due to the latency in data transfer between processor and coprocessor.

Vector processing operations are scored based on the OBCs capability to perform parallel processing for image processing using a dedicated coprocessor. For systems without a dedicated coprocessor, the main processor's capability of performing vector processing (image processing) will be evaluated and scored for comparison. The coprocessor approach is a means to provide hardware-based, parallel processing of vector operations within an integer-based CPU. The proof-of-concept that was selected to show this capability is set of common image processing operations performed on a scaled-down image. A dedicated coprocessor for vector processing will be given a full score of 0.2. If the OBC does not have a dedicated coprocessor, its vector processing capability will be evaluated based on the number of floating-point processing units, fixed-point-processing units, integer processing capability, and clock speed of the OBC with a deduction of 0.02 for not having a dedicated capability. If the OBC is lacking any of the capabilities to perform vector processing, then a default score of 0.075 will be given for software vector processing support to reflect a CPU's capability to perform the processing in software.

For the memory system, it will be assumed that power will be maintained continuously for the OBC. This will allow volatile memory and non-volatile memory to be evaluated for overall memory capability. Scoring will be done on the total bytes ranked on a logarithmic scale from 10 kB to 64 GB. The memory size of 1 kB is the lowest reasonable value where the OBC will still be functional as a small satellite OBC and 64 GB is the largest memory size ranked within the examples. Scoring was performed using the following equation given in Equation 4.4.

$$K_{mem} = 0.2 * \frac{\log_{10}(TotalBytes) - \log_{10}(MinBytes)}{\log_{10}(MaxBytes) - \log_{10}(MinBytes)} \quad (4.4)$$

An example using a memory size of 10 MB is scored on this logarithmic scale of 10 kB to 64 GB is shown in Equation 4.5.

$$K_{mem} = 0.2 * \frac{\log_{10}(10MB) - \log_{10}(10kB)}{\log_{10}(64GB) - \log_{10}(10kB)} = 0.088155177 \approx 0.088 \quad (4.5)$$

These methods of measurements will be used for the OBC examples in this paper. Small satellite OBCs are often custom-built for the application with a variety of architectures and processing nodes. The limitation on this measurement is the lack of systemic testing to a standardized metric for small satellite OBCs. However, it will provide a rough measure of an OBC's capabilities for real-time performance based on the actual data available for these systems.

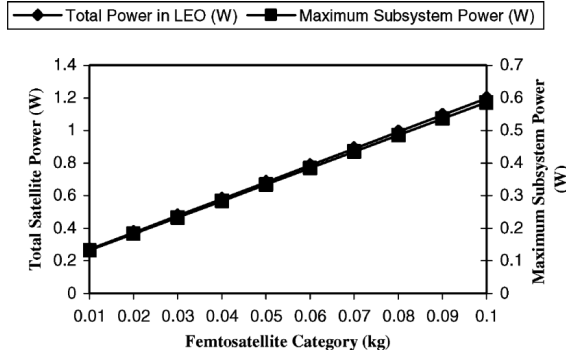
Radiation Tolerance Scoring The radiation tolerance evaluation is based on the TID ratings and SEE ratings of the OBC system using known ratings if available and estimated ratings from similar processors in the same device family if unavailable. The smaller feature sizes make modern electronics more inherently resistant to TID since lower energy particles are unlikely to be trapped in the ultra-thin isolation regions or gate oxides of the device [39]. For reference, most commercial components are radiation-tolerant up to 5 krad with electronics created with transistor nodes < 22 nm achieving as much as 50 krad without intentional hardening [178].

For the scoring, the processor's krad rating will be used as its measure of resistance to TID since this measurement rates the physical degradation rate. A basic bracketing approach will be taken with scoring moving up by factors of 10 or orders of magnitude. Scoring SEE resistance is more involved since multiple factors influence system SEE vulnerability. With small satellite missions generally being under 2 years in duration, SEE will be weighted equal to TID as the chances of failure due to TID are lower than a higher orbit satellite mission.

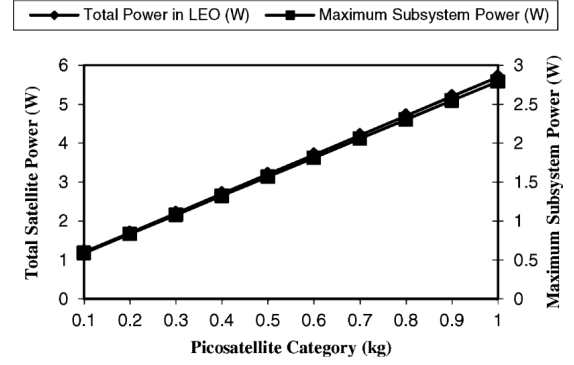
The main consideration will be the linear energy transfer threshold for the device, which describes the ability of a radiation strike to cause a state change on the device, and the associated recovery techniques used when a state change does occur. For the recovery technique priority, hardware-based recovery, such as TMR and ECC, shall be rated highly, followed in priority by scrubbing and CRC, and lowest priority, watchdog timers and resets will be considered. The scoring reflects the time that it takes to repair a fault going from near instantaneous as in the case for TMR and ECC, to periodic on the order of minutes to hours in the case of scrubbing and CRC, to the slowest periodic rate on the order of hours or days for watchdog timers and resets. For RadPC & coprocessors, the radiation environment shall be modeled using the CREME96 software with the assumption that the small satellite is operating in Low Earth Orbit. This scoring approach focuses on providing a reference for OBC comparison rather than perfect accuracy for a small satellite using RadPC as the OBC.

Power Consumption Scoring The power evaluation will utilize the base version of the OBC. The score will be based on the simplest version since small satellite developers generally design with a focus on cost, not capability. Power will be evaluated via the total OBC power requirements or the individual processor/processors integrated circuit chips as a measure of the overall board power performance. For the OBC, power efficiency will be measured as percentage of the total satellite power within the bracket of the satellite mass.

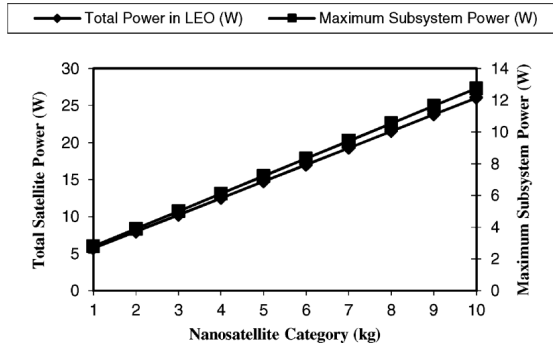
This target was selected as a CubeSat power budget including instruments and computer ranges from 2 W to 8 W depending on design and orbit [180]. The target of 2 W is the maximum possible power consumption for a 1U CubeSat. Furthermore, total small satellite power can be modeled as a function of its mass as shown in Figure 4.3 [179]. Therefore, the OBC power efficiency can be evaluated based on the percentage of satellite power budget that is required for the computer's operation. The power to mass



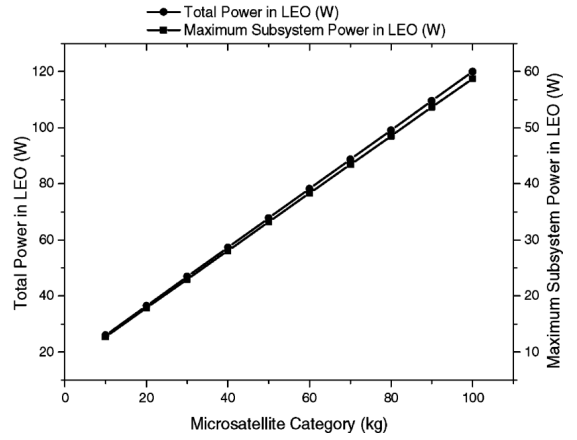
(a) Power to Mass Ratio for Femtosatellites [179]



(b) Power to Mass Ratio for Picosatellites [179]



(c) Power to Mass Ratio for Nanosatellites [179]



(d) Power to Mass Ratio for Microsatellites [179]

Figure 4.3: Power to Mass Ratios for Four Small Satellite Categories

ratio equations from Figure 4.3 are calculated for small satellite mass classifications in Table 4.2.

For small satellite examples that do not have publicly available data for total satellite power, the equations referenced in Table 4.2 will be used to calculate the estimated power. This estimated power will be used as the total satellite power to calculate the power score using Equation 4.6.

$$C_{pwr} = 1 - \frac{OBC_{pwr}}{Sat_{totalpwr}} \quad (4.6)$$

Table 4.2: Power to Mass Ratios for Satellite Mass Classes [181]

Satellite Class	Power(W) to Mass(kg)	Mass(kg)	Estimated Power(W)
Femto	$P_t(W) = 10.3M + 0.167$	0.01 – 0.1	0.27 – 1.197
Pico	$P_t(W) = 5M + 0.7$	0.09 – 1	1.15 – 5.7
Nano	$P_t(W) = 2.26M + 3.44$	1.1 – 10	5.926 – 26.04
Micro	$P_t(W) = 1.044 * M + 15.56$	11 – 200	27.044 – 224.36

Where OBC_{pwr} is the OBC power requirements and Sat_{pwr} is the total power budget of the satellite as reported or as estimated from satellite mass. This equation of the scalar was created to reward power efficiency by the OBC with a higher score for a lower ratio in OBC power requirements to total satellite power requirements.

Cost Scoring Cost will be evaluated via the listed cost for the OBC as a system or by the total cost of the processor/processors used by the OBC system. Since cost is already in dollar amounts, the data will be directly compared as the processor cost is generally the most expensive part of the OBC. The most significant cost driver for an OBC is the level of radiation hardness. As small satellite OBCs focus on affordability with COTS parts, the main processors' radiation tolerance rating is a good indicator of the expense. Traditional cost estimates for a satellite are based on total satellite mass and are invalid for small satellites as the mass to cost ratio of large satellites does not apply to small satellites. Therefore, the affordability of the small satellite OBC will be evaluated via the total cost in 2025 dollars for the processor chips as shown in Equation 4.7.

$$C_{cost} = 1 - 0.9 * \frac{\log_{10}(Cost) - \log_{10}(MinCost)}{\log_{10}(MaxCost) - \log_{10}(MinCost)} \quad (4.7)$$

Communication

The communication stage is the final step in design science research methodology. Communication is critical for ensuring that research findings contribute to academic and practical knowledge [170]. For academic audiences, the strategy emphasizes rigorous dissemination through scholarly channels, namely journal publications and conference publications for both papers and presentations. For practical knowledge, the research findings will be placed under a patent to protect the underlying technology and allow for commercial applications. Other communication avenues that will be used include, journal publications, conference publications and presentations.

Artifact Development

Artifact development follows the iterative nature of FPGA development and aligns closely with design science research methodology as shown in Figure 4.4. All three coprocessor artifacts underwent repeated cycles through the Design & Development, Demonstration, and Evaluation stages of the process. Development focused on reaching certain milestones and building off these milestones to reach the next milestone. For this research, coprocessor artifacts were developed via equation development for operations, VHDL coding, simulation of discrete steps, implementation of discrete steps, evaluation, and iteration to improve performance.

PCB hardware was developed in parallel to the VHDL development of the coprocessors. Following the development of the first coprocessor artifact, FPU, evaluation found the PCB hardware lacked the capability to allow for on-board PR. While the initial PCB hardware platforms, the Nexys A7 FPGA Development Board and Nexys Video FPGA Development, were capable of PR, an external connection to a computer was required to all the Vivado software to order a PR through the Xilinx Platform II Cable. Iteration for the PCB hardware

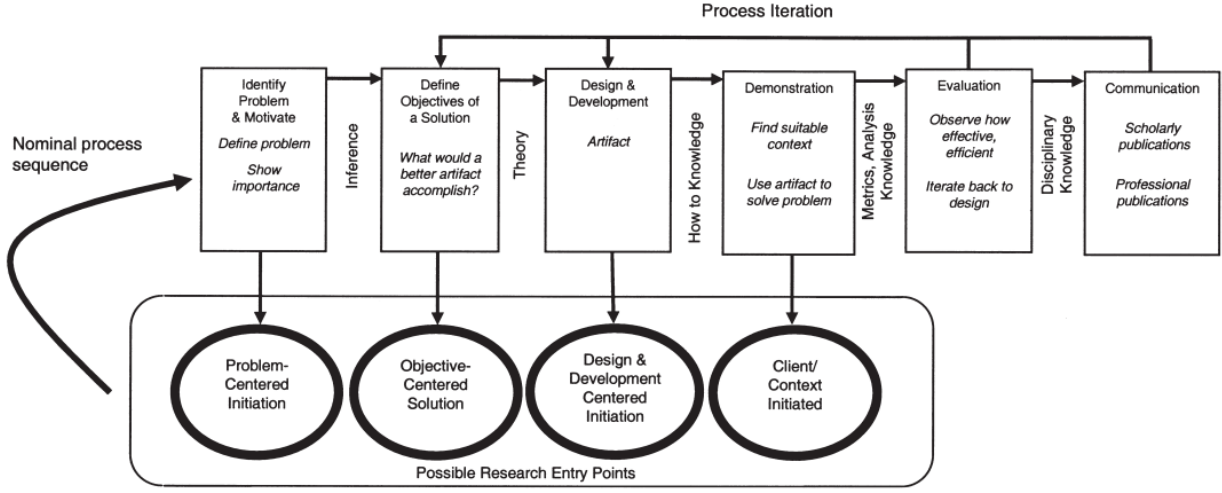


Figure 4.4: Design Science Research Cycles [172]

shifted back into the Design & Development phase, and a new artifact was created to meet the objective of real-time, on-board PR. This PCB artifact was developed in parallel to support the on-board dynamic parallel reconfiguration necessary. A set of breakout boards were developed, and five prototype iterations of the main PCB were fabricated and assembled. The revising through this process provides one example of the research’s iterative nature.

Comparison

To evaluate the research and its resulting artifacts and to accurately place them in the trade space, RadPC implemented on the new PCB with the integrated dynamically reconfigurable coprocessor will be compared to six small satellite examples using the evaluation methodology previously outlined. This comparison will assess all the artifacts complete computer system, focusing on how RadPC performs relative to other solutions in the field. This comparison is not the analysis of the artifacts, that analysis will precede the comparison. The comparison provides a structured approach to evaluate RadPC with coprocessors against selected examples of small satellites that defined the trade space of

computational performance, radiation tolerance, power consumption, and cost. The base version of each board will be used to prevent skewing the results and to reflect the reality of deployed small satellites as opposed to future small satellites that may present yet-to-be-achieved performance metrics. Since RadPC uses dynamic partial reconfiguration of the coprocessor the functionality provided by all different variations of the developed coprocessors will be included since the PR was specifically developed to provide increased hardware versatility. Furthermore, full configurations used by the six example small satellite OBCs will be included when evaluating functionality. This allows any full configurations with additional functionality to be included in the comparison.

Additionally, the RAD750 from BAE and the new HPSC from MicroChip will be evaluated and scored as a comparative benchmark to the large satellite processors that do not have the same constraints as small satellite OBC systems. Specifications for the RAD750 and HPSC are given in Table 4.3. The RAD750 is the historical standard for a radiation hardened single board computer costing $> 200,000$ per processor. These processors have extensive flight heritage in missions from the Mars Rovers to deep space probes. The HPSC, specifically Microchip's PIC64-HPSC microprocessor family developed under a NASA contract and costs $\$80,000$ per processor, represents a next-generation leap with eight SiFive RISC-V X280 64-bit CPU cores supporting virtualization, real-time operations, and vector extensions for AI/ML tasks, offering radiation-hardened and tolerant performance in a scalable, fault-tolerant design.

Table 4.3: Large Satellite Processors

Company	Processor	Type	Result	Clock & Memory	Radiation Hardness	Power (W)	Ref
BAE	Rad-tolerant RAD750 RISC	Typical	260 DMIPS, 2.1 MIPS at 132 MHz	110 to 132 MHz, 128 MB DRAM, 256 kB Start-Up ROM	200 krad (Si), SEU: >1.6 E-10 errors/bit- day	<10.8 W	[182, 183]
BAE	Rad-hard RAD750 RISC	Typical	400 DMIPS 2.1 MIPS at 200 MHz	200 MHz, 128 MB DRAM, 256 kB Start-Up ROM	1 Mrad (Si), SEU: >1.6 E- 10 errors/bit- day	<10.8 W	[182, 183]
MicroChip	Rad-hard PIC64- HPSC1000- RH	Typical	26K DMIPS, 2 TOPS (int8) or 1T FLOPS (float16)	1 GHz,	100 krad (test to 200 krad), SEL 78 MeV	~7-10 W Es- timated	[184, 185]
MicroChip	Rad-tolerant PIC64- HPSC1100- RT	Typical	26K DMIPS, 2 TOPS (int8) or 1T FLOPS (float16)	1 GHz,	50 krad, SEL 42 MeV	~7-10 W Es- timated	[184, 185]
End of table							

The six small satellites OBCs, Flying Laptop, GomSpace Express-3, HTC Nexus One, EnduroSat OBC Type I, ISIS iOBC, AstoDev MarCO C—7DH, will be directly compared to RadPC SBC-100 in terms of computational performance, radiation tolerance, power consumption, and affordability. This comparison will accurately place RadPC SBC-100 relative to its peers in the trade space. RAD750 and HPSC will compare to RadPC SBC-100 directly as well, which will place the RadPC SBC-100 with the wider OBC for both traditional large satellites and the newer small satellites.

RESEARCH ACTIVITIES

The activities performed for this research are the development and individual evaluation of the artifacts. Future sections will consider comparison of the artifacts integrated into a single board computer to the high-end on-board processors and examples of small satellite on-board computers.

Artifact Definition

Artifacts are a product of the research being performed. An artifact does not typically include evaluation results or an in-depth examination of the artifacts themselves. Details of artifacts and evaluation will be performed in future sections that will cover the development of the artifacts, capabilities of the artifacts, evaluation of the artifacts, and limitations of the artifacts.

In design science research, artifacts are the products created to address a specific problem or need. These artifacts are the tangible results that are evaluated as a contribution to knowledge in the field. In this research, four artifacts were created as a solution to the issues of computational performance, radiation tolerance, power consumption, and cost for single board computers in small satellites. As artifacts are the concrete outcome of the design science research process, the artifacts of this research are the PCB, the single-precision, floating-point, mathematical coprocessor, the first-difference edge detection coprocessor, and the first-derivative edge detection coprocessor. The first artifact is a single board computer design around an FPGA as its central processing unit. Developed to support dynamic partial reconfiguration, this PCB focused on providing a low-power small satellite computer that boosts computational performance using coprocessors. The second artifact is a coprocessor for floating-point arithmetic operations. Paired with a RISC-V-32I processor, this coprocessor provides expanded computational capability with sixteen floating-point

operations in addition to the processor’s built in integer operations. The third artifact is a coprocessor for first-difference edge detection in one dimension. This coprocessor increases throughput of image processing by rapidly processing 16x16 pixel RGB images in parallel to the RISC-V processor. Finally, the fourth artifact is a coprocessor for first-derivative edge detection. This coprocessor provides flexible image processing for 16x16 pixel RGB images. It allowed for both one-dimensional and two-dimensional processing. Each of these artifacts is examined and evaluated in depth in the following sections.

Coprocessor: Floating-Point Unit

The floating-point unit was the first artifact developed and was used to define the design constraints and development process for the following artifacts. This artifact was used as a proof-of-concept for coprocessor functionality, implementation of the coprocessor design on an FPGA, dynamic swapping of the coprocessor functionality using partial reconfiguration of the FPGA, integration with RadPC, and implementation of RadPC with coprocessor on an FPGA. The coprocessor approach is a means to provide hardware-based floating-point capability within an integer-based CPU. The proof-of-concept selected to show this capability is a set of 32-bit, floating point functions from the IEEE VHDL proposed library, using the single precision data type from the IEEE 754 Standard. The data type was chosen because RadPC is a 32-bit RISC-V architecture and a single precision float (32-bit) is the largest float size that fits into RadPC’s registers.

Initial Coprocessor Design

The preliminary work demonstrated that a coprocessor based on an FPU was possible within the RadPC integer-based computer architecture using the IEEE Proposed library for VHDL to provide the floating-point operations. Swapping out hardware accelerators was tested in isolation and proved that swapping the floating-point operations via PR

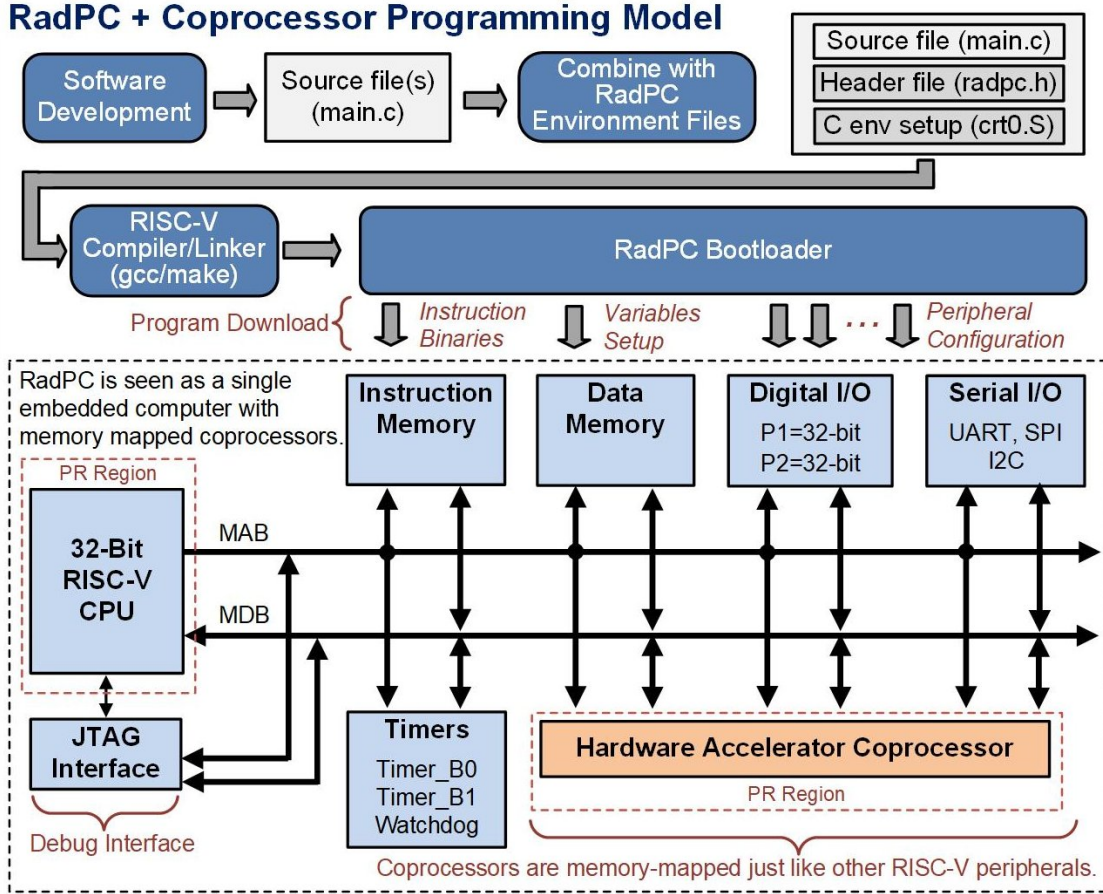


Figure 5.1: RadPC+Coprocessor integrated model

was possible. Based on this work, a coprocessor for floating-point operations with nine (9x) floating-point swappable hardware accelerators was developed in a three-step process. First, the data storage for the coprocessor I/O was tested using different setups, direct memory access, First-In-First-Out (FIFO) storage, peripheral memory addresses, registers, etc. Second, a floating-point coprocessor was developed. This consisted of the architecture for the coprocessor and nine different hardware accelerators to swap in and out of the coprocessor. The I/O for the coprocessor was standardized to allow for the development of future coprocessors. Various PR region locations for the floating-point coprocessor within RadPC's architecture were tested. Third, the coprocessor FPU functions were individually

simulated to verify functionality and then implemented on an FPGA to verify functionality in hardware. Finally, the floating-point coprocessor was integrated within RadPC and implemented on an FPGA development board for verification as shown in Figure 5.1.

Coprocessor Data Storage An alternatives study was performed on the options for storing incoming data to be operated on by the coprocessors within the confines of the RadPC QMR architecture. During operation, the coprocessor needed to receive input data (32-bit single-precision floating-point) from and return results to the CPU (32-bit single-precision floating-point). Input data was constrained to the IEEE 754 Standard for 32-bit floating-point representation and was entered by a user on the software level. For RadPC, the data for both coprocessor inputs and outputs is treated as unsigned to allow the compiler to process the code for RadPC. Results required storage in memory until read by the CPU and then were passed to the software level for delivery to the user. A variety of solutions for coprocessor to CPU data transfers are possible, ranging from external coprocessors that function as peripherals to tightly coupled coprocessors with specific opcodes to peripheral coprocessors with a dedicated interface [186–188].

Figure 5.2 shows the options investigated. Three options were tested: 1) separate data memory for the CPU and coprocessor; 2) shared data memory for the CPU and coprocessor; and 3) fast storage for the coprocessor with dedicated data memory locations. Separate data memory for the CPU and coprocessor was functional, but inefficient. The sequential data movement from CPU data memory to coprocessor data memory resulted in high latency. Total operation time was dominated by data transfer between the two data memory spaces. Adding a second data memory of usable size for the coprocessor requires more FPGA RAM than was available for the targeted FPGAs (Xilinx 100T and 200T). Due to the inefficiencies in a second separate memory for the coprocessor, this approach was tested by not selected for the design.

Determining Optimal Data Storage Location for RadPC+Coprocessor System

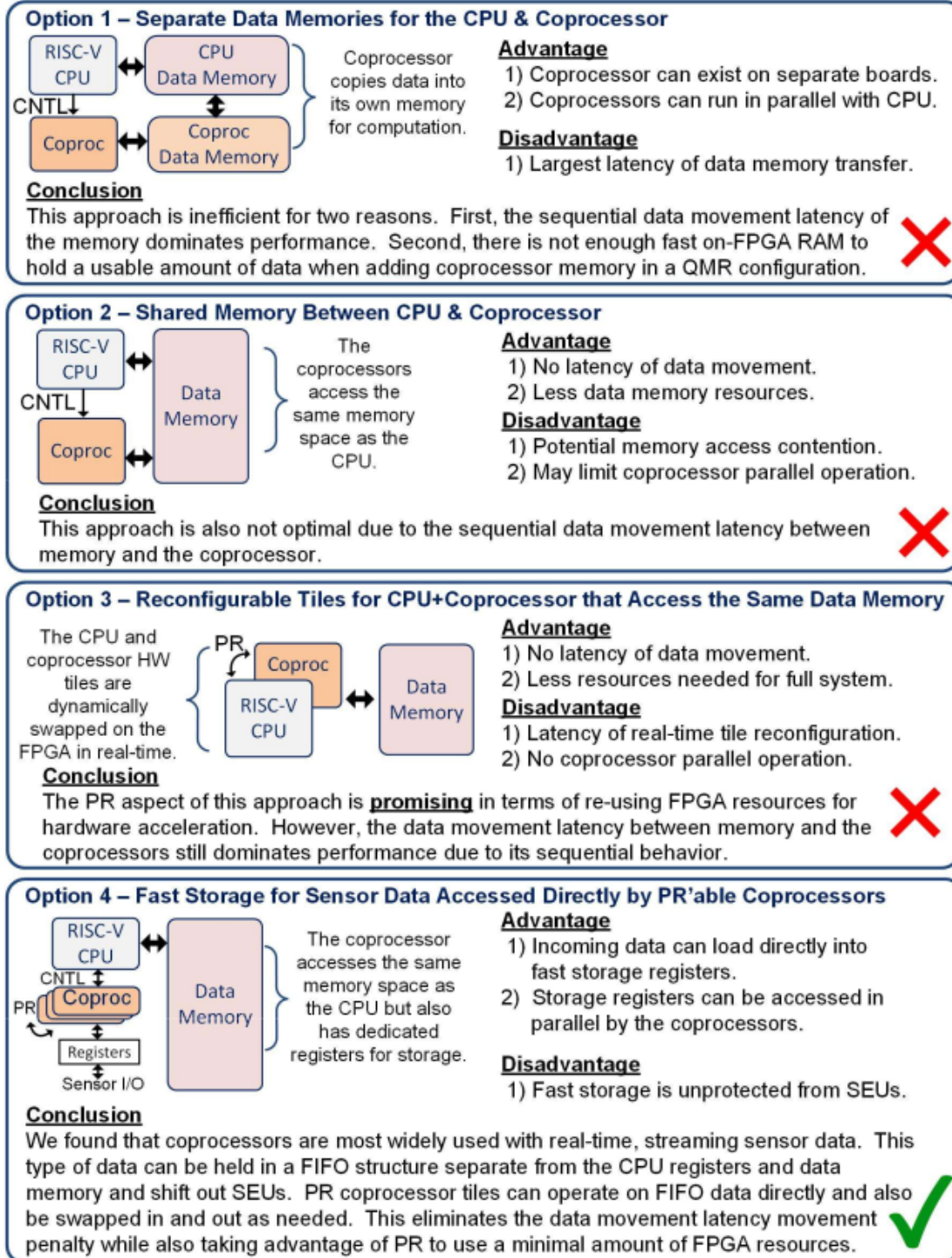


Figure 5.2: Optimal data storage location for RadPC+Coprocessor system

The shared data memory was functional in terms of available on-board FPGA RAM. However, there were issues with the CPU and coprocessor being in contention when accessing the memory. For example, the CPU would read the memory address for the results before the coprocessor had completed writing. Swapping the coprocessor via PR contributed to the contention as PR was unpredictable for the CPU and caused memory contentions.

Fast storage for the coprocessor with dedicated data memory locations was the solution to the issues of data transfer latency and memory access contention. Registers for the coprocessor were implemented inside the CPU architecture. From the coprocessor's perspective, these registers were continuously receiving data and producing results. From the CPU's perspective, registers were accessible in the same manner as standard peripheral registers. Since the coprocessor registers were loaded into data memory on the same clock as the CPU, contention access was avoided and the data transfer from register to data memory was minimized. This approach treated the coprocessor as an external asynchronous peripheral for the RadPC computer system. Once this approach passed timing, access, and PR testing, it was implemented into the RadPC architecture.

Coprocessor Simulation Each hardware accelerator for the coprocessor was simulated using the Vivado tool individually before being implemented on the FPGA. Each simulation includes multiple full operations of the floating-point mathematics with a reset interruption to ensure reset functionality. The simulation results for the floating-point operation with screenshots of the results are shown in Figure 5.3. The inputs and outputs displayed in hexadecimal.

In the above Figure 5.3, a floating-point division is simulated with an active low reset occurring at 165 ns and lasting until 190 ns. The objective of this simulation is to verify that the floating-point division is successful and that the combinational logic design will correctly set its outputs to zero when reset. Equation 5.1 shows the operation performed, with a and

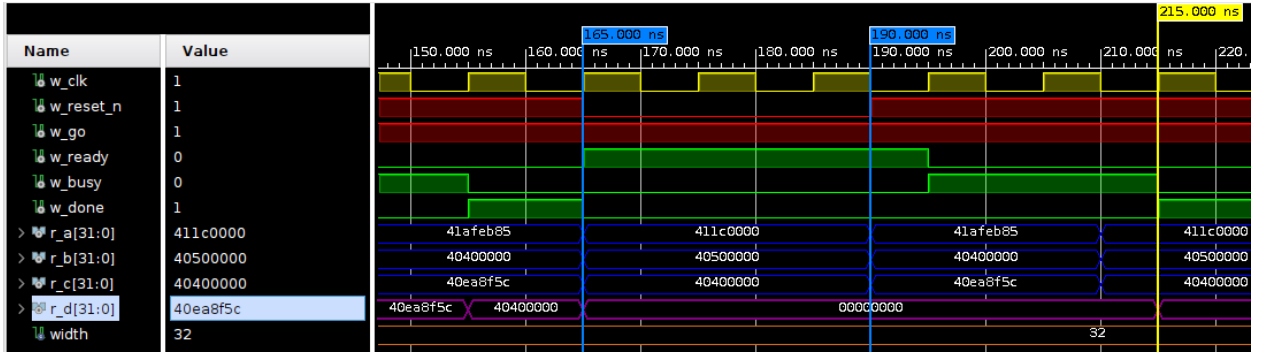


Figure 5.3: Simulation of floating-point division operation with reset interruption at 165 ns

b as inputs, d as the output, and c as a passthrough holding the correct answer.

$$d = \frac{a}{b} \quad (5.1)$$

A 100 MHz clock (yellow waveform in Figure 5.3) was provided in order to closely match the clock for the Nexys A7 that was the target FPGA development board at this point. An active low reset was provided (upper red waveform) and was triggered in the middle of the waveform. The *go* signal (lower red waveform) was used to start operation within the coprocessor. Signals *ready*, *busy*, *done* (green waveforms, upper, middle, lower respectively), were used by the coprocessor to signal when it was ready for more data, when it was processing data, and when it was done processing data. Registers r_a , r_b , and r_c (blue waveforms, upper, middle, and lower respectively) are the floating-point inputs. Hex was used for the inputs as RadPC is an integer-based processor and lacks support for the single floating-point datatype. The hex representations for single floating-point numbers were hard coded into the testbench. Register r_d (purple waveform) is the output of the accelerator. The test operations, timing, and results in hex are shown in Table 5.1.

These simulations were repeated for all nine floating-point functions, addition, subtraction, multiplication, multiply and accumulate, division, division by 2, modulus, remainder,

Table 5.1: Simulated Single Floating-Point Division Results

Name	Function	Test Operation	Hex Result	Duration
DIV	Division	$21.99/3 = 7.33$	d = h40ea8f5c	Start - 155 ns
DIV	Division	$9.75/3.25 = 3.00$	d = h40500000	155 ns - 165 ns
RESET	Resetting	a = X, b = X, c = X, d = 0	d = h00000000	165 ns - 190 ns
DIV	Division	$21.99/3 = 7.33$	d = h40ea8f5c	190 ns - 215 ns
DIV	Division	$9.75/3.25 = 3.00$	d = h40500000	215 ns - End

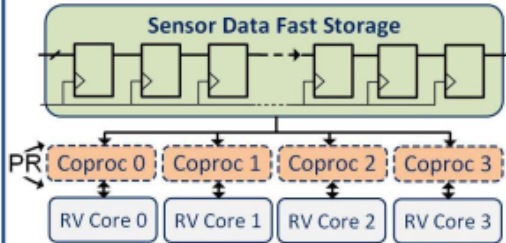
and reciprocal. An additional function from the IEEE Proposed Library for VHDL, square root, was tested. However, the Vivado tool could not successfully synthesis, simulate, or implement that function as the IEEE Proposed Library is experimental and this function (sqrt) was not fully supported.

Coprocessor Integration Following simulation of the individual functions, testing on the Floating-Point Coprocessor was performed using the Nexys A7 FPGA Development Board featuring the Artix-7 100T FPGA. Initially, the floating-point functions were implemented singly with a VHDL wrapper to standardize the inputs and outputs. Once functionality on the hardware was confirmed, PR was tested to swap between hardware accelerators. In this test, the implemented system consisted only of each hardware accelerator with a VHDL wrapper. After verifying a successful swapping via PR, this coprocessor with its nine variations for each hardware accelerator was integrated into the RadPC computer architecture.

This next step required exploring the best option for performing PR on the coprocessors to swap in and out different functionality. Figure 5.4 shows the options investigated. Integration of the FPU coprocessor directly within the RadPC architecture was attempted first. However, it was found that direct integration with RadPC required PR regions

Determining Optimal PR Strategy for Coprocessor Regions

Option 1 – Each Coprocessor is a Separate PR Region Within the RadPC QMR Architecture



Advantage

- 1) The coprocessors are QMR'd giving the ability to produce the correct data if one coprocessor experiences an SEU.

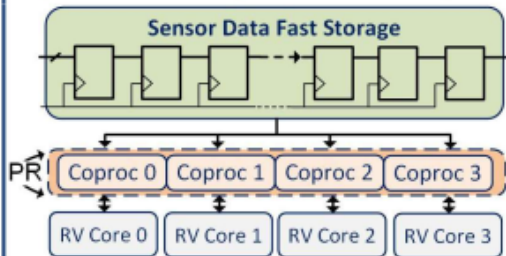
Disadvantage

- 1) Uses the largest amount of FPGA resources.
- 2) Uses the largest number of PR bitstreams to support each function ($4 \times N_{\text{functions}}$).

Conclusion

It was found that the sequential nature of swapping the coprocessor functions using PR (i.e., Coproc 0 is PR'd, then Coproc 1, then Coproc 2, etc.) led to the RV cores becoming unsynchronized. Also, each coprocessor PR bitstream has overhead associated with it to enable accessing a specific location on the FPGA. This led to the off-chip storage requiring an excessive amount of memory to hold the multiple bitstreams. ✗

Option 2 – The Coprocessors are Implemented in QMR Within a Single PR Region



Advantage

- 1) The coprocessors are still in QMR.
- 2) No synchronization issues.
- 3) The single bitstream file overhead is less than using 4x separate PR files.

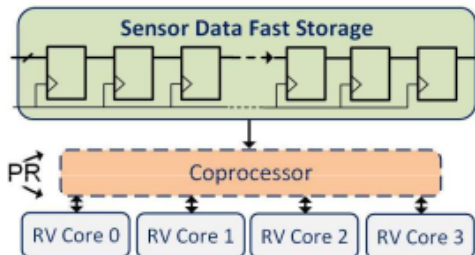
Disadvantage

- 1) The size of the PR region is still relatively large, which leads to long swap times and uses a large amount of FPGA resources.

Conclusion

It was found that even when relatively simple functions (i.e., floating-point operations) were QMR'd into a single PR region, it took a significant amount of FPGA resources. This approach was found to be unscalable to more complex algorithms. ✗

Option 3 – A Single Coprocessor Function is Implemented Within a Single PR Region



Advantage

- 1) Fastest PR time.
- 2) Smallest resource usage.
- 3) No synchronization issues.

Disadvantage

- 1) No QMR for SEU mitigation in coprocessor circuitry.

Conclusion

It was found that a single coprocessor uses a relatively small amount of FPGA resources, so this approach has the ability to scale to more complex algorithms. It was also determined that the PR of the coprocessor region itself will scrub out SEUs, thus giving a relatively quick fault elimination approach. It was decided that a blank PR region can be used for the coprocessor block when not in use to avoid SEUs from accumulating. ✓

Figure 5.4: Coprocessor placement in computer architecture

where all four RISC-V cores had to be individually configured with the nine different FPU configurations. This proved impractical from a timing perspective as it required $4 \times N_{functions} + 1$ partial bitstreams for N coprocessors while also maintaining RadPC's PR recovery mechanisms. The bitstreams needed to hold both a single RadPC softcore and a coprocessor greatly increased the PR region size. Changing the design to a coprocessor operating in parallel with RadPC allowed the number of bitstreams to be reduced to $(4 + (N_{functions} + 1))$ for N coprocessor operations while preserving RadPC's recovery operations via PR. Furthermore, this design verified that a coprocessor could be implemented in parallel with RadPC on resource limited hardware and perform complex mathematical operations.

Three designs for implementing radiation-tolerance in the coprocessors were investigated. Since the coprocessor is designed as part of radiation-tolerant system, the first coprocessor designed applied QMR to the coprocessor and used PR as both a swapping and recovery option. Four copies of a single function are implemented in parallel, each with an individual PR region. This approach had the highest radiation tolerance out of the coprocessor designs tested. An FPU was developed, and preliminary testing of a QMR version was done. After this testing, this approach was abandoned due to the large number of FPGA resources, PR bitstreams ($4 \times N_{functions} + 1$), and PR time required. Routing into the four PR regions introduced significant timing errors and increased the FPGA resource usage. Swapping the functionality of the coprocessor required a minimum of four PRs to be performed sequentially and massively increased swap latency. Furthermore, the number of bitstreams needed an enormous amount of external memory for storage between configurations. The second radiation-tolerant coprocessor design implemented QMR for the coprocessor within a single PR region containing all four coprocessors. Since all four coprocessors are designed with a single floating-point operation, the timing issues for routing around multiple PR regions were avoided. This approach still requires similar FPGA

resources as implementing the coprocessors within their own PR region, however the larger size of the PR region greatly increased the PR time until real-time operation was unfeasible.

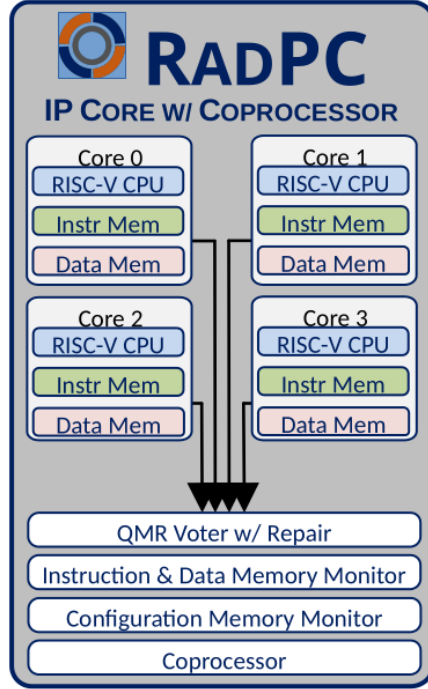


Figure 5.5: RadPC+Coprocessor block diagram.

The conclusion of this step was that it is most useful to have a single coprocessor region that can be partially reconfigured with different hardware accelerator functions as opposed to having QMR on the coprocessors either individually or as a single PR region containing 4x redundant coprocessors. The single coprocessor PR region is memory mapped to each of the 4x QMR computer cores through a voter so that each CPU accesses it as a standard peripheral. This allows the fastest PR time without desynchronizing the CPUs. This also allows the smallest PR bitstreams to be used, that minimizes the coprocessor swap time. It was concluded that while the coprocessor is not put into a QMR form that can vote out SEUs, the rate at that the coprocessor is partial reconfigured during normal operation will likely be faster than the incoming fault rate in the target environments for RadPC (i.e., LEO, cislunar, lunar). This essentially provides a memory scrubbing system

via PR for the coprocessor region. It was decided that the PR region can be loaded with a blank design to avoid SEUs from accumulating over time while the coprocessor is not in use. This further reduces the impact of SEUs by not allowing faults to accumulate in the coprocessor region. The most efficient coprocessor framework was found to be implementing the FPU within a single PR region as a peripheral of RadPC. This approach, shown in Figure 5.5, balanced the processor's fault tolerance, optimal hardware resource usage, coprocessor timing requirements, and usability.

RadPC & Coprocessor Integration

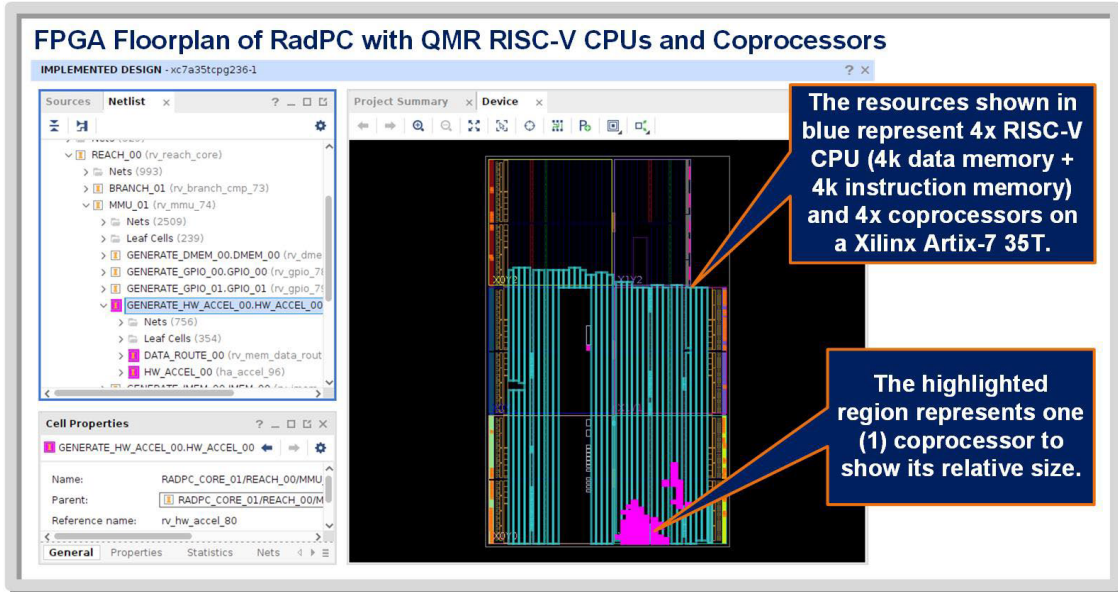


Figure 5.6: RadPC with floating-point coprocessor full configuration implementation

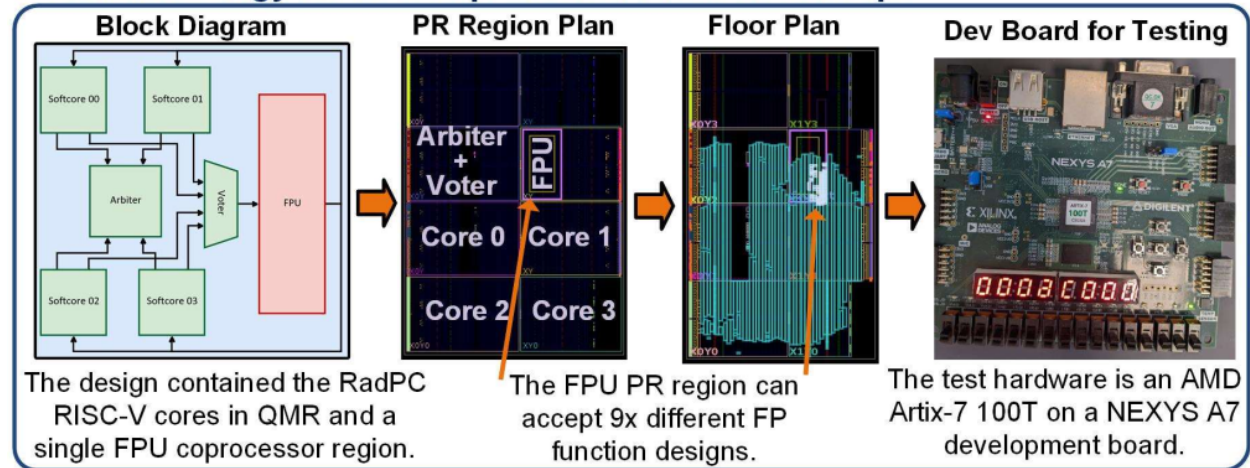
The integrated RadPC with Coprocessor system was loaded onto Nexus A7 FPGA Development Board using the Vivado Tool Design Suite. Configuration of the FPGA was controlled externally to the board via a desktop computer. Both full configuration and PR were performed by the user via the Vivado tool's Hardware Server. Functional testing was initially performed without swapping the coprocessor via PR. Instead, the QMR version

of RadPC was implemented with the single coprocessor consisting only of the hardware accelerator performing floating-point addition as shown in Figure 5.6. Each of the nine floating-point operations were individually implemented in this configuration to perform functional testing of RadPC loading the coprocessors and verifying the outputs for floating-point operations, ADD, SUB, MUL, MAC, DIV, DIVBY2, MOD, REM, and RCP.

Finally, a test program for RadPC was created to exercise each of the nine FPU operations using the following steps. The test program tested all nine of the floating-point operations in series starting with addition, serial report and timing are shown in Figure 5.7.

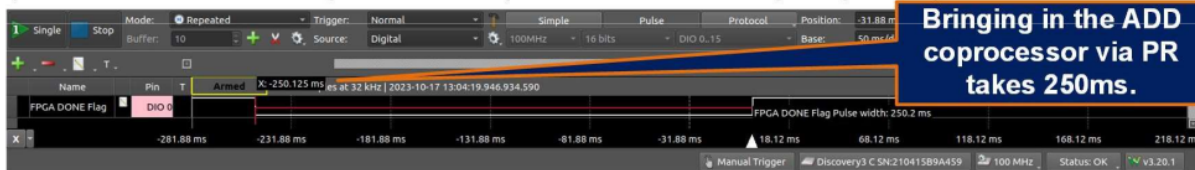
1. RadPC loads inputs into the FPU coprocessor via storing floating-point numbers in the memory map location corresponding to inputs A, B, and/or C.
2. RadPC requests the desired FPU operation from the user using control registers for the coprocessor peripheral and repeats the request over a serial terminal.
3. RadPC waits for the coprocessor to be configured with the desired FPU operation
4. The partial reconfiguration time is measured using an external logic analyzer to monitor the DONE pin on the JTAG configuration interface to the FPGA.
5. The coprocessor begins floating-point computation using the start flag in the control register
6. The compute time is measured using an external logic analyzer to monitor a coprocessor signal flag that is routed to the GPIO of the FPGA development board and is toggled at the start and end of the FPU coprocessor operation.
7. RadPC reads the results from the coprocessor's memory map location and writes the results to a serial terminal for the user to compare the FPU computed results to the expected correct results for the selected operation.

Test Methodology for FPU Coprocessor Proof-of-Concept



Test Methodology

- 1) Load peripheral w/ known input values, select desired FP function, and trigger computation.
(i.e., $A + B = D \rightarrow 10.75 + 4.25 = 15.00 \rightarrow 0xAC000 + 0x44000 = 0xF0000$)
- 2) Measure the time to bring on the PR coprocessor via PR using a logic analyzer.



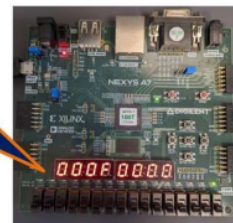
- 3) Measure the time to complete the FP operation.



- 4) Print the results to a serial monitor and the HEX displays to verify proper results.

```
[11:17:09:040] RadPC FPU Demo Program %
[11:17:09:040] PR External FPU Addition Test %
[11:17:09:056] a + b = d %
[11:17:09:056] 10.75 + 4.25 = 15.00 %
[11:17:09:056] Correct Answer = : 0xF0000 %
[11:17:09:056] Input A (Hex): 0xAC000 %
[11:17:09:056] Input B (Hex): 0x44000 %
[11:17:09:056] Input C (Hex): 0x0 %
[11:17:09:056] Output D (Hex): 0xF0000 %
```

The result of the operation is **CORRECT**.



- 5) Continue to the next FP function and repeat test.

Figure 5.7: RadPC with FPU implementation showing FPGA layout, operation, and testing of the ADD on physical hardware (Xilinx 100T FPGA)

8. RadPC writes the result to the HEX displays on the Nexys A7 FPGA board for additional verification.
9. Steps 1-8 are repeated for all nine floating-point operations.

During configuration, the Artix 7 100T's *JTAG Done* pin is automatically pulled low by the FPGA and will go high once the configuration is finished. Timing for full and partial configurations was measured on this *JTAG Done* pin from the JTAG interface by Analog Discovery 2 (AD2). The coprocessor's completed operation flag, *done*, was routed from the coprocessor out to an external FPGA I/O where it is probed externally using the AD2. This measurement is used to track when the floating-point operation started and when it ended using a state machine in the coprocessor to raise the *done* before an operation and lower it afterwards. The flags *ready* and *busy* were routed out and measured to record the time required for data transfer between RadPC and the coprocessor. As shown in Figure 5.5, data comes from the four CPUs to a voter. From that voter, the data is fed into the coprocessor. When the coprocessor completes its operations, the data is routed back to all four of the CPUs. The full testing methodology is shown in Figure 5.7 for the ADD floating-point operation.

All nine floating-point operations were successfully performed by the coprocessor and the results for PR times, operational time with data transfer delay, and operational times of only floating-point operations are shown in Table 5.2. PR times are the total time required to complete the reconfiguration as measured from the start of the bitstream transmission until the coprocessor reports ready for operation. The time taken by the user to manually order the PR to start was not included. Compute time was measured as the time required for the coprocessor to receive an input, perform the floating-point operation, and return the result. This time was measured via a flag raised by the coprocessor upon receiving input data and lowered once results are output. The floating-point operations were implemented

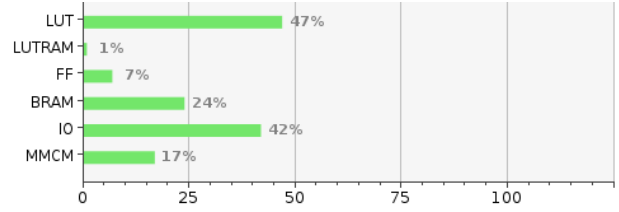
as combinational logic that by default was much faster than if performed by the CPU. To ensure the flag accurately matched the start and end of the computational time, the floating-point operations were implemented in a process where the flag must be raised and lower immediately before and after operation. As shown in Table 5.2, the PR times are a factor of 10^5 greater than compute times and 10 compute times with delay.

Table 5.2: Individual Floating-Point Operation PR and Operation (Op) Times.

Name	Function	Test Operation	PR	Op (Delay)	Op
NOP	No Operation	N/A	250 ms	N/A	N/A
ADD	Addition	$10.75 + 4.25 = 15.00$	250 ms	13.65 ms	3.11 us
SUB	Subtraction	$10.75 - 4.25 = 6.50$	250 ms	13.84 ms	3.11 us
MUL	Multiplication	$6.5 * 4.25 = 27.625$	250 ms	14.12 ms	3.11 us
MAC	Multiply/Accumulate	$2 * 5 + 0.75 = 10.75$	250 ms	15.04 ms	3.61 us
DIV	Division	$319.33 / 6.5 = 49.12769$	250 ms	14.12 ms	3.11 us
DIVBY2	Division-by-2	$319.33 / 16 = 19.958125$	250 ms	15.51 ms	3.11 us
MOD	Modulus	$16 \bmod 5 = 1$	250 ms	13.11 ms	3.11 us
REM	Remainder	$\text{rem } 15 / 4 = 3$	250 ms	13.56 ms	3.11 us
RCP	Reciprocal	$1 / 5 = 0.2$	250 ms	10.68 ms	2.62 us

RadPC & Coprocessor Resource Usage The RadPC and floating-point coprocessor implementation on the Artix 7 100T FPGA used approximately 40% of the available Look-Up Table (LUT) and I/O resources on the FPGA. In Figure 5.8c, the combined totals for RadPC and the floating-point coprocessor are listed under *RADPC_01* as in this version the coprocessor is integrated into RadPC as a component under the top level. Total resource usage of the RadPC and the floating-point coprocessor design is shown in Figure 5.8 and demonstrates the resource limitation for RadPC and an integrated

Resource	Utilization	Available	Utilization %
LUT	29772	63400	46.96
LUTRAM	118	19000	0.62
FF	8953	126800	7.06
BRAM	33	135	24.44
IO	88	210	41.90
MMCM	1	6	16.67



(a) RadPC & FPU Resource Utilization Totals

(b) RadPC & FPU Resource Utilization (%)

Name	^1	Slice LUTs (63400)	Slice Registers (126800)	F7 Muxes (31700)	F8 Muxes (15850)	Slice (15850)	LUT as Logic (63400)	LUT as Memory (19000)	Block RAM Tile (135)	Bonded IOB (210)	BUFGCTRL (32)	MMCME2_ADV (6)	BSCAN2 (4)
top_nexys		29772	8953	1134	244	8916	29654	118	33	88	12	1	1
> CLK_00 (clk_wiz_0)		0	0	0	0	0	0	0	0	0	2	1	0
> dbg_hub (dbg_hub)		450	741	0	0	226	426	24	0	0	1	0	1
> ILA_00 (ila_radpc_0)		693	1207	4	0	362	599	94	1	0	0	0	0
> RADPC_01 (rv_radpc_core)		28597	6985	1126	244	8398	28597	0	32	0	9	0	0
SSD_00 (seven_seg)		18	20	4	0	11	18	0	0	0	0	0	0

(c) RadPC & floating-point Coprocessor Resource Utilization by Component

Figure 5.8: RadPC & FPU Resource Utilization Summary

coprocessor to be implemented on a 100T FPGA. Between both RadPC and the coprocessor needing a significant amount of LUTs, scaling the coprocessor to more complex or multiple simultaneous operations will be difficult without moving to a larger FPGA. The floating-point coprocessor was deliberately designed to implement only one floating-point operation per accelerator in order to keep the resource requirements low since the original targeted FPGA for implemented was an Artix 35T. This targeted FPGA was changed to the Artix 100T because Artix 35T did not have significant resources. Given the 40% LUT usage in this design, the target FPGA will be changed again to the Artix 200T. While not required for the current RadPC and coprocessor implementation, this change will allow future development for both RadPC and the coprocessor. The system will need to scale on both RadPC's side as more radiation tolerance techniques and more computing capability is incorporated into it and on the coprocessor side as it improves from a single implemented function per hardware accelerator to multiple parallel operations of either a single function or several functions.

RadPC & Floating-Point Coprocessor Limitations

This floating-point coprocessor design confirmed that dynamically reconfigurable coprocessors for low-power, real-time operation are possible for small satellites. Custom coprocessors implemented with this design increased the processing capability of low power systems by adding single point floating-point capability to an integer-based system. Furthermore, the circuitry required for an automated dynamically partially reconfigurable system fits within the size, mass, and power constraints for a small satellite. Key future applications includes image preprocessing/digital signal processing where dedicated processing hardware is needed for a short period [189, 190]. An image preprocessing coprocessor could be swapped after use and the hardware could then be used to support other operations. The advantage of this design approach is the capability to change the hardware during real-time operation. The reconfiguration process would also clear out any SEEs while loading the new coprocessor. This adds a limited, but beneficial recovery method from radiation strikes for the coprocessor subsystem.

This design illustrates several issues that need addressing. First, the partial configuration latency is the dominant factor for real-time capability. PR time was 250 ms vs 3.61 μ s for compute time. With the current hardware setup, the CPU is required to pause operation and wait for the user to initiate the PR. Between the PR time and the wait time, the CPU spent the majority of its time during testing paused. Obviously, the CPU being paused for the majority of the time would not be acceptable for actual operations. PR latency needs to be reduced, and the PR process is automated for this design to work practically in real-time operation. Second, the hardware setup was lacking in key areas to support PR. FPGA size limited the size of coprocessors that could be implemented in parallel to the main processor. There were major limiters in the available memory and integrated hardware support for PR. FPGA RAM limited storage for coprocessor data transfer. FPGA size limited to the system size due to routing conflicts on the FPGA. Further, the external memory to store the

FPGA configuration bitstreams limits the number of coprocessor configurations. These issues demonstrated the need for hardware capable of storing larger and more bitstreams, rapidly configuring the FPGA, and a larger FPGA to support more complex coprocessor operation. Third, the coprocessor design did not adequately address radiation tolerance. Constraints on partial bitstreams numbers limited to the redundancy built into the coprocessor component. With the current design, the coprocessor's main radiation protection and recovery method is PR. This method flushes out faults during the PR swapping the coprocessor in then assumes the coprocessor will complete operation and be swapped out before being faulted. Modeling is needed to prove this works in the majority of cases with radiation testing on the finalized design for proof of radiation tolerance.

Another issue was with the software compiler for RadPC. Since RadPC is compiled as a RISC-V 32I system, the compiler will not allow the use of single point float for the code of RadPC. While RadPC is technically now capable of floating-point operations, it requires a custom header file for the developer to use floating-point variables as opposed to defining variables with the standard *float* and *double* datatypes. Finally, the partial reconfiguration process is not supported by the software compiler, so care must be taken as to how the software interface to the coprocessors is handled to avoid syntax errors.

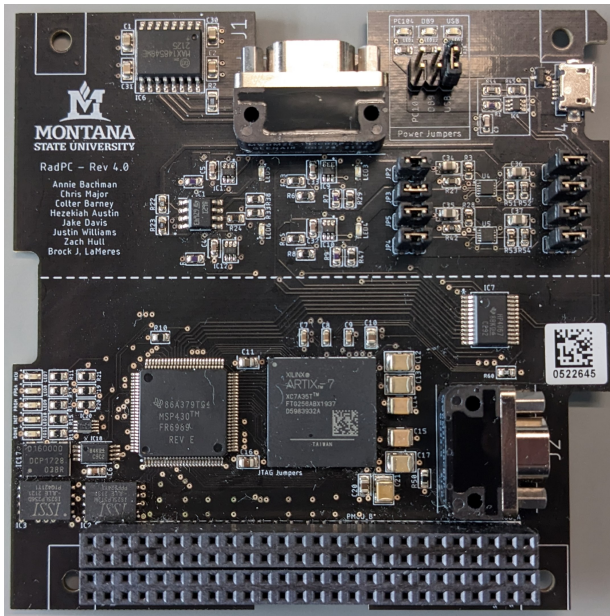
Hardware: Single Board Computer

From the limitations discovered during the development of the floating-point coprocessors, a single board computer was developed to address these issues. This single board computer is a custom PCB and the second artifact developed during the course of this research. This board was given the tradename *Hydra* and will be referred to as such for the remaining of this manuscript. Hydra focuses on addressing the hardware limitations of the publicly available FPGA development boards and previous single board computers designed for RadPC. These limitations were primarily issues with automated PR capability not being supported on the FPGA development boards, leading to the need for external equipment to perform the coprocessor swapping.

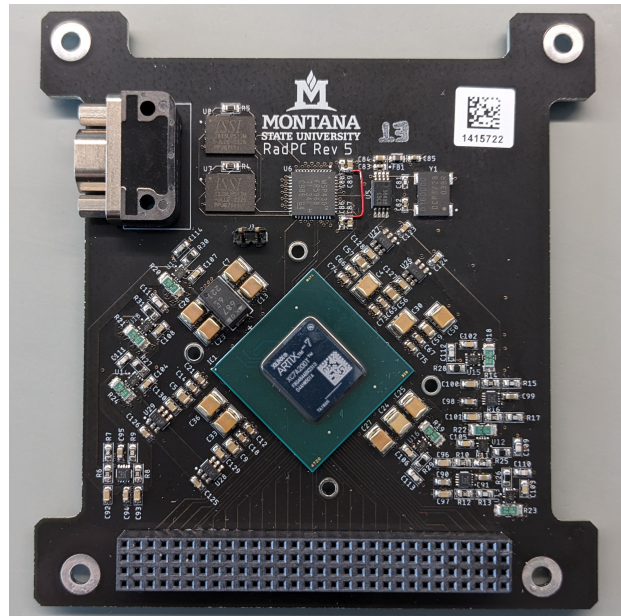
Prior Work

A previous single board computer had been developed by Montana State University (MSU) to support RadPC for space operation. This version, shown in Figure 5.9a, performed well on both the International Space Station and Firefly’s Blue Ghost Lunar Lander.

Unfortunately, this board did not support the required number of GPIO, capability for full and PR, or power monitoring necessary. Revision 4 (Rev4) was used as a starting point for the new hardware design shown in Figure 5.9b. At this point in the design process, the new single board computer consisted of an extension of the same line of boards and was designated the RadPC Revision 5 (Rev5). Primary differences between Rev4 and Rev5 were a larger FPGA, on-board power monitoring, reworked voltage regulation circuits and power islands, switching the microcontroller to one with a radiation hardened version, switching to the PC104-Plus standard connector, and providing external memory for RadPC. Using the 200T FPGA for the Rev5 provided RadPC with a significant amount of logic resources for future development. Estimates during the design phase suggested that RadPC with a



(a) RadPC Revision 4 SBC developed for ISS and Lunar operations



(b) RadPC Revision 5 SBC developed for partial reconfiguration support

Figure 5.9: Single Board Computer & Softcore Processor with Coprocessor

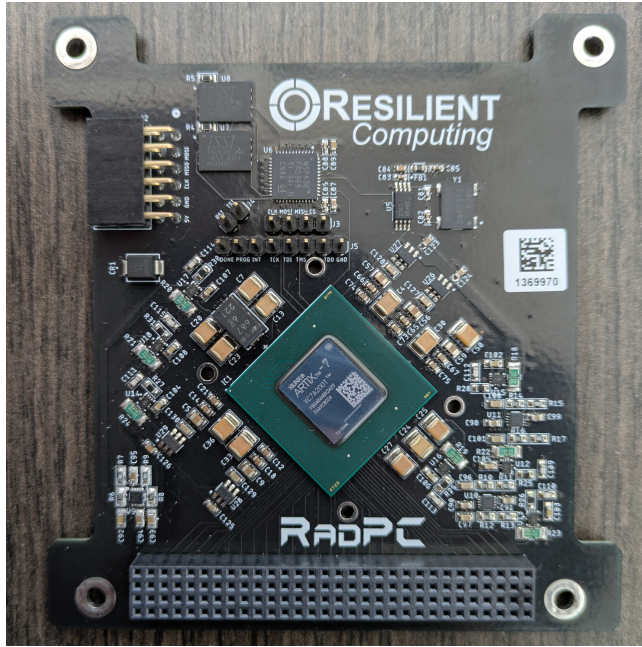
coprocessor would take between 20% and 30% of the 200T. The on-board power monitor measures the individual power rails. Additionally, the microcontroller has controller over the power rails to individual FPGA I/O banks, allowing these banks to be cycled to clear out SEUs on the I/O lines.

Design

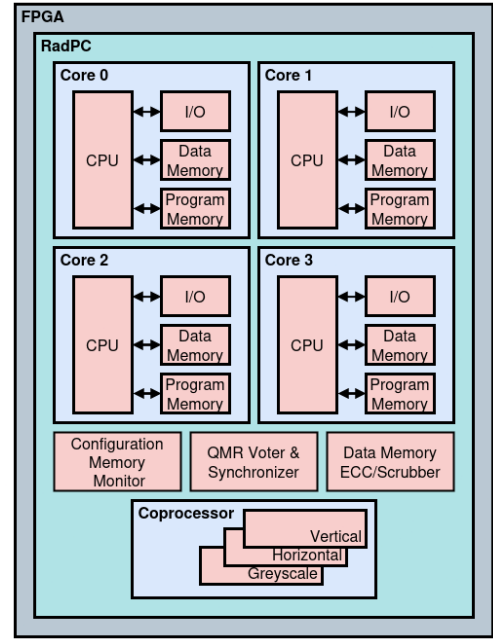
For a small satellite on-board computer, the new Hydra OBC had to provide low power consumption, acceptable computational performance, some radiation tolerance, and be affordable for a small satellite project. Hydra also needed to provide the on-board PR capability needed for a dynamically reconfigurable coprocessor. Hydra consolidates all capabilities used in previous research onto a single board, adding FPGA configuration, power monitoring, UART support, SPI support, and memory storage for bitstreams. For radiation tolerance, Hydra was developed for use with RadPC, where the radiation tolerance

softcore processor would provide reliability in harsh radiation environments. RadPC has an architectural approach to mitigating SEEs from radiation strikes where the detection and recovery is abstracted from the software developer. RadPC is an FPGA-based, flight-proven aerospace processor developed by MSU. This computer was designed for radiation tolerance and uses a 32-bit, reduced instruction set, integer softcore processor (RISC-V RV32I). This processor is an architectural approach to mitigate SEEs that expands triple modular redundancy to quad modular redundancy at the CPU level as shown in Figure 5.10b. TO the user, RadPC functions as a single core microcontroller using a QMR RISC-V architecture implemented on an FPGA [191, 192]. This processor uses a number of recovery systems including, redundancy, voters, memory scrubbers, register-level arbitration, and configuration memory monitoring for maintaining operation during faults. When faults are detected, the faulted core undergoes recovery procedures. Suspect results from the faulted core are outvoted by the three trusted cores and ignored during recovery. In the event of a complete failure of a core from SEEs, the individual core is partially reconfigured, memory and registers are reloaded to synchronize with the other cores and operation is continued [193, 194]. These recovery methods, combined with memory scrubbers, provide resilience from SEE faults. [191, 193, 195]. This architectural approach to recovery from radiation induced faults allows the system to be easily expanded to include dynamic reconfigured coprocessors. Since RadPC uses PR for recovery from radiation strikes, the hardware and softcore architecture is easily adaptable to using PR to dynamically swap coprocessors. Furthermore, RadPC is an aerospace processor developed for small satellites. The coprocessors for RadPC were developed to increase the capability of edge computing for limited resource computing systems operating in harsh radiation environments on small satellites.

The developed board, shown in Figure 5.10a, uses a Xilinx Artix 7 200T FPGA with an implemented CPU, coprocessor, and peripherals. Also, on the board are a



(a) Hydra: Single Board Computer



(b) RadPC+Coprocessor

Figure 5.10: Single Board Computer & Softcore Processor with Coprocessor

separate microcontroller and memory to provide FPGA full and partial configuration with dedicated memory to support full and partial configuration. This board also includes quad redundant user memory for RadPC. A microcontroller, MSP430, functions as the on-board housekeeper. It provides full and partial configuration via the FPGA's JTAG port. FPGAs are configurable via several protocols, Master Serial, Master SPI, Master BPI, Master Select MAP, JTAG, SlaveSelectMAP, and Slave Serial [196]. Of these protocols, only the JTAG protocol is capable of both full and partial configuration and is also an externally driven protocol that allows the configuration controller system to provide additional radiation tolerance or hardening in the design. Furthermore, the memory size for FPGA configuration files and RadPC executable files was increased to support partial configuration and multiple executable files.

Two connectors were selected for Hydra to provide a programming/debugging port and

a PCB stacking port. For the programming/debugging port shown in the upper left of Figure 5.10a, a 12-pin PMOD connector was finally selected over the 9-pin DSUB connector, shown in the upper left of Figure 5.9b. This PMOD connector was selected for the higher number of pins and much lower cost ($\sim \$2$) compared to the DSUB connector ($\sim \$600$). A 120-pin PC104-Plus connector was selected for the PCB stacking port. PC104-Plus connector provides more designated I/O pins compared to the 104-pin CubeSat Kit connector. Shifting to a connector with a tighter pitch. This allows more pins on the PCB and allows the design to take advantage of the higher number of I/O available on the 200T FPGA.

Hydra has a total of seven on-board power rails; 5.0V Board, 1.0V FPGA Core and BRAM, 1.8V FPGA Auxiliary, 3.3V FPGA I/O Banks 34 and 35, 3.3V FPGA I/O Banks 15 and 16, 3.3V Integrated Circuits, and 3.3V PC103-Plus Connector. Each of these rails was monitored for voltage and current. Voltage monitoring used an op-amp directly tapped into the rail upstream of the load. Current monitoring uses shunt resistors and an op-amp to convert the current into a voltage drop. The 5.0V rail provides power for the entirety of the board while all other rails are fed by local voltage regulators. In addition to monitoring power, the microcontroller can selectively shut off power rails either at the voltage regulator or power switches for the individual FPGA I/O banks. Combined these systems give the Hydra the ability to perform a phased power up sequence, selective shutdown of individual power rails, and power cycling chips to clear SEUs.

Finally, a off-FPGA quad modular redundant memory system was designed to provide RadPC with a memory chip per CPU core. The memory system is designed to function as RadPC's hard drive. The memory chips are non-volatile and will retain data through power cycles. Since each of RadPC's CPUs is running in lockstep, these hard drives will also function as redundant copies of each other and provide a radiation-tolerant data storage option.

Hardware Development Summary

This solution underwent four revisions of the PCB before being completed with Hydra. Each revision solved an issue with the previous board. Rev5 had errors in the microcontroller power supply. Rev6 had a fault in the FPGA BRAM power supply that was slowly burning out the BRAM. Rev7 was fully functional without any faults. However, the JTAG for FPGA configuration required a custom bit-level interface on the microcontroller side. This greatly increased the required time for full and partial configuration. To solve this issue, the Hydra used dedicated hardware for the clock and outgoing data pin to speed up the JTAG transaction and configuration speed. In its final form, this artifact was capable of on-board full and partial FPGA configuration, on-board power monitoring, on-board power cycling, boot loading RadPC its executables, and redundant memory for RadPC operation.

Coprocessor: First-Difference Edge Detection

Research for the floating-point coprocessor and PCB verified the system was integrable into the RadPC computer architecture and the hardware allowed for dynamic hot-swapping between coprocessors via PR using an external microcontroller. The next step was the automation of the partial configuration process. Shifting to an on-board configuration process where the microcontroller configures the FPGA via its JTAG port removes the wait time introduced from manual external configuration via the Vivado tool and Xilinx Platform II Cable. Based on the previous work on the floating-point coprocessor and PCB, three (3x) edge detection coprocessors were developed to perform RGBA (Red, Green, Blue, Alpha) to Grayscale Conversion, Horizontal first-difference Edge Detection, and Vertical first-difference Edge Detection.

This research expanded previous work by automating the swapping procedure for the coprocessors, making the coprocessor I/O available to RadPC on the software level, and focusing on image processing applications. These image processing applications will provide a proof-of-concept for dedicated vector processing via a coprocessor. For dynamic partial reconfiguration, an external microcontroller was used to control the FPGA configuration process, and an external memory was used to store multiple PR bitstreams. While automated swapping does not decrease actual PR time, it does decrease effective PR time by removing manual commands required from the user. The coprocessor interface with RadPC was standardized to eight 32-bit input registers and eight 32-bit output registers.

Evaluation criteria for this coprocessor focused on four key points: successful swapping the coprocessor via PR, PR time, coprocessor operation, and coprocessor compute time. Successful completion of these key points demonstrates the potential of dynamic partial reconfiguration as a method to increase capability in small, low power satellites. To verify a successful reconfiguration, test inputs were loaded into the coprocessor, and the results were

compared to the Vivado tool simulations of the coprocessor's operation. The interface for PR was monitored and recorded for reconfiguration time. Operation was tested via comparison between calculated, simulated, and actual output images. Inputs were limited to 16×16 -pixel images (256 pixels total) due to memory constraints on RadPC for this implementation and for easy verification via manual calculation.

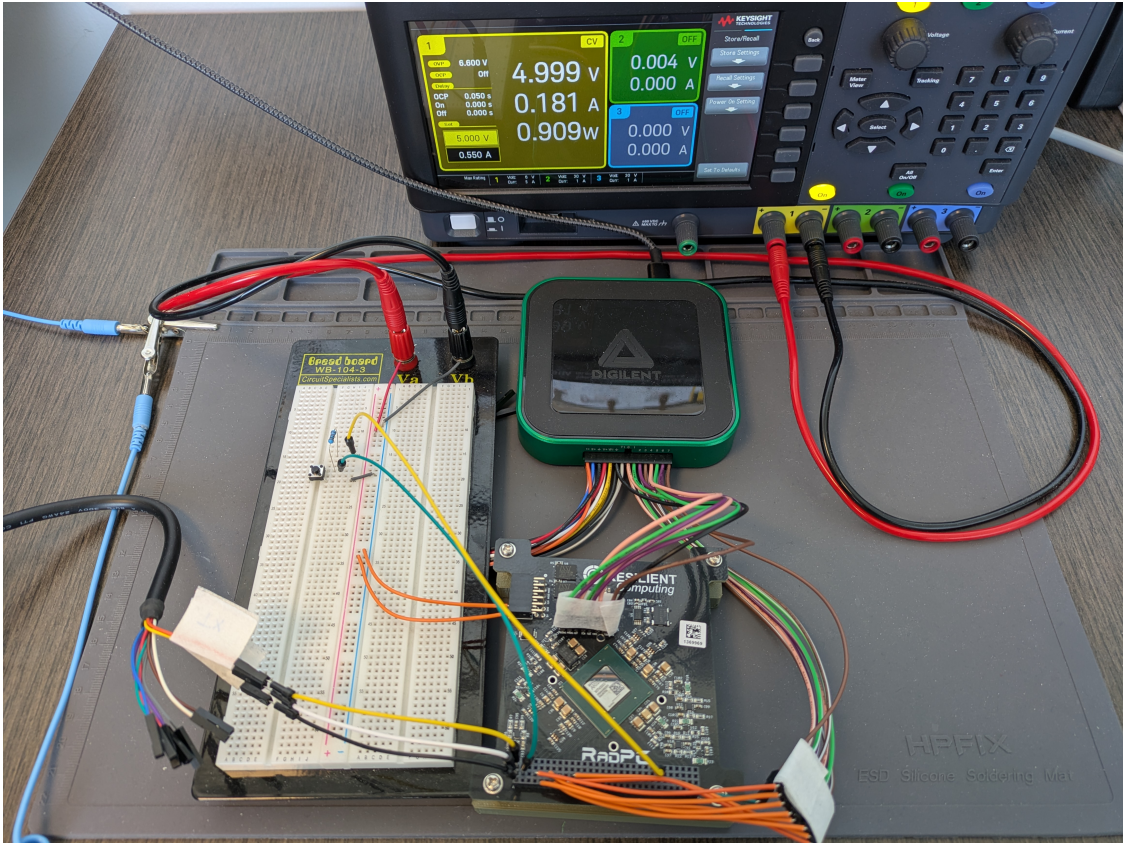


Figure 5.11: Experiment Test setup: power supply, AD3, reset button, FTDI cable, and Hydra

Real-time swapping of the coprocessor via dynamic partial reconfiguration was tested on the Hydra Single Board Computer. The test setup consisted of a power supply, a Hydra OBC, an Analog Discovery 3 (AD3), and a FTDI USB-to-UART cable as shown in Figure 5.11. PR on Hydra was monitored by the Analog Discovery via header pins on the FPGA's JTAG interface. Capturing these signals allowed the configuration process to be recorded

during coprocessor swapping. During operation, the results were recorded on a desktop computer via the UART cable and a terminal logger. The AD3 monitored 9 General Purpose Input/Output (GPIO) pins from the Hydra's PC104 connector. These pins consisted of the most significant byte of the coprocessor output and the coprocessor operation flag. These signals allow verification of the coprocessor functionality during operation and the measurement of total operation time.

During this set of tests Hydra operated on two clock domains, 16 MHz for the microcontroller handling PR over JTAG, and 32 MHz for RadPC. A 16 MHz clock was provided on the board, and this clock was speed up inside the FPGA to a 32 MHz clock for RadPC. The coprocessor was designed with all combinational logic. While a 32 MHz clock was routed into the coprocessor, it was not used. For tapping the 16 MHz JTAG port, the Analog Discovery sampled at 33.34 MHz to avoid aliasing and ensure a stable recording of the swapping process. Timing for the PR was measured using a logic analyzer. Operation was measured via a GPIO pin that was set to raise immediately after the coprocessor was loaded and lowered immediately after the coprocessor outputs were read. This measurement included RadPC's time to load input registers and read output registers since this time is inherently part of the operational time. It does not include the data transfer time RadPC used to read/write pixels from RadPC's data memory. Coprocessor's operation time was measured without time for data transfer added on either end of the coprocessor's operation. For these tests, images were preloaded into RadPC's memory and results from each step of the edge detection were passed over UART to a terminal for logging. Logged results were manually converted from hex codes into full pixels and paired with the respective image header to create a bitmapped image.

RGB-to-Greyscale converter

A compact version of a first-difference edge detector was developed. Using only combinational logic and reworking the coprocessor I/O allowed it to handle multiple pixels at once. Pictures of a coffee cup, the MSU Logo, and the Moon's lunar craters were selected as test images. These images were cropped to a square and down sampled to 16×16 pixels. An RGBA to Greyscale converter was developed using the rightwise bitshift and addition method for efficient FPGA resource utilization [197]. A standard converter uses multiplication with a fraction to achieve the conversion. However, a rightwise bitshift is the equivalent of a division by 2. Performing several series of rightwise bitshifts in parallel and adding the results gives an answer approximately equal to a much longer multiplication by a fraction. Using this method instead of a standard multiplication reduced the FPGA resources required and increased the coprocessor's data throughput. See Table 5.3 for the used conversion factors vs the actual conversion factors.

Table 5.3: Greyscale Conversion Weights

Colors	Standard Weight	Used Weight	Conversion Factor
Red	0.299	0.298828125	C_{red}
Green	0.587	0.585937500	C_{red}
Blue	0.114	0.113281250	C_{red}

This approach illustrates the advantage of using a coprocessor instead of the main processor. The main processor, RadPC, is an integer operation only processor and is not capable of rapidly calculating the greyscale conversion of multiple pixels. This method generates slight incorrect conversion factors for Red, Green, and Blue (RGB) pixels. However, it saves FPGA resources by only using D-Flip-Flop and addition circuitry on

the FPGA. Decimal weights are hard coded into the greyscale conversion operation in the accelerator that implemented Equation 5.2 for RGB pixels conversion into greyscale pixels.

$$\begin{aligned}
Grey &= C_{red} * R_{byte} + C_{green} * G_{byte} + C_{blue} * B_{byte} \\
C_{red} * R_{byte} &= R_{RGB} \gg 2 + R_{RGB} \gg 5 + R_{RGB} \gg 9 + R_{RGB} \gg 9 \\
C_{green} * G_{byte} &= G_{RGB} \gg 1 + G_{RGB} \gg 4 + G_{RGB} \gg 6 + G_{RGB} \gg 7 \\
C_{blue} * B_{byte} &= B_{RGB} \gg 4 + B_{RGB} \gg 5 + B_{RGB} \gg 6 + B_{RGB} \gg 8
\end{aligned} \tag{5.2}$$

These equations were implemented by the RGB-to-Greyscale hardware accelerator inside the First-Difference Coprocessor.

Horizontal Edge Detection

After the 16×16 -pixel images was converted to greyscale, the coprocessor was swapped from RGB-to-Greyscale hardware accelerator to a First-Difference Edge Detection hardware accelerator, and the greyscale image was passed through it. These edge detection operations are performed on a 4×6 pixel matrix; the leading and tailing columns of pixels were prior pixels or next pixels. For the 4×4 blocks of pixels that were on the edge of the 16×16 -pixel images, the first column of pixels was reused as the prior or next pixels. This allowed a generalized algorithm to be used inside the coprocessor with the edges of the image being accounted on the software level. As shown in Equation 5.3, a subtraction is performed between the current pixel and the prior pixel. If the result is greater than a threshold value, then a black ($0 \times FF$) or white (0×00) pixel is output. A total of 16 pixels are handled in parallel, giving a 4×4 matrix as the output.

$$\begin{aligned}
& \begin{bmatrix} P_{-1,0} & P_{0,0} & P_{1,0} & P_{2,0} & P_{3,0} & P_{4,0} \\ P_{-1,1} & P_{0,1} & P_{1,1} & P_{2,1} & P_{3,1} & P_{4,1} \\ P_{-1,2} & P_{0,2} & P_{1,2} & P_{2,2} & P_{3,2} & P_{4,2} \\ P_{-1,3} & P_{0,3} & P_{1,3} & P_{2,3} & P_{3,3} & P_{4,3} \end{bmatrix} \Rightarrow \\
& \begin{bmatrix} P_{-1,0} - P_{0,0} & P_{0,0} - P_{1,0} & P_{1,0} - P_{2,0} & P_{2,0} - P_{3,0} & P_{3,0} - P_{4,0} \\ P_{-1,1} - P_{0,1} & P_{0,1} - P_{1,1} & P_{1,1} - P_{2,1} & P_{2,1} - P_{3,1} & P_{3,1} - P_{4,1} \\ P_{-1,2} - P_{0,2} & P_{0,2} - P_{1,2} & P_{1,2} - P_{2,2} & P_{2,2} - P_{3,2} & P_{3,2} - P_{4,2} \\ P_{-1,3} - P_{0,3} & P_{0,3} - P_{1,3} & P_{1,3} - P_{2,3} & P_{2,3} - P_{3,3} & P_{3,3} - P_{4,3} \end{bmatrix} = \quad (5.3) \\
& \begin{bmatrix} P_a & P_b & P_c & P_d \\ P_e & P_f & P_g & P_h \\ P_i & P_j & P_k & P_l \\ P_m & P_n & P_o & P_p \end{bmatrix}, \text{ where } P_x = \begin{cases} 0 \times FF & > Threshold \\ 0 \times 00 & \leq Threshold \end{cases}
\end{aligned}$$

Setting up the accelerator to perform multiple operations in parallel allows the entire 16×16 -pixel image to be fully processed in 4 operations. This maximizes the coprocessor's operation speed by having the image loaded in 4 chunks for processing. One issue with this method is the pixel offset. Pixels are read left to right with the edge detection comparing the prior pixel to the current pixel, causing the result to be shifted one pixel to the right.

Vertical Edge Detection

Vertical edge detection was performed using the same method with a rotation in the direction of reading. Pixels for the vertical edge detector were read from top to bottom. Operations are performed on a 4×6 pixel matrix to give a 4×4 pixel output as shown in Equation 5.4

$$\begin{aligned}
& \begin{bmatrix} P_{0,-1} & P_{1,-1} & P_{2,-1} & P_{3,-1} \\ P_{0,0} & P_{1,0} & P_{2,0} & P_{3,0} \\ P_{0,1} & P_{1,1} & P_{2,1} & P_{3,1} \\ P_{0,2} & P_{1,2} & P_{2,2} & P_{3,2} \\ P_{0,3} & P_{1,3} & P_{2,3} & P_{3,3} \\ P_{0,4} & P_{1,4} & P_{2,4} & P_{3,4} \end{bmatrix} \Rightarrow \\
& \begin{bmatrix} P_{0,-1} - P_{0,0} & P_{1,-1} - P_{1,0} & P_{2,-1} - P_{2,0} & P_{3,-1} - P_{3,0} \\ P_{0,0} - P_{0,1} & P_{1,0} - P_{1,1} & P_{2,0} - P_{2,1} & P_{3,0} - P_{3,1} \\ P_{0,1} - P_{0,2} & P_{1,1} - P_{1,2} & P_{2,1} - P_{2,2} & P_{3,1} - P_{3,2} \\ P_{0,2} - P_{0,3} & P_{1,2} - P_{1,3} & P_{2,2} - P_{2,3} & P_{3,2} - P_{3,3} \end{bmatrix} = \\
& \begin{bmatrix} P_a & P_b & P_c & P_d \\ P_e & P_f & P_g & P_h \\ P_i & P_j & P_k & P_l \\ P_m & P_n & P_o & P_p \end{bmatrix}, \text{ where } P_x = \begin{cases} 0 \times FF & > Threshold \\ 0 \times 00 & \leq Threshold \end{cases}
\end{aligned} \tag{5.4}$$

The vertical edge detection accelerator also had the issue that the entire image was shifted by one pixel in the direction being read, in this case down one pixel. After completing the image processor, the resulting 16×16 -pixel images are in black and white (B/W) and are using a 1 byte per pixel representation. These pixels were manually converted to an RGBA representation of 4 bytes per pixel or 1 byte per color, resulting in $0 \times 00 \Rightarrow 0 \times 000000FF$ for white pixels and $0 \times FF \Rightarrow 0 \times FFFFFFFF$ for black pixels.

First-Difference Coprocessor

In total, three hardware accelerators were created for the first-difference Edge Detection Coprocessor, RGB-to-Greyscale converter, Horizontal Edge Detection, and Vertical Edge

Detection. Each of the hardware accelerators were tested via simulation and individual implementation on an FPGA before being integrated with RadPC. Figure 5.12 provides a block diagram of the coprocessor with the different hardware accelerators being swapped during operation, and each accelerator generating a different processed image. In this test, the test image used was the MSU Logo and close observation of the black/white images shows a single pixel shift to the right for the horizontal result and a single pixel shift down for the vertical result. Despite this pixel shift issue, the coprocessor successfully performed multiple operations in series and produced a result where the edges are clearly visible.

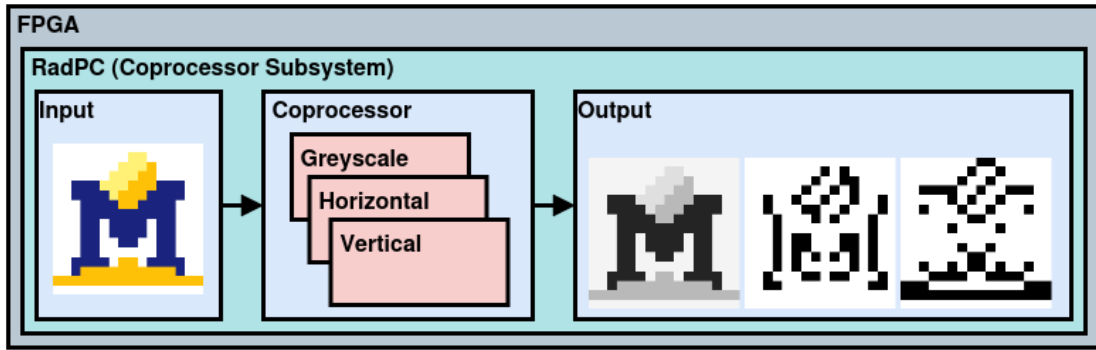


Figure 5.12: first-difference Hardware Accelerators Operations on MSU logo

Using a similar approach to the floating-point coprocessor, the first-difference coprocessor was implemented as PR block in parallel to RadPC with the hardware accelerators being swapped into the coprocessor during operation. Manual PR was the limiting factor for practical use in previous versions of the coprocessor. The previous process required the user to manually initiate a PR in response to RadPC's request for a specific coprocessor. The new process developed in this research uses a on-board microcontroller to receive RadPC's request for a specific coprocessor and automatically initiate a PR to implement the requested hardware onto the FPGA. During operation, the primary latency is data transfer to the coprocessor from RadPC followed by the PR process to swap a new accelerator into the coprocessor.

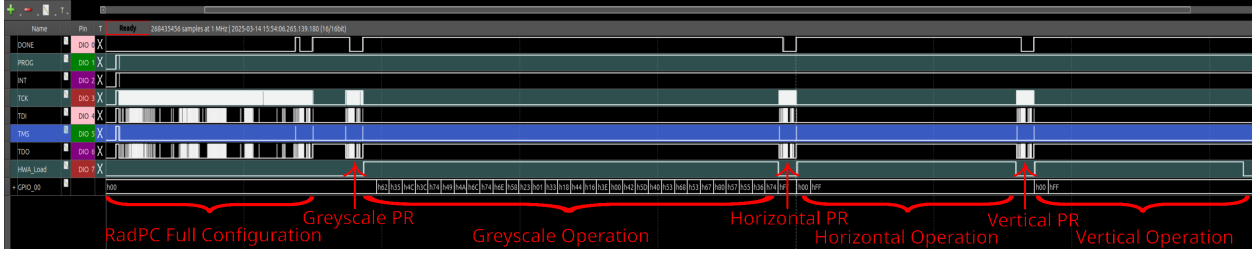


Figure 5.13: RadPC configuration, coprocessor reconfiguration, and image processing

Figure 5.13 shows a rapid series of partial reconfigurations with each coprocessor operating on a 16×16 -pixel image between coprocessor swapping. Three operations were performed in series, greyscale conversion, horizontal first-difference edge detection, and vertical first-difference edge detection. Full configuration of the entire FPGA was performed first to instantiate RadPC. Next, the coprocessor region was PR'd with the RGB-to-Greyscale hardware accelerator. The greyscale output was then used by the horizontal edge detection and vertical edge detection hardware accelerators. Each of these hardware accelerators is swapped into the coprocessor using PR. The overall process of swapping coprocessor functionality and coprocessor image processing captured using a logic analyzer. From this signal capture, configuration and operation times were measured and reported in Table 5.4.

Table 5.4: First-Difference Full Configuration, Partial Reconfiguration, Operation, and Successful/Failure.

Configuration	Success/Fail	Configuration(sec)	Success/Fail	Operation (sec)
RadPC	✓	28.5488	✓	n/a
Greyscale	✓	2.497	✓	60.11
Horizontal	✓	2.497	✓	31.71
Vertical	✓	2.497	✓	30.20

Reconfiguration times for the coprocessor were mainly driven by microcontroller clock frequency (16 MHz), the manual bit level JTAG signaling, and PR region size. The design is relatively slow overall, with a 2.5 s time for swapping. However, the purpose of this design was to demonstrate the concept of hardware flexibility for low power satellites, not speed of configuration. For this purpose, low-speed reconfiguration is acceptable within the design constraints. Operation time is dominated by data transfer in RadPC for reading and writing images to and from memory. Total operation for the program shown in Figure 5.13 was 163.70 seconds for a full configuration, RadPC programming, three coprocessor swaps, and the generation of three processed images of the MSU logo. Three test images were fully processed, an MSU Logo, a Coffee Cup, and a Lunar Crater, and shown in Figure 5.14.

Coprocessor PR was completely successful for all four cases. After PR, coprocessors were individually swapped into the FPGA and RadPC was able to load image data into the coprocessor. The coprocessors were able to generate recognizable images with distinct shapes. Functionality testing on the coprocessor raised some issues with data processing for two out of the five test cases, RGB Conversion and 2D Edge Detection. As shown in Figure 5.14, Greyscale Conversion, Horizontal Edge Detection, and Vertical Edge Detection worked correctly. These three coprocessors each generated a viable 16×16 image with single byte pixels for its result. As an expansion on the work, 2D Edge Detection was also tried and generated a 16×16 image with single byte pixels. However, the output image was not identifiable as a processed version of the input image. The issue was isolated to the pixel shift issue previously mentioned as the images from horizontal and vertical edge detection did not perfectly align. This allows the horizontal and vertical components of 2D Edge Detection to correctly combine all threshold values. A B/W-to-RGB Conversion was tested and successful. However, the current RadPC UART used to pass the data out could not handle the full array for a 16×16 image with 4 byte pixels. Rapidly sending 1024 bytes causes a single bit offset and transmission failure. The RGB Conversion appears to



Figure 5.14: Test images for edge detection coprocessor

be working correctly and has not been verified due to the communication issue. Conversion from single byte pixels to RGB pixels is still performed manually.

First-Difference Power Estimation and Monitoring

Power estimation was performed per logic design components using the Vivado Power Analysis Tool during the synthesis and implementation. This estimates the power requirements for the only FPGA with RadPC and the first-difference edge detection coprocessor implemented. The limitation of this estimation comes from two factors: 1)

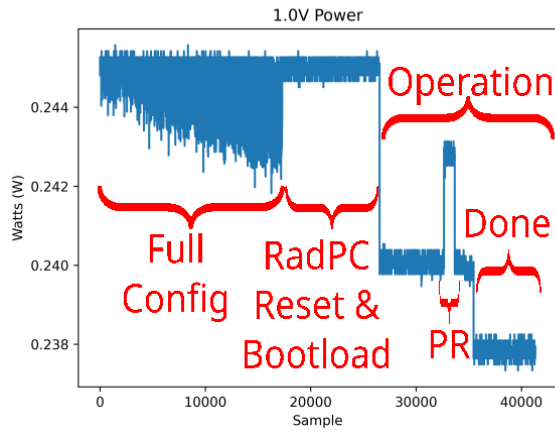
idealized assumptions used by the Power Analysis Tool for the FPGA; 2) only the FPGA's power requirements are estimated. An example of the first limitation is that the FPGA I/O is assumed to draw near zero current during operation. In reality, the FPGA's I/O is driving signals being monitored by the AD3 or driving LEDs. A small power loss occurs in the case of the signal being probed by an external logic analyzer. In the second case, the LED circuit requires power for both the LEDs and the associated resistors. Thus, the actual I/O power consumption is significantly higher on the Hydra board overall. For the individual FPGA components such as the coprocessor, there is also higher power consumption than estimated from losses in the physical circuitry. However, this power consumption estimate does provide a baseline for the overall power consumption because of the main processor (the FPGA) dominating the overall power requirements. The estimated power usage by the FPGA is shown in Table 5.5.

Table 5.5: Estimated FPGA power for components during first-difference operation.

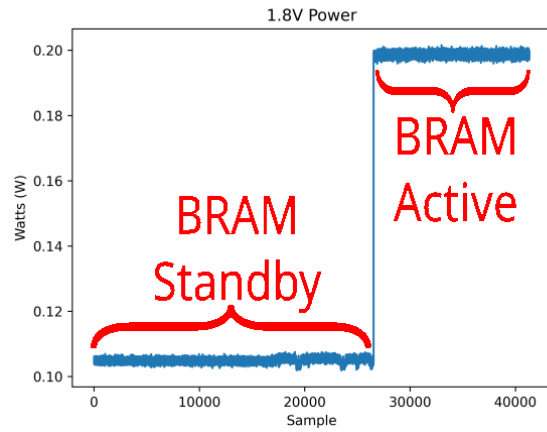
Component	Device	Estimated Power (W)
Total	FPGA	0.414
Implemented Design	FPGA	0.273
Clock	FPGA	0.024
RadPC	FPGA	0.149
Coprocessor	FPGA	0.001
I/O	FPGA	0.001

For actual board measurements, the power consumption is recorded on the board with Hydra's on-board power monitoring system and at the power supply to see the overall power consumption during operation. With the on-board power monitoring system, Hydra monitored voltage and current sensors on the FPGA power rails during a full

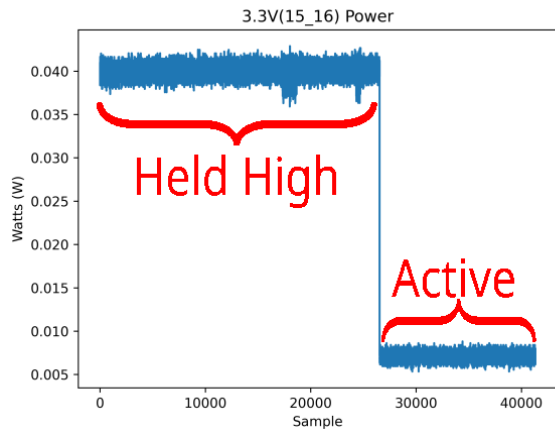
operation sequence, Power On, Startup, Full Configuration with Single Core RadPC, RadPC Reset, RadPC Boot Load, RadPC Operation, Partial Reconfiguration of Coprocessor, and Coprocessor Operation. Graphs of the Hydra's FPGA power during operation are shown in Figure 5.15. The FPGA Core Power, Figure 5.15a, illustrates the whole sequence of operations. BRAM draws power as RadPC is boot loaded and begins operations. FPGA I/O pins, Figure 5.15c, are held high as these pins receive RadPC's clock, reset, and boot loading.



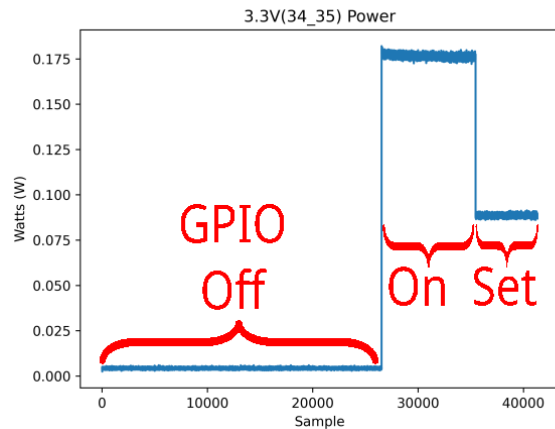
(a) Hydra FPGA Core Power



(b) Hydra FPGA Auxiliary and BRAM Power



(c) Hydra FPGA UART, Clock, & Reset Power



(d) Hydra FPGA GPIO Power

Figure 5.15: Hydra on-board FPGA power for first-difference with manual PR

As the microcontroller was monitoring the power rails and not available for PR, the FPGA was configured and partially reconfigured manually using the Xilinx Platform II Cable. However, the Hydra board power consumption is almost identical between the external FPGA configuration and the on-board configuration. This was verified by monitoring the power supply during the entire sequence of operations with both external configuration and on-board configuration.

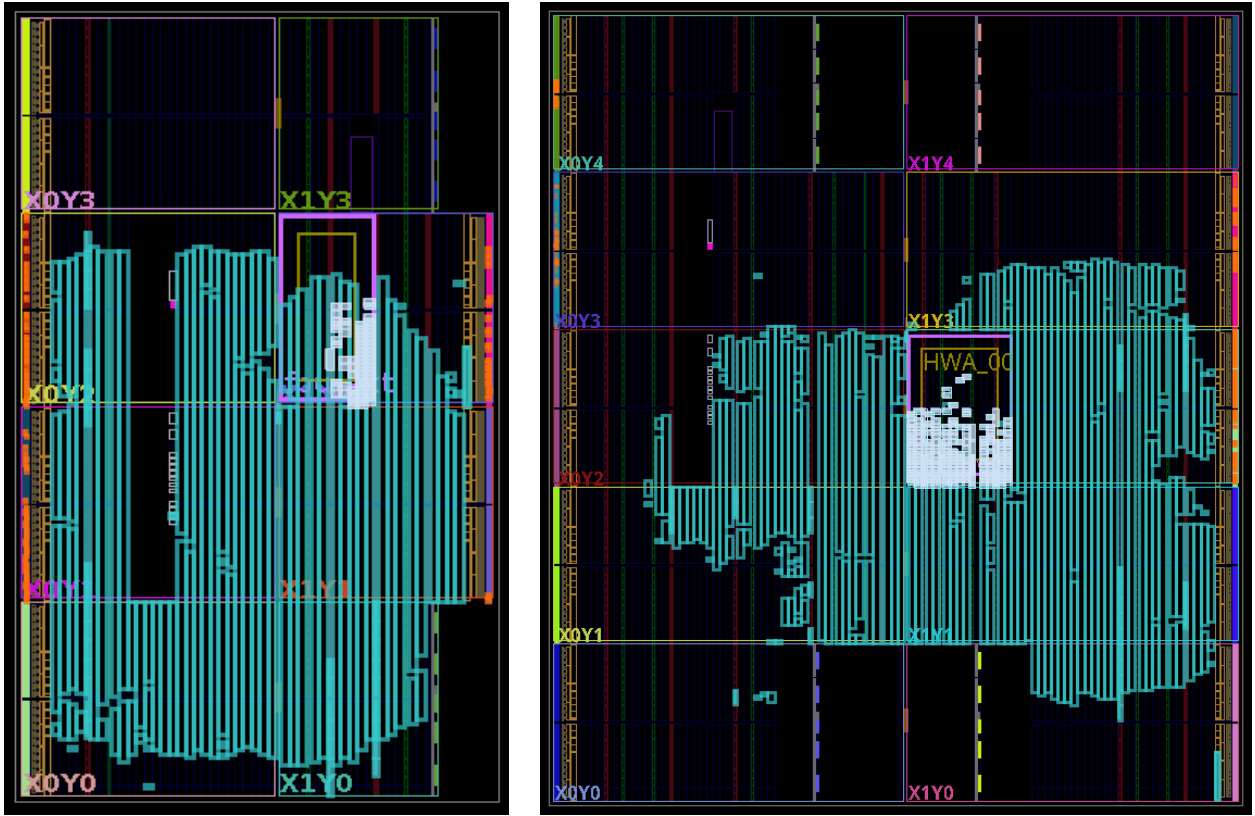
Table 5.6: Measured Hydra board power during first-difference operation.

Operation	Device	Power (W)
Preconfiguration	Hydra	0.995
Full Configuration	Hydra	1.922
RadPC Reset	Hydra	1.473
RadPC Boot Loading	Hydra	1.993
Partial Reconfiguration	Hydra	2.133
Greyscale Operation	Hydra	2.136
Horizontal Edge Detection	Hydra	2.141
Vertical Edge Detection	Hydra	2.137

Total board power consumption during operation was roughly 2.1 W as shown in Table 5.6. This table was created by monitoring the power supply for the entire board and recording power during the same test sequence of Power On, Startup, Full Configuration with Single Core RadPC, RadPC Reset, RadPC Boot Load, RadPC Operation, Partial Reconfiguration of Coprocessor, and Coprocessor Operation. Measured power for the FPGA was higher than estimated at roughly 1 W due to I/O usage by RadPC. The estimated power was accurate for the FPGA, but not the overall circuit with the FPGA I/O driving circuitry.

First-Difference Resource Usage

While RadPC and the first-difference coprocessor were implemented on the larger 200T on the Hydra OBC, instead of the 100T as with RadPC and the floating-point coprocessor, the partial block for the coprocessor was kept roughly the same size as shown in Figure 5.16. Furthermore, the floating-point coprocessor was reimplemented on the 200T of the Hydra. This allowed the FPGA resource usage to be directly compared between different coprocessors.



(a) RadPC+FPU Coprocessor implemented on 100T FPGA

(b) RadPC+Coprocessor (First Diff) implemented on 200T FPGA

Figure 5.16: Comparison of Partial Blocks for Coprocessor on 100T FPGA and 200T FPGA

The power measurements and radiation tolerance measurements for each different coprocessor closely match each other. Monitoring the power during manual external PR

shown the system was capable of < 2.1 W operation while using coprocessors that held true for the floating-point coprocessor as well. Overall, this first-difference edge detection coprocessor was successful in performing on-board swapping of hardware accelerators via dynamic partial reconfiguration. The results of image processing on 16×16 -pixel pictures demonstrated the versatility of the coprocessor concept.

Coprocessor: First-Derivative Edge Detection

The final artifact developed during this research is the first-derivative edge detection coprocessor. This coprocessor is an iteration on the first-difference edge detection coprocessor that addresses the issues of the pixel shift in the results, manual conversion from single bytes to RGBA, and the lack of two-dimensional edge detection. For this coprocessor, four hardware accelerators were developed, a Horizontal first-derivative Edge Detection, a Vertical first-derivative Edge Detection, a Two-Dimensional Edge Detection, and a B/W-to-RGB Conversion. The RGB-to-Greyscale Conversion hardware accelerator from the first-difference coprocessor set of hardware accelerators was reused. The same partial block for FPGA floor planning was reused that makes the power estimates, PR timing, and general operations were similar between coprocessors.

Algorithms

The edge detection algorithm was improved from a first-difference algorithm to a first-derivative algorithm. First-Difference checks between the prior pixel and the current pixels for a difference. If that difference is greater than the threshold, it marks it as an edge. First-Derivative checks the differences between the current pixel, prior pixel, and the next pixel for a difference greater than the threshold for the associated slope of the difference. The results of these two sets of operations are logically OR'ed together. Whether or not the pixel is an edge is determined by the magnitude of the difference between neighboring

pixels and the direction of the slope. This series of operations avoids the issue of pixel shift that the first-difference coprocessor had. Furthermore, these operations correctly assign the edge location within the image provided the image is a dark shape on light background. Horizontal and vertical edge detection was performed using Equation 5.5. Two-dimensional edge detection is the result of Horizontal results being logically OR'ed with Vertical results.

$$\begin{aligned}
 M_{prior} &= |P_{prior} - P_{current}|, & S_{prior} &= \begin{cases} 1; & \text{Positive} \\ 0; & \text{Negative} \end{cases} \\
 M_{next} &= |P_{current} - P_{next}|, & S_{next} &= \begin{cases} 1; & \text{Positive} \\ 0; & \text{Negative} \end{cases} \\
 P_{result} &= \begin{cases} 0 \times FF; & (M_{prior} > Threshold) \wedge (S_{prior} = 0) \\ 0 \times FF; & (M_{next} > Threshold) \wedge (S_{next} = 1) \\ 0 \times 00; & \text{else} \end{cases}
 \end{aligned} \tag{5.5}$$

Equation 5.5 provides the operations performed on a single pixel. On the FPGA, these operations are parallelized and performed on a 4×6 matrix, returning a 4×4 matrix of detected edges. To increase efficiency in processing multiple test images, an additional hardware accelerator was created to convert a black and white single byte pixel representation into RGBA format. Equation 5.6 details the operations used for the B/W-to-RGB conversion hardware accelerator. This accelerator solved the issue of pixels needing to be manually converted from a single byte representation to the RGBA that can be directly appended to a bitmap image header.

$$P_{result} \Rightarrow P_{RGB} = \begin{cases} 0 \times FF & \Rightarrow 0 \times FFFFFFFF \\ 0 \times 00 & \Rightarrow 0 \times 000000FF \end{cases} \quad (5.6)$$

Configuration & Operation Timing

This edge detection algorithm uses a RGB-to-Grayscale conversion, a Vertical edge detection, a new Horizontal edge detection, a 2-Dimensional edge detection, and a B/W-to-RGB conversion as shown in Figure 5.17.

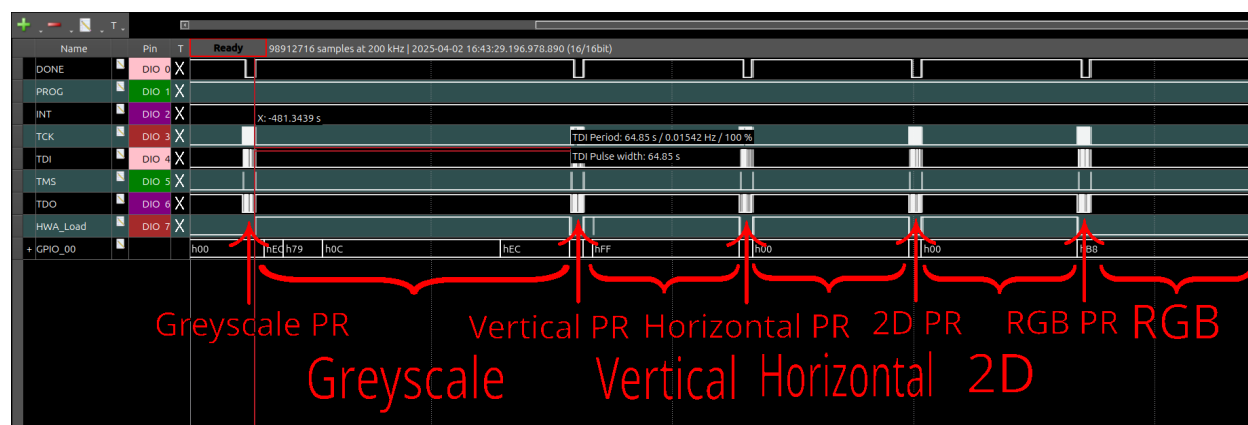


Figure 5.17: first-derivative algorithm processing MSU logo

Figure 5.17 shows the full end to end image processing of the MSU Logo to detect edges vertically, horizontally, and 2-dimensionally. *DONE*, *PROG*, *INT*, *TCK*, *TDI*, *TMS*, and *TDO* are the individual signals of the FPGA's JTAG configuration port and are used to track PR starts and ends. *HWA Load* is a flag from RadPC confirming that it has started writing data to the coprocessor. It also signals that RadPC has finished reading from the coprocessor by lowering the *HWA Load* flag. Finally, *GPIO 00* is the least significant byte of the most significant pixel for the 4x4 block of pixels currently being processed. Image

processing consisted of five operations. First, the RGB-to-Greyscale hardware accelerator was swapped in via PR. Then the 16×16 -pixel images were converted from the RGBA to a single byte per pixel greyscale images. From this greyscale image, the vertical edges were extracted via the operations given in Equation 5.5. Repeating this sequence, the horizontal edges were extracted using the same operations given in Equation 5.5 with the direction of pixel loading rotated 90 degrees. Results from vertical and horizontal accelerators are logically OR'ed in the 2-dimensional accelerator to provide edge detection in both directions. Finally, the black and white single byte representations of pixels are converted by operations shown in Equation 5.6 into RGBA format. The greyscale image, vertical image, horizontal image, and 2-D image are piped out of RadPC via UART and paired with a bitmap header. A wider selection of images was used for testing compared to the previous first-difference coprocessor. In addition to the MSU Logo, Coffee Cup, and Lunar Crater, pictures of the Lunar Surface, the Blue Ghost Landing, one of Jupiter's Moons were also processed.

From Figure 5.17 the timing for both partial configurations and hardware accelerator operations was measured. These times are summarized in Table 5.7 and were verified by repeated measurements during operations with different 16×16 -pixel images. Results were very consistent with variations on the order of $0.1 \mu s$ for PR and no measured variation for successful operations.

The PR and operation times from Table 5.7 are identical to first-difference algorithm times list in Table 5.4 as the same PR block on the FPGA was used. Greyscale and RGB are the RGB-to-Greyscale and B/W-to-RGB converters respectively, and their operation times are double that of the edge detection operations because of the formatting of the data. Both of these conversions are handling RGBA data either as input or output that limits them to processing 8×4 byte pixels in parallel as opposed to the edge detection operations that process 16×1 byte pixels in parallel. The edge detection hardware accelerator was significantly faster as more pixels were processed in parallel. However, there was more

Table 5.7: First-Derivative Full Configuration, Partial Reconfiguration, Operation, and Successful/Failure.

Configuration	Success/Fail	Configuration(sec)	Success/Fail	Operation (sec)
RadPC	✓	28.5488	✓	n/a
Greyscale	✓	2.497	✓	60.11
Horizontal	✓	2.497	✓	31.71
Vertical	✓	2.497	✓	30.20
2Dimensional	✓	2.497	✓	30.15
RGB	✓	2.497	✓	60.15

variation in the timing of operations between the Vertical, Horizontal, and 2-D hardware accelerators. All these accelerators are combinational logic designs and the variation is from the different path lengths on the FPGA for each implementation.

Image Processing

A number of images were generated using the first-derivative algorithm. For this algorithm, the following coprocessors were used in sequence, Greyscale, Vertical, Horizontal, OR, and RGB. Two series of test images were used. The first series of test images from the first-difference coprocessor testing was used. These test images were the MSU Logo, Coffee Cup, and Lunar Crater are shown in Figure 5.18. This allowed for direct one-to-one comparison of the generated images to prove the pixel shift issue was solved with the new first-derivative coprocessor. The second series of test images were to examine the limitations of this edge detection algorithm for light objects on black background or the need for blurring.

The images from the first-derivative coprocessor demonstrate a successful solution to the issues with the first-difference coprocessor. For Vertical and Horizontal images, seen in Figure 5.18d and Figure 5.18e, there is no pixel shift in the direction of image processing.

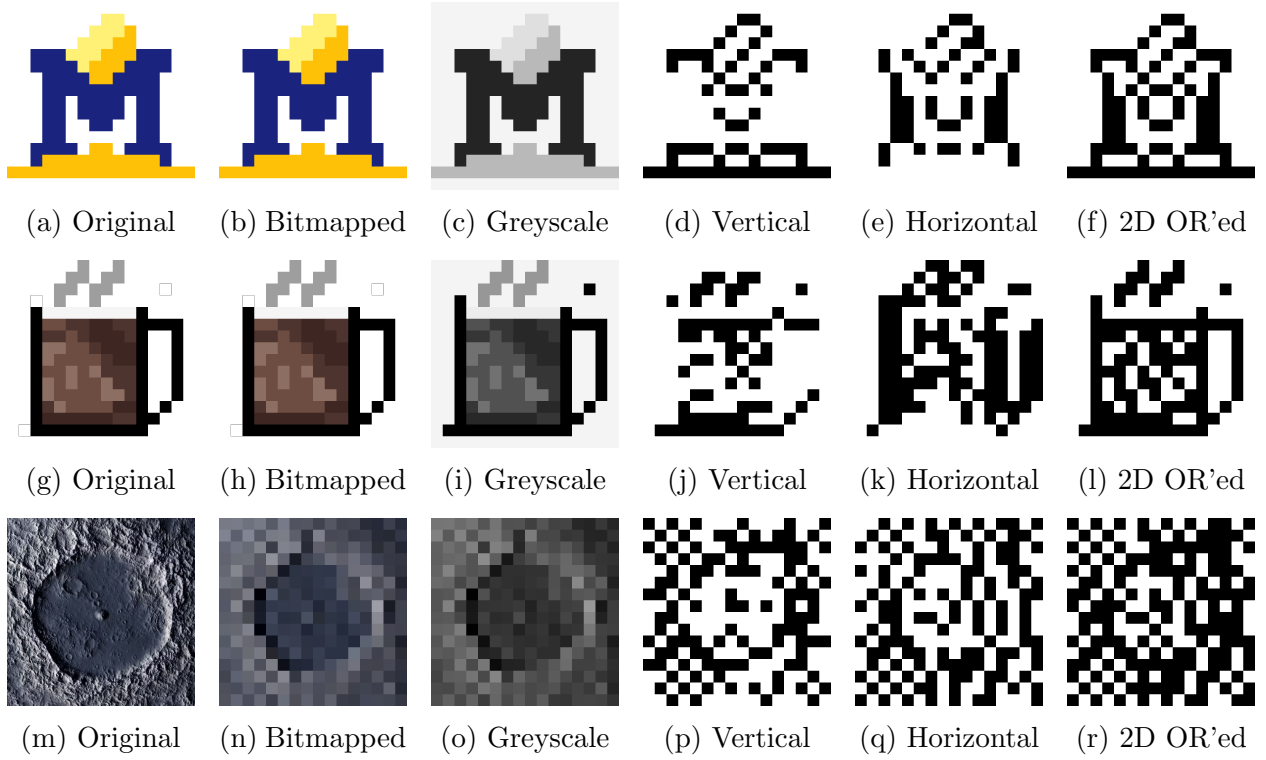


Figure 5.18: Standard set of test images for first-derivative edge detection coprocessor

The combination of these two into the 2D edge detection in Figure 5.18i resulted in a clean image with the MSU Logo clearly visible and outlined. The coprocessor did have issues with noisy images where the overall picture had a relatively low difference for the edge. As apparent in Figures 5.18p, 5.18q, and 5.18r, this produces an image where the edges are not clearly defined. Solutions to this issue are automatic tuning of the threshold based on the difference between the darkest and lightest pixel values or the addition of a sixth hardware accelerator to perform a Gaussian blur on the image before edge detection. Both of these solutions were outside the scope as this research is focused on dynamic partial reconfiguration of a coprocessor in parallel with RadPC, not the development of a perfected edge detection algorithm for implementation on an FPGA.

Four supplementary images were tested to provide a wide range of inputs and prove

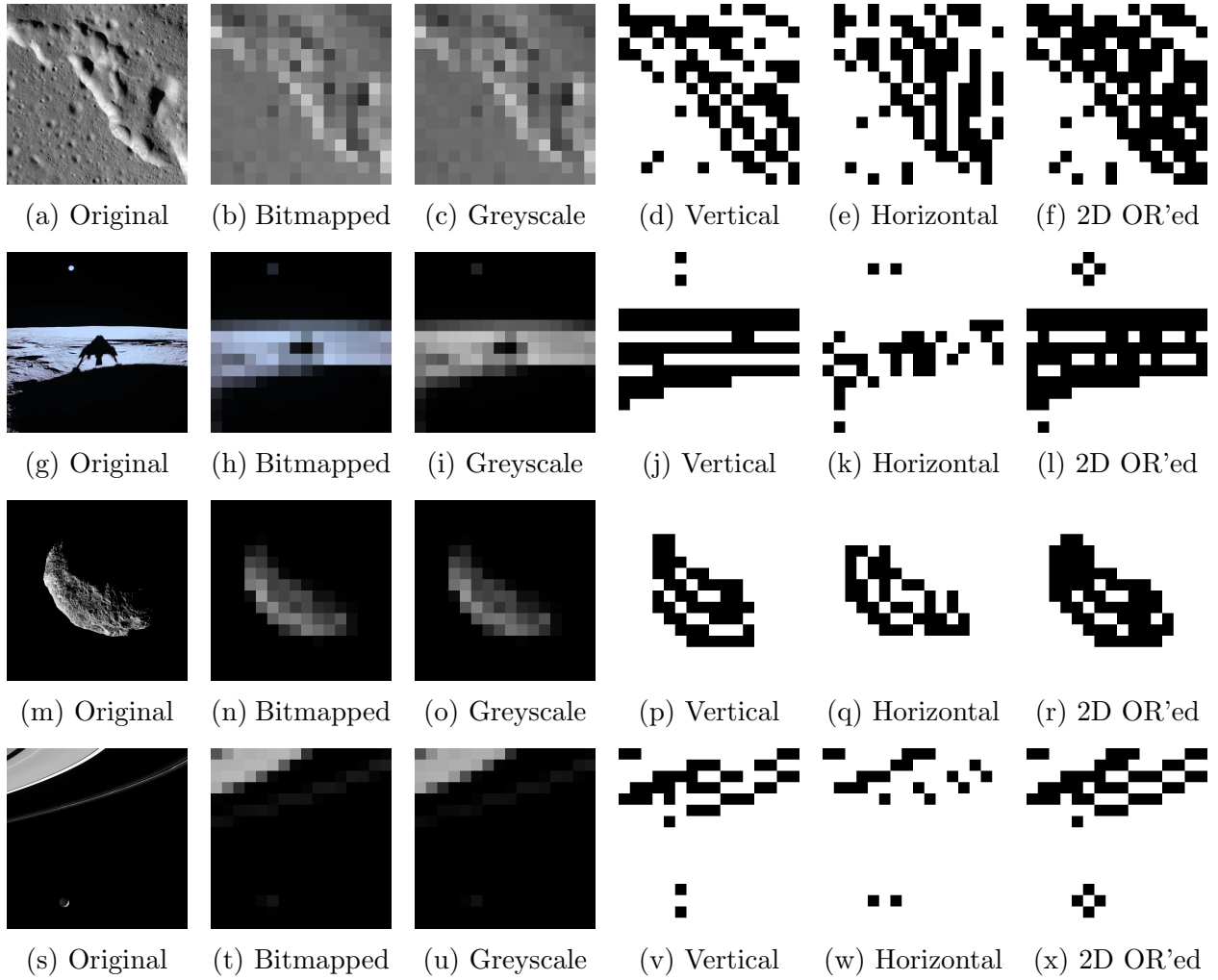


Figure 5.19: Test images for first-derivative edge detection coprocessor

the repeatability of the dynamic partial reconfiguration process over a total of 7 images and 35 runs. In all cases, the swapping between hardware accelerators was successful and the coprocessor output a total of 4 images per test image as shown in Figure 5.19. For these images, a few issues were identified with the coprocessor setup. First, as previously noted, the first-derivative edge detection is more successful on an image with a light background and dark foreground as seen by comparing Figures 5.19a and Figure 5.19f. As opposed to Figures 5.19s and 5.19x with a dark background and light objects where the algorithm

gives edge detection on the pixel beside the actual edge. Second, lighting greatly affects the results. This algorithm works effectively on lunar landscapes where the camera is pointing perpendicular to the surface, and the light source is approximately 15 degrees above parallel to the surface. Third, as seen in Figures 5.19g and 5.19j, images with strong shadows result in the edge detection being dominated by shadows.

Limitations

dynamic partial reconfiguration is a compromise between used FPGA resources and operation time. This imposes limitations, shown in Table 5.8, on the capability of the system.

Table 5.8: Current Limitations and Proposed Solutions

Item	Current	Future Upgrade
PR Speed	2.497 seconds	PR code optimization
PR Request Port	Parallel	Serial
Coprocessor ID	None	Internal Register with ID
PR Verification	None	Acknowledgement, ID, Test Code
Coprocessor Interrupt Register	None	Interrupts in RadPC
Coprocessor Control Register	None	Control register in RadPC

Total PR time for the coprocessors was 2.497 s that is a long delay for a 16 MHz system. The PR code optimization should bring down the PR time. As a reference, the Xilinx Platform II Cable at 6 MHz achieved 250 ms for the same PR Block. Even with the manual bit-level signaling, the total PR time for the PR Block should be reduced to roughly 750 ms. Optimization of the PR code combined with coprocessor preloading in the background can reduce the 2.497 s time. The parallel port between RadPC and the microcontroller limits debug capability and the ability to support multiple coprocessors at

once. Each coprocessor needs its own ID number for requesting PR. Currently, the parallel port only supports $2^3 = 8$ different coprocessors at once. Shifting to a serial port will allow the swapping request to be turned into a packet transmission and will support $2^8 = 256$ coprocessors simultaneously. RadPC needs a method of internally identifying coprocessors, and a test procedure for verifying the coprocessor's PR was successful. Interrupts added to RadPC for the coprocessor would allow RadPC to interrupt its program to service the coprocessor on operation completion. A standardized control register in RadPC would provide a dedicated method for controlling coprocessor operation, particularly given the wide variety of coprocessors that the system already supports.

Reliability and Environment Analysis

In space application, reliability focuses on the probability that a system operates as designed for a specific period dependent on the nature of the mission and function of the system [198]. Reliability analysis for the coprocessor uses Markov chain models to estimate the system's reliability with the likelihood of induced faults and timing of recovery methods. This approach is similar to the Markov chain models used to estimate reliability for RadPC [191, 198, 199]. The Markov model uses a state diagram to describe a system's transitions between modes of operation.

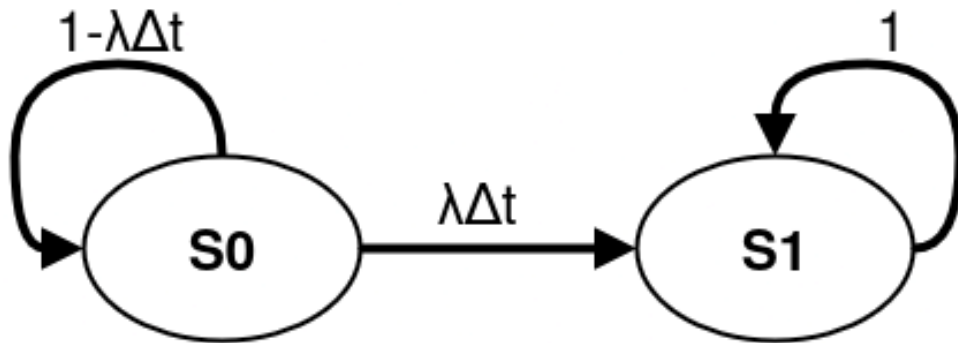


Figure 5.20: Diagram of the Markov model for a simple system

The diagram for a Markov simple system with two states S_0 and S_1 is shown in Figure 5.20. State S_0 is the operational state of the system, and S_1 is the faulted state of the system. The probability for fault rate is algebraically represented by λ for period Δt . The transition matrix T for the simple two state system from Figure 5.20 is given in Equation 5.7

$$T_{simple} = \begin{bmatrix} 1 - \lambda\Delta t & \lambda\Delta t \\ 0 & 1 \end{bmatrix} \quad (5.7)$$

From this matrix, a matrix dot multiplication is used to calculate the probability p of existing in a state S given a time step k as shown in Equation 5.8

$$p_s(t = k\Delta t) = p_{simple} \cdot T_{simple}^k = \begin{bmatrix} p_0(0) & p_1(0) \end{bmatrix} \begin{bmatrix} 1 - \lambda\Delta t & \lambda\Delta t \\ 0 & 1 \end{bmatrix}^k \quad (5.8)$$

Reliability is the probability of residing in a state other than the faulted state S and is given by Equation 5.9. Mean time to failure (MTTF) is also defined in Equation 5.10 as the time t when $R(t) = 0.5$. This is a common metric to evaluate system performance in terms of time until loss of functionality.

$$R(t) = 1 - ps(t = k\Delta t) \quad (5.9)$$

$$R(t)_{MTTF} = 1 - ps(t = k\Delta t) = 0.5 \quad (5.10)$$

Markov Modeling for Coprocessor For the coprocessor, the Markov model represents a radiation strike causing a SEE in either FPGA configuration memory or the implemented logic circuit on the FPGA. This approach limits SEEs to only those that cause faults in the coprocessor; therefore the fault rate is scaled relative to the probability of a strike affecting a critical region inside the coprocessor region on the FPGA. Two repair methods have

been implemented on the coprocessor with both occurring through the FPGA configuration memory. These are modeled in Figure 5.21 with the fastest method generally repairing the system first. However, both repair methods will be computed and plotted for comparison.

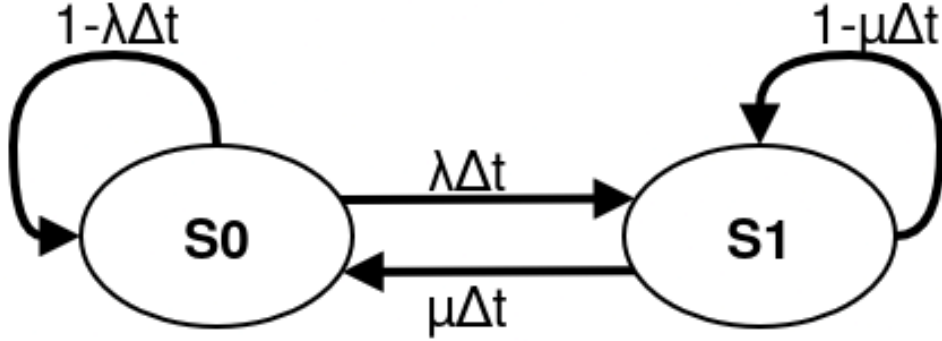


Figure 5.21: Markov Model of Coprocessor System

This model gives the transition matrix T for a simple two state system with repair method from Figure in Equation 5.11.

$$T_{repair} = \begin{bmatrix} 1 - \lambda\Delta t & \lambda\Delta t \\ \mu\Delta t & 1 - \mu\Delta t \end{bmatrix} \quad (5.11)$$

The variable μ is dependent on the repair rate and is calculated using Equation 5.12.

$$\mu = \frac{1}{RepairTime} \quad (5.12)$$

Xilinx's Single Event Mitigation (SEM) system is running in the background. The SEM walks through the configuration memory and checks ECCs for individual blocks of memory, giving a μ_{SEM} for the time required to check all locations in configuration memory. This is working from the worse case, where the SEM is at address n , a fault occurs at address $n - 1$, the memory is of size N , and the SEM restarts at address 0 upon reaching the end of

memory. In this case, the SEM checks $N - n - (n - 1) = N - 1$ addresses before finding the fault. Equations 5.13 and 5.14 gives the scan time and error correction time for the target device [200].

$$ScanTime_{ACTUAL} = ScanTime_{ICAP_{FMAX}} \cdot \frac{ICAP_{FMAX}}{Frequency_{ACTUAL}} \quad (5.13)$$

$$CorrectionLatency_{ACTUAL} = CorrectionLatency_{FMAX} \cdot \frac{ICAP_{FMAX}}{Frequency_{ACTUAL}} \quad (5.14)$$

For the target device of Atrix-7 XC7A200T, the scan time plus the error correction time is the total recovery time. The worst case of recovery time will be assumed. This is a 2-bit correctable fault in the configuration memory [200]. Equation 5.15 gives the total scan and repair time for the longest possible case.

$$\begin{aligned} & ScanTime_{ACTUAL} + CorrectionLatency_{ACTUAL} = \\ & (18.3ms) \cdot \frac{100MHz}{32MHz} + (26900\frac{\mu}{s}) \cdot \frac{100MHz}{32MHz} = \\ & 141.25ms \end{aligned} \quad (5.15)$$

From this recovery time, the μ is calculated using Equation 5.16.

$$\mu_{sem} = \frac{1}{ScanTime_{ACTUAL} + CorrectionLatency_{ACTUAL}} = \frac{1}{141.25ms} = 7.0796\frac{1}{s} \approx 7.08\frac{1}{s} \quad (5.16)$$

The second repair method is the PR that is performed once the coprocessor operation is completed. For the PR repair method, the longest edge detection operation + partial

reconfiguration time will provide μ_{PR} since this is the worse-case for recovery time. The best case for recovery times will be considered using the shortest operational time from the floating-point coprocessor operation. From Table 5.7, this gives Equation 5.17 for edge detection and Equation 5.18 for floating-point.

$$\mu_{pr_{edge}} = \frac{1}{60.15s + 2.49s} = \frac{1}{62.640s} = 0.01596424 \frac{1}{ms} \approx 0.016 \frac{1}{s} \quad (5.17)$$

$$\mu_{pr_{fpu}} = \frac{1}{10.68ms + 2.49s} = \frac{1}{2.50068s} = 0.39989123 \frac{1}{ms} \approx 399.89 \frac{1}{s} \quad (5.18)$$

These three μ will be considered to model the three different cases of a repair by SEM (worst case), repair by partial reconfiguration of the FPU (best case), and repair by partial reconfiguration for the edge detection (worst case). The SEM (best case) is not considered as this case would be an immediate detection and repair of the fault before it would probably affect the implemented logic design.

Radiation Environment for Coprocessor Radiation effects are highly dependent on orbit, time, device orientation, radiation mitigation techniques, etc. While a radiation testing regime is planned for the developed system, it will not be completed by the end of this research project. Therefore, a preliminary analysis of the fault types and rates will be performed to provide some data on the radiation tolerance of RadPC with the coprocessor subsystem implemented. Building off previous modeling of the RadPC computer architecture, the numerical modeling tool Cosmic Ray Effects on Micro-Electronics (CREME96) will be used [201]. Modeling was performed for the ISS Space Radiation Environment based on previous modeling of RadPC in this environment [199]. Scaling from a 35T to a 200T FPGA size increases the fault rate λ significantly from $\lambda = 0.00019$ for 35T to $\lambda = 0.001235$ for 200T.

Reliability Analysis for Coprocessor Shown in Figure 5.22, the Markov model focuses on four cases. These cases are evaluated for 60 seconds of operations as the longest operation during coprocessor operation was 60.11 s. All modelling is done with the assumption that only the coprocessor section of the FPGA configuration memory is vulnerable to radiation induced faults. This isolates the coprocessor from RadPC as RadPC's repair techniques will not operate on the coprocessor. Furthermore, it is assumed that all bits in the FPGA configuration memory for the coprocessor component will affect the implemented logic design. Since the coprocessor component uses 11.35% of the total bits for RadPC & Coprocessor, the fault rate λ will be scaled by 0.1135 to reflect the reduced surface area, allowing the Markov model to only consider the coprocessor's repair techniques.

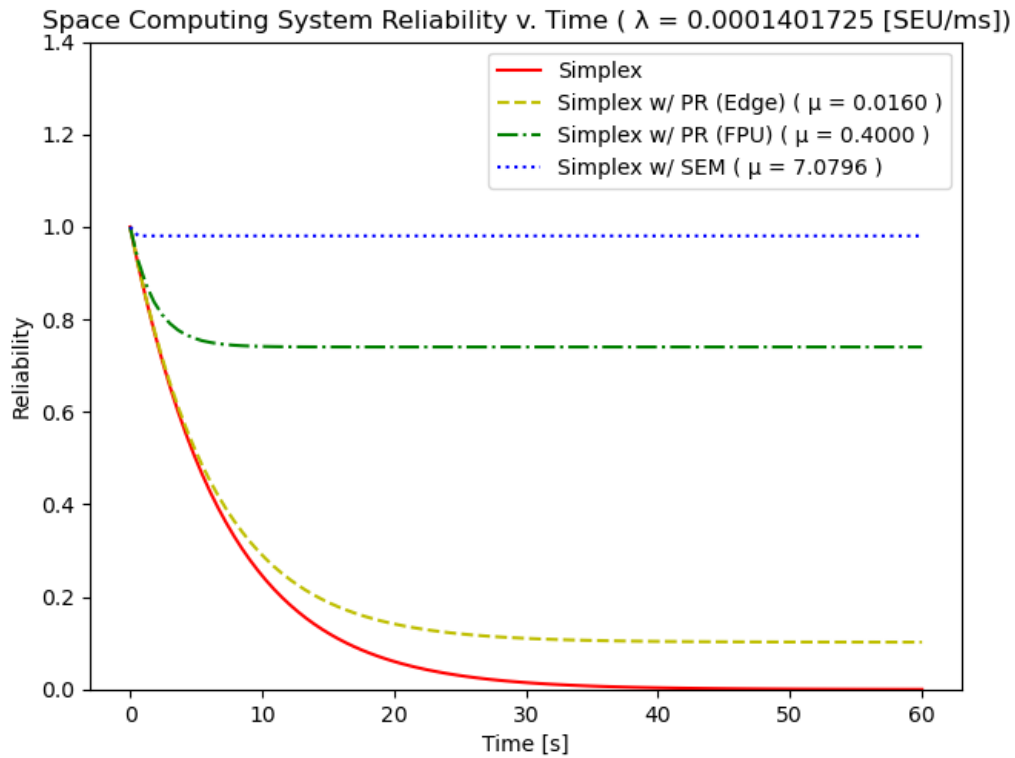


Figure 5.22: Markov Modeling of Coprocessor Reliability for Simplex System, Simplex + PR, Simplex + SEM for 60 s

Based on comparison between the Simplex system without any repair and the Simplex + PR systems, shown for both the longest coprocessor operation and the shortest coprocessor operation, PR is an effective radiation mitigation technique depending on the time between PRs. For the longest coprocessor operation (BW-to-RGB), the Simplex w/ PR (Edge) drops below the MTTF roughly 6 s into operation and the system has an equal chance of failure vs success for the following 54 s of operation. Comparatively, the Simplex w/ PR (FPU) completes its operation and performs PR quickly enough that it stays above 70% reliability. The SEM repair technique is the most effective overall as shown in Figure 5.23 for the 1 hr test.

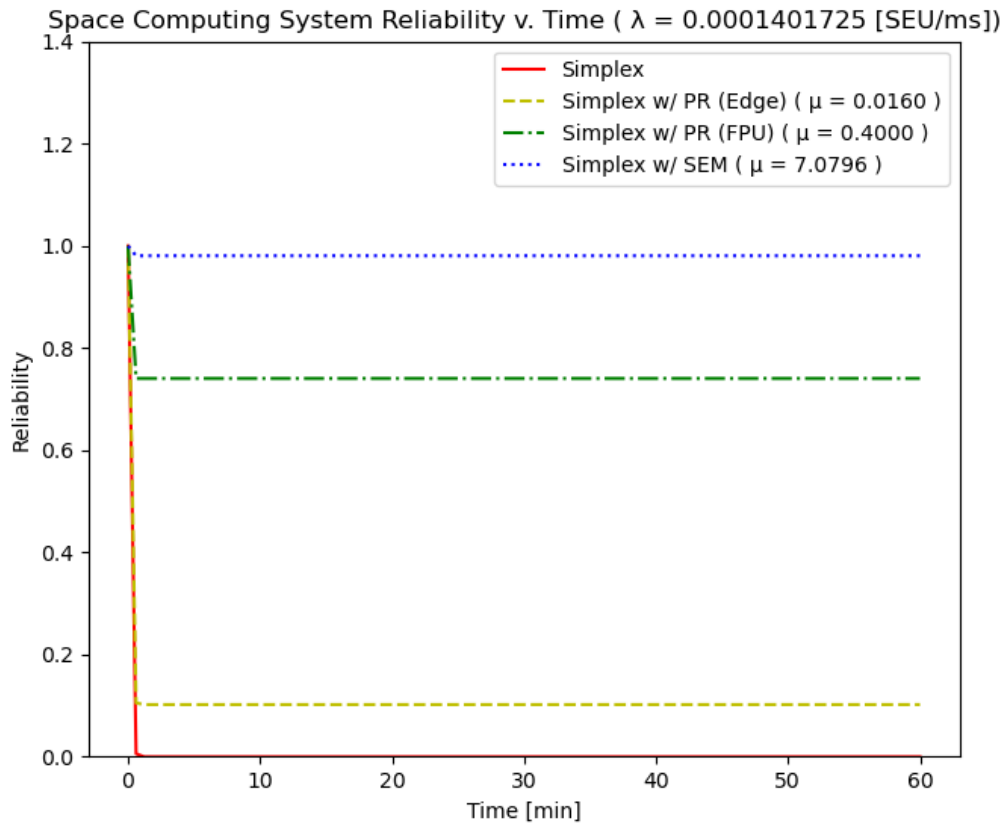


Figure 5.23: Markov Modeling of Coprocessor Reliability for Simplex System, Simplex + PR, Simplex + SEM for 1 hr

In conclusion, the SEM repair technique is effective at mitigating faults from a radiation strike in the FPGA configuration memory for several orders of magnitude longer than the coprocessor is operational. PR is an effective recovery technique. However, it is highly dependent on the time between swapping or refreshing the coprocessor. Given the time to MTTF while waiting for coprocessor operation, PR by itself is not a reliable recovery technique. The combination of PR and SEM presents a highly reliable system with MTTF acceptable for the majority of small satellite missions in LEO, cis-lunar, and lunar environments.

SCORING, COMPARISON, & DISSEMINATION

To provide a relative measure for the success of the RadPC + Coprocessor concept, the finalized system will be directly compared to six examples of small satellite onboard computers and two examples of high-end traditional satellite computers. This comparison will place the developed system within the current small satellite onboard computer systems landscape. The comparison to the high-end computer will provide a reference for the performance and cost tradeoff for small satellites relative to traditional satellites.

Scoring

The six examples selected for this comparison are Flying Laptop, NanoMind A3200, Nexus One, EnduroSat OBC Type I, ISIS iOBC, AstroDev MarCO C&DH. These will be scored, and then the scores will be directly compared to the RadPC & Coprocessor's scoring. Further comparisons will be made between RadPC with the RAD750 and the HPSC. These systems are not full onboard computers, but rather radiation hardened processors that form the main processing node for an onboard computer. This discrepancy between comparing processors vs onboard computers is due to the lack of full information for the RAD750 and HPSC. As high-end radiation hardened processors, there is information on the processors themselves but not the systems as these are generally customized and designed for government programs. HPSC is also a new radiation hardened processor and has not yet been fully released. Therefore, the most complete comparison possible at this time is a comparison of the radiation hardened processors to the small satellite onboard computers.

Computational Performance Scoring

An onboard computer's computation performance shall be scored based on Table 4.1 and Equation 4.1. The primary factors under consideration will be capabilities and speed.

Three types of operations will be considered, integer operations, floating-point operations, and vector processing operations. Integer operations are the operation used by most small satellite computers for system control via setting registers and internal communication packets. Floating-point operations are generally used for navigation, position calculation, and data processing. With the enormous increase in small satellites with cameras, vector processing has become a necessary capability for small satellites. Holding all instrumentation data requires a significant amount of memory. The size of the memory will be considered for scoring. Finally, the clock speed has a major effect on the processing speed for these operations and data transfer from memory and will also be ranked for scoring. Scoring will be performed using Equation 6.1.

$$C_{comp} = K_{clk} + K_{int} + K_{flt} + K_{img} + K_{mem}, 0 \leq K_x \leq 0.2 \quad (6.1)$$

Flying Laptop Scoring Flying Laptop does not have a publicly documented CPU clock speed for the UT699 softcore implemented on it. However, the UT699 datasheet give 66 MHz as the default frequency, and this will be assumed to be the operating frequency as the documentation does not list modifications performed on the UT699 [163]. From this, the Flying Laptop clock score is calculated in Equation 6.2 using Equation 4.2.

$$K_{clk_{laptop}} = 0.2 * \frac{\log_{10}(66) - \log_{10}(1)}{\log_{10}(1000) - \log_{10}(1)} = 0.121302929 \approx 0.121 \quad (6.2)$$

The UT699 provides integer support natively inside its computer architecture and floating-point operations support via peripherals[163]. A peripheral block for an IEEE-754 floating-point unit is implemented to support floating-point operations [163]. While the system does not provide dedicated hardware for vector processing, it does have the capability to perform vector processing on the CPU. Given the assumed clock speed of 66 MHz, the CPU would be capable of relatively fast vector processing at the expense of not being able

to engage in other operations. This gives a vector processing score of $K_{vecLaptop} = 0.075$. Memory is set up by default to allow a max of 512 MB PROM, 1 GB SRAM, and 1 GB SDRAM, which would support image support for later processing.

$$K_{memLaptop} = 0.2 * \frac{\log_{10}(2.512GB) - \log_{10}(10kB)}{\log_{10}(64GB) - \log_{10}(10kB)} = 0.158679895 \approx 0.159 \quad (6.3)$$

The Flying Laptop's overall score is calculated from the individual scores using Equation 4.1 in Equation 6.4.

$$C_{FlyingLaptop} = 0.159 + 0.200 + 0.200 + 0.075 + 0.200 = 0.834 \quad (6.4)$$

GomSpace Express-3 Scoring GomSpace Express-3 uses the NanoMind A3200 operating with a clock between 8 MHz to 64 MHz. For this scoring, the 64 MHz clock will be assumed and the GomSpace Express-3 clock score is calculated in Equation 6.5 using Equation 4.2.

$$K_{clkGomSpace} = 0.2 * \frac{\log_{10}(64) - \log_{10}(1)}{\log_{10}(1000) - \log_{10}(1)} = 0.120411998 \approx 0.120 \quad (6.5)$$

Integer operations are natively supported by the computer architecture, giving an integer processing score of $K_{intGomSpace} = 0.200$. An IEEE 754 FPU is implemented on the NanoMind to support floating-point operations, giving a score of $K_{fltGomSpace} = 0.200$. Vector processing is not supported with dedicated hardware. NanoMind A3200 is capable of performing vector processing at the software level using the CPU as the processing node, giving a vector processing score of $K_{vecGomSpace} = 0.075$. Memory size is decent for a small satellite computer, but not exceptional memory with 128 MB NOR flash, 32 KB FRAM for persistent configuration storage, and 32 MB SDRAM. These memory sizes are used in Equation 6.6 to calculate the memory score for the Express-3

$$K_{mem_{GomSpace}} = 0.2 * \frac{\log_{10}(192MB) - \log_{10}(10kB)}{\log_{10}(64GB) - \log_{10}(10kB)} = 0.125865059 \approx 0.126 \quad (6.6)$$

The GomSpace Express-3's overall score is calculated from the individual scores using Equation 4.1 in Equation 6.7.

$$C_{GomSpace} = 0.120 + 0.200 + 0.200 + 0.075 + 0.126 = 0.721 \quad (6.7)$$

Nexus One Scoring Nexus One has a very fast clock with Qualcomm Snapdragon S1 Processor running at 1 GHz. The Nexus One clock score is calculated in Equation 6.8 using Equation 4.2.

$$K_{clk_{nexus}} = 0.2 * \frac{\log_{10}(1000) - \log_{10}(1)}{\log_{10}(1000) - \log_{10}(1)} = 1.000 \quad (6.8)$$

The Nexus One's Snapdragon S1 Processor is inherently capable of integer processing, giving a score of $K_{int_{Nexus}} = 0.200$. It does not have any hardware level floating-point capability and relies on software emulation for any floating-point operation, giving it a floating-point processing score of $K_{flt_{Nexus}} = 0.000$. It does support basic vector processing to the extent of supporting the built-in camera with a resolution up to 12 megapixels with the memory size to store multiple images either internally or on its 4 GB SD as shown in Table 2.2. However, the Snapdragon S1 Processor does not have any dedicated image or signal processing capability. This gives a low score of $K_{vec_{Nexus}} = 0.075$ for vector processing as the main processor would be required to perform the processing in software only without optimization. Nexus One's memory score is calculated in Equation 6.9 using Equation 4.4

$$K_{mem_{nexus}} = 0.2 * \frac{\log_{10}(4GB) - \log_{10}(10kB)}{\log_{10}(64GB) - \log_{10}(10kB)} = 0.164616863 \approx 0.165 \quad (6.9)$$

The Nexus One's overall score is calculated from the individual scores using Equation 4.1 in Equation 6.10.

$$C_{Nexus} = 0.200 + 0.200 + 0.000 + 0.075 + 0.165 = 0.640 \quad (6.10)$$

EnduroSat OBC Scoring EnduroSat OBC Type I uses an ARM Cortex M4 as its main processor, running at a clock frequency of 180 MHz. From this, the EnduroSat OBC clock score is calculated in Equation 6.11 using Equation 4.2.

$$K_{clk_{Enduro}} = 0.2 * \frac{\log_{10}(180) - \log_{10}(1)}{\log_{10}(1000) - \log_{10}(1)} = 0.1503515 \approx 0.150 \quad (6.11)$$

Integer operations are naturally supported by the computer architecture, giving a score of $K_{int_{EnduroSat}} = 0.200$. The Cortex M4 features a hardware FPU for single precision operations and is compliant with the IEEE 754 Standard [153]. This gives a score of $K_{flt_{EnduroSat}} = 0.200$ for float-point processing. For vector processing, the Cortex M4 does implement digital signal processing instructions like the single instruction multiple data (SIMD). It does not have dedicated vector processing hardware, so these instructions are implemented on the main processor. This gives a very low score of $K_{vec_{EnduroSat}} = 0.075$. For memory, the ARM Cortex M4 has 256 KB RAM, 2 MB Flash Memory, and 256 Mbit Serial NOR Flash Memory [153]. From this memory size, the score is calculated in Equation 6.12 using Equation 4.4.

$$K_{mem_{EnduroSat}} = 0.2 * \frac{\log_{10}(34.256MB) - \log_{10}(10kB)}{\log_{10}(64GB) - \log_{10}(10kB)} = 0.103868445 \approx 0.104 \quad (6.12)$$

The individual component scores for the computational performance of EnduroSat OBC are combined in Equation 6.13.

$$C_{EnduroSat} = 0.104 + 0.200 + 0.200 + 0.075 + 0.120 = 0.699 \quad (6.13)$$

iOBC Scoring The ISIS iOBC is centered on a 32-bit ARM9 CPU running at 400 MHz with 32 MB SDRAM and 2x2 GB SD-card. From this, the ISIS iOBC clock score is calculated in Equation 6.14 using Equation 4.2.

$$K_{clk_{isis}} = 0.2 * \frac{\log_{10}(400) - \log_{10}(1)}{\log_{10}(1000) - \log_{10}(1)} = 0.173470666 \approx 0.174 \quad (6.14)$$

Integer operation is fully supported by the computer architecture, giving a score of $K_{int_{ISIS}} = 0.200$. Floating-point support either single or double precision is not listed in the ISIS iOBC datasheet and is assumed not to be supported [156]. This unfortunately gives a score of $K_{flt_{ISIS}} = 0.000$ for floating-point processing. Vector processing is not supported by dedicated hardware, although the processor does support writing code for vector processing at the software level. This gives a very low score of $K_{vec_{ISIS}} = 0.075$. Memory is substantial with 32 MB SDRAM and 2x2 GB SD-card. The memory score is calculated using Equation 6.15.

$$K_{mem_{ISIS}} = 0.2 * \frac{\log_{10}(4.032GB) - \log_{10}(10kB)}{\log_{10}(64GB) - \log_{10}(10kB)} = 0.164718551 \approx 0.165 \quad (6.15)$$

These individual scores are combined in Equation 6.16.

$$C_{ISIS} = 0.174 + 0.200 + 0.000 + 0.075 + 0.165 = 0.614 \quad (6.16)$$

MarCO C&DH Scoring AstroDev MarCO C&DH uses a MSP430F2618 microcontroller as its processor. This microcontroller is running at 12 MHz. The MarCo clock score is calculated in Equation 6.17 using Equation 4.2.

$$K_{clk_{marco}} = 0.2 * \frac{\log_{10}(12) - \log_{10}(1)}{\log_{10}(1000) - \log_{10}(1)} = 0.071945416 \approx 0.072 \quad (6.17)$$

It fully supports integer and fixed-point operations, giving a score of $K_{int_{MarCO}} = 0.200$. Floating-point operations are not supported, giving a score of $K_{flt_{MarCO}} = 0.000$. Texas Instruments does provide methods to convert a floating-point algorithm into fixed-point code. However, this is method for using the fixed-point capability, not having floating-point capability. Vector processing is limited without any dedicated hardware. The processor itself is capable of performing vector processing at the cost of being unavailable for other operations, giving a $K_{vec_{MarCO}} = 0.000$. This version of the microcontroller provides 116 KB of flash memory and 8 KB of RAM giving it an overall low score for memory as shown in Equation 6.18.

$$K_{mem_{MarCO}} = 0.2 * \frac{\log_{10}(124kB) - \log_{10}(10kB)}{\log_{10}(64GB) - \log_{10}(10kB)} = 0.032130261 \approx 0.032 \quad (6.18)$$

These individual scores for the computational performance of MarCO are combined in Equation 6.19.

$$C_{MarCO} = 0.072 + 0.200 + 0.000 + 0.075 + 0.032 = 0.379 \quad (6.19)$$

RadPC Scoring RadPC with coprocessor is a 32-bit RISC-V CPU running at 32 MHz. RadPC's clock score is calculated in Equation 6.20 using Equation 4.2.

$$K_{clk_{radpc}} = 0.2 * \frac{\log_{10}(32) - \log_{10}(1)}{\log_{10}(1000) - \log_{10}(1)} = 0.100343332 \approx 0.100 \quad (6.20)$$

Integer operations are supported by the computer architecture except for multiplication that requires an expansion of the arithmetic and logic unit. RadPC's score lost 0.02 from

a perfect score because multiplication was not supported, giving a score of $K_{intRadPC} = 0.180$. Floating-point operations are supported via a floating-point operations coprocessor. However, the floating-point types for single and double precision are not supported at the software level for the compiler resulting in a loss of 0.025, giving a score of $K_{fltRadPC} = 0.175$. Vector processing is supported via the coprocessor to provide dedicated parallel processing, given a score of $K_{vecRadPC} = 0.175$ with deductions of 0.025 for the partial reconfiguration latency in swapping functionality between vector processing steps. Finally, onboard memory available to the processor is 12 KB of instruction memory, 12 KB of data memory, and 8 MB of QMD onboard memory. RadPC's score for memory is calculated via Equation 6.21 using Equation 4.4

$$K_{memRadPC} = 0.2 * \frac{\log_{10}(8.048MB) - \log_{10}(10kB)}{\log_{10}(64GB) - \log_{10}(10kB)} = 0.085383812 \approx 0.085 \quad (6.21)$$

These scores on individual aspects of RadPC's computational performance are combined in Equation 6.22 using the scoring system of Equation 4.1.

$$C_{RadPC} = 0.100 + 0.180 + 0.175 + 0.175 + 0.085 = 0.715 \quad (6.22)$$

RAD750 Scoring RAD750 is the historic standard radiation hardened processor, developed by BAE and running on a clock frequency of 200 MHz. RAD750's clock score is calculated in Equation 6.23 using Equation 4.2.

$$K_{clk_{rad750}} = 0.2 * \frac{\log_{10}(200) - \log_{10}(1)}{\log_{10}(1000) - \log_{10}(1)} = 0.153402 \approx 0.153 \quad (6.23)$$

This clock speed supports the RAD750 being capable of 266 MIPS. In addition to the processor core, the RAD750 has two fixed-point units and one floating-point unit

that supports IEEE 754 double-precision operations [202]. These processing units give the RAD750 the scores of $K_{int_{RAD750}} = 0.200$ and $K_{flt_{RAD750}} = 0.200$ for integer and floating-point processing respectively. The system provides a 32 KB cache for the instruction memory, a 32 KB cache for the data memory, and an operational 1 MB external cache. This system is designed for the RAD750 to be provided with an onboard memory chip accessed through either the L2 interface or a PCI interface to increase the memory size. RAD750's memory score is calculated via Equation 6.24.

$$K_{mem_{RAD750}} = 0.2 * \frac{\log_{10}(2.001036GB) - \log_{10}(10kB)}{\log_{10}(64GB) - \log_{10}(10kB)} = 0.155777688 \approx 0.156 \quad (6.24)$$

For vector processing, the RAD750 would have to perform the processing either on the CPU, FPU or using one of the fixed-point Units. The large number of processing units available allows the RAD750 to perform decently at vector processing regardless of not having a dedicated vector processing unit. This gives a score of $K_{vec_{RAD750}} = 0.180$ with the deduction of 0.02 for not having the dedicated vector processing capability.

$$C_{RAD750} = 0.153 + 0.200 + 0.200 + 0.180 + 0.156 = 0.889 \quad (6.25)$$

HPSC Scoring HPSC is the new standard radiation hardened processor, developed by MicroChip using the RISC-V architecture. It provides a wide range of capabilities including 1 GHz, 512-bit vector lengths for AI/ML application, with 2 TOPS for integer operations, and 1 TFLOPS for single float operations with matrix multiplication [184]. These capabilities in integer and floating operations give scores of $K_{int_{hpse}} = 0.200$ and $K_{flt_{hpse}} = 0.200$ for the HPSC. From this clock frequency of 1 GHz, HPSC's clock score is calculated in Equation 6.26 using Equation 4.2.

$$K_{clk_{hpse}} = 0.2 * \frac{\log_{10}(1000) - \log_{10}(1)}{\log_{10}(1000) - \log_{10}(1)} = 0.2 \quad (6.26)$$

The matrix multiplication gives HPSC an exceptional vector processing capability built into the hardware and a score of $K_{vehpse} = 0.200$. HPSC provides NAND Flash, NOR Flash, and SRAM integrated memory. Additionally, HPSC provides interfaces for DDR3 and DDR4 memory chips supporting up to 32 GB per DDR interface. From this memory size, $K_{mem_{HPSC}}$ is calculated in Equation 6.27 using Equation 4.4.

$$K_{mem_{hpse}} = 0.2 * \frac{\log_{10}(64GB) - \log_{10}(10kB)}{\log_{10}(64GB) - \log_{10}(10kB)} = 0.2 \quad (6.27)$$

From these scores on the individual computational performance metric, the overall HPSC score is calculated in Equation 6.28.

$$C_{HPSC} = 0.200 + 0.200 + 0.200 + 0.200 + 0.200 = 1.000 \quad (6.28)$$

Radiation Tolerance Scoring

Radiation tolerance scoring will be balanced between TID rating and SEE resistance. TID will be evaluated using the board radiation dose resistance in krad or Mrad. SEE resistance will be primarily scored off linear energy transfer, which is the energy ionizing radiation deposits into a material per unit of distance traveled. Recovery techniques for the SEE resistance such as TMR, ECC, scrubbing, CRC, watchdog timers, and resets will be considered when LET is not available.

Flying Laptop has two processors to score for radiation tolerance implemented on a Virtex-II Pro FPGA and a RTProASIC3 FPGA. The Virtex-II Pro is not radiation hardened and its TID rating is derived from the feature size of 130 nm CMOS process used for its XC2VP50 FPGA device. Overall, the XC2VP50 has no built-in TID tolerance, a threshold

linear energy transfer of approximately 1 MeV-cm²/mg [203], and achieves fault tolerance to SEEs from the computer architecture of the UT 699 LEON 3 FT microprocessor's TMR. The RTProASIC3 has a 25 – 30 krad TID rating [204]. This FPGA has a linear energy transfer for SEL $LET_{TH} > 57$ and SEU $1.0 < LET_{TH} < 2$ verified via particle testing [205]. This gives an overall rating of 0.7 or a score of Good for low TID, but high LET.

Nexus One has no radiation protection. It is quite literally a phone that was flown to see if it could operate in orbit for a week. Given the lack of inherent or added protection, this onboard computer will be given a rating of Poor for radiation tolerance.

NanoMind A3200 has a microcontroller with a TID rating of 20 krad (Si). There isn't a great deal of linear energy transfer information which leads to a low score of 0.2 (Poor) based on the documentation for the Atmel AT32UC3C device.

EnduroSat OBC Type I use an ARM Cortex M4 processor, which does not have a TID rating or a LET rating. However, the EnduroSat OBC does have flight history. It does provide watchdog timers and resets options to recover from faults. This gives an overall rating of Fair for EnduroSat OBC Type I as it has limited radiation tolerance techniques, but does have flight history.

ISIS iOBC is a flight-proven, high-performance processing unit using an ARM9 processor with a speed of 400 MHz. It operates with a watchdog timer that triggers after a set interval of failure to communicate or perform operations [206]. The overall radiation score for the iOBC is Fair given the lack of listed recovery techniques but proven flight history.

AstroDev MarCO C&DH uses a small MSP430F2618 microcontroller as the processor for the onboard computer. This microcontroller does not have any radiation techniques beyond watchdog timers and reset. Inherent rating tolerance is roughly 5 krad based on the microcontroller family. However, the microcontroller has small cross-section and therefore a low area of vulnerability. This microcontroller does have a successful flight history to Mars. Overall, it is given a rating of Good given the successful flight history in deep space.

RadPC with coprocessor has no radiation hardening, and the CMOS feature size of the FPGA provides about 50 krad worth of protection. On the SEE side, RadPC implements the majority of fault recovery techniques including QMR, ECC checking, memory scrubbing, CRC checks on internal comms, watchdog timers, and resets. Overall, the TID provides slightly higher reliability over prior generation FPGAs and the SEEs are greatly mitigated for a score of 0.8.

RAD750 is a radiation hardened processor design specifically for aerospace operation with a TID rating of 1 Mrad (Si) and a SEU resistance of: $> 1.6E - 10$ errors/bit-day. This is an order of magnitude better than any of the small satellite processor examples. The RAD750 receives a perfect score of Excellent for radiation tolerance.

HPSC is the replacement for the RAD750 with TID rating of 100 krad (tested to 200 krad) and SEL 78 MeV for SEE resistance. Again, these ratings are an order of magnitude better than any of the small satellite processor examples, and the HPSC receives a perfect score of Excellent for radiation tolerance.

Power Scoring

For the Flying Laptop, there are two processors in the onboard computer to consider for power, the Virtex-II Pro, XC2VP50, and the RTProASIC3, RT3PE3000L. For Xilinx's Virtex-II Pro, the power consumption is 0.9 mW/MHz with the UT699 processor from the LEON3FT family operating at 2.5 MHz [207, 208]. The RTProASIC3 requires roughly 0.3075 W of power based on the maximum FPGA core power requirements and maximum FPGA I/O power requirements possible [204]. This comes to a total of ~ 2.5575 W for both processing nodes with total satellite power being 40 W from Table 2.2, giving a score of 0.936 or Excellent from Equation 4.6.

For NanoMind A3200, the microcontroller requires very little power, 0.17 W. Since the power consumption for NanoMind A3200 is not available, power consumption will be

estimated off the satellite mass of 4 kg using $P_t(W) = 2.26(4) + 3.44 = 12.48$ W from Table 4.2. From Equation 4.6, NanoMind A3200's power consumption score is 0.986 and a rating of Excellent.

For Nexus One, the phone requires only 68.6 mW of power without the screen enabled. The satellite for this onboard computer was PhoneSat 1.0 and was powered by a 124 Wh battery. Given the mission duration of 1 week, the satellite consumed an average of 0.738 W. From Equation 4.6, Nexus One's power consumption score is 0.738 or Fair.

EnduroSat OBC Type I has an average power consumption of 0.9 W during operation. The total EnduroSat One satellite has an estimated power consumption from a total mass of 1 kg and Table 4.2, giving $P_t(W) = 5 * 1 + 0.7W = 5.7$ W. From this estimate, Equation 4.6 gives a score of 0.842 or Good.

The ISIS iOBC requires 0.40 W for operation and the overall satellite power consumption is estimated at $P_t(W) = 2.26(3) + 3.44 = 10.22$ W from its mass of 3 kg and the mass to power equation for nanosatellites from Table 4.2. While this estimate cannot be confirmed due to power failure by the satellite, it is a realistic value and gives a power score of 0.961 or Excellent from Equation 4.6.

AstroDev MarCO C&DH uses a microcontroller which consumes ~ 14.454 mW of power for a satellite with a power profile of 35 W when near Earth and 17 W when near Mars [209]. Two scores are calculated, one for Earth orbit at the start of the mission, and one for Mars orbit at the end of the mission. From Equation 4.6, MarCo has two scores of 1.000 for Earth orbit and 0.999 for Mars orbit. Both scores are Excellent. The exceptional power score is a result of a highly efficient design that focuses on reliability for deep space flight.

RadPC has not yet been integrated into a small satellite. However, two of the previous versions of the boards were flown on Firefly's Blue Ghost Lander, and that 2.6 kg payload will be used to estimate the power for a nanosatellite. For a nanosatellite, $P_t(W) = 2.26(2.6) + 3.44 = 9.316$ W from Table 4.2, RadPC with coprocessors takes 2.1 W, and gives a score of

0.775 or Good for RadPC running at 32 MHz with the coprocessor.

For the RAD750 and the HPSC, the power scoring approach does not work well as the power consumption is an order of magnitude higher. RAD750 consumes 20 W during standard operation and the HPSC is higher at 25 W. Both are designed for roughly 100–500 kg satellites. Scoring will use the $P_t(W) = 1.044 * M + 15.56$ from Table 4.2 since the 100–500 kg roughly matches the 11–200 kg range of the microsatellite classification. Mass will be assumed to be 100 kg as it is the lowest value in the actual range that is centered on the microsatellite class where the equation will be valid. This gives 159.56 W and a score of 0.875 or Good for the RAD750 and a score of 0.843 or Good for the HPSC.

Cost Scoring

For onboard computer cost scoring, the price of the onboard computer is directly compared to each other in 2025 dollars. If the onboard computer price is not available as in the case of the Flying Laptop, RAD750, and HPSC then the sum of the prices for the processing nodes will be compared instead of the price of the onboard computer. These scores are provided in Table 6.1 along with the individual small satellite OBCs' costs and radiation hardened processors' costs. This direct comparison provides a rough measure of the overall expense for an OBC.

Comparison

Scores for the six example small satellites, two traditional radiation hardened processors, and RadPC with Coprocessor are compiled in Table 6.1 for comparison. The results categorize these processors into three different groups based on the focuses demonstrated by their scores in the trade space, prioritizing power, prioritizing radiation tolerance, and a compromise between all four categories.

As shown in Table 6.1, RadPC is a solution balancing itself in the center of the trade space. Entries in Table 6.1 can be grouped into three categories, power & cost focused, radiation & performance focused, and balanced. NanoMind 3200, Nexus One, ISIS iOBC, and MarCO C&DH all fit into the power & cost focus with better scores in those columns. These missions all closely align with the small satellite approach of "fly fast, fly often". Frequent, short duration missions are the default. Eventual hardware failure is assumed with the mission length is designed around it. Or, as in the case with the MarCO C&DH, a longer duration mission is performed with a smaller processor and multiple satellites to minimize risk.

As a contrast, RAD750 and HPSC are clearly focused on radiation tolerance, computational capability, and power for all mission classes. Although the RAD750 has a lower score in computation performance than the HPSC, it achieved second place overall despite being a decades old processor. The major downside to this approach and the reason that neither the RAD750 nor in future the HPSC will likely be a widespread processor for small satellites is expense. One HPSC is worth 8-5000 of all the small satellite processor options, making HPSC prohibitively expensive.

Table 6.1: On-Board Computer Comparison

Computer	Class	Compute Score	Radiation Score	Power Score	Cost Score
Flying Laptop	SmallSat	Good 0.834	Good	Excellent 0.936, 0.3075 W	Fair 0.515 \$10,082.13
NanoMind A3200	SmallSat	Good 0.721	Poor	Excellent 0.986, 0.17 W	Excellent 1.000 \$15.75
Nexus One	SmallSat	Fair 0.640	Poor	Fair 0.738, 68.6 mW	Excellent 0.812 \$199.00
EnduroSat OBC Type I	SmallSat	Fair 0.699	Fair	Good 0.842, 0.9 W	Fair 0.476 \$16,000
ISIS iOBC	SmallSat	Fair 0.614	Poor	Excellent 0.961, 0.40 W	Good 0.572 \$5,000
MarCO C&DH	SmallSat	Poor 0.379	Good	Excellent 0.999, 14.45 mW	Excellent 0.986 \$18.23
RadPC & Co-processor	SmallSat	Good 0.715	Good	Good 0.775, 2.1 W	Good 0.576 \$4,727.79
RAD750	LargeSat	Good 0.889	Excellent	Good 0.875, 20 W	Poor 0.100 \$250,000
HPSC	LargeSat	Excellent 1.000	Excellent	Good 0.843, 25 W	Poor 0.330 \$80,000

The final group, Flying Laptop, EnduroSat OBC Type I, and RadPC, are balanced roughly in the center of the trade space. Flying Laptop and EnduroSat both beat RadPC in

power efficiency relative to satellite mass. However, RadPC is more affordable than either option or is better than EnduroSat in radiation tolerance. Although not scored, RadPC also has an advantage in hardware versatility from the dynamic partial reconfiguration system. The ability to swap coprocessors allows RadPC to scale its capabilities with FPGA size and increase computational performance via task specific coprocessors. Overall, RadPC compares well to the rest of its group. RadPC focuses slightly more on reducing overall cost and increasing radiation tolerance. RadPC is the most balanced computing approach across the trade space with its novel architecture being the only small satellite computer that achieves the score of Good across all scoring categories.

Scoring System Reliability

In order to test that the scoring system is reliable across multiple observers, a statistical analysis was performed that involved having five additional raters use the system to score the same group of space computers. Scoring was performed on OBCs for the four categories of computational performance, radiation tolerance, power consumption, and cost using ordinal values. The raters ranked OBC systems within each category on the acceptability of each OBC and provided a score of poor, fair, good, or excellent for the OBC. With ordinal values, the score has a rank or order, but the intervals between the values are not necessarily equal or meaningful. In the case of this scoring system, there is an inherent requirement for the OBC to be capable of performing computational tasks, maintaining operation in a harsh radiation environment, operating on limited power, and being affordable. The OBCs are ranked relative to each other (Poor, Fair, Good, Excellent) with each score showing that individual OBC's standing within the overall sample set. For this coding, a score of 0 will never be given to the OBCs in any category as the OBC has inherent baseline capabilities. An OBC that has no computational performance would not be capable of functioning as a computer. All computers have some level of tolerance to radiation as previously discussed for TID and SEE type of radiation induced faults. Zero power consumption for an operating computer is a physical impossibility. And finally, these OBCs are existing computer systems that are either available for purchase or have been developed and therefore have an associated cost.

For the statistical analysis only, poor will be coded to 1, fair will be coded to 2, good will be coded to 3, and excellent will be coded as 4. A total of five raters performed the scoring on nine OBCs, six example small satellite OBCs, two radiation hardened processors, and the developed solution (RadPC & Coprocessor). The raters used the scoring equations previously described then the scores were mapped into the ordinal system of poor/fair/good/excellent

for the statistical analysis. The results of this scoring by five raters are shown in Table 6.2.

Table 6.2: Multiple Raters On-Board Computer Rankings

Computer	Compute Score	Radiation Score	Power Score	Cost Score
Flying Laptop	Good (0.8340)	Good	Excellent (0.9360)	Fair (0.5150)
	Fair (0.7550)	Excellent	Excellent (0.9361)	Poor (0.2060)
	Fair (0.7300)	Good	Excellent (0.9400)	Fair (0.4000)
	Excellent (0.7553)	Fair	Excellent (0.9360)	Fair (0.3990)
	Good	Good	Excellent	Good
NanoMind A3200	Good (0.7210)	Poor	Excellent (0.9860)	Excellent (1.0000)
	Poor (0.7213)	Good	Excellent (0.9864)	Excellent (1.0000)
	Good (0.7000)	Fair	Excellent (0.9800)	Excellent (1.0000)
	Good (0.72141)	Good	Excellent (0.9860)	Excellent (1.0000)
	Fair	Fair	Excellent	Excellent
Continued on next page				

Computer	Compute Score	Radiation Score	Power Score	Cost Score
Nexus One	Fair (0.6400)	Poor	Fair (0.7380)	Excellent (0.8120)
	Good (0.8396)	Poor	Excellent (0.9070)	Good (0.7640)
	Good (0.8400)	Poor	Excellent (0.9100)	Good (0.7600)
	Good (0.7400)	Poor	Excellent (0.9070)	Excellent (0.7640)
	Good	Poor	Excellent	Excellent
EnduroSat OBC	Fair (0.6990)	Fair	Good (0.8420)	Fair (0.4760)
	Good (0.8542)	Good	Good (0.8421)	Poor (0.3558)
	Good (0.8100)	Fair	Good (0.8400)	Fair (0.3600)
	Excellent (0.83435)	Good	Excellent (0.8420)	Fair (0.3560)
	Excellent	Good	Good	Good
Continued on next page				

Computer	Compute Score	Radiation Score	Power Score	Cost Score
ISIS iOBC	Fair (0.6140)	Poor	Excellent (0.9610)	Good (0.5720)
	Fair (0.7332)	Fair	Excellent (0.9609)	Fair (0.4640)
	Good (0.6100)	Poor	Excellent (0.9600)	Fair (0.4600)
	Good	Poor	Excellent (0.9610)	Fair (0.4640)
	Fair (0.61347)	Good	Excellent	Good
MarCO C&DH	Poor (0.3790)	Good	Excellent (0.9990)	Excellent (0.9860)
	Poor (0.4991)	Fair	Excellent (0.9991)	Excellent (0.9864)
	Poor (0.3750)	Fair	Excellent (0.9900)	Excellent (0.9800)
	Fair (0.37894)	Fair	Excellent (0.9990)	Excellent (0.9860)
	Poor	Fair	Excellent	Excellent
Continued on next page				

Computer	Compute Score	Radiation Score	Power Score	Cost Score
RadPC	Good (0.7150)	Good	Good (0.7750)	Good (0.5760)
	Fair (0.7407)	Excellent	Good (0.7531)	Fair (0.4692)
	Good (0.7400)	Good	Fair (0.7500)	Good (0.4700)
	Good (0.66034)	Good	Excellent (0.0753)	Fair (0.4640)
	Good	Good	Good	Good
RAD750	Good (0.8890)	Excellent	Good (0.8750)	Poor (0.1000)
	Good (0.7842)	Good	Good (0.8333)	Poor (0.2060)
	Good (0.8600)	Excellent	Good (0.8300)	Poor (0.1000)
	Excellent (0.7344)	Excellent	Excellent (0.8330)	Poor (0.1000)
	Good	Excellent	Good	Poor
HPSC	Excellent (1.0000)	Excellent	Good (0.8430)	Poor (0.3300)
	Excellent (1.0000)	Good	Good (0.7916)	Poor (0.2060)
	Fair (0.6800)	Good	Good (0.7900)	Poor (0.2100)
	Excellent (1.0000)	Excellent	Excellent (0.7920)	Poor (0.2060)
	Excellent	Good	Good	Fair
End of table				

Two statistical tests were performed on these scoring, a Weighted Cohen's Kappa test and a Kruskal-Vallis test. The Weighted Cohen's Kappa tested the inter-rater reliability of scoring. The weighted Cohen's Kappa was selected to reflect the nature of the scoring, Poor

<Fair <Good <Excellent, and penalizes greater disagreements between two rater's scores. For example, scores of Poor and Good are in more disagreement than scores of Poor and Fair. The results of the Weighted Cohen's Kappa is shown in Table 6.3 with the score for an average pairwise quadratic weighted kappa (κ).

Table 6.3: Average Pairwise Quadratic Weighted Kappa

	Compute κ	Radiation κ	Power κ	Cost κ
Results	0.5629	0.6457	0.3895	0.8295
	Moderate agreement	Substantial agreement	Fair agreement	Almost perfect agreement

In addition to the Weighted Cohen's Kappa, a Kruskal-Wallis test was also performed on the multi-rater scores. This tests if individual raters are consistently harsher or more lenient than others. The results of the Kruskal-Wallis are shown in Table 6.4.

Table 6.4: Kruskal-Wallis

	Compute Results	Radiation Results	Power Results	Cost Results
Test Statistic (H)	H = 5.5689	H = 1.0458	H = 7.0598	H = 2.8524
p-value (p)	p = 0.2337	p = 0.9028	p = 0.1328	p = 0.5828
Results	Not significant	Not significant	Not significant	Not significant

The results of these statistical analysis show that the raters were in general agreement and none of the raters was consistently harsher or lenient compared with the others. Analysis shows the raters had a high level of agreement for power consumption scoring and cost

scoring. Computational performance scoring and radiation tolerance scoring had a lower, but acceptable levels of agreements.

Dissemination

Beyond this dissertation, the results of this research have been published at two conferences, the 2025 IEEE Aerospace Conference and the 2025 IEEE SCC Conference[210, 211]. Results from this research are currently under review by the Journal Frontiers in Aerospace Engineering[212]. Outside the academic sphere, the RadPC computer architecture with a dynamically swappable coprocessor has been patented in the United States under Patent Number 12,287,713, shown in Figure 6.1.



Figure 6.1: RadPC+Coprocessor Patent

CONCLUSION AND FUTURE WORK

This dissertation presented a dynamically partially reconfigurable coprocessor for an FPGA-based aerospace computer system with a newly developed single board computer capable of providing onboard partial reconfiguration and power monitoring. This approach is an expansion in capability of the RadPC aerospace computer developed by Montana State University. It continues previous work done on the RadPC radiation-tolerant computing architecture by adding dynamic partial reconfiguration of hardware accelerator coprocessors. Four artifacts were developed and demonstrated in support of the novel dynamic partial reconfiguration capability. These new artifacts also supported onboard power monitoring and supported fault-mitigation in the coprocessor component.

A floating-point coprocessor was developed to demonstrate partial reconfiguration during operation was possible. The limitations in this manual approach for partial reconfiguration provided the design requirements for onboard partial reconfiguration. Development of the single board computer focused on providing the onboard capability to dynamically swap hardware accelerator automatically within the coprocessor block on the FPGA. After successful coprocessor functionality swapping was demonstrated, it led to a first-difference edge detection coprocessor being developed. This coprocessor was able to be dynamically reconfigured during operation and performed a series of image processing operations on 16x16 pixel test images. Power monitoring was integrated into the new single board computer, Hydra, and used to observe power consumption during coprocessor operation. This verified the Hydra PCB was capable of supplying significant power during full configuration, RadPC operation, partial reconfiguration, and coprocessor operation. The power monitoring observed each of the individual voltage supplies on the Hydra. Analyzing the power consumption allowed the partial reconfiguration operation to be characterized. Finally, a first-derivative edge detection coprocessor was developed, and a reliability analysis

was performed for the RadPC/coprocessor system and on the coprocessor itself. Analysis showed that the coprocessor was capable of operating in LEO within certain time constraints with low likelihood of a failure.

Finally, the complete integrated solution of RadPC with coprocessor was scored and compared to six example small satellites and two radiation hardened processors for large satellites. Scoring showed that the RadPC with coprocessor system is centered in the radiation tolerance, computational performance, low power, and cost trade space. The RadPC and coprocessor system provided the most balanced approach to small satellite computing. This design successfully created an integrated dynamic partially reconfigurable coprocessor for a fault-tolerant computer that is affordable to a small satellite university development team.

While the current system has demonstrated the concept of dynamic partial reconfiguration for swapping hardware accelerators in small satellite computers. However, some future work is required to incorporate a number of improvements into RadPC and coprocessor system. Partial reconfiguration speed should be increased, and the port for partial reconfiguration requests should be converted from parallel to serial. For coprocessor and RadPC communication, a coprocessor ID code needs to be implemented so that RadPC can independently identify a coprocessor. In addition to the coprocessor identification system, RadPC requires a method to test a coprocessor's operation and allow the coprocessor to inform RadPC of faults. Another important issue to be addressed is the lack of interrupts or control registers in RadPC for the coprocessor. Without a standardized interface for interacting with the coprocessor, RadPC will require custom modification for every new coprocessor.

REFERENCES CITED

- [1] Eric Mahr, Anh Tu, and Anil Gupta. Development of the small satellite cost model 2019 (sscm19). In *2020 IEEE Aerospace Conference*, pages 1–12, March 2020. doi: 10.1109/AERO47225.2020.9172374.
- [2] Robyn M. Millan, Rudolf von Steiger, Meir Ariel, Sergey Bartalev, Maurice Borgeaud, Stefano Campagnola, Julie C. Castillo-Rogez, René Fléron, Volker Gass, Anna Gregorio, David M. Klumpar, Bhavya Lal, Malcolm Macdonald, Jong Uk Park, V. Sambasiva Rao, Klaus Schilling, Graeme Stephens, Alan M. Title, and Ji Wu. Small satellites for space science: A cospar scientific roadmap. *Advances in Space Research*, 64(8):1466–1517, 2019. ISSN 0273-1177. doi: <https://doi.org/10.1016/j.asr.2019.07.035>. URL <https://www.sciencedirect.com/science/article/pii/S0273117719305411>.
- [3] Martin N. Sweeting. Modern small satellites-changing the economics of space. *Proceedings of the IEEE*, 106(3):343–361, March 2018. ISSN 1558-2256. doi: 10.1109/JPROC.2018.2806218.
- [4] Amal Chandran. The smallsat revolution. In *44th COSPAR Scientific Assembly. Held 16-24 July*, volume 44, pages .3–0001–22, July 2022. URL <https://ui.adsabs.harvard.edu/abs/2022cosp...44.3066C>. Provided by the SAO/NASA Astrophysics Data System.
- [5] NASA Science Mission Directorate. Small missions, big lessons: Informing large missions from smallsat/class d lessons learned. Technical report, National Aeronautics and Space Administration, September 2024. URL <https://spacenews.com/smallsats-offer-lessons-for-big-science-missions/>. Presented at the AIAA ASCEND conference, Las Vegas, July 2024.
- [6] Nodir Adilov, Peter Alexander, Brendan Cunningham, and Nikolas Albertson. An analysis of launch cost reductions for low earth orbit satellites. *Economics Bulletin*, 03 2024.
- [7] Federal Aviation Administration. The annual compendium of commercial space transportation: 2018. Technical report, Federal Aviation Administration, January 2018. URL https://www.faa.gov/about/office_org/headquarters_offices/ast/media/2018_ast_compendium.pdf.
- [8] BryceTech. Smallsats by the numbers. Technical report, BryceTech, 2025. URL https://www.brycetek.com/reports/report-documents/smallsats-2025/BryceTech_Smallsats-by-the-Numbers-2025.pdf.
- [9] Francesco Bordi and Christopher J. Scolese. A new approach to mission classification and risk management for nasa space flight missions. Technical report, The Aerospace Corporation and NASA Goddard Space Flight Center, 2018. URL <https://ntrs.nasa.gov/api/citations/20180007026/downloads/20180007026.pdf>.

- [10] Nadim F. Haddad, Ronald D. Brown, Richard Ferguson, Andrew T. Kelly, Reed K. Lawrence, Daniel M. Pirkel, and John C. Rodgers. Second generation (200mhz) rad750 microprocessor radiation evaluation. In *2011 12th European Conference on Radiation and Its Effects on Components and Systems*, pages 877–880, Sep. 2011. doi: 10.1109/RADECS.2011.6131320.
- [11] Edouard Mathieu, Pablo Rosado, and Max Roser. Space exploration and satellites. *Our World in Data*, 2022. URL <https://ourworldindata.org/space-exploration-satellites>.
- [12] Harry W. Jones. The future impact of much lower launch cost. In *International Conference on Environmental Systems (ICES)*, July 2018. URL <https://ntrs.nasa.gov/citations/20200001091>.
- [13] Bernard Kutter. *Commercial Launch Services: an Enabler for Launch Vehicle Evolution and Cost Reduction*. doi: 10.2514/6.2006-7271. URL <https://arc.aiaa.org/doi/abs/10.2514/6.2006-7271>.
- [14] Thyrso Villela, Cesar A. Costa, Alessandra M. Brandão, Fernando T. Bueno, and Rodrigo Leonardi. Towards the thousandth cubesat: A statistical overview. *International Journal of Aerospace Engineering*, 2019(1):5063145, 2019. doi: <https://doi.org/10.1155/2019/5063145>. URL <https://onlinelibrary.wiley.com/doi/abs/10.1155/2019/5063145>.
- [15] Settapong Malisuwan and Borwonrat Kanchanarat. Small satellites for low-cost space access: Launch, deployment, integration, and in-space logistics. *American Journal of Industrial and Business Management*, 12(10), October 2022. doi: 10.4236/ajibm.2022.1210082.
- [16] Elizabeth Buchen and Dominic DePasquale. 2014 nano-satellite report: The nano-satellite market analysis. Technical report, SpaceWorks Enterprises, Inc., 2014. URL <https://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=3018&context=smallsat>.
- [17] Katharine Smith, Nicholas Crisp, and Peter Hollingsworth. Launch and deployment of distributed small satellite systems. In *Proceedings of the International Astronautical Congress (IAC)*. International Astronautical Federation, IAF, September 2014.
- [18] Gil Denis, Didier Alary, Xavier Pasco, Nathalie Pisot, Delphine Texier, and Sandrine Toulza. From new space to big space: how commercial space dream is becoming a reality. . . . *Acta Astronautica*, 166, 10 2019. doi: 10.1016/j.actaastro.2019.08.031.
- [19] Daniel Selva and David Krejci. A survey and assessment of the capabilities of cubesats for earth observation. *Acta Astronautica*, 74:50–68, 2012. ISSN 0094-5765. doi: <https://doi.org/10.1016/j.actaastro.2011.12.014>. URL <https://www.sciencedirect.com/science/article/pii/S0094576511003742>.

- [20] Armen Poghosyan and Alessandro Golkar. Cubesat evolution: Analyzing cubesat capabilities for conducting science missions. *Progress in Aerospace Sciences*, 88:59–83, 2017. ISSN 0376-0421. doi: <https://doi.org/10.1016/j.paerosci.2016.11.002>. URL <https://www.sciencedirect.com/science/article/pii/S0376042116300951>.
- [21] Rob Zimmerman, Deanna Doan, Lawrence Leung, James Mason, Nate Parsons, and Kam Shahid. Commissioning the world’s largest satellite constellation. *Proceedings of the 32nd Annual AIAA/USU Conference on Small Satellites*, August 2018. URL https://s3vi.ndc.nasa.gov/ssri-kb/static/resources/Commissioning%20the%20World_s%20Largest%20Satellite%20Constellation.pdf.
- [22] Georges Labrèche, David Evans, Dominik Marszk, Tom Mladenov, Vasundhara Shiradhonkar, Tanguy Soto, and Vladimir Zelenevskiy. Ops-sat spacecraft autonomy with tensorflow lite, unsupervised learning, and online machine learning. In *2022 IEEE Aerospace Conference (AERO)*, pages 1–17, March 2022. doi: 10.1109/AERO53065.2022.9843402.
- [23] Kevin D. Earle, Matthew J. Carrier, William M. Cirillo, Melanie L. Grande, Christopher A. Jones, Emily L. Judd, Jordan J. Klovstad, Andrew C. Owens, David M. Reeves, and Matthew A. Stafford. Strategic framework for nasa’s space technology mission directorate. Conference Paper AIAA 2018-5136, 2018. URL <https://ntrs.nasa.gov/api/citations/20190033117/downloads/20190033117.pdf>.
- [24] National Aeronautics and Space Administration. Nasa 2022 strategic plan. Strategic Plan NPD 1001.0D, National Aeronautics and Space Administration, 2022. URL <https://www.nasa.gov/wp-content/uploads/2023/09/fy-22-strategic-plan-1.pdf>.
- [25] MOOG, Inc. Cascade single board computer. Datasheet Form 500-1440 0624, MOOG, Inc., 2024. URL <https://www.moog.com/content/dam/moog/literature/sdg/space/avionics/moog-cascade-single-board-computer-datasheet.pdf>.
- [26] Sasha V. Weston, Craig D. Burkhard, Jan M. Stupl, Rachel L. Ticknor, Bruce D. Yost, Rebekah A. Austin, Pavel Galchenko, Lauri K. Newman, and Luis Santos Soto. State of the art small spacecraft technology. Technical report, Millennium Engineering & Integration Services and Ames Research Center and Goddard Space Flight Center, Moffett Field, CA and Greenbelt, MD, 2024.
- [27] B. D. Yost, S. Weston, J. Hines, and C. Burkhard. An overview of the current state of the art on small spacecraft avionics systems. Conference Paper 0521, 2022.
- [28] Marius Meyer, Tobias Kenter, and Christian Plessl. Evaluating fpga accelerator performance with a parameterized opencl adaptation of selected benchmarks of the hpcchallenge benchmark suite. In *2020 IEEE/ACM International Workshop on Heterogeneous High-performance Reconfigurable Computing (H2RC)*, pages 10–18, Nov 2020. doi: 10.1109/H2RC51942.2020.00007.

- [29] Daniel C. Stumpp, Himanshu Akolkar, Alan D. George, and Ryad B. Benosman. harms: A hardware acceleration architecture for real-time event-based optical flow. *IEEE Access*, 10:58181–58198, 2022. ISSN 2169-3536. doi: 10.1109/ACCESS.2022.3172396.
- [30] Tyler Garrett, Diego Wildenstein, Eileen Wang, Evan Gretok, and Alan George. Reliable ml-based image processing and compression for an onboard imaging pipeline. In *2024 IEEE Space Computing Conference (SCC)*, pages 93–103, July 2024. doi: 10.1109/SCC61854.2024.00017.
- [31] MarketsandMarkets. Small satellites market: Transforming space tech in north america. March 2025. Accessed: 2025-08-26.
- [32] Richard Galazzo. Timeline from 1g to 5g: A brief history of cell phones. January 2020.
- [33] Yifan Lu, Qi Shao, Honghao Yue, and Fei Yang. A review of the space environment effects on spacecraft in different orbits. *IEEE Access*, 7:93473–93488, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2927811.
- [34] A Holmes-Siedle and L Adams. *Handbook of radiation effects*. Oxford Univ. Press, Inc., New York, NY (United States), 12 1993.
- [35] H.L. Hughes and J.M. Benedetto. Radiation effects and hardening of mos technology: devices and circuits. *IEEE Transactions on Nuclear Science*, 50(3):500–521, June 2003. ISSN 1558-1578. doi: 10.1109/TNS.2003.812928.
- [36] C.-M. Zhang, F. Jazaeri, A. Pezzotta, C. Bruschini, G. Borghello, S. Mattiazzo, A. Baschiroto, and C. Enz. Total ionizing dose effects on analog performance of 28 nm bulk mosfets. In *2017 47th European Solid-State Device Research Conference (ESSDERC)*, pages 30–33, Sep. 2017. doi: 10.1109/ESSDERC.2017.8066584.
- [37] Rajan Arora, Zachary E. Fleetwood, En Xia Zhang, Nelson E. Lourenco, John D. Cressler, Daniel M. Fleetwood, Ronald D. Schrimpf, Akil K. Sutton, Greg Freeman, and Brian Greene. Impact of technology scaling in sub-100 nm nmosfets on total-dose radiation response and hot-carrier reliability. *IEEE Transactions on Nuclear Science*, 61(3):1426–1432, June 2014. ISSN 1558-1578. doi: 10.1109/TNS.2014.2320494.
- [38] H. J. Barnaby. Total-ionizing-dose effects in modern cmos technologies. *IEEE Transactions on Nuclear Science*, 53(6):3103–3121, Dec 2006. ISSN 1558-1578. doi: 10.1109/TNS.2006.885952.
- [39] Rui Liu, Adrian Evans, Li Chen, Yuanqing Li, Maximilien Glorieux, Richard Wong, Shi-Jie Wen, Joao Cunha, Leopold Summerer, and Véronique Ferlet-Cavrois. Single event transient and tid study in 28 nm utbb fdsoi technology. *IEEE Transactions on Nuclear Science*, 64(1):113–118, Jan 2017. ISSN 1558-1578. doi: 10.1109/TNS.2016.2627015.

- [40] C.-M. Zhang, F. Jazaeri, A. Pezzotta, C. Bruschini, G. Borghello, F. Faccio, S. Mattiazzo, A. Baschiroto, and C.ENZ. Gigarad total ionizing dose and post-irradiation effects on 28 nm bulk mosfets. In *2016 IEEE Nuclear Science Symposium, Medical Imaging Conference and Room-Temperature Semiconductor Detector Workshop (NSS/MIC/RTSD)*, pages 1–4, Oct 2016. doi: 10.1109/NSSMIC.2016.8069869.
- [41] Georgios Kottaras, Theodoros Sarris, Athanasios Psomoulis, Ilias Ioakeimidis, Angelos Papathanasiou, David Pitchford, and Ingmar Sandberg. A low-power, radiation-hardened single event effect rate detection system on a chip for real time monitoring of single event effects on low earth orbit satellites. In *2022 IFIP/IEEE 30th International Conference on Very Large Scale Integration (VLSI-SoC)*, pages 1–6, Oct 2022. doi: 10.1109/VLSI-SoC54400.2022.9939638.
- [42] Daniel H. Lo and Thien C. Tran. Single event effects susceptibilities of select commercial-off-the-shelf components for space. In *2022 IEEE Radiation Effects Data Workshop (REDW) (in conjunction with 2022 NSREC)*, pages 1–7, July 2022. doi: 10.1109/REDW56037.2022.9921706.
- [43] Oleg Khasanov, Edgar Dvilis, Mariya Petyukevich, and Igor Shamanin. Metal-ceramic nanocomposite for radiation shielding of electronics. In *2020 7th International Congress on Energy Fluxes and Radiation Effects (EFRE)*, pages 1076–1079, Sep. 2020. doi: 10.1109/EFRE47760.2020.9242100.
- [44] He Lv, Yi Sun, Pengwei Li, Lei Luo, Hongwei Zhang, Qingkui Yu, and Min Tang. Research on optimization design of radiation dose shield hardening for aerospace components. In *2017 Prognostics and System Health Management Conference (PHM-Harbin)*, pages 1–5, July 2017. doi: 10.1109/PHM.2017.8079240.
- [45] Yu Han, Tao Ying, Jianqun Yang, Yang Zhao, and Xingji Li. Optimization of radiation shielding composite materials tailored for space mission applications. *IEEE Transactions on Nuclear Science*, 72(5):1790–1799, May 2025. ISSN 1558-1578. doi: 10.1109/TNS.2025.3558902.
- [46] Stefania Peracchi, Benjamin James, Federico Pagani, Vladimir Pan, James Vohradsky, David Bolst, Dale Anthony Prokopovich, Susanna Guatelli, Marco Petasecca, Michael L. F. Lerch, Sung Hyun Lee, Taku Inaniwa, Naruhiro Matsufuji, Marco Povoli, Angela Kok, Michael Jackson, Timothy Squire, Anatoly B. Rosenfeld, and Linh T. Tran. Radiation shielding evaluation of spacecraft walls against heavy ions using microdosimetry. *IEEE Transactions on Nuclear Science*, 68(5):897–905, May 2021. ISSN 1558-1578. doi: 10.1109/TNS.2020.3032946.
- [47] Jason Crusan and Carol Galica. Nasa’s cubesat launch initiative: Enabling broad access to space. *Acta Astronautica*, 157:51–60, 2019. ISSN 0094-5765. doi: <https://doi.org/10.1016/j.actaastro.2018.08.048>. URL <https://www.sciencedirect.com/science/article/pii/S0094576517303107>.

- [48] Alexey O. Akhmetov, Georgy S. Sorokoumov, Dmitry V. Bobrovsky, Dmitry V. Boychenko, Alexander Y. Nikiforov, Alexander I. Chumakov, Jean-Charles Thomas, Aleksei U. Maksimov, and Nikolay A. Vasilenkov. The effectiveness of specialized ic packages: Experimental and monte-carlo simulation for basic orbits and spacecraft shielding. In *2015 15th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, pages 1–5, Sep. 2015. doi: 10.1109/RADECS.2015.7365674.
- [49] MohammadReza Baghbanmanesh, Franco Maloberti, and Umberto Gatti. A 10-bit radiation-hardened by design (rhbd) sar adc for space applications. In *2017 New Generation of CAS (NGCAS)*, pages 53–56, Sep. 2017. doi: 10.1109/NGCAS.2017.14.
- [50] Linjiang Tang, Chengan Wan, Xiaofeng Sun, Minghua Zhang, and WeiWei Ye. Research on anti-radiation hardening technology of packaging. In *2022 23rd International Conference on Electronic Packaging Technology (ICEPT)*, pages 1–4, Aug 2022. doi: 10.1109/ICEPT56209.2022.9873468.
- [51] Alberto Martin-Ortega, Santiago Rodríguez, Jose Mingo, Sergio Ibarmia, Joaquín Rivas, Sergio López-Buedo, Celia Ongil, and Marta Portela-Garcia. Data analysis and results of the radiation-tolerant collaborative computer on-board optos cubesat. *International Journal of Aerospace Engineering*, 2019:1–11, 02 2019. doi: 10.1155/2019/1425892.
- [52] A. Makihara, M. Midorikawa, T. Yamaguchi, Y. Iide, T. Yokose, Y. Tsuchiya, T. Arimitsu, H. Asai, H. Shindou, S. Kuboyama, and S. Matsuda. Hardness-by-design approach for 0.15 μm fully depleted cmos/soi digital logic devices with enhanced seu/set immunity. *IEEE Transactions on Nuclear Science*, 52(6):2524–2530, Dec 2005. ISSN 1558-1578. doi: 10.1109/TNS.2005.860716.
- [53] Oliver Schrape, Anselm Breitenreiter, Steffen Zeidler, and Milos Krstic. Aspects on timing modeling of radiation-hardness by design standard cell-based tnr flip-flops. In *2019 22nd Euromicro Conference on Digital System Design (DSD)*, pages 639–642, Aug 2019. doi: 10.1109/DSD.2019.00100.
- [54] Tianwen Li, Hongjin Liu, and Haigang Yang. Design and characterization of seu hardened circuits for sram-based fpga. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 27(6):1276–1283, June 2019. ISSN 1557-9999. doi: 10.1109/TVLSI.2019.2892838.
- [55] Margaret A. Sullivan, Herschel H. Loomis, and Alan A. Ross. Employment of reduced precision redundancy for fault tolerant fpga applications. In *2009 17th IEEE Symposium on Field Programmable Custom Computing Machines*, pages 283–286, April 2009. doi: 10.1109/FCCM.2009.53.

- [56] Kostas Siozios and Dimitrios Soudris. A methodology for alleviating the performance degradation of tmr solutions. *IEEE Embedded Systems Letters*, 2(4):111–114, Dec 2010. ISSN 1943-0671. doi: 10.1109/LES.2010.2083632.
- [57] Abhijit Dhandha and Anju S Pillai. Performance evaluation of primary backup fault tolerant scheduling for single processor systems. In *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 866–871, June 2018. doi: 10.1109/ICCONS.2018.8663173.
- [58] Weiwei Jiang. Software defined satellite networks: A survey. *Digital Communications and Networks*, 9(6):1243–1264, 2023. ISSN 2352-8648. doi: <https://doi.org/10.1016/j.dcan.2023.01.016>. URL <https://www.sciencedirect.com/science/article/pii/S2352864823000299>.
- [59] Katy Mersmann. Dellinger: The little cubesat that could. *NASA Press Release*, October 2018. URL <https://www.nasa.gov/solar-system/dellinger-the-little-cubesat-that-could/>.
- [60] Dawei Li, Shuo Jiang, Xuelong Hou, Linjie Li, Yu Fu, and Lei Wang. Optimization design and application verification of a reset timing control circuit for s3r. In *2022 IEEE International Power Electronics and Application Conference and Exposition (PEAC)*, pages 653–657, Nov 2022. doi: 10.1109/PEAC56338.2022.9959489.
- [61] Chen Qi, Wang Fanyu, He Jinyu, Zhao Yang, Wang Heng, Huang Qichang, and Lei Minjie. Research on the fpga reconfiguration technology based on rs code for nuclear applications. In *2024 Global Reliability and Prognostics and Health Management Conference (PHM-Beijing)*, pages 1–6, Oct 2024. doi: 10.1109/PHM-Beijing63284.2024.10874746.
- [62] Amarendra Edpuganti, Vinod Khadkikar, Mohamed Shawky El Moursi, Hatem Zeineldin, Naji Al-Sayari, and Khalifa Al Hosani. A comprehensive review on cubesat electrical power system architectures. *IEEE Transactions on Power Electronics*, 37(3): 3161–3177, March 2022. ISSN 1941-0107. doi: 10.1109/TPEL.2021.3110002.
- [63] Harold A. Zahl and Hans K. Ziegler. Power sources for satellites and space vehicles. *Solar Energy*, 4(1):32–38, 1960. ISSN 0038-092X. doi: [https://doi.org/10.1016/0038-092X\(60\)90047-5](https://doi.org/10.1016/0038-092X(60)90047-5). URL <https://www.sciencedirect.com/science/article/pii/0038092X60900475>.
- [64] Martin Langer and Jasper Bouwmeester. Reliability of cubesats statistical data, developers’ beliefs and the way forward. 08 2016.
- [65] Timothy M. Lim, Aaron M. Cramer, James E. Lumpp, and Samir A. Rawashdeh. A modular electrical power system architecture for small spacecraft. *IEEE Transactions on Aerospace and Electronic Systems*, 54(4):1832–1849, Aug 2018. ISSN 1557-9603. doi: 10.1109/TAES.2018.2803598.

- [66] Ignacio F. Acero, Jonathan Diaz, Ronald Hurtado-Velasco, Sergio Ramiro Gonzalez Bautista, Sonia Rincón, Francisco L. Hernández, Julian Rodriguez-Ferreira, and Jesus Gonzalez-Llorente. A method for validating cubesat satellite eps through power budget analysis aligned with mission requirements. *IEEE Access*, 11:43316–43332, 2023. ISSN 2169-3536. doi: 10.1109/ACCESS.2023.3271596.
- [67] Jian Chen, Tianyu Feng, Xueqin Chen, and Huayi Li. A fully distributed electrical power system for small satellites. In *2023 IEEE Fifth International Conference on DC Microgrids (ICDCM)*, volume Single, pages 1–6, Nov 2023. doi: 10.1109/ICDCM54452.2023.10433621.
- [68] Bayan Hussein, Ahmed M. Massoud, and Tamer Khattab. Centralized, distributed, and module-integrated electric power system schemes in cubesats: Performance assessment. *IEEE Access*, 10:55396–55407, 2022. ISSN 2169-3536. doi: 10.1109/ACCESS.2022.3176902.
- [69] Sanae Dahbi, Abdelhak Aziz, Smail Zouggar, Mohamed El Hafyani, Oujda Morocco, Ahmed Hanafi, Mohammed Karim, Ibtissam Latachi, and Tajjeeddine Rachidi. Power budget analysis for a leo polar orbiting nano-satellite. In *2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, pages 1–6, May 2017. doi: 10.1109/ATSIP.2017.8075580.
- [70] Ali Jihad Ali, Mohsen Khalily, Ata Sattarzadeh, Ahmed Massoud, Mazen O. Hasna, Tamer Khattab, Okan Yurduseven, and Rahim Tafazolli. Power budgeting of leo satellites: An electrical power system design for 5g missions. *IEEE Access*, 9:113258–113269, 2021. ISSN 2169-3536. doi: 10.1109/ACCESS.2021.3104098.
- [71] NASA Small Spacecraft Technology Program. State of the art of small spacecraft technology. Technical report, National Aeronautics and Space Administration, December 2024. URL <https://www.nasa.gov/smallsat-institute/sst-soa/power-subsystems/#3.2>.
- [72] Eliad Peretz. The importance of solar power technology adaptations to space exploration. *IEEE*, 2024.
- [73] Collin R. Brown, Vincent R. Whiteside, Dmitry Poplavskyy, Khalid Hossain, Mangal S. Dhoubhadel, and Ian R. Sellers. Flexible cu(in,ga)se₂ solar cells for outer planetary missions: Investigation under low-intensity low-temperature conditions. *IEEE Journal of Photovoltaics*, 9(2):552–558, March 2019. ISSN 2156-3403. doi: 10.1109/JPHOTOV.2018.2889179.
- [74] Jing Jun Soon, Jiun Wei Chia, Htet Aung, Jia Min Lew, Shu Ting Goh, and Kay-Soon Low. A photovoltaic model based method to monitor solar array degradation on-board a microsatellite. *IEEE Transactions on Aerospace and Electronic Systems*, 54(5):2537–2546, Oct 2018. ISSN 1557-9603. doi: 10.1109/TAES.2018.2822081.

- [75] Qing Li, Shangguang Wang, Xiao Ma, Ao Zhou, Yue Wang, Gang Huang, and Xuanzhe Liu. Battery-aware energy optimization for satellite edge computing. *IEEE Transactions on Services Computing*, 17(2):437–451, March 2024. ISSN 1939-1374. doi: 10.1109/TSC.2024.3359233.
- [76] Keith B. Chin, Erik J. Brandon, Ratnakumar V. Bugga, Marshall C. Smart, Simon C. Jones, Frederick C. Krause, William C. West, and Gary G. Bolotin. Energy storage technologies for small satellite applications. *Proceedings of the IEEE*, 106(3):419–428, March 2018. ISSN 1558-2256. doi: 10.1109/JPROC.2018.2793158.
- [77] Abderezak Lashab, Mohammad Yaqoob, Yacine Terriche, Juan C. Vasquez, and Josep M. Guerrero. Space microgrids: New concepts on electric power systems for satellites. *IEEE Electrification Magazine*, 8(4):8–19, Dec 2020. ISSN 2325-5889. doi: 10.1109/MELE.2020.3026436.
- [78] Elisa Mostacciuolo, Silvio Baccari, Salvatore Sagnelli, Luigi Iannelli, and Francesco Vasca. An optimization approach for electrical power system supervision and sizing in low earth orbit satellites. *IEEE Access*, 12:151864–151875, 2024. ISSN 2169-3536. doi: 10.1109/ACCESS.2024.3473690.
- [79] C. Cougnet, E. Sein, A. Celeste, and Leopold Summerer. Solar power satellites for space exploration and applications. 567:151, 11 2004.
- [80] Angela Cratere, Leandro Gagliardi, Gabriel A. Sanca, Federico Golmar, and Francesco Dell’Olio. On-board computer for cubesats: State-of-the-art and future trends. *IEEE Access*, 12:99537–99569, 2024. ISSN 2169-3536. doi: 10.1109/ACCESS.2024.3428388.
- [81] Bharadwaj Chintalapati, Arthur Precht, Sougata Hanra, Rene Laufer, Marcus Liwicki, and Jens Eickhoff. Opportunities and challenges of on-board ai-based image recognition for small satellite earth observation missions. *Advances in Space Research*, 75(9):6734–6751, 2025. ISSN 0273-1177. doi: <https://doi.org/10.1016/j.asr.2024.03.053>. URL <https://www.sciencedirect.com/science/article/pii/S0273117724002886>. Science and applied research with small satellites.
- [82] Christo A. Lara, Maximiliano Fragoso, Luis Manuel Juárez, Leonardo Barboni, Rigoberto Reyes, Ricardo Vázquez, Julio Pérez Acle, and Saúl de la Rosa. Fault tolerant architecture design of a cubesat command and data handling system. In *2023 IEEE 24th Latin American Test Symposium (LATS)*, pages 1–6, March 2023. doi: 10.1109/LATS58125.2023.10154496.
- [83] Rebekah A. Austin, Brian D. Sierawski, James M. Trippe, Andrew L. Sternberg, Kevin M. Warren, Robert A. Reed, Robert A. Weller, Ronald D. Schrimpf, Michael L. Alles, Lloyd W. Massengill, Daniel M. Fleetwood, Gerald W. Buxton, Jonathan C. Brandenburg, W. Burns Fisher, and Robert Davis. Radfxsat: A flight campaign for recording single-event effects in commercial off-the-shelf microelectronics. In *2017*

- 17th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, pages 1–5, Oct 2017. doi: 10.1109/RADECS.2017.8696195.
- [84] Robert Merl and Paul Graham. A low-cost, radiation-hardened single-board computer for command and data handling. In *2016 IEEE Aerospace Conference*, pages 1–8, March 2016. doi: 10.1109/AERO.2016.7500849.
 - [85] Tyler M. Lovelly, Travis W. Wise, Shaun H. Holtzman, and Alan D. George. Benchmarking analysis of space-grade central processing units and field-programmable gate arrays. *Journal of Aerospace Information Systems*, 15(8):518–529, 2018. doi: 10.2514/1.I010621. URL <https://doi.org/10.2514/1.I010621>.
 - [86] Brock J LaMeres. *Introduction to Logic Circuits & Logic Design with VHDL*. Springer, 1 edition, 2017.
 - [87] Paul Hershey, Bill Wolpe, Jeffrey Klein, and Charlotte Dekeyrel. System for small satellite onboard processing. In *2017 Annual IEEE International Systems Conference (SysCon)*, pages 1–6, April 2017. doi: 10.1109/SYSCON.2017.7934705.
 - [88] Shubham Agarwal, Rajiv Bhatia, N Padmavathi, S Sudhakar, and S Udupa. System on chip approach for onboard computer. In *2013 IEEE International Conference on Signal Processing, Computing and Control (ISPCC)*, pages 1–6, Sep. 2013. doi: 10.1109/ISPCC.2013.6663446.
 - [89] *AMD Zynq 7000 SoCs*. Advanced Micro Devices, 2025. URL <https://www.amd.com/en/products/adaptive-socs-and-fpgas/soc/zynq-7000.html>. Accessed: Aug 2, 2025.
 - [90] *Microchip PolarFire SoC FPGAs*. Microchip, 2025. URL <https://www.microchip.com/en-us/products/fpgas-and-plds/system-on-chip-fpgas/polarfire-soc-fpgas>. Accessed: Aug 2, 2025.
 - [91] Alan D. George and Christopher M. Wilson. Onboard processing with hybrid and reconfigurable computing on small satellites. *Proceedings of the IEEE*, 106(3):458–470, March 2018. ISSN 1558-2256. doi: 10.1109/JPROC.2018.2802438.
 - [92] David Petrick, Alessandro Geist, Dennis Albaijes, Milton Davis, Pietro Sparacino, Gary Crum, Robin Ripley, Jonathan Boblitt, and Thomas Flatley. Spacecube v2.0 space flight hybrid reconfigurable data processing system. In *2014 IEEE Aerospace Conference*, pages 1–20, March 2014. doi: 10.1109/AERO.2014.6836226.
 - [93] Nicholas G. Franconi, Alan D. George, Alessandro D. Geist, and Dennis Albaijes. Signal and power integrity design methodology for high-performance flight computing systems. In *2021 IEEE Space Computing Conference (SCC)*, pages 27–38, Aug 2021. doi: 10.1109/SCC49971.2021.00011.

- [94] Caleb Adams, Allen Spain, Jackson Parker, Matthew Hevert, James Roach, and David Cotten. Towards an integrated gpu accelerated soc as a flight computer for small satellites. In *2019 IEEE Aerospace Conference*, pages 1–7, March 2019. doi: 10.1109/AERO.2019.8741765.
- [95] AAC Clyde Space. Kryten-m3 command & data handling datasheet. Technical report, AAC Clyde Space, October 2021. Release date: 28 July 2020. Available at https://www.aac-clyde.space/wp-content/uploads/2021/10/AAC_DataSheet_Kryten.pdf.
- [96] AAC Clyde Space. Sirius OBC LEON3FT Command & Data Handling Datasheet. Technical report, AAC Clyde Space, July 2020. URL https://www.aac-clyde.space/wp-content/uploads/2021/10/AAC_DataSheet_Sirius-OBC.pdf. Accessed: 2025-09-15.
- [97] Alén Space. DARA OBC On-Board Computer Datasheet. Technical Report 1.0, Alén Space, November 2024. URL <https://info.alen.space/hubfs/offers/AL-DARA%20OBC-DTS.pdf>. Accessed: 2025-09-15.
- [98] NXP Semiconductors. QorIQ T1040/20 and T1042/22 Communications Processors Datasheet. Technical Report T1FAMILYFS REV 2, NXP Semiconductors, 2016. URL <https://www.nxp.com/docs/en/fact-sheet/T1FAMILYFS.pdf>. Accessed: 2025-09-15.
- [99] Arm Limited. The Cortex-A9 Processor. Web, 2025. URL <https://developer.arm.com/documentation/den0018/a/NEON-Microarchitecture/The-Cortex-A9-processor>. Accessed: 2025-09-15.
- [100] Arm Limited. Arm Cortex-M7 Processor Datasheet. Technical report, Arm Limited, 2020. URL <https://www.arm.com/-/media/Arm%20Developer%20Community/PDF/Processor%20Datasheets/Arm-Cortex-M7-Processor-Datasheet.pdf>. Accessed: 2025-09-15.
- [101] Argotec. FERMI OBC&DH Miniaturized Rad-Hard On-Board Computer & Data Handling Unit Datasheet. Technical Report Version 2, Argotec, 2025. URL https://d2c3t1s1jzmfzb7.cloudfront.net/media/version2/Filtered-argotec-assets/technical-sheets/v3/Exe_ARGOTEC_TECHNICAL_SHEET_OBC_FERMI_vers2.pdf. Accessed: 2025-09-15.
- [102] Argotec. FERMI OBC&DH Miniaturized Rad-Hard On-Board Computer & Data Handling Unit Datasheet. Technical Report Version 2, Argotec, 2025. URL https://satcatalog.s3.amazonaws.com/components/422/SatCatalog_-_Argotec_-_FERMI_OBC_-_Datasheet.pdf?lastmod=20210708055240. Accessed: 2025-09-15.

- [103] Berlin Space Technologies. CDH-110 Command and Data Handling System Datasheet. Technical Report v. 1.0.0, Berlin Space Technologies, 2022. URL https://satcatalog.s3.amazonaws.com/components/418/SatCatalog_-_Berlin_Space_Technologies_-_CDH-110_-_Datasheet.pdf. Accessed: 2025-09-15.
- [104] Berlin Space Technologies. CDH-110 On-Board Computer and Attitude Control Computer. Web, 2025. URL <https://www.berlin-space-tech.com/cdh110/>. Accessed: 2025-09-15.
- [105] C3S Electronics Development LLC. On-Board Computer (OBC) Datasheet. Technical Report v01, C3S Electronics Development LLC, January 2022. URL https://c3s.hu/wp-content/uploads/2022/01/C3S_OBC_datasheet.pdf. Accessed: 2025-09-15.
- [106] CesiumAstro, Inc. SBC-1461 Single-Board Computer Datasheet. Technical Report Hardware Revision A Datasheet Version 072920, CesiumAstro, Inc., 2020. URL https://satcatalog.s3.amazonaws.com/components/426/SatCatalog_-_CesiumAstro_-_SBC-1461_-_Datasheet.pdf. Accessed: 2025-09-15.
- [107] EnduroSat. On-board computer (obc) type i datasheet. Technical Report VSSDFS, EnduroSat, July 2019. URL https://satcatalog.s3.amazonaws.com/components/417/SatCatalog_-_EnduroSat_-_Onboard_Computer_Type_I_-_Datasheet.pdf?lastmod=20210708055038. Revision 1.1, October 2019.
- [108] GOMSpace. Nanomind hp mk3 datasheet. Technical Report DS 1036210 2.1, GOMSpace, March 2023. URL [https://gomspace.com/UserFiles/Subsystems/datasheet/gs-ds-NanoMind_HP_MK3_\(DS_-_1036210_-_1_-_21\)_-_1.PDF](https://gomspace.com/UserFiles/Subsystems/datasheet/gs-ds-NanoMind_HP_MK3_(DS_-_1036210_-_1_-_21)_-_1.PDF). Accessed: 2025-09-16.
- [109] Xilinx, Inc. Defense-grade zynq-7000q soc data sheet: Overview. Technical Report DS196, Xilinx, Inc., July 2015. URL https://docs.amd.com/api/khub/documents/FSoAKRhQz_CuBUZXdQykIw/content. Version 1.3.1.
- [110] Ibeos. Edge-1100 payload processor. Technical report, March 2024. URL <https://www.ibeos.com/edge-1100>. Accessed: 2025-09-16.
- [111] Innoflight. Champs flight computer (mpsoc cfc-400). Technical report, January 2019. URL https://satcatalog.s3.amazonaws.com/components/434/SatCatalog_-_Innoflight_-_CFC-400_-_Datasheet.pdf?lastmod=20210708055852. Accessed: 2025-09-16.
- [112] KP Labs. Antelope dpu. Technical report, May 2025. URL https://cdn.prod.website-files.com/667174a549d8e1e151191810/6821e0a55183029c23f20f0d_Antelope_DPU_Data_Sheet.pdf. Accessed: 2025-09-16.
- [113] NARA Space Technology. Nstobc-a9 datasheet. Technical report, March 2024. URL https://satcatalog.s3.amazonaws.com/components/1340/SatCatalog_-_

- NARA_SPACE_-_NST_OBC_-_Datasheet.pdf?lastmod=20240326064536. Accessed: 2025-09-16.
- [114] NOVI, LLC. Sp240 on-board computer: Enabling edge computing in space. Technical report, NOVI, LLC, Arlington, VA, 2025. URL https://www.novispace.ai/_files/ugd/b76bcf_9e3d1534f355479f800d5e2e14f54d34.pdf. Document Version 1.0.
 - [115] Novo Space Co. Sbc002sv datasheet. Technical Report SBC002SV, Novo Space Co, July 2024. Version 2024-07-22.
 - [116] PassMark Software. Arm cortex-a53 4 core 1800 mhz benchmark, 2025. URL <https://www.cpubenchmark.net/cpu.php?cpu=ARM+Cortex-A53+4+Core+1800+MHZ&id=4128>. Data updated as of September 16, 2025.
 - [117] SkyLabs. Nanohpc-obc product page, 2025. URL <https://www.skylabs.si/products/nanohpc-obc/>. Accessed on September 16, 2025.
 - [118] Space Products and Innovation (SPiN). Ma61c cubesat multipurpose adapter generic interface connector datasheet. Technical report, Space Products and Innovation (SPiN), 2025. URL https://www.spinintech.com/_files/ugd/db39c1_a72a5fbd6f5241499dd5905414585042.pdf. Accessed on September 16, 2025.
 - [119] Spiral Blue. Se-1 space edge one. Technical report, Spiral Blue, January 2024. URL https://www.spiralblue.space/_files/ugd/800665_02ac3932016941fe8f3a34762ac2df6e.pdf. Version Jan 2023.
 - [120] Southwest Research Institute. High-performance single-board computer, 2025. URL <https://www.swri.org/markets/earth-space/space-research-technology/space-engineering/single-board-computers>. Accessed on September 16, 2025.
 - [121] Spacemanic. Eddie obc datasheet. Technical report, Spacemanic, March 2025. URL https://spacemanic.com/wp-content/uploads/2025/08/SM-EDI-DS-0002_3.0_Data-Sheet.pdf.
 - [122] Unibap. Spacecloud ix10-100. Technical report, Unibap, 2025. URL <https://unibap.com/wp-content/uploads/2023/10/spacecloud-ix10-em-product-overview-v15.pdf>.
 - [123] Xiphos. Q8 specifications. Technical report, Xiphos, 2025. URL <https://xiphos.com/hubfs/Xiphos-2023/PDFs/XTI-2001-2024-i-Q8-Rev-C-Spec-Sheet.pdf?hsLang=en>.
 - [124] The White House. Harnessing the small satellite revolution to promote innovation and entrepreneurship in space. website, October 2016. URL <https://obamawhitehouse.archives.gov/the-press-office/2016/10/21/harnessing-small-satellite-revolution-promote-innovation-and>.

- [125] Nano Avionics. How much do cubesats and smallsats cost? website, . URL <https://nanoavionics.com/blog/how-much-do-cubesats-and-smallsats-cost/>.
- [126] Nano Avionics. How many satellites are in space? website, . URL <https://nanoavionics.com/blog/how-many-satellites-are-in-space/>.
- [127] David A. Bearden. Small-satellite costs. *Crosslink*, pages 33–44, January 2001.
- [128] Nicholas Crisp, Katharine Smith, and Peter Hollingsworth. Launch and deployment of distributed small satellite systems. *Acta Astronautica*, 29, 04 2015. doi: 10.1016/j.actaastro.2015.04.015.
- [129] SatCatalog. Cubesat launch costs. website, December 2022. URL <https://www.satcatalog.com/insights/cubesat-launch-costs/>.
- [130] Mike Wall. SpaceX launches 19 starlink satellites from california, lands rocket on ship at sea (video). website, July 2025.
- [131] Casey Dreier. An improved cost analysis of the apollo program. *Space Policy*, 60:101476, 2022. ISSN 0265-9646. doi: <https://doi.org/10.1016/j.spacepol.2022.101476>. URL <https://www.sciencedirect.com/science/article/pii/S0265964622000029>.
- [132] Harry W. Jones. The impact of lower launch cost on space life support. In *AIAA Space and Astronautics Forum*, September 2017. URL <https://ntrs.nasa.gov/citations/20180007067>.
- [133] Melvin A. Broder, Eric M. Mahr, Daniel E.F. Barkmeyer, Erik L. Burgess, Wilmer Alvarado, Samuel B. Toas, and Gregory Hogan. Review of three small-satellite cost models. In *2010 ISPA/SCEA Joint Annual Conference and Training Workshop*, 2010.
- [134] Luis Alberto Aranda, Pedro Reviriego, and Juan Antonio Maestro. Toward a fault-tolerant star tracker for small satellite applications. *IEEE Transactions on Aerospace and Electronic Systems*, 56(5):3421–3431, Oct 2020. ISSN 1557-9603. doi: 10.1109/TAES.2020.2971289.
- [135] Samir A. Rawashdeh and James E. Lumpp. Image-based attitude propagation for small satellites using ransac. *IEEE Transactions on Aerospace and Electronic Systems*, 50(3):1864–1875, July 2014. ISSN 1557-9603. doi: 10.1109/TAES.2014.120808.
- [136] D. Martinez, J. M. Quero, J. García, L. León, P. Castro, and C. Nieto. Attitude determination module for cubesat. In *2015 IEEE International Conference on Industrial Technology (ICIT)*, pages 1979–1983, March 2015. doi: 10.1109/ICIT.2015.7125386.
- [137] Stevan M. Spremo, Alan R. Crocker, and Tina L. Panontin. Small spacecraft overview, 2017.

- [138] Stephen A. Jacklin. Small-satellite mission failure rates, March 2019.
- [139] Kaiqi Xu, Mario Brito, Charlie Young, M.K.S Al-Mhdawi, Nicholas Dacre, and David Baxter. Statistical analysis and modeling of small satellite reliability, January 2024.
- [140] P. C. Adell and L. Z. Scheick. Radiation effects in power systems: A review. *IEEE Transactions on Nuclear Science*, 60(3):1929–1952, June 2013. ISSN 1558-1578. doi: 10.1109/TNS.2013.2262235.
- [141] Christian Fuchs, Florian Moll, Dirk Giggenbach, Christopher Schmidt, Jonas Keim, and Steffen Gaisser. Osirisv1 on flying laptop: Measurement results and outlook. In *2019 IEEE International Conference on Space Optical Systems and Applications (ICSOS)*, pages 1–5, Oct 2019. doi: 10.1109/ICSOS45490.2019.8978984.
- [142] F. Huber, P. Behr, H.-P Röser, and S. Pletner. Fpga based on-board computer system for the "flying laptop" micro-satellite. pages 38–, 08 2007.
- [143] Sabine Klinkner. Flying laptop. URL <https://www.irs.uni-stuttgart.de/en/research/satellitetechnology-and-instruments/smallsatelliteprogram/flying-laptop/>.
- [144] Toshinori Kuwahara. *FPGA-based Reconfigurable On-board Computing Systems for Space Applications*. Phd thesis, Institute of Space Systems: Universitat Stuttgart, Stuttgart, Germany, October 2009.
- [145] Ádria Barros de Oliveira, Johan Hellström, and Lucas Antunes Tambara. Proton see characterization of nexus certuspro-nx fpga implementing grlib leon3ft-based system-on-chip. In *2024 RADECS Data Workshop*, pages 1–5, Sep. 2024. doi: 10.1109/RADECS61975.2024.11017547.
- [146] Phillip Hagel, Jonas Burgdorf, and Sebastian Wenzel. The flying laptop satellite as a low-threshold testbed for computing in orbit. In *ARCS Workshop 2018; 31th International Conference on Architecture of Computing Systems*, pages 1–5, April 2018.
- [147] Herbert J. Kramer. Gomx-3 (gomspace express-3). <https://www.eoportal.org/satellite-missions/gomx-3>, November 2015. Last updated: November 2, 2015 (with mission status updates up to April 2017). eoPortal, European Space Agency. Accessed: August 23, 2025.
- [148] Atmel Corporation. 32-bit AVR Microcontroller AT32UC3C Series Datasheet. Technical Report 32117D-AVR-01/12, Atmel Corporation, January 2012. Available at: <https://ww1.microchip.com/downloads/en/DeviceDoc/doc32117.pdf>, Accessed: August 23, 2025.
- [149] GomSpace A/S. NanoMind A3200 Datasheet. Technical Report 1006901, GomSpace A/S, January 2021. Revision 1.17, Approved February 17,

2021. Available at: https://gomspace.com/UserFiles/Subsystems/datasheet/gs-ds-nanomind-a3200_1006901-117.pdf, Accessed: August 23, 2025.
- [150] Gunter Krebs. PhoneSat 1.0. *Gunter's Space Page*, 2023. URL https://space.skyrocket.de/doc_sdat/phonesat-v1.htm. Accessed: August 23, 2025.
- [151] NASA Ames Research Center. PhoneSat - The Smartphone Nanosatellite. *NASA Technical Reports Server (NTRS)*, 2014. URL <https://ntrs.nasa.gov/citations/20140005553>. Document ID: 20140005553, Accessed: August 23, 2025.
- [152] EnduroSat. EnduroSat One Mission. *EnduroSat One Mission Website*, 2023. URL <https://one.endurosat.com/#the-mission>. Accessed: August 23, 2025.
- [153] EnduroSat. On-Board Computer (OBC) Type I Datasheet. Technical report, EnduroSat, July 2021. Available at: https://satcatalog.s3.amazonaws.com/components/417/SatCatalog_-_EnduroSat_-_Onboard_Computer_Type_I_-_Datasheet.pdf?lastmod=20210708055038, Accessed: August 23, 2025.
- [154] ARC Training Centre for CubeSats, UAVs & Their Applications. CUAVA-1 Mission. <https://www.cuava.com.au/projects/cuava-1/>, 2021. Accessed: August 23, 2025.
- [155] Xueliang Bai, Iver H. Cairns, Patrick Oppel, Youngho Eun, Anthony Monger, Christopher H. Betters, Sergio Leon-Saval, Joe Khachan, James Harpur, Xiaofeng Wu, Yijun Huang, Byung Hoon Cho, Chung Hang Chu, Alex Gaoshan Lian, David Byrne, Flynn Byrant, Daniel Roger Hussey, Mitchell William Galletly, Ada Jiexin Song, Andrew G. Dempster, Joon Wayne Cheong, Ediz Cetin, Tong Wu, Oliver Diessel, Jason Held, Andreas Antoniadis, Alix Murdoch, James Unicomb, and Paris Michaels. The cuava-1 cubesat—a pathfinder satellite for remote sensing and earth observation. In *Proceedings of the AIAA/USU Conference on Small Satellites 2021*, United States, 2021. University of Utah.
- [156] Innovative Solutions In Space. Isis on-board computer (obc) datasheet. Technical report, Innovative Solutions In Space (ISIS), 2021.
- [157] Nanosats Database. Cuava-1 spacecraft. <https://www.nanosats.eu/sat/cuava-1>, 2024. Last modified: December 29, 2024, Accessed: August 23, 2025.
- [158] NASA. Mars cube one (marco). <https://science.nasa.gov/mission/marco/>, 2023. Accessed: August 23, 2025.
- [159] David Sternberg, John Essmiller, Cody Colley, Andrew Klesh, and Joel Krajewski. Attitude control system for the mars cube one spacecraft. In *2019 IEEE Aerospace Conference*, pages 1–10, March 2019. doi: 10.1109/AERO.2019.8741816.
- [160] M. Michael Kobayashi, Mazen Shihabi, and Jim Taylor. Mars cube one telecommunications subsystem design. *Desoanso Design and Performance Summary Series*, Sep 2021.

- [161] NASA Jet Propulsion Laboratory. Mars cube one (marco). <https://www.jpl.nasa.gov/missions/mars-cube-one-marco/>, 2023. Accessed: August 23, 2025.
- [162] Herbert J. Kramer. Marco (mars cube one). <https://www.eoportal.org/satellite-missions/marco>, March 2018. Last updated: March 19, 2018. eoPortal, European Space Agency. Accessed: August 23, 2025.
- [163] Aeroflex. Ut699 32-bit fault-tolerant sparcc tm v8/leon 3ft processor. Technical report, Aeroflex, July 2009.
- [164] EnduroSat. Onboard Computer for CubeSats and SmallSats. <https://www.endurosat.com/products/onboard-computer/>, 2023. Accessed: August 23, 2025.
- [165] Carlos Molina Ordonez. Design and implementation of a single on-board computer for cubesats. masters thesis, Universitat Politecnica de Catalunya, Barcelona, Spain, September 2019.
- [166] Texas Instruments. MSP430F261x, MSP430F241x Mixed-Signal Microcontrollers Datasheet. Technical report, Texas Instruments, 2024.
- [167] Steven M. Guertin, Mehran Amrbar, and Sergeh Vartanian. Radiation test results for common cubesat microcontrollers and microprocessors. In *2015 IEEE Radiation Effects Data Workshop (REDW)*, pages 1–9, July 2015. doi: 10.1109/REDW.2015.7336730.
- [168] Waseem Sajjad, Arooj Shafique, and Rehan Mahmood. Designing of reliable, low-power, and performance-efficient onboard computer architecture for cubesats. *IEEE Journal on Miniaturization for Air and Space Systems*, 5(2):59–72, June 2024. ISSN 2576-3164. doi: 10.1109/JMASS.2023.3342208.
- [169] Björn Osterloh, Harald Michalik, Sandi Alexander Habinc, and Björn Fiethe. Dynamic partial reconfiguration in space applications. In *2009 NASA/ESA Conference on Adaptive Hardware and Systems*, pages 336–343, July 2009. doi: 10.1109/AHS.2009.13.
- [170] Ken Peffers, Tuure Tuunanen, Marcus Rothenberger, and S. Chatterjee. A design science research methodology for information systems research. *Journal of Management Information Systems*, 24:45–77, 01 2007.
- [171] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS Quarterly*, 28(1):75–105, mar 2004.
- [172] Alan Hevner. A three cycle view of design science research. *Scandinavian Journal of Information Systems*, 19, 01 2007.
- [173] Zhuo Ruan, Yuzhang Han, Hongbo Cai, Shengzhen Jin, and Jianguo Han. A dynamically partial-reconfigurable fpga-based architecture for data processing on space solar telescope. In *2007 International Symposium on Industrial Embedded Systems*, pages 194–199, July 2007. doi: 10.1109/SIES.2007.4297335.

- [174] Simen Eine, Dennis Langer, Roger Birkeland, and Milica Orlandić. Reconfiguration of fpga during operation of small satellite for flexible hyperspectral data compression. 10 2024. doi: 10.52202/078365-0077.
- [175] W.J. Price. A benchmark tutorial. *IEEE Micro*, 9(5):28–43, Oct 1989. ISSN 1937-4143. doi: 10.1109/40.45825.
- [176] S. Kumar, L. Pires, D. Pandalai, M. Vokta, J. Golusky, S. Wadi, and H. Spaanenburg. Benchmarking technology for configurable computing system. In *Proceedings. IEEE Symposium on FPGAs for Custom Computing Machines (Cat. No.98TB100251)*, pages 273–274, April 1998. doi: 10.1109/FPGA.1998.707913.
- [177] L. Shannon and P. Chow. Standardizing the performance assessment of reconfigurable processor architectures. In *11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, 2003. FCCM 2003.*, pages 282–283, April 2003. doi: 10.1109/FPGA.2003.1227271.
- [178] Doug Sinclair and Jonathan Dyer. Ssc13-iv-3 radiation effects and cots parts in smallsats. 04 2021.
- [179] Sunday C. Ekpo and Danielle George. A system engineering analysis of highly adaptive small satellites. *IEEE Systems Journal*, 7(4):642–648, Dec 2013. ISSN 1937-9234. doi: 10.1109/JSYST.2012.2198138.
- [180] Scott Sterling Arnold, Ryan Nuzzaci, and Ann Gordon-Ross. Energy budgeting for cubesats with an integrated fpga. In *2012 IEEE Aerospace Conference*, pages 1–14, March 2012. doi: 10.1109/AERO.2012.6187240.
- [181] Sunday C. Ekpo and Danielle George. A system engineering consideration for future-generations small satellites design. In *2012 IEEE First AESS European Conference on Satellite Telecommunications (ESTEL)*, pages 1–6, Oct 2012. doi: 10.1109/ESTEL.2012.6400067.
- [182] BAE Systems. Rad750 3u compactpci single-board computer. Technical report, BAE Sytems, 2025.
- [183] BAE Systems. Processors bae systems’ current processors and single board computers. Technical report, BAE Sytems, 2025.
- [184] MicroChip. Pic64-hpsc series pic64-hpsc1000 and pic64-hpsc1100. Technical report, MicroChip, 2025.
- [185] Wesley Powell. High-performance spaceflight computing (hpsc) project overview. Technical report, National Aeronautics and Space Administration, 2018.

- [186] Jairo Walber Abdala Castro and Aurelio Morales-Villanueva. Exploring dynamic partial reconfiguration in a tightly-coupled coprocessor attached to a risc-v soft-processor on a fpga. In *2021 IEEE XXVIII International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*, pages 1–4, Aug 2021. doi: 10.1109/INTERCON52678.2021.9532810.
- [187] Najdet Charaf, Ahmed Kamaleldin, Martin Thümmel, and Diana Göhringer. Rv-cap: Enabling dynamic partial reconfiguration for fpga-based risc-v system-on-chip. In *2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 172–179, June 2021. doi: 10.1109/IPDPSW52791.2021.00033.
- [188] Jimmy Tarrillo, Fernando A. Escobar, Fernanda Lima Kastensmidt, and Carlos Valderrama. Dynamic partial reconfiguration manager. In *2014 IEEE 5th Latin American Symposium on Circuits and Systems*, pages 1–4, Feb 2014. doi: 10.1109/LASCAS.2014.6820293.
- [189] B. Krill, A. Amira, A. Ahmad, and H. Rabah. A new fpga-based dynamic partial reconfiguration design flow and environment for image processing applications. In *2010 2nd European Workshop on Visual Information Processing (EUVIP)*, pages 226–231, July 2010. doi: 10.1109/EUVIP.2010.5699127.
- [190] Chakradhar V. Borkute, A.Y. Deshmukh, and Chetna N. Kharkar. Dynamic partial reconfiguration in fpgas for dsp applications. In *2011 Fourth International Conference on Emerging Trends in Engineering & Technology*, pages 253–257, Nov 2011. doi: 10.1109/ICETET.2011.70.
- [191] Christopher M. Major, Annie Bachman, Colter Barney, Skylar Tamke, and Brock J. LaMeres. Radpc: A novel single-event upset mitigation strategy for field programmable gate array-based space computing. *J. Aerosp. Inf. Syst.*, 18:280–288, 2021. URL <https://api.semanticscholar.org/CorpusID:233553128>.
- [192] Justin Williams, Colter Barney, Zachary Becker, Jake Davis, Chris Major, and Brock LaMeres. Radpc@scale: A novel approach to the radpc single event upset mitigation strategy. In *2022 IEEE Aerospace Conference (AERO)*, pages 1–7, March 2022. doi: 10.1109/AERO53065.2022.9843540.
- [193] Brock J. LaMeres and Clint Gauer. Dynamic reconfigurable computing architecture for aerospace applications. In *2009 IEEE Aerospace conference*, pages 1–6, March 2009. doi: 10.1109/AERO.2009.4839513.
- [194] Clint Gauer, Brock J. LaMeres, and David Racek. Spatial avoidance of hardware faults using fpga partial reconfiguration of tile-based soft processors. In *2010 IEEE Aerospace Conference*, pages 1–11, March 2010. doi: 10.1109/AERO.2010.5446663.
- [195] Jennifer Hane, Brock LaMeres, Todd Kaiser, Raymond Weber, and Todd Buerkle. Increasing radiation tolerance of field-programmable-gate-array-based computers

- through redundancy and environmental awareness. *Journal of Aerospace Information Systems*, 11:68–81, 02 2014. doi: 10.2514/1.I010106.
- [196] AMD. 7 series fpgas configuration user guide. Technical report, Advanced Micro Devices, December 2023.
 - [197] Kaushal Kumar, Ritesh Mishra, and Durgesh Nandan. Efficient hardware of rgb to gray conversion realized on fpga and asic. *Procedia Computer Science*, 171:2008–2015, 01 2020. doi: 10.1016/j.procs.2020.04.215.
 - [198] Justin A. Hogan, Raymond J. Weber, and Brock J. LaMeres. Reliability analysis of field-programmable gate-array-based space computer architectures. *Journal of Aerospace Information Systems*, 14(4), April 2017.
 - [199] Chris M. Major. *Novel Approach to Fault Tolerance in Space Computers Leverging the RISC-V Architecture*. Phd thesis, Montana State University, Bozeman, MT, May 2023.
 - [200] AMD. Soft error mitigation controller v4.1 logicore ip product guide. Technical report, AMD Xilinx, November 2023.
 - [201] A.J. Tylka, J.H. Adams, P.R. Boberg, B. Brownstein, W.F. Dietrich, E.O. Flueckiger, E.L. Petersen, M.A. Shea, D.F. Smart, and E.C. Smith. Creme96: A revision of the cosmic ray effects on micro-electronics code. *IEEE Transactions on Nuclear Science*, 44(6):2150–2160, Dec 1997. ISSN 1558-1578. doi: 10.1109/23.659030.
 - [202] Richard Berger, David Artz, and Paul Kapcio. Rad750 tm radiation hardened powerpc tm microprocessor. 10 2011.
 - [203] Philippe Adell and Greg Allen. Assessing and mitigating radiation effects in xilinx fpgas. 2008. URL https://nepp.nasa.gov/DocUploads/08A3733D-6D42-428C-824E76B5B5A92B59/07-113%20Adell_JPL%20Assessing%20and%20Mitigating%20Rad%20effects.pdf.
 - [204] Microsemi. Radiation-tolerant proasic3 low power spaceflight flash fpgas with flashfreeze technology. Technical report, MicroChip, March 2020.
 - [205] Martha V. O’Bryan, Kenneth A. LaBel, Jonathan A. Pellish, Jean-Marie Lauenstein, Dakai Chen, Cheryl J. Marshall, Timothy R. Oldham, Hak S. Kim, Anthony M. Phan, Melanie D. Berg, Michael J. Campola, Anthony B. Sanders, Paul W. Marshall, Michael A. Xapsos, David F. Heidel, Kenneth P. Rodbell, Jim W. Swonger, Don Alexander, Michael Gauthier, and Brian Gauthier. Recent single event effects compendium of candidate electronics for nasa space systems. In *2011 IEEE Radiation Effects Data Workshop*, pages 1–13, July 2011. doi: 10.1109/REDW.2010.6062500.

- [206] Stefano Barra. Functional testing of the electrical power system of the mist satellite. Master’s thesis, KTH Royal Institute of Technology, School of Engineering Sciences, Stockholm, Sweden, 2019. URL [https://s3vi.ndc.nasa.gov/ssri-kb/static/resources/FULLTEXT01%20\(2\).pdf](https://s3vi.ndc.nasa.gov/ssri-kb/static/resources/FULLTEXT01%20(2).pdf). TRITA-EECS-EX-2019:570.
- [207] Xilinx. Virtex-ii pro and virtex-ii pro x platform fpgas: Complete data sheet. Technical report, AMD, June 2018.
- [208] CAES. Ut699 leon 3ft /sparctm v8 microprocessor functional manual. Technical report, CAES, September 2018.
- [209] JPL. Mars cube one demo. https://www.jpl.nasa.gov/news/press_kits/insight/launch/appendix/mars-cube-one/. Accessed: October 3, 2025.
- [210] Hezekiah A. Austin, Chris Major, Zach Becker, Tristan Running Crane, Kris Allick, and Brock J. LaMeres. Dynamically reconfigurable coprocessor for floating-point arithmetic capability in small satellites. In *2025 IEEE Aerospace Conference*, pages 1–8, March 2025. doi: 10.1109/AERO63441.2025.11068533.
- [211] Hezekiah A. Austin, Chris Major, Kris Allick, Tristan Running Crane, Benjamin Macht, Clemente Izurieta, and Brock J. LaMeres. Real-time, dynamically reconfigurable coprocessor for image processing in small satellites. In *2025 IEEE SCC Conference*, August 2025.
- [212] Hezekiah Austin, Chris Major, Tristan Running Crane, Kris Allick, and Brock LaMeres. Dynamic partially reconfigurable coprocessor for low-power, radiation-tolerant computer. *Frontiers in Aerospace Engineering - Navigation, Guidance, and Control*, Under Review.

APPENDICES

APPENDIX A

AVERAGE PAIRWISE QUADRATIC WEIGHTED KAPPA PYTHON

This appendix contains the python code to perform an average pairwise quadratic weighted kappa on the data in Table 6.2 and generate the results in Table 6.3.

```

1 import numpy as np
2 import pandas as pd
3 from sklearn.metrics import cohen_kappa_score
4 from itertools import combinations
5
6 # =====
7 # 1. DEFINE YOUR ORDINAL SCALE
8 # =====
9 categories = {
10     'Poor': 1,
11     'Fair': 3,
12     'Good': 4,
13     'Excellent': 5
14 }
15
16 # =====
17 # 2. YOUR DATA: 5 raters x 9 items
18 # =====
19 n_raters = 5
20
21 # Comment out ratings_raw except for the one being evaluated
22
23 # Computational Performance Scores
24 ratings_raw = np.array([
25
26     [3, 3, 2, 2, 2, 1, 3, 3, 4], # Rater 1
27     [2, 1, 2, 3, 2, 1, 2, 3, 4], # Rater 2
28     [2, 3, 2, 3, 3, 1, 3, 3, 3], # Rater 3
29     [4, 3, 2, 4, 3, 2, 3, 4, 4], # Rater 4
30     [3, 2, 2, 4, 2, 1, 3, 3, 4]   # Rater 5
31 ])
32
33 # Radiation Tolerance Scores
34 ratings_raw = np.array([
35
36     [3, 1, 1, 2, 1, 3, 3, 4, 4], # Rater 1
37     [4, 3, 1, 3, 2, 2, 4, 3, 3], # Rater 2
38     [3, 2, 1, 2, 1, 2, 3, 4, 3], # Rater 3
39     [2, 3, 1, 3, 1, 2, 3, 4, 4], # Rater 4
40     [3, 2, 1, 3, 3, 2, 3, 4, 3]   # Rater 5
41 ])
42
43 # Power Consumption Scores

```

```

44 ratings_raw = np.array([
45
46     [4, 4, 2, 3, 4, 4, 3, 3, 3],      # Rater 1
47     [4, 4, 4, 3, 4, 4, 3, 3, 3],      # Rater 2
48     [4, 4, 4, 3, 4, 4, 2, 3, 3],      # Rater 3
49     [4, 4, 4, 4, 4, 4, 4, 4, 4],      # Rater 4
50     [4, 4, 4, 3, 4, 4, 3, 3, 3]      # Rater 5
51 ])
52
53 # Cost Scores
54 ratings_raw = np.array([
55
56     [2, 4, 4, 2, 3, 4, 3, 1, 1],      # Rater 1
57     [1, 4, 3, 1, 2, 4, 2, 1, 1],      # Rater 2
58     [2, 4, 3, 2, 2, 4, 3, 1, 1],      # Rater 3
59     [2, 4, 4, 2, 2, 4, 2, 1, 1],      # Rater 4
60     [3, 4, 4, 3, 3, 4, 3, 1, 2]      # Rater 5
61 ])
62
63 # =====
64 # 3. DISPLAY RATINGS
65 # =====
66 raters = [f"Rater {i+1}" for i in range(n_raters)]
67 items = [f"Item {i+1}" for i in range(9)]
68 df_ratings = pd.DataFrame(ratings_raw, index=raters, columns=items)
69 print("=== RATINGS (5 raters x 9 items) ===")
70 print(df_ratings)
71 print()
72
73 # =====
74 # 4. COMPUTE PAIRWISE QUADRATIC WEIGHTED COHEN'S KAPPA
75 # =====
76 kappa_matrix = np.zeros((n_raters, n_raters))
77 pairs = list(combinations(range(n_raters), 2))
78
79 for i, j in pairs:
80     rater_i = ratings_raw[i, :]
81     rater_j = ratings_raw[j, :]
82     kappa = cohen_kappa_score(rater_i, rater_j, weights='quadratic')
83     kappa_matrix[i, j] = kappa
84     kappa_matrix[j, i] = kappa
85
86 # Diagonal = 1.0 (perfect agreement with self)
87 np.fill_diagonal(kappa_matrix, 1.0)
88

```



```

89 kappa_df = pd.DataFrame(kappa_matrix, index=raters, columns=raters).
    round(4)
90 print("=== PAIRWISE QUADRATIC WEIGHTED COHEN'S KAPPA ===")
91 print(kappa_df)
92 print()
93
94 # =====
95 # 5. AVERAGE INTER-RATER RELIABILITY
96 # =====
97 avg_kappa = kappa_matrix[np.triu_indices(n_raters, k=1)].mean()
98 print(f"Average Pairwise Quadratic Weighted Kappa = {avg_kappa:.4f}"
    )
99
100 # Interpretation
101 if avg_kappa < 0:
102     print("Poor agreement")
103 elif avg_kappa < 0.20:
104     print("Slight agreement")
105 elif avg_kappa < 0.40:
106     print("Fair agreement")
107 elif avg_kappa < 0.60:
108     print("Moderate agreement")
109 elif avg_kappa < 0.80:
110     print("Substantial agreement")
111 else:
112     print("Almost perfect agreement")
113
114 # =====
115 # 6. OPTIONAL: SAVE TO CSV
116 # =====
117 kappa_df.to_csv("cohen_kappa_6raters_9items.csv")
118 print("\nResults saved to 'cohen_kappa_6raters_9items.csv'")

```

APPENDIX B

KRUSKAL WALLIS PYTHON

This appendix contains the python code to perform a Kruskal Wallis test on the data in Table 6.2 and generate the results in Table 6.4.

```

1 import numpy as np
2 import pandas as pd
3 from scipy.stats import kruskal, rankdata
4 from itertools import combinations
5
6 # =====
7 # 1. YOUR DATA (1=Poor, 2=Fair, 3=Good, 4=Excellent)
8 # =====
9 n_raters = 5
10
11 # Comment out ratings_raw except for the one being evaluated
12
13 # Computational Performance Scores
14 ratings_raw = np.array([
15     [3, 3, 2, 2, 2, 1, 3, 3, 4],      # Rater 1
16     [2, 1, 2, 3, 2, 1, 2, 3, 4],      # Rater 2
17     [2, 3, 2, 3, 3, 1, 3, 3, 3],      # Rater 3
18     [4, 3, 2, 4, 3, 2, 3, 4, 4],      # Rater 4
19     [3, 2, 2, 4, 2, 1, 3, 3, 4]       # Rater 5
20 ])
21
22
23 # Radiation Tolerance Scores
24 ratings_raw = np.array([
25     [3, 1, 1, 2, 1, 3, 3, 4, 4],      # Rater 1
26     [4, 3, 1, 3, 2, 2, 4, 3, 3],      # Rater 2
27     [3, 2, 1, 2, 1, 2, 3, 4, 3],      # Rater 3
28     [2, 3, 1, 3, 1, 2, 3, 4, 4],      # Rater 4
29     [3, 2, 1, 3, 3, 2, 3, 4, 3]       # Rater 5
30 ])
31
32
33 # Power Consumption Scores
34 ratings_raw = np.array([
35     [4, 4, 2, 3, 4, 4, 3, 3, 3],      # Rater 1
36     [4, 4, 4, 3, 4, 4, 3, 3, 3],      # Rater 2
37     [4, 4, 4, 3, 4, 4, 2, 3, 3],      # Rater 3
38     [4, 4, 4, 4, 4, 4, 4, 4, 4],      # Rater 4
39     [4, 4, 4, 3, 4, 4, 3, 3, 3]       # Rater 5
40 ])
41
42
43 # Cost Scores

```

```

44 ratings_raw = np.array([
45     [2, 4, 4, 2, 3, 4, 3, 1, 1],      # Rater 1
46     [1, 4, 3, 1, 2, 4, 2, 1, 1],      # Rater 2
47     [2, 4, 3, 2, 2, 4, 3, 1, 1],      # Rater 3
48     [2, 4, 4, 2, 2, 4, 2, 1, 1],      # Rater 4
49     [3, 4, 4, 3, 3, 4, 3, 1, 2]       # Rater 5
50 ])
51
52
53 # =====
54 # 2. MAP TO SCORES (optional - Kruskal works on ranks anyway)
55 # =====
56 code_to_score = {1: 0.50, 2: 0.75, 3: 0.85, 4: 1.00}
57 ratings_mapped = np.vectorize(code_to_score.get)(ratings_raw)
58
59 # =====
60 # 3. RESHAPE TO LONG FORMAT
61 # =====
62 df = pd.DataFrame({
63     'Rater': np.repeat([f'R{i+1}' for i in range(n_raters)], 9),
64     'Score': ratings_mapped.flatten()
65 })
66
67 # =====
68 # 4. KRUSKAL-WALLIS TEST
69 # =====
70 groups = [df[df['Rater'] == r]['Score'].values for r in df['Rater'].
71            unique()]
72 H, p = kruskal(*groups)
73
74 print(f"Kruskal-Wallis Test")
75 print(f"    H = {H:.4f}")
76 print(f"    p = {p:.4f}")
77
78 # =====
79 # 5. MANUAL DUNN'S POST-HOC (only if p < 0.05)
80 # =====
81 def dunn_posthoc(groups, alpha=0.05):
82     """Simple Dunn's test with Bonferroni correction"""
83     k = len(groups)
84     n = [len(g) for g in groups]
85     ranks = [rankdata(g) for g in groups]
86     R = [np.sum(r) for r in ranks]
87     N = sum(n)

```

```

88     p_values = {}
89     comparisons = list(combinations(range(k), 2))
90
91     for i, j in comparisons:
92         z = abs(R[i]/n[i] - R[j]/n[j]) / np.sqrt((N*(N+1)/12) * (1/n
93             [i] + 1/n[j]))
94         # Approximate p-value (standard normal)
95         from scipy.stats import norm
96         p_val = 2 * (1 - norm.cdf(z))
97         p_values[(i, j)] = p_val
98
99     # Bonferroni correction
100    bonferroni_alpha = alpha / len(comparisons)
101    significant = [(f"Rater {i+1} vs Rater {j+1}", p_values[(i,j)],
102        p_values[(i,j)] < bonferroni_alpha)
103        for i, j in comparisons]
104
105    return significant, p_values, bonferroni_alpha
106
107    # =====
108    # 6. RUN POST-HOC ONLY IF NEEDED
109    # =====
110    if p < 0.05:
111        print("\nSignificant - Running Dunn's post-hoc (Bonferroni
112            corrected)")
113        sig_pairs, all_p, adj_alpha = dunn_posthoc(groups)
114        print(f"    Adjusted Alpha = {adj_alpha:.5f}")
115        print("\n    Significant pairs (p < Alpha_adj):")
116        for pair, p_val, is_sig in sig_pairs:
117            star = " *" if is_sig else ""
118            print(f"    {pair}: p = {p_val:.4f}{star}")
119    else:
120        print("\nNot significant (p > 0.05) - No post-hoc needed.")
121
122    # =====
123    # 7. FINAL INTERPRETATION
124    # =====
125    print("\n" + "="*50)
126    if p < 0.05:
127        print("REJECT H_sigma: At least one rater differs significantly.
128            ")
129    else:
130        print("FAIL TO REJECT H_sigma: No evidence of systematic rater
131            differences.")
132    print("="*50)

```