RadPC@Scale: A Novel Approach to the RadPC Single Event Upset Mitigation Strategy

Justin Williams
Electrical Computer Engineering
Montana State University
Bozeman, Montana
justinwilliamsmsu@gmail.com

Colter Barney
Electrical Computer Engineering
Montana State University
Bozeman, Montana
colter.barney@student.montana.edu

Zachary Becker
Electrical Computer Engineering
Montana State University
Bozeman, Montana
bitbytebitco@gmail.com

Jake Davis
Electrical Computer Engineering
Montana State University
Bozeman, Montana
jakejdavis3@gmail.com

Chris Major
Electrical Computer Engineering
Montana State University
Bozeman, Montana
chrismichelmajor@hotmail.com

Brock LaMeres
Electrical Computer Engineering
Montana State University
Bozeman, Montana
lameres@montana.edu

2022 IEEE Aerospace Conference (AERO) | 978-1-6654-3760-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/AERO53065.2022.9843540

***Abstract*—This paper presents the flight test results of a single event upset (SEU) mitigation strategy for computer data memory. This memory fault mitigation strategy is part of a larger effort to build a radiation tolerant computing system using commercial-off-the-shelf (COTS) field programmable gate arrays (FPGAs) called RadPC. While previous iterations of RadPC used FPGA block RAM (BRAM) for its data memory, the specific component of RadPC that is presented in this paper is a novel external memory scheme with accompanying systems that can detect, and correct faults that occur in the proposed data memory of the computer while allowing the computer to continue foreground operation. A prototype implementation of the memory protection scheme was flown on a Raven Aerostar Thunderhead high-altitude balloon system in July of 2021. This flight carried the experiment to an altitude of 75,000 feet for 50 hours allowing the memory in the experiment to be bombarded with ionizing radiation without being attenuated by the majority of Earth's atmosphere. This paper will discuss the details of the fault mitigation strategy, the design-of-experiments for the flight demonstration, and the results from the flight data. This paper may be of interest to engineers that are designing flight computer systems that will be exposed to ionizing radiation and are looking for a lower cost SEU mitigation strategy compared to existing radiation-hardened solutions.**

***Index Terms*—FPGA, Memory, Radiation**

I. Introduction

The demand for computers that can continue operation in space environments has increased with the rapid rise of space exploration missions. One of the leading causes of deteriorating effects to computers in space, specifically CMOS devices, is cosmic radiation. While terrestrial computers are protected from ionizing radiation by the Earth's atmosphere and magnetic field, space computers must be designed to operate in the presence of radiation that can cause logical faults and potentially crashes [2].

Taking into consideration the classification of the effect is important when implementing systems targeting resilience to the effects of radiation. The two primary classifications of space radiation are single event effects (SEEs) and total ionizing dose (TID) — both of which decrease the computer system's reliability [3]. A CMOS device experiences an SEE when high-energy particles strike the device, which in turn, introduces excess charge in the semiconductor material leading to inadvertent logic level shifts. From SEEs, we can derive three subcategories of faults that occur from these logic-level transitions, each of which cause varying levels of damage to the device. The first branch of SEEs can be classified as a single event transient (SET), which occurs when ionized particles deposit charge onto a region of the device and cause an unwanted voltage pulse. The second branch of SEEs can be classified as a single event upset (SEU), which is when an SET is captured within a storage device such as a D-flip-flop or memory cell. If either a SET or SEU occur and introduce a fault that cannot be repaired by conventional recovery strategies, such as a system reset, a more rigorous recovery strategy must be implemented. This final branch is called a single event functional interrupt (SEFI).

A TID effect's root cause is from low-level energy particles depositing charge within a device's insulating regions. This leads to a gradual breakdown of the insulating material over time and can lead to unwanted current flow and permanent transistor biasing. This results in excess power draw and overall degradation of the device itself. The effects of TID emerge over time and are irreversible [4].

SEEs are classified as logical faults because they don't degrade the material of the semiconductor. However, they can lead to system crashes during critical times in a mission that make them lead to catastrophic failure. Our work focuses on the call for using COTS parts in space missions. While it has been shown possible to reverse some of the ionizing dose effects using annealing, it does require specific circuitry to be included in the device that is not present in COTS parts we are using. Our focus is on SEEs, not necessarily TID.

NASA FOP

Montana State University has spent the past decade developing a reconfigurable computing system, denoted "RadPC," that can detect the effects of SEE-induced faults and respond with a suite of recovery mechanisms [1]. A mission ready version of RadPC will be sent to the Moon through NASA's Commercial Lunar Payload Services (CLPS) project as part of the Artemis lunar program in 2023.

In preparation for the Lunar mission, the experiment in this paper serves to validate the RadPC platform in harsh environments through the means of a high-altitude balloon flight. This flight targets the system-wide test of the RadPC architecture, error detection, error recovery, error logging, as well as an external memory system, all of which will increase the NASA technology readiness level (TRL) of the RadPC computer architecture.

II. Motivation

The RadPC platform is designed with the intent of being a cost-effective space computing solution by implementing its architecture on commercial off the shelf (COTS) components. To achieve resilience to cosmic radiation using COTS grade parts, the design itself integrates radiation mitigation technology directly into the architecture. By using COTS products, the platform avoids the use of expensive products that are manufactured for radiation tolerance that tend to be cost-prohibitive for low Earth orbit (LEO) missions.

A traditional method for mitigating TID effects on space computing is shielding. This is achieved by designing an enclosure with external metals of varying thicknesses that meet design tradeoffs including overall system mass versus radiation resilience [5]. Another widely used technique for mitigating TID effects is implemented at the fabrication level including radiation hardening by process (RHBP) and radiation hardened by design (RHBD). RHBP decreases the probability of a strike depositing charge within the semiconductor gate by altering the underlying semiconductor materials [1] [6]. RHBD achieves a similar end through fabricating components with non-standard layout geometries that attempt to reroute the resulting charge from a radiation strike into the power supply nodes of the circuitry [1] [7]. While both RHBP and RHBD prove to be effective in the mitigation of the deteriorating effects of TID and SEEs, they are prohibitively expensive for most commercial space missions.

In recent years, the smaller processing nodes of CMOS transistors have resulted in feature sizes (<65nm) that are less susceptible to TID effects due to the reduced probability of charge getting trapped in the thin insulating regions. Simultaneously the susceptibility of these smaller transistors to SEEs has increased due to the reduced amount of energy needed to cause a logic level transition. As a result, SEEs are becoming the primary concern for future space missions that use modern semiconductor materials.

Given the rapid increase of space exploration missions being performed, the need for cost-efficient radiation resilient computing has become a top priority. Montana State University seeks to satisfy this goal with RadPC. The purpose of this high-altitude balloon experiment was to test a fault recovery procedure for external memory in flight computers. The fault recovery procedure is part of a larger effort at Montana State University to develop a radiation tolerant computer technology for use in space. The computer, called "RadPC", has been matured over the past 12 years through a variety of flight demonstrations on various sub-systems. This balloon flight specifically focused on detecting and correcting errors in external data memory for RadPC.

III. Experiment Overview

The experiment developed consisted of two identical payloads, each containing the RadPC computer, the memory fault recovery system, and interface electronics to the balloon system. The payloads were designed to fly on the Raven Aerostar Thunderhead high-altitude balloon system. The Thunderhead Flight Control Unit (FCU) provided DC power to the payloads and served as a communication bridge to a ground station where telemetry data could be retrieved during flight. Each payload was equipped with a Raspberry Pi 4 (RPi) as the interface between the RadPC computer and the balloon FCU. The payloads received power from the Thunderhead's Flight Control Unit (FCU) which provided a +28VDC power to both payloads. The payloads interfaced to the FCU via Ethernet transmission control protocol (TCP) sockets. The system flew for approximately 50 hours at altitudes between 70,000 feet to 85,000 feet and generated around 20,000 packets of telemetry data. Each telemetry packet contained experiment status information as well as readings of voltages and currents from the various power supplies (5V, 3.3V, 2.5V, 1.8V, and 1.0V DC) and temperature. Figure 1 depicts the physical wiring and communication interface between each RadPC payload experiment and the Raven Thunderhead FCU. The following

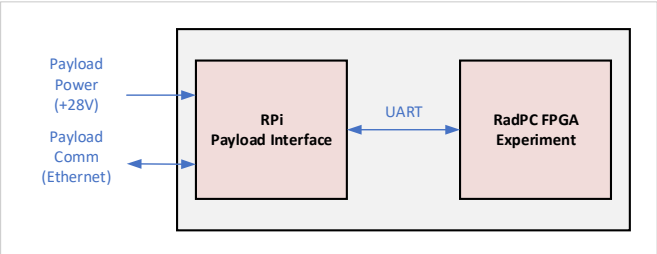


Fig. 1. Payload Interface to Raven FCU

subsections breakdown each interface into 3 distinct levels, the payload software interface, the payload hardware interface, and our ground-station for packet requests and data storage.

A. Software Interface

The software interface implements TCP sockets between the RPi and the Raven FCU for packet telemetry. The RPi serves as a interface bridge by requesting and interpreting data telemetry from the RadPC computer and transmitting them to the FCU. The FCU then sends the telemetry over the Iridium satellite network and to a custom data server that acts as our

ground station. This is achieved by going through Raven's provided API. The TCP interface is routed through the RPi's on-board Ethernet port.

B. Hardware Interface

The hardware interface implements a RPi daughter card that serves three primary functions. The first key function is to regulate the +28VDC input voltage to +5VDC to power the RPi. The second function is to distribute +28VDC from the FCU to the RadPC single board computer (SBC). The RadPC SBC communicates packets via RS422 protocol, which provides the third key function: to transceive UART protocol from the RPi to RS422 to the RadPC SBC.

C. Ground-station Server

The ground-station server houses data that gets funneled from the Iridium comm links through Raven's Application Programming Interface (API) and into our server-side data storage. The ground station server also acts as a data visualizer that any user can login to and view / plot selected data ranges.

D. Experiment Objectives

The RadPC computer executes identical programs that trigger recovery procedures when a fault is identified. This balloon flight serves as a relevant environment that the RadPC architecture is targeting development for. As such, it must detect faults induced by ionized radiation, determine the detected faults effects on the system, and assert the proper recovery procedure. The recovery systems report to an external micro-controller whose primary function is to poll data lines, assemble packets, and serve as a communication peripheral to the outside world. The external micro-controller will additionally inject faults over a serial interface to the Configuration Memory Monitor (CMM) controller on the FPGA fabric.

IV. RadPC Architecture

The existing RadPC architecture employs redundant cores running synchronous to one another on a Xilinx Artix 7 FPGA. This approach is an extension of the widely adopted triple modular redundant (TMR) approach used in space system. RadPC extends TMR with an additional core (NMR) to provide increased reliability. The NMR processors run in conjunction with fault detection and fault mitigation strategies. Each core contains identical program memories that are monitored via external fault detection strategies. The individual cores are referred to as tiles for the rest of this paper as they represent regions that can be reconfigured individually on the Artix 7 FPGA.

This particular experiment is known as RadPC@Scale and is an extension of the existing RadPC architecture by adding additional external memory. Rather than using block RAM instantiated within the FPGA fabric, the tiles communicate to four separate Microchip 23LC1024 external data memories via serial peripheral interface (SPI) protocol. The tradeoff between using external memory is a slower interface with a significantly larger capacity. The top-level system architecture can be visualized in Figure 2. The following sections will show the abstractions of the RadPC hierarchy. The tiles seen in Figure 2 are abstracted in the top level architecture. The Data Memory blocks are external integrated circuit's (IC) from the FPGA. The Data Memory Scrubber (DMS) is a state-machine within the FPGA. The Checkpoint Bus ensures that the program execution of each tile is synchronous to one another. When repairing tiles the checkpoint system allows the repaired core to "catch up" with the other tiles before resuming foreground operation. The voter subsystem compares the outputs of the four tiles and asserts a data memory scrub process if one tile output does not match the other tile outputs. The DMS walks through memory sequentially and compares the outputs of all of the tiles and overwrites any tile with a mismatched memory value to the correct output from the others. Finally, the CMM is a subsystem that monitors any single bit flip events in each tiles configuration memory. Additionally, the CMM can receive commands via serial bus to inject faults in configuration memory and trigger its repair mechanisms. Any time a fault has been repaired, it will send a message over the serial interface to the external micro-controller that it has repaired a single error in configuration memory.

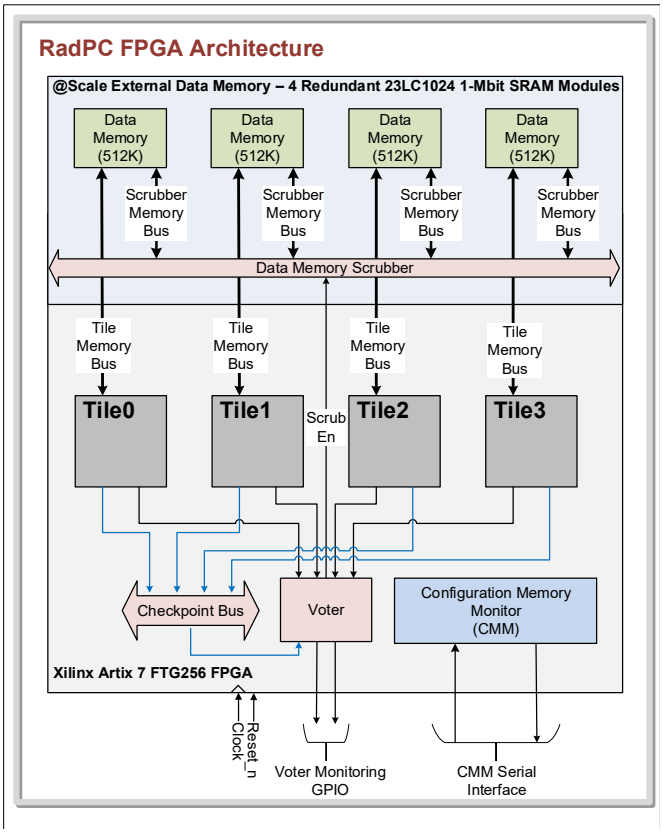


Fig. 2. RadPC at Scale FPGA Architecture

A. RadPC Tile Architecture

Figure 3 shows the tile architecture. Each tile implements a Xilinx Microblaze with program memory, an Advanced eX-

tensible Interface (AXI) peripheral bus controller, and general purpose input/output (GPIO) blocks associated with memory and communication to external devices. The block labelled

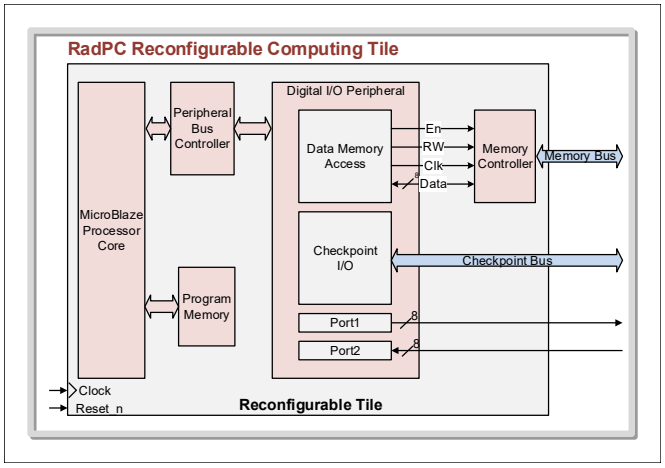


Fig. 3. RadPC at Scale Tile Architecture

'Memory Controller' is abstracted by each tile and is depicted in the following section.

B. RadPC Memory Architecture

The memory architecture is a series of control interfaces that arbitrate between tile memory commands and the data memory scrubber. During normal operation, the memory responds to a tile command to read or write to a memory block. If a fault is detected, the tile repairs the fault using a scrubbing procedure. Figure 4 depicts the tile memory architecture where the Memory Controller block is in charge of the arbitration between tile memory access and DMS access. Error correction codes are also used to immediately repair individually faulted bits within data memory. The memory controller block illustrated in Figure 4 is a finite state machine that arbitrates between the RadPC@Scale memory requests and the DMS memory requests. When a command has been sent to write to the external memory, the memory controller will shift out the 12-bit hamming encoded word via SPI protocol. When a read instruction is being executed, the ECC decoder block will decode the 12-bit word received from the external data memory and pass back a byte sized data value to the system that triggered the transaction (DMS or tile). This creates four separate redundant external memories, one per tile. When the voter invokes the DMS, the DMS will report each redundant memory value and repair any mismatched tile value with the "healthy" value from the majority tiles. Rather than replicating memory values within a singular DRAM, this system is replicating memory values from separate tiles over four separate redundant memory ICs and comparing the inputs and outputs against one another.

The control signals to read and write to data memory were through the Xilinx Microblaze AXI GPIO peripheral bus. When configuring a program within Xilinx Vitis IDE, the programmer invokes GPIO calls to read and write to external data memory systems.

The fault injection blocks serve two purposes: a memory control signal passthrough block when it's inactive, or a memory fault injector block when it's enabled. This block will enable itself when the input address signal from a tile is equal to a registered fault address internal to this block. Once enabled, this subsystem directly modifies the data output from the tile that is being written to the external memory. This subsystem lies between the tile and the memory controller subsystems, and therefore neither the tile nor the memory controller are aware that a memory fault has been injected into the value being written. It is only after this memory address is read from, that the fault is detected by the voter.

V. Results

The RadPC computer assembled data packets on a 10 second cadence. The ground-station server requested packets on a 5 minute cadence, with each payload offset from one another. The data plotted in the following figures is the whole data set that was post-processed after payload retrieval.

A. Packet Structure

Every packet has a 128-byte structure. It begins with a packet header, then moves into power data for each voltage rail. After the power data, it logs pertinent FPGA data. The primary data fields this paper will discuss from Table I are the power monitors, temperatures, Tile Fault Counts, CMM Fault

TABLE I
RadPC Telemetry Packet Structure

Field Length (bytes)	Field Description	Packet Location
24	Packet Header	0-23
3	Packet Number	24-26
4	1.0V Power Rail	27-30
4	1.8V Power Rail	31-34
4	2.5V Power Rail	35-38
4	3.3V Power Rail	39-42
4	5.0V Power Rail	43-46
4	28.0V Power Rail	47-50
2	MCU Temperature	51-52
2	FPGA Temperature	53-54
2	FPGA Tile Count	55-56
1	FPGA Voter Outport	57
1	FPGA Port In (Unused)	58
8	Tile Fault Count	59-66
2	CMM Fault Count	67-68
8	Tile Injection Count	69-76
2	CMM Injection Count	77-78
8	Tile Checkpoint Lags	79-86
8	Tile correctable Faults	87-94
8	Tile Uncorrectable Faults	95-102
2	CMM Correctable Faults	103-104
2	CMM Uncorrectable Faults	104-105
2	MEM Error Correction Codes	106-107
6	MEM Scrubber Outputs	108-114
2	Mission MCU Reset Count	115-116
1	Last Communication Request	117
6	*Reserved*	118-123
2	CRC Codes	124-125
2	End Packet Symbol	126-127

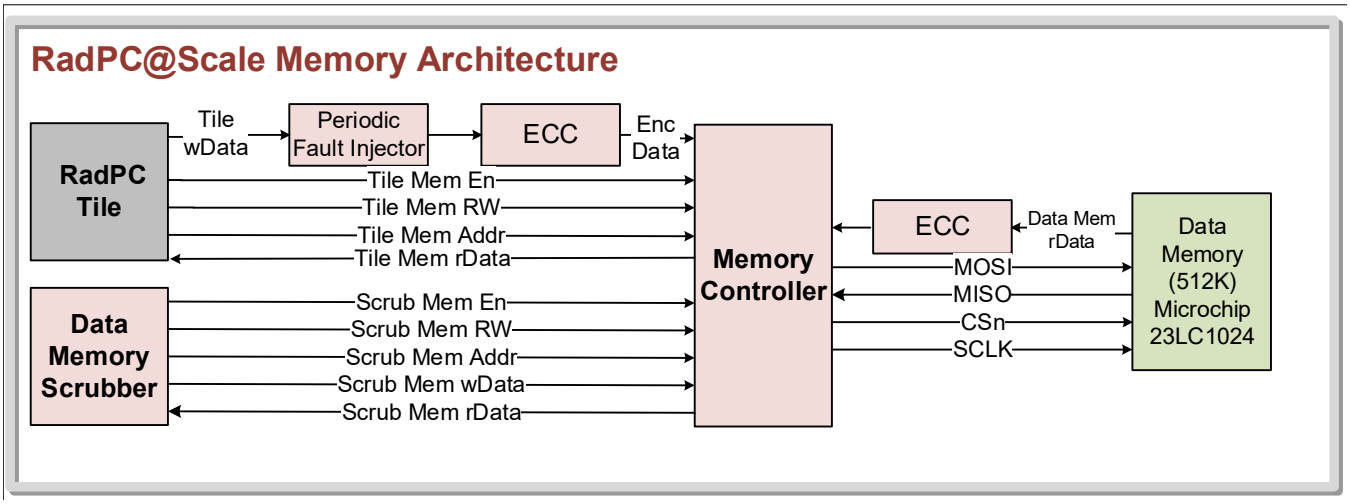


Fig. 4. Simplified tile memory architecture.

Counts, CMM Injection Counts, CMM Correctable Faults, MEM Scrubber Outputs, and Mission MCU Reset Counts.

B. FPGA Data Plots

This section depicts all data fields related to the FPGA computer during flight. It is divided into subsections relevant to different data fields. Payload 1 and Payload 2 exhibited functionally equivalent operation, as such, the data for both payloads is nearly equal. Therefore, the digital data pertaining to the RadPC@Scale tiles will only be shown for Payload 1.

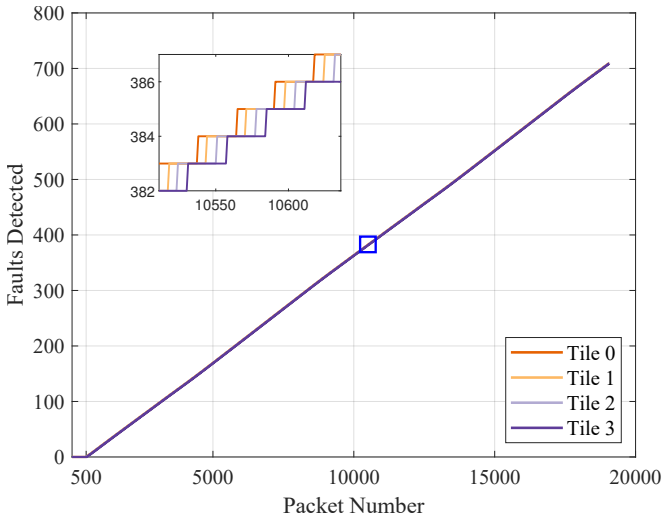


Fig. 5. Payload 1 - FPGA Tile Fault Counts

1) FPGA Fault Injection Detection: It can be seen in the exploded view highlighted by the blue box in Figure 5 that the fault counts increment in a round robin fashion. This is implemented by design and begins with a fault injection on Tile 0, then Tile 1, and so on and so forth. The fault injection block stands between the tile memory control signals and the memory controller, depicted in Figure 4. Due to the inherent design of the fault injection, the system did not detect any external radiation faults. If it did, there would have been additional offsets between the number of faults detected between tiles.

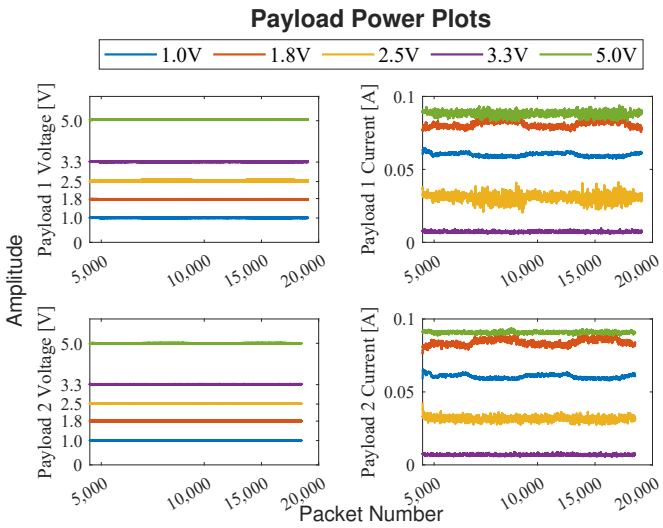


Fig. 6. Payload 1 - RadPC Power Monitoring

2) Power Monitoring: The RadPC Computer implements an analog-to-digital conversion (ADC) measurement scheme for monitoring the voltage supplies. The computer has 5 voltage rail supplies that feed the downstream FPGA and MCU. Notice in Figure 6 that the voltage rails follow the desired voltage levels. The original current data for the power plots was incredibly noisy, so in post-processing, a moving average filter was applied with a window width of 32. Similar to the power plots from Payload 1, Payload 2 behaves in a similar manner. The power monitoring shows that the payloads were in good health during flight.

3) Data Memory Scrubber: This section depicts the data memory scrubber values throughout the flight duration. The

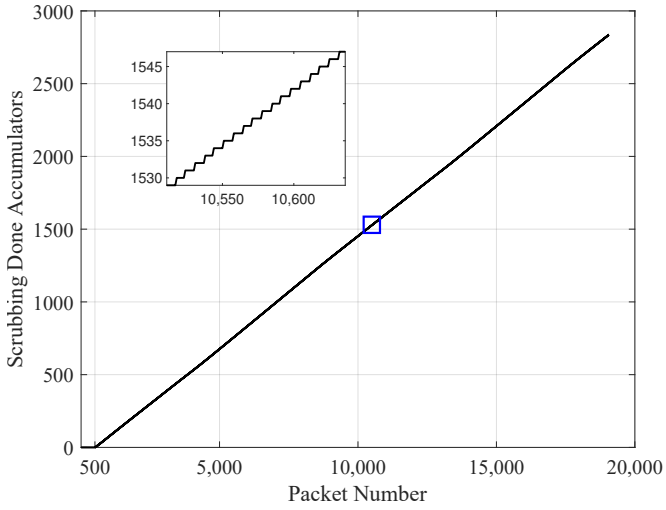


Fig. 7. Payload 1 - Data Memory Scrubber Data Logs

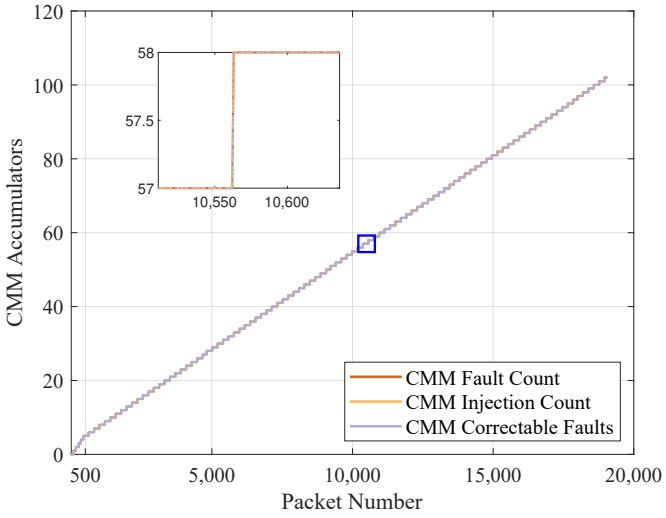


Fig. 8. Payload 1 - CMM Data

data scrubber data depicted in Figure 7 are the accumulations of each time the voter system triggered a data memory scrub. The number of scrubs is modeled by the following equation.

$$S = F_1 + F_2 + F_3 + F_4 \tag{1}$$

Where S is the number of scrubs that has occurred, and F_x is the accumulated faults on any given tile x. Notice that in Figures 5 and 7 that faults did not begin to be introduced until around data packet 500. This was by design as Raven flight operations crew incorporate tests for the balloon launch that include repeated power cycling of the payloads. To avoid data offsets that needed data post-processing due to the flight operations power cycles, a delay of 20 minutes was introduced to begin injecting faults.

4) FPGA CMM: Recall the CMM IP can receive commands via a serial interface to inject errors. The foundation of this IP is to both manually inject faults and detect faults caused by external radiation and then recover from these faults. As denoted on Figure 8, the injections, faults detected, and faults recovered were all detected by our monitoring MCU at the same time and are all overlaid onto one another. Due to the faults detected being the same as the faults injected, it can be seen that there were no faults caused by cosmic radiation.

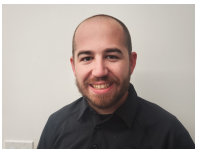
Over the course of the 50 hour flight, the memory experiment injected 2836 faults and successfully recovered from each one.

VI. Conclusions

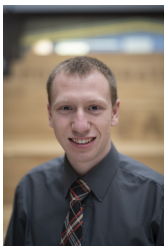
The experiment carried out with the payloads presented in this paper, named RadPC-at-Scale, has demonstrated the performance of a cost-effective, commercial off-the-shelf, radiation tolerant space computing system. Through its suite of fault detection and mitigation strategies, it has shown that it will perform under harsh space environments by recovering from faults, injected or induced by radiation. While this flight has shown that there were no faults induced by radiation onto the payloads, the RadPC architecture still maintained healthy operation as depicted by the data packets shown above.

The system did not detect any external radiation faults due to the rapid rate at which faults were injected onto the system. For this flight, a fault was injected once every 50 seconds onto a singular tile's data memory in a round-robin fashion. When a fault was detected by the voter, the voter would then repair the entire external data memory space across all four tiles. When combining the rapid fault injection, the voter detection and repair mechanisms, this effectively filtered out any faults induced by radiation within the external 23LC1024 SRAM memory devices. To remedy this for the next experiment, the fault injection will be limited to a period of 12 hours. This will allow the system to recognize the fault injection period and interpret whether a detected fault is caused from external radiation or from the memory fault injector block.

VII. Authors



Justin P Williams is a graduate research student studying electrical engineering at Montana State University and the payload director of RadPC@Scale.



Colter Barney is a graduate research student studying electrical engineering at Montana State University.

Zachary Becker is a undergraduate research student studying electrical engineering at Montana State University.

Jake Davis is a graduate research student studying mechanical engineering at Montana State University.

Chris Major is a graduate research student studying electrical engineering at Montana State University.

Brock J. LaMeres is a professor of electrical and computer engineering at Montana State University and the principal investigator of RadPC@Scale.

ACKNOWLEDGMENT

This work was supported by the NASA Flight Opportunities Program under CAN# 80NSSC20K0107.

REFERENCES

[1] C. Major et al., "Overview of the Upcoming RadPC-Lunar Mission," 2021 IEEE Aerospace Conference (50100), 2021, pp. 1-7, doi: 10.1109/AERO50100.2021.9438284.

[2] J. L. Barth, C. S. Dyer and E. G. Stassinopoulos, "Space, atmospheric, and terrestrial radiation environments," in IEEE Transactions on Nuclear Science, vol. 50, no. 3, pp. 466-482, June 2003, doi: 10.1109/TNS.2003.813131.

[3] Claeys, C., and Eddy Simoen. Radiation Effects in Advanced Semiconductor Materials and Devices. Vol. 57. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2013. Springer Ser. in Materials Science. Web.

[4] Rycroft, M.J. "Handbook of Radiation Effects: Holmes-Siedle A. and Adams L., 1994 479 Pp., Oxford Science Publications, £45 Hbk, ISBN 0-19-856347-7." Journal of Atmospheric and Terrestrial Physics, vol. 57, no. 13, 1995, pp. 1672–1673.

[5] R. Uzel and A. Özyildirim, "A study on the local shielding protection of electronic components in space radiation environment," 2017 8th International Conference on Recent Advances in Space Technologies (RAST), 2017, pp. 295-299, doi: 10.1109/RAST.2017.8003007.

[6] G. Anelli et al., "Radiation tolerant VLSI circuits in standard deep submicron CMOS technologies for the LHC experiments: practical design aspects," in IEEE Transactions on Nuclear Science, vol. 46, no. 6, pp. 1690-1696, Dec. 1999, doi: 10.1109/23.819140.

[7] A. Makihara et al., "Hardness-by-design approach for 0.15 /spl mu/m fully depleted CMOS/SOI digital logic devices with enhanced SEU/SET immunity," in IEEE Transactions on Nuclear Science, vol. 52, no. 6, pp. 2524-2530, Dec. 2005, doi: 10.1109/TNS.2005.860716.