SSC17-XIII-02

SSIVP: Spacecraft Supercomputing Experiment for STP-H6

Sebastian Sabogal, Patrick Gauvin, Brad Shea, Daniel Sabogal, Antony Gillette, Christopher Wilson, Ansel Barchowsky, Alan D. George
NSF Center for High-Performance Reconfigurable Computing (CHREC) – University of Pittsburgh
1238D Benedum Hall, 3700 O'Hara Street, Pittsburgh, PA 15261; 412-624-9664
alan.george@pitt.edu

Gary Crum, Tom Flatley
NASA Goddard Space Flight Center
8800 Greenbelt Rd, Greenbelt, MD, 20771; 301-286-3713
gary.a.crum@nasa.gov

ABSTRACT

The Department of Defense Space Test Program (STP) provides spaceflight opportunities for conducting on-orbit research and technology demonstrations to advance the future of spacecraft. STP-H6, the next mission of the program to the International Space Station (ISS), will include a prototype spacecraft supercomputing experiment and framework, called Spacecraft Supercomputing for Image and Video Processing (SSIVP), developed at the National Science Foundation (NSF) Center for High-Performance Reconfigurable Computing (CHREC) at the University of Pittsburgh. SSIVP introduces scalable, high-performance computing (HPC) principles to a CubeSat form-factor to advance the state of the art in space computing. SSIVP adopts the CHREC Space Processor (CSP) concept, a multifaceted design philosophy for a hybrid system of commercial and radiation-hardened (rad-hard) components supplemented with fault-tolerant computing, and a hybrid processor combining fixed-logic CPU and reconfigurable-logic FPGA. SSIVP features five flight-qualified CSPv1 computers as compute nodes, to facilitate this supercomputing concept, and one µCSP smart module, for running a Gallium Nitride (GaN)-based power converter sub-experiment. SSIVP is a versatile, heterogenous platform capable of processing application workloads in the processor or on runtime-reconfigurable FPGA accelerators. In this paper, we present the flight hardware and software, frameworks for parallel and dependable computing, and mission objectives for SSIVP.

I. INTRODUCTION

Space-systems designers are challenged to create reliable, high-performance computers that address the escalating on-board computational demands for both sensor-data and autonomous processing. Design restrictions in size, weight, power, and cost (SWaP-C), prohibitive launch costs, and reliability considerations due to space radiation, further add to the complexity of design. Concurrently, a recent trend is the continued adoption of CubeSat technology for space missions. CubeSats, typically specified at the 1U ($(10\ cm)^3$ and < 1.3 kg) form-factor, are emerging as low-cost platforms enabled by the miniaturization of electronics, sensors, actuators, and instruments. CubeSats may also feature commercial-off-the-shelf (COTS) technology that render high-risk, high-reward missions an affordable possibility. Furthermore, COTS devices often offer substantial performance and energy-efficiency benefits over traditional radiation-hardened (rad-hard) devices but are highly susceptible to space radiation.

Space radiation presents a myriad of challenges for electronic devices and systems deployed into this harsh environment. The effects of radiation on electronic devices are categorized into short-term, *single-event effects* (SEE) and long-term, cumulative effects. SEEs occur when a single ionizing particle strike deposits sufficient energy to influence the device. SEEs may further be classified as destructive and non-destructive effects. Cumulative effects, such as *total ionizing dose* (TID), describe the radiation dosage acquired over time until the device falls out of specification [1].

Recently, the National Research Council (NRC) has identified CubeSats and SmallSats (small satellites) as a disruptive innovation for future space science missions [2]. CubeSats have proliferated substantially in the educational, governmental, and industrial sectors, as shown in Figure 1, and several missions have launched, or are planned, for a wide variety of space science and technology applications. Constellations of CubeSats in formation flight can achieve a distributed computing performance equivalent to that of large, monolithic satellites. Furthermore, the affordability associated with CubeSats enables replenishment over time as spacecraft in the constellation fail.

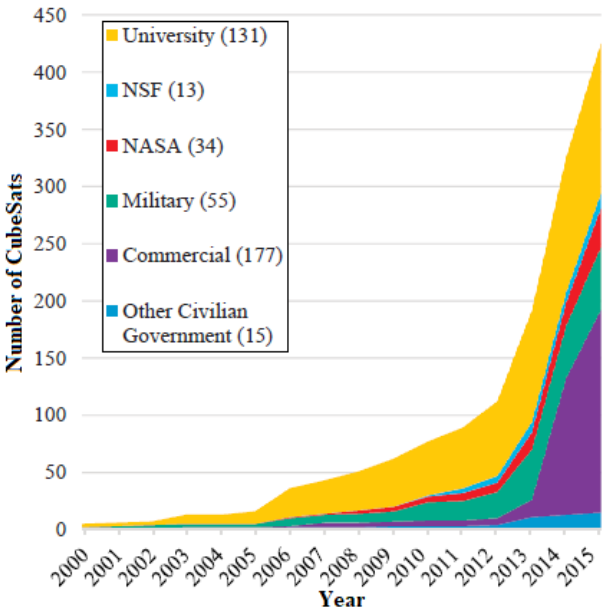

Figure 1: Cumulative Number of CubeSats Launched by Organization [2]

Despite the advantages of CubeSat computing, the NRC report highlighted two key areas in need of improvement: fault protection and high-performance computing for spacecraft operations and payload processing. Several CubeSats in low-Earth Orbit (LEO) focus on Earth observation, including imagery, video, and remote sensing. Due to limitations in downlink bandwidth, it is often desirable to preprocess volumes of data on-board the spacecraft. For example, super-resolution uses several overlapping images to generate a single, relatively-higher resolution image. Image segmentation can be used for classifying image features and identifying the value of the image prior to downlink. To address the implications denoted in the report, we introduce the Spacecraft Supercomputing for Image and Video Processing (SSIVP) experiment for the upcoming Space Test Program – Houston 6 (STP-H6) mission to the International Space Station (ISS).

The SSIVP flight box is under development at the National Science Foundation (NSF) Center for High-Performance Reconfigurable Computing (CHREC) at the University of Pittsburgh. SSIVP introduces *high-performance computing* (HPC) principles to the CubeSat form-factor and beyond, promoting parallel and distributed computing for space applications. Furthermore, SSIVP will investigate GaN-based power converters as part of the GaN sub-experiment. This paper describes the flight hardware and software, framework and experiments for parallel and dependable computing, and mission objectives for SSIVP.

II. BACKGROUND

This section provides a cursory overview of the spaceflight opportunity, provided by the Space Test Program – Houston office, for the STP-H6 mission. Furthermore, key flight software (core Flight Executive and core Flight System) and flight hardware (CHREC Space Processor and its variants) are discussed. A preface on GaN-based power converters is also presented.

Space Test Program - Houston 6

The Department of Defense (DoD) Space Test Program (STP) was created in 1965 to provide an economic and efficient process to offer spaceflight opportunities for the DoD space science and technology community [3]. The STP – Houston office serves to advance future spacecraft by enabling on-orbit research and technology demonstrations on-board the ISS and other human-rated launch vehicles [4].

STP-H6 is the next mission that will feature several payload experiments, including SSIVP. STP-H6 will be integrated and flown under the management and direction of the DoD STP Human Spaceflight Payloads Office. The expected launch window for this mission is late 2018 to early 2019. The STP-H6 Pallet is illustrated in Figure 2.

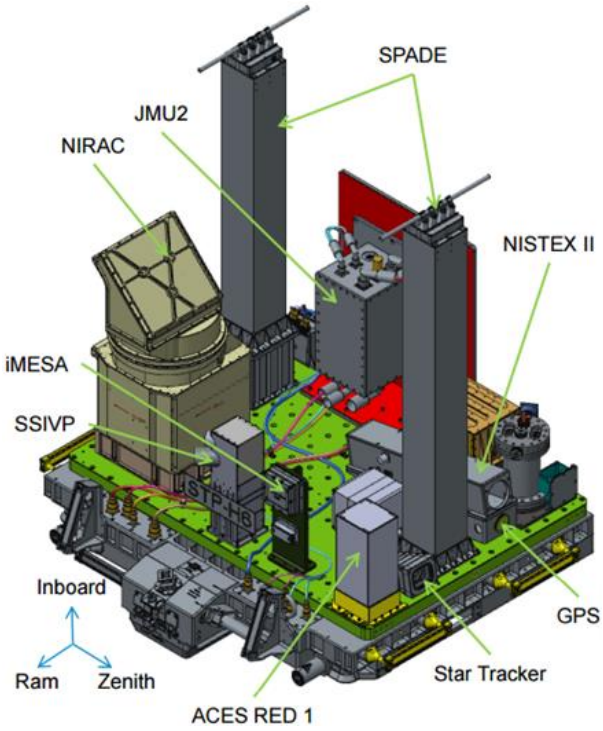

Figure 2: STP-H6 Pallet Model

core Flight Executive and core Flight System

NASA Goddard Space Flight Center's (GSFC) *core Flight Executive* (cFE) and *core Flight System* (cFS) form a software suite that facilitates the development of platform-independent and reusable flight software [5].

The cFE framework resides in the *Core Layer* and provides the core services (i.e., messaging, timing and events, startup and runtime, and table services) necessary to support a run-time environment and development platform for cFE/cFS applications. The Software Bus (SB) provides a messaging system for intra-node communication between cFE/cFS applications. The cFE core services expose an Application Programming Interface (API) for flight software applications. At the *Application Layer*, mission designs can adopt and reuse existing cFE/cFS applications and develop mission-specific applications.

The cFE framework uses software libraries in the *Abstraction Layer* for portability across varied operating environments. The *OS Abstraction Layer* (OSAL) abstracts the operating system (e.g., Linux, RTEMS) and the *Platform Support Package* (PSP) abstracts the target hardware platform (e.g., ARM Cortex-A9, RAD750).

CHREC Space Processor Concept

CHREC Space Processor (CSP) is a concept for hybrid space computing developed by researchers at the NSF CHREC center at the University of Pittsburgh [6]. The CSP concept centers on consolidating a hybrid-processor and hybrid-system architecture. This concept has matured and now features a collection of space-development platforms.

The hybrid-processor facet combines two or more distinct computing architectures (e.g., CPUs, FPGAs) to attain the advantages of each. For example, control-flow orientated algorithms are suited to general-purpose CPUs, whereas dataflow orientated algorithms suited to FPGAs. Several vendors offer System-on-Chip (SoC) devices featuring combinations of different architectures, such as the Nvidia Tegra (CPU+GPU), Texas Instruments KeyStone (CPU+DSP), and Xilinx Zynq SoC (CPU+FPGA).

The hybrid-system facet combines three themes (COTS technology, rad-hard technology, and fault-tolerant computing) to attain the advantages of each. COTS devices offer substantial performance and energy-efficiency benefits over their rad-hard counterparts but are susceptible to space radiation. Rad-hard devices are relatively immune to radiation, but are often generations behinds COTS devices in terms of performance, power, size, and capability. By featuring COTS technology, for its performance and energy-efficiency advantages, supported by rad-hard technology, for reliable monitoring and management of the COTS devices, and augmented by fault-tolerant computing, a reliable, high-performance space processor is achievable.

CSPv1

CSPv1 is the first full realization of the CSP concept, as shown in Figure 3. CSPv1 is designed for the 1U CubeSat form-factor and features the Xilinx Zynq-7020 SoC hybrid-processor, which contains a fixed-logic dual-core ARM Cortex-A9 processor and reconfigurable-logic Artix 7 Series FPGA fabric. The CSPv1 flight board contains rad-hard supervisory, power sequencing, and watchdog components to monitor and manage the COTS processor and memory.

CSPv1 has undergone environmental testing, including both random vibration and thermal vacuum tests. It has also undergone four radiation-beam tests, including two neutron tests at LANL Los Alamos Neutron Science Center (LANSCE) and TRIUMF, and two heavy-ion tests at BNL NASA Space Radiation Laboratory (NSRL).

Finally, CSPv1 has flight heritage and has been operating successfully for months on-board the ISS as part of STP-H5. A dual-CSPv1 flight box was included as a sub-experiment in the ISS SpaceCube Experiment Mini (ISEM), by NASA GSFC's Science Data Processing Branch and SpaceCube Team, for the STP-H5/ISEM mission [7]. STP-H5 was recently launched and mounted onto the ExPRESS Logistics Carrier-1 (ELC-1) on the ISS, advancing the Technology Readiness Level (TRL) of CSPv1 Rev. B to TRL9 in LEO. SSIVP will feature five CSPv1 flight boards, leveraging its high-performance processor and FPGA fabric for HPC.

Figure 3: CSPv1 Rev. B COTS Configuration

µCSP

µCSP is a second realization of the CSP concept but with further reduced SWaP-C [8]. µCSP follows the *Smart Modules* concept, which is a design framework for rapid configuration, integration, and prototyping of modular, reusable hardware cards. Each hardware card provides its own "smart" function (e.g., sensor, instrument, actuator) and conforms to a uniform hardware design specification. µCSP provides each module with low-power processing. Multiple smart modules can be assembled for more complex functionality or distributed computing.

µCSP is designed as a System-on-Module (SoM), as shown in Figure 4, and features the commercial Microsemi SmartFusion2 hybrid-processor, which contains a fixed-logic, single-core ARM Cortex-M3 microcontroller and reconfigurable-logic, flash-based FPGA fabric. Similarly, the µCSP SoM has a hybrid-system architecture, including rad-hard power regulators for the FPGA core voltages, watchdog to monitor the hybrid-processor, and NOR flash memory. SSIVP will include a single smart module, containing the µCSP Rev. A platform, for flight heritage.

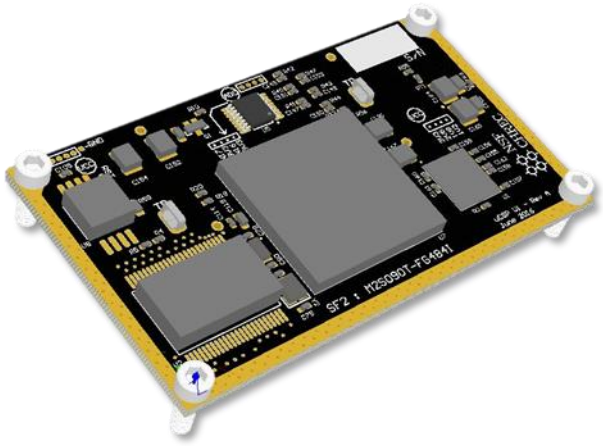

Figure 4: µCSP Rev. A Model

GaN Point-of-Load (PoL) Converters

Spacecraft with large computational loads require many PoL converters to supply the power rails demanded by high-performance processors and FPGAs. Rad-hard PoL converters are outpaced by their COTS counterparts in terms of size, efficiency, and technology. The miniaturization of increasingly large computational loads in the CubeSat form-factor further encourages the use of lightweight, compact, power conversion electronics for future spacecraft.

Gallium Nitride (GaN)-based, high-electron-mobility transistors (HEMT) have demonstrated higher efficiency and lower cost, weight, and volume when compared to conventional Silicon MOSFETs, and show considerable promise for spacecraft use [9]. Furthermore, commercial HEMTs have been tested and shown to be extremely resistant to TID and SEEs, particularly when biased below 100 V [10][11]. COTS GaN HEMTs are becoming widely available, and radiation-hardened FETs are being developed and produced by several vendors.

SSIVP includes the GaN sub-experiment to evaluate GaN-based PoL DC-DC converters for space use. This experiment will evaluate two COTS synchronous buck controllers (Texas Instruments LM25141-Q1 and Linear Technologies LTC3833) paired with three different COTS and rad-hard HEMTs (Efficient Power Conversion EPC2014C, Freebird Semiconductor FBG04N08A, and Teledyne e2v EVG100E15). It will be used to examine the performance of several converters in LEO to investigate lightweight, efficient, and inexpensive power conversion options for future missions.

III. SSIVP HARDWARE DESIGN

The SSIVP flight hardware is accommodated in a 3U flight box, as shown in Figure 5. 1U of the flight box is occupied by dual Camera Link cameras. The remaining 2U of the flight box encapsulates eight boards, including five flight-qualified CSPv1 cards (one CSPv1 Rev. C and four CSPv1 Rev. B), one flight-qualified µCSP Rev. A smart module, one power card, and one backplane interconnect board. This section presents a detailed description for each flight board, including the network topology realized in the backplane.

Figure 5: SSIVP Flight Box Model

Card Description

SSIVP includes five CSPv1 cards, with one designated as the *head node* (CSP0), two as *worker nodes* (CSP1 and CSP2), and two as *camera nodes* (CSP3 and CSP4). The *head node* is assigned to a CSPv1 Rev. C

card, which upgrades several commercial-grade components with rad-hard equivalents from the CSPv1 Rev. B design. Only the *head node* interfaces with the STP-H6 pallet, using UART over RS-422, for commanding and telemetry. The *head node* forwards packets to and from other CSPv1 nodes as necessary. Additionally, the *head node* interfaces to the smart module µCSP and GaN sub-experiment over UART and SPI, respectively, and drives the solid-state relays for powering the cameras and the GaN sub-experiment.

The remaining four nodes are assigned to four CSPv1 Rev. B cards. The *camera nodes* include a Camera Link pipeline, residing in the Zynq FPGA, to interface with the cameras. Lastly, the remaining *worker nodes* are available solely for computing and future uploaded experiments.

SSIVP includes a single smart module, featuring the µCSP Rev. A hybrid-processor. The smart module contains two sensors (resistance temperature detector (RTD) and accelerometer) and the GaN sub-experiment. The smart module includes two sets of three GaN PoL converters differentiated both by the buck controllers and the GaN HEMTs used. For each of the controllers, three converters will be built on drop-in PCBs with power stages comprised of three different GaN HEMTs, filtered with identical passive components. The drop-in converters, as illustrated in Figure 6, are connected to 10 W load resistors on the smart module and monitored in-situ via a network of analog-digital converters connected to the SPI inputs of the µCSP to collect waveform data. A redundant SPI connection interfacing the GaN sub-experiment to the *head node* is available for failover.

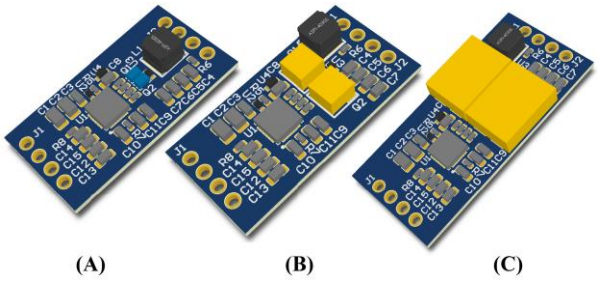

Figure 6: Experimental PoL Converters based on (A) EPC2014C, (B) FBG04N08A, (C) EVG100E15 GaN HEMTs Model

The power card is composed of rad-hard components that provide electromagnetic interference (EMI) filtering and the power rails (12V0, 5V0, and 3V3) necessary to operate the flight cards and cameras. The STP-H6 pallet provides power (28V0) to SSIVP. The power card is illustrated in Figure 7.

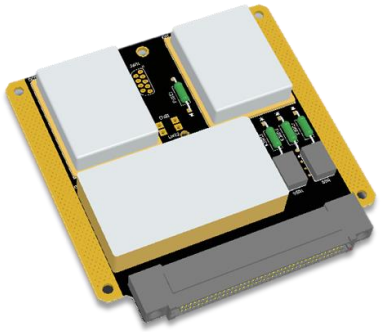

Figure 7: Power Card Model

All flight cards connect onto one backplane interconnect board. The backplane provides interconnects for all interfaces and distributes the power rails provided by the power card. The flight box exposes five external connectors: one RS-422 connector (that interfaces with the STP-H6 pallet for telemetry and commanding); one power connector (that provides power from the STP-H6 pallet); and three debug connectors (JTAG, UART, and status for integration and testing). The external connectors are connected to the backplane by soldered flylead wires. Additionally, the backplane has two internal Camera Link connectors for interfacing both cameras. The backplane is illustrated in Figure 8 and Figure 9.

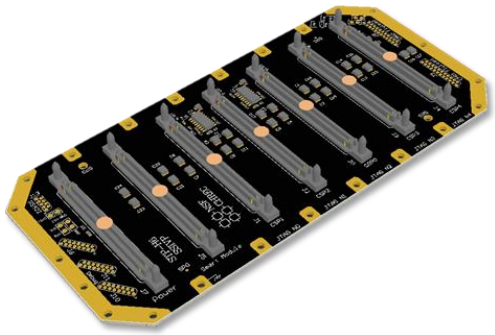

Figure 8: SSIVP Backplane Model

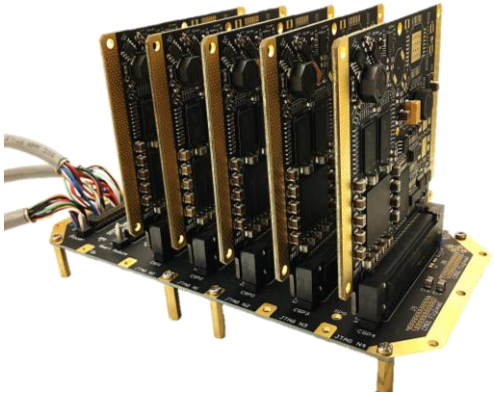

Figure 9: SSIVP FlatSat

Network Topology

The SSIVP backplane interconnects realize two distinct networks. The primary network connects CSPv1 nodes with point-to-point differential signals for high-performance communication. The secondary network connects CSPv1 nodes with point-to-point, single-ended signals for low-power, failover communication. Although the interconnects are physically fixed in the backplane, the networking interface (network peripheral and IO resources) reside in the Zynq FPGA and can be reconfigured post-launch. This reconfigurability enables future research in space networking and mapping network topologies for other data-flow paradigms in parallel computing. The baseline network topology is illustrated in Figure 10, where each node represents a CSP flight card and the number attributed to each directional link denotes the number of data interconnects allocated for that direction.

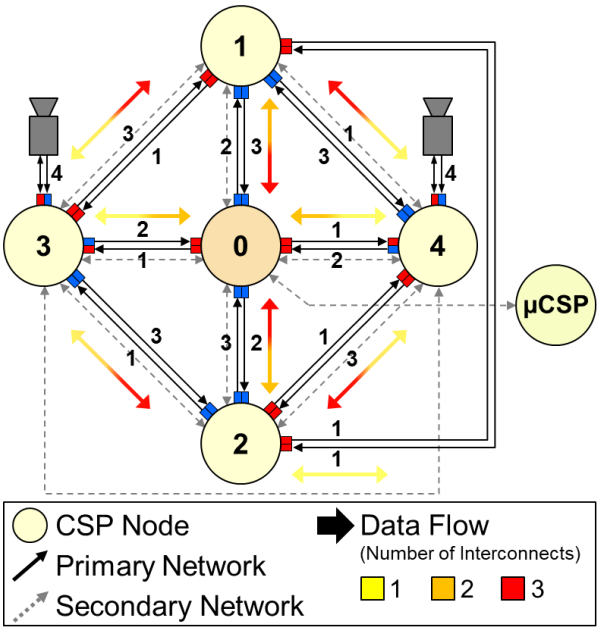

Figure 10: SSIVP Network Topology

Initially, the primary network is designed to optimize communication for scatter-gather applications. The transmission and receiving bandwidths are asymmetric by interface to efficiently map the network topology to the data flow of this parallel programming paradigm. The *camera nodes* have relatively more transmission bandwidth for enhanced scattering of captured image data to adjacent nodes for processing. The *head node* has relatively more receiving bandwidth for enhanced gathering of processed image data from adjacent nodes for downlink. A custom networking interface, called Sabo-Link, leverages the serializer and deserializer (SerDes) resources in the Zynq FPGA for high-performance serial communication between the CSPv1 nodes. Sabo-Link provides a protocol for link connectivity, flow control, 8b10b coding, and parallel bitstream packetization. Sabo-Link also supports variable interconnect widths for the transmitter and receiver interfaces to accommodate alternative network topologies post-launch.

The secondary network is a low-performance, low-power, failover network of UART connections. A serial point-to-point protocol (PPP) is used for communication over UART. A single UART connection links the *head node* with the µCSP board.

IV. SSIVP SOFTWARE DESIGN

This section describes the flight software for the CSPv1 and µCSP compute nodes. The mission manager, for autonomous planning and task scheduling, and the ground-station software, for interacting with SSIVP on-board the ISS, are also discussed.

Flight Software

The CSPv1 flight software consists of Wumbo Linux, a custom operating system based on Xilinx's Linux kernel fork and a BusyBox userland, and a collection of cFE/cFS applications and services. Wumbo Linux is lightweight and resides completely in main memory, using a RAM filesystem (tmpfs). Wumbo Linux is developed using Buildroot.

Wumbo Linux flight images contain NASA GSFC's cFE/cFS framework and applications for platform-independent and reusable flight-quality software. Figure 11 illustrates the cFE/cFS software architecture on CSPv1 for SSIVP.

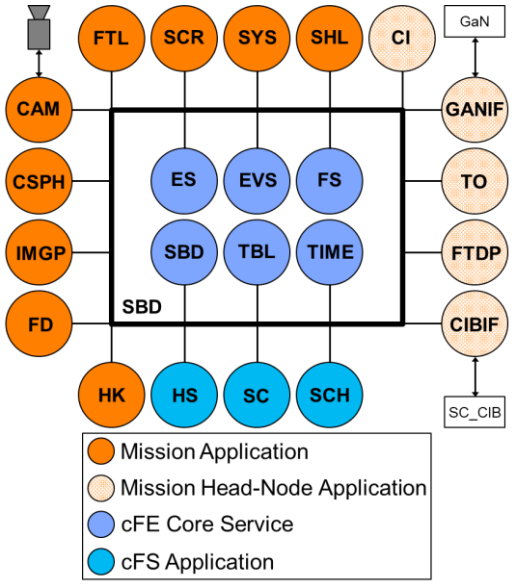

Figure 11: Flight Software Architecture on CSPv1

The cFS applications adopted for SSIVP include Housekeeping (HK), Health and Safety (HS), Stored Commands (SC), and Scheduler (SCH). Additionally, several custom cFE applications unique to this mission were developed, including Camera Control (CAM), Command Ingest (CI), CIB Interface (CIBIF), CSP Health (CSPH), File Downlink (FD), File Transfer Data Processing (FTDP), File Transfer Local (FTL), GaN Interface (GANIF), Image Preparation (IMGP), Scrubber (SCR), Shell (SHL), System Commands (SYS), and Telemetry Output (TO). These applications are described in Table 1.

Table 1: Description of Mission Applications

App	Function
CAM	Controls the connected camera
CI	Disseminates commands from the ground
CIBIF	Handles CIB-specific interactions and data formatting
CSPH	Gathers health information for each node (e.g., Zynq SoC temperature & memory usage statistics)
FD	Downlink files to the ground
FTDP	Interacts with SC_CIB's File Transfer Data Processing interface for file uploads
FTL	Transfers files between nodes
GANIF	Communicates with the GaN experiment
GND	Controls the serial link with the SC_CIB
IMGP	Prepares images for downlink
SCR	Scrubs and reconfigures Zynq FPGA
SHL	Enhanced OS shell interface
SYS	OS-specific management commands
TO	Collects mission telemetry for downlink

SBD is an augmented version of the SB developed by our CHREC team that adds inter-node communication support to cFE's messaging system, without application redesign. SBD uses the peer discovery and publish/subscribe messaging features in the Object Management Group's (OMG) Real-time Publish-Subscribe (RTPS) protocol. Message identifiers from the Consultative Committee for Space Data System's (CCSDS) Space Packet Protocol are used as topics.

A complete boot image contains the first-stage bootloader (FSBL), second-stage bootloader (U-Boot), FPGA bitstream, and flattened image tree blob (Wumbo Linux image). CSPv1 is configured to boot from on-board, radiation-tolerant NAND flash memory. For a reliable boot process, redundant golden boot images are stored in a read-only partition of the flash memory. The RSA authentication feature of the Zynq is used to verify boot image integrity prior to booting, with failover to redundant images. The subsequent partition stores new boot images uplinked post-launch to completely reconfigure the operating flight software and FPGA design.

The µCSP flight board runs µWumbo, a diminutive variation of Wumbo Linux. µWumbo is based on µClinux, a Linux kernel configured for microcontrollers without a memory management unit (MMU), and a BusyBox userland. µWumbo contains application software for periodically reporting health, status, and sensor (thermosensor and accelerometer) information to the *head node* over PPP on a serial link. The µWumbo boot image is stored in on-board, rad-hard NOR flash memory. The bootloader, U-Boot, is stored in on-chip, non-volatile memory. The FPGA bitstream is stored in on-chip, flash-based FPGA configuration memory.

Autonomous Mission Executive

SSIVP includes mission management software that serves to autonomously manage scheduled tasks and appropriately handle unexpected events (e.g., software and hardware faults). Tasks (e.g., function calls, cFE commands, or triggers for external applications) are attributed with time constraints, priority levels, and resource requirement parameters. The task schedule is routinely processed, and tasks are executed as specified while keeping track of the status and resource usage (e.g., sensors, memory, power, etc.) of running tasks. If required resources are unavailable, the scheduler can defer or abort the task execution. Upon completion, tasks are either removed from the schedule or modified with new time constraints if configured to be a routine task. The mission manager is built into a cFE application. Tasks can be added or removed by issuing commands to the cFE application over SBD, or by file reference.

Ground-Station Software

A local ground station is setup to monitor and command SSIVP on-board the ISS. In the backend, NASA Marshall Space Flight Center's (MSFC) Telescience Resource Kit (TReK) is deployed for telemetry, command, and local database capabilities on a local host computer [12]. TReK exposes an API for user applications and tools for handling telemetry and command packets. In the frontend, the Comprehensive Open-architecture Solution for Mission Operations Systems (COSMOS) framework is used for its visualization and support tools [13]. COSMOS provides a graphical user interface to monitor, analyze, and command SSIVP.

V. PARALLEL COMPUTING

This section describes all major communication and computing subsystems in SSIVP that enable high-performance, parallel, and distributed computing. This

includes frameworks for inter-node communication and FPGA-based hardware acceleration, in addition to several existing resources facilitating high-performance computing. Furthermore, a dynamic, heterogenous-aware task scheduler for optimal parallel processing on SSIVP is discussed.

Operating Framework

The operating framework for SSIVP, as illustrated in Figure 12, includes custom networking and acceleration stacks for Wumbo Linux. The networking stack includes the FPGA-based, Sabo-Link network interface and Linux networking driver to provide high-performance communication between CSPv1s. The network driver enables support for native networking applications and libraries, including OpenMPI. OpenMPI is a message-passing interface (MPI) library that provides inter-node communication and synchronization over a network for parallel computing.

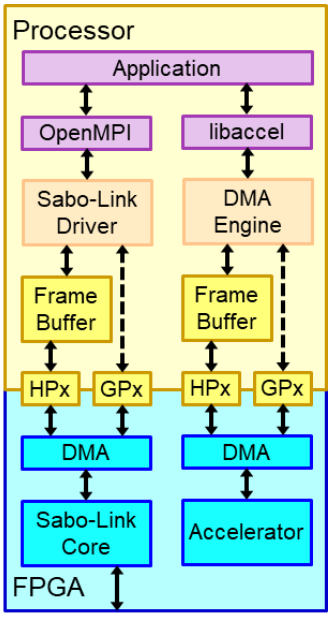

Figure 12: Operating Framework

The acceleration stack includes an FPGA-based image or video hardware accelerator, a DMA-engine character device driver, and a custom userspace library, called libaccel. Hardware accelerators are coded manually using *hardware description language* (HDL) or with *high-level synthesis* (HLS) and leverage the AXI4-Stream interconnect for high-performance stream processing of image data. *Partial reconfiguration* (PR), a feature supported in the Zynq FPGA, is used to alternate between FPGA sub-designs at runtime without interrupting the remainder of the system (e.g., CPU and other FPGA circuits). The FPGA hardware accelerators are constrained to *PR regions* (PRR), which are reconfigured at runtime to enable use of a larger suite of hardware-accelerated applications. The DMA-engine driver offloads image data to the hardware accelerators asynchronously, freeing the processor to perform other tasks in parallel. Finally, the libaccel library provides userspace applications with shared access to the hardware accelerators when available.

In addition to the networking and acceleration stacks, SSIVP enables additional libraries and resources for high-performance computing, including OpenMP and NEON intrinsics. OpenMP is an API for shared-memory multiprocessing, and it provides multithreaded processing to exploit parallelism in fork-join oriented application algorithms. NEON intrinsics access the NEON engines in the ARM Cortex-A9 processor for single-instruction, multiple-data (SIMD) acceleration. Applications on SSIVP can employ one or more components of this framework to achieve maximum performance.

Task Scheduler

SSIVP is a reconfigurable, heterogeneous platform. Application workloads can be processed on the ARM Cortex-A9 processor of the Zynq SoC or offloaded to the FPGA for hardware acceleration. Since FPGA resources are limited and partial reconfiguration is used to alternate between hardware accelerators at runtime, the availability of resources may vary over time.

To optimize OpenMPI application workload distribution across the system, a dynamic, heterogeneous-aware task scheduler is used. The scheduler uses runtime information about the system, including resource availability and utilization, in addition to information known in advance, such as resource performance, networking performance, and partial reconfiguration time, to make scheduling decisions.

Performance and Power Results

To demonstrate the feasibility of parallel computing on SSIVP, a few sample multiprocessor applications were processed on the SSIVP FlatSat for a varied number of compute nodes. For each application, a *camera node* (CSP3) scatters horizontal partitions of a 2448×2050 24-bit RGB image amongst itself and its adjacent nodes (CSP0, CSP1, CSP2). Next, each participating node processes the partition received, and optionally, uses the FPGA hardware accelerators. Finally, the *head node* (CSP0) then gathers all processed partitions and assembles the complete processed image. The sample applications include 2D convolution, bilateral filter, and discrete wavelet transform. These applications were executed for one (CSP0), two (CSP0+CSP3), and four (CSP0+CSP1+CSP2+CSP3) compute nodes, with

scattering and gathering mapped to the SSIVP network topology. The execution times due to parallelization for several system configurations are shown in Figure 13 and Figure 14. For compute-intensive applications, the results show near-linear speedup with the number of nodes. Hardware acceleration also demonstrates significant speedup, however, due to this compute time speedup, communication overhead dominates the execution time for parallelized, hardware-accelerated applications, explaining the nonintuitive slowdown in some cases. Combined, both multicore and FPGA-accelerated parallel computing enable larger and more complex image and video processing applications.

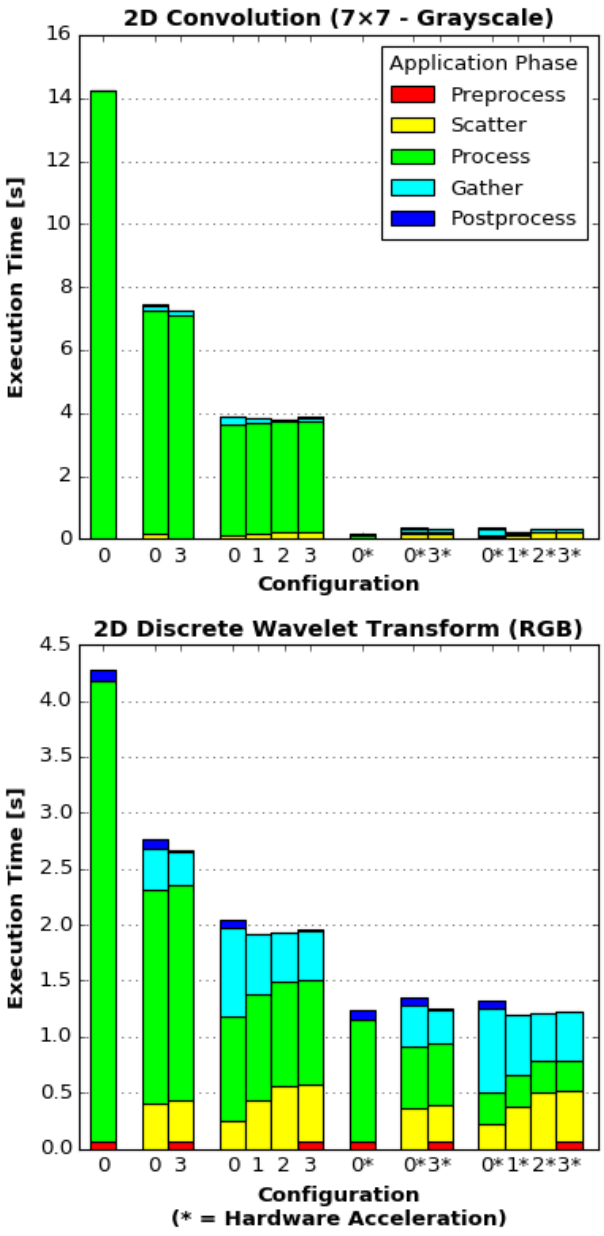

Figure 13: Execution Time Profile

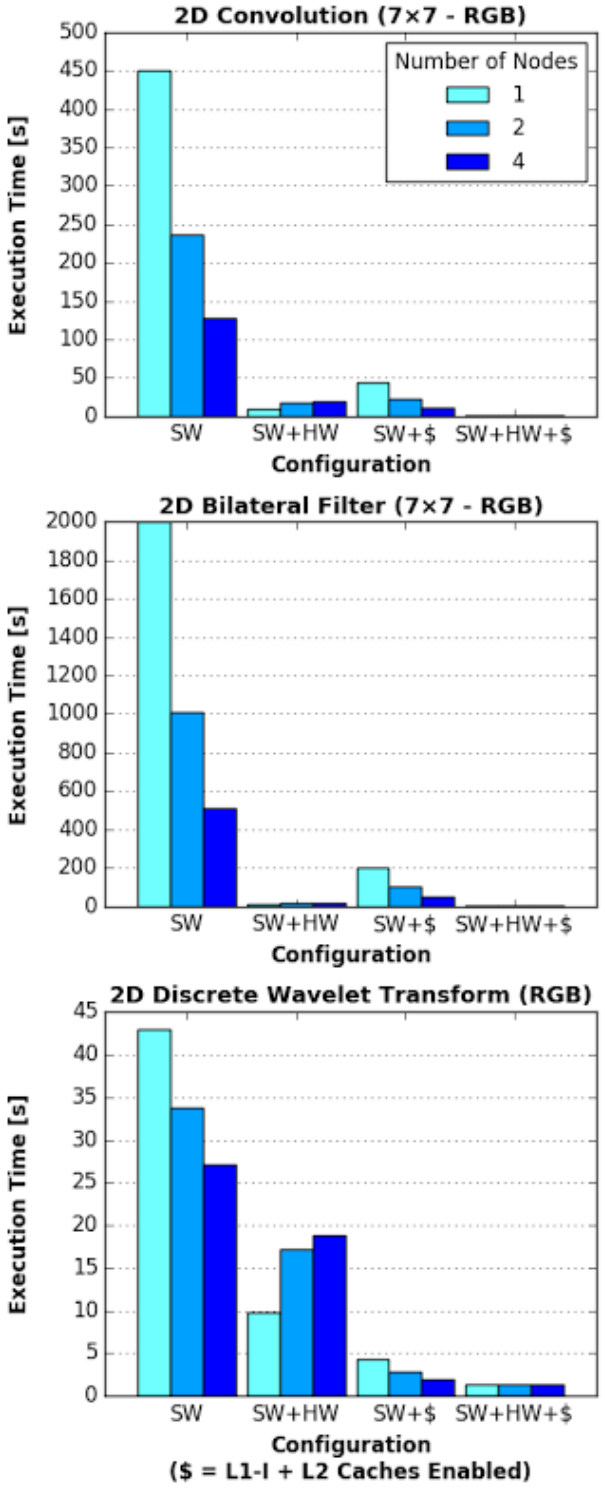

Figure 14: Execution Time Comparison

SSIVP was also measured to obtain preliminary network bandwidth and power usage. The overhead associated with inter-node communication (e.g., scatter-gather) in parallel applications depends significantly on the performance of the networking stack. The iPerf

utility was used to measure the bandwidth for Sabo-Link at 125 Mbps, 175 Mbps, and 215 Mbps for one, two, and three interconnects, respectively. Finally, the SSIVP FlatSat (including five populated CSPv1 nodes and an external evaluation board) measured at 13.65 W idle power and 14.75 load power.

VI. DEPENDABLE COMPUTING

Aside from dependability offered by the CSPv1 and µCSP architectures, two additional experiments are discussed. Adaptive fault tolerance explores environmentally aware reconfigurable systems that accommodate to a changing environment. Fault-tolerant applications and redundant scheduling provide improved data integrity and recovery options for parallel applications.

Adaptive Fault-Tolerance for Hybrid SoCs

SSIVP will include the Hybrid, Adaptive, Reconfigurable Fault Tolerance (HARFT) framework for mitigating SEEs and adapting CPU and FPGA resources to accommodate the current state of the environment [14]. The HARFT framework is illustrated in Figure 15, and comprises of configuration manager, CPU resources, and FPGA resources.

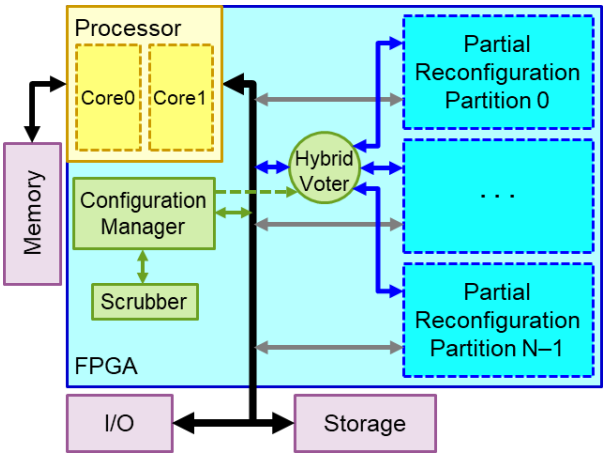

Figure 15: HARFT Framework

The configuration manager uses an FPGA scrubber to detect and correct *single-event upsets* (SEU) in the FPGA configuration memory, frame by frame. A *frame* is the most discrete accessible unit of configuration memory on Xilinx FPGAs. The configuration manager monitors for SEUs detected by the scrubber and uses an aging-window algorithm to approximate the SEU rate for the current state of the environment. This rate is then used to adapt CPU and FPGA resources to maximize overall system performance while providing sufficient redundancy for the current state of the environment.

In the CPU, cores can run in symmetric multiprocessing (SMP) or asymmetric multiprocessing (AMP). CPU cores in SMP operate jointly under one operating system (e.g., Linux), which can improve the performance of multithreaded applications. However, SEEs on a single core can render the operating system inoperable. CPU cores in AMP operate independently under separate operating environments (e.g., Linux + RTOS or Linux + baremetal). With AMP, corruption of one core's execution does not necessarily render both cores inoperable, and thus, fault-tolerance schemes can be explored. AMP on the Zynq SoC is enabled by the OpenAMP and libmetal framework.

In the FPGA, the PRRs are populated with FPGA hardware accelerators or redundant, lockstepped soft-core processors in varied redundancy schemes. The static logic of the FPGA (i.e., FPGA resources not residing in the PRRs) is triplicated using the BYU EDIF Tool [15].

SSIVP uses a hybrid scrubber based on the BYU hybrid scrubbing approach [16]. This scrubber has two stages: the FPGA-based Xilinx Soft Error Mitigation (SEM) controller readback scrubber; and a custom software-based replacement scrubber. The SEM controller rapidly detects and corrects SEUs in the FPGA configuration memory using CRC and ECC codes. When an uncorrectable error has been detected, the software-based scrubber is invoked to replace the entire frame with a golden frame residing in main memory.

Redundant Scheduling and Dependable Applications

In addition to scheduling for parallelism, the task scheduler in SSIVP enables scheduling for redundancy. Application-level workloads can be attributed for redundant scheduling on-demand, or optioned for rescheduling after the application has failed. Future planned research will investigate software redundancy for dependable applications, including user-directed methods in tolerating node failures for parallel applications. The fault-tolerance capabilities proposed for standardization in the MPI specification are candidates for this research.

VII. MISSION OBJECTIVES

The SSIVP experiment has several mission requirements necessary to constitute mission success. SSIVP will provide validation for new technologies and experimental research in high-performance and fault-tolerant space computing.

One directive is to advance the TRL of the CSPv1 Rev. C and µCSP Rev. A flight boards in LEO. The CSPv1 platform, and its featured Xilinx Zynq SoC, has been adopted by a growing number of groups in the CubeSat

community. CSPv1 is featured on the NASA CeREs heliopysics science CubeSat, the Lockheed-Martin SkyFire Lunar Flyby mission, on PlanetIQ's weather satellites, and several other planned missions. µCSP, and its featured SmartFusion2 SoC, will accommodate missions needing high-performance processing but with more stringent SWaP-C requirements.

SSIVP will acquire a profile of daytime and nighttime full-resolution images overlooking areas of Earth. One key directive is to validate SSIVP as a supercomputing platform capable of parallel and distributed computing. This capability includes image and video applications (with multicore, NEON, or FPGA parallelism), frameworks, schedulers, and networks. SSIVP features two high-resolution (5 megapixel) color cameras with different field-of-views (25 mm and 75 mm). Processed images are downlinked as thumbnails or full-resolution images for analysis.

As a reconfigurable platform, another objective for SSIVP is to upload new software and FPGA components post-launch. Each CSPv1 flight board is capable of post-mission uploads (e.g., individual programs or FPGA partial bitstreams) to complete system upgrades (e.g., boot images or FPGA full bitstreams) as desired. This reconfigurability enables SSIVP to be a continuous development platform for subsequent research experiments after primary objectives have been completed.

SSIVP will monitor and record SEUs in the CSPv1 and µCSP flight boards. This tracking includes the CPU and FPGA subsystems of the featured hybrid-processor, and possibly other on-board components. The system will provide environmental information essential for characterizing the reliability of these platforms in LEO. An extended objective is to automate SEU monitoring and adapting on-board CPU and FPGA resources to accommodate to the changing environment.

Finally, the SSIVP GaN sub-experiment will exercise and monitor the resilience of six hybrid PoL converters (three GaN HEMTs and two synchronous buck controllers) in LEO. The µCSP hybrid-processor will collect waveform data to be downlinked for analysis. This data will provide insights on GaN PoL converters as an alternative power solution for future spacecraft.

VIII. CONCLUSIONS

As missions continue to adopt SmallSat technology, there is a need for dependable and high-performance computing. In this paper, we introduced the SSIVP experiment for the STP-H6 mission, including flight software and hardware, frameworks for parallel and dependable computing, and mission objectives. SSIVP will provide validation for the flight system and frameworks for high-performance computing.

This mission will advance the TRL of the CSPv1 Rev. C and µCSP Rev. A space processors to TRL9 in LEO, alongside the CSPv1 Rev. B currently on-board the ISS as part of STP-H5. Valuable radiation data will be collected to gain insights on the CSPv1 and µCSP platforms in this environment for future improvements in the space processor design. Furthermore, radiation data for the GaN sub-experiment will provide insights to alternative power solutions for future spacecraft. Following comprehensive mission success, SSIVP will become a continuous development platform for uploading new software, FPGA designs, and system configurations, accelerating the development and validation of future experiments.

Acknowledgements

This research was funded by industry and government members of the NSF CHREC center, and the National Science Foundation I/UCRC Program under Grant No. IIP-1161022. The authors would like to thank Nicholas Franconi, Christopher Sciosia, Thomas Cook, and Victoria Dulla for assistance in designing the electronics hardware for SSIVP, Schmidt David, Kevin Glunt, Theo Schwarz, Ryan Blair, Matthew O'Connor, David Fudurich, James Ciabattoni, Rina Zhang, and Paul Gatto for designing the mechanical components of SSIVP, and Matthew Barry for thermal analysis. The authors would also like to recognize the STP Houston team for providing support, design assistance, and flight opportunity. Finally, the authors would like to acknowledge Milton Davis and his team at NASA GSFC Code 596 for assembly design support and guidance.

References

1. Sexton, F.W., "Destructive single-event effects in semiconductor devices and ICs," IEEE Transactions on Nuclear Science, Jun. 2003.
2. National Academies of Sciences, Engineering, and Medicine, "Achieving Science with CubeSats: Thinking Inside the Box," The National Academies Press, Washington, DC, 2016.
3. Sims, E., "The Department of Defense Space Test Program: Come Fly with Us," Proceedings of IEEE Aerospace Conference, March 2009.
4. McLeroy, J., "Highlights of DoD Research on the ISS," ISS Research and Development Conf., Jun. 26-27, 2012.

5. *coreflightexec*, NASA Goddard Space Flight Center, May 25, 2017. [Online]. Available: https://sourceforge.net/projects/coreflightexec/

6. Rudolph, D., Wilson, C., Stewart, J., Gauvin, P., George, A.D., Lam, H., Crum, G., Wirthlin, M., Wilson, A., and A. Stoddard, "CSP: A Multifaceted Hybrid Architecture for Space Computing," Proc. of the 28th Annu. AIAA/USU Conf. on Small Satellites, Logan, UT, Aug. 2-7, 2014.

7. Wilson, C., Stewart, J., Gauvin, P., MacKinnon, J., Coole, J., Urriste, J., George, A.D., Crum, G., Timmons, E., Beck, J., Flatley, T., Wirthlin, M., Wilson, A., and Stoddard, A., "CSP Hybrid Space Computing for STP-H5/ISEM on ISS," Proc. of the 29th Annu. AIAA/USU Conf. on Small Satellites, Logan, UT, Aug. 8-13, 2015.

8. Wilson, C., MacKinnon, J., Gauvin, P., Sabogal, S., George, A.D., Crum, G., and Flatley, T., "μCSP: A Diminutive, Hybrid, Space Processor for Smart Modules and CubeSats," Proc. of the 30th Annu. AIAA/USU Conf. on Small Satellites, Logan, UT, Aug. 6-11, 2016.

9. Reusch, D., Gilham, D., Su, Y., and Lee, F.C., "Gallium Nitride based 3D integrated non-isolated point of load module," 27th Annu. IEEE Applied Power Electronics and Exposition (APEC), Orlando, FL, 2012.

10. Scheik, L., "Single-event effect report for EPC Series eGaN FETs: EPC2015, EPC2014, EPC2012," Jet Propulsion Laboratory, National Aeronautics and Space Administration, Pasadena, CA, 2014.

11. Patterson, R., Lauensetin, J-M., Casey, M., Scheik, L., and Hammoud, A., "Radiation & Thermal Cycling Effects on Gallium Nitride and Silicon Carbide Power Transistors," in NEPP 4th Electronics Technology Workshop, National Aeronautics and Space Administration, NASA Goddard Space Flight Center, 2013.

12. *Telescience Resource Kit*, NASA Marshall Space Flight Center, May 25, 2017. [Online]. Available: https://trek.msfc.nasa.gov/

13. Sorensen, T.C., Pilger, E.J., Wood, M.S., Nunes, M.A., and Yost, B.D., "Development of a Comprehensive Mission Operations System Designed to Operate Multiple Small Satellites," Proc. of the 25th Annu. AIAA/USU Conf. on Small Satellites, Logan, UT, Aug. 8-11, 2011.

14. Wilson, C., Sabogal, S., George, A.D., and Gordon-Ross, A., "Hybrid, Adaptive, and Reconfigurable Fault Tolerance (HARFT)," 2017 IEEE Aerospace Conf., Big Sky, MT, Mar. 4-11, 2017.

15. *BYU EDIF Tools Homepage*, Brigham Young University, May 25, 2017. [Online]. Available: http://reliability.ee.byu.edu/edif/

16. Stoddard, A., Gruwell, A., Zabriskie, P., Wirthlin, M., "A hybrid approach to FPGA configuration scrubbing," 2016 IEEE Transactions on Nuclear Science, 2016.