A Framework for Evaluating and Optimizing FPGA-Based SoCs for Aerospace Computing

NICHOLAS WULF, ALAN D. GEORGE, and ANN GORDON-ROSS,
NSF Center for High-Performance Reconfigurable Computing (CHREC), University of Florida

On-board processing systems are often deployed in harsh aerospace environments and must therefore adhere to stringent constraints such as low power, small size, and high dependability in the presence of faults. Field-programmable gate arrays (FPGAs) are often an attractive option for designers seeking low-power, high-performance devices. However, unlike nonreconfigurable devices, radiation effects can alter an FPGA's functionality instead of just the device's data, requiring designers to consider fault-tolerant strategies to mitigate these effects. In this article, we present a framework to ease these system design challenges and aid designers in considering a broad range of devices and fault-tolerant strategies for on-board processing, highlighting the most promising options and tradeoffs early in the design process. This article focuses on the power, dependability, and lifetime evaluation metrics, which our framework calculates and leverages to evaluate the effectiveness of varying system-on-chip (SoC) designs. Finally, we use our framework to evaluate SoC designs for a case study on a hyperspectral-imaging (HSI) mission to demonstrate our framework's ability to identify efficient and effective SoC designs.

CCS Concepts: • **Hardware** → **Modeling and parameter extraction**; *Chip-level power issues*; • **Computer systems organization** → *Availability*

Additional Key Words and Phrases: Aerospace, fault-tolerant, FPGA, Pareto optimal, single-event upset, system design

ACM Reference Format:
Nicholas Wulf, Alan D. George, and Ann Gordon-Ross. 2016. A framework for evaluating and optimizing FPGA-based SoCs for aerospace computing. ACM Trans. Reconfigurable Technol. Syst. 10, 1, Article 1 (September 2016), 29 pages.
DOI: http://dx.doi.org/10.1145/2888400

1. INTRODUCTION

Unmanned, remote-sensing systems are commonly used in air and space environments to sense and collect raw data from the surrounding environment. The system typically then transmits the collected data to a central ground station where high-performance computers process and analyze the data. However, rapidly improving sensor technology has significantly increased the amount of collected data, which may exceed the remote system's transmission bandwidth. Additionally, because remote systems are continually exploring farther-reaching areas, transmission latencies can be on the

This work was supported in part by the I/UCRC Program of the National Science Foundation under Grant No. IIP-1161022. The authors gratefully acknowledge vendor equipment and tools provided by Xilinx that helped make this work possible.
Authors' addresses: N. Wulf, P.O. Box 116200, Gainesville, FL 32611; email: wulf@hcs.ufl.edu; A. D. George, 968 Center Drive, Gainesville, FL 32611; email: ageorge@ufl.edu; A. Gordon-Ross, P.O. Box 116200, Gainesville, FL 32611; email: ann@ece.ufl.edu.

DOI: http://dx.doi.org/10.1145/2888400

order of tens of minutes or more, which hinders remote systems that rely on real-time operating decisions from a ground station.

In order to address increasing bandwidth pressure and transmission latencies, remote systems include on-board processing capabilities to process the raw data in situ and transmit only the smaller, processed data. Additionally, on-board processing empowers remote systems to perform the necessary calculations for making intelligent autonomous operating decisions in real time, thereby reducing the need for high-latency instructions from a distant ground station.

However, incorporating on-board processing into an aerospace mission is challenging when considering stringent size, weight, and power (SWaP) constraints. Power is often the most limiting of these constraints because energy is difficult to collect and store, and increasing the processing performance increases the power consumption. Challenges in aerospace also include radiation effects, which cause unexpected and erroneous behaviors in processing systems and may be exacerbated by decreasing feature sizes and an increasing number of processing elements. Field-programmable gate arrays (FPGAs) are often an attractive option for designers seeking low-power, high-performance devices, but additional considerations are required to mitigate radiation effects. Unlike nonreconfigurable devices, radiation effects can alter the device's functionality instead of just the device's data. Therefore, once a designer has defined an aerospace mission's system platform, environment, and applications (e.g., hyperspectral imaging (HSI), real-time landing, obstacle avoidance), the primary design challenge is device and fault-tolerant (FT) strategy selection. The device must perform well with the mission's applications and be capable of operating effectively in the mission's environment. An appropriate FT strategy is also necessary for most missions in order to guarantee correct operation without excessive resource overhead.

A successful design of an on-board processing system meets or exceeds all mission constraints (maximum power usage, maximum fault rate, minimal processing throughput, etc.). Since these mission constraints have different, and often competing, tradeoffs, the set of successful designs contains many Pareto-optimal designs [Branke et al. 2008]. The designer must choose the best design based on the mission constraints and acceptable tradeoffs. For example, because mission failure may be catastrophic (e.g., loss of life), a designer may increase the system's power consumption in order to lower fault rates. Alternatively, for a sensor-based mission, faults may cause superficial damage to the mission's data (e.g., a few discolored pixels), so sacrificing fault tolerance for increased processing performance might be advantageous. Not only is determining the best design a complex task, but also the designer's reliance on familiar devices, FT strategies, and development-time constraints often limits the design exploration space, which may preclude time to explore new devices and FT strategies. These limitations narrow the design space's scope, possibly resulting in successful yet non-Pareto-optimal designs.

Designers evaluate system designs using evaluation metrics, such as performance, power, dependability, device utilization, mission lifetime, and design cost. Once performance constraints have been met, power, dependability, and lifetime are often the most critical evaluation metrics for on-board processing systems in constrained and harsh environments. Therefore, our work focuses on these three metrics; however, additional metrics could be incorporated. The power metric measures how much power the processing device will consume during the mission. The dependability metric quantifies a system's ability to correctly operate within the mission environment, which designers often represent as the mean time to failure (MTTF), mean time between failures (MTBF), or data-loss rate. The lifetime metric estimates the expected duration of time for which the design will remain functional.

To aid designers in addressing the challenges of designing on-board processing systems, we present a novel framework that determines a set of Pareto-optimal system designs in terms of device and FT strategy based on a mission's constraints. Although we designed our framework to consider a wide range of processing devices, this article focuses on our framework's methodology and analysis with respect to FPGA devices. Furthermore, while this article focuses on system-on-chip (SoC)-level design, the scope of our framework is easily expandable to include other board- and platform-design factors. Our framework considers four system properties: mission, application, device, and FT strategy. The designer specifies the mission and application properties. The mission property defines information about the mission environment and dictates the resources and constraints of the on-board processing system based on design constraints and available platform resources (e.g., sensors, power generation, memory capacity). The application property defines the on-board processing tasks, which are typically sensor data processing and autonomous-operation decisions (i.e., autonomous processing). Once these system properties have been defined within, our framework analyzes these properties with respect to all device and FT strategy combinations stored a priori in our framework's database. From this analysis, our framework produces metric data for power, dependability, and lifetime to determine the Pareto-optimal system designs.

Since the necessary radiation data can be difficult to obtain and/or may not be publicly available for some devices, in these situations our framework's predications may not be Pareto optimal. When radiation data is missing, designers can make educated estimates about this data based on attainable data for similar parts, enabling our framework to determine Pareto-optimal system designs based on these interim estimates. Obviously, results based on these estimates would only be as accurate as the estimates themselves, so any recommended devices would still need to eventually undergo radiation testing before being used in a final design. Furthermore, if and when new data is obtained or made publicly available, the designer can then update the data, enabling our framework to predict more accurate Pareto-optimal system designs. We identify and explain our proposed estimation process in Section 5, where we discuss the article's case studies.

Although our framework uses a resource optimization methodology to estimate the maximum performance of the FPGAs under study, this article does not focus on finding the optimal FPGA designs for a particular application. Instead, our framework uses the maximum performance estimate to prune the design space and focuses instead on finding the optimal set of devices and FT strategies for a particular mission and application.

This article extends upon our previous work [Wulf et al. 2012] in several ways. Previous work focused on power and dependability metrics, with dependability referring to a design's MTBF due to soft-error upsets. Once detected, these soft-error upsets can be repaired through resetting or reconfiguring the device. Conversely, other radiation effects such as total ionizing dose (TID) build up over time in a device, eventually leading to catastrophic failure. Therefore, we added the lifetime metric to our framework to measure a design's response to environmental TID levels and predict a mission's operational lifetime. We also improved the power evaluation metric to account for temperature in calculating a device's static power consumption. This article also extends the previous case study on the EO-1 Hyperion mission by including the lifetime metric in the analysis and greatly expanding the set of devices under study. The previous case study focused on six device families of interest, but due to the effort of manually evaluating every studied device, only one device from each family was studied (for consistency, we used the largest device in each family). However, by automating our device evaluation process, we are able to include every device from each family in our case study, a 60-fold improvement. This larger dataset more accurately represents these six families and provides a greater range of device capabilities, ensuring our framework's accuracy in selecting the optimal designs.

The remainder of this article is organized as follows. Section 2 discusses the background and related work that provides the foundation for our framework. Section 3 presents an overview of our framework, and Section 4 discusses our framework's evaluation metrics. In Section 5, we present a hypothetical case study based on an HSI mission to demonstrate our framework's basic methodology and show our framework's full, design-enhancing potential.

2. BACKGROUND AND RELATED WORK

Our framework leverages previous work related to each of the four system properties and introduces a novel evaluation methodology that combines these properties, producing evaluation-metric results to identify and compare Pareto-optimal system designs. This section discusses important background and research related to each of the four system properties.

Pease et al. [1988] discuss appropriate device selection based on an environment's varying radiation levels. A device database stores radiation data for a set of known devices and allows designers to quickly eliminate inappropriate devices. For the device property, our framework leverages a similar device database to store radiation data, with additional data on the device's processing capabilities and power consumption.

Other works have demonstrated methodologies for predicting the optimal performance of an application design on an FPGA device. The RC Amenability Test (RAT) [Holland et al. 2009] is an analytical methodology that uses three tests for throughput performance, numerical precision, and resource utilization to determine the viability of an algorithm design on an FPGA prior to the use of a hardware description language. RAT measures throughput performance with both communication time for transferring data on and off the FPGA and computation time for processing the data according to an application design, which relies on a user-supplied frequency estimation for the FPGA. Enzler et al. [2000] describe a similar high-level estimation methodology for characterizing the area and performance of an application on an FPGA. Using a priori information about the FPGA's architecture, the methodology creates a set of equations to describe area, frequency, throughput, latency, and input/output (I/O) pin count, enabling the user to quickly test the tradeoffs involved in decomposing parts of the design, replicating those parts, or adding registers for pipelining. Meswani et al. [2013] show how to model and predict the performance of high-performance computing applications on systems that use graphics processing unit (GPU) or FPGA hardware accelerators. Their model evaluates the application's code to find sections that could be easily accelerated and uses simple benchmarks to predict accelerator speedup.

Finally, Williams et al. [2010] define a general methodology for determining the maximum processing capabilities of a given device, referred to as computational density (CD). The CD methodology uses the results of single-instantiated operations to predict the frequency and performance of an application on an FPGA without requiring detailed a priori information about the FPGA supplied by the user. Although these single-instantiated operations are highly optimized and/or vendor provided, the combining of these operations into a full application may be complex and result in different overheads and performances depending on the application, specific design, and method of place and routing. Therefore, the FPGA-based CD methodology represents a theoretical upper-bound estimate of the performance of a given FPGA device and application. Furthermore, the full scope of the CD methodology includes a wide range of device architectures (e.g., central processing unit (CPU), digital signal processor (DSP), FPGA, and GPU) and considers operation types as well as precision when calculating the devices' CDs. Our framework leverages this CD methodology to quickly calculate an upper bound for the optimal performance of a device running a particular application. Although there are numerous designs for a particular FPGA that may satisfy the demands of a mission's application (e.g., DSP or logic centric, short or long pipeline length, speed or area optimized), the CD methodology narrows the framework's analysis to a scaled version of the single design that produces the maximum performance. This analysis of a single design prunes the huge design space associated with SoC-design optimization and enables our framework to focus solely on identifying the optimal device and FT strategy combinations. Additionally, the applicability of CD to a wide range of architectures enables extensions to our framework to widen the scope of analysis beyond comparing only FGPA devices.

For Earth-orbiting missions, our framework requires designers to input the mission property's data into CREME96 [Tylka et al. 1997] to predict the average radiation flux experienced by a processing system. Using user-provided radiation data for specific devices, CREME96 also predicts device upset rates based on the radiation flux effects. For other environments, simpler models based on environmental radiation literature can predict the average radiation flux according to the mission property's data.

FT strategies increase software and hardware fault tolerance using redundant calculations and/or data storage, which allows processing systems to operate correctly despite effects caused by upset-inducing radiation. However, this redundancy incurs processing and/or area overheads, which increase as the FT strategy's fault-mitigating capabilities increase. For example, triple-modular redundancy (TMR) [Neumann 1956; Lyons and W. 1962] is capable of detecting and correcting errors and incurs $\sim$200% area overhead. Application-dependent FT strategies can offer fault-mitigating capabilities with lower overheads, such as algorithm-based FT (ABFT) [Huang and Abraham 1984], which leverages the linear properties of common matrix operations to produce checksums that detect errors in the final calculated matrices. Device-dependent FT strategies, such as reconfigurable FT (RFT) [Jacobs et al. 2012b], use an FPGA's partial reconfiguration capabilities and the time-varying nature of orbital radiation to dynamically increase/decrease the fault-mitigating capabilities. FPGAs can also use partial reconfiguration after detecting an upset to repair transient faults in real time with partial scrubbing or repair permanent faults by reprogramming only the damaged areas to avoid the damaged resources. Furthermore, reduced-precision redundancy [Pratt et al. 2013] is an application- and device-dependent FT strategy that may enable an FPGA to have a significant reduction in overhead compared to TMR with only a small loss in dependability. Our framework considers a wide range of FT strategies, which allows designers to evaluate FT strategies with respect to the specific application and device and view tradeoffs between the fault-mitigating capability and performance/area overhead.

Understanding how the application property impacts a device's performance is paramount in selecting the Pareto-optimal designs. For example, FPGAs are effective for bit-level and fixed-point operations, but potentially less effective than fixed-logic devices for double-precision, floating-point operations due to these operations' much higher reconfigurable resource utilization. Asanovic et al. [2006] address this issue for high-performance computing (HPC) systems by identifying 13 common kernels that represent the essential operations of the vast majority of all HPC applications. By subsetting HPC applications based on the applications' constituent kernels, system designers can quickly and effectively study a broad range of applications and application behaviors with little loss of accuracy by focusing on understanding only these 13 kernels. Our framework leverages this subsetting methodology to identify the most common kernels that represent the majority of all on-board processing applications, which allows our framework to analyze a broad range of on-board processing applications without requiring research into each specific application.

Research institutions with a significant history in designing aerospace systems usually have stringent guidelines in place for the part selection process; however, even these established processes can benefit from our framework's analysis. For example, NASA's state-of-the-art part selection process for a new mission begins from the reference-board design of a previously flown mission, which is compared to the necessary capabilities and requirements of the new mission to identify any parts that need to be upgraded or modified. Then, based on the available technology, the designer tries to determine which new parts would best suit the needs of the new mission. Any new electrical parts are rigorously tested and screened as outlined in NASA's EEE-INST-002 document [Sahu et al. 2003]. In this process, our framework would serve as a preliminary analysis tool, enabling the designer to quickly narrow down his or her device choice scope to the most promising processing devices during the new part selection step without relying on ad hoc selection methodologies or only choosing familiar devices.

3. FRAMEWORK

Our framework determines the Pareto-optimal system designs based on the four system properties (device, mission, FT strategy, and application), allowing a designer to select the best design based on his or her desired tradeoffs, regardless of the designer's familiarity with the devices and FT strategies. Although this article focuses on FPGA devices for aerospace environments, our framework can support a wide range of devices (e.g., CPUs, DSPs, FPGAs, GPUs) as well as a diverse set of environments (e.g., outer space, aerospace, underwater) and is easily extendable to additional devices and environments. Furthermore, while this article focuses on SoC-level design, the scope of our framework is easily expandable to include other board and platform design factors (e.g., recent work shows how external memory devices can be included in our framework [Wulf et al. 2015]). The remainder of this section is organized as follows. Section 3.1 presents an overview of our framework, focusing on overall scope, general concepts, and our framework's components; Section 3.2 details the four system-property components; and Section 3.3 discuses the analysis component.

3.1. Overview

Figure 1 depicts an overview of our framework, which is composed of five components. The first four components are the system-property components, which include the device set, the mission characteristics, the FT strategy set, and the application kernel set components and correspond respectively to the device, mission, FT strategy, and application system properties. The fifth component, the analysis component, corresponds to the power and dependability evaluation metrics.

The system-property components consist of both designer-specified data and research data obtained from the literature. Since our framework does not have a priori knowledge of the system platform, environment, and constraints, the designer provides the mission characteristics, and our framework predefines the device set, FT strategy set, and application kernel set based on literature research data (Section 3.2).

The analysis component combines the data of the system-property components and produces evaluation-metric results, which the designer evaluates to select the best design. Each evaluation metric combines the data from the system-property components in a unique method based on the specific evaluation metric's dependency on the interactions of the system-property components. For example, the power metric evaluates device performance with respect to an application, whereas the dependability metric evaluates device radiation-response data with respect to the mission environment. Alternatively, the dependability metric evaluates the fault mitigation capabilities of the FT strategies, whereas the power metric evaluates the performance and area

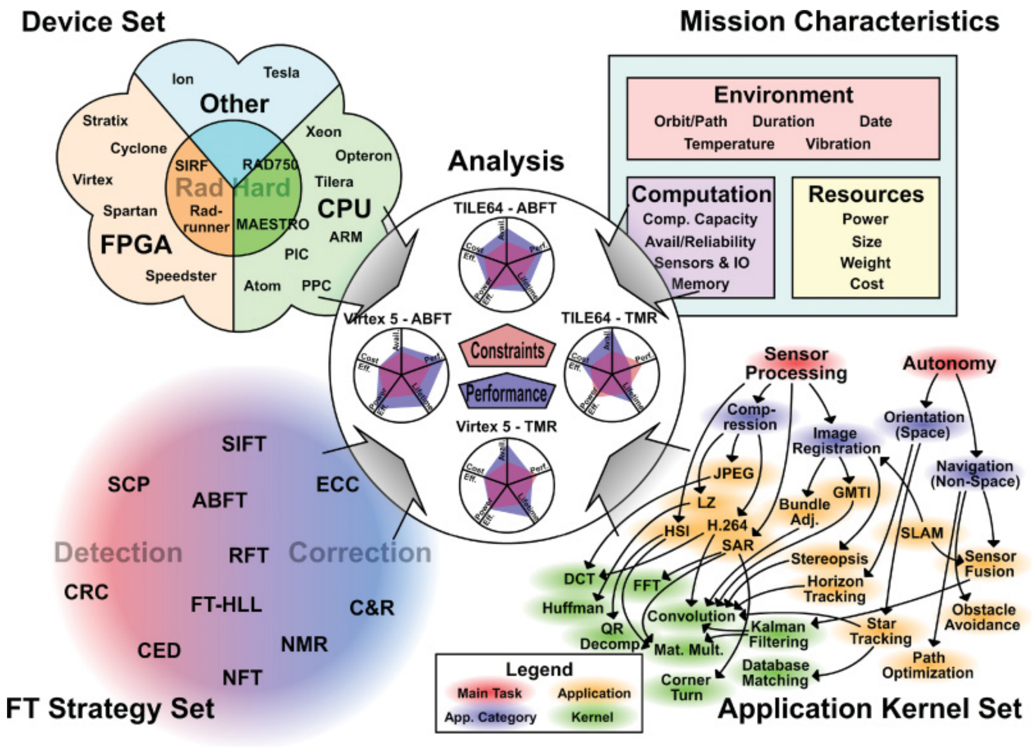


Fig. 1. Framework overview consisting of the four system-property components (corners) and analysis component (center).

overheads of the FT strategies. Finally, evaluation metrics only evaluate valid designs that use device- or application-dependent FT strategies with the corresponding devices and applications.

3.2. System-Property Components

The device set contains a priori data from our framework's database on a broad range of device architectures as well as any available radiation-hardened versions of these devices. The device set's data records three characteristics for each device: power measurements, processing capability, and radiation response. Power measurements include the maximum dynamic power consumption of the device for a given application, the thermal resistance between the inside of the device and the surrounding environment, and information on how the device's temperature affects the device's static power consumption. Our framework represents processing capability using the CD methodology, which depends on the type and precision of the application's operations. The radiation response involves determining the areas of the device that are sensitive to a single radiation particle (proton or heavy ion for space missions) of a given energy. Literature research data provides the radiation-response data, because this data is sufficient for our framework's analysis, and obtaining this data via experimental analysis is difficult and time-consuming.

The mission characteristics define the mission environment, available resources, and computational constraints. Designers must specify this data before our framework can begin mission analysis. The mission environment includes data on the mission's specific path (e.g., an orbit in space or a route along the ocean floor), the mission's duration (e.g., months or years), the mission start date for considering time-dependent environments, and any other harsh conditions that must be considered (e.g., extreme temperatures or excessive vibration). The available resources include the SWaP restrictions and may also include a monetary budget for designing and building the system. Our framework uses the constraints defined in the resource data to test the successfulness of various designs. The computational constraints dictate the acceptable fault rates, required processing throughput based on the incoming sensor data's throughput, and maximum allowable memory usage based on the on-board memory constraints.

The FT strategy set is stored a priori in our framework's database and contains literature research data on the most effective and/or common FT strategies, which includes a wide variety of FT detection and/or correction strategies, some of which are device or application dependent. The FT strategy set records three characteristics for each FT strategy: effectiveness, overhead, and dependencies. The effectiveness is the FT strategy's fault mitigation capability (e.g., detection only, or detection and correction). For example, if a non-fault-tolerant (NFT) system has a 1% chance of experiencing a fault during a certain time interval, adding a TMR FT strategy to the system will correct 97% of these faults over the same time interval. The overhead refers to the extra processing that all FT strategies require due to redundant calculations (e.g., $\sim$200% overhead for TMR). Finally, the dependencies define which devices or applications correspond to a given FT strategy, ensuring that our framework only evaluates valid designs. Adding new FT strategies to our framework only requires the specification of the new methods for calculating the FT strategies' effectiveness and overhead based on the application as well as any application or device dependencies.

The application kernel set is also stored a priori in our framework's database and contains the subset of common kernels (e.g., matrix multiplication and fast Fourier transform) representing the essential operations of the vast majority of on-board processing applications. Identifying the common kernels is a key challenge and important area of research for our framework, which involves analyzing a comprehensive survey of aerospace applications with the goal of identifying the smallest subset of common kernels that encompasses the largest amount of the applications' constituent kernels. If future analysis determines that emerging aerospace applications are not necessarily covered under the current subset of kernels, the subset can easily be expanded to include these new kernels.

In addition to mapping applications to one or more of these kernels, our framework categorizes applications as either sensor processing or autonomous processing. Sensor processing is the processing of the raw data collected from on-board sensors with the purpose of compressing and/or extracting important information before transmission. Autonomous processing is the ability of the on-board processing system to make intelligent decisions and take effective action based solely on in situ analysis of the environment, such as circumnavigating obstacles and locating landing zones. Sensor processing typically focuses on meeting transmission throughput constraints, while autonomous processing focuses on reliably meeting real-time deadlines.

3.3. Analysis Component

Figure 2 shows the analysis component, which uses data from all four of the system-property components to create and output the final Pareto-optimal design set. The designer is responsible for supplying the mission characteristic data to our framework as well as identifying applications and other relevant application parameters (e.g., size of input matrices or arrays), which our framework compares against the application kernel set to understand the application's operations. All data from the device set and FT strategy set exist a priori in our framework's database.

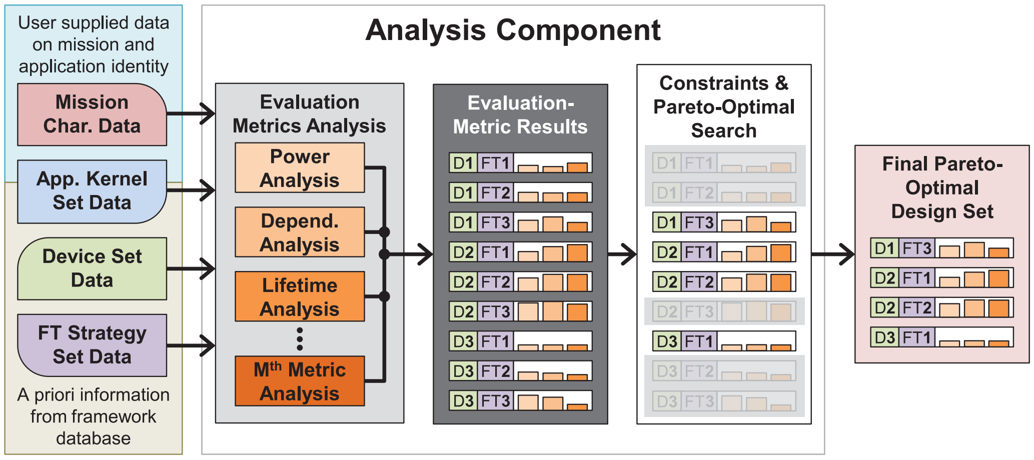

User supplied data on mission and application identity
Mission Char. Data
App. Kernel Set Data
Device Set Data
FT Strategy Set Data
A priori information from framework database
Analysis Component
Evaluation Metrics Analysis
Power Analysis
Depend. Analysis
Lifetime Analysis
Mth Metric Analysis
Evaluation-Metric Results
D1 FT1
D1 FT2
D1 FT3
D2 FT1
D2 FT2
D2 FT3
D3 FT1
D3 FT2
D3 FT3
Constraints & Pareto-Optimal Search
Final Pareto-Optimal Design Set

Fig. 2. Flowchart for the analysis component showing inputs from the four system-property components and an output consisting of the final Pareto-optimal design set.

The analysis component considers every unique and valid device and FT strategy combination as an analyzable design. The analysis component runs each design through all M evaluation metrics, producing M evaluation-metric results for each design. Although the bulk of the analysis occurs automatically within our framework, external tools (e.g., CREME96 and SPENVIS for radiation modeling) may need to be invoked by the designer if automatic controls for these tools have not been integrated into our framework. Although this article focuses on the power, dependability, and lifetime evaluation metrics, these metrics do not represent the entire scope of our framework's capabilities. Any number of evaluation metrics can be added to and used in our framework's analysis if researchers/designers find these metrics relevant to the mission. Furthermore, methods we present for analyzing each evaluation metric are not meant to be final, so future researchers/designers can improve an evaluation metric by using more accurate data sources or replacing the metric's methods with more advanced analysis methods.

A design is successful if the design's evaluation-metric results meet or outperform the mission constraints. After removing all unsuccessful designs that fail the constraints, a Pareto-optimal search selects a small subset of the designs that are Pareto optimal, meaning that these designs are the most preferred designs in some aspect based solely on the design's evaluation-metric results. Finally, the analysis component outputs these designs and the associated evaluation-metric results as the final Pareto-optimal design set, which the designer can use to identify optimal designs and available tradeoffs.

To better understand how the Pareto-optimal search selects Pareto-optimal designs, let the set of all N designs be represented as $\{\boldsymbol{D}_1, \boldsymbol{D}_2, \ldots, \boldsymbol{D}_N\}$, where any particular design $\boldsymbol{D}_x$ in this set is represented by the set of that design's M evaluation-metric results $\{D_{x,1}, D_{x,2}, \ldots, D_{x,M}\}$. Then $\boldsymbol{D}_x$ is a Pareto-optimal design if and only if there is no design that is preferred or equal to $\boldsymbol{D}_x$ for every evaluation metric and is preferred to $\boldsymbol{D}_x$ in at least one evaluation metric (this is known as strongly Pareto optimal [Branke et al. 2008]). More formally, $\boldsymbol{D}_x$ is a Pareto-optimal design if and only if

$$\nexists i \in \{1 \ldots N\} : \left\{\forall j \in \{1 \ldots M\} : D_{i,j} \succeq D_{x,j}\right\} \wedge \left\{\exists j \in \{1 \ldots M\} : D_{i,j} \succ D_{x,j}\right\}. \quad (1)$$

For certain evaluation metrics (e.g., power), a minimal value is preferred, so rather than using a standard inequality sign, the Pareto-optimal definition uses the $\succ$ symbol meaning "is preferred to."

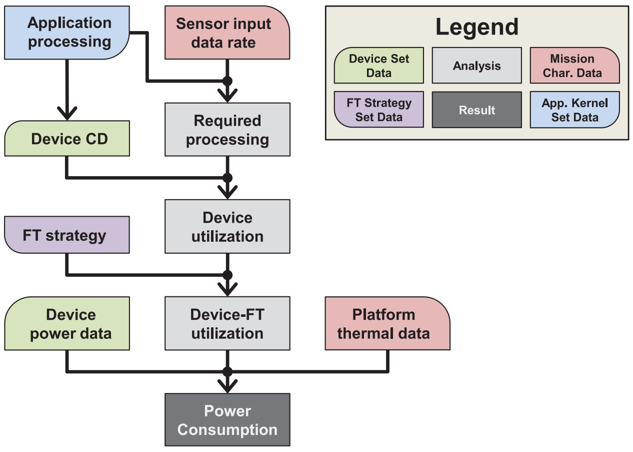


Fig. 3. Flowchart for the power-metric calculation.

4. EVALUATION METRICS

Our framework focuses on power, dependability, and lifetime, which are critical evaluation metrics for aerospace missions. The remainder of this section is organized as follows. Section 4.1, Section 4.2, and Section 4.3 present the design's device power consumption, dependability, and lifetime calculations, respectively.

4.1. Power

Figure 3 depicts the power-metric calculation. Our framework calculates the system's required processing in terms of type and rate of operations performed based on the designer-specified application processing and sensor input data rate. For example, consider a simple on-board image-processing system that uses a camera to capture Earth images from space with a sensor data rate of three images per second, four megapixels per image, and three 8-bit color channels (i.e., red, green, blue) per pixel. The system sums each pixel's three color values to determine if the average brightness of the image exceeds a certain threshold. Since adding multiple 8-bit values produces a result larger than 8 bits, the system processing can be summarized as three 16-bit addition operations per pixel, which is a required processing of 36 million 16-bit addition operations per second.

Our framework uses the required processing result and device CD to calculate the device utilization (U_{device}), which is the amount of device resources a system uses relative to the total amount of device resources available. A device utilization of 100% means that the system is using the device at the device's maximum potential. Our framework calculates the device utilization as the ratio of the required processing to the device's CD. This CD value must correspond to the type and precision of operations used in the required processing. For the image-processing example and a simple, representative, sample device with a 16-bit integer addition CD of 100 million operations per second, the device utilization is 36%.

Updating the device utilization to include the FT strategy's area overhead V_{FT} produces the device-FT utilization U_{dFT}:

$$U_{\text{dFT}} = U_{\text{device}}(1 + V_{\text{FT}}). \tag{2}$$

For the image-processing example and a TMR FT strategy, TMR introduces a ~200% overhead, which results in a device-FT utilization of 108%. Since the exact details of the device configuration dictate TMR's overhead, which could be more or less [Jacobs et al.

2012a] than 200%, and considering that our framework does not evaluate postsynthesis results for specific designs, our framework assumes an average 200% TMR overhead. Since the device-FT utilization is greater than 100%, the system requires either more than one device or a different device with greater resources.

The final result of the power metric calculation is the device's total power consumption P_{total}, which our framework calculates as the sum of the dynamic power consumption P_{dynamic} and static power consumption P_{static}. Our framework calculates P_{dynamic} as the product of the device's maximum dynamic power consumption $P_{\text{max-dyn}}$ (determined as the device's dynamic power consumption during 100% device utilization) and the device-FT utilization value. Therefore, total power consumption is calculated as

$$P_{\text{total}} = P_{\text{static}} + P_{\text{dynamic}} = P_{\text{static}} + (P_{\text{max-dyn}} \times U_{\text{dFT}}). \tag{3}$$

To calculate the static power, which is temperature dependent, our framework requires data on the ambient temperature of the platform T_{ambient}, the thermal resistance from the device to the platform R_{thermal}, and the device's static-power function $f_{\text{sp}}(T_{\text{device}})$, which records how the device's static power varies with respect to the device's temperature. Our framework stores the device's static-power function and nominal thermal resistance a priori in our framework's device database, while the designer supplies the ambient temperature and any special adjustments to the thermal resistance in the mission characteristics data. After calculating dynamic power consumption, our framework calculates the static power by first determining the device temperature needed to ensure that the total power used by the device is equal to the power dissipated as heat. This state of equilibrium is represented as

$$P_{\text{total}} = \frac{T_{\text{device}} - T_{\text{ambient}}}{R_{\text{thermal}}}. \tag{4}$$

Combining Equations (3) and (4), substituting $f_{\text{sp}}(T_{\text{device}})$ for P_{static}, and setting the equation to equal zero produces

$$R_{\text{thermal}}\big(f_{\text{sp}}(T_{\text{device}}) + P_{\text{dynamic}}\big) + T_{\text{ambient}} - T_{\text{device}} = 0. \tag{5}$$

Our framework finds the T_{device} that solves Equation (5) using the Newton-Raphson method, finds static power consumption by evaluating the device's static-power function at T_{device}, and finally computes the total power consumption according to Equation (3).

If the device-FT utilization is greater than 100%, the number of required devices n is

$$n = \lceil U_{\text{dFT}} \rceil. \tag{6}$$

Assuming that the total computation is distributed evenly across the n devices, our framework calculates the utilization for a single device as U_{dFT}/n, calculates the total power consumption for a single device as described earlier, and multiplies the single-device total power consumption by n to produce the total power consumption of all n devices.

To conclude the image-processing example with a TMR FT strategy, we assume an ambient temperature of 25°C and a sample device with a maximum dynamic power consumption of 10W, a thermal resistance of 4°C/W, and a static power consumption that varies linearly from 1W at 0°C to 3W at 100°C (an oversimplification of a standard static-power function). Since the device-FT utilization is 108%, n is 2, and the dynamic power consumption of a single device is 5.4W. Using our example values, Equation (5) becomes $5(\{1+T_{\text{device}}/50\}+5.4)+25-T_{\text{device}} = 0$, which results in a device temperature of 63.3°C and a static power of 2.27W. Finally, for our two example devices running the image-processing example application with a TMR FT strategy, the total power consumption is 15.34W.

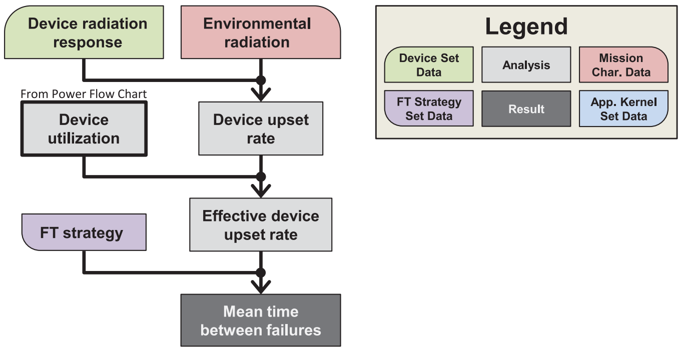


Fig. 4. Flowchart for the dependability-metric calculation.

We note that FPGA device utilization does not necessarily imply fabric utilization, which is the percentage of resources (e.g., lookup tables, flip-flops, DSP units, etc.) that the FPGA is actively using for processing. To illustrate this difference, we consider a device consisting of 125 computational units and two configuration designs: design A uses 100 computational units (80% fabric utilization) and runs at 100MHz, and design B uses 50 computational units (40% fabric utilization) and runs at 200MHz. The device's maximum CD is 20 billion operations per second (GOPS) at 160MHz, which can be achieved by running any arbitrary design that uses all 125 computational units. Using this same device, both design A and B perform 10 GOPS and therefore have a device utilization of 50%, even though both designs differ considerably in the designs' fabric utilizations. Furthermore, since design B's clock rate is double that of design A and design B's fabric utilization is half of design A's, design B's power consumption should realistically be similar to design A because the doubled clock should negate the reduced power consumption from halving the fabric utilization. Ultimately, device utilization allows our framework to abstract away the details of fabric utilization and clock frequency because device utilization is the only relevant factor in relating device performance to power consumption.

In this article, we have elected to discuss this simple power-metric calculation, which is an acceptably accurate estimate for many situations including our case study mission, which clearly and concisely demonstrates our framework's methodology. However, we have also developed a significantly more complex and advanced calculation [Wulf et al. 2013] for more accurately calculating the dynamic power consumption of a device for a given application. Instead of relying on linear interpolation using the device CD, this advanced calculation leverages linear programming techniques similar to those used in the CD methodology to find the optimal dynamic power consumption and consider a wide range of processing components (e.g., hard-core/soft-core processors, hard floating-point units) in addition to traditional FPGA logic. Although we could easily update our framework's simple power-metric calculation with this more accurate and advanced calculation, this would unnecessarily add to the complexity of demonstrating our framework's methodology.

4.2. Dependability

Figure 4 depicts the dependability-metric calculation. Environmental radiation data for heavy ions describes the particle flux (i.e., particles per square meter per second) for varying linear energy transfer (LET) values (or energy levels for protons). LET measures the amount of energy deposited by a particle as the particle passes through each unit length of a material (silicon in this case) in units of MeV·cm^2/g. Figure 5

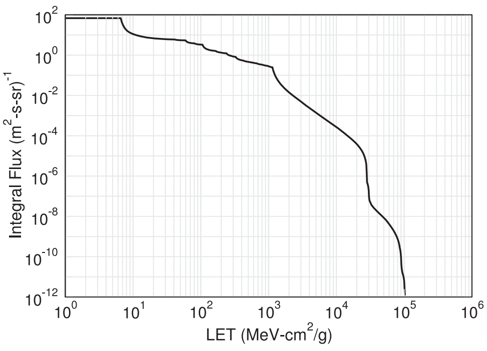


Fig. 5. Environmental radiation data for the ISS orbit assuming 100 mils of aluminum shielding (produced by CREME96).

depicts environmental radiation data for the International Space Station (ISS) orbit assuming 100 mils of aluminum shielding. In this example, a square meter of silicon will experience three 100MeV·cm^2/g particles every second, a 10,000MeV·cm^2/g particle every 24 minutes, and a 100,000MeV·cm^2/g particle every 16 millennia.

The device radiation-response data describes the cross-section of the device areas that are sensitive to single-event upsets (i.e., bit flip) when hit with a particle of a certain LET or energy level. Although temperature may also affect a device's cross-section, this effect is beyond the scope of this journal since there is insufficient data on this effect for our studied devices. Literature research data typically presents cross-section values in units of cm^2/device or cm^2/bit. If radiation data is in terms of per bit rather than per device, our framework calculates the total sensitivity of the device as the product of the bit sensitivity and the number of sensitive bits. Our framework estimates the number of sensitive bits to be equal to the size of the device's bitstream, meaning that our framework accounts for FPGA components (e.g., lookup tables, flip-flops, DSP units, block RAMs, etc.) that the bitstream configures or initializes in the sensitivity estimate. Although there are other FPGA-fabric resources that are nonconfigurable (e.g., internal DSP pipeline stage registers), we do not have access to these resources and cannot assess these resources' impacts on the FPGA's dependability without vendor-provided data.

Our framework determines the device upset rate based on the rate at which various particles hit the device and the effects of the hits, which are part of the environmental and device radiation-response data. The device upset rate measures the rate at which upsets occur in the whole device, including resources in unused device regions. If upsets occur in these regions, the upsets have no effect on the overall system because the design ignores any output from the unused resources. Therefore, the effective device upset rate is the product of the device upset rate and the device utilization (Section 4.1), which measures the relative amount of device resources used. We note that even at 100% device utilization, many device resources will remain unused (e.g., unused routing, unroutable/congested resources), resulting in only about 10% of the bits being vulnerable [Xilinx 2012]. Although it is possible to approximate the precise device upset rate and improve our framework's analysis through vendor tools or fault injection [Cieslewski et al. 2010; Nazar and Carro 2012], analysis based purely on device utilization scaling provides a reasonable worst-case estimate of the device upset rate and is assumed in our framework.

With the effective device upset rate and the applied FT strategy, our framework calculates the final MTBF result, which quantifies the average time a device can operate without experiencing a failure. Our framework calculates MTBF differently for

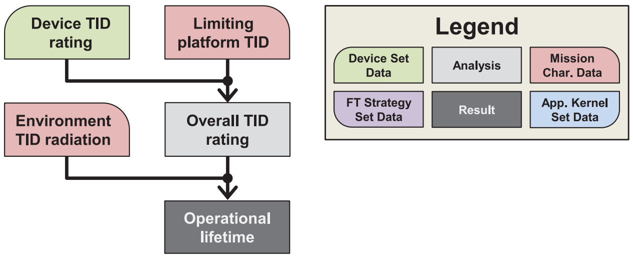


Fig. 6. Flowchart for the lifetime-metric calculation.

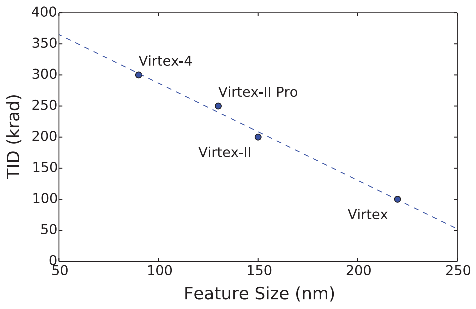


Fig. 7. Device TID ratings for various Virtex devices shows improving TID trend [Fabula et al. 2008].

different FT strategies, which may include variables such as non-FT effective device upset rate and input data size. For example, a TMR system's reliability is the probability that there is no system upset for some period of time and is calculated as

$$R_{\mathrm{TMR}} = 3(R_{\mathrm{Orig}})^2 - 2(R_{\mathrm{Orig}})^3. \quad (7)$$

If a non-TMR system has a reliability of 99.0% after 1 day, then TMR raises the reliability to 99.97%, protecting against 97% of the upsets as compared to the non-TMR system. Conversely, if the non-TMR system has a reliability of 80.0%, TMR raises the reliability to 89.6%, protecting against less than half of the upsets. For other FT strategies, it may not be possible to realistically calculate the FT strategy's fault-mitigating capabilities, requiring either fault-injection testing or literature research data. After calculating the final upset rate for the system, our framework calculates the final MTBF result by inverting the upset rate.

4.3. Lifetime

Figure 6 depicts the lifetime-metric calculation. Our framework requires literature research data for the device's TID rating and the limiting platform TID. The device's TID rating measures the amount of ionizing radiation energy that the device can absorb before becoming nonfunctional. The primary sources of TID radiation are protons, electrons, and bremsstrahlung (high-energy photons released by electron/proton interactions). Typical device TID ratings can range from a few krad for highly sensitive devices to over 1Mrad for hardened devices. Recent trends, as seen in Figure 7, demonstrate that device TID ratings may continue to improve as device feature sizes decrease, although this trend cannot be guaranteed as fabrication processes continue to change.

Essential platform components are any physical platform components (not including the processing device) that must be functional for a mission to remain operational. The limiting platform TID is the TID rating of the essential component with the lowest TID rating. If the platform contains a group of redundant, spare/backup essential

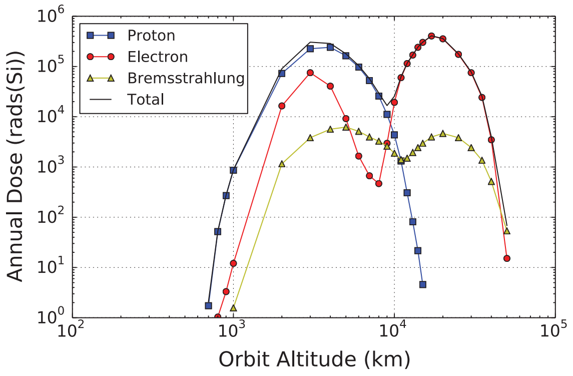


Fig. 8. Annual TID in circular equatorial orbits computed using SHIELDOSE, AE8MAX, and AP8MAX models with 4mm spherical Al shielding [Daly et al. 1996].

components for increased reliability, our framework considers only the group's essential component with the highest TID rating. Typical components with high sensitivity to TID include bipolar transistors, power MOSFETs (increased oxide thickness results in creation of more electron-hole pairs [Poizat 2009]), and Flash memories (charge pumps are sensitive to TID [Kayali]). Displacement damage (DD) may also affect essential platform components, which occurs when nonionizing radiation collides with and displaces atoms, leading to component defects. Our framework treats DD similarly to TID because both accumulate slowly over the component's lifetime, eventually rendering the essential component inoperable. Designers do not usually consider DD for processing devices because TID overshadows the effects of DD [Kayali]. However, other components do show particular sensitivity to DD, including bipolar transistors, photo-detecting charge-coupled devices (CCDs), solar cells, light-emitting diodes, and optocouplers [Poivey and Hopkinson 2009].

Our framework requires either literature research data or modeling results to obtain the environment TID level, which measures the amount of TID a device or component may experience per unit time due to the combination of environmental radiation and platform shielding. Figure 8 shows annual TID levels for equatorial near-Earth orbits of altitudes under 100,000km assuming moderate shielding. Near-Earth TID levels vary from less than 1krad/year for orbits closer than 1,000km up to 400krad/year for orbits with an altitude around 17,000km. Note that TID levels for the ISS at 400km and the Global Positioning System (GPS) orbit at 20,200km are significantly different (0.28 and 110krads/year, respectively) than Figure 8 suggests due to the ISS's and GPS's nonzero inclinations at 51.6° and 55°, respectively.

The overall TID rating is the minimum of the device TID rating and the limiting platform TID. Our framework calculates the final operational lifetime result by dividing the overall TID rating by the environment TID level. The operational lifetime measures the length of time the mission is expected to operate (under normal environmental conditions) before permanent failure of one or more essential components due to excessive radiation exposure. Operational lifetime can vary widely depending on the device/component and environment. For example, radiation-hardened devices rated for over 1Mrad can be expected to operate for several years or more with standard shielding in equatorial orbits around 20,000km altitudes. Furthermore, any Virtex device can be expected to operate for at least several decades before suffering any negative TID-related effects in a typical low Earth orbit (LEO). Although non-TID effects will likely disable the system before the expected TID-based operational lifetime expires, it may still be important to consider the operational lifetime due to solar flares and other sources of ionizing radiation, which can dramatically increase space radiation levels [Brown and Gabbe 1963]. Therefore, the lifetime metric is still useful for LEO-based missions, because during these events, an excessive operational lifetime could lead to months of useful operations as opposed to just days.

Currently, our framework's lifetime metric does not consider destructive single-event effects (SEEs), such as single-event latchup (SEL), single-event burnout (SEB), or single-event gate rupture (SEGR). Just a single occurrence of one of these destructive events can permanently disable an entire device or system, thereby dramatically affecting the expected lifetime of the mission. If radiation data on these effects does exist, the data typically only reports an onset LET, which is the minimum LET required to cause destructive SEEs. Although it is possible to use this data to give a rough estimate of the expected time until a destructive SEE occurs, designers typically opt to use devices that are immune to destructive SEEs rather than risk permanent device damage because, unlike a TID-based lifetime prediction where catastrophic effects do not occur until closer to the end of the predicted lifetime, destructive SEEs can disable a device at any time regardless of the destructive SEE's frequency. Therefore, rather than integrating destructive SEE effects into the lifetime metric, our framework only considers designs that are immune to destructive SEE effects based on the device properties and mission environment.

5. EXAMPLE CASE STUDY

This section introduces a currently deployed HSI mission, which serves as a case study for testing and examining our framework's analysis component. Section 5.1 introduces HSI data collection and materials analysis as well as our case study mission. Section 5.2 details our framework's power-metric, dependability-metric, and lifetime-metric calculation for a Virtex-4 with ABFT for the case study mission. Section 5.3 describes how we set up our case study, and Section 5.4 presents and analyzes the results of our case study.

5.1. Mission Details

Our case study's application involves an HSI analysis algorithm, which attempts to identify certain materials within a scene by comparing known material spectral signatures with observed characteristic spectra. An HSI sensor captures scene data in the form of a 3-dimensional image cube, where two spatial dimensions designate an image pixel, and the spectral dimension designates a specific spectral band for the pixels. As shown in Figure 9, a pixel's characteristic spectrum is the group of data from each spectral band that corresponds to the given pixel. A priori measurements produce spectral signatures for any materials of interest, which define the material's reflectance values for the spectral bands used by the HSI sensor. By comparing each pixel's characteristic spectrum to the set of material spectral signatures, HSI analysis can identify any material of interest and the material's locations in the scene. This process is analogous to humans subconsciously analyzing a scene using an object's color to determine the object's material composition (e.g., brown on an apple indicates rotting). The HSI sensor's greater spectral detail enables HSI analysis to more precisely identify materials (e.g., distinguishing between different types of green vegetation).

Remote HSI systems typically transmit collected image cubes to a ground station where high-performance processing systems perform HSI analysis. However, advances in space-borne electronics and improvements in fault-mitigating technology enable onboard HSI analysis, which may provide several advantages, such as enabling new HSI systems [NASA Jet Propulsion Laboratory 2014] to provide real-time critical information on natural disasters (e.g., volcanoes, wildfires, and drought). HSI analysis also

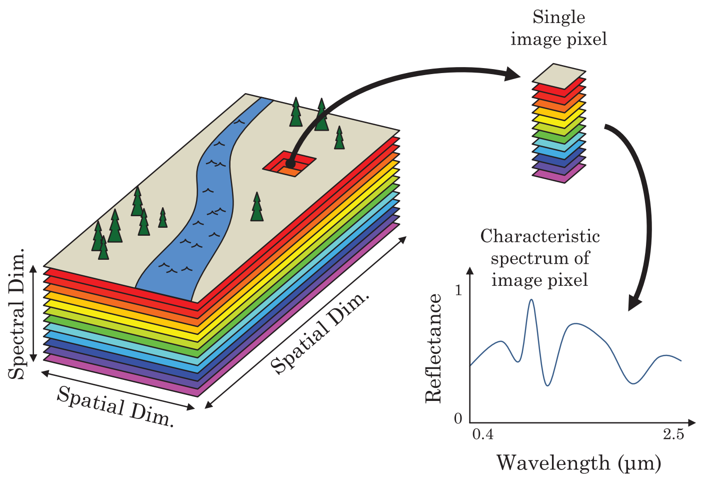


Fig. 9. HSI image cube and the characteristic spectrum of a single image pixel.

reduces image cubes to approximately 1% of the cube's original size, affording more efficient data storage and transmission.

Assessing the feasibility of an HSI on-board processing system requires estimation of the processing required for the HSI analysis on the streaming sensor data. However, because around 97% of the required processing involves a single matrix-multiply operation [Jacobs et al. 2008], these estimations are simplified. The matrix-multiply operation requires calculating the autocorrelation sample matrix $\boldsymbol{R}_{L\times L} = (\boldsymbol{A}_{N\times L})^{\mathrm{T}}(\boldsymbol{A}_{N\times L})$, where N is the number of pixels and L is the number of spectral bands. Matrix $\boldsymbol{A}_{N\times L}$ represents the sensor's image cube because spectral data for each pixel corresponds to a certain row of $\boldsymbol{A}_{N\times L}$. Only half of the values of the output matrix need to be calculated because the output matrix must be symmetric. The number of multiply-accumulate (MAC) operations required to calculate $\boldsymbol{R}_{L\times L}$ for a single image cube is

$$\mathrm{MAC_{HSI}} = \frac{1}{2}NL^2. \tag{8}$$

Data preprocessing prior to HSI analysis increases the system's processing requirements. First, the system preprocesses the HSI sensor's raw data to correct common image sensor defects. Specifically, each value in the image cube must be offset to account for readout noise and dark current and then scaled to adjust for flat-field effects. Since the operations per value are roughly equivalent to a single MAC operation, and there are $N \times L$ values for each image cube, raw data preprocessing requires L times less computation than HSI analysis. Since $L > 100$ for most HSI systems, the raw data preprocessing resource demands are negligible.

Our case study's HSI sensor is the Hyperion [Pearlman et al. 2003, 2001] on the Earth Observing-1 (EO-1) satellite, which orbits Earth at approximately 7.5km/s in LEO at a 680km altitude, capturing single lines of pixels at a time. These lines are perpendicular to the sensor's path, and the combination of many adjacent lines forms an image cube. The Hyperion captures an image every 2.95 seconds and produces an image cube 256 pixels wide, 660 lines long, and 220 twelve-bit spectral bands deep, requiring a total of 1.386 billion 32-bit integer MAC operations per second (OPS). Although temperatures in LEO can vary widely depending on whether a satellite is inside or outside Earth's shadow, we assume the EO-1 is equipped with a passive

Table I. Virtex-4 CREME96 Weibull Parameters
[Engel et al. 2006; Hiemstra et al. 2006]

	Heavy Ion	**Trapped Proton**
Onset	$0.87\ \frac{\mathrm{MeV \cdot cm^2}}{\mathrm{mg}}$	20MeV
Width	30	4
Power	1	0.5
Limiting XS	$1.73\ \frac{\mu\mathrm{m}^2}{\mathrm{bit}}$	$0.0156\ \frac{10^{-12}\mathrm{cm}^2}{\mathrm{bit}}$

thermal control system (e.g., insulation, radiators, thermal fillers) that regulates the ambient temperature to a standard 25°C.

5.2. Calculation of Framework Evaluation Metrics

In order to clearly define our framework's methodologies and contributions, this subsection details the calculation of the power, dependability, and lifetime evaluation metrics for our case study mission using a Virtex-4 LX40-FF668-10 device with the ABFT strategy.

The Virtex-4 LX40-FF668-10 device is a low-mid-range device in the 90nm Virtex-4 family and features 12.3 million configurable bits. The device has a CD of 9.37 billion 32-bit integer MAC OPS with a maximum dynamic power consumption of 4.59W. Based on the Xilinx power estimator tool, the device has a thermal resistance of 6.6°C/W in the airless vacuum of space and has a nonlinear static-power function measuring 0.256W at 25°C, 0.323W at 50°C, and 0.422W at 75°C.

Given the EO-1 Hyperion mission's required 1.386 billion 32-bit integer MAC OPS (Section 5.1), the device utilization for the Virtex-4 LX40-FF668-10 is 14.8%. Pessimistically assuming a 10% overhead [Silva et al. 1998] for the ABFT strategy results in a device-FT utilization of 16.3% and a dynamic power consumption of 0.748W. With an ambient temperature of 25°C, the device's temperature reaches 31.9°C, resulting in a static power consumption of 0.271W and a total power consumption of 1.02W.

The EO-1 Hyperion mission's primary radiation concerns are heavy ions and trapped protons. Most trapped protons originate from the sun's solar winds and are trapped by Earth's magnetosphere, whereas heavy ions are highly charged particles originating from outside of the solar system. Increased solar activity reduces both radiation hazards by causing atmospheric expansion to remove low-orbiting trapped protons and stronger solar winds to repel heavy ions entering the solar system.

CREME96 calculates the effects of these particles on processing devices by reporting the expected upset rate for a device in a given orbit. Engel et al. [2006] give a much more detailed description of a similar example with CREME96. We use the NORAD two-line element (TLE) [Kelso 2011] for EO-1 to supply the orbit parameters, and the solar-minimum model to ensure the dependability metric is accurate for the worst case. From these parameters, CREME96 creates a model of the external space ionizing-radiation environment, which models the proton and heavy-ion flux of various energies around the EO-1. Assuming a typical shielding of 100 mils of aluminum, CREME96 creates a transferred radiation model for the radiation environment inside the EO-1. From the internal radiation model, CREME96 can estimate the device upset rate using the device radiation-response data. Table I shows the heavy-ion and trapped-proton Weibull parameters for CREME96 that define the device's radiation response. The heavy-ion-induced upset rate is 0.538 upsets per day, and the trapped-proton-induced upset rate is 1.12 upsets per day, for a total device upset rate of 1.66 upsets per day. A device utilization of 14.8% results in an effective device upset rate of 0.246 upsets per day.

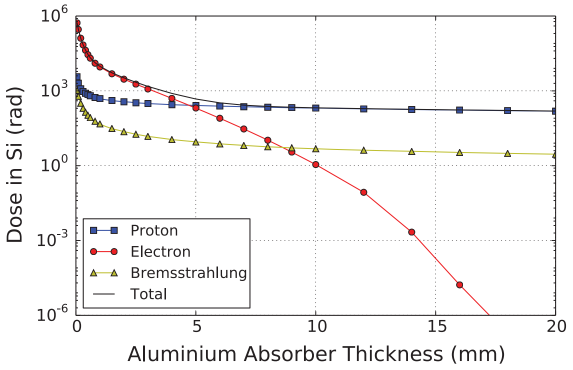


Fig. 10. SPENVIS model of TID contributions from electrons, protons, and bremsstrahlung with varying levels of aluminum shielding for the EO-1 orbit during solar maximum using the SHIELDOSE-2 model.

Due to the ABFT strategy's 10% overhead, the upset rate increases to 0.271 upsets per day. We also assume a pessimistic 90% coverage [Silva et al. 1998] for the ABFT detection. If the system detects an upset, the system restarts processing of the current image cube, resulting in no overall adverse effects for the system assuming that there are no impending, hard, real-time deadlines. With 90% coverage, the effective device upset rate drops to 0.0271 upsets per day, which is equivalent to an MTBF of 36.9 days.

To determine the environmental TID levels, we use the Space Environment Information System (SPENVIS), which was developed by the European Space Agency to model space environment effects. Although CREME96 also models TID effects, this model is insufficient for our case study because CREME96 does not model electron effects, which can account for a significant proportion of the environmental TID. Figure 10 depicts the calculated environmental TID levels for various levels of aluminum shielding in the EO-1 orbit during a solar maximum. For a thin hypothetical aluminum shielding of 2.54mm (equal to 100 mils), SPENVIS calculates worst and best cases of 2.203krad/year and 1.436krad/year during a solar maximum and minimum, respectively. As shown in Figure 7, the Virtex-4 has a TID rating of 300krad. Assuming that no other essential platform components have a lower TID rating, the EO-1 case study mission could operate for 136 years without experiencing failure due to TID.

5.3. Experimental Setup

For the EO-1 Hyperion mission, our framework computes the power and dependability evaluation metrics for six commercial FPGA device families and three FT strategies. We evaluate three standard Virtex families (Virtex-4, Virtex-5, and Virtex-6), two low-power Spartan families (Spartan-3 and Spartan-6), and the radiation-hardened Virtex-5QV device. Our automated data collection tool leverages the Tcl scripting abilities of the Xilinx ISE Design Suite to automatically implement and analyze many various operations on hundreds of different FPGA devices, enabling the calculation of CD after the designer has specified an application and mission. Our tool is currently set up to interface with Xilinx tools, which is sufficient for demonstrating our framework's abilities; however, the tool could be modified to interface with tools from other vendors that support Tcl scripting (e.g., automating Quartus to analyze Altera devices). Although the additional studied devices greatly increase the design space, our identification of Pareto-optimal designs scales well with the design space size, producing a greater design space reduction as a design space increases.

Our framework's current database exists as an SQLite database, which a Ruby-on-Rails-based web server accesses through the ActiveRecord library. When a designer

Table II. Counts of Devices Under Study

Family	Subfamilies	Device Models	Packages	Speed Grades
Virtex-4	3	17	29	58
Virtex-5	5	26	41	105
Virtex-6	4	17	37	95
Spartan-3	1	8	29	58
Spartan-6	2	9	30	59
Virtex-5QV	1	1	1	1
Total	15	78	167	376

accesses our framework's web tool to apply our framework's analysis to a set of devices, the web server collects all relevant information about these devices from our framework's SQLite database and sends this data client-side along with JavaScript code to the designer's computer. The designer's browser executes the JavaScript code, which processes the database data to calculate the CD of all specified devices, calculate the evaluation-metric results for all possible designs, and determine the final Pareto-optimal design set. Currently, designers must operate the CREME96 and SPENVIS web tools themselves, entering in parameters for the mission's orbit and any devices of interest based on data from literature research. Future development of our framework's web tool will enable designers to store these input parameters in the framework's database for data consolidation and may eventually enable automatic result retrieval from the CREME96 and SPENVIS web tools.

Currently, designers must determine the number and type of operations required by the application manually (demonstrated in Section 5.1), and supply this information using our framework's web tool. Ideally, designers would be able to define this application information by only specifying the general parameters of their application (e.g., image cube size, data capture rate). However, this convenience to designers can only be enabled through a full a priori characterization of the most common aerospace kernels, which is a significant area of research and is beyond the scope of this article.

The FT strategies include no fault tolerance (NFT), ABFT, and TMR, each of which performs blind scrubbing (with negligible overhead) after each image cube to remove any remaining errors in the configuration memory, ensuring that upsets during one image cube iteration do not affect the next image cube. Although checkpoint recovery and error correction codes are also common FT strategies, these are best suited for hard-core or soft-core processors and the associated caches, respectively, so these processors are not appropriate for demonstrating our framework with our current case study devices. We assume TMR to have a 200% overhead and include a negligibly small [Xilinx 2006], radiation-hardened, off-chip voter. We also assume that common-mode failures caused by single-event functional interrupts are not a problem for TMR because these failures are predicted to be extremely rare, with expected rates of only one event every 36 to 500 years for a commercial Virtex-4 device in various LEO orbits [Quinn 2008]. The design constraints for the EO-1 Hyperion mission are a power consumption less than 3W and an MTBF greater than 10 days.

Table II depicts our case study's device set. Each FPGA family contains several subfamilies, which are groups of devices optimized for basic logic, signal processing, connectivity, embedded processing, or some combination of these. There are several device models with varying fabric sizes grouped within these subfamilies, with the largest models being roughly an order of magnitude more powerful than the smallest models. For each model, there are often several packages that offer differing package sizes and I/O capabilities without altering the model's internal functionality. Finally, each package typically offers either two or three speed grades, with the faster speed grades sometimes reaching speeds as high as 40% faster than the lowest speed grades.

Table III. Device TID Ratings and Estimated Lifetimes

Family	Feature Size (nm)	TID Rating (krad)	Lifetime (years)
Virtex-4	90	300	136
Virtex-5	65	341	155
Virtex-6	40	381	173
Spartan-3	90	300	136
Spartan-6	45	373	169
Virtex-5QV	65	1,000	454

For clarity, in the remainder of this section, a *device* refers to a unique model, package, and speed grade combination. Therefore, from the original set of six Xilinx FPGA families, there are a total of 376 devices included in our device set, resulting in a design space of 1,128 designs (there are three designs for each device because we consider three FT strategies).

Literature research provides heavy-ion radiation-response data for the Virtex-4 [Engel et al. 2006], Virtex-5 [Quinn et al. 2007], Spartan-3 [Manuzzato et al. 2008], and Virtex-5QV [Xilinx 2012]. Hiemstra et al. also provided proton radiation-response data for all five of our commercial devices [Hiemstra et al. 2004, 2006, 2010; Hiemstra and Kirischian 2012, 2013]. The sources for protons and heavy ions give the radiation-response data for only a single device within each device family because all of the devices within a family share the same bit-level structure. Thus, we reuse radiation-response data for all devices within a family after adjusting for the number of configuration bits per device. Additionally, because we are unable to find sufficient heavy-ion radiation data publicly available for the Virtex-6 and Spartan-6 families, we use a method discussed by Petersen [1998] to accurately estimate heavy-ion limiting cross-sections based on the proton limiting cross-sections of the same device. The limiting cross-section is the most important of the four Weibull parameters. When only the limiting cross-section data is known, we can still obtain sufficiently accurate upset rate estimates by copying the missing three Weibull parameters from a known similar device family.

Since we do not have access to Virtex-5QV tools for generating designs, we estimate the Virtex-5QV's CD by analyzing the Virtex-5 FX130T, which is logically identical to the Virtex-5QV. Xilinx documents specify a block memory maximum frequency of 360MHz for the Virtex-5QV and 550MHz for the Virtex-5. Since the Virtex-5 FX130T's CD is bandwidth limited (the device can fit more MAC operators than the on-chip block memory can supply with inputs), we assume that the Virtex-5QV's CD is 65.45% of the FX130T's CD.

To ensure mission success, it is important that the considered devices are immune to destructive SEEs. SEB and SEGR are primarily the concern of power MOSFETs and BJTs [Sturesson 2003; Ladbury 2007; Schwank et al. 2008], rarely affecting commercial CMOS devices, such as Xilinx FPGAs. SEL is the most common destructive SEE for CMOS devices, so SEL immunity is an important concern for our EO-1 Hyperion mission. Fortunately, the literature shows that all of these commercial families are essentially SEL immune for the levels of radiation in the relatively calm LEO environment of the EO-1 Hyperion mission [Hiemstra et al. 2004, 2006, 2010; Hiemstra and Kirischian 2012, 2013].

The lifetime evaluation metric requires knowledge of the TID ratings of the devices and the expected environmental TID levels (Section 4.3). Table III depicts the TID ratings for the six Xilinx FPGA families in our case study, which were obtained directly from the data shown in Figure 7 or estimated based on the data's linear trendline, which shows a highly linear correlation between feature size and TID rating. This estimation is only appropriate for highlighting the potential suitability of a device for a mission

Table IV. Pareto-Optimal Designs

Fam.	Mod.	Pack.	SG	FT	Power (W)	MTBF (days)	Life (years)
Vir-4	SX55	FF1148	12	ABFT	0.858	109	136
Vir-4	SX55	FF1148	12	TMR	1.35	1,400,000	136
Vir-5	SX35T	FF665	3	ABFT	0.595	49.2	155
Vir-5	SX35T	FF665	3	TMR	0.912	285,000	155
Sp-6	LX16	CSG324	3	ABFT	0.417	38.7	169
Vir-6	LX75T	FF784	2	ABFT	0.696	74.2	173
Vir-6	LX75T	FF784	3	TMR	0.940	650,000	173
Vir-6	LX130T	FF1156	2	TMR	1.35	666,000	173
Vir-6	LX130T	FF784	3	TMR	1.38	746,000	173
Vir-6	LX130T	FF484	3	TMR	1.78	749,000	173
Vir-6	LX195T	FF784	3	TMR	2.19	793,000	173
Vir-6	LX240T	FF1759	3	TMR	2.43	858,000	173
Vir-6	LX240T	FF1156	3	TMR	2.47	867,000	173
5QV	FX130	CF1752	1	NFT	2.41	3,930	454
5QV	FX130	CF1752	1	ABFT	2.44	35,700	454

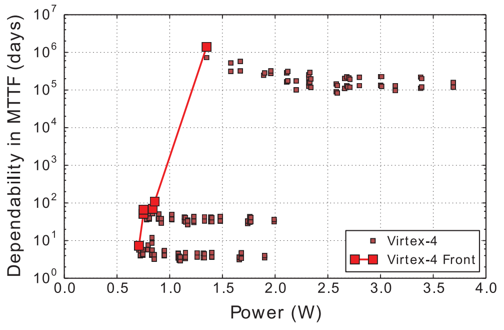
during the early design phase, which is when our framework is most useful. Since these estimations may be inaccurate, any device recommended by our framework must still be tested for TID through radiation injection before the device can be accepted for the final design. Unlike the other families, the Virtex-5QV device is radiation hardened by design and therefore does not follow the same trendline as the other devices. Instead, the Virtex-5QV's product specification states that the Virtex-5QV has a minimum TID rating of 1Mrad. Table III shows the predicted lifetimes (using SPENVIS and the EO-1 mission orbit) for all device families based on the worst-case (solar maximum) value, which are the lifetimes used for our case study.

5.4. Results and Analysis

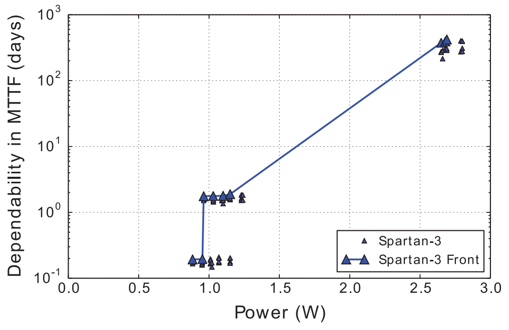
Figures 11, 12, and 13 depict the power and dependability evaluation metric results for our case study. In order of ascending family TID rating, Figures 11(a) to 11(f) depict the results for all designs for each family (Virtex-4, Spartan-3, Virtex-5, Spartan-6, Virtex-6, and Virtex-5QV, respectively) and highlight each family's Pareto-optimal front. Figure 12 collectively depicts the families' designs for cross-family comparison and shows the family-specific Pareto-optimal fronts. Figure 13 depicts our framework's final Pareto-optimal design set after considering lifetime and filtering unsuccessful designs that fail the EO-1 Hyperion mission's power consumption and MTBF constraints (Section 5.3). The results do not show designs requiring more than one device to meet the computational requirements.

Table IV lists the designs included in our framework's final Pareto-optimal design set. The specific device used in a design is described by the device's family, model, package, and speed grade listed under the Fam., Mod., Pack., and SG columns, respectively. The FT column shows a design's FT strategy. The final three columns (Power, MTBF, and Life) show the results of our framework's evaluation metrics for power, dependability, and lifetime, respectively, for each design. We group the designs according to the design's device's family and order by ascending lifetime. Within each device family, we order the designs in ascending order by power and dependability.

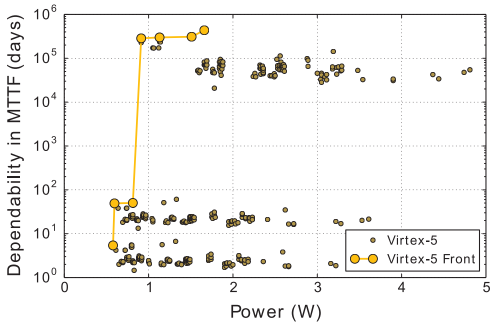
In some cases, clusters of two or more Pareto-optimal designs are nearly identical in power but not in dependability. Within these clusters, designs with less dependability should not be considered because these designs have an insignificant power gain. To address this issue, our framework rounds each evaluation metric result to three significant digits. Therefore, although several designs within these clusters may be



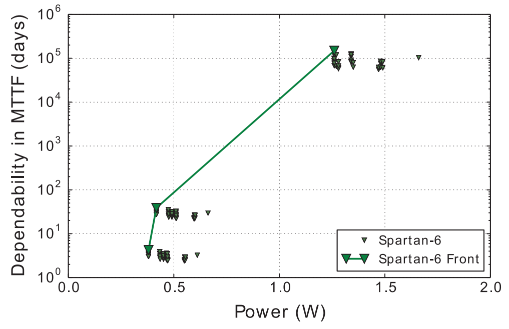
(a) Virtex-4-based designs.



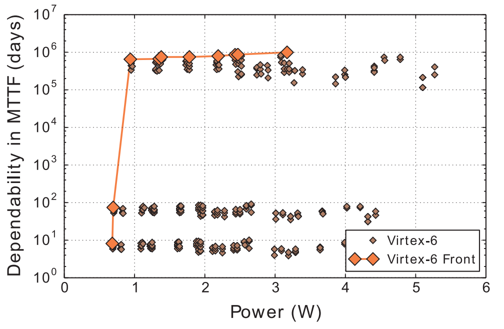
(b) Spartan-3-based designs.



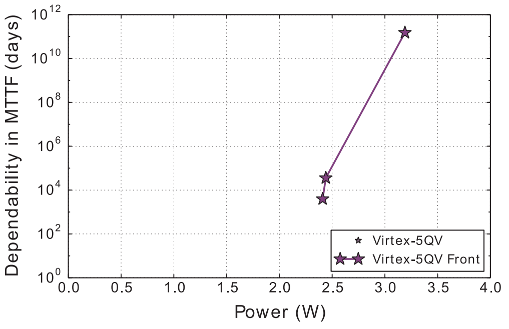
(c) Virtex-5-based designs.



(d) Spartan-6-based designs.



(e) Virtex-6-based designs.



(f) Virtex-5QV-based designs.

Fig. 11. Power and dependability results for all designs including family-specific Pareto fronts.

technically Pareto optimal before rounding, our framework considers only the most dependable design as Pareto optimal because these designs are essentially equal in power but superior in dependability.

In general, there is significant variation between the families and the designs in each family. Within each family, there are three typically horizontally stretched groups that represent the three FT strategies used in our case study. From bottom to top, these FT strategies are the low-power NFT, the middle-ground ABFT, and the highly dependable TMR strategies. The horizontal stretching of these groups is typically the result of variations in static power between the differently sized devices, which affects the power consumption but not the dependability. Within each of the horizontal groups,

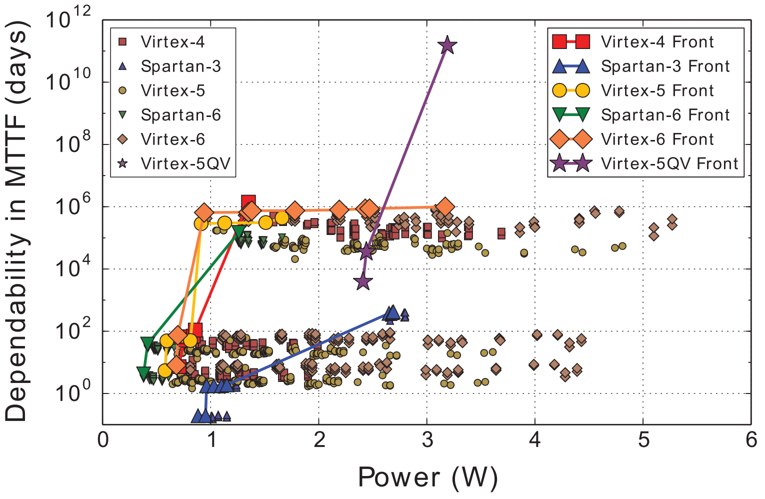


Fig. 12. All designs with six family-specific Pareto fronts.

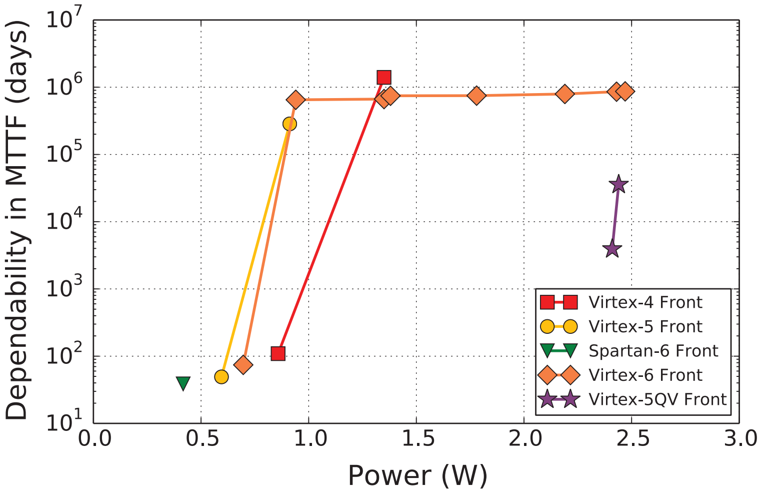


Fig. 13. Power and dependability results for all designs in the final Pareto-optimal design set including family-specific Pareto fronts.

there are typically smaller vertical groupings consisting of designs that differ only in the design's device package and speed grade, with increasing dependability correlating with increasing speed grades. For example, doubling the device's operating speed does not necessarily increase the device's dynamic power because the faster device would require half as much device utilization to achieve the same computational capacity. Conversely, reducing the device utilization would improve dependability (Section 4.2). Finally, varying the design's device package shows a slight yet consistent correlation, with larger packages having a lower thermal resistance and therefore slightly lower static power consumption.

Table IV shows that all designs on the Virtex-4 and Virtex-5 Pareto-optimal fronts use the SX device subfamily. A closer analysis reveals that the SX subfamily is better suited for the HSI application than other subfamilies. For example, we consider the differences between the Virtex-5 SX50T-FF1136-3 and the Virtex-5 LX50TFF1136-3. Both devices are equivalent in package and speed grade but differ in the number of device resources, with the SX device having 13% more logic and 438% more DSP units than the SX device's LX counterpart. The HSI application's multiplication operations are significantly more power efficient when implemented using an FPGA's DSP units rather than only using an FPGA's general logic resources. This results in the SX device achieving a 200% larger CD than the LX device for nearly the same power consumption. Therefore, even though the SX device has 0.18W more static power, the SX device can still perform the required computations with a lower overall power consumption because the SX device's dynamic power consumption scales efficiently.

Conversely, the SX subfamily is not dominating the Virtex-6's Pareto-optimal front for two reasons. First, the Virtex-6 SX subfamily is only available in two large models, the smallest being the SX315T with a static power of 2.08W at 25°C. The smallest Virtex-6 model (the LX75T) has a static power of 0.54W and performs the mission's required computation while staying well under 2W of total power consumption. Second, the Virtex-6 devices have a higher DSP-to-logic ratio, as demonstrated by the Virtex-6 LX75T model having an equal number of DSPs as the Virtex-5 SX50T. Since neither the Virtex-6 LX's nor the Virtex-6 SX's DSP resources saturate when running the HSI application, neither subfamily has a clear advantage over the other.

The Spartan families do not have large horizontally stretched groups for two reasons. First, the Spartan devices offer a smaller, more power-efficient alternative to their Virtex counterparts. Therefore, many of the smaller Spartan designs are too small to handle the mission's required computations, resulting in a reduced design set consisting of only larger devices. Second, there is no significant variation between the subfamilies within each family. The Spartan-3 family does not have different subfamilies and the Spartan-6 has the LX and LXT subfamilies, which differ only in I/O bandwidth and not in computational resources. Therefore, limited differences in device size and specialization potential result in the small variation between different Spartan devices shown in Figures 11(b) and 11(d). This effect also applies to the Virtex-5QV device, which has only one model, package, and speed grade, and thus results in design variation only between the three different FT strategies.

Figure 12 shows how different device families can affect a design's power and performance. The older Spartan-3 family performs poorly in both power and dependability, while the Virtex-4 family provides slightly better dependability and significantly better power consumption than the Spartan-3 family. Although the Virtex-4 devices have higher static power consumption than similarly sized Spartan-3 devices, the Virtex-4 family's DSP units are more power efficient than the Spartan-3 multiplier units, which is important to consider when designing for the HSI application. The Spartan-6, Virtex-5, and Virtex-6 families are superior in low-power consumption, and the Virtex-5 and Virtex-6 families also perform similarly in dependability to the Virtex-4 for designs using TMR. As shown in Table IV, the Virtex-5QV is the most dependable device with 1,000$\times$ greater MTBF for designs with similar FT strategies, but has the highest power consumption of all the devices (also the highest cost at \$50,000 compared to a standard FPGA price of around \$1,000).

Figure 13 shows the final Pareto-optimal design set. Our framework reduces the design space of 1,128 possible designs by 98.7%, determining these 15 designs, consisting of five device families and three FT strategies, as the set of successful Pareto-optimal designs that meet the mission's constraints. The Spartan-3's Pareto-optimal front as well as much of the Pareto-optimal fronts of the Virtex-4, Virtex-5, and Spartan-6 are Pareto inferior to the Virtex-6's Pareto-optimal front. Our framework also rejects all of the Spartan-6 and Virtex-6 NFT designs because none of these designs meet the dependability constraint of an MTBF greater than 10 days. Similarly, the Virtex-5QV's

TMR design and the most power-hungry Pareto-optimal Virtex-6 TMR design consume more than 3W; thus, our framework rejects these designs as well. Of the Pareto-optimal design set, TMRs on the Virtex-4, Virtex-5, and Virtex-6 show the best dependability; ABFTs on the Virtex-4, Virtex-5, Virtex-6, and Spartan-6 show the best power consumption; Virtex-5QV has superior lifetime; and TMR on the Virtex-6 provides a balanced design for all three metrics.

Our system to compute these results uses a single core of an Intel Core i7-2600 CPU running at 3.4GHz with 8GB of RAM to run our framework's JavaScript code running on version 42 of the Mozilla Firefox browser. We evaluate the performance of our framework using 10 trial runs on our system, and we measure the duration of the analysis of each run by querying the system time through the JavaScript Date object before and after the relevant JavaScript code. Our system takes an average of 1.01 seconds to calculate the CD for every device and the evaluation-metric results for each design in our results. After computing all the evaluation-metric results, our system takes an average of 4ms to find the final Pareto-optimal design set. Although general SoC-design optimization approaches may typically require more processing time, our framework leverages the CD methodology to prune the majority of the design-variant options available in the SoC design and focus on a single optimal design variant for each device. Therefore, although the task of finding the final Pareto-optimal design set is on the order of $O(n^2)$, we have reduced the total computation time by first computing CD and the evaluation-metric results for all designs, which instead scale linearly with the number of designs. We predict that even for device sets that are several orders of magnitude larger, calculating the evaluation metrics for all designs will dominate total computation time, meaning our framework's methodology will scale well into the future with the addition of new devices to our framework with increasing device complexity.

6. CONCLUSIONS

In this article, we have introduced a novel framework that leverages past research and successes in device, application, and fault-tolerant (FT) strategy analysis to aid in the design of on-board FPGA-based SoCs for aerospace computing. Our framework considers a designer-defined mission and application, and analyzes a database of literature research and experimental data to provide designers with a final set of Pareto-optimal system designs (device/FT strategy combinations). Our framework's evaluation metrics enable designers to select the best design from this final set depending on desired metric tradeoffs and mission requirements.

To demonstrate our framework's potential given a large design space, we analyzed a design space of 1,128 designs (63× larger than our previous work) and provided a more in-depth analysis of the designs using the new lifetime evaluation metric. Our framework reduced the design space by 98.7%, identifying 15 final Pareto-optimal designs, including designs specializing in low power, high dependability, high lifetime, or a compromise between all three of these attributes.

Our future work includes further framework expansions and enhancements. Realizable Utilization (RU) enables device comparison of the attainable performance a typical designer is able to realize from a device given a certain application as compared to the device's peak performance capability. RU enhances our framework by more effectively measuring the impact of an application on the device, thereby improving the evaluation of a device's CD. Additionally, we will include fault-injection analysis in the calculation of the dependability metric, which will provide greater insight into the true vulnerability of certain applications and the behavior of various FT strategies. Research into device size and cost may also lead to two additional metrics to enhance our framework's evaluation of designs and increase the range of the Pareto-optimal design set. For example, no Spartan-3 designs were Pareto optimal in our analysis because the device family underperformed in all three metrics, but some Spartan-3 designs may be Pareto optimal if we also include a device-cost metric in our analysis, because older Spartan-3 devices are typically less expensive than the newer generations of FPGAs. Finally, device utilization can indicate the potential for a system to advance and mature during the course of a mission and even beyond the mission's original purpose. Although device utilization is an intermediate metric that enables the calculations of the power and dependability evaluation metrics, including device utilization in the set of evaluation metrics would be straightforward and would be worth investigating.

ACKNOWLEDGMENTS

The authors gratefully acknowledge Pietro Sparacino (NASA GSFC) for his help in describing NASA's device selection process for space-based designs.

REFERENCES

K. Asanovic, R. Bodik, B. C. Catanzaro, J. J. Gebis, K. Keutzer, D. A. Patterson, W. L. Plishker, J. Shalf, S. W. Williams, and K. A. Yelick. 2006. *The Landscape of Parallel Computing Research: A View from Berkeley*. Technical Report UCB/EECS-2006-183. University of California, Berkeley. http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.pdf.

J. Branke, K. Deb, K. Miettinen, and R. Slowinski (Eds.). 2008. *Multiobjective Optimization*. Springer-Verlag, Berlin, Germany.

W. L. Brown and J. D. Gabbe. 1963. The electron distribution in the earth's radiation belts during July 1962 as measured by telstar. *J. Geophys. Res.* 68, 3 (1963), 607–618.

G. Cieslewski, A. D. George, and A. Jacobs. 2010. Acceleration of FPGA fault injection through multi-bit testing. In *Proc. of International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA'10)*. 218–224.

E. J. Daly, A. Hilgers, G. Drolshagen, and H. D. R. Evans. 1996. Space environment analysis: Experience and trends. In *ESA 1996 Symposium on Environment Modelling for Space-Based Applications*. Noordwijk, The Netherlands. http://atanar-esa.cdnetworks.net/conferences/96a09/Abstracts/abstract45/paper/

J. Engel, K. Morgan, M. J. Wirthlin, and P. S. Graham. 2006. *Predicting On-Orbit Static Single Event Upset Rates in Xilinx Virtex FPGAs*. Technical Report. All Faculty Publications. http://scholarsarchive.byu.edu/facpub/1307/.

R. Enzler, T. Jeger, D. Cottet, and G. Tröster. 2000. High-level area and performance estimation of hardware building blocks on FPGAs. In *Field-Programmable Logic and Applications: The Roadmap to Reconfigurable Computing*, R. W. Hartenstein and H. Grünbacher (Eds.). Lecture Notes in Computer Science, Vol. 1896. Springer, Berlin, 525–534. DOI: http://dx.doi.org/10.1007/3-540-44614-1_57

J. J. Fabula, J. L. DeJong, A. Lesea, and W. Hsieh. 2008. The total ionizing dose performance of deep submicron CMOS processes. In *Proc. of Military and Aerospace Programmable Logic Devices Conference (MAPLD'08)*. Annapolis, MD. https://nepp.nasa.gov/mapld_2008/presentations/w/08-Fabula_Joseph_mapld08 _pres_2.pdf.

D. M. Hiemstra, G. Battiston, and P. Gill. 2010. Single event upset characterization of the Virtex-5 field programmable gate array using proton irradiation. In *2010 IEEE Radiation Effects Data Workshop (REDW'10)*. 1–4.

D. M. Hiemstra, F. Chayab, and Z. Mohammed. 2006. Single event upset characterization of the Virtex-4 field programmable gate array using proton irradiation. In *2006 IEEE Radiation Effects Data Workshop (REDW'06)*. 105–108.

D. M. Hiemstra, F. Chayab, and L. Szajek. 2004. Dynamic single event upset characterization of the Virtex-II and spartan-3 SRAM field programmable gate arrays using proton irradiation. In *2004 IEEE Radiation Effects Data Workshop (REDW'04)*. 79–84.

D. M. Hiemstra and V. Kirischian. 2012. Single event upset characterization of the Virtex-6 field programmable gate array using proton irradiation. In *2012 IEEE Radiation Effects Data Workshop (REDW'12)*. 1–4.

D. M. Hiemstra and V. Kirischian. 2013. Single event upset characterization of the spartan-6 field programmable gate array using proton irradiation. In *2013 IEEE Radiation Effects Data Workshop (REDW'13)*. 1–4.

B. Holland, K. Nagarajan, and A. D. George. 2009. RAT: RC amenability test for rapid performance prediction. *ACM Trans. Reconfigurable Technol. Syst.* 1, 4 (Jan. 2009), 22:1–22:31.

K. Huang and J. A. Abraham. 1984. Algorithm-based fault tolerance for matrix operations. *IEEE Trans. Comput.* C-33, 6 (June 1984), 518–528.

A. Jacobs, G. Cieslewski, and A. D. George. 2012a. Overhead and reliability analysis of algorithm-based fault tolerance in FPGA systems. In *2012 22nd International Conference on Field Programmable Logic and Applications (FPL'12)*. 300–306.

A. Jacobs, G. Cieslewski, A. D. George, A. Gordon-Ross, and H. Lam. 2012b. Reconfigurable fault tolerance: A comprehensive framework for reliable and adaptive FPGA-based space computing. *ACM Trans. Reconfigurable Technol. Syst.* 5, 4, Article 21 (Dec. 2012), 30 pages. DOI:http://dx.doi.org/10.1145/2392616.2392619

A. Jacobs, C. Conger, and A. D. George. 2008. Multiparadigm space processing for hyperspectral imaging. In *2008 IEEE Aerospace Conference*. 1–11.

S. Kayali. *Space Radiation Effects on Microelectronics*. Technical Report. NASA JPL Radiation Effects Group. http://parts.jpl.nasa.gov/docs/Radcrs_Final.pdf.

T. S. Kelso. 2011. NORAD Two-Line Element Sets Current Data. (Nov. 2, 2011). http://celestrak.com/ NORAD/elements/resource.txt.

R. Ladbury. 2007. *Radiation Hardening at the System Level*. Technical Report. NASA Goddard Space Flight Center. http://radhome.gsfc.nasa.gov/radhome/papers/nsrec07_sc_ladbury.pdf.

R. E. Lyons and W. Vanderkulk. 1962. The use of triple-modular redundancy to improve computer reliability. *IBM J. Res. Dev.* 6, 2 (Apr. 1962), 200–209.

A. Manuzzato, S. Gerardin, A. Paccagnella, L. Sterpone, and M. Violante. 2008. On the static cross section of SRAM-based FPGAs. In *2008 IEEE Radiation Effects Data Workshop*. 94–97.

M. R. Meswani, L. Carrington, D. Unat, A. Snavely, S. Baden, and S. Poole. 2013. Modeling and predicting performance of high performance computing applications on hardware accelerators. *Int. J. High Perform. Comput. Appl.* 27, 2 (May 2013), 89–108.

NASA Jet Propulsion Laboratory. 2014. HyspIRI Mission Study. (2014). http://hyspiri.jpl.nasa.gov.

G. L. Nazar and L. Carro. 2012. Fast single-FPGA fault injection platform. In *2012 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT'12)*. 152–157.

J. V. Neumann. 1956. Probablistic logics and the synthesis of reliable organisms from reliable components. *Automata Stud.* 34 (1956), 43–98.

J. S. Pearlman, P. S. Barry, C. C. Segal, J. Shepanski, D. Beiso, and S. L. Carman. 2003. Hyperion, a space-based imaging spectrometer. *IEEE Trans. Geosci. Remote Sens.* 41, 6 (2003), 1160–1173.

J. Pearlman, S. Carman, C. Segal, P. Jarecke, P. Clancy, and W. Browne. 2001. Overview of the Hyperion imaging spectrometer for the NASA EO-1 mission. In *IEEE 2001 International Geoscience and Remote Sensing Symposium, 2001. (IGARSS'01).* Vol. 7. 3036–3038.

R. L. Pease, A. H. Johnston, and J. L. Azarewicz. 1988. Radiation testing of semiconductor devices for space electronics. *Proc. IEEE* 76, 11 (1988), 1510–1526.

E. L. Petersen. 1998. The SEU figure of merit and proton upset rate calculations. *IEEE Trans. Nuclear Sci.* 45, 6 (Dec. 1998), 2550–2562.

C. Poivey and G. Hopkinson. 2009. *Displacement Damage: Mechanisms and Effects*. Technical Report. Space Center EPFL and European Space Agency. http://space.epfl.ch/webdav/site/space/shared/industry_media/05DDissue3.pdf.

M. Poizat. 2009. *Total Ionizing Dose: Mechanisms and Effects*. Technical Report. Space Center EPFL and European Space Agency. http://space.epfl.ch/webdav/site/space/shared/industry_media/03EPFL_TID_Basic-Mech.pdf.

B. Pratt, M. Fuller, M. Rice, and M. Wirthlin. 2013. Reduced-precision redundancy for reliable FPGA communications systems in high-radiation environments. *IEEE Trans. Aerospace Electronic Syst.* 49, 1 (Jan. 2013), 369–380.

H. Quinn. 2008. *An Introduction to Mission Risk and Risk Mitigation for Xilinx SRAM FPGAs*. Technical Report. Los Alamos National Laboratories. ftp://ftp.lanl.gov/public/hquinn/quinn_intro_to_rad2.pdf.

H. Quinn, K. Morgan, P. Graham, J. Krone, and M. Caffrey. 2007. Static proton and heavy ion testing of the xilinx virtex-5 device. In *2007 IEEE Radiation Effects Data Workshop*. 177–184.

K. Sahu, H. Leidecker, and D. Lakins. 2003. *EEE-INST-002: Instructions for EEE Parts Selection, Screening, Qualification, and Derating*. http://nepp.nasa.gov/DocUploads/FFB52B88-36AE-4378-A05B2C084B5EE2CC/EEE-INST-002_add1.pdf.

J. R. Schwank, M. R. Shaneyfelt, and P. E. Dodd. 2008. *Radiation Hardness Assurance Testing of Microelectronic Devices and Integrated Circuits: Radiation Environments, Physical Mechanisms, and Foundations for Hardness Assurance*. Technical Report. Sandia National Laboratories. http://www.sandia.gov/mstc/services/documents/Sandia_RHA_Foundations_FINAL.pdf.

J. G. Silva, P. Prata, M. Rela, and H. Madeira. 1998. Practical issues in the use of ABFT and a new failure model. In *28th Annual International Symposium on Fault-Tolerant Computing, 1998. Digest of Papers.* 26–35.

F. Sturesson. 2003. *Single Event Effects (SEE) Mechanism and Effects*. Technical Report. Space Center EPFL and European Space Agency. http://space.epfl.ch/webdav/site/space/shared/industry_media/07SEEEffectF.Sturesson.pdf.

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. 1997. CREME96: A revision of the cosmic ray effects on micro-electronics code. *IEEE Trans. Nucl. Sci.* 44, 6 (Dec. 1997), 2150–2160.

J. Williams, A. D. George, J. Richardson, K. Gosrani, C. Massie, and H. Lam. 2010. Characterization of fixed and reconfigurable multi-core devices for application acceleration. *ACM Trans. Reconfigurable Technol. Syst.* 3, 4 (Nov. 2010), 19:1–19:29.

N. Wulf, A. D. George, and A. Gordon-Ross. 2012. A framework to analyze, compare, and optimize high-performance, on-board processing systems. In *2012 IEEE Aerospace Conference*. 1–14.

N. Wulf, A. D. George, and A. Gordon-Ross. 2015. Memory-aware optimization of FPGA-based space systems. In *2015 IEEE Aerospace Conference*. 1–13.

N. Wulf, J. Richardson, and A. D. George. 2013. Optimizing FPGA performance, power, and dependability with linear programming. In *Proc. of Military and Aerospace Programmable Logic Devices Conference (MAPLD'13)*. San Diego, CA.

Xilinx. 2006. *Triple Module Redundancy Design Techniques for Virtex FPGAs*. Xilinx. http://www.xilinx.com/support/documentation/application_notes/xapp197.pdf.

Xilinx. 2012. *Considerations Surrounding Single Event Effects in FPGAs, ASICs, and Processors*. Xilinx. http://www.xilinx.com/support/documentation/white_papers/wp402_SEE_Considerations.pdf.

Xilinx. 2012. *Radiation-Hardened, Space-Grade Virtex-5QV Family Overview*. Xilinx. http://www.xilinx.com/support/documentation/data_sheets/ds192_V5QV_Device_Overview.pdf.

Received March 2015; revised December 2015; accepted January 2016