IAC-22,D1,IP,x72691

# Methods for increasing the dependability of High-performance, Many-core, System-on-Chips*

**Rafal Graczyk, Saad Memon, Marcus Völp**

*Interdisciplinary Center for Security, Reliability, and Trust, University of Luxembourg, Campus Belval, 6, avenue de la Fonte, L-4364 Esch-sur-Alzette, Luxembourg;*
rafal.graczyk@uni.lu; saad.memon@uni.lu; marcus.voelp@uni.lu

**Abstract**

Future space exploration and exploitation missions will require significantly increased autonomy of operation for mission planning, decision-making, and adaptive control techniques. Spacecrafts will integrate new processing and compression algorithms that are often augmented with machine learning and artificial intelligence capabilities. This functionality will have to be provided with high levels of robustness, reliability, and dependability for conducting missions successfully. High-reliability requirements for space-grade processors have led to trade-offs in terms of costs, energy efficiency, and performance to obtain robustness. However, while high-performance / low-robustness configurations are acceptable in the Earth's vicinity, where assets remain protected by the planet's magnetosphere, they cease to work in more demanding environments, like cis-lunar or deep space, where high-energy particles will affect modern components heavily, causing temporary or permanent damage and ultimately system failures. The above has led to a situation where state-of-the-art processing elements (processors, co-processors, memories, special purpose accelerators, and field-programmable-gate arrays (FPGAs), all possibly integrated into System-on-a-Chip (SoC) designs) are superior to their high reliability, space-qualified counterparts in terms of processing power or energy efficiency. For example, from modern, state-of-the-art (SOTA) devices, one can expect a 2-3 order-of-magnitude performance per Watts improvement over space-grade equipment. Likewise, one finds a gap of approximately nine technology nodes between devices, which translates into a factor 25 decrease in operations per Watts. In this paper, we demonstrate how to utilize part of this enormous performance advantage to increase the robustness and resilience of otherwise susceptible semiconductor devices while harnessing the remaining processing power to build affordable space systems capable of hosting the compute-intensive functionality that future space missions require. We are bridging this performance-reliability gap by researching the enabling building blocks for constructing reliable and secure, space-ready Systems-on-a-Chip from SOTA processing elements.

**Keywords:** On-board computing, Dependability, System-on-a-Chip (SoC), Processor Design, Tiled Architecture, Space Architecture, Space Systems

**Nomenclature – Acronyms/Abbreviations**

| | |
|---|---|
| HERA | Hypervisor Enforced Radiation-tolerant Architecture |
| COTS | Common off-the-shelf (processors and system components) |
| PE | Processing Element |

## 1. Introduction

Future spacecraft will integrate among others vision-based navigation, remote sensing, and enhanced onboard autonomy workloads, which partially rely on artificial intelligence, such as deep neural networks. Such workloads come with significant computational demand and are generally data-intensive functionalities benefitting best from heterogeneous processing units and accelerators. Meeting such computational demand in an energy and cost-efficient manner will be among the next challenges for modern spacecraft.

Unfortunately, state-of-the-art (SOTA) methods for constructing space-grade processors and the rigorous development time cycle for getting such systems to space ready are not well suited to meet this computational demand quickly or only at significant costs, also in terms of energy consumption.

There is already a significant performance and power gap between current space grade and commercial-of-the-shelf (COTS) processors, spanning one order of magnitude in the instructions-per-Watt metric. This and the desire to run artificial-intelligence-based application workloads, but also high-bandwidth streaming and telecommunication applications, motivates the inclusion of heterogeneous compute units, such as graphics processing units (GPUs), accelerators, application-specific integrated circuits (ASICs), field-programmable gate arrays (FPGAs), and digital signal processors (DSPs). Moreover, COTS products have

* This work is supported by the Fond Nationale de Recherche in Luxembourg, through grant CS20/IS/14689454 – HERA.

several advantages over space-grade components in terms of energy efficiency, scalability, reusability of hardware and software ecosystems, and shorter production cycles, which has led multiple studies [1]-[4] to suggest using COTS compute systems and tiled architectures in particular.

In heterogeneous compute systems, tiled architectures simplify the integration of heterogeneous processing units by allowing designers to instantiate processing elements (PEs) as tiles and by connecting them through on-chip networks. This is particularly true if the interconnect provides additional means for controlling how PEs can interact with each other and with all tile-external resources (e.g., IO) [5]. However, while programming for multiple instruction set architectures (ISAs) is well understood, adding heterogeneity induces the additional challenge for system architects to also handle potential dependability-related issues within the processing elements (PEs), in particular since their close proximity increases the likelihood of faults and errors propagating. If treated in a classical manner, each PE type would have to be qualified, analysed for faults, hazards, and errors, and tested for the particular space-imposed challenges. In addition, each PE type would need to be adjusted to provide means to tolerate the harsh environment one encounters in space, which today often requires reverting to larger technology nodes.

Ideally, all types of faults (including component failures) should be mitigated by the On-Board Computer & Data Handling System (OBCDHS) since repair capabilities in space remain limited due to the distances spacecraft travel and due to the enormous costs of recovery missions. Instead, dependability should be considered right from the design phase and maintained throughout the Assembly, Integration, Verification, and Test (AIV&T) stages.

Future application processing needs manifest primarily in an increasing demand for faster memory, high-speed I/O channels, high data rates, high-speed communication, and an overall higher throughput when compared to classical fixed-satellite services (FSS), all in the range of gigabits per second. Next-generation avionics such as radar, optical/laser sensors, and high-resolution cameras for onboard intelligence or autonomous functionalities, when combined with AI-based inference, generate large amounts of data that is best processed on board. However, doing so further increases the demand for more high-performance resources.

To achieve the high-performance requirements, we need roughly 2E15 floating point operations per second (TFLOPS) within today's power budget, mostly in terms of matrix-vector multiplication workloads. It is, therefore, necessary to integrate high-performance computing systems in spacecraft. They will enable processing complex big data workloads on board and more accurately, but they will also exhibit a large degree of latent parallelism. This parallelism helps accelerate computational tasks. However, as we shall see, part of such parallelism can also be used to increase the resilience of onboard computing by means of dynamically forming redundant computing units.

Redundancy allows survival and recovery from unforeseen faults, hazards, and errors caused by the rigorous environment one encounters in space. When allocating additional resources over time, results can be validated and re-executed in the presence of faults (temporal redundancy) while exploiting parallelism in this allocation allows computing multiple instances of the same result, which consensus algorithms reduce to the single correct result the system needs while masking faulty instances (spatial redundancy).

We can therefore conclude that shifting from space-grade to high-performance COTS compute platforms will not only give us the performance novel spacecraft workloads need but possibly also the means we would need to sustain the resilience of critical functionality. In this paper, we discuss what additional means will be needed and what levels of resilience one might achieve.

### *1.1. What levels of dependability do we need?*

Dependability, by principle, reflects a user's trust and confidence in the correct operation of a system. This implies operating safely and securely through unexpected situations such as faults but also cyberattacks. Dependable systems are resilient to a certain class of faults if dependability can be maintained for extended periods of time and ideally for the entire mission time. To obtain higher levels of dependability, one can trade energy efficiency and performance to increase the dependability of a system. For example, one might degrade the performance of less critical systems so that critical systems obtain the resources required to mask occurring faults. One might temporarily tap into additional resources to form a sufficient level of redundancy, and one might adapt the system and relocate necessary functionality to compensate for the permanent failure of a processing element. Additional resources are also needed for recovery-related activities [6].

In addition to the design and manufacturing-related dependability concerns, space poses further challenges due to space being a harsh environment. Radiation might induce single (SEU) and multi-event upsets (MEU), altering the computing system's state by changing the charge deposited in memories (mostly flash and RAM, but also of the registers and latches a processor is built of). When interpreting such charges, it reads changed values, and the software might read flipped bits or corrupted data which may even behave arbitrarily by executing functions outside the regular

control flow. The latter is why certification often requires removing all such code.

In mitigation of the effects of SEU and MEU faults, a state is typically encoded such that decoding reveals the correct value despite a certain number of bit-flips and an indication that the value is no longer correct for a larger number. This capability can also be found in SOTA ECC-enabled memories[†]). On top of this, critical functionality can be executed redundantly, and results compared to detect or mask incorrect behaviour.

More complicated to treat is the second class of radiation-induced: single (SELs) and multi-event latch-ups (MELs). They occur as charged particles deposit themselves in electrical circuits, building up a power potential while getting trapped, which, when untreated, may damage the circuit and possibly the entire craft (e.g., through thermal stress). Such faults require disabling and discharging all circuits in a regular manner before undetected latch-ups exceed the thermal threshold of a circuit.

Both techniques have been successfully applied in high-reliability (HiRel) components, including deep in the pipeline, like with LEON-3FT [7]. Unfortunately, applying such techniques within the respective components requires costly re-design and individual qualification for space, which attributes to their costs as well. Instead, we aim at applying such techniques at the PE boundary, preparing them to be effective irrespective of what circuit they protect, which ideally will relieve us from re-qualifying our system each time we re-instantiate a tile with a different PE.

### *1.2 Towards Generic Fault Tolerance for SOTA PEs*

Irrespective of the kind of fault, as long as PEs fail independently, n-fold redundancy can detect faults by comparing the results of a fault-threshold $f$ exceeding the number of replicas ($f+1$) or mask faults by voting on the result of a fault threshold exceeding the majority ($2f+1$). N-fold redundancy has as well be applied successfully in HiRel components, where it multiples the costs and energy consumptions of the singular HiRel component from which the redundant system is constructed. The same multiplicative cost factor applies to SOTA systems. However, due to the enormous power/ performance gap, the overall costs might still be beneficial on the SOTA side, even if more faults have to be tolerated simultaneously (larger $f$) to achieve the same overall resilience.

In this paper, we give partial answers to how SOTA processing elements can be secured from the outside to withstand the harsh environments in space and what the resulting resilience will be if radiation control is reflected into a software problem executed by the OBCDHS.

[†] ECC stands for Error Correcting Codes

## 2. Background and Dependability Requirements

In the following, we introduce the necessary background on avionics, the need and interplay of AI, and the use of COTS systems for these workloads. We introduce the necessary background and terminology as well on the dependability side.

### *2.1 Aerospace control and data processing systems*

Aerospace control and data processing systems (Avionics) are a class of electronic systems specifically designed for aviation-related operations. Typically, avionics systems are built from components and subsystems that, when integrated together, implement the functionality required to operate and manage the craft. As such, avionics are vital for the correct operation of spacecraft and form the backbone of our space infrastructure today. As it is comprehensively stated in [8], typical functions of the avionic system include:

- Mission and Vehicle Management
- Command and Control
- Fault Detection Isolation and Recovery (FDIR).
- Attitude and Orbit Control (for satellites) or Guidance Navigation and Control (for other spacecraft)
- Thermal Control Processing
- Power Distribution Control Processing of a spacecraft's non-vital workloads
- Telemetry and Telecommand handling
- Data Processing, Storage, and Transmission
- Data Handling
- Payload Control and the Payload itself very often mimic a lot of avionics functions

Many of these functions require significant amounts of data processing and control capabilities, in particular, if they have to act autonomously without guidance from the ground. However, fortunately, the componentized design of many avionics systems already indicates well-defined boundaries where individual components can be isolated, executed redundantly, and their results combined for use by other components and to steer the spacecraft. Software control over the degree and magnitude of this redundancy and the required additional recovery procedures already lends a great deal of flexibility, in particular since not all avionics functions are equally critical in all situations, allowing some to be traded against users to overcome difficulties.

### *2.2 Autonomy and Artificial Intelligence*

Next-generation avionics will push aggressively the level of autonomy at which spacecraft will operate. It should thereby be noted that although many autonomous systems leverage artificial intelligence (AI), one does not imply the other and wise versa. For example, rule-based autonomous systems successfully

manage spacecraft without any form of machine learning, while AI finds its application as well in systems that do not contribute to the autonomy of systems. This is not to question the enormous success AI and autonomy achieve in combination. For example, ESA's Active Debris Removal (ADR) mission aims to remove space debris from Low Earth Orbit (LEO) using Vision-Based Navigation (VBN) and Light Detection and Ranging (LiDAR), both being interpreted by AI. However, when increasing the level of automation, one must always balance the increased system complexity against the dependability concerns this entails.

At their core, many AI-based solutions boil down to vector-matrix operations that are best processed by special purpose processing elements, such as GPUs and AI accelerators. Following traditional methods, we would therefore have to harden such circuits individually and qualify them for use in space. Instead, we aim to support high-performance, real-time and high-accuracy image, video, and data processing by taking GPU and AI accelerator circuits as a black box, confining the effects of faults that may occur within them and ensuring PEs continue to fail independently, which includes ensuring latch-ups will not cause a circuit to exceed its thermal threshold.

*2.2 Dependability*

The commonly accepted definition of dependability [9] is the ability to deliver a service that can be trusted, and the trust is justified, meaning that the system delivering the service is trustworthy.

The same seminal work hints that there is an alternative definition stating, more quantitatively, that dependability of a system is its ability to avoid service failures that are more frequent and severe than is acceptable, whoever is concerned. All in all, dependability expresses itself in both robustness of the system and its ability to overcome faults before they can affect the external world. Dependability is often described in the following dimensions:

a) **Availability:** readiness for correct service.
b) **Reliability:** continuity of correct service.
c) **Safety:** absence of catastrophic consequences.
d) **Confidentiality:** absence of unauthorized disclosure of information.
e) **Integrity:** absence of improper system alterations.
f) **Maintainability:** the ability to undergo modifications and repairs; and
g) **Security:** which is covered by a), d), and e) collectively

For space systems, dependability is a key attribute to conducting missions successfully. It demands all key services are available and reliable whenever they are needed. From there, one can start integrating tasks that ensure the safety of the spacecraft as well as the confidentiality and integrity of its command-and-control subsystems and the information it processes. Maintainability is currently not a priority for space systems.

## 3. Dependability of the New Space segment

Small satellite systems market with masses of 600 kg or below experienced exponential market growth in the last decade and are now a major part of our space infrastructure today. They are deployed primarily in Low Earth Orbit. Studies analyzing dependability issues of small satellite systems (e.g., Jacklin, 2016) have shown that a significant fraction of all missions over the past years failed due to system failures. From 2000 to 2016, the fraction of small satellite missions ending in failure or partial failure was as high as 35%. Of those satellites that were successfully placed in orbit between 2009 and 2016, 43% failed for the same reason. Notice that this already excludes launch failures. Excluding partial failures, we still have 33% of small satellites in that period that were no longer operational, short after deployment.

The above is comparable to the numbers reported in [12] for academic space projects. The study by Jacklin explicitly excluded academic endeavors due to their reduced requirements.

Dependability issues are getting more severe and widespread as the complexity of small satellite systems grows. In fact, dependability has become one of the biggest challenges designers of those missions have to tackle. Part of the dependability issues in small satellite systems are due to insufficient verification and validation, risking unnoticed design flaws of the newly developed systems. However, the majority of all failures can still be attributed to reliability and availability problems due to operating cost-effective, performant equipment without sufficient protection in harsh environments.

The more small satellites are used by governments, military, and enterprises to proliferate optical observation, electronic intelligence, vessel monitoring, and communication; the more pressing becomes the need to make them as least as dependable as any other critical infrastructure we utilize on Earth.

## 4. New ways of improving dependability

As we outlined in the introduction, the dependability of a system (here, a computing system on a chip) is its capacity to operate safely and securely through component failures while avoiding failures of the critical services it offers. Therefore, the dependability improvements shall ensure that a sufficiently large rate of faults can be tolerated without the system losing its ability to deliver critical functionality. This can be assured if only all the single points of failure are removed. In the case of redundant processing units, this means each such unit has to be isolated to fail independently, which also implies decoupling the power

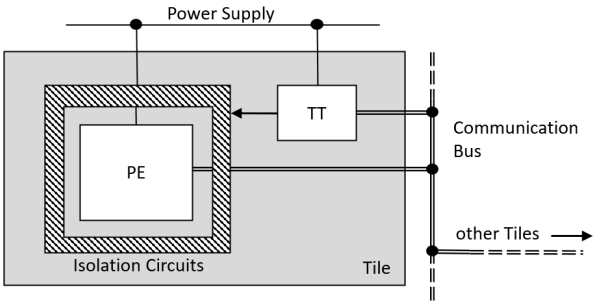


**Figure 1 Tile instantiated with a processing element and the trusted trustworthy circuitry to confine faults (including radiation induced). Shown is also the embedding of this tile in the power supply and communication infrastructure, but not the circuitry that protect the latter.**

supply and clock distribution networks of these units. Both are already seen in system-on-chip designs for safety-critical systems on the ground (e.g., by offering multiple power domains on a chip and by following a globally asynchronous, locally synchronous design with fault-tolerant clocks [11]). Such designs can be pushed forward to the extreme where the only remaining failure mode is loss of structural integrity (e.g., due to critical mechanical damage), which can only be avoided by distributing components on several carriers (e.g., assemblies or boards). For most missions, we consider the residual risk of a mission failure due to loss of structural integrity sufficiently low.

Unfortunately, such a full separation and, thereby, isolation is only feasible for systems where the individual processing elements will not interact or only through traditional, HiRel voting circuits, which consolidate the proposals of individual tiles into a single fault tolerant output. Pushing the design in this direction would severely limit the flexibility of the use of the available processing resources. Instead, we propose an alternative design where the interconnecting components are as well considered to become possibly faulty, where they may need to be repaired, and faults in them recovered and where faults in critical subsystems can be tolerated.

Our key insight, given redundant units and the right architectural support, is that processing elements can cope with faults on their own to ensure the availability of critical services (including repair and recovery) and, thereby, dependability over extended periods of time. In In particular, we strive for a design where the occurrence of a Single-Event Latch-up (SEL) or Single Event Functional Interrupt (SEFI) is not treated at the SoC level but reflected to and handled at the processing element (PE) level that is affected from them.

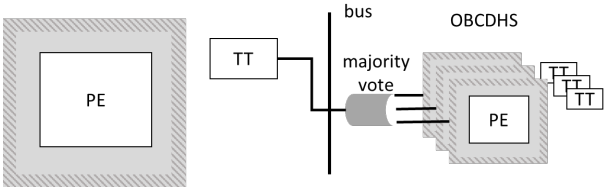


**Figure 2 In the fault resilient configurations, the PEs currently responsible for on-board command and data handling execute the power-cycling and recovery procedure as a sequence of votes to mask faults in one OBCDHS PE.**

Figure 1 shows the architectural support required to obtain PE-level fault handling for arbitrarily affected PEs. For simplicity, we show here only the protection of the PE, but similar mechanisms are applied to the communication bus and to the protection-enabling trusted-trustworthy component TT. Failing PEs might generate arbitrary outputs, with the potential to affect any other PE and, more generally, any state in the system if not properly contained. The first ingredient is, therefore, an isolation circuit, which ensures that despite PE failure, access to the PE is limited only to a subset of this state and possibly only to voting circuits such that the state can be altered only consensually by means of a majority vote. In addition, to fight radiation-induced charge buildup and the faults that may originate from that (SELs, SEFIs, …), the TT may instruct the isolation circuitry to reset and power cycle the PE to remove SEFIs and SELs without parasitically powering the input/output and communication infrastructure.

The TT and the isolation circuit, thereby, do not act on their own but are instructed by other PEs responsible for radiation control to perform these operations after the PEs relocate the accepted functionality and re-establish it after SELs and SEFIs have been handled. Of course, as hinted above, given direct control over a TT, a faulty radiation managing PE could jeopardize critical functionality on PEs, which is why we avoid such direct control and the single point of failure it would imply and subject TT control to voting among redundant radiation-control PEs.

Figure 2 illustrates this consensual control over the affected PE. It should be noted that architecturally, the PEs responsible for radiation control are not at all special, and they may fail as well, requiring power cycling and recovery. Our architecture is, therefore, fully symmetric in allowing in principle, any general-

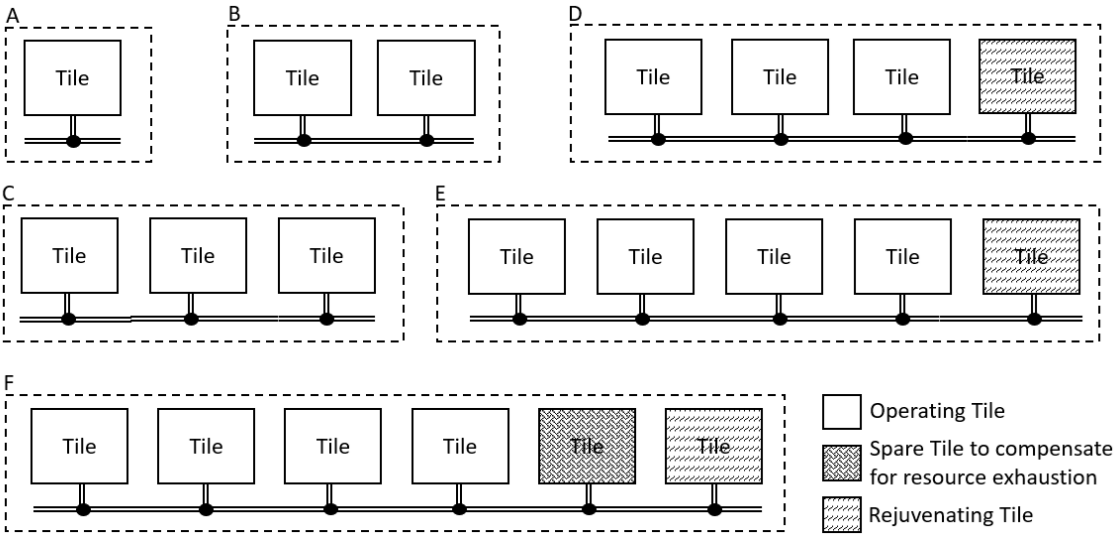


**Figure 3 Fault tolerant configuration variants**

purpose PE[‡] to assume this role and take over the responsibility to power cycle any other PE.

Figure 3 shows six examples of possible configurations. In the following, we discuss the degree to which they are able to tolerate radiation-induced faults before we return to the TT and argue how we can ensure that the TT and isolation circuitry themselves do not become single points of failure.

**Configuration A**: One PE (no redundancy / fault detection capabilities); Simplex or 1oo1. Failure of the PE immediately manifests as system failure. We take this configuration as a reference point for performance and power consumption in the absence of fault tolerance and assume the PE performs all operations.

In a larger system, a subsystem can be configured in this manner if its functionality is not critical.

**Configuration B**: Two PEs; Dual Modular Redundancy (DMR), 1oo2. Both PEs perform the same operation (typically in lockstep). Mismatch of the results indicates the presence of a fault, but without contextual knowledge, it remains unknown which of the replicas failed. This configuration does not support radiation fault handling at the PE level because a single PE would need to trigger this handling, which can jeopardize the other PE if the triggering PE is faulty. Moreover, this configuration cannot guarantee availability in the sense of masking the fault since, in case one PE fails, it remains unknown which output is correct.

In a larger system, this configuration can be used for subsystems that can tolerate becoming temporarily unavailable and where recovery is handled in a different subsystem.

**Configuration C**: Three PEs performing the same operations (e.g., in lockstep); Triple modular redundancy (TMR) or 2oo3. Given a voting mechanism to select the majority of matching outputs, this configuration can tolerate one crash fault, and if operations are triggered externally in all replicas, one arbitrary faulty (i.e., Byzantine) PE. The configuration loses its ability to tolerate faults if one of the PEs is power cycled and recovered. Voting can be either synchronous (i.e., all replicas hold the output simultaneously) or asynchronously (e.g., by buffering the proposals of the individual replicas).

**Configuration D**: Four PEs; TMR (as in configuration C), but with the possibility to tolerate faults during recovery. While the 4th PE is disabled for power cycling, the remaining three can still reach consensus, even in the presence of a fault. This ensures that critical operations (e.g., the power cycling itself) can be completed. Recovery is triggered in a consensual manner so that replicas need to agree on whose turn it is to be power cycled at a given point in time.

In a larger system, the subsystem responsible for radiation control should be configured in that manner and assist other PEs in their recovery.

[‡] Special purpose PEs, like accelerators and GPU tiles have circuits that are dedicated to perform a certain functionality very well. It might therefore be difficult / impossible to implement radiation control on such PEs. However, given enough general-purpose PE, it is sufficient to use only them and exclude the special purpose PEs from assuming the role as radiation control instance. They are still recovered regular, after removing SELs and SEFIs.

| Config: | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| FT | 0 | 1C | 1C | 1C | 1B+1C | 1B+1C |
| A | N | N | Y | Y | Y | Y |
| proRec | N | N | N | Y | Y | Y |
| Sync | sync | sync | sync | sync | sync | async |
| $FT_{conf}$ | 1oo1 (none) | 1oo2 | 2oo3 | 2oo3 | 3oo4, 2oo4 | 3oo5, 3oo4, 2oo4 |
| $A_{ovh}$ | 0 | S + v | 2S + v | $3S + 4S_{isol} + v$ | $4S + 5S_{isol} + v$ | $5S + 6S_{isol} + v$ |

FT – fault tolerance, number of faults tolerated:
(B – Byzantine (including Crash), C – Crash-only)
A – availability of the equipment at fault occurrence
$FT_{conf}$ – fault tolerance configuration
proRec – proactive recovery for dormant fault removal (μSEL, SEFI)
Sync- – type of synchrony of operation is assumed for processing elements
$A_{ovh}$ – area overhead for implementing a given type of fault tolerance over the singular system

**Table 1 Summary of fault tolerance configurations and their features**

**Configuration E**: Five PEs; Byzantine Fault Tolerance (BFT) also during recovery. As mentioned above, BFT requires synchronous activation of the replicas so that no confusion can occur about the operation to perform. If this is not given, additional replicas are required to tolerate faults by first reaching an agreement on the operation before executing it. It is sufficient to add only one replica for recovery on top of the four for BFT as long as the agreement to include a replica in the voting group also reaches an agreement on the replica to be rejuvenated in one indivisible agreement operation. In contrast to D, recovery can be performed reactively upon detection, and the replica is freely selected rather than proactively recovering in a round-robin manner.

**Configuration F**: Six replicas, same as E, but an additional spare is available, ready to be brought in, in case one of the PEs fails permanently. Alternatively, this PE can be enabled to lift the simultaneous agreement constraint of configuration E.

All presented configuration aim at progressively reducing the possibility of remaining single points of failure bringing down the system (mentioned before, losing the mechanical integrity, sustaining critical structural damage). In fact, the presented methodology of constructing a dependable many-core system boils down to following those steps:

1) Overprovisioning the system with tiles
2) Decoupling tiles to reduce common failure modes
3) Protection at the tile level
4) Allowing for partial synchrony
5) Design the distributed decision-making process to remove single points of failure by deploying
    i. trusted-trustworthy components, or
    ii. the consensus among control instances

What remains to be seen is under which conditions the trusted components (TT and the isolation circuitry) can be trusted to not fail themselves due to radiation-induced faults. First, it should be noted that due to their simplicity, the required functionality can be implemented as extremely small cross-section circuits, which reduces the likelihood of being affected by radiation. Second, due to their small cross-section and given the excess transistor budget one obtains from the smaller technology nodes, it will be feasible to replicate these components as well, such that one replica is power cycled together with the PE while the other ensures the operation is performed as intended, switching roles after the recovery completes.

## 5. Dependability analysis of the presented fault-tolerant configurations

To see what level of dependability we achieve, we have evaluated the fault-tolerant configurations introduced in the previous section with a representative processing element and a System-on-Chip SEFI failure rate of 0.1/device-day. Such a rate can be expected for spacecraft in LEO with a standard shielding of 1.5 mm Al and under quiet solar weather conditions [10]. Notice that by further increasing the available resources, it will be possible to sustain even harsher conditions by dynamically adjusting the resources spent to tolerance to the current radiation level.

The reliability of the representative PE is considered exponential; early defects (covered by burn-in and test campaigns) and wear-out (in harsh environments, faults have to be dealt with at much shorter timescales than classical wear-out).

Dependability, to narrow the scope of the analysis, will be hereafter evaluated in the context of reliability as the probability of survival for a given amount of time and in the context of availability as the likelihood of delivering results when requested.

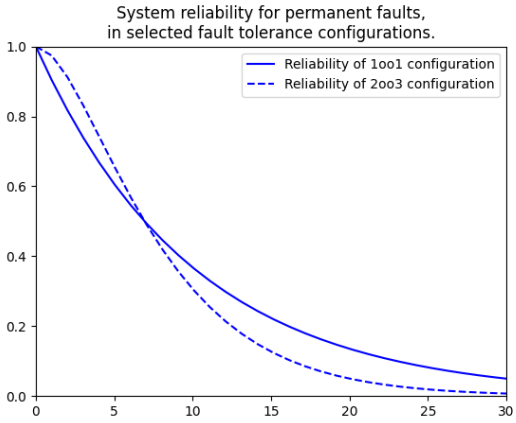


**Figure 4 System reliability for permanent faults, in 1oo1 and 2oo3 fault tolerance configuration**

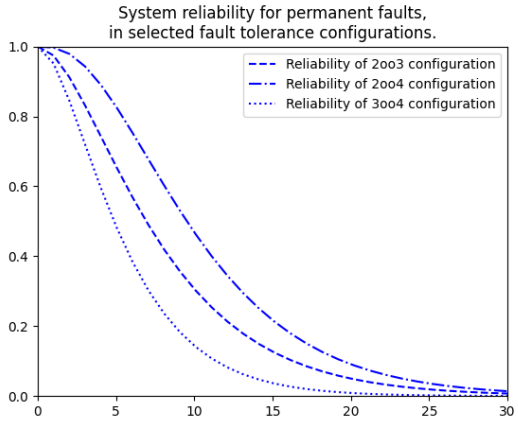


**Figure 5 System reliability for permanent faults, in 2oo3, 2oo4, 3oo4 fault tolerance configuration**

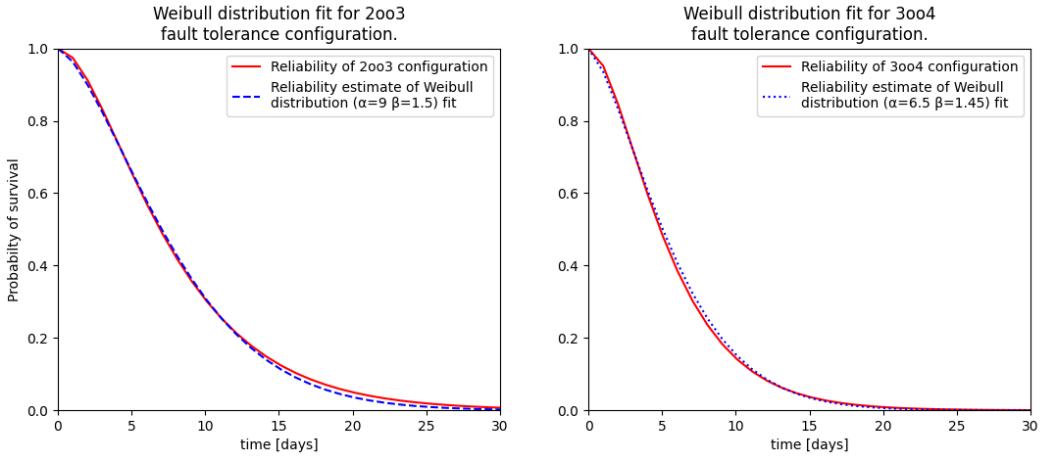


**Fig. 6 Weibull distribution fits for complex system reliability estimations**

As it has been shown in the previous section that designing a system to withstand not only crash faults but also rarer and more critical byzantine types of faults requires a significant amount of resources in terms of redundant processing elements. This cannot be avoided unless the spatial redundancy becomes traded off with temporal redundancy. Savings in components will then result in expenses in processing time; hence, it may negatively affect availability. However, it should be noted that the above holds only for critical operations, which often require less powerful and smaller cores that can be deployed in abundance in many-core systems-on-a-chip. Non-critical functionality that can easily be restarted will experience the same performance as on the ground, although with more disruptions due to radiation faults, and it will be occasionally relocated to power cycle the PE.

It should also be noted that massive redundancy enables the masking of faults of a transient nature, but if the faults are not temporary and immediately become permanent failures, the fault tolerance scheme breaks. Because of the fact that redundancy means several components operating at the same time, their failure rates decrease the expected time to failure of a system.

Figure 4 depicts the reliability of a single processing element (nor redundancy or 1oo1 compared to classic, lockstep triple modular redundancy where 2 out of 3 (2oo3) processing elements have to be operational to deliver expected fault tolerance. The reliability of the 2oo3 variant drops below 1oo1 at 0.7 * $MTTF_{1oo1}$ (expected time to failure of simplex, 1oo1, no-redundancy variant).

In Figure 5 reliability of more advanced fault tolerance schemes is depicted. Classic TMR 2oo3 for reference, and Byzantine fault tolerance in 3oo4 scheme. Unsurprisingly, 3oo4 variant yields even lower reliability if it is set for compensating Byzantine faults.

However, if the system designed for tolerating Byzantine faults (3oo4) is considered in the scope of crash faults as well, then for compensating for the crash faults, its reliability (2oo4) is higher than in classic TMR. This indicates that one of the principles for

increasing reliability, and therefore, dependability, is equipping the systems with the ability to adapt and cover more advanced and simpler fault modes. A system, even in a degraded state after the first permanent failure and crippled ability to compensate for byzantine faults, can still tolerate, more likely, crash fault and operate correctly (in a non-Byzantine context) beyond the first failure.

In space systems, on the other hand, a lot of faults will be temporary (SEU, SETs, SEFIs, and even latch-ups if removed before thermal damage is done to the semiconductor die). If faults are temporary, it means they are removed sometime after the occurrence. Let's investigate how the fault tolerance schemes will affect their top-level system instantaneous failure rate.

First, by looking at the single Processing Element and 2oo3 variant reliability (Figure 4), it can be noticed that even if PE in simplex mode exhibits failures according to an exponential probability distribution, the reliability composition of a couple of PEs in the fault-tolerant scheme is described by a different probability distribution. In Figure 6 (left), we show the survival function fit for the Weibull distribution ($\alpha = 9, \beta = 1.5$) modeling the 2oo3 fault tolerance scheme (crash faults of TMR system), and in Figure 6 (right), we show the survival function fit for the Weibull distribution ($\alpha = 6, \beta = 1.5$) modelling 3oo4 fault tolerance scheme (Byzantine fault-tolerant system). Estimation of reliability distributions of fault-tolerant systems variants allows us to evaluate their behavior in the time domain by observing the hazard function, which provides information about the instantaneous failure rate. Hazard function for 1oo1 (exponential distribution baseline), the 2oo3 and 3oo4 variants have been plotted in Figure 7. It can be noticed that only for some time the instantaneous failure rate of a fault-tolerant system is lower than the constant failure rate of the reference system. Then it exceeds it. This time sets the minimum refresh period for the fault-tolerant system, in which all the redundant components need to be reset or rejuvenated in such a way that the dormant fault modes are removed (e.g., power cycling to remove micro latch-ups). If the recovery is not conducted often enough, the failure rate of a fault-tolerant system becomes worse than that of a system that lacks fault tolerance, putting into question the usefulness of the whole redundancy scheme. Fortunately, the order of magnitude performance gain of SOTA technologies compensates by far the performance costs of both the redundancy and the dynamic recovery compared to HiRel systems.

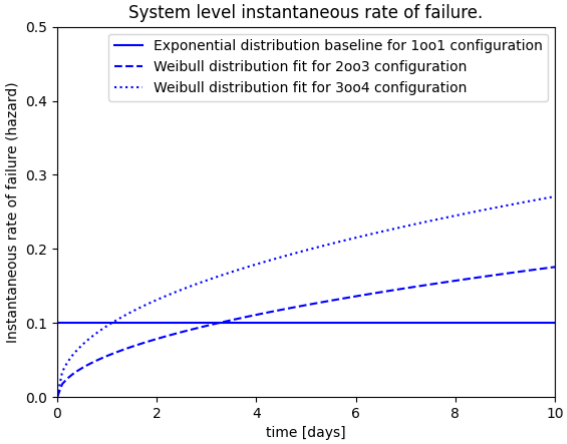


Figure 7 System level instantaneous rate of failure for different system configurations

## 6. Conclusions

In this paper, we have reviewed the existing dependability augmentation techniques widely used in space-embedded systems. We presented an alternative approach for constructing data processing and control systems based on software-controlled recovery, tiled architectures, and trusted-trustworthy components that ensure fault isolation and recoverability.

We have outlined the main reliability characteristics, features, and overheads associated with the classic and new fault-tolerant configurations, especially in the presence of permanent faults, which are unavoidable in harsh environments, such as space.

The reliability analysis indicates that, unlike in ground-based computer systems, more overprovisioning does not necessarily mean better since the degrading failures will come sooner than could be expected by operating single processing nodes without redundancy.

Complex fault tolerance, in the presence of permanent faults, may be claimed to be better than simple fault tolerance, with less redundancy, if it is acceptable to adapt fault tolerance schemes, as the system degrades gracefully. Such an approach would mean that system will start its mission with Byzantine Fault Tolerance, which after sustaining damage, falls back to Triple Modular Redundancy and Crash Fault Tolerance, down to Dual Modular Redundancy with re-execution, until no more faults can be tolerated.

Our analysis shows that regular recovery operations are needed to not follow the above path and instead keep the instantaneous fault rate of the redundant system lower than for the non-fault tolerant system. The rate of such recovery activities must remain significantly higher than simplex system failure rates in order to be able to claim the system's hazard function improvement.

Overall, we show in this paper the constraints for implementing fault-tolerant configurations for mission-

critical, space-embedded systems constructed out of inherently susceptible components.

**References**

[1] Iturbe, Xabier, Didier Keymeulen, Patrick Yiu, Daniel Berisford, Robert Carlson, Kevin Hand, and Emre Ozer. "On the use of system-on-chip technology in next-generation instruments avionics for space exploration." In IFIP/IEEE International Conference on Very Large Scale Integration-System on a Chip, pp. 1-22. Springer, Cham, 2015.

[2] Bokil, Harshad. "COTS Semiconductor Components for the New Space Industry." In 2020 4th IEEE Electron Devices Technology & Manufacturing Conference (EDTM), pp. 1-4. IEEE, 2020.

[3] Budroweit, Jan, and Hagen Patscheider. "Risk assessment for the use of cots devices in space systems under consideration of radiation effects." Electronics 10, no. 9 (2021): 1008.

[4] Sedlmayr, Hans-Juergen, Alexander Beyer, Klaus Jöhl, Klaus Kunze, M. Maier, and T. Obermeier. "COTS for Deep Space Missions." In Radiation Effects on Integrated Circuits and Systems for Space Applications, pp. 381-401. Springer, Cham, 2019.

[5] Asmussen, Nils, Marcus Völp, Benedikt Nöthen, Hermann Härtig, and Gerhard Fettweis. "M3: A hardware/operating-system co-design to tame heterogeneous many cores." In Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 189-203. 2016.

[6] Sousa, P., Bessani, A. N., Correia, M., Neves, N. F., & Verissimo, P. (2007, December). Resilient intrusion tolerance through proactive and reactive recovery. In 13th Pacific Rim International Symposium on Dependable Computing (PRDC 2007) (pp. 373-380). IEEE.

[7] LEON3FT Microcontroller GR716A. Cobham Gaisler AB, September 2021. https://www.gaisler.com/doc/gr716/gr716-ds-um.pdf

[8] IPC-THAG. Rep. Technical Dossier on Avionics Embedded Systems. European Space Agency, October 15, 2020.

[9] Avizienis, Algirdas, J. Laprie and Brian Randell. "Dependability and its threats - A taxonomy." IFIP Congress Topical Sessions (2004).

[10] Adell, P., G. Allen, C. Asbury, C. Barnes, Rob Davies, S. Guertin, F. Irom, W. Parler, and L. Scheick. Guideline for the selection of COTS electronic parts in radiation environments. Pasadena, CA: Jet Propulsion Laboratory, National Aeronautics and Space Administration, 2019, 2019.

[11] Paulitsch, Michael, and Wilfried Steiner. "Fault-tolerant clock synchronization for embedded distributed multi-cluster systems." In 15th Euromicro Conference on Real-Time Systems, 2003. Proceedings., pp. 249-256. IEEE, 2003.

[12] Swartwout, Michael, and Clay Jayne. "University-class spacecraft by the numbers: success, failure, debris.(but mostly success.)." (2016).