Assessing Hardware Fault Tolerance: Comprehensive Study of Fault Detection Methods and Their Trade-Offs

Anonymous Author(s)

Extended Abstract

High reliability is a necessity to a variety of safety-critical applications [5, 8], including on-board computer systems in space applications, automotive, and avionics. Moreover, in the next years, the necessity for high robustness will become even more important [9], as semiconductor feature size will continue to decrease, and emerging applications in the space domain will push more computation from Earth to space [2], where vulnerability constraints are tighter. In all of these application settings, hardware errors arise due to various reasons, e.g., particle radiation or increased susceptibility due to decreased feature size. As a result, critical systems require increasingly heightened robustness for safe and reliable operation to be guaranteed.

To enable reliability, prior works propose conventional [8] and microarchitectural [1, 4, 7, 10] methods to detect and/or correct faults at the hardware level. Conventional fault tolerance techniques, such as component redundancy, typically provide robustness at the expense of increased power consumption and design area, with the latter also exacerbating the system's susceptibility to errors. Given the energy constraints inherent in many critical systems, the associated power and area overheads of fault tolerance methods cannot be typically tolerated. To mitigate this issue, microarchitectural fault tolerance alternatives try to maintain low overheads by leveraging microarchitectural insights.

Designing error detection methods has 4 challenges to be considered: (i) the efficiency in detecting transient and permanent errors, (ii) the error detection latency, (iii) the performance overheads, and (iv) the area/power overheads of the proposed design. While the detection efficiency and latency for conventional methods are well-quantified by field tests or real-world implementations, the same metrics are often overlooked when proposing microarchitectural methods, where emphasis is primarily placed on performance and area. However, in specific applications such as high-risk space and avionics systems, maintaining minimal detection latency may hold greater priority than minimizing performance overheads [3]. We find that detection methods inherently involve tradeoffs between some of these 4 metrics with any particular tradeoff being potentially suitable for one critical usecase, but undesirable for another, depending on the system requirements and operational environment.

To this end, our goal of this work is to *extensively characterize and compare existing detection methods taking into consideration all 4 metrics and unveil corresponding tradeoffs.* We study 3 methods for hardware error detection within a processor: 1 conventional and 2 microarchitectural ones. **1) Dual Modular Redundancy**, a conventional widely adopted scheme in real-world critical systems, where each hardware component is replicated twice and computation is repeated in both copies of the hardware components. **2) Redundant Execution with Simultaneous Multithreading (SMT)** [6], a micorarchitectural method that leverages SMT threads to improve the performance of redundant execution, spawning 2 SMT threads each executing a copy of the same program. This method has not been adequately evaluated regarding error detection efficiency and latency. **3) Parallel Detection (ParDet) with Heterogenous Cores** [1], is considered state-of-the-art micorarchitectural method in terms of performance and area overhead, but has also not been evaluated in terms of detection efficiency. In ParDet, fault detection is parallelized, with each program segment independently evaluated on an auxiliary low-power core.

We compare all methods on the Gem5 simulator, conducting error injection experiments, and demonstrate that microarchitectural methods provide slightly lower detection efficiency and higher latency compared to conventional methods, yet achieving reductions in design area with minimal performance impact but by exhibiting tradeoffs unappealing for high risk scenarios. These results, render microarchitectural techniques more applicable for lower risk dependable systems (like the cheaper nanosatellite missions) where performance needs to be prioritized, while for more critical scenarios conventional methods are still needed because of their with high detectability and low latency.

We believe our study is timely, as it stands at the intersection of critical technological shifts, i.e., increased error rates due to transistor shrinking, and penetration of cloud computing in space. Fundamentally, we aim to bridge the gap between real-world critical systems and fault tolerance approaches and assist effective solutions to the current and future challenges in reliable systems engineering. We hope our analysis can better inform reliability and system engineers on choosing more suitable fault detection methods depending on the application requirements, operational environment, and risk margins, by offering insights into both the detection efficiency characteristics and associated tradeoffs of each existing error detection method.

References

[1] S. Ainsworth and T. M. Jones. 2018. Parallel Error Detection Using Heterogeneous Cores. In *DSN*.

[2] N. Bleier et al. 2023. Space Microdatacenters. In *MICRO*.

[3] Ricky W. Butler. 2008. A Primer on Architectural Level Fault Tolerance. *NASA TM-2008-21510* (2008).

[4] A. Naithani et al. 2022. Reliability-Aware Runahead. In *HPCA*.

[5] A. Nardi and A. Armato. 2017. Functional safety methodologies for automotive applications. In *ICCAD*.

[6] S.K. Reinhardt and S.S. Mukherjee. 2000. Transient fault detection via simultaneous multithreading. In *ISCA*.

[7] E. Rotenberg. 1999. AR-SMT: a microarchitectural approach to fault tolerance in microprocessors. In *FTCS*.

[8] D. Siewiorek and P. Narasimhan. 2005. Fault-tolerant architectures for space and avionics applications. *NASA Ames Research* (2005).

[9] J. Srinivasan et al. 2004. The impact of technology scaling on lifetime reliability. In *DSN*.

[10] C. Weaver and T. Austin. 2001. A fault tolerant approach to microprocessor design. In *DSN*.