A RISC-V Open-Source Platform for Reliability Evaluation in Safety-Critical Operating Systems

POSTER: A RISC-V Open-Source Platform for Reliability Evaluation in Safety-Critical Operating Systems

Giorgio Cora, Simone Tollardo, Sarah Azimi
Department of Control and Computer Engineering (DAUIN)
Politecnico di Torino, Turin, Italy
giorgio.cora@polito.it, simone.tollardo@studenti.polito.it, sarah.azimi@polito.it

ABSTRACT

The growing adoption of RISC-V architectures in safety-critical applications has increased the demand for robust and reliable computing platforms. In operating system-based architectures, radiation-induced faults can alter system memory or register contents, leading to unintended behavior or, in the worst case, system halts. To mitigate such risks, the dependability of operating systems has to be investigated and enhanced, reducing the probability of system failures in the presence of errors. This work proposes a reliability-oriented open-source platform based on the porting of the MIT PDOS xv6-riscv education Operating System for the VisionFive 2 board.

CCS CONCEPTS

• Computer systems organization • Dependable and fault tolerant systems and networks • Reliability

KEYWORDS

RISC-V, Operating Systems, Open-Source, Robustness, Safety-critical systems

ACM Reference format:

Giorgio Cora, Simone Tollardo, and Sarah Azimi. 2025. POSTER: A RISC-V Open-Source Platform for Reliability Evaluation in Safety-Critical Operating Systems. In 22nd ACM International Conference on Computing Frontiers (CF'25), May 28-30, 2025, Cagliari, Italy. https://doi.org/10.1145/3719276.3727950

1 Introduction

The introduction of the open-source RISC-V ISA led to the development of many custom architectures targeting different fields of applications. Among these, safety-critical environments gained particular interest. In such cases, both performance and reliability must be ensured, from both hardware [1] and software [2] perspectives. Over the years, various architectures have been proposed, which embed either the classical mitigation techniques such as Triple Modular Redundancy (TMR) and Dual Modular Redundancy (DWC), as well as custom solutions [3] aiming to improve reliability by taking advantage of the open-source nature of RISC-V cores. However, dependability is not just a hardware concern. Software must also be protected against radiation-induced faults, especially when running an operating system [4]. Errors in system memory or registers can cause unexpected failures or crashes, possibly leading to disastrous consequences in a safety-critical environment.

This paper presents a reliability-oriented platform on a RISC-V board running a modified xv6-riscv, an open-source operating system. To evaluate OS reliability, we developed a custom fault injection platform that assesses key kernel components, focusing on the scheduler and virtual memory manager in this study, though it supports testing other modules as well. The platform enables fault injection into the file system's binary code, providing a comprehensive dependability analysis. While xv6-riscv is a lightweight educational OS, its open-source nature allows in-depth reliability evaluation of individual modules, facilitating targeted mitigation strategies instead of system-wide modifications.

The proposed platform embeds the VisionFive 2 board (VF2), a Commercial-Off-The-Shelf (COTS) RISC-V Single Board Computer, running a modified version of the xv6-riscv operating system.

2 xv6-riscv porting for VF2

The RISC-V architecture defines three main privilege levels to separate hardware control from OS execution: Machine Mode (M-Mode) for firmware and bootloaders, Supervisor Mode (S-Mode) for the OS kernel, and User Mode (U-Mode) for user applications. To enhance the dependability of the proposed platform, the OS was integrated with a Supervisor Binary Interface (SBI), enabling interaction between S-Mode and M-Mode firmware. The chosen SBI implementation, OpenSBI, provides essential services such as CPU bootstrapping, multi-core initialization, I/O handling and

CF '25, May 28–30, 2025, Cagliari, Italy

ACM ISBN 979-8-4007-1528-0/2025/05.
https://doi.org/10.1145/3719276.3727950

system reset. It also introduces reliability features like memory protection, fault isolation, and secure system call handling to prevent privilege escalation. Additional safety mechanisms, including watchdog timers, controlled multi-core scheduling, and secure boot, further improve fault recovery and system integrity.

Adapting the open-source OS to the JH7110 SoC required removing all M-Mode dependencies, enabling full S-Mode execution. The memory map was adjusted for SoC peripherals, while page table initialization and virtual address translation were refined to prevent access faults. Multi-core support was improved to ensure proper CPU wake-up and task scheduling, with modifications to the context switch module to maintain consistency in floating-point registers.

To support fault injections, storage was migrated from QEMU VirtIO disk to SD card via SDSPI, introducing a more efficient disk layout. UART drivers were reworked to ensure stable serial output, reduce IRQ storms, and improve logging.

These enhancements allowed the xv6-riscv OS to run entirely in S-Mode on the VisionFive 2 board, providing a reliable, open-source platform for testing, validation, and OS reliability analysis on a RISC-V processor.

2.2 Fault injection Platform

The fault injection platform is designed to modify the binary files of the platform's data and instruction memories at compile time, using custom scripts to enable fast and efficient testing. The framework systematically introduces random bit flips into binary files to assess software resilience. It reads a binary file, randomly selecting the positions for the bit flips. For each selected bit, a modified copy of the binary file is created with the bit flipped, and the corresponding bit positions are recorded.

The framework generates multiple altered versions of the binary file and stores detailed information about each bit flip in a file. This data is later used to correlate the location of faults with their effects during post-processing. All modified binaries are eventually compiled in the filesystem image together with xv6-riscv user programs. The filesystem is then copied to the SD card, from which fault injected programs are loaded.

Benchmarks are executed and controlled by a custom-developed user program, which manages their execution, checking timeouts and logging progress to disk, allowing to recover from hard faults.

During execution, test results are collected via UART communication and later analyzed using the fault injection framework's analysis tool. This ensures that each test is properly executed, recorded, and evaluated, aiding in the assessment of software robustness against memory corruption faults.

3 Experimental Results

To validate the effects of software-induced faults on the system, four different benchmarks were used: CoreMark, Dhrystone, Dijkstra, and MatMul. For each benchmark, 10,000 random bit flips were injected into the system's data and instruction memory, resulting in a total of 40,000 test runs.

Table I. Fault Injection Runs Results

Benchmark	Trap	Halt	SDC	Others	Total
Coremark	322	5271	1	654	6248
Dhrystone	200	5068	150	689	6107
Dijkstra	124	4939	43	105	5211
Matmul	78	4902	6	15	5001

The output of each run was classified into different categories based on the benchmark results. *Halt* indicates a fault that caused the processor to stop functioning. *Silent Data Corruption (SDC)* is recorded whenever incorrect values are printed via UART. All raised traps are categorized under the *Traps* classification, while *Others* include any unidentifiable system's strange behavior or failures due to multiple events.

Table 1 summarizes these results, while in Fig. 1 an additional classification on the type of traps is highlighted.

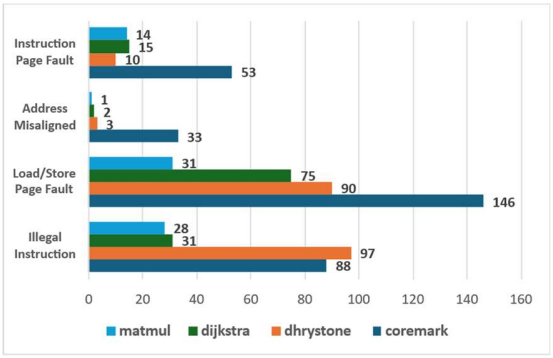

Figure 1: Traps classification

4 Conclusion

This work introduces a reliability-oriented platform based on the porting of the xv6-riscv operating system to the VisionFive 2 board, embedding a custom fault injection environment to evaluate the robustness of the proposed architecture.

Acknowledgement

A special thanks to Augusto Leogrande and Andrea Portaluri, who provided help and support during the porting process of the OS to the RISC-V platform.

REFERENCES

[1] E. Vacca, et al., "On Assessing the Robustness of RISC-V Soft Cores for Space Systems by Mission-Tailored SEU Analysis," *2024 31st IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, Nancy, France, 2024, pp. 1-4, doi: 10.1109/ICECS61496.2024.10848907.

[2] C. De Sio, et al., "SEU Evaluation of Hardened-by-Replication Software in RISC-V Soft Processor," *2021 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, Athens, Greece, 2021, pp. 1-6, doi: 10.1109/DFT52944.2021.9568342

[3] P. R. Nikiema *et al.*, "Towards Dependable RISC-V Cores for Edge Computing Devices," *2023 IEEE 29th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, Crete, Greece, 2023, pp. 1-7, doi: 10.1109/IOLTS59296.2023.10224862.

[4] Wali, A. Sánchez-Macián, et al., "Analyzing the impact of the Operating System on the Reliability of a RISC-V FPGA Implementation," 2020 27th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Glasgow, UK, 2020, pp. 1-4, doi: 10.1109/ICECS49266.2020.9294858