Self-Test Libraries for RISC-V safety-critical applications: recent advances

M. Sonza Reorda, R. Cantoro, Josie E. Rodriguez Condia, A. Ruospo, E. Sanchez
Politecnico di Torino, DAUIN, Torino, Italy

May 7, 2023

Abstract

RISC-V adoption is rapidly expanding even to safety-critical application areas, such as automotive, space, robotics, and health-care. In these areas, it is crucial to guarantee that the probability of a critical failure stemming from a permanent hardware fault falls below a given threshold, often following the guidelines and rules of standards and regulations. In this scenario, possible solutions must combine functional safety goals with constraints related for example to the hardware and performance overhead, the flexibility, the ease of adoption. In the last decade, Self-Test Libraries (STLs) became a commonly adopted solution widely supported by semiconductor, IP and EDA companies, allowing system companies to deploy effective in-field test mechanisms able to detect a high percentage of permanent hardware faults arising during the operational phase (e.g., due to aging). This paper summarizes the latest advancements in the area of STLs for RISC-V architectures, emphasizing the advantages stemming under this perspective from the open instruction set architecture.

Introduction

Among the different areas where RISC-V is expected to play a major role is the one of safety-critical applications, such as automotive, space, robotics, healthcare. In these areas, it is crucial to guarantee that the probability of failures stemming from permanent hardware faults is kept under a defined threshold. To achieve this goal, different solutions can be adopted, including the introduction of fault-tolerant solutions (e.g., ECCs, lockstep, TMR) [1]. However, fault-tolerant approaches are sometimes too expensive in terms of hardware cost, and may lack in flexibility, being implemented in hardware. In some cases, the system company would prefer using the same CPU core for different applications having different levels of functional safety (ASILs in ISO26262 nomenclature): a fully protected CPU core could be unsuitable to cover this range of requirements. For this purpose, adopting a mix of safety mechanisms that can be arranged and tuned at system level would be preferable.

In the last decade, Self-Test Libraries (STLs) became widely adopted in the area: the basic idea is that the semiconductor or IP company develops a set of software test procedures (often at the assembly level). The system company includes them in the application and system software, which launches their execution with the due frequency, checking whether the produced results match the expected ones. In the negative case, a permanent hardware fault is detected, and the application can react accordingly before the fault produces any major consequence. STLs are currently delivered by many semiconductor and IP companies together with their CPU cores, including ARM, Synopsys, NXP, Infineon, STMicroelectronics [2]. The list of permanent faults detected by a given STL is typically assessed by the semiconductor or IP company by performing its functional fault simulation with respect to structural fault models (such as stuck at or transition delay faults). In this way, the STL can be delivered to the system company without disclosing any proprietary information about the core. The STL assessment is done resorting to commercial tools such as Z01X by Synopsys or Xcelium by Cadence, which belong to a new generation of EDA tools targeting functional safety.

Combining STLs with other safety mechanisms, and tuning their frequency of activation, the system company can achieve the desired fault coverage (FC) (Diagnostic Coverage in the ISO26262 nomenclature) without modifying the CPU hardware and avoiding expensive hardware fault-tolerant solutions. Moreover, since STLs are executed at full speed, they can detect delay defects as well, which are known to be the prime consequence of aging. Finally, STL execution can be split in chunks which can easily accommodate into the idle times of the application, thus resulting in a less invasive solution. On the other side, STLs are expensive to develop, since their generation mainly stems from a manual effort, although guided by several test algorithms published in the literature in the last years. The goal of this paper is to underline how the characteristics of the RISC-V open instruction set architecture (ISA) may ideally support the adoption of STLs for safety-critical applications. In fact, it limits the cost for their generation, allows their re-use across different cores, and eases the introduction of ad hoc instructions and hardware for STL support.

STL generation and re-use

Several recent works demonstrated that suitably written STLs for RISC-V cores can detect a high percentage of stuck-at [3], transition [4] and path delay faults [5]. Moreover, the RISC-V open ISA is ideally suited to make STL generation as much independent as possible from the specific core implementation, allowing the adoption of formal techniques to speed up the generation (thus limiting the required manual effort) and easing the STL re-use across different cores. In Table 1, we show the results collected via fault simulation on two RISC-V cores – namely, the CPU modules of the PULP ibex and PULP ri5cy. Both cores have been synthesized with the Silvaco 45nm OpenCell library. The two STLs have been manually crafted, making use of systematic algorithms for specific modules (e.g., register file), while exploiting the vectors from an automatic test patterns generation (ATPG) tool for generating the instructions for some others (e.g., ALU). Noteworthy, the FC values reported in the table do not take into consideration the impact of faults that cannot produce any failure in the mission application (*safe faults* according to the ISO26262 nomenclature). The number of such faults is typically considerable, and evaluated in post-processing by assessing the behavior of the mission application. Hence, the actual *diagnostic coverage* figures for the STLs reported in Table 1 are expected to be much higher than pure FCs. Given the vast amount of material available for the RISC-V architecture, STLs can be in part synthesized starting from existing code (e.g., developed for verification purposes [3]). We are currently working on the development of tools to support test engineers in STL generation (e.g., identifying the parts of the code that require additional instructions to allow the detection of the fault effects).

Ad hoc hardware for STL support

The flexibility characterizing the RISC-V ISA by its open ISA allows also for the inclusion of new instructions aimed at supporting STLs, increasing their fault detection capabilities and reducing their duration. As an example, we propose to add in the RISC-V architecture an ad hoc module able to compute the signature out of all the values flowing on the data bus during a given period. In this way, it is possible to limit the amount of memory required to store the results produced by each STL, and speed-up their execution (implementing in hardware a mechanism which is often implemented in software). To give an idea on the impact of hardware modules specifically added for enhancing the fault observability, we recently conducted a study to evaluate the FC gain given by the possibility of tracing the content of specific flip-flops of the designs. Our results conducted on the ri5cy processor shown that a 128-bit trace buffer able to dynamically select a subset of flip-flops highly increases the FC; in the worst case scenario, we tested 99.15% of the transition delay and 98.75% of the stuck-at faults that were masked and not propagated to observable points by original STLs.

Table 1: *Fault Simulation results on two RISC-V CPUs*

RISC-V Core	Faults	Duration [Clock Cycles]	Stuck-at Fault Coverage [%]
ibex	118,346	109,294	82.59
ri5cy	159,326	80,455	82.18

Conclusions

STLs are widely used as a safety mechanism to be combined with others in order to match the strict requirements existing in many applications areas where reliability and safety are major concerns. RISC-V allows to support the generation and usage of STLs leveraging on its open ISA. This paper summarized the advantages stemming from this characteristic. Work is currently being done by both the academic and the industrial community to improve the available STLs for RISC-V architectures in terms of achieved fault coverage, targeted fault models, and required execution time.

Acknowledgment

This work was partially supported by the Italian ICSC National Research Centre for High Performance Computing, Big Data and Quantum Computing in the frame of the NextGenerationEU program.

References

[1] Fraunhofer IPMS. “RISC-V processor core for functional safety”. In: *White paper*. 2021.

[2] Paolo Bernardi et al. “Development Flow for On-Line Core Self-Test of Automotive Microcontrollers”. In: *IEEE Transactions on Computers* 65.3 (2016), pp. 744–754. DOI: 10.1109/TC.2015.2498546.

[3] Annachiara Ruospo et al. “On-line Testing for Autonomous Systems driven by RISC-V Processor Design Verification”. In: *2019 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*. 2019, pp. 1–6. DOI: 10.1109/DFT.2019.8875345.

[4] Riccardo Cantoro et al. “Effective techniques for automatically improving the transition delay fault coverage of Self-Test Libraries”. In: *2022 IEEE European Test Symposium (ETS)*. 2022, pp. 1–2. DOI: 10.1109/ETS54262.2022.9810392.

[5] Riccardo Cantoro et al. “New Perspectives on Core In-field Path Delay Test”. In: *2020 IEEE International Test Conference (ITC)*. 2020, pp. 1–5. DOI: 10.1109/ITC44778.2020.9325260.