Towards High-Reliability System Design Using Agile Hardware Development Flow

Junchao Chen, Li Lu, Markus Ulbricht, Milos Krstic
IHP, Frankfurt (Oder), Germany

GOALS

- **Propose an iterative agile hardening strategy for the design of high-reliability systems.**
- **Achieve the design trade-off between system reliability, time-to-market, and performance.**

MOTIVATIONS

- **Increasing challenges in ensuring system reliability in safety-critical and mission-critical applications, especially with transistor scaling reaching the deep nanometer range.**
- **Agile hardware development methods have shown potential in reducing hardware development costs.**

METHODS

- **Utilizes agile hardware development platforms.**
- **Chisel-level fault injection platform, like Eris.**
- **Reliability analysis to identify the most vulnerable modules.**
- **Selective harden to tailored the vulnerable components**
- **Faster iterative design cycles.**

OUTLOOK

- **Faster simulation-based fault injection methods based on machine learning.**
- **More accurate module-level reliability analysis methods.**
- **Smarter cross-level selective harden.**

Introduction

- Hardware faults in integrated circuits can cause system performance decline, data corruption, and even system failures. Existing fault mitigation techniques can lead to over- or under-protection, creating a need for more fine-grained methods.
- Agile hardware development, leveraging high-level abstractions and rapid innovation execution, has emerged as a cost-effective alternative to traditional design approaches. However, its application in developing high-reliability hardware has been limited.
- Despite the promise of fine-grained mitigation techniques like selective hardening in designing highly reliable systems, implementing them in the prevalent agile hardware design flow is typically complex, costly, and time-consuming.

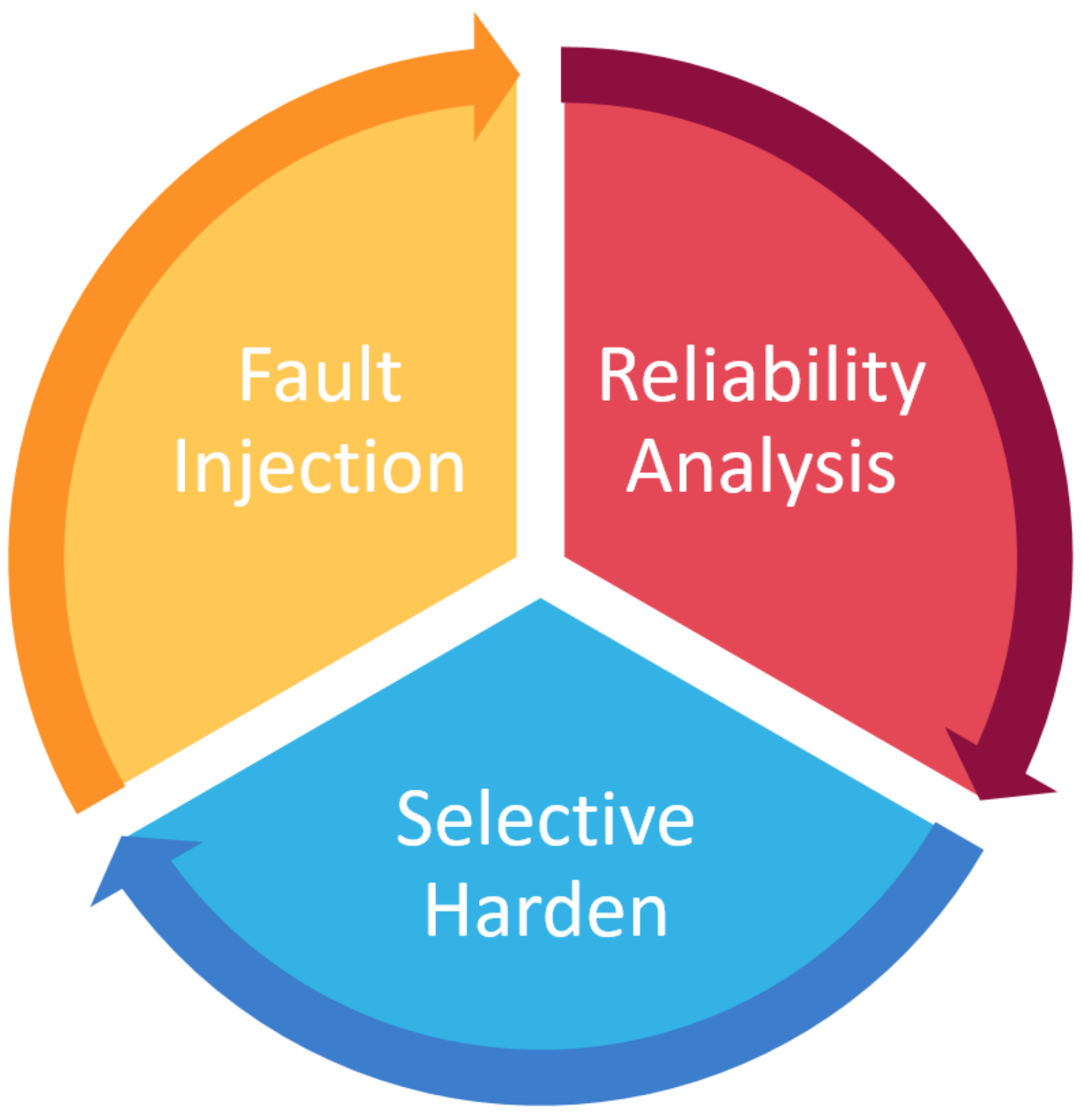

Fig. 1 Agile flow for high reliability system design.

- The paper introduces a strategy involving three iterative steps - fault injection, reliability analysis, and hardening method selection.
- Aims to balance the trade-off between reliability, time-to-market, performance, and other factors in the development of high-reliability hardware.
- The fault injection module simulates both transient and permanent faults at a high abstraction level for the target component.
- The collected fault injection data is analyzed in the reliability analysis module and identify the most vulnerable submodules.
- The hardening method selection module uses the reliability analysis results to determine suitable fine-grained hardening approaches. The selected hardening method is then integrated into the original design, leading to a new design iteration to ensure appropriate protection level.

Fault Injection

- Simulation-based fault injection is an important technique for analyzing system behavior under faults and helps to accurately locate vulnerable hardware components by injecting faults randomly or selectively and monitoring their propagation through the design.
- For agile development, faults can be directly injected into the hardware description level, such as Chisel, for faster vulnerability identification and mitigation impact analysis.
- The Eris platform is an example of a fault injection framework that can analyze any RTL design that can be lowered to FIRRTL (e.g., Chisel and Verilog), converting it to a C model.
- An alternate approach involves the insertion of transient and permanent fault models directly into the target component at the abstraction layer.

Reliability Analysis

- Reliability analysis is at the heart of system design and management, determining the applicability of existing hardening methods and identifying the vulnerability of the system.
- It uses data from design requirements and fault injection results, primarily relying on the Architecture Vulnerability Factor (AVF) to identify vulnerable registers.
- Traditional methods often overestimate vulnerable registers, leading to system over-protection, hence the need for fault propagation tracking during program execution at higher abstraction levels.
- Comparing vulnerability indicators with design requirements helps identify sub-modules requiring optimization, thereby minimizing the impact of increased reliability on system performance, power consumption, and other factors.

Cross-Layer Selective Harden

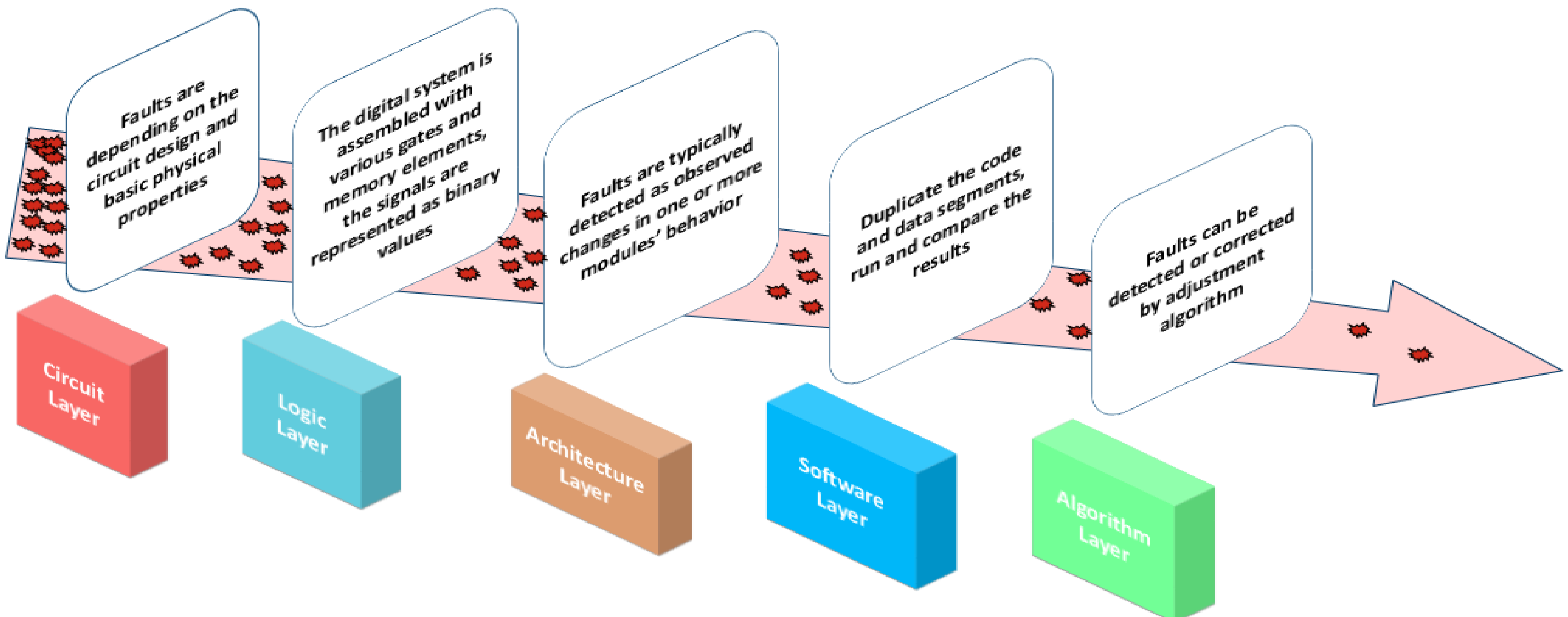
Faults are depending on the circuit design and basic physical properties
The digital system is assembled with various gates and memory elements, the signals are represented as binary values
Faults are typically detected as observed changes in one or more modules' behavior
Duplicate the code and data segments, run and compare the results
Faults can be detected or corrected by adjustment algorithm
Circuit Layer
Logic Layer
Architecture Layer
Software Layer
Algorithm Layer

Fig. 2 Cross-layer faults propagation across abstraction layers of a computing system. During the propagation, different masking effects may block the propagation of the fault, thus, reducing the impact on the system's reliability.

- Cross-layer selective harden system has the potential to achieve higher average performance, more reliable operation, lower cost and energy consumption by taking advantage of the information and resources across different system layers.
- The inherent advantage of hardware redundancy is also convenient for reconfigurable mechanisms at different abstraction layers, such as core level N-Module Redundancy (NMR), adaptive voltage scaling (AVS), dynamic task scheduling, etc.
- The combination of existing technologies on different abstraction layers can result in several modes of operation.

Member of Leibniz Association

A13C