Master Thesis Proposal

A Hardware-based Fault-Injection Framework for QDI Circuits

Julian Spitzer
01325893

April 22, 2023

TECHNISCHE UNIVERSITÄT WIEN

1 Problem Statement

In the context of digital circuits a fault can be described as a bit flipping either temporarily on a wire (Single Event Transient (SET)) or permanently in some memory cell (Single Event Upset (SEU)). Such faults are caused by particles hitting the circuit and cannot be avoided, especially in high-radiation environments. As such, building fault-tolerant circuits is a very important topic especially in safety-critical environments.

Fault injection in digital circuits is a popular tool for testing the robustness of a design against such faults. It is also a very difficult task because it requires the capability to change the value of a single wire or gate deep inside a circuit.

In the synchronous domain, this task is somewhat simplified because time is a discrete factor: Every storage element in the circuit changes on the global clock edge and faults are only relevant if their effect is still observable at a clock edge, meaning small glitches can easily go unnoticed. Toolchain providers like Xilinx [Le12] and Intel [Int] nowadays even provide Intellectual Property (IP) cores for fault injection in synchronous circuits.

In asynchronous circuits (like QDI), fault injection gets much harder. Time is no longer discrete, meaning that any gates and any cells inside the circuit can change their value independently from each other making it much harder to predict when a fault would cause issues. Even a very short glitch can have an immediate impact if it happens at the wrong place. Additionally the correct behaviour is harder to verify because it is impossible to compare against a single state, it can only be verified by checking a sequence of events.

Taking all these factors under consideration, the parameter space for exhaustive fault injection experiments on QDI circuits is huge (fault location, fault length, fault start time, implementation technique) and testing all possible combinations takes millions of individual runs. In simulation, a test framework for automated fault injection experiments on asynchronous circuits was developed in [BHNS21]. The aim of this thesis is the development of a similar test framework in hardware to allow large-scale execution of fault injection experiments and gaining a significant speed-up by avoiding the bottleneck of long simulation times.

2 Expected Results

The expected outcome of this thesis is a hardware test framework for performing fault injection experiments on quasi delay-insensitive (QDI) circuits. The hardware framework is controlled through a UART interface and a Python library running on the host PC.

The QDI circuit under test (UUT) has to be augmented with additional error injection signals. The wires where faults should be injected are then 'cut up' and replaced with logic gates with the second input being one of the error injection inputs. Enabling a fault injection then causes a fault on this wire (depending on the gate the fault is a bit flip or a stuck-at-0/stuck-at-1 error).

The test framework itself is a synchronous circuit which forms a wrapper around the augmented QDI circuit. Depending on the configuration, it will run the circuit with a certain sequence of input words and inject single-bit error pulses on the error injection lines over the configured parameter space in three dimensions (injection gate, injection time, fault duration). To avoid the UART interface being too much of a bottleneck only the runs with observed errors are reported back to the host PC.

The standard sequence of actions for running a fault injection experiment looks like this:

1. Provide a QDI circuit as production rule set (PRS)

2. Use scripts provided by the test framework to augment the circuit according to some parameters

3. Synthesize the test framework with the augmented QDI circuit and program the FPGA with the result

4. Configure the tested parameter space (injection time and fault duration) through the host PC

5. Start the test framework and save the results

In addition to providing the framework this thesis shall also provide results from the execution of such experiments, running on different QDI circuits with different data widths and different implementation techniques to provide an analysis of the fault tol-

erance of different QDI circuits and techniques. The results of the experiment shall also be compared to the results of the simulation framework [BHNS21].

3 Methods

Literature Review

A literature review is done in order to gain a deeper understanding of existing approaches to fault injection in digital circuits. The review covers fault injection approaches in hardware as well as in simulation for both synchronous and asynchronous circuits. The findings are then compared to what the proposed test framework of this thesis can do.

Implementation of Framework

The actual framework for fault-injection experiments is implemented. The framework includes scripts for easily integrating different QDI circuits and a small software stack for executing fault injection experiments and collecting results.

Evaluation and Outlook

Using the implemented hardware framework, fault-injection experiments are executed on different QDI circuits. The results of these experiments are then evaluated to make observations on the fault resistance of different implementation techniques and the results are also compared to the results obtained by the simulation framework [BHNS21]. The actual framework is also evaluated to gain insights on its usability and performance.

4 State of the Art

When it comes to fault injection in digital circuits, the techniques can generally be categorized into three main approaches: hardware-based (physical) fault injection, simulation-based software fault injection and emulation-based fault injection [EGRM20].

While **simulation-based fault injection approaches** are the cheapest in implementation cost, they are expensive in computational effort and run-time as simulating a circuit is very slow. To speed the process up, many approaches use pre-characterization for the simulation ([RDK+08] [HLS09] [KPW10]). There are also static simulation approaches which use specific data structures to track information about effects inside the circuit [EGRM20].

Hardware-based fault injection approaches are the fastest but also the most expensive approaches among all fault injection methods [EGRM20]. This kind of fault injection is either performed in a contact-based fashion (i.e. pin-level probes) or without contact where external energy resources (i.e. beams of laser, heavy ions, protons, and neutrons) are fired towards parts of the design to simulate a real faulty environment [EGRM20]. Hardware-based fault injection always requires an external hardware environment capable of injecting the fault. One such example for a contact-based approach is RIFLE, a general purpose pin-level fault injector [MRMS94]. It provides a framework for injecting pin-level faults and comes with a host software and trace memory for detailed storage of the systems reaction to the injected faults. For an approach without contact, the framework from [VPK+15] shall be mentioned which uses proton beams for radiation testing.

Emulation-based fault injection approaches provide a middle ground between the other two options, benefitting from the speed of hardware-based approaches and the accuracy of software-based approaches [EGRM20]. One of the most popular techniques in this field is using an FPGA to emulate the behavior of a circuit under the presence of faults [EGRM20]. One such approach for emulating SEUs on Xilinx FPGAs uses the Xilinx Essential Bits technology to speed up fault injection [DCPRT14].

The overwhelming amount of literature to be found on the topic is only aimed at synchronous circuits and is thus not applicable for what this thesis intends to do. One specific approach for asynchronous circuits which was already mentioned in the previous chapters is presented in [BHNS21]. It describes an automated setup for simulation-based fault-injection experiments on asynchronous digital circuits (thus falling into the

category of simulation-based fault injection approaches).

The framework developed in this thesis can be seen as an emulation-based fault injection approach for asynchronous (QDI) circuits, for which no previous solution could be found during literature research.

5 Relevance to the Curriculum of the Master in Computer Engineering

The thesis is highly focused on asynchronous circuits and digital design topics. In the curriculum of Computer Engineering, the following courses are most closely related:

- 182.755 Advanced Digital Design VU
- 182.754 Advanced Digital Design LU
- 182.700 HW/SW Codesign VU
- 182.701 HW/SW Codesign LU

Advanced Digital Design provides an introduction to asynchronous circuits while HW/SW Codesign focuses on how to combine Hardware and Software into one system, elevating the advantages of the two worlds.

Bibliography

[BHNS21] Patrick Behal, Florian Huemer, Robert Najvirt, and Andreas Steininger. An automated setup for large-scale simulation-based fault-injection experiments on asynchronous digital circuits. In *2021 24th Euromicro Conference on Digital System Design (DSD)*, pages 541–548. IEEE, 2021.

[DCPRT14] Stefano Di Carlo, Paolo Prinetto, Daniele Rolfo, and Pascal Trotta. A fault injection methodology and infrastructure for fast single event upsets emulation on xilinx sram-based fpgas. In *2014 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, pages 159–164. IEEE, 2014.

[EGRM20] Mohammad Eslami, Behnam Ghavami, Mohsen Raji, and Ali Mahani. A survey on fault injection methods of digital integrated circuits. *Integration*, 71:154–163, 2020.

[HLS09] Daniel Holcomb, Wenchao Li, and Sanjit A Seshia. Design as you see fit: System-level soft error analysis of sequential circuits. In *2009 Design, Automation & Test in Europe Conference & Exhibition*, pages 785–790. IEEE, 2009.

[Int] Intel. Fault injection intel fpga ip core user guide. `https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug_fault_injection`. accessed on 15 April 2023.

[KPW10] Yu-Hsin Kuo, Huan-Kai Peng, and Charles H-P Wen. Accurate statistical soft error rate (sser) analysis using a quasi-monte carlo framework with quality cell models. In *2010 11th International Symposium on Quality Electronic Design (ISQED)*, pages 831–838. IEEE, 2010.

[Le12] Robert Le. Soft error mitigation using prioritized essential bits. *Xilinx XAPP538 (v1. 0)*, page 72, 2012.

[MRMS94] Henrique Madeira, Mário Rela, Francisco Moreira, and João Gabriel Silva. Rifle: A general purpose pin-level fault injector. In *Dependable Computing—EDCC-1: First European Dependable Computing Conference Berlin, Germany, October 4–6, 1994 Proceedings 1*, pages 197–216.

Springer, 1994.

[RDK+08] Rajaraman Ramanarayanan, Vijay Sai Degalahal, Ramakrishnan Krishnan, Jungsub Kim, Vijaykrishnan Narayanan, Yuan Xie, Mary Jane Irwin, and Kenan Unlu. Modeling soft errors at the device and logic levels for combinational circuits. *IEEE Transactions on Dependable and Secure Computing*, 6(3):202–216, 2008.

[VPK+15] Tomá Vanát, Jan Pospiil, Filip Kriek, Jozef Ferencei, and Hana Kubátová. A system for radiation testing and physical fault injection into the fpgas and other electronics. In *2015 Euromicro Conference on Digital System Design*, pages 205–210. IEEE, 2015.