Comparing Voltage and Clock Glitch Attacks on a RISC-V implementation on FPGA

Roua Boulifa, Giorgio Di Natale and Paolo Maistri

Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA, 38000 Grenoble, France

Abstract

Embedded systems are ubiquitous to our daily lives, making them attractive targets for malicious actors. Ensuring their security is crucial. One significant threat is fault injection attacks on microprocessors. Understanding how these attacks affect a system's internal design is essential for assessing their overall security impact. In this paper, voltage glitch and clock glitch campaigns have been carried out on RISC-V processor that is used by researcher community and start to gain popularity in embedded system market. As a result, we provide comprehensive analysis for the glitch set up. We show that some of these models are applicable to both glitch methods. The presented fault models enable better understating of the fault injection effects, and thus, easing the process of analyzing vulnerabilities, and developing cost-effective countermeasures against fault attacks.

Introduction

Nowadays, digital systems are used in every aspect of modern life from energy management and transportation to communications and industrial production. As these systems become ever more interconnected, their security emerges as a critical concern. Among several adversarial techniques, Fault Injection Attacks (FIAs) stand out because they can effectively recover critical information or bypass security features by actively manipulating the underlying hardware during operation to induce errors.

Two of the most low cost FIA techniques are clock glitching and voltage glitching. In a digital circuit, the clock synchronizes sequential elements such as flip-flops, which update their state at each clock edge based on their inputs. The minimum clock period is determined by the worst-case propagation delay through the circuit's combinational logic. In Agoyan et al. [1], deliberately shortening one or more clock cycles is leveraged to make flip-flops sample unstable or erroneous data, then the circuit typically returns to normal operation.

Similarly, voltage glitching leverages the sensitivity of digital devices to a stable power supply. By momentarily perturbing the supply voltage either by under-powering or over-volting the device an attacker can force the transistors into an unstable state, resulting in incorrect computations or corrupted register values. Korak and Hoefler [2] combine glitches on the clock signal with glitches on the power supply bus to improve the success rate of fault injection. Like clock glitches, the voltage perturbation is temporary, yet its effects may extend over several cycles.

For both types of glitching, a detailed understanding of fault effect is essential. This is where fault models come into play: abstract representations that describe the impact of physical disruptions on circuit behavior. By integrating these models, hardware designers and software developers can predict potential vulnerabilities and develop countermeasures to secure embedded systems against such attacks.

The goal of this paper is to study the voltage and clock glitch effects in RISC-V processors implemented on an Artix-7 FPGA. By examining how transient perturbations in clock timing and power supply impact processor behavior, this work aims to provide a deeper understanding of the fault effects on the microarchitecture and support the development of more resilient embedded systems.

Experimental Setup and Protocol

In our experiment we will use the CV32E40P which is 32 bit, in-order RISC-V core written in System-Verilog. It has a 4-stage pipeline: fetch, decode, execute, and register writeback. It has 32 general-purpose 32-bit registers, named from X0 to X31. The core implements a forwarding path between the EX stage output (ALU, Multiplier and Divider results) and ID stage flip-flops, to avoid the latency of writing to register file in case of RAW (Read After Write) conflicts in the executed code. Our core implements the RV32IMC (i.e., Integer, Multiplication/division, and Compressed instructions); compressed instructions, in particular, allow for higher code density. In our experimental setup, we utilized the CW305 board NewAE Technology Inc. [3], which integrates an Artix-7 FPGA. This board features a USB interface for communication with the FPGA, an external PLL for clock management, and a programmable VCC-INT supply.

The experiment protocol is the same for clock and the voltage glitch. The process, shown in Figure 1, repeatedly cycles over several steps for each experiment.

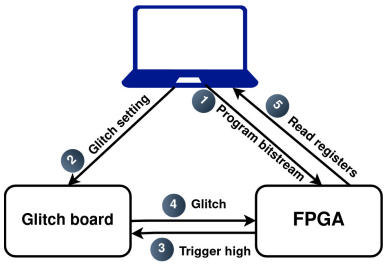

Figure 1: *Experimental Flow*

For voltage glitching, we reduced the FPGA's core voltage to 0.7V to increase the system's susceptibility to faults.

The RISC-V core was implemented by programming the FPGA with the appropriate bitstream (step #1). Subsequently, the control computer configured the glitch parameters on the glitch module (#2). These parameters depend on whether a clock or voltage glitch is injected.

The target CPU then executed the designated program, which generated a synchronization trigger (#3). This trigger initiated the glitch injection based on the specified parameters (#4). After the glitch was introduced, the target CPU read the register values and transmitted them back to the control computer for analysis (#5). This sequence was iteratively repeated with varying glitch settings to assess their effects.

The glitch parameters depend on the implementation and on the target code: in particular, *Width* and *Shift* depend on the physical implementation, and for our experiments they where set in the ranges $[10, 20]$ and $[-10, 10]$, respectively, whereas *Delay* is used to target specific instructions.

The target code is made of 32-bit-wide instructions aligned in memory; they are mainly made by arithmetic operations, fundamental in any software application, and chosen solely to easily characterize error propagation after the fault.

Experimental Results and Analysis

The outcome of a voltage or clock injection in an execution leads to one of the following classes: *Silent*: the execution outcome is equivalent to a normal execution without an injection; *Crash*: a reset or a crash occurs as a result of the injection; *Fault*: the execution outcome is different from the normal execution outcome.

In our experiments several behaviors from the literature could be observed. For instance, in several cases we were able to observe the skipping of a 32-bit instruction. After the glitch, the value of the destination register of the target instruction remains at the initialization value, which suggests that the instruction was completely skipped.

Table 1: *Percentage of silent, crash and fault over the two campaigns.*

class	clock glitch	voltage glitch
Silent	77.4%	10.5 %
Crash	0.01 %	0.17%
Fault	22.6 %	89.5 %

Other outcomes highlighted the occurrence of Early Result Capture R., Di Natale, and Maistri [4], where the unintended propagation of an instruction's result into the following instruction is caused by a timing overlap introduced by the fault. The fault affects the synchronization of the second instruction within the pipeline, leading it to erroneously read the result of the first due to delayed signals.

A consistent part of the observed faulty behaviors is due to a complex combination of more than one fault model. This is more pronounced when injecting voltage glitches than clock glitches, probably because voltage fluctuations may affect larger (and more varied) parts of the implementation.

Table 2: *Fault injection modeling.*

Fault model	clock glitch	voltage glitch
Skip	16.94 %	4.3 %
Early Result Capture	48.62 %	33.33 %
Complex	34.44 %	62.37 %

Conclusion

In this work, voltage and clock glitch campaigns have been performed on the RISC-V core. In order to compare the obtained faulty behaviors, different fault models have been analyzed.

References

[1] M. Agoyan et al. "When Clocks Fail: On Critical Paths and Clock Faults". In: *Smart Card Research and Advanced Application*. Vol. 6035. Springer, 2010, pp. 182–193. DOI: 10.1007/978-3-642-12510-2_13.

[2] T. Korak and M. Hoefler. "On the Effects of Clock and Power Supply Tampering on Two Microcontroller Platforms". In: (2014), pp. 8–17. DOI: 10.1109/FDTC.2014.9.

[3] NewAE Technology Inc. "ChipWhisperer Documentation". In: Available online. 2025. URL: https://chipwhisperer.readthedocs.io/en/latest/index.html.

[4] Boulifa R., G. Di Natale, and P. Maistri. "Early Result Capture: Racing Conditions in Pipeline due to Clock Glitches". In: *European Test Symposium (ETS'25)*. 2025.