AFIx: An Automated Tool for Fault Injection Attack Assessment and Protection

Ramsey Hamed
Caspia Technologies, LLC
Gainesville, FL, USA
rhamed@caspiatechnologies.com

***Abstract*—Several contemporary fault injection (FI) attack methods have been shown to be more effective at smaller process sizes. As this current trend continues and FI methods become more sophisticated, the need to develop an automated industry solution to the threats of FI increases. The Assessment of Fault Injection Extension (AFIx) aims to fill this need by providing the ability to both assess FI vulnerability and apply low overhead countermeasures at the gate level. AFIx achieves this using multiple novel methods, such as the application of FI models for generating potential fault lists, the use of time-to-digital (TDC) sensors for localized FI detection and access control, and the use of TDC sensors for detecting specific forms of FI attacks.**

I. Introduction

With the ever-changing landscape of hardware security, it is important to remain one step ahead of new attack methods that threaten device security. Forgoing hardware level protections can leave designs vulnerable to data corruption, denial of service, and the leakage of assets and design secrets (cryptographic keys, user information, firmware, etc.) [1].

Today, there are numerous methods for breaching device security and extracting sensitive information. Of these methods, fault injection (FI) attacks remain one of the most direct and powerful. These attacks involve purposefully injecting faults into a device to alter its behavior during runtime, with the goal of bypassing traditional chip security measures. Faults can manifest as bit flips in registers or memory, transient interconnect voltages, clock disturbances, supply voltage disturbances, or permanent interconnect changes. Faults can corrupt values in the controller or datapath of a design, changing the process flow or data within a chip. A properly timed and placed FI attack causes incorrect data to be unintendedly propagated throughout the design, potentially revealing sensitive information within [3].

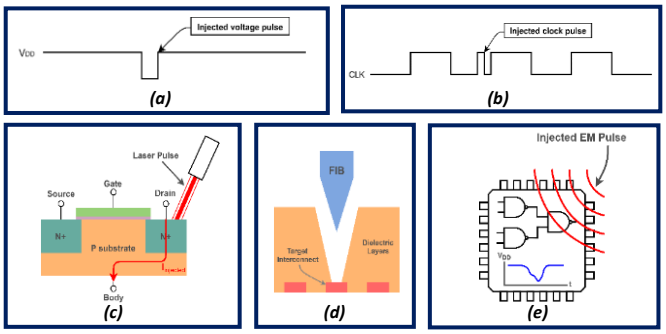

***Figure 1 - Overview of the most common fault injection methods**. These include (a) voltage glitching, (b) clock glitching, (c) laser fault injection (LFI), (d) focused ion beam machining (FIB), and (d) electromagnetic fault injection (EMFI).*

Fault injection can be carried out through several techniques that are invasive, semi-invasive, or non-invasive. The most prevalent techniques [1] are represented in Figure 1 and are described below:

- Clock glitching: an injection technique where the system clock is disturbed, causing setup and hold time violations, leading to the capture of incorrect values.
- Voltage glitching: an injection technique where the chip's voltage supply is disturbed, leading to an increase or decrease in propagation delays.
- Laser fault injection (LFI): uses directed beams of light to inject voltage pulses into the circuit, causing supply variations or flipping bits in registers and SRAM.
- Electromagnetic fault injection (EMFI): uses generated magnetic fields to create voltage pulses in power and ground interconnects. These supply variations affect propagation delays, causing timing failures.
- Focused ion beam (FIB) milling: uses semiconductor editing machinery to alter the structure of a circuit to either cut or add connections.

FI risk for microelectronics is increasing as a result of smaller process sizes and advancements in FI attack technology. Hardware security (and especially fault injection) is often overlooked when training digital designers, regardless of the increasing risks. These issues necessitate an industry-oriented tool for automated FI vulnerability analysis. However, automated design verification and protection is difficult to achieve, leaving a gap between design needs and existing solutions. Changing process sizes, manufacturing methods, and process characteristics means solutions are often only applicable to specific technologies. Additionally, increases in design complexity and size over time means that solutions may become too inefficient and time-consuming.

The Assessment of Fault Injection Extension (AFIx) sidesteps many of these previous limitations. As AFIx's vulnerability analysis and countermeasures are implemented at the gate level, advancements in process size and manufacturing methods do not affect its functionality or effectiveness. Since vulnerability analysis can be performed through simulation, tests during post-silicon validation and potential redesigns to mitigate FI vulnerabilities are avoided. Varying process characteristics also do not hinder AFIx, as FI countermeasures

DISTRIBUTION STATEMENT A. Approved for public release: distribution is unlimited.

are designed to be easily calibrated after production. AFIx can deal with increasingly complex designs by optimizing computation through empirical attack models, fault set reduction, and the use of security properties. Fault set reduction allows AFIx to determine specific locations within a design that are the most susceptible to FI attacks. This pointed analysis allows designers to save time, area overhead, and cost by implementing countermeasures only where necessary. AFIx employs multiple novel methods, including the application of FI models for generating potential fault lists, the use of TDC sensors for localized FI detection and access control, and the use of TDC sensors for detecting specific forms of FI attacks. An overview of AFIx's fault injection vulnerability flow is shown in Figure 2.

II. Methods

A. Security Properties

Every design has a unique set of security requirements that must be considered when identifying vulnerabilities. These requirements, when formalized, are called security properties. Security properties can be used to specifically pinpoint, spatially and temporally, behaviors that must not occur within a design to protect hardware assets [1].

AFIx leverages security properties through fault simulation to identify vulnerabilities. Before fault simulation, they must be created in SystemVerilog and provided to AFIx. As any fault has the potential to change design behavior, it is up to the designers to specify what behaviors are a risk to their design.

B. Fault Simulation

The functional simulation of a design under a set of induced faults, in order to observe its behavior, is known as fault simulation [1]. Within a fault simulator, individual faults are represented by time (how long and when the fault will occur), type (bit flip or stuck-at fault), and location (the specific gates affected).

Similar to the process performed by AFIx, fault simulation is commonly used to perform fault injection vulnerability analysis. Fault simulation tools (such as Synopsys Z01X) are leveraged to simulate a design under a series of faults and to determine design vulnerabilities. This process, traditionally done manually, is not only time consuming to perform but also requires subject matter expertise and leaves the designer with little useful information.

AFIx, on the other hand, is able to automate the fault simulation process. AFIx performs separate simulations of the design for each individual fault to determine which ones violate the provided security properties. The set of desired faults to simulate, the security properties, and the design netlist are all inputs into the fault simulation process. An overview of AFIx's fault simulator is shown in Figure 3.

C. Fault Generation

The set of faults to be simulated is automatically generated by AFIx. AFIx considers fault feasibility, based on empirical attack models, to create this set. This pragmatic approach gives complete fault coverage without needing to simulate every possible fault, greatly increasing simulation efficiency. For example, contemporary laser FI methods typically use one laser to inject faults, with up to two being used for more recent developments. Thus, it is unlikely a real-world fault will result from more than two lasers, reducing the number of location combinations to test.

To further increase simulation efficiency, AFIx only targets locations in the design where injected faults could possibly result in a violation of security properties. Every security property associated with a design includes specific referenced locations. By only considering faults within the fan-in cone of these locations, AFIx greatly cuts down on the number of faults to test.

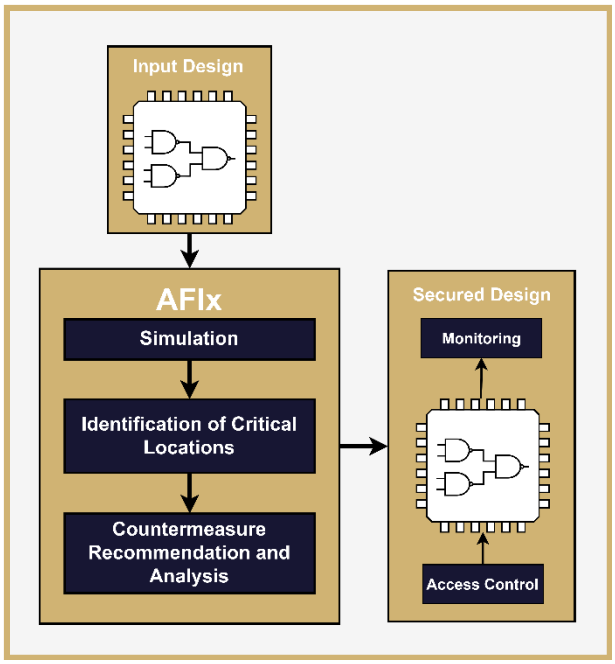

Figure 2 – An overview of AFIx's fault injection vulnerability analysis flow. *This demonstrates how an input design goes through multiple stages before being secured with monitoring and access control hardware countermeasures.*

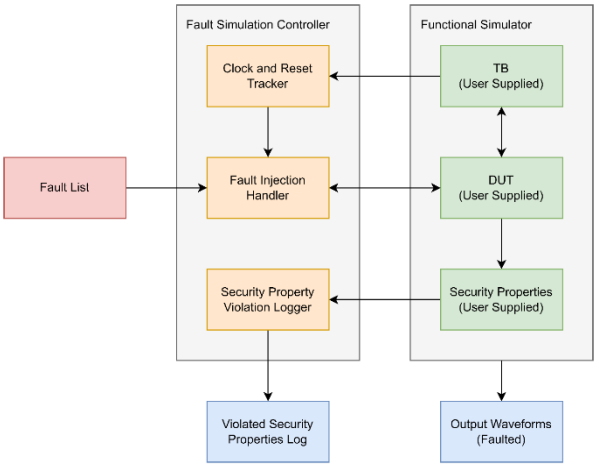

Figure 3 – The architecture of AFIx's fault simulator. *The fault simulation controller is given an omniscient view of the functional simulator, allowing it to synchronize with the simulation, reference locations, and inject faults (as specified by the fault list).*

This informed generation of faults can greatly decrease the number of simulations necessary. Consider, for example, a security property for a design with a fan-in cone of 200 gates. If one was concerned with laser fault injection, it could be assumed that at maximum, two lasers could be used. Applying these lasers to individual gates within the fan-in cone gives 200C2 = 19900 combinations of faults to simulate. If one were to blindly generate faults, for example, by simulating four faults occurring at once, then there are 200C4 = 64684950 combinations to simulate.

D. Fault Set Reduction

Fault simulation results in a list of faults (and the corresponding gates) that violated security properties. This list can be used to determine where countermeasures must be applied to secure a design. Most FI countermeasures incur some trade-offs, which usually appear as increased time, area, or power overhead [3]. To limit the incurred overhead, AFIx finds the smallest number of locations that covers all of the faults that are of concern (critical locations) [1]. While other locations are important, none of the faults would be possible without the critical locations. This way, a greatly reduced number of locations need to be protected to achieve full fault coverage. This decreases the resulting area and power overhead needed, since countermeasures are not required for the remaining locations. Once the critical locations are determined, countermeasures for those locations are then applied.

E. Countermeasures

Current countermeasures to fault injection attacks are quite limited. Some mitigations, such as gate sizing and layout-based countermeasures, aim to reduce the effects of fault injection attacks and prevent them entirely [4]. While these have been demonstrated to be effective in larger process sizes, they quickly drop off in efficacy for modern process sizes [5]. As the efficacy is highly dependent upon process characteristics, these countermeasures also require a custom implementation for each design. Additionally, these countermeasures are not applicable to all fault injection methods and do not consider the implications of successful attacks. Other countermeasures, such as redundancy and error correction, are designed to mitigate the effects of successful fault injection attacks. While these methods can be quite effective, they suffer from very large time and area overhead and can require large architectural modifications to support them.

AFIx's countermeasures, on the other hand, avoid these issues and were created to be extensible, easy to apply, and easy to calibrate. Thus, it was important to create a universal solution that could protect against the majority of fault injection methods. To achieve this, a common indicator between the various FI methods had to be established to aid in detection. One such indicator, shared by the majority of FI attacks, is localized timing variations [2]. As the timing of cells in a design are based on process, voltage, and temperature (PVT) conditions, a change in any of these conditions will result in a change in timing. Following this, supply voltage variations, indirectly caused by many fault injection techniques, can result in timing variations. LFI and EMFI attacks demonstrate this, both of which apply some form of electromagnetic radiation to a design. The applied radiation injects energy into the circuit and causes stray currents, localized around the affected areas. The increased current consumption adds unexpected strain on the power distribution network, dropping the supply voltage in affected areas. Other methods, such as voltage and clock glitching, can create timing variations through more direct means. Voltage glitching inherently uses supply variations as a means of attack, which will affect timing. Clock glitching directly causes timing violations by changing the position of a rising clock edge and can clock the design before logic values can settle.

To take advantage of this phenomenon, AFIx gears its countermeasures towards detecting timing variations. Time-to-digital (TDC) sensors provide a means of measuring timing information and are leveraged by AFIx. These sensors were chosen as they can be easily created using standard cells, eliminating the need for custom manufacturing processes. AFIx

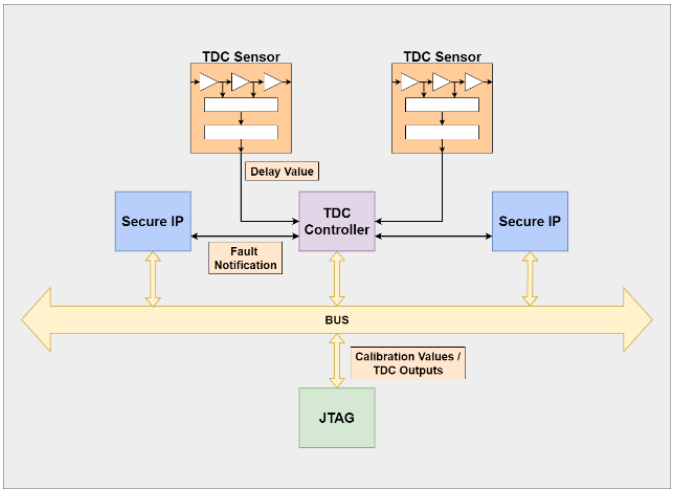

Figure 4 – A representation of AFIx's countermeasure architecture. *TDC sensors distributed through the design are connected to a centralized controller, which manages access control to security critical IPs. The controller can be calibrated through a bus-connected JTAG interface.*

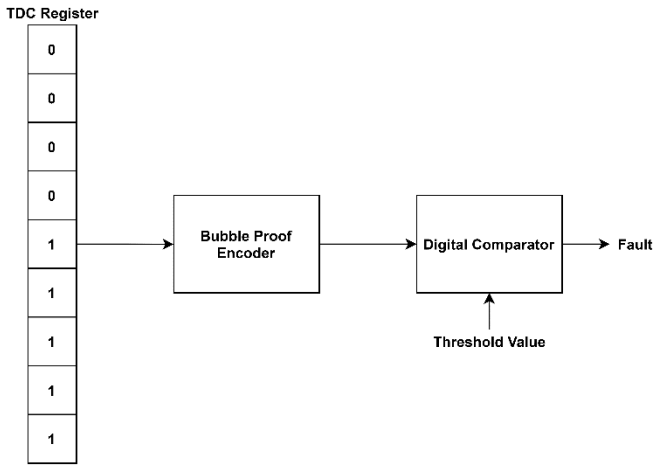

Figure 5 – An overview of how TDC outputs are used by AFIx. *To include the full desired range of timing values, TDC outputs must go through an encoder to lower routing overhead. TDC sensors are prone to bubble errors, necessitating a more sophisticated bubble-proof encoder. These values are compared within the controller using a comparator to determine whether faults have occurred.*

uses TDC sensors by inserting them into the design netlist and connecting them to clock signals near critical locations being monitored. Clock signals offer a stable reference point for timing measurements and allows AFIx's countermeasures to detect timing variations. However, this limits the countermeasures to sampling once per clock period. Designers are required to facilitate this by placing TDC sensors near critical locations during layout, as this cannot be done at the gate level.

These sensors continuously send local timing information to a centralized controller. This timing information is first passed through a bubble-proof encoder to reduce routing overhead and to prevent bubble errors, which are common with TDC sensors. The controller then determines whether measured values exceed thresholds required to constitute a FI attack (demonstrated in Figure 5). These thresholds can be calibrated through a JTAG interface after production. When timing violations are detected, the controller will assert a signal to disable the locations specified by the designer's security properties. This way, even if FI attacks are unpreventable using current methods, they are of little risk as they cannot reveal information from the design. Figure 4 provides an overview of AFIx's countermeasure architecture.

F. Determining Applied Fault Injection Techniques

For some applications, it may be useful to not only know if a fault injection attack occurred, but to also know what form of fault injection was applied. For these applications, the central controller can be adapted to leverage information from multiple TDC sensors. Each applicable fault injection method can then be detected as follows.

Voltage glitching is a global fault injection attack method, so detecting this requires information from TDC sensors across the entire chip. This FI method will be manifested as a voltage drop across all of the TDC sensors within a voltage domain. By comparing all of the TDC sensor values from a single voltage domain, it can be determined by the controller if voltage glitching occurred.

Clock glitching is characterized by a perturbation of a design's clock signal. Detection of this FI attack method can be achieved through the comparison of clock pulse widths. If the previous pulse width and the current pulse width vary by a significant margin, then there is a strong chance that a clock glitching attack occurred. The controller will keep track of the previous pulse width measurements for comparison and to determine if a clock glitching attack occurred.

LFI is a localized attack, mainly affecting only one area of the chip at a time. Most FI attacks do not operate at such a local level, so they can be differentiated from other attacks in this way. If a singular TDC sensor indicates a large voltage drop while no other sensor does, then this high localization indicates an LFI attack. The TDC controller will compare values from TDC sensors across the design to determine if an attack correlates with LFI.

EMFI can take the form of either a global or local FI attack, depending on how the attack is performed. However, it is still expected to occur centered around a single area. Generally, it can be expected that multiple TDC sensors being affected within a small, localized area would be the result of an EMFI attack. As a result of this, the controller can use the location of the TDC sensors to determine if an observed fault correlates with EMFI.

III. Conclusion & Future Work

In this paper, AFIx, an automated solution for FI attack vulnerability assessment and mitigation at the gate level is presented. AFIx is geared towards industry usage, bridging the gap between rising FI attack risk and limited designer knowledge. It provides multiple novel methods, such as the application of FI models for generating potential fault lists and the use of time-to-digital (TDC) sensors for localized FI detection and access control.

In the future, methods of improving AFIx's ability to mitigate FI attacks will be explored. Mainly, the integration of countermeasures at multiple abstraction levels to prevent FI attacks. Additionally, automated methods for security property generation will be investigated to further reduce the need for designer input. Since security properties are somewhat unique to a design, developing and formalizing them can be time consuming. Some developments aimed at reducing the effort and experience necessary to produce them do exist, such as databases that aggregate common security properties (i.e., Trust-Hub [6]). However, automated security property generation is yet to be demonstrated.

References

[1] H. Wang, H. Li, F. Rahman, M. M. Tehranipoor, and F. Farahmandi, "SoFI: Security property-driven vulnerability assessments of ICS Against Fault-Injection Attacks," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, pp. 1–1, 2021.

[2] L. Zussa, J. Dutertre, J. Clédiere, B. Robisson and A. Tria, "Investigation of timing constraints violation as a fault injection means," 27th Conference on Design of Circuits and Integrated Systems (DCIS), pp. 1–6, Nov. 2012.

[3] A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache, "Fault injection attacks on cryptographic devices: Theory, practice, and countermeasures," Proceedings of the IEEE, vol. 100, no. 11, pp. 3056–3076, 2012.

[4] Q. Zhou and K. Mohanram, "Gate sizing to radiation harden combinational logic," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 25, no. 1, pp. 155-166, Jan. 2006

[5] A. Sarafianos, O. Gagliano, V. Serradeil, M. Lisart, J.-M. Dutertre, and A. Tria, "Building the electrical model of the pulsed photoelectric laser stimulation of an NMOS transistor in 90nm technology," 2013 IEEE International Reliability Physics Symposium (IRPS), 2013.

[6] "trust-hub.org," Trust-Hub. [Online]. Available: https://www.trust-hub.org/.