FPGA Fault Injection Platform for Secure Device Design Evaluation
(work in progress)

Anton Biasizzo, Franc Novak
Jozef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
anton.biasizzo@ijs.si, franc.novak@ijs.si

Abstract—In this paper, we describe our current work on developing tools for experimental evaluation of the efficiency of implemented countermeasures against differential fault attacks on cryptographic cores in the FPGA based systems. The developed fault injection platform enables us to analyze the impact of injected faults at the selected points of the FPGA in its run time operation. In its compact version, the platform can also play the role of a Troyan horse imported in a FPGA based system.

Keywords—differential fault attacks; FPGA; fault injection; dynamic partial reconfiguration

I. INTRODUCTION

Recently fault attack strategies based on fault injection targeting different secure devices have been developed [1-3]. While these algorithms describe how to extract the guarded information from the device by inserting faults into the device they do not elaborate how the faults are actually inserted into the secure device. Typically faults are generated using some high energy bursts like LASER beam or electromagnetic interference. The problem with such fault sources is that the faults are random and often multiple faults are generated. Furthermore, the fault attacks usually must be synchronized with the secure device operation (e.g. in some attacks the faults are injected in last round, hence some control over the clock of the secure device must be provided). Because of these demands most evaluations of fault attacks on the devices rely on theoretical models while the reports of actual experimental evaluations performed at operating speed are rare.

FPGA devices are attractive platform for device prototyping as well as for small in mid-size volume production because of their cost and performance. The FPGA devices also enable easy device updates and upgrades using the remote reconfiguration which can be used to tighten the strength of the implemented hardware cores and whole device [4]. Furthermore, the ability of dynamic reconfiguration of SRAM-based FPGA can be used to modify the functionality as a result of detected attack. The dynamic reconfiguration on the other hand can pose a new threat for secure device since it can be also used for the attack. The dynamic reconfiguration can be used for inserting hardware faults into the FPGA application at run-time.

The test infrastructure like scan chains can be used to mount a fault attack [5] as well as to evaluate the resilience of the device to such attacks. While scan chains are efficient means for testing, their application in fault attacks requires switching between test and operational mode and consequently slows down the attack and/or the fault resistance evaluation. On the other hand, dynamic partial reconfiguration offers fast fault injection if it is implemented in-situ. It is to this issue that we turn in our current research.

We have implemented a fault injection platform using the dynamic partial reconfiguration. The platform enables us to inject faults at the selected flip-flops of the FPGA in its run time operation. In this way, differential fault attacks on cryptographic cores in the FPGA can be performed and the efficiency of the implemented countermeasures can be experimentally evaluated. The fault injection platform can also be used for the evaluation of the error mitigation techniques as methods for fault attack prevention. And finally, a compact fault injection core can also play the role of a Troyan horse if the attacker manages to import it in a FPGA based system. Possible case studies in this direction are also foreseen.

II. FAULT INJECTION USING DYNAMIC PARTIAL RECONFIGURATION

The configuration matrix of the SRAM-based FPGA is stored in the SRAM configuration memory. The FPGA configuration matrix is loaded on power-up or on demand. The configuration is loaded by writing either full or partial bitstream on the external configuration port. Fault injection techniques based on dynamic reconfiguration can be divided into two groups:

- Methods which manipulate bitstream based on current fault. Difference bitstream or partial bitstream is determined by software tools like JBits toolset and bitstream is uploaded on the FPGA through Select Map interface.
- Methods which identify configuration frames that have to be altered in order to insert a fault and modify it using internal configuration port (ICAP) [6,7]. The same method can be used for fault recovery when several faults are evaluated in sequence.
Our fault injection platform belongs to the second group and is implemented by two different architectures: microprocessor based architecture, and more compact finite state machine (FSM) based architecture.

Microprocessor based architecture [7] is based on MicroBlaze processor which controls the internal configuration port, synchronizes the execution of the device under test (DUT), and communicates with the external computer. While this architecture is easily tailored to specific demands, it consumes more FPGA resources and is slower compared to the FSM based architecture. This architecture is depicted in Fig. 1.

![Microprocessor based fault injector](image1.png)

**Fig. 1. Structure of microprocessor based fault injector**

As its name indicates, FSM based architecture is controlled by a finite state machine thus any change of DUT synchronization or communication protocol requires hardware modification. More elaborate protocols demand more complex FSM hence it is prudent to use only basic communication protocol. This architecture is very compact and time required to insert fault is minimal. Its structure is shown in Figure 2.

![Finite State Machine based fault injector](image2.png)

**Fig. 2. Structure of FSM based fault injector**

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Slice</th>
<th>LUT</th>
<th>FF</th>
<th>BRAM</th>
<th>ICAP</th>
<th>DSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>MicroBlaze</td>
<td>1565</td>
<td>2687</td>
<td>2652</td>
<td>10</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>FSM</td>
<td>142</td>
<td>293</td>
<td>187</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Both architectures were developed using Xilinx ISE 13.4 suite and tested on Xilinx XUPV5 development platform populated with XC5VLX110T Virtex5 device. Fault injection architectures were originally built to assess the fault coverage of developed tests for DUTs but they were modified to suite the fault attack methods. The microprocessor based fault injector was developed in Xilinx EDK environment, while the FSM based fault injector was developed in VHDL language as an independent device. Both fault injectors are easily portable to other FPGA platforms. The comparison of FPGA resource utilization of both architectures using basic communication protocol is shown in Table 1.

### III. Work in Progress

Currently we are applying the fault injector platform in order to execute different fault attack methods [1-3] on a compact 32-bit AES implementation. In this phase we are using microprocessor architecture since it is more flexible, has more elaborate communication protocol, and the DUT is small enough that it can easily fit onto the FPGA device. We plan to perform the fault attacks on other AES implementation as well, especially 128-bit and pipelined AES implementation.

After this evaluation a similar AES implementation with different mitigation technique like time/space duplication, error correction codes (ECC) [8], triple module redundancy (TMR), and error-recovering technique [9], will be evaluated. The results will be compared in order to assess the improvement gained by each mitigation technique. The selection of appropriate fault injection architecture depends on the utilization of the FPGA resources.

After this evaluation a similar AES implementation with error correction code (ECC) mitigation technique [1] will be evaluated and the results will be compared in order to assess the improvement gained by ECC.

### REFERENCES


