Automated Detection and Analysis of Hardware Trojans in FPGAs

Nicholas Houghton

University of Victoria

nhoughto@uvic.ca

December 2, 2016

Overview

Introduction: Background

- What are hardware trojans?
- How are devices affected by them?
- How do they work?

Introduction: Objectives

1. Devise a method to detect trojans in FPGAs
2. Devise a method to describe descovered trojans
3. Build an application to automate these processes.
4. Automate the visualization technique presented in [Moein, 2016]

Introduction: Two New Applications

- Automated Hardware Trojan Detector (desktop application)
- The Hardware trojan system (website)

Trojan Detector: Use-Case

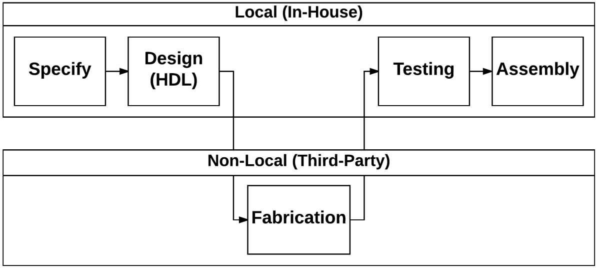

Figure: Use-Case

Trojan Detector: Methodology

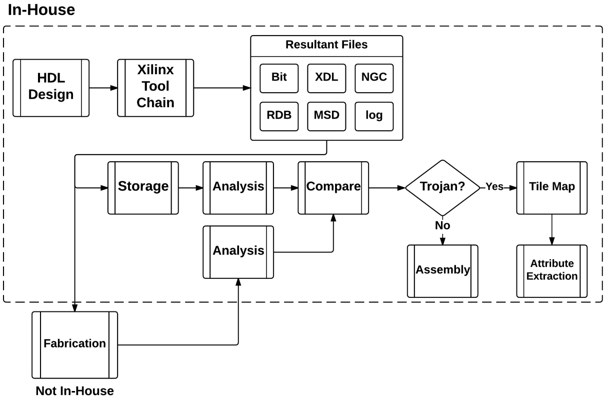

Figure: Methdology Overview

Trojan Detector: FPGA

FPGA: Field Programmable Gate-Array

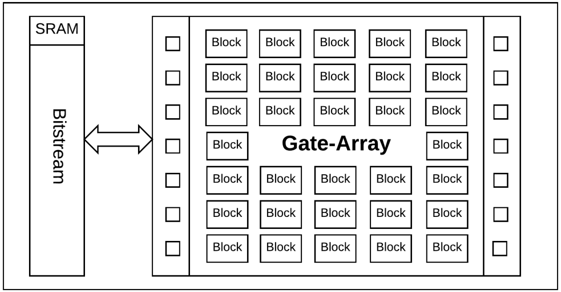

Figure: Simplified FPGA Layout

Trojan Detector: Gate-Array Architecture

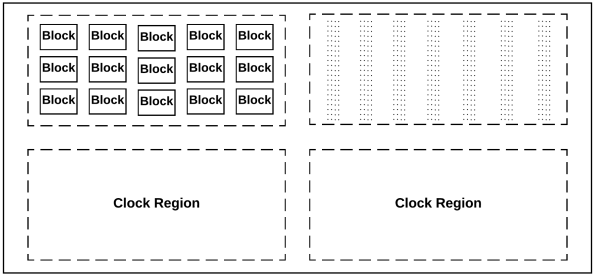

Figure: Gate-Array of Block Columns

Trojan Detector: Sub Columns

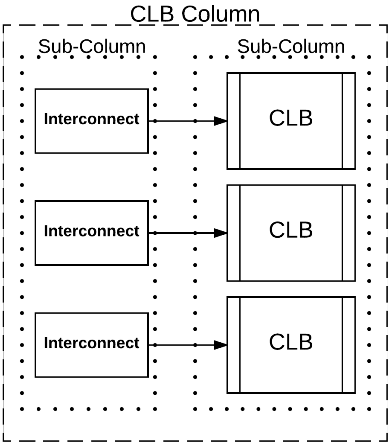

Figure: Column Composition

Trojan Detector: Frame Addressing

Table: Bitstream Frame Address Structure

Unused								BA			T	Row Address					Major Address								Minor Address						
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0

Trojan Detector: Component Mapping

$$n = (W - C) + B \qquad (1)$$

$$i = B - \left\lfloor \frac{w}{n} \right\rfloor \qquad (2)$$

where:

- n: Number of Words per Block
- W: Number of 32-bit words per frame
- C: Number of clock words per frame
- B: Number of blocks per column
- w: Modified Word's number in the frame
- i: Block number in column

Trojan Detector: Trojan Attributes

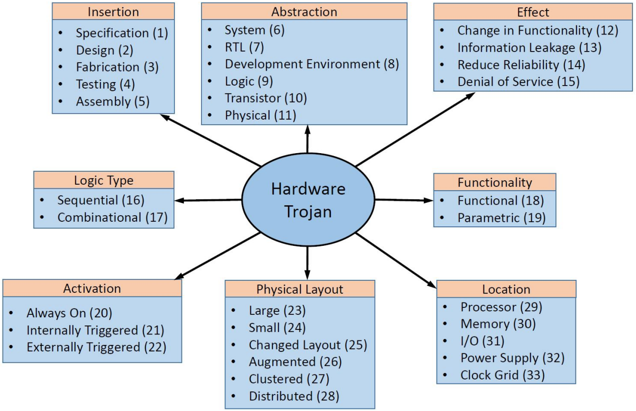

Figure: Hardware Trojan Taxonomy [Moein, 2016]

Relation Matrix

$\mathbf{R} =$

A	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
1	0	1	0	0	0	1	0	0	0	0	0																						
2	0	0	1	0	0	0	1	0	0	0	0																						
3	0	0	0	1	0	0	0	0	0	0	1																						
4	0	0	0	0	1	1	0	0	1	0	0																						
5	0	0	0	0	0	1	0	0	0	0	0																						
6						0	1	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0					
7						0	0	1	0	0	0	1	0	0	1	1	1	1	0	1	1	1	1	1	0	0	0	0					
8						0	0	0	1	0	0	1	0	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1					
9						0	0	0	0	1	0	1	0	0	1	1	1	1	0	1	1	1	0	0	0	0	0	0					
10						0	0	0	0	0	1	1	0	1	0	0	1	1	1	1	0	0	0	1	0	1	1	0					
11						0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	0	1	1	1	1	1	1					
12												0	0	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
13												0	0	0	0	0	1	0	1	1	0	1	0	1	0	1	1	1	1	1	1	1	1
14												0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1
15												0	0	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1
16												1	0	0	1	0	0	1	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1
17												1	1	0	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
18												1	0	0	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
19												0	1	1	0	0	0	0	0	1	0	0	0	1	0	1	0	1	0	0	1	1	1
20												1	1	1	1	0	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1
21												1	0	0	1	1	1	1	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1
22												1	1	0	1	1	1	1	0	0	0	0	0	1	0	1	1	0	1	1	1	1	1
23												1	0	0	1	1	1	1	0	1	1	0	0	0	1	1	1	1	1	1	0	0	0
24												1	1	1	1	0	1	1	1	1	1	1	0	0	0	1	1	0	1	1	1	1	1
25												1	0	0	1	1	1	1	0	1	0	0	1	0	0	1	1	0	1	1	1	1	1
26												1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
27												1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	0	0	1	1	1	1	1
28												1	1	1	1	1	1	1	1	1	1	0	1	0	0	1	0	0	1	1	1	1	1
29																																	
30																																	
31																																	
32																																	
33																																	

Figure: Relation Matrix R

Trojan Detector: User-Interface

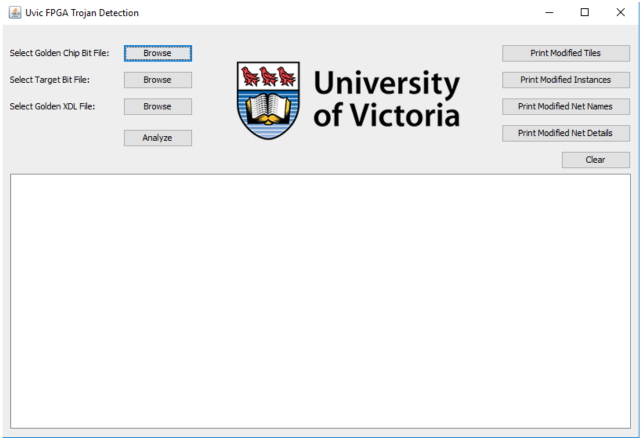

Figure: Hardware Trojan Detector User-Interface

HTS: Classification Tool

- Allows users to pick attributes observed in their trojan using a simple user-interface.
- Generates a directed graph visual.
- Generates a severity vector rating.
- Allows users to save entries to the database for future use.

Case Study: AES-T100

The Trojan leaks the secret key from a cryptographic chip running the AES algorithm through a covert channel. The channel adapts the concepts from spread spectrum communications (also known as Code-Division Multiple Access (CDMA)) to distribute the leakage of single bits over many clock cycles. The Trojan employs this method by using a pseudo-random number generator (PRNG) to create a CDMA code sequence, the PRNG initialized to a predefined value. The code sequence is then used to XOR modulate the secret information bits. The modulated sequence is forwarded to a leakage circuit (LC) to set up a covert CDMA channel in the power side-channel. The LC is realized by connecting eight identical flip-flop elements to the single output of the XOR gate to mimic a large capacitance. [Salmani, 2015]

- Attribute 3: Fabrication
- Attribute 4: Testing
- Attribute 5: Assembly
- Attribute 6: System
- Attribute 7: RTL
- Attribute 13: Information Leakage
- Attribute 16: Sequential
- Attribute 18: Functional
- Attribute 20: Always On
- Attribute 24: Large
- Attribute 26: Augmented
- Attribute 27: Distributed
- Attribute 29: Processor
- Attribute 30: Memory
- Attribute 31: IO
- Attribute 32: Power Supply
- Attribute 33: Clock Grid

Case Study: Visualization

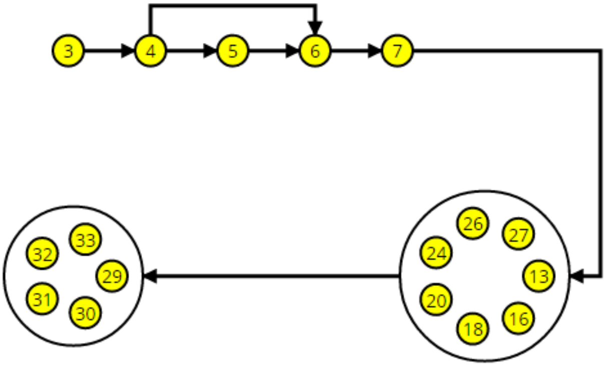

Figure: Directed Graph of the AES-T100 Benchmark

References

S. Moein and T. A. Gulliver and F. Gebali and A. Alkandari (2016)
A New Characterization of Hardware Trojans
IEEE Access v4, 2721 – 2731.

J. Zhang and G. Qu (2014)
A survey on security and trust of FPGA-based systems
Field-Programmable Technology 147 – 152.

H. Li and Q. Liu and J. Zhang and Y. Lyu (2015)
A Survey of Hardware Trojan Detection, Diagnosis and Prevention
2015 14th International Conference on Computer-Aided Design and Computer Graphics (CAD/Graphics) 173 – 180.

H. Salmani and M. Tehranipoor and R. Karri (2013)
On design vulnerability analysis and trust benchmarks development
2013 IEEE 31st International Conference on Computer Design (ICCD) 471 – 474.

The End