SIDE CHANNEL RISK EVALUATION AND MEASUREMENT (SCREAM)

A Major Qualifying Project Report

Submitted to the Faculty of

WORCESTER POLYTECHNIC INSTITUTE

By

Zachary Goddard

Nicholas LaJeunesse

Abstract

While unknown to most people, hardware implementation attacks provide a serious adversary for systems that contain sensitive data. Mission critical information can be extracted from a design with little effort from an attacker when they have access to the physical hardware. Thus designers try to mitigate this problem by using unique countermeasures styles. This work presents the first practical differential power analysis security evaluation of a countermeasure style called t-private logic. A PRESENT block cipher S-Box was implemented on a Virtex 5 FPGA as a reference platform. Both hardware and simulated power traces were collected. Statistical analyses were performed (CPA and Correlation enhanced collision attack) and our results revealed a first-order side channel attack vulnerability.

Table of Contents

Table of Figures

Table of Tables

I. Introduction

Encryption algorithms are widely used for secure computing and communications to protect sensitive data. If the encryption is broken, attackers can gain access to valuable and potentially mission-critical information. These standards for encryption are designed to be highly resistant to traditional cryptanalysis, but can be vulnerable when implemented in physical hardware.

Attackers try to exploit vulnerabilities that arise in hardware by using different implementation attacks. These attacks have three different classifications: invasive, semi-invasive and non-invasive. In an invasive attack, the adversary modifies the physical container by inserting a probe to measure voltages within the encryption core or performing an invasive fault analysis [1]. To counteract these types of attacks, meshes are placed over hardware packages to limit the effectiveness of the probe measurements that the attacker can collect fault tolerant designs have also been proposed. Semi-invasive attacks involve manipulating the outer layers of a device to inject faults and collect differential data through that channel [1]. Passive attacks focus on observing the device over time while it operates [1]. The major advantage of passive attacks is that they are inexpensive and simple to perform. They focus primarily on side-channels because they can leak valuable information and require little hardware specific knowledge or modification of the physical device to perform the attacks, making them difficult to detect.

The two most common side-channels are a device's electromagnetic radiation and its power consumption over time [2]. These non-invasive power side channel attacks are of the most interest because they have a low complexity, low equipment cost, and can be performed undetected. While the device operates, an oscilloscope collects traces measuring the power

consumption which varies with the data (plaintext) that is being encrypted. Statistical analyses have been developed that make guesses to reveal the key that is being used for encryption [2]. These attacks are made possible due to the dependence that the power consumption has on the data that is being operated on.

A number of techniques to limit the data dependent power consumption in hardware have been proposed for Very Large Scale Integration (VLSI) design. Typically for Application Specific Integrated Circuit (ASIC) designs, Complimentary Metal-Oxide Semiconductor (CMOS) logic is used as a standard library. CMOS logic is optimized for low power consumption, but not for security. Every time that an output from a CMOS gate changes state, power is consumed. This fact makes CMOS logic weak to the power side-channel. Currently, there are two overarching techniques regarding logic-level countermeasures. One proposal is based at the transistor level and uses non-standard cells that consume power regardless of the output logic values. The other proposal is to use logic-level countermeasures which are based upon using standard CMOS gates that break the data dependent power consumption [3][4][5]. Logic-level countermeasures that use a standard cell library are appealing to designers because they can be implemented using a standard design flow which saves design time and money.

Two main styles of logic-level countermeasures that have been developed are hiding and masking. The hiding countermeasures aim to make the power consumption consistent for each clock cycle. The most plausible hiding countermeasures rely on a dual-rail precharge (DRP) principle to ensure constant power consumption [3][4]. Masking countermeasures use extra random bits that change the internal logic values at each node to break the dependence [5][7][8].

While many different hiding and masking techniques have been proposed, these countermeasures have not been well characterized in terms of actual design tradeoffs. Differential Power Analysis (DPA) resistance, area, throughput and power consumption are all crucial for a designer to understand when creating a power side-channel analysis resistant ASIC. Some logic level countermeasures have not even been classified for DPA resistance. Designers need to have confidence that their countermeasures are effective otherwise their effort to secure their designs are in vain. One particular countermeasure style that has not been classified for DPA resistance is called t-private circuits. T-private circuits were originally designed to be resistant to invasive attacks but several publications suggest this style as a countermeasure against DPA attacks. The goal of this project is to implement the first encryption modules using this logic style on FPGA and to perform a security evaluation of this style. Different designs will be implemented on an evaluation board developed by National Institute of Advanced Industrial Science and Technology (AIST), upon which side channel attacks will be performed. The results from the study will document how to properly implement this logic style and the effectiveness of its DPA resistance to first order attacks.

II. Background

Evaluating the tradeoffs between different CMOS logic-level countermeasures in hardware design requires background knowledge in several different areas. This study focuses on power side channel attacks and therefore it is important to understand how a device leaks information through the power side channel itself, and how these attacks take advantage of this leakage. However at a base level, it is also necessary to comprehend the nuances of CMOS logic and the different techniques implemented to mitigate the dependency of power consumption on processed data.

A. Digital Hardware Design for ASIC and FPGA

The basic building block of digital design is the Complementary Metal-Oxide Semiconductor (CMOS) transistor pair. CMOS technology has been developed to maximize power efficiency as it consumes power when an output changes values/toggles. When the gate does not change values it consumes little power.

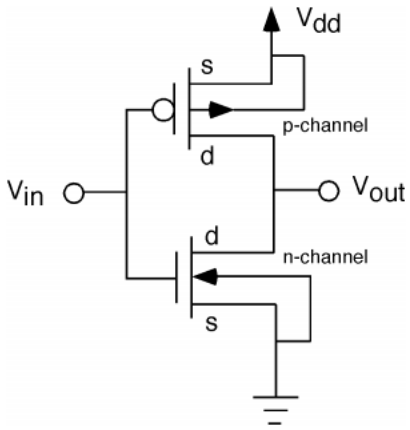

Figure 1: CMOS Logic Pair [6]

However, when the gate goes through its switching period there is a peak in current as the gate's capacitances are charging and discharging. As a result, the majority of a piece of

hardware's power consumption comes from the toggles that occur. Thus the link between the data moving through the logic gates and the power consumed is clear, making power side channel attacks viable. These standard cells are used in most Application Specific Integrated Circuit (ASIC) designs.

A Field Programmable Gate Array (FPGA) is an integrated circuit that is designed to be configured by the consumer. An FPGA is programmed using a hardware description language (HDL) and then acts as a typical IC after it is programmed. The basic building block of an FPGA is a memory element and a Look up Table (LUT). Both ASICs and FPGAs are common for digital hardware design, but there are tradeoffs associated with both. ASICs have a high Non-Recurring Engineering (NRE) cost and have one specific function. FPGAs are more versatile, but have a higher cost per unit in large volumes and higher power consumption. Also they can be less secure because there is less control over what is actually on board because EDA tools are used and the designer has little control of the bitstream that is being programmed into the FPGA.

B. Hamming Distance and Hamming Weight

Hamming Distance is a concept in information theory that takes two strings of equal length and calculates the number of differences between them. This model is based off of the work done by Richard Hamming on Hamming codes used for error detection. Hamming weight is a very similar concept, in that it is the hamming distance of a string and another string of the same length consisting of all zeroes. I can also be thought of as simply the number of 1's present in the string. The following python code produces the hamming distance of two strings:

```
def hamming_distance(str1, str2):
    return = sum(ch1 != ch2 for ch1, ch2 in zip(str1, str2))
```

C. PRESENT Block Cipher

Present is a lightweight block cipher that was proposed in 2007 to provide an encryption solution for small implementations such as RFID tags. The block length of this cipher is 64 bits and there are two key lengths available, 80 and 128 bits. The figure below shows the algorithm as a block diagram.

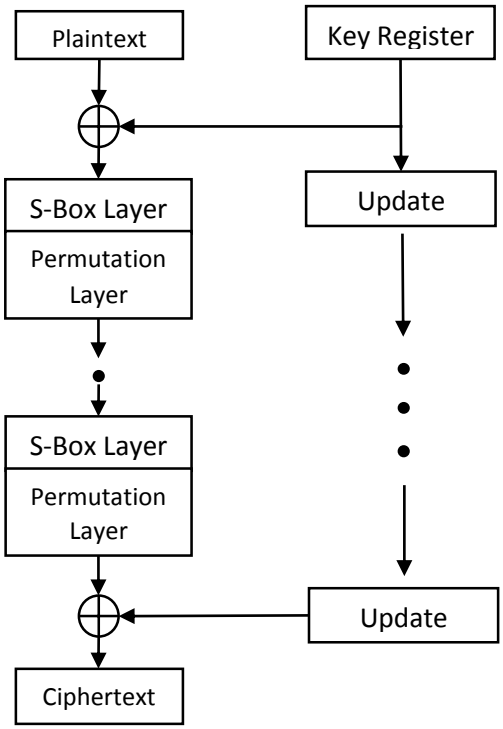

Figure 2. Block Diagram for PRESENT Cipher

At the beginning of each round a round key is introduced by performing the XOR with the plaintext. Then the next layer contains the non-linear substitution box (S-Box). The S-Box is a 4 bit non-linear transformation that is placed 16 times in parallel for 64 bits to be substituted each round. Below shows a table that demonstrates the hexadecimal values for the non-linear mapping for the PRESENT S-box.

Table 1. PRESENT S-Box

X	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S(x)	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Following the algorithm block diagram the operation after the S-Box is a permutation layer. This layer shifts the bits in the form that is shown below in decimal form:

Table 2. PRESENT Permutation Layer

I	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
P(i)	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
I	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
P(i)	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
I	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
P(i)	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
I	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
P(i)	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

The permutation layer is a linear bitwise mapping of bits that is fed into the next round of S-Boxes. Combining S-Boxes and Permutation layers in block ciphers is common to satisfy Shannon's diffusion and confusion properties.

The key schedule of PRESENT for a key of 80 bits is as follows:

1. During the round, the leftmost 64 bits of the 80 bit key register are used for the round;

2. The register is rotated 61 places to the left;
3. the leftmost 4 bits are passed through the S-Box;
4. the values for the register bits 19-15 are XORed with the value of the round counter

The design goal of the cipher is hardware efficiency, thus the S-Box is minimized to 4 bits to ensure a low amount of gates for that design. This cipher is based for moderate security and for being implemented in hardware.

D. DPA Countermeasures

While the use of different hardware implementations styles can provide more security, these alone are not sufficient to mitigate the threat of a power side channel attack. Academics have proposed many different countermeasure styles to break the correlation between the power consumption and the calculations that the hardware is performing. Unique logic cells which consume a constant amount of power and CMOS logic level designs have been proposed. However, logic level countermeasures are of a particular interest to a hardware designer because they can be implemented while using a standard CMOS design flow. The main styles that have been proposed can be classified into two different subsections: hiding and masking [9].

E. Hiding Countermeasures

Hiding countermeasures are based upon creating a constant power consumption per clock cycle to break the data dependent power consumption. To make this possible, dual rail precharge logic (DPL) is instantiated which splits each signal's value into the original value and the inverted value [3][5]. Every clock cycle a pre-charge signal is generated to ensure a single zero to one transition in each of the encoded nets occurs. Theoretically this creates a constant amount

of toggles per clock cycle, which in turn produces a constant amount of power consumption in the CMOS design [3]. However, implementing a DPL in hardware is not as simple as one would expect. When using these types of countermeasures there are strict balancing constraints must be met to ensure capacitive loads are equal between differential pairs [5]. This adds complexity to the design. In an ASIC design the designer has more control over the place and route so it is easier to make differential pairs. An FPGA is based upon a highly regular fabric and is more difficult to control using standard tools. Balancing the capacitive loads of these nets is important and there have been successful attacks against unbalanced DPLs.

F. Masking

Masking countermeasures rely on the use of a random mask bit to add entropy to the design. The mask bit is encoded with the data using an XOR obfuscating the data allowing only those who know the mask bit to retrieve the correct output values. This provides DPA resistance because the values have an equal likelihood that they are 0 or 1 regardless of the original values. It then becomes impossible for the CPA to guess the internal values and determine the power consumption based upon the inputs. This style of countermeasure is beneficial because there is less impact to the designer's flow for place and route. However, due to the need for the Pseudo-Random Number Generators (PRNGs) and number of gates used the amount of hardware used increases. Another problem with masking countermeasures is that glitches that occur propagate and that adds data dependent power consumption [4][8]. The specific type of masking countermeasure that is used in this study is called t-private circuits. This is further explained in the methods section.

G. Side-channel Attack Standard Evaluation Board

The Sasebo-GII, an acronym meaning Side-Channel Attack Standard Evaluation Board, was developed by the Research Center for Information Security (RCIS) and the National Institute of Advanced Industrial Science and Technology (AIST). These boards provide a platform to easily implement an encryption core and take power measurements. On board are two Xilinx FPGAs, the Spartan-3A (XC3S400A-4FTG256) control FPGA and the Virtex-5 (XC5VLX50-1FFG324) cryptographic FPGA. The control FPGA receives a 24MHz clock signal from an onboard oscillator. The block diagram in Fig. 3 depicts the layout of the Sasebo-GII. Communication to the board is handled via USB connection to a host PC.

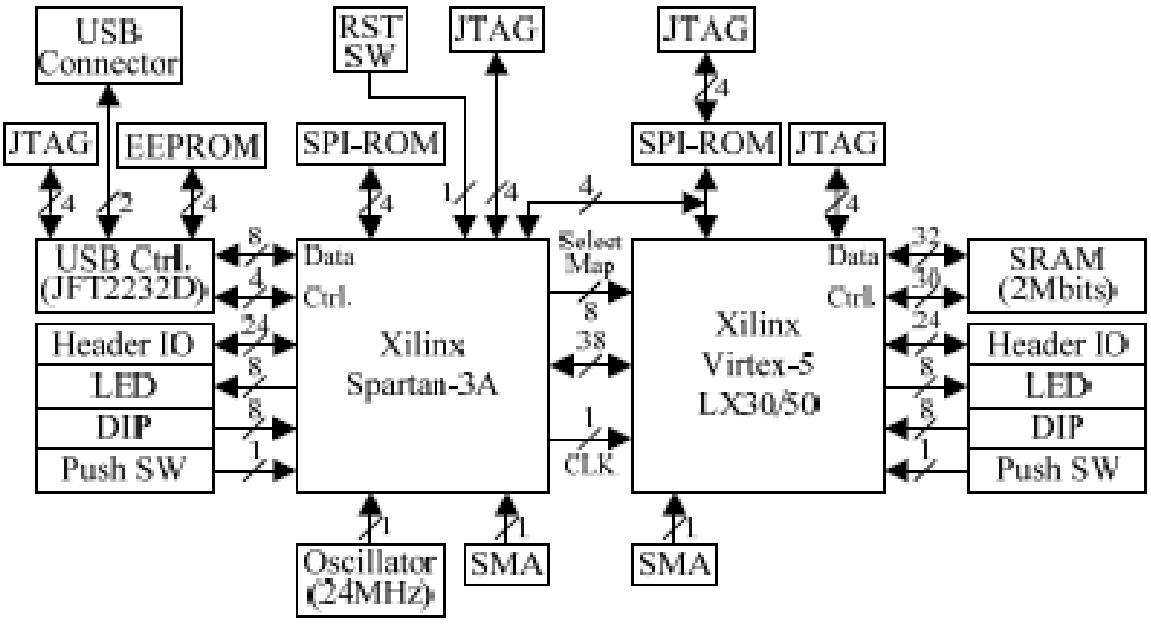

Figure 3 SASEBO GII Block Diagram

A picture of the SASEBO G-II is shown below. Onboard can be seen the cryptographic and control FPGA and SMA connectors used to measuring the voltage across the power line of the cryptographic FPGA.

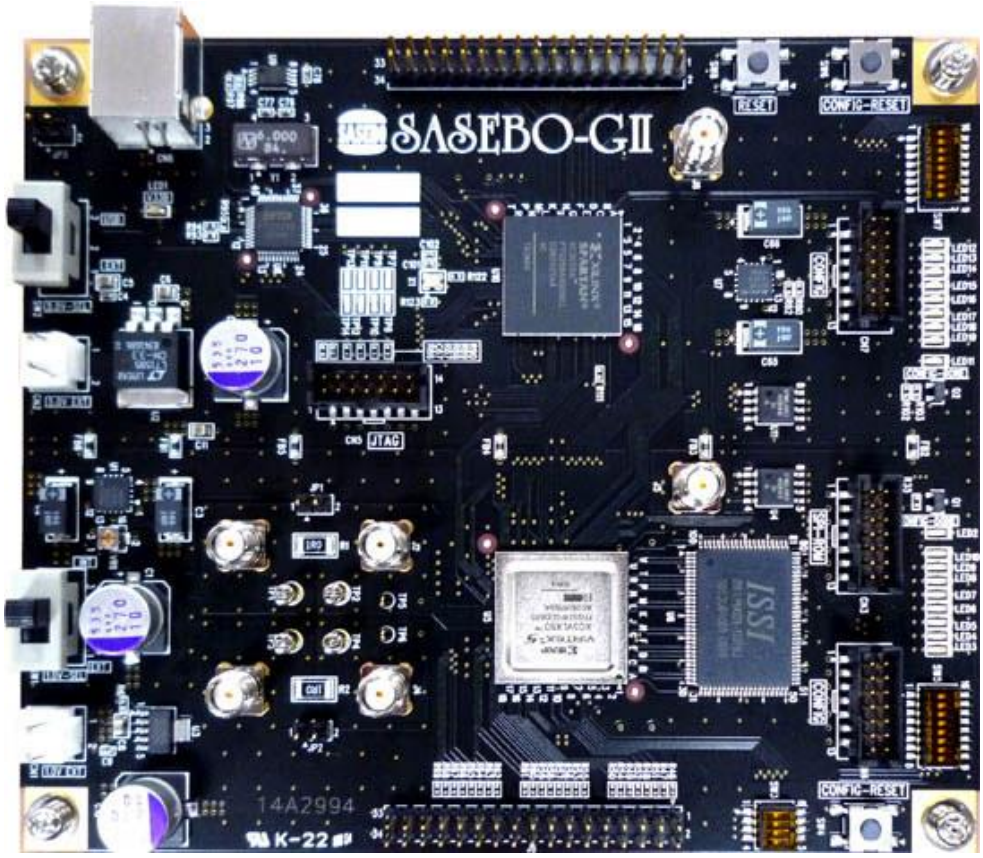

Figure 4. Picture of SASEBO GII [20]

III. Objectives

The goals of this project are as follows:

A. Perform a Prototype Analysis of a T-private Protected PRESENT S-Box Circuit

a) Develop Semi-Automated Implementation tool for T-private Logic Countermeasure Style
b) Create T-private designs
c) Collect simulated power consumption information for designs
d) Implement T-private S-box design on Virtex 5 FPGA
e) Perform physical power consumption data collection on SASEBO GII

Implementing the first documented T-Private PRESENT S-Box on FPGA provides proof of concept for future designs. Also creating a design flow for semi-automatic insertion of CMOS logic level DPA countermeasures is useful to designers to provide a cost-effective way to secure designs. The semi-automated flow allows the designer to have minimal knowledge of the specific countermeasure style as it translates a synthesized netlist to equivalent protected gate level designs.

B. Perform Security Evaluation of T-private logic countermeasure style

a) Perform Correlation Power Analysis
b) Perform Correlation Collision Attack

For this logic style to be useful to designers it must be characterized for DPA resistance. While it may be secure from invasive attacks i.e. probing, it must demonstrate DPA resistance for the logic style to be beneficial to designers concerned about Anti-Tamper qualifications. Understanding the effectiveness of this resistance provided is important for potential future designs.

IV. Methods

A. Design Methods

The main focus of this project was to collect results from attacks performed on the t-private S-box designs. Evaluating this protected logic style is of interest to designers because it is currently uncharacterized for DPA resistance. To create the designs and perform the attacks different steps were taken:

1. Implementing T-private Logic Style

Private circuits were proposed in [15] in 2003 as a countermeasure to invasive attacks. This work states that probing attacks are a more powerful adversary than DPA because the attacker can "read your brain" by directly observing logic values of internal nets. T-private circuits are a masking countermeasure that splits the values into T+1 shares. The value of T determines the security level of the circuit, i.e. the number of internal probes per clock cycle the circuit is resistant to. A circuit designed with T = 1 will be resistant to 1 probe per clock cycle. For t =1 the inputs to a circuit are split into two shares, the original input XORed with a mask value and the mask value itself. The original work described how to make t-private encoders, decoders, PRNGs, and gates, or gates, xor gates and inverters.

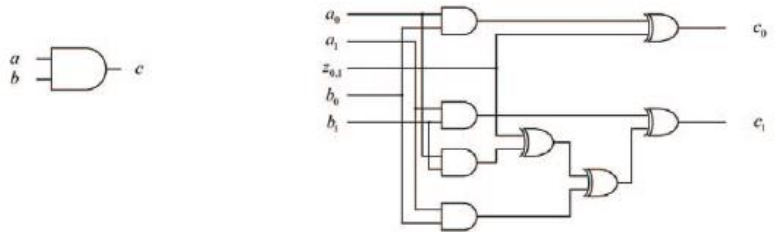

Figure 5. And gate and t-Private equivalent[8]

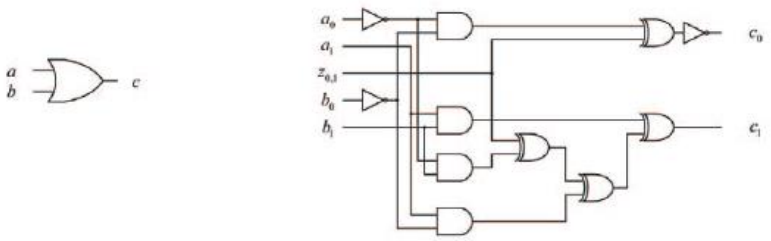

Figure 6. Or gate and t-Private equivalent[8]

AND gate: $c_i = a_i b_i \oplus \sum_{j \neq i} z_{ij}$ where c is of the t+1 outputs. Summation in GF2.

OR gate: $c_i = a_i b_i \oplus \sum_{j \neq i} z_{ij}$ where c is of the t+1 outputs and for all nets of one i values is connected to an inverter. Summation in GF2.

XOR gate: $c_i = a_i \oplus b_i$ where c is one of the t+1 outputs

A later work [8] describes more optimized forms of the gates. The optimized and gate is described below:

AND Gate when t is an odd number:

$$c_i = (a_0 b_i \oplus z_{i \bmod \frac{t+1}{2}}) \sum_{j=1}^{t} a_j b_{(j+1) \bmod t+1}$$;summation in GF2

For i = 0,1…..t

When z is a random bit

AND Gate when t is an even number:

$$c_i = (a_0 b_i \oplus z_{i \bmod \frac{t+2}{2}}) \sum_{j=1}^{t} a_j b_{(j+1) \bmod t+1}$$;summation in GF2

For i = 0,1…..t

$$c_t = (a_0 b_t \oplus z_{t \bmod \frac{t+2}{2}} \oplus z_{t+1 \bmod \frac{t+2}{2}}) \sum_{j=1}^{t} a_j b_{(j+1) \bmod t+1}$$;summation in GF2

When z is a random bit

It is evident that there is an area increase with t-private circuits. To make sure that the size doesn't grow too large, the XOR sum of products can be used to find a space efficient equivalent. The reason that this works is because the XOR gates have only an area increase of t+1 times. This technique is a recommendation for reducing area. Another source of overhead in this countermeasure style is the required entropy. All T-private circuits need t*n masks where n is the number of original inputs. Also required is mask bits for internal states of the combinational logic cloud. T-private and, or, nand and nor gates all require t internal mask values to further obfuscate the internal nets' values.

In further studies T-private circuits are proposed as a DPA resistant circuit. The security proofs in these papers describe how the circuits can be protected against internal probing attacks, however they do not describe the security against DPA.

2. Creating a Tool for Semi-Automatic translation of unprotected RTL to T-private logic RTL

This study was an exploration to not only the effectiveness of the t-private logic style as a DPA countermeasure, but also to see how the actual feasibility of implementing the style. Hardware designers have a tried and true methodology for creating a design. A basic form for the standard design flow is pictured below.

Figure 7. Simplified Design Flow

Ideally security would have minimal impact on the hardware designers flow. Added complexity and knowledge required for a logic level countermeasure is high. Also it is not always clear how to actually implement the countermeasures. Thus a need for a tool is evident. To save time and money the countermeasures would be inserted as early as possible during the design phase. T-private circuits are a good candidate for this because T-private logic works by having gate equivalent circuits. The standard combinational logic designs can be translated to T-private logic by replacing the original CMOS gate with the T-private gate. Also the rest of the design flow is undisturbed because once the equivalents are in place they are treated just treated as CMOS gates. The work that was completed aimed to fill this niche and create a design script that would encompass the t-private logic style. The design script that was developed targeted the part in the flow between RTL and synthesis. Below is a figure showing the design step called "t-private logic synthesis."

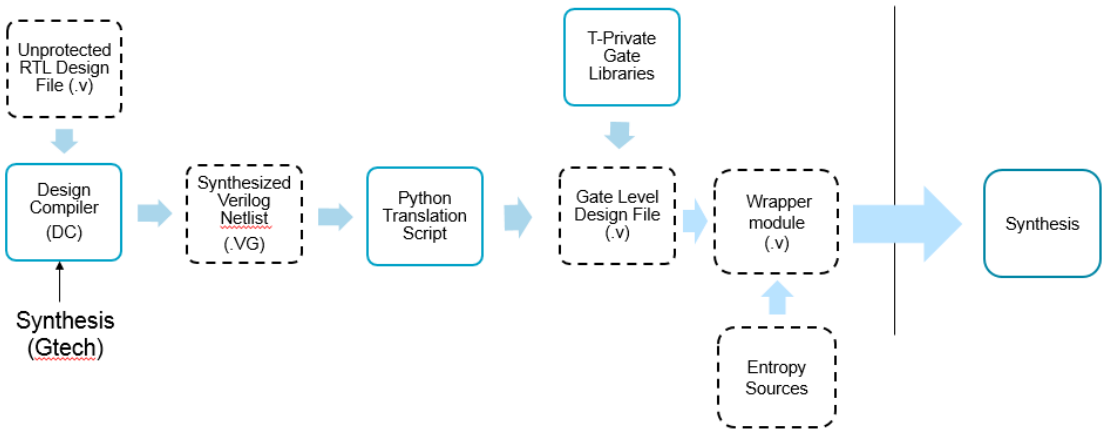

Figure 8. T-private Logic Synthesis Design Flow

This design step takes an RTL design file and synthesizes it using an EDA tool such as Synopsys Design Compiler (DC). DC targeted a hardware unspecific library called GTech. The

GTech library was chosen because it has a very regular form for all of its gates. Also the gates were of their simplest form (i.e. AND2, XOR2 etc.) which made the construction of the t-private equivalent library easy. The output synthesized netlist was then parsed by a python tool that was developed to examine the lines of the design file and create a new output file with its equivalent t-private gate level design Verilog file. The arguments the script has are the path to the original unprotected netlist, the output file name and the security level. The security level corresponds to the t value that was desired. While the script has functionality for all values of t, there was only a (T=1)-private gate primitive library developed during the project. The library contains modules of different basic gates as described in [8]; for example: AND2, OR2, XOR2 and inverters are all located in this library.

The t-private logic style requires all nets in the synthesized design to be split into t+1 shares and requires the addition of entropy (masking) bits. The new design file that is output from the script requires that the new module inputs are encoded properly as described in [8]. For t =1 the design needs to have all of the original inputs XORed with their respective mask bits and the mask bits input to the designs. Also another addition that is required is a 'gate mask' value. AND, OR, NAND and NOR gates all require t internal mask bits. This additional overhead is what gives t-private its security. When breaking up the nets into t+1 shares with equal probability of being either 0 or 1, t probes reveal no information about what the internal operations are.

Using this design flow and script, both a T-private Composite field AES S-Box and a T-private PRESENT S-Box were generated. These designs were easily translated to t-private using this design flow because they are both purely combinational circuits. The necessary individual focus on each design was minimal, however the netlist output from DC had an effect on the overall size of the design. This is further explained in section V.a. If this design flow was to be

applied to a full cipher design sequential elements would be added to the library. The design of sequential elements is described in the original private circuit paper. This was not completed in this study because the overall work focused on combinational elements. Also the addition of PRNGs is something that a designer would have to focus on. For example, since the tool developed would translate the whole design, the overhead associated with creating 128 mask bits for the 128 inputs of AES might prove too costly. The application of t-private circuits as a countermeasure to select nets is described in [17], this might prove to be a more effective solution.

3. Implementing our designs on the SASEBO GII

Using an interface developed by AIST, the protected designs were implemented on the Virtex 5 contained on the SASEBO GII operating at 3 MHz. For measureable power capture 8 PRESENT S-Boxes were placed in parallel in the encryption core to amplify the voltage drop measured on the board. To create an implementation that is independent of the strength of PRNGs created on the FPGA the interface allowed for both plaintexts and mask bits to be written to the encryption core. This design was synthesized, mapped and placed and route using ISE Design Suite 14.6. The implementation targeted the XCV5VLX50-2FF324 contained on the GII and synthesized the modules as a hierarchy to contain the optimization constraints for t-private logic on FPGA.

The Sasebo-GII comes paired with Verilog RTL designs to implement a full AES encryption core. We modified the RTL to create a top level design that allowed us to easily implement different S-Box designs. This data flow for this design is depicted in Fig. 9.

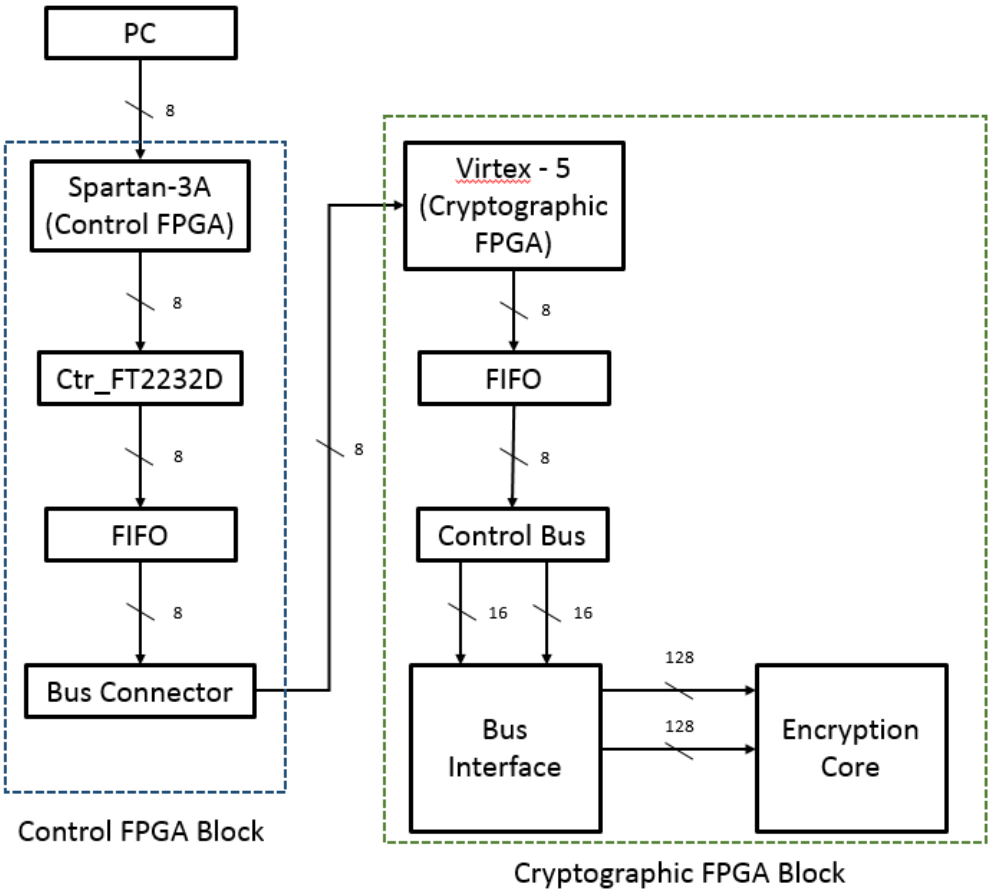

Figure 9. Block Diagram of interface created by AIST for GII

The process starts with the host PC sending plaintext and key data which are packed by the FTD2XX.dll driver to the board via USB. The data is then read into the Control FPGA block by the Ctr_FT2232D module which is then passed through a FIFO to the bus connector. The bus connector passes the data to the Cryptographic FPGA block which is subsequently written to the FIFO and then the control bus. The control bus packs the data into two 16 bit outputs, the input data itself and the corresponding address. The bus interface then uses the address value to pack the data into the appropriate location in either the 128 bit key or plaintext data buses. This address mapping can be seen in Table 3. The 128 bit key and plaintext buses are then read in by

the Encryption core module to be encrypted. This process is reversed to write the encrypted output back to the host PC.

Table 3 Hardware Address Values for SASEBO G-II

ADDR	Data	Description
0x0002	ctrl[2:0]	Control stores key and data ready signals
0x000C	blk_encdec	Sets block encrypt/decrypt functionality
.	Empty	
0x0100	Key in[127:112]	Key Input for encryption core
0x0102	Key in[111:96]	
0x0104	Key in[95:80]	
0x0106	Key in[79:64]	
0x0108	Key in[63:48]	
0x010A	Key in[47:32]	
0x010C	Key in[31:16]	
0x010E	Key in[15:0]	
.	Empty	
0x0140	Data in[127:112]	Plaintext data input for encryption core
0x0142	Data in[111:96]	
0x0144	Data in[95:80]	
0x0146	Data in[79:64]	
0x0148	Data in[63:48]	
0x014A	Data in[47:32]	
0x014C	Data in[31:16]	
0x014E	Data in[15:0]	
.	Empty	
0x0180	Data out[127:112]	Encrypted data output from encryption core
0x0182	Data out[111:96]	
0x0184	Data out[95:80]	
0x0186	Data out[79:64]	
0x0188	Data out[63:48]	
0x018A	Data out[47:32]	
0x018C	Data out[31:16]	
0x018E	Data out[15:0]	
.	Empty	
0xFFFC	0x4522	

4. Simulating Power Traces of our Designs

After the designs were implemented on the GII a post place and route simulation was completed of the modules containing the S-boxes. While the simulation ran, internal values of nets in the design were written to a VCD file. From this file, toggle data over time was collected for the simulations. Since the number of toggles in a CMOS logic circuit is proportional to the dynamic power consumption, this data was used to create a simulated power profile for each input state transition. The time stamped toggle data was low-pass filtered and had Gaussian noise added to it. This was done to create a theoretical power trace similar to what was expected from the physical measurements. Two hundred fifty nine thousand traces were collected to exhaust possible states of the combinational logic.

5. Retrieving power traces from the SASEBO-GII

The entire trace capture setup was encapsulated as a python package and relevant scripts. The first step was to establish communication with the board. This involved creating a basic wrapper around the FTD2xx driver for USB communication, implementing basic functions such as read and write. The next level up was a python module to interface with the SASEBO-GII itself. This involved creating an instance of the board to write commands to and initializing it. It also provides the methods necessary to perform an encryption using the board, including functions that set the key, write the plaintext, and read back the resulting ciphertext.

Another python interface module was also created to communicate and set up the Tektronix MDO4104B-6 oscilloscope. This interface contains numerous methods that allow for complete customization of the oscilloscope settings. However, to streamline the process for our

captures, an initialization method was created which configured a list specific settings from a .json file.

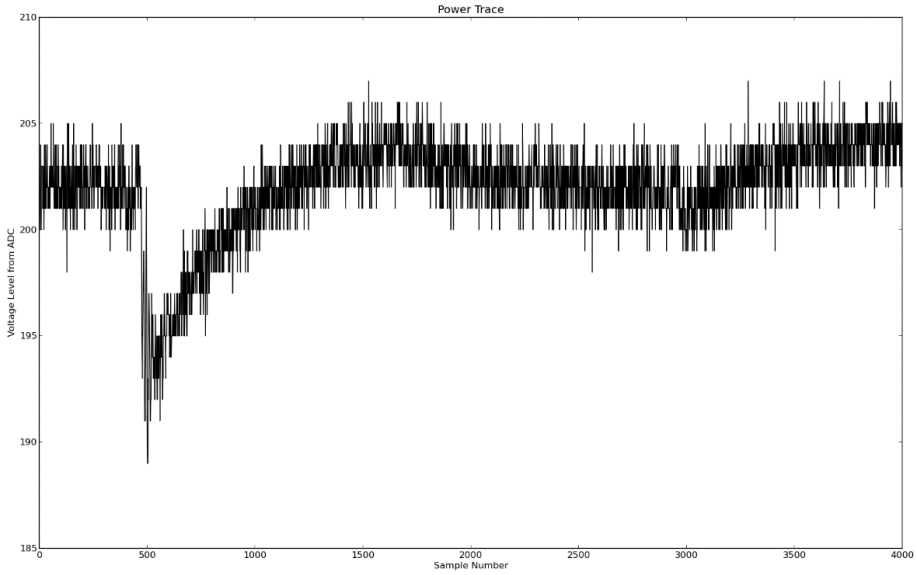

Figure 10. An example power trace from oscilloscope of PRESENT S-Box Implementation

At the top level of the capture is a python script that controls the entire process. The script first configures the scope to prepare it for the current capture. Next the board is initialized and prepared for the encryption process, while the test vector file is opened. The test vector contains a list of plaintexts formatted appropriately for reading to the board. They are listed such that the transition space of all possible inputs from 0-F is covered. The next phase of collection is the encryption loop, where a single line of plaintext from the test vector is written to the board. During the encryption process, a trigger signal is sent from the board to the scope, the scope then saves the power trace being measured. The script reads back the resulting ciphertext from the board, and retrieves the saved waveform from the scope. In Fig. 10 above, you can see what an example trace looks like. This is not the ideal capture due to the bit flickering caused by the low

ADC resolution. However, our assumption is that this does not end up being a problem in our analysis because of the averaging performed on the traces.

Finally, the script packs each trace into an HDF5 file with the appropriate metadata and moves onto the next plaintext. The oscilloscope was connected via SMA-BNC cable to the 1Ω sense resister on the Sasebo-GII. Measurements were taken at a sample rate of 5 GS/s with 8 bits of resolution. The waveform measurement was taken on channel 1 set to a vertical scale of 10e-3 V/div at full bandwidth and DC coupling. Fig. 9 shows a sample trace capture.

B. Performing Attacks on Simulated and Hardware Power Traces

In this section we describe the performed analysis of the t-private reference implementation. We performed two popular attacks, a classical correlation power analysis (CPA) and the correlation-enhanced collision attack (CCA). Both attacks were applied on a simulated leakage based on the toggle count model of the FPGA synthesis output as well as real measurements of the reference design on the SASEBO-GII.

1. Correlation Power Analysis (CPA)

CPA is one of the most popular and simple forms of Differential Power Analysis. It correlates the observed leakage to a hypothetical key dependent leakage. The correlation coefficient is a value used to distinguish or detect the correct key. Our leakage model is the Hamming distance of two consecutive s-box outputs, which is an approximation of the leakage of the output register of the S-Box. This is because the hamming distance is representative of the number of toggles that occur between the previous state to the current state. Toggles or

switching, can be directly related to the power consumption of the circuit, thus making the hamming distance an appropriate model for leakage present during the encryption.

Both attacks were implemented using python modules as a part of the flow step design created by last year's MQP team [18]. The CPA step takes a list of selector functions as an input parameter which is then used to create the leakage models. In this case we used a selector function that calculates the hamming distance between the current and previous ciphertexts. The selector uses the trace's metadata for the calculation. The selector also creates models for each incorrect key guess to be correlated against. This is done in a for-loop which XOR's both the plaintext and previous plaintext inputs for each trace with an offset from 0-15. This new value is then passed through a Present S-Box function to produce the new ciphertext and previous ciphertext outputs. These are then used to create the new hamming distance model for the incorrect key guess.

The correlation was calculated using the following equation:

$$r_{xl} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(l_i - \bar{l})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(l_i - \bar{l})^2}}$$

Where x_i represents the i^{th} sample of a trace and $\bar{x}$ is the sample mean. The leakage model sample is represented by l_i and the sample mean $\bar{l}$. The correlation r_{xl} shows the linear dependence between the power and the leakage model. As the value approaches zero, the relationship decreases, however as it approaches -1,1 the correlation increases.

2. Correlation Collision Attack (CCA)

The Correlation Collision Attack is a first order attack introduced by [15] in which the attacker uses collisions to exploit leakage without knowledge of a hypothetical power model. For a collection of power traces T_i where plaintexts $P_i = \{p_j^i\}_{j=0}^{15}$ are encrypted, the leakage of the S-Box is defined as $s_j^i = S(p_j^i \oplus k_j)$. For a collision between two of the same S-Boxes, the outputs will be equal and thus the inputs are also equal such that: $p_a^{i_1} \oplus k_a = p_b^{i_2} \oplus k_b$. Therefore the input difference is:

$$\Delta_{a,b} = p_a^{i_1} \oplus p_b^{i_2} = k_a \oplus k_b$$

The linear relation between two key bytes reduces the entropy significantly allowing the key to be recovered through trial and error. However this method, known as a linear collision attack is thwarted via the use of masks. To strengthen it against this defense, all traces are averaged by input value to the S-Box, where M_j^α is the average of all traces T_i where input plaintext $p_j^i = \alpha$. A collision between two S-Boxes occurs when $p_a = \alpha$ and $p_a = \alpha \oplus \Delta_{a,b}$. Thus to find the difference $\Delta_{a,b}$ it is necessary to detect all collisions for all inputs at the same time. This is done by correlating the average power consumption for one S-Box's input, M_a^α, to the averaged traces of the second S-Box, $M_b^{\alpha \oplus \Delta_{a,b}}$. The correct value of $\Delta_{a,b}$ is found with:

$$\arg\max \rho_{\Delta a,b} \left(M_a^\alpha, M_b^{\alpha \oplus \Delta_{a,b}} \right)$$

The correlation coefficient ρ is calculated for every point in time and represents the correct distance $\Delta_{a,b}$ when it is maximum.

The CCA step simply takes in the appropriate HDF5 file set and performs the analysis utilizing the metadata. Then the traces from the collections are separated into two halves. Each half is then sorted by the input plaintext to the S-Box and averaged to create two sets of 16 average traces. The correlation between the two sets is calculated for offsets 0-15 representing the key guesses, where an offset of zero is the correct key guess in this case. The correlation is calculated using Pearson's correlation coefficient equation where P is the relation between the correlation coefficient matrix and C is the covariance matrix:

$$P_{ij} = \frac{C_{ij}}{\sqrt{C_{ii} * C_{jj}}}$$

Both attacks produce an HDF5 file of the results, where each selector is represented as a trace of correlation coefficient values.

V. Results

A. Design Results

One point of concern for designers when creating a protected design is the increase in area. The PRESENT S-Box was chosen to be a 4 bit function to minimize the complexity and size in hardware. Post synthesis reports for the t-private implementation show that the design synthesizes and maps to 96 Slice LUTs. The target Virtex 5 device has 28,800 slices, so for a full implementation of PRESENT which requires 16 S-Boxes the percent resource usage would be around 5% for the largest part of the design. While this seems like a small amount of area, the look up table implementation for an unprotected design requires only 4 slice LUTs. However, the PRESENT S-Box design was not in the ESOP form. The majority of the t-private gates in the design were OR and AND gates and gates these have the highest amount of area increase. The present design contained 27 t-private AND gates and 25 OR gates. This overhead seems ridiculous while comparing the two designs, however t-private can be more efficient, as proposed in [17] the ESOP form for other designs performs better in terms of area.

The optimized ESOP t-private Composite field implementation AES S-Box synthesized to 175 Slice LUTs and the unprotected implementation synthesized to 50 LUTs. The contributing factor for this is the AES S-Box contained 36 AND gates and 123 XOR gates. The t-private XOR gates only have a size increase of 2 times (t+1) and that contributes to the more efficient expansion/ size usage for protection. The designs size after synthesis is shown in a table for comparison. The WDDL design was one created by AIST and was synthesized using the same constraints as the t-private design.

Table 4. Size Comparison for Protected Logic Styles Post Synthesis

	AES Composite Field	T-Private AES Composite Field	WDDL AES Composite Field
Number of LUTs	50	175	338
Relative Size	1.0	3.5	6.76

Table 5. Size Comparison for Protected Logic Styles Post Map

	AES Composite Field	T-Private AES Composite Field	WDDL AES Composite Field
Number of Slices	43	148	205
Relative Size	1.0	3.44	4.77

B. Simulation and Hardware Power Traces Attack Results

Using the results from the attacks on simulation and hardware implementation data the effectiveness of t=1 private circuits against several first order attacks was determined. The number of traces required for Correlation Collision Attacks to escape the incorrect key guess noise floor was determined and the correlation values determined from the CPA were compared as well.

Evaluation was performed in three scenarios: once with the input mask set to zero. In the second case the input mask is chosen uniformly at random. As done in [19] we use externally

applied masking to ensure that no leakage is due to input and output logic. The third case is simply an evaluation on an unprotected PRESENT S-Box as a reference.

The CPA was performed on all 259,000 traces for each case. Fig. 10 depicts the CPA of the unprotected Present S-Box. It is clear that the correct key hypothesis is easily distinguishable from the others. However in Figs. 11 & 12, both the t-private and zero mask t-private scenarios show resistance to the classical CPA as the correct key remains hidden among the incorrect hypotheses. However it is important to note that while still hidden, the zero mask CPA results have much higher correlation values than their masked counterparts.

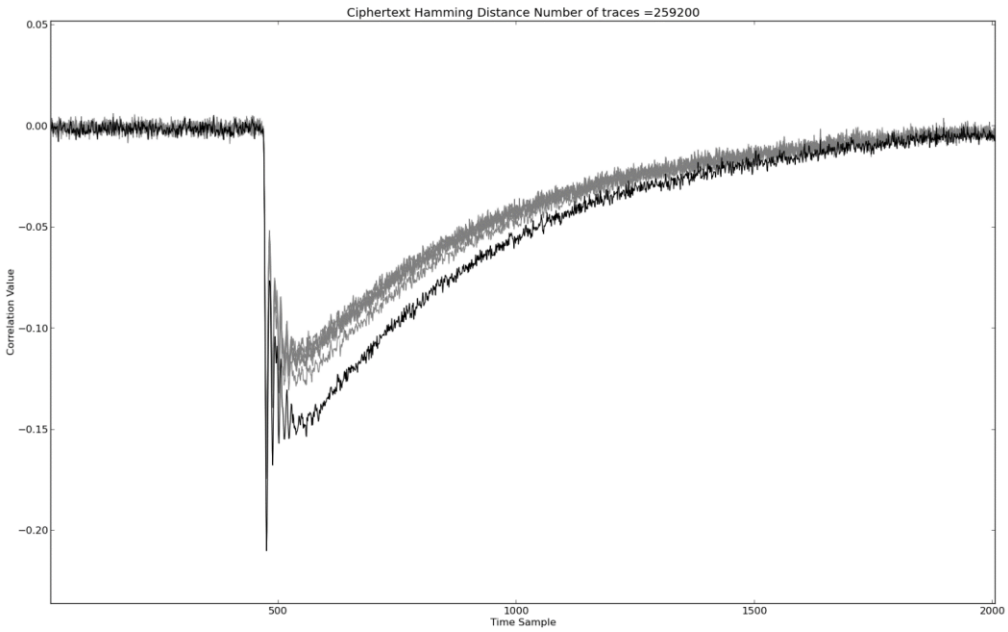

Figure 11. CPA of Unprotected Present S-Box for 259,000 traces

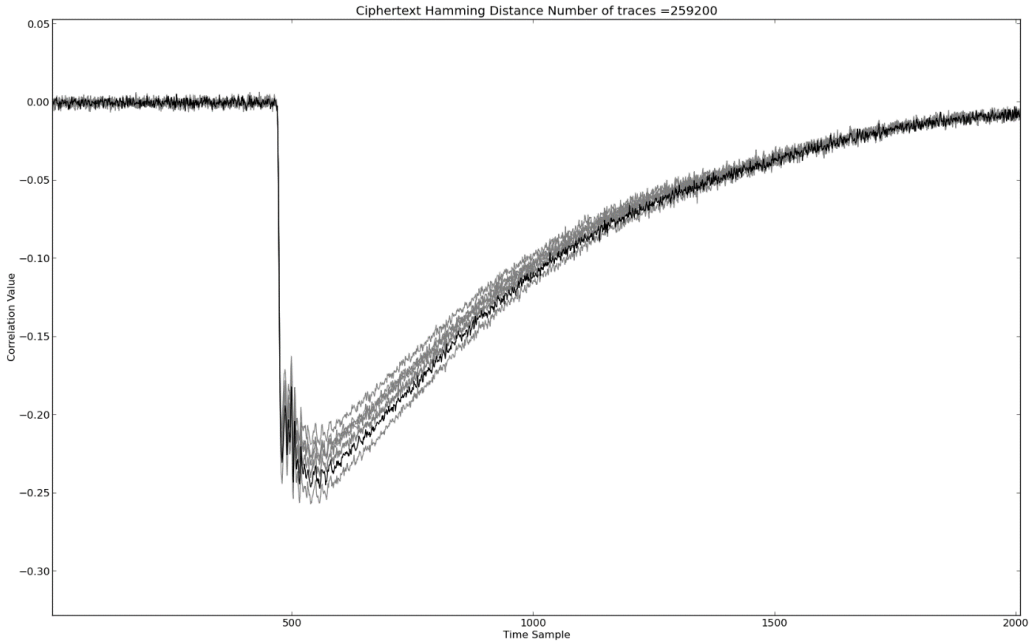

Figure 12. CPA of Zero-Mask t-Private S-Box for 259,000 traces

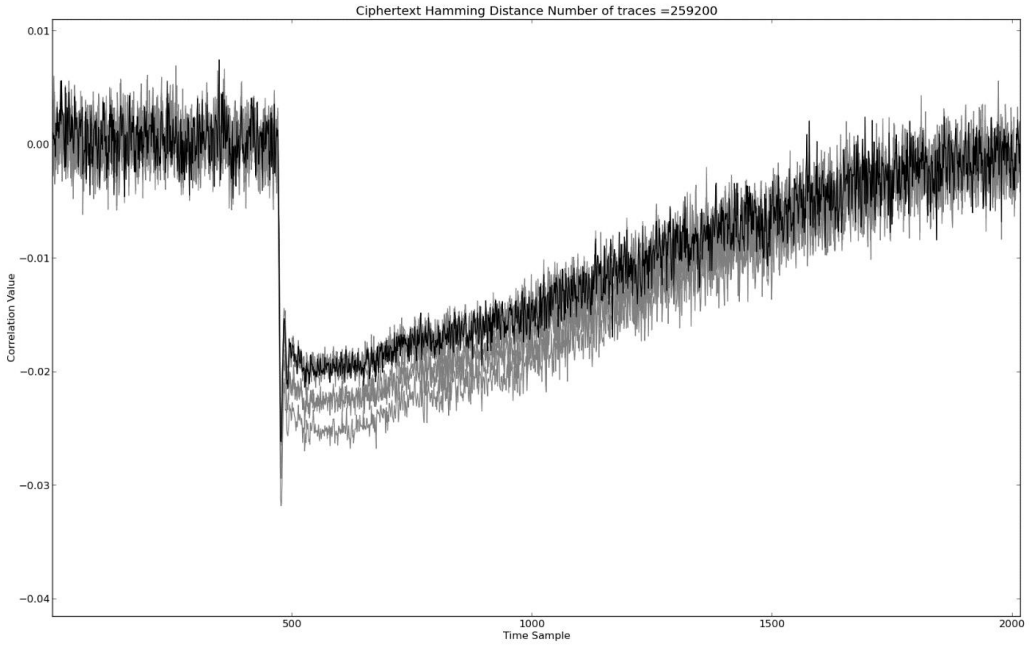

Figure 13. CPA of t-Private S-Box for 259,000 traces

The CCA was performed across all 259,000 traces for each scenario. However, to determine at what point the correct key left the noise floor and became distinguishable, it was necessary to perform the analysis across subsets of increasing increments out of the collections. For example, CCA data was taken at intervals of 256, 512, 1024, 2048, etc. traces. This allowed us to create the graphs seen in the figures that follow. Each figure depicts the correlation value of all 16 key hypotheses at a single point on the power trace versus the number of traces analyzed. Thus looking at Figures 13 & 14, we can see that leakage begins to become evident at about 5,000 traces for the zero mask case and 130,000 traces for the masked case. While Fig. 15 shows the entire CCA analysis trace performed at 259,000 traces. The correct key shows clear separation above all other guesses.

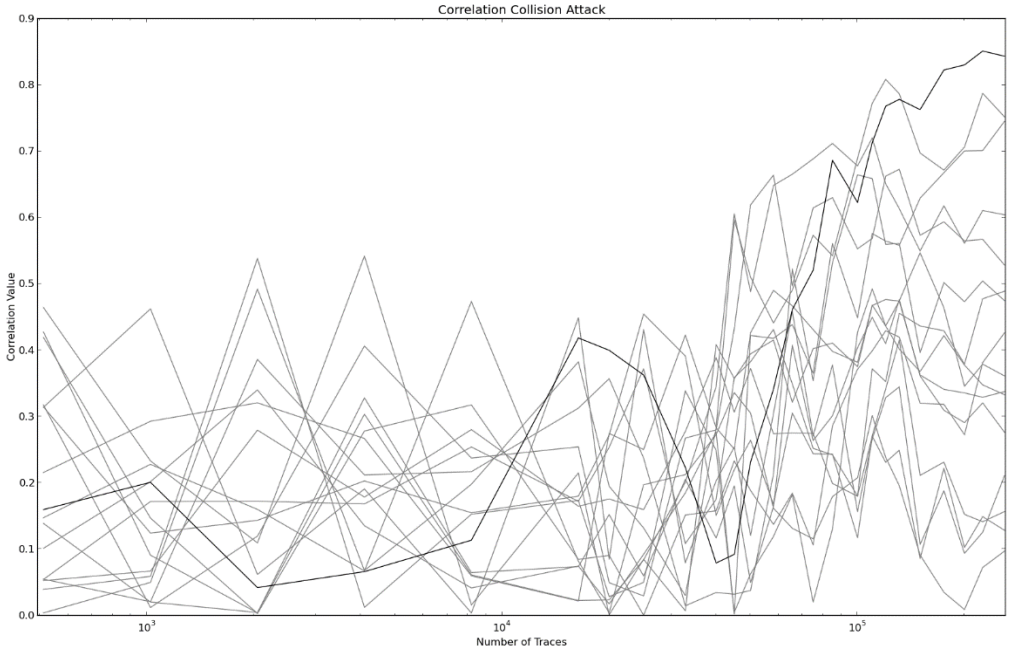

Figure 14. CCA Number of traces vs. Correlation for t-Private S-Box

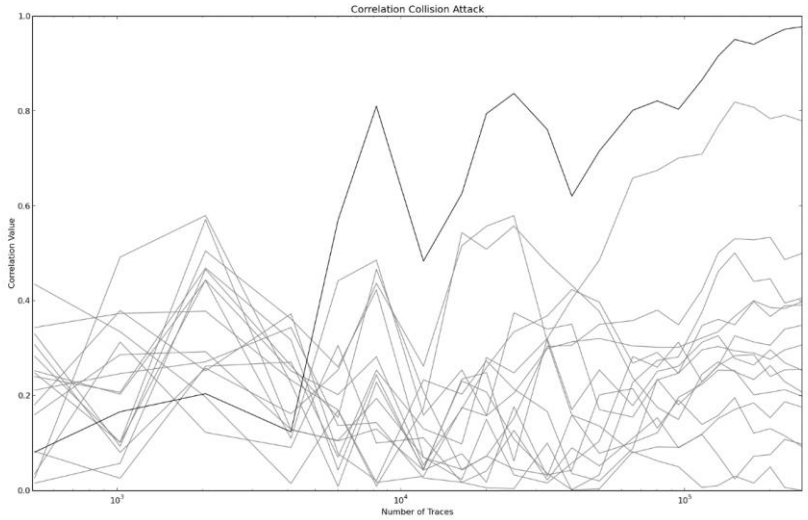

Figure 15. CCA Number of traces vs. Correlation for Zero-Mask t-Private S-Box

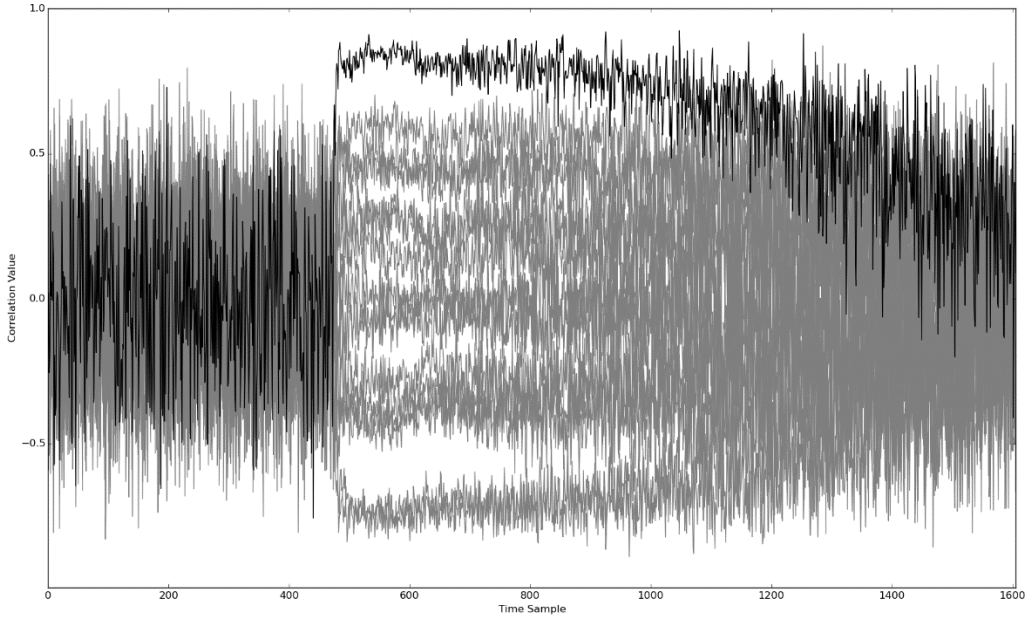

Figure 16. CCA of t-private S-Box for 259,000 traces

The simulation data was put through the same attacks as the above hardware data. First shown is CPA results for the three designs: unprotected PRESENT, T-Private present with

constant mask bits =0 and T-Private with 5 varying mask bits per clock cycle. The solid black line represents correct key guesses in all plots below. The leakage model for the CPA was the hamming distance between S-Box outputs. In the attacks performed on the T-private designs the decoded 4 bit output values for the S-box were used for previous and current ciphertext and 259200 power traces were collected.

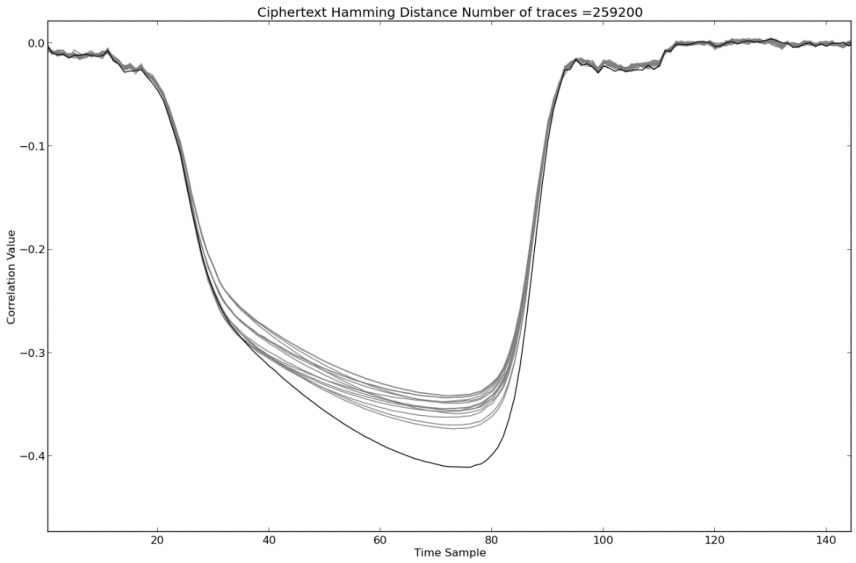

Figure 17. CPA Results for Unprotected PRESENT S-Box

As is shown in the figure above, the simulation data reveals a leakage for the unprotected PRESENT design which is expected.

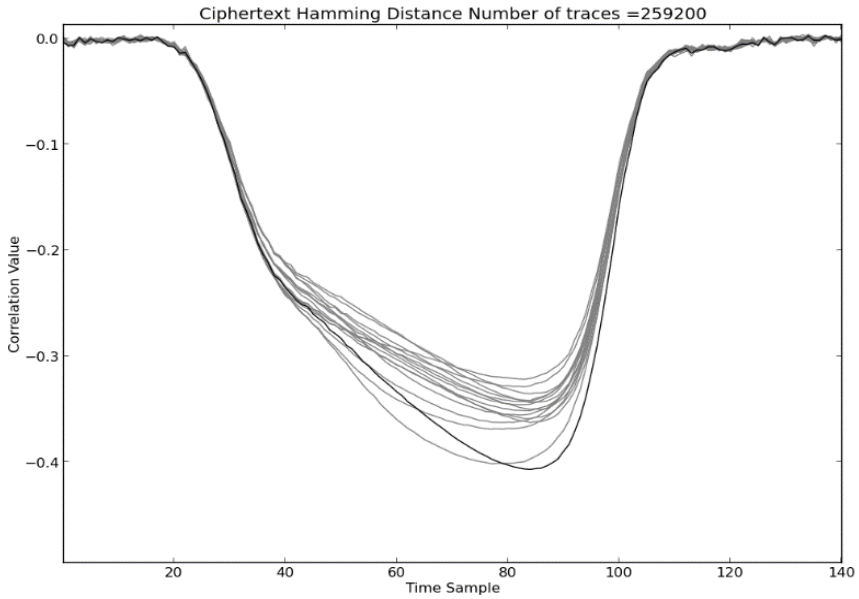

Figure 18. CPA Results for Zero-Masked Present S-Box

Figure 19. CPA Results for T-Private PRESENT S-Box Design with Zero Masks

In the figure for the Zero-masked design the hamming distance between decoded ciphertexts was only able to produce a small leakage which escapes the noise floor for about one tenth of the clock cycle where the power data is collected. Below is shown the figure where the hamming distance selector was used upon the masked outputs of the zero-mask S-Box.

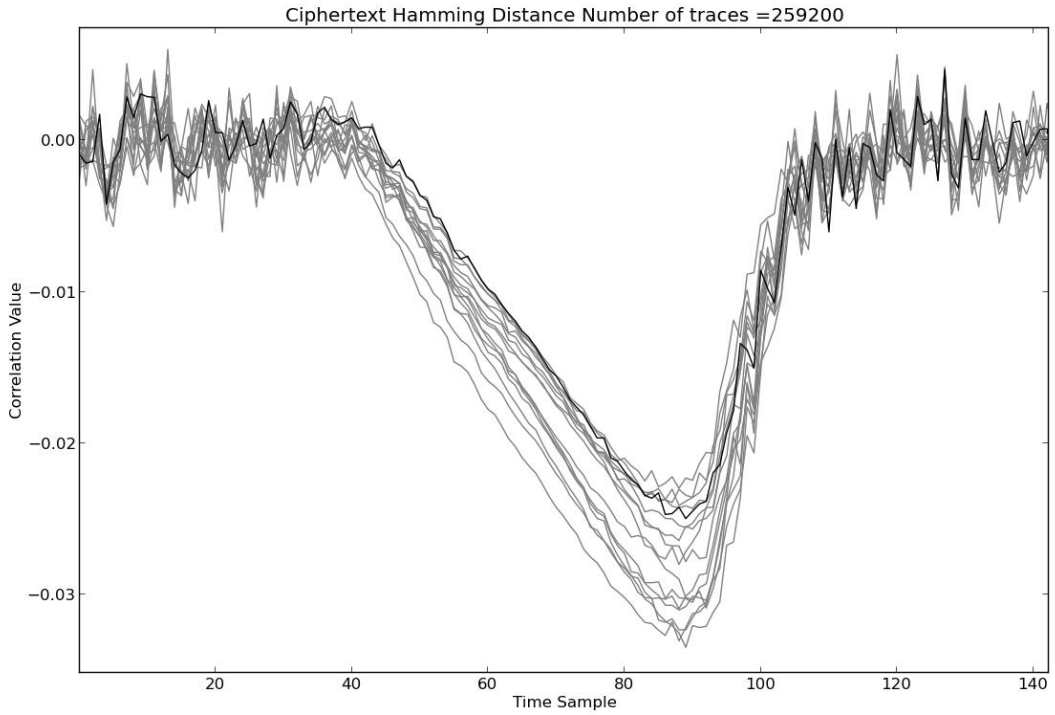

Figure 20. CPA Results for T-Private PRESENT Design

In the CPA for the T-private fully protecteed design the correlation was not able to esacpe the noise floor. This design was protecteed against this leakage model for 259200 traces and lowers the correlation peak correlation by about a factor of 10 compared to the zero mask values.

The next figures show the results for the correlation collision attacks on the 3 designs. Each figure is scaled on the x axis by a logarithmic scale representing the number of traces and the y axis is showing the correlation value for a point determined in the attack results. The graphs show where the values escape the noise floor. As compared to the hardware collection they follow similar trends, but have a fewer number of traces required to determine the correct key. The unprotected design required 1000 traces to reveal the correct guess while the zero mask t-private and fully protected t-private designs required 500 and 5000 respectively.

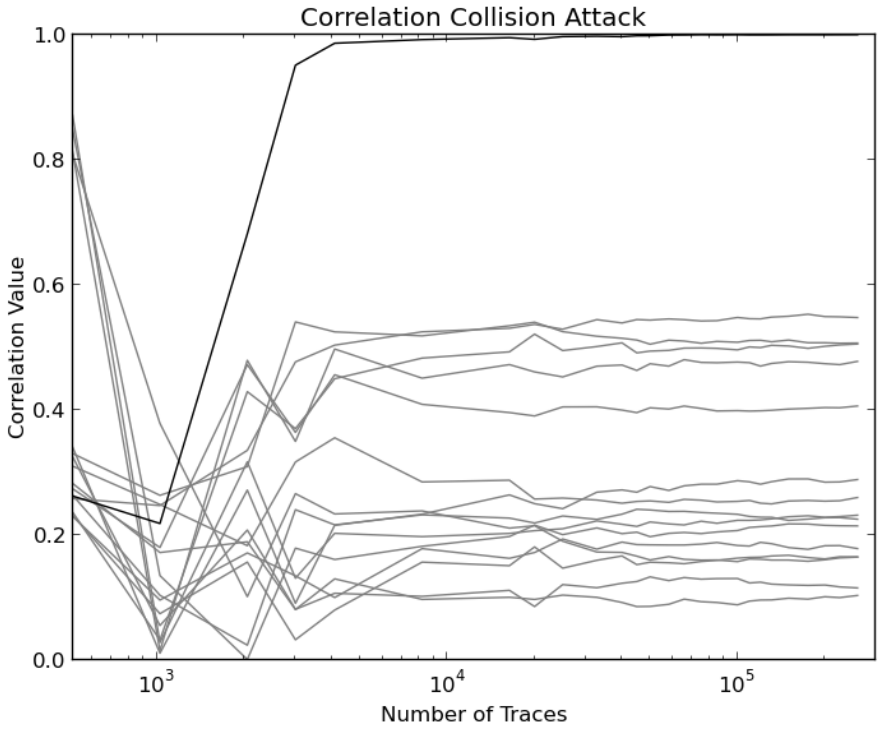

Figure 21. Correlation Collision Attack for Unprotected PRESENT

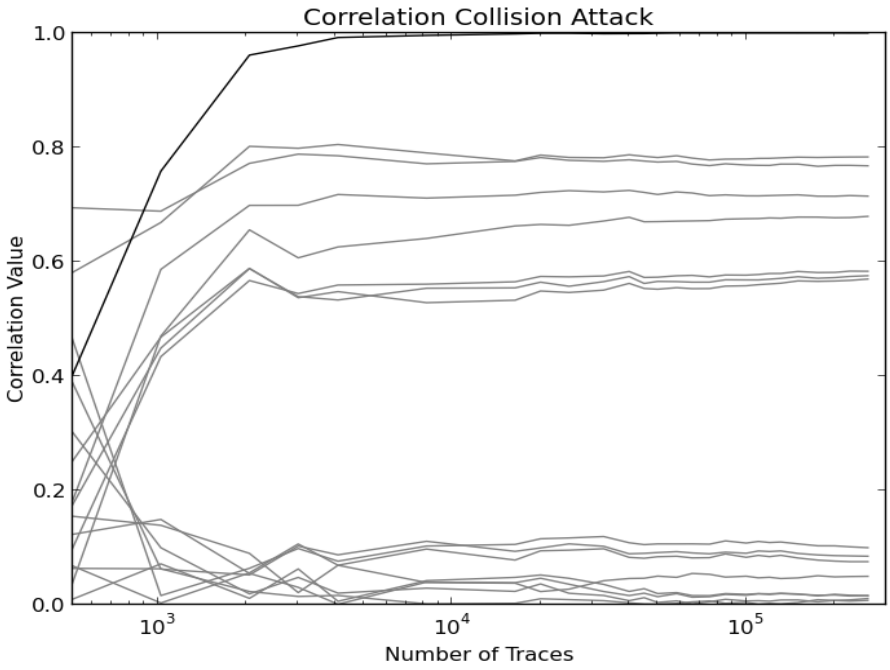

Figure 22. Correlation Collision Attack for Zero-Mask T-private PRESENT

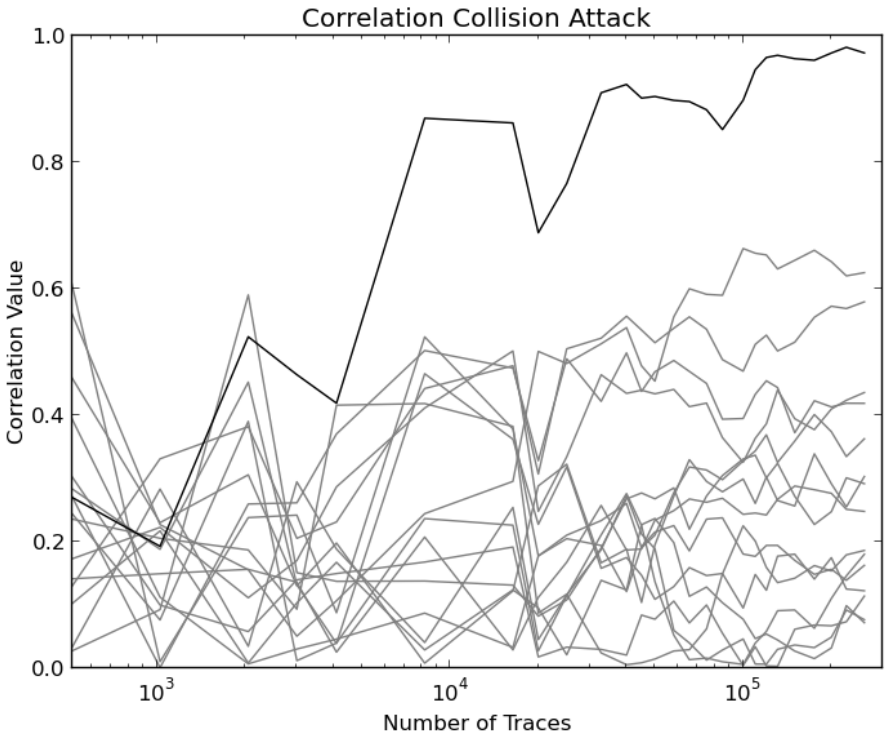

Figure 23. Correlation Collsion Attack for T-private PRESENT Design

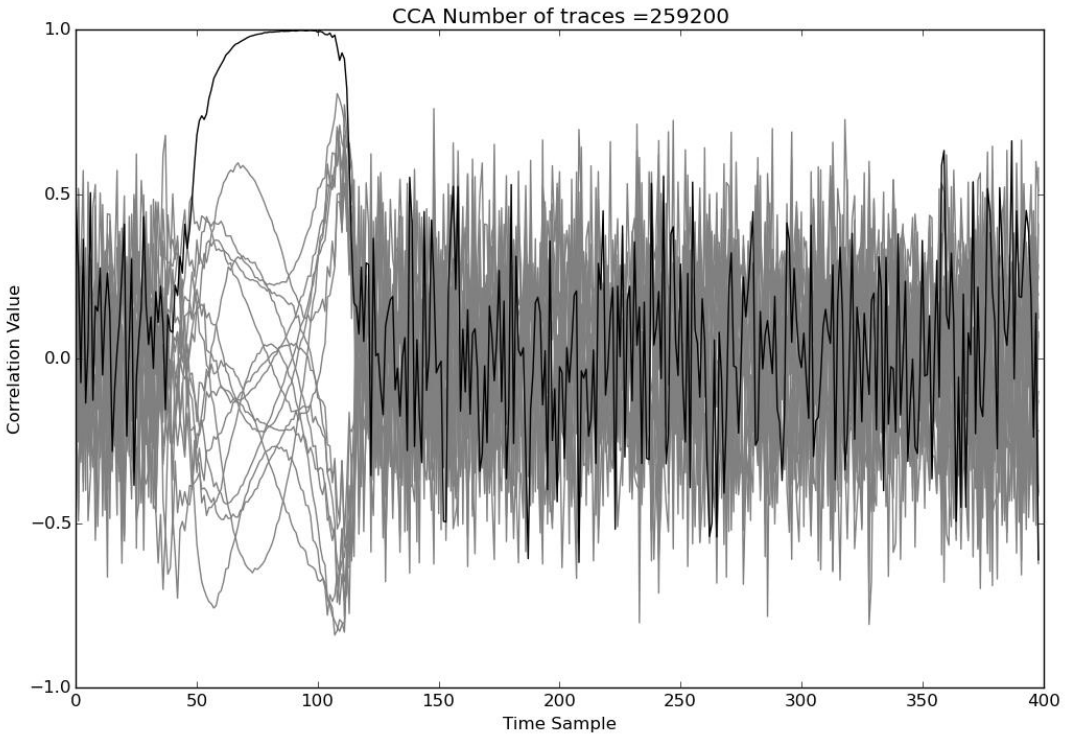

Figure 24. Correlation Collision Attack on T-Private

As can be seen above, the results of the correlation collision attack is shown above. The correct key guess is shown in black and the noise is in gray. This result was for the highest value of number of traces collected. The attack on 259200 traces produced a clear correlation.

C. Differences between Simulation and Hardware Capture Attack Results

As shown in previous sections there are similarities and differences between the simulation and hardware attack results. Mainly of interest is the number of traces required to resolve a correct key guess for the correlation collision attack. This can be due to several different factors. The oscilloscope that was used was the Tektronix MDO4104B-6 and that is equipped with an 8 bit analog to digital converter. This limiting factor caused the max resolution for each trace to be 10mV per division. This accuracy is not equivalent to the filtered toggle values, which was represented by a 64bit float in python. Another difference was the amount of noise present in the simulated design vs. the hardware capture. The amount of Gaussian noise added to the simulated design was a small amount to allow for the variance to be enough for the classical attacks to function without zero variance errors. Finally the simulation data is deterministic for a certain input state transition. The standard delay file contains information about the timing for the design under test. These timing values do not change so if a state transition occurs before that is repeated then the same timing and toggles are used and thus the simulation data would produce the same trace except for the Gaussian noise being different. Also each LUT's toggles were weighted the same for the power consumption, in hardware this isn't always true. Capacitive loads differ for each net and thus a toggle for different nets creates different power consumption profiles.

The reason that these facts contribute to the magnitude of traces being lower for the correlation collision attack is that the correlation collision attack is based upon averages. Averaging the traces for a certain value aims to profile the trace and remove Gaussian noise. While the hardware data has a high amount of noise, contains other hardware's power consumption in the profile and has a non-deterministic power consumption, the simulation data

is very regular and has a low amount of noise. The averages also vary by a small amount which causes the precision to come into play here. All of these contribute to the hardware collection data to require a higher number of traces to extract the key guess opposed to the simulation data.

VI. Conclusions

From our results, we can draw some significant conclusions about the t-Private logic style. In [15] t-Private for T=1 is said to be resistant to 1 probe per clock cycle. This is essentially stating that it is resistant to first-order DPA. However, our results clearly show leakage when analyzed using the correlation enhanced collision attack after only 130,000 traces for the full t-Private S-Box implementation. This result was subsequently duplicated in simulation.

The analysis results are particularly interesting because the CPA was unsuccessful for our total number of traces collected at 259,000. This is an important observation because had a designer only used CPA to analyze the effectiveness of t-Private on their implementation, they might have thought it was secure. This demonstrates the challenge in truly characterizing logic-level countermeasures. For further analysis, it would be interesting to determine at what point the CPA shows leakage, if at all.

Another important observation we made is that our results for hardware match those from simulation. This validates that the measurements gathered from the Sasebo GII were legitimate.

Finally, it may be unrealistic to implement a full design using t-Private. This is due to an increase in overhead of about 3.5x.

VII. Further Work

While conclusions have been drawn in section VI there is still room for further development and avenues for research. These are outlined in this section.

A. Full Encryption Implementation

While it was demonstrated that it is possible to make a non-linear combinational circuit (S-Box) a full cipher implementation could be developed using the tools that were developed during this project. The design for sequential elements has been presented in [15]. Performing a security analysis on a full implementation could also be useful for designers so they can understand how to implement the sequential elements and their effective security.

B. Perceived Information Template Attack

During the duration of the project a perceived information template attack script was developed in python. This script uses expected power consumption from the simulation toggle model and examines the amount of information comes from observing the power traces and matching that to template results. Both mask transitions and input transitions could be examined to explore the effect of information released upon mask values.

C. Evaluating T=2 for DPA Resistance

This study focused on t=1 for first order attack resistance. While it has been shown that this logic style has a first order vulnerability, the effectiveness of t=2 against the correlation collision attack would be interesting to understand. The overhead associated with t=2 is a 2x increase over the t=1 primitive gates. The tradeoffs between power, size, throughput and security would be useful to future designers.

VIII. References

[1] IAIK, “Introduction to Implementation Attacks.” [Online]. Available: http://www.iaik.tugraz.at/content/research/implementation_attacks/introduction_to_impa/.

[2] S. Mangard, M. E. Oswald, and T. Popp, *Power Analysis Attacks - Revealing the Secrets of Smart Cards*, vol. 54. 2007, pp. I–XXIII, 1–337.

[3] K. Tiri, and I. Verbauwhede, A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation, Proc. DATE 2004, pp. 246-251, February 2004.

[4] Suzuki, Saeki, Ichikawa, "Random Switching Logic: A Countermeasure agains DPA based upon Transition Probability," IACR 2004 [online] Available: http://eprint.iacr.org/2004/346

[5] T. Popp and S. Mangard, "Masked Dual-Rail Pre-Charge Logic: DPA-Resistance without Routing Constraints," in Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, Scotland, August 29 - September 1, 2005, Proceedings, ser. LNCS, vol. 3659. Springer, 2005, pp. 172-186

[6] B. Wilson, “CMOS Logic,” 2008. [Online]. Available: http://cnx.org/content/m1029/latest/. [Accessed: 30-Aug-2014].

[7] Fischer, Wieland, and Berndt M. Gammel. "Masking at gate level in the presence of glitches." In *Cryptographic Hardware and Embedded Systems–CHES 2005*, pp. 187-200. Springer Berlin Heidelberg, 2005.

[8] Park, Jungmin, and Akhilesh Tyagi. "t-Private logic synthesis on FPGAs." In *Hardware-Oriented Security and Trust (HOST), 2012 IEEE International Symposium on*, pp. 63-68. IEEE, 2012.

[9] P.-C. Liu , H.-C. Chang and C.-Y. Lee "A low overhead DPA countermeasure circuit based on ring oscillators" *IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 57, no. 7, pp. 546-550, 2010*

[10] W. E. Burr, “Selecting the Advanced Encryption Standard,” *IEEE Security Privacy Magazine*, vol. 1, pp. 43–52, 2003.

[11] K. H. B. K. H. Boey, P. Hodgers, Y. L. Y. Lu, M. O’Neill, and R. Woods, “Security of AES Sbox designs to power analysis,” *Electronics Circuits and Systems ICECS 2010 17th IEEE International Conference on*, pp. 1232–1235, 2010.

[12] E. Prouff, “DPA Attacks and S-Boxes,” pp. 424–441, 2005.

[13] U. S-boxes and C. L. C. Tn, “Ultra-Low Power S-Boxes Architecture for AES,” vol. 15, no. 1, 2008.

[14] K. H. B. K. H. Boey, P. Hodgers, Y. L. Y. Lu, M. O'Neill, and R. Woods, "Security of AES Sbox designs to power analysis," *Electronics Circuits and Systems ICECS 2010 17th IEEE International Conference on*, pp. 1232–1235, 201 (Placeholder1)

[15] Moradi, A., Mischke, O., & Eisenbarth, T. (2010). Correlation-enhanced power analysis collision attack. In *Cryptographic Hardware and Embedded Systems, CHES 2010* (pp. 125-139). Springer Berlin Heidelberg.

[16] Y. Ishai A. Sahai & Wagner, D. Private circuits: Securing hardware against probing attacks. In *Advances in Cryptology- CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21,2003, Proceedings, ser. Lecture Notes in Computer Science,vol. 2729,* **2003**

[17] Park, J, and Akhilesh T. "Towards Making Private Circuits Practical: DPA Resistant Private Circuits." In *VLSI (ISVLSI), 2014 IEEE Computer Society Annual Symposium on*, pp. 528-533. IEEE, 2014.

[18] Haley, V and Huilihan, D, "System Security Metrics Via Power Simulation for VLSI Designs", WPI, 2013

[19] Leiserson, A. and Marson, M. and Wachs, M. "Gate-Level Masking under a Path-Based Leakage Metric" In Cryptographic Hardware and Embedded Systems CHES 2014 Springer Berlin Heidelberg 2014

[20] SATOH," SASEBO-GII" [online] http://satoh.cs.uec.ac.jp/SASEBO/en/board/sasebo-g2.html