2019•2020

Faculteit Industriële ingenieurswetenschappen

master in de industriële wetenschappen: elektronica-ICT

Masterthesis

Remote power analysis attacks on reconfigurable cloud resources

PROMOTOR :

Prof. dr. ir. Nele MENTENS

COPROMOTOR :

dr. ing. Jo VLIEGEN

Dominique Meus

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: elektronica-ICT

Gezamenlijke opleiding UHasselt en KU Leuven

2019 • 2020

Faculteit Industriële ingenieurswetenschappen

master in de industriële wetenschappen: elektronica-ICT

Masterthesis

Remote power analysis attacks on reconfigurable cloud resources

PROMOTOR :

Prof. dr. ir. Nele MENTENS

COPROMOTOR :

dr. ing. Jo VLIEGEN

Dominique Meus

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: elektronica-ICT

UHASSELT KU LEUVEN

Deze masterproef werd geschreven tijdens de COVID-19 crisis in 2020. Deze wereldwijde gezondheidscrisis heeft mogelijk een impact gehad op de opdracht, de onderzoekshandelingen en de onderzoeksresultaten.

Acknowledgements

I would like to thank my advisors prof. dr. ir. Nele Mentens and dr. ing. Jo Vliegen for providing me the opportunity to work on this thesis. I would like to specifically thank them for their openness, patience and encouragement. Their ideas and suggestions played a big role in producing this work.

Contents

List of Figures

Glossary

AES	Advanced encryption standard
CLB	Complex logic block
DES	Data encryption standard
DPA	Differential power analysis
FF	Flip-flop
FPGA	Field-programmable gate array
LUT	Lookup table
MMCM	Mixed-mode clock manager
RO	Ring oscillator
SCA	Side-channel attack
SPA	Simple power analysis
TDC	Time-to-digital converter
TDL	Tapped delay line

Abstract

Field-programmable gate arrays (FPGA) are highly customizable devices which make them interesting devices to deploy in the cloud. To optimally use the hardware, multiple users could share a portion of one FPGA. The customizability is also a risk, as several side-channel attacks have been developed that use the reconfigurable fabric of the FPGA as sensors to gather information on other tenants or even other devices sharing the same power supply. These sensors often rely on time-to-digital converters to measure power consumption at nano-second scale. If one tenant is using such an FPGA as a cryptographic accelerator, another tenant could use power analysis to recover a secret key.

Currently, encryption cores running up to 100 MHz have been attacked using a tapped delay line (TDL) sensor. There are two main limitations to this design, one disadvantage is that the TDL has less time to measure the side-channel leakage as frequency goes up. Another disadvantage is that the TDL has to be calibrated for each device and for clock speed.

This thesis contains measurements of ring oscillators, registers and AES core activity at 25 and 50 MHz on a Xilinx VC707 board. The results show that the side-channel leakage decreases as the frequency goes up. However, the effect of a single AES core is too small to perform power analysis on.

Abstract in dutch

Field-programmable gate arrays (FPGA) zijn programmeerbare logische circuits met interessante toepassingen in de cloud. Om deze hardware optimaal te benutten zouden meerdere gebruikers dezelfde FPGA kunnen gebruiken. Echter is de configureerbaarheid ook een risico, zo zijn er verschillende side-channel attacks ontwikkeld die de FPGA configureren als een sensor om informatie van andere gebruikers of andere chips met dezelfde voeding te meten. Deze sensors maken gebruik van time-to-digital convertors om het vermogenverbruik te meten op nanosecondeschaal. Als één gebruiker de FPGA gebruikt voor cryptografische doeleinden, dan zou een andere vermogensanalyse kunnen toepassen om de geheime sleutel te achterhalen.

Tot op het heden zijn er encryptiemodules tot 100 MHz succesvol aangevallen met tapped delay line sensoren. Er zijn twee nadelen bij deze methode, één nadeel is dat de sensor minder tijd heeft om te meten als de frequentie stijgt. Een ander nadeel is dat deze sensor apart gekalibreerd moet worden voor elk apparaat en elke klokfrequentie.

Deze thesis bevat metingen van ring oscillators, registers en AES core activiteit aan 25 en 50 MHz op een Xilinx VC707 FPGA bord. De resultaten bevestigen dat de side-channel leakage afneemt als de frequentie stijgt. Echter is het effect van een enkele AES core te klein om power analysis op uit te voeren.

Chapter 1

Introduction

Field-programmable gate arrays (FPGAs) are semiconductor devices which are intended to be configured after manufacturing. They are mainly built out of configurable logic blocks (CLB) which can be arbitrarily connected together. This allows a designer to build various functions on a single device and reconfigure the device if the requirements change. The flexibility and performance of these devices makes them attractive candidates to implement in a cloud platform [1].

Many FPGAs support partial reconfiguration [2, 3] which allows a part of the device to be reconfigured instead of reconfiguring the whole device at once. This enables cloud providers to share one FPGA with multiple tenants. However, the reconfigurable nature of FPGAs allows a developer to build sensors that measure the activity of other tenants on the FPGA. This presents a security threat to the other tenants, especially if they are using the FPGA as a cryptographic accelerator. Side-channel attacks, attacks that target the implementation instead of the algorithm, have been developed and tested on various FPGAs [4–8]. These attacks may use a time-to-digital converter (TDC) to measure power usage of the FPGA. Since the power usage depends on the switching activity of the circuit implemented, power analysis attacks can be used to gain knowledge about the other circuit. Figure 1.1 illustrates the setup, the shared power supply is a source of information for tenant 2.

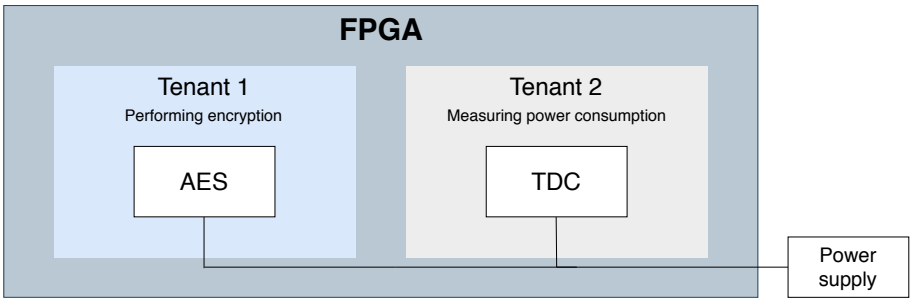


Figure 1.1: FPGA with multiple tenants sharing a power supply

1.1 Problem statement

The main problem is that FPGAs allow users to configure a circuit that can measure the power consumption of the whole device. In a situation where multiple tenants have access to different

parts of the FPGA that share a common power supply, this presents a security vulnerability because a malignant tenant can use the power consumption data to gain information about the circuits of the other tenants. Attacks against hardware vulnerabilities that are inherent to the implementation of the device are called side-channel attacks (SCA). Different techniques such as simple power analysis (SPA), differential power analysis (DPA) or correlation power analysis (CPA) can be used to analyse the power consumption data.

In the worst-case scenario, the attacker may partially or completely recover the secret key. In this scenario, the encryption can be considered defeated and the encrypted data insecure. A partial key could reduce the computational complexity enough for the attacker to decrypt the data in reasonable time.

The limitations of power analysis attacks must be researched to properly assess the risk they pose to multi-tenant FPGAs. There are two common architectures, one is based on a tapped delay line (TDL) and the other on a ring oscillator (RO). In [6], an AES secret key was recovered at frequencies up to 96 MHz using a TDC. Schellenberg et al. [6] note that lower frequencies result in larger variations in their power measurement. Zhao and Suh [5] demonstrate that an RO based architecture works not only on one FPGA, but may also measure the power usage of a CPU on the same SoC. Furthermore, [7] shows that this side-channel may be abused at board-level for chips that share the same power supply. In [4, 8] frequencies up to 50 MHz were successfully tested using an architecture based on an RO and a transmitter-receiver pair of wires. Ramesh et al. [4] also noted that higher frequencies decreases the size of the side-channel, thus requiring more samples to generate a successful key guess. However, the question remains whether attacks using a power side-channel are possible or practical at frequencies over 100 MHz.

1.2 Project objectives

The goal of this thesis is to evaluate the security risks associated to using FPGAs as cloud resources. Currently, the most effective architecture uses a TDL which requires fine-tuning per device. The first objective is to simplify the calibration procedure such that the TDL architecture becomes more portable. Several variations on the architecture exist, but generally two parameters must be experimentally picked by the developer: initial and observable delay. A designer has enough information to calculate an upper bound for initial delay, but the actual delay may be significantly smaller. Thus the need to experimentally calibrate initial delay could be eliminated. The second objective is to characterize the efficacy of TDL sensors at frequencies over 100 MHz.

1.3 Outline

Chapter 2 explains the architecture and limitations of the two main TDC architectures. Next, chapter 3 explains how these TDCs may be used to attack cryptographic algorithms. Chapter 4 describes the tools and methods used to implement and test a TDL sensor. Finally, the results are laid out in chapter 5 and the conclusions and ideas for future work are given in chapter 6.

Chapter 2

Power sensors on FPGA

2.1 Introduction

In order to perform side-channel attacks on an FPGA, the attacker must implement a sensor on the FPGA that measures the side channel. By monitoring the propagation delay of gates in the chip, the attacker can infer the relative power usage of the chip. More specifically, a tapped delay line sensor measures the time it takes for a signal to propagate through a delay line. Another way to measure the propagation delay is to measure the frequency of a ring oscillator, as the duration of each oscillation is directly dependent on the propagation delay.

These types of sensors are TDCs that measure the propagation delay of an internal signal of the FPGA. TDCs are used in any field where accurate time measurements are needed, thus other architectures than the two discussed in this chapter exist. However, this chapter is limited to TDLs and ROs since they are researched extensively for power sensing.

2.2 Tapped delay lines

A tapped delay line is a signal line with delay elements with one or multiple taps. A signal passing through the delay line can be measured by observing the taps, which are simply measuring points between the delay elements. The propagation delay of the signal is measured by storing the logical value of all the taps at the same time. The propagation delay can then be expressed as the time between applying the signal and capturing the values at the taps divided by the amount of taps that changed value.

2.2.1 Working principle

Figure 2.1 illustrates a TDL. The upper row of buffers delay the incoming clock signal. In this implementation the falling edge of the clock signal is used to enable the latches which act as taps between the delay elements. Thus, the row of buffers is actively measuring when the clock signal is high. The clock propagates through the delay line as a function of propagation delay of each buffer. Activity on the board, or another device sharing the same power supply, will cause the

voltage of the power supply to sag momentarily. As a result, the buffers in the delay line exhibit larger propagation delays.

The buffers are separated into two sections, an initial delay line and a observable delay line. The clock period is the most important factor in choosing the length of the initial delay line.

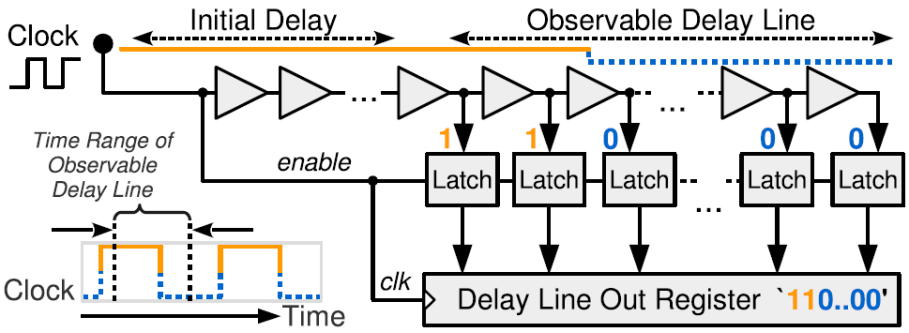


Figure 2.1: Tapped delay line [9, p. 3]

2.2.2 Implementation

To achieve the highest resolution possible, the propagation delay of each delay element should be as small as possible. Furthermore, to get a consistent linear measurement, each delay element should have an equal delay. The designer has little control over the actual delay between primitives. The length of the traces connecting the primitives together is unknown. Hence on Xilinx FPGAs, CARRY4 primitives are commonly used. The dedicated carry logic with carry in and carry out ports allow this primitive to be chained together to make a carry chain of arbitrary size. There are several advantages of using this primitive over other logic elements. One advantage is that a chain of carry elements get consistently fitted to a similar vertical structure and location constraints can be used to guarantee a static structure. This ensures that there is minimal delay between the carry output of one primitive to the carry input of another. Next, carry primitives have low propagation delay. Depending on device and speed grade, actual delays of 17.5 ps have been reported in [10–12]. Combining 4 delay elements into one primitive results in a total delay of around 70 ps per CARRY4 primitive.

Figure 2.2 is a schematic of the innards of a CARRY4 primitive. The delay path is coloured in red. By setting the MUXCY multiplexors input to 1 via S0 to S3, the CIN signal is propagated through each MUXCY to COUT. Each MUXCY constitutes one delay stage. The taps are the carry outputs CO0 to CO3. All taps should be connected to a flip-flop with a common clock. The rising edge of this clock signal constitutes the stop signal of the delay line. The start signal of the delay line is connected to the carry input of the first carry primitive. When the start signal goes high, each carry output bit should sequentially go high. A short while later, the stop clock captures the values of each carry out bit in flip-flops. If the stop clock goes high before the start signal passes through the whole delay line, a relative measurement of delay between start and stop is captured in the flip-flops.

Figure 2.3 illustrates how such a delay line could be implemented. The large rectangular blocks are the carry primitives and the smaller squares are the flip flops capturing the data at the taps. The carry out from the primitive below the selected primitive feeds into the carry in from the selected primitive.

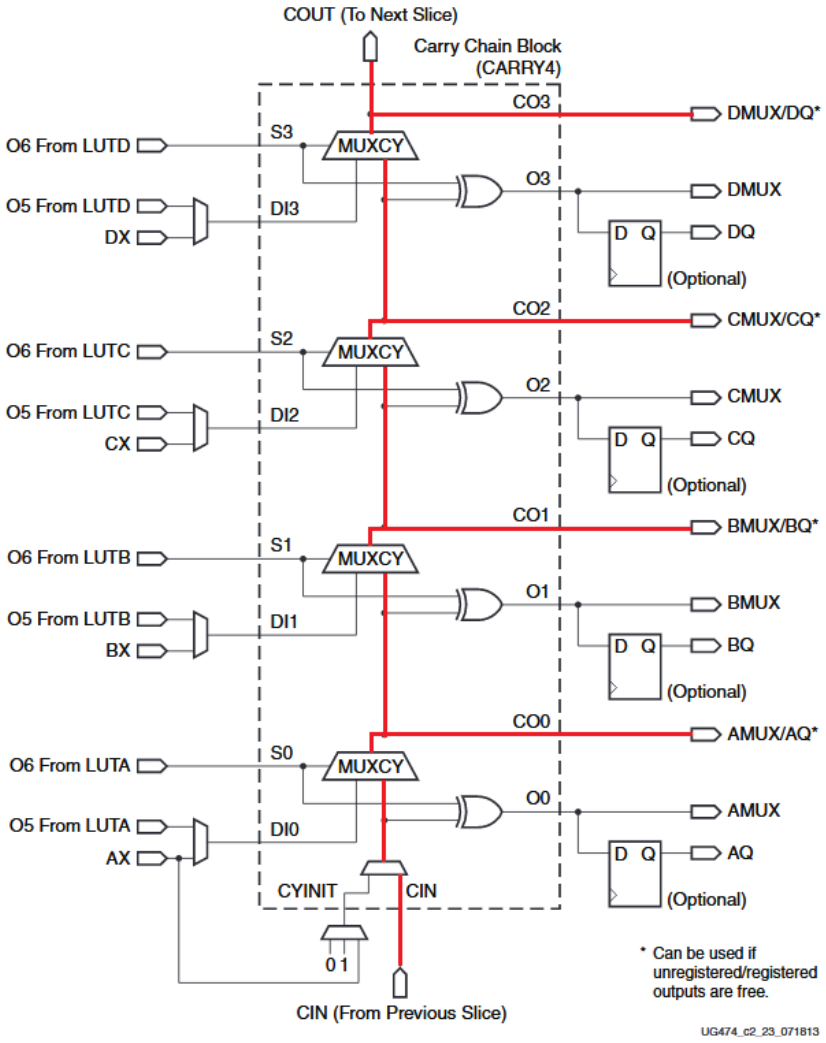


Figure 2.2: CARRY4 primitive [13, p. 43]

2.2.3 Limitations

To get a precise measurement, it is important that the stop signal clocks every tap simultaneously. However, clock lines on an FPGA are divided into different clock regions [14]. Long delay lines could span multiple clock regions and introduce non-linearities between regions. In [15] the start signal is first passed through another kind of delay line which consists of lookup tables (LUTs) and open latches before it enters the carry chain. This extra initial delay line reduces the need for excessive CARRY4 primitives and allows a complete design to fit into one clock region. Zick et al. claim the open latches enhance voltage sensitivity [15, p. 2].

Finally, it is important that the delay between the start and stop signal is consistent at pico-second resolution. Any jitter larger than the smallest delay between taps will result in noisy consecutive measurements, even if the actual propagation delays of the gates are stable.

2.3 Ring oscillators

A ring oscillator (RO) is a ring of an odd numbers of inverters. The most elementary form is an inverter connected to a buffer. Every even number of inverters in an RO could be interpreted as

Figure 2.3: Delay line using CARRY4 primitives

a buffer. Connecting an AND gate into the ring, as in figure 2.4, creates a simple enable signal to turn the oscillator on or off.

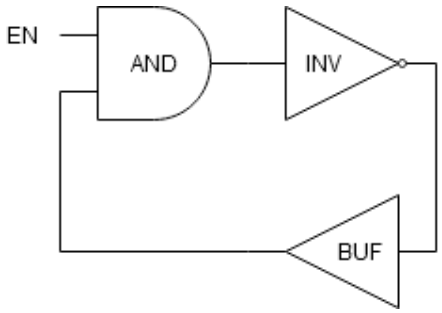


Figure 2.4: Ring oscillator with enable signal

2.3.1 Working principle

If an inverter has a 0 on its input, it will output a 1. After the delay of remaining gates and wires in the ring, the 1 arrives at the input of the inverter, causing it to flip its output to 0. The same process is repeated with inverse values and the circuit is returned to its initial state. When an RO is enabled, any point in the ring oscillates at a frequency at a rate inversely proportional to the total delay of the ring. In figure 2.4 this delay consist of the propagation delay of the inverter, buffer and AND gate plus the delay of the signal travelling through the wires connecting the gates. Propagation delay of CMOS gates are dependent on several factors such as supply voltage, manufacturing technology and temperature. Thus, assuming all other factors are constant, measuring the frequency of a ring oscillator is a way to measure the voltage of the gates in the ring.

2.3.2 Implementation

One way to implement an RO on an FPGA, is to omit the buffer as the AND gate acts as a very small buffer. The inverter and AND gate can each be implemented in one slice. Compared to an ASIC, the routing delay on an FPGA is also likely to be larger. To measure the frequency, [5] suggests using a chain of T-flip-flops (TFF) because the RO oscillates much faster than the system clock. The toggle of each TFF is pulled high and the ring oscillator feeds the clock of the first TFF. The output of the first TFF feeds the clock of the second, as illustrated in figure 2.5. Alternatively [8] suggests using a Johnson Ring Counter.

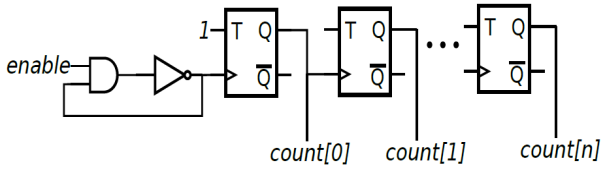


Figure 2.5: Ring oscillator with TFFs to count oscillations [5, p. 4]

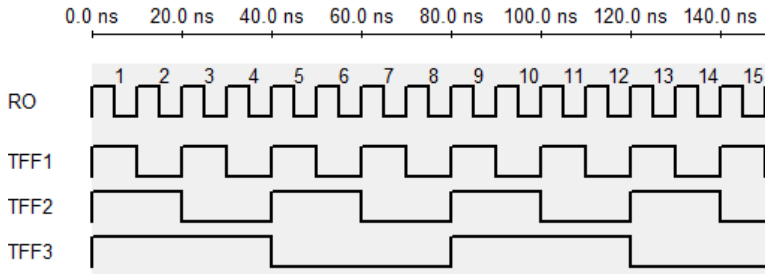


Figure 2.6: Clock diagram for a chain of TFFs

A chain of TFFs with its toggle ports pulled high is essentially a counter, as each TFF halves the frequency of the previous TFF since it only toggles on the rising edge. A clock diagram of a chain of TFFs is given in figure 2.6. Note that TFF3-TFF1 is effectively counting down, but inverting every signal would result in an up counter. The frequency of the RO can be determined by comparing the count, or the amount of toggles, to a clocked counter with a known frequency. The amount of TFF in the chain should be great enough such that the last TFF never gets toggled, this ensures the counter has not rolled over. For example, in figure 2.6, the RO has a frequency of 100 MHz, the first TFF has a frequency of 50 MHz, the second TFF has a frequency of 25 MHz and the third TFF has a frequency of 12.5 MHz. If the reference clock period is more than 80ns, the count would be ambiguous.

2.3.3 Limitations

Compared to a tapped delay line sensor, an RO is simpler to implement and more portable across different FPGAs. However, [15] notes that an RO sensor is best suited for measuring static effects or slow transients. In [8], Gravellier et al. note several disadvantages: frequency dependent resolution, quantization error and counter timing error. The temporal resolution of the TDC depends on the frequency of the RO. For a given measuring period, a higher frequency RO returns a higher resolution measurement. Therefore, it is important to maximize the oscillation

frequency of the RO. Next, the counter measuring the RO count must be specially designed with very high frequencies in mind. Finally, a RO is a combinatorial loop and synthesis tools already recognize this structure, thus it would be easy to detect and forbid from being implemented in a multi-tenant FPGA.

Chapter 3

Power analysis attacks

Power analysis attacks are effective because they target the implementation of a security system rather than the cryptographic algorithm. Since the hardware implementation of a system is often abstracted away from the designer, it is easy to overlook the security implications it has on the system. There are several ways an attacker could exploit fine-grained power measurements on a system performing cryptographic calculations. The simplest technique, simple power analysis (SPA), interprets power measurements over time. Conversely, differential power analysis (DPA) exploits the power data by utilizing statistics on many traces and incorporating the plaintext or ciphertext. Closely related to DPA is correlation power analysis, where power traces are compared to a model of the power leakage. Kocher et al. cover several techniques in [16].

3.1 Simple power analysis

Simple power analysis refers to interpreting a power trace over time. The main idea is that power usage scales with electrical activity. The simplest kind of SPA attack is to identify if a system is in an idle state or not. If a system is idle, the power draw is likely to be constant or follows a consistent pattern. Alternatively, if a device is computing information, the power draw is more chaotic or follows a different pattern than idle. Though this information may seem useless, an attacker could use this data to decide when to attack. If the attacker intends to perform a denial of service attack, it is valuable to know when a system is most active.

A lot more information can be leaked through power traces. Figure 3.1 is a power trace from a system performing encryption using data encryption standard (DES). From visually inspecting the trace, it is immediately clear some action is repeated 16 times. Counting the amount of repeated patterns could help an attacker determine which algorithm is being performed on a device. With access to an accurate power trace, an attacker can compare rounds to each other. If the algorithm is implemented on a CPU, a branch depending on the ciphertext or key value may cause the round to be slightly longer.

With a naive implementation, a great deal can be learned about a system from its power trace. In reality, an accurate power trace often requires physical access. However, even with a noisy power trace, an attacker can likely still determine which algorithm is being used. With this

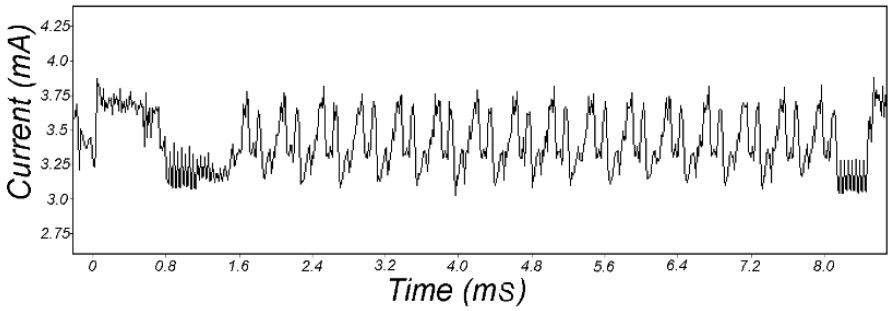


Figure 3.1: SPA trace showing an entire DES operation [17, p. 2]

information, differential power analysis can uncover even more.

3.2 Differential power analysis

In contrast to SPA where information is deduced from the structure of an implementation, DPA targets power differences caused by different data. By collecting many power traces, an average trace may be computed. Depending on if the plaintext or ciphertext is known and what algorithm is being used, the attacker targets a specific step in an encryption algorithm.

Consider a situation like in figure 3.2, which is similar to the last round of AES. Assume signals Key, A and Ciphertext are 8-bit values and that a large amount of power traces and corresponding Ciphertexts are known. Randomly guess the value of Key and compute A under the assumption your guess is correct. Next, invert the Byte substitution operation to get a value for BS0. If there is a difference in power consumption based on data, then the set of power traces with corresponding BS0 calculated to be 0 should differ from the set of traces where BS0 was calculated to be 1. Compute the average for both sets of traces and subtract them from each other. If the key guess is incorrect, the calculated BS0 value is incorrect and the set of power traces is separated into two random sets, which are the same on average. If the key guess is correct, the two sets should have different averages.

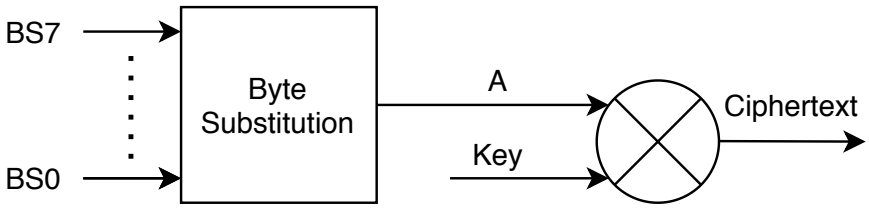


Figure 3.2: Simplified last round of AES for one byte

Repeat this process for every possible key. In this case, for an 8-bit key, there are only 256 different options. The key guess that resulted in the largest peak difference between sets of power traces is most likely to be the actual key. Note that the choice of BS0 is arbitrary here, any other BS bit may be used to separate the set of traces into two sets.

Chapter 4

Experimental procedure

To date, the best performing architecture is the TDL variant. As previously mentioned, this type of sensor requires careful fine-tuning for clock frequency and for different devices. Alternatively, a very long carry-chain without initial delay could be used such that a large time period is covered with one sensor. Ideally the carry-chain is placed into one clock region so that the clock skew is kept to a minimum. Assuming 70 ps per CARRY4 primitive and a chain length of 50 primitives, a delay of less than 3.5 ns could be measured. This kind of architecture would limit the sensor to clock speeds of 150 MHz and up. Note that the architecture in figure 2.1 measures only when the clock is high, thus the delay line is only active for half a clock period.

Since attacks have only been successfully implemented at 100 MHz or slower, it is recommended to include an initial delay line using elements with larger delays such as open latches. The propagation delay of the larger delay element determines how many elements should be added to the delay line. The manufacturer typically specifies pessimistic delay estimates so that a normal design is likely to meet timing if the pessimistic delay is taken into account. No minimum delay is given and the actual delay could be significantly less than the given pessimistic delay, therefore it is necessary to fine-tune for each device.

4.1 Automatic calibration

Figure 4.1 illustrates how the initial delay line from figure 2.1 can be turned into a variable delay line. If Enable_0 is high, then the signal from start only passes through the topmost AND gate and the OR gate. This results in a relatively short initial delay. Alternatively, if Enable_n is high, then the signal has to pass through n delay elements, an AND gate and the OR gate. The initial delay can be set by choosing the right enable signal.

To automatically determine the right enable signal, start by setting Enable_0 high. If the delay line is just right, the clock signal should not propagate to the end of the observable delay line. If the delay line is too short, the last tap of the observable delay line will measure high. Consequently, the last bit of the Delay Line Out Register will be high. Using this last bit, the enable register is shifted right by one. Now Enable_1 is high and the initial delay has increased one delay element. This process is repeated until the last tap of the observable delay line is not high, indicating that the initial delay line is long enough.

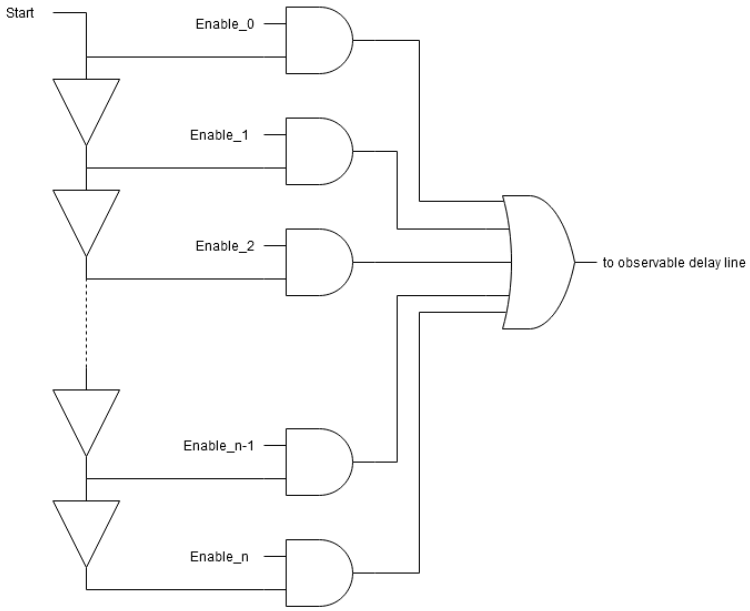


Figure 4.1: Variable initial delay line principle

4.2 Implementation

The final TDL implementation is illustrated in figure 4.2. The output of the CARRY4 chain is captured with two D Flip-Flops (FF) in series to avoid metastability issues. Compared to the negative edge trigged latch used in figure 2.1, using a separate clock is more flexible. By setting the phase shift, the delay data can be captured at any time. By decreasing the phase shift at lower frequencies, a relatively small delay line could be used as "the amount of time during which the side-channel leakage can be leveraged is not bounded to the AES frequency but to the device itself" [8, p. 7]. The disadvantage is that phase shift jitter may impact the measurement. The Mixed-Mode Clock Manager (MMCM) displays a maximum peak-to-peak jitter of 129.198 ps for a 100 MHz clock. If this were the case, the TDL data could swing over 7 bins using only one clock. Fortunately, the jitter is an order of magnitude lower in practice.

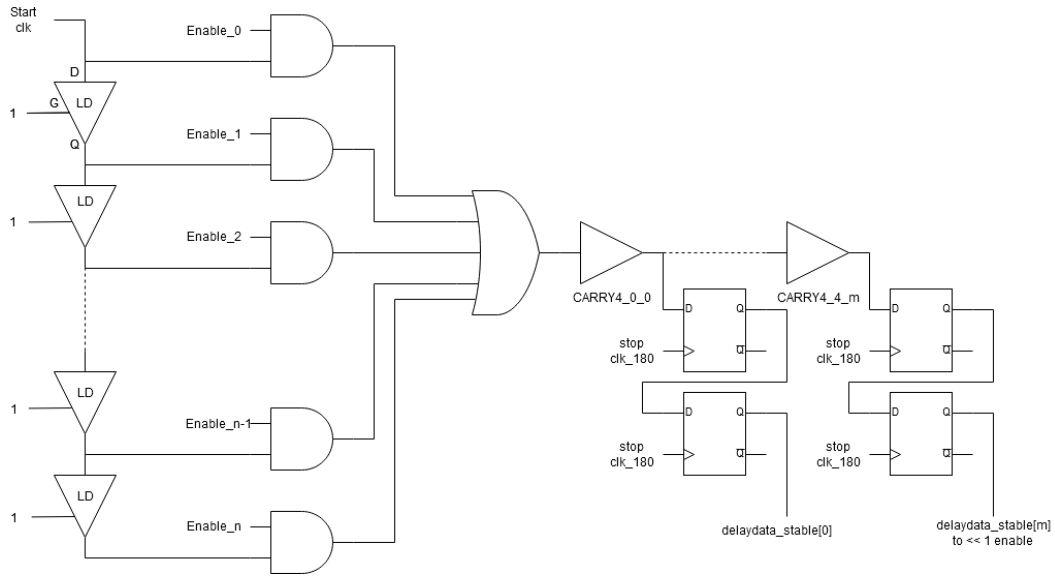


Figure 4.2: Tapped delay line with automatic initial calibration

The primary clock is derived from the MMCM. Additionally a secondary clock phase shifted 180° was used to capture the delay data. The data was recovered from the FPGA using the Xilinx Internal Logic Analyser (ILA) IP core. This IP implements a logic analyser on the FPGA and streams the data to the host computer over a JTAG connection. From Vivado, the data can be exported to a comma-separated values (csv) file for further processing. Using a Python script, bubble correction was applied to the delay data and then the binary values were converted to decimal numbers.

4.3 Hardware and toolchain

Every experiment was run on a Xilinx Virtex-7 VC707 Evaluation Kit. This kit features a VX485T FPGA with a 200 MHz differential clock on board. Synthesis, implementation and bit-stream generation were performed in Vivado 2018.3. The opt_design steps in implementation are turned off to prevent Vivado from optimizing away logic [18]. The FPGA was used at room temperature (20°C) with the fan disabled. The code was written in Verilog. To avoid synthesis removing extra logic, KEEP and DONT_TOUCH constraints were used [19]. Additionally, relative location (RLOC) constraints were used to ensure the CARRY4 elements were implemented as a chain. The webpage [10] provides a working example in VHDL compatible with most 6-series and higher Xilinx FPGAs.

4.4 Benchmarking performance

To verify the functionality and test the performance of the TDL sensor, several test circuits were implemented. A counter was used to enable and disable the test circuits so a comparison between an active and inactive circuit can be made. The test circuits are: ring oscillators, large registers alternating state, and an AES core.

4.4.1 Ring oscillators

An RO consumes a lot of power because it switches state very rapidly. Additionally an RO occupies relatively little area. These properties make it a very good circuit to test the functionality of our sensor since many ROs can fit in same clock domain the sensor is located in. The ROs were implemented with an enable signal similar to figure 2.4. The enable signal was driven by a bit from a counter, causing all ROs to enable and disable at the same time. Since this circuit drives no other logic, it is important to use constraints to prevent it from being synthesized out. To generate many copies of a RO, a for loop was used and the loop variable was passed to the RO module. Passing the loop variable prevents Vivado from trimming identical instantiations.

4.4.2 Toggling registers

Once the TDL sensor is verified to work, a more realistic target is needed. Thus, the second test circuit consists of turning large registers on and off. This circuit has considerably less switching

activity compared to the RO test, since there is only one transition for each bit every clock cycle compared to many transitions every clock cycle.

4.4.3 AES core

Finally an AES core is measured. The 128-bit AES core is implemented with a 128-bit datapath. This means that the input, key and output are 128-bit signals. AES128 consists of ten rounds and each round is computed in one clock cycle. Byte substitution and mix columns are implemented as 8 bit and 32 bit lookup tables. The implementation contains no additional pipelining or other features to improve performance or area.

Chapter 5

Results

The results in the following section are illustrated as a combined figure of an enable signal and the TDL sensor output. The enable signal is active high, i.e. 1 is active and 0 is inactive. The following figures show the output of the carry chain. A delay tap refers to one output from a CARRY4 primitive, four delay taps correspond to one CARRY4 element. As mentioned earlier, on a Xilinx 7-series FPGA, one tap is equal to approximately 17.5 ps of delay. Since an initial delay line of open latches was used, figure 5.1, 5.2 and 5.3a do not show the total delay value.

5.1 Ring oscillators

Figure 5.1 shows the effect of enabling ROs on the TDL sensor. For these figures, the TDL sensor was clocked at 50 MHz. Clearly, enabling ROs decreases the delay. As the amount of ROs increases, the delay gets more and more severe. As the ROs get enabled, the delay continually decreases. When the ROs get disabled, the delay value jumps up a bit and then recovers for a short duration before stabilising. The difference in delay is roughly 8, 12, 16 and 20 taps for 1000, 1500, 2000 and 2500 ROs, which suggests a linear scaling. As the amount of ROs decreases to under 500, the effect becomes harder to visually identify.

5.2 Toggling registers

Figure 5.2 shows the effect of toggling registers on the TDL sensor. Compared to the ROs, the effect of toggling on and off registers on the sensor is less pronounced. Figure 5.2f, 10000 registers at 50 MHz, looks similar to figure 5.1a with 250 ROs. At 2000 registers, it becomes almost impossible to tell if the circuit is enabled from the sensor trace. To increase the delay variance, the clock speed was reduced to 25 MHz. As illustrated in figure 5.2a and 5.2b, the sensors response is somewhat enhanced.

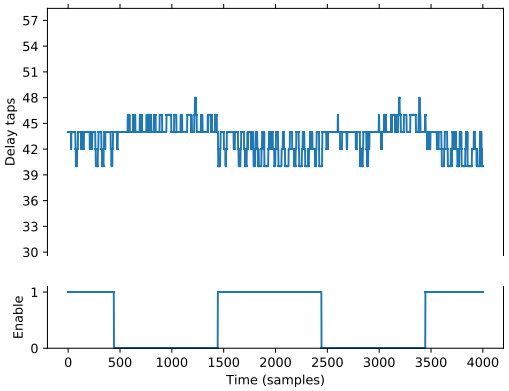
5.3 AES core

Figure 5.3 shows the effect of running an AES core on the TDL sensor at 25 MHz. Visually, in figure 5.3a it is hard to distinguish a difference between enabled and disabled from the delay trace. To amplify the effect, the AES modules were duplicated 5 and 10 times for figures 5.3c, 5.3d and figures 5.3e, 5.3f. With duplicated cores, it is possible to visually distinguish between enabled and disabled.

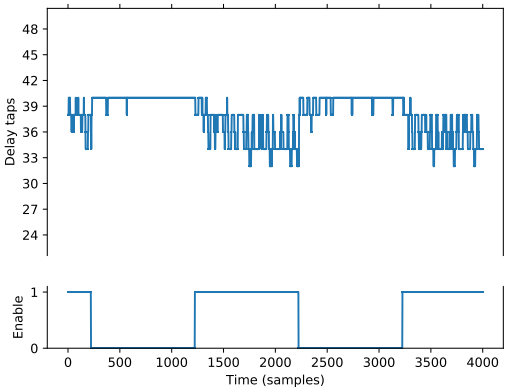
5.4 Discussion

Measuring large amount of ROs clearly indicates that using FPGA resources has an impact on propagation delay. The results show that using a TDL, it is possible to measure this effect under certain circumstances. When measuring registers, the clock speed was lowered from 50 MHz to 25 MHz to reliably measure a difference. The clock speed was kept at 25 MHz when measuring the AES core. Compared to the literature [6, 7], our side-channel leakage is less pronounced.

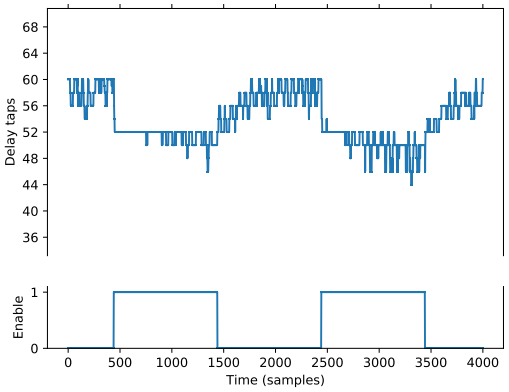
There are several factors that could have influenced this result. First of all, Schellenberg et al. [6,7] use a SAKURA-G FPGA board specially designed for side-channel attack research, though they claim to achieve similar results on an Artix-7 and Zynq-7000 development board. One major difference between these boards and the VC707 board used in this work is size. The VC707 features 485760 logic cells compared to less than 100000 for the boards from the literature. Secondly, there are subtle differences between the TDL implemented in this paper compared to the ones in the literature. My sensor uses a chain of open latches for the initial delay line, compared to a chain of LUTs alternating with open latches. It is possible that the LUTs enhanced the delay response. Finally, the tests in this chapter may not ideally represent the sensors performance. The AES core used for the final test consists of a simple 128 bit implementation with most of the computations pre-computed and stored in LUTs. As a consequence, computing different ciphertexts may have similar power traces. However, in our tests it is not even possible to detect if a single core is on or off.



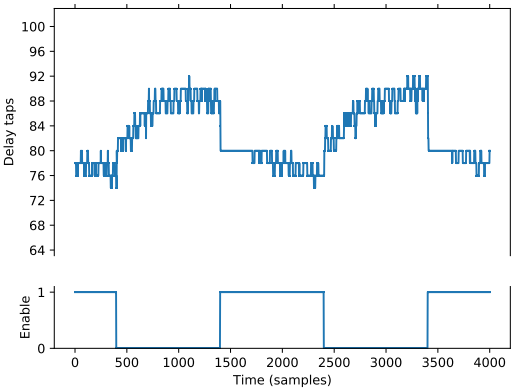
(a) 250 ROs



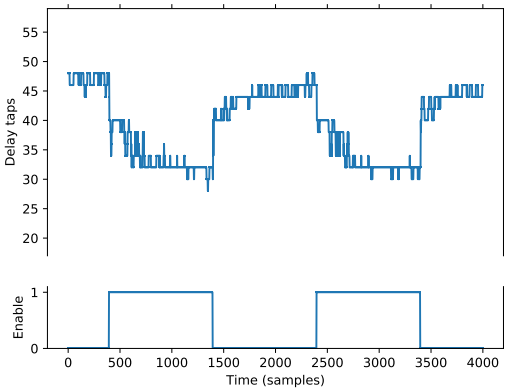
(b) 500 ROs



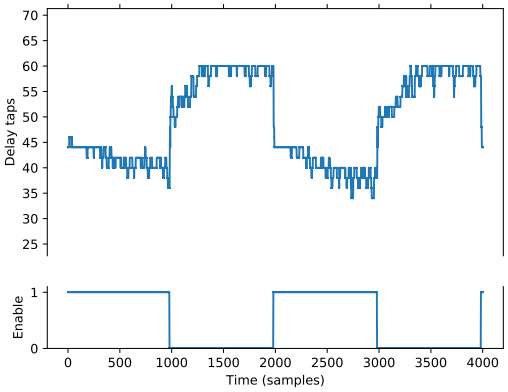
(c) 1000 ROs



(d) 1500 ROs

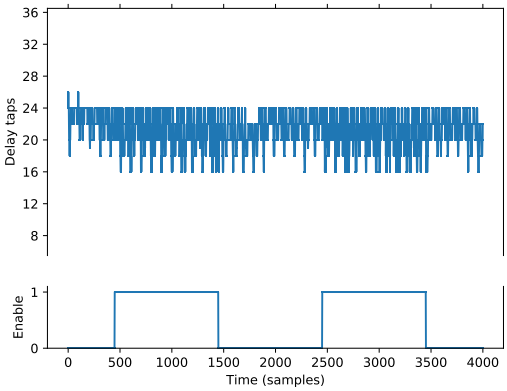


(e) 2000 ROs

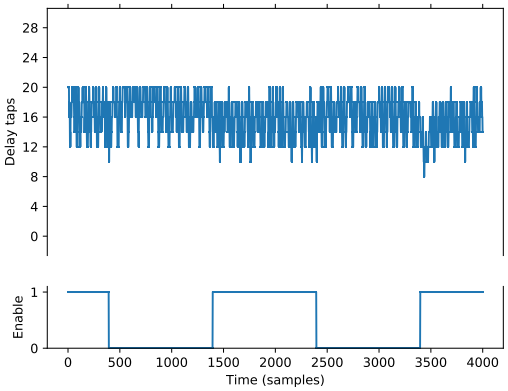


(f) 2500 ROs

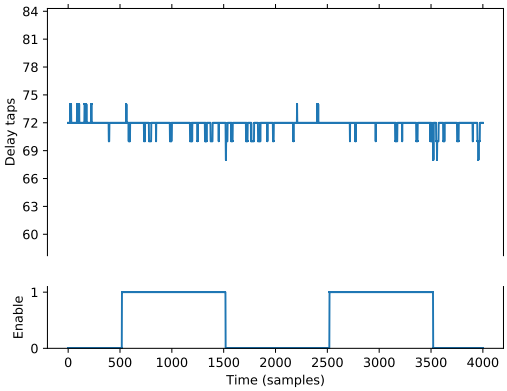
Figure 5.1: TDL delay at 50 MHz from enabling and disabling ROs



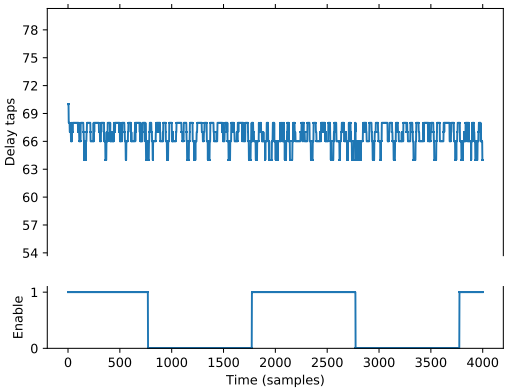
(a) 25 MHz, 2000 registers



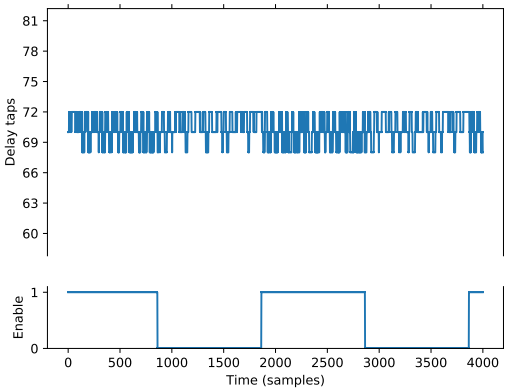
(b) 25 MHz, 4000 registers



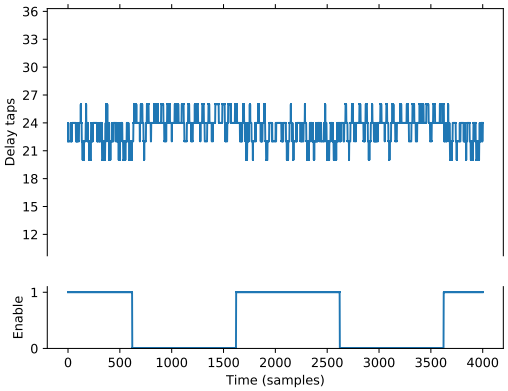
(c) 50 MHz, 2000 registers



(d) 50 MHz, 4000 registers

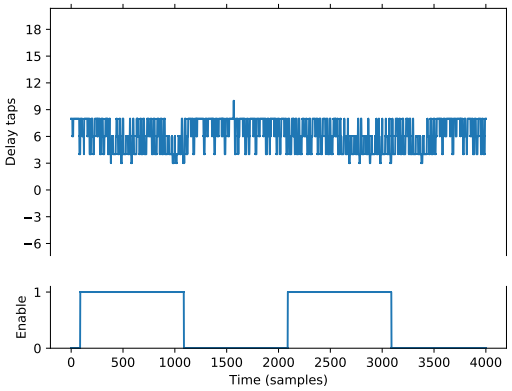


(e) 50 MHz, 6000 registers

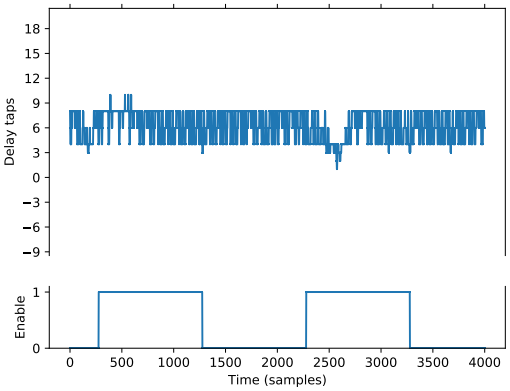


(f) 50 MHz, 10000 registers

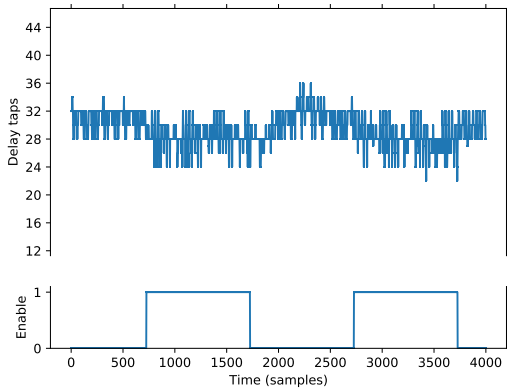
Figure 5.2: TDL delay at 25, 50 MHz from toggling registers



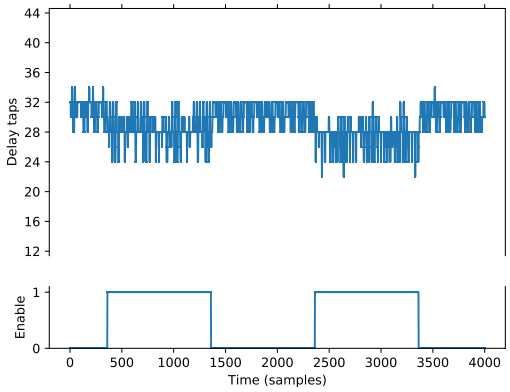
(a) 1 core



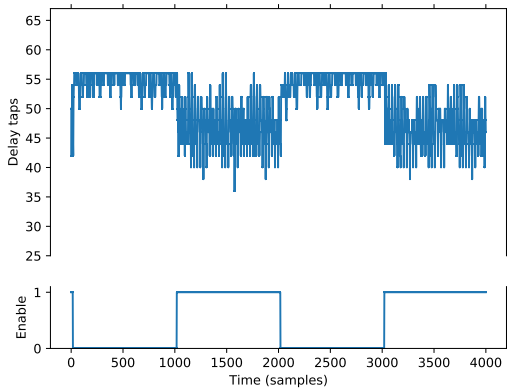
(b) 1 core



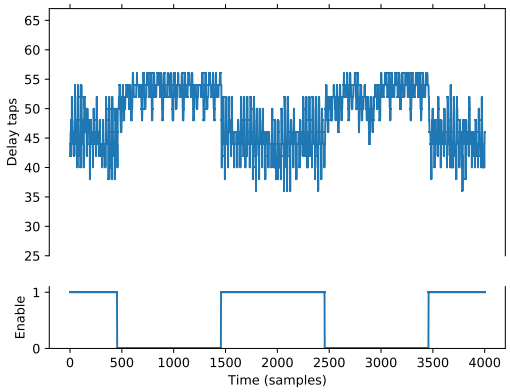
(c) 5 identical cores



(d) 5 identical cores



(e) 10 identical cores



(f) 10 identical cores

Figure 5.3: TDL delay at 25 MHz from enabling and disabling AES core

Chapter 6

Conclusion and future work

In conclusion, a tapped delay line sensor was implemented on a Xilinx VC707 development kit with the goal of measuring its performance at high frequencies. Since this type of sensor requires fine-tuning, an automatic calibration circuit was added to the design. The calibration circuit significantly reduced the development time when regularly switching clock frequencies for testing.

During testing, the effect of various circuits on the FPGA or side-channel leakage turned out to be smaller than anticipated. Compared to the literature, where a small AES core causes a significant effect on the sensor, only a small effect was measured. Since the side-channel decreases as frequency goes up, it was not possible to investigate the viability of power analysis attacks at frequencies over 100 MHz.

Instead, the results suggest other factors may severely impact the performance of this type of sensor, and by extension the risk side-channel attacks pose to tenants in shared FPGAs. To that end, a study comparing different sizes of FPGAs and power filtering circuits for the FPGA could be useful. The VC707 contains a large FPGA compared to the FPGAs typically used to demonstrate power analysis attacks. However, it is quite small compared to the FPGAs offered by cloud providers.

On the other hand, it is not clear what the best architecture is for the initial delay line. A comparison between LUTs, latches and other primitives would be valuable in getting the best signal to noise ratio.

Bibliography

[1] F. Chen, Y. Shan, Y. Zhang, Y. Wang, H. Franke, X. Chang, and K. Wang, "Enabling FPGAs in the cloud," in *Proceedings of the 11th ACM Conference on Computing Frontiers - CF '14*, (Cagliari, Italy), pp. 1–10, ACM Press, 2014.

[2] "Which Intel FPGA devices support Partial Reconfiguration?." `https://www.intel.com/content/www/us/en/programmable/support/support-resources/knowledge-base/tools/2018/which-of-the-intel-fpga-devices-does-supports-partial-reconfigur.html`. Accessed: 2020-05-16.

[3] *Vivado Design Suite User Guide: Partial Reconfiguration (UG909)*, 2018.

[4] C. Ramesh, S. B. Patil, S. N. Dhanuskodi, G. Provelengios, S. Pillement, D. Holcomb, and R. Tessier, "FPGA Side Channel Attacks without Physical Access," in *2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, (Boulder, CO), pp. 45–52, IEEE, Apr. 2018.

[5] M. Zhao and G. E. Suh, "FPGA-Based Remote Power Side-Channel Attacks," in *2018 IEEE Symposium on Security and Privacy (SP)*, (San Francisco, CA), pp. 229–244, IEEE, May 2018.

[6] F. Schellenberg, D. R. Gnad, A. Moradi, and M. B. Tahoori, "An inside job: Remote power analysis attacks on FPGAs," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, (Dresden), pp. 1111–1116, IEEE, Mar. 2018.

[7] F. Schellenberg, D. R. E. Gnad, A. Moradi, and M. B. Tahoori, "Remote inter-chip power analysis side-channel attacks at board-level," in *Proceedings of the International Conference on Computer-Aided Design - ICCAD '18*, (San Diego, California), pp. 1–7, ACM Press, 2018.

[8] J. Gravellier, J.-M. Dutertre, Y. Teglia, and P. Loubet-Moundi, "High-Speed Ring Oscillator based Sensors for Remote Side-Channel Attacks on FPGAs," in *2019 International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, (Cancun, Mexico), pp. 1–8, IEEE, Dec. 2019.

[9] D. R. E. Gnad, F. Oboril, S. Kiamehr, and M. B. Tahoori, "An Experimental Evaluation and Analysis of Transient Voltage Fluctuations in FPGAs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, pp. 1817–1830, Oct. 2018.

[10] "Basic fpga tdc design." `https://cas.tudelft.nl/fpga_tdc/TDC_basic.html`. Accessed: 2020-05-16.

[11] "Tdc on zybo board." https://forums.xilinx.com/t5/General-Technical-Discussion/TDC-on-ZYBO-Board/td-p/817194. Accessed: 2020-05-16.

[12] L. H. Menninga, "Implementation, Characterization, and Optimization of an FPGA-based Time-to-Digital Converter," 2011.

[13] Xilinx, *7 Series FPGAs Configurable Logic Block User Guide*, 2016.

[14] Xilinx, *7 Series FPGAs Clocking Resources User Guide*, 2018.

[15] K. M. Zick, M. Srivastav, W. Zhang, and M. French, "Sensing nanosecond-scale voltage attacks and natural transients in FPGAs," in *Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays - FPGA '13*, (Monterey, California, USA), p. 101, ACM Press, 2013.

[16] P. Kocher, J. Jaffe, B. Jun, and P. Rohatgi, "Introduction to differential power analysis," *Journal of Cryptographic Engineering*, vol. 1, pp. 5–27, Apr. 2011.

[17] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," p. 10.

[18] *Vivado Design Suite User Guide: Implementation (UG904)*, 2018.

[19] Xilinx, *Constraints Guide*, 2013.