Side-channel Testing Infrastructures in Pre- and Post-Silicon Settings

by

Ramazan Kaan Eren

A Thesis Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Electrical and Computer Engineering

May 2022

APPROVED:

Dr. Patrick Schaumont, Major Thesis Advisor

Dr. Berk Sunar, Committee Member

Dr. Shahin Tajik, Committee Member

Abstract

Side-channel analysis methods are growing diverse in practice and more effective in results. Countermeasures against side-channel attacks are able to protect secure hardware devices up to a certain level. Still, some countermeasures are rather hard to penetrate. However, all proposed countermeasures come with a trade-off in area, power, or speed. This situation brings forth the importance of side-channel leakage assessment methods in an effort to understand the origins of the information leakage. For power side-channels, leakage assessment methods range from the early design phases to post-silicon. As the abstraction level of the design goes from early phases to late phases or to post-silicon, the amount and the accuracy of the information that can be extracted from the assessment grows. In addition, a design flaw that is found in later phases can have severe consequences in terms of time and resources to recover from. Therefore, finding a design flaw in the early design phases is advantageous. But, this time, the assessment is short in input material diversity, and this causes inaccurate assessments.

In one way or another, effective tooling can help understand the origins of side-channel leakage. This thesis presents two tools, Side-channel Observer Verification Intellectual Property (SCO VIP) and Saidoyoki, that can help in side-channel leakage assessment efforts in pre- and post-silicon settings. SCO VIP is a functional verification IP that is written for an industry standard, Universal Verification Methodology (UVM). SCO VIP can perform side-channel leakage assessments in register-transfer layer (RTL). Another tool is the Saidoyoki board. Saidoyoki is a highly configurable printed circuit board that houses two in-house designed cryptographic chips and all needed infrastructure to perform side-channel analysis or post-silicon leakage assessment. This thesis also presents two cases in which the capabilities of Saidoyoki and SCO VIP are demonstrated.

Acknowledgements

I would first like to express my deepest gratitude to my research advisor, Dr. Patrick Schaumont, for guiding me with his experience and wisdom throughout this Master's Degree process. Also, I would like to thank my committee members, Dr. Berk Sunar and Dr. Shahin Tajik for their valuable comments and advice.

This thesis was made possible with the collaboration and helps of my friends in Vernam Lab. I would like to thank them and wish them the best of luck in their research.

I would also like to thank Bilal Gece, who is formally my former manager, but inwardly my elder-brother, for his continuous belief and support in me, and in my education.

Finally, I would like to show my warmest gratefulness to my family for their wise decisions they made for me since the day I born, which brought me today as the person I am, their irreplaceable love and their unconditional support. It is not possible to pay back the value they gave me.

Contents

List of Figures

List of Code Listings

Chapter 1

Introduction

Information security has always been a matter of high importance in a timeless manner. When a piece of information needs to be transferred from one point to another, some precautions are usually put into effect to protect the sensitive information from falling into unwanted hands. Before the modern ages, where the advances in the technology-dominated the information security, there were still prominent methods for changing the structure of the information for keeping a possible adversary action unsuccessful [1]. With the advances in technology, means of data transfer have evolved into a combination of mathematics and electronics, and so are the protection methods.

For more than a century, sensitive information has been transferred using several different methods or combinations of them. These methods usually employ electric, electromagnetic (EM), or optical signals. A piece of information is encrypted at one end, sent over the communication channel, and decrypted at the other end. The aim of cryptography is to protect this data, on the way, from anyone who is not the intended recipient. So naturally, encrypted data has been the subject of countless attempts to extract sensitive information. A renowned incidence of an attack on a piece of encrypted information was carried out by Rejewski on Enigma Cipher in 1938 [2]. Although this development was going to be a miraculous advancement for humanity, it still showed a flaw in protecting sensitive data.

Getting closer to today, Data Encryption Standard (DES) became the first cipher to be standardized by the US National Bureau of Standards (NBS), today known as the National Institute of Standards and Technology (NIST), in 1977 [3]. DES is a symmetric-key block cipher [4] that takes a plaintext of 64 bits and a key of 56 bits as inputs and produces a ciphertext of 64 bits. Later, DES is proven to be vulnerable to brute-force attacks because of its short key length [5]. Another well-known cipher is Advanced Encryption Standard (AES), proposed in 1999 and adopted by the NIST in 2001 [6]. AES is also a symmetric-key block cipher that operates on 128-bit plaintext blocks and produces 128-bit ciphertexts. For AES, key size can be 128-bit, 192-bit, or 256-bit. AES was long enough in the key length to prevent brute-force attacks. However, many methods have been proposed not long after its publication to extract the secret key from a device while performing AES encryption or decryption.

Among several attack methods, the group of side-channel attacks (SCA) is prominent for extracting the secret key from a cipher. Side-channel attacks leverage the means of information leakage from the cryptographic device captured in the form of measurable quantities of power, electromagnetic field, time, heat, or light. In side-channel attacks, the adversary records one of the mentioned quantities of the cipher many times. After, they apply a statistical method to find the relation between the measurements and the secret key — e.g., Correlation Power Analysis (CPA) or Differential Power Analysis (DPA) [7].

As the attack methods grew more robust in effect and more efficient in time, the need for testing against information leakage increased as well. As a result, side-channel leakage assessment emerged as an essential part of hardware security research. Test Vector Leakage Assessment (TVLA) is one of the most frequently used methods for testing the cipher in the post-silicon phase [8]. However, finding a flaw in the cipher in the later phases of the design flow is a costly practice in terms of time and money. To address this concern, researchers proposed new methods for testing at gate level (GTL) and register transfer level (RTL). Recently, Yao et al. proposed Architecture Correlation Analysis (ACA) to identify sources of side-channel leakage in GTL [9]. Another work is RTL-PSC from He et al., which focuses on finding vulnerable areas in an AES cipher at the earliest design phase, RTL [10]. Many other methods were also proposed focusing in different aspects of side-channel leakage assessment [11, 12, 13].

When it comes to electrical measurement, several hardware platforms exist. Usually, to observe the power side-channel, an integrated circuit (IC) containing a cipher, or a Central Processing Unit (CPU) performing a cipher algorithm, is monitored for used power in the process of encryption or decryption. After enough traces are collected, an attack method can be employed. Side-channel leakage testing boards usually employ a Field Programmable Gate Array (FPGA) as their target unit. SAKURA-G is one of the well-known hardware platforms for hardware security research [14]. Although FPGAs are useful for flexibility, they come with handicaps of not using the original logic fabric of the design and, consequently, not having control of low-level circuitry.

To address the points mentioned above, this thesis presents two infrastructures for pre- and post-silicon settings of side-channel leakage testing to improve the quality of the test in simulation and measurement and presents implementation examples of the proposed tools. First, side-Channel Observer (SCO) Verification Intellectual Property (VIP) is a Universal Verification Methodology (UVM) VIP that closely monitors toggle counts of the design being tested and provides useful data that can be used in side-channel assessment in RTL. Another infrastructure is Saidoyoki. Saidoyoki is a printed circuit board (PCB) that houses two in-house designed cryptographic Application-Specific Integrated Circuits (ASIC) and provides power measurement and configuration flexibility. With developing SCO VIP and Saidoyoki, this thesis presents the following contributions:

- Toggle counting side-channel leakage assessments are usually meant to be part of the traditional design flow. SCO VIP is designed within UVM. This brings all the design-reuse features of UVM while making a side-channel assessment tool comply with the well-defined design flow.
- An implementation of SCO VIP is made by applying RTL-PSC methods. As a result, SCO VIP finished its analysis in less than a minute, while RTL-PSC took 30-40 minutes.
- Saidoyoki board houses ASICs as its test chips. Unlike FPGA designs implemented on FPGA fabric, ASICs make use of the original logic

design, which is more accurate and free of additional unwanted behavior caused by FPGA fabric.

- Saidoyoki provides easy usage for researchers. Using only one cable, it is possible to configure, program, and control the board.
- The power measurement capabilities of Saidoyoki are highly configurable. It has an optional current probe port, a Low Noise Amplifier (LNA), and a set of shunt resistors to select among.
- Saidoyoki has a variety of clock sources for the test chips. For example, a test chip can receive its clock signal from an on-board clock generator, an SMA connector, or a header.

Outline. The rest of this thesis is structured in the following order. A background discussion on hardware security, side-channel leakage and attack methods, side-channel leakage assessment in different design phases, and UVM is made in Chapter 2. Chapter 3 presents the details of SCO VIP. In Chapter 4, the Saidoyoki board, its capabilities, and its versions are presented. Implementations for SCO VIP and Saidoyoki are presented in Chapter 5. Lastly, Chapter 6 concludes this thesis.

Chapter 2

Background

When a piece of information is desired to be secured, the best practice for gaining this security is to change its structure and content. Cryptographic algorithms are used to convert a plaintext (input data) to a complete non-sensible/non-readable ciphertext (output data). There are many different ways that cryptographic algorithms follow to reach this kind of complexity. Although it is not possible to manually extract any practical information from the ciphertext, it is possible to break the security layers of almost every cipher with enough theoretical knowledge and dedicated equipment. The possibility of losing security levels made side-channel testing/verification research necessary.

This chapter provides the necessary background that the core work of this thesis relies on. The first section presents a general review of the commonly used ciphers, types of side-channel leakage, and the methods used to reveal the secured information. The next part discusses current side-channel assessment techniques in pre-silicon design phases, GTL and RTL. After this comes the post-silicon, when a physical device is present, and the ways of testing its side-channel vulnerabilities. Lastly, an industry-standard design verification technique, UVM, is reviewed.

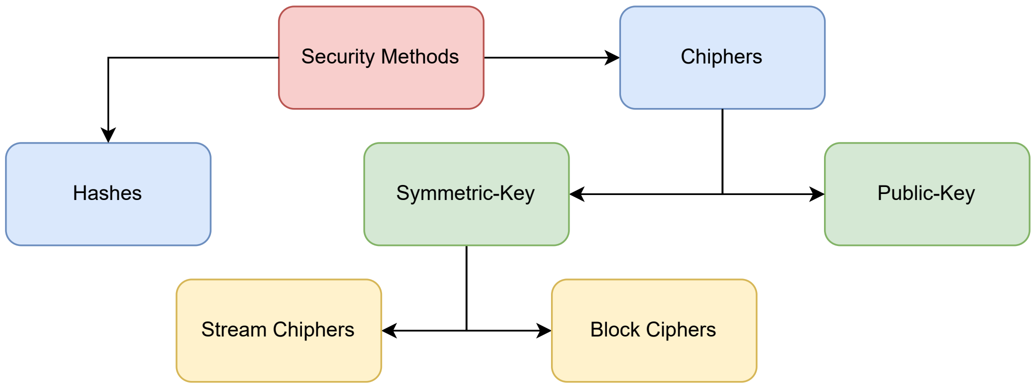


Figure 2.1: Cryptographic algorithms.

2.1 Hardware Security and Threats

2.1.1 Methods for Securing a Hardware

Modern cryptographic methods can be gathered under two main sections, ciphers and hashes. Both ciphers and hashes turn input data into non-readable output data. However, ciphers and hashes differ in using a secret key or keys. While ciphers use secret keys and plaintext to perform the cryptographic operation, hashes generate the output data without a secret key. Hashes are commonly used to generate secret keys [15]. Figure 2.1 shows a general classification of ciphers and hashes.

This thesis mainly focuses on cipher applications. Ciphers can be further split into two sub-sections: symmetric-key ciphers and public-key ciphers. Symmetric-key ciphers make use of one secret key for both encryption and decryption. Both the sender and receiver end of the communication must have the same secret key to use a symmetric-key cipher. In public-key ciphers, this situation is more sophisticated; when a receiver wants to receive data from a sender, it creates a pair of keys, a public key, and a private key. A piece of data that has been encrypted with the public key can only be decrypted using the other pair, the privet key. Visual explanations of symmetric-key and public-key methods can be seen in Figure 2.2. Symmetric-key ciphers are also split into two; block ciphers and stream ciphers. A block cipher encrypts data in chunks called 'block' while a stream cipher encrypts the streaming plaintext bit-by-bit.

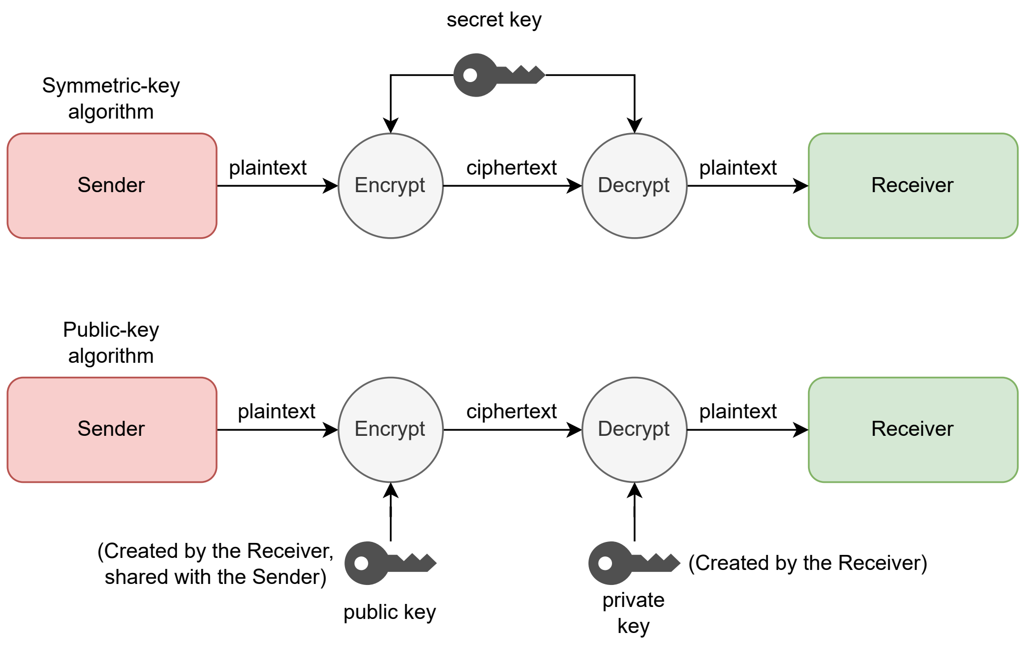


Figure 2.2: Symmetric-key and public-key methods.

Advanced Encryption Standard (AES) is a symmetric-key block cipher that is being highly used by hardware security researchers and is also the core algorithm that the DUTs are being used in this thesis. The input block of the AES consists of 128-bit plaintext in the form of sixteen bytes. Bytes of the plaintext are placed in a 4x4 matrix, also called '*state*,' and the encryption operations are performed on this matrix in rounds. AES state matrix can be seen in Figure 2.3. AES has three different versions in terms of the key size. The ciphertext is calculated after the following number of rounds based on the key size:

- 10 rounds when AES key size is 128
- 12 rounds when AES key size is 192
- 14 rounds when AES key size is 256

b0	*b1*	*b2*	*b3*
b4	*b5*	*b6*	*b7*
b8	*b9*	*b10*	*b11*
b12	*b13*	*b14*	*b15*

Figure 2.3: AES state matrix.

AES encryption starts with a **KeyExpansion** operation in which round keys are generated from the main key using AES key schedule [6]. In each round, several operations are performed on the current state. These operations are as follows:

- **AddRoundKey:** At the very beginning of the encryption and at the end of each round the state is combined with the round key with a bit-wise XOR operation.
- **SubBytes (substitution):** SubBytes operation is where each byte in the state array is replaced with another byte derived from a substitution box (S-box) [16]. SubBytes step adds non-linearity to the cipher.
- **ShiftRows:** In this step, each row of the state array is shifted to left in bytes for the following number of times:
  - Row 1: No shift
  - Row 2: 1
  - Row 3: 2
  - Row 4: 3
- **MixColumns:** In this step, each column of the state is multiplied with a fixed matrix. This operation, together with ShiftRows provides more diffusion to the encryption.

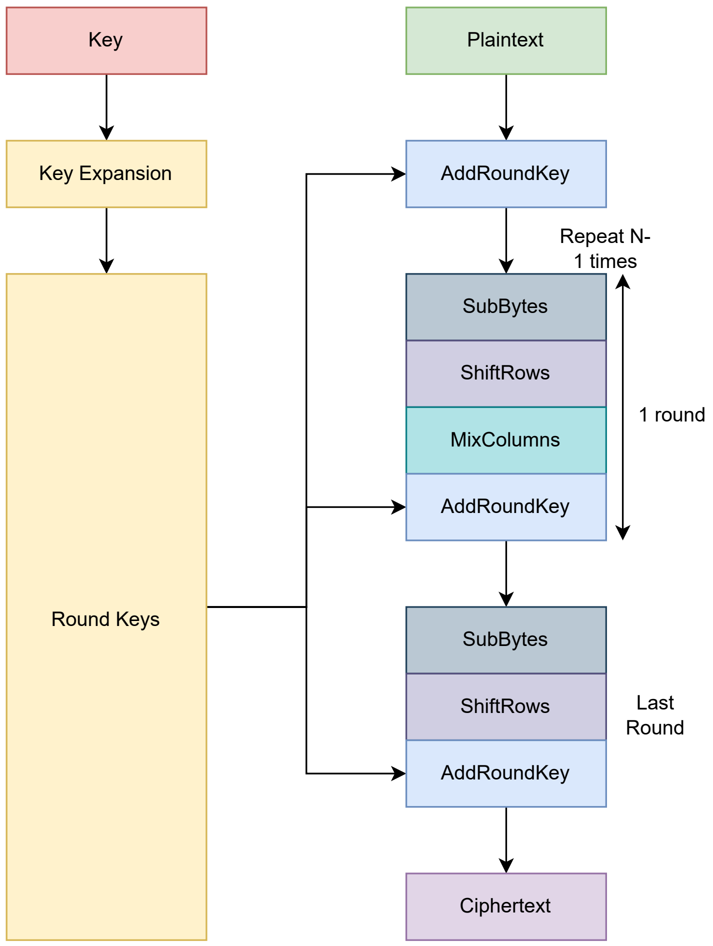


Figure 2.4: AES rounds in encryption.

After the KeyExpansion, an initial AddRoundKey operation is performed to combine each byte of the state with the bytes of the round key. Then, for *N-1* times, where *N* is the round number based on the key length, the above-explained steps are performed in the following order: SubBytes, ShiftRows, MixColumns, AddRoundKey. For the last round, the MixColumns step is skipped. At the end of the encryption, the state array is named ciphertext. Figure 2.4 provides a visual reference for the overall AES encryption process.

2.1.2 Side-channel Leakage in a Secure Hardware

Secure hardware designs aim to keep a sensitive piece of information out of the reach of adversaries. Extracting this sensitive information from encrypted data is visually impossible. However, secure devices tend to present hints of the data while the cryptographic algorithm is performing. When a cryptographic operation is underway, several other channels can be observed with the possibility of revealing a piece of critical information. These channels are called side channels. Some of the well-studied side-channels discussed here are power consumption, EM radiation, and operation time.

Power-side channel leakage is the one that has been extensively used in cryptanalysis. The roots of power-side channel leakage can be explained by analyzing the factors that contribute to the device's power consumption. The power consumption of a digital circuit originated from the power consumption of each transistor in the device. A transistor's power consumption, as seen in Equation 2.1, consists of the power consumed due to short-circuit, current leak, and switching activity. Among them, leaking current and short-circuit consumption are the static ones. Yet, the power consumption caused by the switching activity changes in time, based on the number of transistors switched from 0 to 1 or vice versa. When measuring the power consumption, a component of noise from the measurement equipment or the circuitry itself also adds up to the total. Equation 2.2 shows the measured consumption.

$$P_{total} = P_{short-circuit} + P_{leak} + P_{switch} \tag{2.1}$$

$$P_{measurement} = P_{total} + P_{noise} \tag{2.2}$$

As the switching number is an essential part of the overall power consumption, seeking a relationship between the power consumption and the secret value lays the foundations of power side-channel attacks.

Another information leakage is called EM side-channel leakage. Electric current flowing in a metal wire creates an electromagnetic (EM) field. It is proven that EM traces recorded from a secure design can also reveal secret information [17].

Timing is another quantity that can be observed to read out the secret data. In digital circuits, every logical operation is carried out with different circuit structures, called gates. For example, the number of transistors used in an AND gate is different from the XOR gate. Also, the composition of these transistor networks is different from one to another. This difference brings a distinction in the signal delay times of different operations. This distinction is often exploited in cryptanalysis [18].

2.1.3 Revealing the Secret Key

Methods for measuring a side-channel and analyzing these measurements are diverse. Side-channel attacks are well-defined ways to obtain side-channel measurement and statistical methods in the gathered data to obtain the secret key. In this section, power side-channel attacks are discussed.

Power side-channel attacks are based on making observations on the power consumption of the target device while it is performing the cryptographic operation and estimating a relation between the power consumption and the secret key. This relation is often estimated using a power model. A power model predicts the approximate level of targets power consumption calculated from the known plaintexts. There are different power models used in cryptanalysis, among which Hamming Weight (HW) and Hamming Distance (HD) are the most used ones. Hamming Weight power model counts the number of bits in the high logic state at an output of a computation block; usually an S-box, to make an assumption on the power consumption. Hamming Distance is another power model that counts the number of bits that have switched from the input to the output of the target block in the device. For AES, the S-Box operation of the first round leaks the most information. Based on that, Figure 2.5 shows an example of how HD and HW values are calculated for an attack on an AES cipher. In this example, the hex value at the output of the S-box is FB. The number of high logic bits, and so the Hamming Weight, in FB, is 7. The number of bits that change their state from the input to the output, from 63 to FB, and so the Hamming Distance, is 3.

Side-channel analysis can be made by following one of the attack methods. One of these methods is Simple Power Analysis (SPA) [19], which is a visual

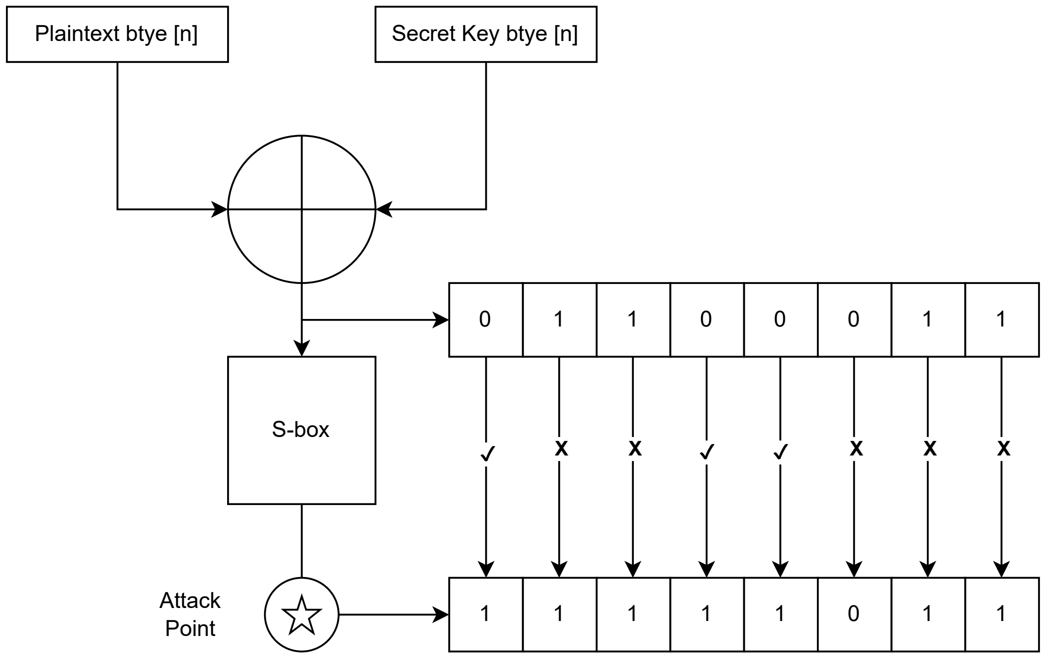


Figure 2.5: Hamming Weight and Hamming Distance calculation for AES.

method to predict the secret key. For more advanced methods, Differential Power Analysis (DPA) and Correlation Power Analysis (CPA) can be counted [20, 21]. DPA and CPA methods make use of statistical analysis of the collected data. This thesis focuses on the CPA method.

CPA method relies on the statistical correlation analysis between the hypothetical power consumption, HW or HD, and the measured power consumption. For AES, hypothetical power consumption is the HW or HD values at the end of the first round S-box, which includes the plaintext and the secret key. Hypothetical power consumption is calculated for every possible key value (key guess). This set of values, along with the measurement results, are used for the calculation of the Pearson Correlation Coefficient using the Formula 2.3, where H and T are the hypothetical power consumption and the measurement values, respectively.

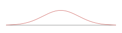
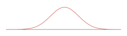
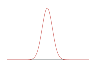
Key Guess No.	Hypothetical Power	Symbolic Measurement	Correlation
1	4		Low
2	2		Low
3	3		High
4	4		Highest

Figure 2.6: A symbolic comparison of correlation values of different key guesses between hypothetical power and average power measurement trends.

$$\rho(H,T) = \frac{cov(H,T)}{\sqrt{V(H).V(T)}} \tag{2.3}$$

After this, the results are expected to show a distinctive difference for one key guess among all the other possible ones. This is because when a correlation coefficient for a key guess with a higher value of hypothetical power consumption and an actual measurement with a high power value is calculated, the result becomes high too. In the case of the correct key, the coefficient value will be the highest. Figure 2.6 is an example of this comparison, in which the key guess number one will be the most likely prediction.

2.2 Pre-Silicon Leakage Assessment

When there are many proven attacks against cryptographic devices, it is necessary to build countermeasures against them. Countermeasures aim to eliminate the relation between the secret key and the measurable side-channels. They mainly developed in two ways, hiding countermeasures and masking countermeasures. Hiding countermeasures are developed to hide the means of the side-channel leakage. This can be achieved, for example, by introducing additional noise into the system. Randomization techniques are often used to create an artificial noise to reduce the Signal-to-Noise Ratio (SNR) [22]. Low SNR reduces the correlation between the secret key and the measurement. Masking countermeasures are based on the idea of splitting the sensitive information into a number of shares so that the attacker needs to find all of them to run a successful attack [23]. Although both countermeasure methods are effective, they introduce significant overheads in design resources. For example, a countermeasure by Das et al. introduces 1.63 times overhead in power and 1.25 times in area [24]. Yet, none of these countermeasures are proven to be non-breakable.

As a result, it is a necessary effort to find the source of the leakage. Once the source of the leakage is found, countermeasures can be applied to a smaller area of the design so that the overhead can be much smaller, or the source of the leakage can be eliminated with an update in the design. Side-channel leakage assessment can be made in different phases of the design process. While the assessments made in the early design processes are relatively faster and requires low resource, they are usually not accurate enough. Assessments in the later design phases can reveal more secrets about the leakage. However, they are relatively more complicated, and finding a security flaw in a later design phase is costly in time and resources. This is an obvious trade-off to be made. To address this trade-off, this section makes a discussion on different methods of side-channel leakage assessment.

2.2.1 Leakage Assessment in RTL

Side-channel leakage assessment in the register-transfer layer is a limited research area. This is because the ways to predict the actual power consumption are limited and very abstract. Yet, finding a security flaw in the early design phase is very desirable as it would be fixed quickly. As power side-channel leakage originates from the hardware's overall power consumption, in RTL, predicting the power consumption primarily relies on the known switching signals. As mentioned in Equation 2.1, power consumption caused by the switching activity is the most critical factor in the measurement. It is possible to count the switching numbers using an RTL simulator software when a cipher is in operation and predict some information from it. Still, simulations are free of real-life noise and other low-level elements like glitches. This is the reason for RTL leakage assessments' non-accuracy.

In [10], He et al. proposes a framework named RTL-PSC that counts the switching number in a simulation of AES and then utilizes Kullback-Leuber (KL) divergence and success rate (SR) metrics based on maximum likelihood estimation to power side-channel analysis. KL divergence is a method used to estimate the statistical distance between two different probability distributions. After collecting the switch count numbers for 1000 encryption cycles of AES, RTL-PSC derives two probability distribution functions. One function is derived from the 1000 cycles of encryption when the key is all zeroes. The other one is derived when the key is all ones. This practice is followed to obtain the largest Hamming Distance between the simulations. RTL-PSC assumes that these functions follow Gaussian Distribution. KL divergence then is calculated between these two functions for each clock cycle and the design unit. High values in the results are suggested to leak more information than the others. The other metric, SR, assumes that the adversary usually selects a key guess that gives the highest maximum likelihood value. Based on this, SR calculates the probability that the chosen key is the correct key. Finally, RTL-PSC sets a threshold value that names them as vulnerable if one or both of these two metrics cross. Inspired by RTL-PCB, the Side-Channel Observer (SCO) VIP, which will be introduced in Chapter 3, implements the KL divergence metric in Chapter 4.

Another side-channel leakage assessment method in RTL is PSC-TG [25], proposed by Zhang et al. This method, unlike the other RTL assessment methods, uses the RTL Information Flow Tracking (RTL-IFT) method instead of toggle counts of the internal design signals. It is also claimed to be effective against masked implementations. By utilizing IFT, PSC-TG does not need a large number of simulations to lower the number of possible exceptions. Instead, it takes the design, the input/output ports to be tracked, the masking information (order), and the attack power model (HW or HD). PSC-TG then uses formal verification methods (assertions) to derive test patterns leading to a maximum leakage. Using these test patterns, they come up with a side-channel vulnerability (SCV) metric for a non-protected design and a pass/fail output for a masked design.

2.2.2 Leakage Assessment in GTL

In the gate-level (GTL) design phase, side-channel leakage assessment provides more accurate results than RTL. This is because GTL is closer to the real-life conditions where the design actually appears in logic cells and wires between them. Still, GTL properties do not include noise but other low-level impacts such as glitches and physical routing. Examples of gate-level assessment methods can be ACA proposed by Yao et al. [9], GLIFT proposed by Oberg at al. [26] and Co-Co proposed by Gigerl at al. [11].

ACA method proposes a technique that is able to analyze and rank logic cells in a gate-level design based on their contribution to the power side-channel leakage. In this method, they aim to localize the position of a leaky cell in the design so that the design engineers can implement a local solution with the advantage of decreasing the overhead caused by the countermeasure. The proposed Leakage Impact Factor (LIF) is calculated for a secure hardware design, and the cells are ranked based on their LIF values. As a result, ACA shows that only a very small number of logic cells actually contribute to a side-channel leakage.

Oberg et al., like PSC-TG in RTL, uses information tracking methods but at the gate-level. In their work, the GLIFT framework detects timing-dependent information leaks. They also present a method to isolate timing information from other information flows toward solving this problem.

Co-Design and Co-Verification (CoCo) proposed by Gigerl et al. leverages formal verification methods to verify a masked software implementation's robust and secure operation on CPUs. Typically, software implementations are more prone to leak information than hardware. Based on this, masked software is used to separate the shares of sensitive information and perform the cryptographic operation in a decentralized manner. However, CPUs often tend to break this rule. CoCo analyzes the execution of a masked software on a gate-level net-list of a CPU. It reports the leakages with the specific gate and cycle information where two or more shares of a sensitive information may combine together.

2.3 Post-Silicon Leakage Assessment

Post-silicon side-channel leakage assessment usually refers to statistical methods, such as Test Vector Leakage Assessment (TVLA), checking if the sensitive variables of the cryptographic operation significantly affected the side-channel measurements.

TVLA is a post-silicon tool that relies on Welch-s T-test to assess whether the cryptographic implementation is safe against side-channel attacks [27]. The T-test itself is used to determine if two sets of data are significantly different from each other [28]. TVLA is implemented in two different ways: general and specific tests. In the general test, the secure hardware is first run with a specific fixed input test vector and then run with a random input test vector. Each of these runs are made for enough times to collect a robust data set. These data sets are then used to calculate t-values. T-values that are higher than a specific value are considered as a leak. Specific tests are made using random vs. random data. They are used to target a leakage for a specific sensitive value. A failing general test indicates a possibility of leakage, while a failing specific test indicates leakage that is immediately exploitable. As a result, TVLA outputs can be used as a tool to determine leakage in the post-silicon level. However, it should be considered that TVLA can output false negatives or false-positive results.

2.4 Universal Verification Methodology

Universal Verification Methodology (UVM) is a standard in functional verification of digital circuits that has been widely used by design verification engineers [29]. UVM is known for its ability to provide strong reusability features and develop functional verification environments in a fast and efficient way. The main idea of UVM is to enable companies to build their modular, reusable, and efficient testbenches.

UVM is developed as a framework written in SystemVerilog in the form of a set of base class libraries. UVM highly takes advantage of object-oriented programming. Packaging a functionality and using it later where it is needed is an essential concept in UVM. Typically, every testbench needs basic components like drivers, monitors, stimulus generators, and checkers. A verification engineer can write all these components and perform the verification process. However, it is possible to write these mentioned components in countless ways. UVM deals with the confusion that this older practice brings. A functionality for a digital interface is implemented in the structures named agents. Agents are connected to both the related parts of the DUT and to the rest of the testbench. For example, an agent can implement the protocol of Serial Peripheral Interface (SPI). The agent, then, is responsible for acting as a formal SPI endpoint.

Such an agent is expected to be able to start transactions, accept transactions, manage bus specifications, observe and report the interface's state, and report flaws in the bus protocol. Figure 2.7. shows an agent structure. An agent usually has three sub-components; monitor, driver, and sequencer. The sequencer receives the sequence items (command pieces) from a sequence object and delivers them to the driver whenever the driver requests a new item. The driver implements the low-level protocol to be followed. For example, when an SPI agent's driver components receive a sequence item requesting one byte written to a register in the DUT, the driver drives the logic ports to select the slave device (DUT), sends the register address, and the data after this. While these operations occur, an observer component records every movement in the interface and reports this information to another component outside of the agent, usually to a scoreboard.

A UVM environment is the component that involves every feature needed in a verification protect. It includes a preliminary structure for basic communications to DUT, agents of needed protocols, scoreboards for checking correct operations or functional coverage collectors to assess the functionality if it is covered in the tests. The UVM environment is instantiated in a UVM test. Each UVM test instantiates the UVM environment, configures its components and agents in the way of its usage, and starts the runtime tests. This is one of the main contributions to reusability. For example, when verifying a System-on-Chip, a test for the CPU and another test for the GPIO module configure and use the same environment instead of having their own. Encapsulating the test, consequently the other components too, UVM test top is the top SystemVerilog module that contains the DUT instantiation,

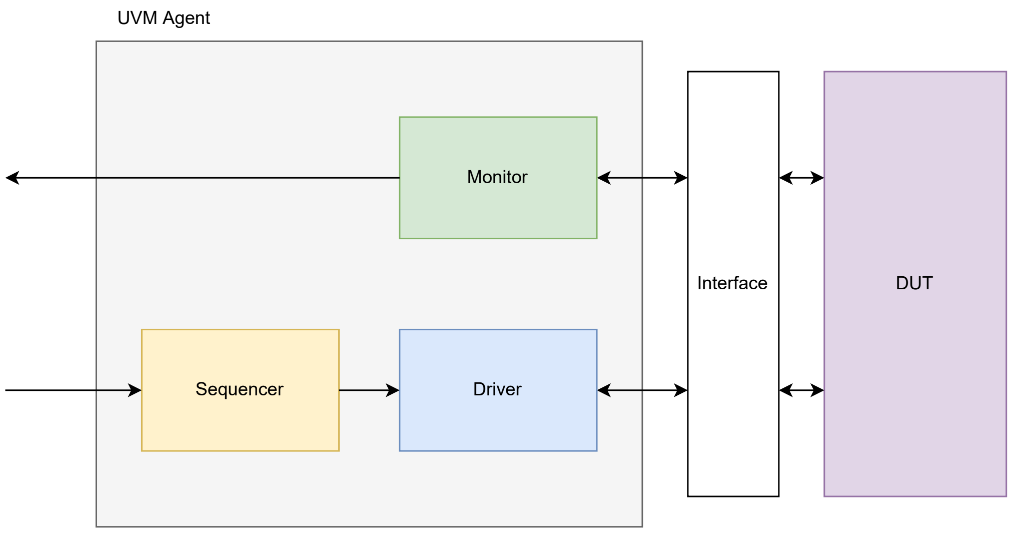


Figure 2.7: A UVM Agent with its sub-components.

clock generation, reset generation, UVM test instantiation, reference models for comparing the results collected from the DUT, or if the the reference model was created using another tool than SystemVerilog, a handle to that tool, such as a SystemC model. Figure 2.8 shows an example UVM testbench structure.

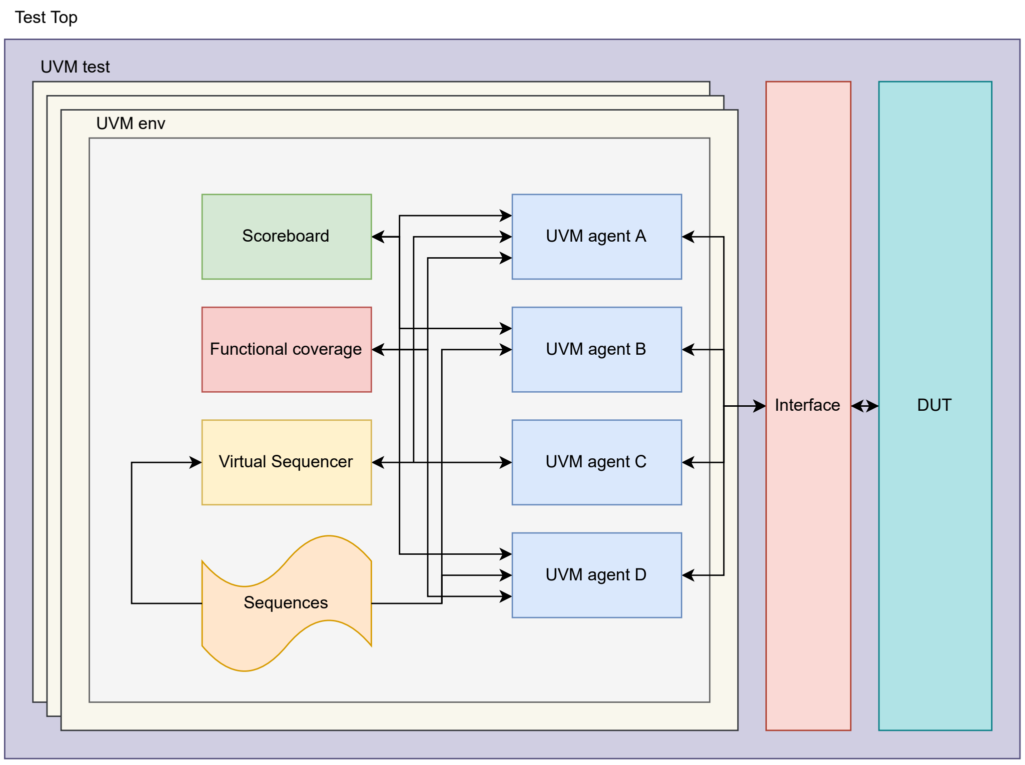


Figure 2.8: An example UVM testbench structure.

Chapter 3

UVM VIP for Leakage Assessment in RTL

3.1 Overview

The idea behind performing a pre-silicon side-channel assessment is to detect information leakage at an early design phase. As the cost of recovering a detected flaw from a late design phase would be so costly in time and resources, previous works in this area are generally proposed methods with the tendency of being compatible with the current design flow followed by almost every chip manufacturer. As an example of this, He et al. in [10], mentions that RTL-PSC, which is a side-channel leakage assessment methodology in RTL based on toggle counting, is developed to be integrated with the traditional ASIC and FPGA design flow. However, traditional chip design flows are well-defined, so making changes is not always feasible. Instead, adding a parallel thread to this process, using its native tools, can be a good idea. Functional verification is a critical process in the chip design flow, and UVM is the most widely used platform for performing this verification. A UVM VIP can be well capable of handling RTL side-channel leakage by collecting and analyzing toggle counts while other VIPs are dealing with the native verification process.

UVM Verification Intellectual Property (VIP) is the term that is used to refer to a UVM Agent and its associated objects. These objects can be sequences, functional coverage models, or functional models. A UVM VIP is usually prepared to implement a specific protocol. It is different from a UVM Agent because, although a UVM Agent is capable of handling low-level signaling with its driver, it cannot work standalone. For example, an agent for a Tightly Coupled Memory does know how to implement the memory writes and reads. However, it needs consumable materials (data and timing information) while performing its function.

In this chapter, a UVM VIP is presented to be used in side-channel leakage assessments in RTL. As side-channel assessments made in RTL are usually base on toggle counting, this VIP (and its agent, SCO) is designed to collect toggle counts from the cryptographic device in an efficient way. The UVM Agent, SCO, is associated with related counters and internal UVM Scoreboard components to evaluate the collected toggle counts without needing an external component to perform a side-channel assessment.

3.2 Verification IP Design

Three sub-component classes form SCO Agent. They are observers, counters, and the internal scoreboard. An observer is extended from the UVM base class, UVM Monitor. There is one observer for every module instantiated in the design. The observers listen to the interfaces to which they are connected and report every movement in the ports of the interface. This reporting is made in the form of time and the state of the port. After the observer stage, counters are tasked to generate meaningful toggle count data from the mere observations made by the observers. To do this, counters stay in touch with external components of the testbench. The external components, for example, can be other agents. The toggle count of a design only needs to be taken while the encryption is ongoing. To understand the right time, counters receive an enable signal, possibly from the agent that is in contact with the design for providing the encryption data and start signal. Also, the number of the clock cycle that the toggle count numbers are taken, and the tag of the encryption number should be provided to the counters block. As a result, an array containing the toggle counts, which are labeled according to their design block name, clock cycle, and encryption number,

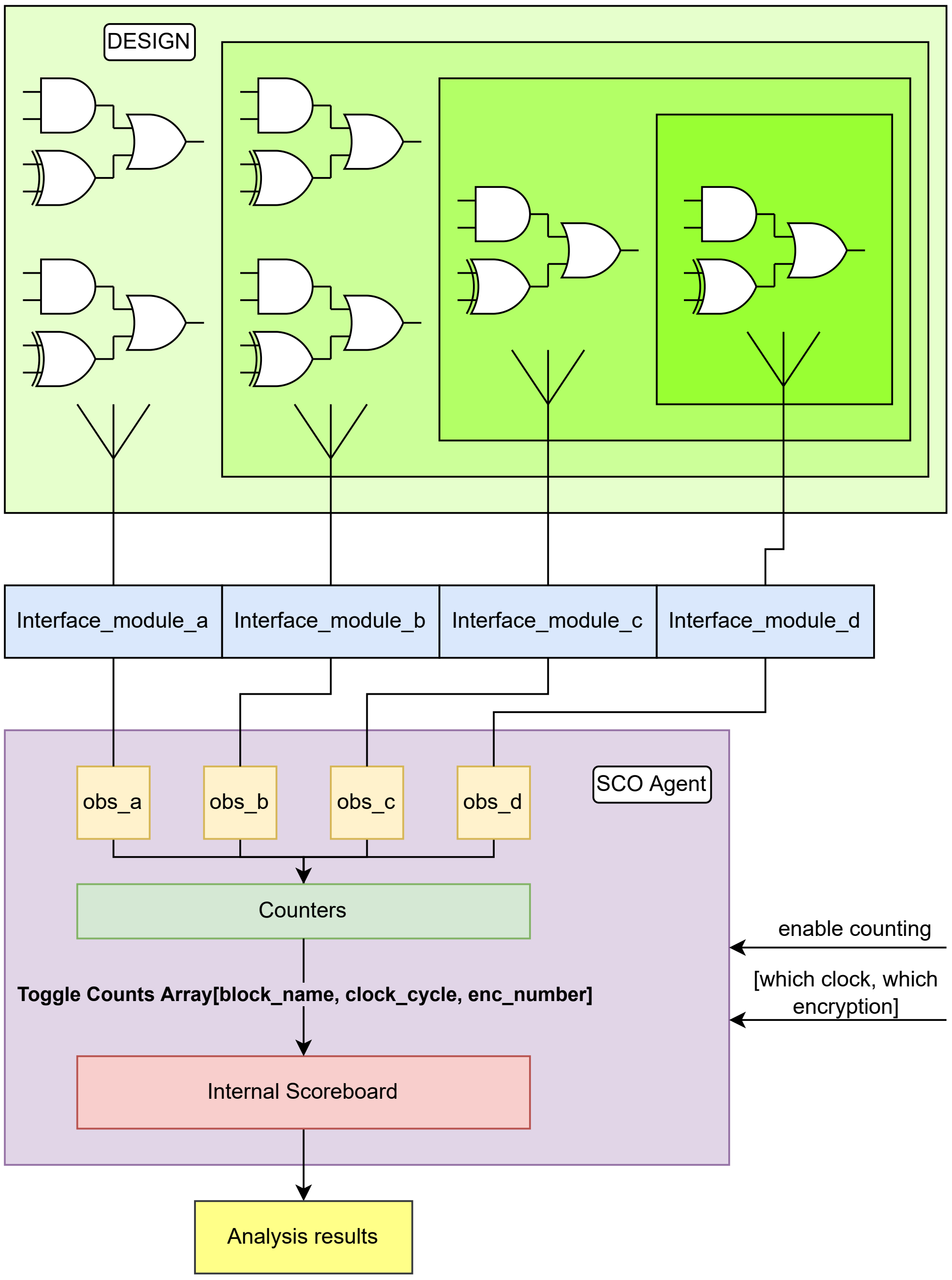


Figure 3.1: UVM Agent, Side Channel Observer (SCO)

is created. Then, this array can be used in pre-silicon toggle counting side-channel assessment applications. Figure 3.1 shows the block diagram of this structure.

Another useful feature of the SCO, thanks to reusability from UVM, is being able to adapt different ciphers. When the signal list of the cipher design is ready, observers can be disconnected from one interface and connected to the other one without the need of designing an agent specifically for one cipher.

3.3 Handling Toggle Counts

When counting signal toggle numbers for side-channel assessments, minor errors in the count numbers can result in wrong assessments. One consideration on this is counting the same net twice. This situation can be originated from the fact that one output of a design module can be the input of another one. When the list of the signals is being made, if all signals appear on the design hierarchy are included, mentioned problem would happen. For example, in Figure 3.2, the signals that are shown by bold lines can be counted for twice.

In order to prevent this confusion, interfaces used in SCO neglected the input ports of the design modules. This is because internal signals of each sub-module can be input into another sub-module too. Therefore, even if the input port of a module is neglected, it was already counted in the one upper-level module. As there is no upper module than the top module, all ports and the internal signals associated with the top module are counted.

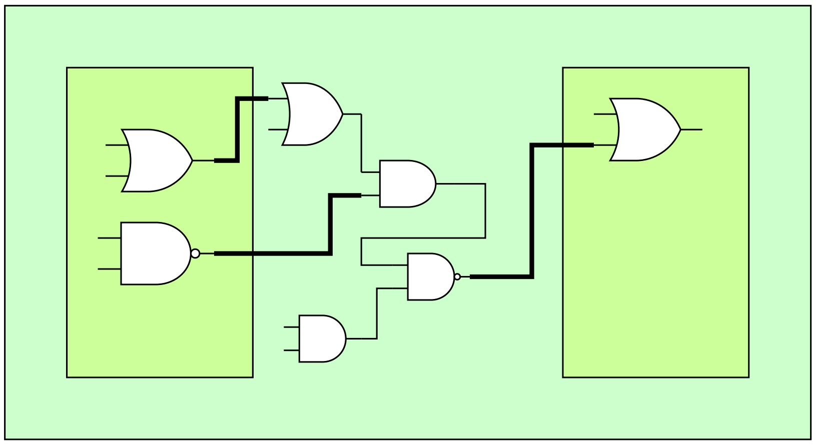


Figure 3.2: Positions of the nets with possibility to be counted for twice in an example circuit.

Chapter 4

Saidoyoki: Post-Silicon Side-Channel Test Platform

4.1 Overview

The most accurate side-channel assessment can be made in the post-silicon setting. This is because all unforeseeable factors that are affecting the measurement values exist in the physical circuit. In RTL, so far, the only useful data seems to be the signal toggle counts, as they more or less represent the dynamic power consumption of transistors. However, RTL lacks most of the other factors that can contribute accurate assessment of side-channel leakage. For example, despite circuit properties being much closer to reality at the gate level, a true noise, which is an important factor in measurements, is still missing.

In this thesis, a post-silicon evaluation tool, Saidoyoki, is presented. Saidoyoki is a printed circuit board (PCB) that houses two in-house designed cryptographic test chips, PICO and FAME. It is designed to be a flexible post-silicon side-channel evaluation tool that can handle all necessary side-channel-related configuration and measurement infrastructures. Saidoyoki can be powered using a barrel jack or a screwdriver terminal. It has four isolated power rails, among which two of them are adjustable in a voltage range. A clock can be supplied to both test chips from an internal clock generator or an external clock input. Saidoyoki can be configured and programmed via a single USB micro-B cable. Power measurement can be done for both

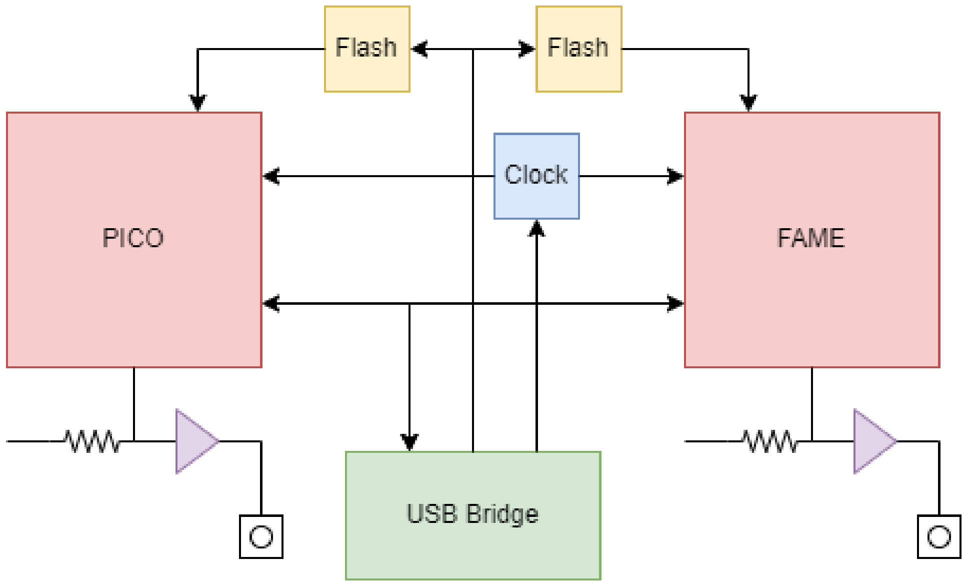


Figure 4.1: Simplified block diagram of Saidoyoki.

chips independently using the optional current probe port or the on-board low noise amplifier (LNA). As Saidoyoki uses ASICs instead of FPGAs, it reveals more concrete facts in side-channel research. Figure 4.1, is a simplified block diagram of Saidoyoki.

Saidoyoki tool has been under development for the last two years. Two versions of Saidoyoki have been developed so far. To prevent confusion, the board mentioned as Saidoyoki in [30] refers to version one. In this thesis, Saidoyoki refers to the most recent design, version two. Figure 4.2 shows an image of the version two board.

4.2 Design Features

4.2.1 Payloads

The Saidoyoki PCB was developed to support the PICO and FAME chips as targets for side-channel measurement campaigns, and it supports all functions mentioned above. Both target chips are ASICs with several cryptographic hardware accelerators. A block diagram of the target chips can be

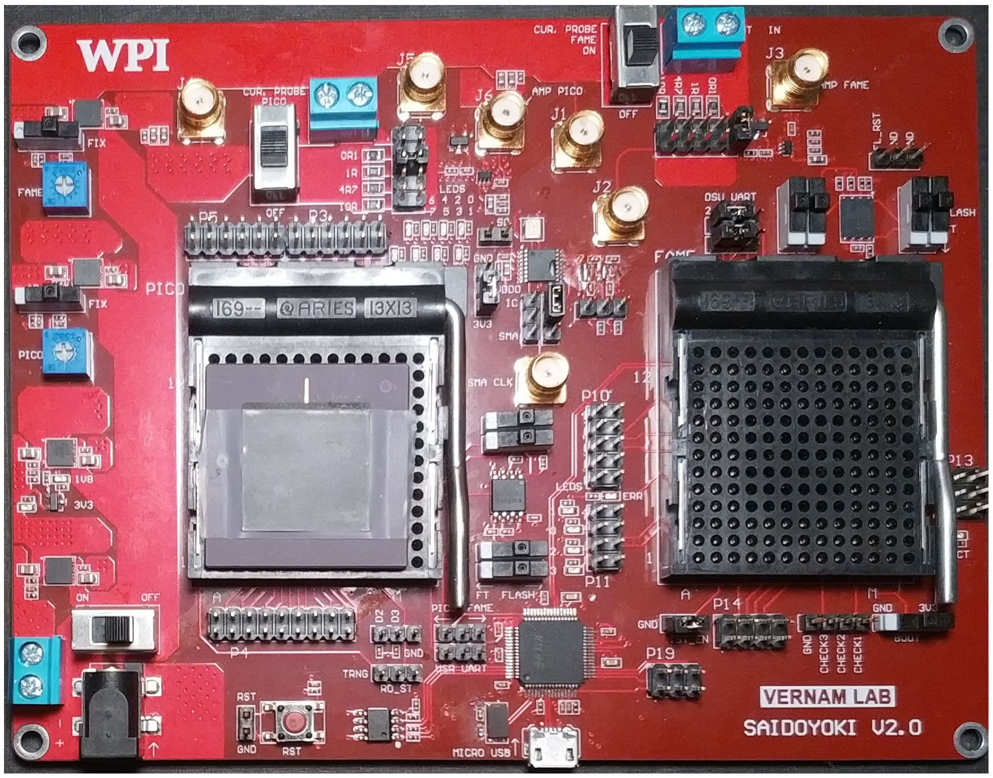


Figure 4.2: A photo of Saidoyoki V2 with a PICO mounted on.

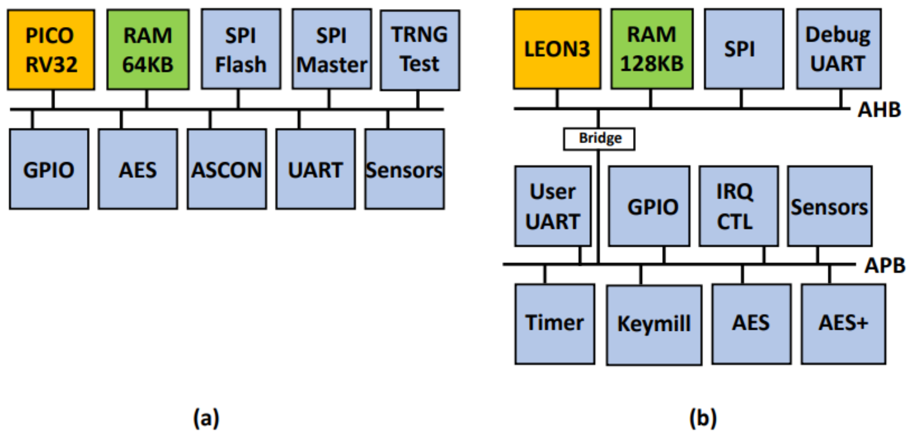


Figure 4.3: (a) PICO Block diagram and (b) FAMEv2 Block diagram.

seen in Figure 4.3.

The PICO ASIC is a 180nm SoC with a RISCV (RV32) core and 64 kByte of internal memory, and several coprocessors. The program exclusively runs from off-chip flash through a Quad-SPI flash ROM. The system is integrated on a single bus. All coprocessors run as bus slaves and communicate with the RISC-V software through memory-mapped registers. PICO contains cryptographic accelerators for symmetric-key encryption (AES), authenticated encryption (ASCON), and hardware testing of true random bitstreams (TRNG test). The sensors in PICO detect fault injection as well as side-channel leakage.

The FAME ASIC is a 180nm SoC with a LEON3 core and 128 kByte internal memory, and several coprocessors. The program can either execute from on-chip SRAM or off-chip flash through an SPI flash ROM. A debug unit, controlled through an on-chip Debug UART, provides program loading, monitoring, and breakpoints. The coprocessors are isolated from the processor through a bus bridge. All coprocessors exclusively operate as bus slaves and communicate with the software through memory-mapped registers. FAME contains cryptographic accelerators for symmetric-key encryption (AES and AES+, a hardened version of AES) and pseudo-random stream generation

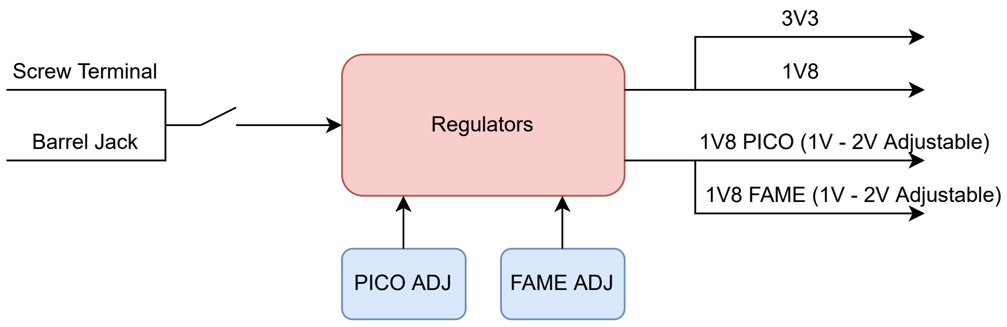


Figure 4.4: Power rails of Saidoyoki.

(KeyMill). The sensors in FAME detect timing faults injected through clock glitching and voltage glitching.

4.2.2 Power

The power supply network of Saidoyoki provides a flexible and reliable power supply for hardware security experiments. The board can be powered by one of the two possible input ports, a barrel jack with a 2.5mm center pin diameter, and a screw terminal. After an on/off slider switch, four Analog Devices LT8083IDF#PBF linear voltage regulators are positioned in parallel. Two of these regulators are used for general purposes 1.8V and 3.3V supply rails. The general purpose rails supply I/O voltage for PICO and FAME and the rest of the board. The other two regulators are dedicated to the core voltage supply to the test chips. They are both set to 1.8V as default, but it is possible to adjust their voltage output for other experiments, like fault attacks. These two regulators' names are written next to them on the board. Users can switch the related slider to the *ADJ* position to enable adjustable operation. After that, the accompanying variable resistor can be set using a screwdriver. The output of these adjustable regulators is limited from 1v to 2V, protecting the test chips from severe off-limits voltage values. A block diagram of the power network can be seen in Figure 4.4.

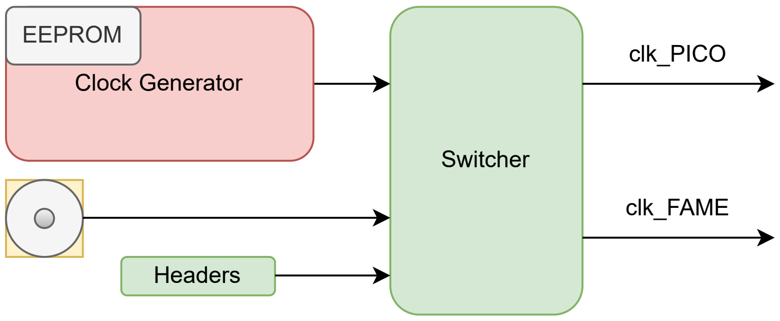


Figure 4.5: Different clock supply options of Saidoyoki.

4.2.3 Clocking

Saidoyoki can supply independently configurable clock signals from several sources for its test chips, PICO and FAME. The primary clock source is the on-board clock generation IC, CDCE925PWR from Texas Instruments. CDCE925PWR is a clock synthesizer IC with two programmable PLLs. Another feature is the internal EEPROM of the clocking IC that can store device configurations permanently. CDCE925PWR is configured over an I2C bus via a USB bridge. A user can program the clocking chip before programming the test chips. If the internal EEPROM was also programmed with the current configuration, the IC would continue to supply the clock after a power-off. A clock signal up to 230 MHz can be supplied through this device. Saidoyoki can receive clock signals from an external source too. It houses an SMA connector and a selection header switch that can also be used as another clock supply port. In Chapter 5, an implementation of Saidoyoki is presented where it receives its clock signal from an external device through this header switch. Figure 4.5 presents a block diagram of the clocking circuit.

4.2.4 Measurements

Power measurement is the most critical part of post-silicon side-channel evaluation. Therefore, Saidoyoki focuses on the power measurement region by providing several different methods to collect power side-channel data. A diagram of the power measurement region on Saidoyoki can be seen in Figure 4.6.

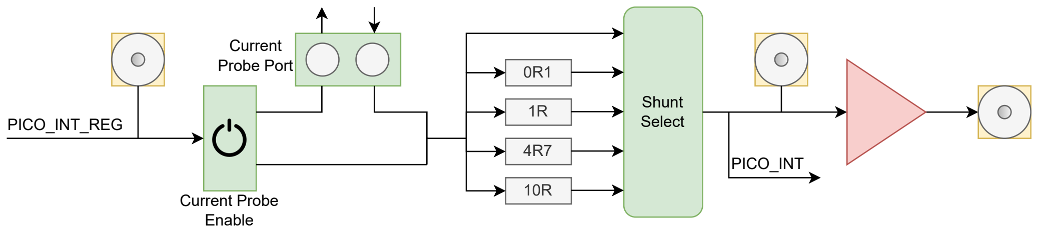


Figure 4.6: Block diagram of the power measurement region of Saidoyoki.

Saidoyoki has two power measurement circuitry, one for the PICO and one for the FAME. Although they are the same, Figure 4.6 shows the PICO part. The power measurement block observes the core voltage supply rail of the test chips. The blocks stand between the output of the regulators and the input of the chip power pins. So, they collect power measurements from the high side of the power network. This structure's two main objects are the current probe port and the low noise amplifier (LNA). The first one is the optional current probe port. This port, when enabled, breaks the serial circuit and drives the current to a screw terminal. The current data is collected externally, and the board receives the current back from the other port of the screw terminal. When disabled, it acts as a closed circuit without letting the current pass through the screw terminal. This prevents unnecessary resistance addition in the network. However, removing the additional resistance caused by the on/off switch is not possible. The next step in the network is the shunt resistor selection block. Users can choose one of the four precise shunt resistors for power measurement. Shunt resistor values are 0.1 Ohm, 1 Ohm, 4.7 Ohm, and 10 Ohm.

There is also an option to bypass the shunt resistor block when it is not needed. After the shunt resistor selection, the next part is the LNA; also, the power rail is routed to the related chip in a 50 Ohms trace (providing better signal quality) after this point. Saidoyoki, as the LNA, has an NXP BGA2801 MMIC wideband signal amplifier providing a gain of 22.2 dB at 250 MHz. The LNA is also internally matched to 50 Ohms. This amplifier is used for side-channel data collection in Chapter 5. The output of the LNA is reachable with an SMA connector. Additionally, there are two other SMA connectors in each power measurement block, one at the block input and one at the LNA input, for observing the power rail when needed.

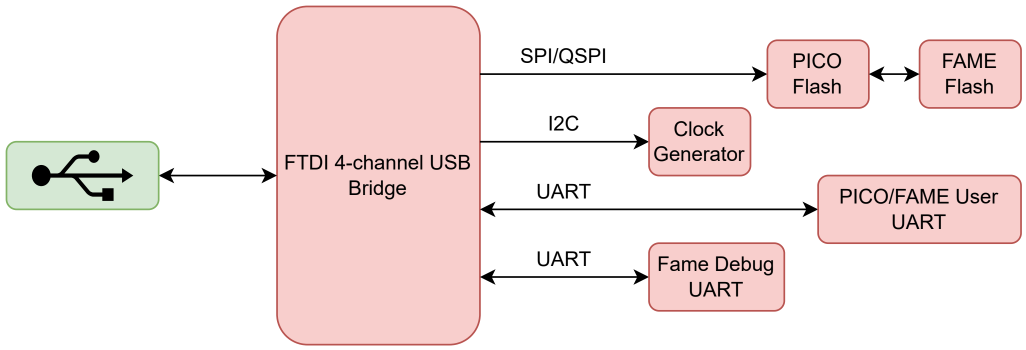


Figure 4.7: Communication channels of FT4232H.

4.3 Interfacing

4.3.1 Communication Interface

One of the most essential features that make Saidoyoki ideal for side-channel research is that it is possible to control every programmable component on the board with a single USB micro-B port. Saidoyoki has a USB bus controller (USB bridge) IC, FT4232H from FTDI, with four channels. Users can access any of those channels through a computer. Figure 4.7 shows a block diagram of the data communication network of Saidoyoki. These channels are as follows:

- **SPI/QSPI:** For programming the flash memory chips.
- **I2C:** For programming the clock generator.
- **UART1:** For the UART communication with the chips.
- **UART2:** For debugging FAME.

Most of these channels are shared between PICO and FAME. This allows easy board configuration without requiring different ports for programming, debugging, and communication. Saidoyoki also has a GitHub page. Users can find example codes and easy programming scripts that do not require more than one Linux command for flashing binaries to PICO's or FAME's program memory.

4.3.2 Physical Interface

A wide range of flexibility comes with many components to be configured. FTDI's USB bridge can handle all the data communication, but there are still many configuration points that should be set by hand. Although some of these physical configuration points were mentioned above, this section gives a complete list of them:

- Adjustable regulators have slider switches to enable the adjustable mood. When enabled user should set an output voltage using the adjustable resistor with a screwdriver.

- Current ports can be enabled or disabled using the slider switch. However, users should remember that when these switches are left at the "enable" position but no current probe is connected, the power supply rail will remain as open-circuit.

- A shunt resistor can be selected among the possible options. When there is no need for one, a jumper should connect bypass pins.

- CDCE925PWR clock chip has a user-defined input port, S0, that can be programmed by the user. This port can be grounded by a jumper. It remains high when left unconnected.

- CDCE925PWR clock chip uses its S1 and S2 user input ports as I2C ports at the same time. If accidentally, the chip is programmed to use its I2C ports as user input ports, it no longer uses the I2C bus. One recovery from this situation is to force the output of the chip voltage supply pin to the ground. Using the related header, this pin is connected to 3.3V as default. It can be connected to the ground header that stands next to it in such a case. At this stage, the chip temporarily restores the I2C bus for use.

- FAMEs flash chip can be reset using the related header.

- Flash chips share the same SPI/QSPI bus as the same slave. Based on this, when programming a flash, its nearby headers should be set to "FT"; when a chip is to read the program from its flash memory, related switches should be set to "FLASH." Switches of both flash memory ships should not be at the "FT" position at the same time.

- User UART is shared between PICO and FAME. Related headers should be set accordingly to connect to the correct chip.
- FAME has two debug UART ports, the needed one should be set accordingly.
- Boot pin of FAME can be set to ground or to 3.3V using the related slider switch.
- Test mode for FAME can be enabled using the related header.
- It is possible to reach all four channels of the USB bridge via headers.

More information about the physical design features and the physical configuration before programming can be found on the GitHub page.

Chapter 5

Implementations

In this chapter, implementations of the above-presented pre-silicon and post-silicon side-channel tools are shown. The first section analyzes a 128-bit AES hardware cipher (the exact hardware used by PICO as an accelerator) using the KL divergence metric from RTL-PSC. KL divergence metric is calculated in the UVM environment using the SCO VIP. To show that SCO VIP can be used for other analysis as well, t values for a t-test are also calculated and the results are presented. In the second section, to show the abilities of the Saidoyoki board, a CPA attack is carried out on the same AES hardware in the PICO.

5.1 Pre-Silicon Leakage Assessment on AES-128

5.1.1 Method

It was mentioned before that side-channel leakage assessment at the RTL level lacks many low-level circuit factors that actually contribute to the target device's information leakage characteristic. However, it is still very profitable to implement a side-channel analysis at RTL if it can find any leakage. Therefore, RTL-PSC is one method that has been proposed to make an RTL side-channel assessment analysis. This method uses the KL divergence metric and the success rate metric based on maximum likelihood estimation to decide which RTL design module leaks at which clock cycle when performing an AES encryption.

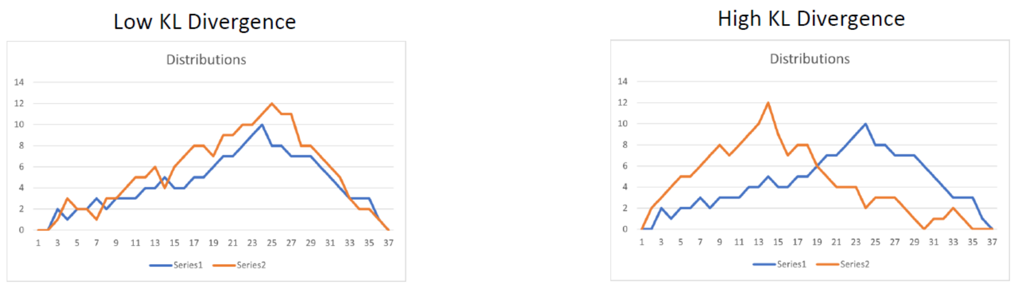


Figure 5.1: KL divergence value tendencies based on two different scenarios.

This thesis used the KL divergence metric for our RTL side-channel leakage assessment. KL divergence is a statistical method that evaluates the statistical distance between two probability distributions. Figure 5.1 presents a visual explanation of the KL divergence metric based on two hypothetical distribution scenarios.

The difference in KL divergence values that is shown in Figure 5.1 is clearly observable that in the low KL divergence scenario, two distributions are much looking alike. In contrast, in the high KL divergence scenario, they are distinctively different from each other. The exact value of KL divergence is calculated using Formula 5.1 where q and p are different probability distributions and $D(q \parallel p)$ is KL the divergence value from q to p.

$$D(q \parallel p) = \int q(x) \log \frac{q(x)}{p(x)} \, dx \tag{5.1}$$

RTL-PSC calculates KL divergence values for two scenarios to create two distributions. In the first scenario, AES encryption is carried out using the secret key value of all zeroes. In the second scenario, the key value is set as all ones, so that the maximum possible HD value can be achieved. This provides the exploration of the state where the target device is most vulnerable against side-channel attacks. To eliminate the effect of the plaintext on the observations and provide randomness for the plaintext, they repeat both encryptions 1000 times, where the output of each encryption is fed back to the input. The first input, the seed, is also chosen randomly. After this, two distribution functions are calculated from the collected toggle count data for the calculation of KL divergence values. First, KL divergence is calculated for each block at each clock cycle of the encryption. Then, the resulting values are normalized by the maximum value. As a result, normalized KL divergence value locations larger than 0.5 are considered as leaky points.

5.1.2 Testbench

Universal Verification Methodology is an industry standard framework that is being used to functionally verify digital circuits at RTL. Therefore, when there is a method for side-channel leakage assessment at RTL that is meant to be a part of the industrial design flow, UVM is the best candidate that can implement this idea.

In this thesis, the KL divergence metric proposed by RTL-PSC has been used to develop a UVM Verification IP that can observe the toggle counts and analyze them in parallel to the rest of the functional verification process. SCO VIP, which was mentioned in Chapter 3, is used to do this. As a reminder, SCO VIP has a UVM Agent with numerous observer units that are connected to the design using specific interfaces, each dedicated to the individual design unit. SCO VIP always listens to the low-level interface for changes in the signal states. Another component is the counter block. This block includes a counter for each observer. When the encryption is ongoing, the related agent sets the count to enable the signal to this block, activating the counter. This related agent is usually an agent that is used to send input data to the cipher. This agent also provides the number of clock cycle that is being executed and the number of encryption cycles. The counter block combines all this data together, and a data block named Toggle Counts Array is generated. This array contains the toggle counts for each specific module, clock cycle, and encryption cycle. Then this array is sent to the internal scoreboard of the SCO VIP to calculate KL divergence values. PICO's AES coprocessor is used independently as the target device in this scenario. AES encryption is carried out 1000 times with the secret key in all zeroes and another 1000 times with the secret key in all ones. Figure 5.2 shows the complete testbench architecture that is used in this experiment.

UVM testbenches are structued on a highly modular basis, this brings the reusability feature to UVM which is one of the strongest factors contributing

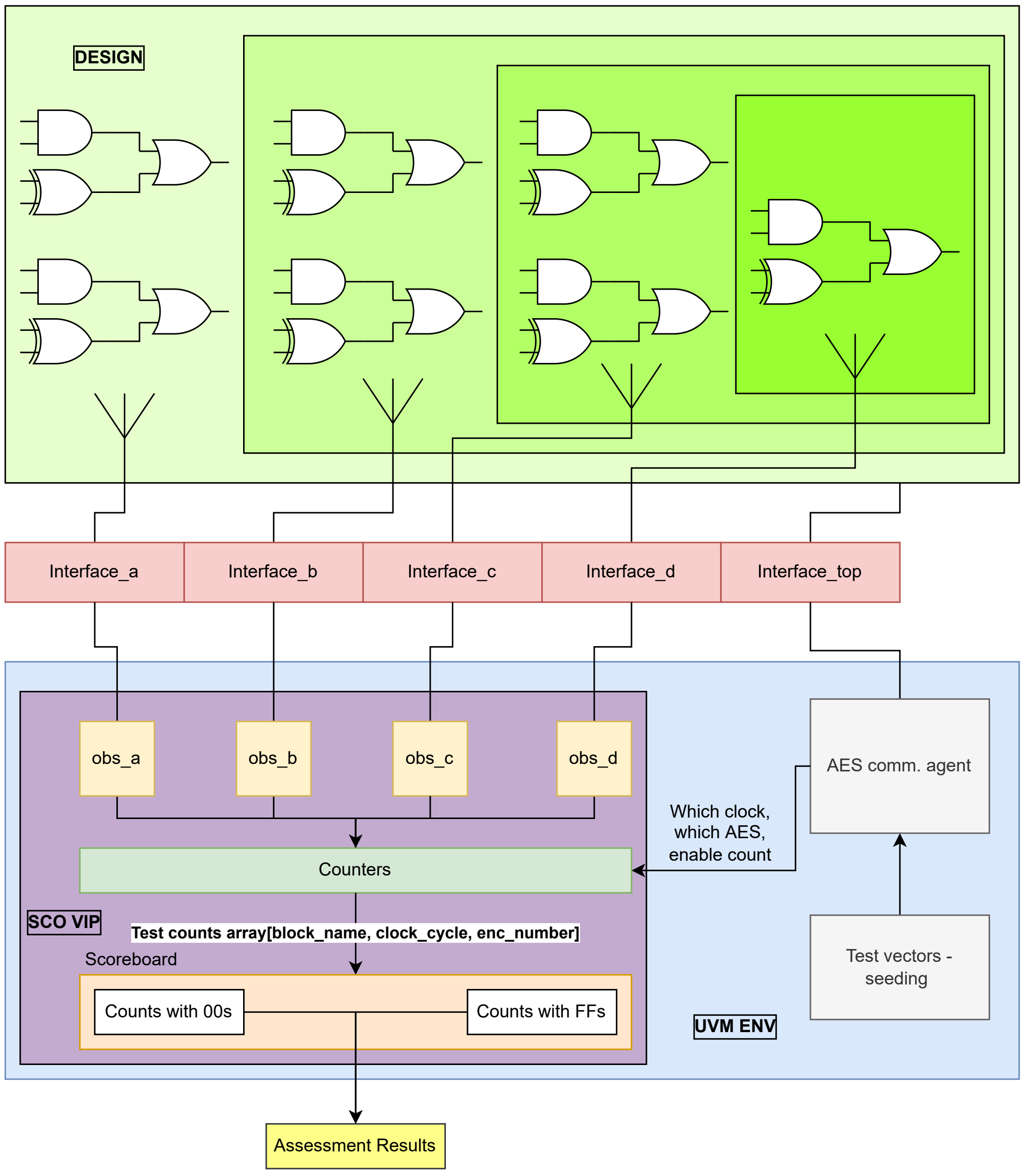


Figure 5.2: UVM testbench arcitecture for RTL side-channel assessment with KL divergence metric.

to its ascension as an industry-standard. In Code Listings 5.1, 5.2, and 5.3, it is shown that how UVM base class libraries and SCO VIP packages can be imported, configured and used to start a simple test. UVM code structures are usually long. Therefore, the mentioned listings show sample coding styles. The rest of the source code of the testbench and the SCO VIP package can be found in the project's GitHub page. Although many RTL simulators have the UVM base class libraries built-in, they are also available as open source at the website of Accellera Systems Initiative.

```
// testbench top
`include "uvm_macros.svh"
module top;
  import uvm_pkg::*;
  import test_pkg::*;
  // Variable declerations
  . . .
  //
  aes_com_if      aes_com_if_0(); // Interface for com agent
  if_aes_comp_core* if_aes_comp_core*_0(); // Interface for observers
  aes_top DUT(/* DUT top level signals to agent interfaces*/);
  bind_dut_internals(); // bind DUT signals to observer interfaces
  initial begin // make interfaces available to the uvm
    uvm_config_db#(virtual aes_com_if)::set(null, "*", "aes_com_vif", aes_com_if_0);
    . . . ;
  end
  initial begin
    generate_clock_and_reset();
  end
  initial begin
    run_test(); // specific method to start uvm_tests
  end
endmodule
```

Code Listing 5.1: A sample testbench top module that imports the UVM base class libraries.

```
//test -- all associated files are combined in test_pkg
class test extends uvm_test;
  `uvm_component_utils(test) // register the class as a uvm
   component
  env_base          m_env; // bring uvm environment
  env_config_base       m_cfg; // bring uvm env config class
  aes_com_env_config      m_aes_com_env_config; // bring
   communication agent
  aes_com_agent_config    m_aes_com_agent_config;
  aes_observer_env_config   m_aes_observer_env_config; //
   bring SCO
  aes_observer_agent_config m_aes_observer_agent_config;
  aes_sequence         m_aes_sequence; // bring test sequence
  // construct and register classes to uvm database
  // configure agents and the environment using the handles.
  task run_phase(uvm_phase phase);
    `uvm_info(get_type_name(), "In run phase of 'test'.",
   UVM_LOW)
    phase.raise_objection(this, "test");
      m_aes_sequence.start(); // start test sequence
    phase.drop_objection(this, "test");
  endtask
endclass
```

Code Listing 5.2: A sample test class that imports the agents, configures and drives the test sequence.

```
//environment -- test calls the env
class env_base extends uvm_env;
  `uvm_component_utils(env_base)
  aes_com_env     m_aes_com_env; // bring aes com agent
  aes_observer_env  m_aes_observer_env; // bring SCO agent
  function void build_phase(uvm_phase phase);
    // build classes
  endfunction
  function void connect_phase (uvm_phase phase);
    // connect uvm harness
  endfunction
endclass
```

Code Listing 5.3: A sample environment class that imports the agents (and other components) and does the internal connections.

One thing to mention is the calculation process of the KL divergence values. As the Formula 5.1 is a continuous function. As our data is discrete, we need to use the discrete version of this formula, which is given in Formula 5.2 where p and q are probability distributions, and D is KL divergence.

$$D(q \parallel p) = \sum_{x \in X} q(x) \log \frac{q(x)}{p(x)} \tag{5.2}$$

After the collection of the toggle counts, every specific module has a data set of toggle count numbers for each clock cycle. Each of those data sets includes 1000 data points as both encryptions were repeated 1000 times. RTL-PSC proposes that these data sets follow Gaussian Distribution. So, then, means and variances values are calculated. After this, mean and variance values are used to calculate probability functions, and finally, the KL divergence values for every module at every clock cycle.

5.1.3 Statistical Analysis

KL Divergence Analysis. After running the above presented UVM testbench, the results showing the KL divergence values of each module at each clock cycle are given in Figure 5.3. According to the Figure 5.3, modules 0 and 1, which are corresponding to the RTL module names *aes_comp_core* and *aes_comp_core*(*enc_block*), show considerable leakage, which is expected. Another plot, combining KL divergence values of every module together on the same output data, is made to see which clock cycles are the most leaky ones. Figure 5.4 shows this plot, where it shows the most leakage values at the first, sixth, and the last rounds of AES. Using UVM for side-channel leakage assessment also increased the runtime of the analysis. According to RTL-PSC, their method that first runs the RTL simulation, then extracts the switching activity report and analyzes them takes around 24 minutes for an AES-LUT and around 46 minutes for AES-GF implementations. On the other hand, as can be seen in the test report in Figure 5.5, SCO VIP does this analysis in less than a minute.

T-Test. This thesis primarily presents a scenario where KL divergence metric, which was used by the RTL-PSC framework, is exploited for side-channel leakage assessment. However, the scoreboard gives a free space for the user to implement other methods too. T-test is a method that is widely used in

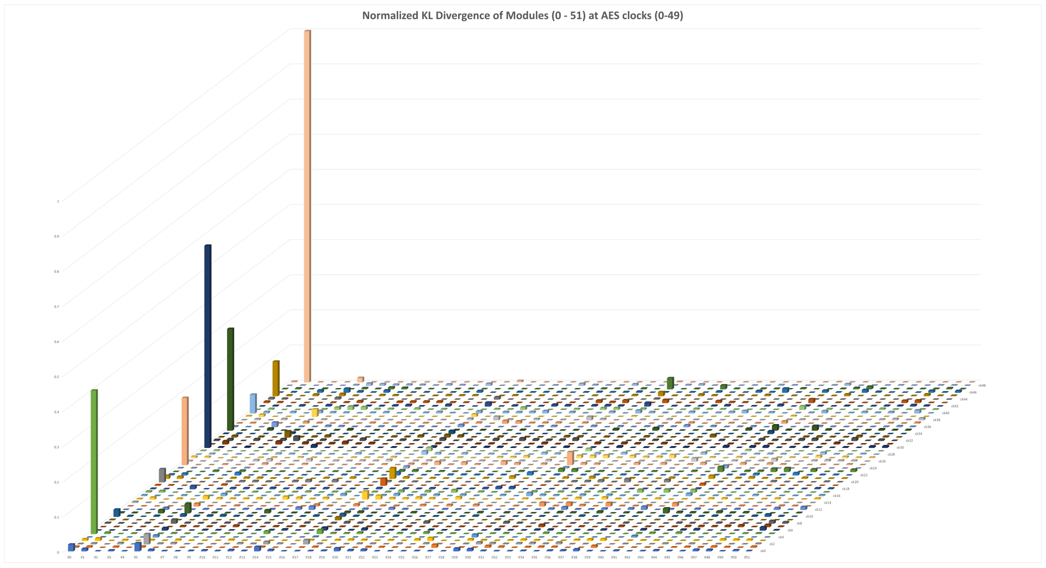


Figure 5.3: Normalized KL divergences values of PICOs AES accelerators RTL modules at every clock cycle.

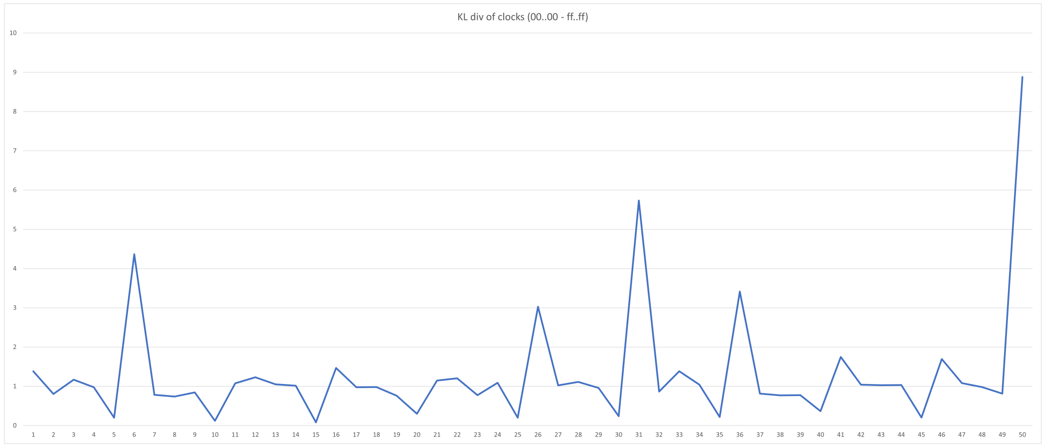
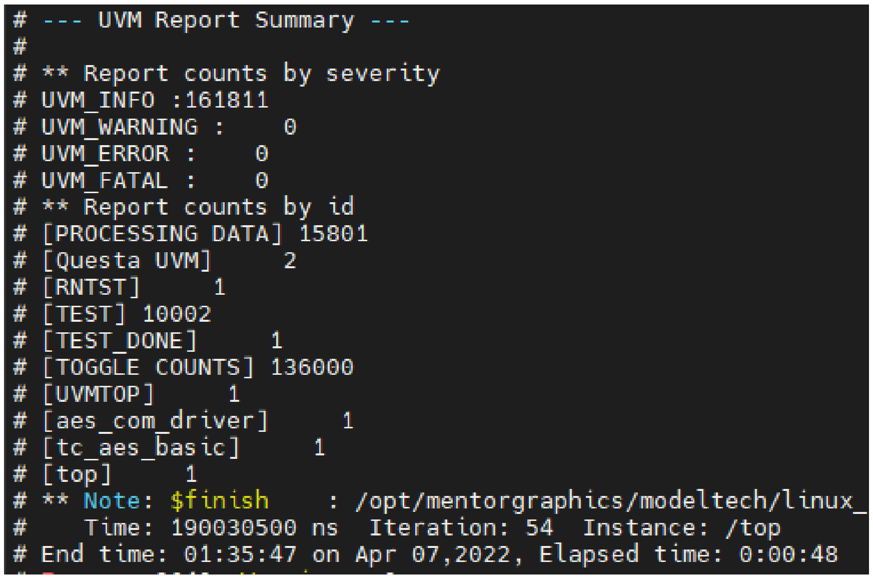


Figure 5.4: KL divergence at clock cycles.



```
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :161811
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [PROCESSING DATA] 15801
# [Questa UVM]     2
# [RNTST]     1
# [TEST] 10002
# [TEST_DONE]     1
# [TOGGLE COUNTS] 136000
# [UVMTOP]     1
# [aes_com_driver]     1
# [tc_aes_basic]     1
# [top]     1
# ** Note: $finish    : /opt/mentorgraphics/modeltech/linux_
#    Time: 190030500 ns  Iteration: 54  Instance: /top
# End time: 01:35:47 on Apr 07,2022, Elapsed time: 0:00:48
```

Figure 5.5: Runtime of SCO VIP in UVM with PICOs AES coprocessor as the target device.

side-channel leakage assessment. In Figure 5.6, a scenario where, instead of KL divergence values, t values between two data sets are calculated in the scoreboard to be used in a t-test. T-test basically tests if two data sets are significantly different than each other. One similarity between the t values and the KL divergence values is that they show significantly high values at the sixth round of AES.

5.2 Attacking PICOCHIP's AES Coprocessor

5.2.1 Method

To demonstrate the capabilities of the Saidoyoki board, a side-channel attack (CPA) is carried out on a 128-bit AES hardware implementation which is used as a coprocessor by PICO. As a reminder, Correlation Power Analysis (CPA) is a statistical method used to extract secret key values from ciphers

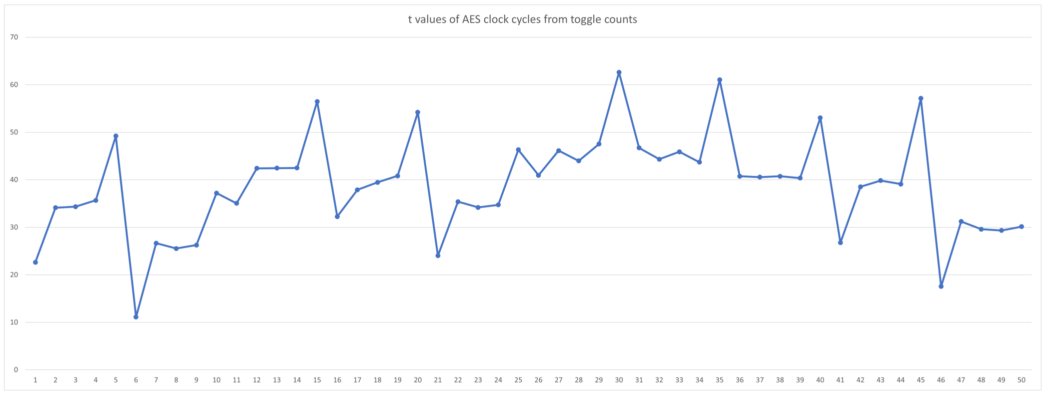


Figure 5.6: T values calculated from the data sets of all-zeros and all-ones.

by analyzing the power traces which are taken from the target device while it is performing a key-based operation. It was shown by previous works that CPA can effectively extract the secret key value from AES ciphers. CPA does this by calculating the Person Correlation Coefficient value for each sample point and key guess. In CPA, the correlation is made between a hypothetical power consumption (power model) and the measured ones. Naturally, the correlation values that were calculated using the hypothetical power consumption of the correct key guess show a distinction from the others. As mentioned above, the power model can be Hamming Weight or Hamming Distance. In this demonstration, we used Hamming Distance between the input and the output of the S-box operation in the first AES round. Formula 5.3 shows the calculation of the power model where HD is the Hamming Distance, HW is the Hamming Weight, p is the plaintext, and k is the key guess. In this attack, all calculations are made byte-wise.

$$HD = HW((p \oplus k) \oplus s\_box(p \oplus k)) \tag{5.3}$$

5.2.2 Hardware Setup

For trace collection and cipher inputs generation (plaintexts and the key), a ChipWhisperer Lite board is used. ChipWhisperer (CW) is able to supply

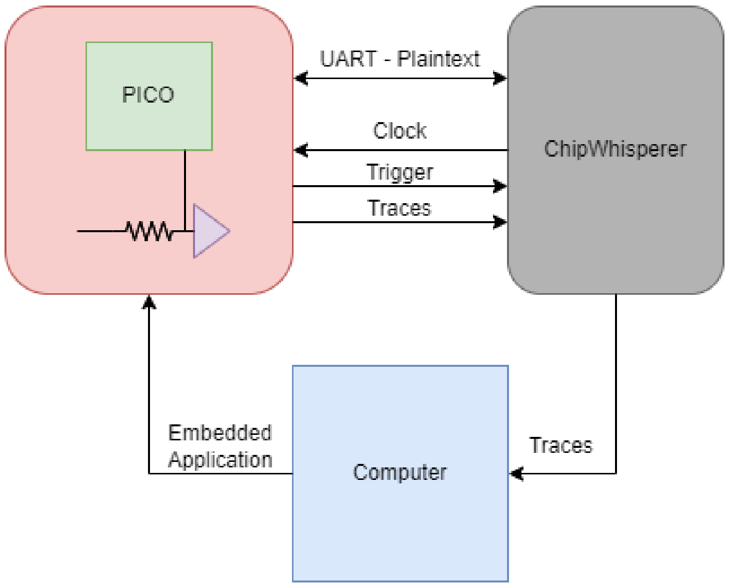


Figure 5.7: Wiring between ChipWhisperer and SaiFdoyoki.

plaintexts and keys to a target device through a UART bus. CW has a 10-bit ADC which can collect samples up to 105 MS/s for trace collection. Whenever the target board notifies the CW of the trigger signal, CW starts to collect a trace for the specified number of samples. CW can also supply clock signals for its target board. In this scenario, Saidoyoki is the target board. Figure 5.7 shows the block diagram of the wiring between CW and Saidoyoki.

In this setting, CW supplies a clock signal of 4 MHz to the Saidoyoki, while the CW sampling rate is 16 MS/s, which means four samples are taken from each clock cycle. A UART bus is used for data communication and handshaking. Saidoyoki provides a trigger signal to inform CW that the encryption has started. CW is also connected to a host computer to send the recorded data. Before the CW operation starts, the PICO chip on Saidoyoki is programmed to perform handshaking with the CW, receive the plaintext-key, and start the encryption while setting the trigger pin high. Using this handshake technique introduces delays in the operation but returns with the elimination of data losses. ChipWhisperer, when started, carries out the handshake protocol first, then sends the input data and waits for the trigger

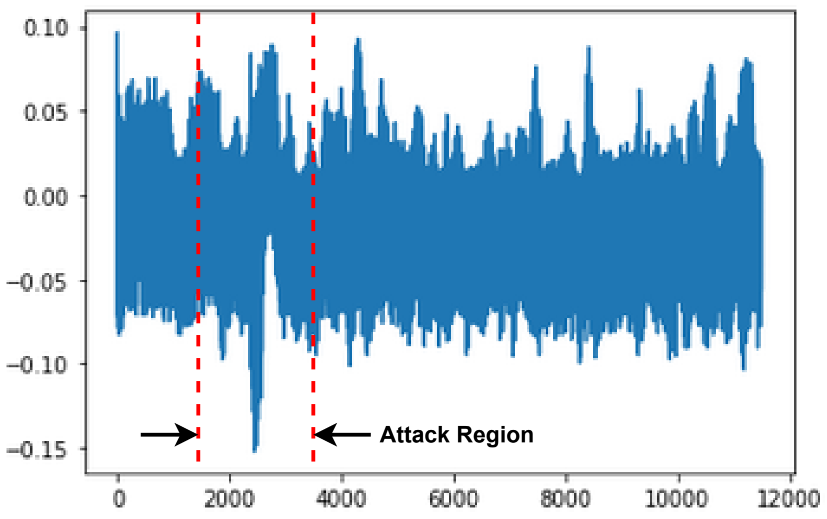


Figure 5.8: A randomly chosen trace and the attack region.

signal. When the trigger signal arrives, it starts to collect the trace. On the side of Saidoyoki, power traces are collected from the LNA using a 4.7 Ohm shunt resistor. This operation repeats as many times as required.

5.2.3 Results

In the initial experiments, it was seen that after Saidoyoki issues the trigger signal, the whole AES rounds take 11500 sample points at most, remember that the sampling rate of CW was 16 MS/s. Provided that, 11500 sample points correspond to 718.75 μs or 2875 clock cycles after the trigger is set. Figure 5.8 shows a random trace. In this trace, it is easy to visually locate the s-box operation from which we calculated the power model. This is the attack region. This is the the attack region.

After this, 400 thousand traces are collected from Saidoyoki. A range of sample points from the sample 1500 to the sample 3500 were cut out from each of these traces to decrease the analysis time. As a result of this analysis, CPA was able to recover fifteen out of sixteen key bytes. For the only key byte that was not able to be recovered, the correct key guess had the second highest correlation with a very small error after the highest one. Figure 5.9, Figure 5.10 and Figure 5.11 show the correlation peaks observed for the key bytes zero to seven, eight to fifteen, and a more detailed plot of the erroneous byte (byte 1), respectively.

One reason for not being able to recover one byte could be the possible electrical noise observed on Saidoyoki. Saidoyoki version two is the improved version which is hardened explicitly against electrical noise. Although it is not possible to eliminate the noise completely, traces collected from version two show more clean trends, and the CPA results show more distinct peaks compared with the results from version one. Moreover, to eliminate the distortion caused by the noise, other experiments using more traces were also done. However, these experiments still showed the same behavior for byte1 - failure to recover with a slight difference. As a result, despite electrical noise being a strong candidate to be the origin of these issues, there could be other factors in effect instead of the noise or together with the noise. This will be a future improvement action point for Saidoyoki version three.

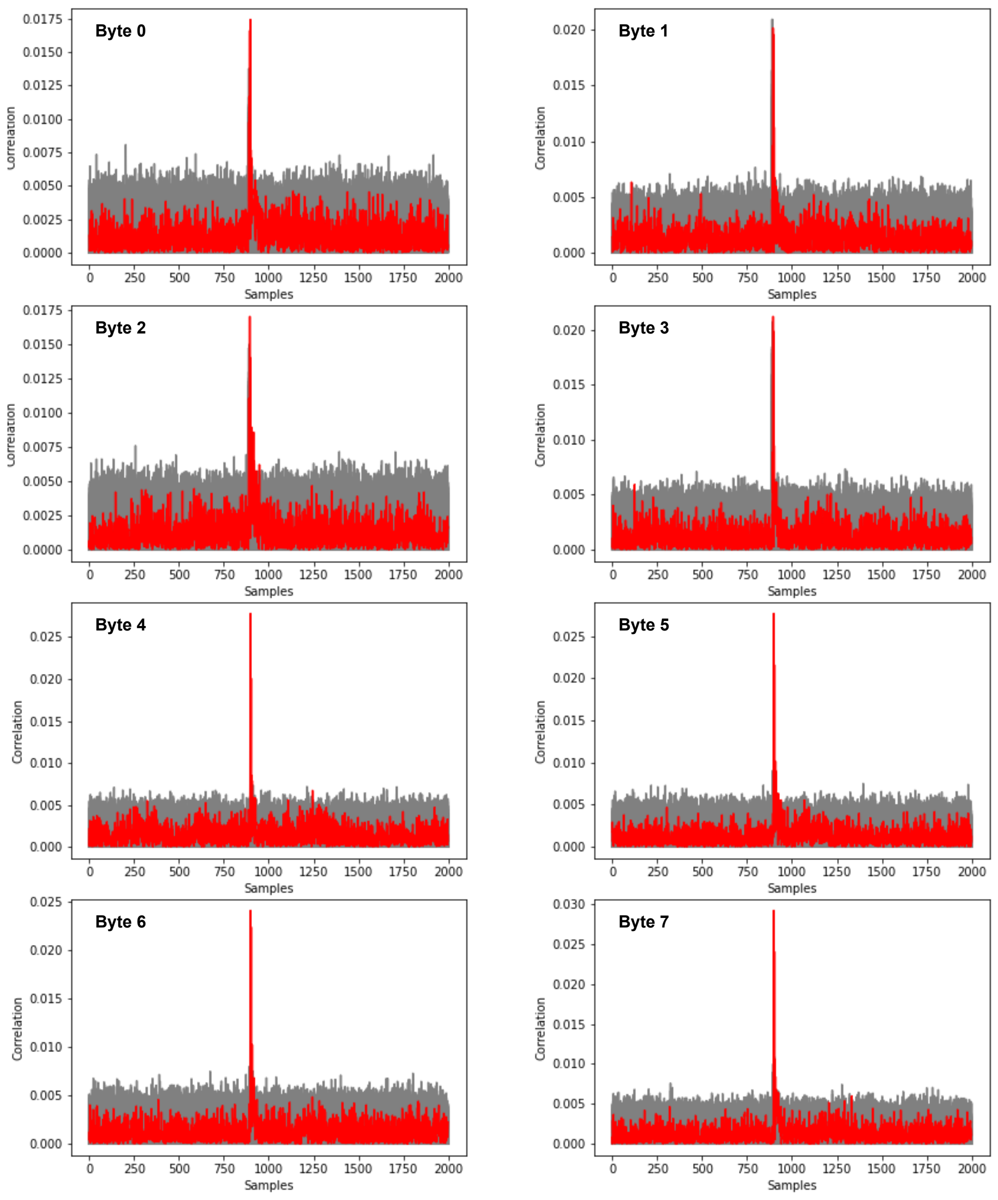


Figure 5.9: Correlation peaks for key bytes from zero to seven.

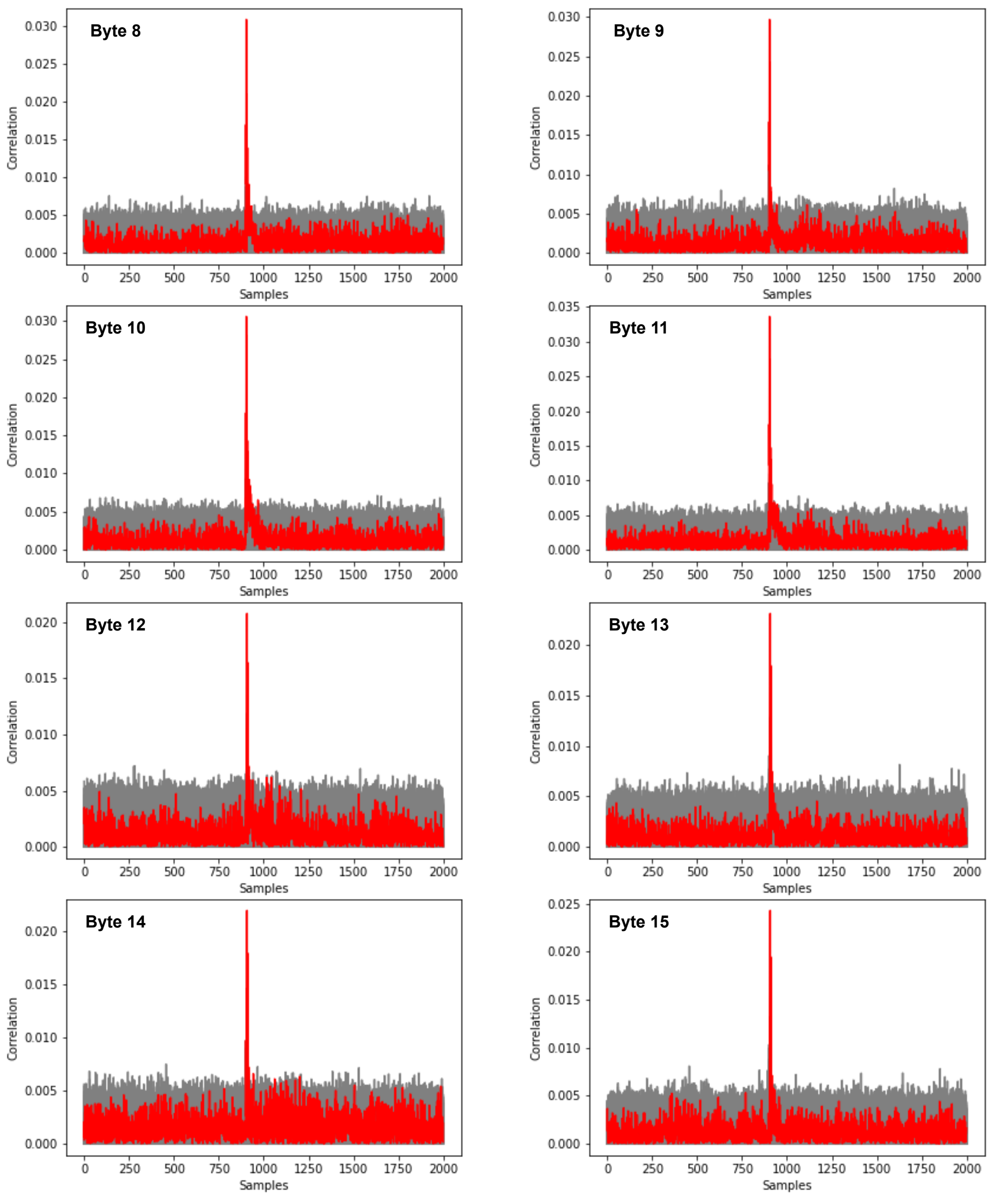


Figure 5.10: Correlation peaks for key bytes from eight to fifteen.

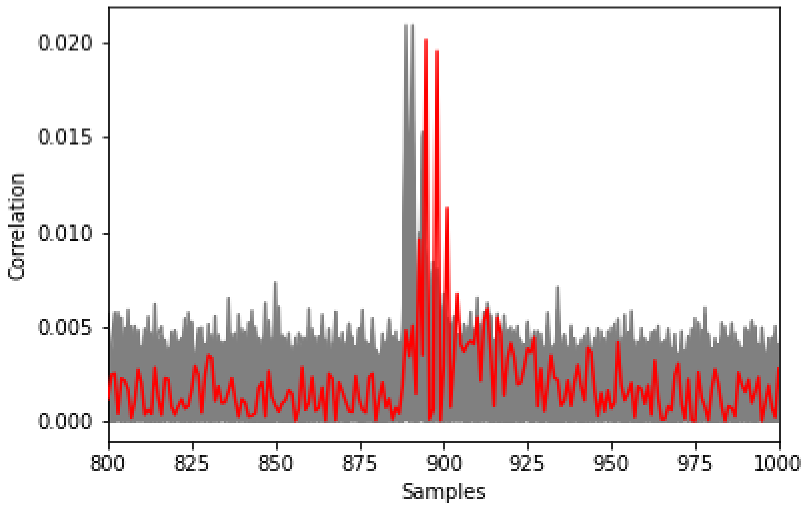


Figure 5.11: Detailed correlation plot of byte 1.

Chapter 6

Conclusion

With the recent advancements in processing technology, the processing capacity of even simple personal machines significantly increased. These advancements made statistical analysis methods utilize the power of brute-force analysis in statistical methods. This means that even commonly available devices are now well capable of performing high processing power requiring methods. Furthermore, side-channel analysis methods, as well, are now getting more dangerous for hardware security. In this state, understanding the origins of side-channel leakage is a matter of great importance. Side-channel leakage assessments methods, accordingly, are getting more complex every day. Although the number of academic works on leakage assessment methods is growing every day, it is crucial to have reliable, flexible, and standards-compatible infrastructures so that these methods can be used on a broader ground, including in industry.

Towards the needs mentioned above, this thesis presented two side-channel leakage assessment tools, SCO VIP and Saidoyoki board. SCO VIP is focused on being a UVM compatible side-channel assessment tool in RTL so that new assessment methods based on toggle counting can easily be implemented in an industry standard verification environment. A demonstration of SCO VIP is made by implementing the recently proposed RTL-PSC method. SCO VIP proved to show results faster than the original work. Saidoyoki, on the other hand, is designed to be a flexible and reliable post-silicon tool for side-channel leakage experiments. Its highly configurable structure is demonstrated to be making side-channel analysis experiments easier. Saidoyoki board presented in this thesis is the second version of the board. Saidoyoki version two also showed itself in a CPA attack on a 128-bit AES hardware accelerator, resulting in more distinct correlation peaks than version one.

Bibliography

[1] Hans Delfs, Helmut Knebl, and Helmut Knebl. *Introduction to cryptography*, volume 2. Springer, 2002.

[2] Marian Rejewski. Mathematical solution of the enigma cipher. *Cryptologia*, 6(1):1–18, 1982.

[3] R Davis. The data encryption standard in perspective. *IEEE Communications Society Magazine*, 16(6):5–9, 1978.

[4] Hans Delfs and Helmut Knebl. Symmetric-key encryption. In *Introduction to Cryptography*, pages 11–31. Springer, 2007.

[5] Matt Curtin and Justin Dolske. A brute force search of des keyspace. In *8th Usenix Symposium, January*, pages 26–29. Citeseer, 1998.

[6] Joan Daemen and Vincent Rijmen. The rijndael block cipher: Aes proposal. In *First candidate conference (AeS1)*, pages 343–348, 1999.

[7] G Joy Persial, M Prabhu, and R Shanmugalakshmi. Side channel attack-survey. *Int J Adva Sci Res Rev*, 1(4):54–57, 2011.

[8] François-Xavier Standaert. How (not) to use welch's t-test in side-channel security evaluations. In *International conference on smart card research and advanced applications*, pages 65–79. Springer, 2018.

[9] Yuan Yao, Tarun Kathuria, Baris Ege, and Patrick Schaumont. Architecture correlation analysis (aca): Identifying the source of side-channel leakage at gate-level. In *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 188–196. IEEE, 2020.

[10] Miao He, Jungmin Park, Adib Nahiyan, Apostol Vassilev, Yier Jin, and Mark Tehranipoor. Rtl-psc: Automated power side-channel leakage assessment at register-transfer level. In *2019 IEEE 37th VLSI Test Symposium (VTS)*, pages 1–6. IEEE, 2019.

[11] Barbara Gigerl, Vedad Hadzic, Robert Primas, Stefan Mangard, and Roderick Bloem. Coco:{Co-Design} and {Co-Verification} of masked software implementations on {CPUs}. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1469–1468, 2021.

[12] Davide Poggi, Philippe Maurine, Thomas Ordas, and Alexandre Sarafianos. Protecting secure ics against side-channel attacks by identifying and quantifying potential em and leakage hotspots at simulation stage. In *International Workshop on Constructive Side-Channel Analysis and Secure Design*, pages 129–147. Springer, 2021.

[13] Rajat Sadhukhan, Paulson Mathew, Debapriya Basu Roy, and Debdeep Mukhopadhyay. Count your toggles: a new leakage model for pre-silicon power analysis of crypto designs. *Journal of Electronic Testing*, 35(5):605–619, 2019.

[14] Hendra Guntur, Jun Ishii, and Akashi Satoh. Side-channel attack user reference architecture board sakura-g. In *2014 IEEE 3rd Global Conference on Consumer Electronics (GCCE)*, pages 271–274. IEEE, 2014.

[15] Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. *Handbook of applied cryptography.* CRC press, 2018.

[16] Iqtadar Hussain, Tariq Shah, Hasan Mahmood, and Muhammad Asif Gondal. A projective general linear group based algorithm for the construction of substitution box for block ciphers. *Neural Computing and Applications*, 22(6):1085–1093, 2013.

[17] Dakshi Agrawal, Bruce Archambeault, Josyula R Rao, and Pankaj Rohatgi. The em side—channel (s). In *International workshop on cryptographic hardware and embedded systems*, pages 29–45. Springer, 2002.

[18] Onur Acıiçmez, Werner Schindler, and Çetin K Koç. Cache based remote timing attack on the aes. In *Cryptographers' track at the RSA conference*, pages 271–286. Springer, 2007.

[19] Stefan Mangard. A simple power-analysis (spa) attack on implementations of the aes key expansion. In *International Conference on Information Security and Cryptology*, pages 343–358. Springer, 2002.

[20] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *Annual international cryptology conference*, pages 388–397. Springer, 1999.

[21] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In *International workshop on cryptographic hardware and embedded systems*, pages 16–29. Springer, 2004.

[22] Yusuke Yano, Kengo Iokibe, Yoshitaka Toyota, and Toshiaki Teshima. Signal-to-noise ratio measurements of side-channel traces for establishing low-cost countermeasure design. In *2017 Asia-Pacific International symposium on electromagnetic compatibility (APEMC)*, pages 93–95. IEEE, 2017.

[23] Emmanuel Prouff and Matthieu Rivain. Masking against side-channel attacks: A formal security proof. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 142–159. Springer, 2013.

[24] Debayan Das, Shovan Maity, Saad Bin Nasir, Santosh Ghosh, Arijit Raychowdhury, and Shreyas Sen. Asni: Attenuated signature noise injection for low-overhead power side-channel attack immunity. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 65(10):3300–3311, 2018.

[25] Tao Zhang, Jungmin Park, Mark Tehranipoor, and Farimah Farahmandi. Psc-tg: Rtl power side-channel leakage assessment with test pattern generation. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 709–714. IEEE, 2021.

[26] Jason Oberg, Sarah Meiklejohn, Timothy Sherwood, and Ryan Kastner. Leveraging gate-level properties to identify hardware timing channels. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 33(9):1288–1301, 2014.

[27] Benjamin Jun Gilbert Goodwill, Josh Jaffe, Pankaj Rohatgi, et al. A testing methodology for side-channel resistance validation. In *NIST non-invasive attack testing workshop*, volume 7, pages 115–136, 2011.

[28] Stephen E Fienberg and Nicole Lazar. William sealy gosset. In *Statisticians of the Centuries*, pages 312–317. Springer, 2001.

[29] Juan Francesconi, J Agustin Rodriguez, and Pedro M Julian. Uvm based testbench architecture for unit verification. In *2014 Argentine Conference on Micro-Nanoelectronics, Technology and Applications (EAMTA)*, pages 89–94. IEEE, 2014.

[30] Pantea Kiaei, Zhenyuan Liu, Ramazan Kaan Eren, Yuan Yao, and Patrick Schaumont. Saidoyoki: Evaluating side-channel leakage in pre- and post-silicon setting. *Cryptology ePrint Archive*, 2021.