SAPIENZA
UNIVERSITÀ DI ROMA

Ingegneria dell'informazione, informatica e statistica

PhD IN ELECTRONIC ENGINEERING
XXIII CYCLE

SIDE-CHANNEL ATTACKS AND COUNTERMEASURES IN THE DESIGN OF SECURE IC's DEVICES FOR CRYPTOGRAPHIC APPLICATIONS

Ing. Luca Giancane

Tutor: Prof. Alessandro Trifiletti

Contents

List of Figures

Chapter 1

Introduction

1.1 History of Cryptography

The most ancient use of coding is probably that one found on the "lacedaemonian scytale" (400 b.C.), a stick on which a leather tape was bundled up; on the tap one wrote for columns parallel to the axis of the stick, letter for letter, the secret text. Removed the tape from the stick, the text was transposed in regular but sufficient way in order to avoid the reading without a second stick equal to the first one.

In the ancient time well-known was hectographed code, in which each letter was ciphered with a number couple, or codes in which each letter was replaced with another letter.

In 800 a.d. Al-Kindi, an Arab mathematician, devised new methods of breaking ciphers of the Qur'an. It was the most fundamental cryptanalytic advance until Second World War. Al-Kindi wrote a book on cryptography entitled "Manuscript for the Deciphering Cryptographic Messages", in which he described the first cryptanalysis techniques.

In Europe, cryptography became (secretly) more important as a consequence of political competition and religious revolution. Although cryptography has a long and complex history, it wasn't until the 19th century that it developed anything more than ad hoc approaches to either encryption or cryptanalysis (the science of finding weaknesses in crypto systems).

The First World War is the first great war after the invention of the radio; this new mass media if from one side allowed to transmit messages practically in an instantaneous way, on the other side it was much exposed to the interception of the enemy: to capture a courier that brought an important message was more difficult than to intercept a radio transmission. For this reason cryptographic became very important and the first cryptography machines (with the relative algorithms) were built.

But it was during the Second World War that research in cryptography experienced a major acceleration and when the war ended cryptography was

studied in many fields, from mathematics to electronics. In 1976, the algorithm Lucifer developed by Horst Feistel at IBM, was officially adopted with the name *DES* (*Data Encryption Standard*) as the first standard encryption algorithm, encouraging the business community to make use of cryptography to secure communication and business transactions. The same year, Whitfield Diffie, Martin Hellman and Ralph Merkle announced a method to solve the key distribution problem introducing the concept of asymmetric encryption and forming the basis for public key cryptography. Unfortunately, they did not succeed to find a one-way function necessary to implement their method. In 1977, Ronald Rivest, Adi Shamir and Leonard Adleman, mathematicians at the MIT laboratory for computer science, introduced the *RSA*, an asymmetric encryption algorithm which finally allowed public key cryptography to be used in practice.

Nowadays cryptography is used to protect personal data: using a mobile telephone, withdrawing money from a bank, storing medical data and so on. All these operations are based on the use of a pocket-sized card with an embedded integrated circuit, commonly named *smart card* or *chip card*.

The interest in secure devices has led to a great numbers of research works on new attacks, on one side, and new countermeasures on the other side. Different levels are considered starting from the algorithms to hardware realizations. In particular, since the introduction of side-channel attacks (physical quantities related with the key), many works have been published on how to exploit or avoid side-channel information leakage, i.e. information that can be retrieved from a cryptographic device by measuring a quantity not directly involved in the secret information treatment. Essentially from 20th century, cryptography makes extensive use of mathematics and engineering, including aspects of physics, information theory, computational complexity, statistics, combinatorial, abstract algebra, number theory, by involving more and more different aspects of sciences in general.

1.2 Smart Card

A smart card consists of a plastic support in which a microchip is inserted; the interface between microchip and external world can be a series of contacts (in this case it is properly named contact-based smart card) or an antenna (contactless smart card) or both (dual-interface smart card).

Probably the beginning of the story of Smart Card is to be found in the novel "La nuit des temps" (The drawn of time), in which the french science-fiction writer René Barjavel wrote in 1968 about a magic ring used as a key by Gondas, a very old and highly advanced civilization. The magic ring was empowered with memory and communications:

> "... Every time a Gonda wanted something new, [...], he would pay with his key. He would bend his middle finger, would enter his key in a location chosen at this effect and his account at the central computer would immediately be reduced by the value of the merchandise or the request service ..."

Figure 1.1: An example of banking smart card.

The proliferation of simple plastic cards started in the USA in the early 1950s. The low price of the synthetic material PVC made it possible to produce robust, durable plastic cards that were much more suitable for everyday use than the paper and cardboard cards previously used, which could not adequately withstand mechanical stresses and climatic effects.

The entry of Visa and MasterCard into the field led to a very rapid proliferation of 'plastic money' in the form of credit cards. This occurred first in the USA, with Europe and the rest of the world following a few years later. Today, credit cards allow travelers to shop without cash everywhere in the world.

At first, the functions of these cards were quite simple. They served as data storage media that were secure against forgery and tampering. General information, such as the card issuer's name, was printed on the surface, while personal data elements, such as the cardholder's name and the card number, were embossed. Many cards also had a signature panel where the cardholder would sign his or her name for reference. The embossed characters on the card can be transferred to paper using simple, inexpensive devices, and they can also be easily read visually (by humans).

The first improvement consisted of a magnetic stripe on the back of the card. The fundamental disadvantage of embossed cards is that their use creates a flood of paper receipts, which are expensive to process. One remedy for this problem is to digitally encode the card data on a magnetic stripe located on the back of the card. Although the storage capacity of the magnetic stripe is only about 1000 bits, which is not very much, it is nevertheless more than sufficient for storing the information contained in the embossing. In this way paper-based transactions were replaced by electronic data processing. This required a different method to be used for user identification, which previously employed the user's signature. The method that has come into widespread general use involves a secret personal identification number (PIN) that is compared with a reference number. The reader is surely familiar with this method from using bank machines (automated teller machines). The main drawback of magnetic-stripe technology is that the stored data can be altered very easily. Manipulating embossed characters requires at least a certain amount of manual dexterity, and such manipulations be easily detected by a trained eye. By contrast, the data

recorded on the magnetic stripe can be altered relatively easily using a standard read/write device, and it is difficult to afterwards prove that the data have been altered. Furthermore, magnetic-stripe cards are often used in automated equipment in which visual inspection is not possible, such as cash dispensers. Most systems that employ magnetic-stripe cards thus use online connections to the system's host computer for reasons of security, even though this generates significant costs for the necessary data transmissions. In order to reduce costs, it is necessary to find solutions that allow card transactions to be executed offline without endangering the security of the system.

The development of the smart card, combined with the expansion of electronic data-processing systems, has created completely new possibilities for devising such solutions. Thanks to enormous progress in microelectronics, in the 1970s it was possible to integrate data storage and processing logic on a single silicon chip measuring a few square millimeters. Smart card characteristic feature is an integrated circuit embedded in the card, which has components for transmitting, storing and processing data. The data can be transmitted using either contacts on the surface of the card or electromagnetic fields, without any contacts.

The first patents were in 1968 and 1970 but the first real progress in the development of smart cards came when Roland Moreno registered his smart card patents in France in 1974. The great breakthrough was achieved in 1984, when the French PTT (postal and telecommunications services agency) successfully carried out a field trial with telephone cards. In this field trial, smart cards immediately proved to meet all expectations with regard to high reliability and protection against manipulation. Significantly, this breakthrough for smart cards did not come in an area where traditional cards were already used, but in a new application.

One of the most important advantages of smart cards is that their stored data can be protected against unauthorized access and manipulation. Such confidential data can be processed only internally by the chip's processing unit. In principle, both hardware and software mechanisms can be used to restrict the use of the storage functions of writing, erasing and reading data and tie them to specific conditions. This makes it possible to construct a variety of security mechanisms, which can also be tailored to the specific requirements of a particular application. In combination with the ability to compute cryptographic algorithms, this allows smart cards to be used to implement convenient security modules that can be carried by users at all times, for example in a purse or wallet. Some additional advantages of smart cards are their high level of reliability and long life compared with magnetic-stripe cards, whose useful life is generally limited to one or two years.

Smart cards can be divided into two groups, which differ in both functionality and price: memory cards and microprocessor cards. Presently, both memory cards and microprocessor cards are available as contactless cards.

In a memory cards the data needed by the application are stored, as the name suggests, in a memory, which is usually EEPROM. Access to the memory

is controlled by the security logic, which in the simplest case consists only of write protection or erase protection for the memory or certain memory regions. However, there are also memory chips with more complex security logic that can also perform simple encryption. The functionality of memory cards is usually optimized for a particular application. Although this severely restricts the flexibility of the cards, it makes them quite inexpensive. Memory cards are typically used for prepaid telephone cards and health insurance cards.

In a microprocessor card the heart of the chip, as the name suggests, is a processor, which is usually surrounded by four additional functional blocks: mask ROM, EEPROM, RAM and an I/O port. The mask ROM contains the chip's operating system, which is 'burned in' when the chip is manufactured. The content of the ROM is thus identical for all the chips of a production run, and it cannot be changed during the chip's lifetime. The EEPROM is the chip's non-volatile memory. Data and program code can be written to and read from the EEPROM under the control of the operating system. The RAM is the processor's working memory. This memory is volatile, so all the data stored in it are lost when the chip's power is switched off. The serial I/O interface usually consists only of a single register, via which data are transferred bit by bit. Microprocessor cards are very flexible in use. In the simplest case, they contain a program optimized for a single application, so they can only be used for this particular application. However, modern smart card operating systems allow several different applications to be integrated into a single card. In this case, the ROM contains only the basic components of the operating system, with the application-specific part of the operating system being loaded into the EEPROM only after the card has been manufactured. Recent developments even allow application programs to be loaded into a card after it has already been personalized and issued to the cardholder. Special hardware and software measures are used to prevent the security conditions of the individual applications from being violated by this capability. Hardware attacks and, on the other hand, hardware countermeasures that will be proposed in this work are all related to applications based on microprocessor smart cards.

Finally, contactless cards, in which energy and data are transferred without any electrical contact between the card and the terminal, have achieved the status of commercial products in the last few years. As contactless cards can work at a distance from the terminal ranging from few centimeters to a meter, this means that such cards do not necessarily have to be held in the user's hand during use, but can remain in the user's purse or wallet. Contactless cards are thus particularly suitable for applications in which persons or objects should be quickly identified such as access control, local public transportation, ski passes, airline tickets. However, there are also applications where operation over a long distance could cause problems and should thus be prevented, as for example for electronic purse application.

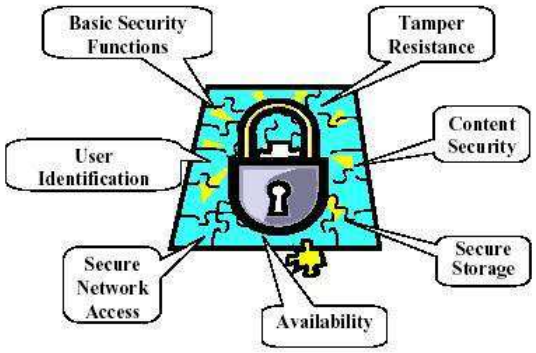


Figure 1.2: Common security requirements of embedded system from an end-user perspective.

1.3 An overview on side-channel attacks

Information technology is evolving at an amazing pace. Personal computers, fax machines, pagers, and mobile phones are in the hands of millions of people worldwide. Similarly, interest in smart card technology has soared in the 1990's, and by the year 2000 the number and variety of smart card-based applications exploded around the world.

In an increasing number of large and important smart card-based systems, from pay-TV through GSM mobile phones and prepaid gas meters to electronic wallets, smart cards are used by millions of cardholders worldwide in more than 90 countries, primarily in Europe and the Far East, processing point-of-sale transactions, managing records, and protecting computers and secure facilities.

For these reasons information engineers have developed an increasing interest in the tamper resistance properties of smart cards and other special purpose security processors. Tamper resistance is not absolute: an attacker with access to semiconductor test equipment can retrieve key material from a smart card controller by direct observation and manipulation of the chip's components. It is generally believed that, given sufficient investment, any chip-sized tamper resistant device can be penetrated in this way.

This is the reason for the increasing interest in new attack methods on one side and in new countermeasures and new cryptographic algorithms on the other side.

Cryptographic algorithms are building blocks of many security protocols and can be implemented both in software and hardware. Software solutions are cheaper and more flexible, while hardware implementations provide higher speed and intrinsic security. A trade-off in cost and speed can be achieved by hardware-software co-design.

On the other hand, the security of the implementation also needs to be considered. Namely, attacks on cryptographic algorithms are usually divided into

mathematical and *implementation attacks*. The latter are based on weaknesses in the implementation and can be passive or active. *Passive attacks* benefit from side channel information, which is collected by measuring some physical quantity.

More precisely, while secret data are being processed they can be deduced by observing execution time, power consumption, electromagnetic radiation, etc.

The second class of implementation attacks, i.e. the *active attacks*, is more invasive as they are based on the introduction of faults, which result in erroneous calculations leading to the exposure of the secret key. The usual cause of these faults can be sudden changes, i.e. glitches, in various parameters such as power supply, clock, temperature, etc. An attacker could also use a light flash with equipment such as a camera flash or a laser in order to induce a fault.

Cryptographic devices have several physical and logical interfaces and therefore a second criterion for categorizing an attack on a cryptographic device is the interface that is exploited by the attack. It is possible to distinguish between invasive and non-invasive attacks.

An *invasive attack* is the strongest type of attack that can be mounted on a cryptographic device. An invasive attack typically starts with the depackaging of the device. Subsequently, different components of the device are accessed directly using a probing station. This part of an invasive attack can be passive if the probing station is only used to observe data signals or active if signals in the device are changed to alter the functionality of the device. In a *non-invasive attack*, the cryptographic device is essentially attacked as it is and only directly accessible interfaces are exploited. Most non-invasive attacks can be conducted with relatively inexpensive equipment, and hence, these attacks pose a serious practical threat to the security of cryptographic devices.

Passive non-invasive attacks are often also referred to as *side-channel attacks*. The three most important types of side-channel attacks are *timing attacks* [51], *power analysis attacks* [53], and *electromagnetic attacks [42, 80]*. The basic idea of these attacks is to determine the secret key of a cryptographic device by measuring its execution time, its power consumption, or its electromagnetic field.

Side-channel attacks can be also active non-invasive attacks. The goal of these attacks is to insert a fault (for example clock or power glitches) in a device without depackaging it.

Regarding attacks, this thesis focuses exclusively on power analysis attacks even if a general overview also of the other types of attacks is treated in the following.

1.3.1 Timing analysis attacks

Timing analysis attacks are based on the fact that algorithms with a non-constant execution time can leak secret information. A non-constant execution time can be caused by conditional branches in the algorithm, various optimization techniques, cache hits, etc. Unlike power attacks, the use of these attacks is not restricted to cryptographic tokens. Timing attacks can also be applied to

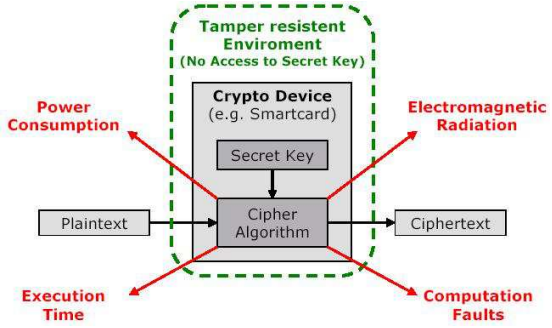


Figure 1.3: Side-channel attack types.

network based cryptosystems and to other applications whenever the attacker can get hold of timing information.

The obvious way to prevent timing attacks is to implement cryptographic algorithms with a constant execution time. Almost all modern implementations are resistant against timing attacks, which makes a timing-only attack impossible. However, the threat remains in combining timing information with other side-channels. For example, timing information can be used by an attacker in order to locate specific parts of the algorithm.

1.3.2 Power analysis attacks

Nowadays, CMOS is by far the most commonly used technology to implement digital integrated circuits. The dominant factor for the power consumption of a CMOS gate is the dynamic power consumption. Two types of power consumption leakage can be observed. The transition count leakage gives information about the number of changed bits, while the Hamming weight leakage is related to the number of 1-bits being processed simultaneously.

Two types of power analysis attacks are distinguished. In a simple power analysis (SPA) attack, an attacker uses the side-channel information from one measurement directly to determine (parts of) the secret key. In differential power analysis (DPA), many measurements are used in order to filter out noise (Figure 1.4). While SPA exploits the relationship between the operations that are executed and the power leakage, DPA exploits the relationship between the processed data and the power leakage.

In DPA, an attacker uses a so-called hypothetical model of the attacked device (see Figure 1.5).

The model is used to predict several values for the side-channel output of a device. These predictions are compared to the measured side-channel output of the device. The most popular comparison ways are the distance-of-mean test and the correlation analysis. For both of these attacks, the model predicts the amount of side-channel leakage for a certain moment of time in the execution.

In Chapter 2, which is fully focused on the power analysis, more details

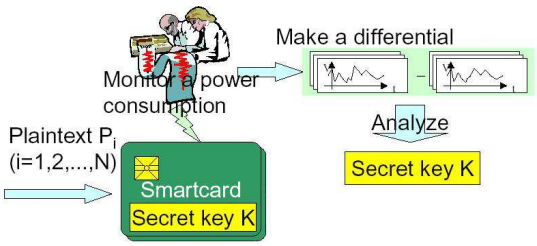


Figure 1.4: Differential Power Analysis to discover the secret key.

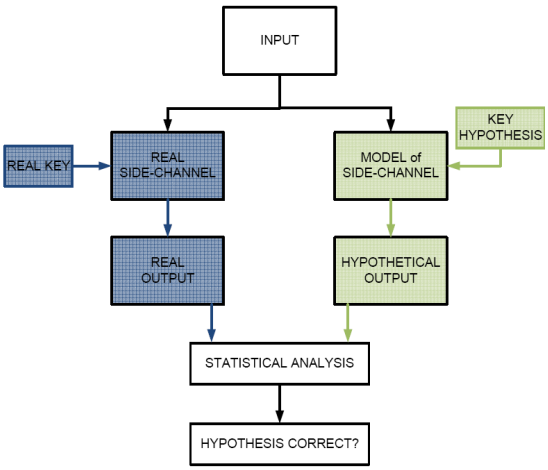


Figure 1.5: DPA behaviour.

about the SPA and DPA will be given.

1.3.3 Electromagnetic analysis (EMA) attacks

A most recently published side-channel attack is based on electromagnetic emanation. The attack exploits the information in the electromagnetic field that is caused by the currents flowing in each component of the cryptographic device.

There are two types of emanations: intentional and unintentional [2]. The first type results from direct current flows. The second type is caused by various couplings, modulations (AM and FM), etc.

It is well known that the US government has been aware of electromagnetic leakage since the 1950s. The first published papers are work of Quisquater and Samyde [80] and Gemplus team [42]. Quisquater and Samyde showed that it is possible to measure the electromagnetic radiation from a smart card. Their measurement setup consisted of a sensor which was a simple flat coil (see Figure 1.6), a spectrum analyzer or an oscilloscope and a Faraday cage. Quisquater also introduced the terms Simple EMA (SEMA) and Differential EMA (DEMA).

Figure 1.6: An example of electromagnetic analysis setup.

Gemplus observed the feasibility of EMA attacks and compared them with PA attacks in favour of the first. Namely, EM emanation can also exploit local information and, although more noisy, the measurements can be performed from a distance. This fact broadens the spectrum of targets to which side-channel attacks can be applied. Of concern are not only smart cards and similar tokens but also SSL accelerators and many other cryptographic devices.

The two papers mentioned above deal exclusively with intentional emanations. On the contrary, the real advantage over other side-channel attacks lies in exploring unintentional emanations [2]. More precisely, EM leakage consists of multiple channels. Therefore, compromising information can be available even for DPA resistant devices for which no contact measurements are available.

Namely, besides carefully exploring all available EM emanations an attacker can also focus on a combination of two or more side-channels. Agrawal et al. defined these so-called multi-channel attacks in which the side-channels are not necessarily of a different kind [12]. For example, they discussed combined power and EM analysis but also multi-channel DPA attacks.

1.3.4 Fault attacks

Fault attacks are based on hardware faults; they were introduced by Boneh et al. [17]. The attack is based on the following idea: if a wrong result is released the adversary can use that information to break the cryptosystem.

Boneh et al. classified the faults into three categories. The first type are transient faults which can occur randomly causing a faulty computation to be executed. The second type are latent faults, which are hardware or software bugs that are difficult to locate. The third type are induced faults for which physical access to the hardware is necessary. These are the most interesting because of the active role of the attacker. For example, optical fault induction attacks, as introduced by Scorabogatov and Anderson [87], use a flashgun targeting a transistor to change the state of a memory cell in a microcontroller. The authors have proved this optical probing to be feasible as they managed to change an arbitrary bit of an SRAM array. They also suggested to use self-timed dual-rail logic as a defense to these attacks, which could also help against power analysis attacks.

Fault attacks can be considered as the almost dangerous implementation attacks as countermeasures usually include complex techniques which are not easy to implement on constrained environment such as smart cards.

1.4 An Overview on countermeasures

Since differential power analysis constitutes a real threat to chipcard security and attacks using DPA have been proved to be successful at breaking many industry-standard cryptographic algorithms, a plethora of countermeasures has been proposed to prevent power analysis attacks on chipcards.

The countermeasures against DPA attacks that have been published so far can essentially be categorized into two groups: *hiding* and *masking*. The basic idea of hiding is to remove the data dependency of the power consumption. This means that either the execution of the algorithm is randomized or the power consumption characteristics of the device are changed in such a way that an attacker cannot easily find a data dependency. The basic idea of masking is to randomize the intermediate values that are processed by the cryptographic device.

A second criterion for categorizing a countermeasure is the level at which the countermeasures are implemented, algorithmic or hardware level. On the *algorithmic* (*software*) level, random masking of intermediate variables [46] is, for example, a widely exploited technique. These are platform-dependent countermeasures and, usually, a substantial processing-time overhead is needed. Hardware countermeasures can be classified according to the involved abstraction level during the design flow.

System-level techniques include adding noise to the device power consumption [28], duplicating logics with complementary operations [94], active supply current filtering with power consumption compensation, passive filtering, battery on chip and detachable power supply [85]. Observe that some of mentioned techniques have a pure theoretical interest since, due to technological and cost constraints, they cannot be employed to design tamper resistant chipcards.

Gate-level countermeasures include circuital techniques which can be implemented using logic gates available in a standard-cell library, e.g. random

masking [43], random pre-charging [21], state transitions and Hamming weights balancing.

At last, the *transistor-level* approach to prevent DPA is based on the adoption of a logic family whose power consumption is independent of the processed data. In literature, many differential and dynamic logic families are available which are suitable to design DPA resistant cryptographic hardware [96].

Although this technique is widely recognized as the most powerful DPA countermeasure, its cost in term of design time is usually very high since a full-custom approach is required.

Finally, in this work (see Chapter 5) hardware transistor-level countermeasures have been considered because they are platform-independent countermeasures and so they have any applicability algorithm.

1.4.1 Algorithmic countermeasures

The basis of power analysis attacks is elementary computations that depend on a part of the secret key and on the input or output data, which are assumed to be known.

Until recently, most of these attacks exploited some specific features of software implementations of cryptographic algorithms, and many countermeasures were designed at a software level. One of the most powerful software techniques to counteract such attacks is to mask all input and intermediate data in order to de-correlate any information leaked through side channels from actual secret data being processed [32]. The idea is simple: the message and the key are masked with some random values at the beginning of computations, and thereafter everything is almost as usual. Of course, the value of the mask at the end of some fixed step must be known in order to re-establish the expected value at the end of the execution. In section 1.4.2.2 is showed how to apply data masking technique at the level of micro operations such as a logical AND.

A typical masking example is to mask implementations of the Advanced Encryption Standard (AES) [70].

The *Advanced Encryption Standard* is a round-based symmetric block cipher. The standard key size is 128 bits, but for some applications 192 and 256-bit keys must be supported as well. The round consists of four different operations namely SubByte, ShiftRow, MixColumn and AddRoundKey, that are performed repeatedly in a certain sequence; each operation in a standard algorithm maps a 128-bit input state into a 128-bit output state. The state is represented as 4x4 matrix of bytes. The number of rounds depends on the key size. In the decryption process, the inverse operations of each basic function are executed in a slightly different order. Figure 1.7 illustrates the general structure of the AES encryption algorithm.

Except for SubBytes(), all transformations are linear, i.e., they have the property that $f(x+m) = f(x)+f(m)$. Hence, for such transformations, it is an easy task to compute how the mask m has changed during the transformation. As a consequence, it is also simple to re-establish the original mask m after a

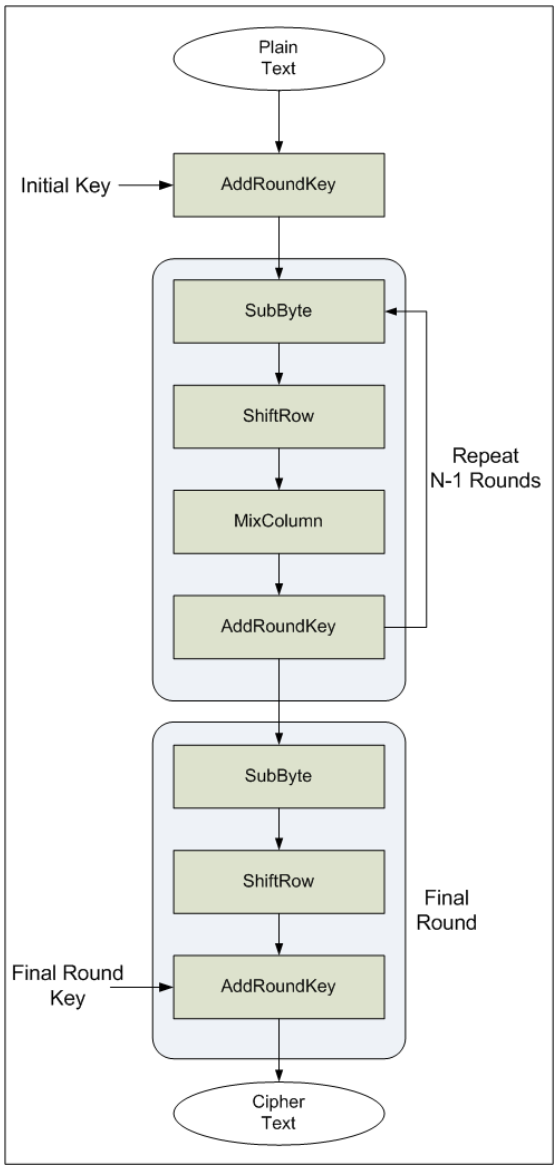


Figure 1.7: Basic operations of the Advanced Encryption Standard algorithm.

(sequence of) transformation(s).

More difficult is the transformation SubBytes(). In fact, as SubBytes() is non-linear, re-establishing the mask after a SubBytes() transformation is difficult. This is because $SubBytes(x + m) = SubBytes(x) + m' \neq SubBytes(x) + SubBytes(m)$. Consequently, SubByte is the main building block of AES. It replaces each byte in a state by its substitute in an S-box that comprises a composition of two transformations: first, each byte in a state is replaced with its reciprocal in the finite field GF(28), except that 0, which has no reciprocal, is replaced by itself and this is the only non-linear function in the AES algorithm; then an affine transformation f is applied. The S-box is usually implemented as a look-up table consisting of 256 entries; each entry is 8 bits wide. Finally, the round key is computed in parallel to the round operation. It is derived from the cipher key by means of key expansion and round key selection operations, which are similar to those of the round operation.

In the following, a method that can be used to mask AES SubBytes() (presented in [73]) is showed. The SubBytes() transformation is implemented as a lookup table. In [73] a masking is proposed such that $SubBytes(x + m) = SubBytes(x) + m$.

In order to achieve this with a lookup table the authors in [73] have to compute a corresponding table MaskedSubBytes() for the mask m as showed in the following algorithm:

INPUT: m
OUTPUT: MaskedSubBytes(x + m) = SubBytes(x) + m,
1: for i = 0 to 255 do
2: MaskedSubBytes(i + m) = Subbytes(i) + m
3: end for
4: Return(MaskedSubBytes)

Such a table needs to be computed for each mask value m_i. If m_i different masks are used, then the complexity of this procedure is $i \cdot 256$. If we ensure that the same masks m_i are re-established before the SubBytes() operation in each round the same i tables can be used throughout the complete AES calculation.

1.4.2 Hardware countermeasures

As shortly discussed in Section 1.4, hardware countermeasure are classified according to the involved abstraction level during the design flow. In the following, examples in the different levels will be briefly showed.

1.4.2.1 System-level countermeasures

The approaches shortly discussed in Section 1.4.1 are valid but it is extremely difficult to guarantee perfect equalization and high-resolution. Moreover the algorithm countermeasures (and the other countermeasures presented in the following) require changes in the implementation of crypto-hardware or require the algorithm itself to be altered in some way.

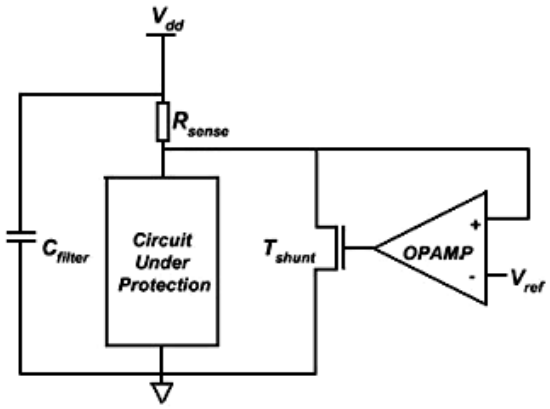


Figure 1.8: Block diagram of the suppression circuit.

System level countermeasures, instead, in general is an approach which does not affect the algorithm implementation in any way. For example Shamir in [85] proposed a method in which the power supply is isolated from the cryptographic hardware of a smart card by using capacitors to supply the current. Two capacitors are used; one supplies current to the chip while the other recharges itself. Examination of the power line will only show a pattern resulting from the charging of the internal supply capacitors.

Another approach, proposed by Ratanpal et al. [83], acts as an auxiliary circuit to increase the difficulty of power analysis attacks against the underlying hardware. This paper presents the DPA suppression circuit which can be added to existing hardware devices implementing cryptographic algorithms. The idea is DPA obtains information in the form of variations in the supply current. If these variations are attenuated, DPA is more difficult to be performed. In other words, the attacker requires more power samples to distinguish the correlation spikes from noise. The circuit showed in Figure 1.8 works as follow: instantaneous current drawn by the device under protection is sensed, and an appropriate current is shunted so that the total current drawn from the supply shows less variation. Ideally, a circuit with infinite bandwidth and zero response time should completely flatten the power supply current trace, and thereby make DPA ineffective.

Obviously, while this ideal behaviour is not possible with practical circuits, feasible circuits can suppress supply current variations enough to make the number of power traces required for DPA prohibitively large. A bit in detail, from Figure 1.8 it is possible to see that when the current demand of the device decreases, the voltage at the bottom of R_{sense} starts to rise. This results in an increase in the output voltage of the op-amp. As a result, transistor M_{shunt} starts drawing more current and pulls the voltage down. Essentially, the feedback loop forces the voltage at the bottom of R_{sense} to a constant reference voltage V_{ref} . The value of the peak current drawn by the device can be set

through V_{ref} to ensure continuous operation of the negative feedback loop. The feedback loop, however, has limited bandwidth. High frequency components of the power signal are filtered through the $R{\cdot}C$ combination of R_{sense} and C_{filter}.

1.4.2.2 Gate-level countermeasures

The masking schemes, which have been discussed in 1.4.1, are all algorithmic countermeasures and can be implemented either in software or in hardware.

However, masking on gate level using a special masked logic has been proposed as well. In [97] and [43] it has been shown how to apply data masking techniques at the level of micro operations such as logical AND, XOR, etc., and how to use these operations as building blocks for implementation of inversion in composite fields directly on masked data.

As it has been already explained, the only operations which are tricky to mask occur during the computation of the AES S-box. In particular, if the S-box is implemented in composite field arithmetic, it is the AND gate which is difficult to mask.

The main contribution of the just cited papers is therefore the design of a masked AND gate and its clever use to secure implementations of the AES S-box. In these papers the bitwise XOR ($\oplus$) random masking is considered, in which data x is randomized into $x \oplus r_x$ by a random mask r_x. In this case the output mask for $z = x \oplus y$ is the bitwise XOR of the two input masks, for x and y. Then, if $z = f(x)$, for an elementary function f, and if we want to obtain a masked output $\tilde{Z} = z \oplus r_z$ from a masked input $\tilde{X} = x \oplus r_x$, then the function f has to be modified into a new, masked function $\tilde{F}$ so that $\tilde{Z} = \tilde{F}\left(\tilde{X}, r_x, r_z\right)$.

The problem considered, for example, in [64] is how to compute securely in hardware, on the logic gate level, the masked function AND. For the AND function $z = f(x, y) = x \bullet y$, by applying the distributive property and by grouping the terms appropriately, it is obtained

$$\tilde{Z} = z \oplus r_z = \left(\left((r_z \oplus (r_x \oplus r_y)) \oplus \left(r_x \bullet \tilde{Y}\right)\right) \oplus \left(r_y \bullet \tilde{X}\right)\right) \oplus \left(\tilde{X} \oplus \tilde{Y}\right) \quad (1.1)$$

in which all the computations are secure (see Figure 1.9).

Another example of hardware countermeasure at gate level is the technique proposed in [19], based on the insertion of random delays on the input signals of each pipeline stage of the processor data-path thus resulting in a randomization of the charge quantity from the power supply. In Figure 1.10 the proposed technique to scramble the current consumption is shown: considering a generic pipeline stage in a cryptographic processor, each input to the combinatorial network is delayed by a random time $\Delta_i = 1, ..., N$. That allows to introduce a variance on the current supply waveform and, above all, the charge quantity transferred during a clock cycle is also decorrelated from the processed data thanks to the modification of the internal sequence of transitions caused by the input delays.

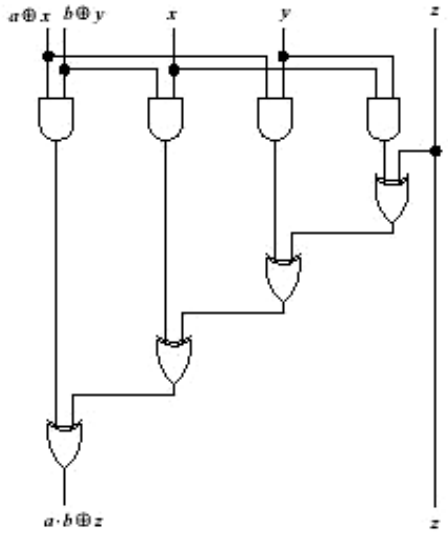


Figure 1.9: Example of masked AND.

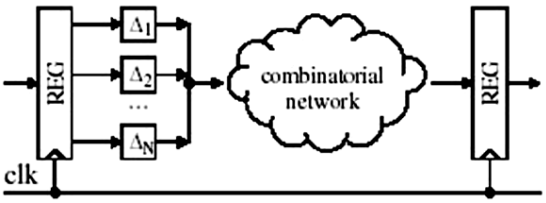


Figure 1.10: Pipeline stage with random delays.

In [19] details about the practical implementation are also reported. In fact, it is also shown how the proposed technique can be easily adopted in the framework of a standard semi-custom design flow. For example a randomized propagation delay D flip-flop is presented in which the delay time depends on a random bit R (see equation (1.2) and Figure 1.11).

$$t_{pd} = t_{pd}^{FF} + t_{pd}^{MUX} + \Delta \cdot R \tag{1.2}$$

For all these presented techniques the presence of a *true random number generator* (*RNG*) is required (see Chapter 6 for details about RNG's). Many works have been published in the technical literature on the implementation of RNG modules ([14], [21], [22], [15] and [25]). In particular, in [23] a new concept for an oscillator-based RBG (Random Bit Generator) which exploits the relative jitter between two identical ring oscillators sharing the same delay elements is proposed. The oscillators start synchronously and a detecting circuit signals when a relative jitter greater than a given threshold has been accumu-

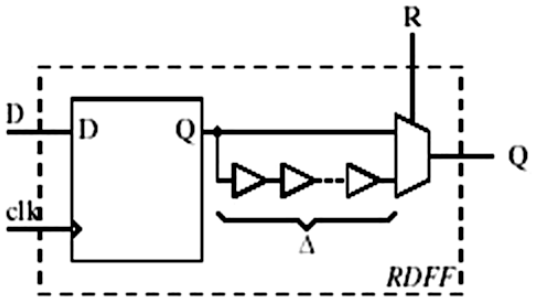


Figure 1.11: Random D flip-flop (RDFF).

lated. Therefore, the generation rate is automatically adapted to the available noise and no pseudo-random patterns are generated thus improving the module testability [24].

1.4.2.3 Transistor level countermeasures

The transistor-level approach is based on the adoption of a logic style whose power consumption is constant or independent of the processed data.

Static Complementary MOS logic, which is the default logic style in standard cell libraries used for security IC's, only consumes energy from the power supply when its output has a 0-1 transition. In fact, during the 1-0 transition the energy previously stored in the output capacitance is dissipated and in the two events of a 0-0 or a 1-1 transition no power is used. This asymmetric power demand provides the information used in DPA to find the secret key.

A logic style with data-independent power consumption does not reveal this information. When logic values are measured by charging and discharging capacitances we need to use a fixed amount of energy for every transition. In this way, Tiri et al. in [96] proposed the *Sense Amplifier Based Logic* (*SABL*), a logic style that uses a fixed amount of charge for every transition, including the degenerated events in which a gate does not change state. In every cycle, a SABL gate charges a total capacitance with a constant value. SABL is based on two principles. First, it is a dynamic and differential logic style and therefore has exactly one switching event per cycle and this independently of the input value and sequence. Since a differential logic family uses the direct and the negated representation of the input and output signals and a dynamic logic family alternates precharge and evaluation phases, both outputs are precharged to 1 in the precharge phase and exactly one of the two outputs evaluates to 0 in the evaluation phase. Second, during a switching event, it guarantees that the load capacitance has a constant value. SABL completely controls the portion of the load capacitance that is due to the logic gate. The intrinsic capacitances at the differential input and output signals are symmetric and additionally it discharges and charges all the internal node capacitances trough a special pull down network.

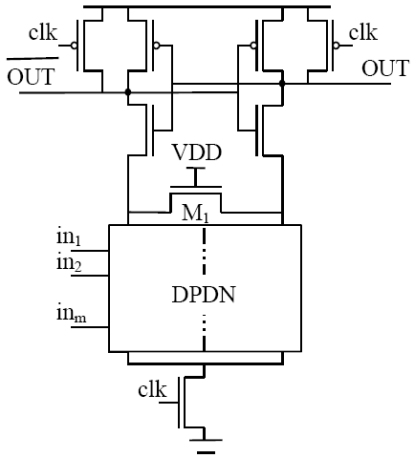


Figure 1.12: SABL inverter.

The SABL gate is based on the StrongArm110 flip-flop (SAFF) [72]. To realize a basic gate, Tiri keeps the sense amplifier half flip-flop and replaces the input differential pair by a differential pull down network (DPDN). Figure 1.12 depicts the SABL inverter gate.

During the evaluation phase (clk high) the cross-coupled inverter will toggle to one state and provides a path to ground. The transistor in the middle with the gate connected to supply (always on) prevents a floating node by serving as a path for subthreshold currents and guarantees that all internal nodes are discharged. Under the assumption that the differential signals propagate in the same environment, the interconnect capacitances are equivalent. Therefore the total output capacitance is a constant. During the precharge phase (clk low) all the discharged nodes and capacitances will be charged. Since in every cycle the same capacitances are discharged and charged that makes the power consumption of the gate independent of the input statistics.

Topics covered in this section will be taken up and further elaborated in Chapter 5.

1.5 Contents of this thesis work

This thesis work reports the results obtained during a Ph.D. in electronic engineering on both attacks and countermeasures side which this kind of research offers.

More in detail, Chapter 2, starting from CMOS powtheer consumption, covers simple and differential power analysis which are the state of the art regarding hardware attacks.

In Chapter 3, an active circuit (the so called Supply and Current Measurement, or SCM) which promises to improve this kind of attack is proposed and

experimental results are reported and the achievable advantage in terms of sensitivity is discussed too. A complete comparison with the standard measurement is performed to underline the improvement achievable using the proposed setup. Finally, real attacks are performed first on a simple crypto core and than on a FPGA implementing a part of the DES algorithm. The obtained experimental results are compared showing the advantages of the proposed solution with respect the standard attack setup.

The setup proposed in Chapter 3 aims to improve the power attacks which are based on dynamic power consumption of the circuits under attack. In Chapter 4, the possibility to use leakage (static) current as a novel side-channel is explored. Assumptions about information leakage in CMOS logic gates due to static current are validated with experimental results and therefore a novel class of attacks is proposed, called Leakage Power Analysis (LPA). A complete analysis of LPA effectiveness under process variations is addressed also in presence of DPA resistant logic styles.

In Chapter 5, hardware countermeasures against power analysis are considered. More in detail, after an introduction about the state of the art on countermeasures at different levels, the work is focused on the transistor level countermeasures by proposing three new logic families for cryptographic applications in which each one offers advantages over the previous one.

Chapter 6 is focused on the Random Number Generators (RNG), hardware building blocks of many cryptographic systems, which generate sequences of random numbers (bits). After a short overview on RNGs, two different works are reported. The first activity is focused on a power attack on a chaos-based Random Number Generator which is designed and realized by the research group of professor Setti of the University of Bologna. The aim of this attack is to verify if it is possible to retrieve information regarding the internal state of the chaotic system used to generate the random bits. As second activity, measures on implementation of a RNG designed by Infineon Technologies (design center in Graz, Austria) are shown proving the randomness of the generated sequence of bits.

Finally, the conclusions of this research work are drawn in Chapter 7.

Chapter 2

Power analysis

2.1 Power consumption of CMOS circuits

The total power consumption of a CMOS circuit is the sum of the power consumptions of the single logic cells which compose the circuit. Hence, the total power consumption essentially depends on the number of logic cells in a circuit, the connections between them, and how the cells are built.

When operating a CMOS circuit, the circuit is provided with the constant supply voltage V_{DD} and with input signals. The logic cells in the circuit process the input signals and draw current from the power supply.

Most modern cryptographic devices are implemented in CMOS logic. The power consumption of CMOS logic gates can essentially be divided into two parts.

The first part is the *static power consumption* P_{stat}. This is the power that is consumed if there is no switching activity in a cell. When a MOS transistor is off current is not exactly zero but a weak (leakage) current flows through the channel producing a power consumption. We denote this leakage current by I_{leak}. Hence, the static power consumption P_{stat} can be calculated according to (2.1). This phenomenon is increasing significantly for modern process technologies with very small channel length.

$$P_{stat} = I_{leak} \cdot V_{DD} \tag{2.1}$$

The second part of the power consumption is the *dynamic power consumption* P_{dyn} composed by switching and short-circuit power consumption. Let us use the simplest CMOS logic cell, the CMOS inverter, to describe these two contributions. As depicted in Figure 2.1, the inverter contains two transistors which act as voltage controlled switches. When the input voltage to the inverter is high, the top switch opens while the bottom switch closes. This grounds the inverter's output and so it goes low. Conversely, when the input voltage is high, the top switch closes and bottom opens setting the output to +V which thus goes high. When performing output $0 \rightarrow 1$ transitions, the CMOS inverter

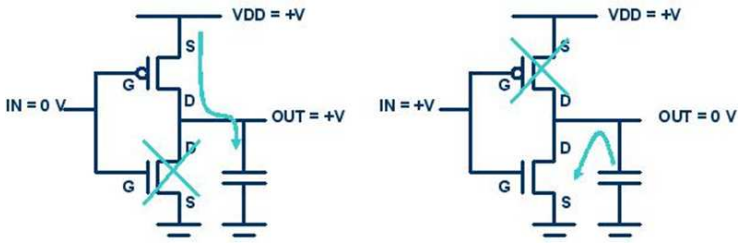


Figure 2.1: CMOS inverter power dissipation: 0-1 (left) and 1-0 (right) output transition.

draws a charging current from the power supply to charge the output capacitance C_L. On the contrary, when performing output $1 \rightarrow 0$ transitions, the CMOS inverter discharges current from the capacitance C_L to power ground. This current flow generates switching power consumption. The average charging power P_{chrg} that is consumed by a cell during the time T can be calculated as shown in (2.2) in which $p_{chrg}(t)$ denotes the instantaneous charging power consumed by the cell, f is the clock frequency, and a is the so-called activity factor of the cell. The activity factor corresponds to the average number of $0 \rightarrow 1$ transitions that occur at the output of a cell in each clock cycle.

$$P_{chrg} = \frac{1}{T} \int_0^T p_{chrg}(t)dt = \alpha \cdot f \cdot C_L \cdot V_{DD}^2 \tag{2.2}$$

Moreover, there is a brief instant, when the inverter is in transition between states, when both transistors conduct current. This causes a short circuit from +V to the ground which temporarily dissipates (short-circuit) power. The average power consumption P_{sc} that is caused by the short-circuit currents in a cell during the time T can be calculated as shown in (2.3) in which $p_{sc}(t)$ is the instantaneous short-circuit power consumed by a cell, I_{peak} denotes the value of the current peak that is caused by the short circuit during the switching event and t_{sc} is the time for which the short circuit exists.

$$P_{sc} = \frac{1}{T} \int_0^T p_{sc}(t)dt = \alpha \cdot f \cdot V_{DD} \cdot I_{peak} \cdot t_{sc} \tag{2.3}$$

2.2 Power Analysis

In 1996 Kocher suggested the idea that power consumption of a cryptographic token can convey sensitive information to an adversary [51]. There, Kocher noted that padding the execution time of operations with dummy computations (e.g., empty loops) may be an ineffective defense against timing attacks since the power consumption of dummy computations can be much different from meaningful ones. In this case, an adversary could plot, or trace, the power consumption of a token as it executes a particular operation and then educe a

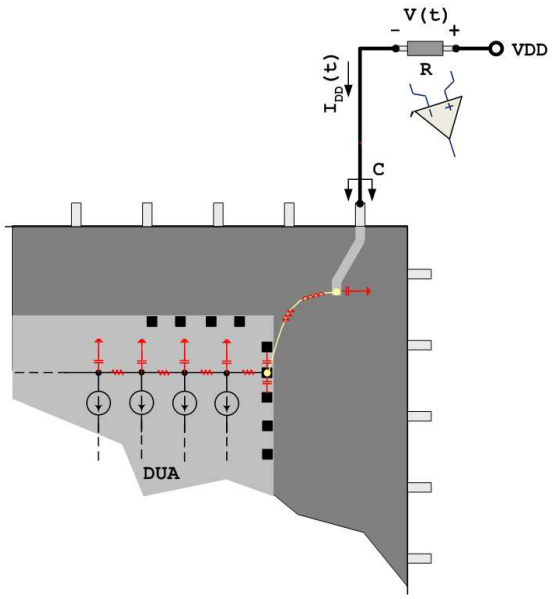


Figure 2.2: Power attack in practice.

valid timing measurement from the length of the initial pattern in the trace. It is not difficult to imagine a situation where an adversary might have the opportunity to collect power consumption data.

In 1998, Kocher and the results of his research were again featured in the New York Times [W98]. The story there summarized some of the details concerning power analysis that Kocher had recently announced. One particularly startling claim was that for some tokens, a power trace of a single cryptographic operation is enough to completely reveal the value of an embedded secret key. Even more startling was the claim that by examining roughly 1000 power traces Kocher and his employees had managed to break every smart card product they had examined in the last year and a half.

As more technical details [52] concerning these discoveries were released, it became clear that power analysis was a serious threat to the security of cryptographic tokens.

To better explain this concept, it is sufficient to consider that the amount of current drawn by electronic devices from a power source during their operation varies depending on the performed computations. To measure the current consumption, a small resistor (approximately $10 - 50\Omega$) is connected in series with the device's power supply. An oscilloscope can be used to measure the voltage difference across the resistor and the current can then be deduced using Ohm's law (see Figure 2.2). Digital oscilloscopes can be used to sample voltage signals at high frequencies giving a trace of the current over an interval of time.

The source of current for most devices is supplied at a constant voltage and

so the power dissipated by these devices is proportional to the flow of current through them. Because of this, power analysis attacks work just as well with current measurements as they do with power measurements. Hence, the only difference between a power analysis attack and a current analysis attack is a constant factor.

The most dominant source of power dissipation (for technology nodes up to 100nm) is usually caused by the charging and discharging of internal capacitive loads attached to gate outputs. A thorough discussion of all three factors is given in [106] and [33].

Power consumption information is useful to an adversary because it is correlated to the calculations the token is performing. Power analysis attacks come in many variants.

The basic PA technique, known as *simple power analysis* (SPA) [51] monitors the power supply of a cryptoprocessor while executing a known encryption algorithm and correlates the time-domain current waveform with various phases of the algorithm. If the algorithm has a key dependent execution flow, for instance if it has branches dependent on the value of key bits, the current waveform presents easily recognizable features that reveal key information. SPA has limited effectiveness if the algorithm flow is data-independent, but it can be effective to crack naive implementations.

Differential power analysis (DPA) [53], and its variants such as higher-order power analysis [59], is significantly more dangerous, in fact it is effective even when execution flow is not data-dependent and for some encryption algorithms it does not even require knowledge of the plaintext.

To perform DPA, an attacker needs a collection of m power traces $T_i[j]$, $i = 1, \ldots, m$ (j is the discrete time index of the values in the trace) and their corresponding ciphertext values C_i. The critical step in applying DPA is the definition of a selection function $D(K_b, C_i) \rightarrow \{0, 1\}$ that, given subkey K_b consisting of a (small) subset of b key bits, can split the set of m traces and ciphertext values in 2 disjoint subsets. The definition of D depends on the encryption algorithm and it is the critical step in a successful DPA attack.

The fundamental premise for the applicability of DPA is that the power profile for an encryption algorithm depends in some parts on the value of the secret key. Infact, as already discussed in Section 1.4, all known DPA countermeasures attempt to falsify this premise. An intuitive approach is to enforce independence of Ti[j] on the value of the secret key, or in other words, modify the algorithm or the hardware to reduce the sensitivity of the power profile to the secret key value. Unfortunately, it is extremely difficult to guarantee perfect equalization and high-resolution as will be discussed in Chapter 5.

2.2.1 Simple Power Analysis

Simple power analysis (SPA) is a technique whereby information about the operation of a cryptographic token is deduced directly from a power trace. Depending on how a cipher is implemented, this information may reveal key material.

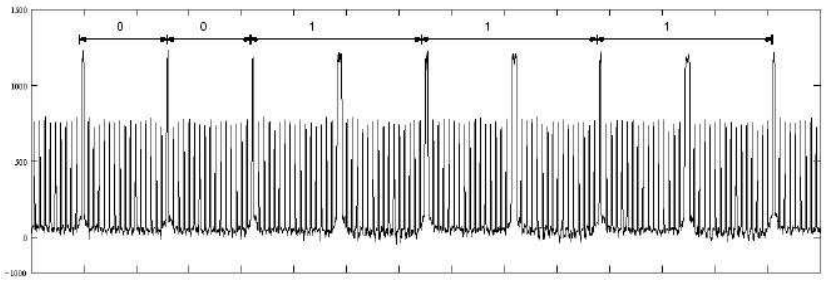


Figure 2.3: An SPA trace of an RSA signature operation.

In order to show an example of how to use SPA, a short presentation of *RSA* algorithms is necessary. The RSA cryptosystem is a public-key algorithm that offers both encryption and digital signatures (authentication).

Ronald Rivest, Adi Shamir, and Leonard Adleman developed the RSA in 1977 [82]; RSA stands for the first letter in each of its inventors' last names. Generally speaking, the RSA algorithm take two large primes, p and q, and computes their product $n = p \cdot q$; n is called the modulus. Choose a number, e, less than n and relatively prime to $(p-1)(q-1)$, which means e and $(p-1)(q-1)$ have no common factors except 1. Find another number d such that $(ed-1)$ is divisible by $(p-1)(q-1)$. The values e and d are called the public and private exponents, respectively. The public key is the pair (n, e); the private key is (n, d). The factors p and q may be destroyed or kept with the private key. It is currently very difficult to obtain the private key d from the public key (n, e). However if one could factor n into p and q, then one could obtain the private key d. Thus the security of the RSA system is based on the assumption that factoring is difficult.

Figure 2.3 shows a portion of a trace from a smart card calculating an RSA signature. Each of the nine spikes indicates the beginning of a square or multiply operation. Initially, registers are loaded with values to be squared or multiplied. Multiplications require additional register loads which increases the width of the leading spike. As a result, square operations (narrow spike) can be distinguished from square-and-multiply operations (narrow spike followed by a wider spike). Thus, five key bits can be determined from the trace: 00111.

Interpreting SPA characteristics is more easily done with some details about the target implementation. With complete details (e.g., source code), an attacker can focus on particular regions of a power trace to try to distinguish the characteristics of specific operations. Generally, any implementation where the path of execution is determined by key bits has a potential vulnerability to this attack.

Kocher et al. [53] were the first to discuss the application of power analysis to implementations of cryptographic algorithms. They were also the first to point out that the dependency of the power traces on the executed instructions can lead to a serious security problem.

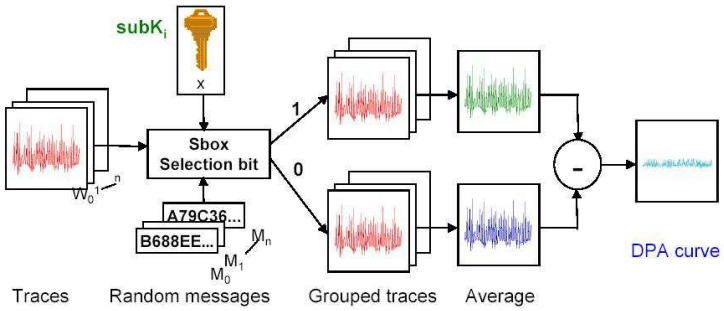


Figure 2.4: DPA basic operations.

2.2.2 Differential Power Analysis

Differential power analysis (DPA) is probably the most threatening attack to result from Kocher's research. The goal of DPA attacks is to reveal secret keys of cryptographic devices based on a large number of power traces that have been recorded while the devices encrypt or decrypt different data blocks.

To carry out a DPA attack, an attacker must have a number of power traces collected from a token as it repeatedly executes a cryptographic operation. The attack proceeds by deducing bits of the secret key, used in each operation, from the observed power consumption. An attacker must also have knowledge of either the inputs or outputs processed by the device during each operation. Usually, an encryption token will use the same key over multiple operations and any generated ciphertext can be freely obtained by an eavesdropper.

The main advantage of DPA attacks compared to SPA attacks is that no detailed knowledge about the cryptographic device is necessary. In fact, it is usually sufficient to know the cryptographic algorithm that is executed by the device. Another important difference between the two kinds of attacks is that the recorded traces are analyzed in a different way. In SPA attacks it is directly analyzed the power consumption trace along the time axis to extract informations about what the cryptographic device is doing. In case of DPA attacks, the shape of the traces along the time axis is not so important. DPA attacks analyze how the power consumption at fixed moments of time depends on the processed data. Hence, DPA attacks focus exclusively on the data dependency of the power traces.

DPA attacks exploit the data dependency of the power consumption of cryptographic devices. They use a large number of power traces to analyze the power consumption at a fixed moment of time as a function of the processed data. The basic DPA technique is shown in Figure 2.4.

Suppose an adversary is able to partition power traces from several cryptographic operations into two groups according to the intermediate value of some bit, b, calculated during each operation. This bit is manipulated during each operation and its value may affect the observed power consumption. If this is the case then the two groups of traces should show respectively different power

biases at locations when b is manipulated. Averaging the traces in each group helps reducing any noise that may be obscuring these usually small biases. Plotting the difference of the two average traces reveals any locations in the traces where these biases occur.

More in detail, the flow to execute a DPA attack consist of five steps.

The first step of a DPA attack is to choose an intermediate result of the cryptographic algorithm that is executed by the attacked device. This intermediate result needs to be a function $D = f(d, k)$, where d is a known non-constant data value and k is a small part of the key.

The second step consists on running the encryption algorithm for N random values of plain-text input. For each of the N plain-text inputs, PTI_i, a discrete time power signal, $S_i[j]$, is collected and the corresponding cipher-text output, CTO_i, may also be collected. The power signal $S_i[j]$ is a sampled version of the power consumed during the portion of the algorithm that is being attacked. The i index corresponds to the PTI_i that produced the signal and the j index corresponds to the time of the sample.

The next step of the attack is to calculate a hypothetical intermediate value D for every possible choice of k. For each possible key, the power traces $S_i[j]$ are divided into two sets using the intermediate value as in 2.4:

$$\begin{aligned} S_0 &= \{S_i[j] \mid D = 0\} \\ S_1 &= \{S_i[j] \mid D = 1\} \end{aligned} \tag{2.4}$$

The next step is to compute for each possible key the average power signal for each set:

$$\begin{aligned} A_0[j] &= \frac{1}{|S_0|} \sum_{S_i[j] \in S_0} S_i[j] \\ A_1[j] &= \frac{1}{|S_1|} \sum_{S_i[j] \in S_1} S_i[j] \end{aligned} \tag{2.5}$$

where $\mid S_0 \mid + \mid S_1 \mid = N$.

Finally, by subtracting the two averages, a discrete time DPA bias, $T[j]$, is obtained for each key:

$$T[j] = A_o[j] - A_1[j] \tag{2.6}$$

When the hypothized key is correct this means that the hypothesis on D matches with the real value assumed by the cryptographic core and thus two sets in (2.4) are split in a right way. Therefore A_0 and A_1 in (2.5) have very different values and thus their difference in (2.6) will be greater than zero. Conversely when a wrong hypothesis on the key is taken in consideration, current traces are splitted in two sets statistically uncorrelated with the value of D. Hence in (2.5) both A_0 and A_1 are the mean of current traces statistically uncorrelated with each other and thus $A_0 \approx A_1$. This means that the difference in (2.6) in this case is approximately zero.

It is evident that the most critical step of a DPA attack is the selection of the function D. Infact the number of power consumption traces that an attacker

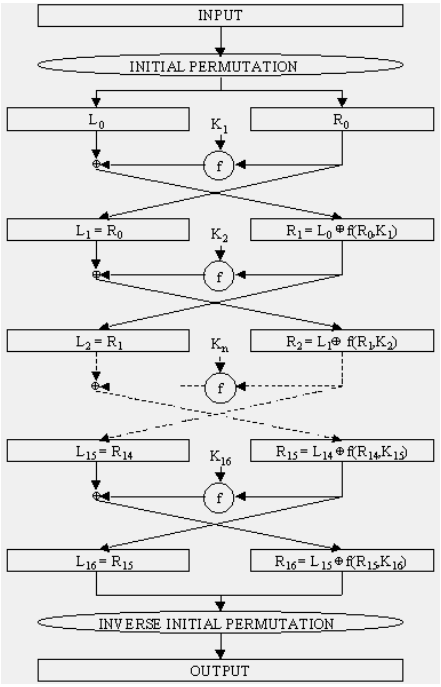


Figure 2.5: Enciphering computation of the DES algorithm.

need for a successful attack will change depending on selected D: more D is related to power consumption and less power traces will be used to find this relation and therefore the unknown part of D, the key.

In order to show an example of how to choose the function D for a DPA attack, a short presentation of *DES* algorithms is necessary.

The Data Encryption Standard (DES) [37] was invented in 1975 by IBM. The algorithm is designed to encipher and decipher blocks of data consisting of 64 bits under control of a 56 bits key (48 bits key and 8 bits for parity check). Deciphering must be accomplished by using the same key as for enciphering, but with the schedule of addressing the key bits altered so that the deciphering process is the reverse of the enciphering process. A block to be enciphered is subjected to an initial permutation IP, then to a complex key-dependent computation and finally to a permutation which is the inverse of the initial permutation (IP-1). The complex key-dependent computation is a Feistel-Structure and consists of 16 rounds (see Figure 2.5):

$$\begin{aligned} L_i &= R_{i-1} \\ R_i &= L_{i-1} \oplus f\left(R_{i-1}, K_i\right) \end{aligned} \tag{2.7}$$

where K_i denotes the subkey of the i-th round.

The cipher function f is defined as $f\left(R_{i-1}, K_i\right) = P\left(S\left(E\left(R_{i-1}, K_i\right)\right)\right)$ (see Figure 2.6) in which P stands for a permutation, S stands for an S-box crossing

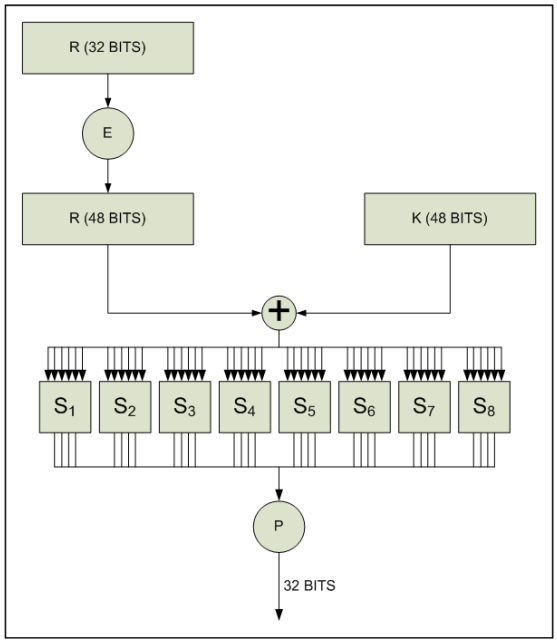


Figure 2.6: Calculation of $f(R_{i-1}, K_i)$ in the DES algorithm.

and, finally, $E(X)$ is the operation to enlarge the X length. The 16 subkeys K_i are derived from a manipulation on the key; L_0 and R_0 are derived from a manipulation on the plaintext.

After the previous short introduction, a practical example of how DPA can be used to attack a DES implementation is described. Selecting an appropriate D function will result in a DPA bias signal that an attacker can use to verify guesses of the secret key. An example of such a D function is as follows:

$$D\left(C_1, C_6, K_{16}\right) = C_1 \oplus SBOX1\left(C_6 \oplus K_{16}\right) \tag{2.8}$$

where C_1 is the first bit of CTO_i that is XORed with one bit of S-box #1; C_6 is the six bits of CTO_i that are XORed with last round's subkey K_{16}; K_{16} represents the six bits of the last round's subkey that feed into S-box #1; $SBOX1(x)$ is a function returning the first bit resulting from looking up x in S-box #1. This particular D function is chosen because at some point during a DES implementation, the software needs to compute the value of this bit. When this occurs or anytime data containing this bit is manipulated, there will be a slight difference in the amount of power dissipated depending on whether this bit is a zero or a one.

From all above-mentioned considerations, starting from one input to the D function was K_{16}, six bits of the subkey, even if the attacker does not know these bits, he can use brute force and try all 2^6 possible values. For each guess, the attacker constructs a new partition for the power signatures and gets a new

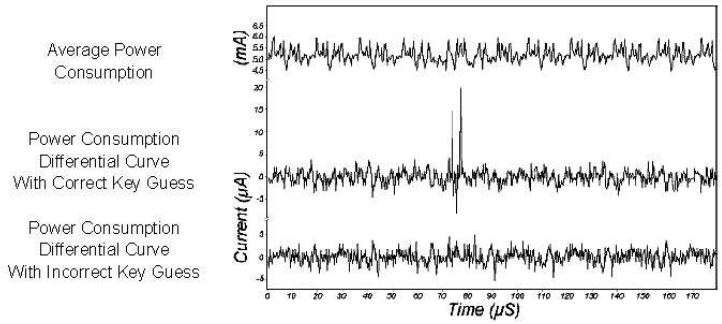


Figure 2.7: DPA traces examples.

bias signal, $T[j]$. If the proper D function was chosen, the bias signal will show spikes whenever the D bit was manipulated. If the D function was not chosen correctly (i.e., the wrong subkey bits were guessed), then the resulting $T[j]$ will not show any biases. Some DPA trace examples that is possible to obtain are showed in Figure 2.7.

Using this approach, an attacker can determine the six subkey inputs to S-box #1 at round 16 of DES. Repeating this approach for the seven other DES S-boxes allows the attacker to learn the entire round 16 subkeys (48 bits). The remaining 8 bits can be found by brute force or by successively applying this approach backwards to previous rounds.

Kocher et al. tested this attack and claimed it was successful on all smart cards they examined.

2.2.3 An overview on Correlation Power Analysis

DPA was later extended to correlation power analysis (*CPA*) in which a correlation coefficient is used to estimate the linear relationship between a power consumption model and the real power traces for each hypothized key.

The power consumption model is a model of the relationship between the power consumption and the input vector. For instance is commonly used Hamming weight of the data being manipulated.

The correlation coefficient measures the linear relationship between two variables.

In statistics, we can express the linear relationship between two points of a trace based on the covariance or the correlation. The covariance quantifies the deviation from the mean and a definition is given in (2.9) as the average of the product of the deviation for the random variables X and Y. The covariance is a linear measure because it is based on the average deviation. The equivalent formula (2.10) shows that the covariance is also related to the concept of statistical dependence. If X and Y are statistically independent, then $E(XY) = E(X)E(Y)$, and therefore $Cov(X, Y) = 0$. On the converse, if $Cov(X, Y) = 0$, then X and Y are independent.

$$Cov\left(X,Y\right) = E\left(\left(X - E\left(X\right)\right) \cdot \left(Y - E\left(Y\right)\right)\right) \tag{2.9}$$

$$Cov\left(X,Y\right) = E\left(XY\right) - E\left(X\right) \cdot E\left(Y\right) \tag{2.10}$$

A related, but more commonly used method to measure a linear relationship between two values is the correlation coefficient $\rho\left(X,Y\right)$. The correlation coefficient is defined in terms of the covariance as can be seen in (2.11). It is a dimensionless quantity and it can only take values between plus and minus one.

$$\rho\left(X,Y\right) = \frac{Cov(X,Y)}{\sqrt{Var(X) \cdot Var(Y)}} \tag{2.11}$$

As in CPA attack term X or term Y is the key and it is normally unknown, this means that in a CPA attack the correlation coefficient is typically unknown and needs to be estimated. The estimator r is defined in (2.12) and this is what is commonly called correlation coefficient in a CPA attack.

$$r = \frac{\sum_{i=1}^{n}\left(x_i - \overline{x}\right) \cdot \left(y_i - \overline{y}\right)}{\sqrt{\sum_{i=1}^{n}\left(x_i - \overline{x}\right)^2 \cdot \sum_{i=1}^{n}\left(y_i - \overline{y}\right)^2}} \tag{2.12}$$

More details about CPA can be found in [61]. In this work, and in special way in Chapter 4, a new attack is presented whose procedure is similar to the CPA.

Chapter 3

The Supply and Current Measurement

3.1 Introduction

When an algorithm or protocol is executed in a cryptographic device (e.g. a chip-card), secret data are manipulated (e.g. cryptographic keys and user PINs). A main requirement for the device is to physically protect these sensitive data against observation thus protecting itself against cloning. The traditional research in cryptanalysis has been mainly focused in investigating weaknesses in algorithms and protocols and, since only few years, the implementation is considered a part of the security evaluation of cryptographic systems.

As discussed in the previous chapter, since differential power analysis (DPA) has been introduced by P. Kocher et al. in 1998 [53], many research activities have been directed in developing countermeasures to enhance the device resistance against DPA while, on the contrary, few contributions aimed to enhance the quality of the measurements needed for the attack itself have been reported in the technical literature [31, 58].

Starting point for any power analysis attack is the measurement of the instantaneous current consumption waveforms from the device under attack. The quality of the available measurements strongly influences the execution time (i.e. the number of waveforms needed for the attack) and, at the end, determines the attack success. The state of the art consists of using a small series resistor between the V_{SS} or V_{DD} pin on the card and the measurement ground. The time varying voltage drop on the resistor $V(t) = RI_{DD/SS}(t)$ can be measured using a differential probe and sampled by a digital oscilloscope [7]. Even if in principle this setup is rather simple to implement, in practice, obtaining good quality measures is not straightforward.

There are some ideas, known in literature, to improve the effectiveness of a power attack with respect the standard resistor-based setup:

- by reducing the possible sources of noise by improving the measuring

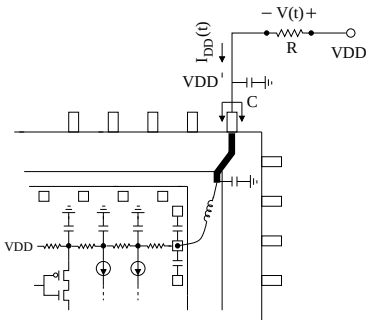


Figure 3.1: Model of the power supply interconnection.

setup;

- by pre-processing the acquired current traces, for example by a filter;
- by using the so called Higher-order attacks [56];
- by considering optimal strategies such as template attacks [34] or stochastic models [89];
- by combining the power-based attack with another side channel leakage source (e.g. electromagnetic radiations).

In this chapter the first solution has been exploited showing that the resistor-based setup is not really the optimal solution to measure the current consumption of an IC and a more effective technique based on an active circuit is introduced. Moreover further experimental results using the proposed circuits are reported and its effectiveness in a practical power analysis attack is assessed.

More in detail the first activity in this work was to confirm experimentally the advantages found in simulation, comparing current measurements using the standard setup with the same measurements using the proposed solution. Secondly, a simple real attack has been performed using both the classical setup and the new one. Finally, a more complex DPA attack on a non-secure software DES has been performed to better compare two setups.

3.2 The proposed idea

A lumped component model for the interconnection between the chip internal power supply network and the external measurement resistor R, used in a classical setup to measure the power consumption, is depicted in Figure 3.1. If the current through V_{SS} is measured, a similar model holds.

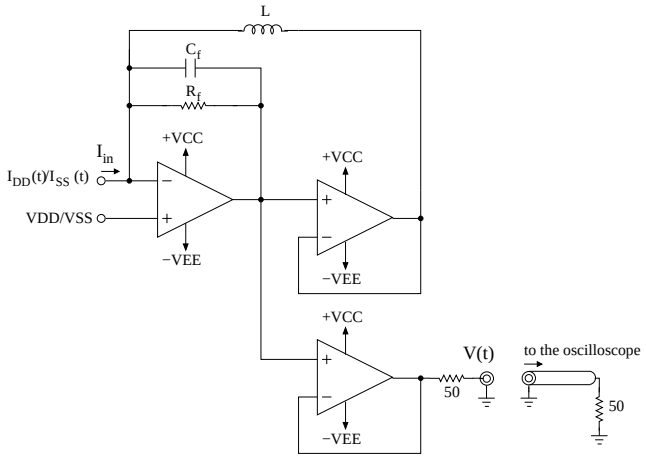


Figure 3.2: Supply and current measuring circuit (SCM).

From the model above results the measuring resistor R and the distributed parasitic capacitance on the chip-internal power connection cause a time constant that limits the bandwidth of the useful signal $I_{DD/SS}(t)$. Therefore, especially for large IC's, low values must be used for R. Typical values are few Ohms. On the other hand, R is also the amplification of the current signal and low values for R imply a low sensitivity of the measurement. As a consequence, a trade-off between bandwidth and sensitivity is required where the best choice depends on the particular device under attack. Moreover, the probing resistor R causes voltage bounces that affect the circuit behavior whose actual voltage supply becomes $V'_{DD} = V_{DD} - I(t)R$ (insertion error).

To overcome the highlighted limitations, an active circuit has been proposed (Figure 3.2) which features a low impedance current measuring input and a high transimpedance gain, while providing, at the same time, the device under measure with a stable voltage supply. In the followings, the circuit is called supplying and current measuring circuit, or simply SCM.

As shown in Figure 3.2, SCM is based on a transimpedance amplifier used as input stage to read the current consumption signal $I_{DD}(t)$ or $I_{SS}(t)$ from a device under analysis. The transimpedance reference input is connected to V_{DD} or V_{SS} and, of course, a balanced power supply $+V_{CC}/-V_{EE}$ is necessary. A second low frequency voltage feedback loop (closed after a voltage buffer), provides the device connected at the probing input with a stable V_{DD}/V_{SS}.

The output signal is taken after a second de-coupling buffer and, within the circuit bandwidth, it holds:

$$V_{out}(t) = -\frac{R_f}{2} I_{in}(t) \tag{3.1}$$

where the 50Ω/50Ω output partition has been taken into account.

From a simplified AC model, the following expression for the circuit transfer function has been obtained:

$$\frac{V_{out}}{I_{in}}(s) = -\frac{1}{2}\frac{A_v L R_f s}{(1+A_v)R_f + (1+A_v)Ls + [C_f L R_f (1+A_v) + C_{in} L R_f]s^2} \tag{3.2}$$

where A_v is the voltage gain of the transimpedance amplifier op-amp, C_{in} is the circuit input capacitance, C_f is a compensation capacitance and the output buffers are supposed ideal. For $A_v \gg 1$, (3.2) becomes:

$$\frac{V_{out}}{I_{in}}(s) \simeq -\frac{1}{2}\frac{L R_f s}{R_f + Ls + C_f L R_f s^2} \tag{3.3}$$

and for the first pole it holds $\omega_p \approx R_f/L$ which can be used to choose L for a given low cut-off frequency.

A discrete component implementation of the proposed circuit has been designed where high slew rate and low noise have been adopted as main criteria for the selection of the components. The op-amps $AD8009$ have been used both for the transimpedance and the two voltage buffers. A 500Ω value has been chosen for the transimpedance resistor R_f thus obtaining a gain of about $250V/A$. A $150\mu H$ inductance L fixes the circuit low cut-off frequency at about 1MHz.

S-parameters measured with the network analyzer are depicted in Figure 3.3 and the corresponding transimpedance gain and input impedance show a flat frequency response and an input resistance below 150Ω up to about 300MHz (Figure 3.4).

A noise analysis of the SCM has been also performed which resulted, within the circuit bandwidth, in the following expressions for the equivalent input noise voltage/current generators:

$$v_{n_i} = \frac{R_f}{2}\sqrt{\left(i_n^-\right)^2 + \left(\frac{\sqrt{4kTR_f} + e_n}{R_f}\right)^2} \tag{3.4}$$

$$i_{n_i} = \sqrt{\left(i_n^-\right)^2 + \left(\frac{\sqrt{4kTR_f} + e_n}{R_f}\right)^2} \tag{3.5}$$

where e_n is the op-amp equivalent input noise voltage generator and i_n^- is the transimpedance input noise current generator (at the non-inverting input). For the minimum detectable signal (MDS), it holds:

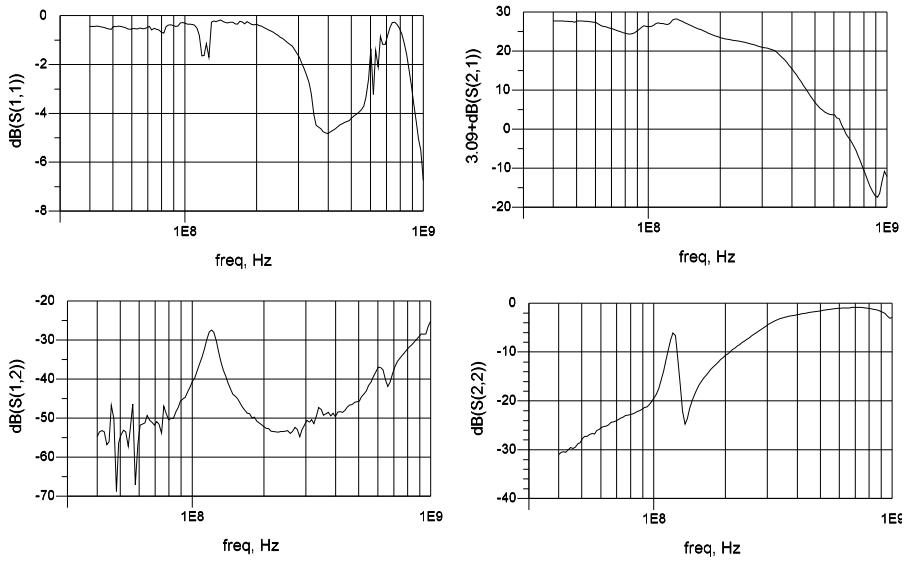


Figure 3.3: S-parameters of the supply and current measuring circuit.

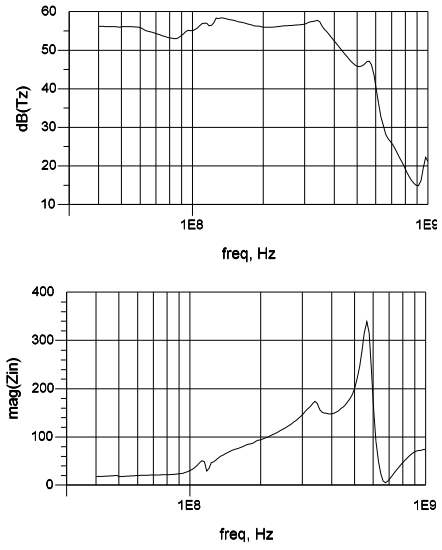


Figure 3.4: Measured transimpedance gain (above) and input impedance frequency response (below).

$$MDS = \sqrt{\left[i_{n-}^2 + \left(\frac{\sqrt{4kTR_f} + e_n}{R_f} \right)^2 \right] B} \tag{3.6}$$

where B is the circuit bandwidth. From the $AD8009$ databook data, it follows $MDS \cong 20 \frac{pA}{\sqrt{Hz}} \sqrt{B}$.

3.3 Focus on the SCM advantages with respect to a resistor

Using the SCM to monitor the current consumption of a device under attack provides several advantages with respect to a standard setup based on a probing resistor.

Firstly, it is possible to reach higher gain-bandwidth products with a higher flexibility: a 50Ω matching does not imply the transimpedance gain.

Moreover the insertion error in the measurement introduced by a resistor is reduced using the SCM due to the smaller transimpedance amplifier input resistance.

Actually, using the classical setup, during the current consumption measurement, the voltage drop over the resistor decreases the voltage supply of the device under measurement. This effect limits the maximum value that is possible to use for the resistor.

It follows that the SCM can in principle reveal smaller current peaks with respect to a resistor-based setup. It means the noise input current level is smaller than in the resistor case. Moreover the presence of parasitic effect introduced by the resistor (RC effect) is limited using the SCM. Furthermore, the frequency components of the current signal below $1MHz$ are filtered: the static consumption of the device under measure is filtered amplifying only the high frequency components which are data dependent. On the other side, the peaking phenomena at high frequency are optimized using the SCM obtaining a smaller trans-characteristic overshoot.

3.4 Measurement setup

The measurement setup used for the experiments is the same in both cases. The SCM has been mounted on a board as depicted in Figure 3.5 together with a MAX3000A FPGA from Altera used as device under attack (DUA). In this prototype the SCM has been connected to the V_{SS} pins of the FPGA.

The measurement setup includes a Tektronix TDS 754d digital oscilloscope, an Agilent 33250A clock generator and a Hewlett-Packard E3631A triple-output power supply to provide $3.3V$ for the FPGA pads and $+5/-5V$ for the SCM itself (Figure 3.6).

As shown in Figure 3.5, the board is divided in two parts: in the lower part the SCM is visible and the FPGA is mounted in the upper part. A 500Ω value has been chosen for the transimpedance resistor R thus obtaining a gain of about $250V/A$. In the right side there is a BNC connector to take the current measurement. Three pins allow to connect the FPGA ground to the resistor (standard resistor-based attack) or to the SCM (proposed attack).

A portion of the Serpent algorithm [3] has been adopted as a case study to prove the effectiveness of the SCM.

Since the Data Encryption Standard algorithm ([37]) is nearing the end of its useful life, the US National Institute of Standards and Technology has issued a call for a successor of the DES, to be called the Advanced Encryption Standard or AES. The *Serpent algorithm* was a candidate for AES.

Serpent encrypts a 128-bit plaintext to a 128-bit ciphertext in 32 rounds under the control of 33 128-bit subkeys. The user key length is variable, 128, 192 or 256 bits; short keys with less than 256 bits are mapped to full-length keys of 256 bits by appending one "1" bit to the MSB end, followed by as many "0" bits as required to make up 256 bits. The cipher itself consists of: an initial permutation IP; 32 rounds, each consisting of a (4-bit) key mixing operation, a pass through S-boxes, and (in all but the last round) a linear transformation (in the last round, this linear transformation is replaced by an additional key mixing operation); a final permutation FP. A set of eight S-boxes is used four times. Each round function uses only a single replicated S-box. Thus after using S7 in round 7, it uses S0 again in round 8, then S1 in round 9, and so on. A Serpent S-box maps four input bits to four output bits. Because of the reduced dimensions of the FPGA on the board, this algorithm is perfect for our intent.

It must be clear that the first measurement target is to demonstrate an improvement in power attack using the SCM circuit with respect to a standard setup (resistor). For this reason it has been chosen to consider only on central round of Serpent algorithm, with the result of the xor function, between a prefixed key and the data, as input signal of a Serpent S-box (the seventh).

As shown in Figure 3.7, the tested circuit includes a 4-bit S-Box with its output register, an input XOR with a 4-bit key and a state machine which generates (in 256 clock cycles) all possible transitions for the 4-bit data xor'ed with the key.

To stimulate the FPGA with all possible input transitions to the S-box, the design into the FPGA includes a state machine, in addition to the S-box, that generates all possible transitions. Using two counters, the state machine increases one counter every clock edge; when this counter reaches the terminal count, the other counter increments. Assigning address signal to one counter at a time, the addresses performs all possible transitions. In this way, in fact, the sequence of address signal values (in decimal) is 0, 0, 0, 1, 0, 2, 0, 3, 0, 4, ..., 0, 15, 1, 0, 1, 1, 1, 2, 1, 3, ..., in which the values in odd positions are the outputs of a counter while the values in even positions are the outputs of the other counter. The state machine provides a trigger signal for the digital oscilloscope too.

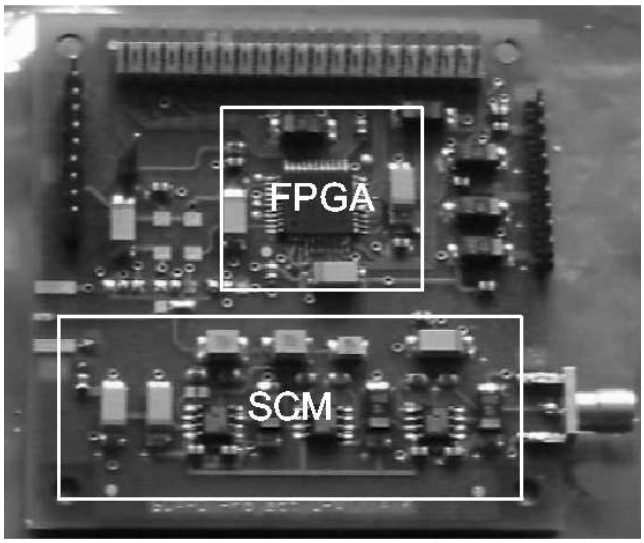


Figure 3.5: Prototype board with the SCM and an Altera MAX3000A FPGA.

The output data is available externally for test purposes but, during the current consumption measurement, the pads are tri-stated to avoid noise inside the current profile due to current spikes on pads, since FPGA core ground and pad ground are connected together in the board.

An advantage of this design is the possibility to study the differences occurring in the current consumption profile changing an external key bit. The 4-bit key is generated externally and, during the measurement, is static ($key = 0001$). The described circuit has been synthesized from a VHDL description and has been mapped on the available FPGA. A 5MHz clock frequency and a 1GS/s sampling rate have been used for the measurements.

3.5 Experimental results

A screenshot from the oscilloscope is shown in Figure 3.8 where the SCM output $V(t)$ is visible. Peak amplitudes ranging from $0.5V$ to about $2V$ can be observed as the processed input data change during the 256 acquired clock cycles (only the first 24 periods are depicted in Figure 3.8).

Analyzing the $V(t)$ waveform, a 14.8MHz oscillation has been observed coming from the FPGA. It is clearly visible when the clock is disabled and it can be also observed in Figure 3.8, superimposed to the current consumption of our case study circuit. Unfortunately, the available FPGA does not provide the possibility to disable this on-chip oscillator.

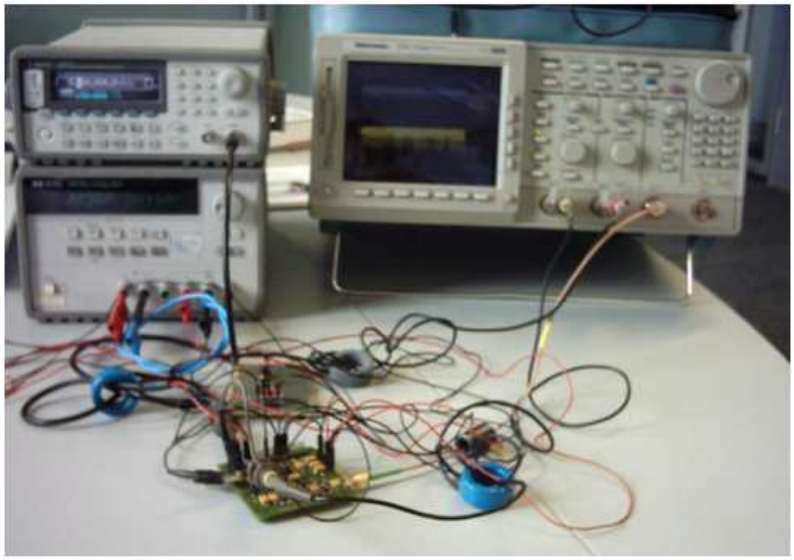


Figure 3.6: Measurement setup.

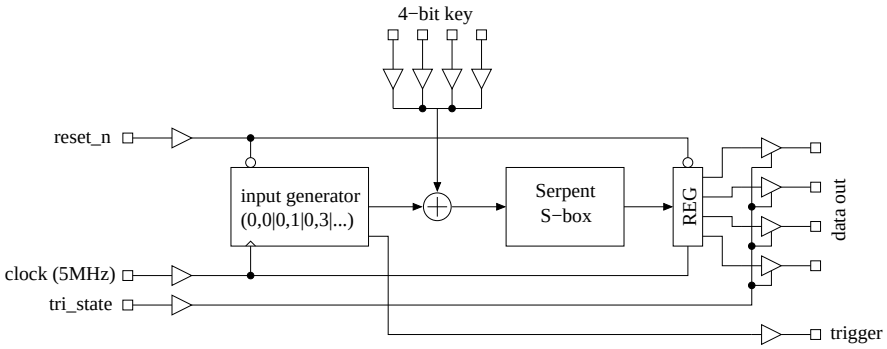


Figure 3.7: Circuit used as a case study.

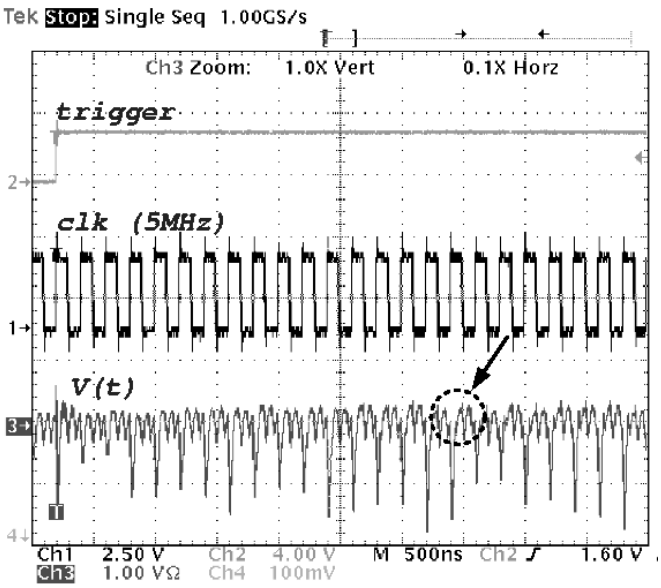


Figure 3.8: Oscilloscope screenshot.

To evaluate the improvement of the introduced circuit with respect to a resistor-based setup, the same measurement has been repeated using a 50Ω resistor and a differential probe to read out the voltage drop. A comparison between the two waveforms when the FPGA is executing the same operations is shown in Figure 3.9 (first 24 clock cycles). When the resistor is used, $V(t)$ peaks show a maximum amplitude of about $50mV$, that means 40 times smaller than the corresponding peaks generated by the SCM. On the other hand, larger resistors cannot be used since, with 50Ω, the available bandwidth is already quite small compared to the SCM, as follows comparing the peak durations in Figure 3.9.

After the data acquisition, the next step was to define the right figure of merit in order to assess the obtained experimental results and to compare them in the two cases. The idea is to compare the SNR (Signal to Noise Ratio) obtained using the SCM circuit and the resistor-base setup respectively, defining a quantity S which is the useful signal and a quantity N which represents the superimposed noise in the set of collected traces.

In order to quantify the achievable improvement in terms of sensitivity to the current consumption variations, the same measurement has been repeated 20 times both with the SCM and the resistor. Therefore the current profile are integrated over the input signal period to extract the energy.

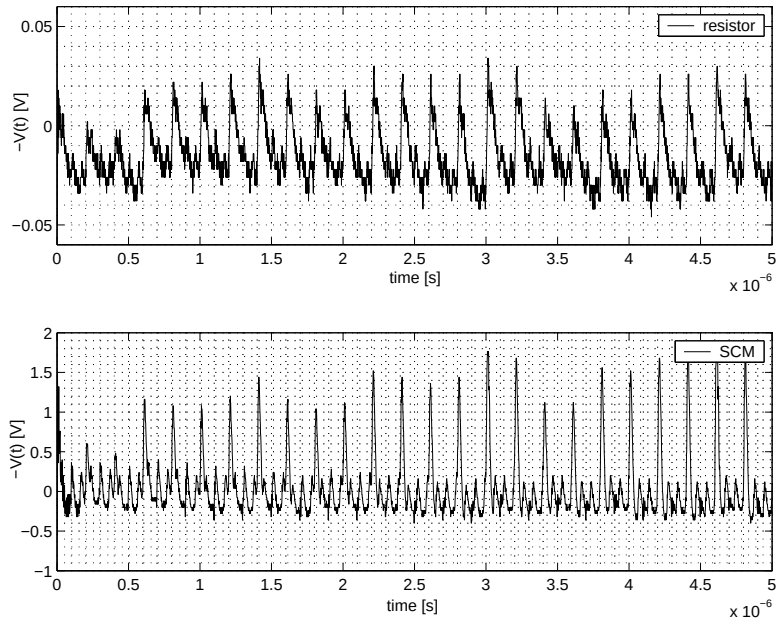


Figure 3.9: Waveform using a 50Ω resistor (above) vs. SCM (below).

In Figure 3.9 a current profile distortion is possible to note. This distortion is due to the oscillation generated by a free running oscillator inside the FPGA. Since the FPGA model used in the test board does not allow to disable this oscillator, this problem has been resolved after the wave acquisition. More in detail, the adopted solution is to integrate not respect the input signal period but respect to the superimposed oscillation period, or a multiple of this period, to cut off this component. Since the exact value of the oscillation period is unknown, an SNR value for different integration windows w has been calculated.

The measurement noise has been evaluated as the average, over the 256 clock cycles, of the energy/cycle variance calculated over the 20 repeated acquisitions:

$$N = \frac{1}{256} \sum_{j=1}^{256} \sigma_{E_j}^2 \tag{3.7}$$

where,

$$E_j = \sum_{i=1}^{w} V[i \cdot T_{clk}] \tag{3.8}$$

with $w \leq 200$, being 200 the number of samples in each clock period.

The useful signal is the variance, calculated over the 256 cycles, of the mean energy/cycle:

$$S = \sigma^2_{\overline{E}} \tag{3.9}$$

where,

$$\overline{E} = \frac{1}{20}\sum_{j=1}^{20} E_j. \tag{3.10}$$

Finally, the signal to noise ratio:

$$SNR = 10log_{10}\frac{S}{N} \tag{3.11}$$

has been adopted as figure of merit.

The obtained results are plotted in Figure 3.10 as a function of the integration window length w for the resistor and the SCM respectively. In both cases, the SNR shows a maximum for $w = 135$. Actually, sampling at 1GS/s, a 135-sample window includes an integer number of periods of the 14.8MHz on-chip oscillator, thus filtering out that asynchronous disturbance. Comparing the maximum values, it follows that a $20dB$ improvement in the SNR can be obtained using the proposed technique.

3.5.1 Attacking a simple crypto-core

The current traces collected for the evaluation of the SNR have been used to implement a differential power analysis too. In particular, the first 16 clock periods of each trace have been extracted from the complete curve since they correspond to the transitions $(0, i)$, $\forall i = 0, \ldots, 15$, for the input 4-bit data.

A Matlab program has been written to implement a DPA starting from a definition of a *S7_serpent(d, k)* function that receives as input a data d and a key k, makes a bit-xor between d and k and calculates the output of the Serpent seventh S-box. After selecting a target-bit (the third bit in this case), for all possible keys (2^4) the current traces are divided in two sets with respect to the target-bit value and for each set the average trace is calculated. Finally, the difference between two average traces is evaluated.

The obtained differential curves for the 16 key hypotheses using the SCM and a 50Ω resistor are reported in Figure 3.11.a and Figure 3.11.b respectively. In the SCM case, the trace corresponding to the correct key hypothesis ($key = 0001$) shows a clearly visible peak which is sensibly higher than the other peaks in the plot (*ghost peaks*). On the contrary, the correct peak is only slightly higher than the ghost peaks in case the resistor-based setup is used. The peaks for each key hypothesis normalized to the peak obtained for the correct key are reported in Table 3.1 for both cases: in the second case, the attack is not able to discriminate with high confidence between the correct key (0001) and keys 0010, 0101.

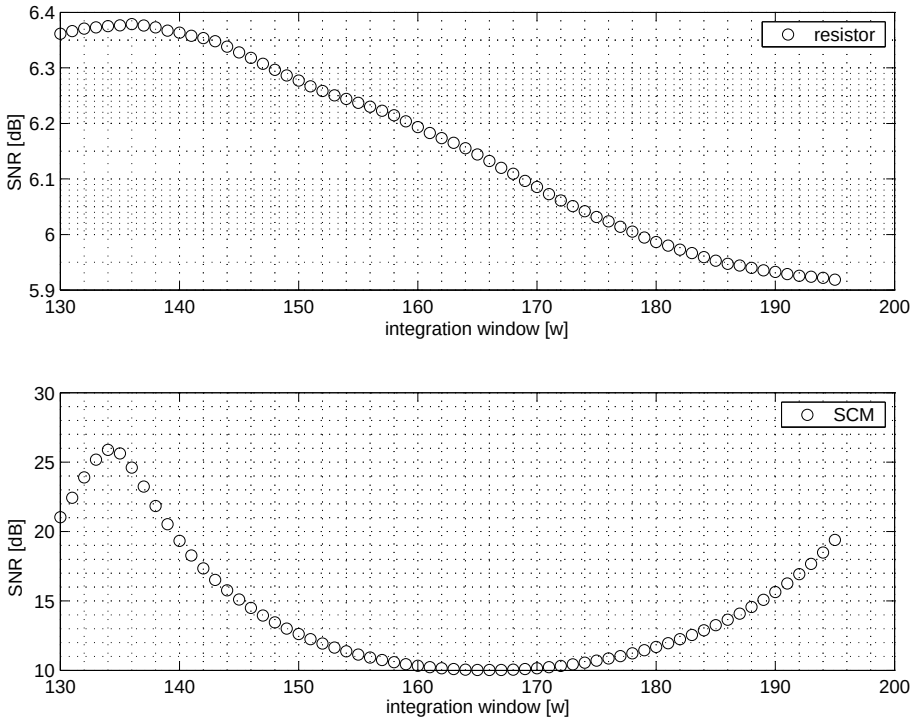
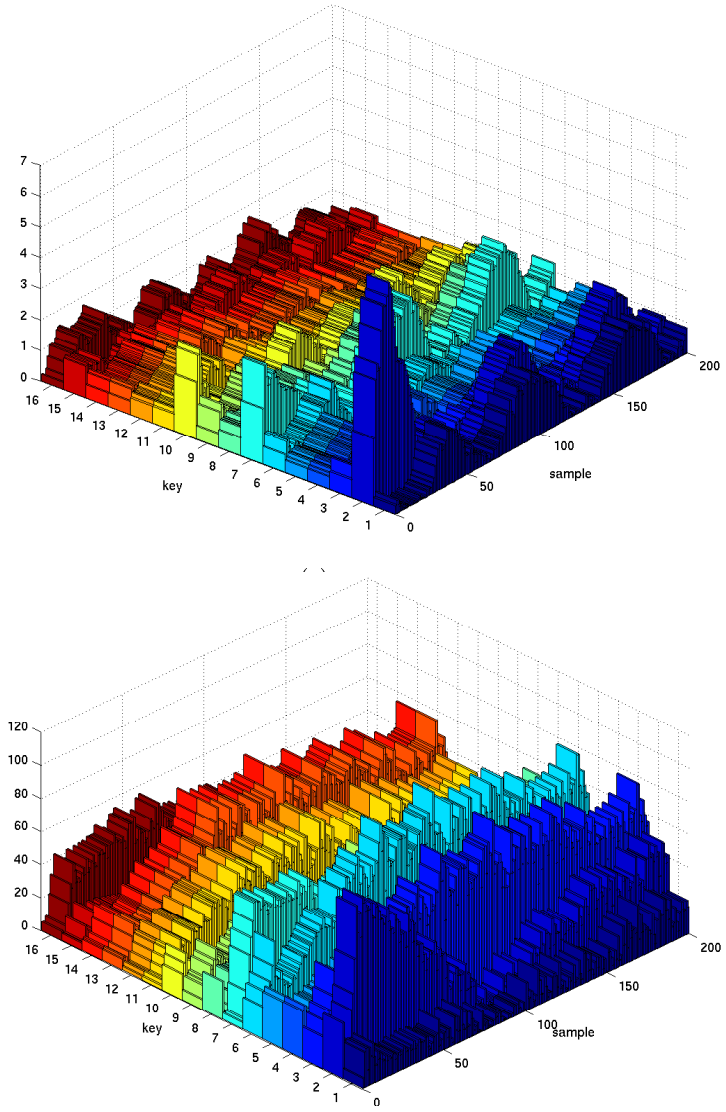


Figure 3.10: SNR comparison - resistor (above) vs. SCM (below).

Table 3.1: Normalized DPA peaks.

key	SCM	50Ω
0000	0.31	0.31
0001	1	1
0010	0.30	0.97
0011	0.19	0.42
0100	0.19	0.42
0101	0.31	0.97
0110	0.45	0.72
0111	0.30	0.57
1000	0.26	0.48
1001	0.41	0.26
1010	0.22	0.54
1011	0.22	0.70
1100	0.22	0.70
1101	0.22	0.70
1110	0.17	0.29
1111	0.25	0.53



(b)

Figure 3.11: Result of the differential power analysis using the SCM (a) and a 50Ω resistor (b).

3.5.2 Attacking a DES

In order to test the proposed current measuring technique in a real scenario, a DPA on a non-secure software DES running on a 8051 like micro-controller has been implemented.

As a preliminary step, a stand-alone version of the SCM which can be connected to the device under analysis was manufactured. A clock frequency of 5MHz has been used for the 8051 and the current traces are sampled at 250MS/s. At the beginning of the encryption, the micro-controller sends a data on the I/O port which is used as trigger for the oscilloscope to start the acquisition of the current trace (250000 samples).

The attack is implemented on the first round of the algorithm using one of the S-boxes outputs as target bit. Different S-boxes and different output bits have been tested obtaining similar results.

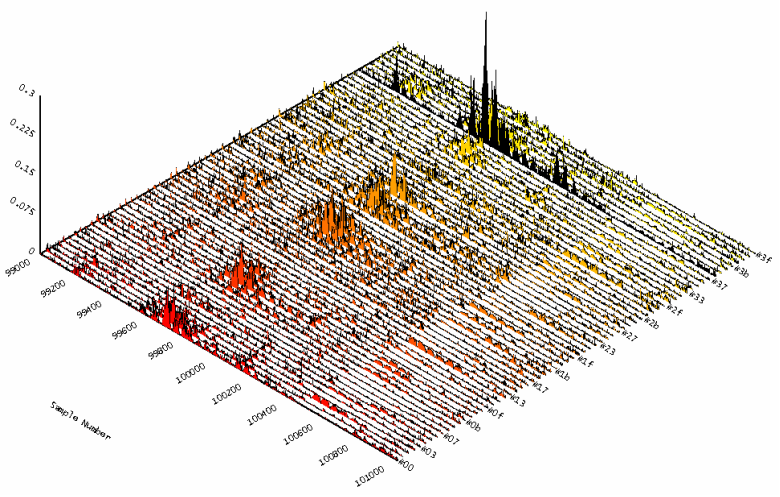
The differential curves for the 64 hypotheses on the 6-bit subkey are shown in Figure 3.12.a and Figure 3.12.b for the SCM and the 50Ω resistor-based setup respectively. In both cases, 512 traces are sufficient to identify the correct key ($0x38$). However, the SCM provides a better correlation coefficient (0.2514) which corresponds to the higher ratio between current key peak and ghost peaks visible in Figure 3.12. This result confirms the analysis on the SNR reported in Section 3.5.

3.6 Conclusions

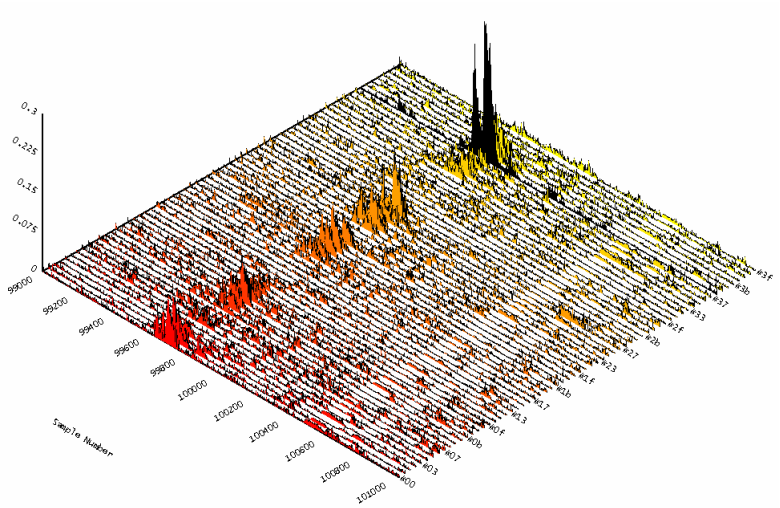
An effective current measuring technique has been introduced which promises to considerably enhance power analysis attacks against cryptographic devices. The proposed idea is based on a transimpedance amplifier to provide a low impedance current input and an additional DC feedback loop to control the voltage at the input pin, thus supplying the device under attack with a stable voltage.

From the analysis and simulation of the SCM, several advantages have been showed with respect to the standard-resistor setup. These advantages were confirmed experimentally.

Compared to a resistor-based measurement, the presented circuit showed a $20dB$ improvement in the sensitivity to the current consumption variations of a device under attacks. Therefore, it is expected a reduction of the required number of acquisitions to implement a successful DPA attack against secure implementations. Attacks performed on a FPGA implementing a section of the Serpent algorithm and on a software DES running on a 8051 micro-controller showed an improvement if the SCM is used.



(a)



(b)

Figure 3.12: Result of the differential power analysis using the SCM (a) and a 50Ω resistor (b).

Next steps in this research activity will be the implementation of DPA attacks against cryptographic co-processors to prove the actual advantage that the introduced technique brings in terms of required number of acquisitions and success of attacks against cores where DPA countermeasures are employed. Moreover, to perform the reported measurements, it has been necessary to pay attention to protect the circuit from external (electromagnetic) noise. This means that a better solution to overcome these problems would be to implement the SCM enclosed in a shielded package thus realizing a sort of current probe. This requires an implementation of the SCM circuit as compact as possible. In fact minimizing the circuit dimensions will allow to keep the interconnections short and therefore to expand the bandwidth minimizing parasitic effects.

Chapter 4

Leakage Power Analysis Attack

4.1 Introduction

As discussed in Chapter 2, Power Analysis attacks have been extensively showed to be a major threat to the security of data that are processed and stored in cryptographic devices, such as Smart Cards. These attacks exploit the dependence of the dynamic power consumption on the inputs of a cryptographic algorithm, i.e. the input ciphertext (plaintext) that is to be decrypted (encrypted) and the secret key. The cost in terms of equipment and computational effort are rather low, hence these attacks can be easily performed. In Chapter 3 an active circuit (the so called SCM) has been proposed which considerably enhances power analysis attacks against cryptographic devices with respect the standard resistor-based setup. The comparison between two setups has been performed by using the standard Differential Power Analysis post-processing technique.

Among the existing post-processing techniques, the Correlation Power Analysis (CPA) is well known to be relatively simple and effective [61, 16]. In CPA, a power model is adopted to estimate the dynamic power consumption required to physically evaluate a signal generated within the crypto-chip as a function of the input and the secret key [16]. Then, a portion of the secret key is guessed, and the resulting dynamic power consumption is estimated with this model. Successively, the correlation coefficient between this estimation and the measured power is evaluated. Finally, the correct key is identified by taking the key guess that leads to the highest value of the correlation coefficient [61, 16]: indeed, the closer to the actual key is the guessed key, the greater is the correlation between the estimated and measured power.

In sub-100nm technologies, the dynamic power is no longer the dominant contribution to the chip power budget, due to the much faster increase of leakage (i.e., static) power at each technology generation ([36, 68, 5, 48]). For example, at the 65nm technology node the leakage power is in the order of half the chip

power consumption, and is planned to be an even greater fraction in successive technologies by the ITRS Roadmap [48]. Hence, the leakage power can be easily measured in the same way as the dynamic power is measured in traditional Power Analysis attacks. Due to the strong leakage dependence on the input of digital circuits [5], leakage can also provide a significant amount of information on the secret key, hence Power Analysis attacks based on leakage can be devised. In the following, these attacks will be referred to as 'Leakage Power Analysis' (LPA) attacks.

Until now, information security issues related to the leakage dependence on processed data were given in [45], and these considerations were applied in [44] to simulate an attack to a simple crypto-core. In [45] and [44], some basic concepts are presented, but LPA attacks are discussed neither in terms of experimental issues, nor from a theoretical point of view. In this thesis, Leakage Power Analysis attacks are formalized and analyzed from both a theoretical and experimental standpoint in a systematic manner. Advantages and practical problems related to the LPA attack are discussed through comparison with traditional attacks targeting the dynamic power. Various examples, simulations and experimental results are reported to better understand LPA attacks, as well as to validate the underlying assumptions. In particular, a practical LPA attack procedure is presented for the first time. A closed-form model of the LPA attack result is also developed to better understand the attack. The impact of technology scaling is also analyzed to evaluate the LPA effectiveness in future technologies.

Analysis shows that LPA attacks are a major threat to information security in sub-100nm technologies. Moreover, since many countermeasures to Power Analysis attacks targeting the dynamic power have been proposed until now [61], LPA becomes the weak point in the information security of cryptographic circuits, if not taken into account during their design.

In this chapter the leakage sources in MOS devices and CMOS logic gates are reviewed, and basic hypotheses are validated with experimental and simulation results. Therefore a well defined procedure to perform LPA attacks is introduced and measurement and practical issues are discussed to validate the proposed methodology. A detailed theoretical analysis about the simple models used in LPA attacks is first treated and then confirmed by experimental results. Real attacks to a register and a cryptographic core are showed. At the end of this Chapter the effect of process variations on LPA attacks are studied and it is also shown that LPA attacks are effective even in the presence of transistor-level countermeasures against DPA. Conclusions and further developments close the Chapter.

4.2 Review of leakage sources in nanometer CMOS logic gates

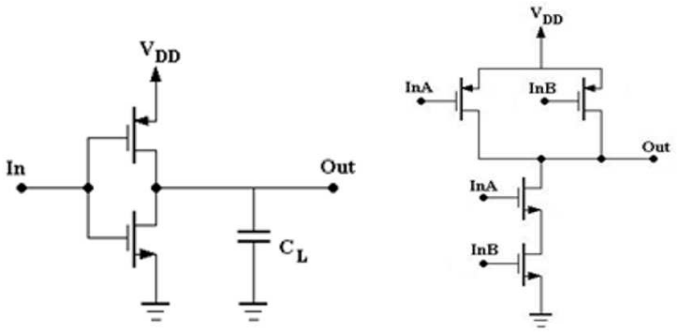
The leakage current conducted by MOS transistors operating in cut-off region consists of three main sources: sub-threshold, gate tunneling and inverse junction current [95]. In current technologies, the sub-threshold current is the most important leakage contribution in a MOS transistor, considering that the gate leakage is reduced with the adoption of high-k materials, and the inverse junction current is two orders of magnitude lower than the former [69]. More specifically, the sub-threshold current $I_{(leak,NMOS)}$ for a single NMOS transistor is given by [95] and [69]

$$I_{leak,NMOS} = I_0 \frac{W}{L} \exp\left(-\frac{V_{TH}}{nk^T/_q}\right) \tag{4.1}$$

where V_{TH} is the transistor threshold voltage, I_0 and n are technology-dependent parameters, W/L is the transistor aspect ratio, T is the temperature, whereas k and q are respectively the Boltzmann constant and the electron charge. An analogous expression holds for the PMOS transistor (with V_{TH} being the threshold voltage magnitude). Due to the exponential dependence in (4.1), the leakage current is very sensitive to temperature variations, as well as to process variations in V_{TH}.

The leakage current of static CMOS logic gates strongly depends on their input [5]. As an example, the leakage of an inverter gate (see Figure 4.1a) is equal to the leakage of the NMOS (PMOS) transistor when this device is in the cut-off region, i.e. if the input is low (high). Since the threshold voltage of the NMOS and PMOS transistors are significantly different in real CMOS technologies, from (4.1) the inverter leakage depends on the input value. This significant leakage dependence on the input is confirmed by data in Tables 4.1-4.2, which respectively report the simulated leakage of a minimum-sized inverter gate in a 90nm and 65nm technology. From Table 4.1, the inverter leakage with a low input is greater than that with a high input by a factor of 3-5 (this is because the NMOS transistor has a lower V_{TH}, compared to the PMOS). From Table 4.2, similar results are obtained in the 65nm technology for realistic operating temperatures of cryptographic device (i.e., at room temperature or slightly greater). The strong dependence on the temperature is also confirmed by Table 4.1 for the 90nm technology, whereas this dependence is weaker for the 65nm technology in Table 4.2, due to the different temperature dependence of the gate leakage.

The above considerations on the leakage dependence on the input value can be extended to general static CMOS gates, whose pull-up and pull down networks are made up of series- and parallel-connected transistors. To understand the leakage dependence on the input, it is sufficient to analyze the case of n series-connected transistors, which are always present in either the pull-up or the pull-down network of static logic gates. For simplicity, let us consider the case of the two series-connected NMOS transistors in the NAND2 gate in Figure



(a) (b)

Figure 4.1: Schematic of the CMOS Inverter (a) and NAND2 gate (b).

Table 4.1: Leakage current (nA) in various CMOS logic gates (90nm technology).

Inverter gate			
A	$T = 0°C$	$T = 25°C$	$T = 50°C$
0	1.36	3.19	6.52
1	0.24	0.73	1.90

NAND2 gate				
A	B	$T = 0°C$	$T = 25°C$	$T = 50°C$
0	0	0.17	0.47	1.1
0	1	1.36	3.19	6.52
1	0	1.02	2.44	5.09
1	1	0.48	1.47	3.79

Table 4.2: Leakage current (nA) in various CMOS logic gates (65nm technology).

Inverter gate			
A	$T = 0°C$	$T = 25°C$	$T = 50°C$
0	2.67	2.98	3.66
1	0.13	0.47	1.40

NAND2 gate				
A	B	$T = 0°C$	$T = 25°C$	$T = 50°C$
0	0	2.37	2.45	2.59
0	1	2.65	2.98	3.66
1	0	2.52	2.77	3.29
1	1	0.26	0.94	2.81

4.1b. For example, let us compare the two cases $B = 0$ and $B = 1$, assuming $A = 1$: when $B = 0$, leakage is greater than that with $B = 1$, as was already observed for the inverter gate. Similar considerations can be easily reiterated and extended to a generic number of series transistors, and hence to generic static CMOS gates. Hence, the leakage current of static CMOS gates tends to exhibit significantly different values depending on the value of each input, once the other inputs are assigned. Obviously, this result applies to both combinational and sequential logic gates, and in the following it will be applied to a broad range of circuits.

4.2.1 Leakage in bit-sliced logic circuits

The strong leakage dependence on the input pattern of basic logic gates is a property that can be exploited to understand the overall leakage of more complex circuits. A practical example of complex circuits that is frequently encountered in real circuits is the case of bit-sliced structures, i.e. circuits with m-bit inputs that are made up of m identical replicas of the same building block. Examples of bit-sliced structures are Arithmetic Logic Units, registers, register files and bus drivers.

In bit-sliced structures, the overall leakage is equal to the sum of the leakage currents of the m bit slices, each of which is assumed to be equal to the high (low) level I_H (I_L) when the corresponding input bit is high (low), as was discussed in the previous subsection (the dual case where I_H is associated with a low input is treated in the same way). Since the number of bit slices having a high leakage current I_H is equal to the number of input bits equal to 1, or equivalently the Hamming weight w of the input word, the overall leakage results to

$$I_{leak,TOT} = w \cdot I_H + (m - w) \cdot I_L = w \cdot (I_H - I_L) + m \cdot I_L \,. \qquad (4.2)$$

From (4.2), the overall leakage linearly depends on the Hamming weight w of the input word, rather than the specific value of each bit.

4.2.2 Leakage dependence on the Hamming weight: simulation and experimental results

The dependence of leakage on the input Hamming weight in (4.2) is confirmed by simulation results in Table 4.3 on a 4-bit register in a 65nm technology, assuming a temperature of $27°C$ (very similar results are obtained for the 90nm technology, hence they are omitted in the following). From this table, it is apparent that the register leakage only depends on the weight w of the input data, and this dependence is confirmed to be approximately linear according to Figure 4.2, which plots the register leakage current versus w. Parameters I_L and I_H in (4.2) that fit the curve in Figure 4.2 are found to be $9.86nA$ and $18.58nA$, respectively. To further assess the result in (4.2), experimental measurements were performed on an off-the-shelf ON Semiconductor 8-bit register of MC74ACT273N family. With these measurements, the effect of process variations and temperature was

Table 4.3: Simulated register leakage for different input data values (65nm technology, $T = 27°C$).

Input X	Hamming weight $w = H(X)$	$I_{leak,TOT}$ $[nA]$
0000	0	39.44
0001	1	47.05
0010	1	47.05
0100	1	47.05
1000	1	47.05
0011	2	54.01
0101	2	54.01
0110	2	54.01
1001	2	54.01
1010	2	54.01
1100	2	54.01
0111	3	63.39
1011	3	63.39
1101	3	63.39
1110	3	63.39
1111	4	74.30

also captured. In particular, one hundred measurements were performed on each of five different chips at different assigned temperatures. As an example, the measured leakage obtained at $T = 43°C$ is plotted versus the input Hamming weight for the five chips in Figure 4.3. To be more specific, the values plotted for each chip are the average value among one hundred repeated measurements. The estimated standard deviation of these measurements was found to be lower than the average by two orders of magnitude, which confirms that the measurements are reliable and repeatable even in the presence of process variations. According to Figure 4.3, the linear trend of leakage approximately holds even under process variations that are seen both within the chip and among the five chips. More comments on process variations will be provided in Section 4.7, where the slight deviation from the linear trend will be explained.

The effect of temperature on (4.2) was experimentally observed by repeating the above measurements in a very wide range of temperatures, from $27°C$ to $85°C$. For example, the leakage measurement for $T = 85°C$ is reported in Figure 4.4, in which the leakage trend is similar to that in Figure 4.3, and this was also observed at other temperatures.

Hence, the simple model in (4.2) is confirmed to be valid regardless of the specific operating temperature. This can be justified from (4.1), by considering that the main impact of temperature on leakage is an equal exponential increase in all transistor leakage currents (or equivalently in I_L and I_H), which preserves the linear dependence in (4.2). This is also apparent from the comparison of Figs. 4.3 and 4.4, in which the trend of $I_{leak,TOT}$ versus w and the curve slope is almost the same regardless of T, and the only difference is the increase of all

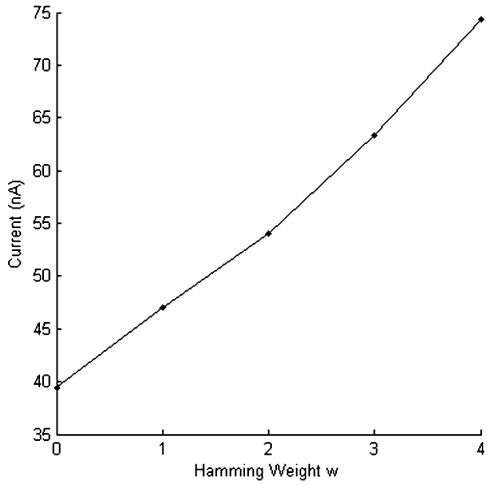


Figure 4.2: Simulated leakage vs. Hamming weight in 4-bit registers at $T = 27^{\circ}C$ (65nm technology).

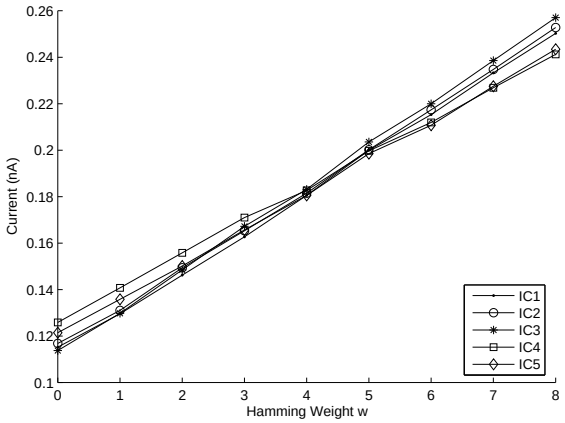


Figure 4.3: Measured leakage vs. Hamming weight in an ON Semiconductor 8-bit register for five different chips ($T = 43^{\circ}C$).

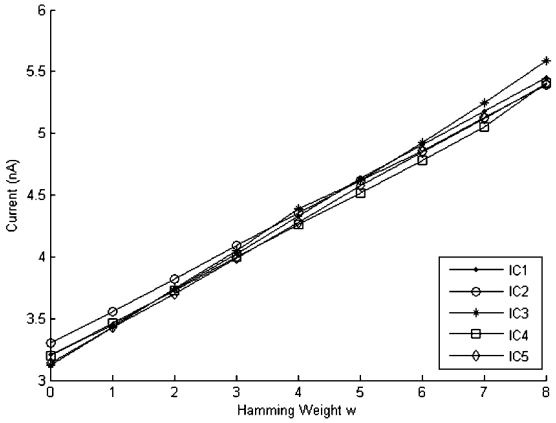


Figure 4.4: Measured leakage vs. Hamming weight in an ON Semiconductor 8-bit register for five different chips ($T = 85°C$).

leakage currents by a factor of about 20.

Summarizing, the leakage current of a bit-sliced structure is directly related to the Hamming weight w of the input data word, thus it can reveal a significant amount of information on the processed data. This fact is exploited in the following section to propose a novel class of Power Analysis attacks.

4.3 Leakage power analysis: a novel class of side-channel attacks

As was discussed in Section 4.2, the leakage current reveals the Hamming weight of the m-bit data X that is processed within a given circuit block. Hence, leakage provides useful information to recover the secret key k of a cryptographic device if the processed data X under attack are a function (or a portion) of k. Hence, a Leakage Power Analysis (LPA) attack can be conceived that exploits the leakage measurement, as discussed in the following.

In real circuits, the processed data X under attack are generated within a given circuit block, which in general is only a part of the entire chip. In practical cases, the power supply node of each block within the chip is not accessible, hence the adversary can only measure the overall chip leakage, which also includes the contribution of the considered block. Hence, the overall chip leakage $I_{leak,TOT}$ depends on the Hamming weight $w = H(X)$ of the signal X under attack (being H the Hamming weight operator), but it also includes many other leakage contributions due to the other blocks within the same chip. As a consequence, when applying random but known input values, the chip leakage $I_{leak,TOT}$ and $H(X)$ are statistically correlated. This is exactly the premise of the Correlation Power Analysis (CPA) attack ([16, 61]), which is based on

the measurement of dynamic power waveform (Section 2.2.3). As a result, a similar attack procedure can be used in Power Analysis attacks that are based on leakage measurements, which in the following will be referred to as "Leakage Power Analysis" (LPA) attacks.

The following LPA procedure is similar to the CPA presented in [61], and consists of five steps according to Figure 4.5. In the first step of LPA attacks, the adversary chooses an internal $m - bit$ signal X that is physically generated within the cryptographic circuit under attack. In general, signal X depends on both the input I and the secret key k of the cryptographic algorithm according to a well-defined function f

$$X = f(I, k) \tag{4.3}$$

where f is set by the algorithm, and hence is known by the adversary.

In the second step, the adversary applies 2^m different input values I_i (with $i = 1 \ldots 2^m$), and measures the corresponding leakage current $I_{leak,i}$ of the cryptographic chip at the point of time in which X is physically evaluated. In principle, this requires the knowledge of the clock period in which X is physically evaluated, as will be assumed in the following for the sake of simplicity. Nevertheless, this assumption can be relaxed, as will be discussed in Section 4.4. As a result of this step, an array $I_{leak,i}$ with size 2^m is obtained (see Figure 4.5).

In the third step, the physical value of X within the chip is estimated for each input I_i according to (4.3). Since the generic input I_i is applied by the adversary, the only unknown variable in (4.3) is the secret key k, hence it must be guessed. For each possible guess k_j of the secret key (with $j = 1 \ldots 2^m$), the resulting value of $X_{ij} = f(I_i, k_j)$ under the generic input I_i is found according to (4.3). As a result of this step, a 2-D array X_{ij} is found.

In the fourth step, the leakage current of the block generating X is estimated. In particular, thanks to the linear relationship between the leakage current within the block generating X and the Hamming weight $H(X)$ in (4.2), the current leakage is estimated by $H(X)$. In other words, $H(X)$ differs from the measured leakage only by an unknown multiplicative constant. The output of this step is a 2-D array $H_{ij} = H(X_{ij})$, with $i = 1 \ldots 2^m$ and $j = 1 \ldots 2^m$, which contains the Hamming weight of X for all applied inputs and key guesses.

In the fifth step, the measured leakage $I_{leak,i}$ and the estimated leakage H_{ij} are compared. For a given key guess k_j, the sequences $I_{leak,i}$ and H_{ij} associated with the random (but known) sequence of inputs I_i (with $i = 1 \ldots 2^m$) can be thought of as random variables. When the key guess is correct (i.e., $k_j = k$), the estimated and measured leakage are maximally correlated. Theoretically, if the linear dependence of $I_{leak,i}$ on $H(X)$ in (4.2) were exact and there were no other leakage contributions, the correlation coefficient $\rho(I_{leak,i}, H_{ij})$ between $I_{leak,i}$ and H_{ij} with $k_j = k$ would be exactly equal to 1 from basic statistics [107]. On the other hand, if the key guess is wrong (i.e., $k_j \neq k$), the measured leakage is no longer linearly related to the estimated $H(X)$, hence the measured leakage and $H(X)$ are loosely correlated and the correlation coefficient $\rho(I_{leak,i}, H_{ij})$ is

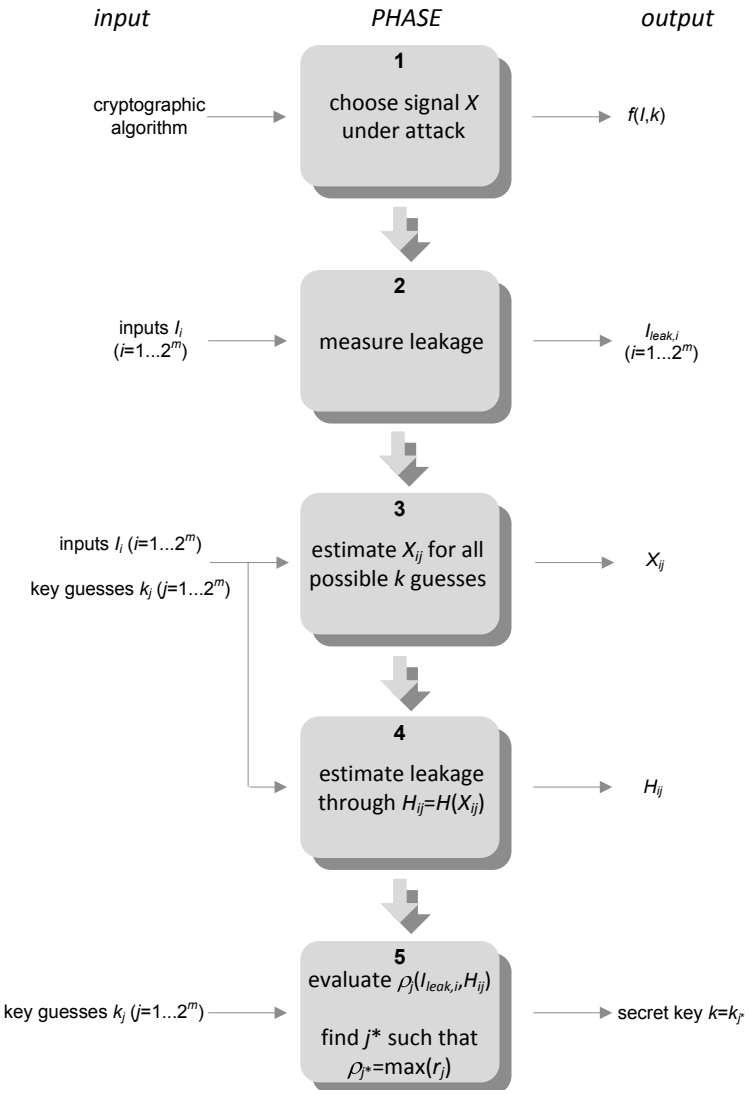


Figure 4.5: LPA attack procedure.

lower than unity. Obviously, leakage is still linearly dependent on the Hamming weight of the signal X that is physically evaluated within the cryptographic circuit but the wrong guess of k leads to an incorrect estimation of X and hence of $H(X)$, thus (4.2) no longer holds.

This means that the correct guess of k (i.e., the secret key) is that leading to the highest value of $\rho(I_{leak,i}, H_{ij})$ among all possible guesses k_j. Hence, the adversary must evaluate the correlation coefficients $\rho_j = \rho(I_{leak,i}, H_{ij})$ between the measured leakage $I_{leak,i}$ and the Hamming weights H_{ij} for $j = 1...2^m$, and identify the value j^* of j that maximizes ρ_j as in (4.4)

$$\rho_{j^*} = \max_j \rho_j \tag{4.4}$$

and the secret key is simply equal to

$$k = k_{j^*}. \tag{4.5}$$

This is in accordance with the final step of CPA attacks, although they are based on dynamic power measurements [1, 16, 61, 92].

4.4 Practical considerations on LPA attacks

In the following, practical issues and examples for each of the steps of the above LPA attack are discussed for cryptographic devices based on a microprocessor and an embedded core.

In regard to the first step in Figure 4.5, the signal X under attack can be easily chosen from the knowledge of the algorithm. To reduce the attack effort, X must be chosen as an internal signal whose dependence $f(I, k)$ on I and k is as simple as possible. For example, in microprocessor-based implementations of DES and AES algorithm, the sub-keys k are loaded into registers to perform the XOR with the ciphertext I (plaintext) during the decryption (encryption) phase [53]. In these cases, the signal X under attack can be chosen as the input of a register in the register file that stores k (or the XOR of k and the input, $I \oplus k$, which is successively evaluated), hence the simple function $f(I, k) = I$ (or $f(I, k) = I \oplus k$ with $k = 000...00$) can be adopted in (4.2). In crypto-devices based on an embedded core, similar considerations can be made, although the register under consideration is not in the register file, but divides two pipeline stages.

An alternative choice for the signal X is the input (or output) of the bus drivers, since most processed data are transferred through busses to the memory, and their leakage is a significant fraction of the overall chip leakage [69] (hence it can be easily measured [10, 11]). As far as the second step in Figure 4.5 is concerned, the leakage measurements must be carefully performed. Indeed, once the input is applied to a logic gate, its leakage current is well-known to have a transient variation, and finally settles to the steady-state value after a period ranging from less than one nanosecond to a few tens of nanosecond [69]. As an example, Table 4.4 reports the leakage settling time of a 2-input CMOS NAND

Table 4.4: Leakage current settling time for each input transition in a 2-input NAND gate (65 nm technology, $T = 27°C$).

From		To		Settling
A	B	A	B	time
0	0	1	0	35.34 ns
0	0	0	1	239.01 ps
0	1	1	1	1.45 ns
0	1	0	0	509.81 ps
1	0	0	0	98.26 ps
1	0	1	1	1.51 ns
1	1	0	1	263.05 ps
1	1	1	0	89.89 ns

gate in a 65nm technology (very similar results were obtained in the 90nm technology). From these considerations, the clock period is generally shorter than the period required to observe the steady-state leakage. As a consequence, the adversary must stop the clock signal at least for a few clock periods to measure $I_{leak,i}$, starting from the period in which X is physically evaluated. Therefore, this period of time with a stopped clock adds to the attack duration, thus slightly reducing the number of attacks that can be performed in a time slot.

It is worth noting that stopping the clock is usually feasible, as the adversary usually has access to the reference clock that is generated outside the chip. Moreover, the exact clock period in which X is evaluated within the chip can be found with a moderate effort, if the adversary has sufficient knowledge of the circuit implementation of the algorithm. For example, in microprocessor-based implementations the correct clock period can be identified by counting the number of memory accesses that are required prior to the evaluation of X. In embedded core implementations, the identification of the clock period in which X is physically evaluated is even easier, as the circuit implementation usually consists of a low number of pipeline stages [108], hence the number of clock cycles to consider is rather low (a few units, typically). Even in the case where the adversary does not exactly know this clock cycle, but he/she knows that it is among a limited number l of clock cycles, he/she can reiterate the procedure in Section 4.3 for each period and then evaluate the maximum correlation coefficients ρ_j for each of the l periods under analysis. Obviously, the maximum ρ_j is expected to be achieved in the correct period, hence the secret key is easily found by selecting the guess of k and the period that maximize ρ_j. Since this increases the number of guesses by a factor of l (and hence the attack effort), the adversary must have at least a rough idea on the clock period in which the signal under attack X is physically evaluated.

Regarding the leakage measurement setup, it should be observed that leakage measurements are in principle simpler to carry out, compared to dynamic power measurements in traditional DPA/CPA attacks. In fact, the latter ones

require the acquisition of power consumption waveforms with a high bandwidth measurement setup and a high sample-rate digital oscilloscope. Leakage measurements do not require either high bandwidth or high sample-rate oscilloscopes, as they are simply DC measurements that can be carried out using a simple ammeter. Typically, ammeters with pico-Ampere accuracy can be found at very low prices (in the range of 500-1,000 $), hence the costs involved in the attack are extremely low (even lower than those in DPA/CPA attacks). As a further advantage of LPA, leakage measurements are rather insensitive to AC additive noise since they are based on measurement averaging over a sufficiently long period of time, as any other DC measurement.

A distinctive feature of LPA attacks is the high sensitivity to temperature, as pointed out in Section 4.2, which deserves some attention during measurements. Indeed, in Section 4.2.2 it was shown that the leakage model in (4.2) holds regardless of the operating temperature. Nevertheless, it should be observed that this is true only if the chip temperature is not time varying, otherwise the leakage current may significantly increase (decrease) despite of a reduction (increase) in the Hamming weight in (4.2), due to an unexpected temperature increase (decrease). For this reason, it is important that the chip temperature is kept constant during LPA attacks. In our experimental results, temperature was set by means of a Peltier cell driven by a proper constant voltage.

The third and fourth step of LPA attacks do not exhibit particular problems from an experimental point of view, but the choice of the bit width m of the signal X under attack is crucial in terms of computational effort. Indeed, the number of guesses exponentially grows as 2^m, hence m must be at most 7-8 for computationally feasible attacks. It is worth noting that this does not limit the effectiveness of LPA attacks, as the overall key can be recovered by reiterating the attack for the remaining m-bit portions of the key. This is feasible in practical cases because the intermediate results of a cryptographic device consist of small words that depend on a few bits of the key and not on the entire key (for example, in microprocessor-based devices the width of the portion of the key that is involved in a single encryption step is no greater than the processor word length). Like any cryptographic operation, these intermediate results are evaluated by combinational logic and stored in a register at a certain time during the computation of the algorithm.

In regard to the last step in Figure 4.5, the correlation coefficient $\rho_j = \rho(I_{leak,i}, H_{ij})$ cannot be exactly evaluated since a finite number of samples of $I_{leak,i}$ and H_{ij} are considered. In practical cases, $\rho_j = \rho(I_{leak,i}, H_{ij})$ is estimated by the sample correlation coefficient (often called "Pearson's Correlation Coefficient") in (4.6), where the number of applied inputs was assumed to be 2^m for simplicity [107]

$$\rho_j = \frac{\sum_{i=1}^{2^m} I_{leak,i} H_{ij} - 2^m \overline{I_{leak,i}} \cdot \overline{H_{ij}}}{(2^m - 1)\, s_{I_{leak,i}} s_{H_{ij}}} \tag{4.6}$$

where the sample mean $\overline{I_{leak,i}}$ and $\overline{H_{ij}}$ are defined as

$$\overline{I_{leak,i}} = \frac{1}{2^m} \sum_{i=1}^{2^m} I_{leak,i} \tag{4.7}$$

$$\overline{H_{ij}} = \frac{1}{2^m} \sum_{i=1}^{2^m} H_{ij} \tag{4.8}$$

and the sample standard deviation $s_{I_{leak,i}}$and $s_{H_{ij}}$ are

$$s_{I_{leak,i}} = \sqrt{\frac{1}{2^m - 1} \sum_{i=1}^{2^m} \left(I_{leak,i} - \overline{I_{leak,i}}\right)^2} \tag{4.9}$$

$$s_{H_{ij}} = \sqrt{\frac{1}{2^m - 1} \sum_{i=1}^{2^m} \left(H_{ij} - \overline{H_{ij}}\right)^2} \tag{4.10}$$

Finally, it is useful to observe that the overall chip leakage contribution includes the leakage current of many other blocks that do not physically evaluate X, as occurs with the dynamic power consumption in traditional CPA attacks [16]. These contributions affect both $I_{leak,i}$ and $\overline{I_{leak,i}}$ in (4.6), and tend to reduce the sample correlation coefficient with respect to the ideal value of 1, as was observed in [16] for CPA attacks. In particular, in the case where the leakage contributions of the other blocks are perfectly constant, they do not effect r_j at all, as is apparent from (4.6). The same result approximately holds also when these contributions randomly vary with a reasonably uniform distribution when applying the inputs I_i, as they tend to average out [16]. This intuitively justifies why LPA attacks can be performed even in the presence of significant leakage contributions that add to the leakage current of the block under attack. This issue will be further discussed through an example in Section 4.6.

4.5 An analytical model of the correlation coefficient in LPA attacks

LPA attacks rely on the assumption that the correlation coefficient ρ_{j^*} associated with the correct key can be discriminated from that of wrong guesses. This requires that the value of ρ_{j^*} under the correct key (i.e., $\rho_{j^*} = 1$) is sufficiently greater than those under the other guesses. For this reason, in the following the correlation coefficient under wrong guesses is analytically evaluated according to the approach in [8].

For the sake of simplicity, let us assume with no loss of generality that $I_L = 0$ and $I_H = 1$ in (4.2), that the signal under attack X is the XOR of the input I and the key portion k (i.e., $f(I,k) = I \oplus k$, for the reasons discussed in Section 4.4), that the key is $k = 00\ldots00$ (so that $f(I,k) = I$), and let us apply all 2^m possible m-bit input values I_i to a bit-sliced circuit. The resulting leakage $I_{leak,i}$ associated with the generic value of X is linearly related to its Hamming

weight $H(X)$ according to (4.2). Hence, the correlation coefficient is equal to 1 when the correct key guess has been chosen, and is lower than 1 when a wrong guess is made.

In the following, it will be matematically demonstrated that the higher is the number of wrong bits in k (or in X, since $X = f(I,k) = I \oplus k$) the lower is the correlation coefficient ρ_{wrong}, and this dependence is linear.

The ρ_{wrong} is evaluated in the general case where the key guess is wrong by an arbitrary number of bits e (with $e = 1 \dots m$). For simplicity, let us assume that the wrong bits in X are the most significant e bits, which define an e-bit string that will be named X^{MSB} in the following. Let us also define W as the word obtained complementing the most significant e bits in X (i.e., W is a guess of X that is wrong by e bits), and W^{MSB} as the string made up of the most significant e bits in W (i.e., the wrong bits of X).

Let us consider all 2^m possible inputs I_i,and assume that they are ordered (from 1 to 2^m) according to the binary value of the resulting X_i (from $00\dots0$ to $11\dots1$), i.e. I_1 is the input leading to $X = 00\dots00$, I_2 is the input leading to $X = 00\dots01$, and so on. For a given input I_i (for $i = 1\dots2^m$), the corresponding values of the correct guess X_i and the wrong guess W_i have a different Hamming weight ($H(W_i)$ and $H(X_i)$, respectively), due to the different contribution of the most significant e bits. However, as discussed above, the bit strings consisting of the $(m-e)$ least significant bits of X_i and W_i are equal, hence their Hamming weight (i.e., $H(X_i) - H\left(X_i^{MSB}\right)$ and $H(W_i) - H\left(W_i^{MSB}\right)$ respectively) is the same, thereby yielding

$$H(W_i) = H(X_i) + \left[H\left(W_i^{MSB}\right) - H\left(X_i^{MSB}\right)\right]. \tag{4.11}$$

Let us an example for the simplest case $e = 1$, with $m = 8$. In this particular case, only the most significative bit make the difference between $H(W_i)$ and $H(X_i)$. By observing that X^{MSB} in this case can be only 0 or 1 (and W^{XMB} only 1 or 0 respectively) it follows that $H\left(W_i^{MSB}\right) - H\left(X_i^{MSB}\right) = \pm 1$. Therefore the relationship between $H(W_i)$ and $H(X_i)$ (defined in (4.11) for the general case) becomes

$$H(W_i) = \begin{cases} H(X_i) - 1 & , X^{MSB} = 1 \\ H(X_i) + 1 & , X^{MSB} = 0 \end{cases} \tag{4.12}$$

Hence this means that the equation (4.11), describing the generic case with e bits wrong, corresponds to 2^e relationship between $H(W_i)$ and $H(X_i)$. Equation (4.12) shows a dependence only on $H(W_i)$ of $H(X_i)$. It is possible to explicit the same dependence in (4.11).

Since W_i^{MSB} is the bit-wise complement of X_i^{MSB}, the sum of the Hamming weight of $X_i^{MSB}$and W_i^{MSB} is equal to e, hence

$$H\left(W_i^{MSB}\right) = e - H\left(X_i^{MSB}\right). \tag{4.13}$$

By substituting (4.13) in (4.11), it immediately follow that

$$H(W_i) = H(X_i) + \left[e - 2 \cdot H\left(X_i^{MSB}\right)\right]. \tag{4.14}$$

Equation (4.14) is useful to evaluate the correlation coefficient ρ_{wrong} between $H(W_i)$ and the leakage $I_{leak,i}$ corresponding to the same value of X_i, which by definition is

$$\rho_{wrong} = \frac{\sum_{i=1}^{2^m}\left(H(W_i) - \overline{H(W)}\right)\cdot\left(I_{leak,i} - \overline{I_{leak,i}}\right)}{\sqrt{\sum_{i=1}^{2^m}\left(H(W_i) - \overline{H(W)}\right)^2 \cdot \sum_{i=1}^{2^m}\left(I_{leak,i} - \overline{I_{leak,i}}\right)^2}}. \tag{4.15}$$

From equation (4.2) follows that the leakage $I_{leak,i}$ in (4.15) is simply equal to its Hamming weight $H(X_i)$ by the assumption that $I_L = 0$ and $I_H = 1$. It is useful to observe that $H(X_i)$ and $H(W_i)$ in (4.15) have the same average, since they are evaluated over the same set of values X_i. Obviously the only difference is the order, as W_i is evaluated by complementing the e most significant bits of X_i, as discussed before. For the same reason $\sum_{i=1}^{2^m}\left(H(W_i) - \overline{H(W)}\right) = \sum_{i=1}^{2^m}\left(H(X_i) - \overline{H(X)}\right)$.

Assuming that inputs are uniformly distributed m-bit symbol, the average $\overline{H(X)}$and variance $\frac{1}{2^m}\sum_{i=1}^{2^m}\left(H(W_i) - \overline{H(W)}\right)^2$ of the Hamming weight result to be equal to $m/2$and $m/4$ respectively. Starting from this consideration and by substituting (4.14) in (4.15) we get

$$\rho_{wrong} = \frac{\sum_{i=1}^{2^m}\left[H(X_i) + \left(e - 2\cdot H\left(X_i^{MSB}\right)\right) - \frac{m}{2}\right]\left(H(X_i) - \frac{m}{2}\right)}{2^m \frac{m}{4}}. \tag{4.16}$$

To further simplify (4.16), observe that all terms X_i with $i = 1 \ldots 2^{m-e}$ have the same X_i^{MSB} (i.e., the first e LSBs), and this can be easily shown to hold also for all terms X_i with $i = (c-1)\cdot 2^{m-e} + 1 \ldots c \cdot 2^{m-e}$, being c an assigned integer number ranging from 1 to 2^e. Accordingly, we can group terms X_i in 2^e different sets (each being identified by $c = 1 \ldots 2^e$). As an example, the case with $e = 2$ is shown in Table 4.5, where there are 2^2 sets, in each of which all X_i's have the same X_i^{MSB}(00 for the first set, 01 for the second, ...). according to this grouping, (4.16) can be written as

$$\begin{aligned}\rho_{wrong} &= \frac{\sum_{c=1}^{2^e}\sum_{i=(c-1)2^{m-e}+1}^{c\cdot 2^{m-e}}\left[H(X_i) + \left(e - 2\cdot H\left(X_i^{MSB}\right) - \frac{m}{2}\right)\right]\left(H(X_i) - \frac{m}{2}\right)}{2^m \frac{m}{4}} \\ &= 1 + \frac{\sum_{c=1}^{2^e}\sum_{i=(c-1)2^{m-e}+1}^{c\cdot 2^{m-e}}\left(H(X_i) - \frac{m}{2}\right)\left(e - 2\cdot H\left(X_i^{MSB}\right)\right)}{2^m \frac{m}{4}}\end{aligned} \tag{4.17}$$

Table 4.5: Grouping terms X_i in 2^e sets according to X_i^{MSB} example with $e = 2$.

terms X_i having X_i^{MSB}=**00**	I	X_i	I	X_i	terms X_i having X_i^{MSB}=**01**
	1	**00**0…00	2^{m-2}+1	**01**0…00	
	2	**00**0…01	2^{m-2}+2	**01**0…01	
	3	**00**0…10	2^{m-2}+3	**01**0…10	
	…	…	…	…	
	2^{m-2}	**00**1…11	$2 \cdot 2^{m-2}$	**01**1…11	
terms X_i having X_i^{MSB}=**10**	I	X_i	i	X_i	terms X_i having X_i^{MSB}=**11**
	$2 \cdot 2^{m-2}$+1	**10**0…00	$3 \cdot 2^{m-2}$+1	**11**0…00	
	$2 \cdot 2^{m-2}$+2	**10**0…01	$3 \cdot 2^{m-2}$+2	**11**0…01	
	$2 \cdot 2^{m-2}$+3	**10**0…10	$3 \cdot 2^{m-2}$+3	**11**0…10	
	…	…	…	…	
	$3 \cdot 2^{m-2}$	**10**1…11	2^m	**11**1…11	

where terms X_i belonging to the same set (according to index c) were grouped under the same summation, and a few analytical manipulations were introduced. Moreover, according to Table 4.5, by construction the $(m-e)$ least significant bits of each term X_i within a given set are exactly the same as the corresponding terms of any other set (i.e., the $(m-e)$ LSBs of the first term are the same for all sets, and the same holds for the second term, the third term...). From the above considerations, after simple but tedious calculations, the following properties can be derive

$$\sum_{i=(c-1)2^{m-e}+1}^{c\cdot 2^{m-e}} H\left(X_i\right) = \sum_{i=1}^{2^{m-e}} H\left(X_{(c-1)2^{m-e}+i}\right) \tag{4.18}$$

$$\left(e - 2 \cdot H\left(X_{(c-1)2^{m-e}+i}^{MSB}\right)\right) = -\left(e - 2 \cdot H\left(X_{(2^e-c)2^{m-e}+i}^{MSB}\right)\right) \tag{4.19}$$

$$H\left(X_{(c-1)2^{m-e}+i}^{MSB}\right) + H\left(X_{(2^e-c)2^{m-e}+i}^{MSB}\right) = e \tag{4.20}$$

for $c = 1...2^e$. By substituting (4.18)-(4.20) into (4.17) and performing a few analytical manipulations, the final expression of the correlation coefficient is found to be

$$\rho_{wrong} = 1 - \frac{2^{m-e} \cdot e \cdot 2^{e-1}}{2^m \frac{m}{4}} = 1 - \frac{2}{m} \cdot e. \tag{4.21}$$

Equation (4.21) mathematically proves that the correlation coefficient linearly depends on the number of wrong bits and therefore the most difficult cases to discriminate from the correct guess are those with a minimum number of wrong bits in X, i.e. wrong by just one bit. Hence the wrong correlation coefficient closer to the correlation coefficient corresponding to the correct guess is

$$\rho_{wrong} = 1 - \frac{2}{m}. \tag{4.22}$$

Moreover, equation (4.22) highlights that ρ_{wrong} is easily discriminated from the correlation coefficient value associated with the correct guess (i.e., 1) if m is a few units. On the other hand, if m is in the order of 8-10, ρ_{wrong} gets very close to unity, hence it is very hard to distinguish the case of a wrong guess from the correct key. For this reasons, LPA attacks can be successful only if the number of bits under attack m is not greater than about 8. It is important to note that this limit does not constitute an inconvenience because the target of an attack is not the whole key but normally the key is divided into some sub-keys to reduce effort of the attack.

4.6 Examples of LPA attacks

Let us apply the LPA attack procedure explained in Section 4.3 to three different circuits. In the first one, the 4-bit register in 65nm technology discussed in Section 4.2 is attacked in simulation. Let us apply a sequence of all possible 4-bit input values in the first column in Table 4.3, and stop the clock signal at its high value. The measured leakage $I_{leak,i}$ and the Hamming weight $H(X)$ for each input value are reported in the second and third column of Table 4.3. The sample correlation coefficient ρ between the measured leakage vector $I_{leak,i}$ (with $i = 1 \ldots 16$) and the Hamming weight vector $H(X)$ is then computed according to (4.6). Using data in Table 4.3, ρ results to 1 for the correct logic vector as expected, while ρ results to 0.41 if only one wrong bit is considered. The latter value reasonably agrees with the theoretical value of 0.5, which is obtained from (4.22) after setting $m = 4$. Hence, the LPA attack allows for identifying the correct guess, as the corresponding correlation coefficient is the highest. Moreover, it can be easily distinguished from the correlation coefficient obtained with a key guess that is wrong by one bit.

In the above attack, a bit-slice circuit (i.e., a register) was considered. The register is an ideal candidate as a block to attack with LPA for the previously explained reasons. Nevertheless, the above considerations and attack procedure can be also applied to circuits that are not bit sliced and involve non-linear transformations. Indeed, even in this case leakage has a strong correlation with the Hamming weight of the input data, although this dependence is no longer linear due to the non-linear transformation. In the following, this is shown by attacking the well-known Serpent 4x4 S-Box transformation, whose truth table is reported in the first and second column of Table 4.6 [3].

As a second example of LPA attack, the simple crypto core in Figure 4.6 based on the Serpent S-Box transformation was considered. This crypto core was synthesized in Cadence environment using a 65nm CMOS cell library. As occurs in many cryptographic algorithms, in Figure 4.6 the plain word and the secret key are mixed in advance by XOR gates and the result is ciphered by S-BOX (a similar structure is observed in many other ciphers, such as DES and

Table 4.6: S-Box truth table and leakage current (65nm technology, $T = 25°C$ and $100°C$).

IN	OUT	$I_{leak,i}(nA)$ @ $T = 25°C$	$I_{leak,i}(nA)$ @ $T = 100°C$
0100	1111	114.6	941.6
1001	1101	115.0	950.0
0000	0011	117.1	950.2
0001	1110	117.9	959.7
1000	1000	118.8	964.0
0011	0111	121.9	999.8
1100	0001	122.9	1007.8
0010	1010	123.4	1016.2
1011	0000	124.0	1024.0
1010	0110	124.8	1026.3
1111	1100	125.4	1031.6
0101	0100	127.1	1031.9
1101	0010	128.2	1032.7
0110	0101	129.6	1051.9
1110	1011	130.7	1052.8
0111	1001	131.6	1071.7

AES). Simulations on this crypto core were carried out by exploring all possible combinations of plain words and keys. The results are shown in Table 4.6, where leakage current values are sorted in an increasing order. Interestingly, sorting the input combinations in an increasing order according to the leakage, the order is the same regardless of the temperature, as was observed over an extremely large range of temperatures (much wider than realistic operating temperatures). Again, the LPA attack effectiveness is expected to be independent of the operating temperature, provided that it is kept constant during the attack. As an example, the cases with $T = 25°C$ and $100°C$ are shown in Table 4.6.

The resulting correlation coefficients in (4.6) of several attacks (i.e., for several keys) for $T = 25°C$ are summarized in Figure 4.7, where the '+' symbol

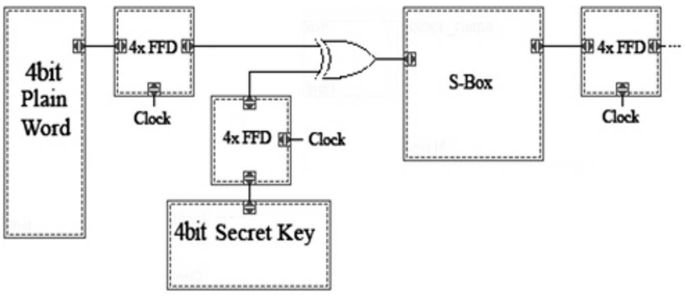


Figure 4.6: Crypto core based on Serpent S-Box.

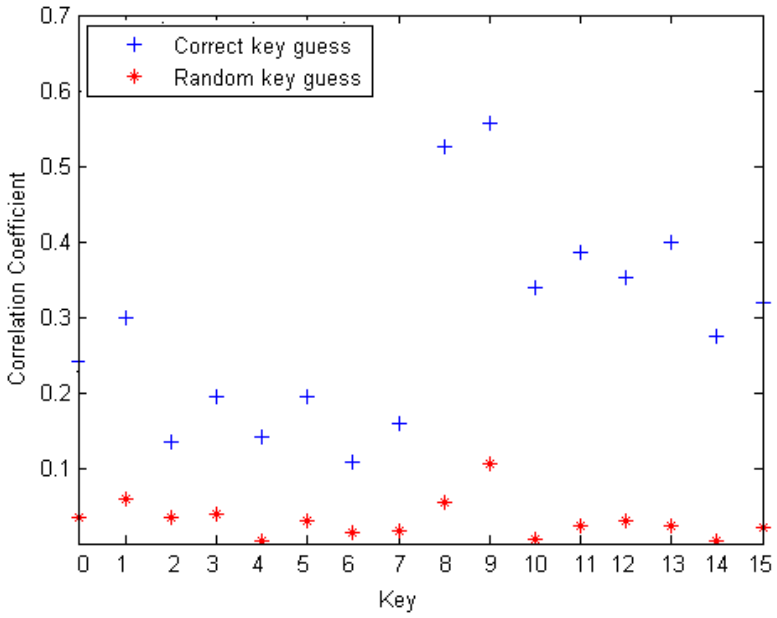


Figure 4.7: Correlation coefficient in a simulated attack.

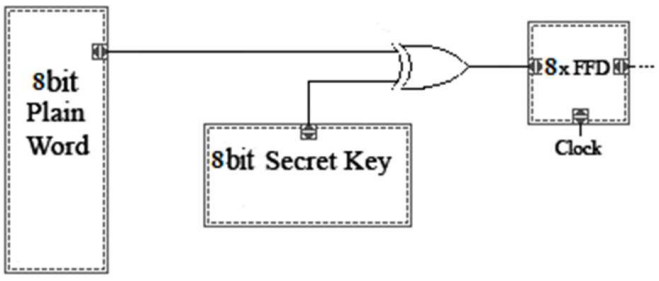


Figure 4.8: Cryptographic circuit under experimental attack.

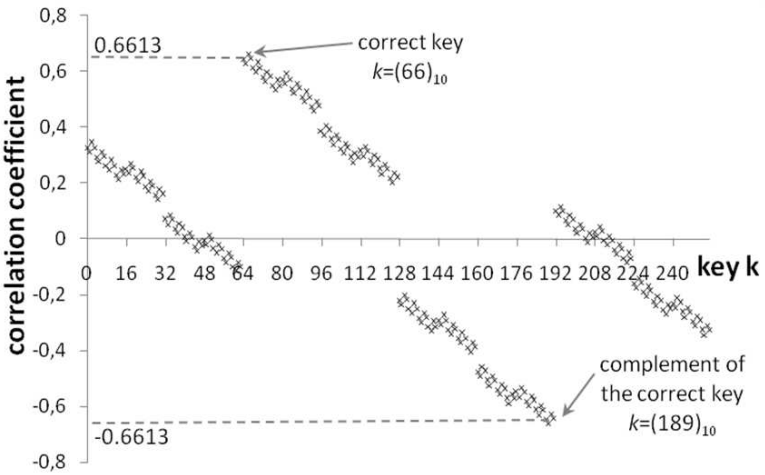


Figure 4.9: Correlation coefficient ρ_j for all possible key guesses.

denotes the right key hypothesis, whereas the '*' symbol denotes a wrong key hypothesis. Very similar results were obtained for $T = 100°C$. From Figure 4.7, the correlation coefficient turns out to be lower than unity, since the circuit is not bit sliced. The same figure reveals that all correct keys are clearly distinguishable from wrong key guesses, as the former ones always lead to greater values of the correlation coefficient. More specifically, the minimum value of the correlation coefficient under a correct key is 0.11, whereas the maximum value under a wrong key guess is 0.10, and in other cases the difference is much larger.

The third example of LPA attack has been conducted on a real circuit. In particular, a portion of the crypto core in Figure 4.6 was implemented in the eight bit architecture in Figure 4.8, employing an off-the-shelf ON Semiconductor 8-bit register of MC74ACT273N family. In particular, 256 leakage measurements have been collected for all possible 8-bit input values with a fixed key. Then, the correlation coefficient between the measured leakage and the predicted Hamming weight H(X) was evaluated for each of the 256 key hypotheses. The resulting magnitude of the correlation coefficient normalized to the maximum value are shown in Figure 4.9 for all key guesses. From this figure, the highest value of the correlation coefficient is obtained under the correct key $(01000010)_2 = (66)_{10}$, as expected. There is also another equal peak with opposite sign that is obtained under the symmetric key $(10111101)_2 = (189)_{10}$, i.e. by complementing all bits of the correct key. This is easily explained by considering that the XOR function is symmetric; hence, by complementing the key bits, the XOR with the input is also bit-wise complemented.

Several other attacks with different keys and at various temperatures were repeated, and results showed that they were always successful, as the correct key was always associated with the greatest value of ρ. This confirms the effectiveness of LPA attacks on real hardware and in real conditions.

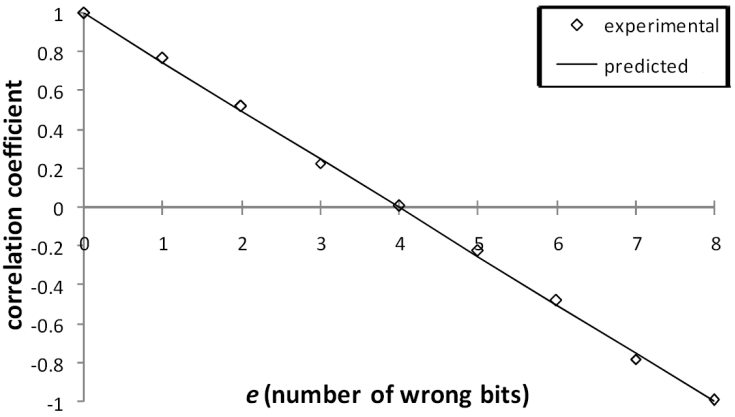


Figure 4.10: Experimental and predicted correlation coefficient vs. e.

Finally, to validate the model in (4.21), the LPA attack, experimentally conducted on the crypto core in Figure 4.8, has been used. The correlation coefficient between the measured leakage and the predicted Hamming weight $H(X)$ was evaluated for each of the 256 key hypotheses, and was normalized to the maximum value (obtained for the correct key). Each measurement was repeated 100 times and averaged to suppress the effect of noise and temperature fluctuations. The resulting correlation coefficient normalized to the maximum value is shown in Figure 4.10 versus the number of wrong bits e. Inspection of this figure shows that equation (4.21) agrees very well with experimental results, thereby confirming the linear dependence of ρ_{wrong} on e in (4.21). The same or even better accuracy was obtained for other chips of the same type at higher temperatures (up to $50°C$, which covers the temperature range of real crypto-chips).

4.7 Analysis of LPA effectiveness under process variations

Until now LPA attacks were analyzed without explicitly accounting for process variations. In the following, their effect of the attack effectiveness is discussed, since their impact tends to increase in nanometer CMOS technologies.

Process variations are usually classified into interdie and intradie. The former ones impact all devices in the same chip in the same manner [39], hence they induce a leakage variation that is equal for all transistors, according to (4.1) [93]. Thus, the leakage contributions of all gates scale by the same factor, which is similar to the effect of temperature variations previously discussed. As a consequence, the linear dependence in (4.2) is preserved, and the sample correlation coefficient (4.6) does not change. Hence, interdie variations do not affect the result of LPA attacks.

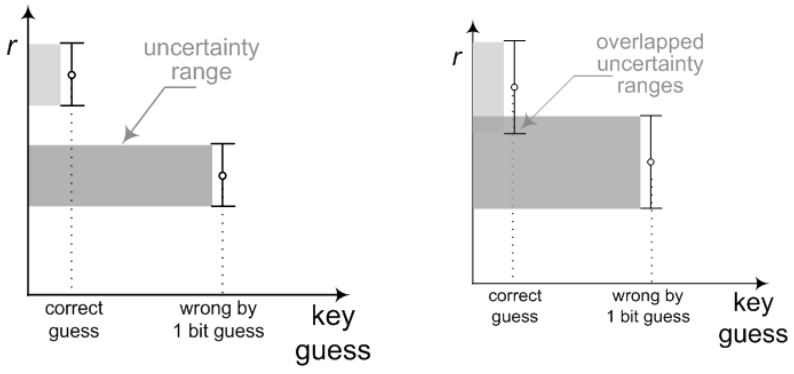
On the other hand, intradie variations affect different transistors in the same chip in a different way [67], hence two flip-flops give a different leakage contribution even when they have the same input. As a consequence, two different input patterns with the same weight w lead to different register leakage currents. Hence, in this case the leakage depends on the specific applied input (not only on w), i.e. a slight deviation from the linear trend in (4.2) is observed, which agrees with the experimental results in Figures 4.3 and 4.4. Hence, intradie variations can potentially reduce the effectiveness of LPA attacks.

To understand the effect of intradie variations, it is useful to observe that they induce a statistical variation in the correlation coefficient ρ in (4.6). Accordingly, the correlation coefficient ρ for a given key guess can be seen as an uncertainty range centered on its nominal value, as shown in Figure 4.11.a. The attack is feasible even in the presence of intradie variations if the value of ρ under the correct guess can be distinguished from that under a wrong guess, i.e. if we are reasonably sure (within a given confidence level, e.g. 99.9%) that the two uncertainty ranges do not overlap, as in Figure 4.11.a. On the other hand, if the two ranges overlap (see Figure 4.11.b) the attack may be unsuccessful.

To evaluate the amplitude of the uncertainty range, we first performed numerical simulations by adopting the simplified leakage model in (4.1) for each bit slice and introducing random intradie variations ΔV_{TH} in V_{TH}. Intradie variations of other parameters (e.g., W, L, ...) can be always described as an equivalent variation in V_{TH} [93]. Hence this assumption does not limit the generality of the subsequent considerations. Variations ΔV_{TH} were randomly generated with normal distribution, zero mean and standard deviation $\sigma_{V_{TH}}$. Obviously, for low (high) values of $\sigma_{V_{TH}}$, the uncertainty width in Figure 4.11 is small (large) and the uncertainty ranges do not (do) overlap, as shown in Figure 4.11.a (Figure 4.11.b). Accordingly, we evaluated the maximum standard deviation $\sigma_{V_{TH},max}$ of V_{TH} above which the uncertainty ranges overlap as in Figure 4.11.b, with a 99.9% confidence level.

Results show that $\sigma_{V_{TH},max}/(nkT/q) \approx 1$ regardless of the value of m, if we consider the uncertainty range of the correct key and of the key guesses that are wrong by one bit. Under the typical value $n = 1.5$ and at room temperature, $\sigma_{V_{TH},max}$ results to $37mV$, which unfortunately is close to typical values of $\sigma_{V_{TH}}$ in 65nm and 45nm CMOS technologies [62, 69]. Hence, in current technologies, variations of V_{TH} may lead to an overlap between the uncertainty range of the correct key and that of a key guess that is wrong by one bit. As a consequence, the attack may be successful in some cases (in a probabilistic sense), whereas in some others the result of the attack may be incorrect (i.e., a key guess that is wrong by one bit is mistaken for the correct key). Even in the latter case, the attack is still successful under the assumption that the correct key and the found key differ by at most by one bit: in this case, the adversary has to search the key in an extremely narrow space with $m+1$ elements (i.e., the key found in the attack, or one of the m keys that differ from it by only one bit), compared to the exhaustive search among the 2^m possible key guesses.

Now, let us justify the above assumption that the correct key and the key found in the attack differ by at most one bit. The above discussed numerical



(a) (b)

Figure 4.11: Correlation coefficient statistical distribution under moderate intradie variations in the case of a successful attack (a) and a unsuccessful attack (b).

simulations showed that $\sigma_{V_{TH},max}/(nkT/q) \approx 1.5$ when we consider the key guesses that are wrong by two bits (and even more if the number of wrong bits is higher): in this case, $\sigma_{V_{TH},max} \sim 56mV$ is significantly greater than $\sigma_{V_{TH}}$ both for current and future technologies (for example, in 32nm technologies, $\sigma_{V_{TH}}$ is projected to be about $40mV$ [62, 93]). This means that the uncertainty range of the key guesses with two or more wrong bits never overlaps with the range of the correct key (as in Figure 4.11.a), neither in current nor in future CMOS technologies. The above numerical results are in good agreement with Monte Carlo circuit simulations, which were performed on 1,000 trials of the same 65nm register circuit affected by process variations. By analyzing the statistical distribution of the resulting 1,000 values of the correlation coefficient, it was found that sometimes the correct key may be mistaken for key guesses with one wrong bit, but it is never confused with guesses with two or more wrong bits. This agrees well with the above considerations found with the simplified leakage model. In particular, it was found that the attack identifies the correct key in 96% of the cases. Hence, in most cases the attack provides the correct result, and in the other few cases the found key has at most one wrong bit, which again agrees well with previous numerical analysis.

From the above considerations, LPA attacks are expected to be successful even in the presence of intradie variations, both in current technologies and in the next technology nodes.

To validate the above analysis of LPA effectiveness under process variations an S-Box block has been considered as a representative example of a block that is vulnerable to power analysis attacks.

In particular, the impact of intra-die variations on the effectiveness of LPA attacks has been evaluated in a 4-input Serpent S-Box built with a standard

CMOS library [3]. The S-Box was designed using a 65nm CMOS cell library from STMicroelectronics in the Cadence dfII environment. BSIM4 models with statistical parameters provided by the foundry were used to guarantee the best accuracy of simulation results for what concerns both leakage current and process variations [29]. To understand how the outcome of an LPA attack is affected by intra-die variations, the attack has been performed on 400 samples of the circuit, each of which was generated by means of Monte Carlo simulations where . The 400 sample circuits generated in each one of the different experiments were all affected by mismatch variations. It is important to observe that each of 400 Monte Carlo iterations represents a realization of the circuit under test with a particular configuration of process random variables.

As a first analysis to intuitively grasp the impact of variations, the leakage current of each of the 400 sample circuits was ordered as a function of the S-Box input value i (for $i = 0 \dots 2^4 - 1$) in ascending order. In principle, under small to moderate process variations, it is expected that the maximum leakage is still obtained for the input value that leads to the minimum Hamming weight w, assuming that $I_L > I_H$. This qualitatively agrees with the plot in Figure 4.14 and Figure 4.15, where leakage is plotted versus the input Hamming weight for the considered realizations of the circuit. Apparently, the trend still resembles a linear dependence as in Figure 4.15, and the maximum leakage is generally obtained for the input value with Hamming weight w equal to 0 (i.e., the input 0000). More quantitatively, the number of circuits that were found to have their maximum (minimum) leakage at a given input value is plotted in Figure 4.12 (Figure 4.13). Apparently, most occurrences have maximum leakage for the input $(0)_{10} = (0000)_2$, as occurs in the case without intra-die process variations. There are also other small peaks around the inputs $(8)_{10}$, $(4)_{10}$, $(2)_{10}$, and $(1)_{10}$, as they differ from $(0)_{10}$ by only one bit, hence the corresponding leakage can sometimes be higher than the case with input $(0)_{10}$ due to variations, although nominally it is lower (of course, this is less likely for inputs that differ by a greater number of bits). Analogous considerations hold for the minimum leakage: for example, leakage is minimum for the input $(15)_{10} = (1111)_2$ as in the case without variations, and there are other small peaks at inputs differing by only one bit (i.e., $(14)_{10}$, $(13)_{10}$, $(11)_{10}$, and $(7)_{10}$). According to these results, variations have a rather limited impact on the leakage dependence on the input, and hence on the LPA attack outcome, which depends on the ability to discriminate high and low values of leakage. In other words, LPA attacks are expected to be still quite effective even in the presence of intra-die variations. As a further elaboration, the leakage current is plotted in Figure 4.14 versus the Hamming weight of the S-Box inputs . The average (with respect to Monte Carlo iterations) leakage current trend is plotted in Figure 4.15.

It is useful to observe that the average curve in Figure 4.15 exhibits an approximately linear trend as in (4.2), although this equation was derived analytically under the assumption of bit sliced circuits. In other words, (4.2) can be a good leakage model also for random logic blocks (i.e., non bit sliced). Actually, analogous considerations were made in the past for traditional DPA attacks, which were extended to random logic blocks in a very similar way. In

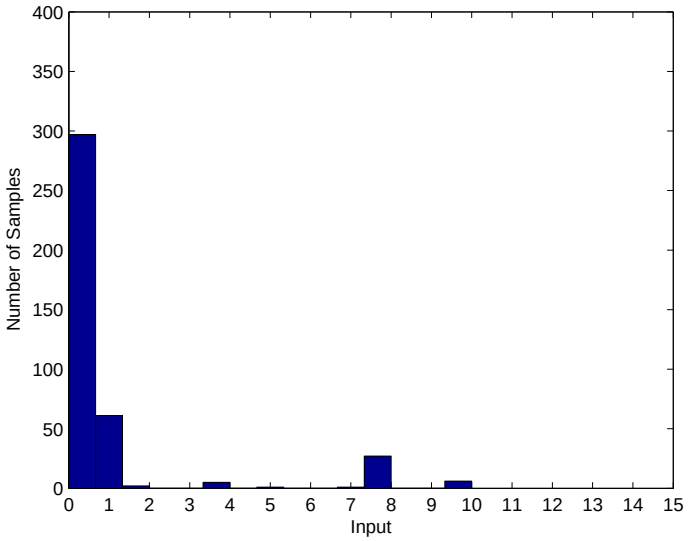


Figure 4.12: Histogram distribution of highest leakage current versus the S-Box input (standard CMOS logic style).

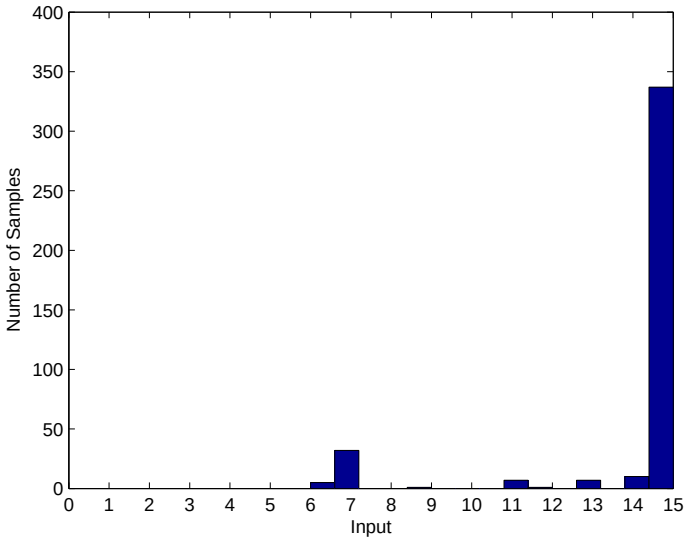


Figure 4.13: Histogram distribution of lowest leakage currents versus the S-Box input (standard CMOS logic style).

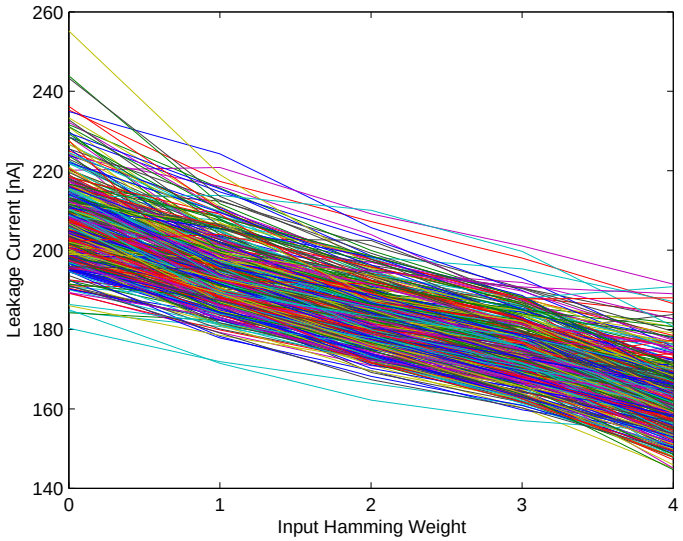


Figure 4.14: S-Box leakage current trend versus input Hamming weight for standard CMOS logic style.

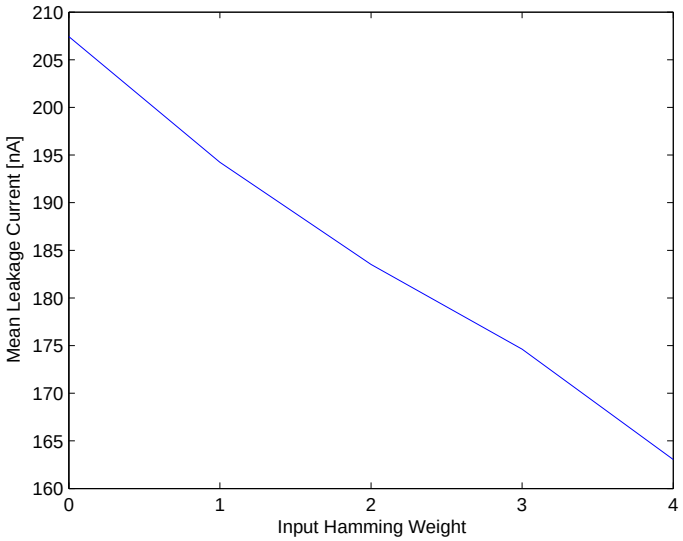


Figure 4.15: S-Box leakage current trend versus input Hamming weight for standard CMOS logic style (average over 400 Monte Carlo iterations).

this respect, LPA and DPA attacks are similar, as the linear model of power consumption in (4.2) generally applies. Analysis of Figure 4.14 also shows that the leakage dependence on the input Hamming weight in (4.2) is rather insensitive to process variations. Indeed, other than keeping approximately a linear trend, the extreme experience a rather small change due to process variations. More specifically, from Figure 4.12 the left (right) extremum changes by 14% (19%) compared to the average value. Again this intuitively confirms that variations are expected to have a rather limited impact on LPA attacks.

To better understand the impact of intra-die process variations in practical cases, let us analyze the statistics of the outcome of an LPA attack that was repeated on 400 sample circuits implementing the simple crypto core depicted in Figure 4.16. In this figure, the S-Box is the same used for the previous considerations. These attacks were performed according to the five-step procedure described in Section 4.3, and setting the secret key of the crypto core to $(5)_{10} = (0101)_2$. For each of the 400 sample circuits and for each key hypothesis, the correlation coefficient ρ between the measured leakages and the Hamming weight vector $\rho(I_{leak,i}, H_{ij})$ was evaluated. According to the block diagram in Figure 4.16, for each sample the Hamming weight H_{ij} was evaluated on the signal that results from the XOR between the i-th input and the j-th conjectured key. Observe that only the first half of the possible keys (i.e., $j = 0 \ldots 7$ in this case) has to actually be considered, as $\rho(I_{leak,i}, H_{ij})$ has the same magnitude for symmetric keys because of the symmetry of the XOR function (i.e., $|\rho(I_{leak,i}, H_{ij_1})| = |\rho(I_{leak,i}, H_{ij_2})|$ for $j_1 + j_2 = 15$ in this case).

The resulting values of $\rho(I_{leak,i}, H_{ij})$ for the 400 sample circuits are plotted in Figure 4.17 versus the conjectured key, considering only the first half of the possible keys, as discussed above. Figure 4.17 shows the impact of intra-die process variations on the outcome of an LPA attack. From Figure 4.17, it is apparent that the correlation coefficient for the correct key can be lower than that for other key guesses. However, this does not mean that the LPA attack necessarily fails, as failure is observed only if the correlation coefficient associated with the correct key is not the highest for a given sample circuit. For example, the correlation coefficient obtained for the sample #135 (highlighted in Figure 4.17) is such that the maximum value is obtained for the correct key, which means the attack is successful for this sample. Hence, in general the overlap in Figure 4.17 between the data for the correct and incorrect keys is not an issue, and does not convey information on the LPA effectiveness.

According to the above considerations, to better understand the impact of variations on the effectiveness of LPA, we have to check if the correct key is associated with the highest correlation coefficient for each sample circuit, instead of looking at all samples at the same time. To this aim, we plotted the key that leads to the highest correlation coefficient versus the considered sample in Figure 4.18. From this figure, it is apparent that in most cases the LPA attack was successful (i.e., the highest correlation coefficient is obtained for the correct key $(5)_{10}$), with only very few exceptions. More specifically, the LPA attack was successful for 375 sample circuits out of 400, i.e. in 94% of the

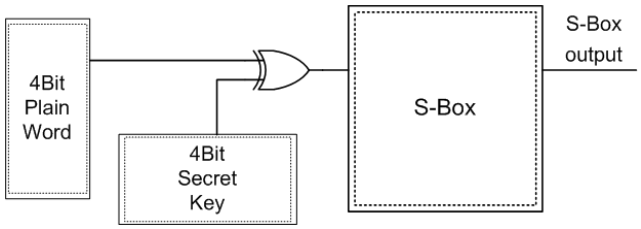


Figure 4.16: Crypto core used to perform the LPA attack under process variations.

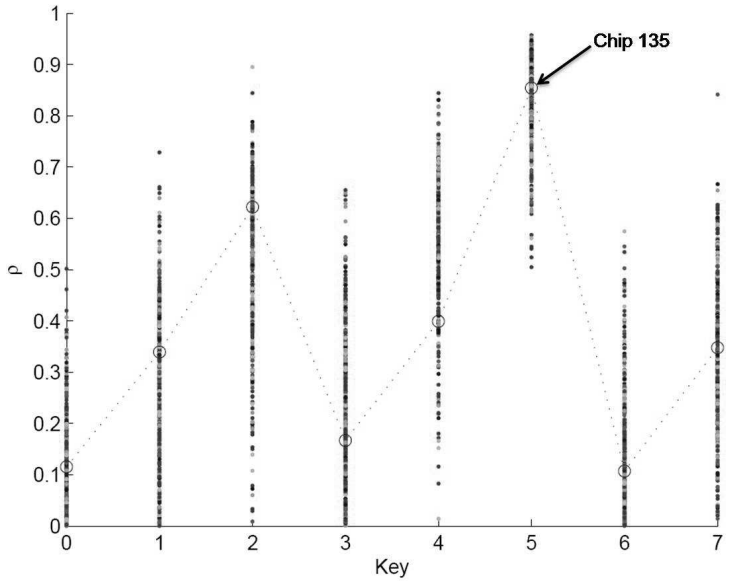


Figure 4.17: Correlation coefficients versus the key guess for all the considered sample circuits (standard CMOS logic style).

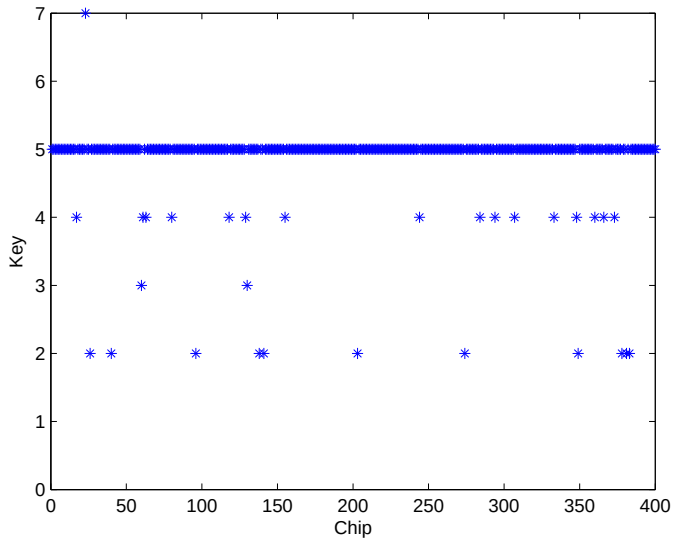


Figure 4.18: Plot of the key leading to the highest correlation coefficient versus the considered sample circuit (standard CMOS logic style).

cases. Furthermore, the probability of success of LPA attack rapidly increases if we broaden the search to the keys leading to the two highest values of the correlation coefficients . In this case, the LPA attack is successful for 394 out of 400 chips, hence in the 99% of the cases. This shows that LPA attacks can be very effective in recovering the secret key of standard CMOS cryptographic chips, even in the presence of intra-die process variations.

4.8 Analysis of LPA in presence of DPA resistant logic styles under process variations

The analysis reported in Section 4.7 was extended to DPA resistant logic styles.

In Section 1.4 has been discussed that DPA can be counteracted through a number of countermeasures at various levels of abstraction. At the transistor level, the countermeasures are based on the adoption of logic styles whose power consumption is constant or independent of the processed data, which is typically obtained through differential signaling and precharged logic (see Chapter 5 for details). Wave Dynamic Differential Logic (WDDL) is an example of a state-of-the-art DPA-resistant logic style that can be implemented with a standard CMOS cell library [102]. On the other hand, more resistant logic styles require the development of a custom cell library as in the case of Sense Amplifier Based Logic (SABL). Unfortunately SABL requires a perfectly balanced capacitive load at the two nodes of each differential pair, which is not trivial to obtain

during synthesis, placement and routing [96]. Three-phase Dual-rail Pre-charge Logic (TDPL), which will be shown and discussed in Section 5.3, overcomes these problems.

WDDL and TDPL were chosen as representative state-of-the-art logic styles based on standard and custom cell libraries, respectively. The S-Box and the simple crypto core in Figure 4.16 were designed using the reference 65nm CMOS technology and using an automated design flow. A standard cell library available from the foundry was used to synthesize the WDDL blocks, whereas a custom cell library was adopted to synthesize the TDPL blocks. For each circuit, 400 different sample circuits were generated through Monte Carlo simulations.

4.8.1 Analysis of results for the WDDL logic style

The analysis presented in Section 4.7 has been repeated for the S-Box and crypto core in the WDDL logic styles. In particular, the resulting statistical distribution of the highest (lowest) leakage current of the S-Box is plotted in Figure 4.19 (Figure 4.20) for the WDDL design. From Figure 4.19, it is apparent that most sample circuits have maximum leakage for the input $(0)_{10} = (0000)_2$, as occurs in the case without intra-die process variations. However, there are more peaks distributed over the other input values, compared with the standard CMOS block (see Figures 4.12 and 4.13). This means that the process variations have a more significant effect than in the standard CMOS version, although the input value $(0000)_2$ still maximizes leakage in typical cases. Figure 4.20 leads to similar conclusions, although the peak of occurrences is less distinguishable. Accordingly, process variations have a more significant impact on the leakage dependence on the input in WDDL circuits, and hence on the LPA attack outcome. This means that LPA attacks are expected to be less effective than in standard CMOS circuits, but still successful in typical cases. This agrees with the trend of the leakage current versus the Hamming weight of the S-Box inputs for each sample circuit in Figure 4.21 and Figure 4.22 (Figure 4.22 is the average on the 400 Monte Carlo iterations of the leakage current trend for WDDL circuit reported in Figure 4.21). From Figure 4.22, a roughly linear trend is still observed, but the curves are significantly flatter than in Figure 4.15. This means that the leakage dependence on the input in WDDL circuits is significantly weaker than that of standard CMOS logic. As a consequence, leakage reveals less information about the key, and hence the LPA attack is expected to have a lower probability of success, compared to standard CMOS.

The above considerations on the WDDL logic style are confirmed by the results obtained for the crypto core in Figure 4.16 designed by the WDDL logic style. More specifically, the resulting correlation coefficient for the 400 sample circuits are plotted in Figure 4.23 versus the conjectured key. From this figure, the correlation coefficient values computed using the right key $(0101)_2$ under process variations are completely overlapped with the correlation coefficients computed using wrong keys. As explained in Section 4.7, this does not necessarily mean that the LPA attack always fails. As an example in the Figure 4.23, the sample WDDL circuit #180 still has the maximum leakage under the correct

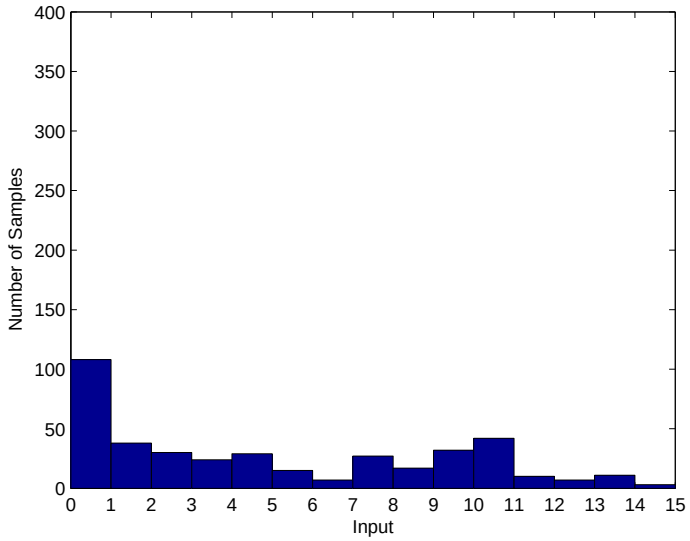


Figure 4.19: Histogram distribution of highest leakage current versus the S-Box input (WDDL logic style).

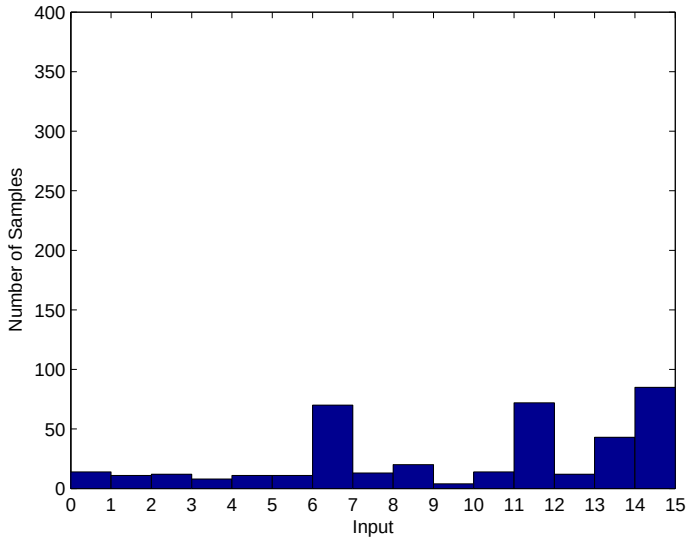


Figure 4.20: Histogram distribution of lowest leakage currents versus the S-Box input (WDDL logic style).

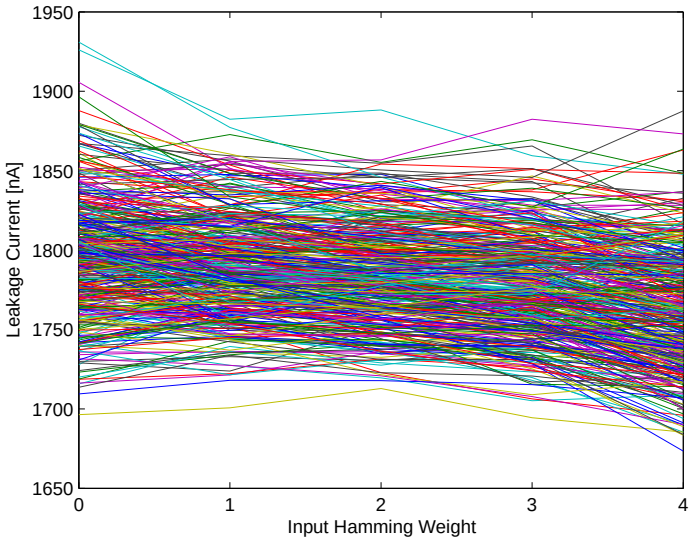


Figure 4.21: S-Box leakage current trend versus input Hamming weight (WDDL logic style).

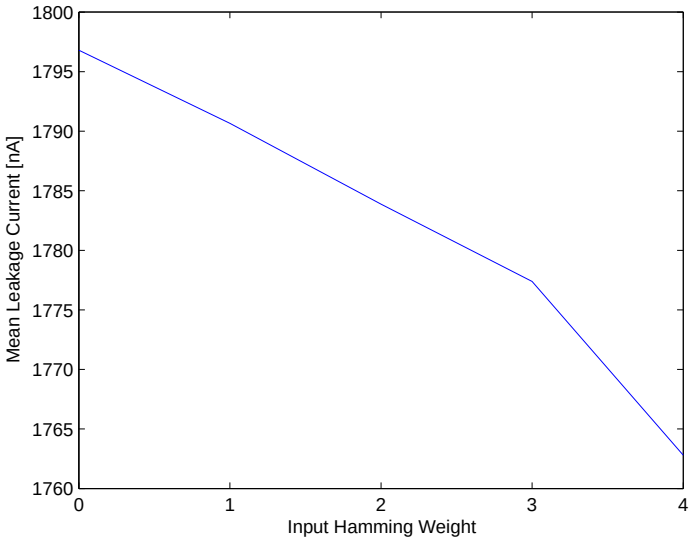


Figure 4.22: S-Box leakage current trend versus input Hamming weight for standard WDDL logic style (average over 400 Monte Carlo iterations).

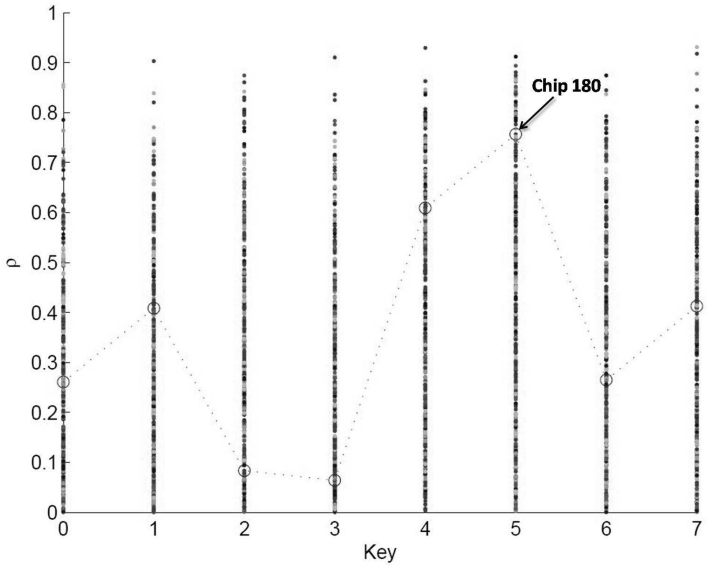


Figure 4.23: Correlation coefficients versus the key guess for all the considered sample circuits (WDDL logic style).

key, i.e. the LPA attack is successful for this circuit. However, the probability of a successful LPA attack across all sample circuits will be certainly lower than in the case of standard CMOS logic case.

More specifically, from Figure 4.24 we achieve a successful attack in 85 sample circuits out of the 400 considered circuits, i.e. LPA attack works in 21% of the cases. On the other hand, the probability of success of LPA attack increases if we broaden the search to the keys leading to the two highest values of the correlation coefficients. In this case, the LPA attack is successful for 161 sample circuits, i.e. in 40% of the cases. From the above considerations, LPA attacks can still be successful in a significant portion of a set of real cryptographic chips, which agrees with the qualitative considerations made previously. In other words, although WDDL circuits have a rather high resistance to DPA attacks, they are quite vulnerable to LPA attacks. Obviously, this is not acceptable in applications requiring a high level of security, like those where DPA-resistant logic styles are adopted. In those applications, LPA attacks must be counteracted as well. Hence, a significant research effort will be required in the future to deal with vulnerability to LPA attacks, as traditional techniques to counteract DPA attacks are quite ineffective.

4.8.2 Analysis of results for the TDPL logic style

The resulting statistical distribution of the highest (lowest) leakage current of the S-Box is plotted in Figure 4.25 (Figure 4.26) for the TDPL design. From

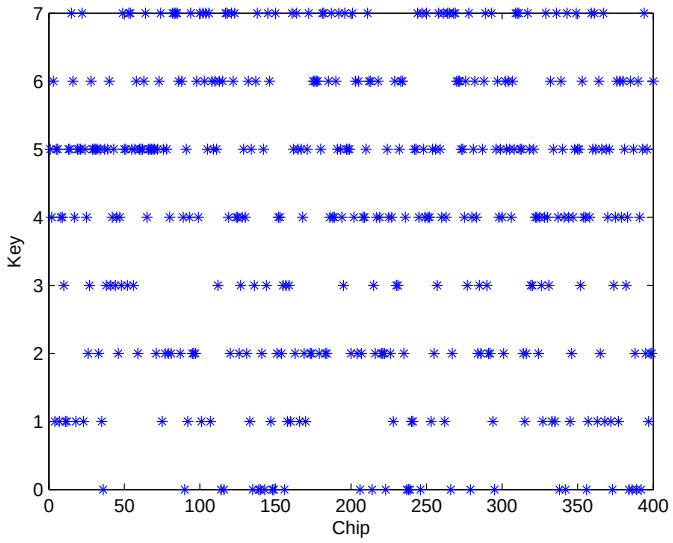


Figure 4.24: Plot of the key leading to the highest correlation coefficient versus the considered sample circuit (WDDL logic style).

these figures, it is no longer possible to distinguish the peak associated with the correct key from those with incorrect key. Hence, TDPL is expected to be more robust against LPA attacks, compared with WDDL and standard CMOS logic styles. This agrees with the trend of leakage versus the input Hamming weight in Figure 4.27 and Figure 4.28 (Figure 4.28 is the average on the 400 Monte Carlo iterations of the leakage current trend for TDPL circuit reported in Figure 4.27), which is no longer monotonic for a significant fraction of sample circuits.

Hence, the model in (4.2) can be rather inaccurate when estimating leakage, which intuitively leads to a degradation in the probability of success of LPA attacks.

This is also consistent with the plot of the correlation coefficient for the 400 sample circuits versus the conjectured key in Figure 4.29. Indeed, in this figure correlation coefficient values computed using the correct key $(0101)_2$ under process variations are again completely overlapped with the correlation coefficients computed using wrong keys. Nevertheless, LPA attacks can still be successful for a portion of the sample circuits, as shown in the example of sample #217 in Figure 4.29. Now, let us analyze the plot of the key leading to the highest correlation coefficient versus the considered sample circuit in Figure 4.30 for the TDPL logic style. From this figure, the attack is successful in 94 sample circuits (i.e., in 23% of the cases). If we broaden the search to the keys leading to the two highest values of the correlation coefficients, the LPA attack is successful for 150 sample circuits, i.e. in 38% of the cases.

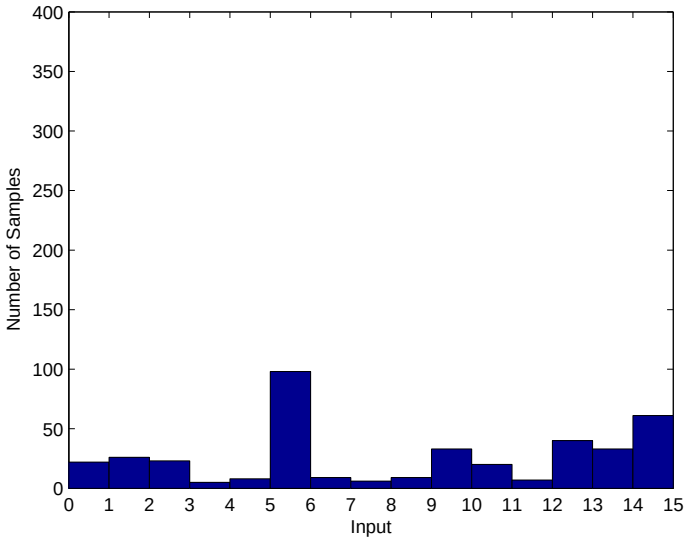


Figure 4.25: Histogram distribution of highest leakage current versus the S-Box input (TDPL logic style).

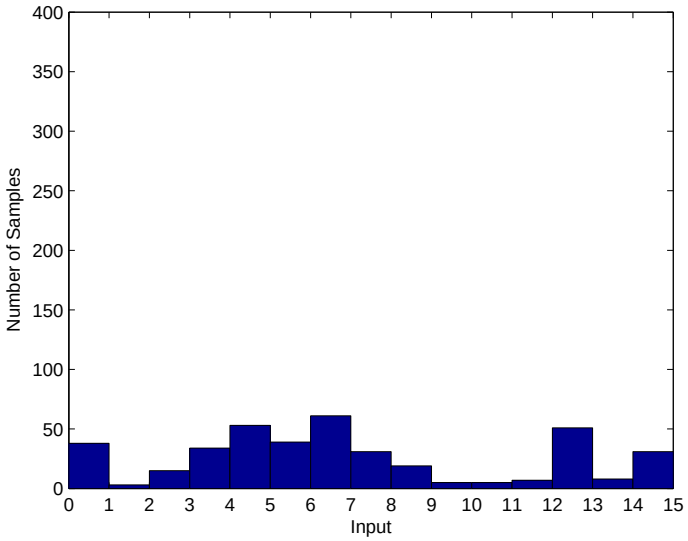


Figure 4.26: Histogram distribution of lowest leakage currents versus the S-Box input (TDPL logic style).

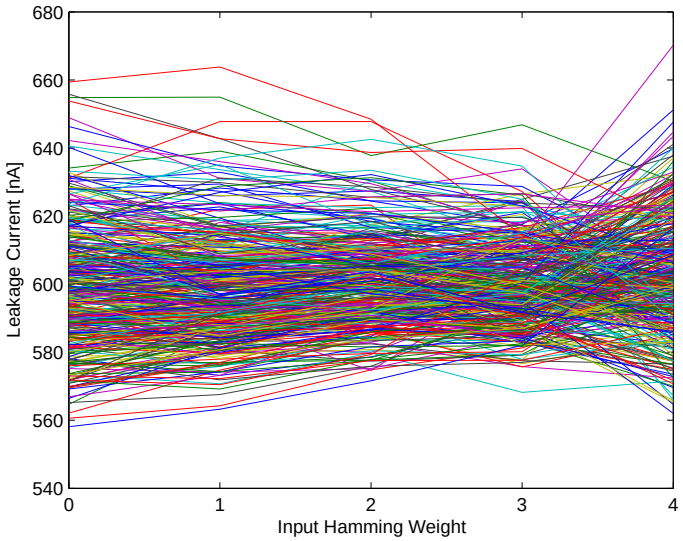


Figure 4.27: S-Box leakage current trend versus input Hamming weight (TDPL logic style).

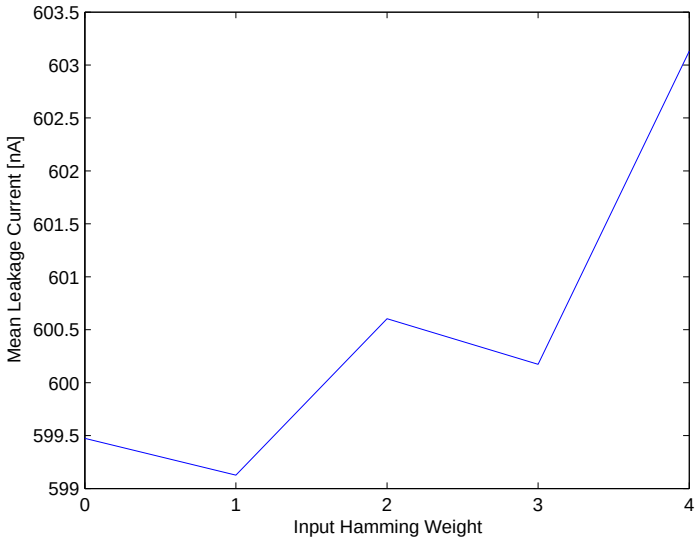


Figure 4.28: S-Box leakage current trend versus input Hamming weight for standard TDPL logic style (average over 400 Monte Carlo iterations).

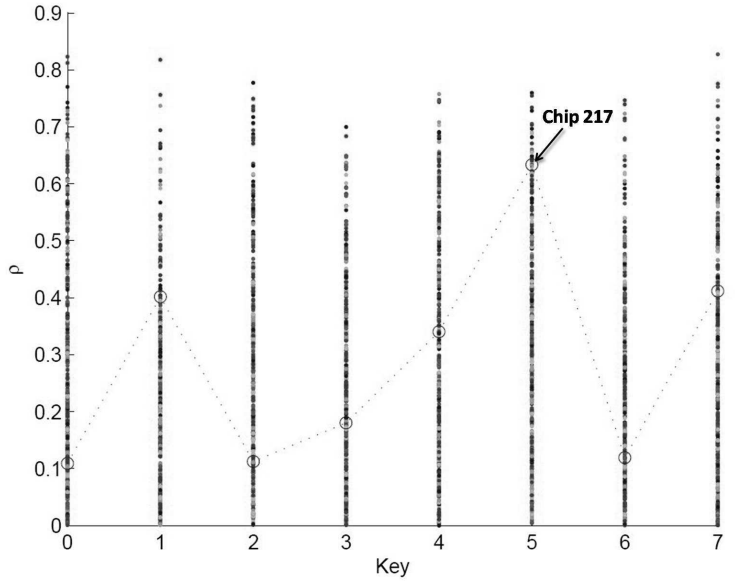


Figure 4.29: Correlation coefficients versus the key guess for all the considered sample circuits (TDPL logic style).

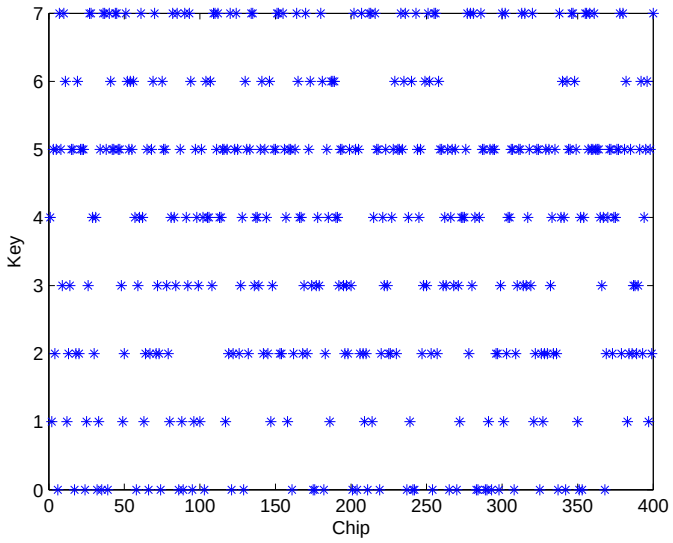


Figure 4.30: Plot of the key leading to the highest correlation coefficient versus the considered sample circuit (TDPL logic style).

Table 4.7: Comparison of the LPA attack outcome for different logic styles.

Logic Style	attack based on key with max. leakage	attack based on 2 keys with max. leakage
std. CMOS	375 (94%)	394 (99%)
WDDL	85 (21%)	161 (40%)
TDPL	94 (24%)	150 (38%)

Finally, as was observed for WDDL, the TDPL logic style is vulnerable to LPA attacks in a significant portion of real chips, which is clearly unacceptable in the applications where TDPL is used. This confirms that the vulnerability to LPA attacks is a serious concern in cryptographic chips requiring a high level of security, even in the presence of countermeasures against DPA. In other words, once countermeasures are introduced for DPA attacks, vulnerability to LPA attacks becomes the weakness that severely limits the security against power analysis attacks.

Table 4.7 summarizes the results of the presented LPA attacks for the three different logic styles (CMOS, WDDL, TDPL) showing the successful cases when the attack is performed considering only the key corresponding to the highest correlation coefficient and when the attack is based on the two keys with the highest correlation coefficient.

4.9 Conclusions

Leakage Power Analysis attack has been presented for the first time in the literature.

Starting from a model which describes a relationship between the leakage current and the Hamming weight of the input word, a simple attack procedure based on the correlation coefficient evaluation has been introduced for the first time.

In a first time, various LPA attacks to simple (bit sliced) circuits have been performed in simulation and experimentally. Results confirm the validity of the underlying assumption, and the effectiveness of this kind of attacks. Experimental issues have been discussed, including the effect of temperature and process variations, which strongly affect the leakage current of sub-100 nm VLSI circuits. In particular, it is shown that the temperature of the cryptographic chip must be kept constant during the LPA attack, in order to recognize the secret key.

In a second time analysis has shown that the Hamming weight as leakage power model is not restricted to bit sliced circuits, but can also be used for combinational circuits (e.g., S-Boxes) and more complex circuits (e.g., crypto cores). In regard to the impact of process variations, Monte Carlo simulations have shown that intra-die variations can influence the outcome of LPA attacks.

In particular, the analysis of a simple crypto core in standard CMOS logic allowed to successfully infer the secret key on 94% of sample circuits.

The effectiveness of LPA has been studied also attacking crypto cores designed with DPA resistant logic families (WDDL and TDPL) under process variations. In particular, results demonstrate that, even if transistor-level countermeasures developed for DPA have some beneficial effects in preventing LPA attacks, the success rate for LPA attacks is intolerably high (many orders of magnitude greater than levels of security that are commonly required).

A theoretical analysis of LPA attacks has been developed to predict the result of practical attacks. In particular, a closed-form expression of the correlation coefficient between the estimated and measured leakage is provided as a function of the parameters related to the attack. The model is shown to agree well with experimental and simulation results.

Although this work clarifies various aspects of LPA attacks, it has to be considered a starting point for further investigation, as many issues are not fully understood. For example, the effect of temperature on attacks is not clear, whereas simple criteria to properly set the chip temperature to maximize the attack effectiveness would be desirable, in order to test cryptographic circuits under worst conditions. For the same reason, criteria to build measurement setups for fast and effective attacks are required. Moreover, experimental attacks on more complex circuits should be performed to better understand the robustness against LPA attacks versus their complexity. Furthermore, criteria to consciously select the number of measurements to have a reasonably accurate estimation of the correlation coefficient should be identified. Finally, much work will be required in the future to devise countermeasures that are able to improve the robustness of cryptographic circuits against LPA attacks.

Chapter 5

Transistor Level Countermeasures

5.1 Logic families for cryptographic applications

Side channel attacks can disclose confidential data (i.e. cryptographic keys and user PIN's) looking at the information leaked by the hardware implementation of cryptographic algorithms. In particular, DPA exploits the fact that digital circuits feature a power consumption profile dependent on the processed data: even small correlations between the circuit switching activity and the key material can be revealed by measuring the current consumption over repeated computations (see Chapter 2 for details).

Since the introduction of DPA, several hardware countermeasures have been proposed in the technical literature at different logic levels: system-level, gate-level and transistor-level countermeasures. In Section 1.4.2 a short description of different solutions adopted in each level is reported, whereas, in this Chapter, transistor-level countermeasures only are studied.

The transistor-level approach is based on the adoption of a logic style whose power consumption is constant or independent of the processed data.

As shown in Chapter 2, standard-CMOS logic has a power consumption which strongly depends on processed data. To weaken the relationship between processed data and power consumption two different solutions has been introduced.

A first adopted solution is the dynamic (or precharge) logic, where each signal has an evaluation phase in which the signal assumes the logic value depending on the realized logic function and a precharge phase in which each signal goes to V_{DD}. An additional clock signal as input to the gate defines the different phases. To understand the advantage of this solution in cryptographic applications, let us compare two 2NOR gates designed in CMOS logic and dynamic logic respectively. If the inputs do not change or if they change, for example, from 01 to 10, the output of the CMOS 2NOR does not change remaining stable at

V_{SS} and this results in a constant power consumption revealing the behavior of the logic; in the same circumstances, the output of 2NOR gate designed in dynamic logic goes to V_{SS} during the evaluation phase (like in CMOS design) but everytime the gate enters precharge phase, it sets the output value to V_{DD}, producing a power consumption. The first drawback of this solution is the added signal "clock" to each gate. From cryptographic point of view, when the gate output in evaluation phase is high, there is no change with respect to the precharge phase (this signal is high in both cases) and, therefore, the power consumption profile reveals this state.

Another adopted solution is the dual-rail logic style in which each signal X is replaced by the pair of signals X and its inverse $\overline{X} = not(X)$. In this logic style, a 2NOR gate has four inputs (A, $not(A)$, B, $not(B)$) and two outputs (NOR and $not(NOR) = OR$). To understand the advantage of this solution in cryptographic applications, just think that each time one output goes low the other goes high producing a power consumption. The first drawback of this solution is that each gate needs additional area to duplicate the logic. From a security point of view, when the inputs do not change, the output does not change too, as in CMOS.

To capture the strengths of both solutions, the dual-rail pre-charge (DRP) logic style is often used to counteract DPA. In a dual-rail pre-charge (DRP) logic style (e.g. Sense Amplifier Based Logic (SABL) [96], Wave Dynamic Differential Logic (WDDL) [102], Dual-Spacer DRP [90]), signals are spatially encoded as two complementary wires and power consumption is constant under the assumption that the differential outputs of each gate drive the same capacitive load. Dual-rail pre-charge logics are not affected by glitches but building two balanced wires requires a full-custom approach thus increasing design and maintenance costs. A DRP logic style can be implemented by using standard CMOS gates, as in the case of the Wave Dynamic Differential Logic (WDDL) [102], or by using a custom design in which each gate is transistor level designed, as in the case of the Sense Amplifier Based Logic (SABL) [96].

Semi-custom design flows supporting differential logic families have been proposed in the technical literature. For instance, a technique for the automatic routing of balanced complementary lines has been introduced in [101]. Even if an automatic place and route could reduce design time and increase the portability, the proposed balanced routing technique does not take into account the dependence of the capacitive load on a line on the logic state of the adjacent wires and, furthermore, introduces additional constraints for the routing tool thus limiting its efficiency and, likely, causing an area overhead especially if only few metal layers are available for the inter-cell routing. In addition, in a modern deep sub-micron technology, local process gradients cannot be neglected and they are the limiting factor for the load matching accuracy.

Another technique proposed in [77, 78] is based on a masked dual-rail precharge logic style (MDPL) where, due to the random masking at the gate level, power consumption is randomized. Moreover, since MDPL is a dual-rail precharge logic, glitches are avoided and, at the same time, the complementary wires do not need to be balanced thus removing the main drawback of the dual-

rail circuits. As reported in [76], a first implementation of MDPL showed a DPA leakage due to the early propagation of the input data with respect to the masking ones. The authors propose an improved implementation (iMDPL) where SR-latches are used to resynchronize the inputs thus forcing a combinatorial cell to evaluate only when all the inputs are in an valid differential state. The penalty with respect MDPL is a factor 3 and 1.5 in terms of area and power consumption respectively.

The effectiveness of a DPA-resistant logic style is normally quantified by means of two figures of merit: the Normalized Energy Deviation (NED) and the Normalized Standard Deviation (NSD). The energy per cycle is computed as

$$E = V_{DD} \cdot I_{AV} \cdot T_{CK} = V_{DD} \cdot \int_0^T I_{DD}(t)dt \tag{5.1}$$

where $I_{DD}(t)$ is the current drawn from the supply.

Starting from (5.1), the normalized energy deviation (NED) estimates the gap between the maximum and the minimum value of the energy consumption (normalized to the maximum value). The lower the NED, smaller are the differences of energy consumption among the different processed data and therefore better is the effectiveness of the logic in counteracting power analysis. The NED is defined as:

$$NED = \frac{\max(E) - \min(E)}{\max(E)}. \tag{5.2}$$

The normalized standard deviation (NSD) estimates how much the different values of power consumption, for different processed data, deviate from the average consumption. The lower the NSD, smaller are the distances of different energy consumptions from the mean energy and therefore better is the effectiveness of logic to counteract power analysis:

$$NSD = \frac{\sigma_E}{\overline{E}}. \tag{5.3}$$

where σ_E and $\overline{E}$ is the standard deviation and the average of E respectively.

In this Chapter three different logic families to counteract DPA attacks will be presented. Starting from the standard CMOS logic gates, it will be shown that each proposed solution is an improvement with respects to the previous one concerning security and design criteria.

The first proposed solution (Section 5.2) is a novel dynamic differential logic style (*3sDDL*) which relies on the use of signals with three possible states: logic '1' (V_{DD} voltage), logic '0' (GND voltage) and 3rd state ($V_{DD}/2$ voltage) following the idea proposed in [6].

A DRP logic gate presents a data independent power consumption under the ideal hypothesis the two output loads are perfectly balanced. In a real circuit the output loads are unbalanced by the mismatch induced by the different routing of two outputs and by the input capacitance mismatch of the following gate.

This means all logic families for cryptographic applications present this second order relationship between processed data and power consumption.

The second proposed solution is a logic insensitive to unbalanced routing capacitances (Section 5.3). It is obtained by introducing a three-phase dual-rail pre-charge logic (*TDPL*) with an additional discharge phase where the output which is still high after the evaluation phase is discharged as well. Since both outputs are precharged to V_{DD} and discharged to V_{SS}, a TDPL gate shows a constant energy consumption over its operating cycle. The main drawback of this solution is the additional area for the routing of the three control signals.

The last solution consists of a completely novel approach to the design of a secure logic family which is based on a standard two-phase operation (precharge/evaluation) while being at the same time insensitive to unbalanced load conditions (Section 5.4). In this proposed logic family, called *DDPL*, the information is represented in the time domain by forcing a positive (logic-1) or negative (logic-0) relative delay between the differential lines. Therefore, as in TDPL, both outputs are precharged and discharged inside the operating cycles but, due to the chosen data encoding, a single control signal is sufficient as in a standard dual-rail logic.

For each logic style, implementation details and simulation results on a basic set of logic gates are first reported. Simple and more complex case studies are then discussed and extensive comparisons with others logic styles which are the state of the art are carried out.

5.2 The 3-state Differential Dynamic Logic Family

In a general DRP style each signal is replaced by a pair of wires: one wire usually carries the non-inverted signal a while the other wire carries the inverted (complementary) signal $\overline{a}$. Moreover all logic signals alternate between a precharge value (in this case the signals are in the so called precharge phase) and the logic values that are processed (in this case the signals are in the so called evaluation phase). The precharge value is either V_{DD} (logic 1) or Ground (logic 0). The sequence of the precharge phase and the evaluation phase is typically controlled by the clock signal. In most cases, the logic value of the clock signal defines the phase in which a circuit currently is.

The proposed logic style is a novel dual rail CMOS called 3-state Differential Dynamic Logic (3sDDL). The logic style can be regarded as a differential and dynamic logic style with precharge value equal to $V_{DD}/2$.

The schematic of a generic 3sDDL gate is reported in Figure 5.1 (capacitors represent output loads). The connections of the gate to V_{DD} and ground nodes are controlled by clocked MOS devices (note that differently from normal dynamic logic the pull-up device is clocked by $notCK$); a pass-transistor connects the two output nodes. In a 3sDDL gate, the operations proceeds through two phases.

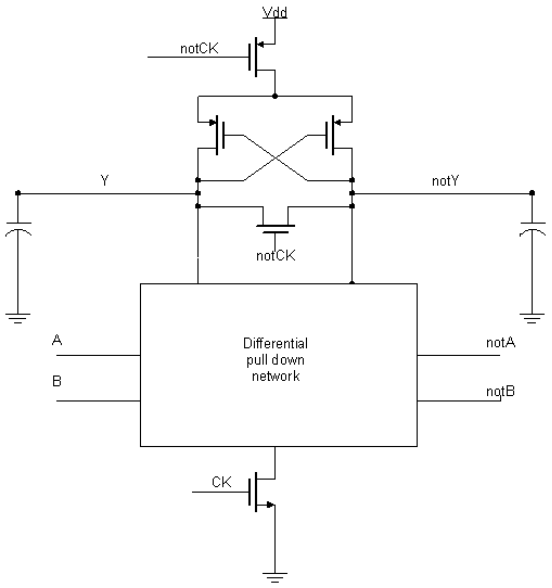


Figure 5.1: Schematic of the generic 3sDDL gate.

During the Precharge phase ($CK = 0V$, $notCK = V_{DD}$), the pass-transistor shorts the two output nodes Y and $notY$; since one output node is initially at $0V$ and the other is at V_{DD} (due a previous evaluation phase), at the end of the precharge phase both output nodes are at $^{V_{DD}}/_{2}$ by charge redistribution, assuming perfect matching between the output capacitances.

During the Evaluation phase ($CK = V_{DD}$, $notCK = 0V$), the pass-transistor is an open circuit and the logic gate is connected to V_{DD} and ground; in this phase the differential pull down network evaluates its outputs (charging one of the output capacitances from $^{V_{DD}}/_{2}$ to V_{DD} and discharging the other from $^{V_{DD}}/_{2}$ to $0V$).

The above operating mode presents some relevant features with respect to a normal DRP used in literature:

- the voltage swing required in the transition from precharge to evaluation is halved with respect to standard dynamic logic: this potentially allows faster operation and lower power consumption;
- each data line is loaded only by a single NMOS input capacitance;
- the third state in the logic signals can be easily detected by suitable circuits, which allows us obtaining a clock signal from data signals: this could avoid clock routing to each gate, by locally generating the clock at each gate;
- the use of cross-coupled pull-up transistors guarantees static operation in

the evaluation phase allowing a direct connection of 3sDDL gates (differently from dynamic gates).

Obviously a single to dual-rail netlist conversion is required and interconnection capacitances at Y and $notY$ outputs have to be balanced for best performance (as in all DRP logic styles known in literature).

The power analysis of the proposed logic can be easily carried out: at each clock cycle one of the two outputs is charged from $V_{DD}/2$ to V_{DD} by the pull-up of the gate, while the other is discharged from $V_{DD}/2$ to zero by the differential pull-down. The total charge required to the supply is given by $C_Y \cdot V_{DD}/2$ or $C_{notY} \cdot V_{DD}/2$, where C_Y is the capacitance at node Y and C_{notY} is the capacitance at node $notY$. In the precharge phase, no current is drawn from the supply (since charge redistribution is used). As a result, assuming $C_Y = C_{notY}$, at each clock cycle always the same current is drawn from the power supply.

Some design guidelines to optimize 3sDDL cell project from DPA resistance point of view should be followed: a good balancing of dual rail wires and of internal nodes parasitic capacitance; design of a differential pull down network layout which is perfectly symmetrical from the viewpoint of series or parallel connected devices and so that the number of series-connected transistors from output to ground does not depend on the input pattern. It has to be noted that a technique which allows the routing of a dual rail circuit by using standard place and route tools has been recently proposed [99, 104]. This technique guarantees a matching of the interconnection capacitances of two differential lines within a few percent.

5.2.1 3sDDL cells library: design and simulations

The technology chosen for the present study is a $0.13\mu m$ CMOS. Circuit simulations have been carried out by using the design kit of the HCMOS9 process from STMicroelectronics in CADENCE IC 4.4.6 environment. A $1.2V$ supply has been assumed. All n-channel and p-channel devices have been sized for minimum area.

The schematic of a 3sDDL XOR gate is reported in Figure 5.2. It is evident that the differential pull down guarantees the same number of series connected devices between each output and ground for all input patterns. Furthermore, due to the symmetry of the logic function, a good balance of parasitic capacitance of internal nodes has been easily obtained. The power consumption of the XOR gate has been studied for a 4 3sDDL XOR gate fan-out and taking into account an interconnection capacitance of $12.5fF$ both at Y and $notY$ outputs (estimated from the layout). A clock period $T_{CK} = 10ns$ has been assumed for all tests.

For each of the four output transitions the energy per cycle has been computed as in (5.1). As figures of merit for the logic gates with respect to DPA attacks, we have used the Normalized Energy Deviation (NED) and the Normalized Standard Deviation (NSD) as calculated in (5.2) and (5.3) respectively.

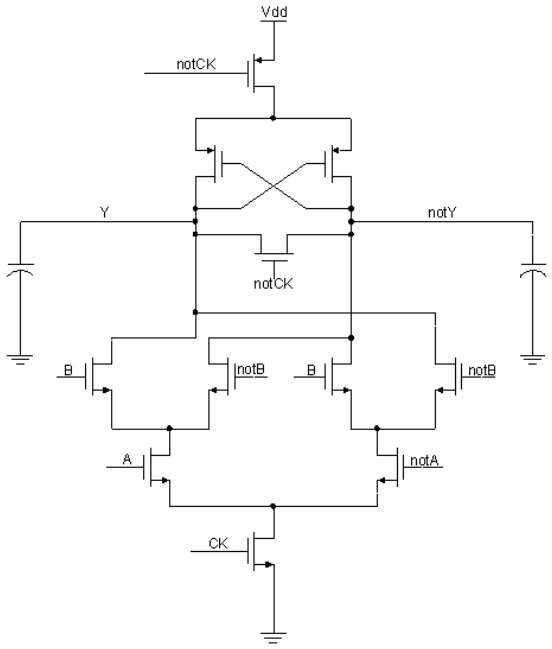


Figure 5.2: XOR-XNOR gate in 3sDDL.

Table 5.1: I_{AV} with perfectly balanced interconnection capacitances (XOR gate).

Transition	I_{AV} (μA)	*NED*
$0 \rightarrow 0$	1.020	0%
$0 \rightarrow 1$	1.020	
$1 \rightarrow 0$	1.020	
$1 \rightarrow 1$	1.020	

The average current I_{AV} has been measured for each of the four output transitions; results are summarized in Table 5.1. In order to take into account interconnection capacitance mismatch, the average current I_{AV} has been measured for a 10% mismatch in the interconnection capacitances at the two outputs; the results are summarized in Table 5.2.

A further analysis of the power consumption of the XOR-XNOR gate has been carried out by simulating random input sequences of length 500 clock cycles for the input signals as in [100]. In Figure 5.3 the output of the gate and the current drawn from the supply I_{DD} is reported. The average current per cycle I_{AV} has been computed for each of the 500 bits. A standard static CMOS (sCMOS) XOR gate has been tested in the same conditions as for the 3sDDL gate. A summary of the comparison between sCMOS and 3sDDL XOR gate is reported in Table 5.3, where E_{AV} is the Energy per cycle averaged on

Table 5.2: I_{AV} for a 10% interconnection capacitance mismatch (XOR gate).

Transition	I_{AV} (μA)	*NED*
$0 \rightarrow 0$	1.026	14%
$0 \rightarrow 1$	1.102	
$1 \rightarrow 0$	0.942	
$1 \rightarrow 1$	1.012	

Table 5.3: Comparison between sCMOS and 3sDDL XOR gates.

	sCMOS	3sDDL
NED	95%	0.02%
NSD	124%	0.006%
E_{AV}	$17\,fJ$	$12\,fJ$
t_p/t_{p0}	1	0.883

the 500 clock cycles, t_p is the propagation delay of the 3sDDL gate and t_{p0} the propagation delay of the sCMOS gate.

We considered different implementations of the NAND gate in our work. The simpler topology for the NAND gate in 3sDDL presents the asymmetry of the differential pull down network which provides two series connected devices for the NAND output and two parallel connected devices for the and output. An analysis of the power consumption of this topology of a four AND-NAND gate fan out has been carried out by using random bit sequences for the input signals. The NED achieved by using this topology is about 14%. In order to obtain better results in terms of NED new 3sDDL NAND gates have been designed and tested. The topology which allows obtaining the better trade off between area increase and NED improvement is reported in Figure 5.4: the pull down stage has been designed to have the same number of series-connected devices for each input pattern. Actually, the parasitic capacitances at Y and $notY$ nodes do

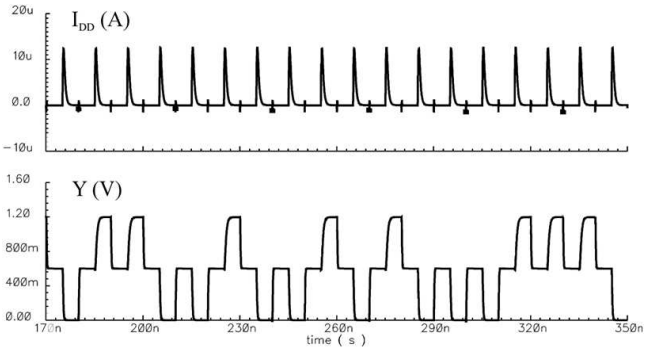


Figure 5.3: Output voltage and current drawn from the supply.

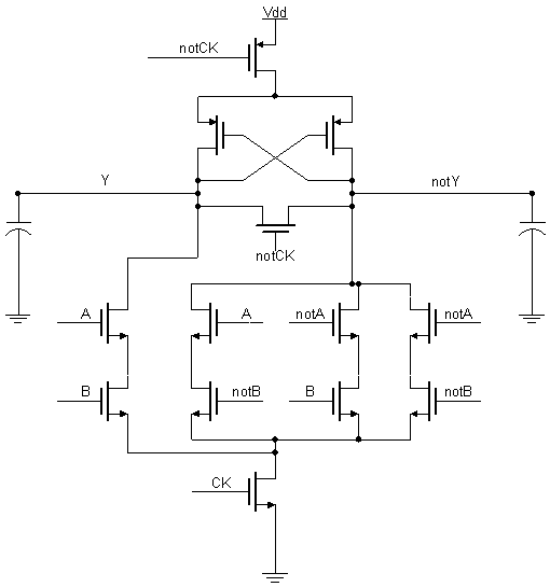


Figure 5.4: Proposed AND-NAND gate schematic.

not perfectly match; better results could be achieved by a different sizing of the MOS devices connected to Y and connected to $notY$, but this would increase area and input capacitance. Power consumption of the AND-NAND gate in Figure 5.4 has been studied for a four gate fan-out and taking into account an interconnection capacitance of about $10fF$ both at Y and $notY$ outputs. The NED achieved by using this topology is 3.6%. For a 10% and a 20% capacitance mismatch the NED is 12% and 22% respectively. The random input test has been carried out also for circuit in Figure 5.4 obtaining an NSD of 1.6%.

In order to compare 3sDDL against the main previously published SCA proof logic styles, a Serpent Substitution Box (S-Box) [3] has been implemented in sCMOS, DyCML [4], SABL [100] and 3sDDL logic styles. The S-Box has been synthesized in the Synopsis Design Vision environment: libraries containing only inverters, NAND gates and XOR gates for all the considered logic styles have been used.

The synthesized S-Box circuit is made up of 43 NAND gates, 1 XOR gate and 17 inverters which are necessary only for the sCMOS implementation. Interconnection capacitances have been extracted from the layout and $30fF$ has been assumed as load for each of the four output lines. A summary of the comparison between the different S-Boxes implemented in various logic styles is reported in Table 5.4.

Table 5.4: Comparison between the S-boxes designed with the different logic styles.

Logic Style	NED	NSD	$E_{AV}\,(fJ)$	# of MOS
sCMOS	95%	34.1%	173	216
DyCML	8.5%	1.9%	863	574
SABL	6.6%	1.5%	1381	706
3sDDL	2%	0.4%	544	570

5.2.2 Interfacing 3sDDL and standard CMOS

To implement the critical part of the encryption module in the proposed logic we have to interface standard CMOS signals with the 3sDDL gates by using registers composed of D-type flip-flop with conversion capability.

The flip-flops have been implemented as dynamic master-slave edge-triggered flip flops based on 3sDDL inverters and pass-transistors. These flip flops receive standard CMOS inputs and perform the conversion to the 3sDDL data.

An architecture implementing 3sDDL logic requires the routing of the clock to each cell in the 3sDDL combinational logic, and therefore is not directly compatible with a standard cell-based CMOS digital design flow. If we look at format of data signals in 3sDDL it is apparent that the information of clock transitions is present in the data signals. To obtain the clock signal, a circuit which is able to detect the third ($V_{DD}/2$) state in the data signal is needed. Since this circuit – that we will call "3^{rd} state detector" – has to be included within each logic gate, it has to be as compact as possible.

The aim of the 3^{rd} state detector circuit is to distinguish between the 0 or V_{DD} levels and the third state level $V_{DD}/2$ and to provide the clock signal. The schematic diagram of the proposed 3^{rd} state detector circuit is reported in Figure 5.5. The PMOS device in the first inverter after the detector core has to be high threshold MOS transistors in order to have no static power consumption when the inputs A and $notA$ (see Figure 5.5) are at $V_{DD}/2$.

Circuit simulations have been carried out by using the design kit of the HCMOS9 process from STMicroelectronics (devices with different threshold voltages are available) in CADENCE IC 4.4.6 environment.

5.3 The Three-Phase Dual-Rail Pre-Charge Logic Family

The 3sDDL logic family (discussed in the previous Section 5.2), like the others DRP logic styles for cryptographic applications, presents a power consumption independent of the processed data under the assumption that load capacitances are perfectly balanced. Each one of the load capacitances C_L and $C_{\overline{L}}$ of a DRP cell consist of the output capacitance of the cell itself C_o, the capacitance C_w of the wire to the subsequent cells and the sum of the input capacitances

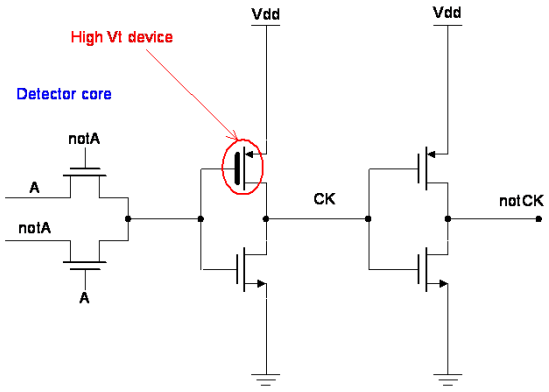


Figure 5.5: 3^{rd} state detector circuit.

C_i of these cells. In order to balance C_L and $C_{\overline{L}}$, the three parts of these capacitances need to be pairwise balanced. For modern process technologies of digital circuits, the biggest contribution to the capacitance at the output of a logic cell is typically C_w. Therefore, balancing $C_{L,w}$ and $C_{\overline{L},w}$ is the most essential task and this is done during placing and routing of the DRP cells. For instance, a technique for the automatic routing of balanced complementary lines has been introduced in [101]. Even if it pays attention to the P&R of the DRP cells, the proposed balanced routing technique does not take into account the dependence of the capacitive load on a line on the logic state of the adjacent wires and, furthermore, introduces additional constraints for the routing tool thus limiting its efficiency. Moreover, in this way the input capacitances C_i of the subsequent cells become predominant in C_L.

The Three-Phase Dual-Rail Pre-Charge Logic (TDPL) style is a further approach to the design of a dual-rail pre-charge logic family which is insensitive to unbalanced load conditions thus allowing adopting a semi-custom design flow (automatic place & route) without additional constraints on the routing of the complementary wires.

The proposed concept is based on a three phase operation where an additional discharge phase is performed after the precharge/evaluation steps typical of any dynamic logic style. Although the concept is general, it can be implemented as an improvement of the standard DRP logics with a limited increase in circuit complexity.

In the TDPL logic, during a first phase (precharge), the output lines of a generic logic gate are both charged to V_{DD}, then (second phase, evaluation) the proper line is discharged to V_{SS} according to the input data, thus generating a new output data. Finally, during the last phase (discharge), the other line is discharged too. As a consequence, since both wires are precharged to V_{DD} and discharged to V_{SS}, a TDPL logic gate shows a constant energy consumption over its operating cycle (independent of the input data), even if unbalanced capacitive loads to V_{DD} and/or V_{SS} are taken into account.

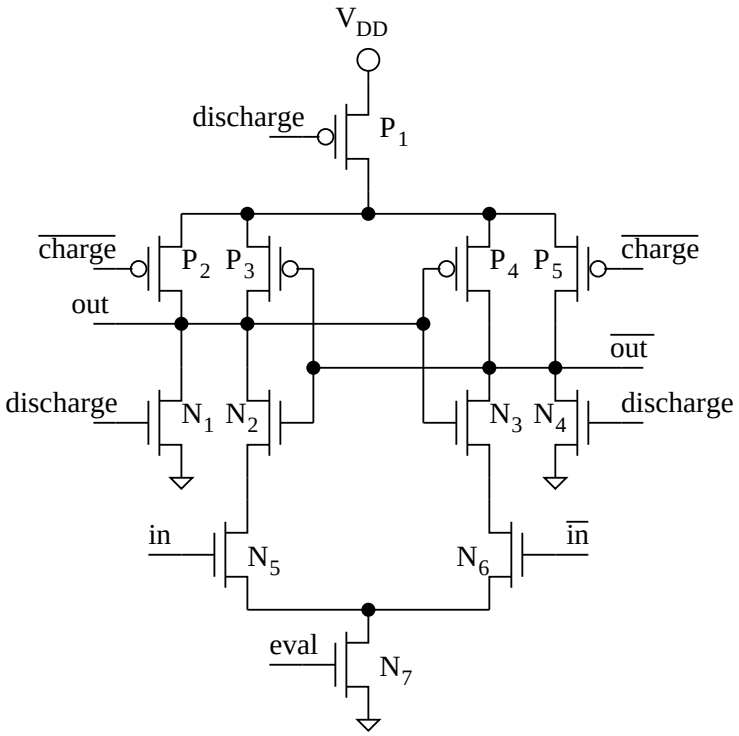


Figure 5.6: TDPL inverter.

The proposed logic style has been implemented as an enhancement of the SABL logic style with a minimum increase in the required area. Therefore, SABL cells are assumed as the benchmark for the equivalent TDPL cells in this work. For instance, an TDPL inverter is shown in Figure 5.6, where the pull-down NMOS transistors (N_1, N_4) and a PMOS switch (P_1) have been added to the SABL inverter in order to implement the discharge phase.

With reference to the timing diagram shown in Figure 5.7, the circuit operation is the following:

1. Charge: at the beginning of each cycle, signal *discharge* goes low, thus closing P1. Signal $\overline{charge}$ goes low too and both output lines are precharged to V_{DD}.

2. Evaluation: during the charge phase new input data (in, $\overline{in}$) are presented to the circuit. On the raising edge of signal *eval*, N_7 is closed thus discharging one of the output lines according to the input data.

3. Discharge: at the end of each operating cycle, input *discharge* is activated in order to pull down (through the additional pull-down transistors N_1, N_4) the output line which has not been discharged during the evaluation phase.

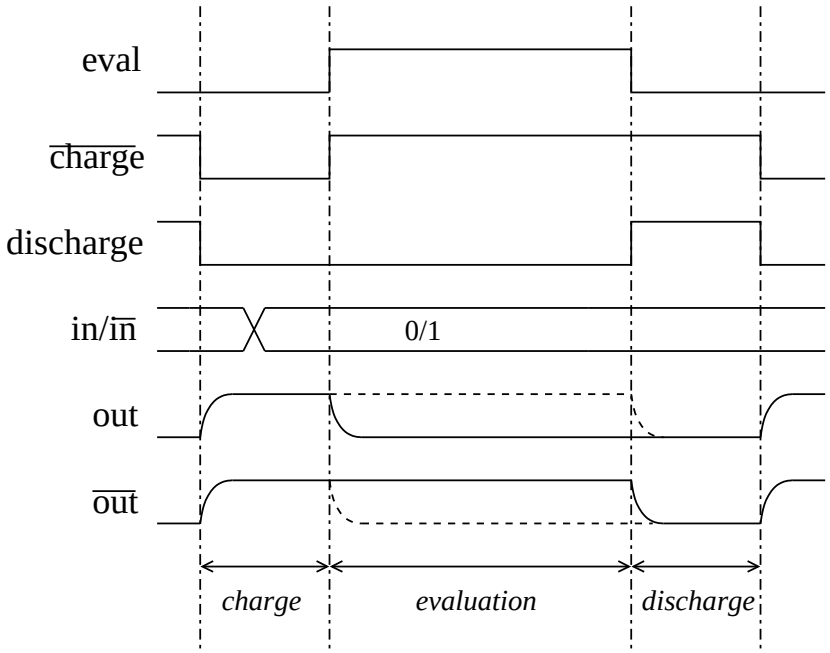


Figure 5.7: Timing diagram of the TDPL inverter.

Table 5.5: Capacitive loads.

	to V_{DD}	to V_{SS}
from out	$C_{out}^{VDD} = 8fF$	$C_{out}^{VSS} = 4fF$
from $\overline{out}$	$C_{\overline{out}}^{VDD} = 1fF$	$C_{\overline{out}}^{VSS} = 3fF$

More complex gates are obtained changing the pull-down logic. As an example, a 2-input NAND/AND and a XOR/NXOR are depicted in Figure 5.8.

A basic set of cells has been designed in a $65nm$ CMOS process. A $1.2V$ supply voltage and a 100MHz operating frequency are adopted. Each transistor is designed with a width $W = 0.12\mu m$ and the minimum gate length $L = 65nm$ is assumed. Simulations are done in Spectre, using BSIM4 transistor models.

In order to simulate the cells in a real operating condition, the testbench shown in Figure 5.9 has been defined, where each input to the gate under analysis is driven by a TDPL inverter and unbalanced load capacitances to V_{DD} (C_{out}^{VDD}, $C_{\overline{out}}^{VDD}$) and V_{SS} (C_{out}^{VSS}, $C_{\overline{out}}^{VSS}$) are assumed on the output lines (out, $\overline{out}$). Typical values for the local routing parasitic capacitances in a standard-cell semi-custom layout are used (Table 5.5). The same testbench, with SABL inverters on the inputs, has been used to simulate the corresponding SABL cells. In both cases, only the current consumption of the gate under analysis is taken into account and every input data transition is simulated.

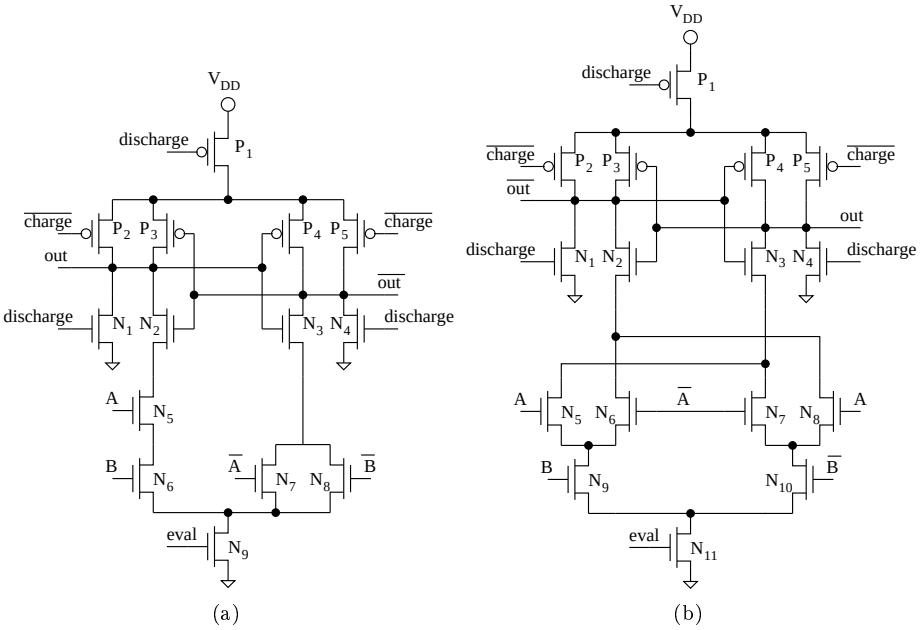


Figure 5.8: NAND/AND (a) and XOR/NXOR (b).

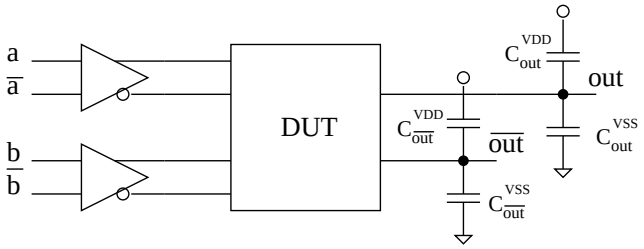


Figure 5.9: Simulation testbench.

Table 5.6: Simulation results for the three basic gates.

	INV		NAND/AND		XOR/NXOR	
	SABL[96]	TDPL	SABL[96]	TDPL	SABL[96]	TDPL
$\min(E)$[fJ]	8.30	16.79	8.52	16.97	8.75	17.16
$\max(E)$[fJ]	10.86	16.86	10.92	17.17	11.25	17.31
Δ_E[fJ]	2.55	0.08	2.40	0.20	2.51	0.15
NED	23.5%	0.5%	26.4%	1.4%	26.7%	1.0%
$\overline{E}$[fJ]	9.58	16.83	9.17	17.09	10.00	17.23
σ_E[fJ]	1.26	0.04	0.96	0.08	1.12	0.07
NSD	13.2%	0.3%	12.5%	0.5%	13.5%	0.5%
# transistors	10	12	12	14	14	16

As in [96], the energy per cycle reported in (5.1) is adopted as figure of merit to measure the resistance against power analysis attacks. The obtained results for the three analyzed gates are summarized in Table 5.6, where Δ_E is the difference $(\max(E)-\min(E))$, the normalized energy deviation (NED) is defined as in (5.2) and NSD is the normalized standard deviation defined as in (5.3). As expected, SABL gates are sensible to unbalanced load conditions (NED$>$ 25%, NSD$>$ 12%) thus confirming that a balanced routing must be necessary employed to obtain a constant energy consumption. On the contrary, TDPL cells show an extremely balanced energy consumption (NED$<$ 2%, NSD$<$ 1%) in spite of unbalanced load capacitances.

From Table 5.6, it follows that, as expected, an increase in the mean energy per cycle must be taken into account since both output lines are discharged in each cycle. On the contrary, the penalty in terms of silicon area is minimal (16% for the NAND/AND), especially if compared with what is reported for MDPL [78]. With respect to SABL, TDPL requires the routing of two additional signals ($\overline{charge}$ and $discharge$). However, if at least four metal layers are available for signal routing, an increase in silicon area is not expected, especially in regular structures such as data-paths. Notice that MDPL is affected by a similar drawback due to the routing of the random data for masking.

5.3.1 TDPL flip-flop implementation

The flip-flops in DRP circuits, as the SABL flip-flop reported in [96], consist of two stages, as shown in Figure 5.10. With respect to the flip-flop, when the previous DRP combinational logic is in one of the two phases (that is the same of Stage 2 and the following combinational DRP cells) Stage 1 is in the other phase. To clarify this behavior, let's assume Stage 1 is in the precharge phase when Stage 2 and the combinational DRP cells are in the evaluation phase. In this case, Stage 2 of the DRP flip-flops provides the stored logic values to the combinational DRP cells. The combinational DRP cells calculate their output values according to the respective input values. Just before Stage 2 of the DRP flip-flops and the combinational DRP cells are precharged, Stage 1 of the DRP

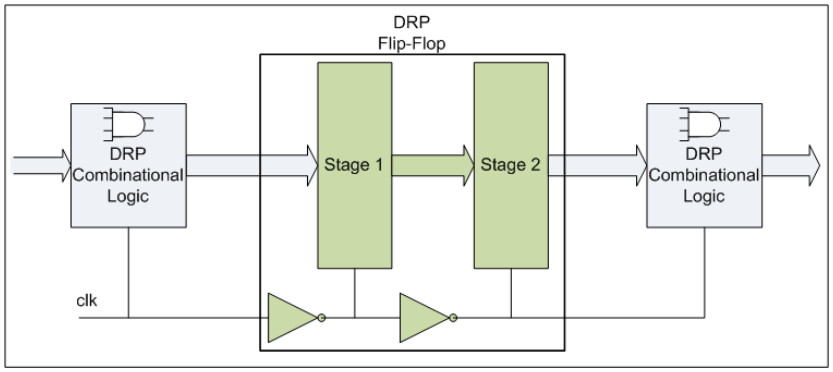


Figure 5.10: Generic DRP Flip-Flop.

flip-flops stores the logic values at its inputs. This behavior ensures that all cells of the DRP circuit are precharged during a clock cycle and that the processed logic values are not lost during the precharge phase.

The implementation of a data flip-flop compatible with TDPL gates is based on the scheme shown in Figure 5.11: it includes a first TDPL inverter, an intermediate circuit which drives the $\overline{set}/\overline{reset}$ inputs to a CMOS SR-latch and a second TDPL inverter on the output.

Its operation is similar to the SABL flip-flop reported in [96] where the additional circuitry after the first TDPL inverter avoids the latch invalid input ($\overline{set} = \overline{reset} = 0$) by forcing a logic-1 during both the phases of discharge and charge. In other words, the input TDPL inverter with the additional circuitry on its outputs behaves as a SABL inverter. The timing diagram shown in Figure 5.12 clarifies the flip-flop operation where $eval\_n$ is equal to $eval$ negated and $discharge^*$, $\overline{charge}^*$ are the discharge and charge signals referred to $eval\_n$. Therefore, as $discharge$, $\overline{charge}$ are active when $eval = 0$, $discharge^*$, $\overline{charge}^*$ are active when $eval\_n = 0$.

With reference to Figure 5.12, on the $eval$ falling edge, the input inverter enters its evaluation phase and $node_1$ is set to $\overline{D}$. Further changes of the input data during the evaluation phase do not affect $node_1$. Transistors N_1,N_3 are closed (P_2,P_4 are open) and the CMOS SR-latch is either set or reset depending on $node_1$ thus storing the input data ($node_2$). Meanwhile, the output inverter is in the discharge and charge phases thus being insensitive to its inputs.

When the input inverter enters its discharge phase (on the $eval$ raising edge), N_1 and N_3 are open and P_2, P_4 are closed thus forcing the hold state of the SR-latch ($\overline{set} = \overline{reset} = 1$). Hereafter, changes on $node_1$ due to the discharge/charge phase in the first invert do not affect $node_2$. The output inverter enters its evaluation phase thus setting Q according to $node_2$.

Ultimately, the value assumed by the output Q during the evaluation phase on the $eval$ raising edge is equal to the value taken by the input D during the evaluation phase on the previous $eval$ raising edge according to the master-slave

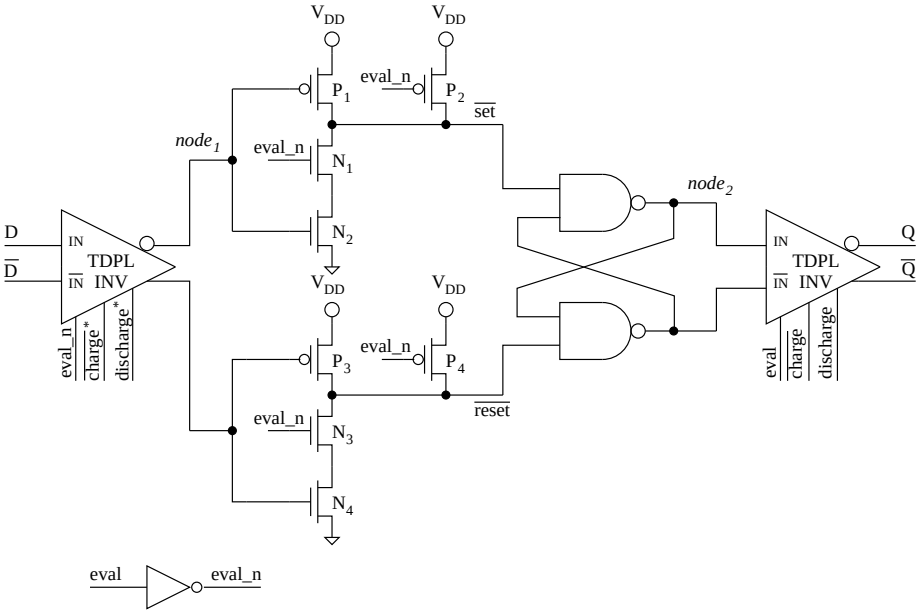


Figure 5.11: TDPL flip-flop.

operation.

5.3.2 Case study

As a case study, the circuit shown in Figure 5.13 has been simulated. It includes two 4-bit input registers (data_i, key_i), four XOR gates, the S-box S_0 from the Serpent algorithm [3] and a 4-bit output register (data_o). The circuit has been implemented in TDPL and SABL as well. In order to take into account the unbalanced routing in a semi-custom layout, every gate output is loaded with unbalanced load capacitances to V_{DD} and V_{SS}. In details, the asserted lines are loaded with $4fF$ and $2fF$ to V_{DD} and V_{SS} respectively, while $0.5fF$ and $1.5fF$ have been used for the negated lines. These are reasonable parasitic capacitance values for the local routing in the adopted technology.

A histogram of the observed energies per cycle reported in Figure 5.14 shows that TDPL guarantees a balanced energy consumption, independent of the processed data, even in presence of unbalanced interconnections. Notice that the same scale has been used for the energy per cycle.

The obtained simulation results are summarized in Table 5.7 for both SABL and TDPL, where the energy consumption has been evaluated on both the clock cycles necessary to process the input data. These results confirm that, as for the single logic gates, TDPL shows an extremely balanced energy consumption in spite of unbalanced semi-custom routing.

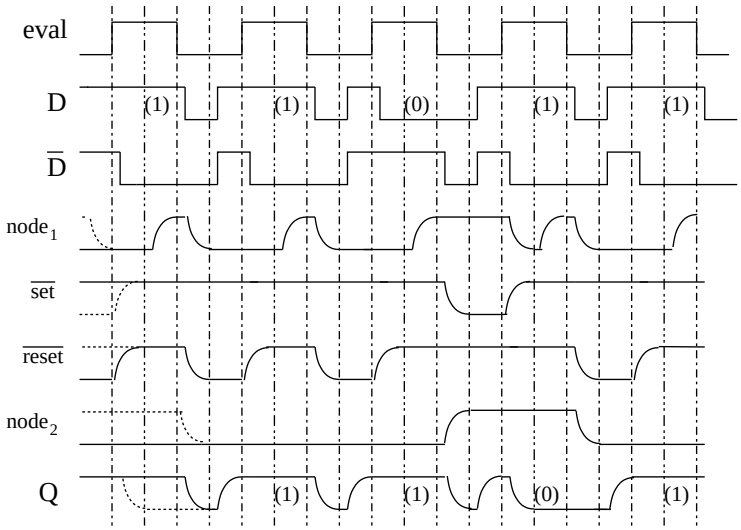


Figure 5.12: Flip-flop operation.

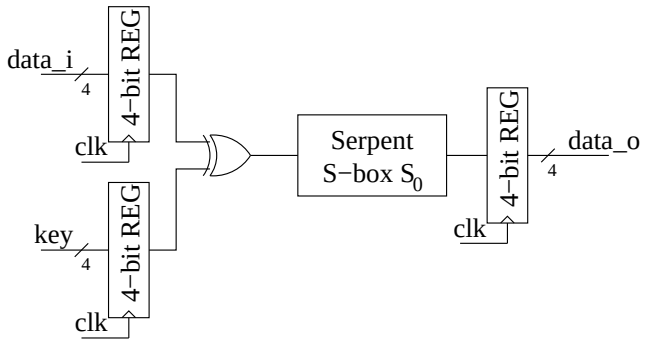


Figure 5.13: Circuit used as case study.

Table 5.7: Simulation results for the case study circuit (Figure 5.13).

	I clock cycle		II clock cycle	
	SABL[96]	TDPL	SABL[96]	TDPL
max(E)[fJ]	746.53	1305.23	750.50	1305.59
min(E)[fJ]	726.91	1302.44	722.53	1302.63
ΔE[fJ]	19.62	2.79	27.97	2.97
NED	2.63%	0.21%	3.73%	0.23%
$\overline{E}$[fJ]	739.35	1304.07	738.32	1303.94
σ_E[fJ]	4.16	0.54	5.54	0.66
NSD	0.56%	0.04%	0.75%	0.05%

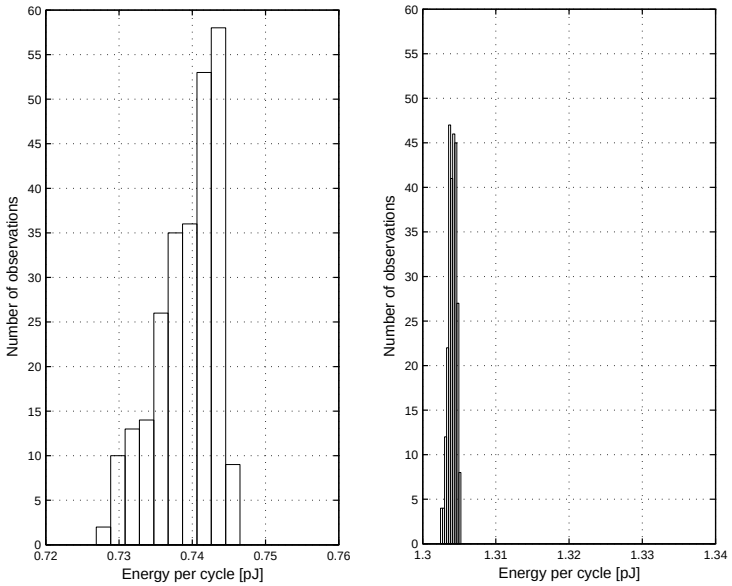


Figure 5.14: Case study - energy consumption per cycle: SABL (left) vs. TDPL (right).

5.4 The Delay-based Dual-rail Pre-charge Logic Family

In the previous section it has been shown how the problem concerning the unbalanced load conditions is addressed by the TDPL logic family in which however two additional control signals are needed (Section 5.3).

In this section, a completely innovative logic style is proposed as a further approach to the design of a DRP logic family which is insensitive to unbalanced load conditions. This logic style is called Delay-based Dual-rail Pre-charge Logic (DDPL) and it can be considered an improvement of the TDPL logic style.

DDPL exploits the time domain data encoding shown in Figure 5.15: during the precharge phase both differential lines are charged to V_{DD} and, in the evaluation phase, are both discharged to V_{SS}. The information is encoded in the order with which the lines are discharged. For a logic-1, the negated line is discharged after a delay Δ with respect to the asserted one. Conversely, for a logic-0, the negated line is discharged first. Since over the operating cycles both lines are charged and discharged once, the total current consumption is data-independent.

A 2-input NAND/AND and a XOR/NXOR which operate accordingly to the introduced data encoding are depicted in Figure 5.16.

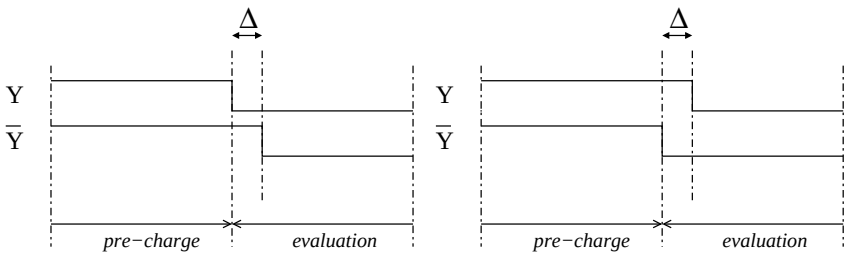


Figure 5.15: Time domain data encoding.

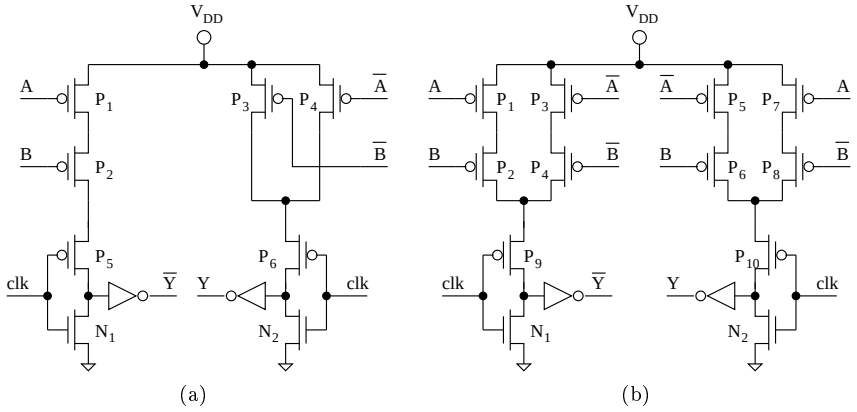


Figure 5.16: NAND/AND (a) and XOR/NXOR (b).

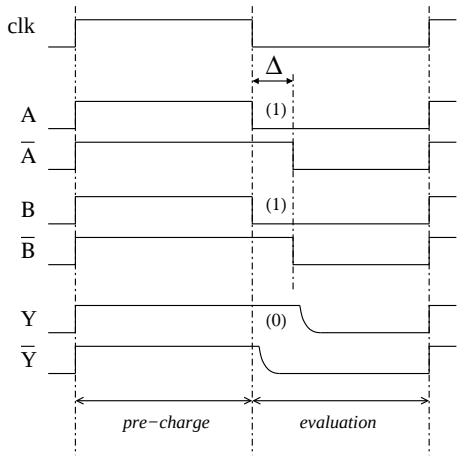


Figure 5.17: Timing diagram of the DDPL NAND.

With reference to the timing diagram shown in Figure 5.17, the NAND operation is the following:

1. Precharge: at the beginning of each cycle, signal *clk* goes high, thus closing N_1 and N_2 and pre-charging both output lines V_{DD}. Since during this phase the input lines are high (outputs from another DDPL gate), the pull-up logic is open.

2. Evaluation: the new DDPL encoded input data (A, $\overline{A}$), (B, $\overline{B}$) are presented to the circuit on the falling edge of signal *clk*. Since A, B go low before $\overline{A}$,$\overline{B}$ (both inputs are logic-1's), the negated output $\overline{Y}$ is discharged before Y, thus generating a logic-0, as expected in a NAND gate.

The NAND operation for the other input combinations is similar.

In order to convert a single-rail CMOS data to the DDPL format, the converter shown in Figure 5.18 is used: during the precharge phase ($clk = 1$), both outputs are charged to V_{DD} while, on the clock falling edge, they are discharged to V_{SS} forcing a delay Δ between them, accordingly to the single-rail input data A. If $A = 1$, $Y = 0$ and $\overline{Y} = 0$ after a delay Δ. Conversely, for $A = 0$, $\overline{Y} = 0$ and $Y = 0$ after a delay Δ. By construction, the CMOS-to-DDPL converter has a data independent current consumption.

This basic set of cells has been designed in a $65nm$ CMOS process. A $1V$ supply voltage and a 100MHz operating frequency are adopted. Each transistor

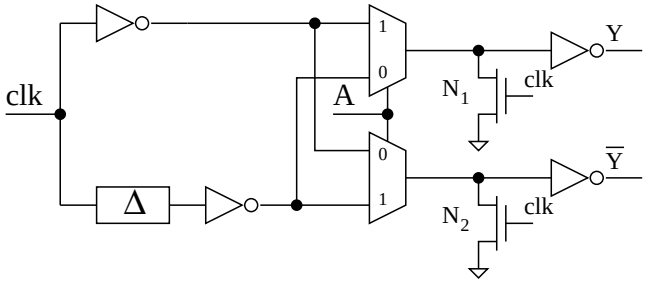


Figure 5.18: CMOS-to-DDPL converter.

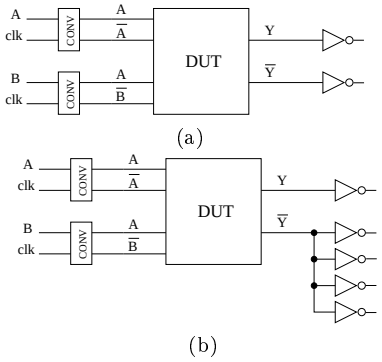


Figure 5.19: Simulation testbenches: balanced (a) and unbalanced (b) loads.

is designed with a width $W = 0.12\mu m$ and the minimum gate length $L = 65nm$ is assumed. A delay $\Delta = 1ns$ is used for the CMOS-to-DDPL converter. Simulations are done in Spectre, using BSIM4 transistor models.

In order to simulate the cells in a real operating condition, the testbenches shown in Figure 5.19 have been defined where, each input to the gate under analysis is driven by the CMOS-to-DDPL converter. To simulate both balanced and unbalanced loads, a different number of CMOS inverter is used on the two outputs. The same testbench has been used to simulate the corresponding SABL and WDDL cells. In both cases, only the current consumption of the gate under analysis is taken into account and every input data transition is simulated.

For the NAND/AND gate, a superimposition of the power supply current traces $I_{DD}(t)$ for the 16 input transitions is depicted in Figure 5.20 in case of unbalanced loads. In both the cells SABL and DDPL, each operation phase can be clearly identified in the supply current profile. Notice that, in unbalanced load conditions, SABL cells show a data dependent current consumption especially during precharge. In the DDPL cells, the precharge current pulse is constant. In the evaluation phase, two pulses are visible which correspond to the transitions at distance $\Delta = 1ns$ of the outputs lines. These two pulses show some data dependency but the sum of their energies is constant (Table 5.8).

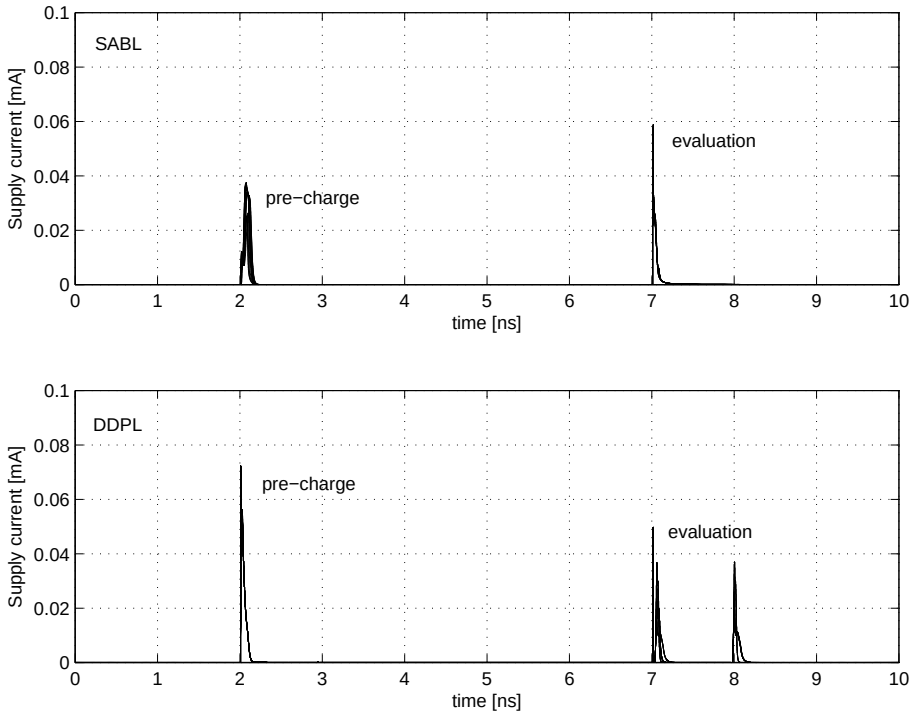


Figure 5.20: NAND/AND - superimposition of the power supply current traces: SABL (above) vs. DDPL (bottom).

Clearly, if a measuring setup is available which allows to resolve the single peaks within one operating period, both logics can be attacked. However, it is worth noting that DDPL shows an advantage with respect to SABL since the two evaluation peaks at distance $\Delta = 1ns$ must be resolved in order to detect a data dependency while, for SABL, resolving the precharge peak is sufficient.

In other terms, in a standard precharge logic like SABL, the operating frequency constraints the logic synthesis of the design and determines, at the same time, the achievable security level. On the contrary, in DDPL, the clock frequency does not fix the security since it depends on the delay Δ which must be chosen considering only the critical path t_{crit} of the design ($\Delta > t_{crit}$). Therefore, a cryptographic core in DDPL can run at a low frequency (for instance in a power constrained application) having, in spite of that, a high resistance against DPA.

As in [96], the energy per cycle assumed as in equation (5.1) is adopted as figure of merit to measure the resistance against power analysis attacks. The obtained results for the analyzed gates are summarized in Table 5.8 and Table 5.9, where the normalized energy deviation (NED) and the normalized standard deviation (NSD) are defined as in equations (5.2) and (5.3) respectively.

Table 5.8: Simulation results of DDPL NAND and XOR gates.

	Balanced loads		Unbalanced loads	
	NAND	XOR	NAND	XOR
$\max(E)$[fJ]	3.756	4.007	5.375	4.896
$\min(E)$[fJ]	3.720	3.986	5.337	4.873
ΔE[fJ]	0.036	0.021	0.038	0.023
NED	1%	0.5%	0.7%	0.5%
$\overline{E}$[fJ]	3.739	3.996	5.358	4.883
σ_E[fJ]	0.010	0.008	0.011	0.007
NSD	0.3%	0.2%	0.2%	0.2%
# transistors	12	16	12	16

Table 5.9: NAND - comparison with SABL and WDDL.

	Balanced loads			Unbalanced loads		
	SABL[96]	WDDL[102]	DDPL	SABL[96]	WDDL[102]	DDPL
$\max(E)$[fJ]	3.121	8.613	3.756	4.615	10.24	5.375
$\min(E)$[fJ]	2.958	7.983	3.720	2.958	8.014	5.337
ΔE[fJ]	0.163	0.630	0.036	1.657	2.223	0.038
NED	5.2%	7.3%	1%	35.9%	21.7%	0.7%
$\overline{E}$[fJ]	2.989	8.261	3.739	4.195	9.491	5.358
σ_E[fJ]	0.041	0.176	0.010	0.699	0.801	0.011
NSD	1.4%	2.1%	0.3%	16.7%	8.4%	0.2%
# transistors	16	30	12	16	30	12

As expected, SABL and WDDL gates are sensitive to unbalanced load conditions (NED$>$ 21.7%, NSD$>$ 8.4%) thus confirming that a balanced routing must be necessary employed to obtain a constant energy consumption. Conversely, DDPL cells show an extremely balanced energy consumption (NED$<$ 0.7%, NSD$<$ 0.2%) in spite of unbalanced load capacitances.

From Table 5.9, it follows that, as expected, an increase in the mean energy per cycle must be taken into account since both output lines are discharged in each cycle. In terms of silicon area (see transistor count in Table 5.9), DDPL shows a certain improvement with respect to SABL (25% for the NAND/AND) and a relevant advantage with respect to WDDL (60%). Compared to TDPL, a lower area consumption is also expected since DDPL does not require the routing of additional control signals.

5.4.1 A combinational case study

In order to confirm the results discussed in the previous section, a DDPL full adder, designed as depicted in Figure 5.21, has been tested and compared with the equivalent SABL design. An implementation based on XOR/NXOR and NAND/AND gates is employed and cascaded gates are connected using

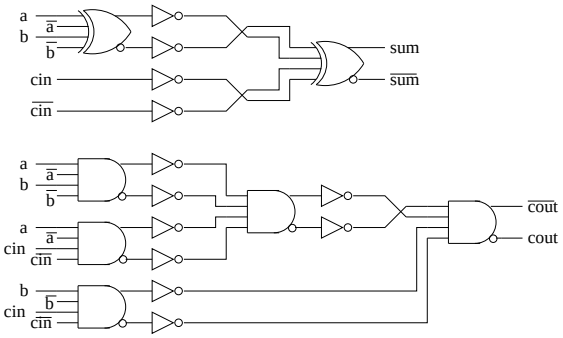


Figure 5.21: DDPL/SABL full adder.

Table 5.10: Simulation results for the FULL ADDER.

	Balanced loads		Unbalanced loads	
	SABL[96]	DDPL	SABL[96]	DDPL
$\max(E)$[fJ]	20.263	24.778	22.922	27.909
$\min(E)$[fJ]	19.573	24.676	19.768	28.001
ΔE[fJ]	0.690	0.102	3.154	0.092
NED	3.41%	0.41%	13.76%	0.33%
$\overline{E}$[fJ]	19.709	24.731	21.309	27.959
σ_E[fJ]	0.136	0.027	0.734	0.025
NSD	0.69%	0.11%	3.44%	0.09%

a Domino logic where the static inverters are included inside the gates (Figure 5.16) and they do not cause an unbalanced energy consumption because, in each cycle, both inverters on each couple of output wires switch two times (0-1 commutation during the precharge phase and a 1-0 event during the evaluation). On the contrary, in the SABL approach balanced interconnections between inverter and the following gate are necessary.

As for the simulation of a single gate, balanced and unbalanced load conditions have been used on the outputs (Figure 5.19). A superimposition of the power supply current traces $I_{DD}(t)$ for the 64 possible transitions of the 3-bit input $\{A, B, C_{in}\}$ is depicted in Figure 5.22 for both implementations SABL and DDPL in the unbalanced case.

A histogram of the observed energies per cycle reported in Figure 5.23 shows that DDPL guarantees a balanced energy consumption, independent of the processed data, even in presence of unbalanced loads. Results summarized in Table 5.10 confirm the improvement which has been obtained with respect to SABL.

5.4.2 DDPL flip-flop implementation

The implementation of a data latch for the proposed logic style is based on the scheme shown in Figure 5.24: the DDPL encoded data input is converted to a

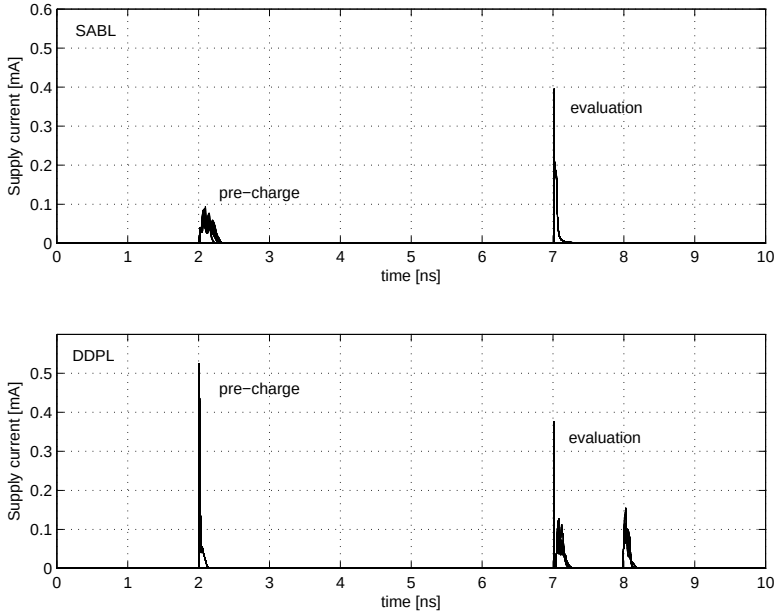


Figure 5.22: FULL ADDER - superimposition of the power supply current traces: SABL (above) vs. DDPL (bottom).

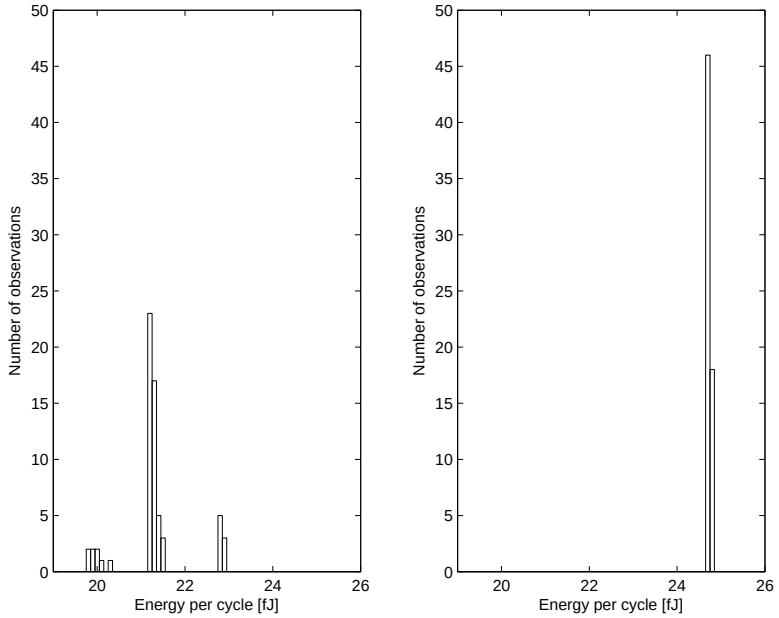


Figure 5.23: FULLADDER - energy consumption per cycle: SABL (left) vs. DDPL (right).

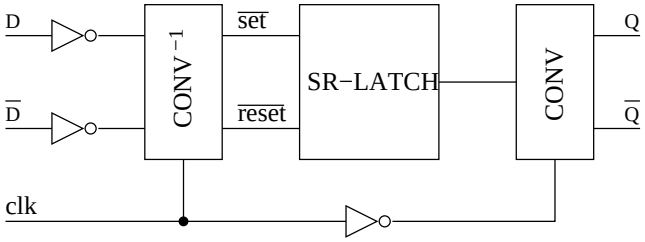


Figure 5.24: DDPL latch.

dual-rail CMOS format by an input converter ($CONV^{-1}$) whose outputs are the $\overline{set}/\overline{reset}$ inputs to a CMOS SR-latch. On the outputs, a CMOS-to-DDPL converter as in Figure 5.18 is used to regenerate a DDPL data for the following combinatorial logic. A DDPL data flip-flop is obtained cascading two latches in master-slave configuration.

The input DDPL-to-CMOS converter and the corresponding timing diagrams are shown in Figure 5.25(a) and 5.25(b) respectively: during the first semi-period of the clock signal *clk*, P_1 is open and the outputs $\overline{set}$, $\overline{reset}$ are both forced high by N_1, N_4 (hold state for the SR-latch). On the clock falling edge, P_1 is closed, the outputs are released and the DDPL encoded input data (A,$\overline{A}$) is presented to the circuit. As soon as the first input line goes low, P_2 is closed and, depending on which input line goes low first, the corresponding output line goes low as well thus storing the correct logic value in the CMOS SR-latch. The cross-coupled NMOS transistors N_2, N_3 force the other output in the opposite logic state.

In Figure 5.25(b), Δ is the delay forced by the CMOS-to-DDPL converter of the previous flip-flop. Obviously, the starting delay Δ changes during the propagation through the combinatorial logic. It must be ensured that $\Delta_f > \Delta_{setup}$, where Δ_f is the final delay on the flip-flop inputs and $\Delta_{setup} = t_p^{HL} + t_p^{LH} + R_{on}C_o$, being t_p^{HL} and t_p^{LH} the NXOR propagation delays, R_{on} the resistance of the series (P_1,P_2,$P_{3/4}$) and C_o the total capacitance on the $P_{3/4}$ drains. However, as said above, the flip-flop regenerates a valid DDPL data thus ensuring that the propagation through the next combinatorial network starts with the original delay Δ. In addition, since the input converter evaluation phase is driven by the input signals themselves, the proposed logic is insensitive to a skew on the clock signal *clk* as long as it does not exceed Δ_f.

It is worth noting that NXOR gate in Figure 5.25(a) switches two times in each cycle. Therefore, under the assumption that a symmetric full-custom layout is used for the internal interconnections, the DDPL flip-flop is by construction DPA resistant.

5.4.3 A sequential case study

As a second case study, the circuit shown in Figure 5.13 has been simulated. It includes two 4-bit input registers (data_i, key_i), four XOR gates, the S-box

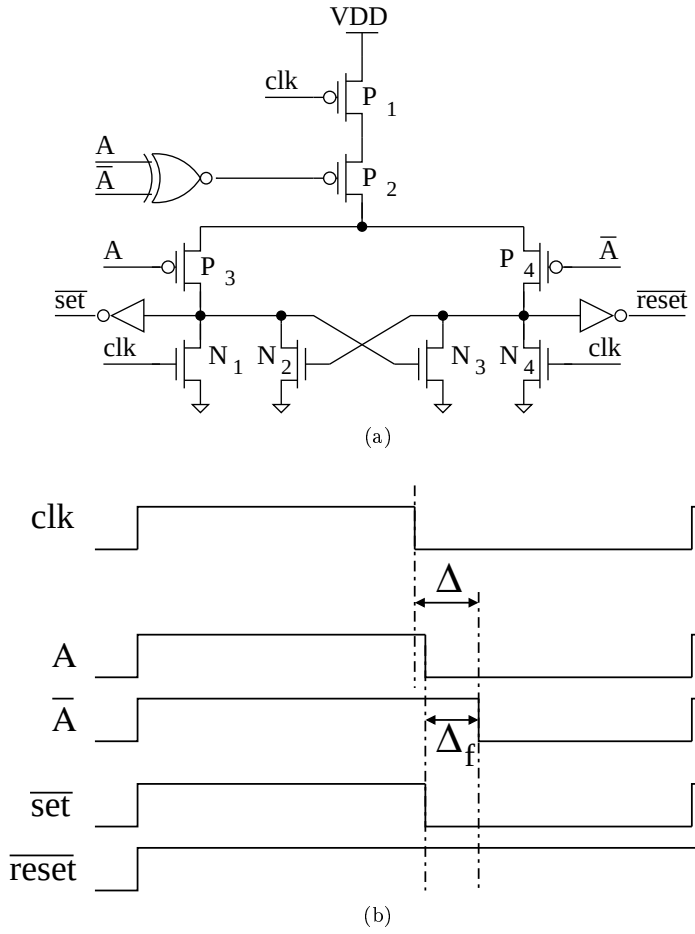


Figure 5.25: DDPL-to-CMOS converter: implementation (a) and timing diagram (b).

Table 5.11: Simulation results for the sequential case study circuit (Figure 5.13).

	I clock cycle		II clock cycle	
	SABL[96]	DDPL	SABL[96]	DDPL
$\max(E)$[fJ]	622.11	1151.74	625.42	1152.52
$\min(E)$[fJ]	605.76	1147.63	602.11	1147.63
ΔE[fJ]	16.35	4.12	23.31	4.89
NED	2.63%	0.36%	3.73%	0.82%
$\overline{E}$[fJ]	616.12	1149.92	615.27	1150.52
σ_E[fJ]	3.47	0.72	4.62	0.82
NSD	0.56%	0.06%	0.75%	0.07%

S_0 from the Serpent algorithm [3] and a 4-bit output register (data_o). The circuit has been implemented in DDPL and SABL as well. In order to take into account the unbalanced routing in a semi-custom layout, every gate output is loaded with unbalanced load capacitances to V_{DD} and V_{SS}. In details, the asserted lines are loaded with $4fF$ and $1.5fF$ to V_{DD} and V_{SS} respectively, while $1fF$ and $3fF$ have been used for the negated lines. These are reasonable parasitic capacitance values for the local routing in the adopted technology.

The obtained simulation results are summarized in Table 5.11 for both SABL and DDPL, where the energy consumption has been evaluated on both the clock cycles necessary to process the input data. These results confirm that, as for the single logic gates, DDPL shows an extremely balanced energy consumption in spite of unbalanced semi-custom routing.

5.5 Conclusions

Differential power analysis attacks work because the power consumption of cryptographic devices depends on processed data. The aim of a hardware countermeasure is to remove this dependency. Transistor level countermeasures have been analyzed in this Chapter showing how the DRP logic styles counteract DPA attacks. Three dynamic differential logic styles to protect security devices against power attacks have been designed and in-depth evaluated.

The 3sDDL, based on signals with three allowed states, has been shown as first. The designed gates and flip-flops have achieved a strong reduction of both the NED and the NSD parameters with reasonable penalty in terms of area and total power dissipation. A circuit which is able to recover the clock from 3sDDL data signals has also been proposed. Simulation results have shown a strong reduction of both the NED and the NSD with respect to the standard CMOS and others logic styles for cryptographic applications. The main drawback of 3sDDL has been shown to be that a 3sDDL gate has an independent power consumption on processed data under the assumption that load capacitance are perfectly balanced.

To address this drawback a second logic style has been proposed, the TDPL.

The design of an innovative DRP logic family whose power consumption is insensitive to unbalanced load conditions has been investigated by the adoption of an additional third (discharge) phase after precharge and evaluation. In this way, the TDPL logic style allows to adopt a semi-custom design flow (automatic place & route) without any constraint on the routing of the complementary wires. In particular a flip-flop compatible with the TDPL logic family has been introduced and compared to the state of the art in the technical literature. Experimental results on a case study in a 65nm CMOS process confirm that the proposed implementation shows a constant energy consumption even in presence of asymmetric interconnections. The simulated energy consumption per cycle shows an improvement in the energy consumption balancing in excess of 10 times with respect to the corresponding SABL implementation without requiring any constraint on the geometry of the complementary wires. The main drawback of TDPL is represented by the additional two control signals defined to set the three phases. These additional signals make more difficult the place and route processes.

The third proposed solution, the DDPL logic family, promises to be a completely innovative logic style. The DDPL style is based on a novel encoding concept where the information is represented in the time domain rather than in the spatial domain as in a standard dual-rail logic. Even if a DDPL gate works with only one control signal (as a standard DRP logic), it presents a power consumption which is insensitive to unbalanced load conditions as in the case of TDPL logic. A complete set of combinatorial gates and a data flip-flop have been designed and simulated showing that the proposed logic family has a constant energy consumption even in presence of asymmetric interconnections. The simulated energy consumption per cycle is up to 50 times more balanced than in the corresponding SABL gates without requiring any constraint on the geometry of the complementary wires. In terms of area, DDPL is comparable to SABL and 60% smaller than WDDL. The introduced time domain data encoding allows to set the DPA-resistance independently from the operating frequency by choosing the delay parameter Δ according to the expected resolution of current consumption measurements.

Chapter 6

Random Number Generator

6.1 RNG and PRNG

A random number generator (often abbreviated as RNG) is a computational or physical device designed to generate a sequence of numbers or symbols that lack any pattern, i.e. appears random. Random numbers are extensively used in many crypto-graphic operations. Public/private key pairs for asymmetric algorithms are generated from random bit streams; a random bit generator (RBG) is also needed for key generation in symmetric algorithms, for generating challenges in authentication protocols, and for creating padding bytes and blinding values [63].

The many applications of randomness have led to the development of several different methods for generating random data. Many of these have existed since ancient times, including dice, coin flipping, the shuffling of playing cards, the use of yarrow stalks (by divination) in the *I Ching*, and many other techniques. Because of the mechanical nature of these techniques, generating large amounts of sufficiently random numbers (important in statistics) required a lot of work and/or time. Thus, results would sometimes be collected and distributed as random number tables. Nowadays, after the advent of computational random number generators, a growing number of government-run lotteries, and lottery games, are using RNGs instead of more traditional drawing methods, such as using ping-pong or rubber balls.

There are two principal methods used to generate random numbers. In a True or physical RNG (*TRNG*) a physical phenomenon, that is expected to be random, is measured and then used to generate sequences of zeros and one. In a so called Pseudo RNG (*PRNG*) a computational algorithm that produce long sequences of apparently random results is used; in this case the generated sequences are completely determined by a shorter initial value, known as a seed or key. A physical random number generator can be based on an essentially random atomic or subatomic physical phenomenon whose unpredictability can be traced to the laws of quantum mechanics. Sources of entropy include radioactive

decay, thermal noise, shot noise, avalanche noise in Zener diodes, clock drift, the timing of actual movements of a hard disk read/write head, and radio noise. However, physical phenomena and tools used to measure them generally feature asymmetries and systematic biases that make their outcomes not uniformly random. A randomness extractor, such as a cryptographic function, can be used to obtain uniformly distributed bits from a non-uniformly random source, though at a lower bit rate.

An example of a PRNG is the linear congruential generator, which uses the recurrence

$$X_{n+1} = (aX_n + b) \; mod \, m \tag{6.1}$$

to generate numbers where m is the maximum number of numbers the formula can produce.

For random numbers used in cryptography, a flat statistic is not sufficient and their unpredictability is the main requirement.

A random bit generator (RBG) is a system whose output consists of fully unpredictable (i.e., statistically independent and unbiased) bits. In security applications, the unpredictability of the output also implies that it must not be possible for an attacker to observe or manipulate the generator. An RBG basically differs from a pseudo-random generator because the complete knowledge of the generator structure and of whatever previously generated sequence does not result in any knowledge of any following bit. In other terms, the entropy of an n-bit output sequence should be ideally equal to n. On the contrary, the entropy of a n-bit output sequence from a pseudo-random generator cannot exceed the entropy of its seed, whatever n is. While pseudo-random generators are suitable in those applications where just a flat statistic is needed [84], random number generators are required in security applications, where unpredictability is the main goal.

A true RBG must be necessarily based on some kind of nondeterministic phenomena that could act as the source of the system randomness. Electronic noises and time jitter are usually the only stochastic phenomena that are suitable for the integration in embedded systems as chipcard integrated circuits (ICs).

When designing an RBG for a chipcard IC, a wide spectrum of implementation issues has to be considered and fulfilled. Due to cost reasons and mechanical stress requirements, the silicon area is a limited resource in a chipcard microcontroller (a typical area is $5 - 10mm^2$ for a 8/16-bit card) and, at the same time, there is the demand to integrate nonvolatile memory blocks of ever-increasing size. As a consequence, the silicon area for integrating the CPU core and its peripheral devices (including the RBG module) must be minimized. Furthermore, no external components can be used due to packaging constraints and security reasons: any externally accessible circuit node seriously affects the chip tamper resistance [71].

To avoid complex power management policies, power consumption is another stringent constraint, especially in a contactless chipcard IC. A related issue is the chip resistance against power analysis attacks [53]: when the RBG is employed

in a key generation algorithm, a current consumption profile highly correlated to the RBGs output bit stream can be exploited by an attacker to infer the generated secret values.

Four different techniques for generating random streams are reported in the technical literature: direct amplification of a noise source [14, 22, 47], jittered oscillator sampling [23, 38, 50, 75, 74], discrete-time chaotic maps [88, 91] and metastable circuits [13, 40].

The direct amplification of a white noise source has been proved to be an effective technique to obtain high-speed random streams. Nevertheless, the sampling of a jittered oscillator is the preferred method because of its higher robustness and lower sensitivity to external disturbances [75].

An oscillator-based RBG exploits the cycle-to-cycle time drift (jitter) in free running oscillators to produce a random bit sequence. In the basic scheme, a fast oscillator is sampled by a lower frequency oscillator in a D flip-flop and, under the hypothesis that the standard deviation of the slow oscillator period is greater than the fast oscillator period, the states of the latter in two successive sampling times can be assumed uncorrelated, thus generating a random bit stream. To fulfill the mentioned constraint and, at the same time, to have a high generation speed, in [21] the low frequency oscillator is provided with an amplified noise source. The main drawback of this solution is the increased power consumption which is a main constraint, especially for the integration in contact-less chipcards.

To maximize the exploitation of the oscillator jitter when an explicit noise source is not employed, a new concept has been introduced in [15] where the fast oscillator phase is controlled to force the sampling close to one of its edges (offset-compensated oscillator-based RBG). That allows relaxing the jitter requirement and, in order to generate random bits, a jitter greater than the phase control resolution becomes the condition to be fulfilled in this case. Moreover, since in [15] both oscillators are reset every time a new bit is produced, the output stream is not affected by frequency beating between the sampled and sampling oscillator which, enhancing the pseudo-random behavior of the sequence, makes difficult detecting a lack of randomness. As a further advantage, this noise source is suitable to be implemented as a fully digital standard-cell based circuit, with a substantial advantage in terms of design time, power and area requirements with respect to other designs where a significant analog part is present.

Hardware RBGs can feature a very high throughput, but the random sources commonly used present several statistic defects, due to physical limitations (noise intensity, bandwidth limitation, fabrication tolerances, aging and temperature drifts), deterministic disturbances (power supply and substrate interference), and external attacks aimed at manipulation. Therefore, the post-processing of the raw bit stream from the source with a carefully designed compressing algorithm must be employed. A lower speed bit stream with increased statistical quality is generated from a high-speed near-random input stream by concentrating its entropy.

Even if, historically, the only requirement for an RBG was to fulfill bunches of statistical tests aimed to reveal defects in the generated data, nowadays, in

the technical community, it is well accepted that, for random numbers used in cryptography, the focus must be on the verification of a minimum entropy requirement and statistical tests are significant only if the statistical model of the random source under evaluation is known [86].

The German IT security certification authority (BSI) has adopted this approach in its AIS 31 publication [55] where the physical noise source is separated from the digital post-processing and criteria and statistical tests are defined for the noise source output (digitized noise signal) in order to verify a minimum entropy limit for the post-processor's output (internal random numbers). Namely, the entropy requirement on the final data is guaranteed by defining a minimum entropy limit for the raw data from the source and, at the same time, requesting that the adopted post-processing algorithm does not reduce its input entropy.

In [24] and [26], the authors go further ahead with this approach defining an RBG based on a stateless (memoryless) noise source and a stateless post-processing algorithm.

Since the noise source is assumed memoryless, the generated symbols are independent and, since the post-processor is also memoryless, the internal random numbers are independent too. Therefore, the entropy limit can be verified directly after the post-processor, controlling that the assumed compression ratio in the post-processor is well chosen with respect to the available entropy per bit from the source. In this scheme, very fast noise sources, but with a low entropy per bit ("spread" entropy sources), can be adopted, provided that a sufficient compression is applied. In other terms, the relevant figure of merit becomes the entropy throughput (entropy/second) instead of the entropy per bit, and the design of the noise source is not any more constrained by the statistical quality, but efficiency and robustness can be taken as the main goals.

Finally, in [23] the authors propose a novel concept for an oscillator based RBG which exploits the relative jitter between two identical ring oscillators sharing the same delay elements. The oscillators start synchronously and a detecting circuit signals when a relative jitter greater than a given threshold has been accumulated. Therefore, the generation rate is automatically adapted to the available noise and no pseudo-random patterns are generated thus improving the module testability.

In this Chapter two works on two distinct RNGs are presented.

The first part of the Chapter (Section 6.2) reports a side-channel attack on a chaos-based Random Number Generator based on power consumption analysis. The analyzed RNG has been designed by the research group of Professor Setti (University of Bologna, Italy) in a $3.3V$, $0.35\mu m$ CMOS technology. The aim of this attack is to verify if it is possible to retrieve information regarding the internal state of the chaotic system used to generate the random bits. In fact, one of the most common arguments against this kind of RNGs is that, due to the deterministic nature of the chaotic circuit on which they rely, the system cannot be truly unpredictable. The power consumption profile of a chaos-based RNG prototype analyze is analyzed, showing that a side-channel attack based on a power analysis can determine only the external (digital) state of the system

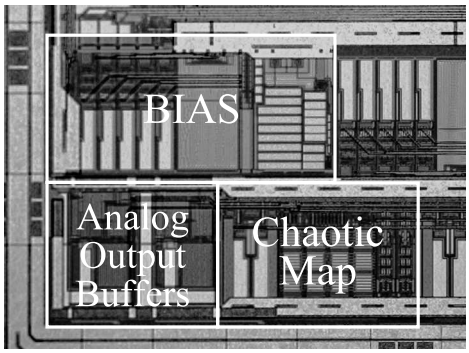


Figure 6.1: Micro-photograph of the $0.35\mu m$ CMOS prototype (detail).

but not the internal (analog) state.

In the second part of the Chapter (Section 6.3) measures on a RNG prototype chip are analyzed. The RNG manufactured in a $0.12\mu m$ CMOS process by Infineon Technologies is based on the shot noise associated to the leakage current of a reverse biased p-n junction. After a short presentation of the RNG, the obtained experimental results are evaluated to prove the randomness of the prototype chips.

6.2 Attacking a Random Number Generator

A power consumption based side-channel analysis on a chaos-based RNG is presented in this Section. After an explanation on how the RNG works, results about the side-channel analysis demonstrate power consumption reveals the generated bit but not the internal state of the RNG.

6.2.1 Architecture of the designed RNG

The RNG analyzed in this work has been designed in a 3.3V $0.35\mu m$ CMOS technology. A detail micro-photograph of it can be seen in Figure 6.1.

The core of this RNG is a chaotic map, formally a 1D discrete-time dynamical system whose state evolution is described by:

$$x_k = M\left(x_{k-1}\right) \tag{6.2}$$

with $M : I \rightarrow I$ while the random output bit D_k is given through the quantization function $Q : I \rightarrow 0, 1$ from the state of the map:

$$D_k = Q\left(x_{k-1}\right) \tag{6.3}$$

In the prototype the state x_k of the chaotic map is implemented as a differential voltage ranging in $I = [-1, 1]V$ and the two functions M and Q are respectively:

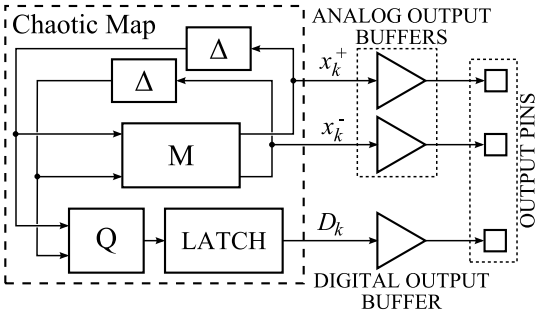


Figure 6.2: Basic architecture of the chaos-based RNG prototype. The map state x_k is implemented with the two differential analog voltages x_k^+ and x_k^- , while the random bit is the digital signal D_k.

$$M(x) = \begin{cases} 2x+2 & if\, x \leq -\frac{1}{2} \\ 2x & if\, -\frac{1}{2} < x \leq \frac{1}{2} \\ 2x-2 & if\, x > \frac{1}{2} \end{cases} \tag{6.4}$$

$$Q(x) = \begin{cases} 1 & if\, -\frac{1}{2} < x \leq \frac{1}{2} \\ 0 & elsewhere \end{cases} \tag{6.5}$$

Despite the deterministic evolution of the analog state x_k, that is regulated by equation (6.2), the succession of the quantized state D_k can be theoretically proved to be effectively a random, unpredictable bitstream. The proof can be found in [35]; here it is enough to recall that the only assumption required is that the initial condition of the system is unknown and randomly drawn according to a continuous probability density function (that is verified assuming the initial condition is affected by noise).

A block diagram of the prototype is depicted in Figure 6.2. The core of the circuit is the chaotic map, implementing both M and Q functions, and the unity delay blocks required to achieve the dynamic behavior as in equation (6.2). It is designed as a fully-differential switched capacitors circuit, a detailed description of which can be found in [79]. Due to testing purposes both the random generated bit D_k and the differential analog chaotic map state x_k are made available to output pins. The buffers used to drive these output pins have to be taken into account when analyzing the power supply current of the circuit, since due to limitation in the standard I/O cells available for this technology, they share the power supply line with the core circuit.

An example of waveforms generated by the prototype can be seen in Figure 6.3, showing at the same time the state of the chaotic map, the generated random bit and the current profile on the power supply line. Note that the state of the chaotic map is shown along with the two thresholds of the Q function: according to (6.4) and (6.5) when the state is in between the threshold, the random bit

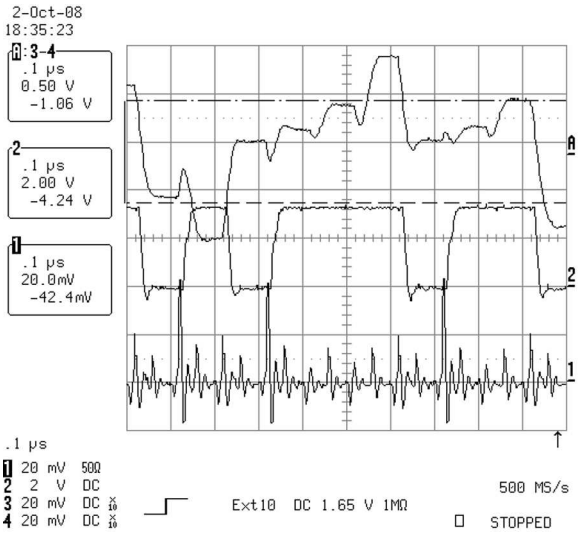


Figure 6.3: Prototype measurements. From top to bottom: differential analog internal state of the chaotic map (channel A); generated random bits (channel 2); power supply current (channel 1). The probe used for the the current sensing has a sensitivity of 5mV/mA.

at the next time step is high, while when it is outside, the next random bit is low. Note also that, since the probe used for sensing the power supply current requires an AC coupling, we are observing only the dynamic power, i.e. the variations with respect to the mean value of the current. Peaks in the current profile are present in correspondence to each clock edge (not shown in the figure); highest peaks can be found when the random output bit has a transition from low to high. To understand what is the expected current profile, let us consider the diagram of Figure 6.4. At the rising edge of the clock the analog state of the chaotic map switches with a short transient from the value x_k to the value $x_k + 1 = M(x_k)$, and the output random bit from $D_k = Q(x_k - 1)$ to $D_k + 1 = Q(x_k)$. During these transients, we can observe a peak in the power supply current due to the contribution of these three subcircuits: a) the chaotic map; b) the analog output buffers; and c) the output digital buffer. Generally, if the chaotic map gives the highest contribution to static power supply (it is designed only with class-A amplifiers), its contribution to the dynamic power is not particularly high. The same behavior can be observed for the analog buffers, while the digital buffer requires only dynamical power. Furthermore, due to the large capacity of the output pins, and to the single-ended configuration, this power is expected to be quite high only when observing a low-to-high transition. This is exactly what we can observe in Figure 6.3.

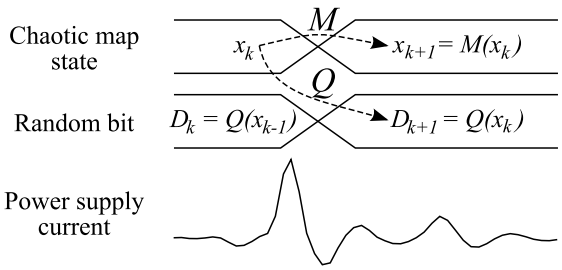


Figure 6.4: Typical current profile during a transients.

6.2.2 Analysis of the power supply current

As observed in Section 6.2.1, due to I/O cells constraints we have a dominant contribution to the dynamic current from the output random bit buffer (i.e. the digital buffer). Consequently, we can expect four kinds of current profiles, according to the random bit transition, which have been shown in Figure 6.5. Two major peaks can be observed, the first one corresponding to the rising edge of the clock, and the second one to the falling edge. Additionally, if we compare the two profiles associated with the low-to-low and high-to-high transition (i.e. the cases where there is no transition in the output random bit), we can see they are very similar.

We can verify from prototype measurements that the dynamic current profile is not directly related to the analog state of the chaotic map, but only to the generated random bit. More precisely, and using the same symbols of Figure 6.4, i.e. indicating respectively with x_k and x_{k+1} (D_k and D_{k+1}) the chaotic map state (random output bit) before and after the transient, the current profile during the transient does not depends directly on x_k and x_{k+1} but only on D_k and D_{k+1}. To prove this, we have to isolate the contribution given by the $D_k \rightarrow D_{k+1}$ transition considering separately the four cases corresponding to the four transitions considered in Figure 6.5. For all these cases, we have considered two scatter plots where the internal (analog) and the external (digital) state of the chaotic map measured from a long acquisition are compared with the observed current profile. Actually, to this purpose we need a numeric indicator of the current profile, which has been chosen as the charge required by the transient in a period T [96]:

$$\Delta q_k = \int_0^T i_D\left(t - kT\right) dt \tag{6.6}$$

where i_D is the measured dynamic current, so Δq_k is actually a differential charge.

In order to check if any relation exists between the state of the map and the power consumption it is useful to scatter plot the $\triangle q_k$ corresponding to a

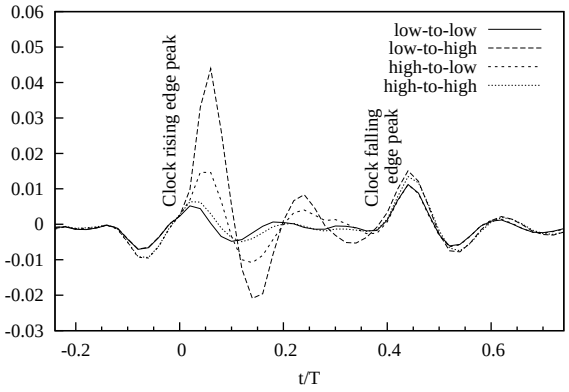


Figure 6.5: Example or the four kinds of dynamic current profile, according to the random output bit transition.

state transition. In this way in a scatter plot where $\triangle q_k$ is in y-axis and the RNG state is in x-axis, if the clusters points corresponding to the different RNG states (x_k) are at different levels along y-axis this means that each RNG state can be identified by its power consumption. Otherwise, if the clusters points corresponding to the different RNG states (x_k) cover indistinctly all area long y-axis this means that the same $\triangle q_k$ value corresponds to different RNG states and thus the RNG internal state can not be catched.

In Figure 6.6 the relation between the analog (internal) state and the power consumption is shown where the value of $\triangle q_k$ is associated to the internal state (x_k); conversely, in Figure 6.7 the relation between the digital (external) state and the power consumption is shown where the value of $\triangle q_k$ is associated to the external state $D_{k+1} = Q(x_k)$.

Note that in both the figures the values of (x_k) are correctly in the range $-1 < x_k < -1/2$ or $1/2 < x_k < 1$ when $D_{k+1} = Q(x_k) = 0$, instead when $D_{k+1} = Q(x_k) = 1$ the values of (x_k) are compelled in the range $-1/2 < x_k < 1/2$.

By observing the Figure 6.6 it is evident that the same charge $\triangle q_k$ corresponds to both RNG states. On the contrary, by observing the Figure 6.7 the charges $\triangle q_k$ corresponding to two different RNG states are clearly distinct. This means that no relation between the current profile and the internal state of the chaotic map can be measured, whereas the digital output $D_{k+1} = Q(x_k)$ is clearly traceable. Therefore, the RNG is internally secure but the digital output buffers are sensitive to side-channel attack.

As an additional test, it may be interesting checking if a relation exists between the observed current profile and the successive random bit D_{k+2}. This test effectively plays the role of the prediction of the following bit given the random bitstream and the current profile.

To get an intuitive idea, let us refer to Figure 6.8, showing the two distributions of the measured charge $\triangle q_k$, which has been separated in two groups

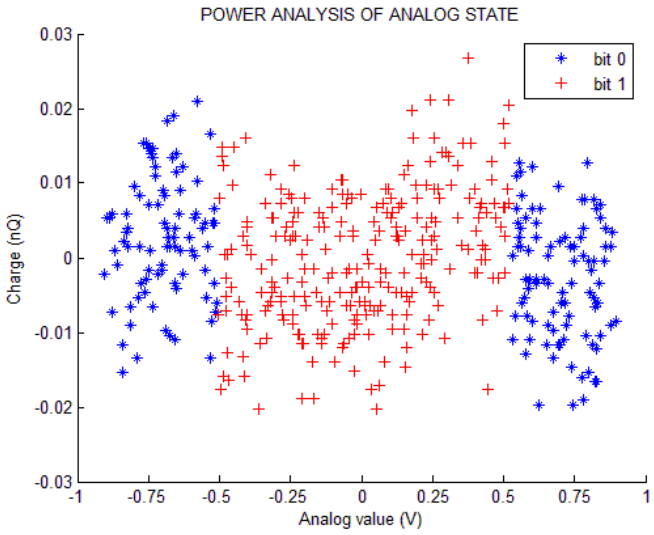


Figure 6.6: Scatter plot of the charge Δq_k required when the successive random bit D_{k+1} is high and low compared with the internal (analog) state of the chaotic map (x_k).

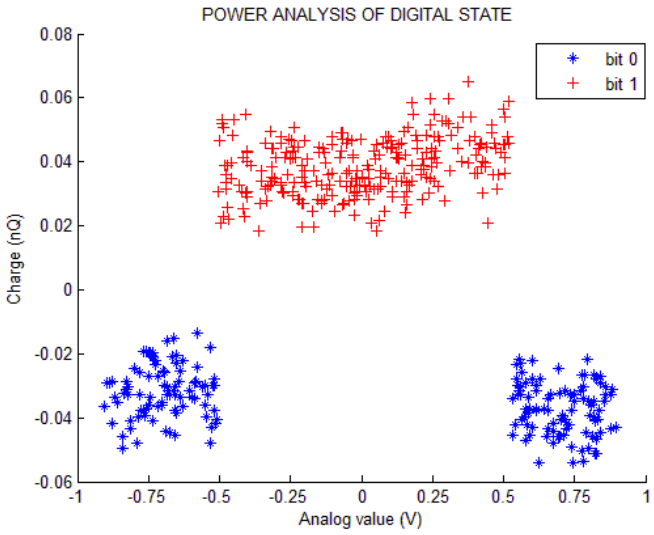


Figure 6.7: Scatter plot of the charge Δq_k required when the successive random bit D_{k+1} is high and low compared with the external (digital) state of the chaotic map $D_{k+1} = Q(x_k)$.

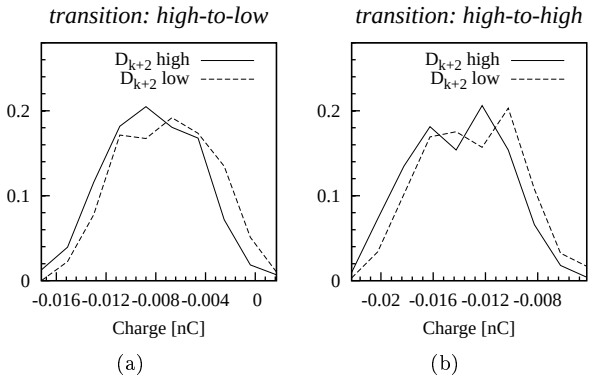


Figure 6.8: Conditional distribution density of the charge Δq_k during a transient, assuming the successive random bit D_{k+2} is high (continuous line) or low (dotted line), in the case of a high-to-low random bit transition (case a) and of an unchanged high random bit (case b, referred to as high-to-high). The distributions are obtained with a 10 bins histogram of frequencies.

according to the value of D_{k+2}. Roughly speaking, given an observed transition $D_k \rightarrow D_{k+1}$ (more precisely, high-to-low transition for the case of Figure 6.8a, and a constant high value for the case of Figure 6.8b) we want to know if the group of $\triangle q_k$ giving rise to $D_{k+2} = 0$ is distributed as the group of $\triangle q_k$ giving rise to $D_{k+2} = 1$.

Obviously, if these two distributions were defined in two different (or slightly overlapping) regions it would be possible to determine with a high probability if the successive random bit D_{k+2} would be high or low from the measure of $\triangle q_k$. On the contrary, as in the case of Figure 6.8, from two distributions almost superimposing it would not be possible to predict if D_{k+2} would be high or low with an accuracy much greater then the one given by pure chance. This means, by definition, that D_{k+2} is unpredictable.

To formally obtain a numerical measure of the similarity of the two obtained distributions, and therefore of the unpredictability of D_{k+2}, we can use the concept of entropy and mutual information.

While the entropy of a random variable X is defined as the average information provided by each of its realization, the average mutual information between two random variables X and Y is defined as the average information provided about a realization of X by the occurrence of the realization of Y [41]. Mathematically, given a discrete random variable X with possible values $x1, x2, \ldots$ and associated probabilities $P_X(x_1), P_X(x_2), \ldots$, its entropy H(X), measured in bit, is defined as

$$H(X) = -\sum_k P_X(x_k) \log_2 P_X(x_k) \tag{6.7}$$

while the average mutual information $I(X;Y)$ between the two discrete ran-

dom variables X and Y is

$$I(X;Y) = \sum_k \sum_j P_{XY}(x_k, y_j) \log_2 \frac{P_{XY}(x_k, y_j)}{P_X(x_k) P_Y(y_j)} \tag{6.8}$$

where P_Y is the probability distribution of Y, and P_{XY} is the joint probability distribution of X and Y .

If we consider D_{k+2} as the random variable X, and $\triangle q_k$ as Y (actually, since $\triangle q_k$ is a continuous random variable, we need to discretize it in a limited number of bins in order to apply the above definition, as already done in Figure 6.8), $I(X;Y)$ is the quantity of average information about D_{k+2} we can get from the observation of $\triangle q_k$, i.e. the indicator of how well $\triangle q_k$ can be used to predict D_{k+2}. Note that, according to this notation, the plots of Figure 6.8 represent the two conditional probability densities $P_{Y|X}$, with $X = 0$ and $X = 1$.

Assuming X and Y are unrelated, $I(X;Y) = 0$; if instead X and Y are completely dependent, $I(X;Y) = H(X) = 1$ bit. In the two cases of the example of Figure 6.8 the mutual information is measured in $I = 0.03$ bits for the high-to-low (low-to-high) transition case, and $I = 0.021$ bits for the high (low) constant value case. These very low values clearly show the lack of mutual information, i.e. the impossibility to retrieve information on D_{k+2} from the observation of $\triangle q_k$.

6.2.3 Conclusions

In this Section power analysis on a prototype of a chaos-based RNG has been considered. Measures on a prototype showed that it is not possible to get information on the internal state either from the observation of the generated bitstream, or from a side-channel attack based on power analysis. The power supply current trace is shown to depend only on the random digital bit, principally for the presence of the buffer driving the chip pad, while a dependency between the current trace and the internal state of the chaotic map cannot be observed.

6.3 Measurements on the leakage based RNG

An RNG based on the p-n junction leakage current is presented in this Section. The proposed RNG is first introduced. Measurements on some test chips are shown to prove its functionality and its randomness.

6.3.1 The proposed RNG

A reverse biased p-n junction (e.g. a n/n+ resistance diffused in a p-type substrate) shows a leakage current (dark current) I_d due to thermally generated minority carriers. A shot noise is produced by such a current:

$$i_d = \sqrt{2qI_d\Delta f} \tag{6.9}$$

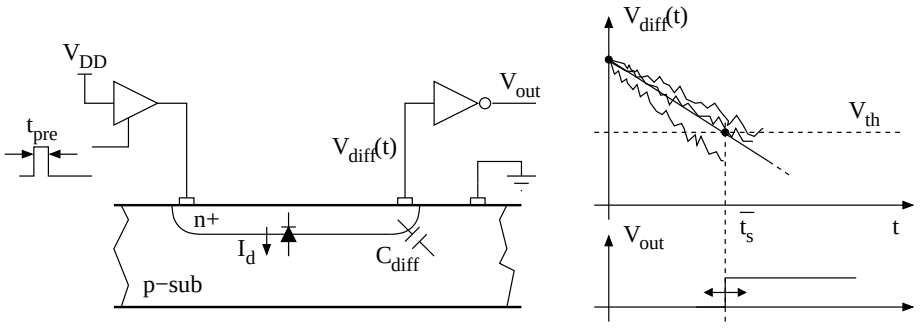


Figure 6.9: Noisy discharge of a reverse biased p-n junction.

where $q = 1.6 \times 10^{-19} C$ is the electron charge.

If a tri-state buffer is used to force a voltage V_{DD} for a time t_{pre} (Figure 6.9), when the buffer is disabled, the capacitance C_{diff} of the depletion region discharges from V_{DD} with the noisy current $(I_d + i_d)$. A CMOS inverter is used to detect when the diffusion voltage $V_{diff}(t)$ reaches the inverter threshold V_{th}. During the discharge time, the shot noise $i_d(t)$ is integrated by C_{diff} thus obtaining a variance for $V_{diff}(t)$ which increases as the integration time increases. As a consequence, the inverter switching time t_s is a random variable whose variance can be exploited to generate a random bit.

As shown in Figure 6.10, two nominally identical diffusions are precharged by two nominally identical drivers and, from the inverter outputs, a random bit is obtained using a DFF as sampling device. Due to process mismatches (e.g. between the diffusions, the driver timings, the inverter thresholds, interconnection delays, etc...), a mismatch in the mean integration times $\overline{t_{s1}}$ and $\overline{t_{s2}}$ is expected which results in an unbalanced bit stream $\{s[i]\}$. In spite of that, it is worth observing that the noise source is stateless since the diffusions are re-charged to V_{DD} thus cancelling the memory of the previously generated bit. As long as the jitter $J_{rms} = \sqrt{2}\sigma_{t_s}$ is larger than the mean integration time mismatch, the output bits are biased but independent and an unbiased bit sequence can be extracted using a proper compression function [49].

However, in order to obtain a design robust against process variations, the alignment of the mean switching times is controlled by the feedback loop shown in Figure 6.11, where the upper inverter output is delayed by $\Delta T_1 = (T_2 - T_1)/2$, whereas the lower one is followed by a variable delay $\Delta T_2 \in [T_1, T_2]$, controlled by an up/down counter which estimates the mean value of the random stream $\{s[i]\}$. A precision δ smaller than J_{rms} is required to adjust ΔT_2. When a random bit is generated, a new charge pulse is required to charge again the diffusion capacitances. The DFF clock signal is used to trigger a new charge pulse thus obtaining a continuous operation.

The mismatch compensation loop makes the noise source non-stationary (non-stateless). However, the resulting random process can be modeled as a discrete-time Markov chain and a post-processor which produces i.i.d. output

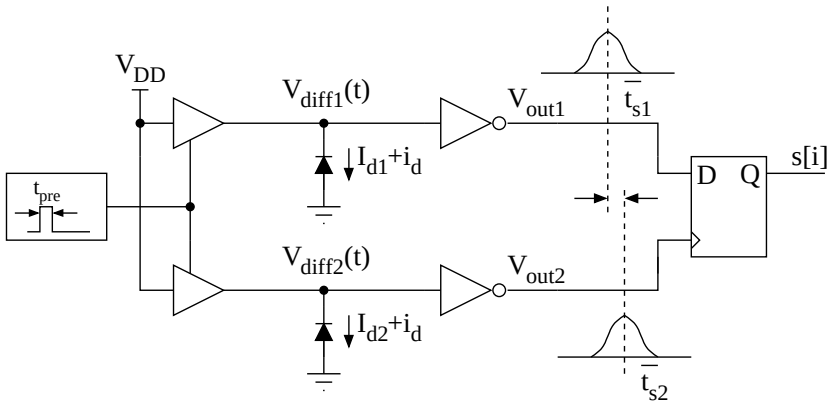


Figure 6.10: Generation of a random bit using two n+ diffusions.

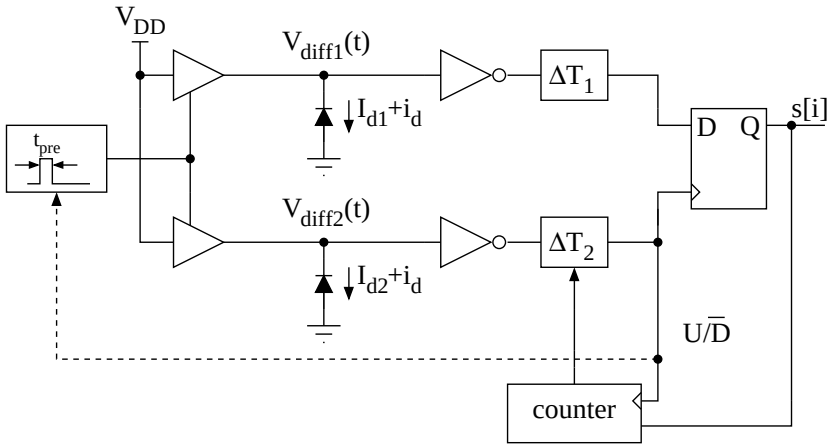


Figure 6.11: Feedback loop to cancel the output sequence bias.

words can be still defined, how it is reported in [27].

The output throughput is fixed by the mean integration time:

$$\overline{t_{s1}} \cong \overline{t_{s2}} = \overline{t_s} = \frac{V_{DD} - V_{th}}{I_d / C_{diff}} \tag{6.10}$$

where C_{diff} is assumed constant between V_{DD} and $(V_{DD} - V_{th})$ and equal to $C_{diff}(V_{DD})$.

Since both I_d and C_{diff} are directly proportional to the diffusion area A_{diff}, the difference $V_{DD} - V_{th}$ is the only circuit parameter which can be used to control the data-rate. For the variance of the diffusion voltage at the end of the integration time t_s, it holds [98]:

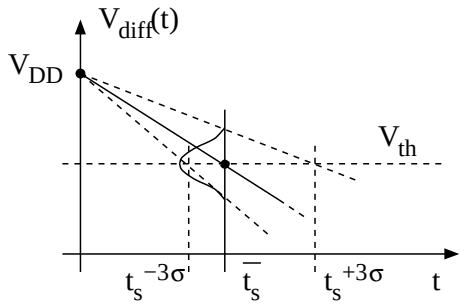


Figure 6.12: Integration time variance.

$$\sigma^2_{V_{diff}} = \frac{qI_d}{C^2_{diff}} t_s \tag{6.11}$$

where $q = 1.6 \times 10^{-19} C$ is the electron charge.
Being $\overline{t_s} \gg \sigma_{t_s}$, from (6.11) it follows:

$$\sigma^2_{V_{diff}} \cong \frac{qI_d}{C^2_{diff}} \overline{t_s}. \tag{6.12}$$

As shown in Figure 6.12, the integration time has a non-symmetrical probability distribution and it holds:

$$t_s^{+3\sigma} - \overline{t_s} = \frac{3\sigma_{V_{diff}}}{V_{DD} - V_{th} - 3\sigma_{V_{diff}}} \overline{t_s} \tag{6.13}$$

$$\overline{t_s} - t_s^{-3\sigma} = \frac{3\sigma_{V_{diff}}}{V_{DD} - V_{th} + 3\sigma_{V_{diff}}} \overline{t_s}. \tag{6.14}$$

However, being $\sigma_{V_{diff}} \ll V_{DD} - V_{th}$, it follows $t_s^{+3\sigma} - \overline{t_s} \cong \overline{t_s} - t_s^{-3\sigma}$ and the available jitter is

$$J_{rms} \cong \sqrt{2} \frac{\sigma_{V_{diff}}}{V_{DD} - V_{th}} \overline{t_s}. \tag{6.15}$$

From (6.10), (6.12) and (6.15) it follows:

$$J_{rms} = \frac{\sqrt{2C_{diff} q (V_{DD} - V_{th})}}{I_d}. \tag{6.16}$$

Using the values in Table 6.1, where the leakage current I_d is a measured data, a bit generation time $\overline{t_s} = 0.84ms$ and a rms jitter $J_{rms} = 2.6\mu s$ is obtained. Since J_{rms} is less than 1% of $\overline{t_s}$, the control loop is actually necessary to make the design robust. On the other hand, the available jitter is large enough to request only a coarse-grained delay line whose design is not critical.

The obtained data-rate (about 1.2Kb/s) is quite low but sufficient in most cryptographic applications. Moreover, the circuit is extremely compact and multiple instances can be used if a higher throughput is required.

Table 6.1: Parameters for a sample design in a $0.12\mu m$ CMOS process.

Parameter	Symbol	Value
Power supply	V_{DD}	$1.2V$
Inverter threshold	V_{th}	$0.8V$
Depletion capacitance @V_{DD}	C_{diff}	$84fF$
Leakage (dark) current	I_d	$40pA$

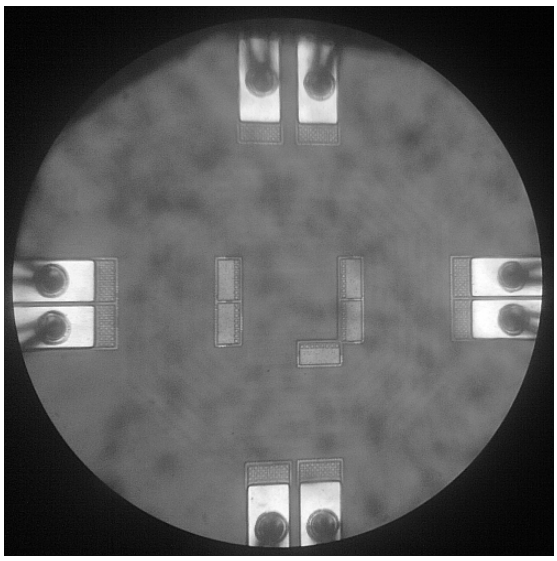


Figure 6.13: Testchip micro-photograph.

6.3.2 Testchip design

A prototype testchip has been designed and manufactured in a $0.12\mu m$ CMOS process from Infineon Technologies (a micro-photograph is shown in Figure 6.13).

Post-processing and compression are based on a hash function implemented by means of a 32-bit LFSR. Even if not strictly required since the LFSR is not operated in free evolution, a primitive polynomial is adopted having care to choose a polynomial with a spread distribution of its terms. That provides a better diffusion after the reset.

In order to estimate on-line the entropy of the generated bit sequence, the concept of *deterministic model* of the noise source was introduced in [27], namely, a model which describes the source behavior under the hypothesis that the stochastic part of the system is neglected.

Due to the re-charge to V_{DD} of both diffusions, the adopted noise source has a straightforward deterministic behavior. In particular, if $J_{rms} = 0$ (i.e. $J_{rms} \ll \delta$), the device produces the periodic sequence "010101..." resulting from the

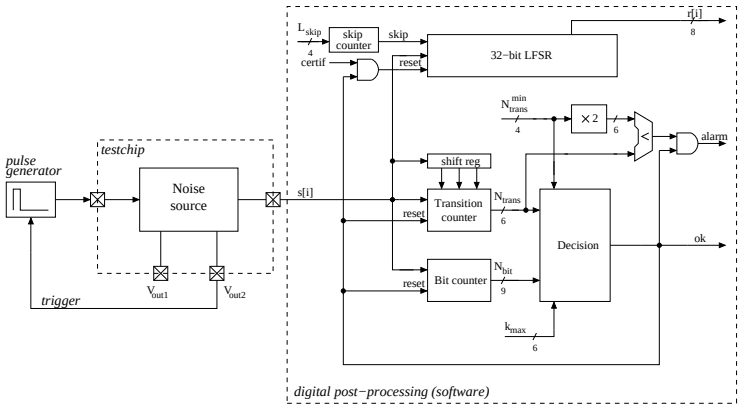


Figure 6.14: RBG testchip and digital post-processing.

mismatch compensation loop. Other potential deterministic behaviors can arise because of transients or failures in the feedback loop. In this case, a constant sequence stuck at one or zero is expected. The entropy is estimated by counting the number of transitions in the raw stream $\{s[i]\}$ that are unexpected according to the discussed deterministic model. As in [27], the following transition counter is adopted:

$$N_{trans} = \sum_i (s[i] \oplus s[i-2]) \cdot (s[i] \oplus s[i-1] \oplus s[i-2] \oplus s[i-3]). \qquad (6.17)$$

As depicted in Figure 6.14, the entropy estimator includes the transition counter (6.17), a bit counter N_{bit} and a *decision* algorithm to signal that the required amount of random bits has been generated by the source. The *ok* reset both counters to start the generation of a new byte and, if the signal *certif* is active, the LFSR is reset as well (*certification* mode).

Two configuration parameters are used: the maximum compression ratio k_{max} after which the output byte is delivered anyway, and the minimum number of transitions on the bit stream N_{trans}^{min} to be counted before releasing a new output byte. In Figure 6.14, N_{trans}^{min} is multiplied by 2 (i.e. the transitions counted by (6.17) in a maximal entropy byte) thus implicitly fixing a minimum compression $k_{min} = N_{trans}^{min}$. The output byte is released if $k = k_{max}$ even if the required number of transitions has not been reached yet and, in this case, an *alarm* is raised. Therefore, a time-out is implemented, as requested in a real application to avoid software deadlocks.

Additionally, a minimum number of input bits $N_{bit}^{min} = 8 \cdot N_{trans}^{min}$ is requested too. In other terms, the input sequence is compressed at least how much a maximal entropy sequence would be. As a further condition, always an even number of bits are processed, as discussed in [27].

Algorithm 6.1 Decision algorithm.

```
decision (in: N_trans, N_bit, N_trans^min, k_max;
          out: ok) {
   if (k_max > 0)&((N_bit ≥ 8 · k_max) |
      (N_trans ≥ 2 · N_trans^min))&(N_bit ≥ 8 · N_trans^min)&
      (N_bit mod 2 == 0) then
    ok = 1;
   else ok = 0;
  }
```

Table 6.2: Measured mean integration time mismatch and relative jitter.

$Chip\#$	$\overline{t_{s2}} - \overline{t_{s1}}[\mu s]$	$J_{rms}[\mu s]$
1	-256	4.95
2	372	11.8
3	-422	19.1
4	-522	15.6
5	415	26.4
6	194	6.32

6.3.3 Experimental results

A first set of measurements has been performed to evaluate the available jitter J_{rms} and the mismatch in the mean integration times $\overline{t_{s1}}$ and $\overline{t_{s2}}$. In Figure 6.15, the two output voltages V_{out1}, V_{out2} (trace 1 and 2, respectively) and their difference (trace M) are visible. By measuring average and standard deviation of the pulse width in M, mismatch $\overline{t_{s2}} - \overline{t_{s1}}$ and jitter J_{rms} have been obtained.

The measured values for six dies from different wafers (different process corners) are reported in Table 6.2. It can be observed that the measured jitter is always larger than the expected value derived from (6.16). This results from the fact that the leakage current reported in Table 6.1 is a worst case value (with respect to process variations). The values in Table 6.2 confirm that the relative jitter is a fraction of the mean integration time mismatch and, therefore, the bias cancellation feedback loop in Figure 6.11 is actually needed.

Raw bit sequences $\{s[i]\}$ for each tested die have been collected and the estimated coefficients of the autocorrelation function $R_{SS}[j]$, for $j = 1, 2, \ldots, 10$, are evaluated as in (6.18) and reported in Table 6.3. It is worth noting that, since $J_{rms} \gg \delta$, the effect of the feedback loop on the autocorrelation coefficients is negligeble.

$$R_{SS}[j] = \frac{E\{(S[i] - \mu) \cdot (S[i+j] - \mu)\}}{\sigma^2} \tag{6.18}$$

In order to evaluate the entropy of the post-processed random bytes $r[i]$, raw sequences of 10×10^6 bits have been generated for each of the tested dies and the corresponding random bytes have been calculated. The post-processor

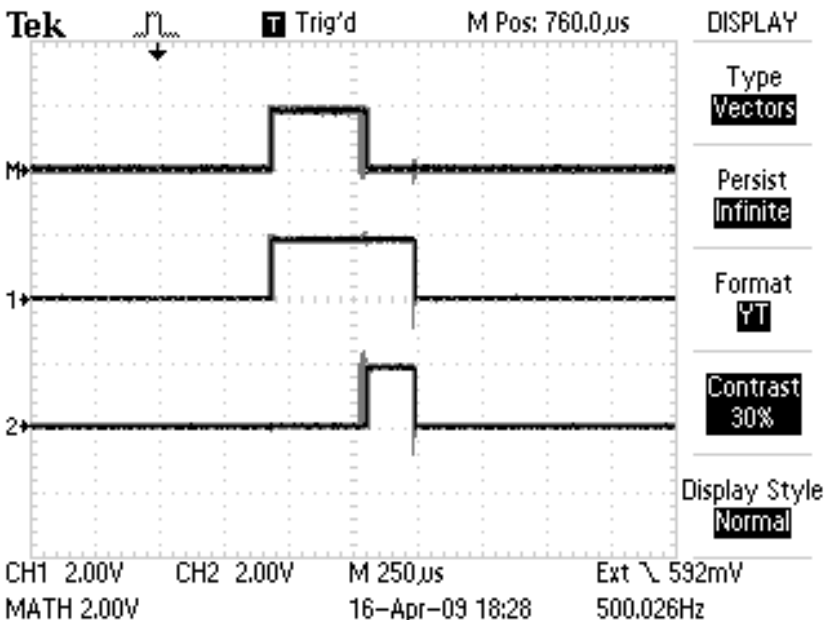


Figure 6.15: Oscilloscope screenshot (chip #20).

Table 6.3: Correlation coefficients.

Chip #	$r[1]$	$r[2]$	$r[3]$	$r[4]$	$r[5]$
1	−0.075	−0.065	−0.055	−0.047	−0.038
2	−0.033	−0.031	−0.029	−0.028	−0.025
3	−0.020	−0.020	−0.019	−0.020	−0.017
4	−0.025	−0.024	−0.022	−0.023	−0.021
5	−0.014	−0.015	−0.014	−0.015	−0.013
6	−0.061	−0.053	−0.047	−0.041	−0.036
Chip #	$r[6]$	$r[7]$	$r[8]$	$r[9]$	$r[10]$
1	−0.035	−0.029	−0.024	−0.020	−0.018
2	−0.024	−0.022	−0.021	−0.019	−0.019
3	−0.017	−0.017	−0.015	−0.015	−0.015
4	−0.020	−0.019	−0.017	−0.016	−0.017
5	−0.013	−0.013	−0.012	−0.012	−0.012
6	−0.032	−0.029	−0.025	−0.021	−0.019

Table 6.4: χ^2 test and entropy estimate in *certification* mode.

$Chip\#$	N_{trans}^{min}	k_{max}	L_{skip}	N_r	$\overline{k}$	THR(kb/s)	χ^2	$\widehat{H}$
1	4	32	16	1199701	6.25	0.19	260.28	7.99984
2	4	32	16	1199690	6.25	0.19	254.43	7.99985
3	4	32	16	1199725	6.25	0.19	284.52	7.99983
4	4	32	16	1199732	6.25	0.19	218.99	7.99987
5	4	32	16	1199853	6.25	0.19	282.04	7.99983
6	4	32	16	1199672	6.36	0.18	260.26	7.99984

is operated in certification mode, configured to release a byte after counting 8 transitions in the raw stream ($2 \times N_{trans}^{min} = 8$) and a maximum compression ratio $k_{max} = 32$ is chosen. For the number of raw bits to be skipped between two input sub-sequences we chose the conservative setting $L_{skip} = 16$, thus obtaining a mean compression $\overline{k} = 6$ for an ideal sequence.

Since the processed bytes $r[i]$ have been proved to be independent, their distribution can be tested using Pearson's chi-square test (χ^2) with 255 degrees of freedom (confidence level $\alpha = 0.05$):

$$\sum_{i=0}^{255} \frac{(N_r \cdot P_r(i) - N_r/256)^2}{N_r/256} < \chi^2_{0.95}(255) = 293.25 \tag{6.19}$$

where $P_r(i)$ is the empirical estimate of the distribution of the output bytes $r[i]$ and N_r is the number of generated output bytes $r[i]$. The obtained results are summarized in Table 6.4, which shows that the collected sequences pass the applied test. An estimate of the entropy per byte is also reported using the plug-in estimate:

$$\widehat{H} = -\sum_i P_r(i) \log P_r(i). \tag{6.20}$$

For the final throughput after compression, it holds:

$$THR = \frac{1}{\overline{t_s} \cdot \overline{k}} \tag{6.21}$$

where $\overline{k}$ is the average post-processor compression (including the 16 skipped bits) and $\overline{t_s} = 0.84ms$.

The quality of the post-processed bytes in certification mode has been also tested using the statistical tests defined in the AIS 31 publication for random bit generators belonging to the functionality class P2 [55]. The obtained results are reported in Table 6.5 and they are largely within the acceptance ranges, thus confirming the results in Table 6.4.

As a final verification, the AIS 31 tests have been repeated on a sequence of processed bytes collected operating the post-processor in normal mode (post-processing reset disabled). As discussed in [27], since a linear post-processing is employed, the entropy is expected to be not lower than what is measured when

Table 6.5: AIS 31 statistical tests (functionality class P2) in *certification* mode.

Test	Description	Acceptance range	*Chip* #1	*Chip* #2	*Chip* #3
P2.i.vii.a	one-dimensional distribution	< 0.025	0.0012	0.0002	0.0003
P2.i.vii.b	one-step transition distribution	< 0.020	0.0012	0.0021	0.0004
P2.i.vii.c	2-step dependency test	< 15.13	0.1066	1.6474	0.7683
P2.i.vii.d	3-step dependency test	< 15.13	3.4778	6.3394	4.4935
P2.i.vii.e	Coron test	> 7.976 && < 8.026	8.0027	8.0006	7.9994
Test	Description	Acceptance range	*Chip* #4	*Chip* #5	*Chip* #6
P2.i.vii.a	one-dimensional distribution	< 0.025	0.0018	0.0009	0.0025
P2.i.vii.b	one-step transition distribution	< 0.020	0.0004	0.0005	0.0025
P2.i.vii.c	2-step dependency test	< 15.13	0.1883	2.1517	2.9338
P2.i.vii.d	3-step dependency test	< 15.13	1.0858	1.6937	1.8242
P2.i.vii.e	Coron test	> 7.976 && < 8.026	7.9950	8.0023	8.0007

Table 6.6: AIS 31 statistical tests (functionality class P2) in *normal* mode.

Test	Description	Acceptance range	*Chip* #1	*Chip* #2	*Chip* #3
P2.i.vii.a	one-dimensional distribution	< 0.025	0.0001	0.0013	0.0029
P2.i.vii.b	one-step transition distribution	< 0.020	0.0003	0.0002	0.0029
P2.i.vii.c	2-step dependency test	< 15.13	1.3834	0.7683	0.2509
P2.i.vii.d	3-step dependency test	< 15.13	2.2849	1.3005	0.9418
P2.i.vii.e	Coron test	> 7.976 && < 8.026	8.0056	7.9997	7.9987
Test	Description	Acceptance range	*Chip* #4	*Chip* #5	*Chip* #6
P2.i.vii.a	one-dimensional distribution	< 0.025	0.0013	0.0002	0.0023
P2.i.vii.b	one-step transition distribution	< 0.020	0.0008	0.0041	0.0017
P2.i.vii.c	2-step dependency test	< 15.13	3.5617	2.2445	1.9220
P2.i.vii.d	3-step dependency test	< 15.13	2.2984	4.9601	1.4906
P2.i.vii.e	Coron test	> 7.976 && < 8.026	7.9999	8.0000	8.0023

operating the RBG in certification mode. The results in Table 6.6 confirm the assumption.

6.3.4 Conclusions

A noise source for a random bit generator has been proposed which exploits the shot noise associated to the leakage current of a reverse biased p-n junction. A model of the source is discussed and a sample design in a $0.12\mu m$ CMOS process has been presented which is expected to provide a 1.2kb/s high quality random bit stream with limited area and power requirements.

Chapter 7

Conclusions

Cryptographic devices have become essential building blocks of many systems that we daily use. Consequently, the research on security topics became more and more important and today it covers different aspects of human sciences at different levels. Many works on the implementation of new kind of attacks and, at the same time, new countermeasures have been reported in the technical literature. In 1996 Kocher suggested the idea that the secret key of a cryptographic device could be discovered by using some physical quantities indirectly correlated with the key. The dynamic power consumption is the side-channel extensively used in literature. The analysis (directly or by statistical evaluations) of power traces of the cryptographic device under attack is the foundation of the so called power analysis attack. Consequently, the SPA and much more the DPA constitute the major threat to the cryptographic devices. On the other side a number of software and hardware countermeasures have been proposed to prevent power analysis attacks on chipcards. DPA-resistant logic styles are a class of hardware countermeasures that is implemented at the cell level. The basic idea is to build logic cells with a power consumption that is independent of the processed logic values.

This thesis work is focused on both aspects by proposing new ideas about the power attacks and different techniques to counteract these attacks.

A circuit, named SCM, to improve the DPA attack has been proposed as first activity in Chapter 3. The SCM provides a low impedance current input and an additional DC feedback loop to control the voltage at the input pin, thus supplying the device under attack with a stable voltage. From the analysis and simulation of the SCM, several advantages have been showed with respect to the standard resistor-based setup. These advantages have been confirmed by measuring the current consumption of an FPGA implementing a part of Serpent algorithm.

Compared to a resistor-based measurement, the presented circuit shows a 20dB improvement in the sensitivity to the current consumption variations of a device under attacks. Finally, an attack has been implemented on the adopted

case study and on a DES realization showing a noticeable improvement if the SCM is used.

After proposing a novel circuit to improve DPA, a novel side-channel has been investigated in Chapter 4. Since in sub-100nm technologies leakage current is becoming more predominant in the total power consumption, a power attack based on leakage current (named LPA) has been proposed. A model in which leakage is linearly related to the Hamming weight of the signal under attack has been proposed and experimentally evaluated. A systematic attack procedure based on the correlation coefficient evaluation has been introduced. Experimental issues involved in each step of this procedure have been discussed. Moreover, due to the strong leakage sensitivity to temperature, it was shown that the operating temperature must be kept constant during the LPA attack. It was also shown that LPA attacks are extremely simple and require inexpensive equipment (even cheaper than that used in traditional power analysis attacks based on the dynamic power consumption). An experimental LPA attack to a simple circuit has been performed for the first time. Results confirm the validity of the underlying assumption, and the effectiveness of this kind of attacks. Moreover analysis has shown that the Hamming weight as leakage power model is not restricted to bit sliced circuits, but can also be used for combinational circuits (e.g., S-Boxes). In regard to the impact of process variations, Monte Carlo simulations have shown that intra-die variations can influence the outcome of LPA attacks. In particular, the analysis of a simple crypto core in standard CMOS logic allowed to successfully infer the secret key on 94% of sample circuits. This probability of success drops to 21% in crypto cores implemented in WDDL, and to 24% in TDPL logic style. The percentage of success has been found to respectively rise to 99%, 40% and 38% for CMOS, WDDL and TDPL, if the adversary accepts to broaden the search of the secret key to those leading to the two highest correlation coefficients. Results demonstrate that, even if transistor-level countermeasures developed for DPA have some beneficial effects in preventing LPA attacks, the success rate for LPA attacks is intolerably high (many orders of magnitude greater than levels of security that are commonly required). This means that the body of work focused on countermeasures to DPA attacks does not solve the security issues arising with LPA attacks. Hence, in the future a significant research effort will be required to devise appropriate solutions and countermeasures against LPA that keep the security level to the current standards.

DPA-resistant logic styles are a class of hardware countermeasures that is implemented at the cell level. A DPA-resistant logic family is based on the idea that each build logic cell must have a power consumption that is independent of the processed logic values.

In Chapter 5 three different logic styles to counteract power analysis have been proposed. The 3sDDL is a dual rail precharge logic with a precharge value at $V_{DD}/2$. The figures of merit considered in literature to prove the efficiency of a DPA-resistant logic style, NED and NSD, have been taken into account

showing an improvement with respect to the standard CMOS and the others secure logic styles.

All DRP gates proposed in literature for cryptographic applications have a power consumption independent on processing data under the ideal assumption that the load at two differential outputs are perfectly matched. This means that all proposed DPA-resistant logic family have a leakage of information. The second logic style proposed in this thesis work, named TDPL, addresses this problem by introducing a third phase to the two normally used in the DRP logic gates. A complete set of cells has been proposed and a TDPL flip-flop has been designed. Experimental results on a case study in a 65nm CMOS process confirm that TDPL shows a constant energy consumption even in presence of unbalanced loads. NED and NSD show an improvement in the energy consumption balancing in excess of 10 times with respect to the corresponding SABL implementation without requiring any constraint on the geometry of the complementary wires.

The main drawback of TDPL logic style is represented by the presence of the additional two control signals needed to set the three phases which make more difficult the placing and routing. The third proposed logic style is the DDPL which is based on a completely novel encoding concept where the information is represented in the time domain rather than in the spatial domain as in a standard dual-rail logic. DDPL permits to have the same improvements of the TDPL with respect to the others DPA-resistant logic styles (and thus a DDPL cell has a power consumption which is insensitive to unbalanced load conditions) but with only one control signal, as in the standard DRP logic styles. Much attention has been placed in the design of the DDPL sequential blocks. Complete case studies have been proposed showing that the simulated energy consumption per cycle is up to 50 times more balanced than in the corresponding SABL gates without requiring any constraint on the geometry of the complementary wires. A main advantage of DDPL logic style is that the introduced time domain data encoding allows to set the DPA-resistance independently from the operating frequency by choosing the delay parameter according to the expected resolution of current consumption measurements.

Finally, in Chapter 6, two works on RNG prototypes are presented. In the first part of this Chapter, power analysis on a prototype of a chaos-based RNG has been proposed. Measurements prove that the considered RNG is sensitive to power analysis because of digital output drivers. Experimental results proved that power consumption is not related to the internal state but only to the externally generated bit. In the second part of Chapter 6 a leakage based RNG is presented. Measurements on six samples of the RNG and post-processing tests of the acquired output data have proved its functionality and randomness.

List of publications

Journals

- M. Alioto, L. Giancane, G. Scotti, A. Trifiletti, "Leakage Power Analysis Attacks: a Novel Class of Attacks to Nanometer Cryptographic Circuits", IEEE Transactions on Circuits and Systems I, Volume: 57 , Issue: 2, Publication Year: 2010 , Page(s): 355 – 367.
- M. Bucci, L. Giancane, R. Luzzi, G. Scotti and A. Trifiletti, "Delay-Based Dual-Rail Pre-Charge Logic", accepted for publication at Transactions on Very Large Scale Integration (VLSI) Systems.
- M. Bucci, L. Giancane, R. Luzzi, M. Marino, G. Scotti, A. Trifiletti, "Enhancing power analysis attacks against cryptographic devices", IET Circuits, Devices & Systems, Volume 2, Issue 3, June 2008 Page(s): 298 – 305.
- M. Bucci, L. Giancane, R. Luzzi and A. Trifiletti, "A Flip-Flop for the DPA Resistant Three-Phase Dual-Rail Pre-Charge Logic Family", submitted to Transactions on Very Large Scale Integration (VLSI) Systems.
- M. Djukanovic, L. Giancane, G. Scotti, A. Trifiletti, M. Alioto, "Analysis of LPA Effectiveness on DPA Resistant Logic Styles under Process Variations", submitted to Microelectronics Journal.
- M. Bucci, L. Giancane, R. Luzzi and A. Trifiletti, "A Leakage-Based Random Bit Generator", submitted to IEEE Transactions on Circuits and Systems II.

International Conferences

- L. Giancane, P. Marietti, M. Olivieri, G. Scotti and A. Trifiletti, "A New Dynamic Differential Logic Style as a Countermeasure to Power Analysis Attacks", 14th IEEE International Conference On Electronics, Circuits, And Systems (ICECS 2008).

- F. Pareschi, G. Scotti, L. Giancane, R. Rovatti, G. Setti and A. Trifiletti, "Power Analysis of a Chaos-Based Random Number Generator for Cryptographic Security", 2009 IEEE International Symposium on Circuits and Systems (ISCAS 2009), 24-27 May 2009, pp. 2858 – 2861.
- M. Djukanovic, L. Giancane, G. Scotti and A. Trifiletti, "Impact of Process Variations on LPA Attacks Effectiveness", 2nd IEEE International Conference On Computer and Electrical Engineering (ICCEE 2009), Dubai, UAE, 28 - 30 December 2009.
- M. Alioto, L. Giancane, G. Scotti and A. Trifiletti, "Leakage Power Analysis Attacks: Well-Defined Procedure and First Experimental Results", 21th IEEE International Conference on Microelectronics (ICM 2009), Marrakech, Marocco, 19-22 December 2009.
- M. Bucci, L. Giancane, R. Luzzi, G. Scotti and A. Trifiletti, "Delay-Based Dual-Rail Pre-Charge Logic", 16th IEEE International Conference on Electronics, Circuits, and Systems (ICECS 2009), Medina Yasmine Hammamet, Tunisia, 13-16 December 2009.
- M. Alioto, L. Giancane, G. Scotti and A. Trifiletti, "Leakage Power Analysis Attacks: Theoretical Analysis and Impact of Variations", 16th IEEE International Conference on Electronics, Circuits, and Systems (ICECS 2009), Medina Yasmine Hammamet, Tunisia, 13-16 December 2009.
- M. Alioto, M. Djukanovic, L. Giancane, G. Scotti and A. Trifiletti, "Leakage Power Analysis Attacks: Effectiveness on DPA Resistant Logic Styles under Process Variations", accepted as lecture at 2011 IEEE International Symposium on Circuits and Systems (ISCAS 2011).

Bibliography

[1] S. Aumonier, "Generalized Correlation Power Analysis", in Proc. of Crypto'07, Krakow (Poland), Sept. 2007.

[2] D. Agrawal, B. Archambeault, S. Chari, J. R. Rao and P. Rohatgi, "Advances in Side-Channel Cryptanalysis", RSA Laboratories Cryptobytes, Vol. 6, number 1, pages 20-32, 2003.

[3] Anderson R., Biham E., and Kundsen L., "A proposal for the Advanced Encryption Standard", AES proposal, 1998, available at http://www.cl.cam.ac.uk/ftp/users/rja14/serpent.pdf.

[4] M.W. Allam, M.I. Elmasry, "Dynamic Current Mode Logic (DyCML): a New Low-Power High-Performance Logic Style", IEEE Journal of Solid State Circuits, vol 36, no. 3, pp 550-558, March 2001.

[5] A. Abdollahi, F. Fallah, M. Pedram "Leakage current reduction in CMOS VLSI circuits by input vector control", IEEE Trans. on VLSI Systems, vol. 12, no. 2, 2004, pp. 140-154.

[6] M.Aigner, S. Mangard, R. Menicocci, M. Olivieri, G. Scotti, A. Trifiletti, "A novel CMOS Logic style with data independent power consumption", IEEE International Symposium on Circuits and systems, Kobe, May 2005, pp. 1066-1069.

[7] Aigner M., and Oswald E., "Power analysis tutorial", available at http://www.iaik.tugraz.at.

[8] M. Alioto, M. Poli, S. Rocchi, "A General Model for Differential Power Analysis Attacks to Static Logic Circuits", in Proc. of ISCAS 2008, pp. 3346-3349, Seattle (USA), May 2008.

[9] M. Alioto, M. Poli, S. Rocchi, "A general power model of Differential Power Analysis attacks to static logic circuits", IEEE Trans. on VLSI Systems, vol. 18, no. 5, pp. 711-724, May 2010.

[10] M. Alioto, M. Poli, S. Rocchi, V. Vignoli, "Power Modeling of Precharged Address Bus and Application to Multi-bit DPA Attacks to DES Algorithm", in Proc. of PATMOS 2006, pp. 593-602, Montpellier (France), Sept. 2006.

[11] M. Alioto, M. Poli, S. Rocchi, V. Vignoli, "A General Model of DPA Attacks to Precharged Busses in Symmetric-Key Cryptographic Algorithms", in Proc. of ECCTD 2007, pp. 368-371, Sevilla (Spain), Aug. 2007.

[12] Agrawal, J. R. Rao and P. Rohatgi, "Multi-channel attacks", in C. Walter, Ç. K. Koç and C. Paar, editors, Proceedings of 5th International Workshop on Cryptographic Hardware and Embedded Systems (CHES), number 2779 of LNCS, pages 2-16, 2003, Springer-Verlag.

[13] M. J. Bellido, A.J. Acosta, M. Valencia, A. Barriga, J.L. Huertas, "A simple binary random number generator: New approaches for CMOS VLSI", in Proc. 35th Midwest Symp. Circuits Syst., Aug. 1992, vol. 1, pp. 127–129.

[14] V. Bagini and M. Bucci, "A design of reliable true random number generator for cryptographic applications", Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES 99) Heidelberg, Germany: Springer-Verlag, 1999, vol. 1717, pp. 204-218.

[15] H. Bock, M. Bucci, and R. Luzzi, "An offset-compensated oscillator based random bit source for security applications", Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES '04), Lecture Notes on Computer Science, Springer-Verlag, vol. 3156, pp. 268-281, 2004.

[16] E. Brier, C. Clavier, F. Olivier, "Correlation Power Analysis with a Leakage Model", in Proc of CHES 2004, Lecture Notes in Computer Science vol. 3156, pp. 16-29, Boston, Aug. 2004.

[17] D. Boneh, R. A. DeMillo and R. J. Lipton, "On the Importance of Checking Cryptographic Protocols for Faults", in W. Fumy, editor, Advances in Cryptology: Proceedings of EUROCRYPT '97, number 1233 of LNCS, pages 37-51, Springer-Verlag.

[18] M. Bucci, M. Guglielmo, R. Luzzi and A. Trifiletti, "A Power Consumption Randomization Countermeasure for DPA-Resistant Cryptographic Processors", Proc. Int.l Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS '04), Lecture Notes in Computer Science, vol. 3254, Springer-Verlag, pp. 481-490, 2004.

[19] M. Bucci, M. Guglielmo, R. Luzzi, A. Trifiletti, "A Countermeasure against Differential Power Analysis based on Random Delay Insertion", Circuits and systems 2005, ISCAS 2005, Vol. 4, pp. 3547 – 3550, 2005.

[20] M. Bucci, L. Giancane, R. Luzzi and A. Trifiletti, "Three-Phase Dual-Rail Pre-charge Logic", Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES '06), Lecture Notes in Computer Science, vol. 4249, Springer-Verlag, pp. 232-241, 2006.

[21] M. Bucci, L. Germani, R. Luzzi, A. Trifiletti, M. Varanonuovo, "A High-Speed Oscillator-Based Truly Random Number Generator for Cryptographic Applications on a Smart Card IC", IEEE Trans. Computers, vol. 52, no. 4, pp. 403-409, Apr. 2003.

[22] M. Bucci, L. Germani, R. Luzzi, P. Tommasino, A. Trifiletti, M. Varanonuovo, "A high-speed IC random-number source for SmartCard microcontrollers", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, Vol. 50, Issue 11, Nov. 2003 pp. 1373–1380.

[23] M. Bucci, L. Giancane, R. Luzzi, A. Trifiletti, M. Varanonuovo, "A novel concept for stateless random bit generators in cryptographic applications", Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on 21-24 May 2006, pp. 317-320.

[24] M. Bucci and R. Luzzi, "Design of Testable Random Bit Generators", Proc. Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES '05), Lecture Notes on Computer Science, Springer-Verlag, vol. 3659, pp. 147-156, 2005.

[25] M. Bucci and R. Luzzi, "A Leakage-Based Random Bit Generator with On-line Fault Detection", Design and Diagnostics of Electronic Circuits and systems, 2006 IEEE April 18-21, 2006, pp.232-233.

[26] M. Bucci and R. Luzzi, "A testable random bit generator based on a high resolution phase noise detection", in Proc. 10th IEEE Workshop Design Diagnosti. Electron. Circuits Syst. (DDECS'07), 2007, pp. 23–27.

[27] M. Bucci and R. Luzzi, "Fully Digital Random Bit Generators", IEEE Trans. Circuits and Systems, vol. 55, no. 3, pp. 861-875, April 2008.

[28] L. Benini, E. Omerbegovic, A. Macii, M. Poncino, E. Macii, F. Pro, "Energy-aware design techniques for differential power analysis protection", Proc. Design Automation Conf. (DAC '03), pp. 36-41, 2003.

[29] http://www-device.eecs.berkeley.edu /~bsim3/bsim4.html.

[30] J. Coron, "Resistance Against Differential Power Analysis for Elliptic Curve Cryptosystems", Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES '99), Lecture Notes in Computer Science, vol. 1717, Springer-Verlag, pp. 292-302, 1999.

[31] Clavier C., Coron J., and Dabbous N., "Differential Power Analysis in the Presence of Hardware Countermeasures", Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES '00), Lecture Notes in Computer Science, vol. 1965, Springer-Verlag, 2000, pp. 252-263.

[32] Chari, S. Jutla, C. Rao, J. Rohatgi, "Towards sound approaches to counteract power-analysis attacks", Proc. Advances in Cryptology – Crypto'99, volume 1666 of Lecture Notes in Computer Science, pp. 398-412, Springer-Verlag, 1999.

[33] J.S. Coron, P. Kocher and D. Naccache, "Statistics and Secret Leakage", in Financial Cryptography '00, 2000.

[34] S. Chari, J. Rao, P. Rohatgi, "Template Attacks", in the Proceedings of CHES 2002, LNCS, vol. 2523, pp. 13-28, CA, USA.

[35] S. Callegari, R. Rovatti, and G. Setti, "Embeddable ADC-Based True Random Number Generator for Cryptographic Applications Exploiting Nonlinear Signal Processing and Chaos", in IEEE Transaction on Signal Processing, vol. 53, no. 2, pp. 793–805, Feb. 2005.

[36] Z. Chen, L. Wei, M. Johnson, K. Roy "Estimation of standby leakage power in CMOS circuits considering accurate modelling of transistor stacks", in Proc. of ISLPED 1998, pp. 239-24 Monterey, Aug. 1998.

[37] National Bureau of Standards, "Data Encryption Standard", U.S. Department of Commerce, FIPS pub. 46, Jannuary 1977.

[38] M. Dichtl and N. Janssen, "A high quality physical random number generator", in Proc. Sophia Antipolis Forum Microelectron. (SAME'00), 2000, pp. 48–53.

[39] M. Eisele, J. Berthold, D. Schmitt-Landsiedel, R. Mahnkopf, "The impact of Intra-Die Device Parameter Variations on Path Delays and on the Design for Yield of Low Voltage Digital Circuits", IEEE Trans. on VLSI Systems, vol. 5, no. 4 pp. 360-368, December 1997.

[40] M. Epstein, L. Hars, R. Krasinski, M. Rosner, and H. Zheng, "Design and implementation of a true random number generator based on digital circuit artifacts", in Proc. 5th Workshop Cryptograph. Hardware Embed. Syst. (CHES'03), Heidelberg, Germany, 2003, vol. 2779, pp. 152–165.

[41] R. G. Gallager, "Information theory and reliable communication", Wiley and Sons, 1968.

[42] K. Gandolfi, C. Mourtel and F. Olivier, "Electromagnetic Analysis: Concrete Results", in Ç. K. Koç, D. Naccache and C. Paar, editors, Proceedings of 3rd International Workshop on Cryptographic Hardware and Embedded Systems (CHES), number 2162 of LNCS, pages 251-261, 2001, Springer-Verlag.

[43] J. Dj. Golic and R. Menicocci, "Universal masking on logic gate level", Electronics Lett., vol. 40, no. 9, April 2004.

[44] L. Giancane, M. Jovanovich, G. Scotti, A. Trifiletti, "Leakage power analysis of cryptographic devices implemented in nanometer CMOS technologies", Konferencija 9-a 07: Konferenciju za Elektroniku, Telekomunikacije, Racunarstvo, Automatiku I Nuklearnu Tehniku, Herceg Novi, Igalu (Montenegro), 4-8 June 2007.

[45] J. Giorgetti, G. Scotti, A. Simonetti, A. Trifiletti, "Analysis of Data Dependence of Leakage Current in CMOS Cryptographic Hardware", in Proc. of Great Lake Symposium on VLSI (GSLVLSI 2007), Stresa (Italy), March 11 2007.

[46] M. A. Hasan, "Power Analysis Attacks and Algorithmic Approaches to Their Countermeasures for Koblitz Curve Cryptosystems", IEEE Trans. Computers, vol. 50, no. 10, pp. 1071-1083, Oct. 2001.

[47] W. T. Holman, J. A. Connelly, and A. B. Downlatabadi, "An integrated analog/digital random noise source", IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol. 44, no. 6, pp. 521–528, Jun. 1997.

[48] International Technology Roadmap for Semiconductors. 2006 Update. Available at http://public.itrs.net.

[49] A. Juels, M. Jakobsson, E. Shriver, and B. Hillyer, "How to Turn Loaded Dice into Fair Coins", IEEE Trans. Information Theory, vol. 46, no. 3, pp. 911-921, May 2000.

[50] B. Jun and P. Kocher, "The Intel random number generator", Cryptographic Research Inc., San Francisco, CA, white paper prepared for Intel Corp., Apr. 1999. Available Online: http://www.cryptography.com/resources/white papers/IntelRNG.pdf

[51] P. C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems", In Neal Koblitz, editor, Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings, number 1109 in Lecture Notes in Computer Science, pages 104-113. Springer, 1996.

[52] P. Kocher, and B. Jun, "Introduction to Differential Power Analysis and Related Attacks", Technical report, Cryptography Research Inc., 1998.

[53] Kocher P., Jaffe J., and Jun B., "Differential power analysis", Proc. Advances in Cryptology (CRYPTO '99), Lecture Notes in Computer Science, vol. 1666, Springer-Verlag, 1999, pp. 388-397.

[54] Koemmerling, M. Kuhn, "Design Principles for Tamper- resistant Smart Card Processors", USENIX Workshop on Smart Card Technology, Chicago, May 1999, pp. 9-20.

[55] W. Killmann and W. Schindler, "AIS 31: Functionality classes and evaluation methodology for true (physical) random number generators, version 3.1", Bundesamt fur Sicherheit in der Informationstechnik (BSI), Bonn, Germany, 2001.

[56] T.S. Messerges, "Using second-order power analysis to attack DPA resistant software", in the Proceedings of CHES 2000, LNCS, vol. 2523, pp. 238-251, Worcester, MA, USA.

[57] S. Moore, R. Anderson, M. Kuhn, "Improving Smart Card Security using Self-Timed Circuit Technology", IEEE International Symposium on Ansychnronous Circuits and Systems, 2002, pp. 120-126.

[58] Moyart D., and Bevan R., "A method for resynchronizing a random clock on smart cards", Proc. Smart Card Security Conf., 2001.

[59] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Examining smartcard security under the threat of power analysis attacks", IEEE Trans. Comput., vol. 51, no. 5, pp. 541–552, 2002.

[60] E. Menendez and K. Mai, "A High-Performance, Low-Overhead, Power-Analysis-Resistant, Single-Rail Logic Style", Proc. IEEE Int. Workshop on Hardware-Oriented Security and Trust (HOST '08), pp. 33-36, 2008.

[61] S. Mangard, E. Oswald, and T. Popp, "Power Analysis Attacks: Revealing the Secrets of Smart Cards", Springer-Verlag, 2007.

[62] H. Masuda, S. Ohkawa, A. Kurokawa, M. Aoki, "Challenge: variability characterization and modeling for 65- to 90nm processes", in Proc. of CICC 2005, pp. 593-599, Sept. 2005.

[63] A. J. Menezes, P. C. Oorschot, P. C., and S. A. Vanstone, Handbook of applied cryptography, CRC Press, Boca Raton, FL, 2001.

[64] R. Menicocci and J. Pascal, "Elaborazione crittografica di dati digitali mascherati", Italian patent IT MI0020031375A, July 2003.

[65] S. Mangard, T. Popp and B. M. Gammel, "Side-Channel Leakage of Masked CMOS Gates", Proc. Cryptographers' Track at the RSA Conference (CT-RSA '05), Lecture Note in Computer Science, vol. 3376, Springer-Verlag, pp. 351-365, 2005.

[66] S. Mangard, N. Pramstaller and E. Oswald, "Successfully Attacking Masked AES Hardware Implementations", Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES '05), Lecture Notes in Computer Science, vol. 3659, Springer-Verlag, pp. 157-171, 2005.

[67] S. R. Nassif, "Modeling and forecasting of manufacturing variations", in Proc. of ASP-DAC, pp. 145-150, 2001.

[68] S. Narendra, S. Borkar, V. De, D. Antoniadis, A. Chandrakasan, "Scaling of stack effect and its application for leakage reduction", International Symposium on Low Power Electronics and Design 2001, pp. 195-200.

[69] S. G. Narendra, A. Chandrakasan (Eds.), "Leakage in Nanometer CMOS Technologies", Springer, 2006.

[70] National Institute of Standards and Technology, FIPS -197: Advanced Encryption Standard, November 2001.

[71] D. Naccache and D. M'Raibi, "Cryptographic smart cards", IEEE Micro, vol. 16, no. 3, pp. 14–24, Jun. 1996.

[72] B. Nikolic, V. G. Oklobzija, V. Stojanovic, W. Jia, J. K. Chiu and M. M. Leung, "Improved Sense-Amplifier-Based Flipflop: Design and Measurements", IEEE J. Solid-State Circuits, vol. 35, pp. 876-883, June 2000.

[73] E. Oswald, S. Mangard and N. Pramstaller, "Secure and Efficient Masking of AES - A Mission Impossible?", SCA-Lab Technical Report Series, 2003.

[74] C. S. Petrie and J. A. Connelly, "A noise-based IC random number generator for applications in cryptrography", IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol. 47, no. 5, pp. 615–621, May 2000.

[75] C. S. Petrie and J. A. Connelly, "Modeling and simulation of oscillator-based random number generators", in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS'96), 1996, vol. 4, pp. 324–327.

[76] T. Popp, M. Kirschbaum, T. Zefferer and S. Mangard, "Evaluation of the Masked Logic Style MDPL on a Prototype Chip", Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES '07), Lecture Notes in Computer Science, vol. 4727, Springer-Verlag, pp. 81-94, 2007.

[77] T. Popp and S. Mangard, "Masked Dual-Rail Pre-Charge Logic: DPA-Resistance without Routing Constraints", Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES '05), Lecture Notes in Computer Science, vol. 3659, Springer-Verlag, pp. 172-186, 2005.

[78] T. Popp and S. Mangard, "Implementation Aspects of the DPA-Resistant Logic Style MDPL", Proc. IEEE Int.l Symp. Circuits and Systems (ISCAS '06), pp. 2913-2916 , 2006.

[79] F. Pareschi, G. Setti and R. Rovatti, "A Fast Chaos-based True Random Number Generator for Cryptographic Applications", in Proceedings of ESSCIRC2006, pp 130–133. Montreux, Switzerland, Sep. 2006.

[80] J. Quisquater and D. Samyde, "Electromagnetic analysis (EMA): Measures and countermeasures for smart cards", in I. Attali and T. Jensen, editors, Proceedings of Smart Card Programming and Security (E-smart 2001), number 2140 of LNCS, pages 200-210, 2001, Springer-Verlag.

[81] W. Rankl and W. Effing, "Smart Card Handbook" Third Edition, John Wiley & Sons, 2003.

[82] R.L. Rivest, A. Shamir, and L.M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM (2) (1978), 120-126.

[83] G.B. Ratanpal, R.D. Williams and T.N. Blalock, "An On-Chip Signal Suppression Countermeasure to Power Analysis Attacks", IEEE Transactions On Dependable And Secure Computing, Vol. 1, no. 3, July-September 2004.

[84] B. Schneier, Applied Cryptography, 2nd ed. New York:Wiley, 1996.

[85] A. Shamir, "Protecting Smart Cards from Passive Power Analysis with Detached Power Supplies", Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES '00), Lecture Notes in Computer Science, vol. 1965, Springer-Verlag, pp. 71-77, 2000.

[86] W. Schindler, "Efficient online tests for true random number generators", in Proc. 3rd Workshop Cryptograph. Hardware Embed. Syst. (CHES'01), 2001, vol. 2162, pp. 103–117.

[87] S. P. Scorabogatov and R. J. Anderson, "Optical fault induction attacks", in B. S. Kaliski Jr., Ç. K. Koç and C. Paar, editors, Proceedings of 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES), number 2523 of LNCS, pages 2-12, 2002, Springer-Verlag.

[88] T. Stojanovski and L. Kocarev, "Chaos-based random number generators— Part I: Analysis", IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol. 48, no. 3, pp. 281–288, Mar. 2001.

[89] W. Schindler, K. Lemke, C. Paar, "A stochastic model for differential side-channel cryptanalysis", in the Proceedings of CHES 2005, LNCS, vol. 3659, pp. 30-46, Edinburgh, Scotland.

[90] D. Sokolov, J. Murphy, A. Bystrov and A. Yakovlev, "Improving the Security of Dual-Rail Circuits", Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES '04), Lecture Notes in Computer Science, vol. 3156, Springer-Verlag, pp. 282-297, 2004.

[91] T. Stojanovski, J. Pihl, and L. Kocarev, "Chaos-based random number generators—Part II: Practical realization", IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol. 48, no. 3, pp. 382–385, Mar. 2001.

[92] F. Standaert, E. Peeters, G. Rouvroy, J.-J. Quisquater, "An Overview of Power Analysis Attacks Against Field Programmable Gate Arrays", Proceedings of the IEEE, vol. 94, no. 2, pp. 383-394, Feb. 2006.

[93] A. Srivastava, D. Sylvester, D. Blaauw, Statistical Analysis and Optimization for VLSI: Timing and Power, Springer, 2005.

[94] H. Saputra, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, R. Brooks, S. Kim, and W. Zhang, "Masking the energy behaviour of DES encryption", Proc. Design, Automation and Test in Europe Conf. (DAT '03), pp. 84-89, 2003.

[95] Y. Tsividis, Operation and Modeling of the Transistor MOS (2nd ed.), Oxford University Press, 2003.

[96] K. Tiri, M. Akmal, and I. Verbauwhede, "A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards", in Proc. of ESSCIRC '02, 2002.

[97] E. Trichina, E. De Seta, L. Germani, "Simplified Adaptive Multiplicative Masking for AES and its secure implementation", in Cryptographic Hardware and Embedded Systems: CHES 2002, Vol. 2523 of Lecture Notes in Computer Science, pp. 277-285, Springer-Verlag, 2002.

[98] H. Tian, B. Fowler, and A. El Gamal, "Analysis of Temporal Noise is CMOS Photodiode Active Pixel Sensor", IEEE J. Solid State Circuits, vol. 36, no. 1, pp. 92-101, Jan. 2001.

[99] K. Tiri, D. Hwang, A. Hodjat, B Lai, S Yang, P. Schaumont, I. Verbauwhede, "A side-channel leakage free coprocessor IC in 0.18/spl mu/m CMOS for embedded AES-based cryptographic and biometric processing" Proceedings of Design Automation Conference, Munich, June 2005, pp. 222 – 227.

[100] K. Tiri and I. Verbauwhede, "A Dynamic and Differential CMOS Logic with Signal Indipendent Power Consumption to Withstand Differential Power on Smart Cards", European Solid-State Circuits Conference, Florence, Sept. 2002.

[101] K. Tiri and I. Verbauwhede, "Place and route for secure standard cell design", Proc. Smart Card Research and Advanced Application IFIP Conf. (CARDIS '04), 2004.

[102] K. Tiri and I. Verbauwhede, "A Logic Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation", in Proc. Design, Automation and Test in Europe Conference and Exposition (DATE '04), pp. 246-251, 2004.

[103] K. Tiri and I. Verbauwhede, "Charge recycling sense amplifier based logic: securing low power security ICs against DPA" European Solid-State Circuits Conference, Leuven, September 2004, pp. 179 – 182.

[104] K. Tiri and I. Verbauwhede, "Place and Route for Secure Standard Cell Design" Smart Card Research and Advanced Application IFIP Conference, Toulouse, 2004.

[105] P. Wayner, "Code Breaker Cracks Smart Cards' Digital Safe", New York Times, page D1, 22 June 1998.

[106] N. Weste and K. Eshraghian, "Principles of CMOS VLSI Design: A Systems Perspective", Addison-Wesley, 2nd edition, 1994.

[107] R. E. Walpole, R. H. Myers, S. L. Myers, K. Ye, Probability & Statistics for Engineers & Scientists, Prentice Hall, 2006.

[108] X. Zhang, K. K. Parhi, "High-speed VLSI architectures for the AES algorithm", IEEE Trans. on VLSI Systems, vol. 12, no. 9, pp. 957-967, Sept. 2004.