Power Side Channels in Security ICs: Hardware Countermeasures

Lu Zhang[1], Luis Vega[2], and Michael Taylor[3]

Computer Science and Engineering
University of California, San Diego
{luzh[1], lvgutierrez[2], mbtaylor[3]}@eng.ucsd.edu

ABSTRACT

Power side-channel attacks are a very effective cryptanalysis technique that can infer secret keys of security ICs by monitoring a chip's power consumption. Since the emergence of practical attacks in the late 90s, they have been a major threat to many cryptographic-equipped devices including smart cards, encrypted FPGA designs, and mobile phones. Designers and manufacturers of cryptographic devices have in response developed various countermeasures for protection. Attacking methods have also evolved to counteract resistant implementations. This paper reviews foundational power analysis attack techniques and examines a variety of hardware design mitigations. The aim is to highlight exposed vulnerabilities in hardware-based countermeasures for future more secure implementations.

1. INTRODUCTION

Side-channel attack scenarios take advantage of information leaked from channels other than the main communication channel and discover small but critical secrets from the leakages. Attacks target chips that incorporate cryptographic functions in order to reveal the secret keys using side-channel analysis techniques. Well-known side channels include variations in execution time, power consumption, and electromagnetic emission during different encryption and decryption steps. Because of the non-invasive properties of side-channel attacks they are often easy to mount without expensive equipment, and hard to be distinguished from normal chip operations. Kocher *et al.,* [KJJ99] in 1999 presented practical approaches to performing power analysis on cryptographic devices. Two kinds of methods were developed to extract secret keys: *Differential Power Analysis (DPA)* and *Simple Power Analysis (SPA)*. The DPA method employs statistical analysis using many power measurements, and the SPA is applicable when the leak is so evident that simple analysis techniques such as visual inspection can disclose secrets.

Modern cryptographic ciphers such as AES and RSA are designed to resist adversaries that supposedly have knowledge of both plaintext and ciphertext data. For example, to break AES, one must exhaustively enumerate all possible cryptographic keys which is computationally prohibitive. The power analysis techniques circumvent such difficulties by intercepting *intermediate values* that are calculated for encryption or decryption processes. The intermediate values are much shorter in bit length than either the plaintext or the ciphertext, allowing the adversary to take a *divide-and-conquer* strategy to recover portions of the key separately. Power analysis attacks can be successfully mounted on a variety of cipher implementations. This paper primarily uses the AES (Advanced Encryption Standard) block cipher as an example for power analysis discussions due to its wide adoption. The appendix has an introduction of the AES round functions and the cipher implementation choices. The AES standard is specified in [Nat01].

This paper focuses on the differential power analysis because it is much harder to defend against than the simple power analysis attacks. The major contribution is to evaluate emerging hardware countermeasures that have been developed since the disclosure of power side channels and summarize their potential vulnerabilities. The rest of this paper is organized as follows. Section 2 briefly introduces the concept of power traces and how to set up an attack. Section 3 reviews basic methods for differential power analysis. Section 4 surveys the most commonly used power models. Section 5 presents metrics for evaluating the effectiveness of attacks or defenses. Section 6 outlines various directions for mitigations. Section 7 discusses a variety of hardware-based countermeasures, and section 8 summarizes the paper and proposes other promising solutions.

2. POWER TRACES

Modern cryptographic algorithms are typically implemented in an integrated circuit (IC) chip that consists of numerous logic gates composed of CMOS (Complementary Metal-Oxide Semiconductor) transistors. Transistors *switch* on and off to represent the logic function of a gate and these transitions draw electric current from the chip's power supply. The power analysis attack is based on the fact that the overall chip's power variation reflects the aggregated switching activity of each gate.

The first step for power analysis attacks is to acquire one or more *power traces* from the target device. A power trace

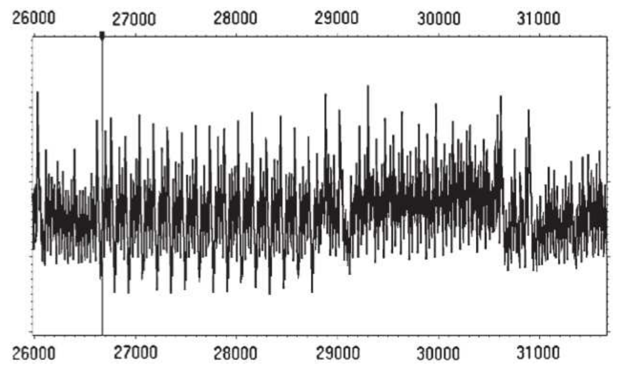


Figure 1: An Example Power Trace in [KJJR11]. The vertical line marks the time of the first S-Box output.

(Figure 1) is a digitized sampling sequence of instantaneous power consumption values over a period when a series of cryptographic operations take place, for example, the first round of the AES. Once a power trace has been acquired some optional digital signal processing can be used to improve the trace quality.

2.1 The Measurement Setup

Many cryptographic chips require a single constant voltage supply, e.g., 5V, 3.3V or 1.8V, often denoted as V_{dd}. The current drawn from V_{dd} by the chip is a time-variant value $I_{dd}(t)$ in Amperes that sinks to the ground lines of the chip. $P(t) = V_{dd} \cdot I_{dd}(t)$ describes the instantaneous power in Watts of the whole chip.

The most commonly used sampling device is an oscilloscope that measures voltages. A small resistor R is usually inserted into the V_{dd} or ground line, and the voltage drop V_R across the resistor is sensed by probes of the oscilloscope and stored. The voltage drop is proportional to the power consumption: $V_R = I_{dd}(t) \cdot R \propto P(t)$. Figure 1 shows an example power trace representing the first round of an AES-128 encryption on a smart card. From the time marker, it is visible that there are 16 spikes happening in sequence, indicating the microcontroller is working on state bytes one after another. Recorded power traces are represented by the matrix in (1).

$$\mathbf{P} = \begin{pmatrix} P_1(1) & P_1(2) & \cdots & P_1(T) \\ P_2(1) & P_2(2) & \cdots & P_2(T) \\ \vdots & \vdots & \ddots & \vdots \\ P_N(1) & P_N(2) & \cdots & P_N(T) \end{pmatrix} \tag{1}$$

In $\mathbf{P}$ each row represents one power trace measured within a period T, for the same sequence of operations performed on N different sets of data. For example, the first round of AES-128 encryptions using the *same* cipher key. The instantaneous value $P_n(t)$ in the matrix is digitized, and stored as binary integers. Because of sampling, the time values represented in the matrix are discrete. One important property that must be preserved between rows of the matrix is the *time consistency*, meaning the values of each column must preserve the same time offset relative to the start of sampling.

In practice, many measurement setups have to be carefully performed to get clear power traces. For example, a highly stable supply voltage is preferred. The serial resistor should be inserted close to the target chip's supply or ground pins, not the circuit board or device level ones. Modern SoCs usually have several power supplies for different parts of its internal structures, which, on the other hand, all share the same ground. Some chip I/O pins being pulled down are also electrically grounded and may shunt some amount of return currents. In such cases, the ground lines are noisier than the power line that mainly supplies the internal cipher module. Unless specially noted this paper assumes attacks are executed in the power line, and higher values of matrix $\mathbf{P}$ indicate higher power consumptions from the source V_{dd}.

2.2 The Signal Processing

To suppress noise and improve time consistency, optional signal processing can be performed on the raw power traces. Noise signals can be filtered by applying appropriate filters. Time consistency is often preserved by a *trigger* signal [KJJR11] that indicates the beginning of certain functions, for example, the first AES round. This signal can trigger the oscilloscope to start sampling, and the resulting power traces will be well aligned. In case such a trigger signal is not available, *trace alignment* techniques are employed. A simple correlation test could be helpful to find the time shift τ that minimizes the differences between $P_i(t)$ and $P_j(t+\tau)$. Occasionally more complex alignment methods are needed in the presence of hardware countermeasures [CCD00] or for a very noisy device such as a smartphone [NSN$^+$14]. The power trace matrix shown in (1) can be considered as filtered and aligned, and each row corresponds to the encryption process for one data block of 16 bytes.

3. DIFFERENTIAL POWER ANALYSIS

The first differential power analysis introduced by [KJJ99] uses a method called *Difference of Means (DoM)*. Later versions using more effective statistical tools have been developed to improve the performance of differential power analysis. One easy to mount and very powerful attack calculates correlation coefficients and thus it is often called the *correlation power analysis (CPA)*. This section mainly introduces these two methods because hardware countermeasures often use them for the evaluations. A few advanced attacking techniques are also briefly introduced.

3.1 Difference of Means

The differential power analysis published by [KJJ99] targeted a DES (Data Encryption Standard) cipher, and later the same authors extended their work to AES [KJJR11]. They introduced a *known-plaintext* attack: the adversary has a set of plaintext data, encrypted by an AES cipher using the *same* secret cipher key. For each plaintext input, the adversary can observe a power trace and for all the N inputs (e.g. 4,000 in [KJJR11]) the power traces are stored in a matrix like $\mathbf{P}$. Each power trace represents the encryption of a data block of 16 bytes (Figure 1). In this scenario, the ciphertext can be used at the end to verify the correctness of the key recovered, but for the extraction process the ciphertext is not used.

Observations show that the power trace values sampled at a specific point in time (a column of the power trace matrix) is

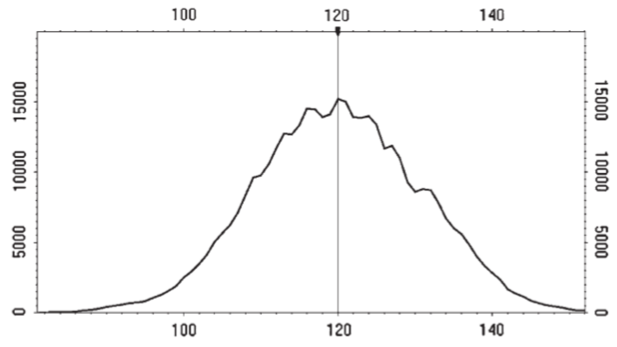


Figure 2: The Power Consumption Distribution at the Time Marker of Figure 1

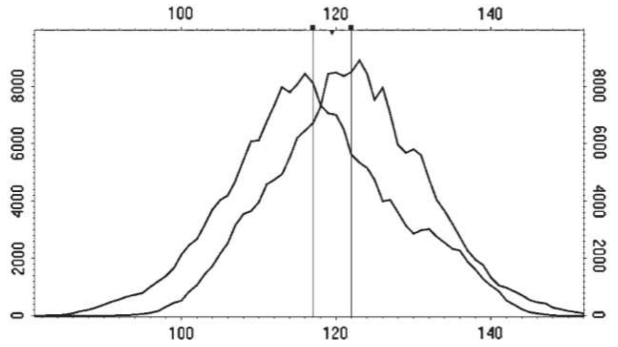


Figure 3: Distributions of the Two Subsets at the Time Marker of Figure 1

close to a *normal distribution*. Figure 2 shows the distribution at the time of the first S-Box output in the initial AES round. The distribution is a combined effect caused by all switching transistors of the chip and electronic noises. Now *suppose* the cipher key is known then the statistical analyze is as follows. For each data input, the result after the first AddRoundKey and SubBytes can be calculated. Then according to one arbitrary bit of the output, for example, the least significant bit (LSB), separate the power traces into two subsets. One subset has the traces where the LSB is 0, and the other has traces where the LSB equals 1. Now the distributions of the two subsets are still close to normal but with distinguishable means, shown in Figure 3.

The way to separate traces is known as the *selection function* (2) for DoM based attacks. The reason for the difference of means is circuit design details make it possible for either LSB value to consume more power when data is being processed. Switching to the attack scenario, the adversary, without knowing the cipher key, can do exhaustive testing of all key hypotheses, and each time use the *hypothetical* LSB to separate power traces into two subsets. Then the adversary calculates the difference of means as in equation (3). When the key guess is correct, the difference of the subsets' means usually stands out as shown in Figure 3. Even if the difference is very small, it will become *statistically significant* given a sufficient number of power traces (large N values in matrix $\mathbf{P}$). If the key guess is not correct, each subset is essentially a random sampling of the full distribution and their DoM often vanishes as the number of trace increases.

$$F(d_n,\ x_k) = LSB(\ S(d_n \oplus x_k)\ ) \tag{2}$$

In equation (2) and (3), $S(\cdot)$ indicates the SubBytes function, d_n represents one data *byte* of the initial state of the n^{th} power trace, and x_k ($k \in [1 \cdots 256]$) is a guessed round key *byte* corresponding to d_n. The $\Delta\bar{P}(t)$ is known as the *differential trace*, and the predicted byte $x_{\hat{k}}$ of the first round key is chosen by the greatest difference of means at certain time $\hat{t}$. Thus, a successful attack also reveals the time $\hat{t}$ when the key byte $x_{\hat{k}}$ is used. The same DPA process is repeated for each key byte independently. Because this attack scenario targets the S-Box output of the *first* round, the recovered round key is also the predicted cipher key.

$$\begin{aligned}\Delta\bar{P}(t) =& \frac{\sum_{n=1}^{N} F(d_n,\ x_k)\cdot P_n(t)}{\sum_{n=1}^{N} F(d_n,\ x_k)} \\ &- \frac{\sum_{n=1}^{N}(1 - F(d_n,\ x_k))\cdot P_n(t)}{\sum_{n=1}^{N}(1 - F(d_n,\ x_k))}\end{aligned} \tag{3}$$

$$(\hat{k}, \hat{t}) = \underset{(k,t)}{\operatorname{argmax}}\ |\Delta\bar{P}(t)| \tag{4}$$

Mounted DPA attacks, in practice, are often more complex. The target bit does not have to be the LSB, and different bits could have varying degrees of leakages. There could also be more than one guess that leads to equally strong spikes in the differential trace, and a decision based on equation (4) is hard to make. The false spikes caused by incorrect key bytes are termed as *harmonics* or *ghost peaks*, which are essentially caused by several key hypotheses all correlate to the chosen bit. Nevertheless, such a simple method worked devastatingly well on many smart cards when the paper [KJJ99] was published. Later methods have evolved to make great improvements in this classical DPA technique.

3.2 Correlation Coefficient

The correlation power analysis [BCO04] introduced by Brier *et al.,* was the first to explicitly use the *correlation coefficient* (also known as the *Pearson Product-Moment Correlation Coefficient*) to make decisions among key hypotheses. To deploy a correlation analysis, the adversary needs to build a hypothetical power consumption matrix $\mathbf{H}$, for every key guess in $\{x_1, x_2, \cdots, x_K\}$ ($K = 256$ in case only one key byte is targeted in one attacking attempt). In $\mathbf{H}$, each column is calculated using one value of x_k, and the entry $H_n(k)$ is a modeled power value estimated by using a plaintext byte d_n and the hypothetical key byte x_k.

$$\mathbf{H} = \begin{pmatrix} H_1(1) & H_1(2) & \cdots & H_1(K) \\ H_2(1) & H_2(2) & \cdots & H_2(K) \\ \vdots & \vdots & \ddots & \vdots \\ H_N(1) & H_N(2) & \cdots & H_N(K) \end{pmatrix} \tag{5}$$

$$R_k(t) = \frac{\sum_{n=1}^{N}(H_n(k) - \bar{H}(k)) \cdot (P_n(t) - \bar{P}(t))}{\sqrt{\sum_{n=1}^{N}(H_n(k) - \bar{H}(k))^2 \cdot \sum_{n=1}^{N}(P_n(t) - \bar{P}(t))^2}}$$
$$with : \bar{H}(k) = \frac{\sum_{n=1}^{N} H_n(k)}{N},\ \bar{P}(t) = \frac{\sum_{n=1}^{N} P_n(t)}{N} \tag{6}$$

Therefore, the computing of $H_n(k)$ depends on how the adversary models the power consumptions according to the cipher's intermediate values. The paper [BCO04] suggested a *Hamming Distance* model, which is the count of different bits between two bit-vectors. The model could target the state register of the AES cipher, and assumes its power consumption has a linear relationship with the number of flipped bits between two successive states. For the known-plaintext attack scenario, the two successive state values can be the results of AddRoundKey and SubBytes in the first round. Hence $H_n(k)$ is computed by the Hamming distance between $(d_n \oplus x_k)$ and $S(d_n \oplus x_k)$.

$$(\hat{k}, \hat{t}) = \underset{(k,t)}{\text{argmax}}\ |R_k(t)| \tag{7}$$

The correlation coefficient $R_k(t)$ ($[-1.0, 1.0]$) between each column $\mathbf{H}(k)$ of $\mathbf{H}$ and each column $\mathbf{P}(t)$ of $\mathbf{P}$ is computed as in (6). The greatest value of $|R_{\hat{k}}(\hat{t})|$ given by (7) predicts the secret key byte $x_{\hat{k}}$ and the time $\hat{t}$ of its leakage.

The explained DPA and CPA approaches are suitable for a known-plaintext scenario. For encryption power traces with *ciphertext-only*, the attacking process needs to be adjusted. For example, the target intermediate value becomes the *last round S-Box input*, and the attacker tries to predict the *last round key*. Since the key expansion for AES-128 is invertible with any round key, having the last round key also reveals the cipher key. Similar attack flows can be derived for the decryption process as well. The first or the last round of the block cipher is often targeted by DPA because the data complication is sufficiently weak for some of the intermediate values to be easily enumerated. It is possible that an attack does not uniquely predict a key, or the most likely choice is not the right one. In such cases, a list of candidates can be supplied to some key enumeration and validation steps. Nevertheless, the attack results should have reduced the remaining brute-force effort to feasible levels.

3.3 Advanced Power Analysis Attacks

Since the debut of practical power analysis attacks, many evolved approaches have been proposed to improve either DPA or SPA. For instance, the selection function for the DoM method can assign weights to different traces or divide traces into more than two categories. Using functions other than the ordinary averaging could be useful when data sets have unusual statistical distributions. Brief introductions about a few other major improvements to power analysis techniques are as follows.

Higher-Order DPA: The attacks introduced in the previous section only analyze the statistics of samples at *one* point in time, i.e., correlate $\mathbf{H}(k)$ with each column of $\mathbf{P}$ independently. This is known as the *first-order* attack. [KJJ99] also introduced *higher-order* DPA that works on samples at multiple times within $\mathbf{P}$ and their cross-correlations. The *attacking order* is roughly the number of intermediate values of different time being considered in one attack. Practical results ([Mes00] and [OMHT06]) have reported that higher-order attacks can compromise implementations with certain countermeasures.

Collision Attacks: A collision attack [SLFP04] requires a pair of encryption runs with two known and different inputs. It checks if the two encryption runs compute some equal intermediate values, known as *intermediate value collisions*. The essential idea is that for two different inputs, a collision cannot occur for all key values, but only for a limited subset of keys. Hence, the collisions reduce the guessing space and possibly uniquely identify the cipher key. Adversaries using this attacking method may prefer to choose some plaintext inputs for efficient collision detection, and such a scenario is called the *chosen-plaintext attack*. In practice, one pair of chosen plaintexts might only allow the recovery of one key byte. A few more measurements are required to predict the whole cipher key, but the number of required power traces is usually significantly reduced. Enhanced collision attack using correlation analysis has also been developed by [MME10].

Profiled Attacks: This series of attacking methods aim to solve the problem when only one or two power traces of the victim device are observable. The template attack [CRR03] builds an accurate power consumption model using an identical or similar device as the victim. The attack consists of two steps. First a *profiling* phase is performed by precisely characterizing power traces as a *multivariate normal distribution*. Next during the on-site attack a *key extraction* phase is executed using the available templates and key hypothesis testing on a *single* power trace observed from a victim device. The researchers successfully used the template attack to break an RC4 stream cipher and a DES block cipher. Schindler *et al.,* [SLP05] introduced the stochastic models for the device profiling. It achieves the efficiency of the template attack in the key extraction phase but requires far fewer measurements in the profiling phase.

Algebraic Side-Channel Attacks: The algebraic side-channel attack [RS10] combines template attacks and algebraic cryptanalysis. Besides building templates, it also exploits the information leakage of many cipher rounds (not limited to the first or last round) to build a boolean satisfiability problem to solve. The authors managed to break a PRESENT block cipher with the observation of a single power trace. A later improved version [MBZ+12] also successfully attacked an AES-128 implementation.

4. POWER MODELS

Most power side-channel attacks exploit the *linear relations* between the observed power traces and the hypothetical power consumption that is dependent on intermediate values the circuitry processes. Analysis of transistor-based logic gates reveals a clear dependency between its output value and the power it consumes. Figure 4 illustrates a basic CMOS inverter gate and simplified models of its transition

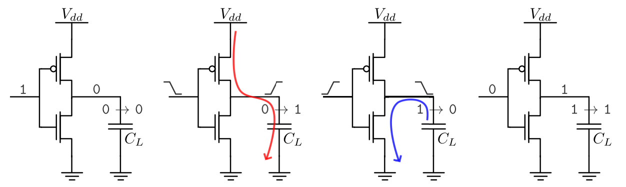


Figure 4: CMOS Inverter Transitions

activities. Its power consumption includes three parts: the *static power*[1], P_{static}, the *short-circuit power*, P_{short} and the *switching power*, P_{switch}. The sum of the latter two components is also known as the *dynamic power* P_{dyn}. According to [RE10], the semiconductor technology used for the fabrication of smart card microcontrollers lies in the range of 0.18-μm to 90-nm. And the dominant factor of power consumption for a technology node in this range is the dynamic power [MSSE09]. This paper only considers the dynamic power, since most power side-channel attacks rely on this portion. Attack results based on sub-threshold power are still preliminary [Mor14].

The capacitor C_L in Figure 4 is a lumped model including the transistor's intrinsic capacitance, its load capacitance, and the parasitic capacitance of the interconnections. When its output switches from `0` to `1`, C_L gets charged by a current from V_{dd}. The switching power the gate consumes is proportional to $\alpha \cdot f \cdot C_L \cdot V_{dd}^2$, where α is the active factor of `0` $\to$ `1` transitions in one clock cycle, and f is the clock frequency of the circuit. When the output switches from `1` to `0`, C_L discharges and ideally there are no current flows from the power supply. However in both `0` $\to$ `1` and `1` $\to$ `0` transitions there is a momentary short-circuit current flowing from V_{dd} to the ground when both the transistors are conducting. The `0` $\to$ `0` and `1` $\to$ `1` transitions do not incur power variations if not considering the sub-threshold current. Accordingly, if measure the power consumption in the V_{dd} line, `0` $\to$ `1` transitions contribute the most, and then the `1` $\to$ `0` cases. Now consider an `AND` which computes $y = a \cdot b$ and its switching activity shown in Table 1 (just the a, b, and `AND` columns). Given a uniform distribution of switching probabilities for a and b, obviously the probabilities for the four output switching activities are not uniform anymore. Assume a is a fixed secret bit (like a key) and b is a random bit (like a plaintext or ciphertext). If a is `0` (i.e., the four `0` $\to$ `0` rows of a), y has neither `0` $\to$ `1` nor `1` $\to$ `0` switching; otherwise if a is `1`, y has one `0` $\to$ `1` and one `1` $\to$ `0` switching (i.e., in the four `1` $\to$ `1` rows of a). Therefore, by knowing b and monitoring the mean power consumption of the `AND` gate it is possible to deduce what value a has. In contrast, the `XOR` function still has uniform output switching activities and does not easily reveal the value of a in this way.

Scaling up this simple example, a whole cryptographic circuit's power consumption is the aggregation of power consumed by all its logic gates. For AES, the AddRoundKey function can be implemented only using `XOR` gates, but the finite field inversion of the S-Box must employ logic functions like `AND` or `OR` ([WOL02]). Hence power analysis attacks work and often favor the S-Box because of the distortions in the odds of switching activities make distinguishable power profiles between the right and wrong key guesses. With a chosen power analysis method, e.g., the CPA, the quality of mapping intermediate values to power consumptions has an important impact on its effectiveness. This section gives a brief survey of commonly used mapping methods, also known as the *power models.*

Single Bit: Power modeling using a single bit is the most basic approach based on the principle explained using the `AND` gate. Nevertheless, its effectiveness had been proved by the classical DoM based DPA attacks. Different target bits have been observed showing a different amount of leakages. The differences root from variations in data paths that signals travel along, and the ghost peak phenomena happen because multiple keys can correlate to the same bit.

Hamming Weight: The Hamming weight power model is a multi-bit extension of the single bit model. The weight is simply the number of `1`s in a bit vector. For this model to work effectively, it should assume all the bits are in a constant state (either all `0`s or all `1`s) before a target value appears. Therefore the Hamming weight of an intermediate value is roughly linear to the power consumed to process it. Messerges *et al.,* [Mes00] first introduced this model to power analysis attacks and found the fact that many smart card processors around that time (year 2000) precisely exhibited these characteristics. The reason is many microcontrollers precharge their buses before carry the real data, and this even leads to simple `MOV` instructions leak the Hamming weights of their operands [MS00]. Leakage of Hamming weight is especially severe for look-up table based S-Box implementations because the I/O ports of memories usually have much higher capacitance than other cells.

Hamming Distance: The Hamming distance is the count of different bits in two bit-vectors, and the model was first used by the correlation power analysis in [BCO04]. It does not require a priori state of all `0`s or all `1`s before the target value appears. But it supposes the adversary to guess two successive values of a register to calculate the Hamming distance between them. The CPA attack using the AddRoundKey and the SubBytes outputs exemplifies its use. It assumes `0` $\to$ `1` and `1` $\to$ `0` transitions equally contribute to the power consumption. Though still not very accurate, its applicability has been proved by many practical attacks and widely used as a generic power model.

Switching Distance: Peeters *et al.,* [PSQ07] introduced a power model called the *switching distance.* It exploits the fact that `0` $\to$ `1` and `1` $\to$ `0` transitions draw a different amount of current from the supply. For an inverter, a `0` $\to$ `1` switching draws some current from the V_{dd} for charging the C_L and the short-circuit effect while during a `1` $\to$ `0` switching only some short-circuit current appears. Hence, the switching distance is defined as a tunable factor δ, indicating the power consumption difference between `0` $\to$ `1` and `1` $\to$ `0` transitions.

Toggle Count: The above models are good approximations for registers that usually change states only once in a clock

[1]The static power is also called the *sub-threshold leakage power.* To avoid confusion with the information leakage in this context, this paper refers to it as the static or just sub-threshold power.

cycle. Mangard *et al.,* [MPO05] noted the fact that glitches in combinational circuits are the dominant factor of power consumptions in a clock cycle. Glitches are unwanted output toggles before it settles due to different arrival times of input signals and the gate's function. Therefore, this model evaluates the number of gate toggles in the S-Box circuit instead of its final output. It works the best on hardware-arithmetic based S-Box implementations due to the complex combinational data paths. The model supposes the attacker knows certain design details of the cipher, for example, the gate-level circuit netlist, and builds an accurate power model using circuit simulations.

Power Simulation: More accurate modeling of the CMOS gates is obtained by advanced power simulation. Commercial IC design tools provide power simulation capabilities at different levels. From high to low, simulations can be categorized into behavioral, netlist, layout and analog levels. Accordingly in the same order the simulation requires greater details of the IC design and hence provides more accurate results.

Profiled Model: The profiling of template and stochastic attacks can be viewed as power modeling as well. Different from hypothetical modeling, the profiling utilizes real measurements performed an identical or similar device as the victim. While other attack forms consider noises as hindrance, profiled attacks precisely model noises as well. The probability distributions of noises using different cryptographic keys are characterized during the profiling. Power side-channel analysis usually considers profiled attacks as the strongest form.

The power consumption values predicted by hypothetical models do not have to be in Watts as long as they preserve a (partial) linear relationship to the actual data-dependent power. It is reasonable that the better a model fits the actual power consumption of the chip, the more effective an attack becomes, and obviously better power modeling requires more knowledge of the adversary about the victim device. A comparative study in [MBMT13] showed that a Hamming weight model failed while the Hamming distance and switching distance models succeeded in the same CPA attack setup. Advanced power models such as toggle counts and power simulations are not infeasible since present time witnesses a growth in open-source hardware and hardware as IP businesses. It would be a difficult problem to distinguish benign and malicious users.

5. METRICS

Fair evaluation methods for side-channel attacks and countermeasures are useful to compare their quality. The most commonly used metric for attacks and countermeasures is the **number of power traces**, i.e., the value N of $\mathbf{P}$. However this metric is subject to the power monitoring method, the quality of tools and the effectiveness of attacks. Mangard *et al.,* [MOP08] developed a more analytical way to evaluate the vulnerability of a device to power analysis attacks. They model the power consumption as the sum of the *exploitable power* P_{exp}, the *switching noise* $P_{sw.noise}$, the *electronic noise* $P_{el.noise}$, and the constant component P_{const}. The four components are independent of each other, and their effects are additive.

$$P_{total} = P_{exp} + P_{sw.noise} + P_{el.noise} + P_{const} \tag{8}$$

The P_{exp} is the component that depends on intermediate values (e.g., an S-Box output during an encryption) that the attacker is looking for, or, in other words, the power that relates to secrets. The exploitable power is intrinsic to a cipher implementation and independent of attacking methods. For attackers, they try to use power models to approximate P_{exp}, and from the defensive perspective this P_{exp} should be carefully evaluated and minimized. The switching noise $P_{sw.noise}$ results from transistor switches that are independent of the sensitive values, for example, another processor core doing irrelevant tasks. The electronic noise $P_{sw.noise}$ describes variations caused either internal or external sources. The P_{const} is the constant part such as the static power of the cryptographic IC. Given an attack scenario, the signal-to-noise ratio is defined by (9), and the $Var(\cdot)$ function calculates the variances of its operand.

$$SNR = \frac{Var(Signal)}{Var(Noise)} = \frac{Var(P_{exp})}{Var(P_{sw.noise} + P_{el.noise})} \tag{9}$$

The researchers also derive the correlation coefficient $R_k(t)$ in terms of P_{exp} and the SNR as equation (10). The $\rho(X, Y)$ function calculates the correlation coefficient ($[-1.0, 1.0]$) of two random variables.

$$R_k(t) = \rho(\mathbf{H}(k), \mathbf{P}(t)) = \frac{\rho(\mathbf{H}(k), P_{exp})}{\sqrt{1 + \frac{1}{SNR}}} \tag{10}$$

The SNR is independent of the power model used by an attacker while the $\rho(\mathbf{H}(k), P_{exp})$ is dependent on the power model, and it describes how well an attacker approximates the P_{exp}. Equation (10) gives several directions for defending correlation-based attacks, and the goal is to make $R_k(t)$ approaching zero so that adversaries do not easily get conclusive predictions. Apparently a weak correlation between $\mathbf{H}(k)$ and P_{exp}, or strong noises both reduce the amplitude of $R_k(t)$, and consequently it becomes difficult for an adversary to predict the right key. Usually, a lot more power trace measurements are needed in case of a low SNR. Their exact relations still depend on practical details.

Standaert *et al.,* [SMY09] developed improved security metrics using some information theoretic concepts. A few popular variations have been adopted by the DPA Contest[2] which is an online benchmark for power side-channel attacks. In the Non-Invasive Attack Testing Workshop host by NIST in 2011, Goodwill *et al.,* [GJJ+11] introduced a leakage detection method called the *T-Test*. It uses a statistical hypothesis testing method to detect if one or some sensitive intermediate values can influence the measurement data. Hardware countermeasure evaluations have not been widely using the new metrics and so this survey does not discuss them in detail. Subsequent sections still use the number of power

[2]www.dpacontest.org

traces because they appear the most commonly in publications. However, unless the numbers are from the same authors and the same measurement setup, they should not be considered as a precise metric for the effectiveness of corresponding countermeasures.

6. OVERVIEW OF COUNTERMEASURES

Ideally, making the cryptographic device physically secure or frequently changing keys should be the most effective countermeasures for side-channel attacks. However, such mitigations are difficult to guarantee in a large number of cases. For example, an encrypted secret is shipped to an adversarial user with the decryption key hardened in the device (e.g., an encrypted FPGA design). Or a strong attack already reveals the key before its session expires (e.g., an authentication smart card). Aggressive physical shielding is often difficult for compact devices to adopt. Therefore, it is strongly desired for the cryptographic circuit to have intrinsic fail-safe solutions in case it is in the hands of adversaries.

Power side-channel attacks work because the power consumption of a cryptographic device is dependent on its processed intermediate values, as indicated by the P_{exp}. Such data dependencies are amplified by the non-linear functions of modern ciphers, for example, the AES S-Box. The S-Box was designed to resist conventional *linear cryptanalysis.* But experimental results of Guilley *et al.,* in [GHP04] showed the paradox that the better protected against linear cryptographic attacks a block cipher is, the more vulnerable it is to differential power analysis. Prouff also formally proved in [Pro05] the resistance of an S-Box to DPA attacks and the classical cryptographic criteria, such as high non-linearity, cannot be satisfied simultaneously.

Hence, one obvious goal of counteracting techniques is to eliminate such power-data dependencies, especially around the non-linear cipher functions. One idea is to perform computations in a way such that different data processing consumes the equal amount of power. Such techniques are known as *hiding*, which brings down $Var(P_{exp})$ ($SNR \rightarrow 0$). Thus for all key hypotheses the correlation coefficients $R_k(t)$ are close to zero and not conclusive. The capability to balance power consumptions is limited for software. Hardware design can effectively modify a chip's power characteristics at various levels.

In practice, the data dependency can not be completely removed, and then according to the *SNR* definition another direction for mitigation is to increase the variance of the *amplitude noise*. Noise can be achieved by *on-chip noise generation*, including excessive switching noises and electronic noises. Hardware implementations using wide data paths (e.g., parallel S-Boxes) for cryptographic algorithms are example approaches to increase switching noises. The added noises also do not completely hide the exploitable power component, but they can fairly complicate an attack. Programmers can increase noise by executing parallel activities to the cryptographic functions. But doing multiple tasks in software is often limited by the cost and footprint constraints on many security ICs.

Masking, also known as *secret sharing*, is a general countermeasure for both software and hardware that randomizes the intermediate values while still keeps the input and output of a cipher the same as the unmasked version. For example, in the AES encryption algorithm, the plaintext and the key are added or multiplied with random values, i.e., the masks. The transformations of masks through round functions are tracked, and before the ciphertext output the masking effect is removed properly. Consequently, it becomes difficult for adversaries to correlate the P_{exp} using hypothetical power models ($\rho(\mathbf{H}(k), P_{exp}) \rightarrow 0$). Masking techniques can apply to the algorithmic level without changing the power consumption characteristics of the cryptographic hardware, and therefore it is very commonly used in software. The same algorithmic masking can be ported to hardware, and besides algorithmic masking, hardware design also utilizes gate-level masking. Because masking is suitable for multiple design levels, there is a large body of research literature on this topic. This survey will focus on gate-level masking in the next section.

Another technique that is implicit in the *SNR* expression is to break the trace alignment effectively (see Section 2.2), and therefore it prevents attackers from building a meaningful power trace matrix. The software can implement this countermeasure can be implemented by inserting random number of *dummy instructions* before and after protected functions, or *shuffling* the execution order of S-Boxes for every data block. Hardware design can adopt similar techniques. Effects that cause trace misalignment is also called *temporal noise* for attacks.

Usually, a cipher implemented in hardware is more resistant to power analysis attacks than its software version. Microprocessors are often highly pipelined and equipped with a data bus that consumes the majority part of the whole chip's dynamic power. Simple power models such as Hamming weight and Hamming distance often fairly represent the exploitable power of their cipher implementations. The parallel nature of hardware, complex data paths, and irregular transistor toggles in one clock cycle all increase the attacking effort. Mangard *et al.,* [MOP08] designed an AES core in ASIC using four parallel S-Boxes and found it took significantly more effort to attack the hardware than the software counterpart on an 8051 microcontroller.

Complex SoCs and devices may not be more resistant to power side-channel attacks than a small chip. Successful side-channel attacks mounted to modern smartphone processors[3] are demonstrated by [JK12] and [NSN$^+$14]. A large design usually has many power and ground pins surrounding the chip. Some of the supply pins are close or dedicated to a security circuit, and thus a quality measurement can be done with these pins. Even worse, for power efficiency and hence usability, contemporary large-scale chips must use power gating techniques to shut down unwanted switching of transistors. From a security perspective these techniques effectively turn off switching noises and leave the running cryptographic module as vulnerable as in a small chip. Therefore appropriate countermeasures must also be employed cryptographic circuits of large-scale SoCs as well.

[3]These two cases actually use electromagnetic signals for attacking but the principle of leakage is similar to the power side-channel, both induced by rapidly changing currents.

7. HARDWARE COUNTERMEASURES

Hardware design can effectively implement a variety of countermeasures. Since power consumption properties can be largely changed in hardware design, many approaches have been trying to conceal the real activities of the protected circuit. This section first introduces logic gate masking, followed by the dual-rail precharge logic that intends to equalize power consumption for all switching activities. Gate masking and dual-rail/precharge designs often combine to take merits of both. Other techniques such as power line isolation, noise generation and randomized dynamic voltage and frequency scaling have also been proposed to thwart DPA attacks.

7.1 Masked Logic

Alternative to masking at algorithmic level is the use of *masked logic gates* so that at the gate-level no values stored in hardware are correlated to secret intermediate values. The basic idea of gate masking is to mask properly a gate's input and output signals and never let the true secret values expose until the final output. For example, a regular 2-input `AND` gate ($y = a \cdot b$) is transformed to a 5-input (a_m, m_a, b_m, m_b, m_y), 2-output (y_m, m_y) `Masked-AND` [Tri03] as shown in Figure 5 and equation (11). The m_a, m_b, and m_y are random 1-bit masks that can be either equal or different. In the same manner, other basic gates can be converted into masked versions and consequently any circuit can be built using masked gates. Ishai *et al.,* [ISW03] formally proved that any logic circuit can transform into corresponding securely masked versions. Before entering the masked circuit inputs like a and b are masked as $a_m = a \oplus m_a$ and $b_m = b \oplus m_b$ (remember that `XOR` functions are not easily attacked by DPA). Masked values and mask bits propagate together along the circuitry (e.g., one AES round) and at the end of the protected computing the true values are recovered by removing masks (e.g., $y = y_m \oplus m_y$).

$$y_m = a_m \cdot b_m \oplus (m_a \cdot b_m \oplus (m_b \cdot a_m \oplus (m_a \cdot m_b \oplus m_y))) \\ with: \; a_m = a \oplus m_a, b_m = b \oplus m_b, y_m = y \oplus m_y \tag{11}$$

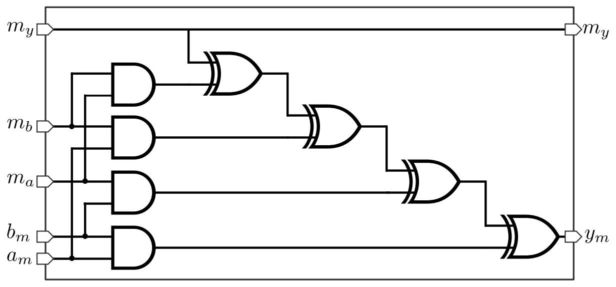


Figure 5: The Masked `AND` Gate

Compared to algorithmic masking, masking at gate level does not have to consider the higher level functions of using logic gates. Given a cryptographic circuit implemented by regular gates, masked gates can be swapped for them to form a less leaky version, at a cost of increased gate count and extra power consumption. Standard logic synthesis tools do not well support this design flow, whose primary goals are usually optimizing timing and area. Therefore, additional effort is required for the functional and timing verification of the masked gate netlist.

Masking is a mathematically provable securing scheme without considering real hardware behaviors. It effectively hides dependencies between the secret data and *settled* final values of gate outputs, and counteracts basic power models such as the Hamming weight and the Hamming distance because such attacks also just consider the final values. Unfortunately, the real hardware is complicated, and a masked design often overlooks glitches caused by the different arrival times of input signals to logic gates. It is obvious that the masked gate in Figure 5 has severely unbalanced signal paths, and worse variations may already exist before they enter the first level of the masked layers. Consequently some unwanted glitches appear at the output before the final value settles, and they are found correlated to the secret key values. Mangard *et al.,* [MPG05] demonstrated that simply masked gates are vulnerable to attacks using the more precise toggle count power model. A formal explanation of how glitches leak information is given by Nikiva *et al.,* [NRR06].

Masking methods, including algorithmic masks, are also susceptible to various advanced DPA attacks, such as higher-order DPA ([Mes00]), template attack ([ARRS05]), and algebraic side-channel attack ([RS10]. These attacks exploit multiple intermediate values in a power trace and cross-correlate them to remove the effect of masks. On the other hand higher-order masking methods ([RP10]) have been proposed to counteract higher-order power analysis. The *masking order* is determined by how many independent data *shares* are needed to represent the true secret value. If the number of shares is $d + 1$, e.g., $X = \bigoplus_{i=1}^{d+1} x_i$, the scheme is known as the d^{th}-order masking. For example a masked gate's output y is split into two shares, i.e., y_m and m_y; this is called a first-order masking. A d^{th}-order masking is supposed to resist a d^{th}-order attack. Higher-order masking is difficult for efficiently hardware implementations and flaws [CPRR14] are also found due to over complex designs. Although simple masking is often not enough to prevent successful attacks, it certainly increases the difficulty level and usually achieves better resistance when combined with other countermeasures.

7.2 Dual-Rail Precharge (DRP) Logic

The DRP logic cells are designed to make output switching activities independent of inputs to achieve constant power consumptions at the gate level. In a dual-rail gate, logic `0` and `1` are represented by complementary pairs `(0, 1)` and `(1, 0)` respectively. The precharge state is defined by `(0, 0)`, but `(1, 1)` is not a valid circuit state. The promise is exactly the same count of `0` $\rightarrow$ `1` and `1` $\rightarrow$ `0` transitions appear per gate and per clock cycle, regardless of the input activity. The last four columns of Table 1 describes switching activities of a dynamic `(AND, NAND)` pair. The shaded transitions occur when a *precharge* signal asserts and the unshaded ones happen when the it de-asserts. In dynamic CMOS logic design, the second step is the *evaluation* phase.

Classical dual-rail domino logic gates satisfy the desired pattern of switching. But one essential design metric is to ensure all charging and discharging paths have the same ca-

a	b	AND	NAND	XOR	Dyn. AND		Dyn. NAND	
0 → 0	0 → 0	0 → 0	1 → 1	0 → 0	0 → 0	0 → 0	1 → 0	0 → 1
0 → 0	0 → 1	0 → 0	1 → 1	0 → 1	0 → 0	0 → 0	1 → 0	0 → 1
0 → 0	1 → 0	0 → 0	1 → 1	1 → 0	0 → 0	0 → 0	1 → 0	0 → 1
0 → 0	1 → 1	0 → 0	1 → 1	1 → 1	0 → 0	0 → 0	1 → 0	0 → 1
0 → 1	0 → 0	0 → 0	1 → 1	0 → 1	0 → 0	0 → 0	1 → 0	0 → 1
0 → 1	0 → 1	0 → 1	1 → 0	0 → 0	0 → 0	0 → 1	1 → 0	0 → 0
0 → 1	1 → 0	0 → 0	1 → 1	1 → 1	0 → 0	0 → 0	1 → 0	0 → 1
0 → 1	1 → 1	0 → 1	1 → 0	1 → 0	0 → 0	0 → 1	1 → 0	0 → 0
1 → 0	0 → 0	0 → 0	1 → 1	1 → 0	0 → 0	0 → 0	1 → 0	0 → 1
1 → 0	0 → 1	0 → 0	1 → 1	1 → 1	0 → 0	0 → 0	1 → 0	0 → 1
1 → 0	1 → 0	1 → 0	0 → 1	0 → 0	1 → 0	0 → 0	0 → 0	0 → 1
1 → 0	1 → 1	1 → 0	0 → 1	0 → 1	1 → 0	0 → 0	0 → 0	0 → 1
1 → 1	0 → 0	0 → 0	1 → 1	1 → 1	0 → 0	0 → 0	1 → 0	0 → 1
1 → 1	0 → 1	0 → 1	1 → 0	1 → 0	0 → 0	0 → 1	1 → 0	0 → 0
1 → 1	1 → 0	1 → 0	0 → 1	0 → 1	1 → 0	0 → 0	0 → 0	0 → 1
1 → 1	1 → 1	1 → 1	0 → 0	0 → 0	1 → 0	0 → 1	0 → 0	0 → 0

Table 1: Switching Activities of Gates. For dynamic gates the shaded column indicates the precharge phase.

pacitance. Tiri *et al.,* [TAV02] reported the earliest design of DRP cells in academia for counteracting DPA attacks. On the basis of (modified) dual-rail domino logic, their solution is to use a pair of cross-coupled inverters for capacitance balancing, shown in Figure 6. Since the cross-coupled inverters are also known as a sense amplifier, this transistor-level DRP cell design is called **Sense Amplifier Based Logic (SABL)**. An SABL cell has the following advantages: 1) outputs switching activities independent of inputs, 2) immunity to glitches, and 3) balanced internal capacitances. The immunity to glitches is because during the precharge phase all outputs only monotonically fall, and in the evaluation phase the outputs can only monotonically rise (nature of dynamic logic). However, a design based on SABL cells still has to balance external loads and especially the parasitics on routing wires. To relax the routing constraint, a *Three-Phase Dual-Rail Pre-Charge Logic (TDPL)* [BGLT06] was developed, by enforcing an additional discharge[4] phase after the evaluation. Consequently in TDPL either differential output exactly charges once and discharges once. As a result, unbalanced external load capacitance has a much lower effect.

The SABL design does achieve nearly constant power consumption in circuit simulation but it requires a tedious and expensive full-custom IC design flow, especially for its dynamic logic. Consequently, many later developments of DRP cells intend to use static CMOS standard cells to emulate the switching activities of SABL cells. Tiri *et al.,* [TV04] also introduced the **Wave Dynamic Differential Logic (WDDL)** that is compatible with standard cell based semi-custom ASIC design. The authors first demonstrated a *Simple Dynamic Differential Logic (SDDL)* built by basic static CMOS gates of a standard cell library to emulate the dual-rail precharge behavior. An SDDL `AND` gate is illustrated in Figure 7. The dual gate to the `AND`, the `OR` gate, is derived

[4]Depending on different nodes for circuit analysis, the precharge or discharge could lead to either `0` or `1`. Looking at the outputs of gates, precharge drops all outputs to `0` but discharge brings them to `1`.

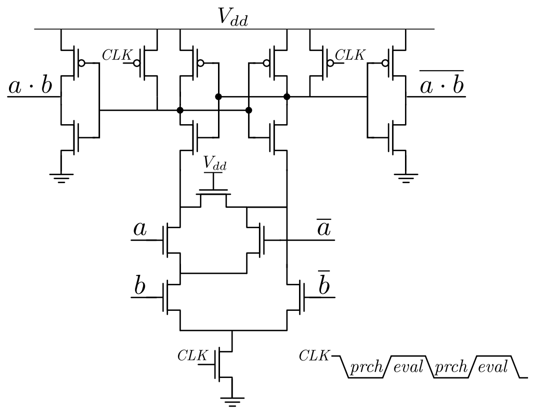


Figure 6: The SABL `AND` Gate

using the De Morgan's law: $\overline{a \cdot b} = \overline{a} + \overline{b}$; and their outputs are `AND`-ed with the precharge signal. To avoid glitching, the dual-rail functions must implement positive monotonic gates (outputs always swing in the same direction as inputs). The original publication [TV04] only allows the `AND` and the `OR` gates[5]. Inversion is inherent in a dual-rail logic gate by swapping the two outputs.

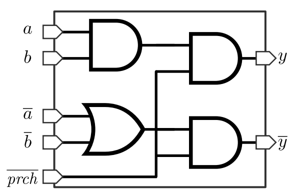


Figure 7: The SDDL `AND` Gate

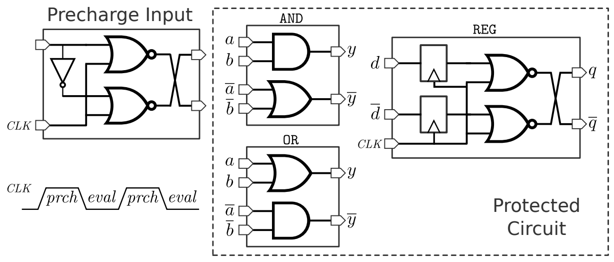


Figure 8: The WDDL Encryption Module Design

By virtue of the positive monotonic gates, the precharge signal does not have to be globally distributed to every combinational gate. During the precharge phase, the inputs of any compound gate are precharged to logic `0`s and any `AND` or `OR` gate outputs logic `0` as well. Accordingly, the two `AND` gates at the output stage of an SDDL gate are removed, and the precharge signal (logic `0`) can propagate like a wave to all combinational gates and stop at clocked flip-flops. Hence comes the name WDDL. The precharge waves are launched from register cells as shown in Figure 8. Inverting gates are not allowed along the way because they halt the `0`-waves. A WDDL gate has the same switching activity as in SABL. But one good property that WDDL loses is the balanced

[5]Any boolean logic can be implemented with the `AND`, `OR`, and `INV` operators.

internal capacitance: the AND and OR gates that compose a WDDL gate are not equal. Therefore, in general not all internal nodes of a WDDL gate are charged and discharged in the same way. The authors argue that with technology shrinking the interconnection capacitance is more dominant, and the focus should be on routing balance.

Although a WDDL design can be implemented using existing standard cells, normal IC design tools still do not natively support it. Suppose a cryptographic circuit is designed by hardware description languages (Verilog or VHDL), the designer first has to restrict the use of target cells to only AND, OR and INV. Once have the synthesized cell netlist, a custom script is executed to replace each gate by its WDDL counterpart and remove inverters by exchanging the outputs. From the gate netlist to the physical layout strict design constraints must be applied, especially for differential wire routing. The constraints for routing the differential wires include but are not limited to the follows [TV06]: 1) always on the same metal layers, 2) always on adjacent routing tracks, 3) always have the same lengths and widths, 4) always have the same number of vias for resistance balancing, 5) must be shielded with V_{dd} or ground lines to remove crosstalk, 6) making every other metal layer a ground plane to control the inter-layer capacitance. Therefore, the designers have to force commercial IC design tools in many ways to meet the physical design constraints. In each design iteration, parasitic capacitance analysis must be rigorous performed. Moreover, due to custom editing of the gate netlist, formal verification and timing closure require greater effort than in a normal IC design. For FPGAs, the constraints are even harder to meet than the ASIC case due to the limited degree of freedom in the FPGA physical design.

An SoC[6] [TV06] using WDDL design as its AES portion was fabricated using a TSMC 0.18-μm process. On the same chip, an insecure AES counterpart exists as well, comprising regular standard cells and regular physical design rules. Both parts have been implemented starting from the same synthesized gate netlist. In the evaluation, for the insecure AES core on average 2,000 power traces are required to disclose a key byte. The AES in WDDL successfully protects 5 out of 16 key bytes up to 1,500,000 power traces; the 11 leaked bytes on average needs 255,000 traces each. The increased security is at a cost of 3x increase in area, 4x increase in power consumption and a limited operating speed up to 125MHz. The residual leakage of this WDDL design was due to imperfect routing and the lack of IC design tool support on security.

In theory, the power consumption of an SABL or WDDL gate would be independent of the data values processed and so thwart power analysis attacks. However in practical IC design the capacitances of the complementary outputs are impossible to be perfectly balanced due to all the complications in logic synthesis, placement, and routing. Even if the most powerful IC design tool can make such a balance at design time when the chips are being fabricated there are still manufacture process variations that effectively break the balance. Consequently, there are always residual leakages and given sufficient number of power traces the secrets can still be extracted. A DES cryptoprocessor implemented in FPGA using WDDL without place and route constraints was successfully attacked by [SGD$^+$09].

[6]The authors claim this is the first practical DPA countermeasure implemented and tested in actual silicon.

7.3 Masked Dual-Rail/Precharge Logic

The strong constraints in IC physical design for DRP logic cells motivate research for more efficient solutions. Suzuki *et al.,* [SSI04] proposed the **Random Switching Logic (RSL)** that combines the mask and the precharge, but it uses single-rail logic. It introduces the precharge signal for partially resisting glitches, and the random switching bit (mask) is employed to avoid dual-rail and routing balance constraints. An RSL gate satisfies two properties: 1) executes masked operations for all input/output signals using the same 1-bit random mask, 2) executes operations while the precharge (like enable) is logic 1; otherwise the output drives 0. For example a 2-input NAND/NOR gate in RSL is expressed by equation (12). The e signal is for precharge, and the m is a 1-bit mask that switches the two functions randomly. When $m = 0$, the RSL-NAND/NOR works as a regular NAND and when $m = 1$ it works as a regular NOR. To encrypt a data byte, the m bit is randomly generated and expanded to a byte, XOR-ed with the input data byte, the round key byte, fed into the protected circuit to switch logic roles, and in the end XOR-ed with the masked output to get the real encrypted value. Because the RSL gates do not confine to positive monotonic functions, the design is not inherently free of glitches. To ensure glitch-free, the precharge signals must de-assert after all input data signals settle. Suzuki *et al.,* claims this can be achieved easily even by automatic place-and-route tools if enough delay time is given.

$$\begin{aligned} \texttt{RSL NAND/NOR}:\ & y_m = \overline{\overline{e} + a_m \cdot b_m + (a_m + b_m) \cdot m} \\ with:\ & a_m = a \oplus m, b_m = b \oplus m, y_m = y \oplus m \end{aligned} \tag{12}$$

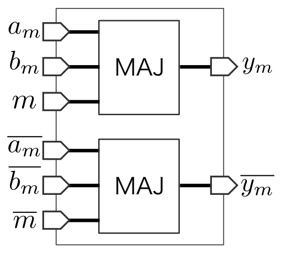
RSL gates do not have uniform switching activity as shown in Table 1, but it intends to hide the data induced variations by randomness. The authors of RSL evaluated their design in FPGA and compared with a WDDL design[7], the RSL needs 1/3 of the area of WDDL with better timing performance. Under DoM based attacks, the WDDL design leaks with 60,000 traces but the RSL resists 200,000 queries.

Masked Dual-Rail Precharge Logic (MDPL) [PM05], as its name explains, intends to combine merits of both mask and DRP to halt DPA attacks and not pose any tedious IC design constraints. The MDPL adopts dual-rail design to have data-independent output switching activity and intends to use a random mask bit to compensate for the loss of routing balance. An MDPL AND gate takes six dual-rail inputs (a_m, $\overline{a_m}$, b_m, $\overline{b_m}$, m, $\overline{m}$) and produces two outputs (y_m, $\overline{y_m}$). The truth table is in Figure 9a and the outputs are computed as $y_m = ((a_m \oplus m) \cdot (b_m \oplus m)) \oplus m$, and $\overline{y_m} = ((\overline{a_m} \oplus \overline{m}) \cdot (\overline{b_m} \oplus \overline{m})) \oplus \overline{m}$ (just for the truth table, the implementation should not be done this way). From the truth table, it is observed that the y_m and $\overline{y_m}$ can be computed by the *majority* function (MAJ) which outputs 1 if

[7]The WDDL design was implemented in the same FPGA, but the authors did not mention any control in balancing the WDDL loads.

a_m	b_m	m	y_m	$\overline{a_m}$	$\overline{b_m}$	$\overline{m}$	$\overline{y_m}$
0	0	0	0	1	1	1	1
0	0	1	0	1	1	0	1
0	1	0	0	1	0	1	1
0	1	1	1	1	0	0	0
1	0	0	0	0	1	1	1
1	0	1	1	0	1	0	0
1	1	0	1	0	0	1	0
1	1	1	1	0	0	0	0

(a)



(b)

Figure 9: (a): The MDPL AND Gate Truth Table (b): The MDPL AND Gate Schematic

the inputs have more 1s than 0s, or it outputs 0. A gate that performs the majority function is supposed to exist in a standard cell library. In MDPL, all signals are precharged to 0 and similar to WDDL the precharge waves are launched from registers and sweep all combinational logic. Since the majority gate is a positive monotonic function, the MDPL gates are immune to glitches. Moreover, because both the dual-rail outputs can be computed by the same majority function, MDPL has internally balanced current paths.

Like RSL, MDPL also uses a single-bit mask m (and its complementary $\overline{m}$) for each data byte encryption. MDPL does not require balanced loads and routing constraints. The imbalance is supposedly compensated by the random masks. The authors simulated their design using a 0.35-μm standard cell library, evaluated that by mounting DoM attacks on the gate-level netlist power simulation and concluded the secret key was not discovered. Compared to a regular but leaky standard cell design, the MDPL version is 4.5x bigger in the area and half the operating speed.

Although SABL, WDDL, RSL and MDPL are all designed to be glitch-free, their implementation still can not prevent different arrival times of signals that depend on cell and route delays. Many of the normal boolean logic gates have the property that their logical outputs can be uniquely determined without necessarily knowing all the inputs. Accordingly a gate sometimes evaluates its logical output early, before waiting for all of its logical inputs (not a timing violation). This shifted time of evaluation, called *early propagation* (or *early evaluation*), causes data-dependent power dynamics that are exploitable. Suzuki and Saeki [SS06] explained the leakage using WDDL and MDPL as examples. They pointed that a cell that avoids early propagation must delay the evaluation moment until all input signals have arrived, e.g., all inputs of MDPL have settled to differential values. Popp *et al.,* [PKZM07] confirmed the leakage on a prototype MDPL chip and proposed improved MDPL (iMDPL) to counteract early propagations. The remedy in iMDPL significantly increased the area compared to MDPL. For example, one more NAND, three more OR and six more NOR gates were added to the MDPL AND gate in Figure 9b.

Tiri and Schaumont [TS07] found another source of leakage in RSL and MDPL that is the mask bit itself. For example in RSL, the encrypting process with $m = 0$ consumes more (or less) power than the encrypting process with $m = 1$. This is expected as the mask bit puts the circuit in one of the two complementary forms. In other words, the distribution of samples in some columns of the power trace matrix $\mathbf{P}$ already looks like Figure 3 with an observable notch in the middle. Finding the time when the two distributions clearly separate deserves some searching effort. Tiri and Schaumont [ST07] suggest folding one part around the notch on top of the other and then perform a regular DoM based attack. Therefore, later research work often refers to this technique as the *folding attack*. The authors used toggle count simulation and proved the concept. Because the Gaussian curve is a probability density function, using it to filter the mask bit is called *probability density function filtering*. For MDPL, a similar approach is applicable too. Practical folding attacks on MDPL were reported by De Mulder *et al.,* [DMGPV09] on a prototype chip. Moradi *et al.,* [MKEP12] further found leakage in iMDPL using the folding attack and a few other advanced mask detections methods. So once the mask effect disappears from RSL or MDPL countermeasures, the circuit downgrades to simple precharge or dual-rail logic without balanced load capacitance and is susceptible to attacks.

7.4 Power Line Isolation

Alternative to manipulating the logic gates, another approach intends to isolate/decouple the cryptographic circuit's power supply from the external voltage source. Shamir [Sha00] introduced a power line decoupling method using two capacitors that work in an alternating manner under a proper switch control. During half the time, one capacitor is charged by the external power source while the other capacitor is discharged by supplying the security chip. During the other half of time, the two capacitors switch roles. Because the charges stored in the supplying capacitor can only sustain a limited number of circuit operations, this paper mainly discussed the feasibility of using a large capacitive but separate circuit module for securing the protected chip. Both can fit in the cavity of a smart card.

Since externally placed switch capacitors are easily tampered by manipulating their pins, later developments focus on pushing them into the same chip. Corsonello *et al.,* [CPM06] proposed an integrated *charge pump* also using switches and capacitors. Although not evaluated under practical attacks, the authors claimed a strong effectiveness of their design based on simulation results.

Tokunaga and Blaauw [TB10] introduced a switching capacitor based current equalizer and practically evaluated that in actual silicon. Figure 10a shows the design consisting three switching capacitor modules, each having a 100-pF capacitor C_P for charge storage and three controlled pass-transistor switches namely S_{Supply}, S_{Logic}, and S_{Shunt}. Each capacitor module has three cyclic switching states: 1) Only S_{Supply} conducts and replenishes charge from the external supply, 2) Only S_{Logic} is closed to provide charge for the protected circuit, and 3) Only S_{Shunt} conducts and provides a path for C_P to discharge. The three modules have staggered switching pattern, shown in Figure 10b, to ensure uninterrupted operation of the protected circuit. The storage capacitor was implemented using a combination of NMOS and metal-to-metal capacitors. During a discharge phase, its voltage is monitored by a comparator, and when its voltage discharges to a preset constant V_{Ref}, a trigger will open S_{Shunt} to stop discharging. Thus V_{Ref} is to ensure no remainder variable

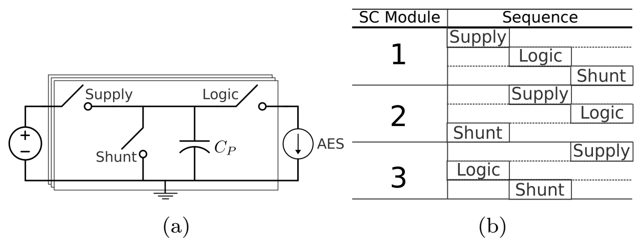


Figure 10: (a): There Switching Capacitor Modules of [TB10] (b): The Switching Pattern Diagram. Named blocks indicate conducting of corresponding switches.

voltage is detectable at the beginning of the next charging phase, and it prevents discharging to zero to save energy.

The test chip fabricated using a 0.13-μm CMOS process has a switching capacitor protected AES core as well as a unprotected one for practical comparison. The chip design flow allows the AES core placed and routed using standard CAD tools, and the current equalizer module is integrated after that. Before fabrication, some mixed-signal circuit simulation was performed to verify the correct operations. The protected AES circuit did not leak any secret during CPA attacks up to 10 million power traces while the insecure AES core breaks at 10,000 measurements. The switching capacitor block incurs a 7.2% increase in the area, 33% increase in the power consumption, and half the performance loss.

While [TB10] was designed as a bulk unit sitting aside the protected circuit, Gornik *et al.,* [GMOP15] bring switching capacitors to individual logic gates. They pointed out two issues of [TB10]: 1) its effectiveness depends on the quality of switches, which in semiconductor is not perfect and their sub-threshold currents in cutoff mode is exploitable by adversaries. 2) the mixed-signal design flow is hardly reused for other chips. To resolve the first problem, Gornik *et al.,* improved the external cutoff circuit (Figure 11) to reduce the impact of sub-threshold currents in switches. Compared to Figure 10a, the S_1 and S_3 switches work like the S_{Supply}, the S_4 is equivalent to the S_{Shunt}, and the S_2 and S_5 switches resemble the S_{Logic}. S_2 is the improvement to accommodate sub-threshold currents of S_1 and S_3: when S_1 and S_3 are open but leaking, S_2 should effectively lead the currents to ground to avoid crosstalk between V_{dd} and $V_{dd,Int}$.

For effortless integration in digital IC design, the switching capacitors are compressed into a new standard cell and can be distributed to each logic gate to supply it for several transitions. The concept is to design the decoupling cell once, and it can be reused and compatible with automated layout tools. However, using their target 0.15-μm CMOS technology the decoupling cell was 259.78 μm^2, 10x larger than a 2-input `XOR` gate of the standard cell library. Existing physical design tools did not support automated layout of such decoupling cells, and so they performed manual placement and routing of the test circuit which incurred only moderate effort due to its small size. The authors anticipated the area overhead could be reduced using a smaller technology node, and they could use automation tools for future chip layout. The detailed transistor-level design of the cell is in [GMOP15] but the paper does not disclose specifications for

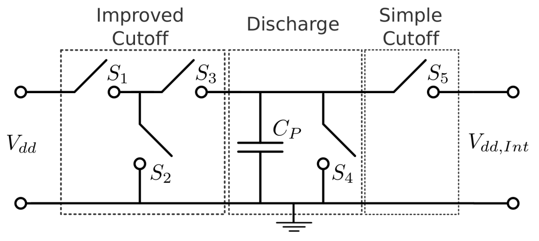


Figure 11: The Decoupling Circuit of [GMOP15]

C_P, e.g., how large capacitance it should have.

Chips each having a protected S-Box of the PRESENT block cipher were fabricated and evaluated. Under assessment, the batch of chips from the same design, same wafer, and the same package exhibited quite diverse resistance to DPA attacks. Some of them showed a high level of robustness while some of them provided nearly no protection. The authors analyzed their chips, and they thought the reason was the quality of chip bondings. They received the manufactured dies unpackaged and performed packaging and wedge bonding themselves. By repeating the bonding process for the same test chip, they confirmed different results using the same setup for power line attacks. The bonding issue was expected to be resolved using advanced bonding machines.

Two things the recent publications did not precisely discuss are: 1) the switch control signal generation, and 2) possible leakage through non-power pins. A figure of the [TB10] paper indicates it uses an on-chip ring oscillator for the switch control clock generation and a programmable pattern generator for each switching capacitor module. The [GMOP15] work exposed the switch controls to chip I/O pins and let a microprocessor generate the control signals. Supposedly the exposed control signals are for the testing chip only, for practical defense they should be implemented on-chip to avoid adversarial manipulations. Since a ring oscillator's frequency is sensitive to temperature, and so are the clock skews to switches, it is unclear if switching capacitors would leak information when their activities are not as well aligned as shown in Figure 10b. Prior to [TB10] and [GMOP15], Plos [Plo09] evaluated early designs using switching capacitors and identified data-dependent information can also leak through a device's I/O pins (non-power pin), due to coupling effects (crosstalk) where the switching activity of one wire influences the voltage of neighboring wires. The [TB10] paper did not address this problem; attack measurements were only performed on a serial resistor in the chip's V_{dd} supply line. On the other hand [GMOP15] cited the Plos paper yet still did not explicitly test their design's I/O leakage. The power measurements were performed by a current probe also in the power line only.

7.5 Noise Generation

In the time dimension, the most commonly used techniques are randomly inserting dummy instructions and shuffling the order of operations, and as a result power traces are not aligned anymore. For example, a microcontroller-based AES cipher can insert random dummy instructions before and after the S-Box lookup, and the execution sequence of the 16 S-Boxes is different for each data block encryption. The price to pay for using dummy instructions is the throughput of the cipher, and on the other hand shuffling does not affect the throughput much but the operations that can effectively shuffle are limited. Multiple other approaches also try to manipulate the clocks of the protected circuit, including: 1) use clock gating technique to skip randomly clock pulses [BLOW10], 2) randomly change the clock frequency [PH+10] and phase [GM11], 3) use multiple clock domains for different parts of the protected circuit [BVR+13]. Another form of temporal distortion coupled with voltage scaling is discussed in Section 7.6.

For countermeasures that change the timing behavior of power traces, it is critical that the attackers can not identify them. Dummy instructions such as `NOP`s have detectable low power profile than others, and the attacker can quickly filter them and restore the alignment of power traces. Due to the limited degree of shuffling options, the right order of segments in a power trace can be recovered by signal processing. Pattern matching techniques are commonly used, such as using a sliding window to perform correlation tests on one power trace with respect to a reference. Clavier *et al.,* [CCD00] developed a sliding window based integration method that can effectively attack shuffled power traces. Newer and faster alignment techniques have also been published such as the elastic alignment [VWWB11] and the horizontal alignment [TH12]. In general, the insertion of dummy instructions and shuffling do not provide a high level of protection against power analysis. A technique to identify changing clock frequencies is in Section 7.6.

In the amplitude dimension, the obvious goal is to increase the noise variance $Var(P_{sw.noise} + P_{el.noise})$ so that the SNR drops. Software, if possible, can run multiple independent threads in parallel. Besides parallelism hardware architecture also uses dedicated noise engines that perform random switching activities while the cryptographic circuit operates. A variety of ways can generate excess noise on the chip. Le Masle *et al.,* [LMCL11] use long wires with many buffers along the route, and feed each wire with controlled switching activities to draw currents in the buffers. Güneysu *et al.,* [Gün10] find in dual-ported memories write contentions result in metastability within the storage cells and lead to increased power consumption. They also developed a way to modify the Xilinx FPGA bitstream to generate controlled short circuits for a very limited amount of time. It is important for noise creators to cover uniformly the entire protected circuit but not only sit in a corner of the chip. Cornered noises may only affect measurements from nearby power lines, yet still leave backdoors in other power/ground pins and electromagnetic emissions. Another critical requirement for amplitude noises is they must have the identical frequency spectrum as the exploitable power; otherwise they are simply removed by a frequency filter. Equation (10) shows the correlation coefficient in CPA is proportional to $\sqrt{SNR}$, and so empirically added amplitude noise results in quadratic number of power traces to get the same attack outcomes. In practice, no countermeasure can make noise variance to infinity, and for CPA only the relative amplitude of correlation coefficients matters not the absolute. So amplitude noise is often considered ([MOP08] and [GM11]) not an optimum way to thwart power side-channel attacks. They play more assisting roles in defense.

7.6 Random Voltage and Frequency Scaling

The Dynamic Voltage and Frequency Scaling (DVFS) technique is generally used for digital systems to achieve power efficiency. From the defense perspective, it could be randomized to halt power side-channel attacks. Compared to amplitude noises that mainly manipulate C_L, scrambling both V_{dd} and f is supposedly more efficient ($P_{dyn} \propto f \cdot C_L \cdot V_{dd}^2$). Yang *et al.,* [YWV+05] first introduced DVFS for counteracting DPA attacks. Their proposed cryptosystem consists of three parts: 1) a cryptoprocessor, 2) a DVFS Feedback Loop (DVFSL), and 3) a DVFS Scheduler (DVFSS). During critical operations of the cryptoprocessor the DVFSS unit randomly generates (V_{dd}, f) configurations and sends them to the DVFSL, which implements the support for DVFS. The DVFSL unit thereafter can apply the (V_{dd}, f) changes to the cryptoprocessor with optional random delays in between. The authors proved their concept using a software implementation of the DES cipher running by the SimplePower tool, an RTL power estimator. No actual power analysis attacks were mounted. Using custom metrics for power trace properties the authors claimed the effectiveness of randomized DVFS (RDVFS) as a DPA-resistant technique.

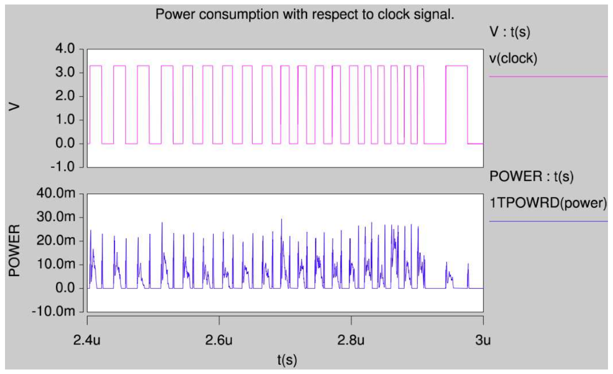


Figure 12: Frequency Detection [BZ07] of RDVFS. Bottom: Simulated Power Traces with RDVFS. Top: The Predicted Clock Frequencies.

Baddam *et al.,* [BZ07] evaluated the concept of [YWV+05] using more realistic experiment setup and actual DPA attacks. They implemented an RDVFS-enabled AES (partial) test circuit using an AMS 0.35-μm technology library and run the SPICE netlist simulation without routing parasitics (for simulation speed). The simulated power traces were then tested using DoM and CPA based attacks. In the power traces, they notice distinguishable higher spikes, each followed by a slope of setting down towards the next one (Figure 12 Bottom). The reason is most transitions in a sequential circuit happen right after clock edges and cause a burst of currents. Therefore, the clock frequencies can be sequenced by reading the spikes in the power traces (Figure 12 Top). Because in DVFS commonly there is a one-to-one mapping between each operating voltage and frequency, the voltages of RDVFS can be derived. The attacker then scales power models accordingly and concludes successful key extraction. This experiment also confirms that simply manipulating clocks as temporal noises do not provide much protection. Observing the limitations of RDVFS Baddam *et al.,* tried only to scale the voltage while keeping the frequency constant. However for their experiment with 10,000 encryption rounds the correlation strength was never below a point where the key was undetectable.

Avirneni and Somani [AS13] tried to overcome the limitations of RDVFS by introducing Aggressive Voltage Scaling (AVS) and Aggressive Frequency Scaling (AFS). These techniques break the one-to-one connection between V_{dd} and f. For operating frequency beyond the timing margins, the authors suggest self-correction circuit design. For the evaluation, they set up SPICE simulation using one S-Box circuit of AES and apply random (V_{dd}, f) pairs for collecting 10,000 power traces. The traces are evaluated by CPA based attacks. The same experiment was repeated 500 times and only two times the actual key had the strongest correlation.

DVFS shows potential for mitigating power side-channel attacks but the results are still preliminary. For commercial CPUs and FPGAs that already have DVFS support the random DVFS methods should be feasible options. Otherwise, the on-chip design of the voltage regulator, clock management, and error-correction logic is not yet a trivial task for custom ASICs.

7.7 Combining Countermeasures

It is widely accepted that a single countermeasure can not effectively protect the security hardware against a variety of side-channel attacks. Accordingly, a straightforward thought is to combine protections. Unfortunately, some countermeasures do not simply add up, as explained by the folding attack in Section 7.3. So choosing a combination of suitable countermeasures is a rather challenging and tedious work in practice. A positive example of using combined mitigations is proposed by Güneysu *et al.,* [GM11] for FPGAs. They use signal masking, clock randomization, and amplitude noise together for protection. The design could resist a variety of first-order DPA attacks, with up to 100 million power traces. Combining masked and precharge/dual-rail logic gates is another popular direction. Besides the RSL and MDPL, *LUT-Masked Dual-Rail with Precharge Logic (LMDPL)* [LMW14] is the most recent tested hardware implementation by the time of writing this survey. LMDPL uses on gate-level masking, dual-rail precharge logic without the need for routing constraints. It employs monotonic gates to avoid glitches, and it fights against leakage of mask bits and early propagation using a novel *activity image analysis* for combinational data paths. In its FPGA evaluation, it shows no significant leaks up to 200 million power traces.

Obviously using multiple mitigations further increases the circuit area, power, cost and complicates the design flow. Improving the security level is never free and never perfect. The cost and effort for the defense should match the value of the device. For a chosen combination of mitigations, it should still be used with corresponding key updating policies to make allowed power trace queries of the same key within its tolerance.

8. SUMMARY

Making a device resistant to power side-channel attacks is not trivial. In academic publications, resistant logic styles have been the most popular form of hardware countermeasures. Moreover, according to [MOP08] in the semiconductor industry, counteracting power side-channel attacks at the cell level was one of the first reactions. It is reasonable because logic styles are close to solving the information leaks fundamentally in power traces. Hence, this survey reviewed foundational designs using logic style countermeasure and their vulnerabilities. Since the exposure of flaws in DPA-resistant logic styles, a large number of proposals have been raised for improvements, at a higher cost of the area, speed, and power. Some logic styles such as self-timed (asynchronous) logic design [TMA+02] and current-mode logic [AE01] are not discussed because they are not quite compatible with contemporary mainstream IC design flows.

Although numerous logic style countermeasures exist, each of them features a unique set of overheads and constraints. The lack of security support in CAD tools has limited the implementation of many defensive circuit techniques. Hardware defenders have to swap gates, use customized routers and dictate existing tools for specialized layout requirements. Moreover extra effort on functional and timing verification has been incurred for secure implementations. To circumvent complications in the design flow the rest countermeasures are engaged with separate objects to relax the constraints, for example, random number generators, clocks, and voltage regulators that themselves must be tamper-resistant. Once random numbers are biased, or clock sources are compromised, it is anticipated that corresponding countermeasures are weakened. Table 2 summarizes the potential vulnerabilities in existing hardware countermeasures. The report aims to provide a survey of exposed vulnerabilities in hardware-based countermeasures, and serve as a reference for future more secure IC implementations.

9. ACKNOWLEDGEMENT

This work was partially supported by NSF Awards 1018850, and 1228992, and by C-FAR, part of STARnet, a Semiconductor Research Corporation program. We also thank the members of Michael Taylor's Bespoke Silicon Group for feedback on the paper.

10. REFERENCES

[AE01] Mohamed W Allam and Mohamed I Elmasry. Dynamic Current Mode Logic (DyCML): A New Low-Power High-Performance Logic Style. *Solid-State Circuits, IEEE Journal of*, 36(3):550–558, 2001.

[ARRS05] Dakshi Agrawal, Josyula R Rao, Pankaj Rohatgi, and Kai Schramm. Templates as Master Keys. In *Cryptographic Hardware and Embedded Systems - CHES 2005*, pages 15–29. Springer, 2005.

[AS13] Naga Durga Prasad Avirneni and Arun K Somani. Countering Power Analysis Attacks using Reliable and Aggressive Designs. *IEEE Transactions on Computers*, 99:1, 2013.

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation Power Analysis with a Leakage Model. In *Cryptographic Hardware and Embedded Systems - CHES 2004*, pages 16–29. Springer, 2004.

[BGLT06] Marco Bucci, Luca Giancane, Raimondo Luzzi, and Alessandro Trifiletti. Three-Phase Dual-Rail Pre-Charge Logic. In *Cryptographic Hardware and Embedded Systems - CHES 2006*, pages 232–241. Springer, 2006.

[BLOW10] Kean Hong Boey, Yingxi Lu, Máire O'Neill, and Roger Woods. Random Clock against

Countermeasures	Potential Vulnerabilities
Masked Logic	**Glitches:** Gate toggles before final values settle can leak information. **Higher-Order Leaks:** Combining samples at multiple times can remove the effect of masks. **Random Numbers:** Random number generators can be biased to weaken the countermeasure.
DRP Logic	**Routing Imbalance:** Imbalanced load and parasitic capacitances cause leakage in power traces. **Early Propagation:** Gates evaluate output before all inputs settle can leak information.
Masked DRP Logic	**Masking Bits:** Masking bits could be detected and removed by a variety of means. **Routing Imbalance:** Imbalanced load and parasitic capacitances cause leakage in power traces. **Early Propagation:** Gates evaluate output before all inputs settle can leak information. **Nonuniform Switchings:** If masking bits are compromised, nonuniform switchings can be attacked. **Random Numbers:** Random number generators can be biased to weaken the countermeasure.
Switching Capacitors	**Switches:** Imperfect semiconductor switches can leak information when they are open. **Control Signals:** Signals controlling capacitor switchings can be biased to weaken the countermeasure.
Temporal Noises	**Detectable Patterns:** Signal processing techniques can detect and remove temporal distortions. **Random Numbers:** Random number generators can be biased to weaken the countermeasure.
Amplitude Noises	**Amplitude Variance:** Amplitude variances can not be infinity and have limited effectiveness. **Frequency Spectrum:** Noises having different frequency components than signals can be filtered. **Random Numbers:** Random number generators can be biased to weaken the countermeasure.
RDVFS	**Detectable Patterns:** Signal processing techniques can detect and remove temporal distortions. **Frequency-Voltage Bonding:** Once frequencies are detected, voltages can be derived and scaled. **Random Numbers:** Random number generators can be biased to weaken the countermeasure.

Table 2: Existing Hardware Countermeasures and Potential Vulnerabilities

Differential Power Analysis. In *Circuits and Systems (APCCAS), 2010 IEEE Asia Pacific Conference on*, pages 756–759. IEEE, 2010.

[BVR+13] Ali Galip Bayrak, Nikola Velickovic, Francesco Regazzoni, David Novo, Philip Brisk, and Paolo Ienne. An EDA-Friendly Protection Scheme against Side-Channel Attacks. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013*, pages 410–415. IEEE, 2013.

[BZ07] Karthik Baddam and Mark Zwolinski. Evaluation of Dynamic Voltage and Frequency Scaling as a Differential Power Analysis Countermeasure. In *VLSI Design, 2007. Held jointly with 6th International Conference on Embedded Systems., 20th International Conference on*, pages 854–862. IEEE, 2007.

[CCD00] Christophe Clavier, Jean-Sébastien Coron, and Nora Dabbous. Differential Power Analysis in the Presence of Hardware Countermeasures. In *Cryptographic Hardware and Embedded Systems - CHES 2000*, pages 252–263. Springer, 2000.

[CPM06] Pasquale Corsonello, Stefania Perri, and Martin Margala. An Integrated Countermeasure against Differential Power Analysis for Secure Smart-Cards. In *Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on*, pages 4–pp. IEEE, 2006.

[CPRR14] Jean-Sébastien Coron, Emmanuel Prouff, Matthieu Rivain, and Thomas Roche. Higher-Order Side Channel Security and Mask Refreshing. In *Fast Software Encryption*, pages 410–424. Springer, 2014.

[CRR03] Suresh Chari, Josyula R Rao, and Pankaj Rohatgi. Template Attacks. In *Cryptographic Hardware and Embedded Systems - CHES 2002*, pages 13–28. Springer, 2003.

[DMGPV09] Elke De Mulder, Benedikt Gierlichs, Bart Preneel, and Ingrid Verbauwhede. Practical DPA Attacks on MDPL. In *Information Forensics and Security, 2009. WIFS 2009. First IEEE International Workshop on*, pages 191–195. IEEE, 2009.

[GHP04] Sylvain Guilley, Philippe Hoogvorst, and Renaud Pacalet. Differential Power Analysis Model and Some Results. In *Smart Card Research and Advanced Applications Vi*, pages 127–142. Springer, 2004.

[GJJ+11] Gilbert Goodwill, Benjamin Jun, Joshua Jaffe, Pankaj Rohatgi, and Others. A Testing Methodology for Side-Channel Resistance Validation. In *NIST Non-Invasive Attack Testing Workshop*, 2011.

[GM11] Tim Güneysu and Amir Moradi. Generic Side-Channel Countermeasures for Reconfigurable Devices. In *Cryptographic Hardware and Embedded Systems - CHES 2011*, pages 33–48. Springer, 2011.

[GMOP15] Andreas Gornik, Amir Moradi, Jurgen Oehm, and Christof Paar. A Hardware-based Countermeasure to Reduce Side-Channel Leakage-Design, Implementation, and Evaluation. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 2015.

[Gün10] Tim Güneysu. Using Data Contention in Dual-Ported Memories for Security Applications. *Journal of Signal Processing Systems*, pages 1–15, 2010.

[ISW03] Yuval Ishai, Amit Sahai, and David Wagner. Private Circuits: Securing Hardware against Probing Attacks. In *Advances in Cryptology - CRYPTO 2003*, pages 463–481. Springer, 2003.

[JK12] Benjamin Jun and Gary Kenworthy. Is Your Mobile Device Radiating Keys? In *RSA Conference*, volume 2, page 2012, 2012.

[KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In *Advances in Cryptology - CRYPTO'99*, pages 388–397. Springer, 1999.

[KJJR11] Paul Kocher, Joshua Jaffe, Benjamin Jun, and Pankaj Rohatgi. Introduction to Differential Power Analysis. *Journal of Cryptographic Engineering*, 1(1):5–27, 2011.

[LMCL11] Adrien Le Masle, Gary Chun Tak Chow, and Wayne Luk. Constant Power Reconfigurable Computing. In *Field-Programmable Technology (FPT), 2011 International Conference on*, pages 1–8. IEEE, 2011.

[LMW14] Andrew J Leiserson, Mark E Marson, and Megan A Wachs. Gate-Level Masking under a Path-Based Leakage Metric. In *Cryptographic Hardware and Embedded Systems - CHES 2014*, pages 580–597. Springer, 2014.

[MBMT13] Hassen Mestiri, Noura Benhadjyoussef, Mohsen Machhout, and Rached Tourki. A Comparative Study of Power Consumption Models for CPA Attack. *International Journal of Computer Network and Information Security (IJCNIS)*, 5(3):25, 2013.

[MBZ$^+$12] Mohamed Saied Emam Mohamed, Stanislav Bulygin, Michael Zohner, Annelie Heuser, Michael Walter, and Johannes Buchmann. Improved Algebraic Side-Channel Attack on AES. In *Hardware-Oriented Security and Trust (HOST), 2012 IEEE International Symposium on*, pages 146–151. IEEE, 2012.

[Mes00] Thomas S Messerges. Using Second-Order Power Analysis to Attack DPA Resistant Software. In *Cryptographic Hardware and Embedded Systems - CHES 2000*, pages 238–251. Springer, 2000.

[MKEP12] Amir Moradi, Mario Kirschbaum, Thomas Eisenbarth, and Christof Paar. Masked Dual-Rail Precharge Logic Encounters State-of-the-Art Power Analysis Methods. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 20(9):1578–1589, 2012.

[MME10] Amir Moradi, Oliver Mischke, and Thomas Eisenbarth. Correlation-Enhanced Power Analysis Collision Attack. In *Cryptographic Hardware and Embedded Systems - CHES 2010*, pages 125–139. Springer, 2010.

[MOP08] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*, volume 31. Springer Science & Business Media, 2008.

[Mor14] Amir Moradi. Side-Channel Leakage through Static Power-Should We Care about in Practice? *IACR Cryptology ePrint Archive*, 2014:25, 2014.

[MPG05] Stefan Mangard, Thomas Popp, and Berndt M Gammel. Side-Channel Leakage of Masked CMOS Gates. In *Topics in Cryptology - CT-RSA 2005*, pages 351–365. Springer, 2005.

[MPO05] Stefan Mangard, Norbert Pramstaller, and Elisabeth Oswald. Successfully Attacking Masked AES Hardware Implementations. In *Cryptographic Hardware and Embedded Systems - CHES 2005*, pages 157–171. Springer, 2005.

[MS00] Rita Mayer-Sommer. Smartly Analyzing the Simplicity and the Power of Simple Power Analysis on Smartcards. In *Cryptographic Hardware and Embedded Systems - CHES 2000*, pages 78–92. Springer, 2000.

[MSSE09] Amir Moradi, Mahmoud Salmasizadeh, Mohammad Taghi Manzuri Shalmani, and Thomas Eisenbarth. Vulnerability Modeling of Cryptographic Hardware to Power Analysis Attacks. *Integration, the VLSI Journal*, 42(4):468–478, 2009.

[Nat01] National Institute of Standards and Technology. Advanced Encryption Standard (AES). *FIPS-197*, 2001.

[NRR06] Svetla Nikova, Christian Rechberger, and Vincent Rijmen. Threshold Implementations against Side-Channel Attacks and Glitches. In *Information and Communications Security*, pages 529–545. Springer, 2006.

[NSN$^+$14] Yuto Nakano, Youssef Souissi, Robert Nguyen, Laurent Sauvage, Jean-Luc Danger, Sylvain Guilley, Shinsaku Kiyomoto, and Yutaka Miyake. A Pre-processing Composition for Secret Key Recovery on Android Smartphone. In *Information Security Theory and Practice. Securing the Internet of Things*, pages 76–91. Springer, 2014.

[OMHT06] Elisabeth Oswald, Stefan Mangard, Christoph Herbst, and Stefan Tillich. Practical Second-Order DPA Attacks for Masked Smart Card Implementations of Block Ciphers. In *Topics in Cryptology - CT-RSA 2006*, pages 192–207. Springer, 2006.

[PH$^+$10] Jihan Park, Dongsoo Har, et al. Random Clocking Induced DPA Attack Immunity in FPGAs. In *2010 IEEE International Conference on Industrial Technology*, pages 1068–1070, 2010.

[PKZM07] Thomas Popp, Mario Kirschbaum, Thomas Zefferer, and Stefan Mangard. *Evaluation of the Masked Logic Style MDPL on a Prototype Chip*. Springer, 2007.

[Plo09] Thomas Plos. Evaluation of the Detached Power Supply as Side-Channel Analysis Countermeasure for Passive UHF RFID Tags. In *Topics in Cryptology–CT-RSA 2009*, pages 444–458. Springer, 2009.

[PM05] Thomas Popp and Stefan Mangard. Masked Dual-Rail Pre-charge Logic: DPA-Resistance without Routing Constraints. In *Cryptographic Hardware and Embedded Systems - CHES 2005*, pages 172–186. Springer, 2005.

[Pro05] Emmanuel Prouff. DPA Attacks and S-Boxes. In *Fast Software Encryption*, pages 424–441. Springer, 2005.

[PSQ07] Eric Peeters, François-Xavier Standaert, and Jean-Jacques Quisquater. Power and Electromagnetic Analysis: Improved Model, Consequences and Comparisons. *Integration, the VLSI Journal*, 40(1):52–60, 2007.

[RE10] Wolfgang Rankl and Wolfgang Effing. *Smart Card Handbook, 4th Edition*. John Wiley & Sons, 2010.

[RP10] Matthieu Rivain and Emmanuel Prouff. Provably Secure Higher-Order Masking of AES. In *Cryptographic Hardware and Embedded Systems - CHES 2010*, pages 413–427. Springer, 2010.

[RS10] Mathieu Renauld and François-Xavier Standaert. Algebraic Side-Channel Attacks. In *Information Security and Cryptology*, pages 393–410. Springer, 2010.

[SGD$^+$09] Laurent Sauvage, Sylvain Guilley, Jean-Luc Danger, Yves Mathieu, and Maxime Nassar. Successful Attack on an FPGA-Based WDDL DES Cryptoprocessor without Place and Route Constraints. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 640–645. European Design and Automation Association, 2009.

[Sha00] Adi Shamir. Protecting Smart Cards from Passive Power Analysis with Detached Power Supplies. In *Cryptographic Hardware and Embedded Systems - CHES 2000*, pages 71–77. Springer, 2000.

[SLFP04] Kai Schramm, Gregor Leander, Patrick Felke, and Christof Paar. A Collision-Attack on

AES. In *Cryptographic Hardware and Embedded Systems - CHES 2004*, pages 163–175. Springer, 2004.

[SLP05] Werner Schindler, Kerstin Lemke, and Christof Paar. A Stochastic Model for Differential Side Channel Cryptanalysis. In *Cryptographic Hardware and Embedded Systems–CHES 2005*, pages 30–46. Springer, 2005.

[SMY09] François-Xavier Standaert, Tal G Malkin, and Moti Yung. A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks. In *Advances in Cryptology - EUROCRYPT 2009*, pages 443–461. Springer, 2009.

[SS06] Daisuke Suzuki and Minoru Saeki. Security Evaluation of DPA Countermeasures using Dual-Rail Pre-Charge Logic Style. In *Cryptographic Hardware and Embedded Systems - CHES 2006*, pages 255–269. Springer, 2006.

[SSI04] Daisuke Suzuki, Minoru Saeki, and Tetsuya Ichikawa. Random Switching Logic: A Countermeasure against DPA based on Transition Probability. *IACR Cryptology ePrint Archive*, 2004:346, 2004.

[ST07] Patrick Schaumont and Kris Tiri. *Masking and Dual-Rail Logic Don't Add Up*. Springer, 2007.

[TAV02] Kris Tiri, Moonmoon Akmal, and Ingrid Verbauwhede. A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to withstand Differential Power Analysis on Smart Cards. In *Solid-State Circuits Conference, 2002. ESSCIRC 2002. Proceedings of the 28th European*, pages 403–406. IEEE, 2002.

[TB10] Carlos Tokunaga and David Blaauw. Securing Encryption Systems with a Switched Capacitor Current Equalizer. *Solid-State Circuits, IEEE Journal of*, 45(1):23–31, 2010.

[TH12] Qizhi Tian and Sorin A Huss. A General Approach to Power Trace Alignment for the Assessment of Side-Channel Resistance of Hardened Cryptosystems. In *Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), 2012 Eighth International Conference on*, pages 465–470. IEEE, 2012.

[TMA$^+$02] George Taylor, Simon Moore, Ross Anderson, Robert Mullins, and Paul Cunningham. Improving Smart Card Security using Self-Timed Circuits. In *2002 IEEE 18th International Symposium on Asynchronous Circuits and Systems*, pages 211–211. IEEE Computer Society, 2002.

[Tri03] Elena Trichina. Combinational Logic Design for AES SubByte Transformation on Masked Data. *IACR Cryptology ePrint Archive*, 2003:236, 2003.

[TS07] Kris Tiri and Patrick Schaumont. Changing the Odds against Masked Logic. In *Selected Areas in Cryptography*, pages 134–146. Springer, 2007.

[TV04] Kris Tiri and Ingrid Verbauwhede. A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation. In *Proceedings of the conference on Design, automation and test in Europe - Volume 1*, page 10246. IEEE Computer Society, 2004.

[TV06] Kris Tiri and Ingrid Verbauwhede. A Digital Design Flow for Secure Integrated Circuits. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 25(7):1197–1208, 2006.

[VWWB11] Jasper GJ Van Woudenberg, Marc F Witteman, and Bram Bakker. Improving Differential Power Analysis by Elastic Alignment. In *Topics in Cryptology–CT-RSA 2011*, pages 104–119. Springer, 2011.

[WOL02] Johannes Wolkerstorfer, Elisabeth Oswald, and Mario Lamberger. An ASIC implementation of the AES SBoxes. In *Topics in CryptologyCT-RSA 2002*, pages 67–78. Springer, 2002.

[YWV$^+$05] Shengqi Yang, Wayne Wolf, Narayanan Vijaykrishnan, Dimitrios N Serpanos, and Yuan Xie. Power Attack Resistant Cryptosystem Design: A Dynamic Voltage and Frequency Switching Approach. In *Design, Automation and Test in Europe, 2005. Proceedings*, pages 64–69. IEEE, 2005.

APPENDIX

A. THE AES CIPHER

The AES [Nat01] block cipher encrypts 128 bits fixed block size of data, using a *cipher key* that can be either 128, 192 or 256 bits long. The differences between the uses of the three key sizes are the number of encryption/decryption *rounds* and the *key expansion* process. The three different AES key sizes use 10, 12, and 14 rounds respectively. This paper mainly discusses the AES-128 version that uses a 128-bit cipher key.

A.1 The Round Functions

Each round of the AES encryption comprises of four different functions, named as *AddRoundKey*, *SubBytes*, *ShiftRows*, and *MixColumns*. The 128-bit data being operated in each round contains 16 bytes $\{D_1, D_2, \cdots, D_{16}\}$, known as the *state*. An AES state is stored as intermediate values in memory units for subsequent transformations. The encryption process of one data block starts with the plaintext state and performs an initial AddRoundKey operation. Then nine rounds are executed, and each round sequentially performs SubBytes, ShiftRows, MixColums, and AddRoundKey on the state. After that, a *last round* is executed with only three functions: SubBytes, ShiftRows, and AddRoundKey. The encryption flow of AES is in Figure 13, and each function of an AES round is briefly explained below.

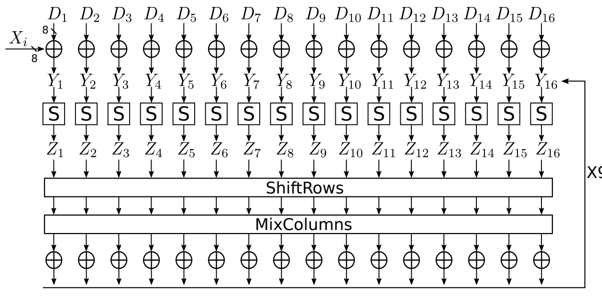

D_1 D_2 D_3 D_4 D_5 D_6 D_7 D_8 D_9 D_{10} D_{11} D_{12} D_{13} D_{14} D_{15} D_{16}
X_i
8
8
Y_1 Y_2 Y_3 Y_4 Y_5 Y_6 Y_7 Y_8 Y_9 Y_{10} Y_{11} Y_{12} Y_{13} Y_{14} Y_{15} Y_{16}
S
Z_1 Z_2 Z_3 Z_4 Z_5 Z_6 Z_7 Z_8 Z_9 Z_{10} Z_{11} Z_{12} Z_{13} Z_{14} Z_{15} Z_{16}
X9
ShiftRows
MixColumns

Figure 13: The AES Encryption

AddRoundKey: In AES-128 a *round key* is also a 128-bit value that is derived from the cipher key by the key expansion algorithm. Round keys are different in each round. The AddRoundKey function performs a bitwise XOR ($D_i \oplus X_i$, $i \in [1 \cdots 16]$) of a state byte D_i and a corresponding round key byte X_i. Note that the round key used in the initial AddRoundKey operation is identical to the cipher key.

$$S(Y_i) = \mathbf{A} \cdot Y_i^{-1} + \mathbf{b} \qquad (13)$$

SubBytes: Also known as the S-Box, this function is to perform byte substitution on each byte of the state *independently* after the AddRoundKey operation. The function (13) for each state byte (Y_i) is to compute its multiplicative inverse (Y_i^{-1}) in the finite field of $GF(2^8)$, followed by an affine transform. Definitions of binary matrix $\mathbf{A}$ and vector $\mathbf{b}$ are given by [Nat01]. Due to the computational complexity of finite field inversions and matrix multiplications, the S-Box is often precomputed as a lookup table and stored in memories for quick reference in each round.

ShiftRows: This transformation views the 16 state bytes as a 4×4 array and performs cyclic left shifts for each row of the state bytes. The first state row is not shifted, the second row is shifted to the left by one byte position, the third row is shifted by two byte positions and the fourth row is shifted by three.

MixColumns: This column-wise function provides further mixing of bytes in a state. It produces a new state column using a constant matrix defined by AES to multiply a previous state column.

The state gets updated by each round function, and after ten rounds of transformations the initial 128-bit plaintext is turned into the 128-bit ciphertext. More plaintext bits are encrypted in the same way as 128-bit data blocks. The AES decryption process executes the inverse of these functions in reversed order, using the *same cipher key*. The complete specification of the AES is in the FIPS-197 document [Nat01]. The provided information in this section should suffice most discussions covered in the survey.

The SubBytes function has the property that small changes in its input produce large differences in the output, and vise versa. Therefore, most power analysis attacks target the S-Box input or output because wrong key guesses lead to clearly distinguishable S-Box outputs compared to the actual values. In contrast, cryptographically weak functions such as the AddRoundKey are more resistant to power analysis because similar key guesses result in small variations in these functions.

A.2 The Cipher Implementations

The AES cipher can be efficiently implemented in either software or hardware. Many smart cards have software implementations using 8051, PIC or ARM family microcontrollers [RE10]. A simple microprocessor with 8-bit data paths does not create any significant limitations for running the AES algorithm. Some modern smart cards could also use 32-bit microprocessors for advanced features such as running Java programs. For ultra low-power, performance, and some security advantages, hardware implementations of cipher modules are also found available. Instead of stored lookup tables, the S-Box can also be efficiency implemented in hardware using finite field arithmetic [WOL02]. There is usually more parallelism in the hardware implementation than in software. For example, up to 16 S-Boxes can be executed simultaneously, while an 8-bit microcontroller can only process one byte after another. Both software and hardware ciphers often lack countermeasures for power analysis attacks unless specially designed for such purposes.

The cipher was designed to resist algorithmic breaking attempts: brute-forcing 2^{128} guesses is computationally prohibitive. However despite its software or hardware implementations, it is obvious that its internal computations often operate on smaller data widths than 128 bits, especially the S-Box function that only works on 8-bit chunks independently. It should be emphasized that these intermediate values are exactly on target in power analysis attacks, and their short length in bits greatly reduces the effort of exhaustive testing.