Review of gate-level differential power analysis and fault analysis countermeasurees

Hamad Marzouqi, Mahmoud Al-Qutayri, Khaled Salah

Electrical and Computer Engineering Department, Khalifa University of Science, Technology and Research, Abu Dhabi, United Arab Emirates
E-mail: khaled salah@kustar.ac.ae

Abstract: Hardware implementation of modern crypto devices paves the way for a special type of cryptanalysis, which is known as side channel analysis (SCA) attacks. These attacks are designed to extract critical information from the physical leakage of the digital circuitry such as the power consumption and electromagnetic emissions. Differential power analysis (DPA) attacks are considered the most efficient form of SCA attacks that require special types of countermeasures. Another form of attacks, known as fault analysis (FA), is based on forcing the circuit to produce faulty results in order to extract useful information about the secret. Several countermeasures have been proposed in the literature to address and mitigate SCA attacks at different levels of abstraction. They include algorithmic, gate and transistor-level countermeasures. Leakage originates at every level according to the implemented crypto system and attack methodology. Countermeasures at gate level and transistor level are more generic than those at the algorithmic level as they tend to be specialised for certain implementations. Complication of the design process increases down to the basic abstraction layers, however, gate-level countermeasures provide the balance of generality and design complication. The major state-of-the-art gate-level countermeasures against DPA and FA attacks are reviewed here.

1 Introduction

The computation requirement for carrying out the operations within a cryptographic algorithm necessitates the need for hardware accelerators. The strength of the symmetric cryptographic algorithms depends highly on the repetition of round operations, whereas asymmetric cryptography is built on large integer arithmetic. Hardware implementations of cryptographic algorithms may provide information leakage and backdoors using cryptanalysis techniques that are not based on the mathematical structure of the algorithm. In general, a provable secure algorithm can be attacked easily if it is poorly implemented, either as a soft code or as a hardware module. There are several types of implementation attacks, as shown in Fig. 1, which can be categorised into either interrupting the crypto-operations or just observing them. Some hardware attack techniques may require invasive measures that usually involve damaging several units to carry out a successful attack. As an observation attack, differential power analysis (DPA) is considered the most efficient form of attack in terms of setup simplicity and cost. On the other hand, fault analysis (FA) attack may require some precision and sophisticated setup in order to achieve the expected results.

Since the publication of the inspiring work on DPA attacks by Kocher et al. [1], several articles have been proposed in the literature to attack cryptographic applications in embedded systems using DPA [2]. DPA attacks are based on the concept that a cryptographic system running on chips (smart card, ASIC or field programmable gate array (FPGA)) may leak secret information through the power consumption pattern of the chip. The idea is that the amount of current withdrawn from the power source of the chip has direct correlation with the processed data (plain-texts and secrets). Several power traces are collected for many different inputs, then these power traces are statistically analysed based on a certain power model as for instance, Hamming weight or Hamming distance. The correct partial key can be determined by spotting current peaks from differentiating the power traces based on the power model.

Different countermeasures are proposed at different levels of abstraction as shown in Fig. 2. Countermeasures can be applied at the algorithm level, gate level and transistor level. Countermeasures at the algorithmic level are based on providing some redundant operations or random data to be processed with the original data. These types of countermeasures are called masking and they are based on randomising the power consumption in order to eliminate or reduce possible correlation between the power traces and the processed data. Masking techniques applied to advanced encryption standard (AES) [3] and data encryption standard (DES) [4] are proposed in [5, 6]. Masking with randomised look-up tables (LUT) is proposed in [7]; their technique is based on refreshing tables as a pre-computation step. Algorithmic-level countermeasures can be applied to the source code of the cryptographic algorithms for some implementations involving embedded processors like smart cards.
The second type of DPA countermeasures takes place at the gate level of the digital circuitry. Logic styles that provide constant power consumption regardless of the processed data are built on standard logic gates such as AND, OR and XOR. This type of countermeasure is known as a hiding technique, and it utilises the concept of dual-rail with pre-charge logic (DPL). On the other hand, masking countermeasures can be applied at this level by performing the logical operations of the original inputs masked by some random data. Countermeasures at gate level can be easily accommodated into standard-cell design flow within FPGA or ASIC. The drawback of gate-level counter measures is the area and performance overhead that is introduced by the extra redundant logic. Notice that gate-level DPA countermeasures are not sufficient for asymmetric cryptography, where algorithmic-level leakage is highly noticeable.

The third type of DPA countermeasures is based on building customised cells from transistor networks. These transistor-level countermeasures are based on either masking or hiding methods. The design target of such countermeasures is to build new logic gates that perform the standard logical function based on a different transistor layout. The new transistor layout should provide either constant power consumption regardless of the processed data or masking at transistor level. The primary drawback of transistor-level countermeasures is the complexity of incorporating new designs on the standard cell-based design flow. On the upside, such designs are lighter and provide better performance in terms of speed and power when compared to gate-level countermeasures. The first logic style that was introduced to defend against DPA attacks is the sense amplifier-based logic (SABL) [8]. The concept of a single transition per clock cycle through a pre-charging logic with dual rail is first introduced in the SABL logic style. Different logic styles that are based on the same concept are proposed later in the literature such as look up table-based LBDL [9], complementary metal-oxide semiconductor (CMOS) LUT [10] and SecLib [11]. Another design that is based on the complementary pass-transistor logic (CPL) rather than CMOS is presented as SDMLp in [12]. The inherited differential property of the CPL logic is utilised to provide a logic style with reduced area and power consumption at the expense of reduced performance.

Owing to some physical characteristics of the CMOS-based digital circuits, intended faulty operations are used to attack cryptographic systems built using such technology. Fault attacks are known as the procedure of making the cryptographic system deviate from its normal operations to produce faulty (unexpected) output in order to be used in the analysis to reveal the secret. There are several ways to inject faults in a digital circuit. A comprehensive list of techniques is listed in [13]. Faults are injected by either (i) manipulating the internal signals such as the clock of the circuit, or the supply voltage, or (ii) by physically affecting the electrons operation by using white light, laser and X-rays.

Fault attacks are considered a threat to hard-wired cryptographic systems and some measures need to be adopted to prevent such attacks [14, 15]. These measures are categorised into three different levels as shown in Fig. 3: (i) fault prevention, (ii) fault detection and (iii) fault resilience. Fault prevention is implemented within the chip by the chip manufacturer or on the printed circuit board as intrusion detection techniques as photo detectors and clock (supply voltage) watchdogs. When an intrusion is detected, the circuit may respond by zeroing out the secret, stopping the operations or outputting non-useful results. On the other hand, fault detection is widely used due to the simplicity and ease of implementation within the design of the digital circuit. Basically, fault detection depends on redundancy at different levels, including information, time and resources. The third level of fault countermeasures is the fault resilience where the fault output is allowed to be delivered with the assurance that this faulty result does not provide any useful information to the attacker. Different techniques and strategies for fault resilience are described in detail in [16].

Timing redundancy fault-detection technique is based on generating the result twice or more using the same block and comparing the multiple outputs before delivering the final output. A timing redundancy is provided, since the output requires extra redundant time to be delivered. On the other hand, space (or resources) redundancy is based on having multiple blocks implemented within the same
platform to generate multiple outputs for comparison before delivering the correct output. A space redundancy is provided by the added extra hardware for multiple output generation. As for information redundancy, the idea is based on parity codes where redundant parity codes [17] are used to detect and/or correct faults [18]. Although, parity codes are known for their inability to provide a 100% detection capability [19], they are considered the least costly techniques in terms of space and time when compared to the space and timing redundancy techniques.

Another way to categorise fault attacks countermeasures is at the implementation level. Countermeasures reported in the literature are either implemented at the algorithmic level or at the gate level of the digital circuit. Examples of countermeasures that are implemented at the algorithmic level are presented in [20, 21] for asymmetric cryptography, and [22, 23] for symmetric cryptography. Moreover, there are several fault attacks countermeasures at the gate level as discussed in [24, 25], where the countermeasures are considered sufficiently generic and can be used on any cryptographic system. Some countermeasures are specific to a certain cryptographic system and are implemented at the architecture level as those proposed for the AES symmetric algorithm in [26].

This paper reviews gate-level countermeasures against DPA and fault attacks. Section 2 provides background on DPA attacks and different leakage sources on CMOS circuits. Section 3 presents the basic concept of masking and hiding techniques at gate level with overview of several logic styles to thwart DPA attacks. Section 4 discusses and summarises the most popular techniques used to mitigate FA attacks. Section 5 discusses the option of combining both countermeasures in one unified scheme. Section 6 concludes the paper.

2 Background

To understand the concept behind different countermeasures for side channel attacks on crypto systems, we give a brief background on the power consumption of CMOS circuit along with some related concepts as those of early evaluation (EE), glitches in power traces and coupling capacitance.

2.1 Power consumption in CMOS digital circuits

The power consumption in CMOS logic consists of three sources [27] as stated in (1)

\[
\text{Total Power} = p_{th}(C_{l} V_{dd} F_{clk}) + I_{SC} V_{dd} + I_{Leakage} V_{dd}
\]

(1)

The first term in (1) is the most dominant. It consists of the switching components such as the capacitance, the clock frequency and the probability of the transition to occur. The second term is the power consumed when the current flow from the source to the ground through an active \( n \)-type MOS (NMOS) or \( p \)-type MOS (PMOS) transistors. The third term is determined by the leakage current that is directly related to the fabrication process. Therefore, the switching factor is the major term that should be taken into consideration since it contributes highly to the DPA traces. However, owing to advances in manufacturing technologies where the transistors continue to shrink in size, making the leakage power (static power) contributes also to the DPA traces. Most of the assumptions made in the literature eliminate this fact, hence such assumptions need to be reconsidered to reflect the state-of-the-art implementations [28, 29].

2.2 Glitches

In CMOS circuits, opposite switching activities in the inputs of the gates cause unnecessary activities known as glitches. Also, different arrival times of the inputs due to the propagation delays of the predecessor gates and the wires lengths cause such activities [30]. These glitches contribute highly to the DPA traces since they are data dependent. Therefore, DPL, place and route constraints and synchronisation methods are the key solutions to avoid, or at least minimise; such glitches in the logic styles will be discussed thoroughly in this paper.

2.3 Early evaluation

Different logic gates have different propagation delays, and the wires connecting these gates have their own propagation delay proportional to their lengths. Such characteristics cause different arrival times of the inputs to the gates; hence, some unnecessary evaluations are performed by the gate before desired evaluation. This phenomena is known as early evaluation (EE) [31], and some designs of DPA-resistant logics are based on addressing such activities.

2.4 Coupling capacitance

In DPL logic style, it is important to have equivalent routing for the true and the complementary paths in order to avoid the effect of coupling capacitance as illustrated in Fig. 4. The loading capacitance has three components, as stated in [32], which consist of (i) the intrinsic output capacitance, (ii) the interconnect capacitance and (iii) the intrinsic input capacitance of the load. The intrinsic input and output capacitances are controlled by the DPL logic, whereas the interconnect capacitance should be controlled by the place and route process. It is worth noting that the place and route process is much more difficult to control within an FPGA design than in ASIC implementation. Hence, the loading capacitance requires further back-end FPGA methods to ensure equivalent routing of the true and the complementary paths of the DPL logic.

2.5 Concept of DPA attack

DPA is considered a simple, cost-effective and efficient attack [30]. Even with cryptographic core with countermeasures against such attack, the DPA can still perform a successful attack by carrying out additional traces. The typical setup involves only a PC, an oscilloscope and the cryptographic device hooked to a power supply. An acquisition tool running on the host PC acquiring power traces from the

![Fig. 4 Loading capacitance for unbalanced DPL routing](image-url)
oscilloscope for different input data is needed as well. Fig. 5 depicts the basic procedure of the DPA attack.

The basic procedure of a DPA attack consists of four main steps. The first step involves collecting as much as possible power traces from the cryptographic core by applying different input data. The efficiency of a DPA attack is measured by the number of traces that are used to finally extract a meaningful secret. The second step is based on targeting an intermediate partial key related function within the cryptographic algorithm and calculating the outcome of that function for different inputs (that are used in the first step) and all possible partial keys that are used by the function.

The third step is based on finding a power model that best suits the circuit that runs the cryptographic algorithm. Mostly, the Hamming distance is used as a power model for CMOS circuits, others may use the hamming weight model. This hypothetical power model is applied to the intermediate results (step 2 outcome) to calculate the power value for every input data \( D \) used and all possible partial key \( K \), hence, a table of size \( D \times K \) is generated. For sure, one column \( k_i \) relates to the exact operation performed by the cryptographic core itself. Using the measured power from the core and the table created in step 3, the fourth step is based on statistical analysis of those gathered data to find the best possible key. A successful statistical analysis is based on increased number of power traces.

The attraction of the DPA attack is that instead of performing a brute force attack on all possible key value, which has the cost of \( 2^n \) with \( n > 80 \), the range is reduced to \( 2^8 \) or a maximum of \( 2^{16} \) as per the function that directly operates on part of the key. An example of such function is the substitution box in symmetric algorithms, where size ranges from 4-bits in lightweight cryptography \([33]\) up to 32-bits in some algorithms like MARS. The same attack is performed on the other partial key bits using the same gathered power traces. A successful DPA attack is based on multiple factors: (i) Careful power traces collection by reducing as much as possible the noise from other circuit components. (ii) Increased number of gathered power traces. (iii) Careful choice of the intermediate function that is related to partial key. Finally, (iv) well-established power model.

### 3 Gate-level DPA countermeasures

The general concept behind DPA countermeasures at any abstraction level is to minimise the correlation between the power consumption of the circuit and the processed data. It is almost not possible to zero out this dependency. However, it is sufficient to reduce it to the minimum level that will complicate the attack. Manipulating the dependency between the power consumption and the processed data is accomplished in time and amplitude domains. Time-domain countermeasures try to shuffle the operations sequence or desynchronising them to harden the necessary alignment procedure for the DPA attack as in \([34, 35]\), whereas, the amplitude domain countermeasures focus on either randomising the power consumption, or making it consistent regardless of the processed data. Amplitude domain countermeasures are based on DPL or masked logic. Fig. 6 shows the list of different DPA
countermeasures at gate level that are to be discussed within this section.

3.1 Dual-rail with pre-charge logic (DPL)

Two concepts define the operation of the DPL logic. First, the signals in the DPL logic style are represented by their true and false values, and thus named dual-rail. Second, these signals exhibit dynamic behaviour within one clock cycle operation. At the lower edge of the clock cycle, the signals are forced to absorb their null values (pre-charged), whereas, at the higher edge, the signals gain their true values (Evaluation).

3.1.1 Wave dynamic differential logic (WDDL): The design of the WDDL is based on the design of the SABL. The SABL [36] is a transistor-level logic style that is designed based on two principles: single switching event per cycle that is independent of the input signals and consistent total capacitance charging and discharging. Therefore, the designers of the SABL proposed a new scheme at the gate level to be easily embedded in the standard design process. The proposed WDDL scheme [37] is built totally from standard-cell library that can be adapted to the design flow for an ASIC or FPGA.

WDDL is differential as every signal in the architecture is presented by its true and false values. Therefore, using the DeMorgan Law, the differential output is presented by the true value and its complement is presented by the false value which will ensure the single transition event. The pre-charge and the evaluation phases are occurring within one clock cycle which will make the logic dynamic as they change their states concurrently. The inputs are pre-charged by applying the inputs with the inverted clock signal to a NOR gate. Hence, the output will be zero whenever the clock is low (pre-charge), and the output of the pre-charge logic will be the true value of the input whenever the clock is high (evaluation). For the registers, the same pre-charge logic is applied to the outputs of the registers. However, a master–slave (two cascaded registers) model of the registers can be applied to form a pre-charge circuit. This model will cost double the clock frequency in order to obtain the same data rate as the first model.

Fig. 7 shows the schematic of the AND gate WDDL logic style with the pre-charge logic for the input and the registered outputs. Note that the pre-charge is applied to the system inputs, not to the input of each logic gate individually. Fig. 7 shows a logic depth of one and the pre-charge can be applied to any logic depth. For instance, the pre-charged input ‘A’ will propagate through the cascaded logic depth making the pre-charging event occur for every logic on the data path. To give an example, consider an input ‘A’ is pre-charged to zero, making the true and the false outputs of the pre-charge logic equal to zero. The pre-charged input will cause the outputs of the first logic equal to zero. These outputs are connected as the inputs to the second logic depth making them pre-charge to zero, hence producing the zero output. The same analysis can be applied to the next logic depth and so on.

Having the single transition event will ensure the total capacitance charging/discharging consistently for every clock cycle, and thereby, the correlation will be reduced between the processed inputs and the power traces used on the DPA attack. The main drawback of the WDDL scheme is the area and the performance overhead introduced by the additional gates and the differential routes. It is worth noting that the area is almost three times that of the single ended logic. Another drawback is the restriction to the usage of limited gates as the AND and OR logics. Notice that in order to preserve the pre-charge wave generation, the inverter should not be used within the data path which will stop the propagation wave. Therefore, to invert a signal within the data path, the differential output of the WDDL gate can be swapped so that the true output will be connected to the false input of the next logic and vice versa.

As for FPGA implementations, different variants of WDDL are proposed to eliminate the effect of the unbalanced differential routing and the EE. One is called double WDDL (DWDDL) [38], which costs almost four times the resources of the basic module. Another approach isolates the true and the complementary paths into separate regions within the FPGA fabrics. This method is called isolated WDDL (iWDDL) [39] and extra registers are required for the negative signals to avoid stopping the pre-charge wave. Extra clock cycles will be the cost. Similar to the iWDDL approach there is another called divided backend WDDL (DBWDDL) [40] that utilises XOR gates as inverters instead of registers. To ensure balanced routing, one may use dual-output programmable blocks within some FPGA as discussed in [41].
The results of an ASIC implementation of AES algorithm protected with balanced WDDL can be found in [32]. The area overhead is increased from 79 K of gate-equivalent up to 245 K. Most of the key is recovered using on average 250 000 traces, whereas the full key recovery of the unprotected AES required only 2200 traces. Notice that the maximum frequency is also affected, the reduction is almost by four times.

3.1.2 Dual-rail logic – dual spacer: The dual spacer DRL countermeasure is based on having the pre-charge value alternate between the all-zeros signal and the all-ones signals [42]. The all-zeros and the all-ones are called spacers since they carry no information. The principle behind the dual spacer DRL design is the switching factor of the OR gate and the AND gate differ intrinsically. When the gate switches between the pre-charge and the evaluation phase, only one gate will switch its state from null to valid value state, in case of single spacer (00). Therefore having a different switching factor between the OR and the AND gates can leak some information regarding the processed data.

The protocol of the dual spacer scheme works by having the DRL gate return to the all-zeros and the all-ones spacers every two clock cycles. Hence, the gate will be initially at the all-zeros ‘00’ spacer, then the evaluation phase will start and the gate will produce its true value, either (01) or (10). On the second pre-charge phase, the gate will go to the all-ones (11) state, then the second evaluation phase will produce either (01) or (10). Therefore, the dual spacer will ensure no correlation between the different switching factors of the gates and the processed data since the valid data (01) or (10) can be produced by switching the AND or the OR gates, depending on the state of the previous spacer.

The drawback of the dual spacer scheme is that it requires extra gates to control the value of the spacer and to alternate between the two spacers. This area overhead will degrade the performance since the conversion from single spacer to dual spacer is done in the critical path. The converter from single-to-alternating spacer is basically controlled by a toggle that will decide to which spacer it will alternate according to the previous states. Hence, two latches are used on the toggle to reserve the previous states. Another toggle is needed to control the alternation between spacers on the output of the registers. This toggle operates according to the system clock and a single toggle can be used for all registers on the system.Fig. 8 shows the converter proposed in [42]. The toggle will control the alternating process depending on the value of the previous states using latches. The same converter is used for the output of the registers with a global toggle controlled by the system clock. It should be noted that only simulation results are provided in [42] without any physical implementation in either ASIC or FPGA and no DPA attack were performed on the protected circuit.

3.1.3 Reduced complementary dynamic and differential logic (RCDDL): The RCDDL logic style depends on reusing the logic of the true data path to generate the complementary output [43]. It provides improved sizing results on logics that are composed of sum of products terms rather than single-depth logic gates as AND and OR gates. The RCDDL style does not limit the usage of negative logic as in other DPL as these logics will stop the pre-charge propagation wave.

The RCDDL logic consists of two data paths. First, the original data path generates the true value and is composed of the product terms segment and summation segment. The
3.1.4 Secure triple track logic (STTL): The signals in the STTL [44] style are encoded by three wires rather than two wires as in the dual-rail logics. The third wire is used to determine the validity of the signal and does not contribute to the information content of the signal. Hence, the valid signal will control the evaluation process on the output of the valid signal generation logic that computes the output according to the valid signals. The delay of the output valid signal is ensured by cascading five LUTs on the output of the valid signal generation logics. The logic that computes the output according to the valid signals is implemented via C-elements each occupying one LUT. The total LUT occupancy is 11 LUTs, out of which five used for the delay and six used for the logic. An optimised version with six LUTs constructing the STTL is implemented also in [44]. It is to be noted that the extra LUT-NAND gate driving the signal Z1 is used to equate the logical depth with the other signal Z0. The optimised version can be implemented by generating the Z0 signal with one 3-LUT, implementing the generalised C-element with eliminating the need for the extra NAND gate driving the Z1 signal. Also, this will yield to reduce the buffer depth to three LUTs for the ZV signal.

3.1.5 Balanced cell-based differential logic (BCDL): The (BCDL) [46] is another type of dual-rail logic with pre-charge that attempts to avoid the early propagation problem by using synchronisation cells with global pre-charge signal. This global pre-charge signal is generated by a phase locked loop source within the FPGA with double the speed of the system clock to ensure distribution to all BCDL cells and faster arrival than other input signals. The global pre-charge signal avoids using the pre-charging logics for every input signals since it will force the BCDL logic to fire a zero output at the pre-charge phase on every cell. Unlike the synchronisation process of the STTL design that is based on the extra validity signals of the inputs, the BCDL synchronisation is performed at all differential signals of the inputs that can be easily suited for FPGA implementations using the general multi-input LUT.
general, the synchronisation should assure that the evaluation phase does not start until all inputs are valid. Also, the pre-charge will not start until all inputs become NULL.

As shown in Fig. 11, the synchronisation is performed by first XOR-ing the differential inputs to check for the states of the inputs, then these signals are applied to an AND gate along with the global pre-charge signal. The resultant pre-charge/state signal is used to force the outputs of the true and the complementary logics to pre-charge or evaluate accordingly. Since the pre-charge signal is faster than any other input which undergoes different logic depths, no EE issues arise. An important requirement for mapping the pre-charge/state signal into the LUT is to connect this signal to the MSB of the LUT to avoid any unnecessary activities in the LUT that may cause glitches.

The BCDL can be considered as very promising DPA-resistant logic style since it is considered compact in terms of performance and size in comparison with the other DPL logic styles. Also, the synchronisation process reduces the effect of EE with minimum resources, especially in FPGA. In addition to this, the synchronisation is done with completely non-hysteresis logic (no memory) as opposed to STTL which uses the C-element. Furthermore, BCDL is considered robust against simple fault attacks as analysed in [46]. Although, the global pre-charge signal can raise some issues regarding the routing complexity to all logical cells, the BCDL remains a very attractive logic style which requires further analysis and robustness evaluation.

3.1.6 WDDL without EE: An implementation at the design level of the WDDL model with no EE property has been proposed in [47]. The basic idea of the WDDL w/o EE methodology is to map both the true and false signals of the inputs to the direct and complementary logic. In this case, the gate does not change its output if the inputs are in the transitional states. The gate outputs a valid signal if all inputs are in the steady state and valid. For instance, if one of the inputs provides \((0, 0)\) (pre-charge phase), and the other input provides \((0, 1)\) (transitional phase), then the output will stay at the \((0, 0)\) state. This implementation provides protection intrinsically against the EE problem and at the expense of LUTs occupancy.

3.1.7 Homogeneous dual-rail logic (HDRL): The HDRL style [48] is designed based on the observations of the ground voltage (VSS) current instead of the usual supply voltage (VDD) current of the circuit. The hypothesis used is that the VSS current drawn by a cell is indistinguishable for different inputs. Hence, the same cells are used for the complementary and the true data paths, where one is fed by the true values of the signal and the other is fed by the complemented signal. As stated, the energy and design complexity overhead is much improved over the WDDL logic style. Also, clock speed is improved since the pre-charging phase is not necessary. Fig. 12 shows the design of the HDRL AND gate where two AND cells are used for both data paths.

All CMOS gates suffer from glitches by nature [49], and having a CMOS logic style without pre-charge characteristics is a major flaw for a DPA-resistant scheme. Another concern of such logic style is the fact that all the assumptions are made based on observing the VSS current without taking into consideration the source current that is drawn by the circuit. Results shown in [48] are based on simulation tools; further practical experiments on silicon is necessary to back up the assertions made regarding the HDRL logic style.

3.1.8 Pre-charge absorbed DPL: The (PA-DPL) [50] is another DPL logic that attempts to eliminate the effect of the EE. Unlike the basic concept of the wave pre-charge propagation, the pre-charge signal should be connected to all LUTs that encode a logic function to be protected. Using the digital clock management within the FPGA, a signal is generated with double the frequency of the system clock and with slight timing ahead. This signal is inverted
and logically AND-ed with the pre-charge signal, which is usually generated by the system clock. The output of the AND gate is connected to the LUT as a pre-charge absorbed input.

The pre-charge absorption process requires another LUT to perform the AND operation, which in turn can be avoided by utilising two inputs on the LUT and encoding the process within that LUT itself. It is highly recommended to ensure equivalent routing for the complementary paths to avoid unbalanced capacitance. In [50], the method of DBWDDL is illustrated to perform this task. Unlike the WDDL, two networks need to connect all LUTs that require protection, the pre-charge and the extra double frequency signal. This requirement is considered a drawback along with the constraint that needs to be assured by doubled-frequency clock and timing-ahead synchronisation signal. This constraint reduces the maximum allowable frequency for a system to be implemented within the FPGA by 50%.

3.2 Masking logic

Instead of operating over the true values of the input signals, masking logics force the operation to be performed over masked inputs by randomly generated data. It has been observed that purely masked logic does not provide sufficient protection against DPA attacks [51]. Therefore, several logic styles are proposed in the literature that utilise the advantage of the DPL logic with the masked logic as discussed in this subsection.

3.2.1 Masked dual-rail pre-charge logic (MDPL): The MDPL [51] logic style involves randomising the power traces. The masking method and the principle of the DPL are combined into one logic style. The DPL method is used to avoid glitches that is introduced in other masking-based logic [49]. Similarly, every signal is presented by the true and complementary value. The mask is generated by a TRNG and PRNG blocks and is also presented by its true and complementary value.

The authors in [51] claim that the implementation of the MDPL does not require any place and route constraint as in the other DPL logic styles, since it is mostly built of majority gates that are considered monotonic positive gates. A monotonic positive gate provides an equivalent transition at the output as per the transitions on the input. However, balanced load capacitance is necessary to increase the resistance against DPA attacks, which is done through controlled place-and-route constraints [38, 40].

The differential outputs of the MDPL gate is produced from two MAJ gates. Two gates are used to produce the differential output for the AND gate. The true value $Q_0 = \text{MAJ}(a_m, b_m, m)$ is produced by one MAJ gate and the complemented output $Q = \text{MAJ}(a_m, b_m, m)$ is produced by the other MAJ gate. Notice that the signals $a_m$ and $b_m$ are the masked signals of the original inputs. Fig. 13 shows the schematic of the MDPL AND gate. Other logic gates can be implemented from the MDPL-AND gate.

For the completeness of the DPL style, a pre-charging phase is applied to every MDPL gate. The pre-charge wave is generated at the inputs and at the registers outputs and will propagate through the logic since the MAJ gates are positive logics that will not stop the pre-charge wave. The strength of the MDPL logic randomness depends on the PRNG block that is used to generate different masks at every clock cycle. Therefore, area and design time overhead is introduced by the necessity of the PRNG implementation.

3.2.2 Improved MDPL: An evaluation of the MDPL implementation of a cryptographic core with AES implementation is done by performing a DPA attack in comparison with the WDDL balanced wires implementation [52]. A severe leakage on the MDPL implementation due to the EE problem can be observed by the attack results. The DPA attack on the AES core with MDPL implementation only traces 471 in comparison with the balanced WDDL (43 201 traces) [52]. Notice that power traces are measured when the micro-controller performs 1-byte move operation. It has been shown in [52] that some peaks occur at the beginning of the evaluation phase due to the difference in time arrival of the input signals of the MDPL cell.

An improvement on the MDPL unit has been proposed by adding an evaluation-precharge detection unit (EPDU) before the input of the original MDPL cell. The EPDU works by producing a zero if all inputs that are at a differential state, that is, ready to be evaluated. Fig. 14 shows the schematic of the iMDPL cell with the EPDU unit. The EPDU consists of two parts. First is the combinational logic with three OR gates and one NAND gate to detect the states of the MDPL inputs. The second part consists of three latches that will force the MAJ gate to produce zero whenever the input signals are in the pre-charge phase (all-zeros). Therefore the EPDU unit will ensure that the evaluation phase will not start until all the input signals are in evaluation-ready state. Note that an additional area overhead gets introduced over the original MDPL to solve the problem of EE. Another improvement of MDPL logic style is presented in [53], where an FPGA implementation of masked memory with DPL logic style is proposed. An all-in-one style that packs all the necessary functionality of the MDPL style in one LUT to eliminate the need for certain routing procedures.

A recent attack on the iMDPL scheme that exploits the leakage from the mask tree is provided in [54]. The attack is performed on an AES-128 chip that is protected by an iMDPL logic style. A pre-processing step of the power traces as an integral computation of the power consumption values in a clock cycle is performed by specifying a certain window of a specific size. Improved results where the key is recovered by only 200 000 traces is based on the fact that the propagation of a single mask of zero value differs than the propagation of the mask with a value of one. This is shown on the histogram of the power values when the mask is zero and the mask is one.

3.2.3 Dual-rail random switching logic (DRSL): The DRSL is another masked logic that is built on the random switching logic (RSL) proposed in [55]. The design of the
RSL gate involves building a standalone logical gate that does need complementary control. The RSL gate performs the masked logical operation having the mask bit and the masked signals as inputs. However, an enable signal that is used to allow or disallow the operation of the RSL gate is needed. The reason for the enable signal is that, if the enable signal is high, then the RSL will execute the operation; otherwise, the output will be zero. There is an extreme condition that needs to be satisfied as the enable signal needs to be set to ‘1’ after all other inputs are settled. The RSL NAND gate can be realised using (2), and other logical operations can be built out of this gate.

\[ z = \text{enable} + xy + x'r + yr \]  

(2)

The generation of the enable signal requires a global enable signal to be passed to all RSL gates in the architecture, which may require amplifications to overcome the fanout problem. A transistor layout of the RSL NAND gate is shown in [55]. The gate can be realised according to (2) and can be implemented in a FPGA LUT. Fig. 15 shows the schematic of the RSL NAND gate realised by standard logic.

An attack on the RSL logic is proposed in [56]. Since the RSL methodology is based on equalising the transition probability by the random mask, the attack is based on attempting to reveal the mask to disturb the equalisation and then extract the secret. It has been concluded in [56] that a single mask bit is not enough to protect against DPA attacks on a masked logic. A DRSL has been presented in [57]. It is based on the principle of combining dual-rail logic with pre-charge and the basic RSL gates. A local pre-charge generation based on the states of the masked inputs is used to synchronise the input arrivals and generate the enable signal for the RSL gate. The local pre-charge logic works by first generating the pre-charge signal (Enable = 0) when one of the inputs is in the pre-charge phase, and then disabling the pre-charge when all inputs are in the evaluation phase. A major shortcoming of the DRSL is the requirement of both local and global pre-charge signals where overhead and synchronisation issues are introduced. An improvement of the DRSL that is built from positive logics is proposed in [58].

### 3.2.4 Pre-charge masked Reed–Muller logic (PMRML):

The PMRML [59] is a masked logic style that takes advantage of the fixed polarity Reed–Muller (FPRM) form of the Boolean functions. The FPRM form states that a Boolean function in general can be composed of the XOR sum of the ANDs in which every variable has either negative or positive polarity. The analysis of the design of the PMRML presented in [59] is based on having logic functions free of glitches and dissipation timing skews (DTS), a definition that is introduced in [59]. The authors claim that the Masked-AND gate is not DTS-free, whereas the masked-XOR gate is DTS-free, since it has equal occurrence of the zeros and ones. Therefore, using the FPRM form, the AND part is replaced by dual-rail selection 4 × 1 multiplexer (MUX-DS). The XOR part is replaced by a Masked-XOR gate.

\[ z = \text{AND} + \text{XOR} \]  

(3)
A correction mask generator (CMG) is used to recover the plain data, where the initial masks are generated by a RNG. The pre-charge process is performed on the signals that form the dual-rail selection of the multiplexer. These are formed by one of the inputs (differential and pre-charged) and the updated mask of this input. In a cascaded PMRML cells, separate pre-charge signals need to be generated to meet timing constraints and to ensure the correct starting of the evaluation and pre-charge phases. This pre-charge timing constraint would require separate control and, consequently, firing procedure. The control is also affected by the architecture of the cryptographic core implementation and the logical depth of the architecture. These control constraints are considered major drawbacks of the PMRML logic style and very difficult for implementation in either FPGA or ASIC implementation.

4 Gate-level FA countermeasures

Hardware systems are vulnerable to faults that are either malicious or accidental in which deviation from the normal operating parameters of the system occurs. It has been shown that secrets can be revealed from cryptographic hardware systems by carefully inducing faults that result in faulty output [14, 15]. An extensive discussion of fault attacks techniques and countermeasures is given in [60]. Single or several faulty outputs can be analysed to deduce some information on the secret. Countermeasures are essentials in such systems. As shown in Fig. 16, these countermeasures can be classified into three types: (i) fault prevention, (ii) fault detection and (iii) fault resilience. Prevention is implemented at the physical level or at the printed circuit board (PCB) level by chip coating or sensors and watchdogs. Detection and resiliency can be implemented at gate, architecture and algorithmic level. Detection is mainly based on redundancy at three levels [61], as depicted in Fig. 17: (i) hardware (space), (ii) information through parity codes and (iii) timing. Resiliency is the concept of revealing non-useful information or preventing the fault propagation. It is mainly based on DPL logic and homomorphic encryption at the algorithmic level.

4.1 Fault detection

Fault detection is the most anticipated method to resist fault attacks in cryptographic hardware systems [62]. Hardware (space) redundancy is based on duplicating the cryptographic hardware module in order to compute the two outputs in parallel. An example of a fault-tolerant ECC system using a parallel computation of the scalar multiplication is presented in [63]. A successful attack requires a highly timed and localised perturbation on both modules. The second type of redundancy is timing. Sequential computations are performed using the same module to generate the two outputs to be compared. The second operation performs the inverse operation that might be available in the system. As an example, a timing redundancy used in AES fault-tolerant implementation that
utilises the inverse blocks of the decryption module is presented in [64], and round and operation level are presented in [26]. The same operation can be repeated twice as implemented in ECC [63]. The homomorphic property of the RSA algorithm is exploited to introduce a concurrent error detection scheme [65].

The detection capability and the strength of the information redundancy scheme is based on the type of the code used. In general, there are three types of codes that are used in a cryptographic system to detect faults that include: (i) parity, (ii) linear and (iii) robust codes. Codes are used to detect faults in registers and logical gates. In registers, the input and output bits are applied to a specific code with a comparator that is used to check for a difference that may occur within one clock cycle. For the combinatorial block, a prediction block that copies the main operation is used to generate assumingly similar output. The two outputs are applied to the parity codes in order to compare for any differences in the outputs. A comparator is also used to detect whether a fault has been induced in one of the two data paths: the main combinatorial block or the prediction block.

Different parity codes [19] have been proposed in the literature for asymmetric and symmetric cryptography. As an example of parity codes implementation in symmetric cryptography, a double parity code for the AES for the whole round has been proposed in [66]. In asymmetric cryptography, the multiplier is the main block that is protected through parity codes. Binary field Montgomery multiplier is applied to parity codes to provide a concurrent error detection in [67], whereas a multi-parity scheme is used in [68]. A low-density parity check is used in a polynomial bases multiplier with three-bit detectable errors and a Reed-Solomon correcting code has been proposed in [18].

Linear arithmetic codes provide a middle solution for performance when compared to parity and robust codes. In [69], a multi-linear code system has been implemented by randomly choosing one of the available linear codes to detect faults in asymmetric cryptography systems. A BCH code with multi-bit error detection and correction is implemented for binary fields Montgomery multiplier in [70]. An extended AN + B code for the linear blocks and redundant table lookup for the non-linear blocks of an AES implementation is provided in [71]. On the other hand, robust (non-linear) codes provide better performance with higher complexities. Examples of the AMD code has been presented in [72], and quadratic residue code for a general arithmetic unit has been presented in [73].

In the literature, different logic styles can be classified as gate-level fault-analysis countermeasures. We summarise these as follows:

4.1.1 Self-checking alternating logic (SAL): SAL involves providing a complemented output for a complemented input. A normal logic function is transformed into SAL logic by adding one extra input. If the extra added input is set to zero, then the function performs its regular operation; otherwise, a complemented output is provided. Accordingly, a timing redundancy scheme can be established by sequentially performing the regular and the complemented operations of the SAL logic to check whether the second output is a complemented version of the first output. Consider a function \( f \) with an input \( X \) of length \( m \), the SAL logic function \( f^* \) is presented as follows

\[
f^*(X, 0) = f(X)/\overline{X}(X, 1) = f(X)
\] (3)

The SAL logic is used in [74] to build an error-detection scheme for a dual-basis polynomial multiplier for ECC implementations. The multiplier is based on a systolic array architecture made of processing elements that perform bitwise operations. The processing elements have three inputs and one output, where an extra input is added to transform the regular processing elements into self-checking alternating processing elements. Consider a processing element with inputs \( a, b \) and \( S_{in} \), the output \( S_{out} = ab \oplus S_{in} \).

With an extra input \( t \), the output \( S_{out} \) is calculated as follows

\[
S_{out} = \overline{t}bS_{in} + bS_{in}\overline{a} + tS_{in}a + \overline{t}bS_{in}a + \overline{t}bS_{in}\overline{a}
\] (4)

Accordingly, the regular processing element schematic and the SAL version of the processing element are shown in Fig. 18. The regular output is calculated and stored in a register by setting the input \( t = 0 \). Then, all inputs are complemented along with \( t \) to calculate the complemented output. Finally, the complemented output is compared with

Fig. 18 Regular and SAL processing element for the dual-basis multiplier

\( a \) Regular processing element
\( b \) SAL processing element
the regular output in order to detect any transient or permanent faults at the cost of time and area overhead.

4.1.2 Asynchronous logic: Asynchronous logics have intrinsic resistance capability to faults. In [25], an asynchronous circuit design has been presented to take advantage of the fault resistivity of the asynchronous logics to resist fault attacks. The design is focused on quasi delay insensitive asynchronous circuits that are based on four phases self-synchronising protocol. Such circuits use the Muller gates (C-element), which evaluate only if all inputs are valid. Hence, an instantaneous fault in one of the inputs will not propagate if it gets triggered prior to the evaluation phase where other inputs are not valid yet.

The signals that are processed by the combinational part are registered into memory based on the synchronisation protocol. Hence, two parallel data paths are mutually synchronised by the acknowledgement and the validity signals. When a fault is induced in one of the data paths, it will most likely generate an invalid state of the protected data paths. The problem with asynchronous circuits is the increased design complexity that makes them unattractive.

4.1.3 Parity preserving logic: Parity preserving logic or reversible circuits [75] are a type of logic functions with an equal number of inputs and outputs. The main advantage of such logic is the parity preserving feature where the parity of the inputs is always equal to the parity of the outputs. Hence, parity checking can be performed directly on the inputs and outputs in order to detect faults that might happen. Fig. 19 illustrates the principle of parity preserving gates. The functionality of these gates are limited to certain operations; however, different functions can be built by combining different parity preserving logics.

In [76], redundant signed digit (RSD) adder is built from parity preserving gates as a fault-tolerant full adder for prime field ECC processor. In general, the RSD adder is propagation free and consists of two layers of full adders. In each layer, five Feynman Double (F2G) and three Fredkin (FRG) parity preserving gates are used with different layouts, to generate the actual data for computation and redundant bits for parity checking. In [77], 5 × 5 signed multiplier is built based on 12 F2G, 13 FRG and 8 modified new fault tolerant (MNFT) gates. Such logic gates are slightly expensive with an inherited parity checking capability. Note that the detection capability of these gates are limited since a simple parity checking scheme is utilised.

![Fig. 19 Different parity preserving logic gates](image)

*Fig. 19 Different parity preserving logic gates*

a) Fredkin gate  
*b) Feynman gate  
*c) Feynman double gate  
*d) Modified new fault tolerant gate*

4.2 Fault resilience

The concept of fault resiliency is presented in adequate details in [16] where a non-meaningful faulty result is provided to the attacker. It is very important to make sure that the faulty output cannot be used to perform FA and DFA attacks. DPL logic with no EE characteristic that is encoded within a LUT function is used as a fault injection resilience technique at gate level. Another technique is introduced in [16] at protocol and algorithmic levels that utilises a homomorphic encryption and decryption scheme. In DPL logic, each signal is encoded by two signals where logic ‘1’ is presented by ‘10’ and logic ‘0’ is presented by ‘01’. The signals ‘00’ and ‘11’ convey no information since they are used on the pre-charge phase.

In case of a one bit fault in the signal ‘10’ for instance, the signal is altered to ‘00’ which in turn does not provide any information based on the DPL pre-charge/evaluation concept. To make sure that this fault does not affect the consequent cycles, the DPL with no EE is used. The DPL with no EE makes sure that the gate does not evaluate or pre-charge unless a valid signal at the input is provided. A single case where a fault can propagate is where a dual fault on the same signal triggers a transition from a valid signal ‘10’ to another valid signal ‘01’.

5 Combined countermeasures

To increase the security of the cryptographic system that is implemented in hardware, it should be protected against known attacks such as DPA and FA. However, extra security means extra cost in terms of hardware resources and performance. Also, some countermeasures may introduce back doors for other attacks, as in the case of C-safe and M-safe attacks on simple power analysis (SPA) protected circuits [78]. Having a combined scheme that can provide somehow a unified protection against DPA and FA would be desirable. This section discusses the different schemes and methodologies that are proposed in the literature, especially at the gate level.

The basic principle of a combined countermeasure is to provide a fault detection scheme that can decrease the correlation between the power consumption traces and the processed data. It can also be done also the other way around where a DPA-resistant logic style may have a fault-resistance capability. Either way, the combined countermeasure should provide the necessary protection with optimised hardware and performance costs.

In the literature, there are few research articles that address this challenge where a unified scheme is able to reduce the effect of both DPA and FA attacks on cryptographic hardware. Most of the proposed work is focused on cryptographic algorithms implemented at the algorithm level. Examples of algorithmic level combined countermeasures for RSA and ECC simultaneously have been described in [79, 80].

5.1 Parity codes and DPA attacks

A study on the effect of different parity codes on the correlation between the power consumption traces and the processed data has been presented in [19]. In [19], it has been shown that different parity checking codes provide different sensitivity levels of the protected circuit against DPA attacks. Complementary parity codes has demonstrated the best resistance against DPA attacks in which the correlation between the power traces and the processed data is minimal because of its complementary path.
Table 1 WDDL AND gate w/o early evaluation: fault resilient implementation [47]

<table>
<thead>
<tr>
<th>At</th>
<th>Af</th>
<th>Bt</th>
<th>Bf</th>
<th>AND (true)</th>
<th>OR (false)</th>
<th>Input state</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>pre-charge</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>from pre-charge</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>from pre-charge faulty</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>from pre-charge evaluation</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>evaluation</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>from invalid</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>from pre-charge evaluation</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>evaluation</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>from invalid faulty</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>from invalid faulty</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>invalid or faulty</td>
</tr>
</tbody>
</table>

Fig. 20 Fault induced in a DPL logic during a pre-charging phase

5.2 DPL logic style and fault attacks

The DPL styles showed some immunity against fault attacks [81, 82]. The alternation between the pre-charge value (00) which is considered as a null value and the evaluation value (10) or (01) gives this special feature to such logic styles. The DPL logic exhibits two phases during one clock cycle. At the first half of the clock cycle, all DPL logic gates in the path are forced to be pre-charged to null value. At the second half of the clock cycle, the gates are evaluated to provide the corresponding signals. This phenomena is forced on all logics in the path through a dynamic wave of pre-charge and evaluation.

Two scenarios can be considered in the DPL logic as follows: The first scenario is where a fault is induced in the pre-charge phase in which a null signal is altered either to a valid signal or to another null signal (11). In this case, the faulty signal will not propagate to the next logic depth because it will be suppressed when the evaluation phase starts at the next clock edge rising (the register is not triggered and the fault is not stored), as shown in Fig. 20. In the second scenario, the fault is induced in the evaluation phase where a valid signal is altered to either a null signal ((11) or (00)), or to another valid signal. In this case, the faulty signal might be stored and propagated to the next clock cycles resulting in a successful attack. Notice that the window for a successful attack is reduced in the case of DPL logic.

Fault resiliency has been described in [47] where WDDL implementation without EE is proposed. The LUT encoding of the WDDL gate evaluates only valid signals. Hence, only a successful attack is performed when a valid signal is altered to another valid signal; that is, (10) to (01) or vice versa. Table 1 shows the implementation reported in [47] of the WDDL w/o EE with fault attacks countermeasures. The DPL w/o EE tend to prevent any evaluation activities on the logic when the inputs are in the transitional phase. Hence, it can be clearly seen in the LUT implementation that the outputs are either (00) or (11) when there are invalid signals on the input ports. We can see that there are some invalid inputs that may be presented due to a fault injection that produce invalid outputs, revealing no useful information to the attacker. Since BCDL is another DPL logic style with EE prevention characteristics, then it is considered also prone to fault attacks as shown in [46]. In general, the same analysis can be applied to any DPL logic style without EE characteristics.

6 Conclusions

This paper reviewed the existing gate-level DPA and FA countermeasures. It was shown that combining these types of countermeasures can minimise area and performance overhead. Gate-level DPA countermeasures are based on either DPL logic or masked DPL logic. Gate-level FA countermeasures are based on either detection through parity codes or timing redundancy. In addition, the paper discussed fault resiliency through DPL logic as an alternative countermeasure against fault attacks at gate level.

The target technology affects the choice of the DPL logic style that can be implemented. In FPGA, the BCDL can be a suitable choice in terms of space and effectiveness against DPA attacks, where a LUT approach is adopted. However, in ASIC implementations, WDDL tends to be more appropriate since the process of balancing the dual-rail wires is applicable. In FA countermeasures, it was shown that parity codes provide the least area and time overhead at the expense of reduced fault-detection capability. On the other hand, the PPL and SAL, provide higher detection with excessive area overhead.

The review demonstrated that the various countermeasures proposed in the literature attempt to overcome area overhead, which nearly doubles a device’s overall area, or leakage issues. However, both issues remain a challenge for gate-level DPA and FA countermeasure and hence, need concerted future research to overcome them.

7 References

Several researchers have proposed various countermeasures to protect cryptographic devices against power analysis attacks. For instance, Monnet, Renaudin, and Leveugle [25] introduced randomized look-up tables to mask the power consumption of cryptographic devices. Similarly, Mathew, Banerjee, Mahesh, Pradhan, Jabir, and others [19] proposed using differential power analysis (DPA) techniques to counteract DPA attacks.

Another approach to protect against side-channel attacks is the use of masking techniques. For example, Sining, King, Wang, and others [21] suggested using masking with differential CMOS lookup table with data independent power consumption for cryptographic applications. This technique helps to withstand differential power analysis attacks.

Furthermore, Baddam and Zwolinski [30] proposed using a digital design approach to design secure circuits. They suggested using a combination of techniques such as random clock insertion, reduced complementary delay, and isolated WDDL to protect against side-channel attacks.

Moreover, Boey, Lu, O'Neill, and Woods [8] introduced a random clock against leakage technique to protect against side-channel attacks. They used this technique in their Securing AES implementation work, where they used a high-speed block cipher. Their work was presented at the CHES 2007 conference.
81 Selmane, N., Bhasin, S., Guille, S., Graha, T., Danger, J.L.: ‘WDDL is protected against setup time violation attacks’. 2009 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC), 2009, pp. 73–83