Glitch-free is not Enough

Revisiting Glitch-Extended Probing Model

Daniel Lammers*, Nicolai Müller* and Amir Moradi

Ruhr University Bochum, Horst Görtz Institute for IT Security, Bochum, Germany
firstname.lastname@rub.de

Abstract. Today, resistance to physical defaults is a necessary criterion for masking schemes. In this context, the focus has long been on designing masking schemes guaranteeing security in the presence of glitches. Sadly, immunity against glitches increases latency as registers must stop the glitch propagation. Previous works could reduce the latency by removing register stages but only by impractically increasing the circuit area. Nevertheless, some relatively new attempts avoid glitches by applying Dual-Rail Pre-charge (DRP) logic styles. Promising works in this area include LUT-based Masked Dual-Rail with Pre-charge Logic (LMDPL), Self-Synchronized Masking (SESYM) – both presented at CHES – and Self-Timed Masking – presented at CARDIS – enabling to mask arbitrary circuits with only one cycle latency. However, even if glitches no longer occur, there are other physical defaults that may violate the security of a masked circuit. Imbalanced delay of dual rails is a known problem for the security of DRP logic styles such as Wave Dynamic Differential Logic (WDDL) but not covered in formal security models.
In this work, we fill the gap by presenting the delay-extended probing security model, a generalization of the popular glitch-extended probing model, covering imbalanced delays. We emphasize the importance of such a model by a formal and practical security analysis of LMDPL, SESYM, and Self-Timed Masking. While we formally prove the delay-extended security of LMDPL and Self-Timed Masking, we show that SESYM fails to provide security under our defined security model what causes detectable leakage through experimental evaluations. Hence, as the message of this work, avoiding glitches in combination with d-probing security is not enough to guarantee physical security in practice.

Keywords: Side-Channel Analysis · Masking · Hardware · Dual-Rail Pre-charge Logic · Robust Probing Model

1 Introduction

Side-Channel Analysis (SCA) and Masking. Even 25 years after their first discovery by Kocher [Koc96], SCA attacks continue to pose a serious threat to the confidentiality of sensitive data on cryptographic devices. Inspired by Kocher's discovery, various attacks were presented in the following years, exploiting the observation of different physical characteristics [KJJ99, GMO01], so-called side channels. For this reason, industry and academia have spent much effort to design efficient countermeasures that reliably prevent such physical attacks. In particular, masking [CJRR99], an algorithmic approach inspired from secret sharing [Sha79], stands out. Masking randomizes all sensitive intermediates of an algorithm by splitting them into multiple shares such that no incomplete set of shares gives information about any sensitive intermediate. A great advantage of masking over other countermeasures is that formal security and adversary models abstract its

*These authors contributed equally to this work.

physical security. Hence, realizing and verifying a masked implementation becomes much simpler resulting in an accelerated design process. The basic d-probing model defines a simple and abstract adversary with the ability to place probes on up to d arbitrary circuit wires [ISW03]. Each probe records one stable signal during a fixed time instance. Further, a masked circuit is secure under the d-probing model if such an adversary cannot get any information about sensitive variables.

Physical Defaults and the Robust Probing Model. However, a masked cryptographic implementation, satisfying security under the d-probing model, may become vulnerable when practically realized. For example, Mangard et al. [MPO05] mounted successful first-order Differential Power Analysis (DPA) attacks on theoretically secure masked Advanced Encryption Standard (AES) designs [AG01, OMPR05] implemented on an Application-Specific Integrated Circuit (ASIC) prototype. It turned out that unexpected combinational recombinations due to different delays in non-ideal circuits, so-called glitches, leak information on sensitive variables. In the underlying circuits, glitches propagate through the S-box computation leading to a toggle count depending on (unmasked) sensitive variables. The d-probing model itself assumes an ideal circuit and does not cover physical defaults, e.g. glitches. Hence, a d-probing secure design may still leak information due to glitches. To prevent glitch-related leakages, Mangard et al. proposed two solutions which we define as glitch-freedom and glitch-immunity [MPO05].

Definition 1 (Glitch-freedom)**.** A circuit is glitch-free iff no glitches occur inside a circuit.

Definition 2 (Glitch-immunity)**.** A circuit is glitch-immune iff it stays probing secure for every possible occurrence of a glitch.

To verify the probing security even with the addition of glitches, the robust d-probing model [FGP+18] extends the traditional d-probing model. Apart from other parameters, the robust d-probing model technically introduces glitch-extended probes allowing an adversary to get access to all intermediates that could be potentially revealed when probing an arbitrary wire. We remark that glitch-free and glitch-immune circuits are robust-probing secure. Today, a couple of masking schemes, including Threshold Implementation (TI) [NRR06] and Domain-Oriented Masking (DOM) [GMK16], guarantee glitch-immunity.

Preventing Glitches and Wave Dynamic Differential Logic (WDDL). While one branch of research focused on developing glitch-immune circuits, another had formerly tried to build glitch-free circuits for power-equalization purposes, i.e., hiding [MOP07]. A well-studied approach in this area is the application of Dual-Rail Pre-charge (DRP) logic. In DRP, a signal x is transmitted by its original wire x_t and a complementary one x_f. Hence, if both lines are zero, switching to a valid state requires a constant and data-independent amount of energy. To force the zero state, all signals are pre-charged, i.e. set to zero, before evaluating the inputs. Moreover, DRP enables to construct monotonic gadgets (see Lemma 1) building the basis of WDDL [TV04a], a DRP-based equalization technique that builds on custom cells made by standard cell libraries.

Lemma 1. *A gadget is positive and monotonic iff it performs exactly a single zero-to-one transition during evaluation and exactly a single one-to-zero transition during pre-charge.*

As all WDDL gadgets are monotonic, a circuit based on only WDDL gadgets is glitch-free. However, it was shown in [SS06] that WDDL gadgets suffer from data-dependent time-of-evaluation. This means that a WDDL gadget makes its valid output at different time instances depending on the value of the given inputs. Consequently, its dynamic power consumption still (slightly) shows a dependency on the processed data, hence, not fully achieving the power-equalization goal when observing instantaneous power consumption.

(In-)Complete Modeling of Timing Dependencies. Although WDDL is not sufficient as a stand-alone countermeasure, its glitch-free property – achieved due to employing DRP cells – offers promising possibilities in combination with other countermeasures like masking. In recent publications, e.g., [SBHM20], [SBB+22] and [NGPM22], it is believed that if a circuit is glitch-free, register stages, whose only purpose is to stop the glitch-propagation, are no longer needed. Hence, a combination of DRP and masking should allow to build masking schemes evaluating an arbitrary function within a single clock cycle. Such constant-time low-latency schemes are of particular industrial benefit due to their high potential for real-time applications, e.g. fast memory encryption. Moreover, if it is true, proving the security of a masked implementation becomes simplified as the underlying security model does not need to consider glitches. However, registers have another feature besides stopping the propagation of glitches. They synchronize intermediate states *independent of the data*, as they are controlled by the global clock signal. Without this synchronization, differences in time can propagate through the circuit, which – in the end – can lead to a data-dependent time-of-evaluation. We remark that the robust probing model does not cover timing differences in DRP logics. These effects have to be considered separately.

Our Contributions

We point out that glitch-freedom in conjunction with d-probing security is not enough to guarantee physical security. More concretely, we deal with the early propagation effects and routing imbalances on the physical security of glitch-free masked circuits. Our findings are listed as follows.

- We present the delay-extended probing model as a direct extension of the glitch-extended probing model. In short, the glitch-extended probing model is a subset of the delay-extended model. Our definition allows engineers to formally verify masked hardware circuits in a glitch-free environment.
- We analyze the security of three masking schemes that apply DRP logic and guarantee to be glitch-free by design, namely LUT-based Masked Dual-Rail with Pre-charge Logic (LMDPL) [SBHM20], Self-Synchronized Masking (SESYM) [NGPM22], and Self-Timed Masking [SBB+22]. We perform a theoretical analysis based on our defined delay-extended probing model and timing simulations. In short, we can confirm the security of LMDPL and Self-Timed Masking while SESYM becomes insecure due to routing imbalances.
- We confirm our theoretical observations by measurements on Field Programmable Gate Array (FPGA) prototypes. In particular, we detect significant leakage in the SESYM designs published by the authors.

2 Background

2.1 Notation

We denote single-bit variables by lower case letters, e.g. $x \in \mathbb{F}_2$, and vectors by upper case letters, e.g. $X \in \mathbb{F}_2^n$. For shared variables, we use superscripts to denote particular shares, e.g. x^0 as the first share of x. If a variable is represented in dual-rail logic, we use subscripts to denote specific rails. In particular, x_t denotes the original and x_f the corresponding inverted rail.

2.2 Dual-Rail Pre-charge (DRP) Logic

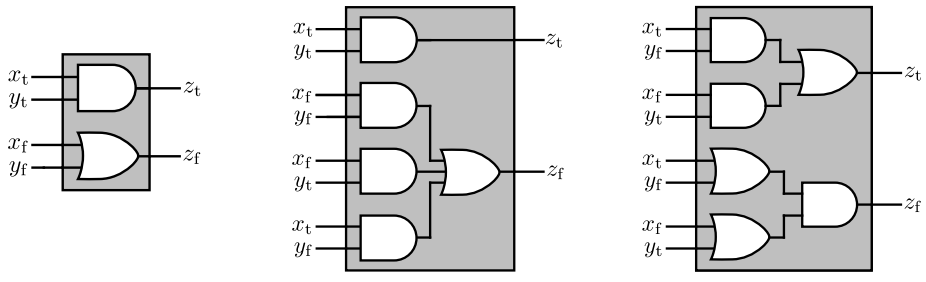
DRP logic tries to equalize the power consumption of a circuit by guaranteeing a constant number of toggles per clock cycle. Therefore, a DRP circuit should hide the data-dependent power consumption in the amplitude domain, making it resistant to DPA attacks. A circuit implementing DRP processes all signals based on two wires. The first wire drives the original value and the second one its complement. This encoding, shown below, guarantees that one rail always carries logical one while the other rail carries logical zero.

$$\left.\begin{matrix} x_\mathrm{t} = 0 \\ x_\mathrm{f} = 0 \end{matrix}\right\} x = \textsc{null}, \qquad \left.\begin{matrix} x_\mathrm{t} = 0 \\ x_\mathrm{f} = 1 \end{matrix}\right\} x = 0, \qquad \left.\begin{matrix} x_\mathrm{t} = 1 \\ x_\mathrm{f} = 0 \end{matrix}\right\} x = 1, \qquad \left.\begin{matrix} x_\mathrm{t} = 1 \\ x_\mathrm{f} = 1 \end{matrix}\right\} x = \textsc{invalid}$$

To equalize power consumption, it is necessary to apply a pre-charge technique, which sets all circuit wires, i.e. both rails of all signals, to logical zero, i.e. the NULL state. This so-called pre-charge phase alternates with an evaluation phase, which computes the actual outputs. As all signals switch from NULL to a valid state, and all valid states result in a single toggle per signal, the number of toggles in the entire circuit (and ideally power consumption) becomes constant for every possible input value.

2.3 Wave Dynamic Differential Logic (WDDL)

WDDL [TV04a] is a known design technique that replaces standard cells with compound standard cells, i.e. gadgets. The pursued goal is a circuit with a constant power consumption, independent of the processed data. The block diagram of different WDDL gadgets are depicted in Figure 1. It is noteworthy that it has been shown by Suzuki et al. in [SS06] that WDDL gadgets may fail to keep the instantaneous power consumption constant when the inputs arrive at different times instances. The underlying concept is referred to as *early propagation* effect or *data-dependent* time of evaluation.



(a) WDDL-AND2 (b) WDDL-AND2-noEE (c) WDDL-XOR2

(d) WDDL-AND2 symbol (e) WDDL-XOR2 symbol

Figure 1: WDDL gadgets.

Definition 3 (Early Propagation Effect)**.** Some basic gates can uniquely determine their logical output without considering their entire logical inputs. Accordingly, such gates can propagate their output early, i.e. before all input signals have arrived at the gate.

Example 1 (Early Propagation Effect)**.** Let us consider the OR gate in the WDDL-AND2 shown in Figure 1(a) with the input signals x_f and y_f and output signal z_f. If we assume that $x_\mathrm{f} = 1$, then $z_\mathrm{f} = 1$ is propagated without any knowledge about y_f.

Bahsin et. al [BGF+10] tried to avoid early propagation and introduced WDDL-noEE, shown in Figure 1(b). However, this only avoids early propagation from the pre-charge to the evaluation phase, while a global pre-charge signal (connected to all gadgets) can mitigate the early propagation from the evaluation to the pre-charge phase. Subsequently, Moradi et. al [MI14] introduced AWDDL, an asynchronous variation of WDDL, which handles early propagation effect in both phases under the assumption that both rails of the same variable have no delay imbalance. Additionally, the authors presented a custom router, minimizing the risk of routing imbalances by taking care of the placement in an FPGA design. However, they pointed out that the routing is not perfectly balanced and remains a challenge in practice. It is worth mentioning that the fat wire approach [TV04b] also aims to avoid delay imbalances in ASIC design processes, though the capacitances of dual-rail wires in a fabricated circuit can still slightly differ.

2.4 Masking

Masking based on the concept of secret sharing [Sha79] is the predominant approach to protect hardware circuits against SCA attacks [CJRR99]. In *d-order Boolean Masking*, every sensitive variable $X \in \mathbb{F}_2^n$ is randomized by being split into a set of $d+1$ *shares* $\{X^0, ..., X^d\} \in (\mathbb{F}_2^n)^{d+1}$ which are distributed uniformly and randomly. Every sharing can exemplary be initialized by randomly sampling the first d shares and computing X^d such that $X^d = (\bigoplus_{i=0}^{d-1} X^i) \oplus X$. To achieve the desired security with $d+1$ shares, the target algorithm is transformed into a masked algorithm by performing every operation on the shared representation of sensitive variables.

2.4.1 Domain-Oriented Masking (DOM)

The DOM scheme allows to create d-order secure Boolean masked circuits by utilizing $d+1$ shares [GMK16]. The idea is to split a circuit into $d+1$ independent share domains, receiving at most one share per variable. While linear operations achieve domain independence by processing each share separately, non-linear operations, such as the first-order secure DOM multiplier shown in Figure 4(a), must combine shares across domain borders. To keep domain independence, some intermediates are blinded by fresh randomness and synchronized by registers to avoid combining both shares of the same variable.

2.5 Security Models

The basic d-probing model [ISW03] defines an adversary getting access to up to d intermediates of an ideal circuit. Based on the capabilities of the so-called *d-probing adversary*, *d-probing security* is defined as follows.

Definition 4 (d-probing security)**.** A circuit is d-probing secure iff every d-tuple of intermediates does not give any information about any sensitive variable.

To model the observation set, an attacker places up to d probes on arbitrary wires. Afterwards, each probe records a stable signal at one point in time.
The robust d-probing model [FGP+18] builds on top of the above-given d-probing model and extends it to cover physical defaults, i.e. glitches, transitions, and couplings.

Definition 5 (Glitch)**.** If the inputs of a gate arrive asynchronously, the gate processes an unstable input state leading to temporary changes in the output. Accordingly, a gate propagates an unexpected signal before reaching its stable output. We refer to such a short-time transient faulty output as a glitch [MPG05].

Example 2 (Glitch)**.** As given in [NRR06], we consider a single AND gate with the input signals x and y and output signal z. If we assume that x arrives significantly later than y

and a transition from stable inputs $(x, y) = (1, 0)$ to $(x, y) = (0, 1)$, then z will temporary change from 0 to 1 – due to the transient input state $(x, y) = (1, 1)$ – before reaching the stable output $z = 0$.

Definition 6 (Transition)**.** If the circuit overwrites a memory element, i.e. a register, the corresponding power consumption depends on both the old and new memory states. This overwriting effect is referred to as a transition.

Example 3 (Transition)**.** Consider a variable with two shares as $x = x^0 \oplus x^1$. If a circuit consecutively stores x^0 and x^1 in the same register cell, a transition leaks information about x.

Definition 7 (Coupling)**.** Consider a pair of two adjacent wires (w, t). It holds that the switching activity of w influences t and vice versa through their inter-wire capacitance [CBG+17]. Such an influence is known as coupling.

Example 4 (Coupling)**.** Consider a variable with two shares as $x = x^0 \oplus x^1$. If a circuit drives x^0 and x^1 on adjacent wires, coupling may lead to crosstalk leaking information about x.

Again, the adversary can create a d-probing set but, moreover, all probes are extended in order to cover the desired physical defaults [FGP+18].

Definition 8 (Glitch-extended Probe)**.** A glitch-extended probe on wire w models the impact of glitches by recording all stable signals that contribute to w.

Definition 9 (Transition-extended Probe)**.** A transition-extended probe on wire w models the impact of transitions by recording two consecutive values carried by w.

Definition 10 (Coupling-extended Probe)**.** A coupling-extended probe on wire w models the impact of couplings by recording multiple wires adjacent to w.

This leads to the following definition.

Definition 11 (Robust d-Probing Security [FGP+18])**.** A circuit is robust d-probing secure iff every d-probing set, containing up to d extended probes, does not give any information about any sensitive variable.

In this work, we consider the robust probing model including glitches but excluding transitions and couplings, colloquially named as glitch-extended d-probing model.

2.5.1 Worst-Case vs. Fine-Grained Leakage Models

It holds that the robust probing model covers glitches and transitions by assuming a worst-case scenario, i.e. every possible occurrence of a glitch (resp. transition) is considered. We remark that only particular glitches (resp. transitions) occur in a physical circuit while others never occur. Hence, worst-case models tend to be over-conservative as they consider physical defaults that may never occur in a concrete physical realization of a design. However, worst-case models are the only way to prove the security of arbitrary circuits without any fine-grain characterization. Note that couplings cannot be modeled with a worst-case model. If we consider all possible couplings, i.e. all wires jointly couple, every design becomes insecure as the wires driving all different shares jointly couple. To successfully prove security in presence of couplings, the adversary is restricted in a way that only particular couplings, e.g. between at most t wires, are recorded. Such a fine-grained model requires a careful definition as the restrictions must still fit the particular physical reality.

2.6 Composability

The design and evaluation of provably robust-probing secure circuits become infeasible with increasing design complexity, e.g. for higher-order security. To avoid this problem, tiny and provably secure building blocks were introduced which can be put together to realize provably secure functions of arbitrary order. We refer to these building blocks as composable gadgets [CGLS21]. Such gadgets allow the transformation of any unprotected gate-level netlist into a masked one by simply replacing any unprotected gate with a composable gadget. In practice, the transformation can be performed automatically, e.g. by AGEMA [KMMS22].

Definition 12 (Composability)**.** A set of gadgets, which are individually secure under a certain security model, are composable iff all their possible combinations are still secure under the same security model.

Therefore, an arbitrary circuit made out of composable gadgets is d-probing secure if all underlying gadgets are composable. To formally prove the gadget's composability, Cassiers et al. introduced the d-Probe-Isolating Non-Interference (PINI) security notion based on restricting the probe propagation to the probe's share domain [CS20].

Definition 13 (Perfect Probe Simulation)**.** Consider a set $\mathcal{P}$ of d (extended) probes on a gadget. $\mathcal{P}$ is perfectly simulatable with a set $\mathcal{S}$ of input shares if there exists a probabilistic polynomial time simulator calculating a joint probability distribution based on S which is identical to the distribution of $\mathcal{P}$.

Definition 14 (d-Probe-Isolating Non-Interference (PINI))**.** Consider a set $\mathcal{P}$ of d_0 (extended) probes on a gadget and a set $\mathcal{S}$ of d_1 (extended) probes on the gadget's outputs while O contains all share indices probed by probes in $\mathcal{S}$. A gadget is d-PINI iff for every $\mathcal{P} \cup \mathcal{S}$ with $d_0 + d_1 \leq d$, there exists a set $\mathcal{I}$ of d_0 share indices such that the wires probed by probes in $\mathcal{P} \cup \mathcal{S}$ is perfectly simulatable from shares with indices in $\mathcal{I} \cup \mathcal{O}$ [CS20].

3 Single-Cycle Masking in Hardware

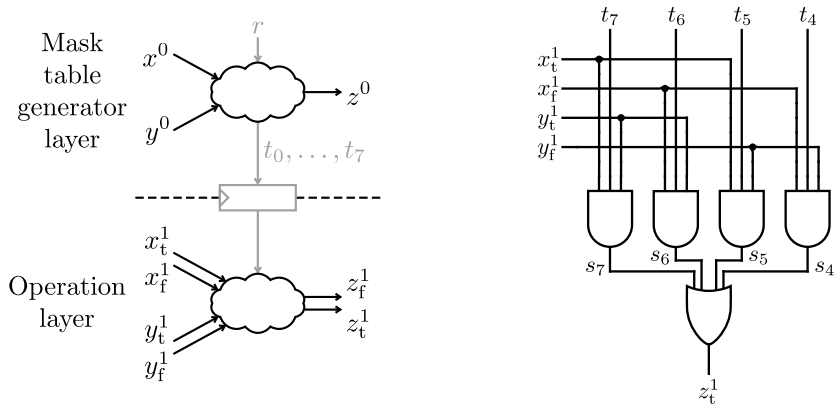
Besides area and randomness requirements, efficiency concerning latency tends to be relevant for conventional masking schemes, particularly those grounded on composable gadgets. We review three composable masking schemes, the widely-considered LMDPL, the recently proposed SESYM, and the asynchronous Self-Timed Masking. All incorporate DRP principles.

3.1 LUT-based Masked Dual-Rail with Pre-charge Logic (LMDPL)

LMDPL [LMW14] enables the first-order secure masking of arbitrary circuits while achieving a constant latency of two clock cycles, i.e. with a dedicated pre-charge and evaluation cycle. In a follow-up work, Sasdrich et al. [SBHM20] proved the formal security of LMDPL under the glitch-extended probing model and showed that gadgets based on LMDPL are provably composable. Each gadget processes two shares per variable, and a single fresh mask blinds one share if the gadget is non-linear. All LMDPL gadgets are divided into two layers, as depicted in Figure 2(a). The mask table generator layer handles the first share of all input variables (x^0, y^0) to compute the first output share z^0. The operation layer takes the second shares in their dual-rail representation ($x^1_\mathrm{t}, x^1_\mathrm{f}, y^1_\mathrm{t}, y^1_\mathrm{f}$) and computes z^1_t and z^1_f. As the operation layer implements the DRP style, it is guaranteed that no glitches occur.

In case of a non-linear gadget, a register stage separates both layers, and a fresh mask blinds all masked table outputs $t_0, \ldots, t_7$ which are computed as follows:

$$z^0 = r, \qquad t_{4+2i+j} = F(x^0 \oplus j, y^0 \oplus i) \oplus r, \qquad t_{2i+j} = t_{4+2i+j} \oplus 1, \qquad \text{where } i, j \in \{0, 1\}.$$



(a) LMDPL gadget layers, gray colored components are needed if the gadget is non-linear.

(b) One output rail of a non-linear LMDPL gadget operation layer. The second output rail z^1_f is generated analogously.

Figure 2: LMDPL gadget overview.

The operation layer processes all $t_0, \ldots, t_7$ by instantiating two 4-to-1 multiplexers while x^1 and y^1 act as a 2-bit select signal. We remark that the authors analyzed the timing of the multiplexers and claim that these multiplexers satisfy a time-of-evaluation independent of any sensitive variable. In addition to the second share variables (x^1_t, x^1_f, y^1_t, y^1_f), all $t_0, \ldots, t_7$ which are stored in registers should follow a pre-charge/evaluation fashion per clock cycle to guarantee no glitches.

3.2 Self-Synchronized Masking (SESYM)

Nagpal et al. [NGPM22] described a generic procedure to convert any Boolean-masked circuit into a single-cycle glitch-extended d-probing secure circuit with no need for additional randomness. The process involves converting single-rail inputs to dual-rail and employing asynchronous primitives to achieve single-cycle processing. A so-called completion detector module (a product-of-sums form of all dual-rail outputs) detects the completion of the evaluation phase for every output variable, and immediately induces a pre-charge wave into the masked circuit. In parallel, an array of Muller C-elements holds the stable output signals and acts as a dual-to-single-rail converter. Figure 3 gives a rough overview.

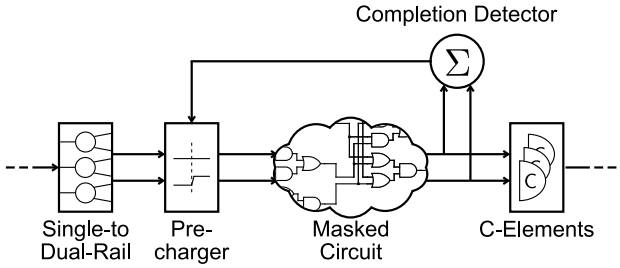
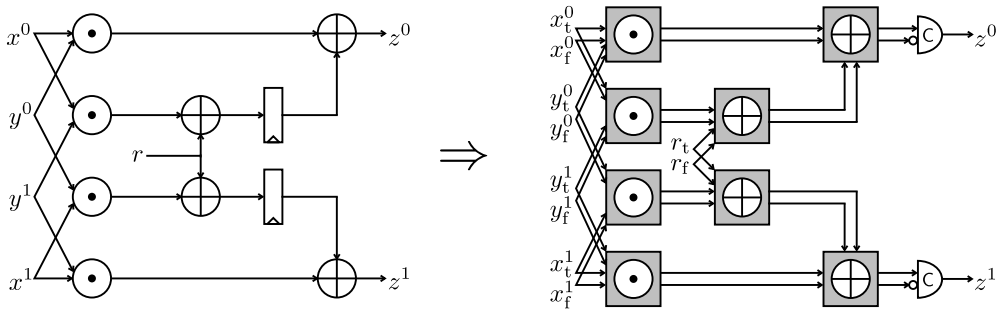


Figure 3: SESYM components.

The authors showed their conversion based on a DOM based masking of two S-box designs. SESYM replaces all gates in DOM gadgets by their WDDL counterparts (cf. Figure 4). As the transformed circuit already guarantees glitch-freeness, SESYM removes the re-sharing registers.



(a) DOM-AND2 (b) SESYM-AND2

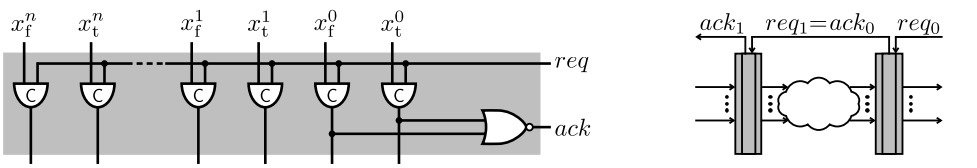
Figure 4: Conversion of a DOM-AND2 into a SESYM-AND2.

The authors justified the register removal with the glitch-freeness. However, the impact of other timing-dependent effects is assumed to be unproblematic and not covered by their underlying adversary model. We highlight that the idea is based on the following observation.

> "... the only purpose of registers within a masked non-linear circuit is to prevent glitches from propagating to the share compression stage. We can safely remove these registers if we can guarantee glitch-free share compression through other means." [NGPM22]

3.3 Self-Timed Masking

The generic approach of Simoes et al. [SBB+22] incorporates DOM gadgets for masking, Muller C-elements for synchronization and a DRP logic style with WDDL-noEE. The basic idea is to replace registers with self-timed latches, which use the state of dual rails to handle a handshaking mechanism. The authors presented two variants, namely weakly indicating and strongly indicating latches. Every latch encompasses C-elements that process both rails of n inputs together with a *req* signal (see Figure 5(a)). Since a C-element only forwards equal inputs, the latches synchronize the pre-charge or evaluation results of a preceding combinatorial circuit and propagate them as soon as *req* arrives. Immediately afterwards, a completion detector toggles the *ack* signal based on the dual-rail state of all C-element outputs. Alternatively, the weakly indicating latch simplifies the handshake by processing the dual-rail state of only one output share. Thus, the completion of a phase can be guaranteed for the preceding circuit, and the computation of the succeeding circuit can be induced. Successively, consecutive latches receive the *ack* signal of the succeeding latch mapped to their *req* input signal in reverse order, constructing a pipeline (see Figure 5(b)). As a result, arbitrary combinational circuits can be translated into circuits following the Self-Timed Masking scheme and evaluate in a single clock cycle by multiple runs of the pipeline equivalent to its depth. The authors presented a serialized design, where only



(a) Latch internals (b) Pipeline with latch symbols

Figure 5: Weakly-indicating self-timed latches.

the S-box follows the introduced scheme with weakly indicating latches and computes four S-boxes consecutively by means of a multiplexer. The S-Box inputs are transformed from single-rail to dual-rail and vice versa for its outputs. As the authors stated, this construction does not need a clock signal per computation (i.e., self-timed), but suffers from low throughput due to its higher delay compared to synchronous designs which can run at a high clock frequency. Hence, the underlying concept may not be considered as a low-latency approach.

4 A Deeper Look

In this section, we deeply analyze the effects of asynchronously arriving inputs in glitch-free circuits. To formally discuss the issues due to imbalanced timings, we abstract the considered effects with delays according to Definition 15.

Definition 15 (Delay)**.** A delay Δ_w of wire w describes the time lag between a combinational input transition, i.e. register-state or primary-input toggles, and the toggle on wire w with respect to the observation spot.

Without loss of generality, we define delays in terms of wires and neglect the time required for a gate to compute its outputs. However, we model lags on gates by adding them to the delay of the gate's output wire. We define the observation spot as the wire cross-section where a toggle is observed with a particular delay. While the exact delay depends on the observation spot, we ignore focusing on a particular observation spot on w since all assumptions hold for any arbitrary observation spot on the same wire. According to Section 2, for circuits that implement DRP we can make Definition 15 more explicit, resulting in Lemma 2.

Lemma 2. *During the pre-charge phase, Δ_w describes the delay between the start of the pre-charge phase and the toggle of w from 1 to 0. During the evaluation phase, Δ_w describes the delay between the start of the evaluation phase and the toggle of w from 0 to 1.*

Initially, based on concrete examples, we visualize the underlying problem originating from delays.

Example 5 (WDDL-AND2 with delay)**.** Based on the sketch in Figure 1(a), consider a single WDDL-AND2 gadget with primary signals x, y, and z in dual-rail representation, i.e. $(x_\mathrm{t}, x_\mathrm{f})$, $(y_\mathrm{t}, y_\mathrm{f})$, and $(z_\mathrm{t}, z_\mathrm{f})$. Moreover, we assume that y_f is delayed compared to the other inputs. Hence, it holds that $\Delta_{y_\mathrm{f}} > \Delta_{x_\mathrm{f}}$ while $\Delta_{x_\mathrm{f}} = \Delta_{x_\mathrm{t}} = \Delta_{y_\mathrm{t}}$. Figure 6 visualizes the timing for all possible input vectors. For simplicity, we assume that $\Delta_{x_\mathrm{f}} = \Delta_{x_\mathrm{t}} = \Delta_{y_\mathrm{t}} = 0$, i.e. we ignore all delays except Δ_{y_f}.

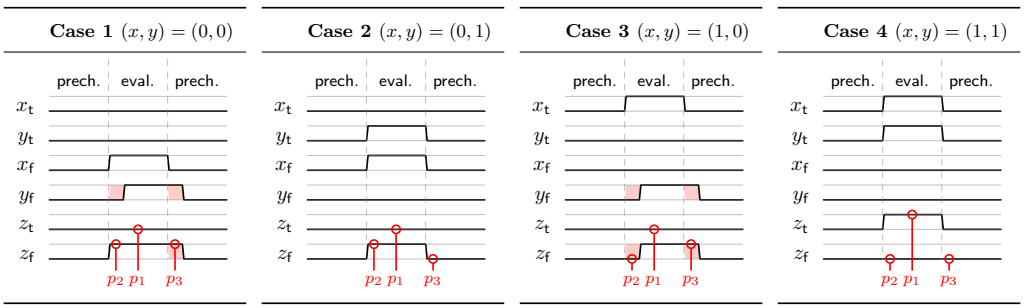


Figure 6: WDDL-AND2 timings, y_f delayed by $\Delta_{y_\mathrm{f}} > \Delta_{y_\mathrm{t}} = \Delta_{x_\mathrm{t}} = \Delta_{x_\mathrm{f}} = 0$.

Case 1 (Probe on z_t). Consider an adversary placing a probe p_1 on z_t as shown in Figure 6. If p_1 records 1, x_t and y_t must be 1. Moreover, the adversary gets information about Δ_{x_t} and Δ_{y_t} as both arrived before p_1 records.

Case 2 (Probe on z_f). Consider an adversary placing a probe p_2 on z_f as shown in Figure 6. If p_2 records 1 (resp. 0), the adversary recovers $x_f = 1$ (resp. $x_f = 0$) and information about Δ_{x_f}. Moreover, consider an adversary placing a probe p_3 on z_f as shown in Figure 6. If p_3 records 1 (resp. 0), the adversary recovers $y_f = 1$ (resp. $y_f = 0$) and information about Δ_{y_f}.

Example 6 (WDDL-XOR2 with delay)**.** Following Figure 1(c), consider a single WDDL-XOR2 gadget with primary signals x, y, and z in dual-rail representation, i.e. (x_t, x_f), (y_t, y_f), and (z_t, z_f). Moreover, we assume that y_t is delayed compared to the other inputs. Hence, it holds that $\Delta_{y_t} > \Delta_{x_t} = \Delta_{x_f} = \Delta_{y_f}$. Figure 7 visualizes the timing for all possible input vectors. For simplicity, we assume that $\Delta_{x_t} = \Delta_{x_f} = \Delta_{y_f} = 0$.

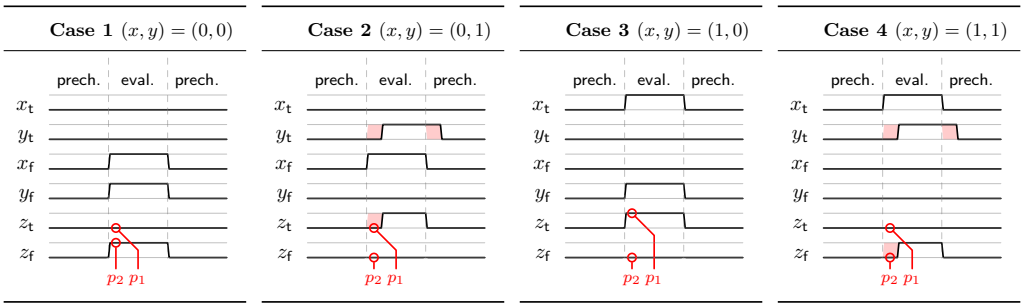


Figure 7: WDDL-XOR2 timings, y_t delayed by $\Delta_{y_t} > \Delta_{y_f} = \Delta_{x_t} = \Delta_{x_f} = 0$.

Case 1 (Probe on z_t). Consider an adversary placing a probe p_1 on z_t as shown in Figure 7. If p_1 records 1, it holds that $x_t = 1$ and $y_f = 1$. Moreover, the adversary gets information about Δ_{x_t} and Δ_{y_f} as both arrived before p_1 records.

Case 2 (Probe on z_f). Consider an adversary placing a probe p_2 on z_f as shown in Figure 7. If p_2 records 1, x_f and y_f must be 1. Moreover, the adversary gets information about Δ_{x_f} and Δ_{y_f} as both arrived before p_2 records.

Despite the glitch-freedom of WDDL-AND2 and WDDL-XOR2, Example 5 illustrates the information propagation due to different timings which are not considered by the robust probing model. In particular, probes on the gadget's outputs propagate to (one or many) gadget inputs while glitch-extended probing model implies that the probes on DRP gadgets' outputs do not propagate as there is no glitches. Hence, even if glitches are completely avoided, an information flow through the combinational logic is still possible which means that security under the robust d-probing model is not accurate when evaluating a DRP circuit. We formalize our findings on Example 5 and Example 6 in Lemma 3.

Lemma 3. *If a circuit is glitch-free, it is not enough to guarantee physical security by means of probing model.*

We remark that this problem does not solely originate from the early propagation effect. For example, if WDDL-AND2-noEE is employed in Example 5, the same issue can be observed when the delays of the rails are not balanced.

4.1 Delay-Extended Probing Model

We start to study the propagation of delays based on atomic gates and continue with more advanced constructions, such as WDDL. Finally, we focus our analysis on LMDPL, SESYM,

and Self-Timed Masking, reviewed in Section 2. Here, we introduce the (worst-case) delay-extended probing model to capture the effect of delays in a glitch-free environment and investigate the security of aforementioned schemes under the new model. Initially, we define a probe capturing the effect of all possible delays.

Definition 16 (Delay-extended Probe)**.** A delay-extended probe on wire w models the impact of all possible delays of w.

4.1.1 Relation to the Glitch-Extended Probing Model

As glitches are a particular effect of delays, the delay-extension is strongly related to the known glitch-extension. In particular, both extended probes are restricted to record only at one time instance. To formally show the relation between glitches and delays we introduce Lemma 4 and Lemma 5.

Lemma 4. *Consider a circuit which is not glitch-free, i.e. no DRP. Then, it holds that, glitch-extended probes $\Leftrightarrow$ delay-extended probes. In particular, it holds that every glitch-immune circuit is delay-extended secure and vice-versa.*

Proof. First, we prove that every glitch-extended probe covers a delay-extended probe ($\Rightarrow$). Then, we prove that every delay-extended probe covers a glitch-extended probe ($\Leftarrow$).

- $\Rightarrow$ Consider a glitch-extended probe. Following Definition 8, a glitch-extended probe on wire w records the whole combinational logic affecting w. As delays only propagate through combinational logic, the glitch-extended probe models delays as well.
- $\Leftarrow$ Consider a delay-extended probe. According to Definition 5, glitches are caused by imbalanced delays. As glitches are a subcategory of delay effects, the impact of glitches is modeled by a delay-extended probe.

□

We should highlight that Lemma 4 only holds for circuits that are not glitch-free. If a circuit is glitch-free, e.g. by DRP, our model complements the glitch-extension, i.e. the delay-extension covers more effects due to delays than just glitches.

Lemma 5. *Consider a circuit which is glitch-free and d-probing secure. Then, Lemma 4 does not hold as the circuit is not necessarily delay-extended d-probing secure.*

Lemma 5 formalizes that other delay-caused effects besides glitches exist. For example, the early propagation effect (see Definition 3) may violate the security even if a circuit is glitch-free and d-probing secure.

4.1.2 Delay-Extended Probe Propagation

To formally argue about the effect of delays in glitch-free circuits, we investigate the information propagation due to imbalanced wire delays. To keep it simple, we start with pre-charged atomic gates, namely AND2 and OR2. We assume that the gates start from the pre-charge phase, i.e. all signals drive 0. Then, we consider the time-of-evaluation, i.e. the point in time when the output reaches its valid state.

Example 7 (AND2 with delay)**.** Consider one AND2 gate with inputs x, y initially driving 0, and output z. Logically, AND2 only evaluates z to 1, iff $x = y = 1$. Therefore, if an adversary places a probe on z and records 1, it holds that $x = 1$ and $y = 1$, i.e. both inputs are revealed. Moreover, assume that AND2 evaluates to 1 with a delay of Δ_z. Hence, an adversary that probes z and records 1 learns as follows.

- $(\Delta_x = \Delta_z \Rightarrow \Delta_y \leq \Delta_z)$ or

- $(\Delta_y = \Delta_z \Rightarrow \Delta_x \leq \Delta_z)$, i.e. AND2 evaluates as soon as the subsequent input arrives.

Hence, in a worst-case scenario, i.e. $\Delta_x = \Delta_y = \Delta_z$ with $z = 1$, an adversary learns (x, Δ_x) and (y, Δ_y) by probing z with a delay of Δ_z. Thus, observing Δ_x and Δ_y allows to further back-propagate the delay-extended probes on x and y as formalized in Lemma 6.

Lemma 6. *When all inputs initially drive 0, a delay-extended probe on the AND output cascades to delay-extended probes on all its inputs.*

Example 8 (OR2 with delay)**.** Consider one OR2 gate with inputs x, y initially driving 0, and output z. Logically, OR2 evaluates z to 1, iff at least one input is 1. Therefore, if an adversary places a probe on z and records 1, it holds that $x = 1$ or $y = 1$. Moreover, assume that OR2 evaluates to 1 with a delay of Δ_z. Hence, an adversary that probes z and records 1 learns as follows.

- $(x = 1, \Delta_x = \Delta_z) \Rightarrow (y = 0 \text{ or } \Delta_y \geq \Delta_z)$ or
- $(y = 1, \Delta_y = \Delta_z) \Rightarrow (x = 0 \text{ or } \Delta_x \geq \Delta_z)$, i.e. OR2 evaluates as soon as one input with logical value 1 arrives without knowing the subsequent input.

Due to early evaluation, it holds that the time when z evaluates to 1 only depends on one of the inputs. Hence, an adversary learns either (x, Δ_x) or (y, Δ_y) as summarized in Lemma 7.

Lemma 7. *When all inputs initially drive 0, a delay-extended probe on the OR output cascades to exactly one delay-extended probe on one of its inputs.*

For simplicity, Example 7 and Example 8 considered only gates with two inputs. However, we remark that Lemma 6 and Lemma 7 hold for gates with an arbitrary number of inputs.

4.1.3 Delay-Extended Security Notions

Based on Definition 16, we can adapt Definition 11 and Definition 14 to the delayed environment.

Definition 17 (Delay-extended d-Probing Security)**.** A circuit is delay-extended d-probing secure iff every d-probing set, containing up to d delay-extended probes, does not give any information about any sensitive variable.

Definition 18 (Delay-extended d-Probe-Isolating Non-Interference (PINI))**.** Consider a set $\mathcal{P}$ of d_0 delay-extended probes on a gadget and a set $\mathcal{S}$ of d_1 delay-extended probes on the gadget's outputs. A gadget is delay-extended d-PINI iff for every union set $\mathcal{P} \cup \mathcal{S}$ with $d_0 + d_1 \leq d$ there exists a set $\mathcal{I}$ of d_0 input indices and a set $\mathcal{O}$ of d_1 output indices such that $\mathcal{P}$ is perfectly simulatable from shares with indices in the union set $\mathcal{I} \cup \mathcal{O}$

In short, it is noteworthy to restate that when dealing with not glitch-free circuits, the delay-extended probing model is totally the same as the glitch-extended probing model. The delay-extended probing model captures more effects only when evaluating glitch-free circuits utilizing DRP.

4.2 WDDL Gadgets

To simplify the analysis of the following schemes and to show the relation between our theoretical findings and a practical adversary we apply our model to the WDDL gadgets presented in Example 5 and Example 6.

First, consider Example 5 and an adversary placing a delay-extended probe p_{z_t} (resp. p_{z_f}) on z_t (resp. z_f). Due to Lemma 6 and Lemma 7, it holds that

$$p_{z_\mathrm{t}} \to \{p_{x_\mathrm{t}}, p_{y_\mathrm{t}}\}, \qquad p_{z_\mathrm{f}} \to \{p_{x_\mathrm{f}}\}, \{p_{y_\mathrm{f}}\}. \tag{1}$$

Equation (1) reflects the practical results of Example 5. In particular, it holds that $p_1 = p_{z_\mathrm{t}}$ while $p_2 \to \{p_{x_\mathrm{f}}\}$ and $p_3 \to \{p_{y_\mathrm{f}}\}$.

Further, consider Example 6 and an adversary placing a delay-extended probe p_{z_t} (resp. p_{z_f}) on z_t (resp. z_f). Due to Lemma 6 and Lemma 7, it holds that

$$p_{z_\mathrm{t}} \to \{p_{x_\mathrm{t}}, p_{y_\mathrm{f}}\}, \{p_{x_\mathrm{f}}, p_{y_\mathrm{t}}\}, \qquad p_{z_\mathrm{f}} \to \{p_{x_\mathrm{t}}, p_{x_\mathrm{f}}\}, \{p_{x_\mathrm{t}}, p_{y_\mathrm{t}}\}, \{p_{x_\mathrm{f}}, p_{y_\mathrm{f}}\}, \{p_{y_\mathrm{t}}, p_{y_\mathrm{f}}\}. \tag{2}$$

Equation (2) reflects the practical results of Example 6 as $p_1 \to \{p_{x_\mathrm{t}}, p_{y_\mathrm{f}}\}$ and $p_2 \to \{p_{x_\mathrm{f}}, p_{y_\mathrm{f}}\}$, i.e. one possible probing set that results from extending p_{z_t} and p_{z_f}. However, for each concrete example (as Example 6) p_{z_t} and p_{z_f} only expand to a single probing set while the other possible probing sets are relevant for other examples with different delays.

4.3 Non-linear LMDPL Gadget

Consider a non-linear LMDPL-AND2 gadget with shared primary inputs (x^0, x^1) satisfying $x = x^0 \oplus x^1$, (y^0, y^1) satisfying $y = y^0 \oplus y^1$, and a shared primary output (z^0, z^1) satisfying $z = z^0 \oplus z^1$. The mask table generator layer processes (x^0, y^0) together with a fresh mask r to compute z^0. The operation layer processes the dual-rail representation of the second share $(x^1_\mathrm{t}, x^1_\mathrm{f}, y^1_\mathrm{t}, y^1_\mathrm{f})$ and returns $(z^1_\mathrm{t}, z^1_\mathrm{f})$.

4.3.1 Probing Security

As LMDPL gadgets are restricted to only first-order security, we also consider only the first-order delay-extended model.

Lemma 8. *An LMDPL-AND2 gadget achieves delay-extended* 1*-probing security.*

Proof. First, we prove the delay-extended probing security if the adversary places on probe in the mask table generator layer. Afterwards, we consider adversaries placing one delay-extended probe on a wire in the operation layer.

- The mask table generator layer does not operate in DRP. Hence, glitch- and delay-extensions are equal. In case an adversary places a glitch-extended probe p on a table output $t_0, \ldots, t_7$, p records all primary inputs of the layer, i.e. $p \to \{p_{x^0}, p_{y^0}, p_r\}$. However, this gives no information about any sensitive variable, as only variables within the first share domain are observed.
- The operation layer (cf. Figure 2(b)) is separated by a register stage from the mask table generator layer, implements the DRP logic, and purely consists of monotonic gates. Thus, the operation layer is glitch-free what makes the delay-extended probing model applicable. Consider an adversary placing a delay-extended probe p on z^1_t (a probe on z^1_f behaves analogously). According to Definition 16, we substitute z^1_t by the following probing sets.

  $$p_{z^1_\mathrm{t}} \to \{p_{x^1_\mathrm{f}}, p_{y^1_\mathrm{f}}, p_{t_4}\}, \{p_{x^1_\mathrm{t}}, p_{y^1_\mathrm{f}}, p_{t_5}\}, \{p_{x^1_\mathrm{f}}, p_{y^1_\mathrm{t}}, p_{t_6}\}, \{p_{x^1_\mathrm{t}}, p_{y^1_\mathrm{t}}, p_{t_7}\}$$

  Note that the observed $t_4, \ldots, t_7$ is blinded by r in the mask table generator layer, and the blinding cannot be removed meaning that every probing set derived from $p_{z^1_\mathrm{t}}$ (resp. $p_{z^1_\mathrm{f}}$) is independent of x and y.

□

4.3.2 Composability

The authors of LMDPL already proved the composability of their design by showing glitch-extended 1-SNI [BBD+16]. Additionally, we show that LMDPL also satisfies delay-extended 1-PINI.

Lemma 9. *It holds that an LMDPL-AND2 gadget satisfies delay-extended* 1*-PINI.*

Proof. As $d = 1$, based on Definition 18 it holds that either $d_0 = 1, d_1 = 0$ or $d_0 = 0, d_1 = 1$. We prove every relevant wire in both layers separately and confirm their simulatability.

- **Mask table generator layer:** Consider a set with one glitch/delay-extended probe on an arbitrary intermediate wire or z^0. It holds that every possible wire in the mask table generator layer is independent of the second share $(x^1_\mathrm{t}, x^1_\mathrm{f}, y^1_\mathrm{t}, y^1_\mathrm{f})$, i.e. it depends only on x^0, y^0, and a random bit. Hence, every wire in the mask table generation layer is perfectly simulatable by the shares with index 0 (x^0, y^0) and by tossing a fair coin, i.e. with a set of share index $\{0\}$.
- **Operation layer:** Consider a set with one delay-extended probe on an arbitrary intermediate wire in the operation layer or on either z^1_t or z^1_f. As before, it holds that such a delay-extended probe only observes shares with share index 1, i.e. $(x^1_\mathrm{t}, x^1_\mathrm{f}, y^1_\mathrm{t}, y^1_\mathrm{f})$ and the randomly blinded output of a single register $t_0, \ldots, t_7$. Hence, every wire in the operation layer is perfectly simulatable by the shares with index 1 and by tossing a fair coin, i.e. with a set of share index $\{1\}$.

□

4.4 Non-linear SESYM Gadget

For our analysis, we investigate a SESYM-AND2 gadget receiving x and y as inputs while the output is denoted by z. The gadget is d-order masked, i.e. $x = x^0 \oplus \cdots \oplus x^d$, $y = y^0 \oplus \cdots \oplus y^d$, and $z = z^0 \oplus \cdots \oplus z^d$, and operates on the dual-rail representation of each intermediate. Hence, each signal corresponds to a tuple of signals.

$$x^i \to (x^i_\mathrm{t}, x^i_\mathrm{f}), \qquad y^i \to (y^i_\mathrm{t}, y^i_\mathrm{f}), \qquad z^i \to (z^i_\mathrm{t}, z^i_\mathrm{f})$$

We visualize the last share domain of a d-order SESYM-AND2 gadget in Figure 8.

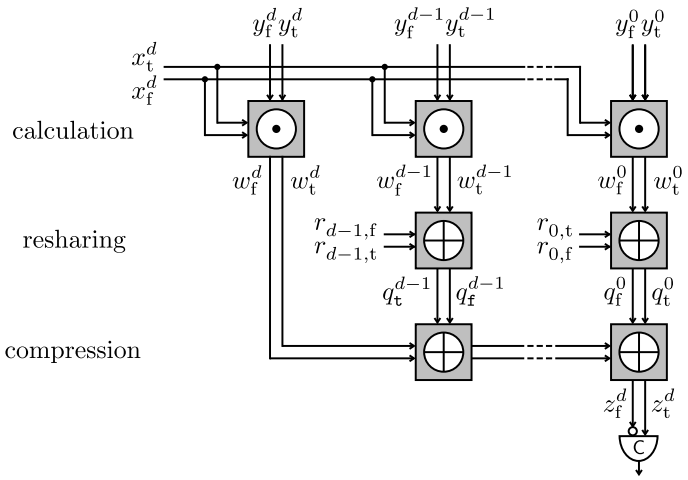


Figure 8: Last share domain of a d-order SESYM-AND2 gadget.

4.4.1 Probing Security

Initially, we formally analyze the gadget's security under the delay-extended probing model, i.e. by proving the following lemmas.

Lemma 10. *Consider a d-probing secure and glitch-free SESYM-AND2 gadget. It holds that the SESYM-AND2 gadget does not achieve delay-extended* 1*-probing security.*

Proof. Consider one delay-extended probe $p_{z_\mathrm{t}^d}$ on z_t^d. According to Equation (2), it holds that a probe placed on z_t^d propagates to one rail of every share compression layer and, further, to every input of the re-sharing stage, i.e.

$$p_{q_\mathrm{f}^0} \rightarrow \{p_{w_\mathrm{t}^0}, p_{r_{0,\mathrm{t}}}\}, \cdots, p_{q_\mathrm{f}^{d-1}} \rightarrow \{p_{w_\mathrm{t}^{d-1}}, p_{r_{d-1,\mathrm{t}}}\}$$
$$p_{z_\mathrm{t}^d} \rightarrow \{p_{w_\mathrm{t}^d}, p_{q_\mathrm{f}^{d-1}}, \cdots, p_{q_\mathrm{f}^0}\} \rightarrow \{p_{w_\mathrm{t}^d}, \cdots, p_{w_\mathrm{t}^0}, p_{r_{d-1,\mathrm{t}}}, \cdots, p_{r_{0,\mathrm{t}}}\}$$

Hence, $p_{z_\mathrm{t}^d}$ propagates back to probes on at least one output of each WDDL-AND2 gadget, i.e. $w_\mathrm{t}^d, \cdots, w_\mathrm{t}^0$. According to Equation (1), it holds that a probe placed on one output rail of a WDDL-AND2 gadget propagates to one rail of both gadget inputs, i.e.

$$p_{w_\mathrm{t}^d} \rightarrow \{p_{x_\mathrm{t}^d}, p_{y_\mathrm{t}^d}\}, \cdots, p_{w_\mathrm{t}^0} \rightarrow \{p_{x_\mathrm{t}^d}, p_{y_\mathrm{t}^0}\}, \quad p_{z_\mathrm{t}^d} \rightarrow \{p_{x_\mathrm{t}^d}, p_{y_\mathrm{t}^d}, \cdots, p_{y_\mathrm{t}^0}, p_{r_{d-1,\mathrm{t}}}, \cdots, p_{r_{0,\mathrm{t}}}\}.$$

Hence, besides other probing sets, one particular result of the delay-extended probe-propagation of $p_{z_\mathrm{t}^d}$ contains all shares of y. □

Lemma 11. *Consider a d-probing secure and glitch-free SESYM-AND2-noEE gadget which avoids the early evaluation effect. It holds that the SESYM-AND2-noEE gadget does not achieve delay-extended* 1*-probing security.*

Proof. It holds that WDDL-AND2 and WDDL-AND2-noEE calculate their non-complement output rail in the same way. Thus, the probe propagation already shown to prove Lemma 10 is valid here as well. □

We remark that we prove the insecurity of a SESYM-AND2 (resp. WDDL-AND2-noEE) gadget by exemplary setting a probe on the last output share. However, the proof is independent of the share domain, i.e. probing the output of an arbitrary share domain leads to the same results.

4.4.2 Practical Examples

As a practical case study, which will be confirmed experimentally, we investigate a first-order SESYM-AND2 gadget receiving x and y as inputs while the output is denoted by z. The gadget is first-order masked, i.e. $x = x^0 \oplus x^1$, $y = y^0 \oplus y^1$, and $z = z^0 \oplus z^1$, and operates on the dual-rail representation of each intermediate. Hence, each signal corresponds to a tuple of signals.

$$x^0 \rightarrow (x_\mathrm{t}^0, x_\mathrm{f}^0),\ x^1 \rightarrow (x_\mathrm{t}^1, x_\mathrm{f}^1),\ y^0 \rightarrow (y_\mathrm{t}^0, y_\mathrm{f}^0),\ y^1 \rightarrow (y_\mathrm{t}^1, y_\mathrm{f}^1),\ z^0 \rightarrow (z_\mathrm{t}^0, z_\mathrm{f}^0),\ z^1 \rightarrow (z_\mathrm{t}^1, z_\mathrm{f}^1)$$

Here, we focus on the computation of z^0 as shown in Figure 9. We assume that y_t^0 and y_t^1 arrive asynchronously at SESYM-AND2's internal WDDL-AND2 gadgets. In particular, we assume that y_t^0 (resp. y_t^1) arrives with a delay of Δ_0 (resp. Δ_1) at `WDDL-AND2#0` (resp. `WDDL-AND2#1`). Thus, both delays propagate to the same share domain.

Example 9 (SESYM-AND2)**.** Table 1 shows the effect of Δ_0 and Δ_1 on the output delay, i.e. the point in time when the output z^0 reaches a valid dual-rail state. As y_t^0 (resp. y_t^1) is an input of the internal AND2 gate of `WDDL-AND2#0` (resp. `WDDL-AND2#1`), it holds that Δ_0 (resp. Δ_1) only propagates to the output w_t^0 (resp. w_t^1) if $x_\mathrm{t}^0 = 1$ and $y_\mathrm{t}^0 = 1$ (resp. $y_\mathrm{t}^1 = 1$).

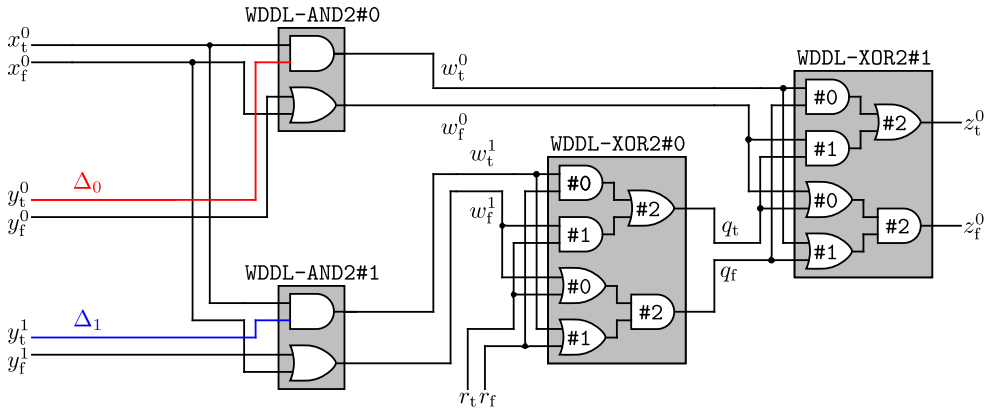


Figure 9: First-order SESYM-AND2 gadget (only first output share), red wire delayed by Δ_0 and blue wire delayed by Δ_1.

Case 1 (Only Δ_0 propagates). If Δ_0 propagates through `WDDL-AND2#0` to w_t^0, w^0 reaches its valid dual-rail state with a delay of Δ_0. Further, `WDDL-XOR2#1` receives the delayed $w_t^0 = 1$ as an input for both rails. Depending on q (output of `WDDL-XOR2#0`), the delay is either propagated to z_t^0 or z_f^0.

Case 2 (Only Δ_1 propagates). If Δ_1 propagates to w_t^1, w^1 reaches its valid dual-rail state with a delay of Δ_1. Further, `WDDL-XOR2#0` receives the delayed $w_t^1 = 1$ as an input for both rails. Similar to Case 1, Δ_1 propagates either to q_t or q_f, depending on r.

Case 3 (Both Δ_0 and Δ_1 propagate). If w_t^0 and w_t^1 are delayed by Δ_0 and Δ_1, q will reach its valid dual-rail state with a delay of Δ_1. Hence, two delays affect the timing of `WDDL-XOR2#1`. In this case, it holds that $w_t^0 = 1$ while q depends on r.

Table 1: Timing dependencies of SESYM-AND2 gadget with and without early evaluation (noEE), y_t^0 delayed by Δ_0 and y_t^1 delayed by Δ_1 (only first output share z^0).

							SESYM-AND2		**SESYM-AND2-noEE**	
y	x	y^1	y^0	x^1	x^0	r	Δ	$\bar{\Delta}_{\Delta_0<\Delta_1}$	Δ	$\bar{\Delta}_{\Delta_0<\Delta_1}$
0	0	0	0	0	0	0/1	0	$\frac{\Delta_1}{4}$	0	$\frac{\Delta_1}{2}$
		0	0	1	1	0/1	0		0	
		1	1	0	0	0/1	0		$\max(\Delta_0, \Delta_1)$	
		1	1	1	1	0/1	$\max(\Delta_0, \Delta_1)$		$\max(\Delta_0, \Delta_1)$	
0	1	0	0	0	1	0/1	0	$\frac{\Delta_1}{4}$	0	$\frac{\Delta_1}{2}$
		0	0	1	0	0/1	0		0	
		1	1	0	1	0/1	$\max(\Delta_0, \Delta_1)$		$\max(\Delta_0, \Delta_1)$	
		1	1	1	0	0/1	0		$\max(\Delta_0, \Delta_1)$	
1	0	0	1	0	0	0/1	0	$\frac{\Delta_0+\Delta_1}{4}$	Δ_0	$\frac{\Delta_0+\Delta_1}{2}$
		0	1	1	1	0/1	Δ_0		Δ_0	
		1	0	0	0	0/1	0		Δ_1	
		1	0	1	1	0/1	Δ_1		Δ_1	
1	1	0	1	0	1	0/1	Δ_0	$\frac{\Delta_0+\Delta_1}{4}$	Δ_0	$\frac{\Delta_0+\Delta_1}{2}$
		0	1	1	0	0/1	0		Δ_0	
		1	0	0	1	0/1	Δ_1		Δ_1	
		1	0	1	0	0/1	0		Δ_1	

In summary, it can be seen in Table 1 that the time-of-evaluation of z^0 depends on the unshared y. In this example, it holds that $\bar{\Delta} = \frac{\Delta_1}{4}$ if $y = 0$ and $\bar{\Delta} = \frac{\Delta_0+\Delta_1}{4}$ if $y = 1$.

noEE. Based on the impressions of Example 9, we exchange all WDDL-AND2 gadgets by an alternative design avoiding the early propagation effect, depictured in Figure 10. Again, we formally analyze the gadget's security under the delay-extended probing model.

Example 10 (SESYM-AND2-noEE)**.** The resulting effect of Δ_0 and Δ_1 on the output delay is also visualized in Table 1. The most important difference compared to Example 9 is that y_t^0 and y_t^1 affect the inverted output rail of `WDDL-AND2#0` and `WDDL-AND2#1`. Consequently, Δ_0 and Δ_1 can propagate to both output rails of w^0 and w^1. For the sake of simplicity, we do not repeat the cases with $w_t^0 = 1$ or $w_t^1 = 1$ as they are already covered in Example 9. In particular, we focus on all cases where the delay propagates to w_f^0 or w_f^1.

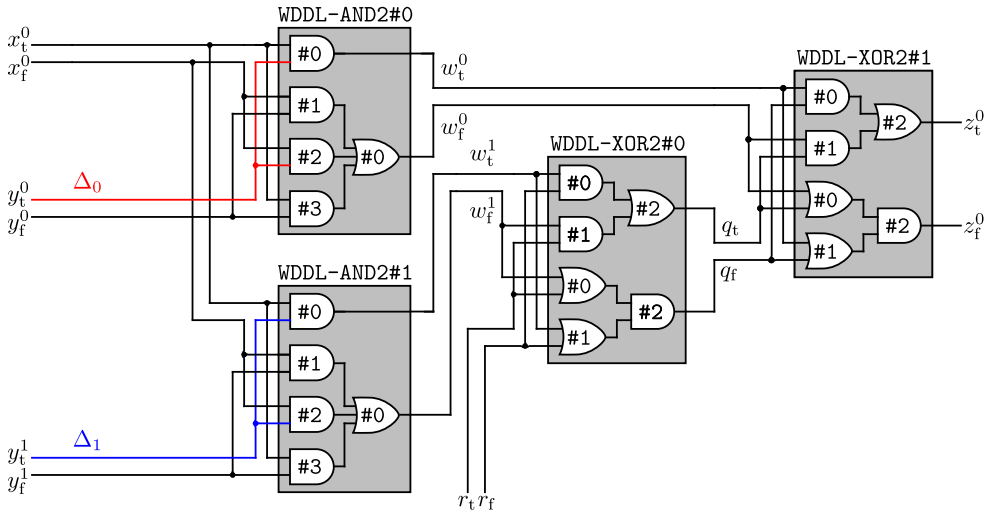


Figure 10: First-order SESYM-AND2-noEE gadget (only first output share), red wire delayed by Δ_0 and blue wire delayed by Δ_1.

Case 1 (Only Δ_0 propagates). We assume that Δ_0 propagates to w_f^0, i.e. $x_f^0 = 1$ and $y_t^0 = 1$ while `AND#2` of `WDDL-AND2#0` evaluates to 1. Next, w_f^0 is forwarded to both rails of `WDDL-XOR2#1`. Further, `WDDL-XOR2#1` propagates Δ_0 either to z_t^0 or z_f^0. If $q = 1$, `AND#1` evaluates to 1 with a delay of Δ_0 which is further propagated to z_t^0. If $q = 0$, `OR#0` and `OR#1` evaluate to 1 but `OR#0` with a delay of Δ_0. Hence, `AND#2` must wait for both signals before evaluating to 1 what propagates Δ_0 to z_f^0.

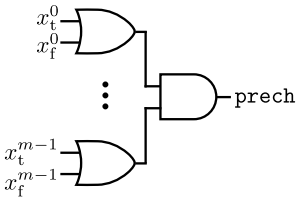
Case 2 (Only Δ_1 propagates). We assume that Δ_1 propagates to w_f^1, i.e. $x_f^0 = 1$ and $y_t^1 = 1$ similar to `WDDL-AND2#0` in Case 1. Moreover, `WDDL-XOR2#0` behaves similar to `WDDL-XOR1#2` in Case 1 and propagates Δ_1 to either q_t or q_f depending on r. Finally, `WDDL-XOR2#1` behaves similar to `WDDL-XOR2#1` in Case 2 of Example 9. Hence, Δ_1 propagates to either z_t^0 or z_f^0.

Case 3 (Both Δ_0 and Δ_1 propagate). Δ_0 and Δ_1 propagate to w_f^0 and w_f^1 only if $x_f^0 = 1$, $y_t^0 = 1$, and $y_t^1 = 1$. As we know from Case 1 and Case 2, both delays are propagated to the final output. Hence, the longer delay must be considered.

We summarize the final delays in Table 1 and observe that the time-of-evaluation of z^0 still depends on the unshared y.

4.5 SESYM Completion Detector

In a synchronous setting, the two consecutive phases of DRP logic are usually driven by control signals generated with some link to the clock signal. SESYM merges the pre-charge and evaluation phases asynchronously into a single clock cycle by controlling the pre-charge logic with an asynchronous `prech` signal. The control signal generation is implemented by a product-of-sums (see Figure 11(a)) to ensure that every dual-rail output is evaluated before the next pre-charge phase.



(a) Generic SESYM completion detector

x_t^0	x_t^1	x_f^0	x_f^1	$\Delta_{\texttt{prech}}$	
0	0	1	1	$\Delta_{x_f^1}$	
1	0	0	1	$\Delta_{x_f^1}$	
0	1	1	0	$\Delta_{x_f^0}$	$\Rightarrow x = 1$
1	1	0	0	$\Delta_{x_t^{0/1}}$	$\Rightarrow x = 0$

(b) Timing dependencies of $\Delta_{\texttt{prech}}$, completion detector with two shares, assuming $\Delta_{x_t^{0/1}} < \Delta_{x_f^0} < \Delta_{x_f^1}$

Figure 11: SESYM completion detector's functionality.

We review the so-called completion detector as a security-critical design component, because the value of `prech` and its delay $\Delta_{\texttt{prech}}$ should depend on all masked circuit outputs. This includes all shares per sensitive variable, basically all secret data, where signal delays cross share boundaries. Thus, the combinational conjunction into a single signal can potentially leak information. The example in Figure 11(b) shows how $\Delta_{\texttt{prech}}$ can be utilized as a distinguisher for a secret in some cases under a specific assumption.

Lemma 12. *Consider a d-order SESYM completion detector which is glitch-free. It holds that the SESYM completion detector does not achieve delay-extended* 1*-probing security.*

Proof. Consider a delay-extended probe $p_{\texttt{prech}}$ on `prech`. This probe propagates to all inputs of the underlying AND gate (see Figure 11(a)). It holds that one possible probing set becomes $p_{\texttt{prech}} \rightarrow \{x_t^0, \cdots, x_t^{m-1}\}$ which records all shares of x. Hence, the insecurity is confirmed. □

4.6 Non-linear DOM Gadget with Self-Timed Masking

The authors of [SBB+22] attempted to embed asynchronous logic in a globally synchronized and round-based design, i.e. a loop evaluating multiple times in a single clock cycle. Therefore, the synchronous to asynchronous (`S2A`) module converts the single-rail inputs to DRP and makes use of a multiplexer to forward smaller chunks of the state to the asynchronous logic alternating with NULL. Contrary, the asynchronous to synchronous (`A2S`) module converts the output chunk back to single-rail logic, and restructures the chunks back to a valid state. Based on the design choices in [SBB+22], we consider the last share domain of a self-timed d-order masked AND2 gadget shown in Figure 12. Our design includes a 3-stage pipeline with weakly-indicating latches $\texttt{L}_0$, $\texttt{L}_1$, and $\texttt{L}_2$, i.e. the control signals req_i and ack_i depend on the first input share of the corresponding latch.

Here, we assume that the `S2A` module forwards x and y as inputs to the asynchronous logic while the `A2S` module receives an output denoted by z. The whole circuit is d-order masked, i.e. $x = x^0 \oplus \cdots \oplus x^d$, $y = y^0 \oplus \cdots \oplus y^d$, and $z = z^0 \oplus \cdots \oplus z^d$ while the asynchronous part operates on the dual-rail representation of each intermediate. Hence, each signal corresponds to a tuple of signals.

$$x^i \rightarrow (x_t^i, x_f^i), \qquad y^i \rightarrow (y_t^i, y_f^i), \qquad z^i \rightarrow (z_t^i, z_f^i)$$

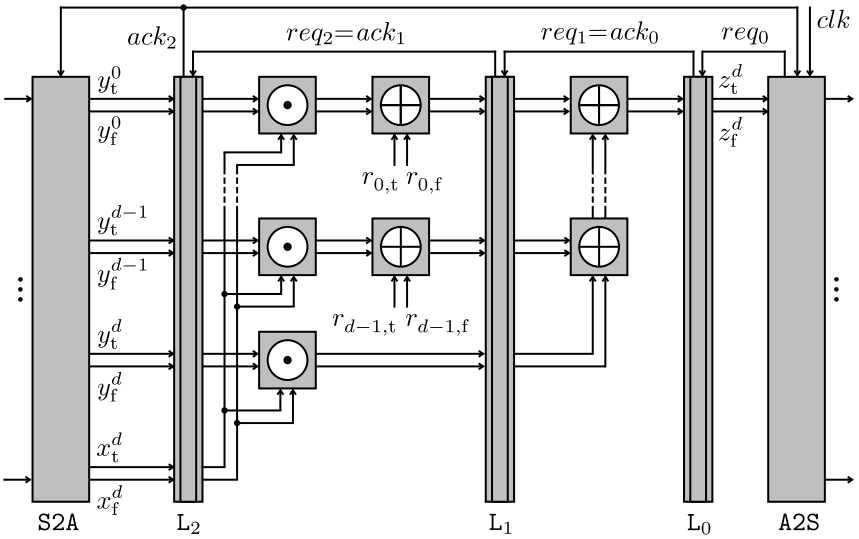


Figure 12: Last share domain of a self-timed masked AND2 gadget with asynchronous 3-stage pipeline.

4.6.1 Probing Security

As for SESYM, we prove the following lemma.

Lemma 13. *Consider a d-probing secure and glitch-free self-timed masked AND2 gadget. It holds that the self-timed masked AND2 gadget does not achieve delay-extended 1-probing security.*

Proof. Consider one delay-extended probe $p_{z_\mathrm{t}^d}$ on z_t^d. In contrast to SESYM, asynchronous latches – made by C-elements as shown in Figure 5 – synchronize the intermediates. However, a delay-extended probe on a C-element output propagates to all inputs of the C-element (see Figure 1 of [SBB+22]). Hence, it holds that a delay-extended probe on one output of the latch propagates to the respective control and data input of the latch. If a delay-extended probe propagates through a latch, the resulting probing set is the same as for SESYM[1]. □

Again, we prove the gadget by exemplary setting a probe on the last output share. However, the proof is independent of the share domain, i.e. probing the output of an arbitrary share domain leads to the same result.

4.6.2 Practical Examples

The proof above points out that the asynchronous latches do not synchronize the data for all possible delays occurring at the latch inputs. We recap that a C-element forwards a new output only if both inputs, i.e. a data signal and a control signal, are equal and keeps its previous state otherwise. The control signals req_0, req_1, and req_2 imitate enable signals of the respective latches $\mathtt{L}_0$, $\mathtt{L}_1$, and $\mathtt{L}_2$. For example, $\mathtt{L}_0$ forwards only the NULL state if $req_0 = 0$, and a valid dual-rail state if $req_0 = 1$. Now, we assume that `S2A` pre-charges the data inputs of $\mathtt{L}_2$, i.e. x_t^d, x_f^d and $y_\mathrm{t}^0, y_\mathrm{f}^0, \cdots, y_\mathrm{t}^d, y_\mathrm{f}^d$ while $\mathtt{L}_2$ itself still keeps a valid dual-rail state (similar to Figure 7 of [SBB+22]). Therefore, it holds that $req_2 = 1$. As the circuit operates in an alternating manner, $\mathtt{L}_1$ stores NULL, and $\mathtt{L}_0$ stores another valid

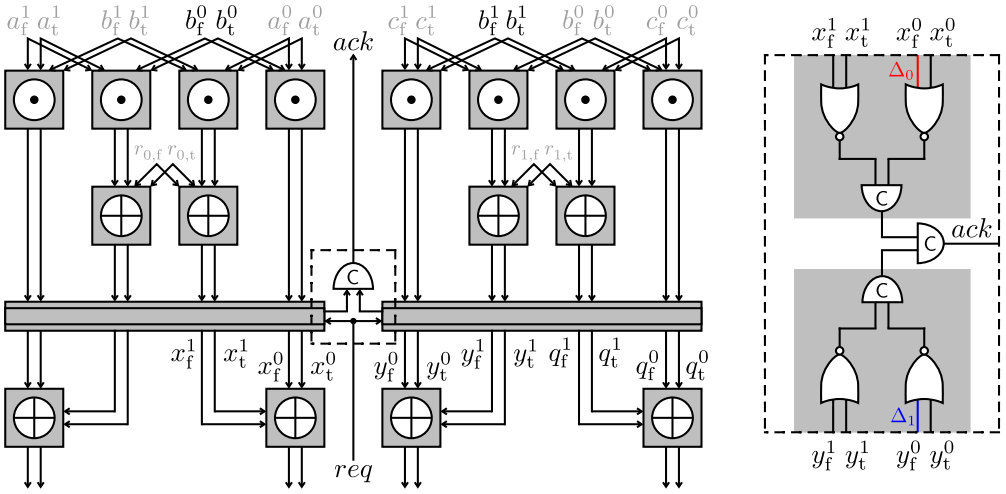
[1]Self-Timed Masking uses basic gadgets which may differ slightly from the described WDDL gadgets. However, the probe-propagation rules used for the proof also apply to these gadgets.

dual-rail state ($req_1 = 0$ and $req_0 = 1$). To reach the next state (similar to Figure 8 of [SBB+22]) asynchronously, ack_2 toggles from 1 to 0 as soon as $\mathtt{L}_2$ stores a valid dual-rail state. As soon as $ack_2 = 0$, the toggle of req_0 from 1 to 0 is also initiated and enables $\mathtt{L}_0$ to store NULL. As $\mathtt{L}_0$ changes from a valid dual-rail state to NULL, req_1 toggles from 0 to 1 to enable the storage of a valid dual-rail state in $\mathtt{L}_1$. The toggle of control signals propagates back until $\mathtt{L}_2$ triggers the next toggle of ack_2. However, the latches only synchronize the data if the output of the combinational circuit arrives at the latch before the corresponding req signal. Otherwise, if the new req signal arrives before the new state, it can pass the latch without any synchronization.

For example, consider the combinational circuit between $\mathtt{L}_2$ and $\mathtt{L}_1$, i.e. the non-linear operation and re-sharing stage of the multiplier. If we assume that particular output wires of $\mathtt{L}_2$ are delayed, e.g. Example 10, the delays will propagate through $\mathtt{L}_1$ and through the compression stage, if req_1 arrives at $\mathtt{L}_1$ before the combinational circuit finishes its computations. Consequently, the same security flaw as for SESYM shown in Example 10 and Table 1 can occur. To fix this issue, the designer must guarantee that all combinational parts are already done before the next req signal arrives at their respective latch. If this holds, the delays of the control signals become independent of the processed data which indicates delay-extended probing security. However, it will probably require manual adjustments.

4.7 Non-linear Self-Timed Masking Completion Detector

Weakly-indicating latches as shown in Figure 5 process only one share of a sensitive variable to achieve an evaluation time which is independent of the unshared variable. Therefore, we would assume that a weakly-indicating latch with completion detector achieves delay-extended probing security. However, according to Figure 10 of [SBB+22], the authors exhibited a Canright S-box implementation, which uses a common input variable for two parallel DOM multipliers. Moreover, a C-element receives the acknowledge signal of both parallel latches and combines them to forward a new req signal to the previous pipeline stage. We argue that combining multiple acknowledge signals without special care may lead to leakage even if the latches synchronize the data. For example, consider two parallel self-timed masked AND2 gadgets as shown in Figure 13.



(a) Parallel gadgets with merged acknowledge signals.

(b) Acknowledge.

Figure 13: Parallel first-order self-timed masked AND2 gadgets.

The first gadget computes $ab = x$ while the second gadget computes $bc = y$. Both gadgets are first order masked, i.e. $a = a^0 \oplus a^1$, $b = b^0 \oplus b^1$, $c = c^0 \oplus c^1$, $x = x^0 \oplus x^1$, and $y = y^0 \oplus y^1$, and operate on the dual-rail representation of each intermediate, e.g. $a^0 \to (a^0_\mathrm{t}, a^0_\mathrm{f})$. Further, both gadgets receive a common input $b^0_\mathrm{t}, b^0_\mathrm{f}, b^1_\mathrm{t}, b^1_\mathrm{f}$. The completion detector is built as indicated in Figure 9 of [SBB+22], i.e. processes two dual-rail outputs of the latch $(x^0_\mathrm{t}, x^0_\mathrm{f}, x^1_\mathrm{t}, x^1_\mathrm{f})$ and $(y^0_\mathrm{t}, y^0_\mathrm{f}, y^1_\mathrm{t}, y^1_\mathrm{f})$ and combines them to a single *ack* signal. According to our model, a probe p_{ack} would now record both shares of b making this particular completion detector insecure. As a practical example, we assume that x^0_f is delayed by Δ_0 and y^0_f is delayed by Δ_1 compared to the other inputs of the completion detector. Then, it holds that one of the delays propagates to *ack* if $x^0_\mathrm{t} = 0$ or $y^0_\mathrm{t} = 0$ while *ack* is not delayed if $x^0_\mathrm{t} = 1$ and $y^0_\mathrm{t} = 1$. As $x^0_\mathrm{t} = a^0_\mathrm{t} b^0_\mathrm{t}$ and $y^0_\mathrm{t} = b^1_\mathrm{t} c^1_\mathrm{t}$, a non-delayed *ack* signal gives information about the unshared b, namely $b = 1$. In summary, improper placement and routing can lead to delays similar to the ones we provoked for our formal analysis of SESYM in Section 4.4, but on the completion detector inputs instead of the multiplier inputs. It is important to mention that the leakage on *ack* is independent of the *req* signal. More concretely, this issue occurs even if the latches achieve a data-independent synchronization. To prevent this issue, the designer should pay attention to how the shares of a variable processed by several parallel instances are given to these instances. For example, following Figure 13, if the completion detector processes $(q^0_\mathrm{f}, q^0_\mathrm{t}, q^1_\mathrm{f}, q^1_\mathrm{t})$ instead of $(y^0_\mathrm{f}, y^0_\mathrm{t}, y^1_\mathrm{f}, y^1_\mathrm{t})$ only $(b^0_\mathrm{f}, b^0_\mathrm{t})$ contribute to the non-reshared inputs of the completion detector. Hence, a probe on *ack* gives no further information on $(b^1_\mathrm{f}, b^1_\mathrm{t})$ as they only contribute to the re-shared inputs of the completion detector. Another solution is to only take the re-shared synchronized value for the completion detector, which guarantees data-independent delay propagation. For example, if the completion detector would only process the synchronized values $(x^1_\mathrm{f}, x^1_\mathrm{t})$ (blinded by r_0) and $(y^1_\mathrm{f}, y^1_\mathrm{t})$ (blinded by r_1), the adversary cannot get any information by probing *ack*.

5 Case Studies

In order to experimentally verify our findings, we conducted a rich set of FPGA-based analyses. After explaining the specifications of the underling setup, we provide detailed expression of each experiment individually, as follows.

5.1 Setup

Our entire analyses have been conducted on the Xilinx Kintex-7 FPGA of an SCA-evaluation board SAKURA-X [SAK]. We made use of a digital sampling oscilloscope to monitor the voltage drop over a 1 Ω shunt resistor placed on the Vdd path of the underlying Kintex-7 FPGA, which is proportional to the current passing through the FPGA, hence a fraction of its instantaneous power consumption. We collected such traces at a sampling rate of 500 MS/s while the FPGA was being supplied by a stable oscillator at a frequency of 3 MHz, hence minimizing the effect of adjacent clock cycles on each other via power traces.

As the analysis scheme, we followed the state of the art, and collected traces suitable for fixed-vs-random first-order leakage assessment t-test [SM15] to examine the existence of first-order leakages without conducting any attack. Since all our designs under test are first-order masked, i.e., with two shares, for every measurement the fixed or random input is given to the Kintex-7 FPGA in an independently and freshly masked form. When required, the fresh randomness (sometimes called online masks in various literature) are generated inside the Kintex-7 FPGA. To this end, we instantiated an individual 31-bit Linear Feedback Shift Register (LFSR) for each required fresh mask bit, which is seeded randomly at the power-up cycle of the Kintex-7 FPGA. This choice is also based on the

state of the art, particularly the highly efficient FPGA-specific implementation of such an LFSR [MMW18].

5.2 Keccak χ Function

As the first experiment, we constructed the Keccak 5-bit χ S-box using SESYM multipliers. In short, it consists of five instances of such a multiplier and five masked XORs in DRP logic. The required inverters are just realized by swapping the complementary rails of one of the shares. Naturally, this design requires five fresh mask bits, each of which is individually associated to a SESYM multiplier.

The original SESYM multiplier design has been made for ASIC platforms. In order to be as close as possible to the original construction, we made sure that every single gate is realized by a dedicated Look-Up Table (LUT). We further turned off all optimization features and kept the hierarchy of the design to avoid the synthesizer merging multiple gates into a LUT.

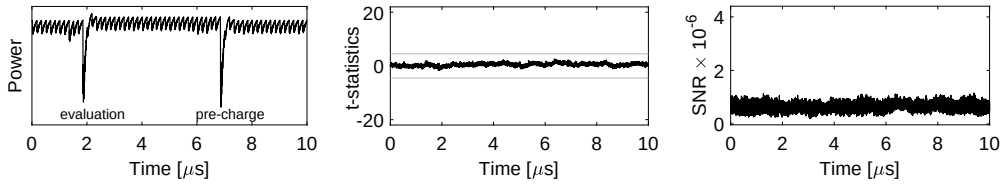
Since such a tiny design is very small for the underlying Kintex-7 FPGA, in order to increase the visibility of the power peaks we instantiated 25 copies of such a SESYM χ S-box[2], and conducted the aforementioned experiment, i.e., measurements suitable for fixed-vs-random t-test. We controlled the evaluation and pre-charge phases and intentionally made a long delay between such phase changes to identify them in the power traces, as can also be seen in the mean trace shown in Figure 14(a). The corresponding t-test result using 100 million traces is shown in Figure 14(b), implying no detectable first-order leakage. We further used half of the collected traces (which belong to the group with random input) to estimate the Signal-to-Noise Ratio (SNR) [MOP07] based on the 5-bit unmasked input of the χ S-box. The corresponding SNR curve is shwon in Figure 14(c), also indicating not a visible dependency. We would like to highlight that the underlying design does not contain any cascaded SESYM multipliers, and has a very low power consumption.

In order to examine our claims given in Section 4.4, i.e. the effect of delay imbalances, we manually added delay elements (realized by cascaded inverters) to y_t^0 and y_t^1 of each WDDL-AND2 gadget of every SESYM multiplier (see the description and notation in Section 4.4). To this end, we instantiated 6 inverters[3] in signal y_t^0 and 10 inverters in signal y_t^1. Repeating the same experiment led to the results shown in Figure 15, clearly showing the first-order leakage of the design as well as a visible data-dependency via SNR. We should highlight that if the design is provably secure, its security should not depend on the delay of its internal signals, or let say on the placement and routing. This can be seen as a counterexample for the claim of the authors in [NGPM22] that imbalanced dual rails do not have any effect on the security of SESYM circuits.

As stated above, we did our best to emulate the ASIC-based designs on the underlying FPGA. As given in Section 2, WDDL-AND2 gadgets have the early evaluation issue, which is also the case in the experiments explained above. The noEE version can mitigate this issue on FPGAs by instantiating one 5-to-2 LUT for the entire operation of a WDDL-AND2 gadget (see Section 2.3 and [MI14]). Hence, we repeated the last experiment by replacing every WDDL-AND2 gadget with WDDL-AND2-noEE while keeping the intentional imbalances between the aforementioned rails. As shown in Figure 16, avoiding the early evaluation does not have much effects on the detectable first-order leakage. This confirms our claims and statements given in Section 4, that even in case of no early propagation, the imbalances between dual rails would falsify the SESYM's security arguments.

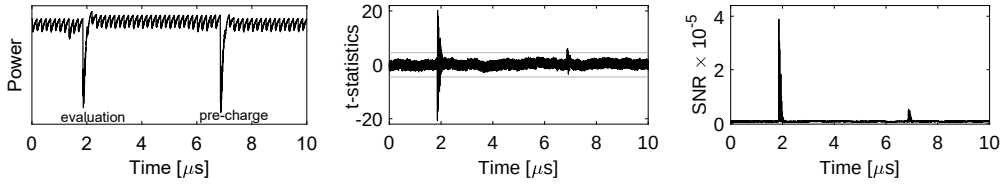
[2]This does not pose any security issues. Multiple parallel-working instances of a secure circuit would not falsify any security argument.

[3]Each of these inverters is realized by a single LUT in FPGAs.



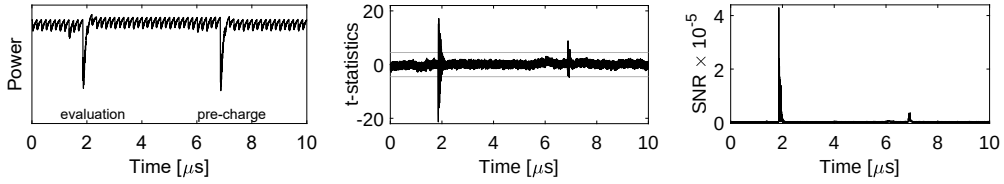
(a) Mean trace, over 100k traces (b) t-test, over 100 million traces (c) SNR, over 50 million traces

Figure 14: SESYM Keccak χ, uncontrolled placed and routed.



(a) Mean trace, over 100k traces (b) t-test, over 100 million traces (c) SNR, over 50 million traces

Figure 15: SESYM Keccak χ, manually delayed internal signals.



(a) Mean trace, over 100k traces (b) t-test, over 100 million traces (c) SNR, over 50 million traces

Figure 16: SESYM Keccak χ, without early evaluation (noEE), manually delayed internal signals.

5.3 AES S-box

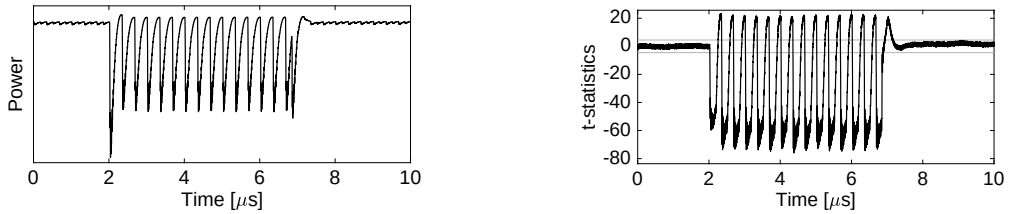
As given above, the Keccak χ function contains just a few non-cascaded SESYM multipliers, with very little power consumption. In order to examine a more complex case, we also considered the AES S-box in our experiments. To this end, we focused on the SESYM designs that the authors have provided on a public repository on GitHub[4].

We first took the ASIC-based design, and realized the circuit similar to that of Keccak χ function, as explained in Section 5.2. The architecture of the underlying S-box is known as Boyar-Peralta design [BP12] which consists of 34 AND2 instances and some XOR gates. The authors of [NGPM22] have replaced the gates with their SESYM counterpart resulting in a SESYM circuit requiring 34 fresh mask bits (for the first-order).

We instantiated 8 copies of such an AES S-box design and performed the aforementioned experiments. The design further contains a completion detector (see Section 4.5), which automatically switches between evaluation and pre-charge phases. Note that we just made sure that every gate is realized by a dedicated LUT and kept the hierarchy of the design. We did not add any delay elements to any signal, and have not controlled the placement and routing. As shown in Figure 17(a), we kept the S-box operating for a couple of clock cycles. The t-test results presented in Figure 17(b) imply a highly detectable first-order leakage with less than 1 million traces.

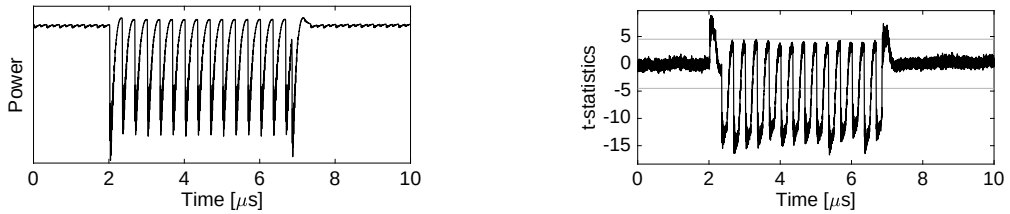
As the second design to examine, we replaced the WDDL-AND2 and WDDL-XOR2 gadgets with their LUT-based variants without early evaluation. As stated, each of these

[4] https://extgit.iaik.tugraz.at/sesys/self-synchronized-masking



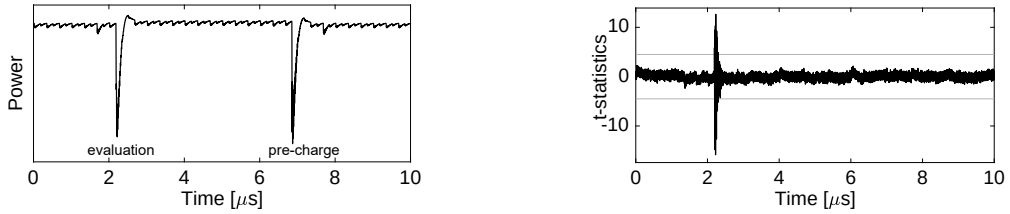
(a) Mean trace, over 100k traces (b) t-test, over 1 million traces

Figure 17: SESYM AES S-box, original ASIC design.



(a) Mean trace, over 100k traces (b) t-test, over 1 million traces

Figure 18: SESYM AES S-box, ASIC design without early evaluation.



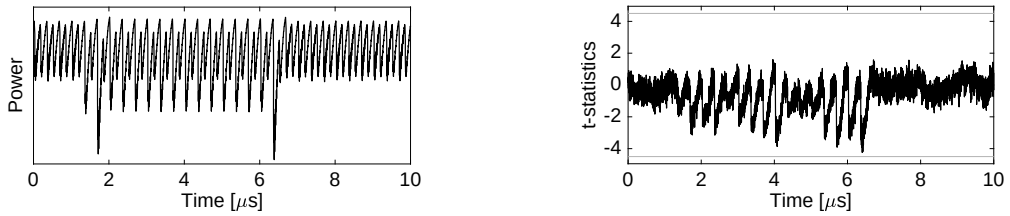
(a) Mean trace, over 100k traces (b) t-test, over 100 million traces

Figure 19: SESYM AES S-box, original FPGA design.

gates is realized by a single 5-to-2 LUT. Note that the other gates used in the pre-charge units and completion detector stay unchanged, i.e., one individual LUT for each gate. This leads to less FPGA resource utilization and hence less power consumption, which turns into a slight decrease in amount of detectable first-order leakage as shown in Figure 18.

After seeing that the original ASIC designs exhibit strong first-order leakage (confirming our findings), we further examined the FPGA-based designs that the authors also provided on GitHub[4]. We first noticed that the FPGA designs are entirely different to the ASIC ones. For example, the WDDL gadgets are realized by their LUT-based version without early evaluation and each LUT is also controlled by a global pre-charge signal, which forces all WDDL gadgets to move to the pre-charge phase simultaneously. Further, there is no completion detector module, and the switch between pre-charge and evaluation phases is controlled by a Finite State Machine (FSM) just based on the AES rounds and its required operations.

Similar to the other experiments, we took the AES S-box design, which is based on the architecture known as Canright S-box [Can05], and requires 18 fresh mask bits. As stated above, such a SESYM S-box design is controlled by a pre-charge signal. Therefore, similar to the former experiments on Keccak χ function, we made a relatively long delay between activation of evaluation and pre-charge phases as shown in Figure 19(a). Although the design exhibits less leakage compared to the ASIC-based designs, its first-order leakage using 100 million traces is still clearly observable as shown in Figure 19(b). Interestingly, no leakage is detected at the start of the pre-charge phase. This is due to the aforementioned



(a) Mean trace, over 100k traces

(b) t-test, 100 million traces

Figure 20: LMDPL AES S-box.

fact that all WDDL gadgets go to the pre-charge phase by a global signal simultaneously, while the start of the evaluation phase of every WDDL gadget depends on the arrival time of its inputs, causing the leakage in SESYM circuits.

5.3.1 Discussions

An important question is why the authors of [NGPM22] have not observed any first-order leakage in their experiments. First, the authors have conducted their experiments only on their FPGA-based designs which are – based on the source codes available on GitHub[4] – not fully equivalent to their ASIC-based designs. Second, their ASCON design – similar to our Keccak χ function – contains just a few non-cascaded SESYM multipliers with very low power consumption. As shown in Figure 14, the leakage of such a small circuit may not be easily detectable. Third, the FPGA-based AES design available on GitHub[4] is a round-based design, i.e., 16 S-boxes in parallel excluding the key schedule, etc. If the authors have taken the same design in their experimental evaluations, the first-order leakage might be harder to detect compared to that in our experiments. Forth, the quality of the employed measurement setups might be different.

5.4 LMDPL

As stated in Section 4.3, the LMDPL gadgets are secure under the delay-extended probing model. For the sake of completeness as well as sanity check, we also conducted a similar experimental evaluation on an LMDPL circuit. To this end, in order to do a fair analysis, we implemented 8 instances of the Boyar-Peralta AES S-box using LMDPL gadgets, and – similar to the SESYM ASIC-based AES S-box design – kept the S-box active for a couple of clock cycles (see Figure 20(a)). The corresponding analysis results using 100 million traces are shown in Figure 20(b), which are consistent with our statements as well as with the results the authors reported in [SBHM20], i.e., no detectable first-order leakage.

6 Conclusions

In short, we made it clear that, in addition to glitches, signal delays such as routing imbalances can violate the security of masked hardware designs. We used many examples built on each other to show how imbalanced delays are propagated by a combinational circuit and thus reveal sensitive values. Based on this insight, we created a model allowing formal verification of glith-free masked hardware designs with imbalanced delays. The underlying delay extension of our model naturally extends the robust probing model with the ability to cover an unconsidered physical default. Moreover, we investigated the security of LMDPL and SESYM in theory and practice revealing security flaws in SESYM. Also, we extended our work with the investigations on Self-Timed Masking, which follows the DOM principles, but has similar issues as SESYM in certain corner cases, and generally

suffers from a rather slow handshaking mechanism, which may avoid it to be considered as a low-latency solution.

Finally, we would like to clarify some points. By addressing imbalanced delays as a potential security flaw, we made use of knowledge known since more than 15 years in the context of WDDL [SS06]. However, as masking schemes like TI keep their security guarantees for all possible occurrences of glitches, they implicitly cover imbalanced delays as well. Nevertheless, imbalanced delays become interesting again, in case there are no glitches inside the circuit, and must be considered separately. The authors of LMDPL incorporated delays as part of their theoretical discussion, what the authors of SESYM did not. The authors of SESYM formally verified the (glitch-extended) probing security of their approach through COCO [GHP+21]. We confirm that (1) SESYM is indeed probing secure, (2) SESYM circuits are glitch-free, and (3) that COCO reports its security if the probes are not propagated due to glitches. However, the probes are propagated due to imbalanced delays, which COCO or other formal verification tools do not cover. Hence, integrating the delay-extended probing security model in a formal verification tool would be helpful to formally verify the security of DRP-based masking schemes. Integration of this concept in PROLEAD [MM22] seems straightforward, which is among our plans in the future.

As our delay-extended probing model covers all possible delays, it is probably over conservative. However, this also holds for glitches and transitions in the original robust probing model, but it is the only way to guarantee the security of *arbitrary* circuits, especially independent of placement and routing. A decisive question is whether such a conservative delay-extended model leaves room for fully asynchronous circuits at all, i.e. circuits without any clock-driven synchronization. To achieve delay-extended probing security in asynchronous circuits, we remark that every gate (or let say every fundamental operating unit) must achieve a data-independent time-of-evaluation and that delay of dual rails must be balanced. As a side note, Self-Timed Masking can achieve practical security as long as its practical instantiation avoids particular delays. Generally, the same can be done for glitch-extended security as well. If a design guarantees the avoidance of particular glitches, it can achieve practical security without being glitch-extended secure. Unfortunately, manual placement and routing are challenging, time-consuming, and error-prone tasks. It is true that LMDPL circuits are experimentally and provably secure under the delay-extended probing model, but LMDPL requires a single register stage to achieve this. Although it is definitely a low-latency approach, it cannot be considered as a fully asynchronous self-timed solution, hence no need to pay attention to placement, routing and delay imbalances.

Acknowledgments

The work described in this paper has been supported in part by the German Research Foundation (DFG) under Germany's Excellence Strategy - EXC 2092 CASA - 390781972, and through the projects 435264177 (SAUBER) and 456967092 (SecFShare).

References

[AG01] Mehdi-Laurent Akkar and Christophe Giraud. An Implementation of DES and AES, Secure against Some Attacks. In *CHES*, volume 2162 of *LNCS*, pages 309–318. Springer, 2001.

[BBD+16] Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, Pierre-Yves Strub, and Rébecca Zucchini. Strong Non-

Interference and Type-Directed Higher-Order Masking. In *CCS*, pages 116–129. ACM, 2016.

[BGF+10] Shivam Bhasin, Sylvain Guilley, Florent Flament, Nidhal Selmane, and Jean-Luc Danger. Countering early evaluation: an approach towards robust dual-rail precharge logic. In *WESS*, page 6. ACM, 2010.

[BP12] Joan Boyar and René Peralta. A Small Depth-16 Circuit for the AES S-Box. In *SEC*, volume 376 of *IFIP*, pages 287–298. Springer, 2012.

[Can05] David Canright. A Very Compact S-Box for AES. In *CHES*, volume 3659 of *LNCS*, pages 441–455. Springer, 2005.

[CBG+17] Thomas De Cnudde, Begül Bilgin, Benedikt Gierlichs, Ventzislav Nikov, Svetla Nikova, and Vincent Rijmen. Does Coupling Affect the Security of Masked Implementations? In *COSADE*, volume 10348 of *LNCS*, pages 1–18. Springer, 2017.

[CGLS21] Gaëtan Cassiers, Benjamin Grégoire, Itamar Levi, and François-Xavier Standaert. Hardware Private Circuits: From Trivial Composition to Full Verification. *IEEE Trans. Comp.*, 70(10):1677–1690, 2021.

[CJRR99] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards Sound Approaches to Counteract Power-Analysis Attacks. In *CRYPTO*, volume 1666 of *LNCS*, pages 398–412. Springer, 1999.

[CS20] Gaëtan Cassiers and François-Xavier Standaert. Trivially and Efficiently Composing Masked Gadgets With Probe Isolating Non-Interference. *IEEE TIFS*, 15:2542–2555, 2020.

[FGP+18] Sebastian Faust, Vincent Grosso, Santos Merino Del Pozo, Clara Paglialonga, and François-Xavier Standaert. Composable Masking Schemes in the Presence of Physical Defaults & the Robust Probing Model. *IACR TCHES*, 2018(3):89–120, 2018.

[GHP+21] Barbara Gigerl, Vedad Hadzic, Robert Primas, Stefan Mangard, and Roderick Bloem. Coco: Co-Design and Co-Verification of Masked Software Implementations on CPUs. In *USENIX*, pages 1469–1468. USENIX Association, 2021.

[GMK16] Hannes Groß, Stefan Mangard, and Thomas Korak. Domain-Oriented Masking: Compact Masked Hardware Implementations with Arbitrary Protection Order. In *TIS @ CCS*, page 3. ACM, 2016.

[GMO01] Karine Gandolfi, Christophe Mourtel, and Francis Olivier. Electromagnetic Analysis: Concrete Results. In *CHES*, volume 2162 of *LNCS*, pages 251–261. Springer, 2001.

[ISW03] Yuval Ishai, Amit Sahai, and David A. Wagner. Private Circuits: Securing Hardware against Probing Attacks. In *CRYPTO*, volume 2729 of *LNCS*, pages 463–481. Springer, 2003.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In *CRYPTO*, volume 1666 of *LNCS*, pages 388–397. Springer, 1999.

[KMMS22] David Knichel, Amir Moradi, Nicolai Müller, and Pascal Sasdrich. Automated Generation of Masked Hardware. *IACR TCHES*, 2022(1):589–629, 2022.

[Koc96] Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *CRYPTO*, volume 1109 of *LNCS*, pages 104–113. Springer, 1996.

[LMW14] Andrew J. Leiserson, Mark E. Marson, and Megan A. Wachs. Gate-level masking under a path-based leakage metric. In *CHES*, volume 8731 of *LNCS*, pages 580–597. Springer, 2014.

[MI14] Amir Moradi and Vincent Immler. Early propagation and imbalanced routing, how to diminish in fpgas. In *CHES*, volume 8731 of *LNCS*, pages 598–615. Springer, 2014.

[MM22] Nicolai Müller and Amir Moradi. PROLEAD A Probing-Based Hardware Leakage Detection Tool. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(4):311–348, 2022.

[MMW18] Lauren De Meyer, Amir Moradi, and Felix Wegener. Spin Me Right Round Rotational Symmetry for FPGA-Specific AES. *IACR TCHES*, 2018(3):596–626, 2018.

[MOP07] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power analysis attacks - revealing the secrets of smart cards*. Springer, 2007.

[MPG05] Stefan Mangard, Thomas Popp, and Berndt M. Gammel. Side-Channel Leakage of Masked CMOS Gates. In *CT-RSA*, volume 3376 of *LNCS*, pages 351–365. Springer, 2005.

[MPO05] Stefan Mangard, Norbert Pramstaller, and Elisabeth Oswald. Successfully Attacking Masked AES Hardware Implementations. In *CHES*, volume 3659 of *LNCS*, pages 157–171. Springer, 2005.

[NGPM22] Rishub Nagpal, Barbara Gigerl, Robert Primas, and Stefan Mangard. Riding the waves towards generic single-cycle masking in hardware. *IACR TCHES*, 2022(4):693–717, 2022.

[NRR06] Svetla Nikova, Christian Rechberger, and Vincent Rijmen. Threshold Implementations Against Side-Channel Attacks and Glitches. In *ICICS*, volume 4307 of *LNCS*, pages 529–545. Springer, 2006.

[OMPR05] Elisabeth Oswald, Stefan Mangard, Norbert Pramstaller, and Vincent Rijmen. A Side-Channel Analysis Resistant Description of the AES S-Box. In *FSE*, volume 3557 of *LNCS*, pages 413–423. Springer, 2005.

[SAK] SAKURA. Side-channel Attack User Reference Architecture. http://satoh.cs.uec.ac.jp/SAKURA/index.html.

[SBB+22] Mateus Simoes, Lilian Bossuet, Nicolas Bruneau, Vincent Grosso, and Thomas Sarno Patrick Haddad. Self-Timed Masking: Implementing Masked S-Boxes Without Registers. In *CARDIS 2022*, Lecture Notes in Computer Science. Springer, 2022.

[SBHM20] Pascal Sasdrich, Begül Bilgin, Michael Hutter, and Mark E. Marson. Low-latency hardware masking with application to AES. *IACR TCHES*, 2020(2):300–326, 2020.

[Sha79] Adi Shamir. How to Share a Secret. *Commun. ACM*, 22(11):612–613, 1979.

[SM15] Tobias Schneider and Amir Moradi. Leakage Assessment Methodology - A Clear Roadmap for Side-Channel Evaluations. In *CHES*, volume 9293 of *LNCS*, pages 495–513. Springer, 2015.

[SS06] Daisuke Suzuki and Minoru Saeki. Security Evaluation of DPA Countermeasures Using Dual-Rail Pre-charge Logic Style. In *CHES*, volume 4249 of *LNCS*, pages 255–269. Springer, 2006.

[TV04a] Kris Tiri and Ingrid Verbauwhede. A logic level design methodology for a secure DPA resistant ASIC or FPGA implementation. In *(DATE*, pages 246–251. IEEE Computer Society, 2004.

[TV04b] Kris Tiri and Ingrid Verbauwhede. Place and route for secure standard cell design. In Jean-Jacques Quisquater, Pierre Paradinas, Yves Deswarte, and Anas Abou El Kalam, editors, *Smart Card Research and Advanced Applications VI*, pages 143–158, Boston, MA, 2004. Springer US.