GliFreD: Glitch-Free Duplication
Towards Power-Equalized Circuits on FPGAs

Alexander Wild, Amir Moradi, Tim Güneysu
Horst Görtz Institute for IT Security, Ruhr University Bochum, Germany
{alexander.wild, amir.moradi, tim.gueneysu}@rub.de

***Abstract*—Designers of secure hardware are required to harden their implementations against physical threats, such as power analysis attacks. In particular, cryptographic hardware circuits are required to decorrelate their current consumption from the information inferred by processing (secret) data. A common technique to achieve this goal is the use of special logic styles that aim at equalizing the current consumption at each single processing step. However, since all hiding techniques like Dual-Rail Precharge (DRP) were originally developed for ASICs, the deployment of such countermeasures on FPGA devices with fixed and predefined logic structure poses a particular challenge.**

In this work, we propose and practically evaluate a new DRP scheme (GliFreD) that has been exclusively designed for FPGA platforms. GliFreD overcomes the well-known early propagation issue, prevents glitches, uses an isolated dual-rail concept, and mitigates imbalanced routings. With all these features, GliFreD significantly exceeds the level of physical security achieved by any previously reported, related countermeasures for FPGAs.

I. Introduction

Hiding [15] is known as common class for countermeasures to protect against Side-Channel Analysis (SCA). A subset of hiding countermeasures aims at equalizing the power consumption of the cryptographic device to keep it independent from the processed data – thwarting attacks such as Differential Power Analysis (DPA). These countermeasures, also known as DPA-resistant logic styles, usually implement the Dual-Rail Precharge (DRP) concept. Examples for this are SABL [28], WDDL [29], DRSL [5], MDPL [23], iMDPL [22] that are specifically tailored to be used in Application-Specific Integrated Circuit (ASIC) devices. However, due to predefined structures and restrictions in routing, the techniques of these schemes cannot be easily applied to Field Programmable Gate Arrays (FPGAs). Therefore, most of the efforts to equalize the power consumption on FPGAs have been put in the direction of *duplication*. Fortunately, an FPGA contains similar blocks formed by a couple of slices with (nearly) equal inter- and intraconnections. Hence, re-instantiating a part of a circuit at another location on the FPGA and converting it to its dual function seems to be a viable option. Previous works investigated this concept of duplication [2], [8], [9], [10], [12], [17], [21], [24], [32], but all reported schemes still show some vulnerabilities against certain power analysis attacks. We provide a comprehensive overview of these schemes and their corresponding issues in Section II.

This work aims to design a scheme that rules out previous weaknesses to provide an SCA-resistant implementation of cryptographic circuts on FPGAs. Our scheme, denoted as *GliFreD*, avoids (1) glitches in combinatorial circuits, (2) forms a pipeline architecture, and (3) efficiently instantiates the duplication concept. We show in practical experiments on a Xilinx Spartan-6 FPGA how to combine Xilinx design tools and *RapidSmith* [11] to finally convert an unprotected circuit into a corresponding DPA-protected one under the definitions of the GliFreD scheme.

Side-channel analysis of the converted circuits implemented on the SAKURA-G platform indicates the success of GliFreD to significantly mitigate the success of DPA attacks. We further elaborate the limitations of the duplication concept and provide reasons for the leakages that cannot be completely avoided. In particular, we present practical delay measurements of duplicated routes which are notably different to those reported by the *Xilinx FPGA Editor*. Apparently, the efficiency of duplication is crucially affected by process variations that cannot be completely avoided without device-specific back annotation.

II. Background

The DRP logic schemes for FPGAs proposed in the past have to overcome three major problems. One is the *early propagation* effect, which is related to different delays of a gate input signals. Hence, the gates of some logic styles evaluate the output at different points in time depending on the input values [27]. The second phenomenon and problem to be avoided is the one of *glitches*. They are likely to occur at a gate output if an input signal changes during evaluation phase. Hence, a DRP scheme has to ensure that the evaluation phase is not initiated until all input signals have become settled. The third issue to deal with is *imbalanced routing*. Routes of different lengths and thus capacitive loads have different contributions to the amount of power consumption on signal toggle. Therefore, dual rails in a DRP scheme should use balanced routes to minimize the corresponding data-dependent leakage (see [30] as an approach for ASIC platforms). Despite of the wealth of related research published so far, all previously reported schemes suffer from at least one of the aforementioned problems. In the following we now briefly introduce the latest and most relevant DRP schemes related to FPGAs.

DRP Logic Styles

In [21] a dual-rail logic style called Balanced Cell-based Dual-rail Logic (BCDL) was introduced that employs a rendezvous box to connect all gate inputs with a global precharge signal. One of these rendezvous boxes is placed in front of

TABLE I: Overview on side-channel resistant logic styles for reconfigurable hardware

Reference	**Defects**			**Case Study**						**Evaluation**	
	EP	**Glitch**	**Routing**	**Platform**	**Target**	**Resources**			**Freq.**	**Method**	**Traces**
						ALM/Slice	LUT	FF	(MHz)		
Logic Styles											
BCDL [21]			x	Stratix II	AES-128	1841		1024	50.64	CPA [3]	150k
Triple-Rail [12]			x	Spartan 3	DES Sbox	501			n/a	CPA [3]	4k
DPL-noEE [2]	*		x	Stratix I	AES-128	14574			19.7	CPA [3]	40k
AWDDL [17]			*	Virtex 5	AES Sbox	n/a			n/a	IT [25]	200k
Duplication Schemes											
DWDDL [32]	x			Spartan 3E	AES Sbox	818			50	CPA [3]	256
Partial SDDL [10]		x		Spartan 3E	AES-128	928			n/a	CPA [3]	12k
PA-DPL [8]		x		Virtex 5	AES Sbox		286		104.8	CEMA [6]	100k
[9]			*	Virtex 5	AES Sbox	n/a			n/a	CEMA [6]	1M
[18]	single rail			Virtex 5	AES Sbox		100	649	320.51	CEPACA [19]	50M

x:problematic *:problem partially solved

each gate to fire the gate evaluation as soon as all dual input signals become stable. BCDL prevents early propagation but is not aware of different routing delays which end up with different route capacities between the `true` and `false` networks of the dual-rail circuit – finally leading to information leakage.

The work presented in [24] deals with different placement strategies for a DRP scheme on an Altera Stratix-II FPGA. Here, the authors tried to place the components of the original circuit as close as possible together and did the same for the dual circuit. Another strategy in [24] was to place the related components of the original and the dual circuits as close as possible. Both strategies do not avoid early propagation and do not allow to take control over rail delays. Besides, both strategies are based on signal delays given by the Standard Delay Format (SDF) file generated by the Quartus-II tools.

The authors of [12] employed an asynchronous approach by making use of the so-called triple-rail approach, and evaluated their implementation on a Spartan-3 FPGA. In this approach an asynchronous control signal is connected to the gate that is considered as being the latest arriving signal. This control signal fires the gate in order to prevent early propagation. No routing strategy is considered in this work so that different routing delays will show up between the original and the dual circuit. Additionally, imbalanced toggles inside internal Look-Up-Tables (LUT) are input-dependent that also results in exploitable leakage.

In [2] the authors introduced a WDDL-like scheme without early propagation called DPL-noEE. It connects the `true` and `false` gate signals (dual-rail logic) to the same LUT that fires the output when all four input signals are available. No information is given about the placement and routing strategies for the `true` and `false` gates which has been done by a tool developed by themselves called vDuplicate. The evaluation of this logic style was done on an Altera Stratix-II and reported to halve the leakage compared to WDDL.

Beside any routing issues, the work of [17] showed that DPL-noEE only prevents the early propagation at the evaluation phase. However, the transitions at the precharge phase are still data-dependent. Their improved architecture removes the data-dependent precharge by emulating an S-R latch by means of a LUT which keeps the LUT output until all input signals are precharged [17]. To mitigate the routing imbalances the authors of [17] developed a customized router to find the best match with minimal routing differences for a design. The authors reported that although the leakage is reduced using their customized router, but it cannot be completely avoided due to nonexistence of perfectly-identical dual-rail routes.

Duplication

The seminal work in [32] introduces Double WDDL (DWDDL) which duplicates a fully placed-and-routed WDDL circuit resulting in a massive resource utilization. In [10] asynchronous latches are inserted to precharge the dual-rail circuit, but the authors have stated that their style can still be broken due to upcoming glitches.

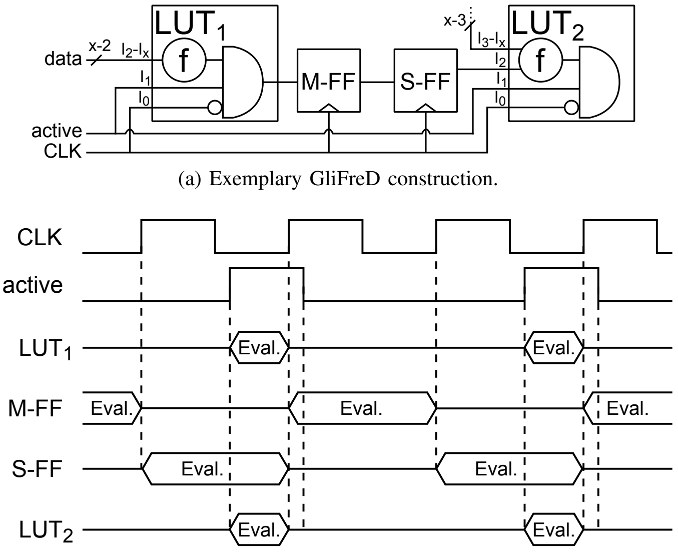
The authors of [8] mainly focused on the well-known early propagation effect by connecting control signals to every LUT in a fully combinatorial circuit with duplicated routings. However, as we show in Section V, the circuit starts to glitch right after the first LUT stage if no register is present between the subsequent LUTs to stop the signal propagation.

In a more recent work [9], the authors applied duplication while minimizing the area overhead. Their underlying strategy is to duplicate the original circuit to realize the dual one, but – in contrast to all other similar works – these two sub-circuits are interleaved. Hence, they provided algorithms to find and correct the overlapping parts which leads to differently-routed `true` and `false` networks, resulting finally in information leakage that can be exploited by an SCA. An overview of the aforementioned related works is given in Table I.

III. General Concept

In this section we explain our proposed technique denoted as Glitch-Free Duplication (*GliFreD*). GliFreD is combining two different techniques: first the work of [8], where each LUT is enabled by at least one global signal, and second, the scheme presented in [18] that considers a globally-enabled register at the output of each LUT. Recall that the work by [8] follows the duplication scenario with the issue that glitches arise for large circuits. The concept of [18] avoids propagation of glitches but does not include techniques for duplication.

GliFreD uses primarily two components: *Look-Up-Tables* (LUTs) and *Master-Slave-Flip Flops* (MS-FFs). Right after each LUT at least one MS-FF has to be placed so that a transition at a LUT output does not directly propagate into other LUTs. An exemplary instantiation is depicted in Figure 1a.



(a) Exemplary GliFreD construction.

(b) Waveforms of the given GliFreD construction.

Fig. 1: GliFreD Construction and Behavior.

Note that Figure 1b shows the corresponding waveforms and transitions between the precharge and evaluation of the given example.

Two global control signals active and CLK are connected to each LUT to control the precharge and evaluation phases. Following the statement given in [18], we assume that each LUT consists of a multiplexer tree and therefore the control signals should be connected to the very first I_0 and second multiplexer stage I_1 to avoid internal data-dependent LUT glitches. CLK is the clock signal of the circuit and is also connected to each MS-FF. active is a continuous repetitive toggling signal that runs at half of the CLK frequency. LUTs are configured in a way that data signals connected to I_2-I_X are only processed if active=HI and CLK=LO. Otherwise, LUT output is set to zero, i.e., $O = \texttt{active} \cdot \overline{\texttt{CLK}} \cdot f(I_2, \dots, I_X)$.

We should highlight that our construction (Figure 1a) avoids any glitches at the LUTs' output. The inputs of each LUT are provided by the slave flip flops (S-FFs) that become stable right after the positive edge of CLK. Since the active signal enables the output of the LUT only after a negative edge of CLK (when the LUT inputs are stable), the LUT output toggles at most once at start of the LUT evaluation phase (see Figure 1b). The same holds at the start of the LUT precharge phase as the LUT becomes disable at the positive edge of CLK short before the S-FFs go to the precharge phase.

Note that in this step, all input and output signals are computed in a single-rail fashion. After duplication, the LUTs in the dual circuit are configured to work complementary to the inverted incoming data signals, i.e., $\overline{O} = \texttt{active} \cdot \overline{\texttt{CLK}} \cdot \overline{f(\overline{I_2}, \dots, \overline{I_X})}$. Clearly, CLK and active – as control signals – are not dualized, and both original and dual circuits receive the same control signals.

CLK is connected to FFs and routed via a special clock tree for minimal signal delays. To connect CLK to the LUT input I0, the clock tree must be expanded. This is done by connecting the clock tree to the closest switch box adjacent to the target LUT to route CLK to the LUT input. We assume that the delay differences between CLK signals connected to LUTs and FFs are small and negligible.

Without any restrictions on the routing of the active signal, the routing delays of active are clearly larger than that of CLK (see Figure 1b for a conceptual view). CLK triggers the S-FFs before the falling edge of active. Without connecting CLK to the LUTs, glitches would appear right before the start of the LUTs precharge phase. Apparently, such a connection shortens the LUTs evaluation phase and avoids glitches at the LUT output. As explained, it is not necessary to consider any routing constraints for the active signal. However, as it is a global signal and FPGAs usually contain several clock trees, it can – optionally – be routed via an available clock tree to minimize routing delays.

Generally, DRP schemes follow the predicate that the number of toggling gates in the entire circuit (i.e., original and dual) is always constant at each transition between precharge and evaluation phases. For LUTs, this is guaranteed by adding a dual circuit. But as stated in [16] having only one FF stage after each LUT (even after dualization) is not sufficient to keep the number of toggles constant. As a result, all storage elements should also implement the precharge-evaluation concept as realized by MS-FFs.

As mentioned in Section II, an important issue of the DRP schemes is imbalanced routing. In order to equalize routing delays of complementary signals, in GliFreD the single-rail circuit is placed and routed in a restricted area, then its dual is added to the design by duplication, i.e., by copying the dual of all the components and corresponding routings. Such a restriction is to make sure there is room available for the dual circuit. Due to the vertically-identical architecture of the FPGAs, the strategy which we followed for duplication is to put the dual circuit below the original one. In other words, two identical and vertically-adjacent parts of the FPGA are chosen. The placement of the original circuit is limited to the upper part, and after placement and routing its dual is copied to the lower part. Note that only the LUT configurations need to be dualized as explained above ($\overline{O} = \texttt{active} \cdot \overline{\texttt{CLK}} \cdot \overline{f(\overline{I_2}, \dots, \overline{I_X})}$).

In summary, GliFreD achieves the following:

- Some dedicated modules available on FPGAs e.g., multiplexer and carry chains cannot be used due to their fixed behavior. To generate a dual circuit it is necessary to invert the behavior of all elements. Hence, any such module must be converted to its LUT-equivalent.
- Two control signals have to be connected to each LUT that results in routing and resource overhead due to a reduction from X-to-1 LUTs to (X-2)-to-1 LUTs.
- The FF utilization of a design is increased due to the required placement of at least one MS-FF between two connected LUTs. It is noteworthy that in each slice of the Xilinx FPGAs there exists a FF right after each LUT, and in non-GliFreD designs with large combinatorial circuits such FFs are left unused. In recent Xilinx 7 series, even two FFs per LUT exist in each slice [31] that can be used to form the MS-FFs. Hence, the FF-utilization overhead of GliFreD is limited to the extra FFs required to form

the pipeline architecture.

- The number of clock cycles required for the operation of the circuit depends on the LUT depth of the circuit.
- A high clock frequency can be easily achieved due to a minimal LUT depth δ (i.e., depth $\delta = 1$).
- With additional MS-FFs the circuit can be easily transformed into a fully-pipelined circuit.
- Not only fully-unrolled designs can be implemented by GliFreD. Indeed, regardless of existence of a loop in the design it can be transformed into a GliFreD circuit.

IV. Case Study

To evaluate our proposed scheme we transformed a Canright AES Sbox [4] into a GliFreD circuit for a Xilinx Spartan-6 FPGA. Multiple steps had to be performed as explained below.

GliFreD requires that each LUT is connected to two control signals. Therefore, only 4 of 6 available input pins of each LUT can be used as data pins. We therefore restrict synthesis to map the circuit into 4-to-1 LUTs only. In fact, the Sbox was simply synthesized for a Spartan-3 that natively only uses 4-to-1 LUTs. Next, a Hardware Description Language (HDL) representation was re-generated from the resulting netlist using the *ISE Design Suite*. By means of a simple script, all 4-to-1 LUT macros (in the recreated HDL file) were transformed into 6-to-1 LUT macros and adopted to consider two CLK and active control signals at $\mathtt{I}_0$ and $\mathtt{I}_1$. Finally, the control signals were connected to each LUT, and MS-FFs are added to build a fully-pipelined circuit.

Synthesizing the modified HDL file results in a netlist containing the single-rail circuit of the GliFreD Sbox. The netlist was then converted into a textual description specified by Xilinx Design Language (XDL). To build the dual counterpart of the Sbox, we prepared a tool (based on the *RapidSmith* library [11] processing the XDL file) that automatically duplicates the design (including all components and their routes). All LUTs content of the dual circuit were changed accordingly as stated in Section III. This duplication process guarantees the equivalence between the single-rail circuit and its dual counterpart. Hence, the resulting netlist contains a fully-pipelined GliFreD circuit; finally we obtained a GliFreD-enabled Sbox. Note that tools for all steps exist so that the entire process can be efficiently automated.

In the following the theoretical resource overhead is computed that is required to transform an arbitrary unrolled design based on LUTs and FFs into a GliFreD circuit. The overhead of the LUT utilization is design- and synthesizer-dependent and can be determined by the following formula:

$$u_{\mathrm{LUT}}/i_{\mathrm{LUT}} = 2\big(i_{\mathrm{LUT}}(1+p_{\mathrm{LUT}}) + m_{\mathrm{LUT}}\big)/i_{\mathrm{LUT}},$$

where u_{LUT} represents the number of LUTs used in the GliFreD circuit, i_{LUT} the number of 6-to-1 LUTs of the original design, m_{LUT} the number of elements e.g., multiplexers to be replaced by LUTs, and p_{LUT} refers to the resource utilization penalty that is inferred by using 4-to-1 LUTs instead of 6-to-1 LUTs. p_{LUT} depends on the underlying circuit as well as the synthesizer, but it can be estimated to be at most $2^6/2^4 = 4$ if all 6 inputs of all 6-to-1 LUTs are used in the original design.

TABLE II: Resource cost and comparison

Sbox Design	**Slice**	**LUT**	**FF**	**Clock** (cycles)	**Freq.** (MHz)
Canright	23	54	16	1	110
GliFreD	726	126	1176	20	250
Control	9	21	17	-	-

The overhead of the FF utilization is also design-dependent and can be given by u_{FF} as the number of FFs used in the GliFreD design and i_{FF} as the number of FFs already available in the original design by

$$u_{\mathrm{FF}}/i_{\mathrm{FF}} = \Big(2\big(u_{\mathrm{LUT}}\big) + e_{\mathrm{FF}}\Big)/i_{\mathrm{FF}}.$$

When transforming an unrolled design into a not-fully-pipelined GliFreD circuit, the number of additional FFs mostly depends on the number of used LUTs and is approximately four times the number of single-rail LUTs (so, $e_{\mathrm{FF}} = 0$). For a fully-pipelined design, this number is much higher as additional MS-FFs should be added to form the pipeline in all circuit stages. In such a case, e_{FF} refers to the number of extra FFs which are required to balance all the inputs of each LUT. This can be seen in Table II that compares the resource utilization and runtime properties of the GliFreD Sbox and an unmodified Canright Sbox. All values are post-place-and-route results, generated with *Xilinx ISE 14.7*.

It can be seen that the LUT overhead is roughly 2.33, mainly due to the dual circuit and the use of 4-to-1 instead of 6-to-1 LUTs. With the pipelined structure the FF utilization drastically increases, here we state an overhead of 73.5. The not-fully-pipelined Sbox requires 252 FFs which results in an overhead of 15.75.

As given in Section III a GliFreD circuit requires active and CLK signals to control the data flow. Hence, the resource utilization of the control logic (given in Table II) is almost constant and independent of the design transformed into GliFreD, but it needs to be added to the overall resource consumption. Beside the generation of active and CLK, the control logic contains a small Finite-State Machine (FSM) and a counter that handles the formed pipeline.

V. Practical Evaluation

In order to better examine the impact of each component in our construction, we created five slightly different architectures of the GliFreD Sbox circuits. All modifications are applied to the netlist and keep the routing of the original design untouched.

Profile 1: The GliFreD Sbox as described above.

Profile 2: LUT configurations are changed so that CLK signal has no impact on the LUT behavior.

Profile 3: LUT configurations are modified so that active signal is *don't care*.

Profile 4: The dual part of the GliFreD Sbox circuit is removed.

Profile 5: All FFs are configured as constantly transparent latches.

We consider *profile 1* as reference and compare it with all others to determine the effect of each feature that is part of our proposed scheme. With *profile 2* and *3* we evaluate the necessity of routing `CLK` and `active` signals to each LUT. *Profile 4* and *5* provide an insight on the efficiency of the dual-rail concept and the developed pipeline architecture, respectively.

A. Side-Channel Analysis (SCA)

For practical SCA we used the SAKURA-G [1] side-channel evaluation platform based on a Xilinx Spartan-6. Power traces have been collected by monitoring the voltage drop over a $1\,\Omega$ resistor at the `Vdd` path. The target FPGA operates at a frequency of $3\,\mathrm{MHz}$; power traces have been obtained by means of a digital oscilloscope at a sampling rate of $1\,\mathrm{GS/s}$. Due to the small scale of the power consumption signal, we made use of the amplifier embedded on the SAKURA-G board to acquire low-noise signals.

For each of the aforementioned design profiles we collected $n = 1\,000\,000$ traces while the Sbox input $x_i \in \{0,1\}^8$ with $i \in 1 \ldots n$ was randomly chosen from a uniform distribution. Prior to each measurement the zero input is applied to the Sbox circuit until all pipeline stages of the Sbox are initialized. While the randomly selected x_i is constantly applied to the Sbox input, the associated power trace is recorded. This setup avoids any other switching noise due to the involvement of other data being processed by the Sbox circuit in the previous or next cycles and enables to clearly evaluate the design in a worst-case scenario.

In order to fairly compare the design profiles, we considered the following two evaluation metrics. First, an information-theoretic (IT) metric [25] which examines the amount of exploitable information using mutual information between processed data and the associated leakages. The results of an IT analysis can be directly translated to the success rate of a univariate template attack. Second, we applied a moments-correlating DPA (MCDPA) [20] that provides insights on the data complexity (i.e., the number of required traces) of a successful attack by targeting a specific statistical moment of the leakage probability distributions.

For the IT analysis, we considered the first $500\,000$ collected traces of each profile and estimated for each Sbox input value $x_i \in \{0,1\}^8$ the probability distribution of the leakages independently at each sample point. The Probability Density Functions (PDFs) are estimated by Gaussian distributions where the means and variances at each sample point were estimated by the *sample mean* and *sample variance*. Similar to [13] we evaluated the amount of available information based on the conditional entropy and by means of the integral over the leakages. We, in fact, measured the amount of perceived information [7] as 1) we assume a Gaussian distribution for the leakages and 2) follow the leakage model based on Sbox input values. The obtained perceived information diagrams for all the design profiles are shown in Figure 2.

As a first observation, the GliFreD design (*profile 1*) has the lowest exploitable leakage. Note, however, that the leakage is not completely eliminated. *Profile 2* has a considerably high exploitable leakage at one of the last stages of the pipeline what confirms the necessity of routing both `CLK` and `active` signals to each LUT, as discussed in Section III. In order to pinpoint the source of this leakage, we collected the difference between the delay of `CLK` and `active` signals at each LUT input from the *Xilinx FPGA Editor*. As it can be observed from Figure 2 we identified a couple of LUTs at the 9th stage of the circuit pipeline with a delay difference $> 300\,\mathrm{ps}$ while it is at most $57\,\mathrm{ps}$ for all other LUTs in the design.

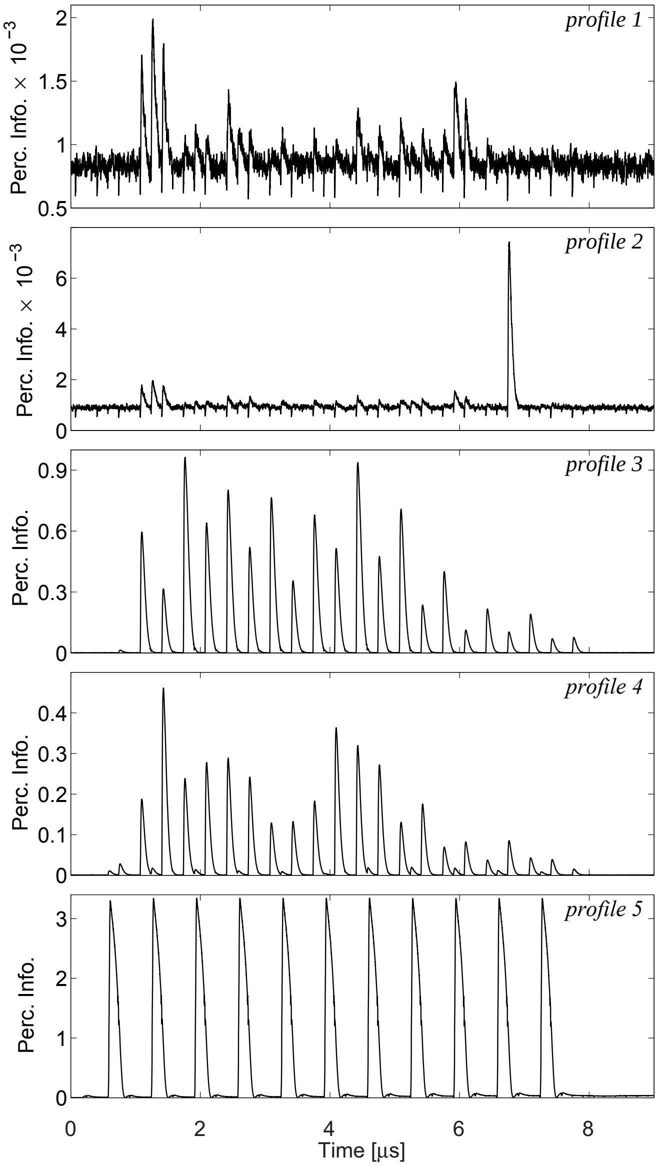


Fig. 2: Perceived Information Diagrams

In order to evaluate the data complexity, we used the first $500\,000$ traces of each profile to estimate the first-order moments and the second $500\,000$ traces to mount a profiling MCDPA at the first order. Figure 3 shows the obtained results that confirm the feasibility of an attack. When excluding routes of `CLK` to the LUTs (*profile 2*), this leads to an attack that can be successful with much less number of traces. The design of *profile 5* based on transparent latches emulates a circuit adopting the concept in [8] and evaluates the Sbox output right after `CLK=LO` and `active=HI`. Thanks to the rule-of-thumb proposed in [14], [26] and the metric feature of MCDPA [20], we can conclude that a first-order attack on GliFreD requires

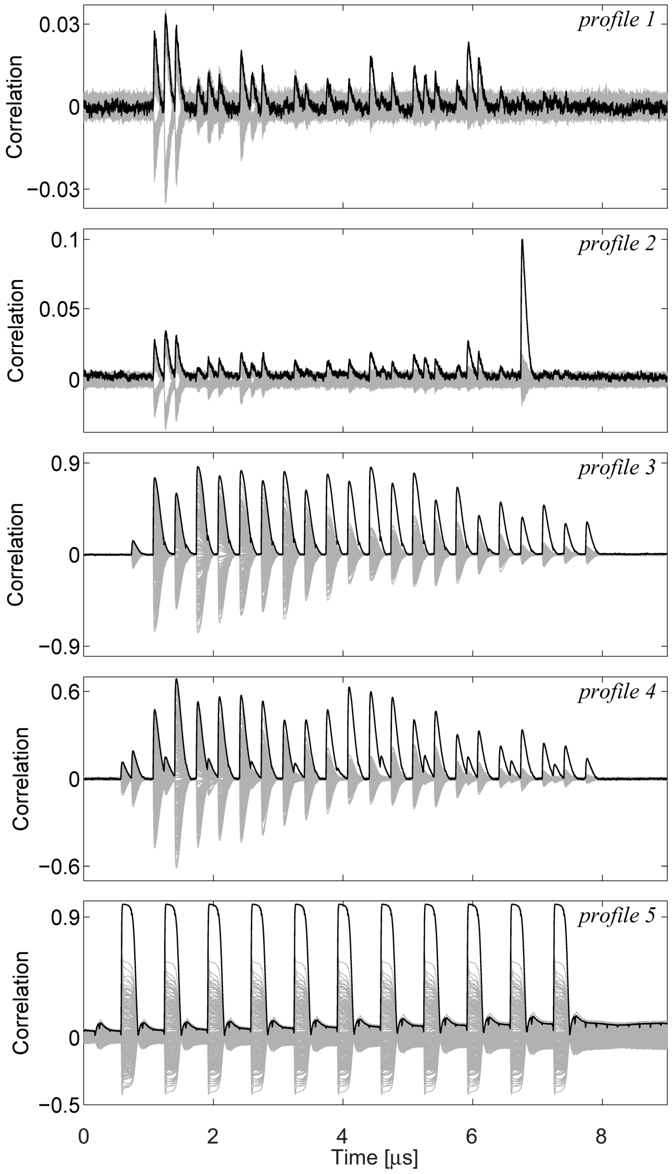


Fig. 3: Result of first-order Moments-Correlating DPA (profiled setting)

$\left(\frac{1}{0.03}\right)^2 \approx 1100$ times more traces compared to the same circuit as in [8] based on the same platform. Nevertheless, we still need to face the small first-order leakage of *profile 1* that can be exploited by a potential power attack. As a next step we investigate and identify the source for this remaining leakage.

B. Determining Device-Specific Signal Delays

GliFreD and many other similar schemes, e.g., [8], [32], are founded on the assumption that duplicated routes have exactly the same signal delays and wire capacities on an FPGA. Tools like the *Xilinx FPGA Editor* confirm this assumption *in theory*. Since all reported values are worst-case simulation times, the actual signal propagation delay of routes can differ in practice due to process variations and other influences (e.g., temperature, supply voltage or aging).

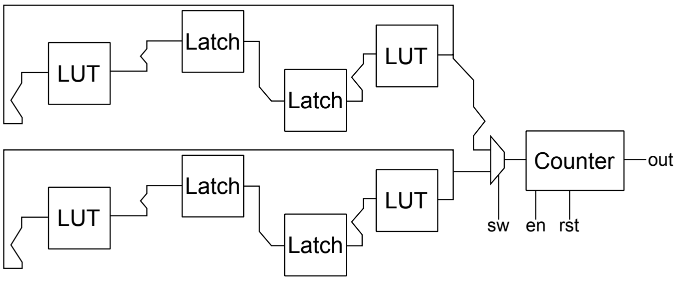


Fig. 4: Ring oscillators made by LUT-to-LUT routes

We therefore claim that the remaining leakage obtained for *profile 1* is based on the differences of individual signal delays in routing counterparts. In order to confirm this claim and to quantize the real delay differences in practice we tried to measure signal delays by using ring oscillators. Two identically-routed ring oscillators are selectively connected to an asynchronous counter. For a certain period of time each ring oscillator drives the counter to obtain its frequency and hence the delay of each ring oscillator.

For the sake of simplicity, we measured the delay of the paths in the Sbox from LUT to LUT. The first LUT in each path is configured as an inverter, while the second one is used as pass-through. To route the output of the second LUT back to the first LUT (realizing a loop), an additional net (that is not a part of the GliFreD Sbox) has to be added to the original and dual part of the circuit. Note that this feedback net is identically routed for both parts but also contributes in the delay difference between the original and dual ring oscillators. We further need to configure the FFs between two connected LUTs as transparent latches to avoid blocking the oscillation (see Figure 4).

To automate these measurements, we developed a tool using *RapidSmith* to extract all LUT-to-LUT routes of our GliFreD Sbox. For each of the extracted routes the following procedure is performed:

- The route including the associated LUTs and FFs are inserted into a new design at the same location that also contains the aforementioned asynchronous counter.
- LUTs and FFs configurations are modified as explained above to support oscillation.
- The XDL file is converted to Native Circuit Description (NCD), and the feedback path is added to the design by a *Xilinx FPGA Editor* script.
- The NCD file is converted back to XDL, and the fully routed ring oscillator is then copied using *RapidSmith* to the corresponding dual part of the original GliFreD Sbox design.
- Finally, by another *Xilinx FPGA Editor* script two ring oscillators are connected to the switch driving the asynchronous counter. Finally, the bitstream is generated.

Our tool generated 301 bitstream files, each of which contains two identically-routed ring oscillators with a LUT-to-LUT route of the GliFreD Sbox design (*profile 1*), a 24-bit asynchronous counter, and surrounding logic to communicate with the control FPGA.

Each of the generated bitstreams was loaded by a script into the target FPGA of the SAKURA-G. Next, the two 24-bit values (c_0 and c_1) that are associated with the asynchronous counter driven by the ring oscillators for the period of time T

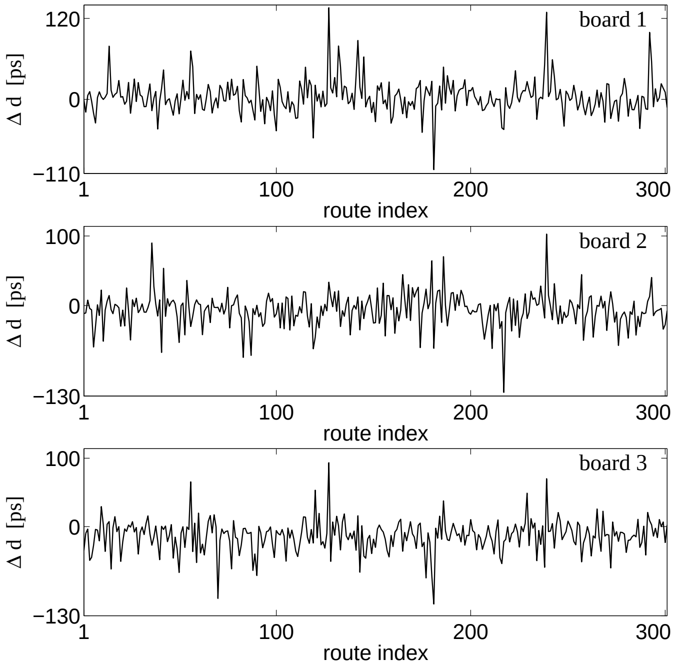


Fig. 5: The delay difference between the LUT-to-LUT dual routes, examined on 3 SAKURA-G boards

were evaluated. Environmental noise that influences measurements were mitigated by repeating this process 1000 times to compute the means. Finally, $\bar{c}_0$ and $\bar{c}_1$ were achieved which determine the delay of each ring by $d_0 = \frac{T}{2\bar{c}_0}$ and $d_1 = \frac{T}{2\bar{c}_1}$.[1] Finally, $\Delta d = d_0 - d_1$ provides the difference in the delay of the two rings. Figure 5 shows Δd values we observed for all 301 bitstreams (LUT-to-LUT routes) that significantly differ from the equal route delay as reported by *Xilinx FPGA Editor*. The delay difference of some identically-routed rings exceed even $100\,$ps.

Since we expect that such differences are predominantly due to process variations, we repeated the same experiment on two other SAKURA-G boards using the same supply voltage $1.2\,$V at room temperature. The corresponding results are very diverse and shown in Figure 5. By changing the supply voltage to $1.0\,$V, we expected an increase in Δd that was confirmed by our experiments.

As a result, the delay values reported by *Xilinx FPGA Editor* are only a very rough estimate to the reality. However, still copying signal routes in FPGAs that follow the same structure and the same shape in *Xilinx FPGA Editor* seems to provide the best option and the most straightforward way to produce similar (yet not identical) routes in FPGAs.

VI. Discussions and Future Work

In this work, the DRP scheme called GliFreD was proposed to eliminate leakages of critical information while processing cryptographic operations due to early propagation, glitches or imbalanced routing.

We successfully showed that vertical copies of routing structures are optimal to obtain virtually identical routes with minimal leakage. We further quantified the remaining leakage that cannot be completely avoided due to routing-specific signal delays. In fact, the delay report given by Xilinx tools does not match with the reality by far. In this context accurate and device-specific routing information (such as device-specific back annotation) is required to improve the nature of this countermeasure for FPGA devices. This additional information can be considered by a tool that automatically translates any HDL-based design (not only an unrolled circuit) to GliFreD as future work.

[1] The factor of two is due to the nature of the ring oscillator as the counter is clocked once when a transition passes the ring two times.

References

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

[2] S. Bhasin, S. Guilley, F. Flament, N. Selmane, and J. Danger. Countering early evaluation: an approach towards robust dual-rail precharge logic. In *WESS 2010*, page 6. ACM, 2010.

[3] E. Brier, C. Clavier, and F. Olivier. Correlation Power Analysis with a Leakage Model. In *CHES - 2004*, volume 3156 of *LNCS*, pages 16–29. Springer, 2004.

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

[5] Z. Chen and Y. Zhou. Dual-Rail Random Switching Logic: A Countermeasure to Reduce Side Channel Leakage. In *CHES 2006*, volume 4249 of *LNCS*, pages 242–254. Springer, 2006.

[6] G. L. Ding, J. Chu, L. Yuan, and Q. Zhao. Correlation Electromagnetic Analysis for Cryptographic Device. In *PACCS - 2009*, pages 388–391. IEEE, 2009.

[7] F. Durvaux, F. Standaert, and N. Veyrat-Charvillon. How to Certify the Leakage of a Chip? In *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 459–476. Springer, 2014.

[8] W. He, E. de la Torre, and T. Riesgo. A Precharge-Absorbed DPL Logic for Reducing Early Propagation Effects on FPGA Implementations. In *ReConFig 2011*, pages 217–222. IEEE, 2011.

[9] W. He, A. Otero, E. de la Torre, and T. Riesgo. Automatic generation of identical routing pairs for FPGA implemented DPL logic. In *ReConFig 2012*, pages 1–6. IEEE, 2012.

[10] J. Kaps and R. Velegalati. DPA Resistant AES on FPGA Using Partial DDL. In *FCCM 2010*, pages 273–280. IEEE, 2010.

[11] C. Lavin, M. Padilla, J. Lamprecht, P. Lundrigan, B. Nelson, B. Hutchings, and M. Wirthlin. RapidSmith – A Library for Low-level Manipulation of Partially Placed-and-Routed FPGA Designs. Technical report, Brigham Young University, September 2012.

[12] V. Lomné, P. Maurine, L. Torres, M. Robert, R. Soares, and N. Calazans. Evaluation on FPGA of triple rail logic robustness against DPA and DEMA. In *DATE 2009*, pages 634–639. IEEE, 2009.

[13] F. Macé, F.-X. Standaert, and J.-J. Quisquater. Information Theoretic Evaluation of Side-Channel Resistant Logic Styles. In *CHES 2007*, volume 4727 of *LNCS*, pages 427–442. Springer, 2007.

[14] S. Mangard. Hardware Countermeasures against DPA ? A Statistical Analysis of Their Effectiveness. In *CT-RSA 2004*, volume 2964 of *LNCS*, pages 222–235. Springer, 2004.

[15] S. Mangard, E. Oswald, and T. Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, 2007.

[16] A. Moradi, T. Eisenbarth, A. Poschmann, and C. Paar. Power Analysis of Single-Rail Storage Elements as Used in MDPL. In *ICISC 2009*, volume 5984 of *LNCS*, pages 146–160. Springer, 2009.

[17] A. Moradi and V. Immler. Early Propagation and Imbalanced Routing, How to Diminish in FPGAs. In *CHES 2014*, volume 8731 of *LNCS*, pages 598–615. Springer, 2014.

[18] A. Moradi and O. Mischke. Glitch-free implementation of masking in modern FPGAs. In *HOST 2012*, pages 89–95. IEEE, 2012.

[19] A. Moradi, O. Mischke, and T. Eisenbarth. Correlation-Enhanced Power Analysis Collision Attack. In *CHES - 2010*, volume 6225 of *LNCS*, pages 125–139. Springer, 2010.

[20] A. Moradi and F.-X. Standaert. Moments-Correlating DPA. Cryptology ePrint Archive, Report 2014/409, 2014. http://eprint.iacr.org/.

[21] M. Nassar, S. Bhasin, J. Danger, G. Duc, and S. Guilley. BCDL: A high speed balanced DPL for FPGA with global precharge and no early evaluation. In *DATE 2010*, pages 849–854. IEEE, 2010.

[22] T. Popp, M. Kirschbaum, T. Zefferer, and S. Mangard. Evaluation of the Masked Logic Style MDPL on a Prototype Chip. In *CHES 2007*, volume 4727 of *LNCS*, pages 81–94. Springer, 2007.

[23] T. Popp and S. Mangard. Masked Dual-Rail Pre-charge Logic: DPA-Resistance Without Routing Constraints. In *CHES 2005*, volume 3659 of *LNCS*, pages 172–186. Springer, 2005.

[24] L. Sauvage, M. Nassar, S. Guilley, F. Flament, J. Danger, and Y. Mathieu. DPL on Stratix II FPGA: What to Expect? In *ReConFig 2009*, pages 243–248. IEEE, 2009.

[25] F. Standaert, T. Malkin, and M. Yung. A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks. In *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 443–461. Springer, 2009.

[26] F.-X. Standaert, E. Peeters, G. Rouvroy, and J.-J. Quisquater. An overview of power analysis attacks against field programmable gate arrays. *Proceedings of the IEEE*, 94(2):383–394, 2006.

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

[28] K. Tiri, M. Akmal, and I. Verbauwhede. A dynamic and differential CMOS logic with signal independent power consumption to withstand differential power analysis on smart cards. In *ESSCIRC 2002*, pages 403–406, 2002.

[29] K. Tiri and I. Verbauwhede. A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation. In *DATE 2004*, pages 246–251. IEEE, 2004.

[30] K. Tiri and I. Verbauwhede. Place and Route for Secure Standard Cell Design. In *CARDIS 2004*, volume 153 of *IFIP*, pages 143–158. Kluwer/Springer, 2004.

[31] Xilinx. *7 Series FPGAs Configurable Logic Block: User Guide*, November 2014.

[32] P. Yu and P. Schaumont. Secure FPGA circuits using controlled placement and routing. In *CODES+ISSS 2007*, pages 45–50, 2007.