# Information Leakage from FPGA Routing and Logic Elements

Ilias Giechaskiel
Independent Researcher
London, United Kingdom
ilias@giechaskiel.com

Jakub Szefer
Yale University
New Haven, CT, USA
jakub.szefer@yale.edu

## ABSTRACT

Information leakage in FPGAs poses a danger whenever multiple users share the reconfigurable fabric, for example in multi-tenant Cloud FPGAs, or whenever a potentially malicious IP module is synthesized within a single user's design on an FPGA. In such scenarios, capacitive crosstalk between so-called *long* routing wires has been previously shown to be a security vulnerability in both Xilinx and Intel FPGAs. Specifically, both static and dynamic values on long wires have been demonstrated to affect the delays of the adjacent long wires, and such delay changes have been exploited to steal sensitive information such as bits of cryptographic keys. While long-wire leakage is now well-understood and can be defended against, this work presents two other, new types of information leaks that pose similar risks, but which have not been studied in the past, and for which existing defenses do not work. First, this paper shows that other types of routing resources (namely *medium* wires) are also vulnerable to crosstalk, with changes in their delays also measurable fully on-chip. Second, this work introduces a novel source of information leaks that originates from logic elements within the FPGA Configurable Logic Blocks (CLBs) and is likely not the result of the capacitive crosstalk effects investigated in prior work. To understand the potential impact of the two new leakage sources, this paper experimentally characterizes and compares them in four families of Xilinx FPGAs, and discusses potential countermeasures in the context of existing attacks and defenses.

## CCS CONCEPTS

**• Security and privacy → Hardware attacks and countermeasures**; **Embedded systems security**; **• Hardware → Reconfigurable logic and FPGAs**.

## KEYWORDS

FPGA security, information leakage, wire leakage, routing resources, logic resources, crosstalk, covert channels, side channels

**ACM Reference Format:**
Ilias Giechaskiel and Jakub Szefer. 2020. Information Leakage from FPGA Routing and Logic Elements. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD '20), November 2–5, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3400302.3415695

## 1 INTRODUCTION

Information leakage in FPGAs presents a security threat whenever an FPGA is shared between different users, or when a design incorporates untrusted third-party IP modules. For example, multi-tenant Cloud FPGAs have been recently proposed [1, 2, 4, 15–17, 27, 28], where designs from multiple users are synthesized onto the same FPGA. To maximize the usage of the FPGA resources, a number of the multi-tenant proposals do not enforce any physical separation between the logic circuits integrated from different users [1, 2, 28]. Although this can enable better and more cost-effective utilization of the cloud-based FPGAs, it also allows for different users' designs to be routed close to each other and share the reconfigurable FPGA fabric. Capacitive crosstalk and other unintended electrical interactions can then inadvertently leak information.

The potential information leakage between different users' designs can happen not only due to dynamic activity across the FPGA chip [10], but also because of short-lived static signals when parts of the designs are routed on adjacent wires in the FPGA [7–9, 20, 21]. Moreover, as we show in this paper for the first time, information leaks also occur when these signals are routed through multiplexers in the same Configurable Logic Block (CLB). Often, during synthesis, the whole design is flattened and then routed. This flattening erases any module boundaries, and logic belonging to different modules or users can end up on the same CLB. Likewise for routing, even without flattening, wires from different designs are inevitably routed on adjacent resources. Previously, only routing-related information leakage due to the use of long wires had been considered, while CLB-based vulnerabilities and medium-wire-based leaks had been completely ignored.

Specifically, existing attacks and defenses focus predominantly on capacitive crosstalk between these so-called *long* routing wires. Information leakage due to long-wire crosstalk has been shown to be a security vulnerability in Xilinx [7–9] and Intel [20, 21] FPGAs, leading to covert- and side-channel attacks. Both static and dynamic values on long wires affect the delays of the adjacent wires, which can then be estimated using on-chip Ring Oscillators (ROs). These changes in the delays of long wires have been exploited to steal sensitive information, such as cryptographic keys from AES [21] or other [7] algorithms when the key bits simply happen to be carried on the victim wires.

Because of the dangers of information leakage, various countermeasures could be deployed, but have short-comings that prevent them from addressing the threats introduced in this work. Using constraint specifications in the design tools, each user's design could be confined to a fixed region in the shared fabric, for example through Pblocks, as well as `EXCLUDE_PLACEMENT` and `CONTAIN_ROUTING` commands in Xilinx's Vivado flow. However, as there is no `EXCLUDE_ROUTING` flag, other Pblocks may still route their signals through the interconnect provided by the first Pblock's switch matrices.

Alternatively, defense mechanisms have been presented to specifically tackle the security threats of information leakage in FPGAs. They focus broadly on two types of countermeasures. One set of proposals demands isolating different circuits through physical partitioning, e.g., [14], but physical isolation can lead to poor resource utilization, and still does not prevent information leakage due to dynamic activity, e.g., [10]. The other approaches use design rule checks (DRCs) that prevent long wires in mutually-untrusting designs from being routed next to each other [7–9, 19, 24]. These existing security proposals that protect against long-wire leakage do not currently defend against the novel sources of vulnerability introduced in this paper which use medium wires or multiplexers in CLBs.

More precisely, this work identifies that *medium* wires are affected by crosstalk. These medium wires are the VLONG12s, which span 12 CLBs in the 7 Series Xilinx FPGAs, compared to VLONG18 long wires, which span 18 CLBs. Although shorter than long wires, we show for the first time that medium wires of victim designs leak information about the values carried on them to attacker designs routed on adjacent wires, despite differences in the electrical characteristics between long and medium wires. Second, this work introduces an entirely different source of information leaks that originates from the logic elements within Configurable Logic Blocks (CLBs). The vulnerability we present is due to routing and multiplexers (MUXes) within the CLBs. As design tools pack the logic into CLBs to minimize area, the new information leakage can be abused by attacker logic that is placed onto the same CLB as the victim logic to deduce information about slow-changing values. In other words, this phenomenon is not one of voltage drops due to high switching activity. The two new types of information leakage are evaluated on four families of Xilinx FPGAs (Artix 7, Kintex 7, Zynq-7000, and Kintex UltraScale), and countermeasures are discussed for both leakage sources.

## 1.1 Contributions

This work aims to advance the understanding of potential information leakage sources in FPGAs, and to help design better defenses to protect against them. The contributions of this work include:

(1) The evaluation of *long*-wire leakage in two additional Xilinx FPGA families not previously studied.
(2) The identification of *medium* wires as a new class of routing resources that leak information about their values, in spite of electrical differences compared to long wires.
(3) The introduction of logic elements (multiplexers) as a novel and stronger source of information leakage within CLBs.
(4) An experimental evaluation of the new sources of vulnerability, and a discussion of possible causes and effects.
(5) An analysis of the limitations of existing defense frameworks, and the proposal of new countermeasures.

## 2 BACKGROUND

To help understand the new sources of information leakage, this section discusses the logic and routing resources of the Xilinx 7 Series (Section 2.1) and UltraScale (Section 2.2) architectures. It also summarizes prior work in long-wire leakage (Section 2.3) to contextualize the rest of the paper.

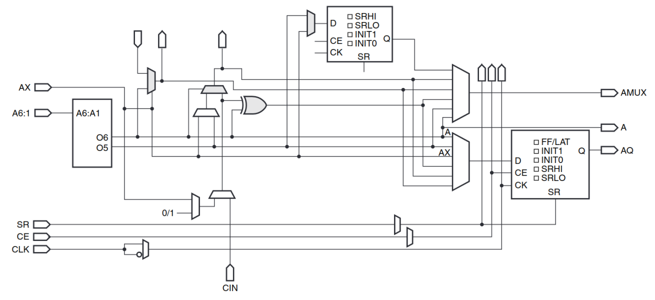

**Figure 1: A Xilinx 7 Series slice contains four lookup tables, eight storage elements, a carry chain, and wide multiplexers. Only a quarter of a slice is shown here. Figure adapted from a Xilinx User Guide [29].**

## 2.1 7 Series Architecture

The Xilinx 7 Series generation, first introduced in 2010, is comprised of four FPGA families with different cost, performance, and price tradeoffs [31]. These FPGAs, built on a 28 nm, high-k metal gate (HKMG) process, consist of the Spartan 7, Artix 7, Kintex 7, and Virtex 7 families [31]. The 7 Series is also related to the Zynq-7000 System-on-Chip (SoC) family. The Zynq-7000 extends an Artix 7 or Kintex 7 Programmable Logic (PL) fabric with ARM Cortex-A9 CPU cores in its Processing System (PS) [33]. These five FPGA families have similar internal layouts, whose primary building element is the Configurable Logic Block (CLB) and its associated switch matrix, which is used for routing.

A CLB is composed of two slices, each of which contains four lookup tables (LUTs) and eight registers, along with a dedicated carry chain and multiplexers (MUXes) [29]. Slices come in two forms, SLICEL and SLICEM, with SLICEM being usable for distributed memory and shift registers. Figure 1 shows some of the SLICEL resources.

The CLB's switch matrix contains routing resources to connect elements within a CLB, and enables CLBs to communicate with each other. There are multiple types of such communication wires with different orientations, directions, and lengths, but prior work on FPGA security has primarily focused on vertical *long* routing resources (VLONG18s). Long wires in 7 Series FPGAs are used to efficiently connect switch matrices that are 18 CLBs apart. They can be driven from either end (i.e., they are bidirectional), and have an intermediate read-only tap halfway through. Each CLB can only drive one of two VLONG18s, one towards the bottom of the device, and one towards the top, with adjacent long wires originating from adjacent CLBs.

The 7 Series FPGAs, however, have an additional type of a slightly shorter wire, which spans 12 CLBs. These are the VLONG12s, which we call *medium* wires for brevity, and show in Section 4 that they pose a similar, though weaker, information leakage threat as their long-wire counterparts. The VLONG12s are bidirectional but tapless and CLBs again only have access to two of them. We further discuss their implications for existing defense mechanisms in Section 6.

**Table 1: Properties of the boards used in the experiments. Clocks can be differential (D) or single-ended (S).**

| Board | Family | FPGA Chip | Clock |
|---|---|---|---|
| Nexys4 DDR | Artix 7 | xc7a100tcsg324-1 | 100 MHz S |
| PYNQ-Z2 | Zynq-7000 | xc7z020clg400-1 | 125 MHz S |
| KC705 | Kintex 7 | xc7k325tffg900-2 | 200 MHz D |
| KCU1500 | Kintex UltraScale | xcku115-flvb2104-2-e | 300 MHz D |

## 2.2 UltraScale Architecture

The Kintex and Virtex UltraScale FPGAs from Xilinx are built on a smaller 20 nm process, and also organize their logic resources in CLBs. The UltraScale CLBs have combined the two slices into one, which contains all 8 LUTs and 16 registers, along with the carry chain and multiplexers [30].

## 2.3 Long-Wire Leakage

Prior work has shown that the values carried by long wires affect the delays of their adjacent long wires in a way which can be measured on-chip by estimating the frequency of ring oscillators (ROs). The victim wire may be in sync with the RO for crosstalk measurement [6], or completely independent for covert- and side-channel attacks. The main insight behind these attacks is that when a long wire carries a 1, adjacent long wires become slightly faster compared to when the same wire is carrying a 0. This phenomenon is present in many types of Intel [20, 21] and Xilinx [7–9] FPGAs, and long-wire leakage in two additional families (Kintex 7 and Zynq-7000) is shown in this paper. This work further demonstrates that medium wires also leak information about their state, but that the extent of the leakage differs from that of long wires, even if their relative length is taken into account (Section 4). This suggests electrical and/or physical differences in their characteristics. It is worth noting that the threats posed by long-wire leakage have prompted defense frameworks [19, 24]. These countermeasures are discussed in Section 6, along with their limitations against the new sources of information leakage introduced in this paper.

## 3 EXPERIMENTAL SETUP

In this section we describe the boards (Section 3.1) and the design (Section 3.2) of our experiments, and briefly discuss the metric used to evaluate routing and logic information leakage (Section 3.3).

## 3.1 Devices Used

Experiments are performed on four boards from different manufacturers and device families: a Nexys4 DDR from Digilent with an Artix 7 chip [3], a PYNQ-Z2 from TUL with a Zynq-7000 [26], a KC705 from Xilinx with a Kintex 7 [34], and a KCU1500 from Xilinx with a Kintex UltraScale [32]. The clock frequencies for these four devices range between 100-300 MHz, with some being differential and others single-end. We use the default clock configuration on each board and do not attempt to improve clock accuracy with a Mixed-Mode Clock Manager (MMCM) or Phase-Locked Loop (PLL). The properties of the boards tested are summarized in Table 1.

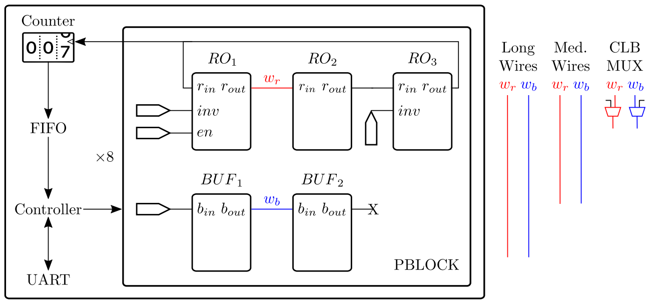

**Figure 2: Experimental setup. A three-stage ring oscillator (configurable as three inverters, or one inverter and two buffers), and a two-stage buffer are placed on one Pblock. There are eight such Pblocks with ring oscillator and buffer pairs. Control signals and measurements are taken through counters in a controller, which stores them in a FIFO and transfers them to a PC over the UART for analysis. The three routing configurations of wires $w_r$ and $w_b$ are shown on the right, where for long- and medium-wire leakage $w_{\{r,b\}}$ are adjacent, while for logic-based information leakage, they use the CLB's multiplexers (MUXes).**

## 3.2 Architectural Design

The basic building block for both the medium-wire and CLB-logic information leakage experiments consists of a 2-stage "victim" buffer and a 3-stage measurement Ring Oscillator (RO). For the 2-stage victim, the buffer lookup tables (LUTs), $BUF_{\{1,2\}}$, mirror their input to their output ($b_{out} = b_{in}$), with the output of the second LUT, $BUF_2$, remaining unconnected. The RO contains one stage, $RO_2$, which is a fixed inverter ($r_{out} = \neg r_{in}$), and two stages, $RO_{\{1,3\}}$, which can be configured as either buffers or inverters. Although the concrete routing and placement of the victim buffer and measurement RO depends on the experiment (e.g., whether the routing between buffer or RO stages uses adjacent medium wires), the five LUTs ($RO_{\{1,2,3\}}$ and $BUF_{\{1,2\}}$) and their intermediate wires are placed in a Pblock, with no other logic present through the `EXCLUDE_PLACEMENT` and `CONTAIN_ROUTING` constraints. The logic contained in a Pblock (Figure 2) produces one output (the RO signal), and has three inputs: the buffer value, an enable signal for the ring oscillator, and a signal which controls whether all three RO LUTs act as inverters, or whether the RO consists of one inverter and two buffers. Although the RO type does not have an effect on the medium-wire leakage experiments of Section 4, it does affect the information leakage within CLBs, as shown in Section 5.

For testing, eight of these measurement RO and victim buffer pairs are instantiated on different SLICEM and SLICEL locations of the device. A controller measures the ROs' frequencies through counters, stores them in a FIFO, and transfers the data to a PC for analysis over the UART. The counters, driven through the RO outputs, are sampled every $2^t$ clock cycles, with $t = 21$ by default (7-21 ms, depending on the board). As in prior work on long-wire leakage [7–9, 20, 21], the signal to the buffer toggles after each measurement period. Figure 2 summarizes the experimental setup.

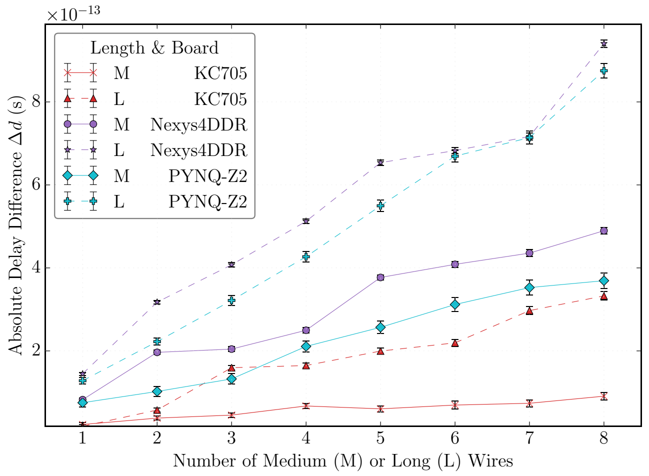

**Figure 3: Absolute wire delay differences $\Delta d$ showing that both medium (M) and long (L) wires leak information about their state to adjacent wires.**

For each experiment, we collect 4,096 data points five times, for a total of 20,480 measurements from each RO per testing configuration. All bitstreams are created with Vivado 2019.2, and results are reported with 99% confidence intervals.

## 3.3 Measurement Metric

As different ROs can have different frequencies depending on minor process, voltage, and temperature (PVT) variations across the die [12], to quantify the extent of the information leakage across different setups, we need to use a metric that is insensitive to the clock and RO frequencies.

To overcome this limitation, we can estimate the absolute delay difference $\Delta d$ of each RO by accounting for the clock frequency $f_c$ and the measurement period $2^t$, and adapting an equation derived in previous works [9, 10, 20]:

$$\Delta d = \frac{2^t}{2f_c} \cdot \frac{|C^1 - C^0|}{C^0 C^1} \tag{1}$$

where $C^i$ represents the RO count when the buffer carries value $i \in \{0, 1\}$.

## 4 MEDIUM-WIRE LEAKAGE

In this section we first measure long-wire leakage in the 7 Series devices (and especially show long-wire leakage in the Zynq-7000 and Kintex 7 families for the first time) and compare it to the new source of routing information leakage due to medium wires. To do so, we use the setup of Figure 2 and make the wires $w_r$ and $w_b$ use adjacent medium (or long) wires. In our experiments, the $i$-th ($1 \leq i \leq 8$) RO and buffer both use $i$ chained medium (respectively long) wires between two of their stages, with the $i$ medium (respectively long) wires in the RO routed parallel to the $i$ medium (respectively long) wires in the buffer. We plot the absolute delay difference $\Delta d$ of each RO as calculated by Equation (1) in Figure 3.

As Figure 3 shows, both medium- and long-wire leakage are present in the devices tested, although medium-wire leakage is weaker than its longer counterpart, and for some devices may require chaining multiple wires together to be exploitable. However, the differences between medium and long wires are not just due to the length of the wire. If we let the normalized absolute delay difference due to long and medium wires be $\Delta d_n^L = \Delta d^L/18$ and $\Delta d_n^M = \Delta d^M/12$ respectively, then, as Table 2 indicates, long-wire leakage is $1.19 - 1.85\times$ stronger than medium-wire leakage (corresponding to $1.01 - 1.73$ fs more information leakage per wire segment), even when the relative lengths of the wires are taken into account. Figure 4 shows an example of this by comparing raw data for 4 long and 6 medium wires, both spanning $4 \cdot 18 = 6 \cdot 12 = 72$ CLBs. This suggests that the two types of wires are not the same physically or electrically, as also supported by the lack of intermediate taps in medium wires (Section 2.1). We discuss this insight further in Section 6.

It should be noted that the same tests were repeated with measurement periods of $2^{19}$ and $2^{23}$ clock cycles, as well as with ROs made up of three inverters or one inverter and two buffers, and the results remained within 0.1-13% of each other, depending on the device. Finally, it is also worth highlighting that the extent of the information leakage seems to be broadly correlated with the size of the FPGA chip, with wires in larger, higher-end devices generally being less susceptible to information leakage (subject to process variations between and within boards of the same family). This observation is in line with the results of Giechaskiel et al. [7], where long-wire leakage in Basys 3 boards was more pronounced than it was in larger (in terms of resources) Nexys4 DDRs, both of which use Artix 7 chips.

**Table 2: Normalized long- and medium-wire leakage $\Delta d_n^L$ & $\Delta d_n^M$: long-wire leakage is stronger, even when accounting for the length of the wires.**

| Board | $\Delta d_n^L$ | $\Delta d_n^M$ | $\Delta d_n^L - \Delta d_n^M$ | $\Delta d_n^L / \Delta d_n^M$ |
|---|---|---|---|---|
| Nexys4 DDR | 7.23 fs | 6.08 fs | 1.15 fs | 1.19× |
| PYNQ-Z2 | 6.23 fs | 4.50 fs | 1.73 fs | 1.39× |
| KC705 | 2.19 fs | 1.18 fs | 1.01 fs | 1.85× |

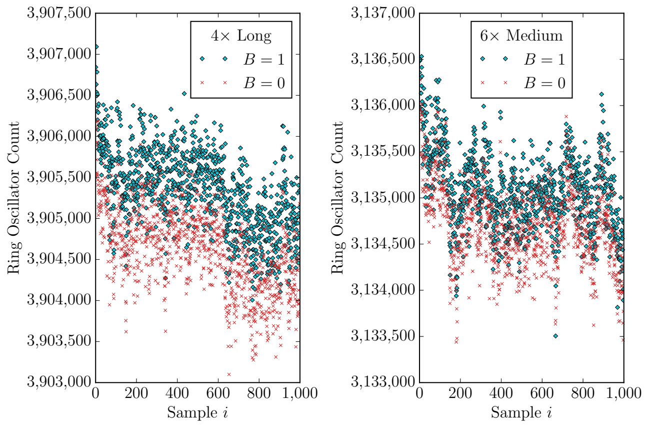

**Figure 4: Long- (left) and medium- (right) wire leakage on the Nexys4 DDR board for an equal span of 72 CLBs.**

## 5 CONFIGURABLE LOGIC BLOCK LEAKAGE

This section introduces the second new source of static information leakage, which unlike medium- and long-wire leakage depends on the internal structure of Configurable Logic Blocks (CLBs) rather than capacitive crosstalk between different routing resources. Lookup Tables (LUTs) in Xilinx CLBs have 6 inputs, but can be configured as either 6-input, 1-output elements or as 5-input, 2-output ones. Moreover, each LUT has an associated carry-in and carry-out bit from the carry chain. These values, along with a register output and the outputs of the two adjacent LUTs, are selectable through a multiplexer (MUX) in Xilinx FPGAs (see the `AMUX` output in Figure 1). As we show for the first time in this paper, this additional routing and logic within CLBs can lead to unintentional information leakage. This vulnerability, much like the routing leakage of Section 4 is static, i.e., exists when the values being spied upon remain constant. This distinguishes it from prior work which depends on voltage drops due to dynamic activity, which are easier to detect (see Section 6).

We first introduce the placement and routing setup in Section 5.1, establish the information leakage for all four boards in Section 5.2, and then compare the vulnerability with and without the MUX outputs in Section 5.3. Finally, we analyze the information leakage for all possible instantiations of RO and buffer LUTs within a Kintex UltraScale CLB in Section 5.4.

### 5.1 Placement and Routing

The basic setup for the logic information leakage experiments remains the same as that of Figure 2 in Section 3, but the concrete placement of the RO and buffer LUTs and the manual routing of the $w_r$ and $w_b$ wires changes. Specifically, all five LUTs are now placed within the same CLB, in one slice for the Kintex UltraScale board, and in two slices for the other boards respectively. Both SLICEL and SLICEM locations are used. The $BUF_1$ and $RO_1$ LUTs occupy adjacent LUTs: `A` and `B` in the 7 Series devices, while `H` and `G` for the Kintex UltraScale. The specific LUTs used are varied in Section 5.4, while the remaining three LUTs are fixed. In the experiments of Section 5.2, both $w_r$ and $w_b$ are routed through their respective MUX outputs as shown in Figure 2, but Section 5.3 uses alternative wiring to pinpoint the cause of the information leakage.

### 5.2 Baseline Results

The first results we present in Figure 5 contain raw measurements of the logic information leakage from the Kintex UltraScale in two different measurement setups: with the RO configured as an inverter and two buffers (left), or as three inverters (right). The bitstream used and the RO displayed on the graph are identical—only the runtime configuration of the RO has changed between the two measurements. As Figure 5 indicates, the information leakage is strong in both cases, and easily distinguishable even with a simple thresholding mechanism. However, the "direction" of leakage (whether RO counts are higher when spying on a 0 or a 1) is dependent upon the RO inputs themselves. This is due to the fact that the vulnerability affects different LUTs within the same CLB differently, and thus counteracts the information leakage towards the other RO LUTs. This aspect of the phenomenon is further explored in Section 5.4.

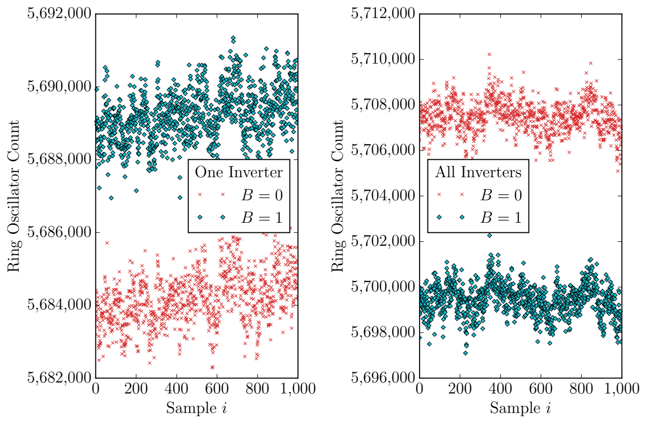

**Figure 5: Information leakage on the Kintex UltraScale due to multiplexers with one inverter and two buffers (left) or three inverters (right).**

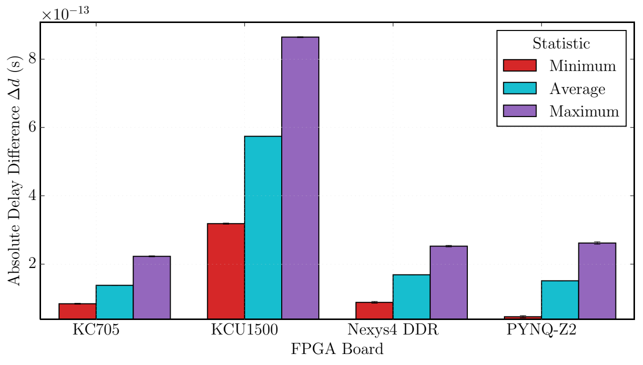

**Figure 6: Minimum, average, and maximum information leakage due to CLB logic as measured by the eight ring oscillators on the four FPGA boards.**

Figure 6 plots the minimum, maximum, and average delay difference $\Delta d$ as calculated using Equation (1) for the 8 ROs on each of the four devices under test using the three-inverter setup. As Figure 6 indicates, all devices exhibit this logic-based information leakage, with broadly consistent results among the various device locations. That said, the vulnerability is much stronger with the Kintex UltraScale device compared to its 28 nm counterparts.

### 5.3 Effect of LUT Output Type

We further investigate the cause of the information leakage by testing four different routing combinations, where the ring oscillator and/or the buffer are allowed to use either the regular or the multiplexed output. In addition, we implement a setup where the buffer only has one stage, with its output $w_b$ being unconnected. The maximum leakage under the six possible setups for the four boards is plotted in Figure 7 for the three inverters setup (the information leakage for the average, minimum, and one inverter/two buffers setup is similar, but less pronounced).

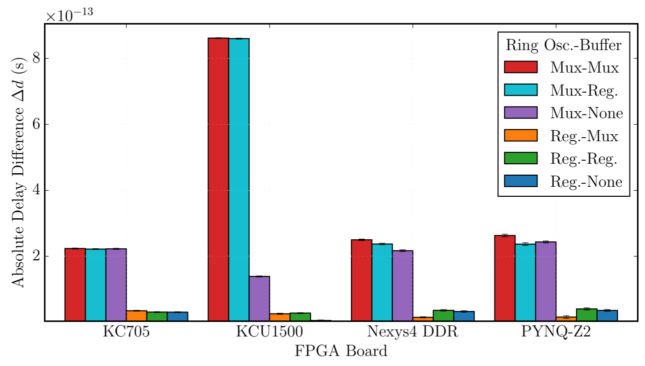

**Figure 7: Maximum information leakage in the three inverters setup for different routing combinations of the RO and buffer LUT outputs.**

Figure 7 demonstrates that the vulnerability is much smaller (to virtually non-existent) when the ring oscillator uses the regular output, but remains broadly the same (with the exception of the KCU1500 board) regardless of the buffer output, if any. In other words, if the wires in an attacker's measurement RO get routed through the MUX outputs (manually, or automatically by the tools, as explained in Section 6), then the adversary can infer the input to a nearby LUT, whether that LUT outputs a signal or not. As the information leakage is more pronounced on the Kintex UltraScale board and behaves differently when removing the buffer output, we investigate it more extensively in Section 5.4.

## 5.4 Kintex UltraScale Analysis

The final set of experiments measures information leakage in all possible RO and buffer LUT placements on a CLB. Specifically, we fix $RO_2$ and $RO_3$ in a CLB, completely remove the second buffer LUT $BUF_2$, and test all $8 \cdot 7 = 56$ possible combinations of $(RO_1, BUF_1)$ placements on LUTs in an adjacent CLB. Figure 8 plots the signed absolute delay difference for all setups in the two buffers, one inverters setup, with negative (respectively positive) values denoting that the RO counts are higher when the input to the buffer is value 0 (respectively 1).

Figure 8 allows us to draw three main conclusions. First, for any possible buffer location, there exists at least one RO location that can infer its value. For instance, when the buffer is on LUT H, an RO stage on LUT G can detect its value. Moreover, some buffer locations are more "leaky" (e.g., LUT A), and can be detected from multiple RO locations. Second, some RO locations (e.g., LUT H) can detect buffer values on more locations than others. Third, there does not seem to be a clear pattern for the information leakage, or even a pattern describing when a buffer input of 1 causes the RO frequency to increase vs. decrease. However, unlike the results of Figure 5, no "inversions" are possible in this setup: the results using the three inverters setup are nearly identical, with no changes in the direction (sign) of leakage.

One curious aspect of Figure 8 is the disproportionately large values when the RO logic is placed on LUTs E or F, with the buffer on LUT A. To further determine the cause of this discrepancy, we repeated the above experiment in the three-inverters setup, when

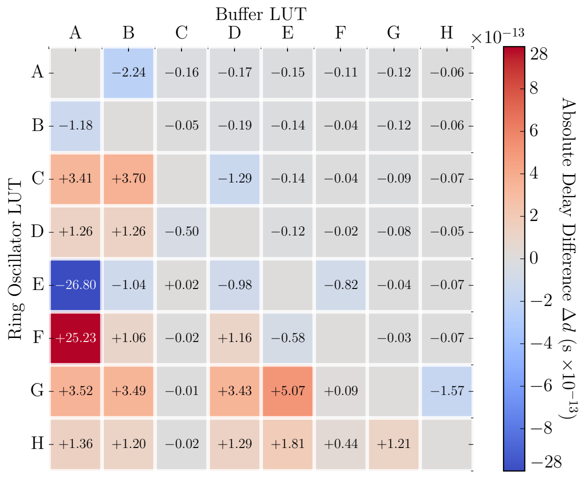

**Figure 8: Information leakage for all possible instantiations of the RO and buffer LUTs in the two buffers, one inverter setup, without a second buffer stage.**

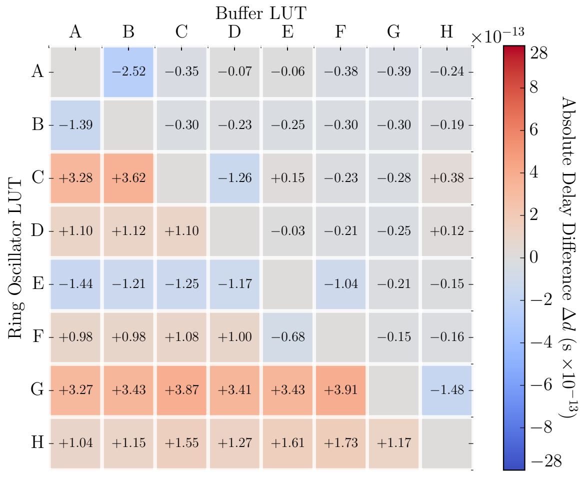

**Figure 9: Information leakage for all possible instantiations of the RO and buffer LUTs in the three inverters setup, with a second victim buffer stage.**

adding back the second buffer LUT in the victim logic. The results are plotted in Figure 9 with the same scale as Figure 8. The results are generally similar both in terms of which pairs of LUTs leak information, but also in terms of the magnitude and sign of the information leakage. However, the two outliers have disappeared.

The reason for this disparity comes down to a subtle-yet-crucial routing decision that the tools took in the first setup: Vivado also routed the buffer signal to the EX CLB wire (see AX in Figure 1) in both outliers, which also appears as an input to the multiplexer, among other elements. The ensuing leakage due to these logic inputs is almost 10× stronger than the (already strong) leakage detailed in this section. An example of this second source of multiplexer-based information leakage is presented in Figure 10, for ROs with

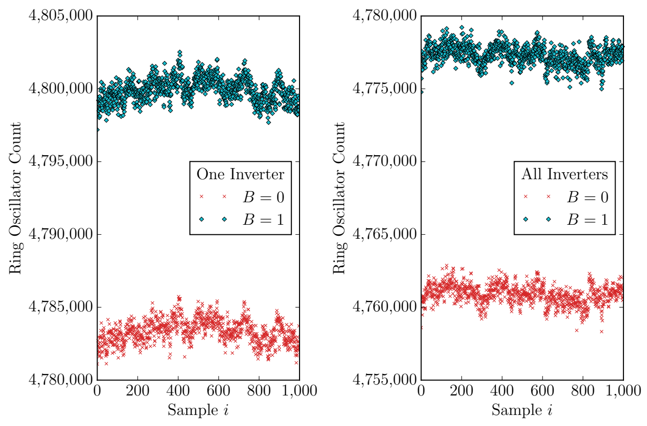

**Figure 10: Information leakage on the Kintex UltraScale due to the EX multiplexer input with one inverter and two buffers (left) or three inverters (right).**

three inverters as well as one inverter and two buffer setups. As explained above, as only one RO LUT is on the same CLB as the buffer LUT in this experimental setup, the "direction" (sign) of the information leakage remains the same for both measurements. It also highlights that these sources of vulnerability can be unintentionally revealed due to routing tool choices, a fact which is further discussed in Section 6.

## 6 DISCUSSION

In this section, we first discuss some aspects of the practicalities involved in exploiting the vulnerabilities discovered (Section 6.1), potential causes of the information leakage (Section 6.2), and then propose some countermeasures (Section 6.3).

### 6.1 Exploiting the Vulnerability

The new sources of information leakage introduced in this work require the routing of signal resources in adjacent channels or within the switch matrices of the same CLB. It is further worth repeating that the adversarial and victim designs do not share any logic resources, as the LUTs and multiplexers used are distinct.

Although it would be unusual for the adversary and the victim to be placed on the same CLB, routing through other pblocks is possible since the tools (Vivado) do not provide an `EXCLUDE_ROUTING` constraint, as explained in the introduction. Moreover several existing proposals for virtualization of the FPGA resources do not require physically isolating the logic of different users [1, 2, 28], hence making the attacks of this paper even easier in practice.

In addition, it is worth highlighting that the routing tools often activate several programmable interconnect points (PIPs) leading to different wires, even if the those wires are not actually used by the user logic. Indeed, as explained in Section 5.4, Vivado inadvertently routed the buffer value to the EX input, which ended up revealing an even stronger source of information leakage. We thus expect that dense designs might be particularly vulnerable as tools will be required to make unconventional routing choices.

Finally, another interesting aspect of the specific sources of information leakage is that they depend on static, short-lived nearby signals, and are thus not easily disturbed by dynamic activity. Indeed, Giechaskiel et al. showed that large nearby adders [7, 8] or cloud shells [9] cannot prevent long-wire leakage from being exploitable. This has implications for potential defense mechanisms, which are discussed in Section 6.3.

### 6.2 Potential Information Leakage Causes

Despite numerous works on long-wire leakage [7–9, 19–21, 24], its cause remains elusive without access to proprietary FPGA layout and manufacturing information. The same is true of the new sources of vulnerability identified in this work, but we briefly discuss some insights that arose from our investigations.

First of all, our discovery of information leakage due to medium wires reveals electrical differences between long and medium wires: long wires leak more information per-wire segment than medium wires. If the phenomenon is indeed caused by capacitive coupling as prior work has theorized, these differences could be explained by fewer *RC* parasitics, potentially resulting from changes in the diameter, distances, or metal layers for different types of wires. It is further worth noting that the lack of an intermediate tap in medium wires also appears to be significant: Virtex 5 long wires (which had two intermediate buffers) resulted in the most pronounced information leakage in prior work [8], compared to long wires with one buffer in subsequent FPGA generations and medium wires with no intermediate taps.

The source of the multiplexer leakage is not as clear, though it is important to note that the results of Section 5.4 suggest that it is not due to capacitive coupling. High ground bounce, poor substrate tie-offs, or other sources of static leakage could potentially explain some of the issues identified in this work, but much like prior work, to conclusively determine the root cause of the leakage, further simulation and experimentation is needed with the aid of confidential FPGA chip information.

### 6.3 Countermeasures

A number of defenses have been proposed to prevent information leaks in FPGAs. Some of them focus on limiting static information leakage from long wires, while others aim to detect voltage-related attacks due to dynamic switching activity. However, no work has specifically focused on information leakage due to medium wires or logic elements where the measured value remains constant.

Countermeasures for long-wire leakage prevent security-critical wires from being routed adjacent to attacker wires [7, 8]. HILL [19] is such a framework implementing the isolation of security-sensitive long wires. Based on user input, HILL places and routes modules so that they do not use long wires that could be possibly exposed to attackers. For designs that cannot avoid long wires (e.g., ones connecting to external I/O), HILL reserves the wires which are adjacent to the security-critical long wires and sets them to constant or random values. Another solution is a modified PathFinder [22] algorithm that considers different sensitive and non-sensitive nets (as labeled by the designers) during the routing computations [24], and ensures they are not routed next to each other. These approaches do not account for medium-wire leakage, and are vulnerable to novel sources of routing information leakage, such as the ones of this work. As a result, we recommend that tools *prevent routing any wires from untrusted sources next to sensitive wires, independent of the wire type or length.*

A more coarse-grained proposal to isolation is to partition the FPGA into non-overlapping regions and to create buffer zones called *moats* between them [13, 14]. Routing is disabled in the moats, preventing crosstalk among different modules, except possibly for the input and output wires that need to leave the moat region. In a similar approach, tools could wrap the user logic with *soft shells* [35]. The soft shells encrypt data going out of user logic, preventing crosstalk attacks even for inputs and outputs. To be effective, these two schemes would need to be combined: the former to ensure the security of incoming and outgoing wires, and the latter to ensure that non-encrypted wires inside the user logic are never adjacent to potential attacker wires. Although physical isolation can prevent crosstalk attacks, it significantly constraints how the designs are routed and has an impact on area and performance overhead. Instead, for static information leakage, we propose that a more fine-grained approach should be preferred which *prohibits sharing any CLB resources (logic elements, or routing elements in the CLB's switch matrix), if any of the CLB's inputs are from mutually untrustworthy sources.* That said, neither approach prevents information leakage due to *dynamic* activity when the attacker and victim are: (a) on different dies within the same chip [10], (b) different FPGAs on the same board [23], or (c) even different boards with a shared power supply [11]. Such attacks require improvements in the voltage regulation circuits, or hiding the useful signal under artificially-introduced random noise.

It should be noted that some proposals call for trusted bitstreams through message authentication codes and encryption [5]. However, such approaches assume that design tools are able to find malicious circuits to prevent attacks. As an example, previous works have proposed prohibiting ring oscillators and other structures from being instantiated [18]. Nevertheless, recent ring oscillators designs using latches or flip-flops can bypass these design rule checks [9, 25], and suggest that banning specific types of logic may not be effective as a defense mechanism.

Finally, changes at the hardware level are necessary, and include *routing different wires on different metal layers in the FPGA chip and increasing the spacing between them in a routing channel.* However, as implementing the changes in the underlying hardware can take years to deploy, current tools must be modified to account for existing and prospective sources of information leaks during the design or programming time.

## 7 CONCLUSION

This work presented two new classes of information leaks that are present in four families of Xilinx FPGAs. First, it evaluated *long*-wire leakage in two families which had not been investigated in prior work. It then identified an additional type of routing resource, *medium* wires, whose crosstalk behavior is similar to that of its longer counterpart, and can thus be used to extract sensitive information from FPGAs. This work showed that even if one accounts for differences in the lengths of the wires, long-wire leakage is stronger than medium-wire leakage, suggesting that the two types of wires are not the same physically or electrically. This paper further introduced and analyzed a novel source of information leakage that originates from the elements within a Configurable Logic Block (CLB). It showed that the delays of elements using multiplexer (MUX) outputs are sensitive to the values of nearby LUTs. As a result, an adversary exploiting this fact can use a ring oscillator to detect changes in the inputs and outputs of victim logic. As current tools do not prevent the adjacent routing of resources from different users, this work introduced two new attack vectors that need to be considered in the context of single-tenant designs with untrusted IPs, or multi-tenant FPGAs—especially ones that do not enforce physical isolation of logic placement. As preventing the underlying issues in hardware is a years-long process, the software itself must account for potential vulnerabilities due to information leaks by prohibiting the sharing of logic and routing resources if *any* of the inputs or outputs come from untrusted sources, independent of the wire type or length.

## ACKNOWLEDGMENTS

This work was supported in part by NSF grants 1716541 and 1901901.

## REFERENCES

[1] Stuart Byma, J. Gregory Steffan, Hadi Bannazadeh, Alberto L. Garcia, and Paul Chow. 2014. FPGAs in the Cloud: Booting Virtualized Hardware Accelerators with OpenStack. In *Field-Programmable Custom Computing Machines (FCCM).*

[2] Fei Chen, Yi Shan, Yu Zhang, Yu Wang, Hubertus Franke, Xiaotao Chang, and Kun Wang. 2014. Enabling FPGAs in the Cloud. In *ACM Conference on Computing Frontiers (CF).*

[3] Digilent, Inc. 2016. Nexys4 DDR FPGA Board Reference Manual. https://reference.digilentinc.com/_media/reference/programmable-logic/nexys-4-ddr/nexys4ddr_rm.pdf. Accessed: 2020-05-24.

[4] Esam El-Araby, Ivan Gonzalez, and Tarek El-Ghazawi. 2008. Virtualizing and Sharing Reconfigurable Resources in High-Performance Reconfigurable Computing Systems. In *International Workshop on High-Performance Reconfigurable Computing Technology and Applications (HPRCTA).*

[5] Rana Elnaggar, Ramesh Karri, and Krishnendu Chakrabarty. 2019. Multi-Tenant FPGA-Based Reconfigurable Systems: Attacks and Defenses. In *Design, Automation & Test in Europe Conference & Exhibition (DATE).*

[6] Martin Gag, Tim Wegner, Ansgar Waschki, and Dirk Timmermann. 2012. Temperature and On-Chip Crosstalk Measurement using Ring Oscillators in FPGA. In *IEEE Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS).*

[7] Ilias Giechaskiel, Ken Eguro, and Kasper B. Rasmussen. 2019. Leakier Wires: Exploiting FPGA Long Wires for Covert- and Side-Channel Attacks. *ACM Transactions on Reconfigurable Technology and Systems (TRETS)* 12, 3 (Sept. 2019), 1–29.

[8] Ilias Giechaskiel, Kasper B. Rasmussen, and Ken Eguro. 2018. Leaky Wires: Information Leakage and Covert Communication Between FPGA Long Wires. In *ACM ASIA Conference on Computer and Communications Security (ASIACCS).*

[9] Ilias Giechaskiel, Kasper B. Rasmussen, and Jakub Szefer. 2019. Measuring Long Wire Leakage with Ring Oscillators in Cloud FPGAs. In *International Conference on Field Programmable Logic and Applications (FPL).*

[10] Ilias Giechaskiel, Kasper B. Rasmussen, and Jakub Szefer. 2019. Reading Between the Dies: Cross-SLR Covert Channels on Multi-Tenant Cloud FPGAs. In *IEEE International Conference on Computer Design (ICCD).*

[11] Ilias Giechaskiel, Kasper B. Rasmussen, and Jakub Szefer. 2020. C$^3$APSULe: Cross-FPGA Covert-Channel Attacks through Power Supply Unit Leakage. In *IEEE Symposium on Security and Privacy (S&P).*

[12] Ali Hajimiri, Sotirios Limotyrakis, and Thomas H. Lee. 1999. Jitter and Phase Noise in Ring Oscillators. *IEEE Journal of Solid-State Circuits (JSSC)* 34, 6 (June 1999), 790–804.

[13] Ted Huffmire, Brett Brotherton, Nick Callegari, Jonathan Valamehr, Jeff White, Ryan Kastner, and Tim Sherwood. 2008. Designing Secure Systems on Reconfigurable Hardware. *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 13, 3 (July 2008), 1–24.

[14] Ted Huffmire, Brett Brotherton, Gang Wang, Timothy Sherwood, Ryan Kastner, Timothy Levin, Thuy Nguyen, and Cynthia Irvine. 2007. Moats and Drawbridges: An Isolation Primitive for Reconfigurable Hardware Based Systems. In *IEEE Symposium on Security and Privacy (S&P).*

[15] Ahmed Khawaja, Joshua Landgraf, Rohith Prakash, Michael Wei, Eric Schkufza, and Christopher J. Rossbach. 2018. Sharing, Protection, and Compatibility for Reconfigurable Fabric with AMORPHOS. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*.

[16] Oliver Knodel, Paul R. Genssler, Fredo Erxleben, and Rainer G. Spallek. 2018. FPGAs and the Cloud – An Endless Tale of Virtualization, Elasticity and Efficiency. *International Journal on Advances in Systems and Measurements* 11, 3-4 (2018), 230–249.

[17] Oliver Knodel, Paul R. Genssler, and Rainer G. Spallek. 2017. Virtualizing Reconfigurable Hardware to Provide Scalability in Cloud Architectures. In *International Conference on Advances in Circuits, Electronics and Micro-Electronics (CENICS)*.

[18] Jonas Krautter, Dennis R. E. Gnad, and Mehdi B. Tahoori. 2019. Mitigating Electrical-level Attacks towards Secure Multi-Tenant FPGAs in the Cloud. *ACM Transactions on Reconfigurable Technology and Systems (TRETS)* 12, 3 (Sept. 2019), 1–26.

[19] Yukui Luo and Xiaolin Xu. 2019. HILL: A Hardware Isolation Framework Against Information Leakage on Multi-Tenant FPGA Long-Wires. In *International Conference on Field-Programmable Technology (FPT)*.

[20] George Provelengios, Chethan Ramesh, Shivukumar B. Patil, Ken Eguro, Russell Tessier, and Daniel Holcomb. 2019. Characterization of Long Wire Data Leakage in Deep Submicron FPGAs. In *ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA)*.

[21] Chethan Ramesh, Shivukumar B. Patil, Siva N. Dhanuskodi, George Provelengios, Sébastien Pillement, Daniel Holcomb, and Russell Tessier. 2018. FPGA Side Channel Attacks without Physical Access. In *IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM)*.

[22] Jonathan Rose, Jason Luu, Chi Wai Yu, Opal Densmore, Jeffrey Goeders, Andrew Somerville, Kenneth B. Kent, Peter Jamieson, and Jason Anderson. 2012. The VTR Project: Architecture and CAD for FPGAs from Verilog to Routing. In *ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA)*.

[23] Falk Schellenberg, Dennis R. E. Gnad, Amir Moradi, and Mehdi B. Tahoori. 2018. Remote Inter-Chip Power Analysis Side-Channel Attacks at Board-level. In *International Conference on Computer-Aided Design (ICCAD)*.

[24] Zeinab Seifoori, Seyedeh Sharareh Mirzargar, and Mirjana Stojilović. 2020. Closing Leaks: Routing Against Crosstalk Side-Channel Attacks. In *ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA)*.

[25] Takeshi Sugawara, Kazuo Sakiyama, Shoei Nashimoto, Daisuke Suzuki, and Tomoyuki Nagatsuka. 2019. Oscillator without a Combinatorial Loop and its Threat to FPGA in Data Centre. *Electronics Letters* 15, 11 (May 2019), 640–642.

[26] TUL Corporation. 2019. PYNQ-Z2 Reference Manual. http://www.tul.com.tw/download/PYNQ_Z2_User_Manual_v1.1.pdf. Accessed: 2020-05-24.

[27] Anuj Vaishnav, Khoa D. Pham, and Dirk Koch. 2018. A Survey on FPGA Virtualization. In *International Conference on Field Programmable Logic and Applications (FPL)*.

[28] Jagath Weerasinghe, François Abel, Christoph Hagleitner, and Andreas Herkersdorf. 2015. Enabling FPGAs in Hyperscale Data Centers. In *IEEE International Conference on Ubiquitous Intelligence and Computing, Autonomic and Trusted Computing, Scalable Computing and Communications (UIC-ATC-ScalCom)*.

[29] Xilinx, Inc. 2016. 7 Series FPGAs Configurable Logic Block: User Guide (UG474). https://www.xilinx.com/support/documentation/user_guides/ug474_7Series_CLB.pdf. Accessed: 2020-05-24.

[30] Xilinx, Inc. 2017. UltraScale Architecture Configurable Logic Block: User Guide (UG574). https://www.xilinx.com/support/documentation/user_guides/ug574-ultrascale-clb.pdf. Accessed: 2020-05-24.

[31] Xilinx, Inc. 2018. 7 Series FPGAs Data Sheet: Overview (DS180). https://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf. Accessed: 2020-05-24.

[32] Xilinx, Inc. 2018. KCU1500 Board: User Guide (UG1260). https://www.xilinx.com/support/documentation/boards_and_kits/kcu1500/ug1260-kcu1500-data-center.pdf. Accessed: 2020-05-24.

[33] Xilinx, Inc. 2018. Zynq-7000 SoC Data Sheet: Overview (DS190). https://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf. Accessed: 2020-05-24.

[34] Xilinx, Inc. 2019. KC705 Evaluation Board for the Kintex-7 FPGA: User Guide (UG810). https://www.xilinx.com/support/documentation/boards_and_kits/kc705/ug810_KC705_Eval_Bd.pdf. Accessed: 2020-05-24.

[35] Sadegh Yazdanshenas and Vaughn Betz. 2019. The Costs of Confidentiality in Virtualized FPGAs. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems (TVLSI)* 27, 10 (Oct. 2019), 2272–2283.