Cross-board Power-Based FPGA, CPU, and GPU Covert Channels

Ilias Giechaskiel, Kasper Rasmussen, and Jakub Szefer

7.1 Introduction

Field-programmable gate arrays (FPGAs) have become increasingly popular in cloud deployments, and this transition has also resulted in a threat model shift from one of physical attacks that require physical proximity to the FPGA board and external equipment (e.g., high-end oscilloscopes) to one of remote attacks using only on-chip logic. The majority of recent work has so far shown that remote fault, covert-channel, and side-channel attacks are indeed possible between designs belonging to different users co-located within the same FPGA chip [4, 8–10, 13, 24, 26, 27, 29, 30, 32, 37, 45, 46]. However, as boards are currently allocated on a per-user basis in commercial clouds, this multi-tenant threat model remains theoretical, with little practical impact.

In this chapter, we instead tackle a more pressing scenario that is applicable to existing cloud FPGA deployments, where boards are co-located within the same server rack unit. Users renting FPGAs from such FPGA cloud providers assume that their designs are safely isolated from potentially malicious designs by other users running in the same data center. However, as we show, the assumption of isolation can be broken due to leakage through the shared use of power supply units (PSUs). Specifically, we introduce a new class of remote covert-channel attacks between

Ilias Giechaskiel
Independent Researcher, London, UK
e-mail: ilias@giechaskiel.com

Kasper Rasmussen
University of Oxford, Oxford, UK
e-mail: kasper.rasmussen@cs.ox.ac.uk

Jakub Szefer
Yale University, New Haven, CT, USA
e-mail: jakub.szefer@yale.edu

single-tenant FPGAs on different FPGA boards that are merely powered through the same PSU. Moreover, we show that if this PSU also powers the host computer, the same sink FPGA (receiver) can detect high levels of CPU and GPU activity, creating new CPU-to-FPGA and GPU-to-FPGA channels. These channels allow one system, which may (GPU, FPGA) or may not (CPU) contain an accelerator, to leak information such as private encryption keys to an entirely different system (the sink FPGA), which is fully isolated, except for the shared power supply.

The first crucial observation of our work is that although causing variable power consumption to transmit information is easy, detecting voltage fluctuations without external equipment is non-trivial. However, the reconfigurability of FPGAs provides access to the hardware at a much lower level and can be used to implement circuits that detect voltage changes that are imperceptible to fixed silicon chips such as CPUs and GPUs. Indeed, cloud providers are aware of the impact of such low-level hardware access, so besides allocating FPGAs on a per-user basis, they also keep several features such as voltage and temperature monitors inaccessible to end users.

The second key observation is that ring oscillators (ROs) are capable of both causing and sensing voltage fluctuations. This chapter therefore introduces a novel way of monitoring changes in voltage caused by the source FPGA, CPU, or GPU. Specifically, both properties of ROs are used in the sink (receiver) FPGA, whereby stressing the voltage regulator of the sink FPGA allows one to detect transmissions by the source (transmitter) FPGA.

Using these insights, we demonstrate the first cross-FPGA covert channel between off-the-shelf, unmodified Xilinx Artix 7, and Kintex 7 boards in either direction of communication. We also characterize the bandwidth–accuracy tradeoffs across different measurement periods and sizes of the covert-channel ROs on the source and sink FPGAs. We further test our covert channel on two PSUs running under normal operating conditions (i.e., without being overloaded) and introduce CPU-to-FPGA and GPU-to-FPGA covert channels by modulating their respective loads. We finally discuss countermeasures to mitigate this source of leakage.

7.1.1 Contributions

Our contributions can be summarized as follows:

1. We identify sharing of PSUs as a new source of vulnerability, even for unprivileged FPGA designs without access to voltage or temperature system monitors.
2. We introduce a novel measurement setup and classification metric that uses ring oscillators (ROs) on the sink FPGA to stress its voltage regulator and therefore reliably detect external voltage fluctuations.
3. We exploit this setup to create the first remote covert-channel attack between FPGAs on distinct physical boards that are dedicated on a per-user basis, reaching accuracies of up to 100%.

4. We evaluate the strength of the information leakage across different architectural choices and perform a bandwidth–accuracy tradeoff analysis.
5. We introduce the first CPU-to-FPGA and GPU-to-FPGA covert channels using high loads of activity on their respective processors, opening up new avenues for remote FPGA attacks.
6. We propose hardware- and software-level countermeasures to reduce the impact of the leakage.

7.1.2 Chapter Organization

The rest of the chapter is organized as follows. Section 7.2 introduces the threat model, while Sect. 7.3 details the experimental setup, including hardware properties, the measurement procedure, and the high-level architectural FPGA design. Section 7.4 then describes the need for our novel classification metric and explains why it works where the naive approach of looking at absolute ring oscillator counts fails. Section 7.5 then evaluates cross-FPGA covert communication over shared PSUs, varying the number of source and sink ring oscillators used, and performing an analysis of bandwidth–accuracy tradeoffs. Section 7.6 then covers CPU-to-FPGA and GPU-to-FPGA information leakage, while Sect. 7.7 discusses potential defense mechanisms. We place our work in the context of related research in Sect. 7.8, before we conclude in Sect. 7.9.

7.2 Threat Model

Prior work on attacks without physical access to the FPGA hardware has primarily investigated security in the context of multi-tenant FPGAs. It has shown that when a single FPGA chip is shared among multiple users concurrently, designs are vulnerable to temperature and voltage attacks (Sect. 7.8). Although these attacks highlight potential issues with future architectures, they remain theoretical at the moment, as FPGAs are currently allocated on a per-user basis. In this chapter, we are thus concerned with covert-channel attacks against platforms where the entire logic is allocated to a single user. Design logic therefore cannot access any voltage or thermal system monitors present on the FPGA fabric, as these inaccessible in a cloud environment.[1] Compared to multi-tenant attacks on FPGA designs that share the same power distribution network, adversarial attacks to infer any information about the activity or data (e.g., encryption keys) of other users necessitate that side-

[1] In cloud FPGAs, part of the fabric is reserved by a cloud-provided "shell" that hides implementation details, including physical pinouts, identification primitives, and system monitors. User logic is forced to interact with external hardware through the shell's AXI4 interfaces.

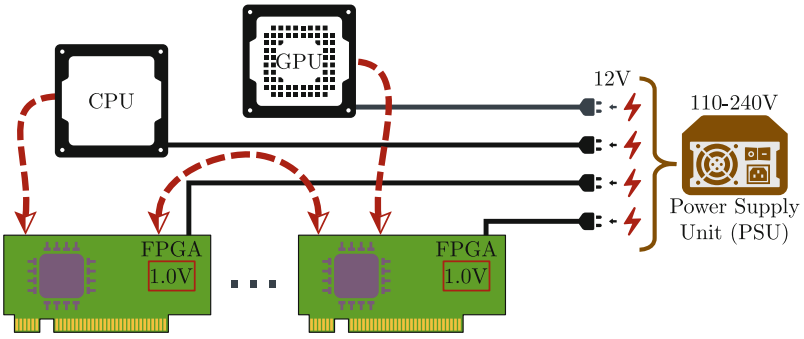


Fig. 7.1 System model for FPGA-to-FPGA, CPU-to-FPGA, and GPU-to-FPGA leakage in co-located environments. The CPU, GPU, and one or more (potentially malicious) FPGAs are powered through the same PSU but do not share any logic and do not have access to system monitors for measuring voltage or temperature changes

channel leakage be measurable across extensive physical separation (as opposed to logic on the same FPGA chip), and with multiple intermediate components (passive capacitors, inductors, voltage regulators, etc.) on the path between the source and sink FPGA boards.

In this chapter, we specifically investigate remote voltage-based attacks, where a shared PSU provides an indirect connection between FPGA boards. We do not consider reverse-engineering attacks on the bitstream itself or the contained logic, but instead focus on how to initiate a communication channel through modulating the load on the PSU itself. We mainly consider FPGA-to-FPGA attacks between otherwise unconnected devices, but also investigate CPU-to-FPGA and GPU-to-FPGA attacks. This is because the same PSU might also power the host computer, and, by extension, its internal components including CPUs and GPUs, as shown in the high-level system model of Fig. 7.1. We make no assumptions regarding how the FPGAs are connected to the computer. In other words, we do not assume that FPGAs are attached to the motherboard over PCIe, to a USB controller over a serial chip, or, in fact, if they are even (logically) connected to the computer at all. Our only assumption is that of a shared PSU between the two communicating parties.

Within an FPGA, and in accordance with prior work [9, 10, 27, 46], (potentially adversarial) users can place and route any designs of their choice, such as different types of ring oscillators. This is allowed by current FPGA cloud deployments, as long as the logic is placed outside of the cloud-provided shell. In this chapter, we show that by relying only on on-chip FPGA logic (i.e., ring oscillators), we are able to demonstrate FPGA-to-FPGA, CPU-to-FPGA, and GPU-to-FPGA covert communication, without physical access to the FPGA boards. One of the key contributions of our work is therefore the ability to communicate across unmodified devices, without external equipment or access to internal voltage monitors, which are off-limits to unprivileged FPGA designs.

It should be noted that some cloud providers such as Amazon Web Services (AWS) place restrictions on the types of circuits that can be instantiated on their FPGAs and prohibit combinatorial loops including ring oscillators [9, 35]. Although in this chapter we primarily use conventional ring oscillators, Sect. 7.5.5 shows that they can be easily replaced by alternate designs proposed in recent work [9, 10, 22, 35], which bypass such cloud countermeasures, and could therefore be used to attack the isolation mechanisms that separate physical hardware is supposed to provide.

7.3 Experimental Setup

In this section, we detail our experimental setup, starting with the ring oscillators employed in the source and sink FPGAs (Sect. 7.3.1) and delving into the architectural design of the FPGA transmission and reception circuitry (Sect. 7.3.2). We then describe the hardware properties of the FPGA boards used (Sect. 7.3.3), as well as the computer PSUs, CPUs, and GPUs, which are effectively turned into covert-channel transmitters (Sect. 7.3.4). We finally discuss the process followed for data collection (Sect. 7.3.5).

7.3.1 Ring Oscillators

Ring oscillators are comprised of an odd number of NOT gates in a ring formation and therefore form a combinatorial loop, whose value oscillates. The frequency of oscillation changes based on process variations, as well as voltage and temperature conditions [16], making ROs good temperature [38] and voltage [46] monitors. ROs also cause voltage fluctuations, which stress power circuits, and can potentially crash the FPGA or inject faults [12, 24, 26, 27, 30].

In this chapter, we use ROs as both transmitters and receivers and implement them using lookup tables (LUT-RO) with one inverter and three buffer stages as shown in Fig. 7.2. We chose to use this RO design instead of more common ROs with three inverters or one inverter and two buffer stages because preliminary experiments showed that they resulted in more stable measurements. Alternative types of ROs are evaluated in Sect. 7.5.5.

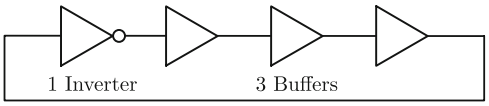


Fig. 7.2 The ring oscillators are implemented using lookup tables (LUT-ROs) and contain one inverter and three buffer gates

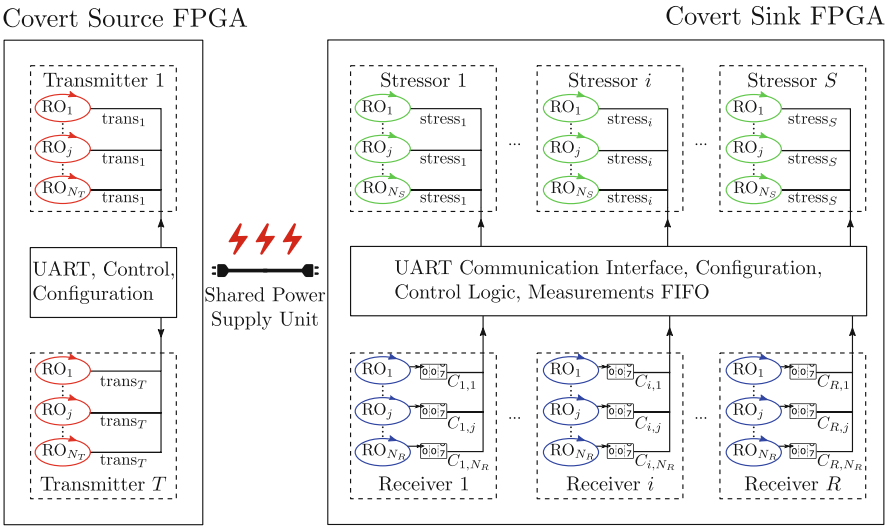


Fig. 7.3 Experimental setup: the covert source (left) uses $T \cdot N_T$ ROs, while the sink (right) has $R \cdot N_R$ measurement ROs and $S \cdot N_S$ stressor ROs. The same power supply unit powers both boards

7.3.2 Architectural FPGA Design

We now give a high-level overview of the covert-channel source and sink FPGA designs, which are summarized in Fig. 7.3.

7.3.2.1 Covert-Channel Source

To cause detectable changes on the sink, the source FPGA employs ring oscillators organized as T *transmitters*, which can be controlled independently. These transmitters are placed on separate clock regions to make power consumption more evenly spread throughout the FPGA. They contain N_T ROs each, for a total of $T \cdot N_T$ ROs, as shown in the left part of Fig. 7.3.

7.3.2.2 Covert-Channel Sink

To receive transmissions, we employ R *receivers*, placed on separate clock regions of the sink FPGA, and each containing N_R ROs. We estimate the RO frequency by counting the number of RO signal transitions in a fixed measurement interval of 2^t clock cycles through counters placed outside of the RO clock regions.

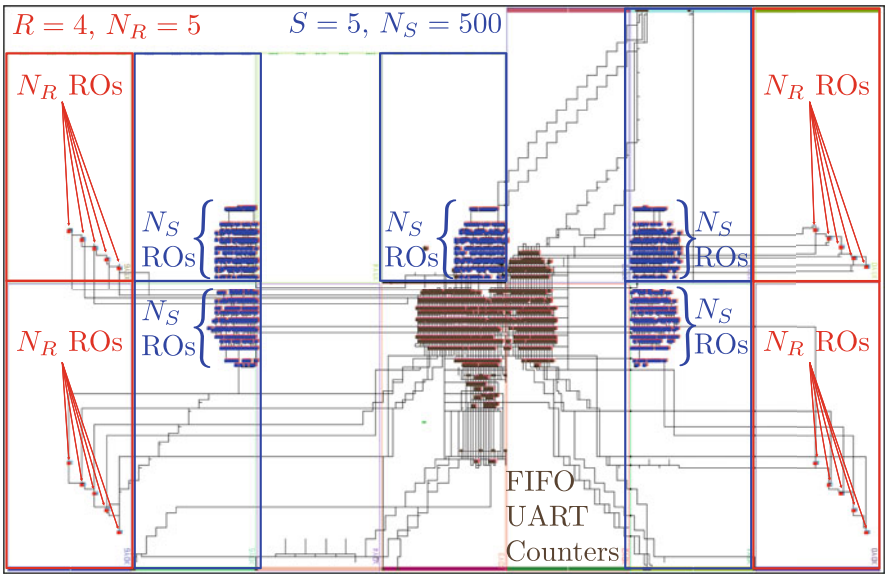


Fig. 7.4 Annotated Vivado screenshot of the sink architecture on the Kintex 7 board, with receiver ROs in red, stressor ROs in blue, and other logic (counters, UART, FIFO) in brown

However, this setup is not sufficient to decode covert transmissions, due to inherent noise in the power supply and environmental fluctuations. Instead, it is necessary to introduce additional circuitry on the sink FPGA that stresses the board's voltage regulator, making maintaining a constant voltage harder. This fact allows us to sense voltage changes induced by the source FPGA, or even by CPU and GPU activity, as presented later in Sect. 7.6. Specifically, we include S *stressors*, each with N_S ROs. As with the source transmitters, these S stressors are placed on separate clock regions and can also be controlled independently. The block diagram for the sink design is shown in the right part of Fig. 7.3, while Fig. 7.4 shows a concrete instantiation of the sink architecture on the Kintex 7 board. Section 7.4 further demonstrates the need for stressor ROs.

7.3.3 FPGA Boards

For our experiments, we use Xilinx Kintex 7 KC705 and Artix 7 AC701 boards. The 28 nm chips these devices contain are similar, but the Kintex 7 is more performant, while the Artix 7 is optimized for low power [41, 44]. Both FPGAs have a 200 MHz oscillator and operate at a core `VCCINT` voltage of 1.0 V, but the boards use different regulators to convert the 12 V PSU output into 1.0 V [42, 43].

Table 7.1 Properties of the FPGA boards used, along with fixed compile-time choices for the source and sink circuit configurations

Property	Artix 7	Kintex 7
Board	AC701	KC705
Part Number	XC7A200T	XC7K325T
Slices	33 650	50 950
Clock Regions	2×5	2×7
Core Voltage, `VCCINT`	1.0 V	1.0 V
Voltage Regulator	LMZ31710	PTD08A020W
Clock Frequency	200 MHz	200 MHz
# of Boards Tested	2	2
# of Transmitters, T	10	14
# of Stressors, S	5	5
# of Receivers, R	4	4
# of ROs per Receiver, N_R	5	5

For the source FPGA designs, we place a transmitter on each clock region of the FPGA. As the Artix 7 board has 10 clock regions, while the Kintex 7 has 14, the numbers of transmitters on these devices are $T = 10$ and $T = 14$, respectively. The sink FPGAs contain $R = 4$ receivers in the corners of each chip, each with $N_R = 5$ ROs. Sink FPGAs also contain $S = 5$ stressors, one of which is placed in the center of the device, while the remaining four are next to the receiver clock regions (Fig. 7.4 shows an example with $N_S = 500$). Although not shown to be significant in our experiments, these early architectural choices were made to ensure that the power draw was approximately equally spread across the FPGA fabric.

These decisions and other FPGA properties are summarized in Table 7.1. More compile- and run-time parameters, such as the measurement period and the number of source transmitters ROs N_T and sink stressor ROs N_S, are varied in Sect. 7.5.

7.3.4 Power Supply Units and Computer Transmitters

To verify that the covert channel is not due to faulty design in a line of specific power supply units, we test communication on two PSUs made by different manufacturers (Corsair and Dell), rated for different loads (850 W and 1300 W, respectively), and both with a Gold 80 Plus Certification (which guarantees 90% efficiency at 50% load). These PSUs are integrated in two computers, the first of which contains two Xeon E5645 CPUs for a total of 24 threads, while the second contains a single Xeon E5-2609 with 4 threads. They also contain Nvidia GeForce GPUs, with 96 and 640 CUDA cores, respectively. The CPU and GPU cores are used as the covert-channel sources in Sect. 7.6 for CPU-to-FPGA and GPU-to-FPGA communication over the shared power supply. The properties of the computers used are summarized in Table 7.2.

Table 7.2 Hardware properties of the two computers used, with their corresponding PSUs, CPUs, and GPUs

Property	PC-A	PC-B
PSU Brand	Corsair	Dell
Power Rating	850 W	1300 W
80 Plus Certification	Gold	Gold
Motherboard	SuperMicro X8DAL-i	Dell Precision T7600
Xeon CPU Model	E5645	E5-2609
# of CPU Cores	6 @ 2.4 GHz	4 @ 2.4 GHz
# of Threads	12	4
# of CPUs	2	1
GeForce GPU	ZOTAC GT 430	EVGA GTX 750 Ti
GPU Memory	1 GB GDDR3	2 GB GDDR5
# of CUDA Cores	96 @ 0.7 GHz	640 @ 1.0 GHz

7.3.5 Data Collection and Encoding

For our data collection process, we made several choices to make the communication scenario realistic. For instance, the computers attached to the PSUs were used normally during experimentation, including running and installing other software. Moreover, to ensure leakage is not due to temperature, the FPGAs were placed outside the computer case, and away from computer fans, which may affect measurements by turning on or off based on the computer temperature. We similarly placed the FPGAs next to each other horizontally (as opposed to stacking them vertically), further minimizing cross-FPGA temperature effects. In addition, to control for other voltage effects, the FPGAs were not connected to the computer over PCIe, which would likely increase the potential for leakage. However, as we show in Sect. 7.5.5, our covert channel operates with similar accuracy, even when the FPGAs are connected to the computer over PCIe and are enclosed in it without accounting for temperature variations. Finally, to verify that the leakage is not caused through the UART interface, we often used one computer to take the measurements, and the other to power the source and sink boards through its PSU.

As there is inherent noise in the measurements, (a) the absolute RO frequency is not well-suited for comparison, and (b) the RO counts need to be averaged over repeated measurements to produce meaningful results. To address both concerns, we use Manchester-encoding, where to send a 1, the source transmitters are enabled for one measurement period and disabled for the next (a 0 is similarly encoded by first disabling transmitters during the first measurement period and enabling them in the second period). These measurement periods are $M \cdot 2^t$ clock cycles long, where we average M RO counts collected by ROs enabled for 2^t clock cycles (see Sect. 7.4). The bandwidth can thus be calculated as

$$b = \frac{f_c}{2 \cdot 2^t \cdot M}, \tag{7.1}$$

where $f_c = 200\,\text{MHz}$ is the FPGA clock frequency.

In most experiments, we transmit the 20-bit number `0xf3ed1` across the covert channel, Manchester-encoding it in 40 bits. Additional patterns are evaluated in Sect. 7.5.4. To ensure that perfect synchronization is not needed between the source and the sink, for each of the 40 periods, we take four sets of M measurements, where M is in the order of a few hundred counts (see Table 7.3 and Sect. 7.5.3). The four sets of repetitions create $4^2 = 16$ Manchester-encoded pairs per bit to be transferred, for a total of $16 \times 20 = 320$ pairs to estimate the covert-channel accuracy.

7.4 Classification Metric

This section introduces a novel methodology to detect changes in the power supply voltage through the sink's "stressor" ROs. Section 7.4.1 first motivates why the naive approach of using the absolute ring oscillator counts is insufficient for classification of transmissions in this scenario. Section 7.4.2 then introduces the metric using stressors, while Sect. 7.4.3 finally explains why our technique works.

7.4.1 Why Absolute Counts Are Not Enough

Broadly speaking, when the transmitters are activated on the source FPGA, CPU, or GPU, there is a voltage drop that is visible not just at the board regulator, but also at the 12 V rail PSU input to the FPGA board. Indeed, Fig. 7.5 demonstrates this

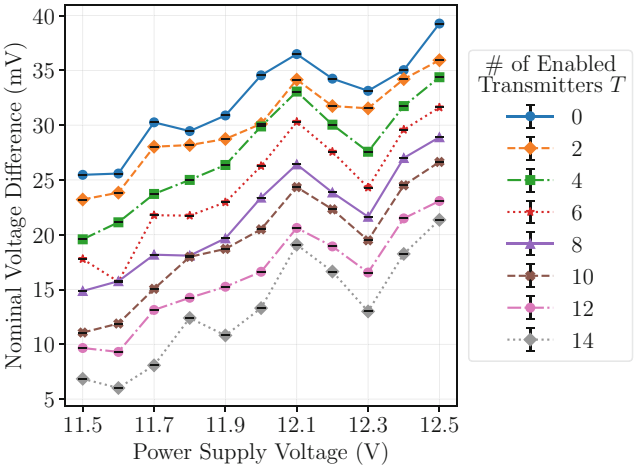


Fig. 7.5 Voltage as set by the power supply and measured by the oscilloscope for various numbers of enabled transmitters T on the KC705-2 source, with 99% confidence intervals

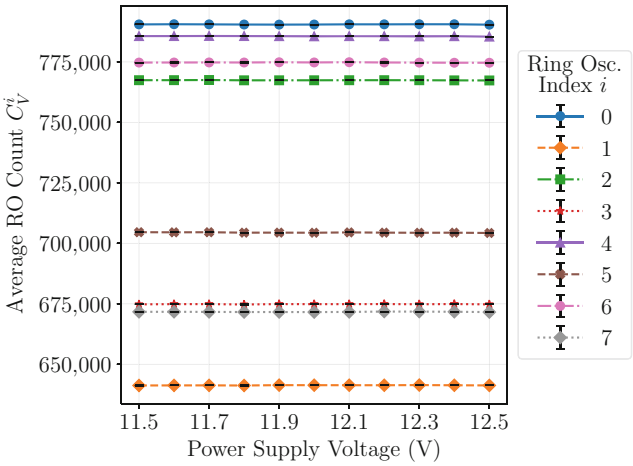


Fig. 7.6 The average ring oscillator counts C_V^i (at 99% confidence) on the AC701-1 sink remain approximately the same for different power supply voltages V and all eight ring oscillators R_i

for a Kintex 7 source without a sink FPGA present across multiple input voltages and different numbers of enabled transmitters T. Specifically, we power the board using a Keithley 2231A power supply and measure the voltage at the power rail of the board using a Tektronix MDO3104 Mixed Domain Oscilloscope with TPP1000 1 GHz passive probes, taking 10 000 data points. Figure 7.5 indicates that at any voltage level provided by the power supply (11.5 V to 12.5 V), as the number of enabled source transmitters T increases, the voltage measured by the oscilloscope decreases. For example, at 12.5 V, the oscilloscope measures 12.539 V when no transmitters are enabled, but only 12.521 V when 14 transmitters are enabled, for a voltage drop of approximately 18 mV. At 11.5 V, the measured voltage similarly drops from 11.525 V to 11.507 V.

Although one would expect RO frequency to increase with higher voltages [16], this is not the case. For a ring oscillator i, let its average count be C_V^i when the voltage provided by the power supply is $11.5\,\text{V} \leq V \leq 12.5\,\text{V}$. We would expect that $C_{V_1}^i > C_{V_2}^i$ whenever $V_1 > V_2$, but Fig. 7.6 suggests that the RO counts remain approximately the same for all eight ring oscillators and voltages V tested on an Artix 7 sink, likely because the regulator is able to deal with such input voltages. As a result, the absolute RO frequency cannot be used to decode cross-FPGA covert-channel transmissions.

7.4.2 A New Metric Based on Count Differences

To solve the issues identified above, we introduce ROs to "stress" the voltage regulator and make external changes in the power supply voltage measurable. For any bit transmission (say the i-th one), we take M measurements as follows:

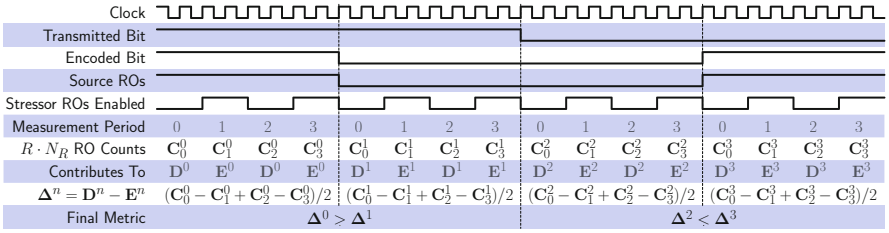


Fig. 7.7 Timing diagram for a Manchester-encoded transmission of the two bits `10`, with $M = 4$ measurement periods. Half of the ring oscillator counts are taken when the stressors are enabled ($\boldsymbol{E}$), and the other $M/2 = 2$ counts when they are disabled ($\boldsymbol{D}$) to compute $\boldsymbol{\Delta} = \boldsymbol{D} - \boldsymbol{E}$. The receiver uses the sign (positive or negative) of the difference $\boldsymbol{\Delta}^{2n} - \boldsymbol{\Delta}^{2n+1}$ between the two parts of the encoded transmission of the n-th bit to determine if it should be decoded as a 0 or as a 1. For example, $(\boldsymbol{C}_0^0 - \boldsymbol{C}_1^0 + \boldsymbol{C}_2^0 - \boldsymbol{C}_3^0)/2 = \boldsymbol{\Delta}^0 > \boldsymbol{\Delta}^1 = (\boldsymbol{C}_0^1 - \boldsymbol{C}_1^1 + \boldsymbol{C}_2^1 - \boldsymbol{C}_3^1)/2$, so the first bit is decoded as a 1. Similarly, $\boldsymbol{\Delta}^2 < \boldsymbol{\Delta}^3$, so the second bit is decoded as a 0

1. For the first measurement period, we disable all stressor ROs, and let the receiver ROs run for 2^t clock cycles, producing counts $\boldsymbol{C}_0^i = (C_0^0, \ldots, C_0^{R \cdot N_R - 1})$.
2. In the second period, we enable all (or some, see Sects. 7.4.3 and 7.5.3) stressor ROs and estimate the RO frequencies through their counts, $\boldsymbol{C}_1^i$.
3. In the third measurement period, we disable all stressor ROs, re-enable them in the fourth period, and so forth.

This procedure produces $M/2$ measurements $\boldsymbol{C}_0^i, \boldsymbol{C}_2^i, \ldots$ corresponding to disabled stressors, and $M/2$ measurements $\boldsymbol{C}_1^i, \boldsymbol{C}_3^i, \ldots$ corresponding to enabled stressors, as also shown in the timing diagram of Fig. 7.7. Figure 7.7 represents Manchester-encoded transmissions of the 2 bits `10`, averaging over $M = 4$ measurements and only repeating transmissions once (actual measurements have $M = 500$, with 4 repetitions). We take the average of each set per RO, thereby calculating the disabled-stressor average $\boldsymbol{D}^i = 2/M \cdot \sum_{k=0}^{M/2-1} \boldsymbol{C}_{2k}^i$ and the enabled-stressor average $\boldsymbol{E}^i = 2/M \cdot \sum_{k=0}^{M/2-1} \boldsymbol{C}_{2k+1}^i$. We then use $\boldsymbol{\Delta}^i = \boldsymbol{D}^i - \boldsymbol{E}^i$ to recover the transmitted bit.

Specifically, assume that we wish to recover the n-th bit, corresponding to transmissions $2n$ and $2n + 1$, as each bit b is Manchester-encoded as the pair $(b, 1 - b)$. In each transmission pair, there is always a 1 bit and a 0 bit, so we can compare the $R \cdot N_R$ counts of $\boldsymbol{\Delta}^{2n}$ and $\boldsymbol{\Delta}^{2n+1}$. If the majority of the RO differences in the first set of measurements is bigger than the corresponding differences in the second set of measurements (i.e., $\Delta^{2n} > \Delta^{2n+1}$ for most ROs), we classify the n-th bit as a 1, while if the majority is smaller, ($\Delta^{2n} < \Delta^{2n+1}$ for most ROs), we classify it as a 0.

Figure 7.8 demonstrates the need for this more complicated procedure in practice for a transmission of a Manchester-encoded 1 bit. Specifically, it compares our new metric with stressor ROs, $\boldsymbol{\Delta}^{2n} - \boldsymbol{\Delta}^{2n+1}$, against the naive bit-recovery metric $\boldsymbol{D}^{2n} - \boldsymbol{D}^{2n+1}$ for all 20 receiver ROs. As Fig. 7.8 (blue circles) shows, $\boldsymbol{\Delta}^{2n} - \boldsymbol{\Delta}^{2n+1} > 0$ for all 20 receiver ROs $R_0, R_1, \ldots$, so our metric correctly

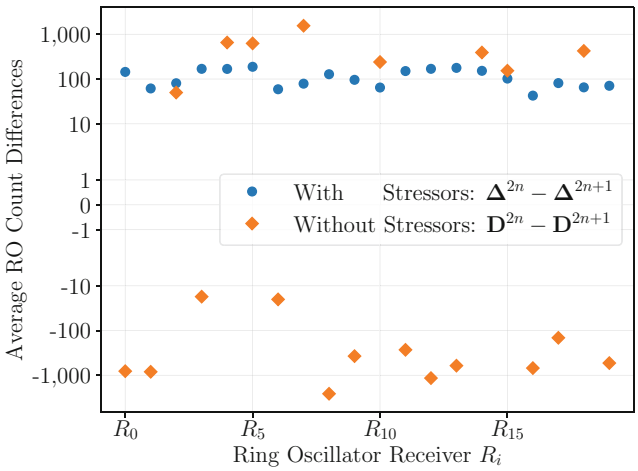


Fig. 7.8 All RO count differences with stressors $\boldsymbol{\Delta}^{2n} - \boldsymbol{\Delta}^{2n+1}$ (blue circles) are positive, correctly decoding a transmission of 1. However, the naive metric without stressors $\boldsymbol{D}^{2n} - \boldsymbol{D}^{2n+1}$ (orange diamonds) behaves randomly, with only about half being positive

recovers this bit transmission. However, the $\boldsymbol{D}^{2n} - \boldsymbol{D}^{2n+1}$ values with stressors disabled (orange diamonds) behave randomly, and indeed, in the experiment in which these measurements originated, our metric successfully recovered over 98% of transmissions, compared to 53% using the naive method without the stressors. Section 7.4.3 further expands on why the new technique makes for a good approach in detecting transmissions.

7.4.3 Characterization of the Proposed Metric

In this section, we test the receiving circuit (sink FPGA) on its own to characterize its behavior. We first plot in Fig. 7.9 the average metric Δ_V^i for the eight ring oscillators of Fig. 7.6 across the same power supply voltages $11.5\,\text{V} \leq V \leq 12.5\,\text{V}$. As expected, for all ROs, $\Delta_{V_1}^i < \Delta_{V_2}^i$ whenever $V_1 > V_2$: When there is an external voltage drop (e.g., when the source FPGA enables the transmitter ROs), the $\boldsymbol{\Delta}$ metric increases compared to when there are no external transmissions.

We additionally test the behavior of the receiver FPGA across different measurement times of 2^t clock cycles and the numbers of enabled stressors S. Specifically, we conduct measurements on an Artix 7 sink and calculate the average value of our $\boldsymbol{\Delta}$ metric over all 20 receiver ROs at two voltage levels: 11.5 V and 12.5 V. Figure 7.10 plots our results, which lead to several observations.

First of all, the average difference $\Delta = \Delta_{11.5} - \Delta_{12.5}$ is close to zero for time periods up to 41 µs, indicating that prolonged measurement times are necessary to distinguish between transmissions of zero and one, which in practice result in

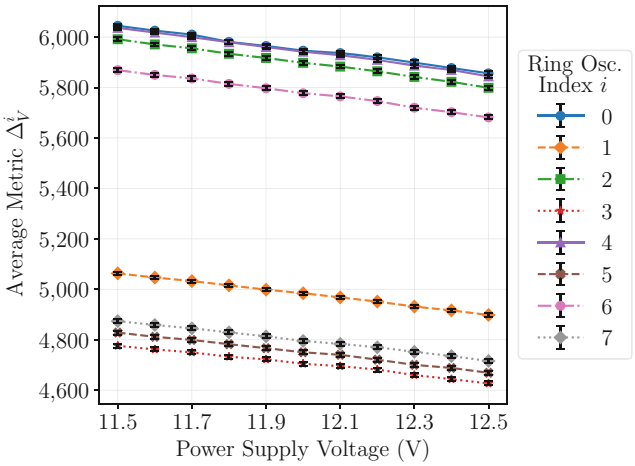


Fig. 7.9 The average metric Δ_V^i on the AC701-1 sink decreases with higher power supply voltages V for all eight ring oscillators R_i

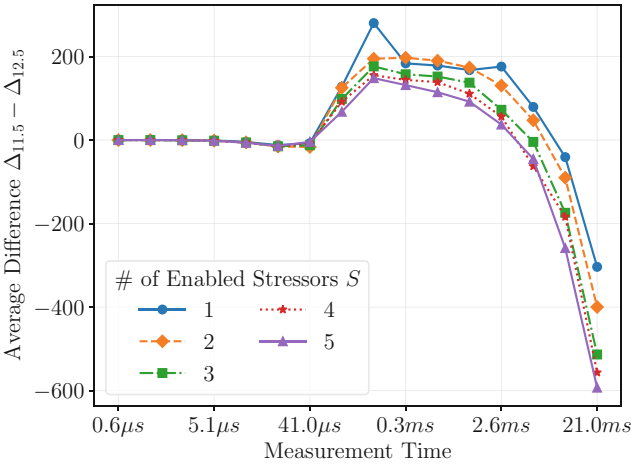


Fig. 7.10 Difference between the average Δ metric as measured at 11.5 V and 12.5 V for different measurement times and numbers of stressors enabled on the AC701-1 sink

much smaller voltage drops of ≈20 mV. Moreover, until 2.6 ms, $\Delta > 0$ for all choices of how many stressors S to enable simultaneously, with fewer stressors resulting in a larger effect. However, for even larger time periods, $\Delta < 0$, with more stressors resulting in a bigger effect in magnitude. Consequently, the choice of number of stressors and measurement time is intricately linked with the accuracy of the covert channel and, in fact, helps explain why in some experimental setups (e.g., the KC705-1 receiver on PSU-B of Table 7.3), the recovered pattern is flipped, i.e., a 0 bit is identified as a 1 bit and vice versa.

Table 7.3 Default values for accuracy- and bandwidth-related parameters, and the chapter sections in which they are varied. Bandwidth is calculated using Eq. (7.1)

Property	Artix 7	Kintex 7	Section
# of Transmitter ROs, N_T	1000	1000	7.5.2
# of Enabled Transmitters	10	14	7.5.2
Transmitted Pattern	`0xf3ed1`	`0xf3ed1`	7.5.4
Transmitter Types	LUT-RO	LUT-RO	7.5.4
# of Stressor ROs, N_S	500	500	7.5.2
# of Enabled Stressors	1	5	7.5.3
Stressor and Receiver Types	LUT-RO	LUT-RO	7.5.5
# of Repetitions per Bit, M	500	500	7.5.3
Measurement Cycles, 2^t	2^{15}	2^{21}	7.5.3
Channel Bandwidth b ($\mathrm{b\,s^{-1}}$)	6.1	0.1	7.5.3

7.5 Cross-FPGA Communication

In this section, we explore FPGA-to-FPGA covert communication, presenting a summary of our results with the default experimental parameters in Sect. 7.5.1. We then vary the number of source transmitter and sink stressor ROs in Sect. 7.5.2. We further evaluate bandwidth–accuracy tradeoffs in Sect. 7.5.3 and test the performance of the covert channel across different transmitter patterns and cabling setups in Sect. 7.5.4. We finally test the covert channel using different types of ROs and under different experimental conditions in Sect. 7.5.5.

7.5.1 Overview of Results

In this section, we give an overview of our cross-FPGA results. The values for the default experimental parameters used in these experiments and the corresponding covert-channel bandwidths are summarized in Table 7.3. These values were chosen based on exploratory testing, as they represent a good tradeoff between accuracy and bandwidth. However, in some cases, better accuracy can be achieved at the cost of bandwidth, or the same accuracy can be maintained despite increasing the bandwidth (see Sect. 7.5.3).

The results of our measurements across all 12 combinations of source and sink FPGAs on both PSUs are summarized in Table 7.4. As the table shows, covert communication is possible with high accuracy between any two boards, in either direction, and on both PSUs. The table also allows us to draw various conclusions. First of all, the behavior is not the same for identical boards. This is likely due to both process variations internal to the FPGA chip (which affect RO measurements), and because of different component tolerances. As an example, the AC701-2 board

Table 7.4 Accuracy for cross-FPGA covert channels on PSUs A and B, using the default experimental parameters

PSU	Transmitter	Receiver			
		AC701-1	AC701-2	KC705-1	KC705-2
A	AC701-1	–	79%	92%	100%
A	AC701-2	99%	–	93%	100%
A	KC705-1	100%	86%	–	100%
A	KC705-2	100%	98%	99%	–
B	AC701-1	–	100%	†98%	100%
B	AC701-2	100%	–	†99%	100%
B	KC705-1	100%	95%	-	100%
B	KC705-2	100%	100%	†98%	-

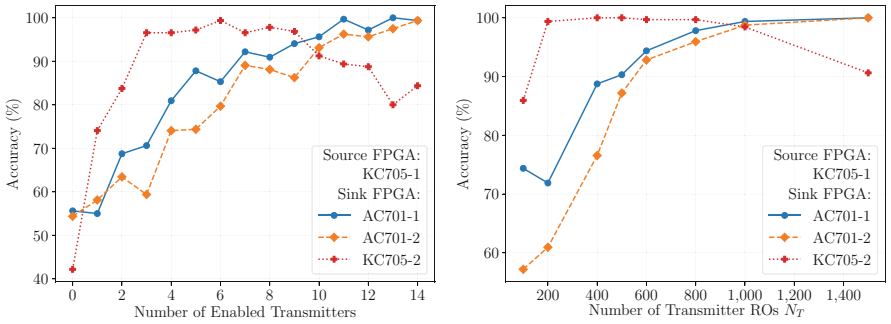
† signifies that the recovered bit pattern is flipped

is a worse sink than the AC701-1 board, while the KC705-1 board is a worse source than the KC705-2 board.

Moreover, the Kintex 7 boards are generally better sources than the Artix 7 boards, due to the higher count of transmitters they contain ($T = 14$ as opposed to $T = 10$). As we show in Sect. 7.5.2, more transmitters tend to improve the quality of the covert channel. Finally, we notice that although the information leakage remains strong in both PSUs, the accuracy of the recovered data on the 1300 W PSU-B is generally higher than the accuracy on the 850 W PSU-A. This is perhaps somewhat surprising, given that we would have expected the higher-rated PSU to produce more stable output under sudden changes in the load, but this appears to not be the case.

7.5.2 Transmitter and Stressor ROs

In this section, we evaluate the effect of changing the size of the transmitting and receiving circuits in the source and sink FPGAs, respectively, on the accuracy of the covert channel. Since each of the T transmitters (with N_T ROs each) can be controlled independently (Fig. 7.3), we first vary the number of simultaneously enabled transmitters on the KC705-1 board and plot the results across all receiver boards in Fig. 7.11a. We also change the number of transmitter ROs N_T on KC705-1 with all T transmitters enabled at the same time and plot the results in Fig. 7.11b. Both experiments show that increasing the number of effective transmitter ROs $T \cdot N_T$ increases the accuracy of the covert channel. This is because the ensuing voltage drops are more pronounced and can thus be more easily detected by the receiving boards. However, for the KC705-2 sink board, too much activity on the transmitter can decrease the accuracy of the channel. This is because although the magnitude of the voltage drop increases in isolation (Fig. 7.5), the stressor ROs are also causing a voltage drop that can overshadow that of the source FPGA.



(a) Transmitters (b) Transmitter ROs

Fig. 7.11 Increasing the number of (**a**) simultaneously enabled transmitters and (**b**) transmitter ROs N_T on the KC705-1 source board generally increases the accuracy of the cross-board covert channel, except for the KC705-2 sink past a certain threshold

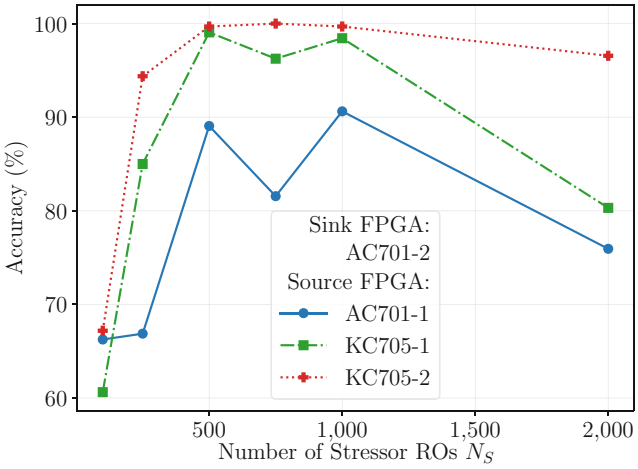


Fig. 7.12 Increasing the number of stressor ROs N_S on the AC701-2 sink board can decrease accuracy, as the additional activity can hide external transmissions under the noise floor

We additionally evaluate the effect of changing the number of stressor ROs N_S on the sink AC701-2 board and plot the accuracy of the covert channel in Fig. 7.12. Consistent with Fig. 7.10, although stressor ROs are necessary to detect covert transmissions, further increasing N_S can have the opposite effect: the voltage drop caused by the stressors overpowers any effect caused by the source transmissions and starts pushing the average difference from positive to negative.

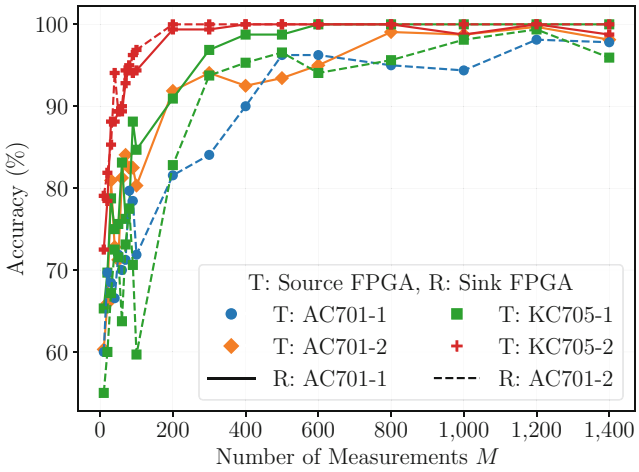
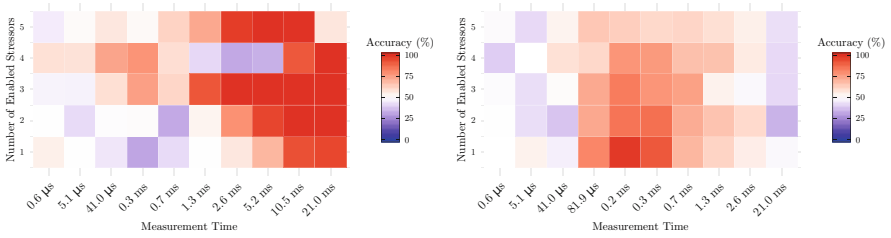


Fig. 7.13 Increasing the number of measurements M improves accuracy to any AC701 sink R, from any FPGA source T

7.5.3 Bandwidth–Accuracy Tradeoffs

In this section, we investigate accuracy–bandwidth tradeoffs by varying both the measurement period of 2^t clock cycles and the number of measurements M over which the RO counts are averaged. We first experiment with both the AC701-1 and the AC701-2 boards as sinks and plot the results from all other possible FPGA sources in Fig. 7.13. In general, increasing the number of measurements increases the accuracy of the covert channel, but at a cost of lower bandwidth. $M = 500$ represents a good tradeoff between accuracy and bandwidth (over 90% accuracy at $6.1\,\mathrm{b\,s^{-1}}$ for the Artix 7 boards), but $M \geq 1000$ results in higher accuracy at half the bandwidth.

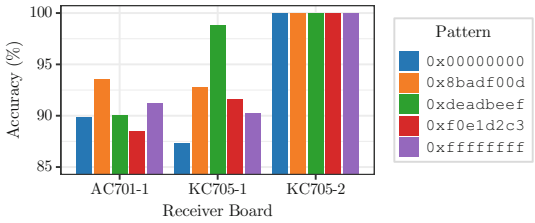
The second aspect we investigate is varying the number of clock cycles 2^t for which each RO is counting. At the same time, we also change the number of enabled stressors on the sink FPGA and test the accuracy of the covert channel with the AC701-2 FPGA source. The results for the KC705-1 and AC701-1 sinks are shown in Figs. 7.14a and b, respectively. These results indicate that the parameters for the receivers need to be carefully tuned for different types of boards. For example, the Artix 7 board necessitates that fewer stressors be driven, which is consistent with the results of Sects. 7.4.3 and 7.5.2. On the other hand, the KC705-1 sink remains accurate across a wider range of enabled stressors but requires longer measurement periods for acceptable accuracies.



(a) KC705-1 (b) AC701-1

Fig. 7.14 Accuracy for different measurement times and the number of enabled stressors on the (**a**) KC705-1 and (**b**) AC701-1 sinks

Fig. 7.15 The accuracy of the covert channel with the AC701-2 source remains similar across five different 32-bit patterns



7.5.4 Transmitted Patterns and Cabling Layouts

We test the transmission of longer patterns by communicating five 32-bit patterns (64 encoded bits). The patterns were chosen to have different Hamming Weights and runs of zeros and ones to show that the channel does not fundamentally depend on the values transmitted. The results, plotted in Fig. 7.15 for the AC701-2 source, indicate that the covert channel remains similarly accurate for all three sink boards and five transmitted patterns.

In the majority of the previous experiments, the source and sink FPGA boards were connected to the same PSU output through a Corsair peripheral cable with four Molex connectors. This cable was attached to one of the "bottom" 6-pin outputs of the PSU. However, to verify that the information leakage persists across different cable setups, we also use a 12-pin output of the PSU splitting into two 6-pin PCIe cables, denoted by "left" and "right." We then test communication from the KC705-1 board to the KC705-2 board across different cable setups, using the default measurement time of 2^{21} clock cycles, enabling all 5 stressors, but also increasing the number of measurements to $M = 1000$. The results of our experiments are summarized in Fig. 7.16, which demonstrates that a covert channel is possible in all setups tested. This is perhaps to be expected, since the PSU uses a "dedicated single +12 V rail" [5], but the results further indicate that there are differences among the ports tested. Specifically, the covert channel is most accurate between FPGA boards on the same cable (as they are at exactly the same electric potential difference) and

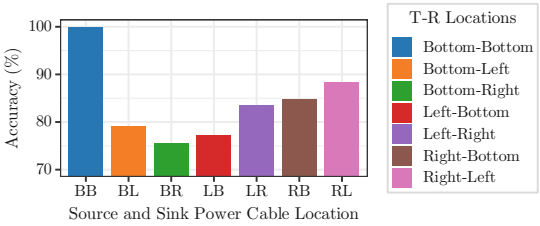


Fig. 7.16 The accuracy of communication between the transmitter (T) and the receiver (R) Kintex 7 boards depends on how they are connected to the Power Supply Unit

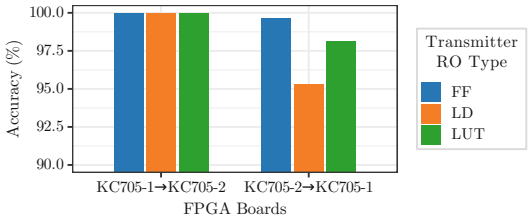


Fig. 7.17 The accuracy between the two Kintex 7 boards is consistently high for all types of source ROs tested

least accurate between the single location on the bottom of the PSU and either of the dual outputs. Finally, it should be noted that the recovered pattern is flipped in all setups, except when sharing the cable on the bottom output.

7.5.5 Ring Oscillator Types and Alternative Experimental Setup

We finally test communication using alternative types of ROs on the Kintex 7 boards, which we measure in a more realistic setup. Specifically, both boards are connected to PC-A over PCIe and are enclosed in the computer tower to avoid isolating thermal effects. The ROs used were proposed by Giechaskiel et al. [9, 10] to bypass currently deployed cloud countermeasures that prohibit combinatorial loops such as the LUT-RO used so far. One of them replaces a buffer gate with a latch (LD-RO), while the other one with an inverter and a flip-flop (FF-RO). The setup otherwise uses the default experimental parameters of Table 7.3. Figure 7.17 first shows that for all three types of transmitter ROs, the accuracy of the cross-KC705 channel remains above 95%, despite potential noise introduced by thermal conditions and the shared PCIe buses. Similarly, Fig. 7.18 shows that accuracy remains above 95% when using these alternative ROs for stressors and receivers on a KC705 sink. Although in many cases bits are again flipped, blocking combinatorial loops and introducing environmental noise cannot prevent our channel from operating.

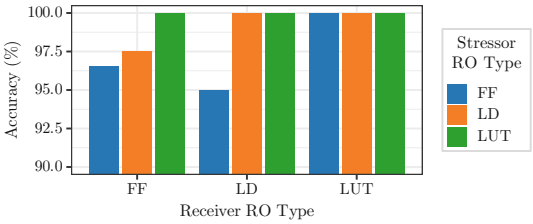


Fig. 7.18 The accuracy from the KC705-1 source to the KC705-2 sink using different receiver and stressor ROs also remains high

7.6 Additional Covert Channels

In this section, we explore CPU-to-FPGA (Sect. 7.6.1) and GPU-to-FPGA covert channels (Sect. 7.6.2).

7.6.1 CPU Transmissions

In order to test the CPU-to-FPGA communication channel, we replace the power draw of the FPGA source with heavy CPU loads. To that end, we use the open-source `stress` program, which is available on Debian-based Linux distribution package managers [40]. We vary the number of threads that `stress` uses from 0 (i.e., no transmissions, corresponding to random measurements), up to the number of threads available on each computer, i.e., 24 on the CPU attached to PSU-A, and 4 on the CPU attached to PSU-B.

The measurement process and classification metric remain the same as for the cross-FPGA channels, but we introduce an additional delay of 3 seconds after the `stress` program has started to ensure full utilization of the cores, and an additional 3 seconds after killing the process, to ensure that the usage has returned to normal. Moreover, when testing with PSU-A, and to increase accuracy, we reduce the measurement period for the KC705 receivers to $2^t = 2^{18}$ clock cycles (1.3 ms) from 2^{21} (10 ms), and the number of stressors to 4 instead of 5 (we use the default parameters on PSU-B but increase measurements for the AC701 boards to $M = 1200$). This increases the bandwidth of the covert channel by a factor of $8\times$ to $0.8\,\mathrm{b\,s^{-1}}$ compared to the cross-FPGA channel.

We plot the results for the two PSUs in Fig. 7.19, which allows us to draw three main conclusions. First of all, there is a critical CPU activity threshold that is necessary to make the covert channel possible. On PSU-A, this requires about 4 threads for the AC701 boards and 7 threads for the KC705 boards. Moreover, increasing the number of threads does not always make the covert channel more accurate. For example, increasing the number of CPU threads from 0 to 10 increases accuracy, but the accuracy generally plateaus between 10 and 17 CPU threads, and then decreases, perhaps due to hyper-threading. Finally, we notice that for a similar number of threads used, the accuracy on PSU-B is often higher compared to that

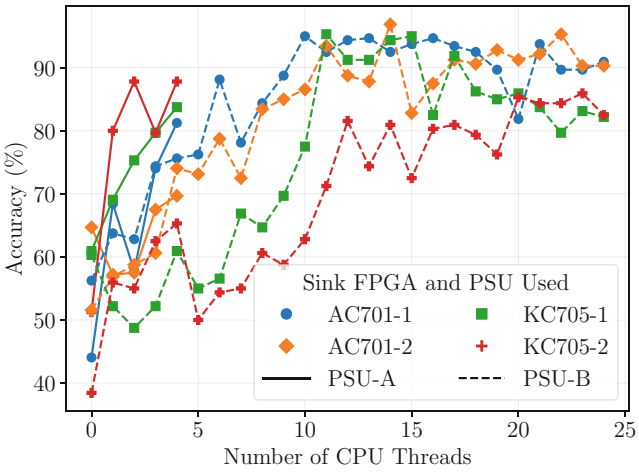


Fig. 7.19 CPU-to-FPGA accuracy for the four FPGA sink boards on both PSUs for different numbers of CPU threads used as transmitters. As PSU-A powers a CPU with only 4 threads, no more than 4 threads can be dispatched for testing

Table 7.5 Maximum accuracy of transmissions from a CPU source to the four FPGA sinks on the two PSU and PC setups, along with the parameters for which the accuracy is achieved

PSU	Parameter	AC701-1	AC701-2	KC705-1	KC705-2
A	Accuracy	95%	97%	95%	86%
A	Bandwidth	6.1 b s^{-1}	6.1 b s^{-1}	0.8 b s^{-1}	0.8 b s^{-1}
A	# of Threads	10	14	11	23
A	# of Enabled Stressors	1	1	4	4
A	# of Measurements	500	500	500	500
A	Measurement Cycles	2^{15}	2^{15}	2^{18}	2^{18}
B	Accuracy	81%	70%	†84%	88%
B	Bandwidth	2.5 b s^{-1}	2.5 b s^{-1}	0.1 b s^{-1}	0.1 b s^{-1}
B	# of Threads	4	4	4	4
B	# of Stressors	1	1	5	5
B	# of Measurements	1200	1200	500	500
B	Measurement Cycles	2^{15}	2^{15}	2^{21}	2^{21}

† signifies that the recovered bit pattern is flipped

for PSU-A. This parallels our cross-FPGA results of Sect. 7.5 and indicates that PSU-B is generally more prone to covert communication. The maximum accuracy achieved, the number of CPU threads used, and other experimental parameters are summarized in Table 7.5.

Table 7.6 Parameters for GPU testing with `gpu_burn`

Property	GPU-A	GPU-B
Architecture	Fermi	Kepler
Technology	40 nm	28 nm
Driver Version	390.87	418.67
CUDA Version	8.0	10.1
Compiler Flag	`compute_20`	`compute_50`

Table 7.7 Maximum accuracy of transmissions from a GPU source to the four FPGA sinks on the two PSU and PC setups, along with the parameters for which the accuracy is achieved

PSU	Parameter	AC701-1	AC701-2	KC705-1	KC705-2
A	Accuracy	76%	70%	94%	89%
B	Accuracy	97%	87%	96%	†100%
A&B	Bandwidth	$2.0\,\mathrm{b\,s^{-1}}$	$2.0\,\mathrm{b\,s^{-1}}$	$0.03\,\mathrm{b\,s^{-1}}$	$0.03\,\mathrm{b\,s^{-1}}$
A&B	# of Enabled Stressors	1	1	5	5
A&B	# of Measurements	1500	1500	1500	1500
A&B	Measurement Cycles	2^{15}	2^{15}	2^{21}	2^{21}

† signifies that the recovered bit pattern is flipped

7.6.2 GPU Transmissions

The process for testing GPU-to-FPGA transmissions is similar to that of CPU-to-FPGA transmissions. We stress the GPUs with the open-source `gpu_burn` [39] program, which uses Nvidia's CUDA platform to fully utilize the GPU cores. As the two GPUs use different architectures, we compile and run the `gpu_burn` program against different Nvidia drivers and CUDA versions. These differences are summarized in Table 7.6. Moreover, we return to the default measurement period of $2^t = 2^{21}$ cycles for the Kintex 7 boards and increase the number of measurements for all boards to 1500, reducing bandwidth by a factor of $3\times$. These parameters and the corresponding results are summarized in Table 7.7. As in the CPU case, 3 seconds of delay are added after before and after the program, to allow usage to return to normal.

Figure 7.20 plots the results of our experiments for the four boards on both GPUs. We find that it is possible to create a communication channel to all four boards, on both PSUs. As expected, since there are fewer GPU cores attached to PSU-A, the covert channel is weaker, but the accuracy is over 95% for three of the four boards when using the GPU attached to PSU-B, which is larger. Moreover, we notice that the AC701 boards are worse sinks than the KC705 boards. Although this pattern is not entirely identical across the three communication channels (FPGA-to-FPGA, CPU-to-FPGA, and GPU-to-FPGA), it broadly remains consistent, potentially due to the differences in the voltage regulators themselves or other aspects of board design and component tolerances.

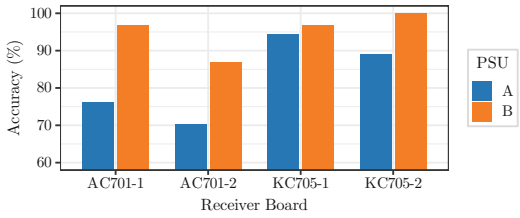


Fig. 7.20 GPU-to-FPGA accuracy for the four FPGA sink boards on both computers and PSUs

7.7 Discussion

In this section, we discuss how practical the covert channels we introduced are (Sect. 7.7.1) and propose some software- and hardware-level countermeasures to mitigate the impact of the information leakage (Sect. 7.7.2).

7.7.1 Practicality of Attacks

There are two aspects of how practical our communication scheme is, which we evaluate in this section. The first is how costly transmissions are in terms of resources used on the FPGA boards. The amount of logic instantiated is moderate, but not negligible. On the transmitting end, $G \cdot T \cdot N_T$ lookup tables (LUTs) are used, where $G = 4$ is the number of ring oscillator stages. In particular, the source design (including the UART and other logic) utilizes 16.6% of LUT resources on the Artix 7 FPGA chip. Similarly, the sink design uses $G \cdot (R \cdot N_R + S \cdot N_S)$ LUTs for the receiver and stressor ROs, and $L \cdot R \cdot N_R$ registers for counting, where $L = 32$ is the length of the counters. Only 7.8% of the Artix 7 resources are used in this case—a number that can be reduced to 3.4%, as the AC701 boards only enable one stressor for higher accuracy.

The second aspect is the channel capacity, which lies between that of thermal attacks, which can transmit under 15 bits in an hour [14, 38], and power attacks within CPUs that can transfer between 20 and 120 bits per second [1, 19].

Although the Kintex 7 boards were shown to be better sinks (often with 0% error rate), the Artix 7 boards were faster by a factor of 7.6× (6.1 b s^{-1} vs. 0.8 b s^{-1}). This difference is significant in practice: Table 7.8 shows how long it would take to transmit keys for different popular cryptographic algorithms. Even assuming that the channel is not noisy, it would take almost 45 minutes to transfer a 256-bit AES key to a KC705 board, and 3 hours to transfer a 1024-bit RSA key. However, the AC701 board would need less than 3 minutes to transfer the same RSA key, despite the potential drop in accuracy.

To increase accuracy, one can either tweak the parameters of the source and sink FPGA designs (including the number of measurements M over which RO counts are averaged) or instead change the communication scheme itself. For example, a 3-repetition code decreases bandwidth by a factor of 3, but also lowers the error rate

Table 7.8 Time to leak cryptographic keys of different sizes to the Artix 7 and Kintex 7 boards

Algorithm	Key size	AC701	KC705
AES	256	0.7 min	44.7 min
ECDSA	521	1.4 min	91.1 min
RSA	1024	2.8 min	179.0 min

e to $3e^2 - 2e^3$: a 10% error rate is reduced to under 3%. The channel capacity is $1-H(e) = 1+e\log_2 e+(1-e)\log_2(1-e)$, and for smaller bitflip probabilities, other error correcting codes such as Hamming and Golay codes can be used to improve accuracy.

7.7.2 Defense Mechanisms

In this section, we discuss potential software and hardware defense countermeasures against voltage-based covert- and side-channel attacks. To start with, some countermeasures might revolve around preventing intentional transmissions from the covert-channel source. However, doing so would be particularly hard without huge sacrifices in terms of power and performance. Although we used ring oscillators to cause fluctuations in the voltage of FPGAs sharing the same PSU, other switching activity can also result in voltage over- and under-shoots. For example, prior work has shown that switching large sets of programmable interconnect points [47] or self-oscillating circuits consisting of flip-flops or carry chains [21] can cause voltage fluctuations outside of the allowed operating voltage range for an FPGA device. Moreover, we demonstrated CPU-to-FPGA and GPU-to-FPGA channels, which show that the problem is not FPGA-specific, but can be found in other types of activities that result in large power draws. Consequently, unless power is equalized among all possible algorithm implementations, some leakage that can differentiate between levels of activity will persist.

To prevent side-channel attacks from being possible, designers may remove the power-draw dependence on the data being processed and increase the noise level. Although several masking and hiding techniques have been proposed, leakage on FPGAs persists due to variations in placement and routing [6]. Consequently, a better approach is to prevent the leakage from being measurable on the FPGA sinks.

Current FPGA cloud providers prevent voltage and temperature monitors from being accessible by user logic and prohibit traditional LUT-ROs from being instantiated on their infrastructure [2]. However, alternative ring oscillator designs can bypass cloud restrictions [9, 10, 21, 22, 35] and can also replace LUT-ROs (Sect. 7.5.5). Moreover, time-to-digital converters (TDCs) can also be used instead of ring oscillators to monitor voltage fluctuations and conduct side-channel attacks [33]. Although compiler tools that check for combinatorial loops and latches [21, 22] would prevent some of the above monitoring logic, it would not necessarily catch all forms of self-oscillating logic.

Given that designing effective countermeasures against side- and covert-channel receivers is an arms race, defense-in-depth would dictate run-time solutions in addition to any preventive approach. One feature of the covert channel is the high switching activity on the receiver. Built-in voltage monitors (such as those proposed for shared FPGAs [12, 25, 31]) could thus be used by cloud providers to detect abnormal fluctuations—with the caveat that legitimate circuits may also cause similar patterns, and that, at least on the AC701 boards, the number of enabled stressor ROs was small ($N_S = 500$). In fact, proposals to "detect the insertion of power measurement circuits onto a device's power rail" [23] are similar, though the challenge is to reduce false positives.

Finally, better hardware (at a higher cost) can also help hide the useful signal under the noise floor. For example, independent, fully separate power supplies for different boards would require that the leakage be detectable even over the AC power line, and through two different AC-to-DC rectifiers. Moreover, better isolation of power circuits within the same PSU, as well as voltage regulators with better transient responses on both the source and the sink FPGAs, or differently designed powering circuits with more filters and smoothing capacitors can also reduce the signal available to an attacker.

7.8 Related Work

This section summarizes prior work in remote FPGA attacks without physical access to the boards (Sect. 7.8.1), as well as voltage- and temperature-based covert channels (Sect. 7.8.2).

7.8.1 Remote FPGA Attacks

Although attacks on FPGA systems have traditionally required physical access to the FPGA board, a recent class of remote attacks has emerged. These attacks have used ring oscillators and TDCs as covert- and side-channel receivers, and ROs and other circuits as covert-channel transmitters and fault attack inductors.

Most of the proposed attacks operate in the multi-tenant setting, from a weak threat model where logic resources of different tenants are adjacent [8, 9, 11] to progressively stronger ones where the attacker and victim are physically separated on the same FPGA die [30, 45, 46] or even across separate dies on 2.5D-integrated FPGA chips [10]. The target applications are equally diverse, from covert channels [10] and fingerprinting different applications [13] to recovering cryptographic keys [33, 46] and inferring machine learning parameters [29, 37, 45]. In parallel, fault attacks have been used for similar purposes, from biasing True Random Number Generators (TRNGs) [27] and causing errors in CPU-FPGA SoC hybrids [26] to attacking neural networks [4, 24, 32].

So far, there have only been few works that consider remote attacks in the single-tenant setting. One such attack by Tian and Szefer introduced a temporal thermal channel, where different users receive time-shared access to the same FPGA fabric [38]. A different attack by Schellenberg et al. considered cross-chip side-channel attacks to recover RSA keys [33]. However, the chips were located on the same FPGA board that is explicitly "designed for external side-channel analysis research" [33], and hence shared the same voltage regulator, making them easier to influence directly, due to the lack of additional intermediate components between their power distribution networks.

7.8.2 Power and Temperature Covert Channels

It is well-known that data-dependent power consumption can be used to recover cryptographic keys through differential power analysis and other techniques by acquiring and analyzing power traces [20]. The same principles can be applied to create covert communication, for example, from a malware app on a phone to a malicious USB charger [34], or from a program that modulates CPU utilization to an attacker measuring the current consumption of the computer [15]. Similarly, measuring voltage ripple on the power lines can be used to track the power usage pattern of other data center tenants [17]. Although these works exploit the same source of information leakage, they require external equipment to detect these data-dependent power variations and are thus not applicable to cloud environments in practice. However, it is possible to use the reconfigurable fabric of FPGAs as a covert-channel sink, allowing for accurate transmission of data remotely, without physical access.

Another category of power attacks that has recently been discovered is related to dynamic voltage and frequency scaling (DVFS) on modern processors, which regulates the voltage and frequency of CPUs in accordance with usage demands. Malicious software can exploit DVFS to cause faults in computations [36], or create covert channels between CPU cores, where the source core modulates frequency, and the sink core measures a reduction in its own performance [19].

Thermal attacks can also be used to create covert channels between CPU cores [28], but they require access to CPU thermal sensors and are slower than their power counterparts, having a capacity of up to $300\,\mathrm{b\,s^{-1}}$ [3]. Temperature-based covert channels need not be limited to communication within a single computer. Assuming computers are sufficiently close, a covert channel between nearby yet air-gapped devices is also possible with access to temperature sensors on the sink computer [14]. Finally, thermal information can also be used as a proxy estimate for power consumption in data centers. This information can alert potential adversaries to opportune moments to attack the availability of servers, either by exceeding the power capacity [18], or by more generally degrading performance [7]. Although these attacks require privileged thermal sensors, FPGA ROs could also be used for similar purposes, complementing our work.

7.9 Conclusion

In this chapter, we presented the first FPGA-to-FPGA, CPU-to-FPGA, and GPU-to-FPGA voltage-based covert channels, achieving transmission accuracies of up to 100%. Unlike prior work, which unrealistically assumes that different users share the same FPGA fabric, our work considered a stronger threat model, where the FPGA chip and board are allocated on a per-user basis. Our covert channel exploited properties of the response of power supply units (PSUs) and voltage regulators to changes in their load. To detect these changes, we introduced a novel architectural design and classification metric that depends on stressor ring oscillators on the covert-channel sink FPGA. We showed that ring oscillators also performed well in the source FPGA and further showed that heavy CPU and GPU activity could also be used as an effective transmitter. We demonstrated our covert channel on four Artix 7 and Kintex 7 boards, creating a channel of communication between any two of them in either direction, with high accuracy. We also performed an analysis of bandwidth–accuracy tradeoffs and further explored the accuracy of the covert channel across different sizes and types of the sink and source FPGA circuits, different measurement patterns and setup layouts, and PSUs with different power ratings from two manufacturers. We finally proposed potential countermeasures to prevent the information leakage we discovered from being exploitable. Overall, our remote covert-channel attacks highlight the dangers of shared power supply units, and therefore a need to re-think FPGA security, even for single-user monolithic designs.

Acknowledgments This work was supported in part by NSF grant 1901901.

References

1. Alagappan, M., Rajendran, J., Doroslovački, M., & Venkataramani, G. (2017). DFS covert channels on multi-core platforms. In *IFIP/IEEE international conference on very large scale integration (VLSI-SoC)*.
2. Amazon Web Services (2021). AWS EC2 FPGA HDK+SDK errata. https://github.com/aws/aws-fpga/blob/master/ERRATA.md. Accessed: 2023-05-21.
3. Bartolini, D. B., Miedl, P., & Thiele, L. (2016). On the capacity of thermal covert channels in multicores. In *European conference on computer systems (EuroSys)*.
4. Boutros, A., Hall, M., Papernot, N., & Betz, V. (2020). Neighbors from hell: Voltage attacks against deep learning accelerators on multi-tenant FPGAs. In *International conference on field-programmable technology (FPT)*.
5. Corsair (2010). Professional series Gold AX850–80 PLUS Gold certified fully-modular power supply. https://www.corsair.com/p/CMPSU-850AX. Accessed: 2023-05-21.
6. De Cnudde, T., Ender, M., & Moradi, A. (2018). Hardware masking, revisited. *IACR transactions on cryptographic hardware and embedded systems (TCHES), 2018*(2), 123–148.
7. Gao, X., Xu, Z., Wang, H., Li, L., & Wang, X. (2018). Reduced cooling redundancy: A new security vulnerability in a hot data center. In *Network and distributed system security symposium (NDSS)*.

8. Giechaskiel, I., Rasmussen, K. B., & Eguro, K. (2022). Long-wire leakage: The threat of crosstalk. *IEEE Design and Test (D&T), 39*(4), 41–48.
9. Giechaskiel, I., Rasmussen, K. B., & Szefer, J. (2019). Measuring long wire leakage with ring oscillators in cloud FPGAs. In *International conference on field programmable logic and applications (FPL)*.
10. Giechaskiel, I., Rasmussen, K. B., & Szefer, J. (2019). Reading between the dies: Cross-SLR covert channels on multi-tenant cloud FPGAs. In *IEEE international conference on computer design (ICCD)*.
11. Giechaskiel, I. & Szefer, J. (2020). Information leakage from FPGA routing and logic elements. In *International conference on computer-aided design (ICCAD)*.
12. Glamočanin, O., Mahmoud, D., Regazzoni, F., & Stojilović, M. (2021). Shared FPGAs and the holy grail: Protections against side-channel and fault attacks. In *Design, automation & test in Europe (DATE)*.
13. Gobulukoglu, M., Drewes, C., Hunter, W., Kastner, R., & Richmond, D. (2021). Classifying computations on multi-tenant FPGAs. In *Design automation conference (DAC)*.
14. Guri, M., Monitz, M., Mirski, Y., & Elovici, Y. (2015). BitWhisper: Covert signaling channel between air-gapped computers using thermal manipulations. In *IEEE computer security foundations symposium (CSF)*.
15. Guri, M., Zadov, B., Bykhovsky, D., & Elovici, Y. (2020). PowerHammer: Exfiltrating data from air-gapped computers through power lines. *IEEE Transactions on Information Forensics and Security (TIFS), 15*, 1879–1890.
16. Hajimiri, A., Limotyrakis, S., & Lee, T. H. (1999). Jitter and phase noise in ring oscillators. *IEEE Journal of Solid-State Circuits (JSSC), 34*(6), 790–804.
17. Islam, M. A., & Ren, S. (2018). Ohm's law in data centers: A voltage side channel for timing power attacks. In *ACM conference on computer and communications security (CCS)*.
18. Islam, M. A., Ren, S., & Wierman, A. (2017). Exploiting a thermal side channel for power attacks in multi-tenant data centers. In *ACM conference on computer and communications security (CCS)*.
19. Khatamifard, S. K., Wang, L., Das, A., Köse, S., & Karpuzcu, U. R. (2019). POWERT channels: A novel class of covert communication exploiting power management vulnerabilities. In *IEEE international symposium on high-performance computer architecture (HPCA)*.
20. Kocher, P., Jaffe, J., Jun, B., & Rohatgi, P. (2011) Introduction to differential power analysis. *Journal of Cryptographic Engineering, 1*(1), 5–27.
21. La, T., Pham, K., Powell J., & Koch, D. (2021). Denial-of-Service on FPGA-based cloud infrastructures: Attack and defense. *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES), 2021*(3), 441–464.
22. La, T. M., Matas, K., Grunchevski, N., Pham, K. D., & Koch, D. (2020). FPGADefender: Malicious self-oscillator scanning for Xilinx UltraScale+ FPGAs. *ACM Transactions on Reconfigurable Technology and Systems (TRETS), 13*(3), 1–31.
23. Le Masle, A., & Luk, W. (2012). Detecting power attacks on reconfigurable hardware. In *International conference on field programmable logic and applications (FPL)*.
24. Luo, Y., Gongye, C., Fei, Y., & Xu, X. (2021). DeepStrike: Remotely-guided fault injection attacks on DNN accelerator in cloud-FPGA. In *Design automation conference (DAC)*.
25. Luo, Y. & Xu, X. (2020). A quantitative defense framework against power attacks on multi-tenant FPGA. In *International conference on computer-aided design (ICCAD)*.
26. Mahmoud, D., Hussein, S., Lenders, V., & Stojilović, M. (2022). FPGA-to-CPU undervolting attacks. In *Design, automation and test in Europe (DATE)*.
27. Mahmoud, D., & Stojilović, M. (2019). Timing violation induced faults in multi-tenant FPGAs. In *Design, automation and test in Europe (DATE)*.
28. Masti, R. J., Rai, D., Ranganathan, A., Müller, C., Thiele, L., & Čapkun, S. (2015). Thermal covert channels on multi-core platforms. In *USENIX security symposium*.
29. Moini, S., Tian, S., Holcomb, D., Szefer, J., & Tessier, R. (2021). Remote power side-channel attacks on BNN accelerators in FPGAs. In *Design, automation and test in Europe (DATE)*.

30. Provelengios, G., Holcomb, D., & Tessier, R. (2020). Power distribution attacks in multi-tenant FPGAs. *IEEE transactions on very large scale integration systems (TVLSI), 28*(12), 2685–2698.
31. Provelengios, G., Holcomb, D., & Tessier, R. (2021). Mitigating voltage attacks in multi-tenant FPGAs. *ACM transactions on reconfigurable technology and systems (TRETS), 14*(2), 1–24.
32. Rakin, A. S., Luo, Y., Xu, X., & Fan, D. (2021). Deep-Dup: An adversarial weight duplication attack framework to crush deep neural network in multi-tenant FPGA. In *USENIX security symposium*.
33. Schellenberg, F., Gnad, D. R. E., Moradi, A., & Tahoori, M. B. (2018). Remote Inter-chip power analysis side-channel attacks at board-level. In *International conference on computer-aided design (ICCAD)*.
34. Spolaor, R., Abudahi, L., Moonsamy, V., Conti, M., & Poovendran, R. (2017). No free charge theorem: A covert channel via USB charging cable on mobile devices. In *Applied cryptography and network security (ACNS)*.
35. Sugawara, T., Sakiyama, K., Nashimoto, S., Suzuki, D., & Nagatsuka, T. (2019). Oscillator without a combinatorial loop and its threat to FPGA in data centre. *Electronics Letters, 15*(11), 640–642.
36. Tang, A., Sethumadhavan, S., & Stolfo, S. (2017). CLKSCREW: Exposing the perils of security-oblivious energy management. In *USENIX security symposium*.
37. Tian, S., Moini, S., Wolnikowski, A., Holcomb, D., Tessier, R., & Szefer, J. (2021). Remote power attacks on the versatile tensor accelerator in multi-tenant FPGAs. In *IEEE international symposium on field-programmable custom computing machines (FCCM)*.
38. Tian, S., & Szefer, J. (2019). Temporal thermal covert channels in cloud FPGAs. In *ACM/SIGDA international symposium on field-programmable gate arrays (FPGA)*.
39. Timonen, V. (2020). *Multi-GPU CUDA stress test*. http://wili.cc/blog/gpu-burn.html. Accessed: 2023-05-21.
40. Waterland, A. P. (2014). Stress. https://web.archive.org/web/20190502184531/https://people.seas.harvard.edu/~apw/stress/. Accessed: 2023-05-21.
41. Xilinx, Inc. (2012). 7 Series Product Brief. https://www.xilinx.com/publications/prod_mktg/7-Series-Product-Brief.pdf. Accessed: 2023-05-21.
42. Xilinx, Inc. (2019). AC701 evaluation board for the Artix-7 FPGA (UG952). https://www.xilinx.com/support/documentation/boards_and_kits/ac701/ug952-ac701-a7-eval-bd.pdf. Accessed: 2023-05-21.
43. Xilinx, Inc. (2019). KC705 evaluation board for the Kintex-7 FPGA (UG810). https://www.xilinx.com/support/documentation/boards_and_kits/kc705/ug810_KC705_Eval_Bd.pdf. Accessed: 2023-05-21.
44. Xilinx, Inc. (2020). 7 series FPGAs data sheet: Overview (DS180). https://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf. Accessed: 2023-05-21.
45. Zhang, Y., Yasaei, R., Chen, H., Li, Z., & Al Faruque, M. A. (2021). Stealing neural network structure through remote FPGA side-channel analysis. *IEEE Transactions on Information Forensics and Security (TIFS), 16*, 4377–4388.
46. Zhao, M., & Suh, G. E. (2018). FPGA-based remote power side-channel attacks. In *IEEE symposium on security and privacy (S&P)*.
47. Zick, K. M., Srivastav, M., Zhang, W., & French, M. (2013). Sensing nanosecond-scale voltage attacks and natural transients in FPGAs. In *ACM/SIGDA international symposium on field-programmable gate arrays (FPGA)*.