University *of* Massachusetts Amherst

Physical Security in FPGA circuits

Daniel Holcomb on behalf of many collaborators and sponsors

11/17/2025

Faculty Positions In AI Systems

- Two tenure-track faculty positions in Computer Engineering for scholars advancing AI systems from edge to datacenter.

Hardware for AI

- GPUs / custom accelerators / FPGAs
- Memory, interconnect, disaggregation
- ML-centric CAD / co-design

AI Systems & Architectures

- Schedulers, runtimes, compilers
- Datacenter/cluster design, TEEs
- Performance modeling & efficiency

Secure & Trustworthy AI

- Supply chain & tamper resilience
- Side-channels & physical safeguards
- Model/IP protection, red-teaming
- Human-in-the-loop decisions
- Misuse mitigation

Apply by 12/15 for full consideration

Search chair Dan Holcomb
dholcomb@umass.edu

Field Programmable Gate Arrays

- A fabric of programmable logic, routing, memory, and fixed-function blocks.
- Configured by a user-generated bitstream
- Dominated by Intel (Altera) and AMD (Xilinx)
- $10B/year and 10% growth
- Applications from embedded to cloud

Field Programmable Gate Arrays

- A fabric of programmable logic, routing, memory, and fixed-function blocks.
- Configured by a user-generated bitstream
- Dominated by Intel (Altera) and AMD (Xilinx)
- $10B/year and 10% growth
- Applications from embedded to cloud

www.wengerna.com

Position

Configurability exposes circuit-level capabilities for interactions that are beneath the usual abstraction of digital computation.

Appropriately harnessing and mitigating these non-digital behaviors can contribute to the security of hardware systems that incorporate reconfigurability.

Presentation shows four examples on this theme

- **Part 1: Security in Multi-Tenant FPGAs**
  - Remote side channel attacks using wire coupling
  - Fault attacks by adversarial power consumption

- **Part 2: Inter-chiplet delay PUF**
  - Implemented on Xilinx FPGAs locally and AWS

Security for Multi-Tenant FPGAs

Daniel Holcomb,
Russell Tessier,
and our students

Work made possible by grants from NSF and Intel's Corporate Research Council

Multi-Tenant FPGA

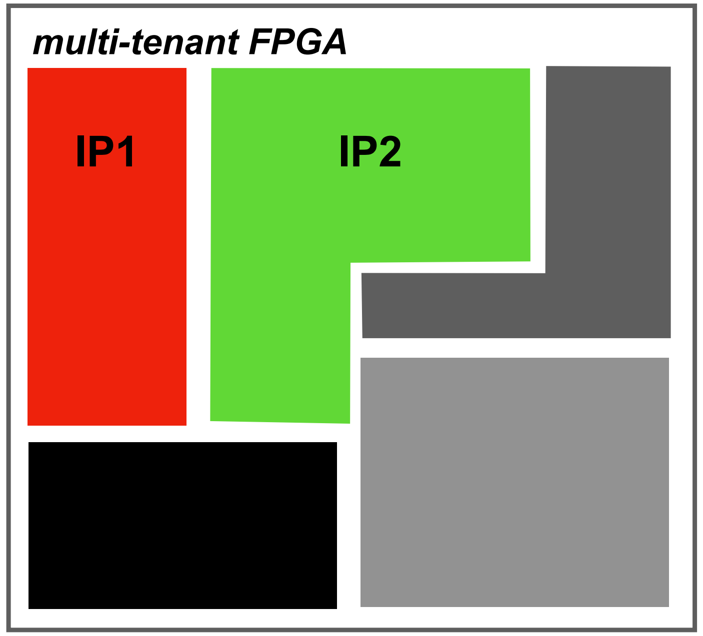
- Making efficient use of large FPGAs requires sharing FPGA fabric among different users, or untrusting IPs
- Mimics processor usage model in software systems
- Infrastructure for sharing among users is starting to come online [1]
- Flexibility of FPGA circuits allows new threats [2,3]
- Understand and mitigate threats to enable secure multi-tenant FPGA usage

[1] A. Khawaja, J. Landgraf, R. Prakash, M. Wei, E. Schkufza, and C. J. Rossbach, "Sharing, protection, and compatibility for reconfigurable fabric with AmorphOS," OSDI'18

[2] M. Zhao and G. E. Suh, "FPGA-Based Remote Power Side-Channel Attacks," S&P'18

[3] F. Schellenberg, D. Gnad, A. Moradi and M. Tahoori, "An inside job: Remote power analysis attacks on FPGAs," DATE'18

Threats in Multi-Tenant FPGAs

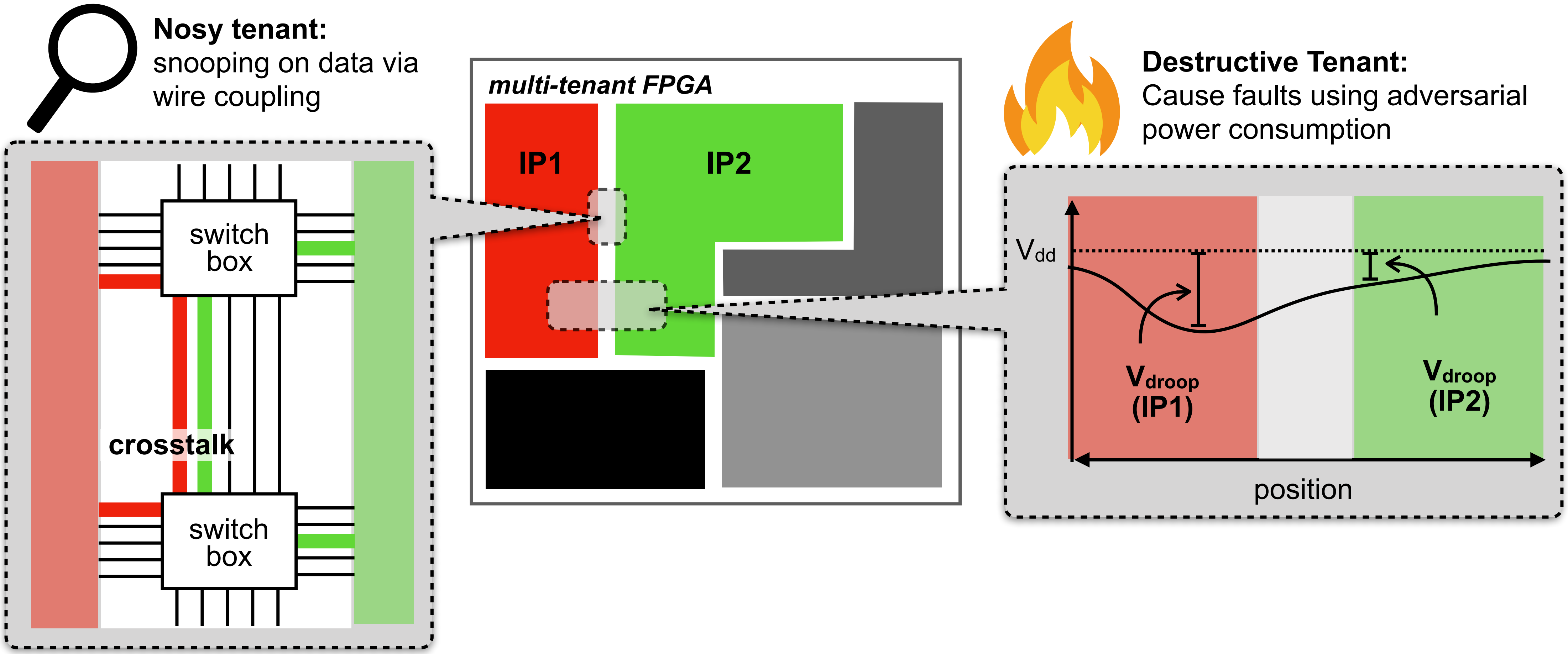


[1] C. Ramesh, S. B. Patil, S. N. Dhanuskodi, G. Provelengios, S. Pillement, D. Holcomb and R. Tessier. "FPGA Side Channel Attacks without Physical Access", FCCM'18

[2] G. Provelengios, C. Ramesh, S. B. Patil, K. Eguro, R. Tessier, and D. Holcomb. "Characterization of Long Wire Data Leakage in Deep Submicron FPGAs", FPGA'19

[3] G. Provelengios, D. Holcomb, and R. Tessier, "Characterizing Power Distribution Attacks in Multi-User FPGA Environments", FPL'19

Threats in Multi-Tenant FPGAs

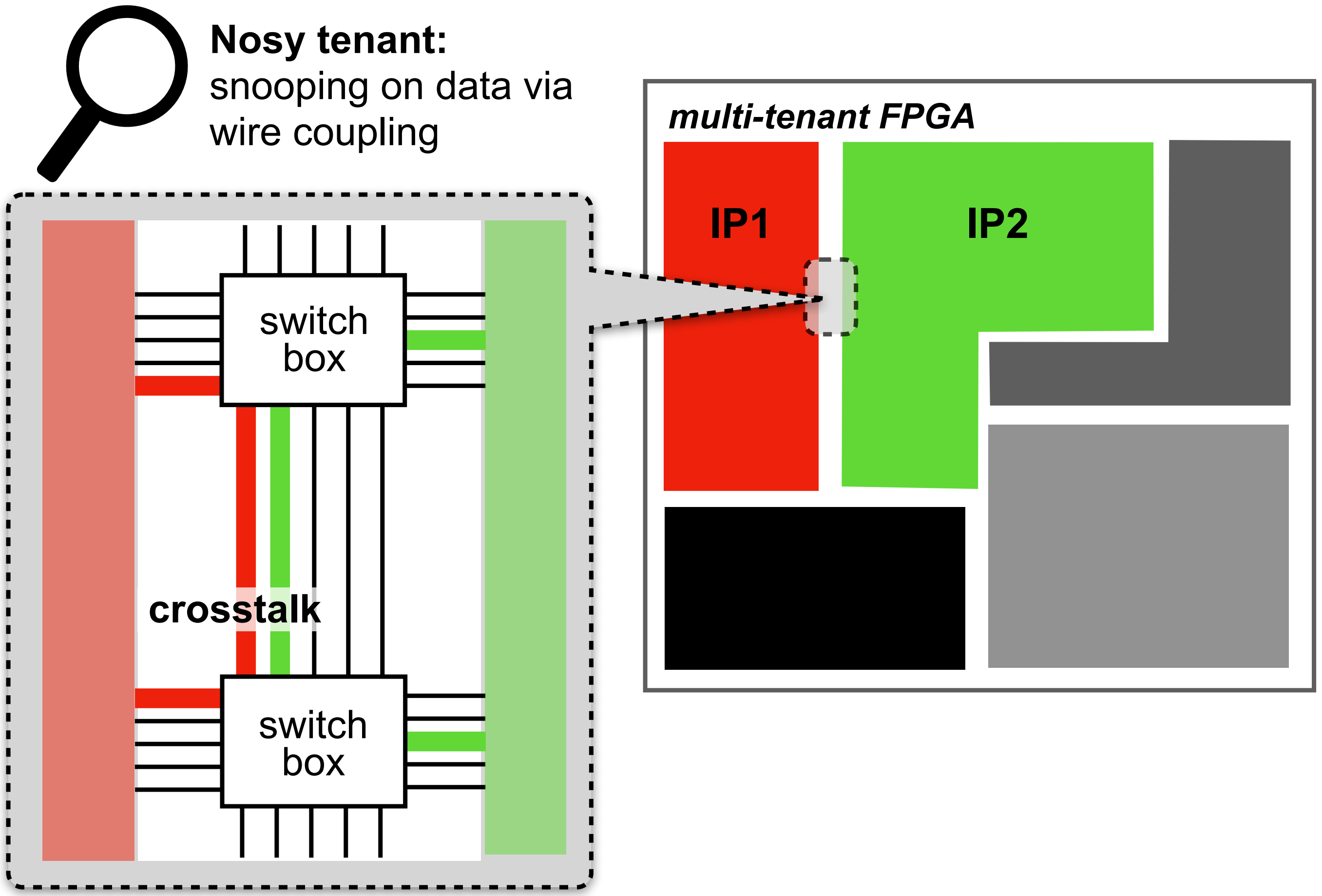


[1] C. Ramesh, S. B. Patil, S. N. Dhanuskodi, G. Provelengios, S. Pillement, D. Holcomb and R. Tessier. “FPGA Side Channel Attacks without Physical Access", FCCM'18

[2] G. Provelengios, C. Ramesh, S. B. Patil, K. Eguro, R. Tessier, and D. Holcomb. "Characterization of Long Wire Data Leakage in Deep Submicron FPGAs", FPGA'19

[3] G. Provelengios, D. Holcomb, and R. Tessier, "Characterizing Power Distribution Attacks in Multi-User FPGA Environments", FPL'19

Threats in Multi-Tenant FPGAs



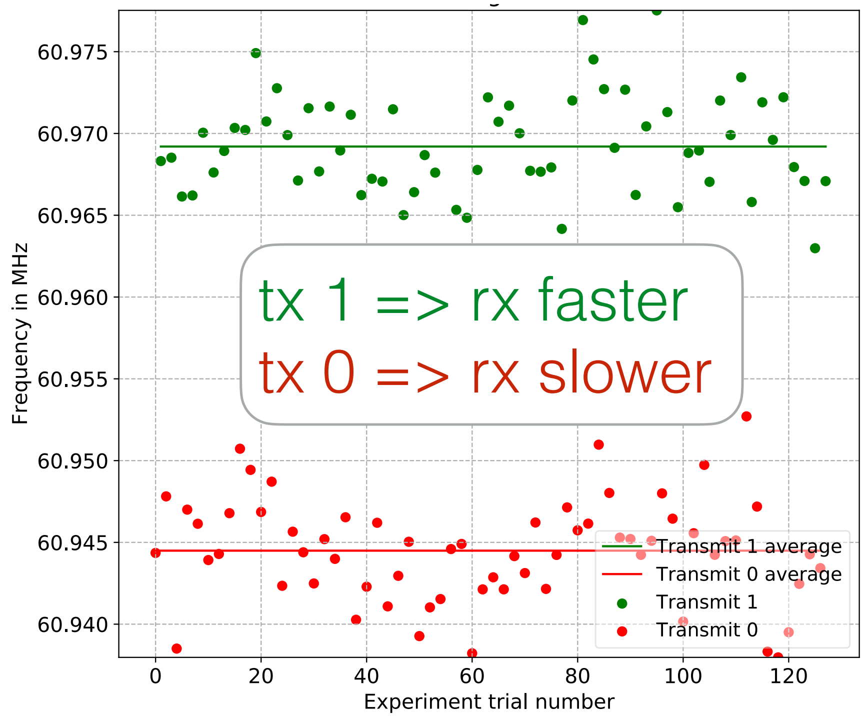
[1] C. Ramesh, S. B. Patil, S. N. Dhanuskodi, G. Provelengios, S. Pillement, D. Holcomb and R. Tessier. “FPGA Side Channel Attacks without Physical Access", FCCM'18

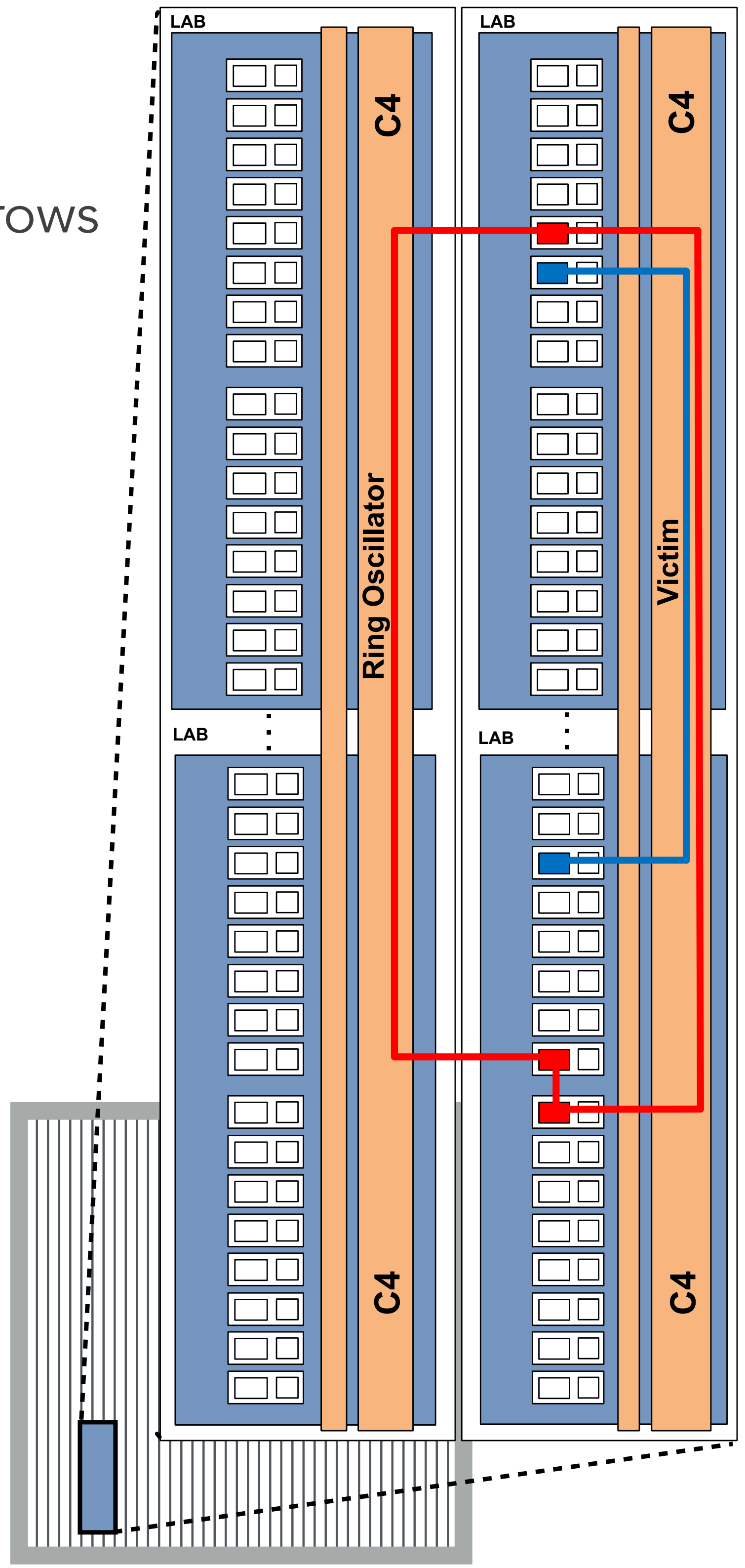
[2] G. Provelengios, C. Ramesh, S. B. Patil, K. Eguro, R. Tessier, and D. Holcomb. "Characterization of Long Wire Data Leakage in Deep Submicron FPGAs", FPGA'19

[3] G. Provelengios, D. Holcomb, and R. Tessier, "Characterizing Power Distribution Attacks in Multi-User FPGA Environments", FPL'19

Wire Coupling

- FPGA long wires span multiple LABs along columns or rows
- Ring oscillator (receiver) routed next to signal wire (transmitter) changes its frequency depending on the transmitter's DC value
- Observed on both Xilinx [1] and Intel [2] FPGAs

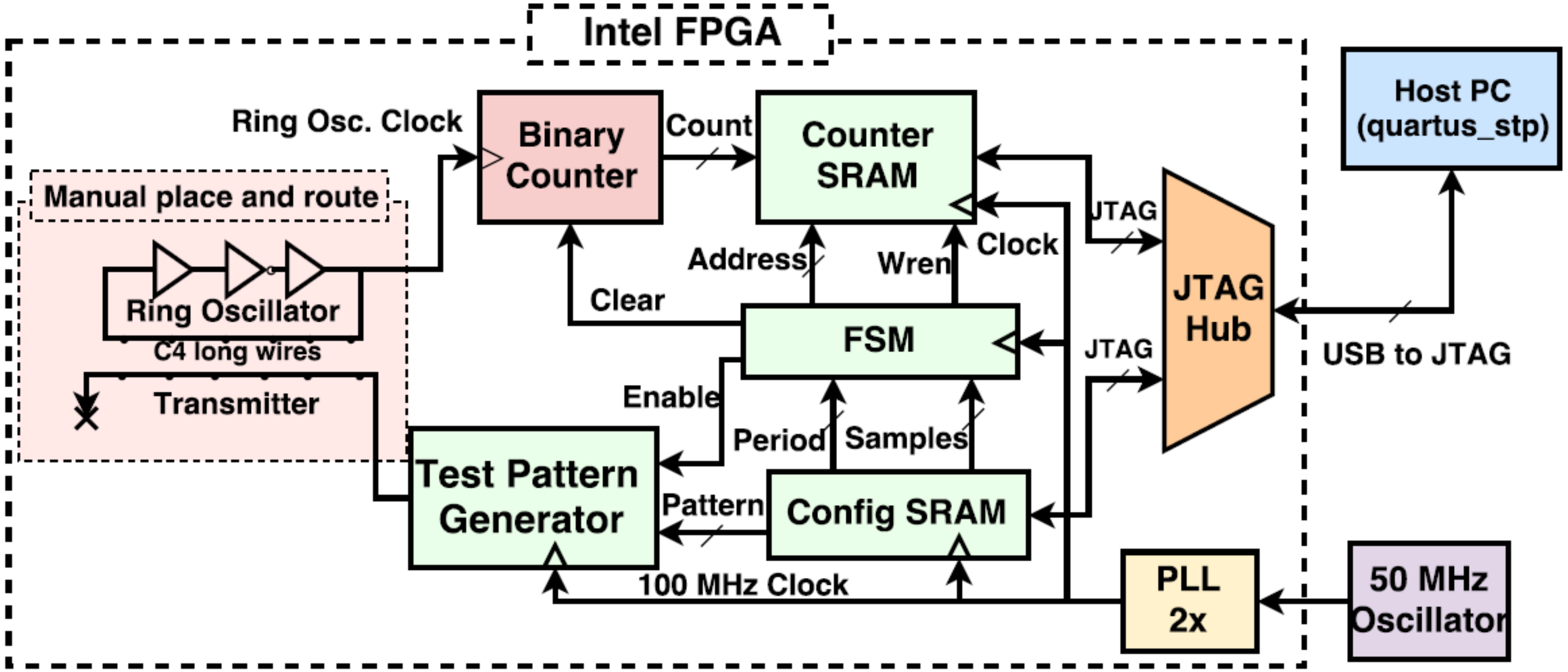




[1] I. Giechaskiel, K. Rasmussen, and K. Eguro. "Leaky Wires: Information Leakage and Covert Communication Between FPGA Long Wires." ASIACCS'18

[2] C. Ramesh, S. B. Patil, S. N. Dhanuskodi, G. Provelengios, S. Pillement, D. Holcomb and R. Tessier, "FPGA Side Channel Attacks without Physical Access", FCCM'18

Characterization Setup



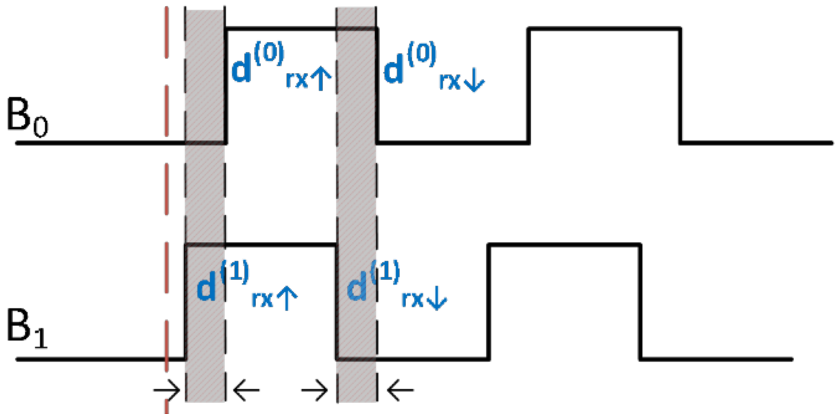
$$\Delta RC = \frac{C^1 - C^0}{C^1}$$

- Relative change in RO count
- On order of 0.0001

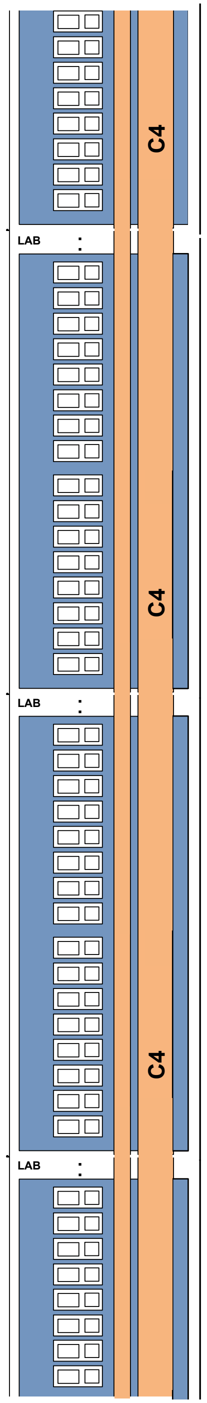
$$\Delta t = \left(\frac{1}{f^0} - \frac{1}{f^1}\right)/2$$

$$= \left(\left(d^0_{rx\uparrow} - d^1_{rx\uparrow}\right) + \left(d^0_{rx\downarrow} - d^1_{rx\downarrow}\right)\right)/2$$

- Change in propagation delay
- On order of femtoseconds

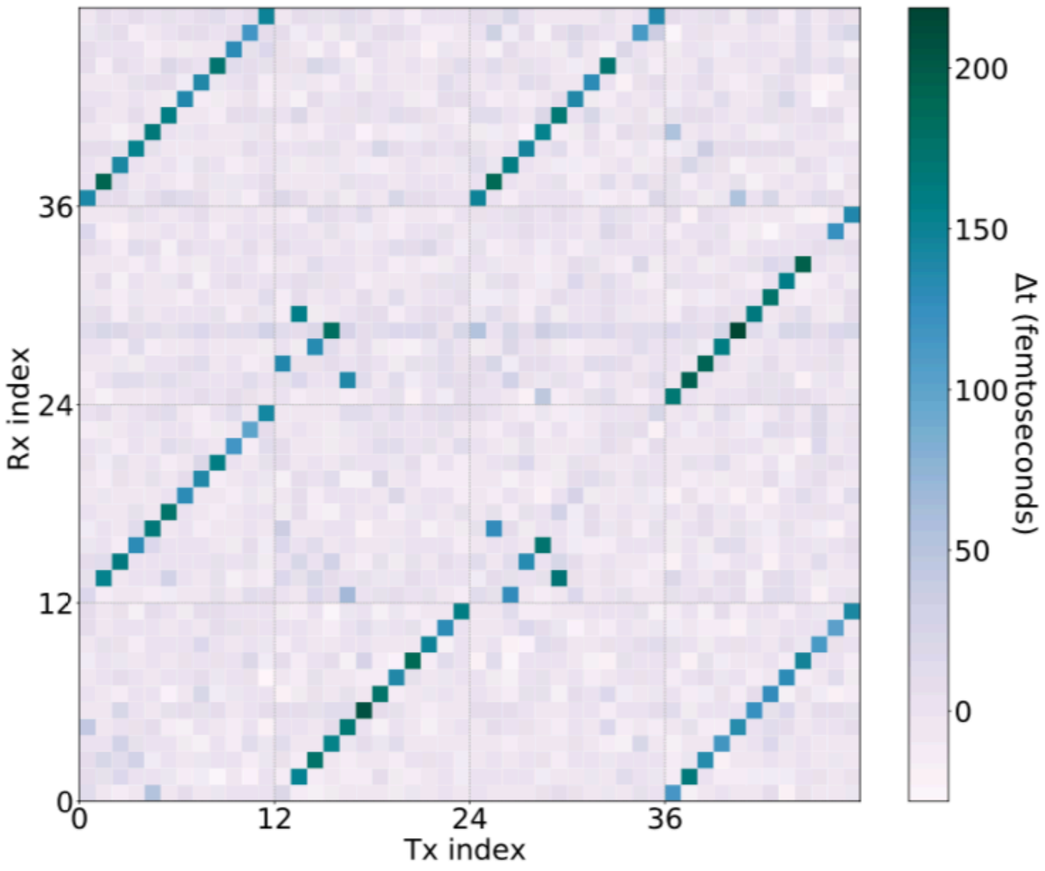


Meet the Neighbors

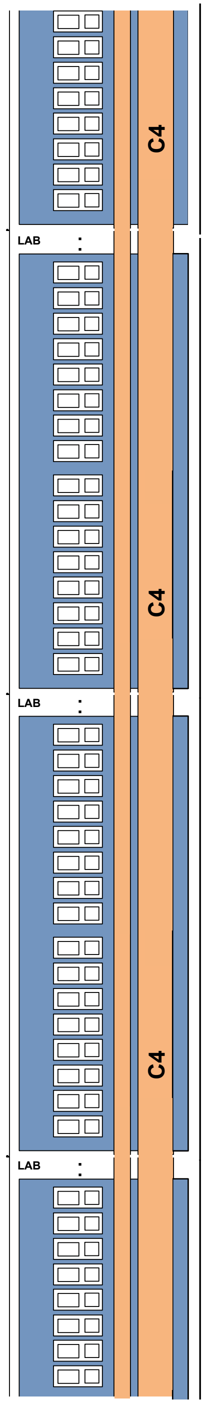


- Quartus doesn't reveal adjacency in channel, but can infer it by searching for pairings with coupling
- Testing all wire pairings in CycloneIV C4 channel:
  - Two neighbors per wire
  - Coupling is bidirectional
  - Comparable magnitude

[Provelengios *et al.* FPGA'19]

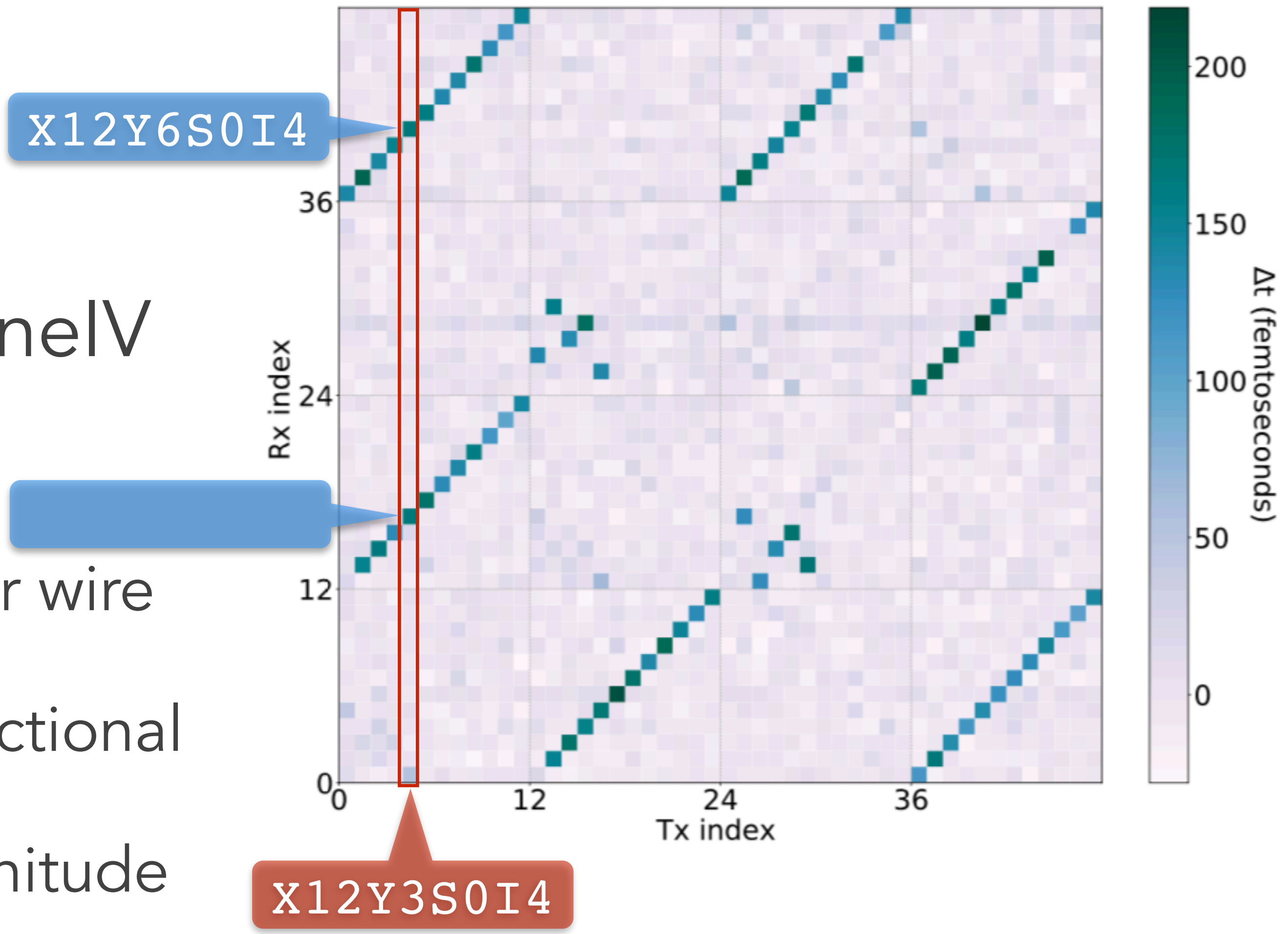


Meet the Neighbors

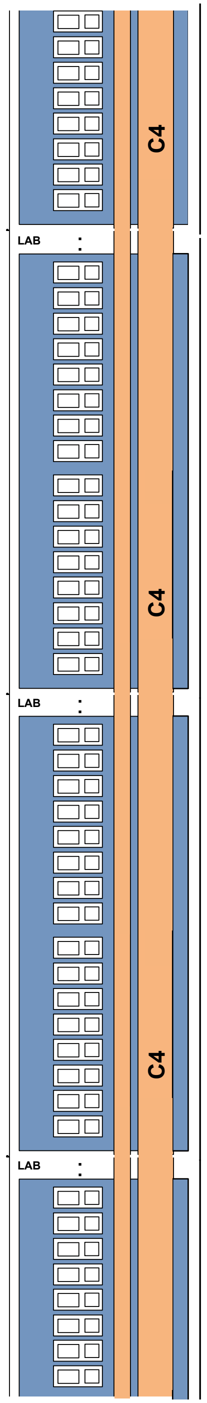


- Quartus doesn't reveal adjacency in channel, but can infer it by searching for pairings with coupling
- Testing all wire pairings in CycloneIV C4 channel:
  - Two neighbors per wire
  - Coupling is bidirectional
  - Comparable magnitude

[Provelengios *et al.* FPGA'19]

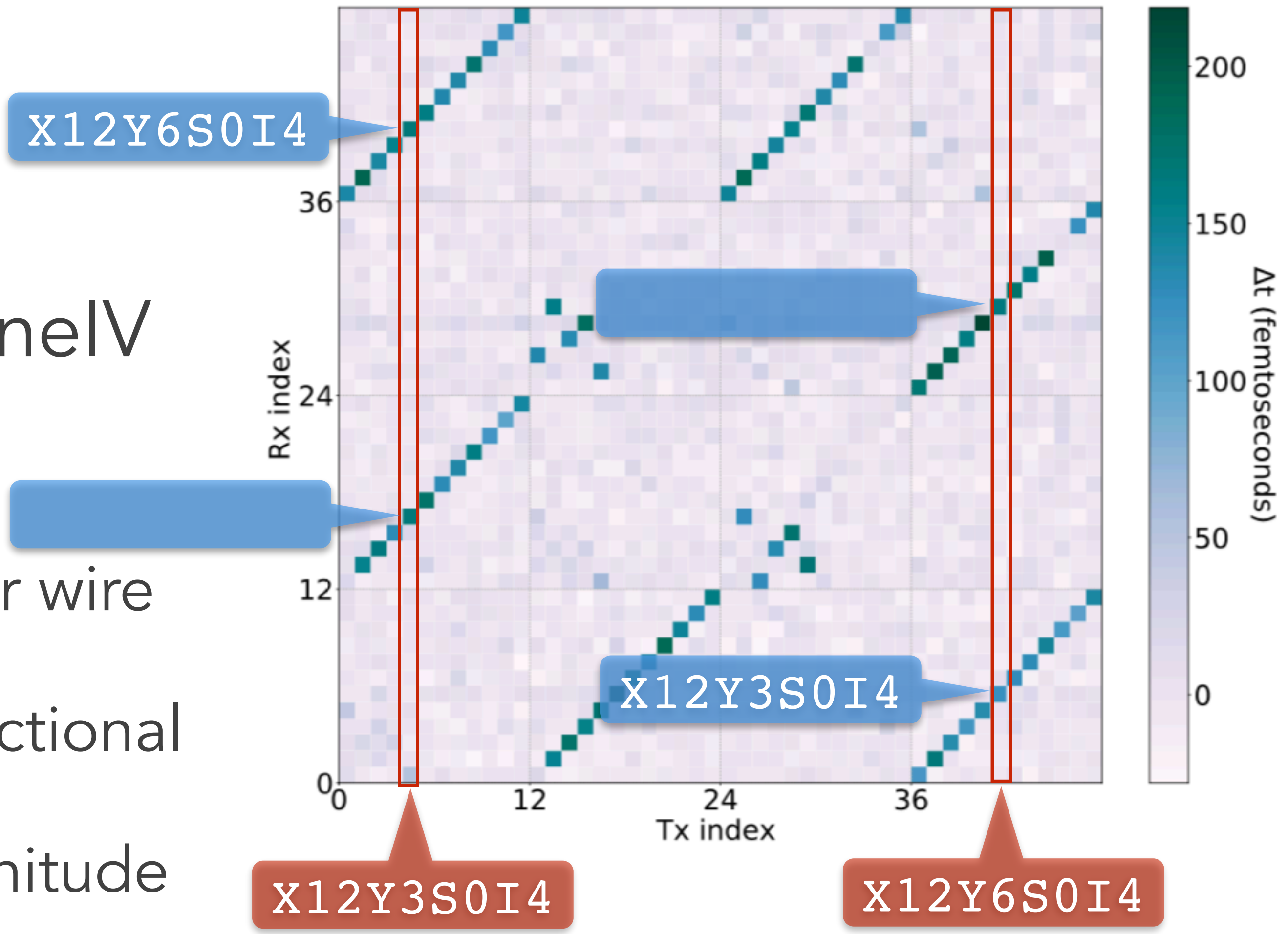


Meet the Neighbors



- Quartus doesn't reveal adjacency in channel, but can infer it by searching for pairings with coupling
- Testing all wire pairings in CycloneIV C4 channel:
  - Two neighbors per wire
  - Coupling is bidirectional
  - Comparable magnitude

[Provelengios *et al.* FPGA'19]

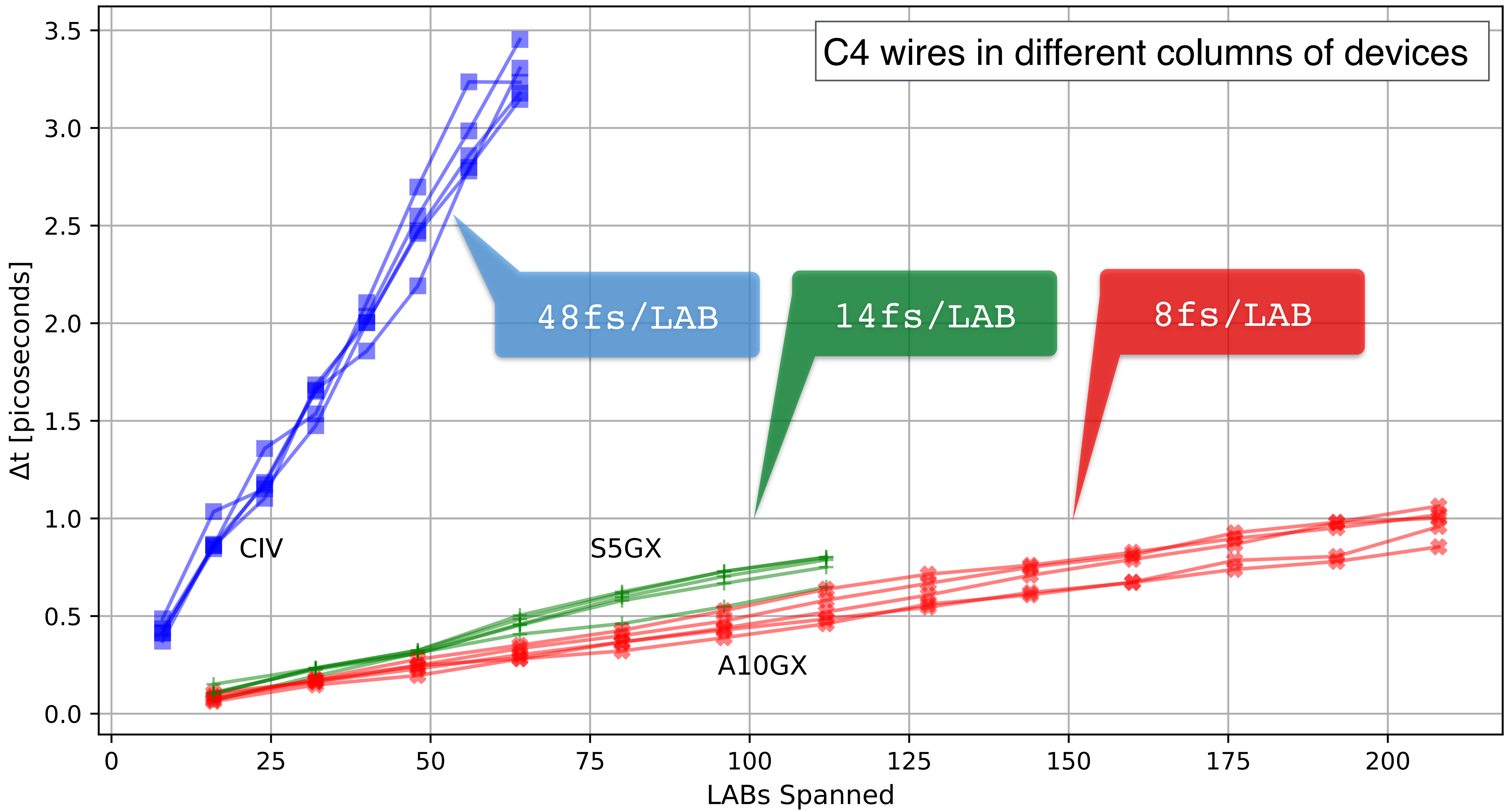


Immediate Neighbors Only

configuration								impact
baseline	0	0	0	RO	0	0	0	-
1 neighbor	0	0	0	RO	1	0	0	1x
1 neighbor	0	0	1	RO	0	0	0	1x
2 neighbors	0	0	1	RO	1	0	0	2x
non-immediate neighbor	1	1	0	RO	0	1	1	0

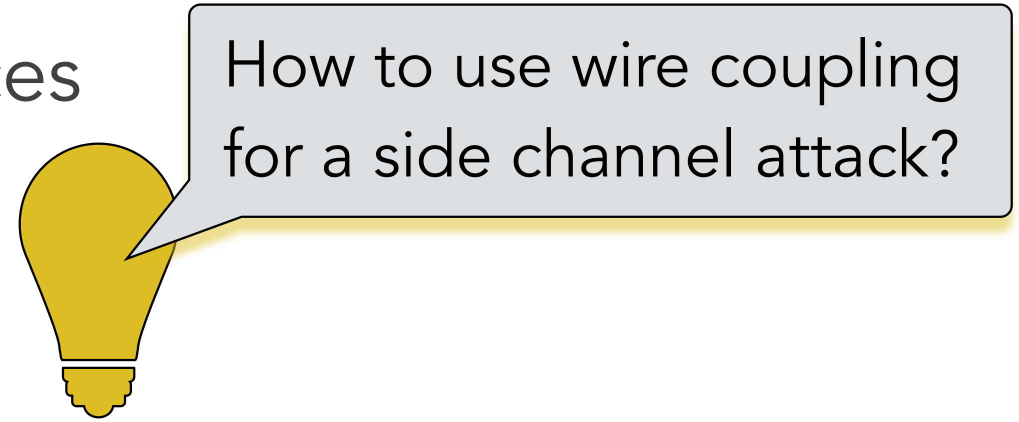
Proportional to Adjacency Length

[Provelengios *et al.* FPGA'19]



Summary of Coupling

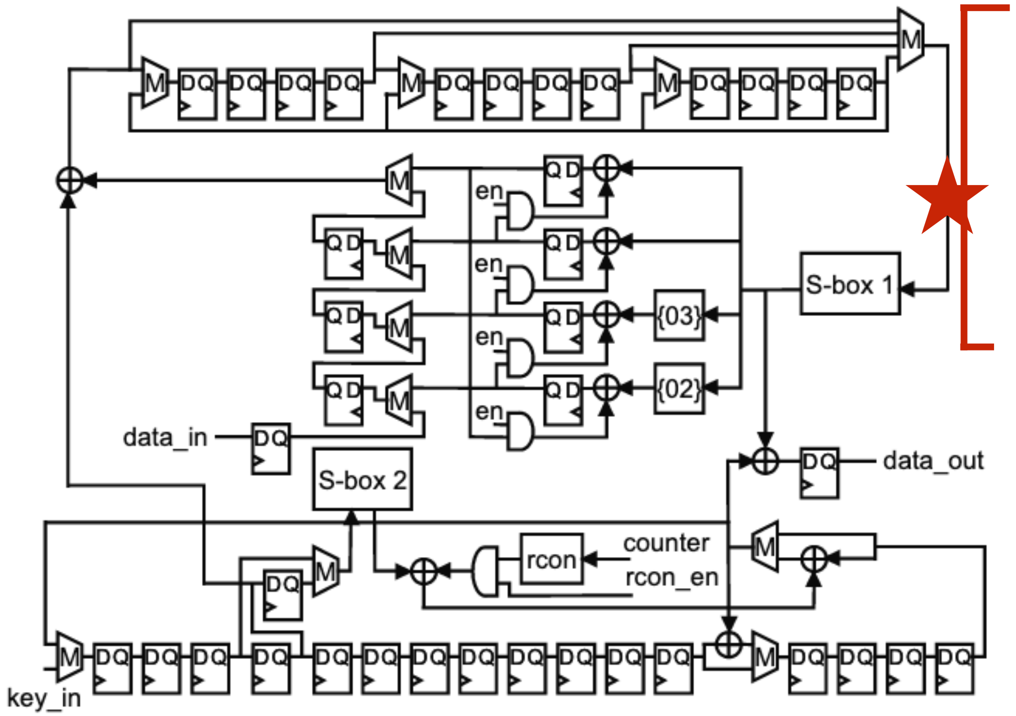
- Exists across devices and various long wire types
  - Cyclone IV (60nm), Stratix V (28nm), Arria 10 (20nm)
  - 10-50 fs per LAB of adjacency
  - 0.01% slowdown in typical RO
- Bidirectional, DC, only between immediate neighbors
- Consistent across chip instances



Cyclone IV GX (EP4CGX150DF31)
Altera DE2-115 (EP4CE115F29)
Stratix V (5SGXEA7K2F40C2N)
Arria 10 GX (10AX115N2F45E1SG)

FPGA Side Channel Attack

- Victim circuit is AES with 8-bit datapath [1][2]
- Attacker knows placement and routing of AES design
- Attacker can route sensor/receiver wire next to one of 8 S-Box inputs
- Extract last round subkey using coupling as side channel



Why 8-bit AES?

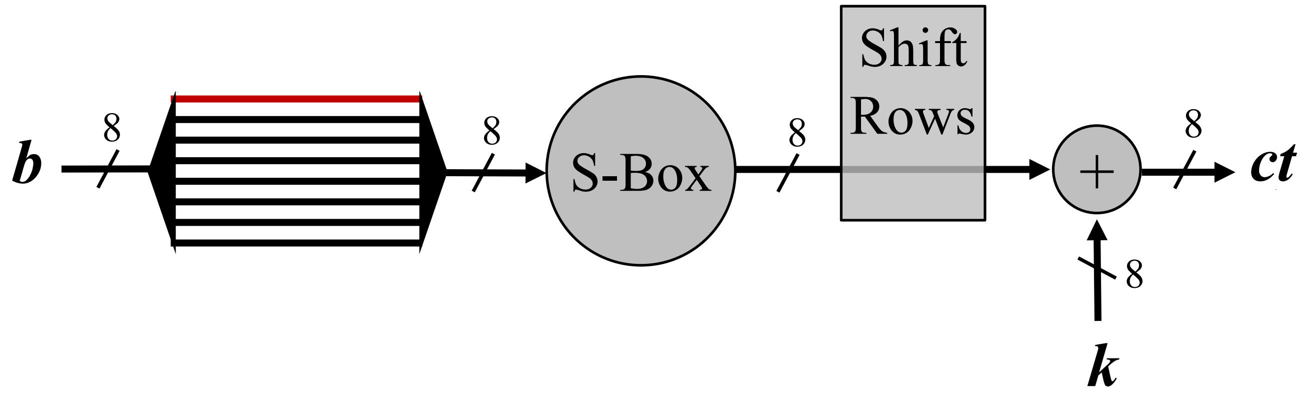
- 16 Bytes use same S-Box
- Leakage from S-Box input can reveal all bytes
- Attack each byte in different cycle of last round

[1] C. Jin, "8bit datapath hardware implementation of AES," https://github.com/ChengluJin/8bit_datapath_AES

[2] P. Hamalainen, T. Alho, M. Hannikainen and T. D. Hamalainen, "Design and Implementation of Low-Area and Low-Power AES Encryption Hardware Core," DSD'06

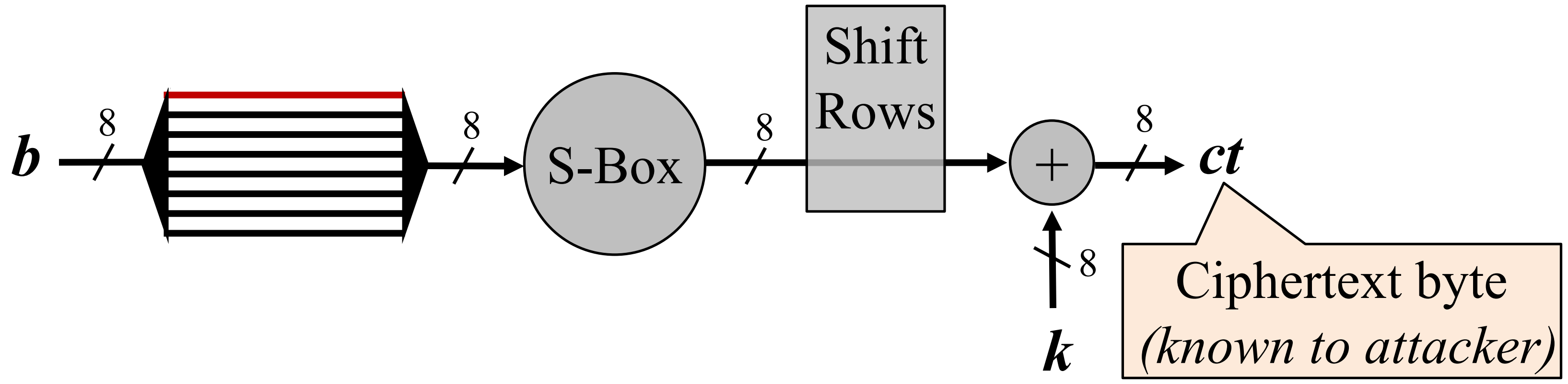
FPGA Side Channel Attack

- Victim circuit is AES with 8-bit datapath [1][2]
- Attacker knows placement and routing of AES design
- Attacker can route sensor/receiver wire next to one of 8 S-Box inputs
- Extract last round subkey using coupling as side channel



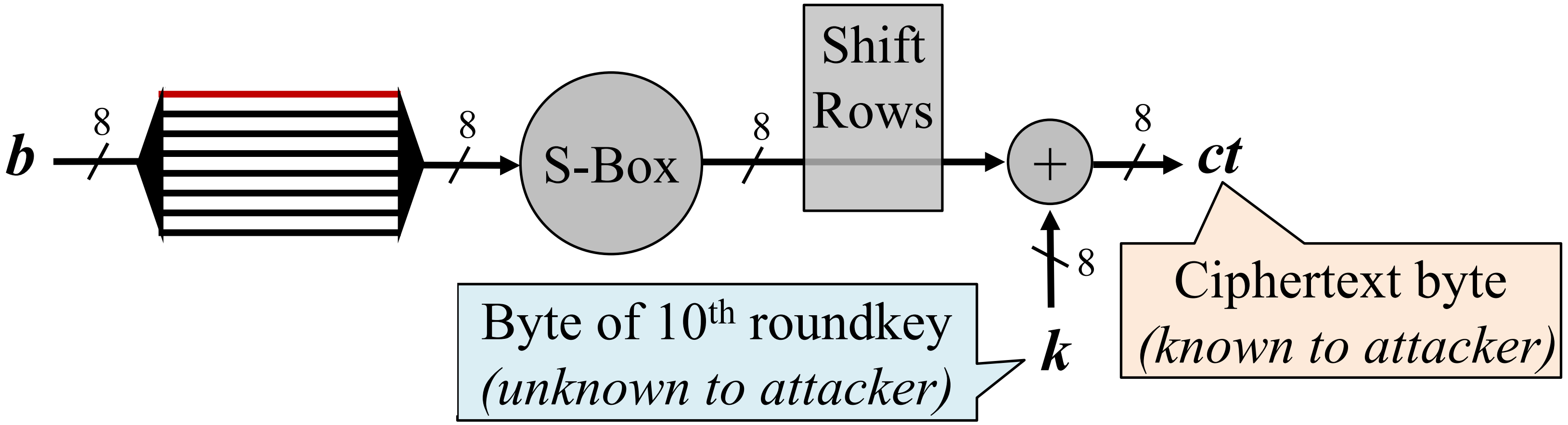
FPGA Side Channel Attack

- Victim circuit is AES with 8-bit datapath [1][2]
- Attacker knows placement and routing of AES design
- Attacker can route sensor/receiver wire next to one of 8 S-Box inputs
- Extract last round subkey using coupling as side channel



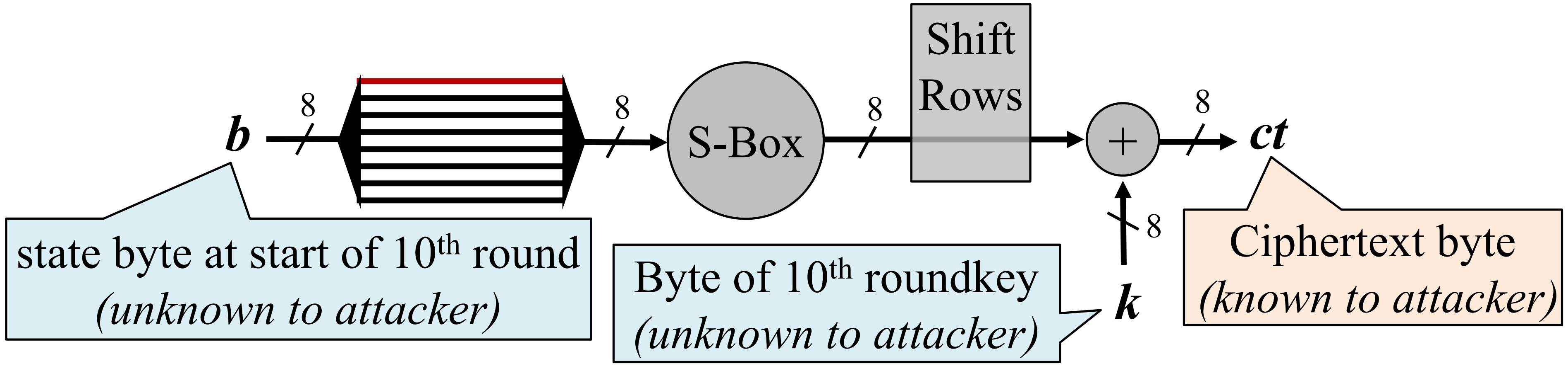
FPGA Side Channel Attack

- Victim circuit is AES with 8-bit datapath [1][2]
- Attacker knows placement and routing of AES design
- Attacker can route sensor/receiver wire next to one of 8 S-Box inputs
- Extract last round subkey using coupling as side channel



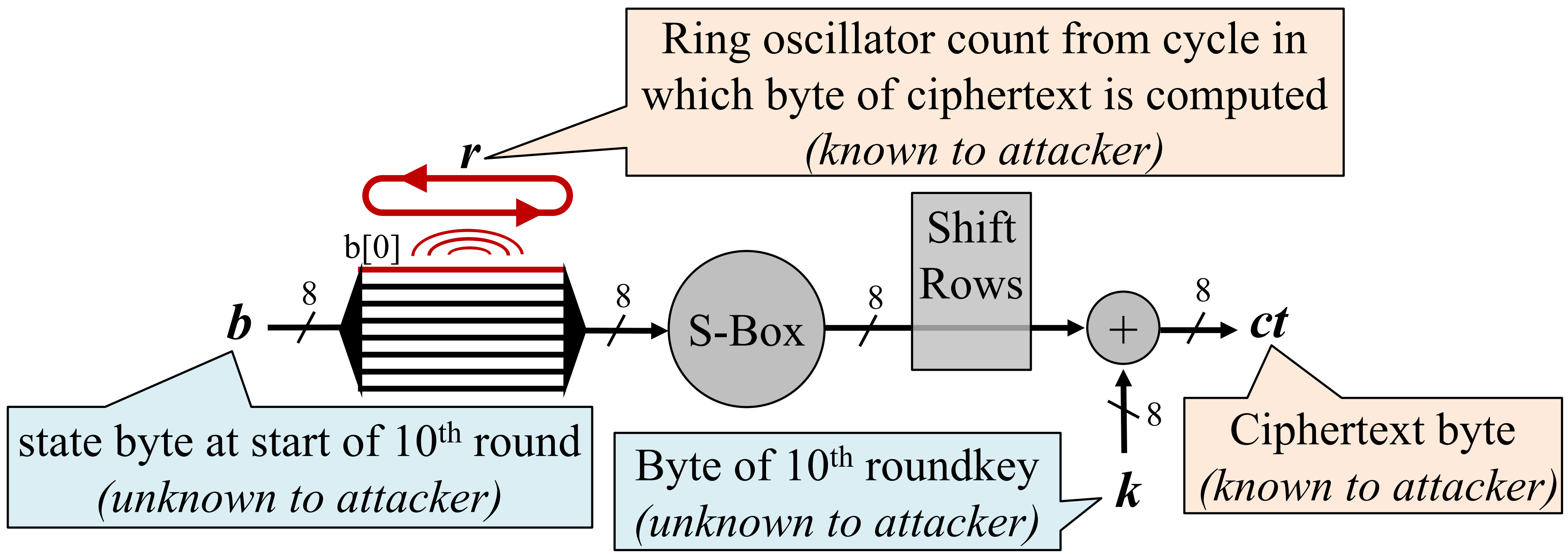
FPGA Side Channel Attack

- Victim circuit is AES with 8-bit datapath [1][2]
- Attacker knows placement and routing of AES design
- Attacker can route sensor/receiver wire next to one of 8 S-Box inputs
- Extract last round subkey using coupling as side channel



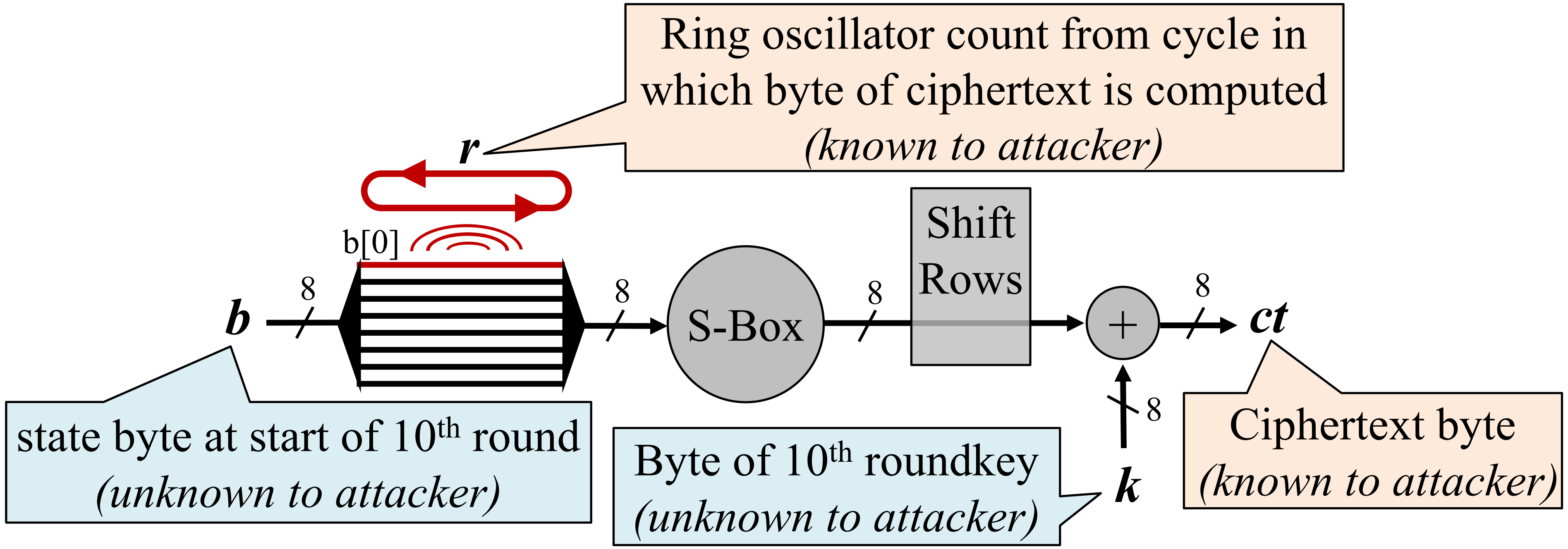
FPGA Side Channel Attack

- Victim circuit is AES with 8-bit datapath [1][2]
- Attacker knows placement and routing of AES design
- Attacker can route sensor/receiver wire next to one of 8 S-Box inputs
- Extract last round subkey using coupling as side channel

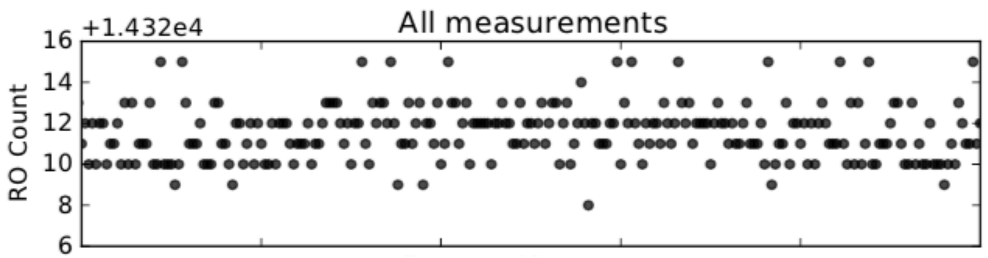


FPGA Side Channel Attack

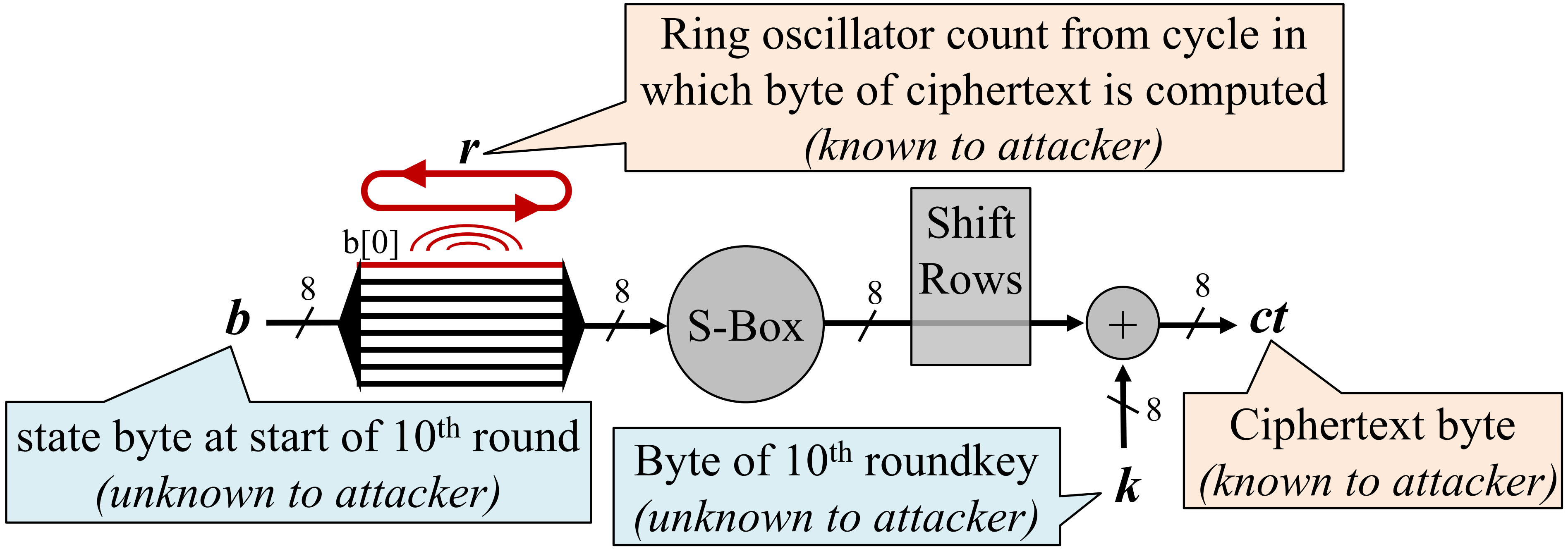
- Correlation between osc. counts and key-predicted wire values confirms one guess as correct key byte
- Repeat for each of 16 bytes by targeting different cycles of encryption



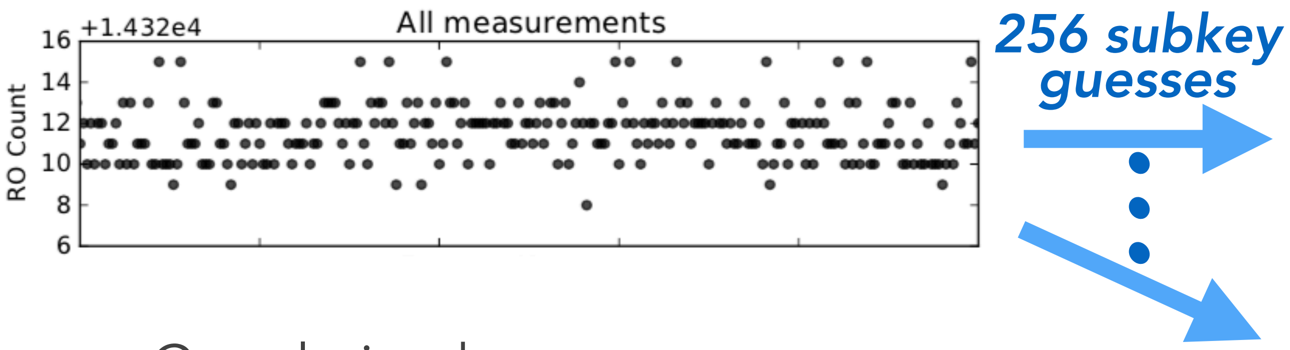
FPGA Side Channel Attack



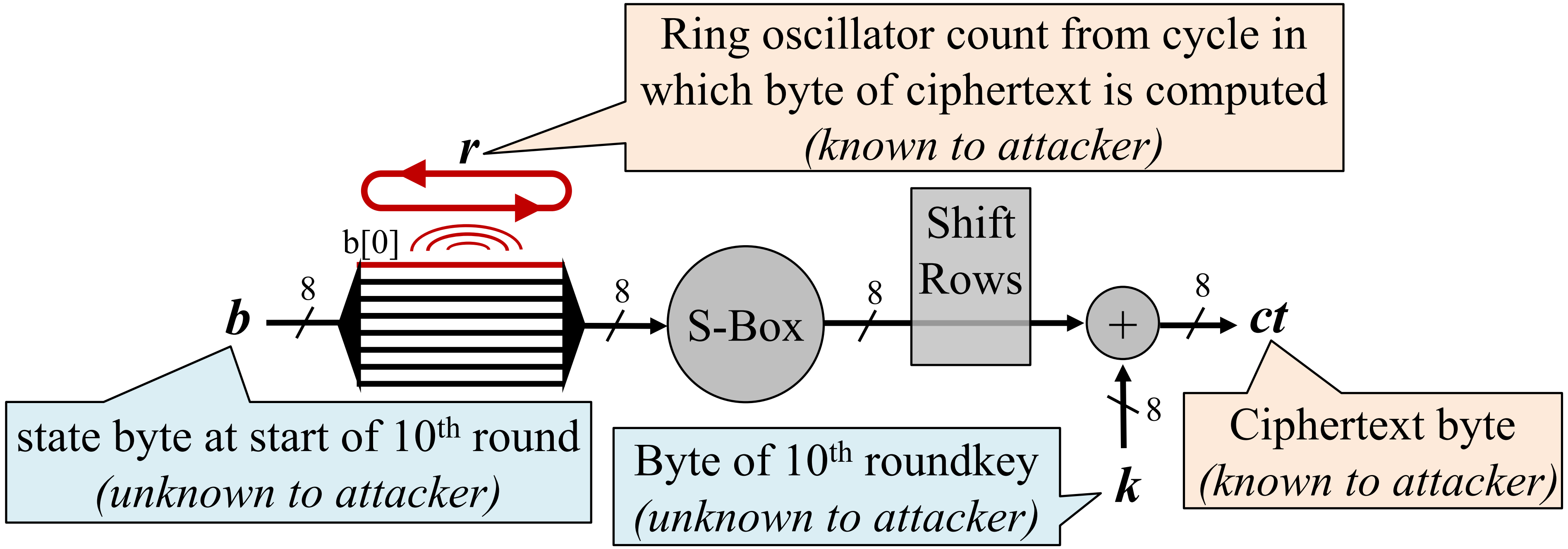
- Correlation between osc. counts and key-predicted wire values confirms one guess as correct key byte
- Repeat for each of 16 bytes by targeting different cycles of encryption



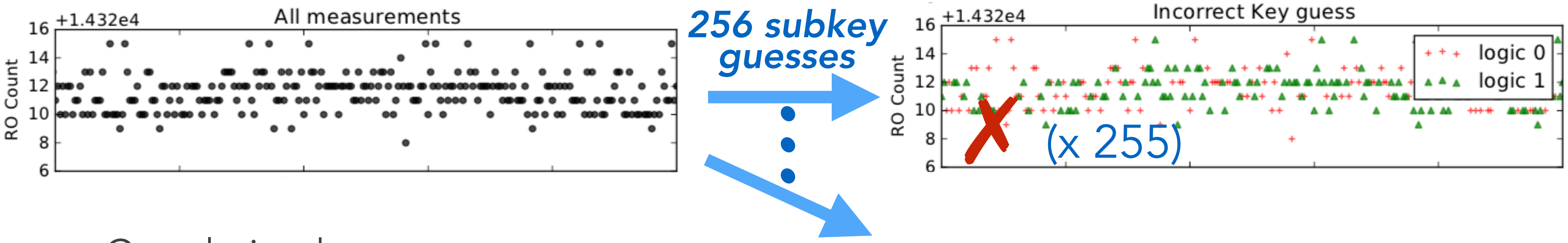
FPGA Side Channel Attack



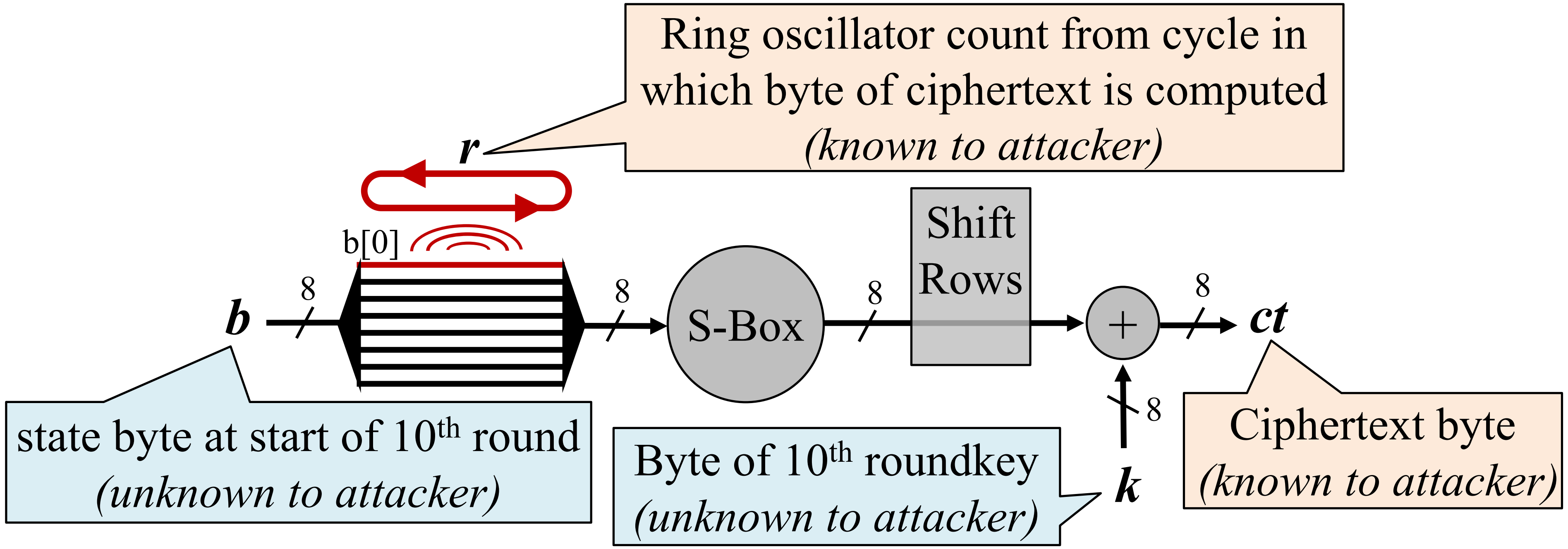
- Correlation between osc. counts and key-predicted wire values confirms one guess as correct key byte
- Repeat for each of 16 bytes by targeting different cycles of encryption



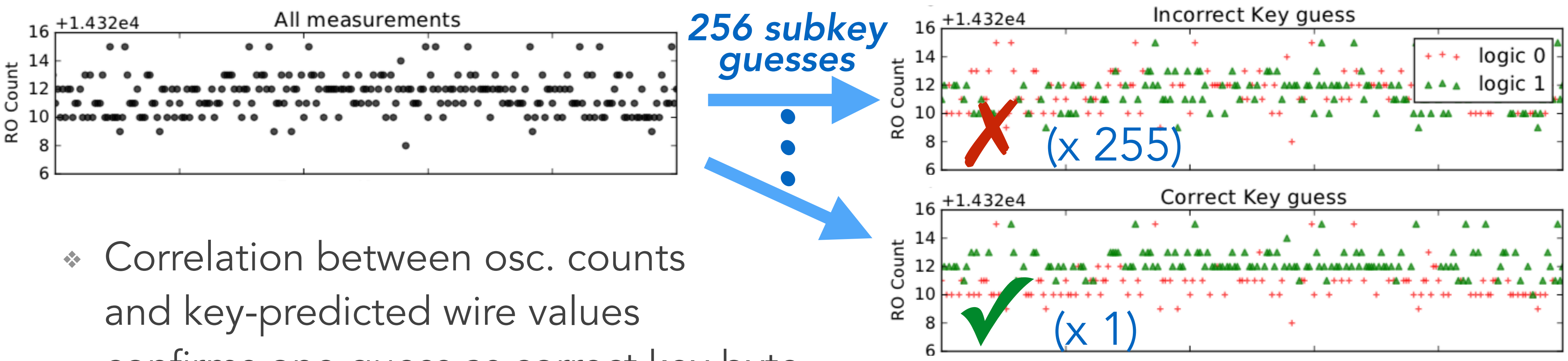
FPGA Side Channel Attack



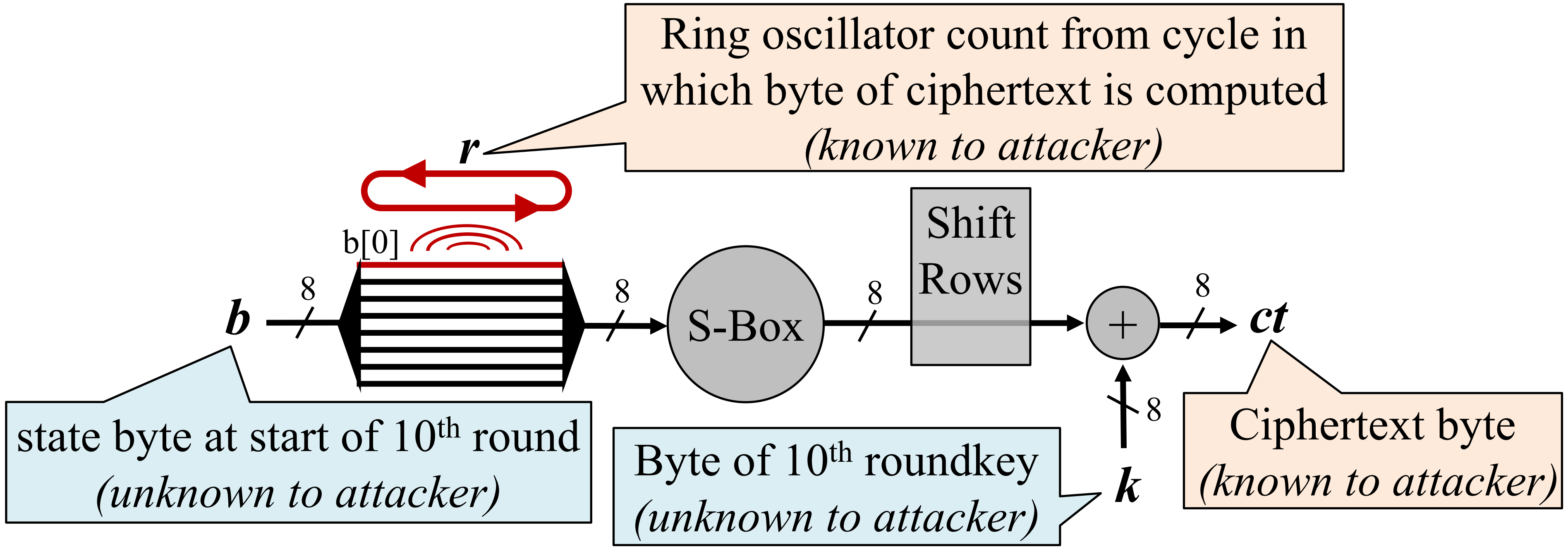
- Correlation between osc. counts and key-predicted wire values confirms one guess as correct key byte
- Repeat for each of 16 bytes by targeting different cycles of encryption



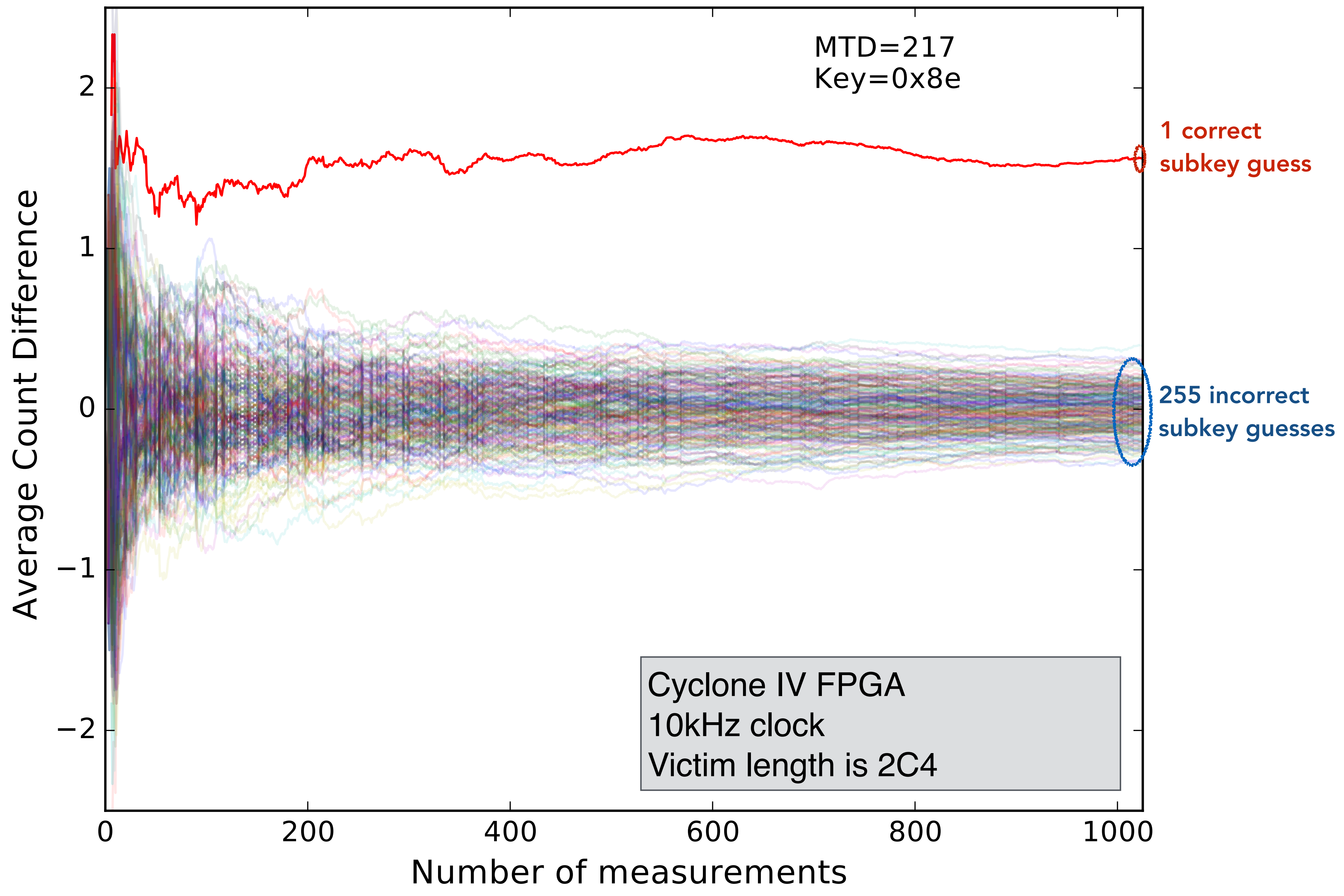
FPGA Side Channel Attack



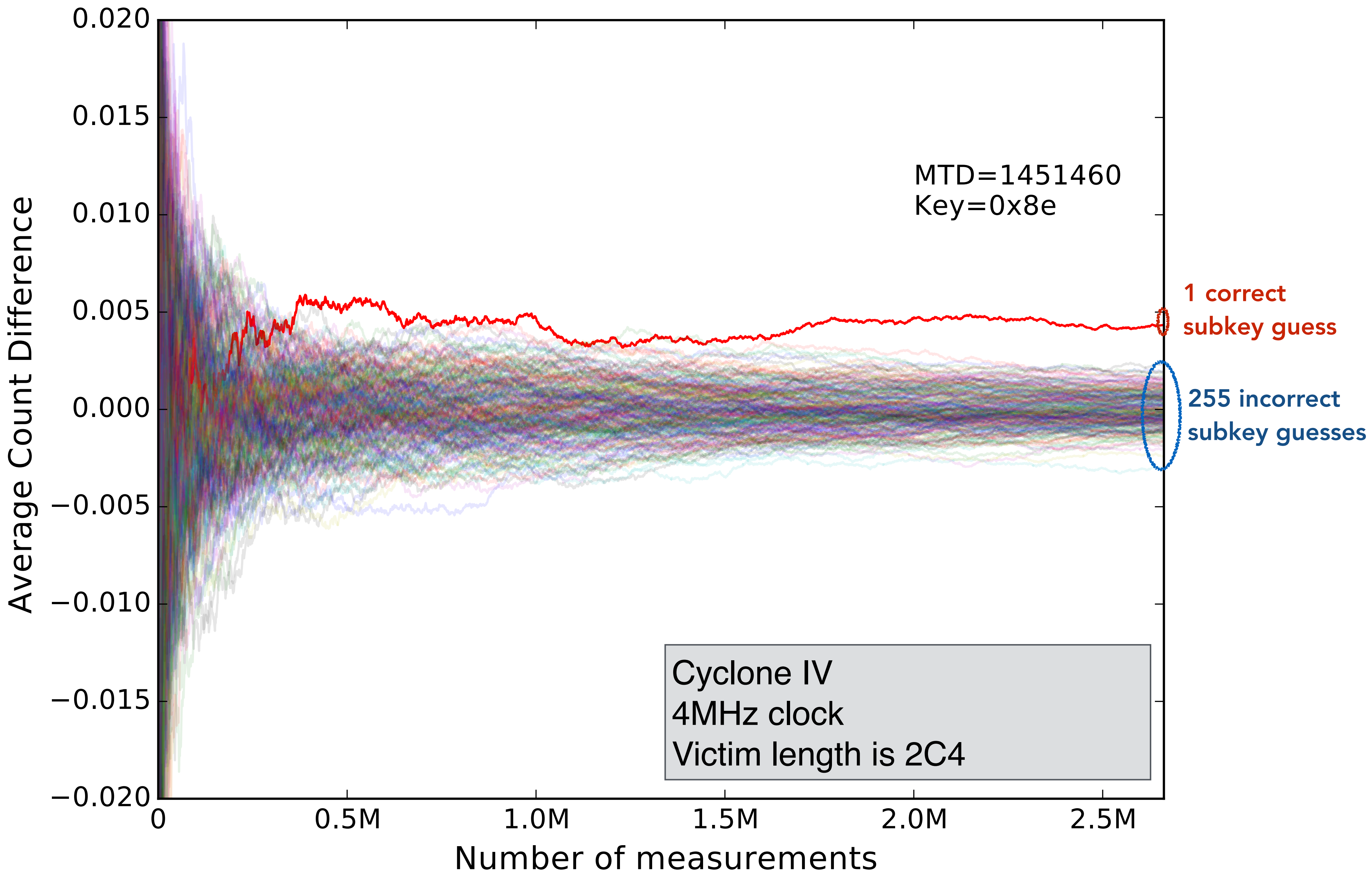
- Correlation between osc. counts and key-predicted wire values confirms one guess as correct key byte
- Repeat for each of 16 bytes by targeting different cycles of encryption



FPGA Side Channel Attack

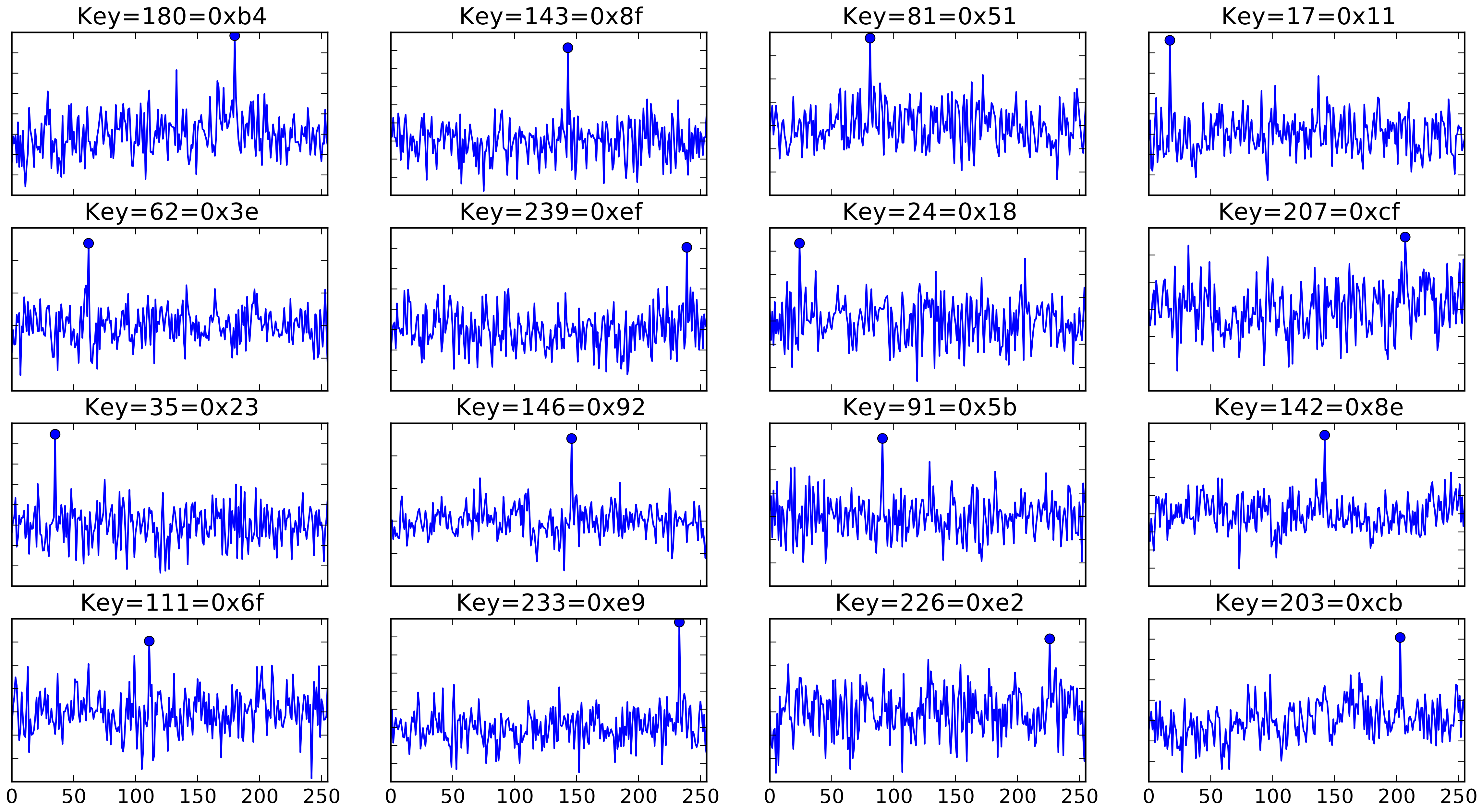


FPGA Side Channel Attack

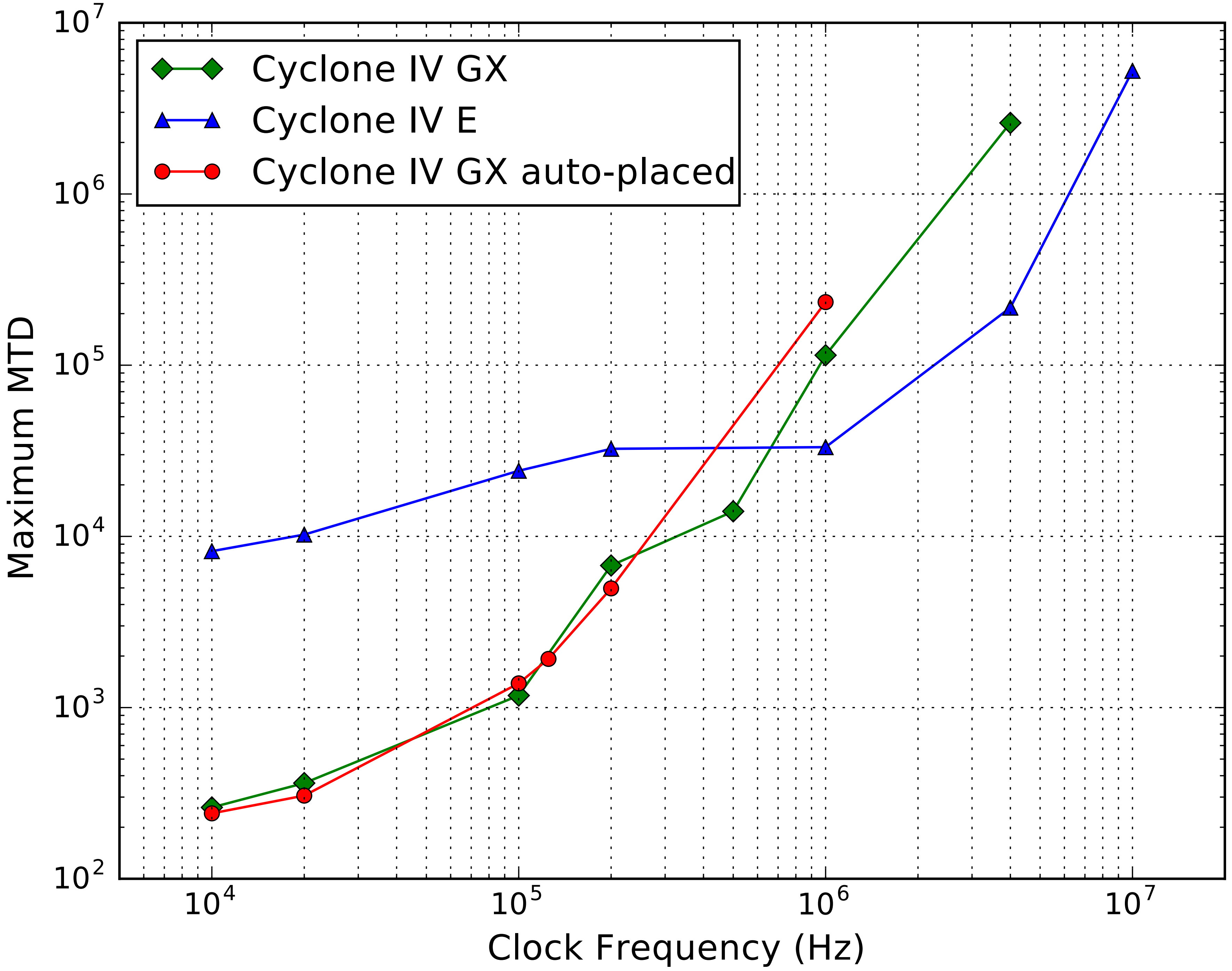


Side Channel Attack

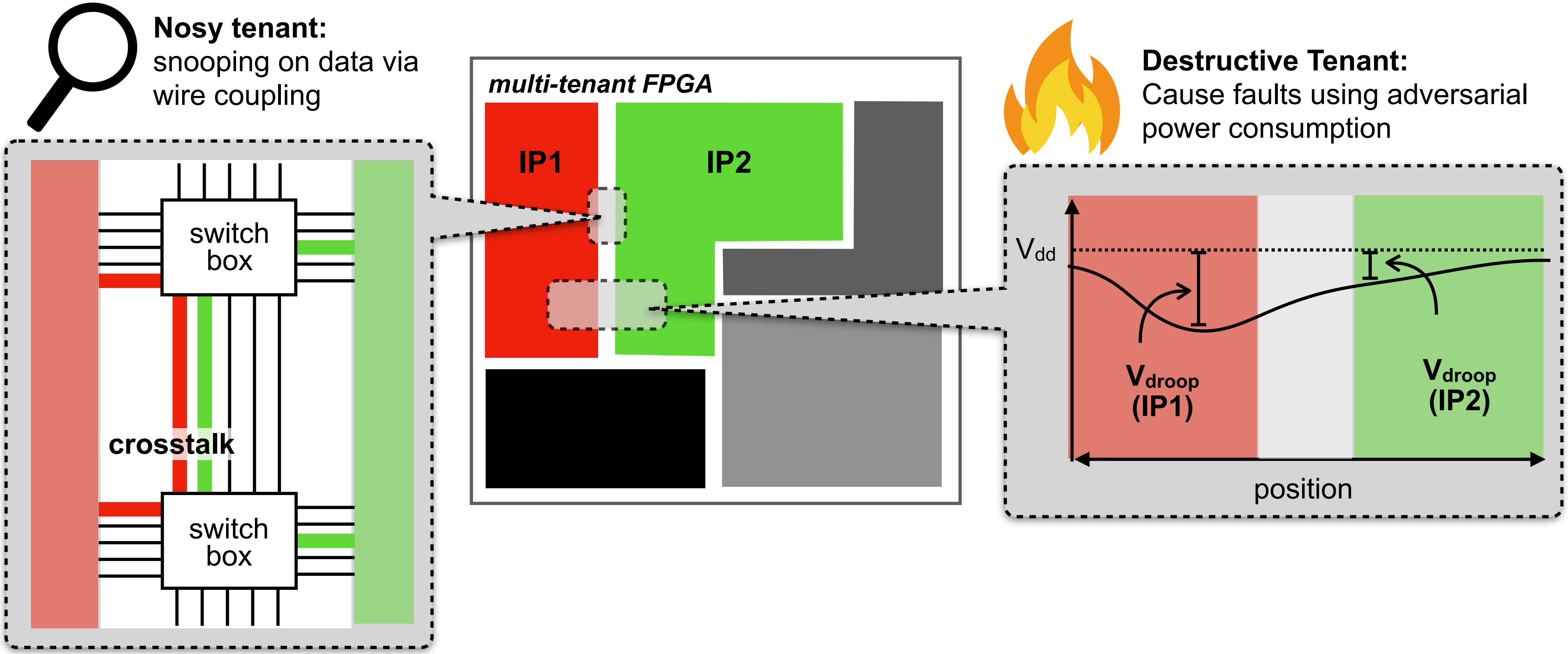
Cyclone IV FPGA
2.6M encryptions
4MHz clock
Victim length is 2C4 wires



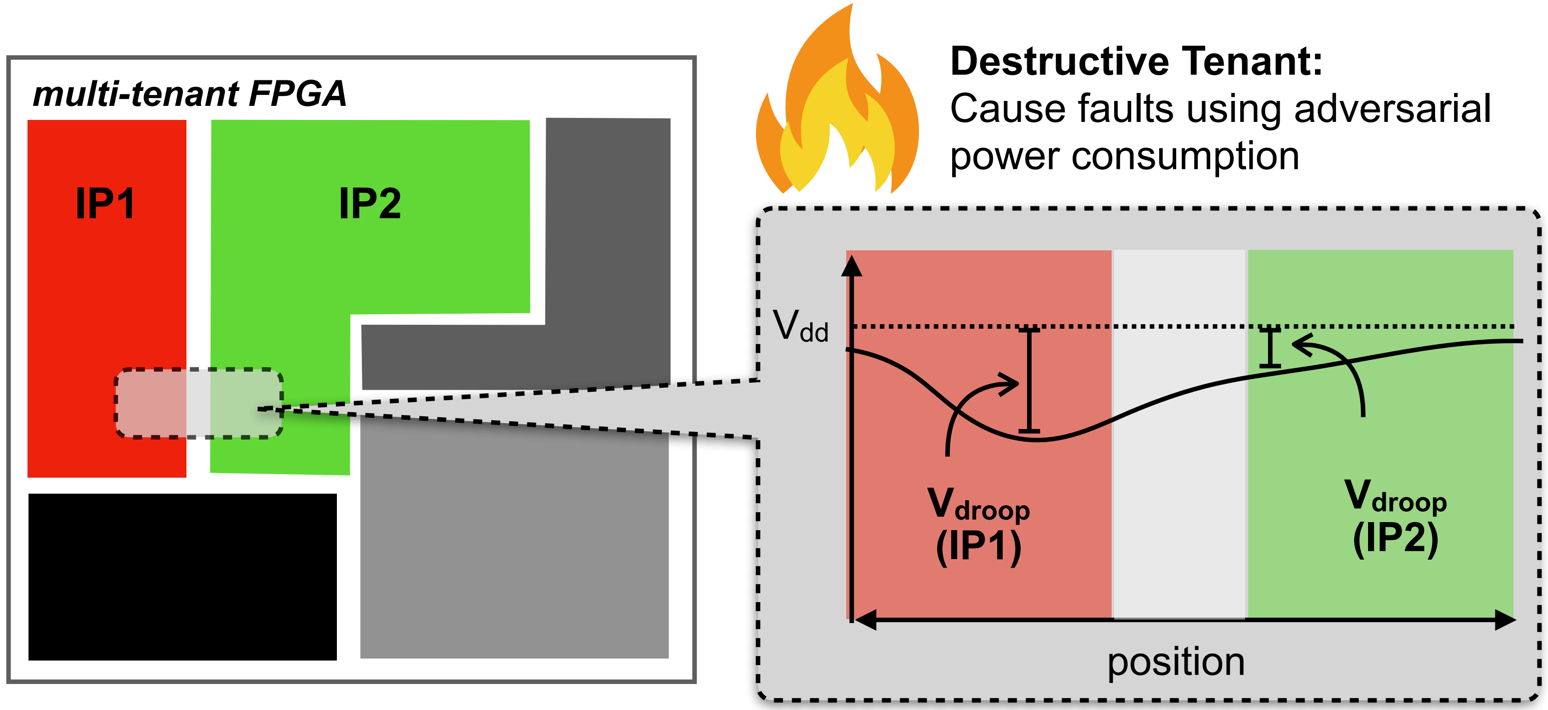
Side Channel Attacks



Threats in Multi-Tenant FPGAs

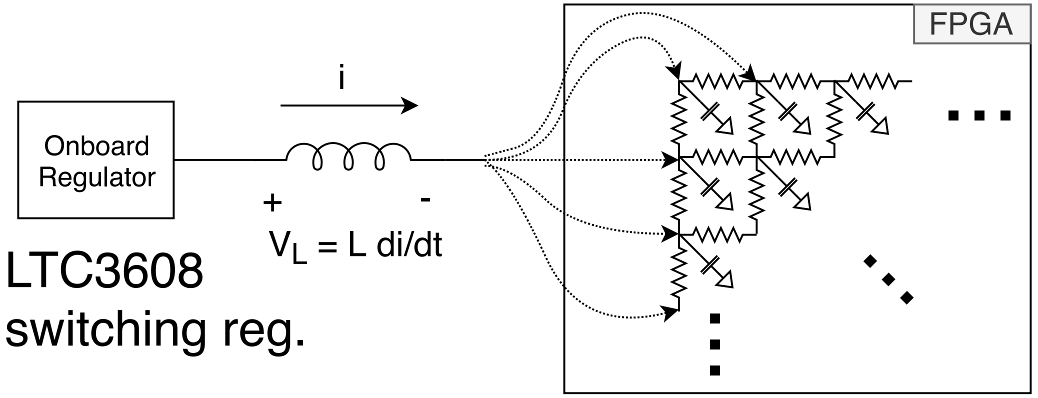


Threats in Multi-Tenant FPGAs

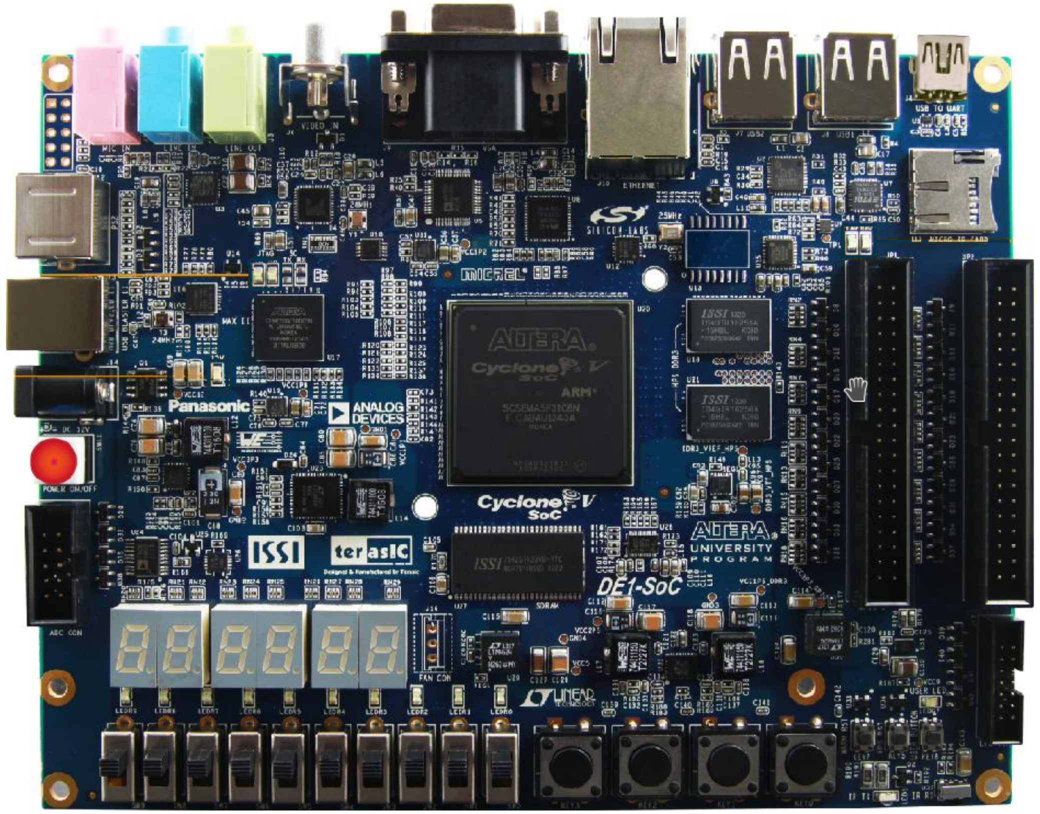


Destructive Tenant

- Tenant 1 is destructive, wants to cause faults in tenant 2
- Logical isolation but circuits share power distribution
- Voltage brownouts through adversarial power consumption
- Evaluate with combination of hardware measurements and on-chip sensors



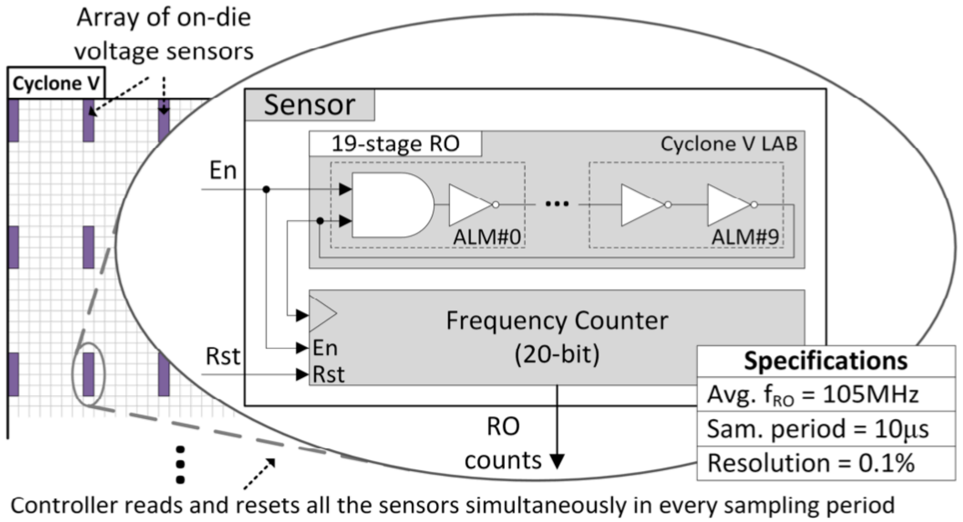
LTC3608 switching reg.

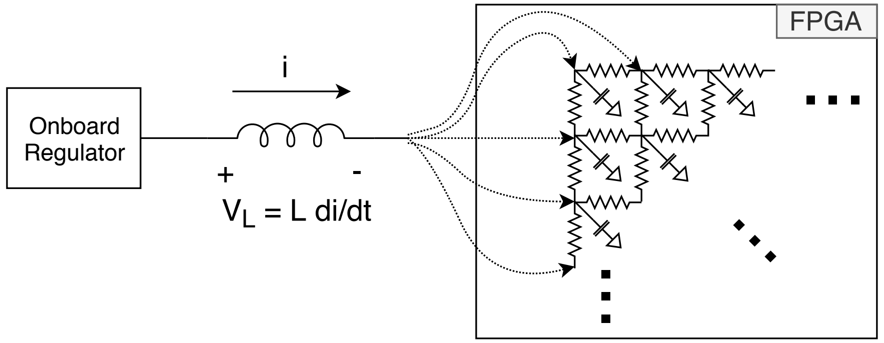


Cyclone V FPGA (28nm) on DE1-SoC board

Voltage Sensors and Calibration

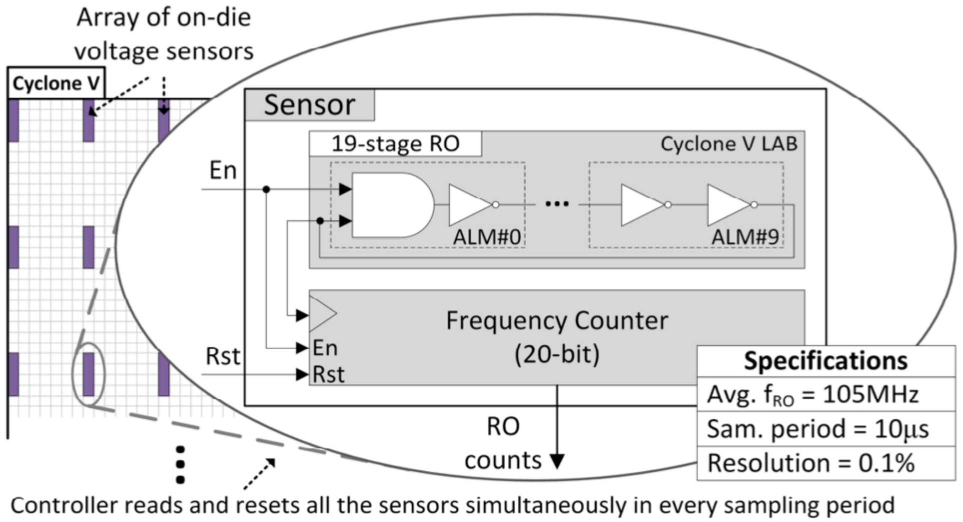
- Oscillator frequency as proxy for supply voltage
- Spatial resolution, low cost, no hard sensors

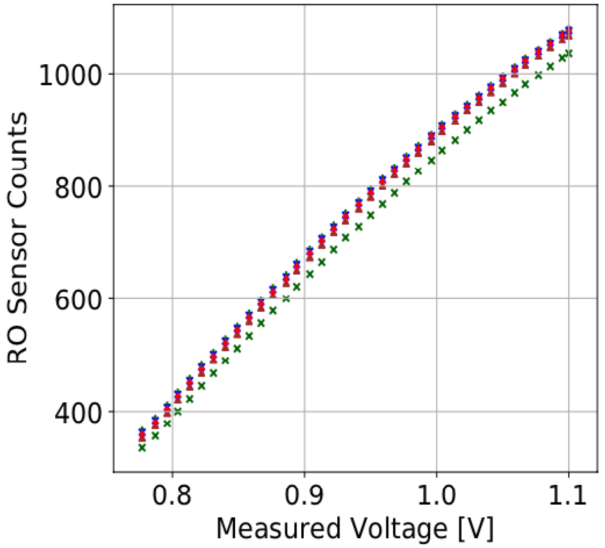


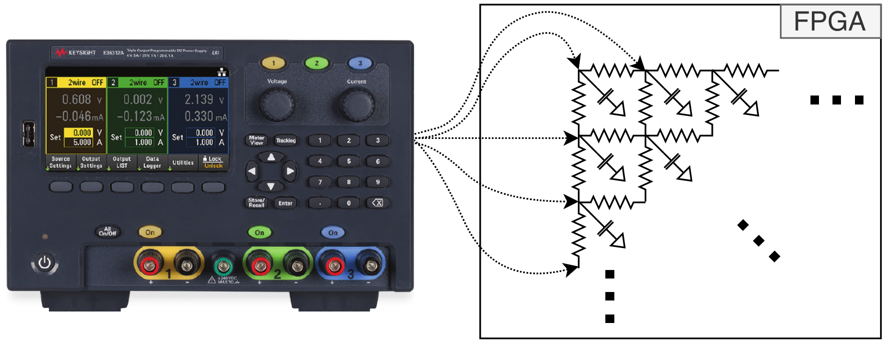


Voltage Sensors and Calibration

- Oscillator frequency as proxy for supply voltage
- Spatial resolution, low cost, no hard sensors

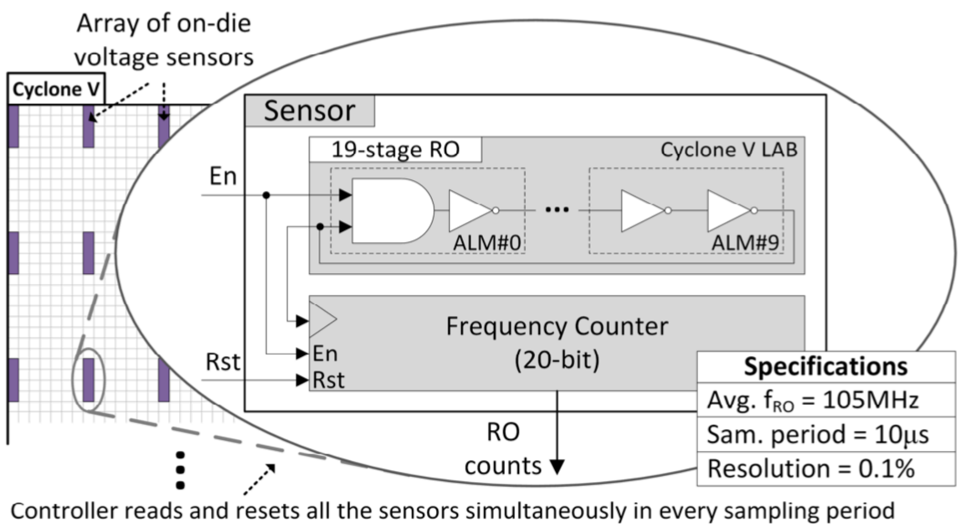


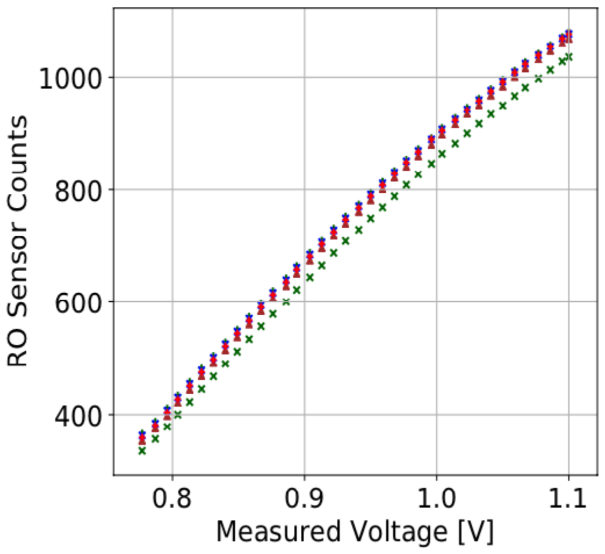


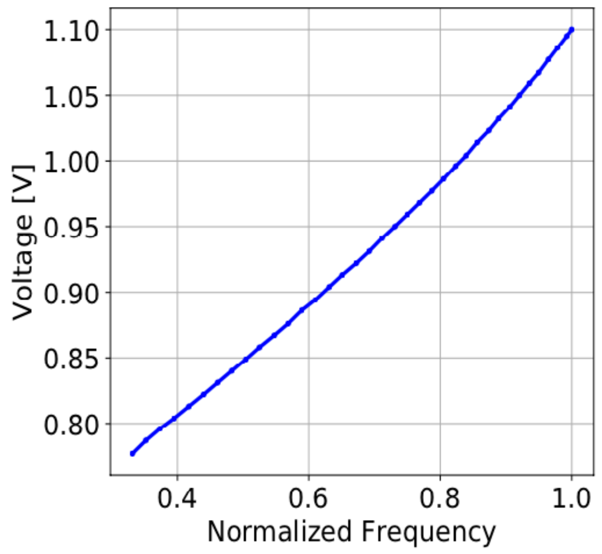


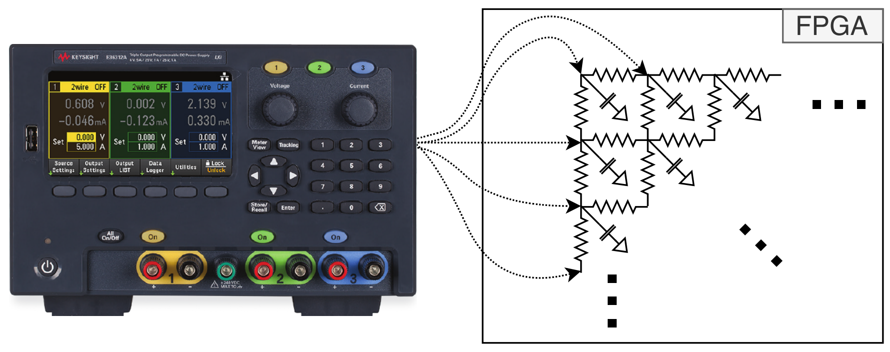
Voltage Sensors and Calibration

- Oscillator frequency as proxy for supply voltage
- Spatial resolution, low cost, no hard sensors









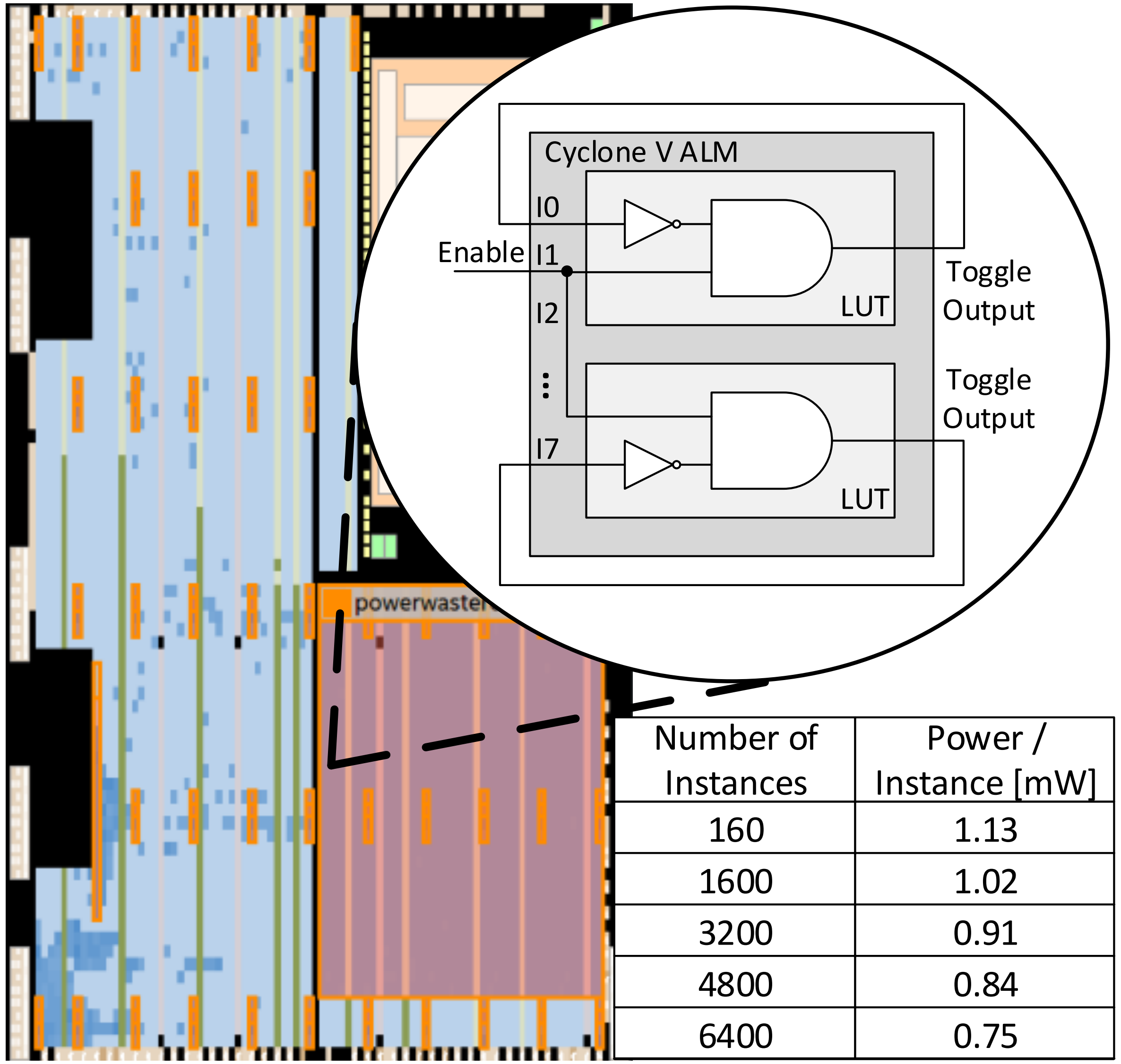
- 10µs period yields sub-mV resolution
- Benchtop supply for calibration only. Fault attacks use unmodified board

Power Waster Circuits

- Attacker trying to cause voltage faults should maximize power with density, speed, activity

$$P_{dyn} = C \times V_{DD}^2 \times f_{SW}$$

- 1-stage ROs are ideal power wasting circuits
  - Can pack 20 per LAB
  - Enable to switch on/off in synch
- AWS EC2 F1 doesn't allow ROs
  - Can use stealth ROs
  - Or comb. circuits designed to maximize glitching [1]

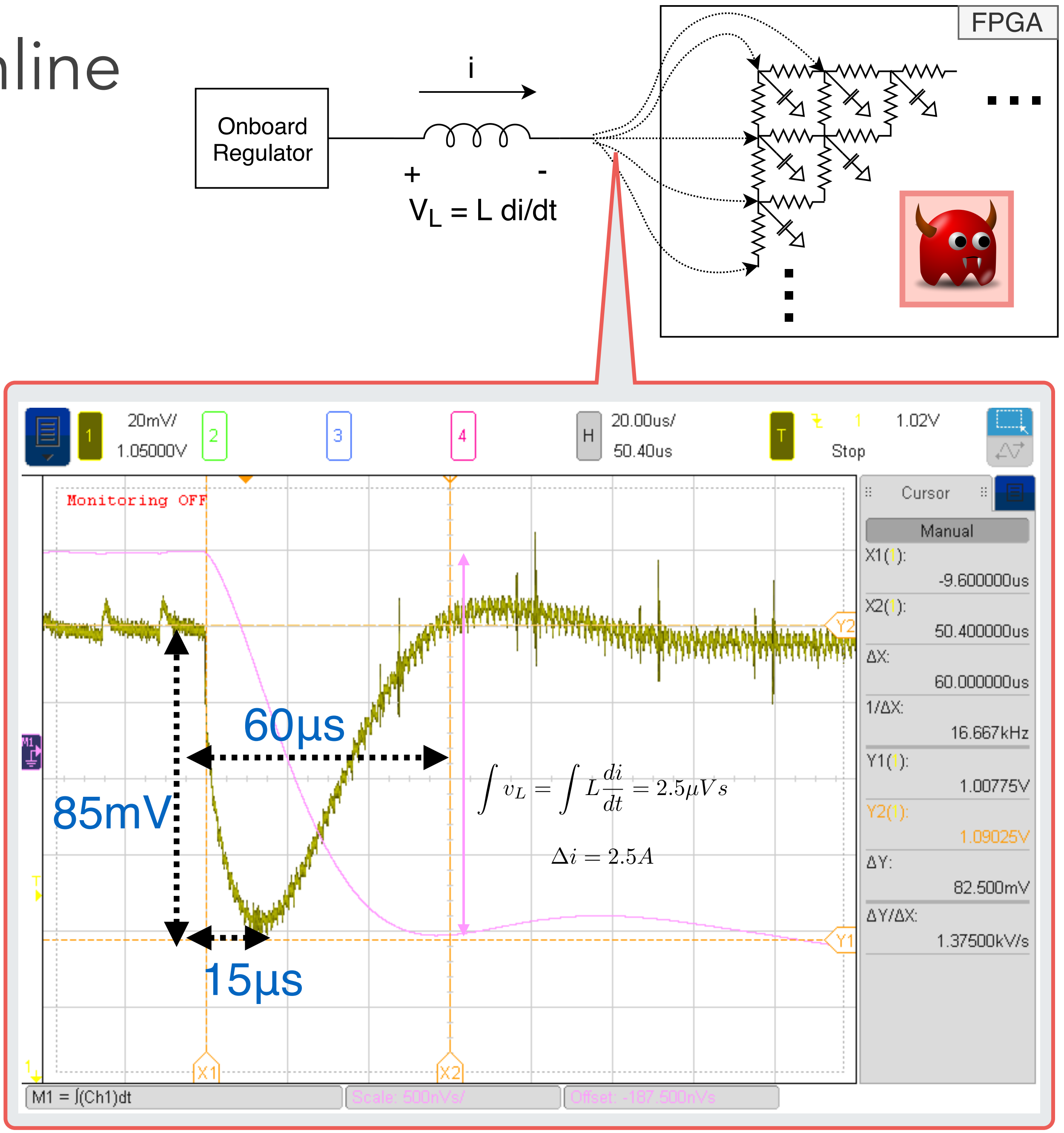


Number of Instances	Power / Instance [mW]
160	1.13
1600	1.02
3200	0.91
4800	0.84
6400	0.75

[1] Provelengios, Holcomb, Tessier, "Power wasting circuits for cloud FPGA attacks" FPL'20

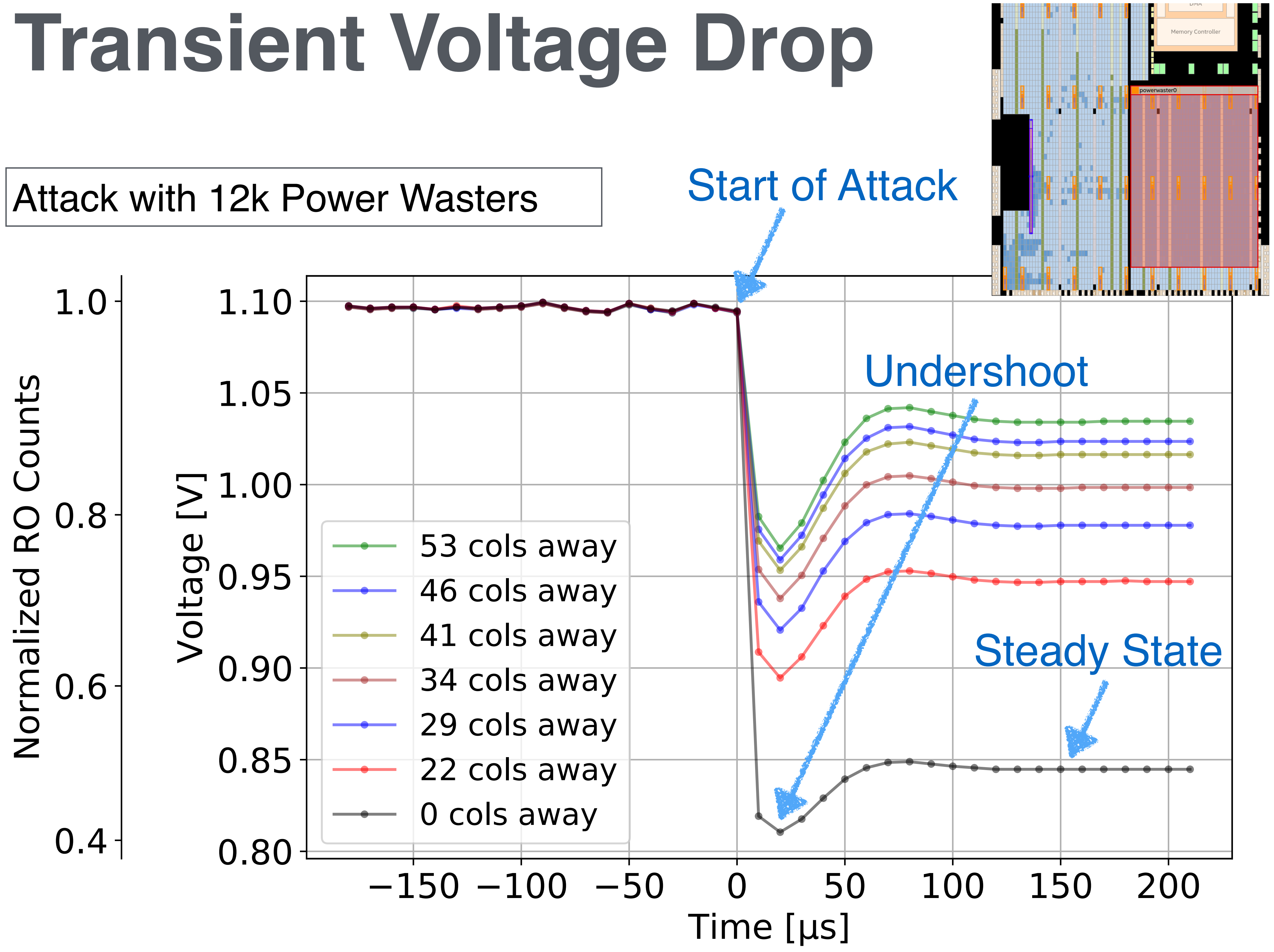
Voltage Undershoot from Inductor

- L di/dt drop on 1μH inline inductor of switching regulator from inrush current when power wasters turn on
  - +2.5A in 60μs
  - 85mV drop to entire chip
- Peaks at 15μs, recovers in 60μs

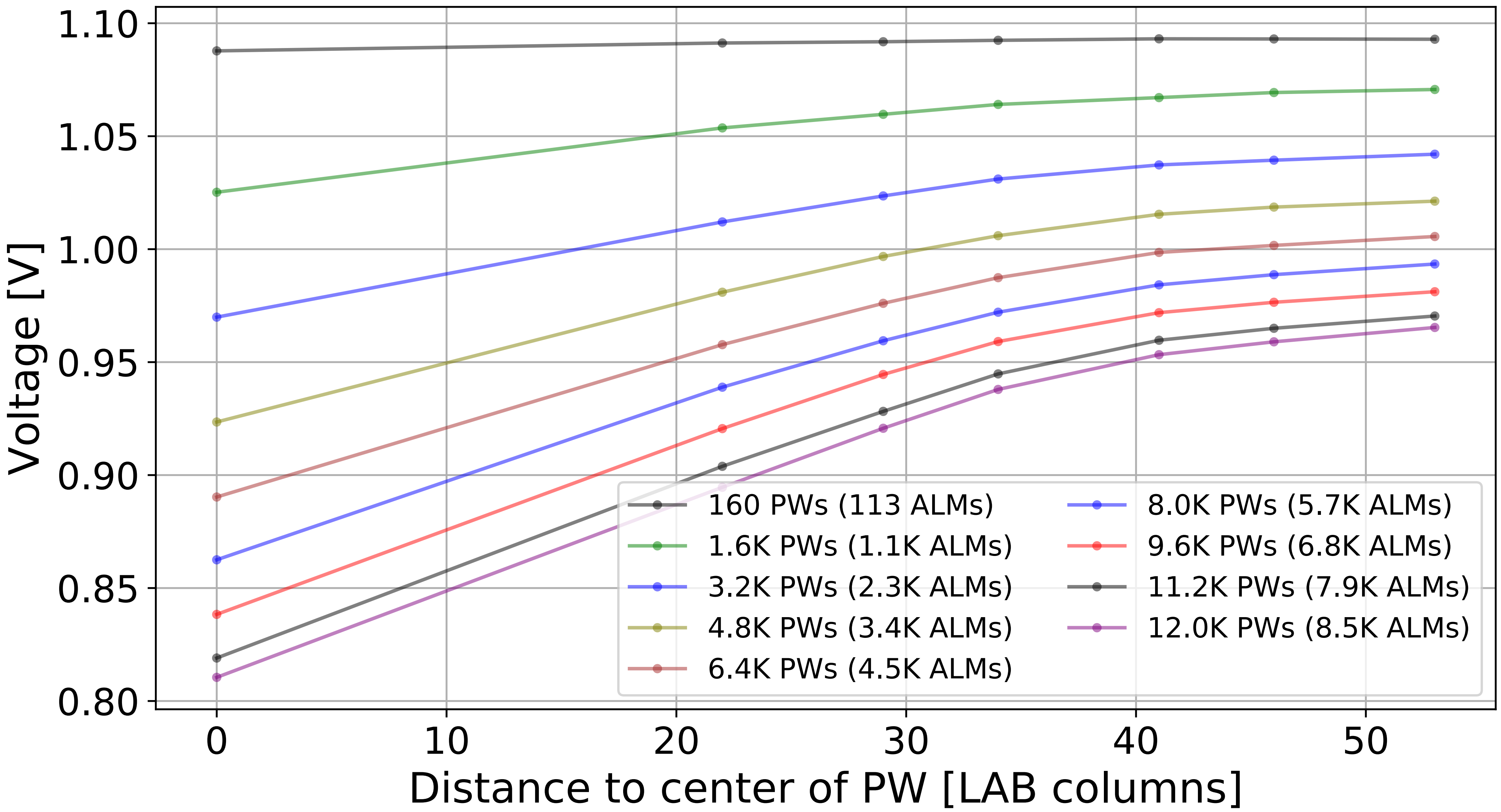


Transient Voltage Drop

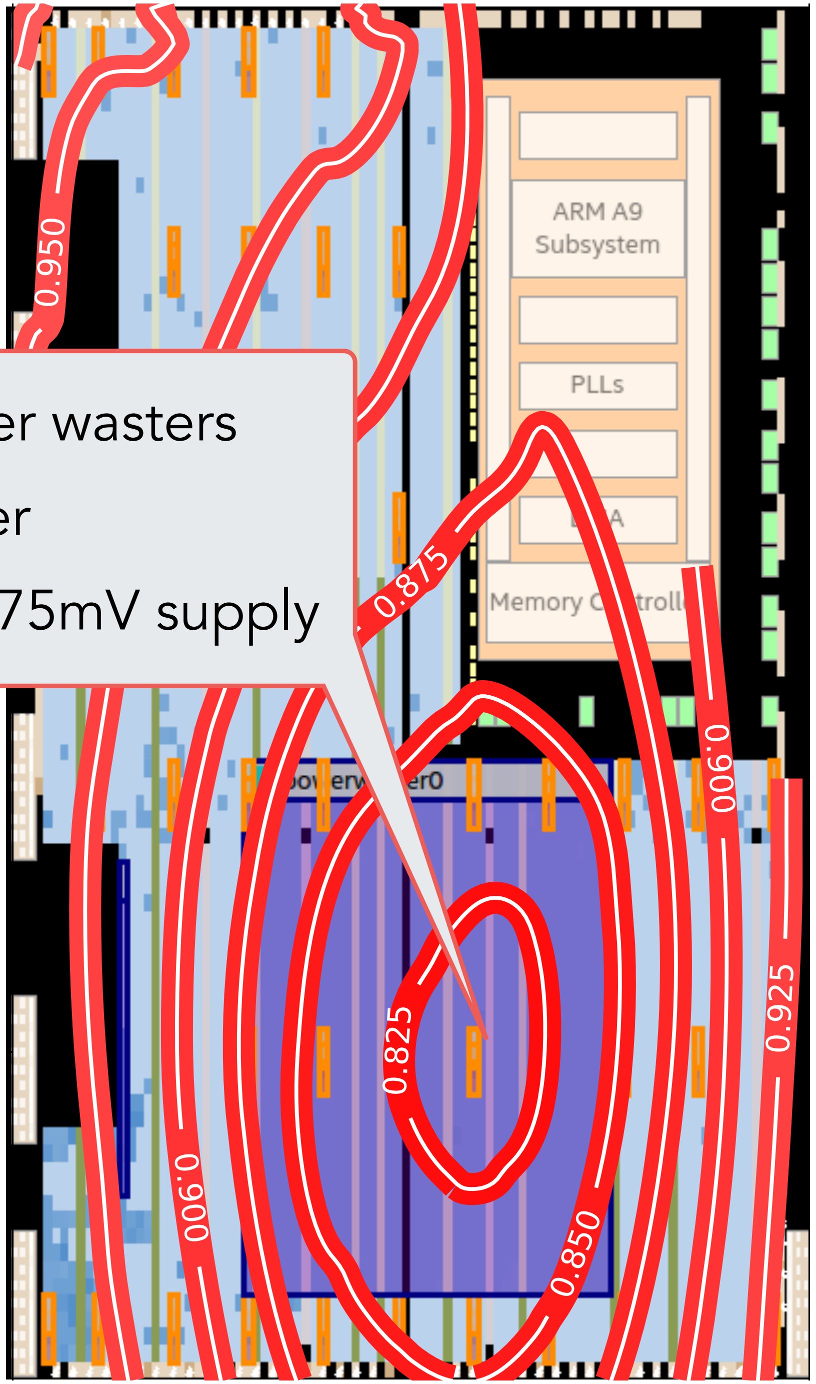
Attack with 12k Power Wasters



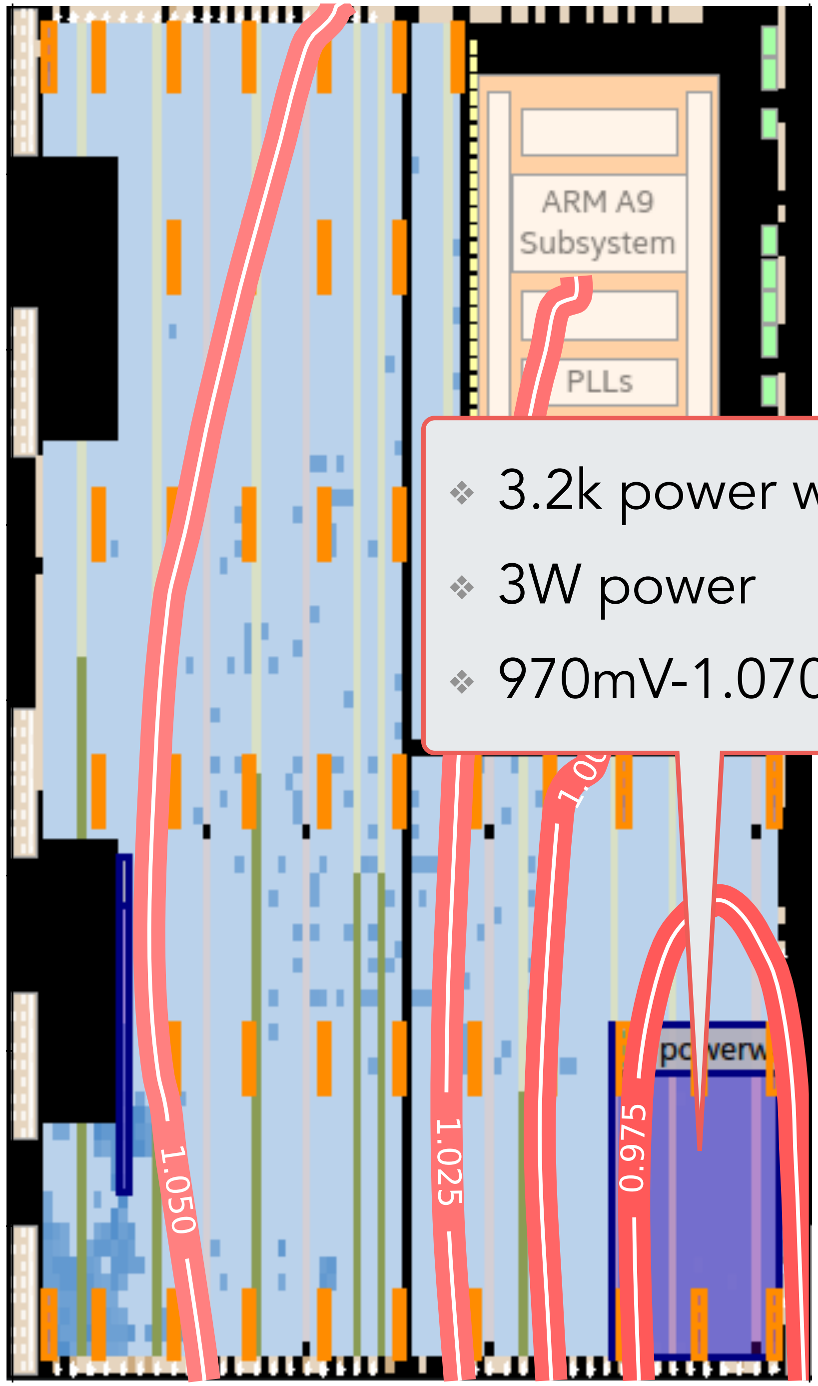
Steady-State Voltage vs Distance



Voltage Surface During Power Attack



- 12k power wasters
- 9W power
- 800mV-975mV supply

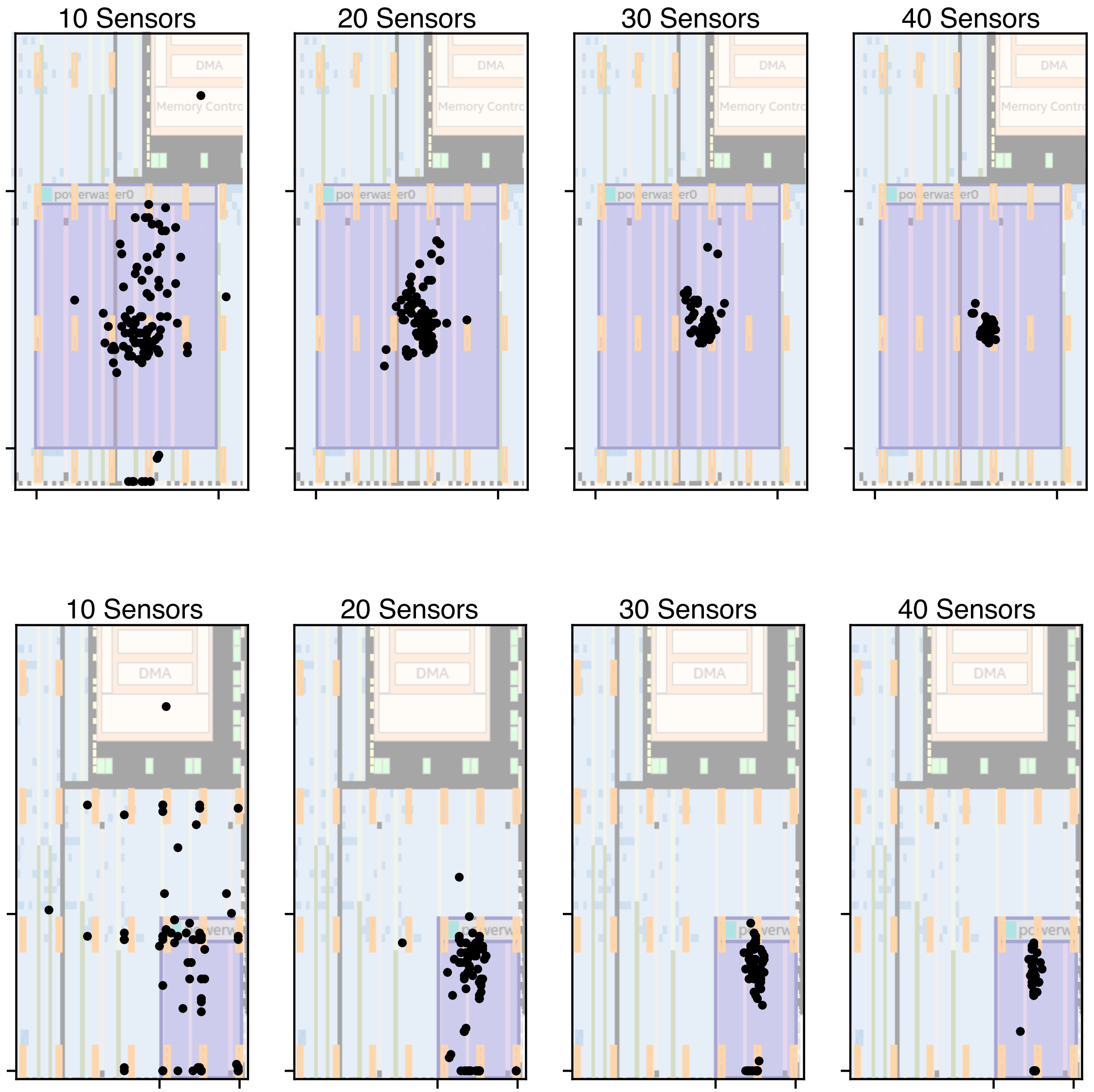


- 3.2k power wasters
- 3W power
- 970mV-1.070V supply

Cyclone V FPGA on DE1-SoC board

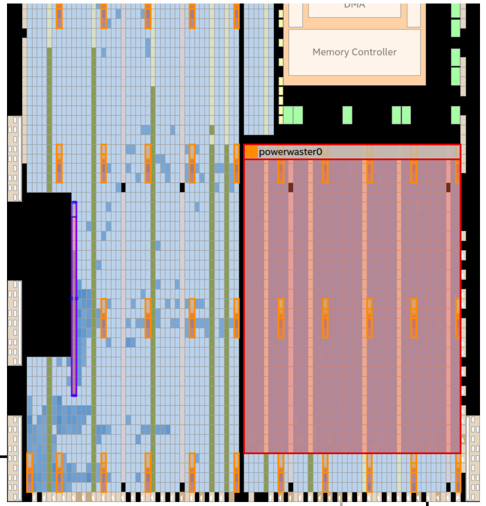
Detection and Mitigation

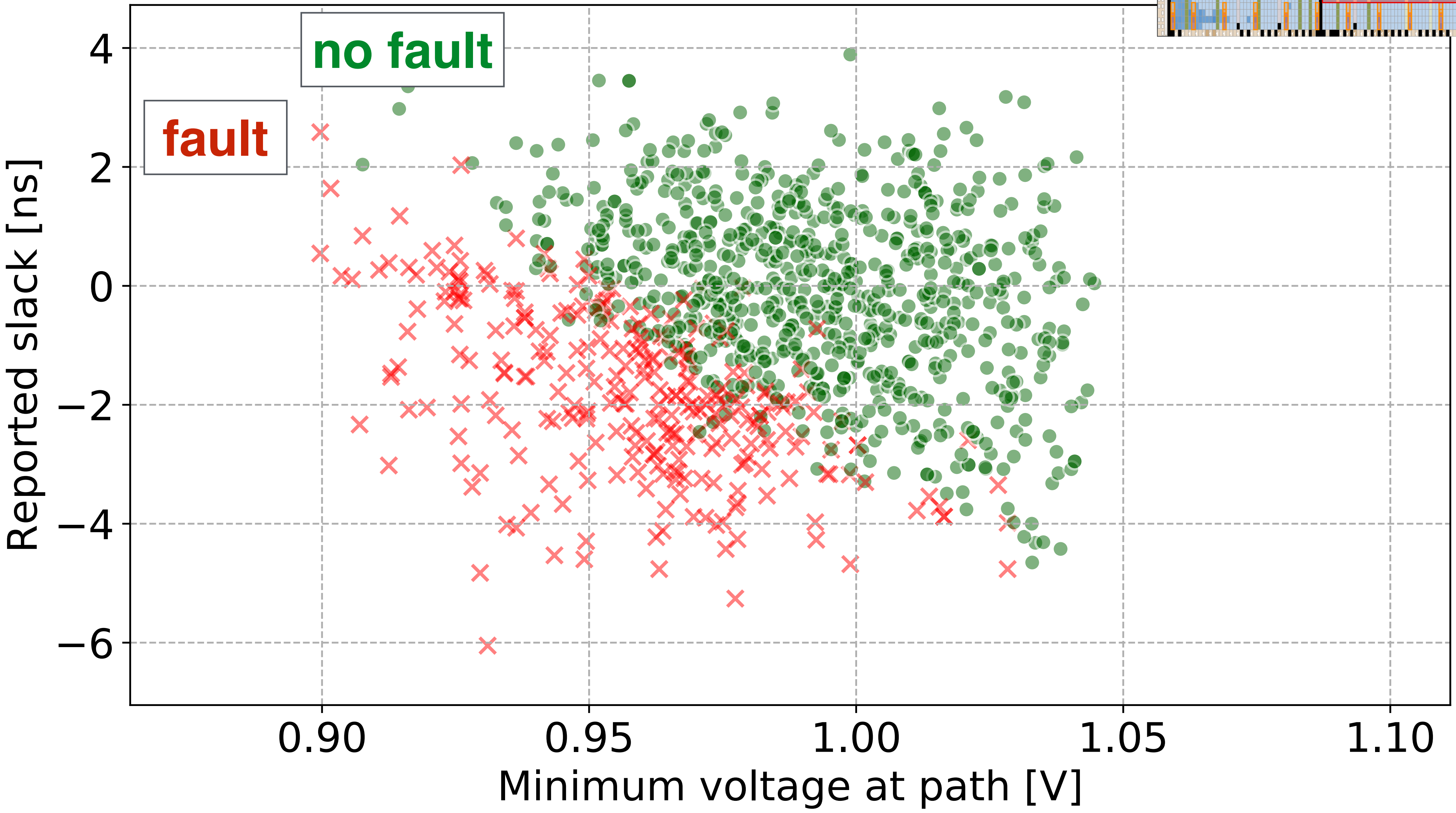
- How many of sensors are needed to locate the attack?
- Dot is predicted center of attack from voltage gradients using random sensor subset
- Voltage gradients give away attacker position, especially in larger attack
- Use fewer sensors to save cost
- Can detect attacks in real time before faults occur?



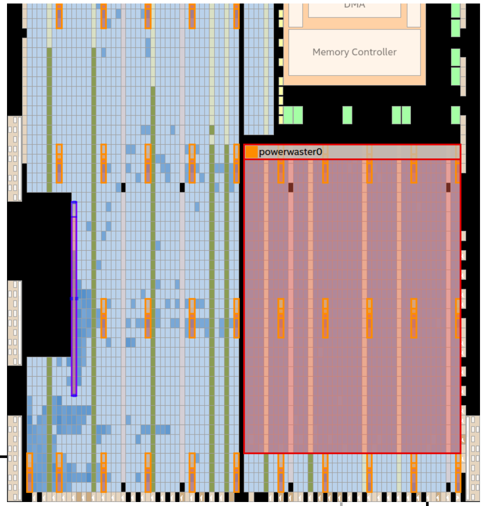
Num. RO sensors	ALMs (Avail.: 32,070)	Flip flops (Avail.: 128,280)
10	390 (1.2%)	200 (<1%)
20	780 (2.4%)	400 (<1%)
30	1,170 (3.6%)	600 (<1%)
40	1,560 (4.9%)	800 (<1%)
46	1,794 (5.6%)	920 (<1%)
Controller	430 (1.3%)	111 (<1%)

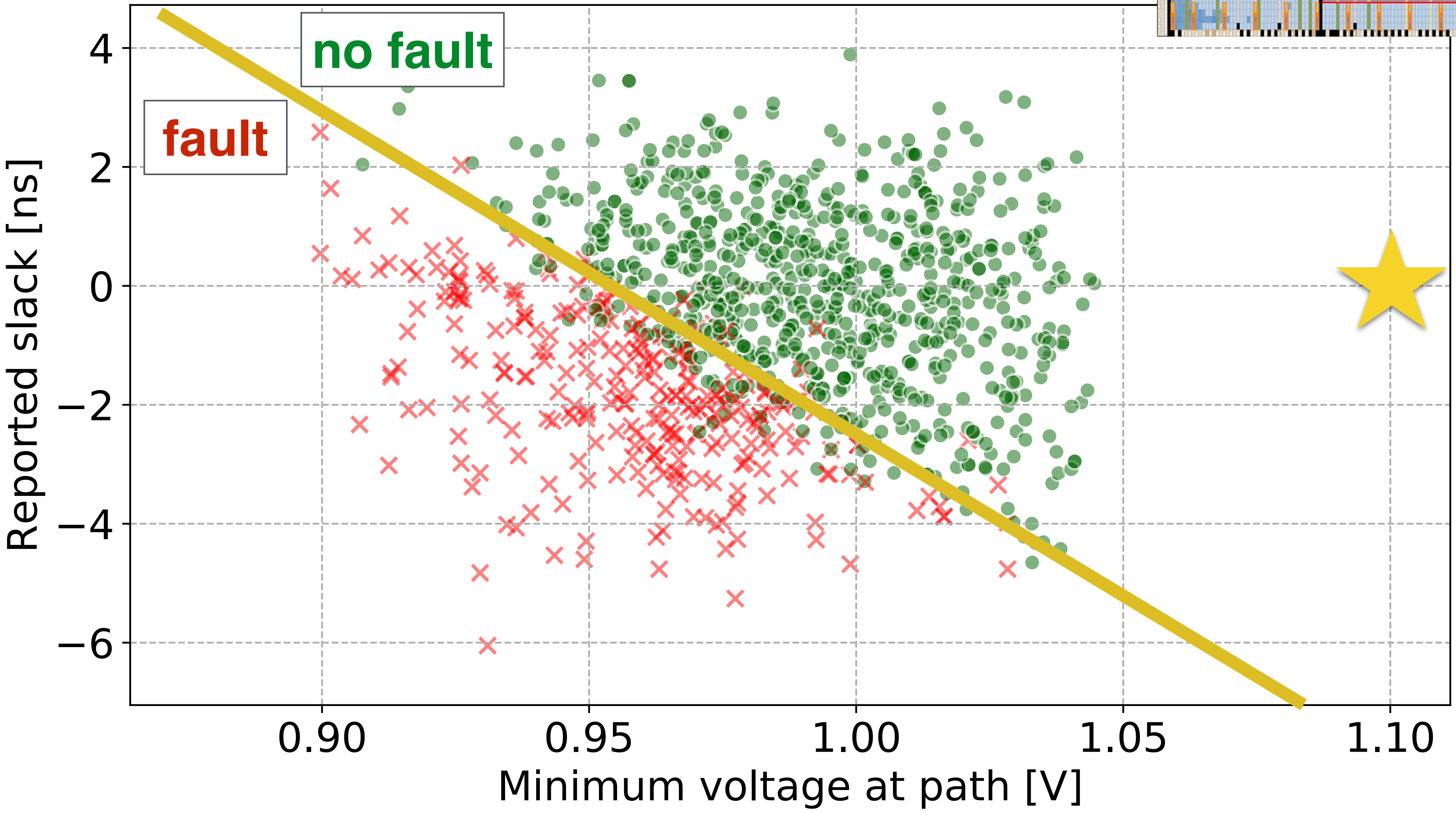
Delay Faults



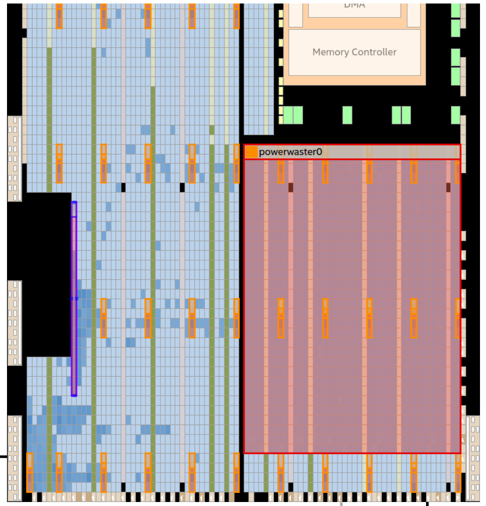


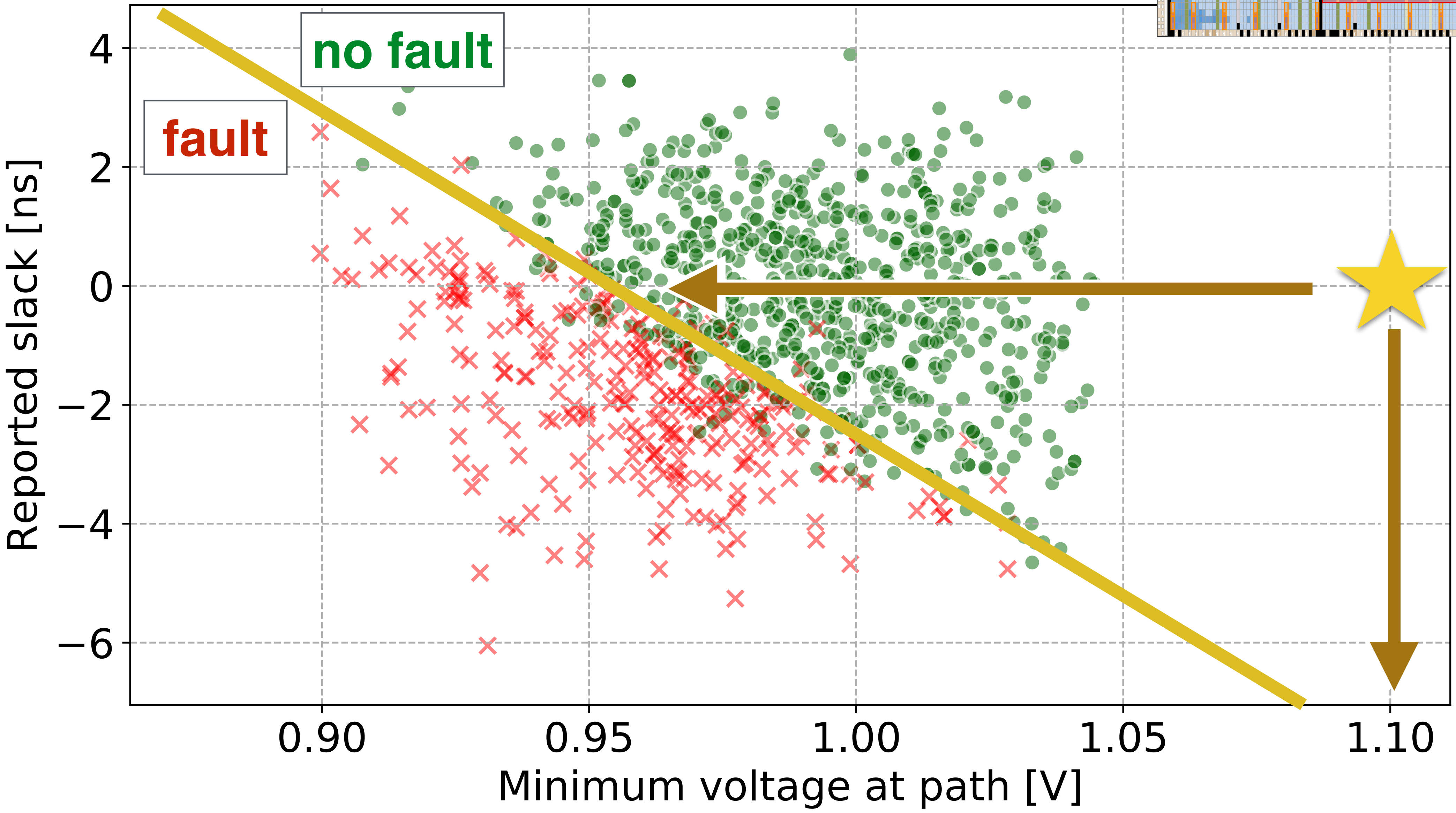
Delay Faults



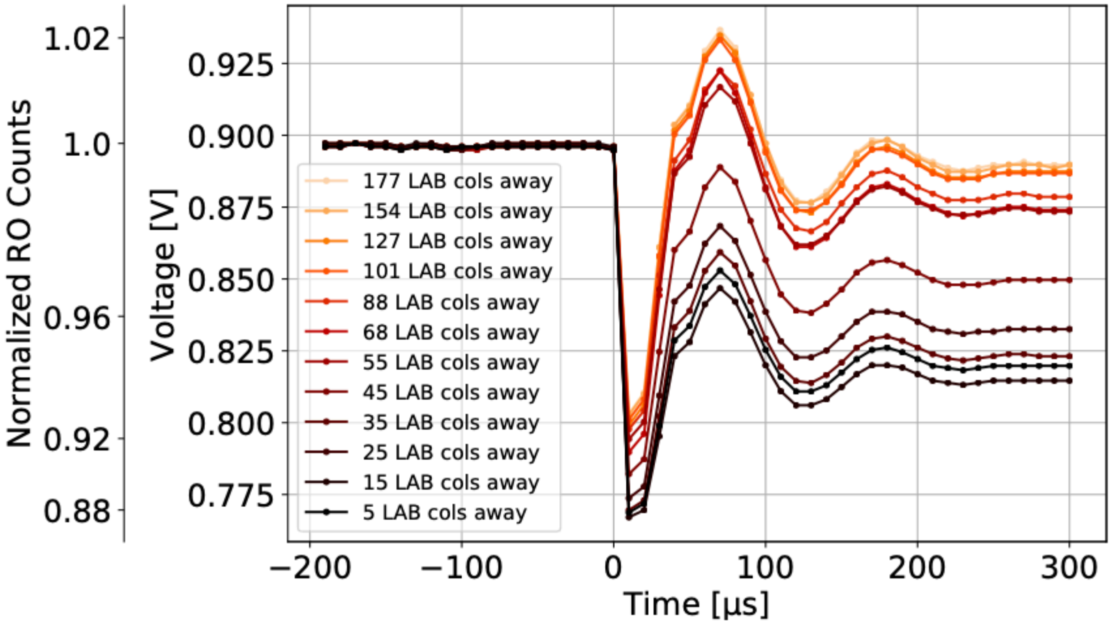


Delay Faults

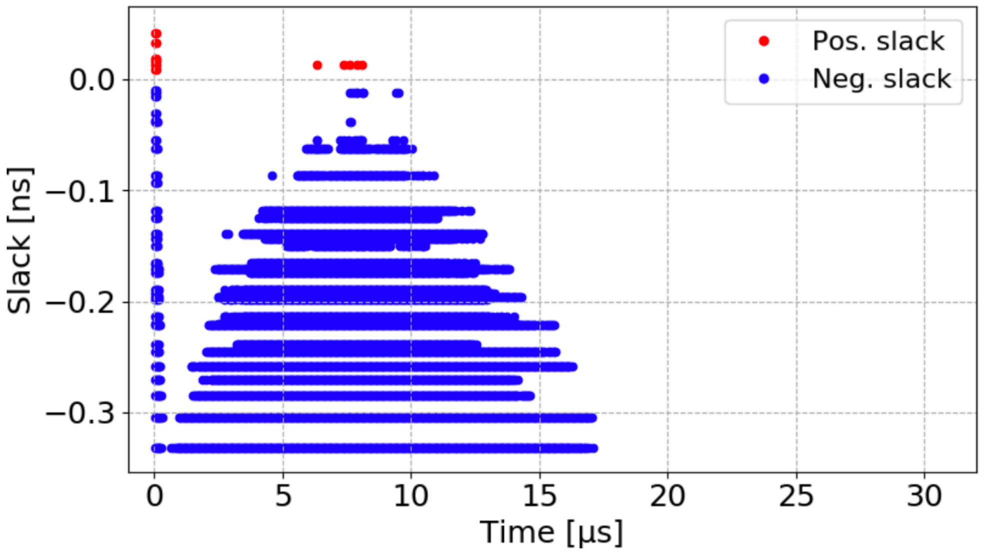


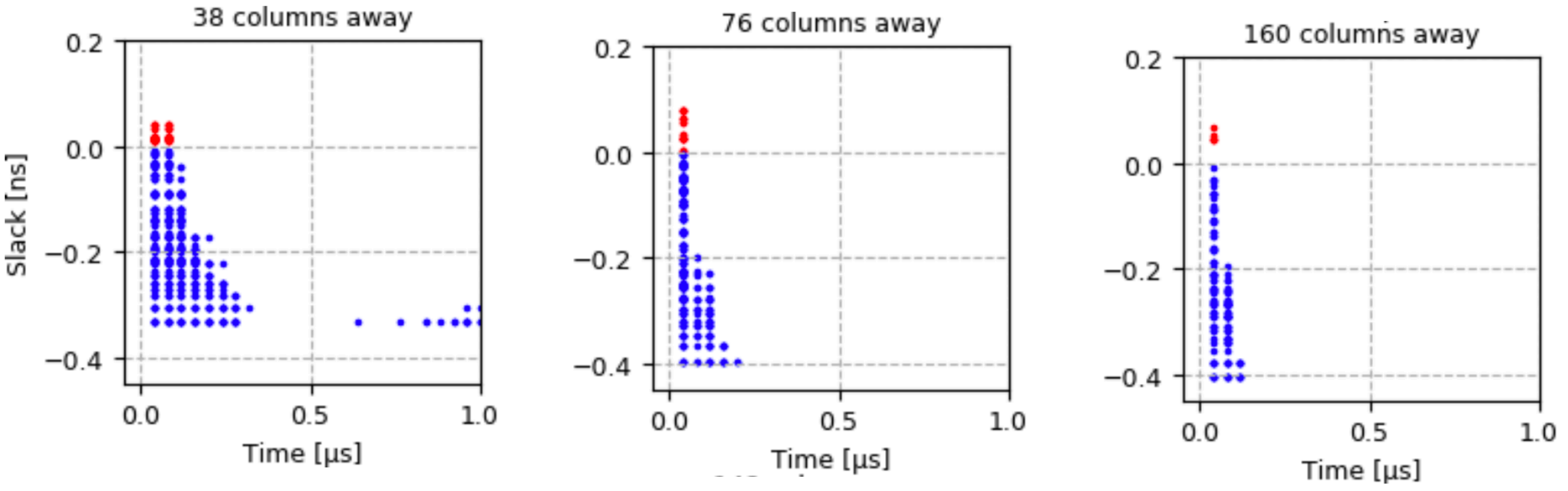


Faults in Arria 10 FPGAs



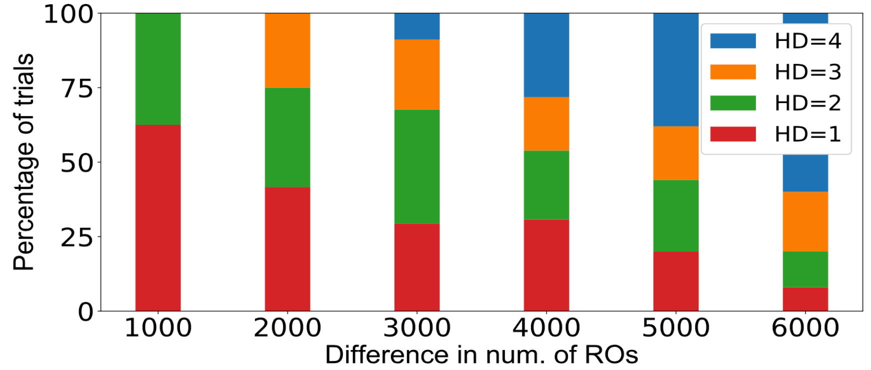
(b) Arria 10

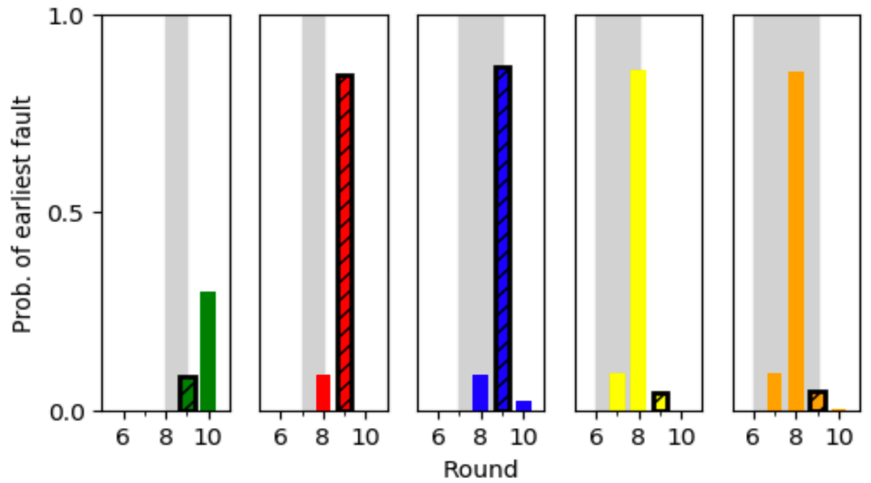


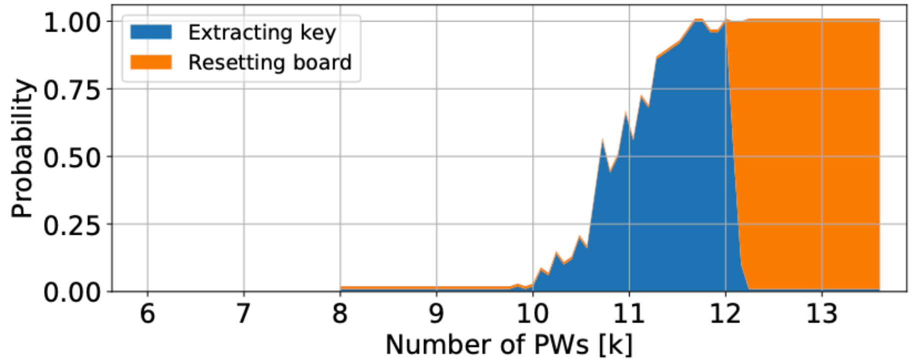


Adapting Voltage for Fault Attack

- Differential Fault Intensity Analysis [1]
  - Vary fault intensity by glitching clock in specific cycles [1]
  - DFIA by on-chip voltage attacks [2]
    - num of ROs
    - timing of enable/disable
- Attack RSA by faulting CRT [3]
  - Previously by supply tuning [4]
  - Now by voltage attack [5]







[1] Ghalaty, Yuce, Taha, and Schaumont, "Differential fault intensity analysis", FDTC'14

[2] Li, Tessier and Holcomb, "Precise Fault Injection to Enable DFIA for Attacking AES in Remote FPGAs", FCCM'22

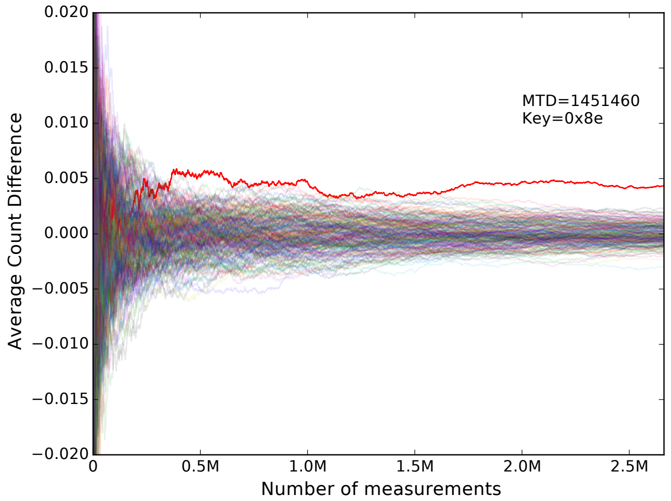
[3] Lenstra, "Memo on RSA signature generation in the presence of faults," 1996.

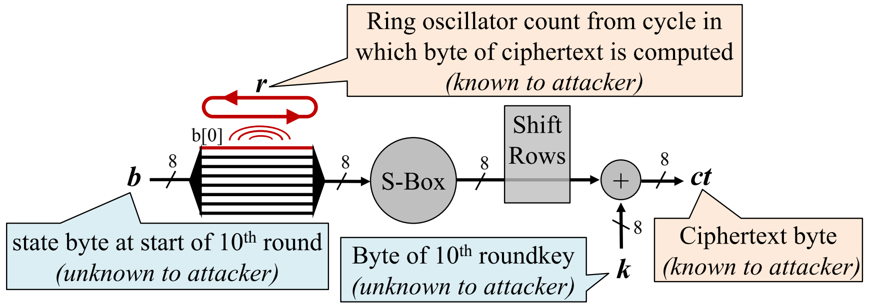
[4] Pellegrini, Bertacco, and Austin, "Fault-based attack of RSA authentication," DATE'10

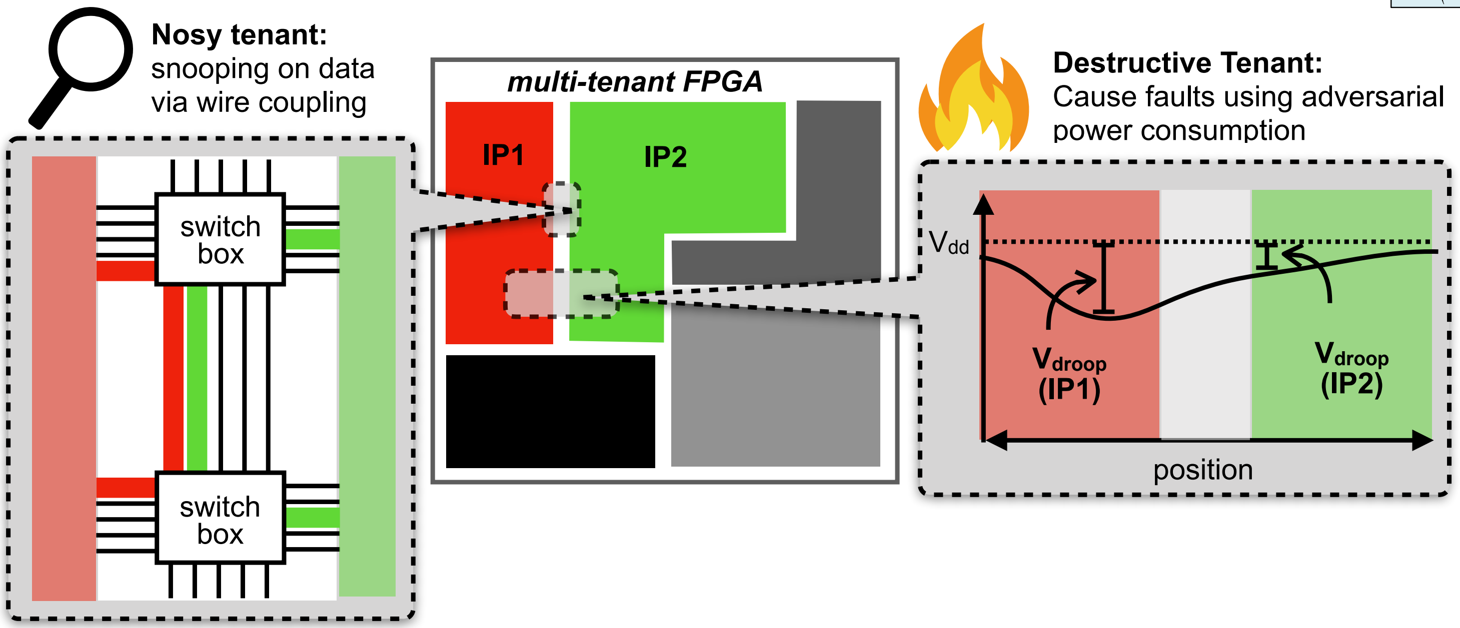
[5] Provelengios, Holcomb, and Tessier, "Power Distribution Attacks in Multi-Tenant FPGAs " TVLSI'20

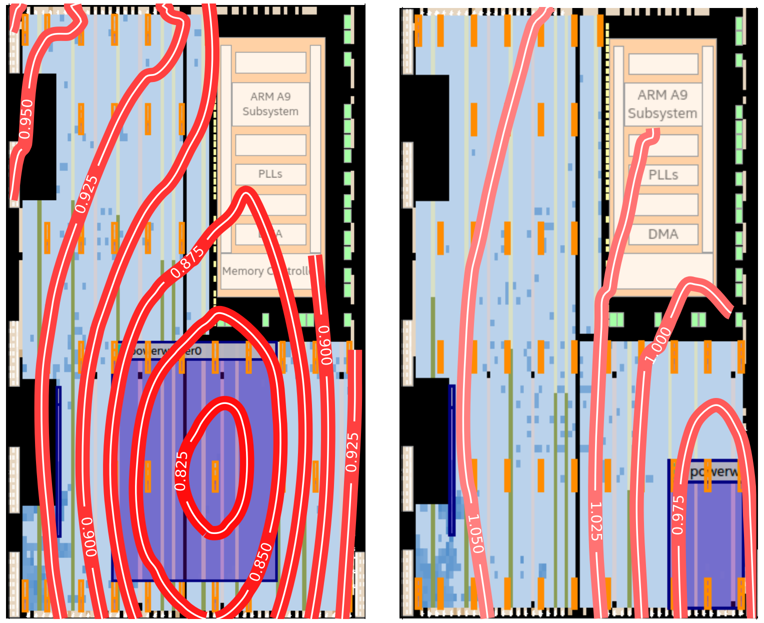
Multi-Tenant FPGA Security Recap

- Nosy Tenant: wire leakage
  [Ramesh *et al.* FCCM'18] [Provelengios *et al.* FPGA'19]
  - AES key exfiltration without physical access
  - Characterization and inferring channel layout
- Destructive Tenant: Faults using power wasters
  [Provelengios *et al.* FPL'19]
  - Shaping voltage drops and reconstructing voltage from sensors









- **Part 1: Security in Multi-Tenant FPGAs**
  - Remote side channel attacks using wire coupling
  - Fault attacks by adversarial power consumption

- **Part 2: Inter-chiplet delay PUF**
  - Implemented on Xilinx FPGAs locally and AWS

Know Time to Die – Integrity Checking for Zero Trust Chiplet-based Systems Using Between-Die Delay PUFs

Aleksa Deric, Daniel E. Holcomb
University of Massachusetts Amherst

CHES'22

University *of* Massachusetts Amherst

MITRE

Introduction

Industry trends toward chiplets as a replacement for monolithic fabrication

The modularity of chiplets brings new and interesting security threats

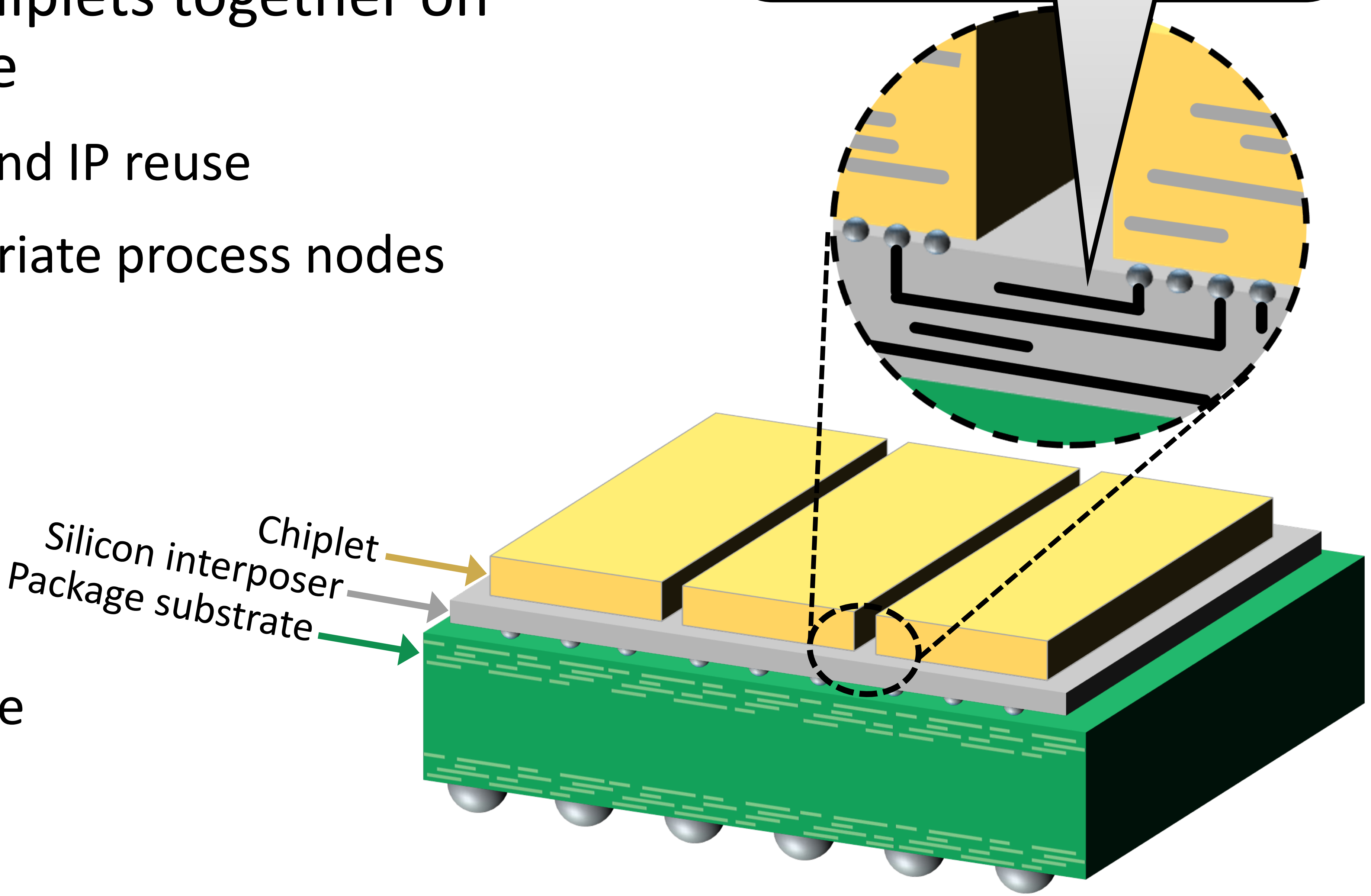
This work: inter-die delay PUF as a security primitive for chiplets

Chiplets - Overview

- Each chiplet is a separately-produced silicon die
- SoC created by packaging chiplets together on a silicon interposer or bridge
  - Heterogeneous integration and IP reuse
  - Able to leverage cost-appropriate process nodes
  - Increased yield

Interposer wires:

- 500-1k wires/mm shoreline
- 25-50 µm microbump pitch
- 0.5 pJ/bit
- 1 Gbps/wire



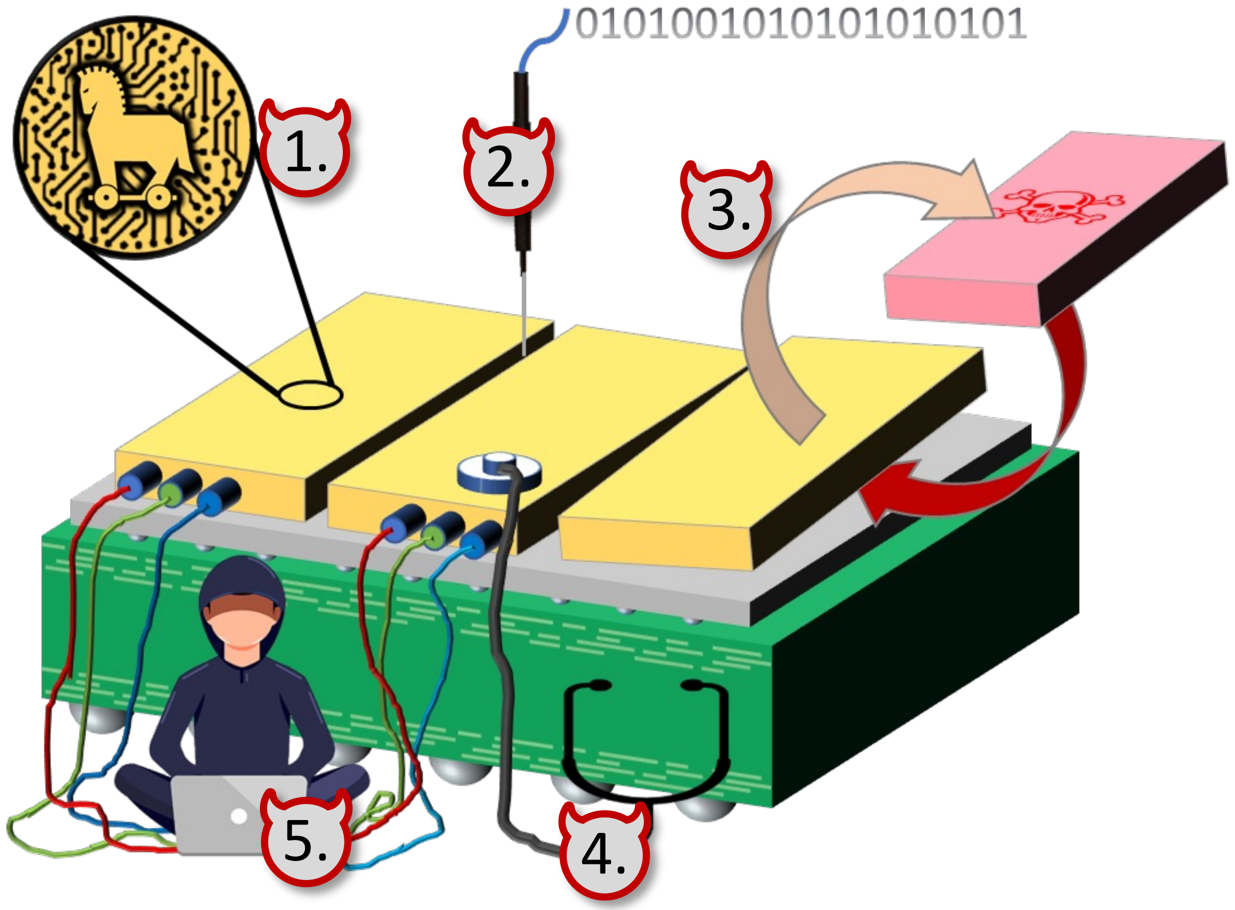
- Recent examples
  - AMD Ryzen
  - Intel Meteor Lake, Arrow Lake
  - Xilinx Virtex Ultrascale+

Chiplets – Motivation & Related Work

- Different threats possible with chiplets vs monolithic fabrication
  - Which are critical and how to defend?

1. Trojans in co-packaged chiplets
2. Probing exposed interposer wires
3. Die-swapping
4. Side-channels from within package
5. Man-in-the-middle

- Zero trust: chiplets cannot blindly assume they are operating in a friendly environment
  - Root of trust needed
  - Using cryptography and PUFs [1]
  - Trusted security-enforcing interposer with active traffic policing [2]
  - Secure-by-construction interposer Networks-on-Chip with message checking [3]



- This work: inter-chiplet delay fingerprints through interposer for physical security

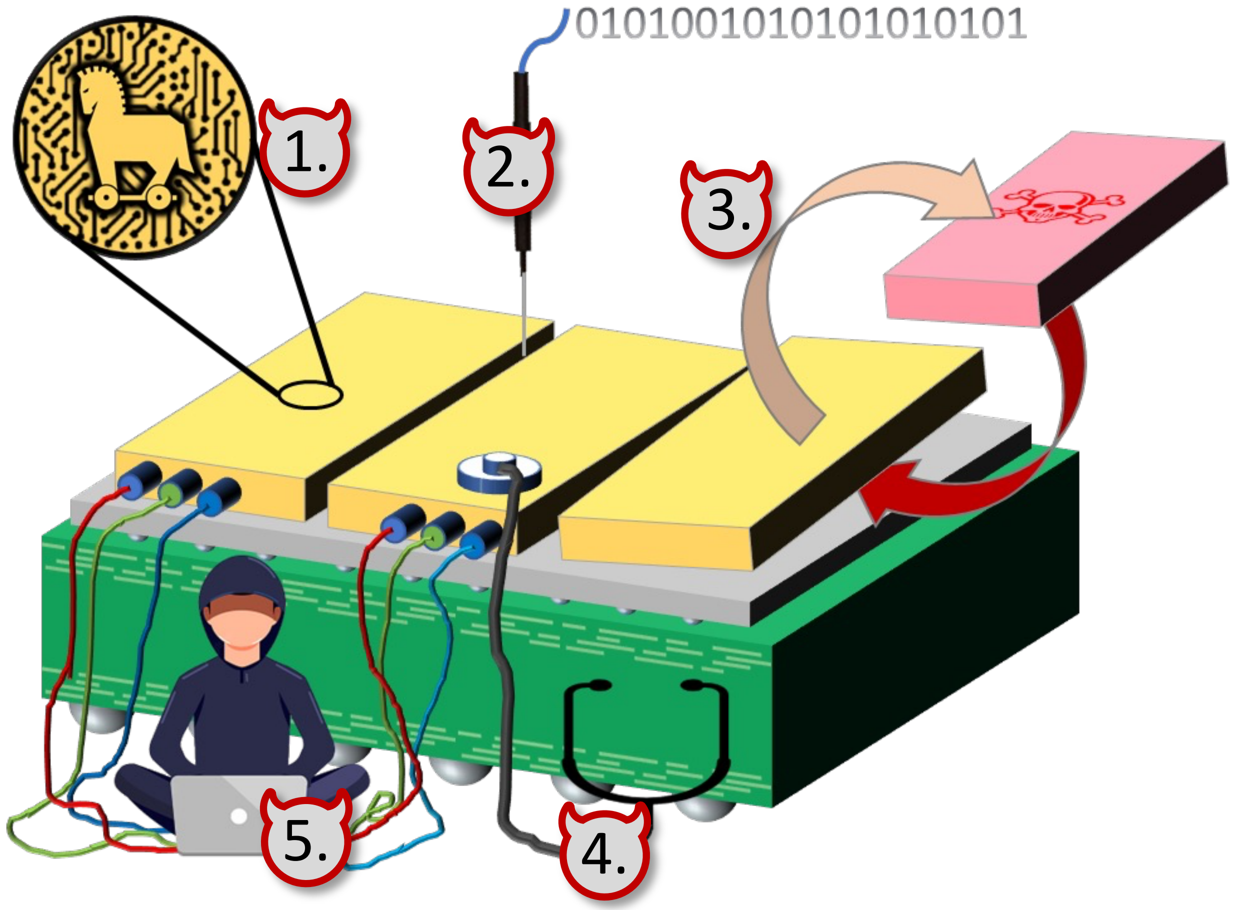
[1] CEVA. Fortrix: Self-contained IP platform for Root-of-Trust and cybersecurity in chiplets and SoCs. Product note. 2022
[2] Nabeel et al. 2.5d root of trust: Secure system-level integration of untrusted chiplets. IEEE T-Comp, 2020
[3] Chacon et al. Coherence attacks and countermeasures in interposer-based systems, 2021

Chiplets – Motivation & Related Work

- Different threats possible with chiplets vs monolithic fabrication
  - Which are critical and how to defend?

1. Trojans in co-packaged chiplets
2. Probing exposed interposer wires
3. Die-swapping
4. Side-channels from within package
5. Man-in-the-middle

- Zero trust: chiplets cannot blindly assume they are operating in a friendly environment
  - Root of trust needed
  - Using cryptography and PUFs [1]
  - Trusted security-enforcing interposer with active traffic policing [2]
  - Secure-by-construction interposer Networks-on-Chip with message checking [3]



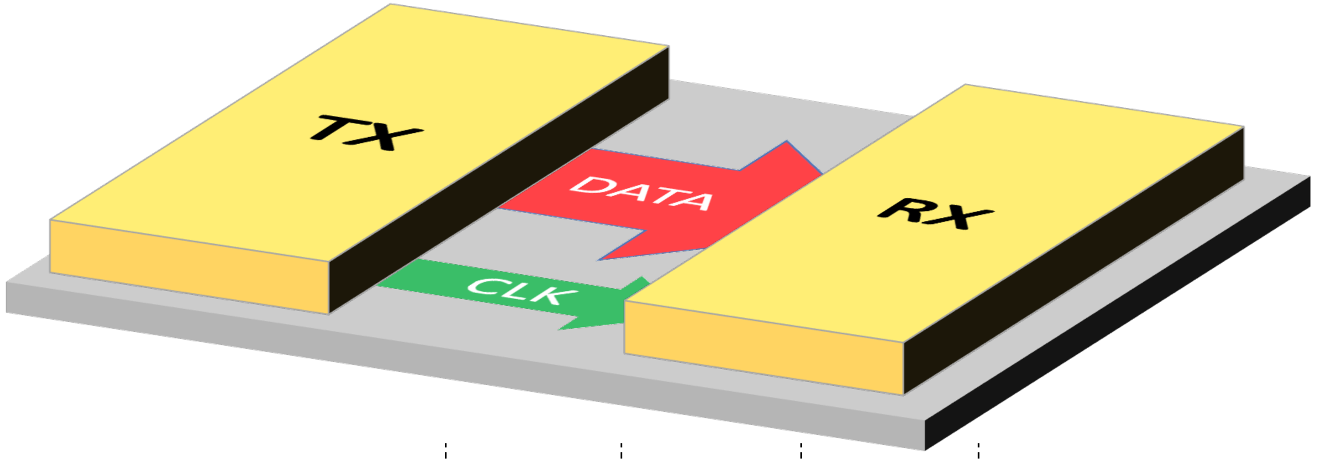
- This work: inter-chiplet delay fingerprints through interposer for physical security

[1] CEVA. Fortrix: Self-contained IP platform for Root-of-Trust and cybersecurity in chiplets and SoCs. Product note. 2022
[2] Nabeel et al. 2.5d root of trust: Secure system-level integration of untrusted chiplets. IEEE T-Comp, 2020
[3] Chacon et al. Coherence attacks and countermeasures in interposer-based systems, 2021

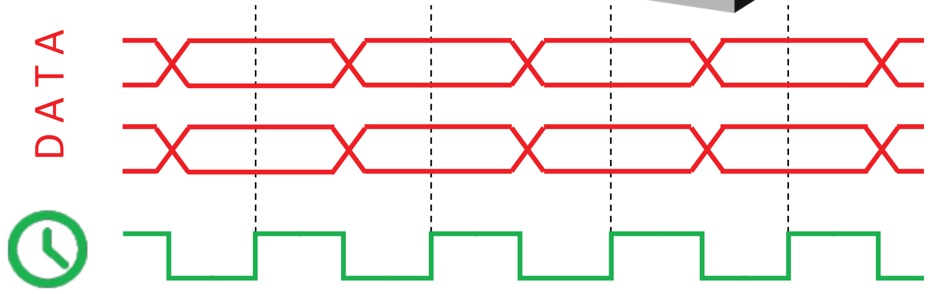
Overview

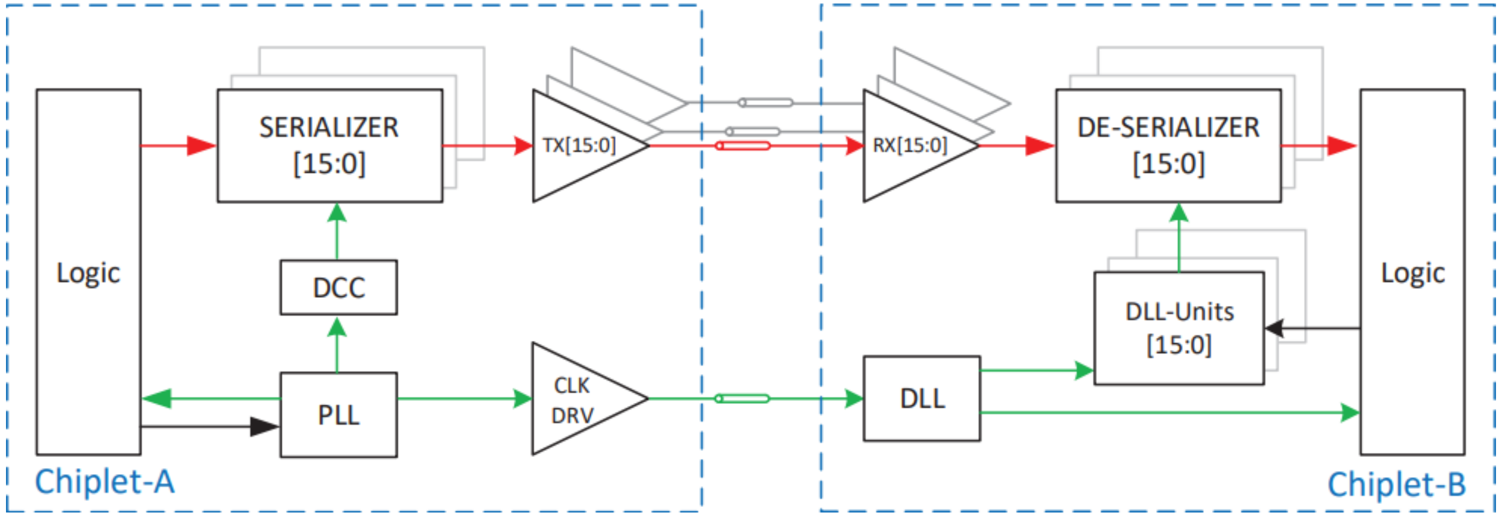

Measuring Delay
Design & Implementation
Statistics
Characterization Experiments
Summary

Communication Between Chiplets



- Typically source synchronous clocking
  - Data and clock forwarded from TX
  - Wires crossing through interposer
  - Registered I/O
  - Tunable delay on RX deskews sampling clock
- Emerging standards
  - Intel AIB [1]
  - TSMC LIPINCON [2]
  - UCIe [3]
  - Bunch-of-Wires [4]





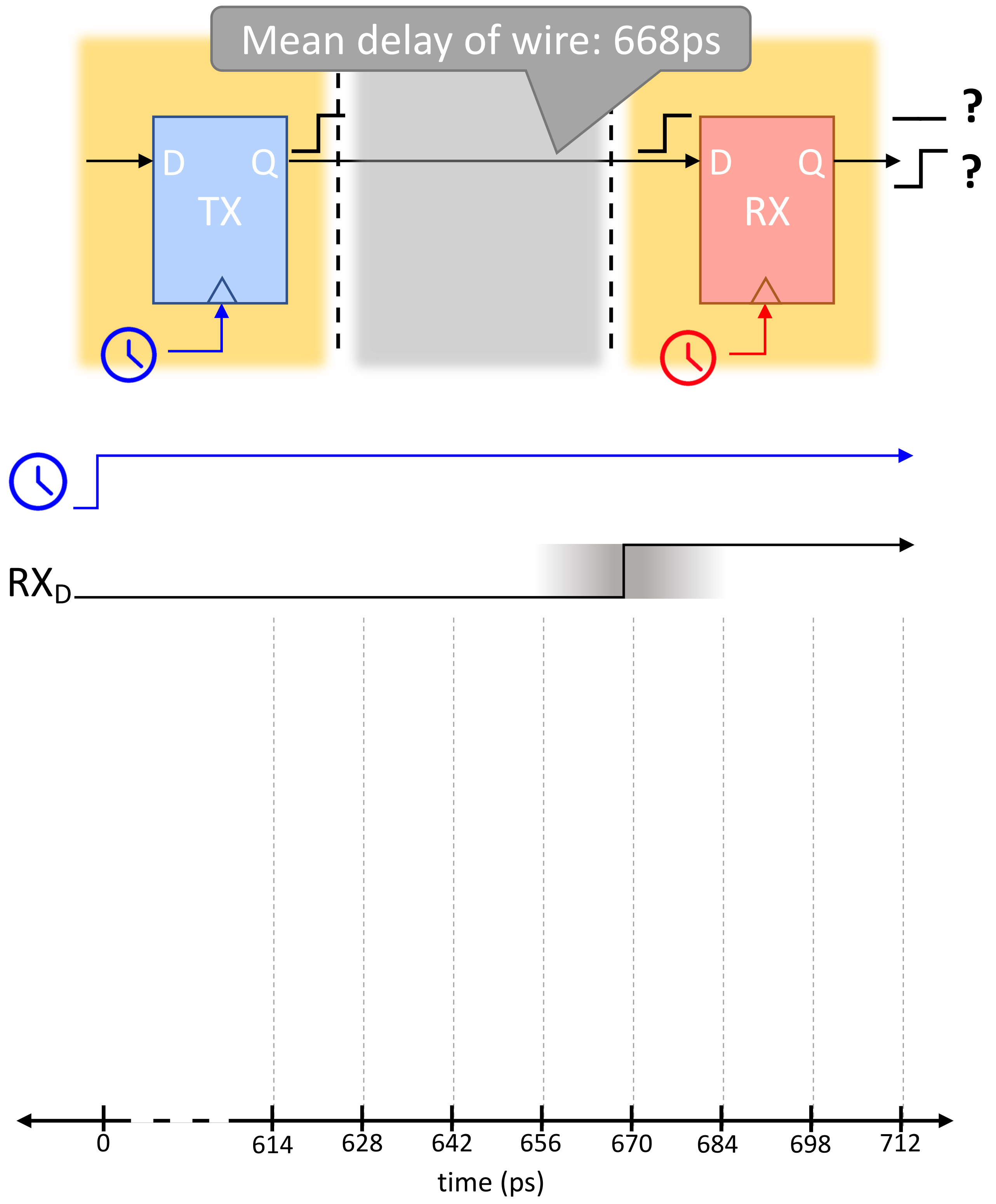
[Farjadrad et al.]

[1] David Kehlet. Accelerating innovation through a standard chiplet interface: The advanced interface bus (AIB). Intel White Paper, 2017
[2] Lin et al. A 7nm 4GHz Armcore-based CoWoS chiplet design for high performance computing. In 2019 Symp on VLSI Circuits, 2019
[3] D. Das Sharma, “Universal Chiplet Interconnect express (UCIe)® : Building an open chiplet ecosystem”, UCIe Consortium White paper, 2022
[4] R. Farjadrad et al., "A Bunch-of-Wires (BoW) Interface for Interchiplet Communication," IEEE Micro, 2020

Measuring Delay

- Use phase compensation to measure propagation delay of signal from neighboring chiplet
  - Transmit repeatedly
  - Sweep receiver phase
  - Find phase with 50% failure

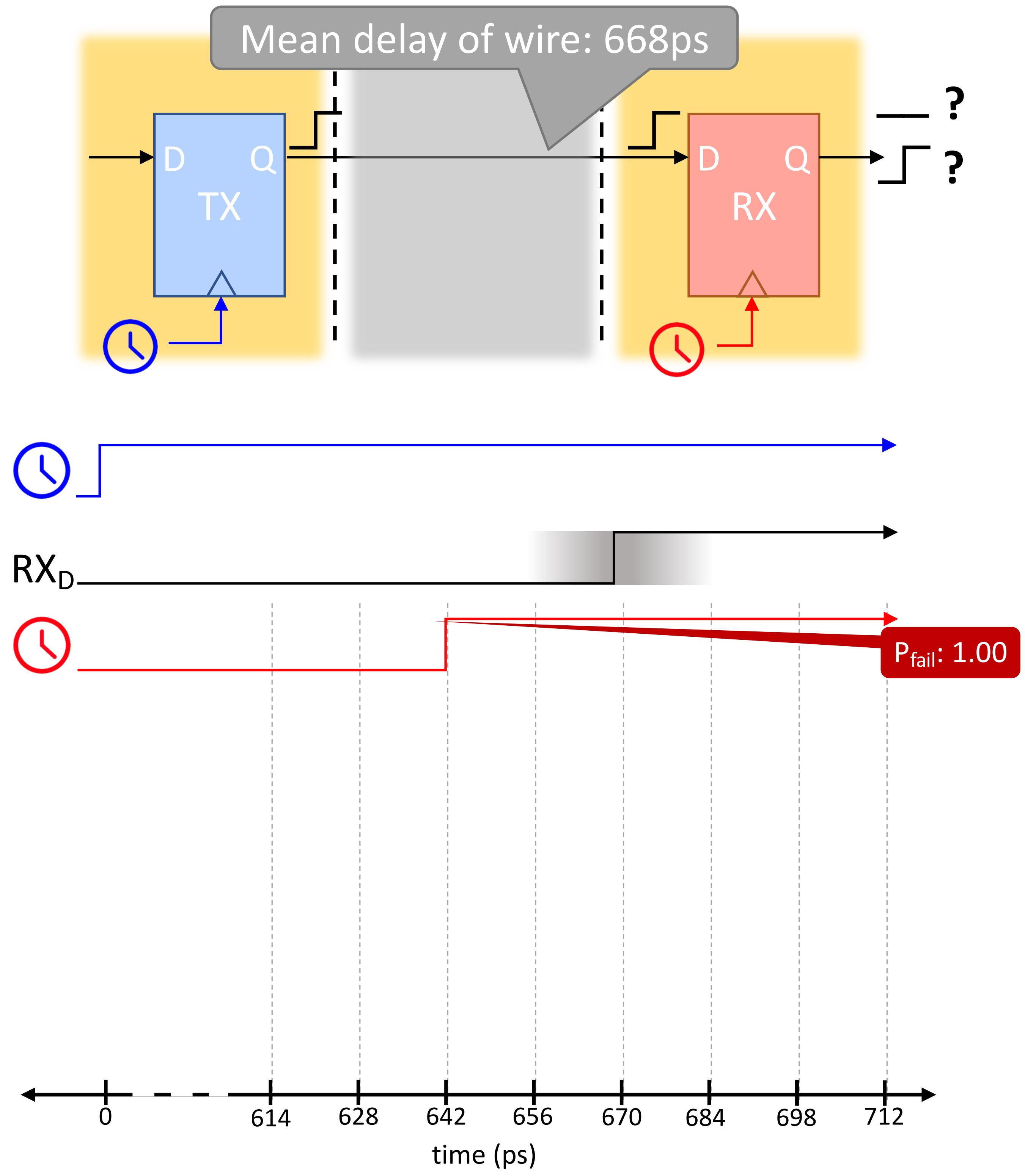
- Delay defined as the skew between TX and RX clock that causes rising transition to be received as 0 and 1 with equal probability



Measuring Delay

- Use phase compensation to measure propagation delay of signal from neighboring chiplet
  - Transmit repeatedly
  - Sweep receiver phase
  - Find phase with 50% failure

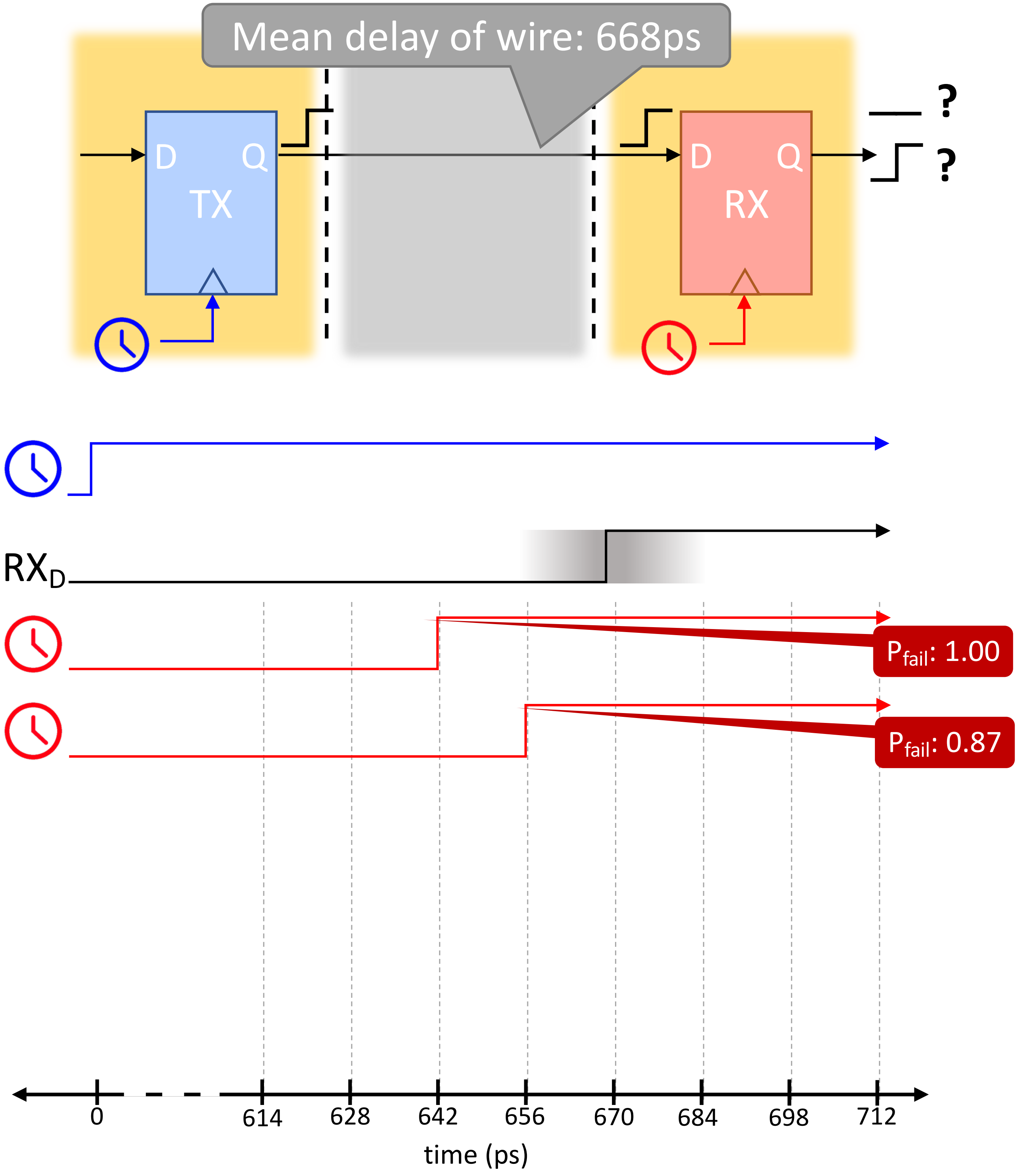
- Delay defined as the skew between TX and RX clock that causes rising transition to be received as 0 and 1 with equal probability



Measuring Delay

- Use phase compensation to measure propagation delay of signal from neighboring chiplet
  - Transmit repeatedly
  - Sweep receiver phase
  - Find phase with 50% failure

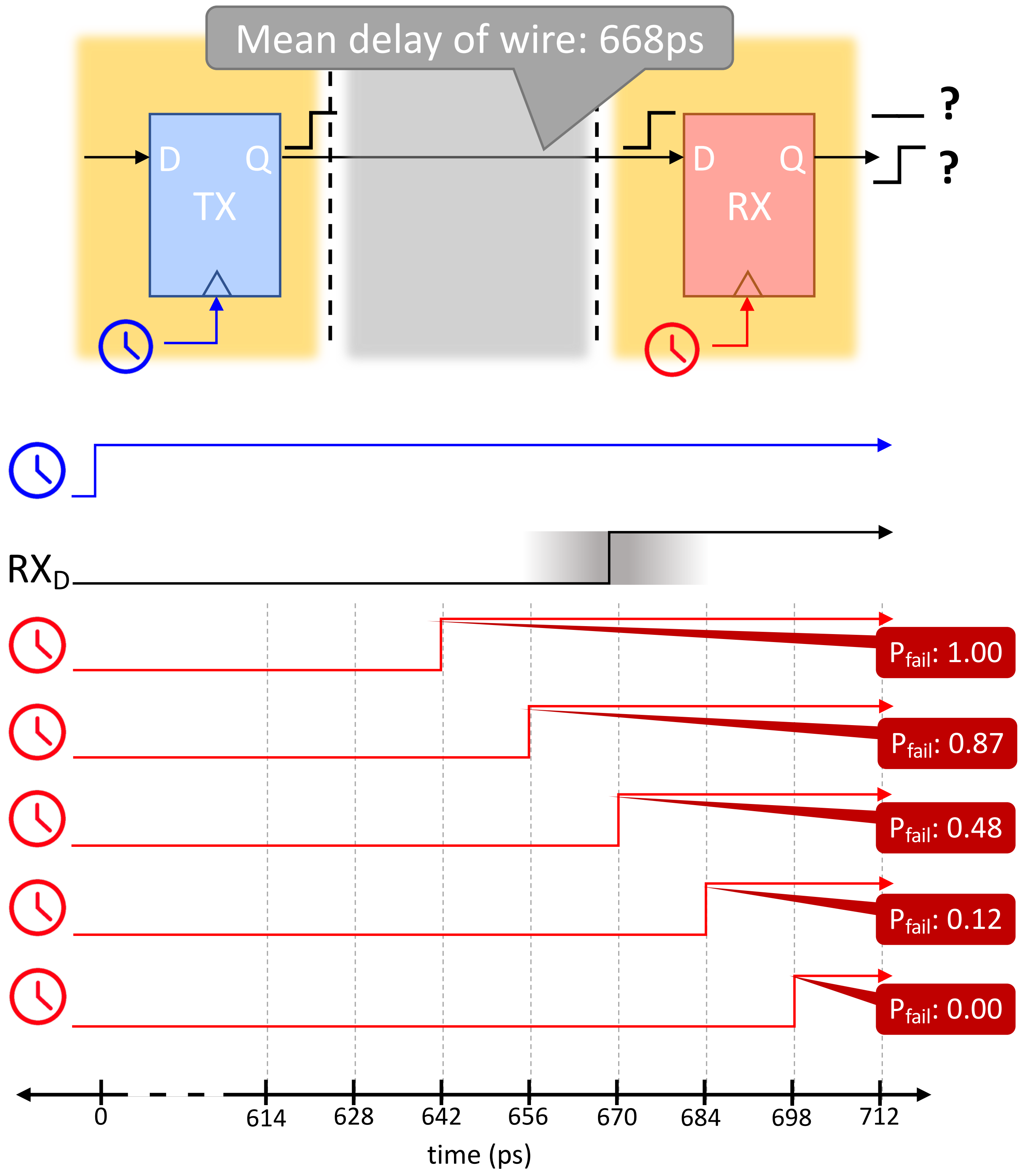
- Delay defined as the skew between TX and RX clock that causes rising transition to be received as 0 and 1 with equal probability



Measuring Delay

- Use phase compensation to measure propagation delay of signal from neighboring chiplet
  - Transmit repeatedly
  - Sweep receiver phase
  - Find phase with 50% failure

- Delay defined as the skew between TX and RX clock that causes rising transition to be received as 0 and 1 with equal probability

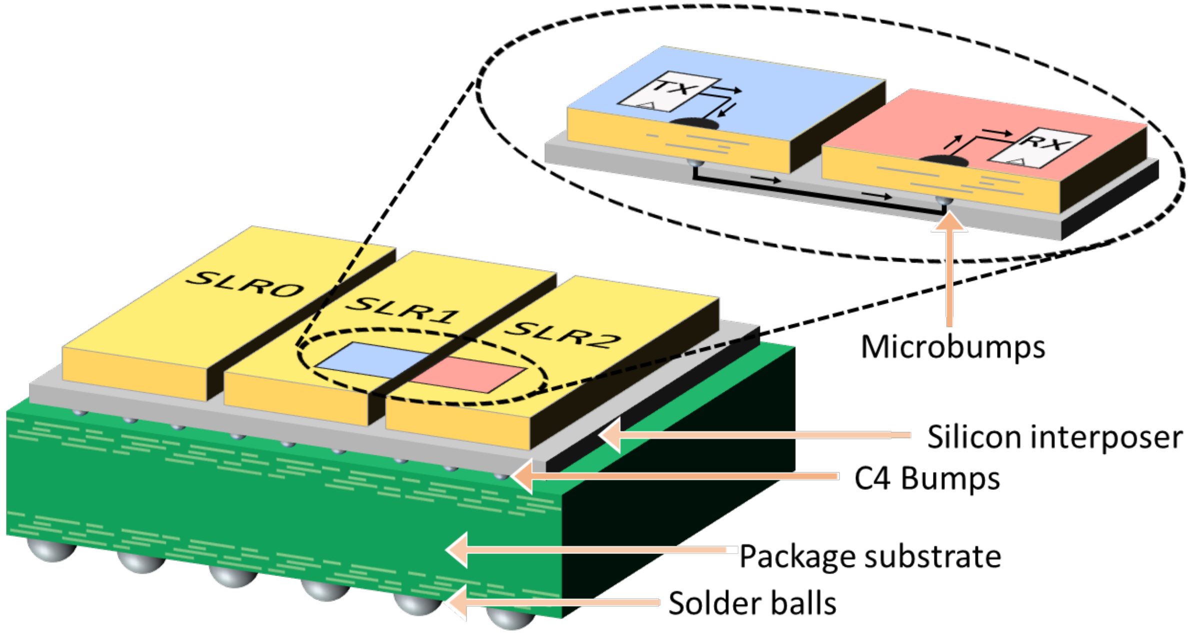


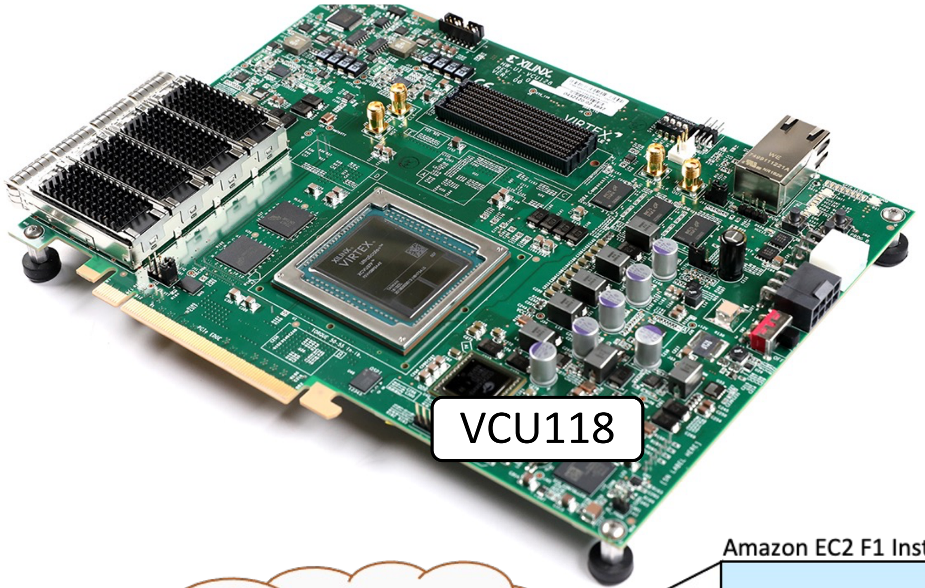
Experimentation Platforms

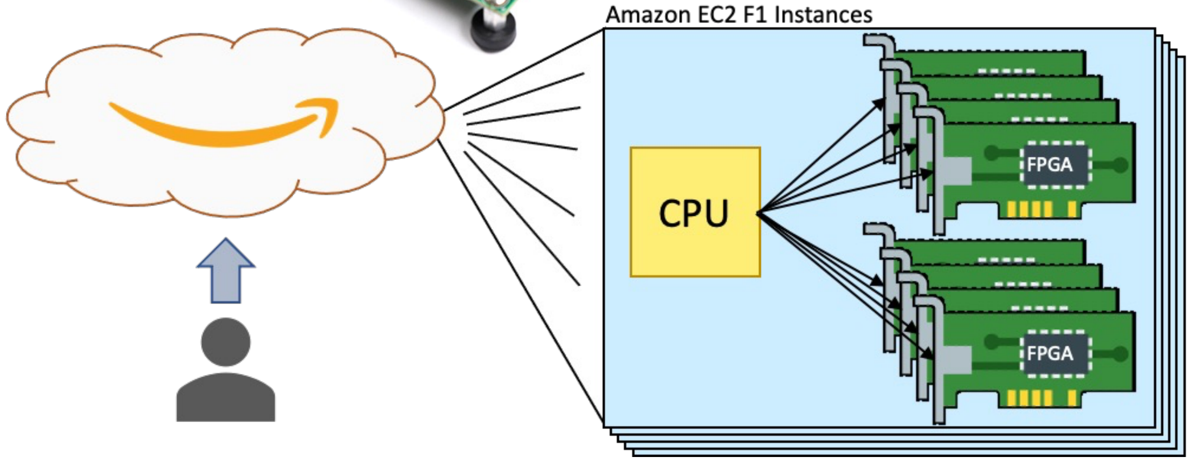
- FPGA as prototype and test platform
  - Provides control over clocking
  - Logic programming enables transmitting arbitrary patterns between chiplets

- Xilinx Virtex Ultrascale+ FPGAs
  - part# xcvu9p-flgb2104-2-i
  - Chiplets organized in Super Logic Regions (SLR)
  - Interposer wires called Super Long Lines (SLL)

- In-lab testing using VCU118 kit
- AWS EC2 F1 instances in cloud to test on larger population







Overview

Measuring Delay

Design & Implementation

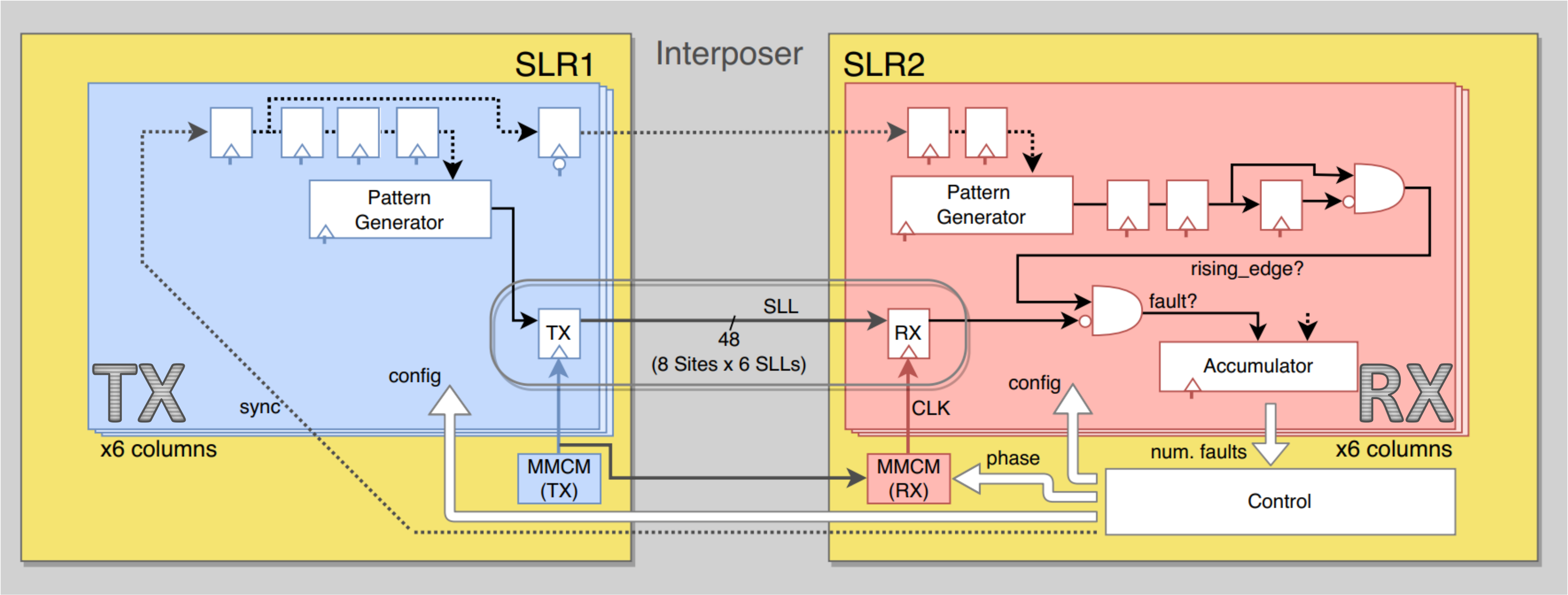
Statistics

Characterization Experiments

Summary

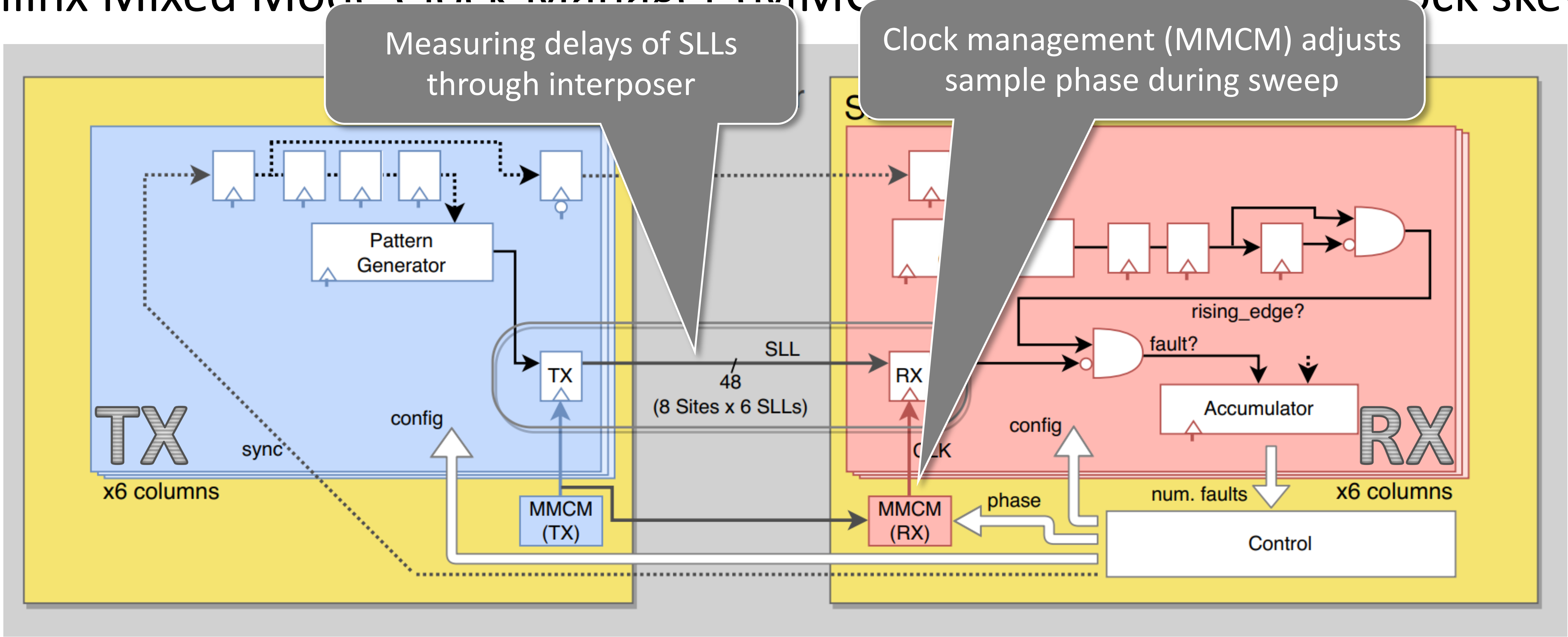
Chiplet PUF - Schematic

- Column-based design with 48 SLLs per column
- Instantiated on multiple columns
- Xilinx Mixed Mode Clock Manager (MMCM) to adjust sampling clock skew



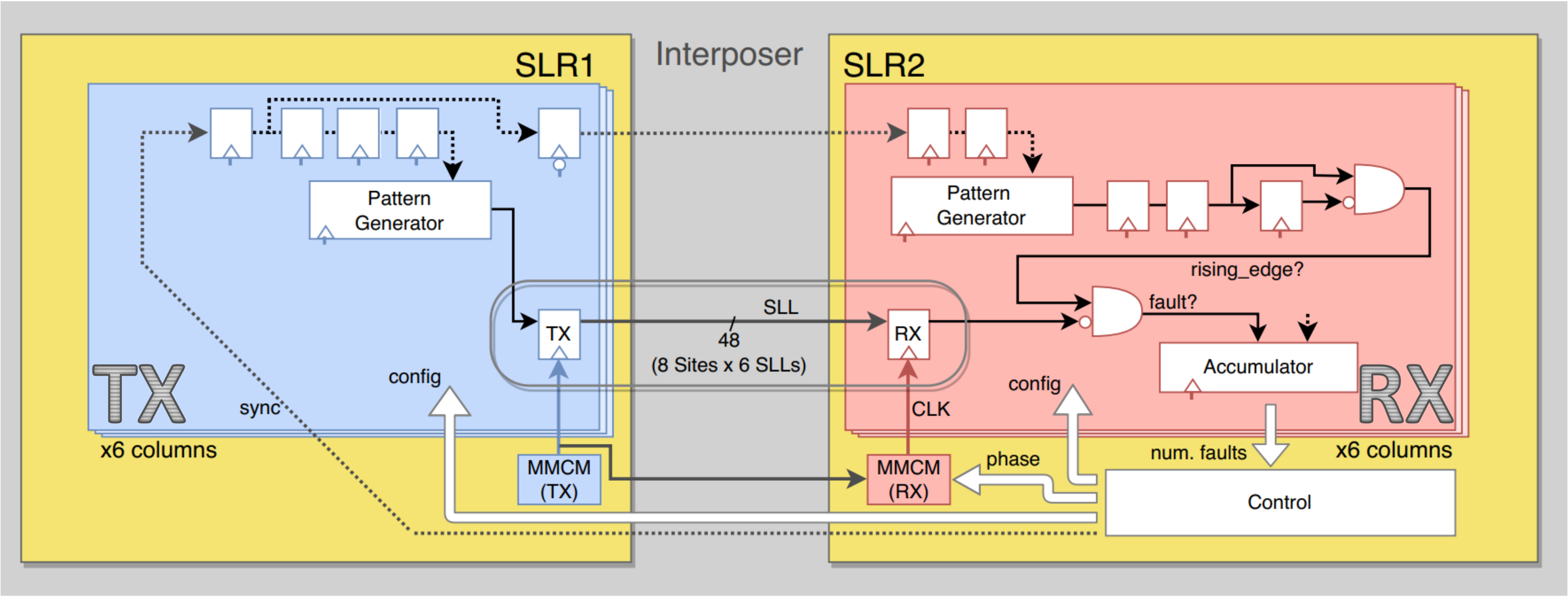
Chiplet PUF - Schematic

- Column-based design with 48 SLLs per column
- Instantiated on multiple columns
- Xilinx Mixed Mode Clock Manager (MMCM) to adjust sampling clock skew



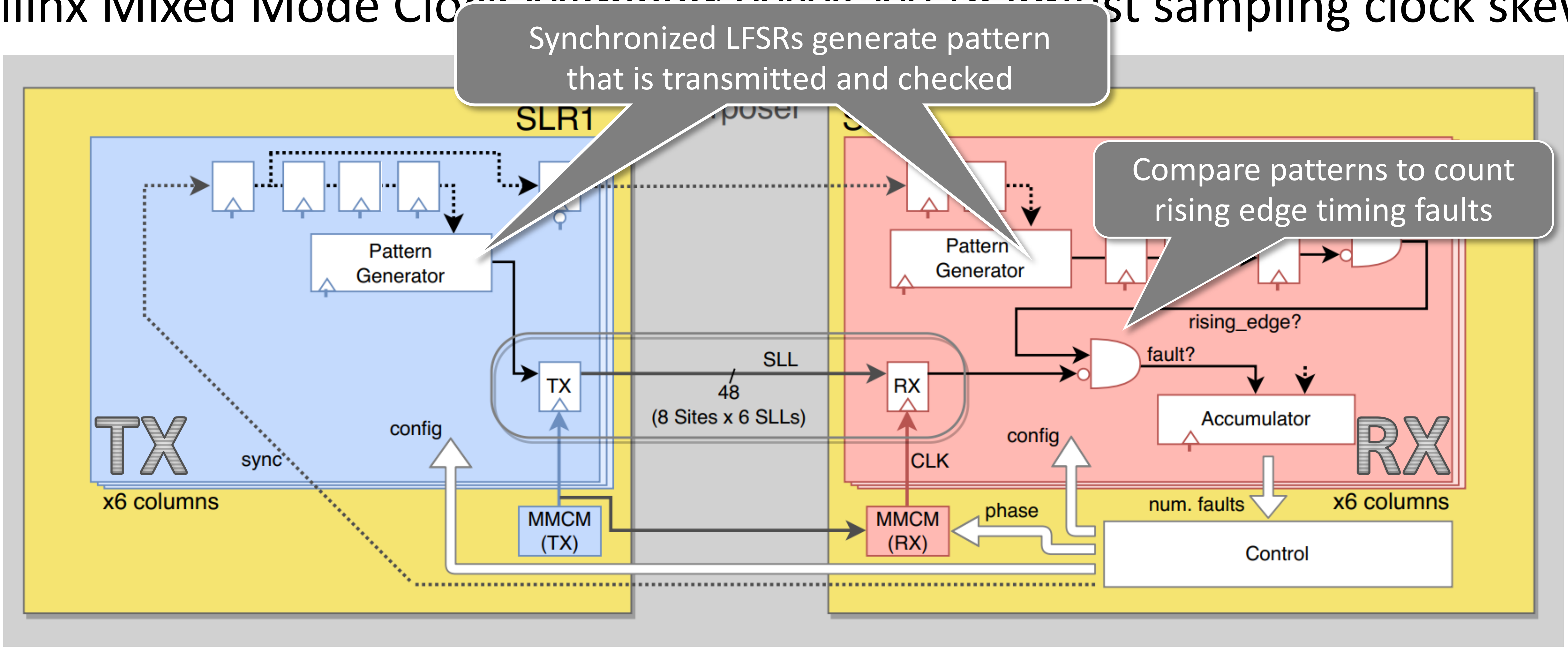
Chiplet PUF - Schematic

- Column-based design with 48 SLLs per column
- Instantiated on multiple columns
- Xilinx Mixed Mode Clock Manager (MMCM) to adjust sampling clock skew



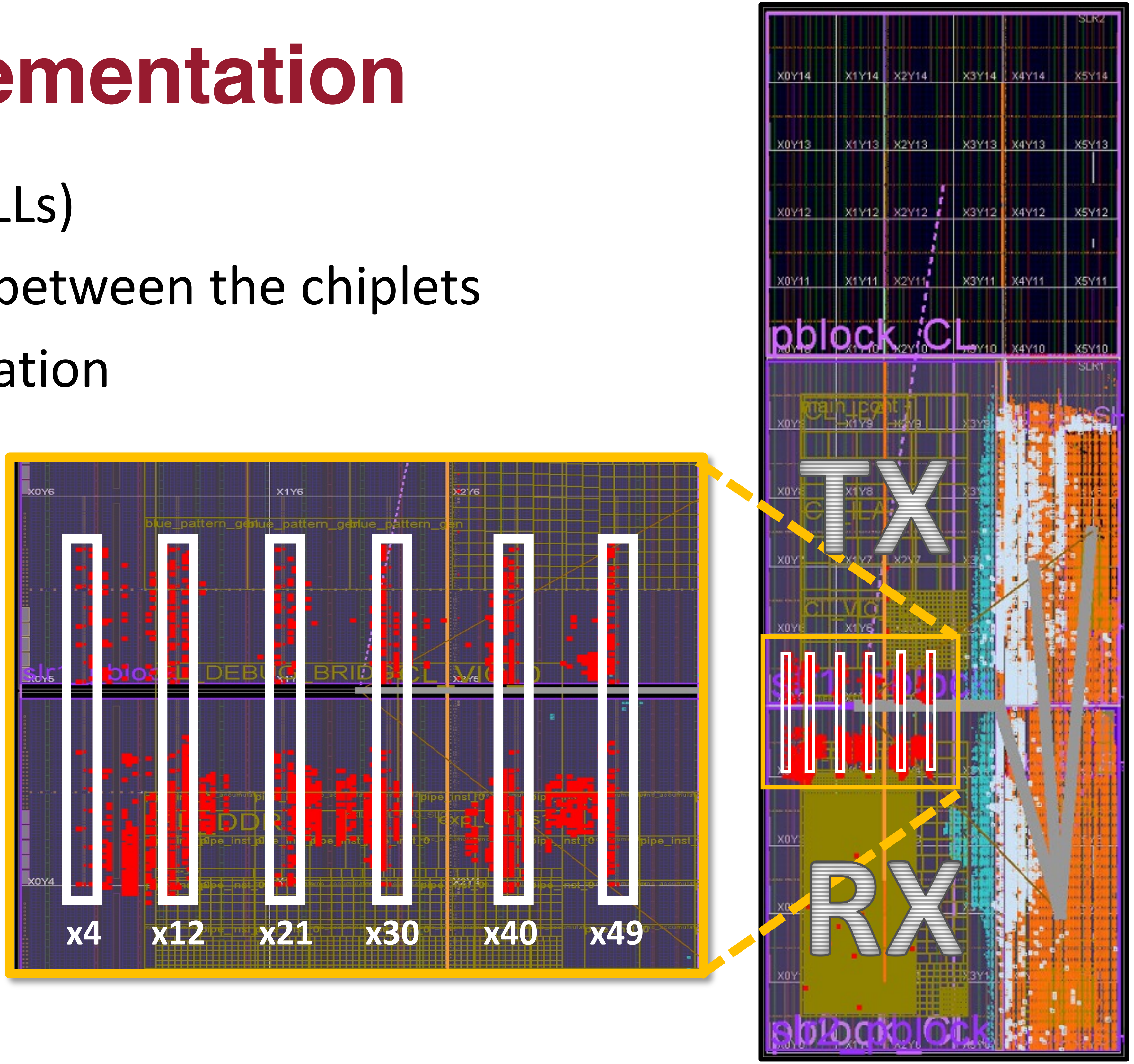
Chiplet PUF - Schematic

- Column-based design with 48 SLLs per column
- Instantiated on multiple columns
- Xilinx Mixed Mode Clock Manager (MMCM) to adjust sampling clock skew



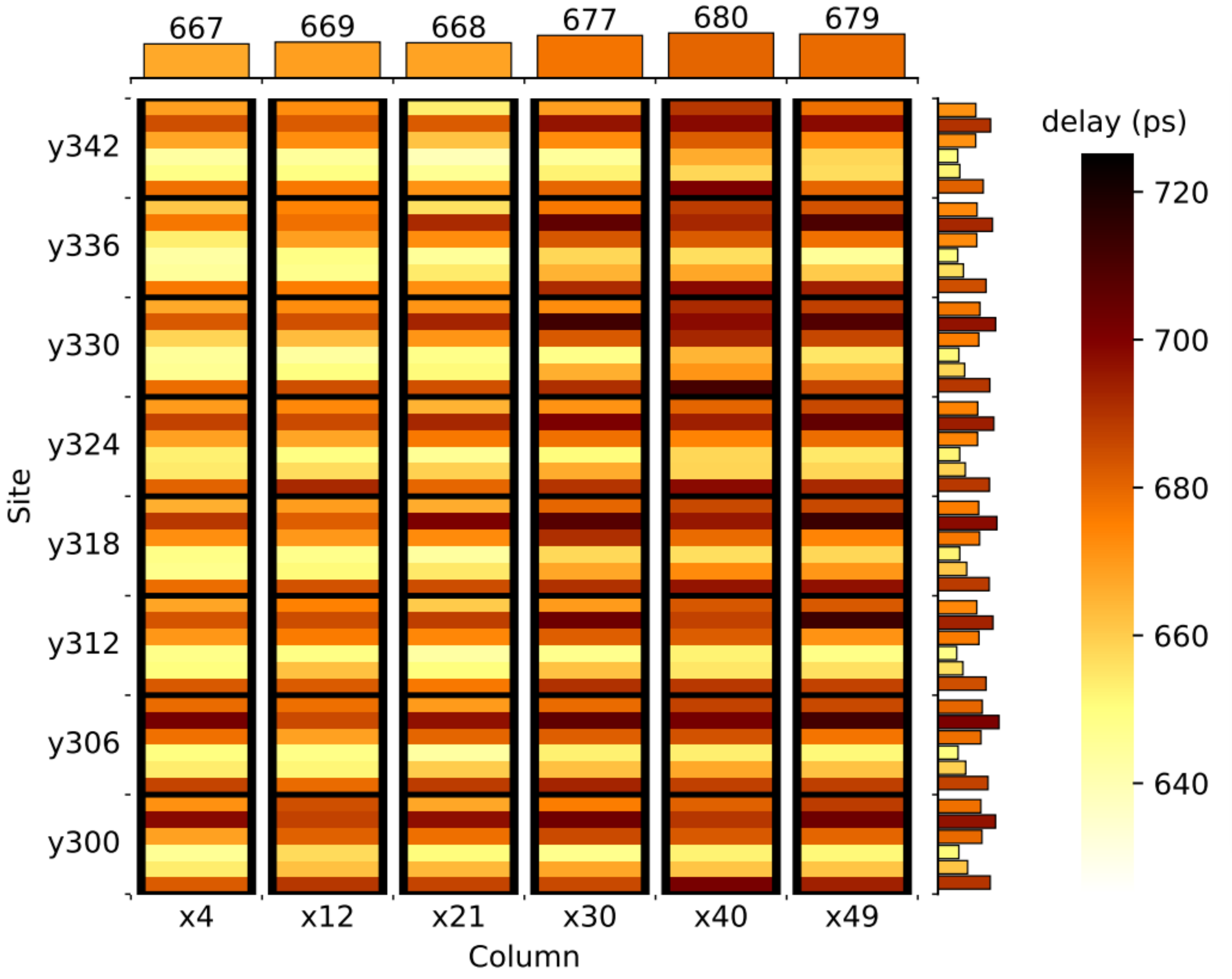
Chiplet PUF - Implementation

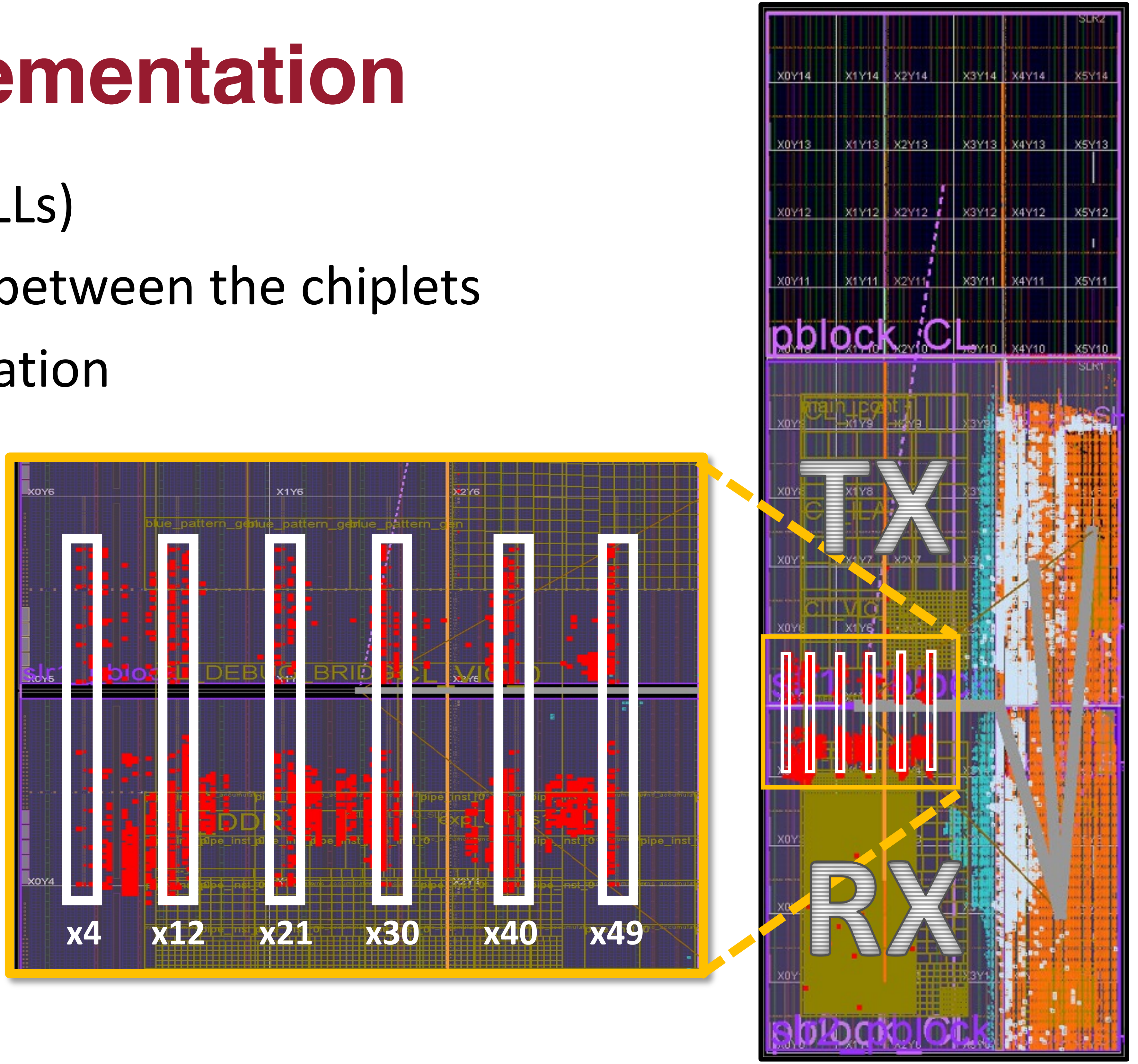
- 6 columns instantiated (288 SLLs)
- Using <2% of the 17,280 SLLs between the chiplets
- 0.27% LUT and 0.34% FF utilization



Chiplet PUF - Implementation

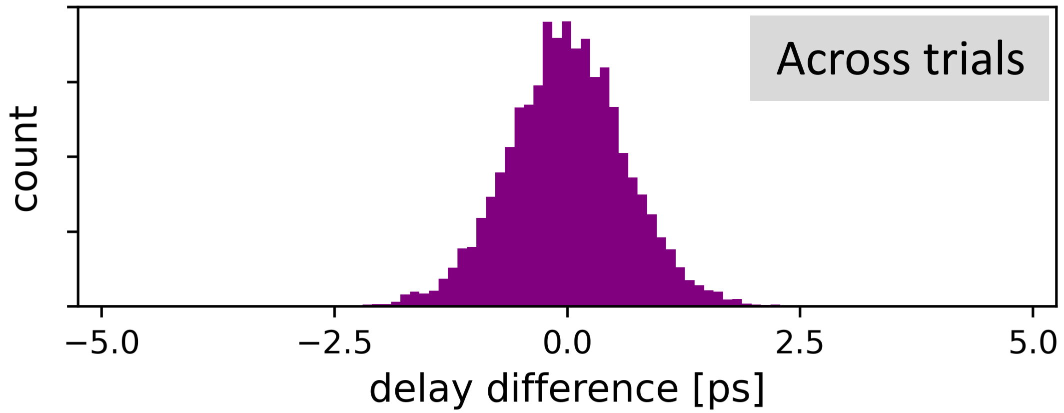
- 6 columns instantiated (288 SLLs)
- Using <2% of the 17,280 SLLs between the chiplets
- 0.27% LUT and 0.34% FF utilization

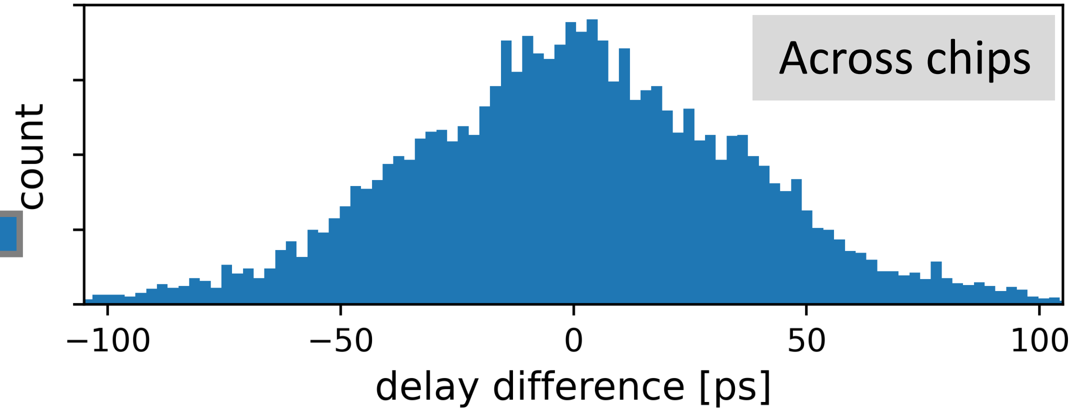


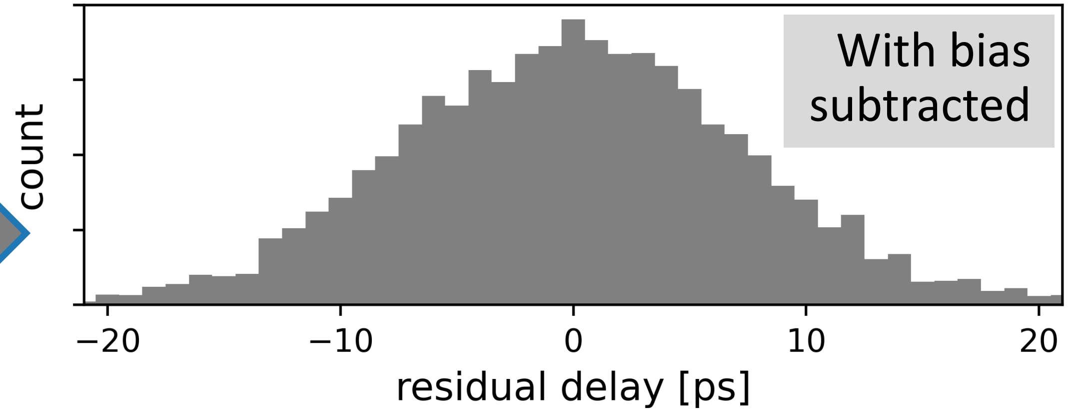


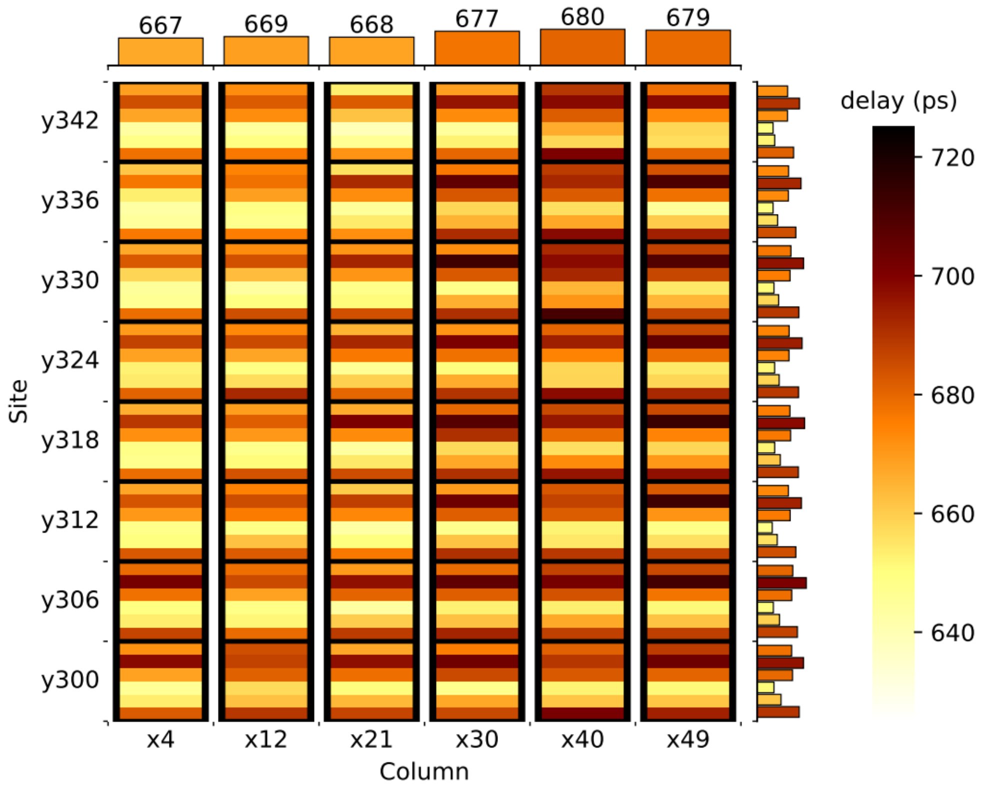
Measured SLL Delays

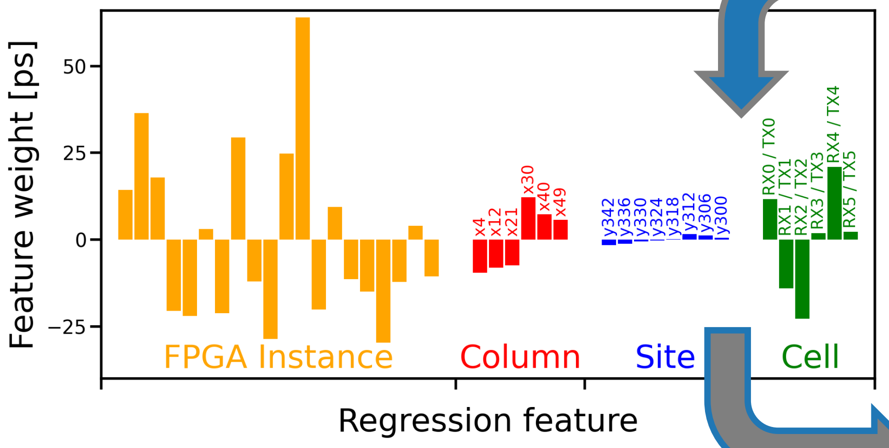
- SLL delay measurements in 630-720 ps range
- Reliable and instance-specific
  - 0.5 ps difference across trials
  - 29.4 ps difference across chips
  - 5.8 ps difference after removing biases





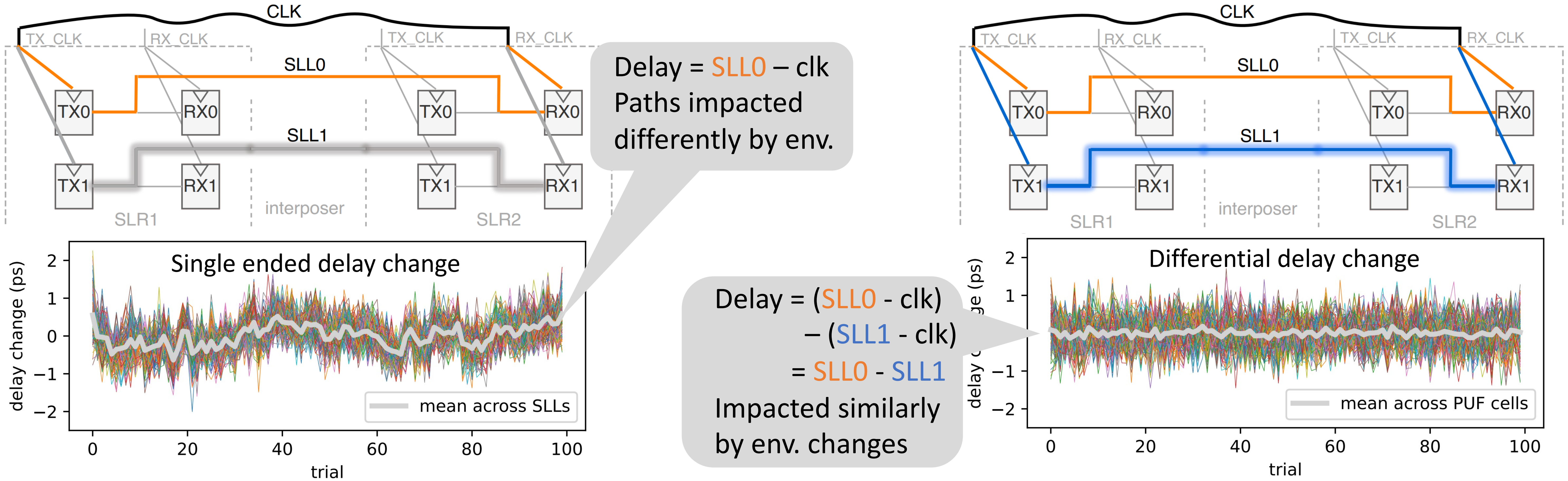






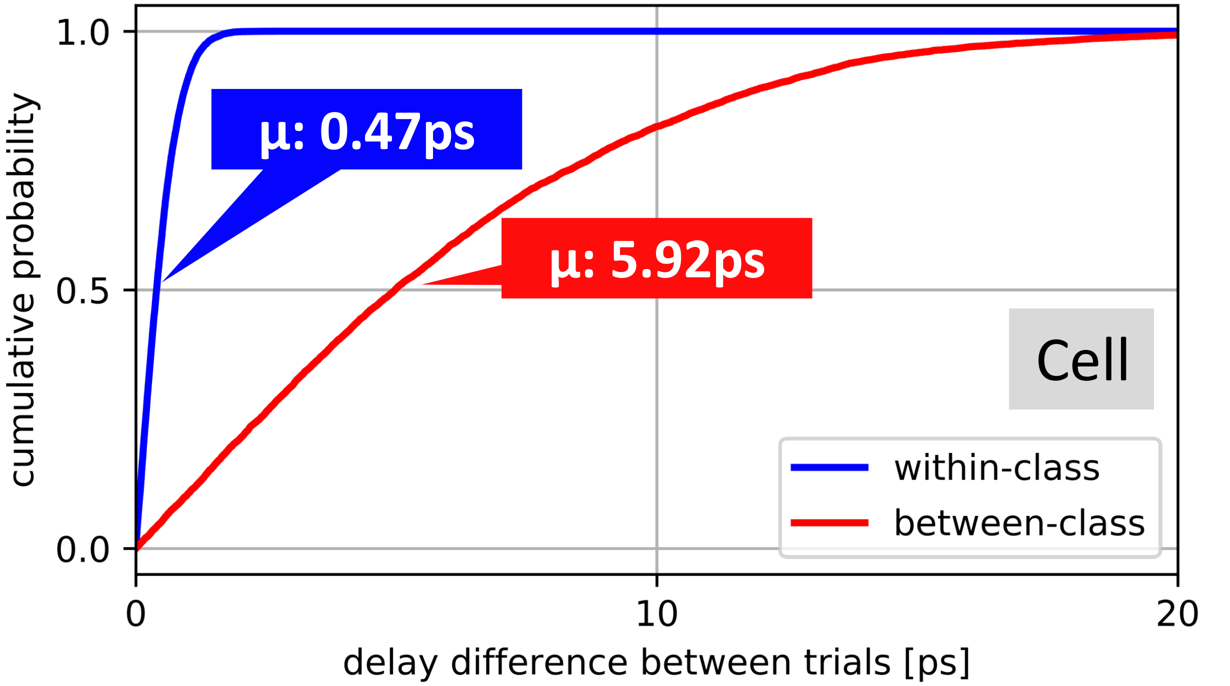
Robust Delay Measurement

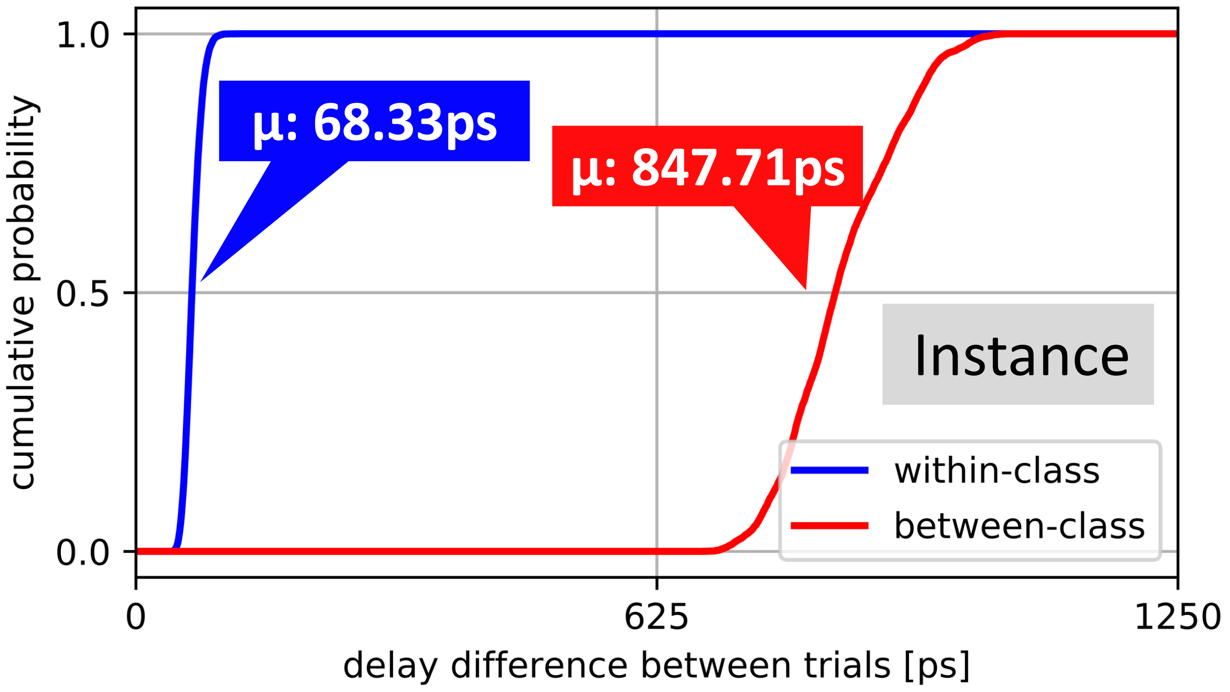
- Differential delays between SLL pairs outperform single-ended SLL delays
- Delay measurement becomes independent of clock path
  - Less delay drift because clock changes become common mode
  - Clock path reused across SLLs ➜ Avoid miscounting skew variation as uniqueness



Within and Between-Class Distances

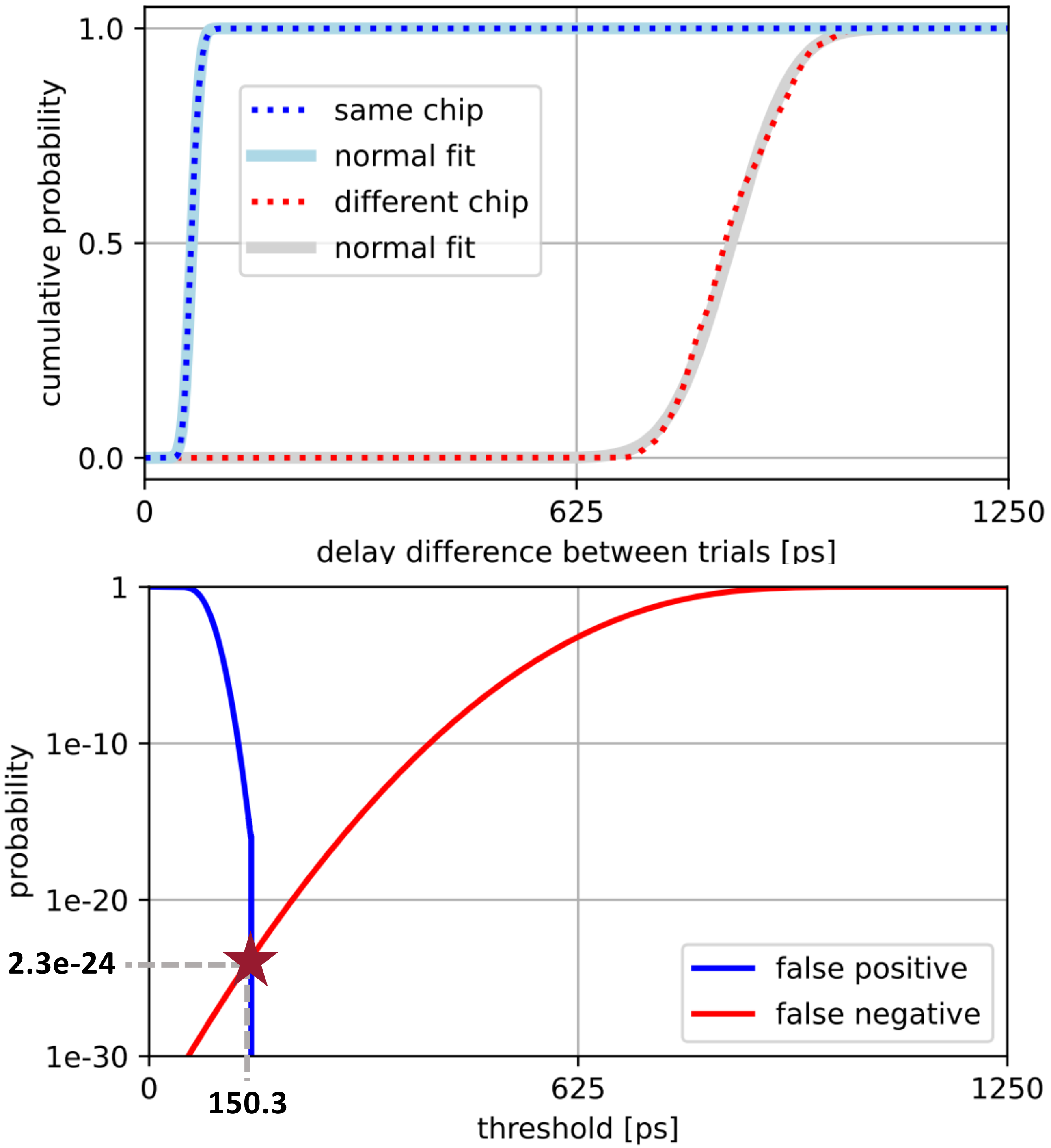
- Data from 20 AWS EC2 F1 instances
  - Same VU9P part used for local testing
- Cumulative distributions
  - PUF cell difference = $\left|D_{t,s} - D_{t',s}\right|$
  - Instance difference = $\sum_{s=1}^{144}\left|D_{t,s} - D_{t',s}\right|$
- Separation of within-class and between-class distances is consistent with the PUF being a reliable and unique fingerprint





Type I and II errors

- Empirical data approximately normally distributed
- Fitted normal distributions used to estimate false positive and false negative rates in a larger population
- Equal error point occurs at threshold = 150.3 ps
- Type I and II error rates are 2.3e-24



Overview

Measuring Delay

Design & Implementation

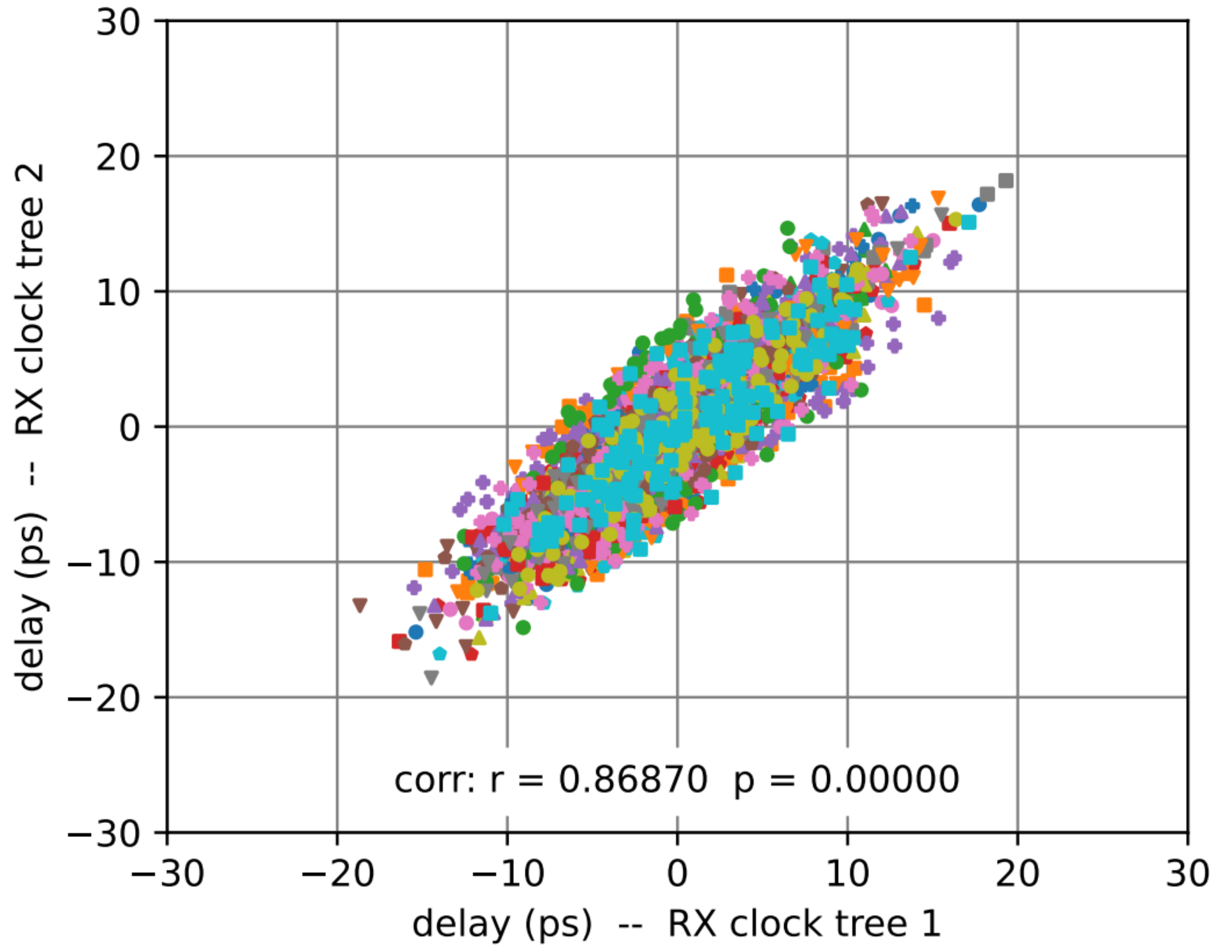
Statistics

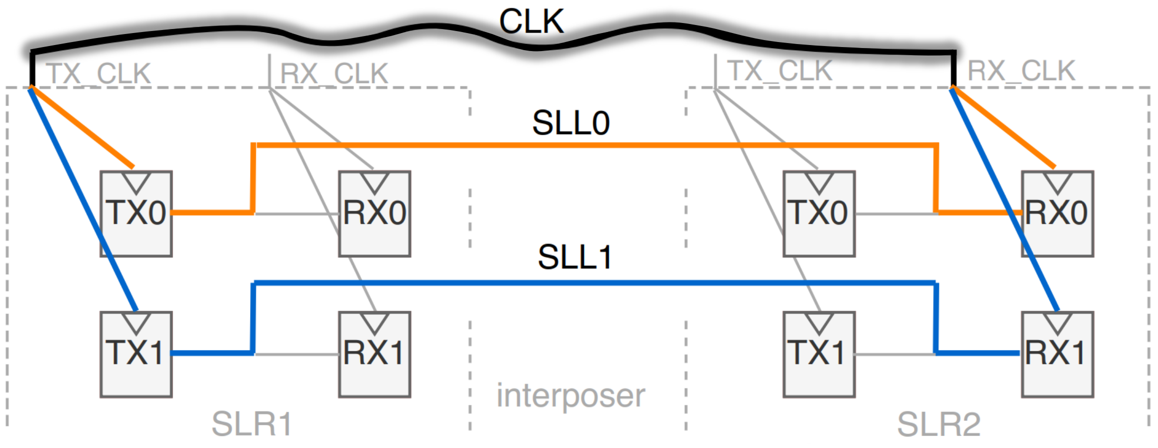
Characterization Experiments

Summary

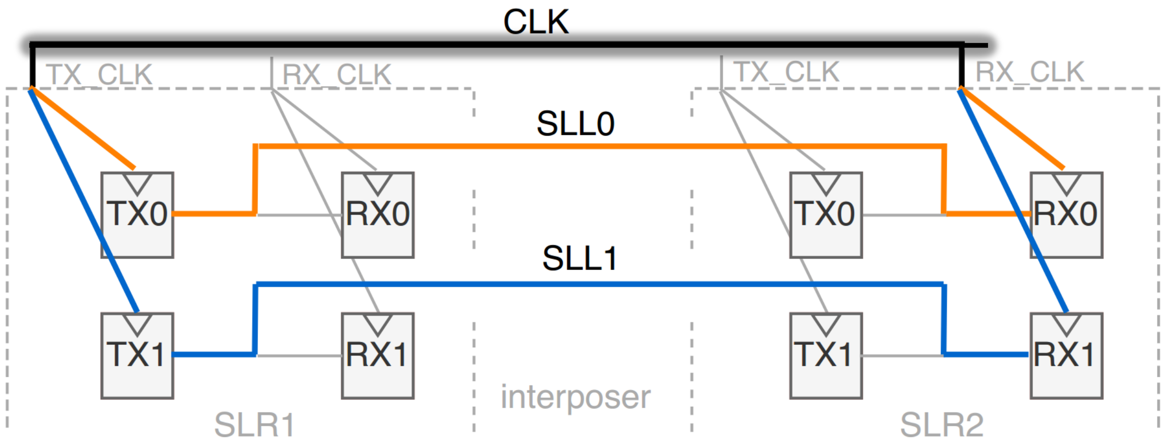
Characterization – Using different clock trees

- Testing whether differential PUF output is insensitive to clock
  - Crucial for minimizing impact of environmental fluctuations and of variation on clock tree
- Compare PUFs between two variants:
  - **Same** interposer wires, **same** drivers
  - **Different** clock distribution path
- Highly correlated outputs (r = 0.869) in experiments on 20 cloud instances x 144 cells
- Conclusion: PUF insensitive to clock, as intended



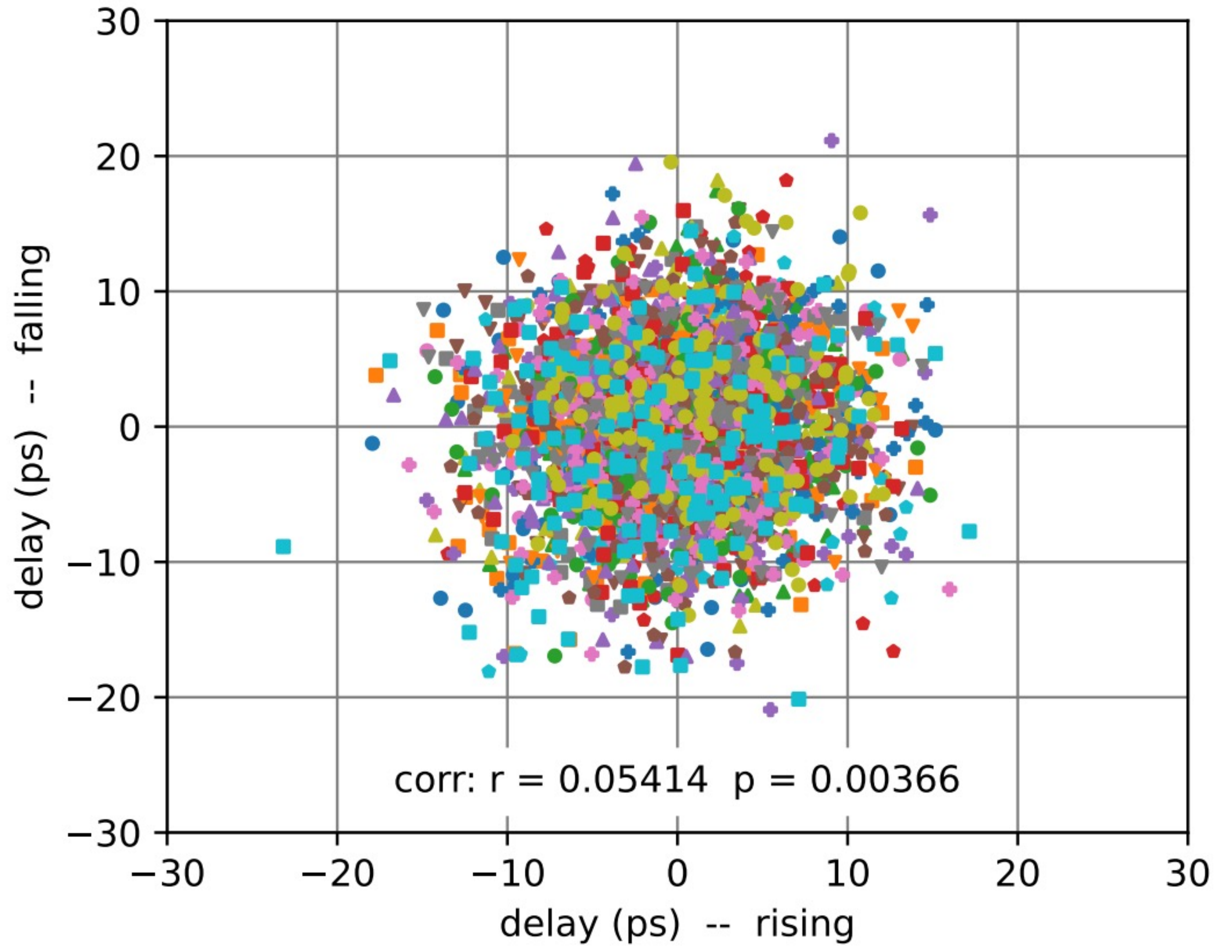


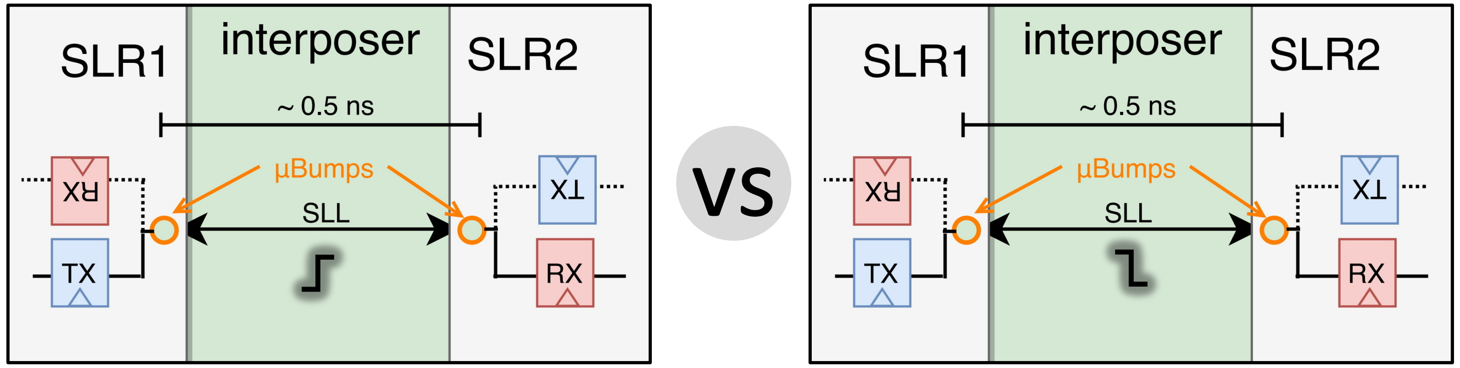
VS



Characterization – Rising vs Falling transition

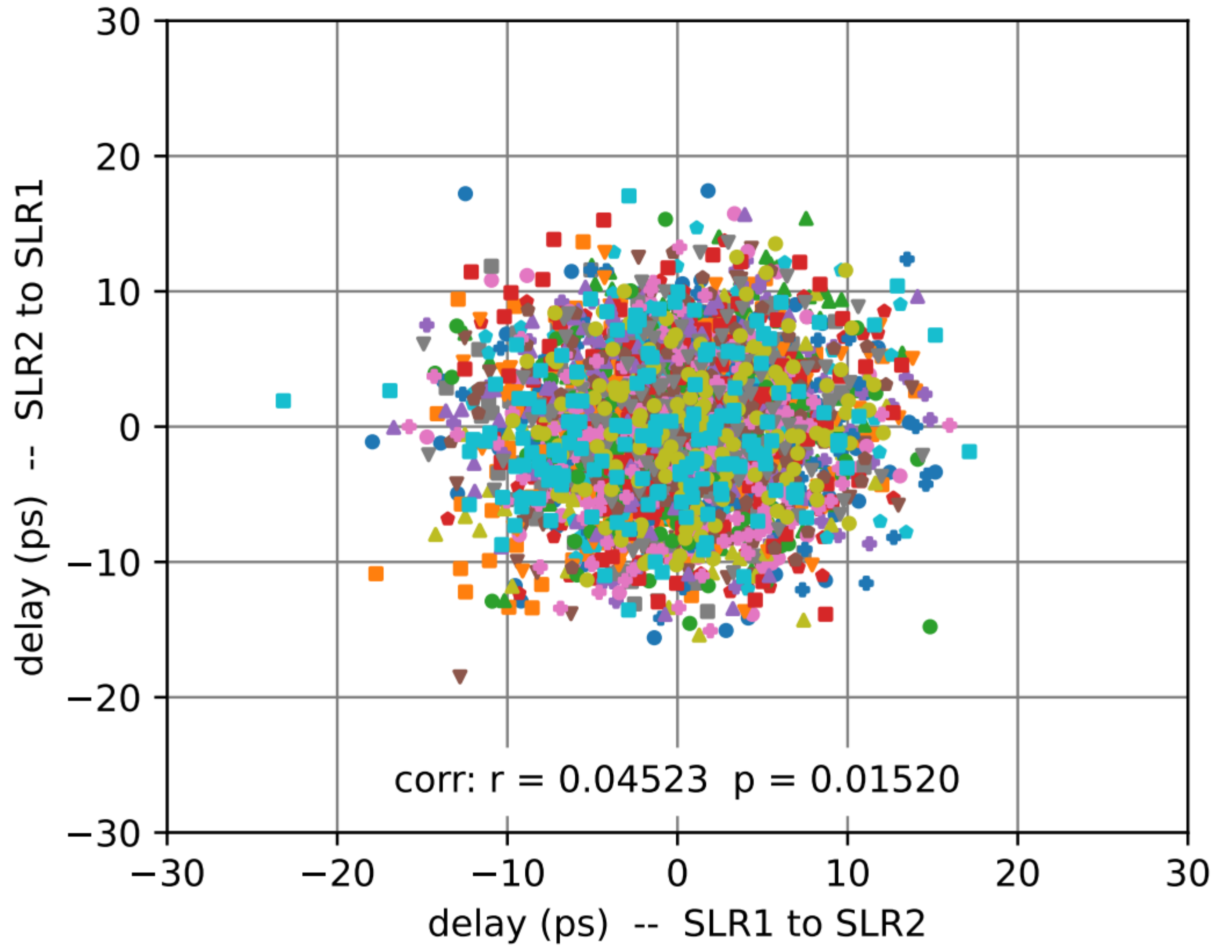
- Testing whether drivers or interposer wires dominate variability
- Compare PUF variants using rising or falling transition to measure delays:
  - **Same** interposer wires in both cases
  - **Different** transistors driving wires
  - **Different** transistors in sampling flops
- Weaker correlation (r=0.054) implies that variation of interposer wires is not dominant factor
- Conclusion: Transistor variation is a significant source of entropy

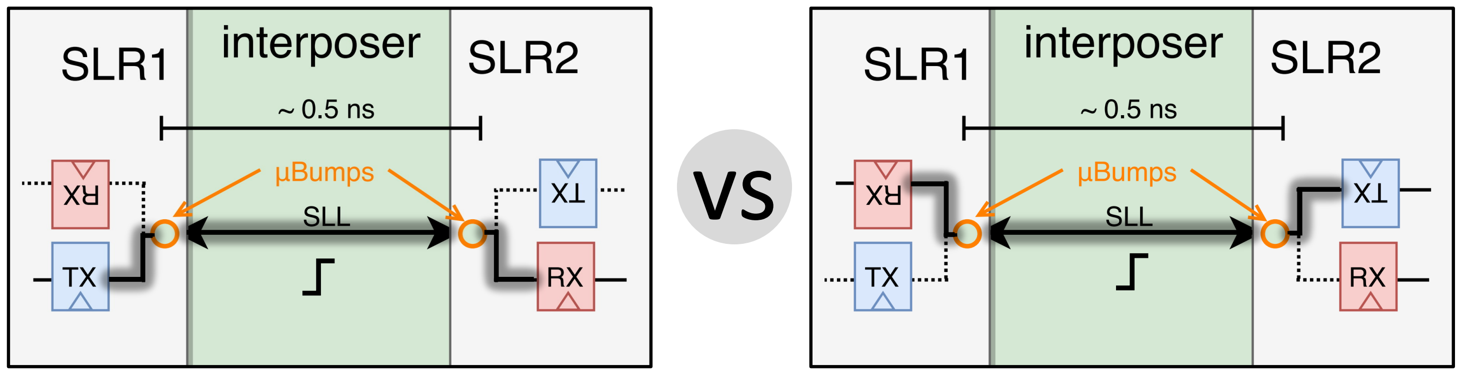




Characterization – Swapping TX and RX

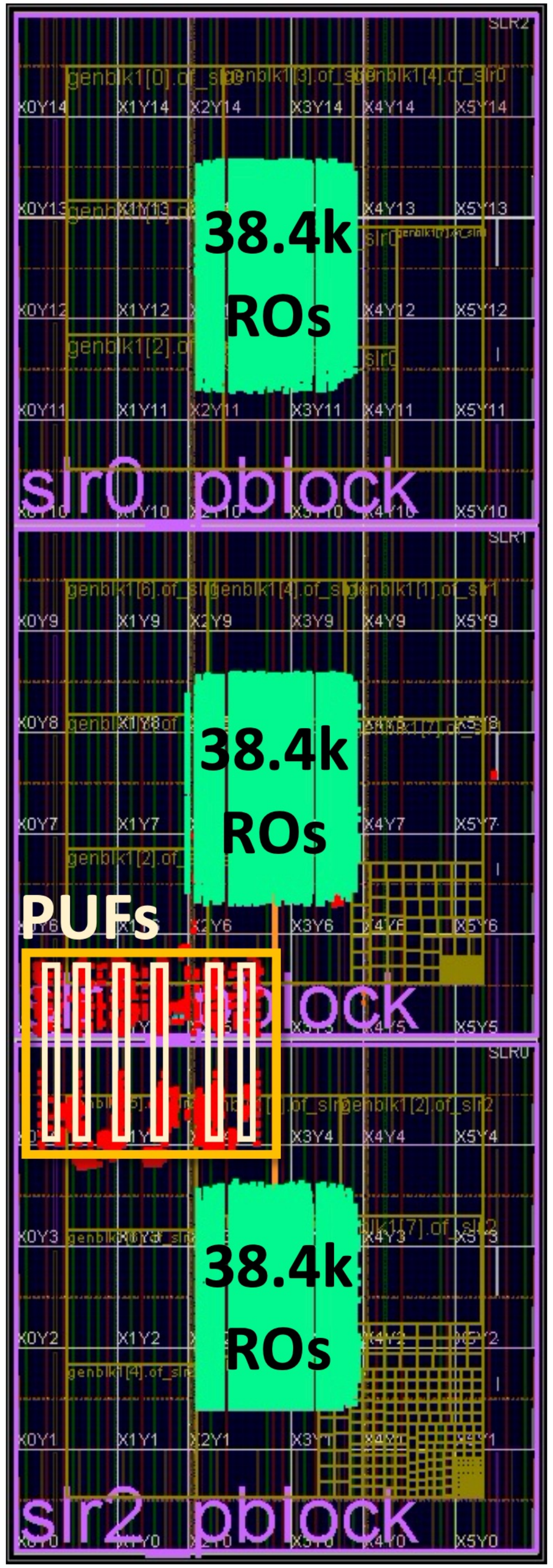
- Testing impact of driving same wire from each end
  - Possible in Xilinx architecture because SLLs are bidirectional
- Comparing two variants with:
  - **Same** interposer wires
  - **Different** transistor instances
  - **Different** environment for TX and RX
- Weak correlation (r=0.045) again implies that variation of interposer wires is not dominant factor
- <u>Conclusion</u>: Transistor variation is a significant source of entropy





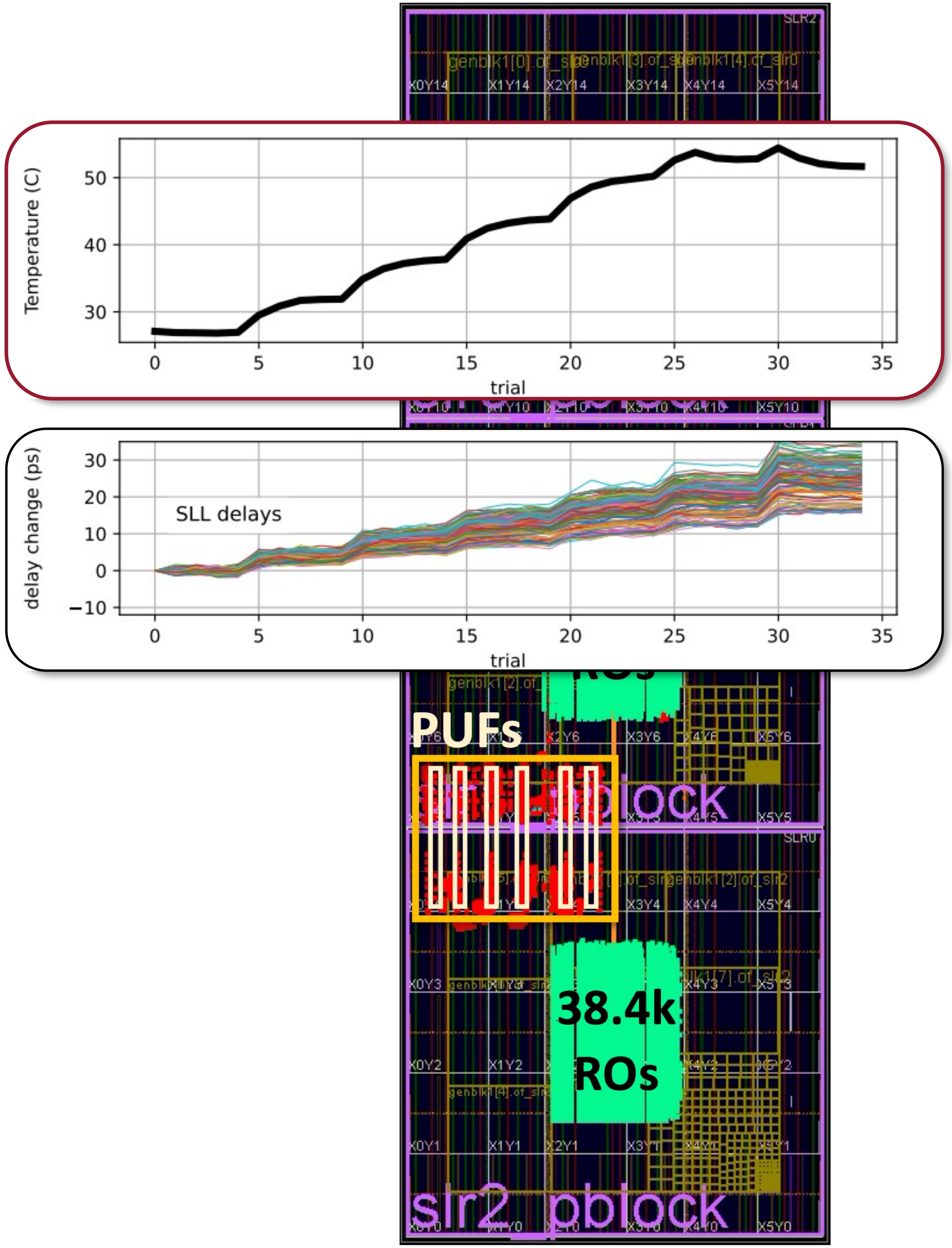
Heating & Compensation

- 38.4k power wasting ring oscillators (ROs) added to each SLR
  - Controlled in groups of 4.8k
- SLL delays increase proportional to die temperature
  - Sensitivity is non-uniform
  - Causes error in output of differential PUF cells
- Compensate delay by learning and applying per-SLL delay coefficient
  - Does not use temperature sensor



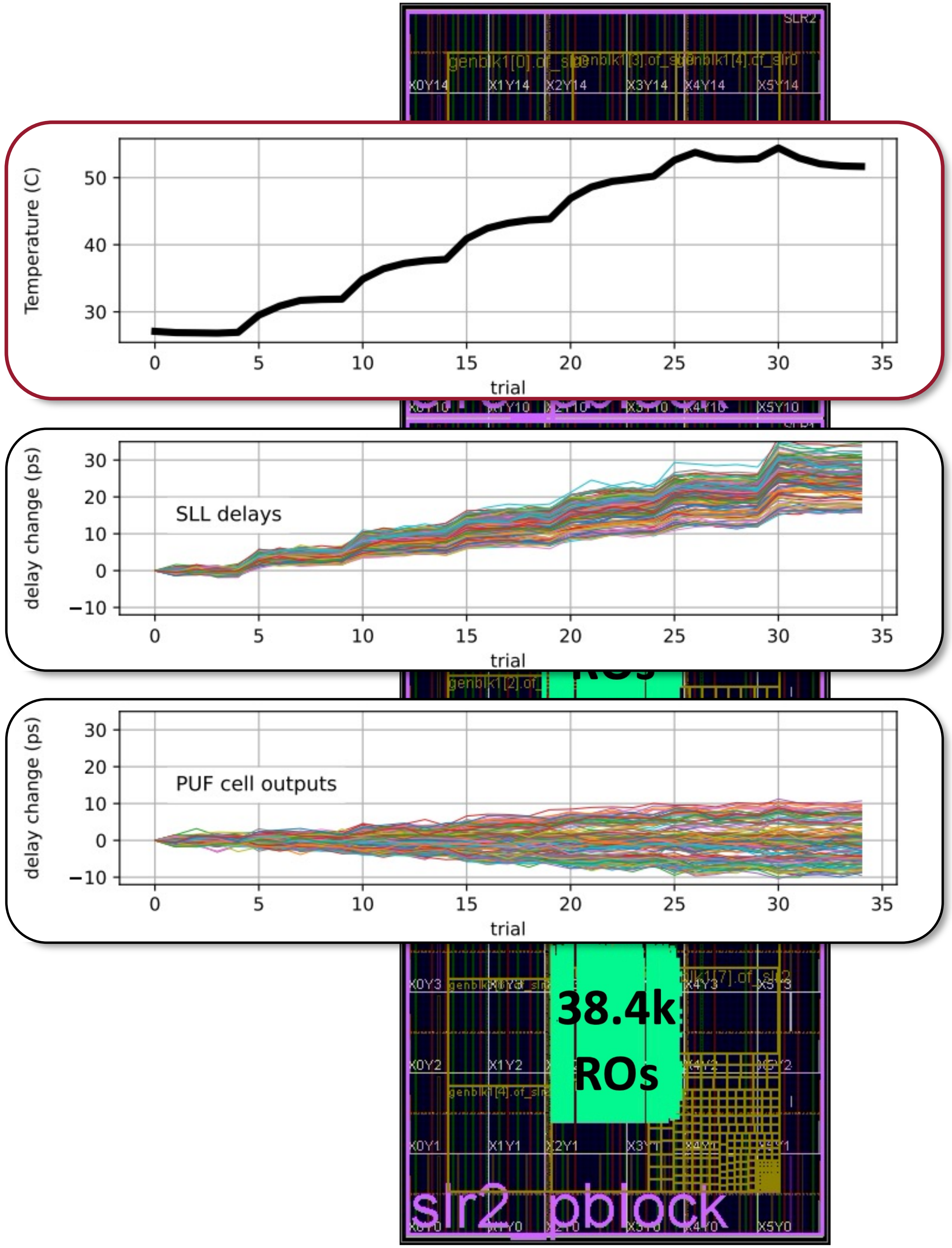
Heating & Compensation

- 38.4k power wasting ring oscillators (ROs) added to each SLR
  - Controlled in groups of 4.8k
- SLL delays increase proportional to die temperature
  - Sensitivity is non-uniform
  - Causes error in output of differential PUF cells
- Compensate delay by learning and applying per-SLL delay coefficient
  - Does not use temperature sensor



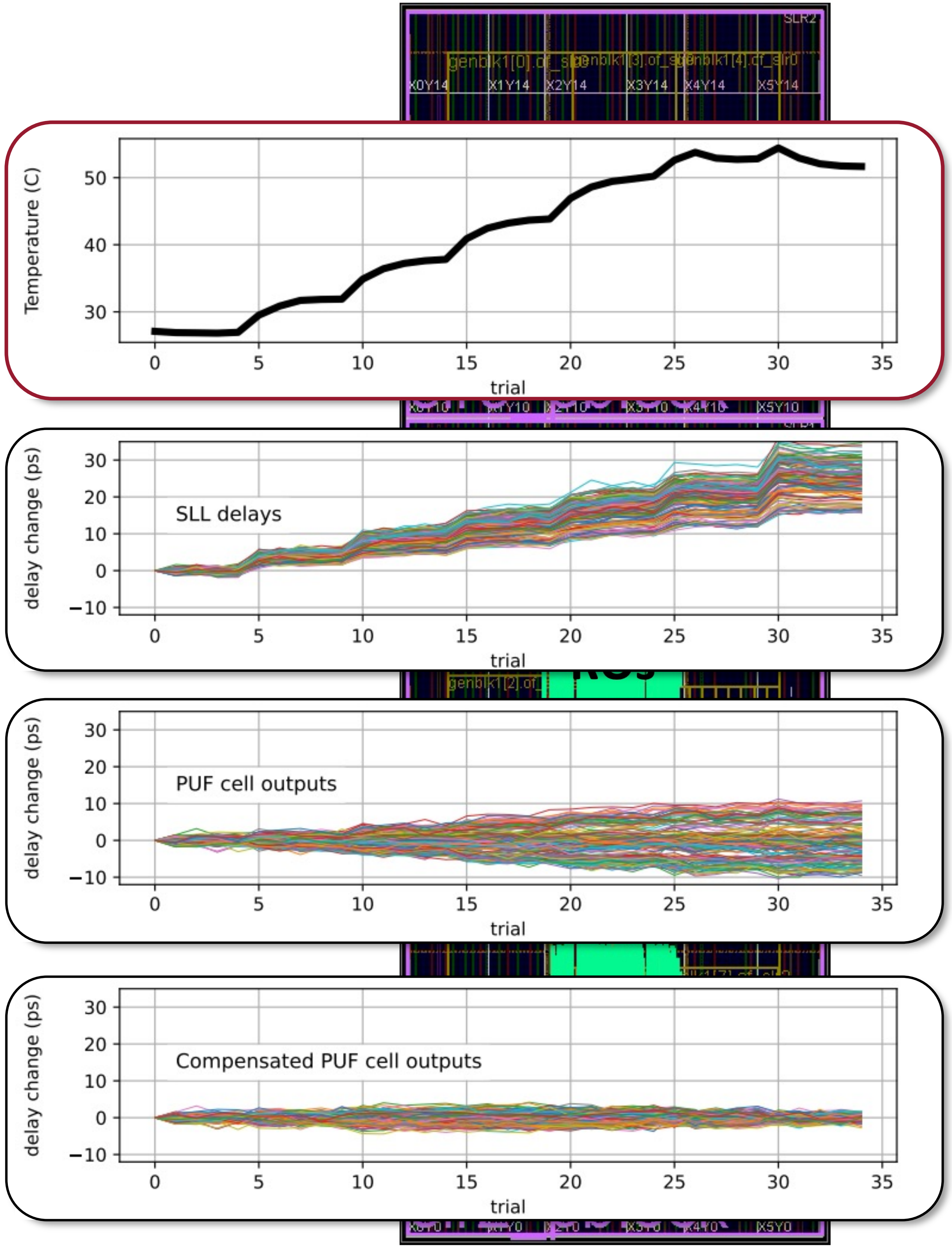
Heating & Compensation

- 38.4k power wasting ring oscillators (ROs) added to each SLR
  - Controlled in groups of 4.8k
- SLL delays increase proportional to die temperature
  - Sensitivity is non-uniform
  - Causes error in output of differential PUF cells
- Compensate delay by learning and applying per-SLL delay coefficient
  - Does not use temperature sensor



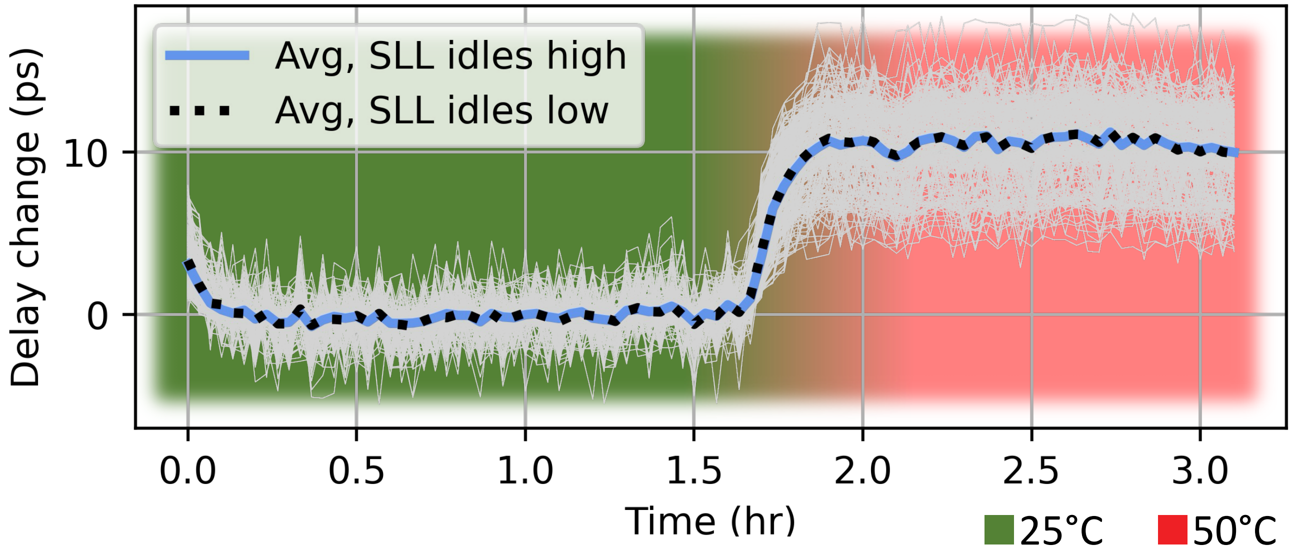
Heating & Compensation

- 38.4k power wasting ring oscillators (ROs) added to each SLR
  - Controlled in groups of 4.8k
- SLL delays increase proportional to die temperature
  - Sensitivity is non-uniform
  - Causes error in output of differential PUF cells
- Compensate delay by learning and applying per-SLL delay coefficient
  - Does not use temperature sensor



Testing for Impact of Aging

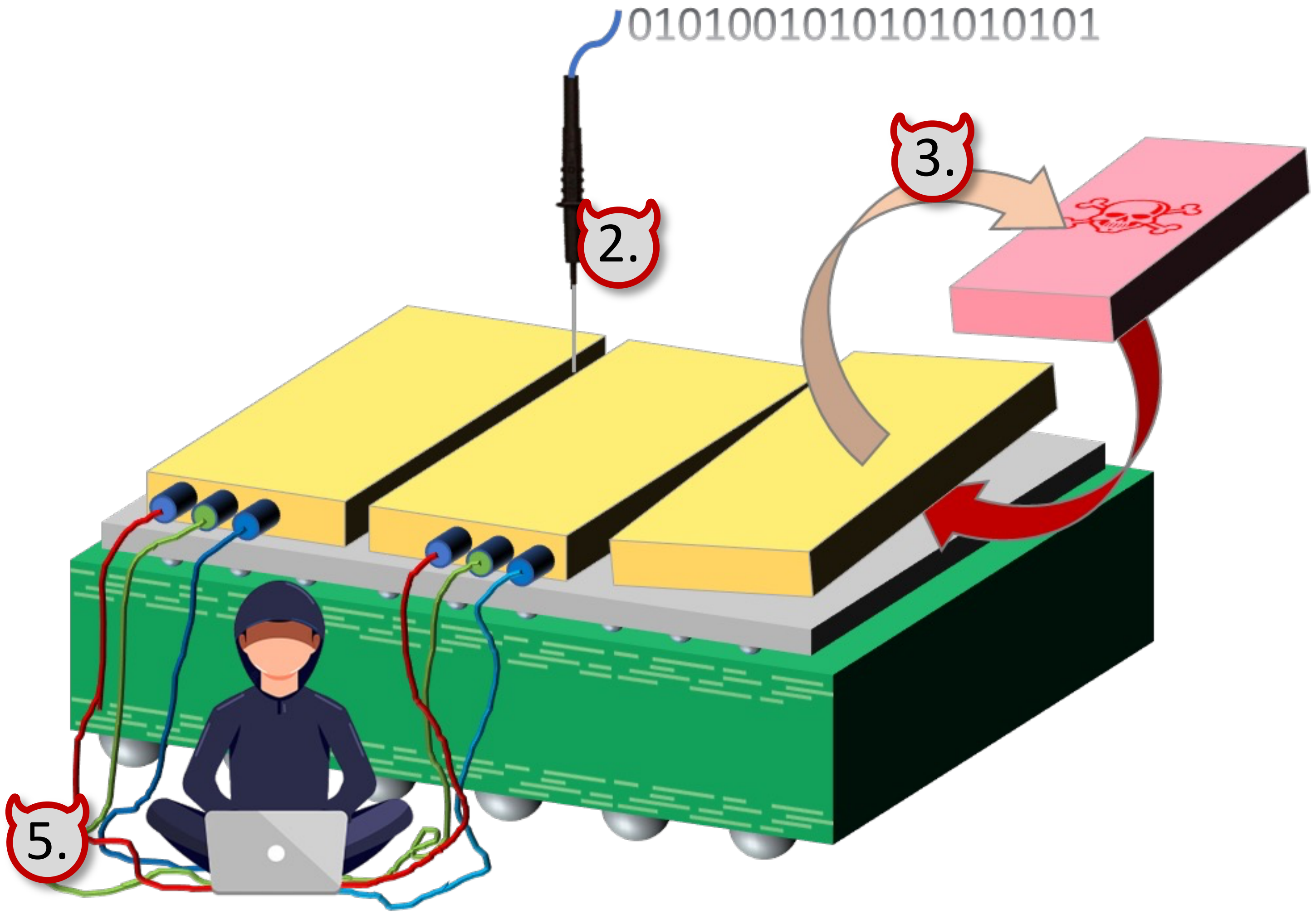
- Aging can change circuit delay
- Potentially detrimental to PUF response stability
- Test: randomly assign SLLs to two groups, which are aged in opposite directions
  - Pull-high vs pull-low when idling between measurements
- Conclusion: groups do not diverge, implying that aging has little to no effect



Threats Addressed by Chiplet PUF

- PUF responses stable at picosecond level
  - Provides evidence of package integrity

1. Trojans in co-packaged chiplets
2. Probing exposed interposer wires
3. Die-swapping
4. Side-channels from within package
5. Man-in-the-middle



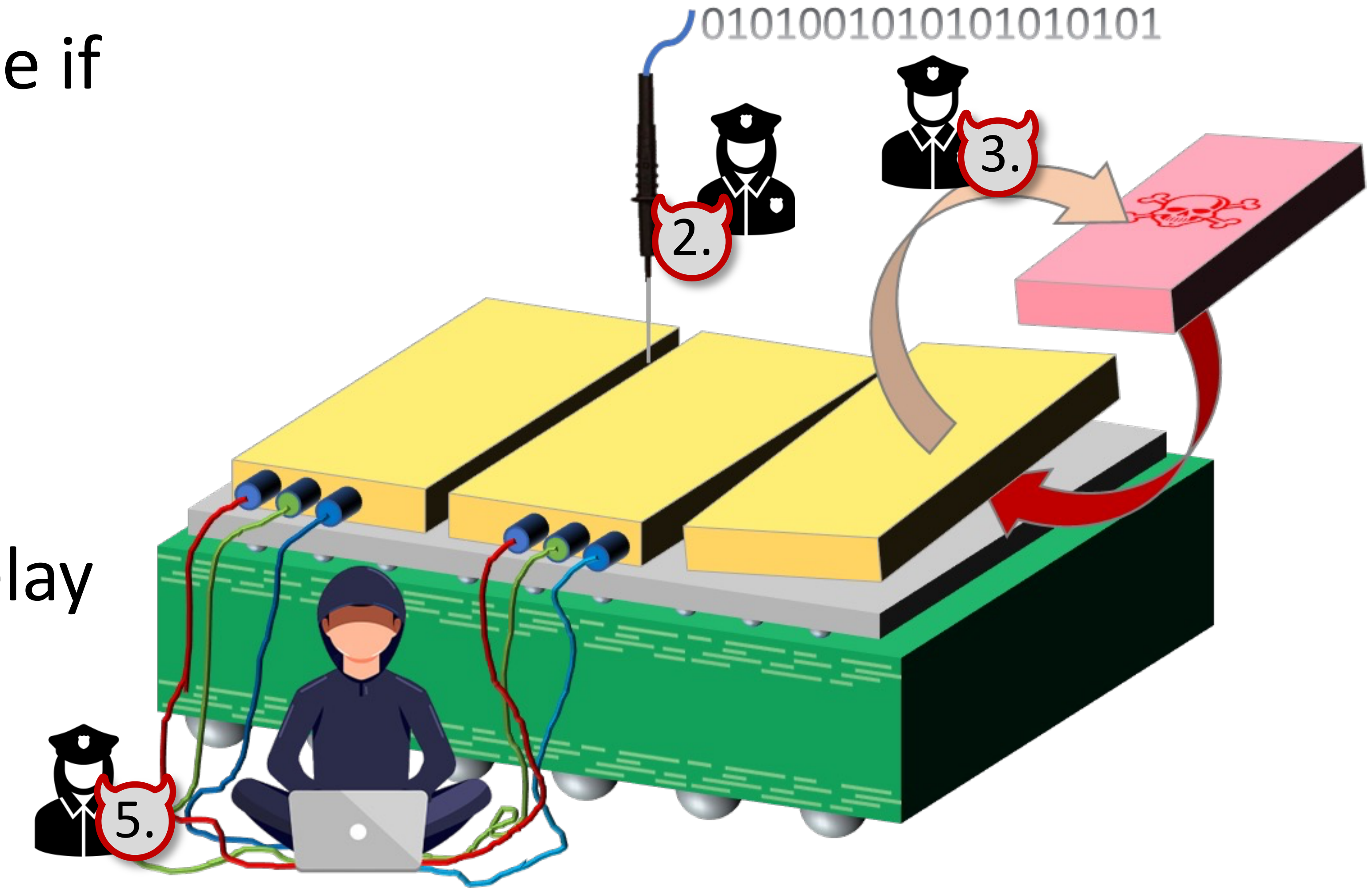
Threats Addressed by Chiplet PUF

- PUF responses stable at picosecond level
  - Provides evidence of package integrity

1. Trojans in co-packaged chiplets
2. Probing exposed interposer wires
3. Die-swapping
4. Side-channels from within package
5. Man-in-the-middle

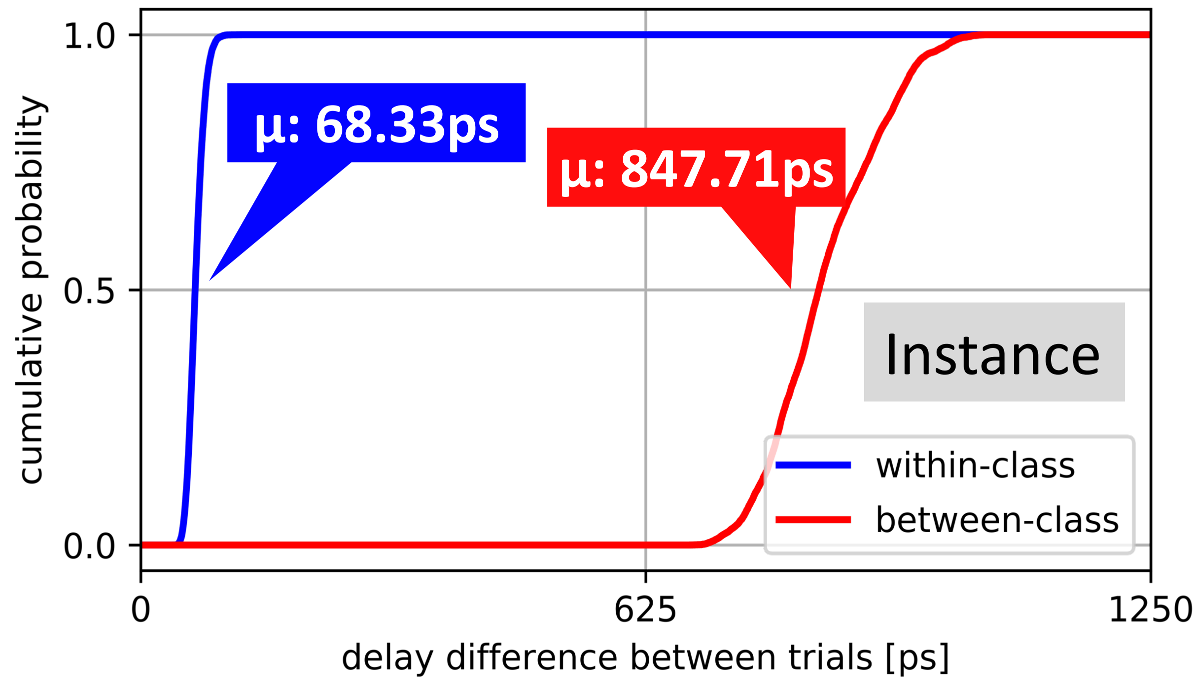
- Physical probes and MITM detectable if causing delay changes that exceed within-class distances

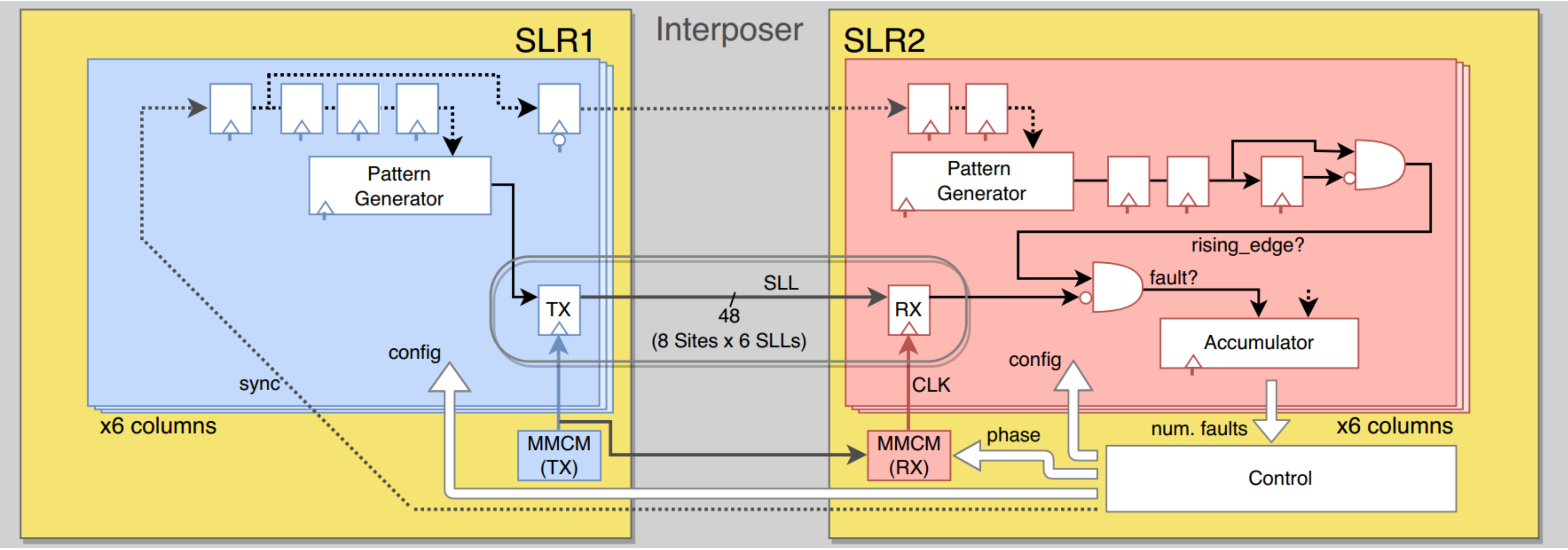
- Die swapping detectable because drivers contribute to entropy and delay measurements exist only on RX die

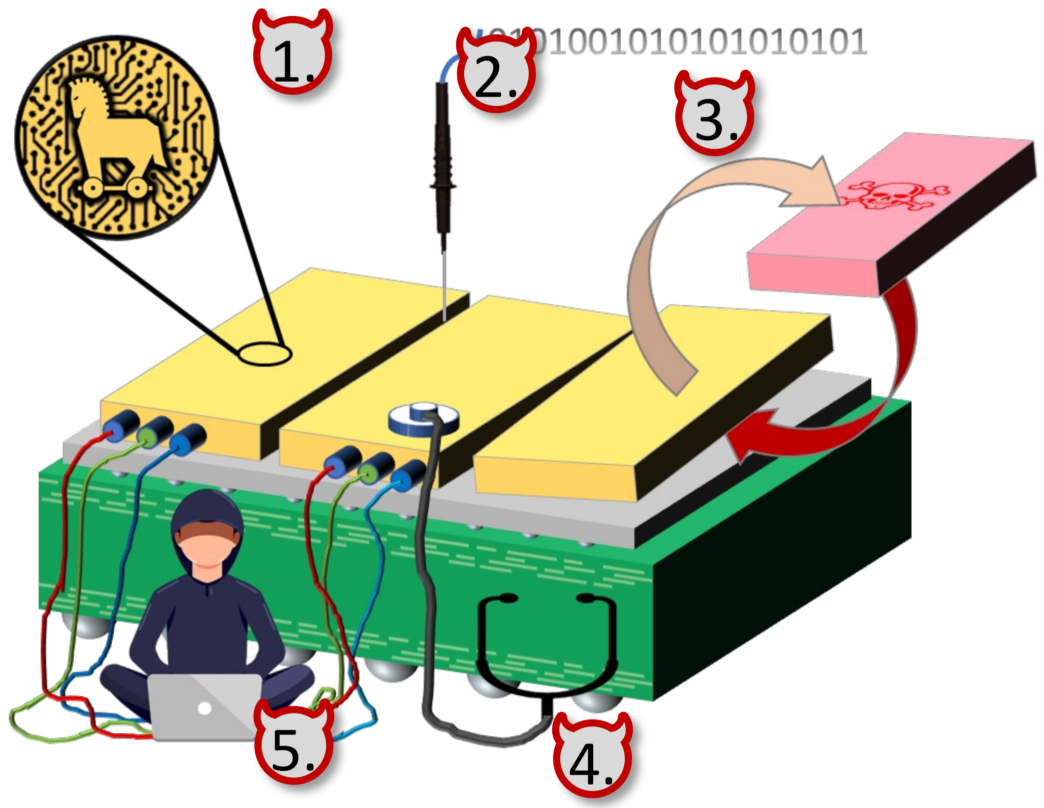


Conclusion

- Presented a security primitive to extract delay fingerprints from connections between chiplets
- Prototyped using Xilinx Ultrascale+ FPGAs locally and across a population on AWS EC2 F1
- Performed analysis across a variety of design manipulations to identify the specific sources of entropy in the system

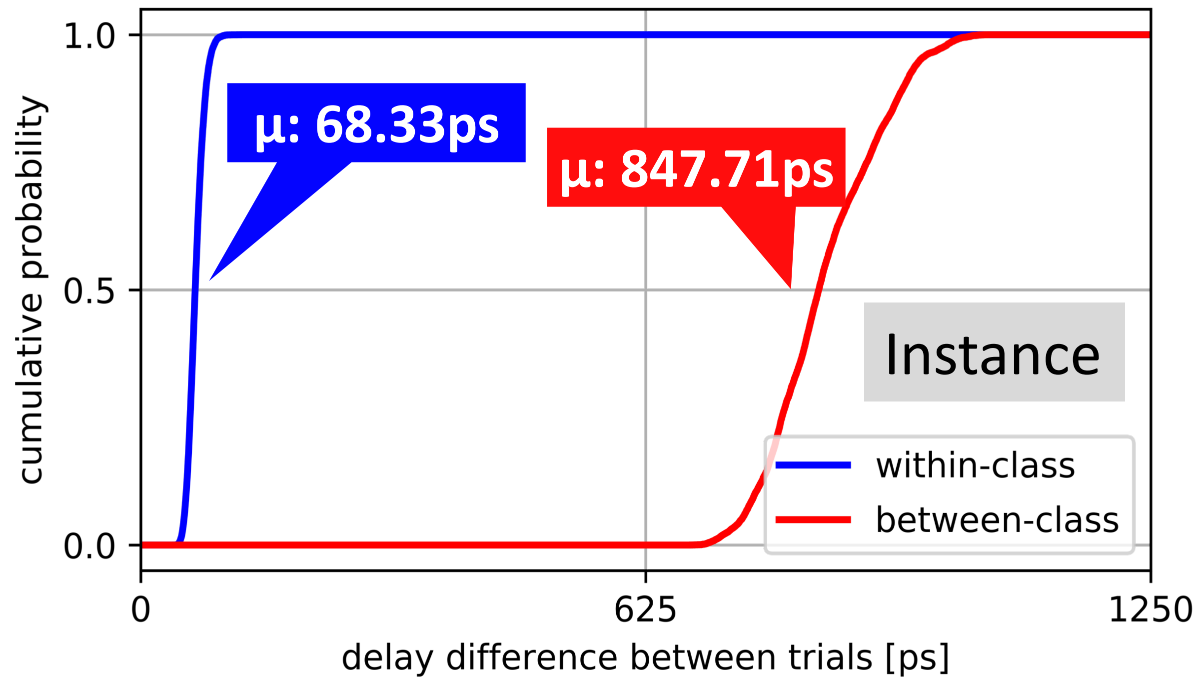




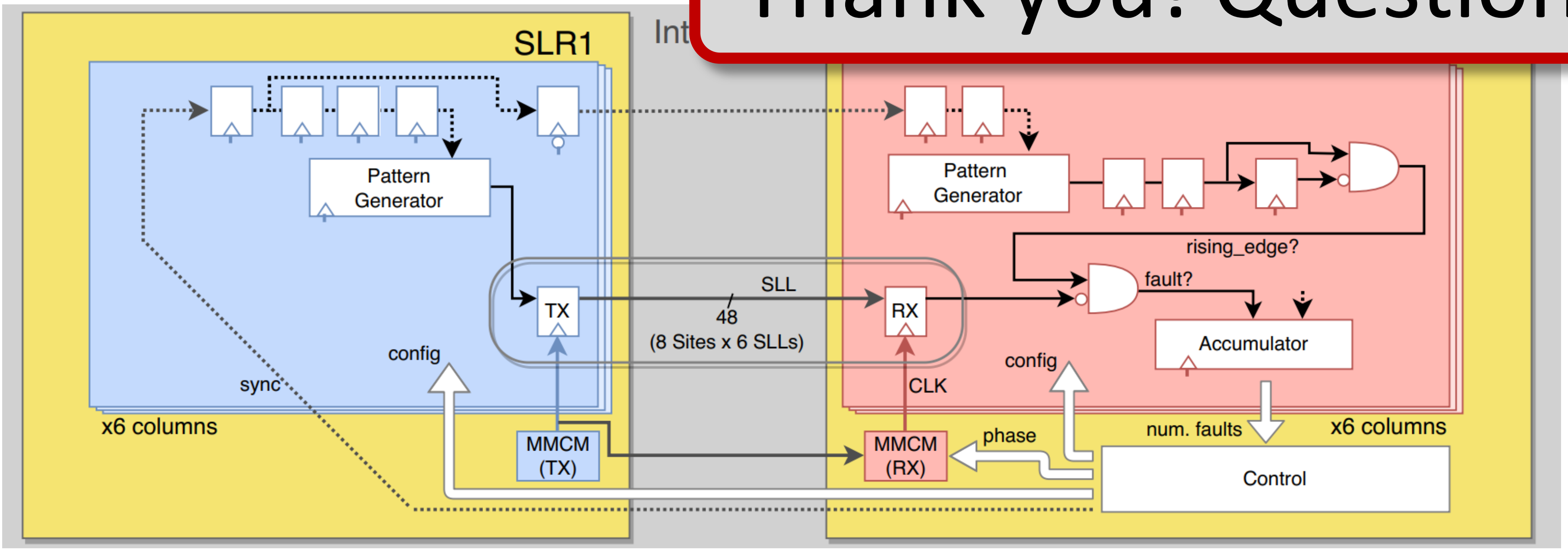


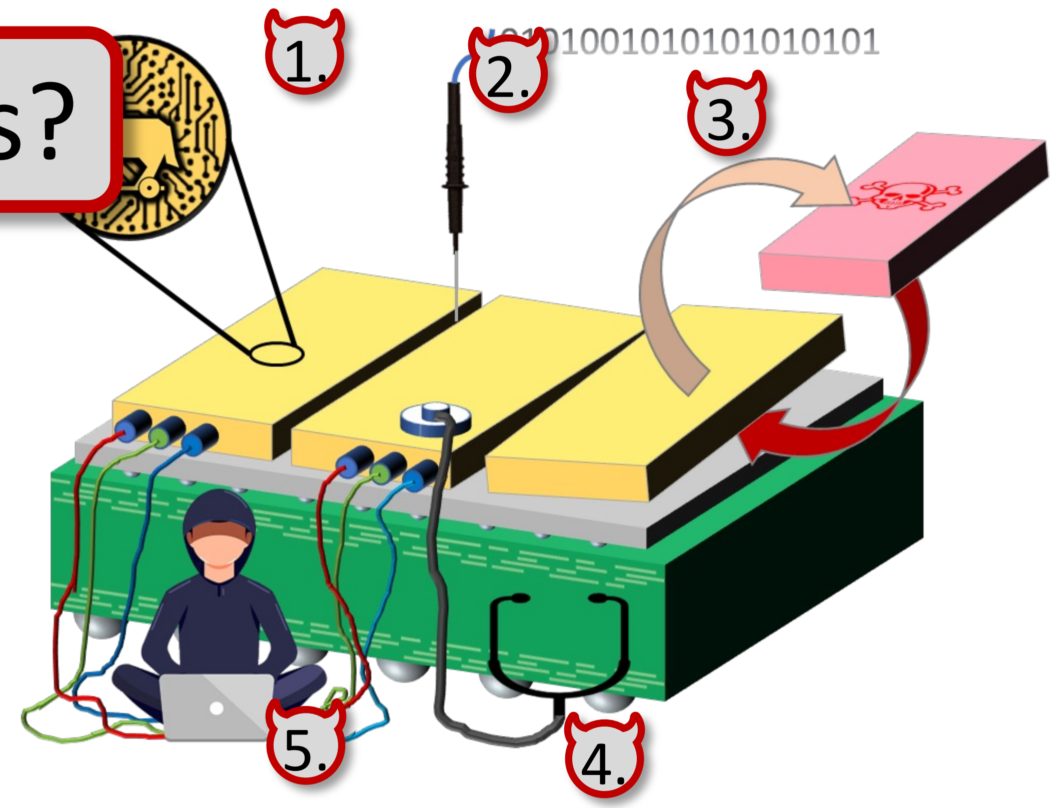
Conclusion

- Presented a security primitive to extract delay fingerprints from connections between chiplets
- Prototyped using Xilinx Ultrascale+ FPGAs locally and across a population on AWS EC2 F1
- Performed analysis across a variety of design manipulations to identify the specific sources of entropy in the system



Thank you! Questions?

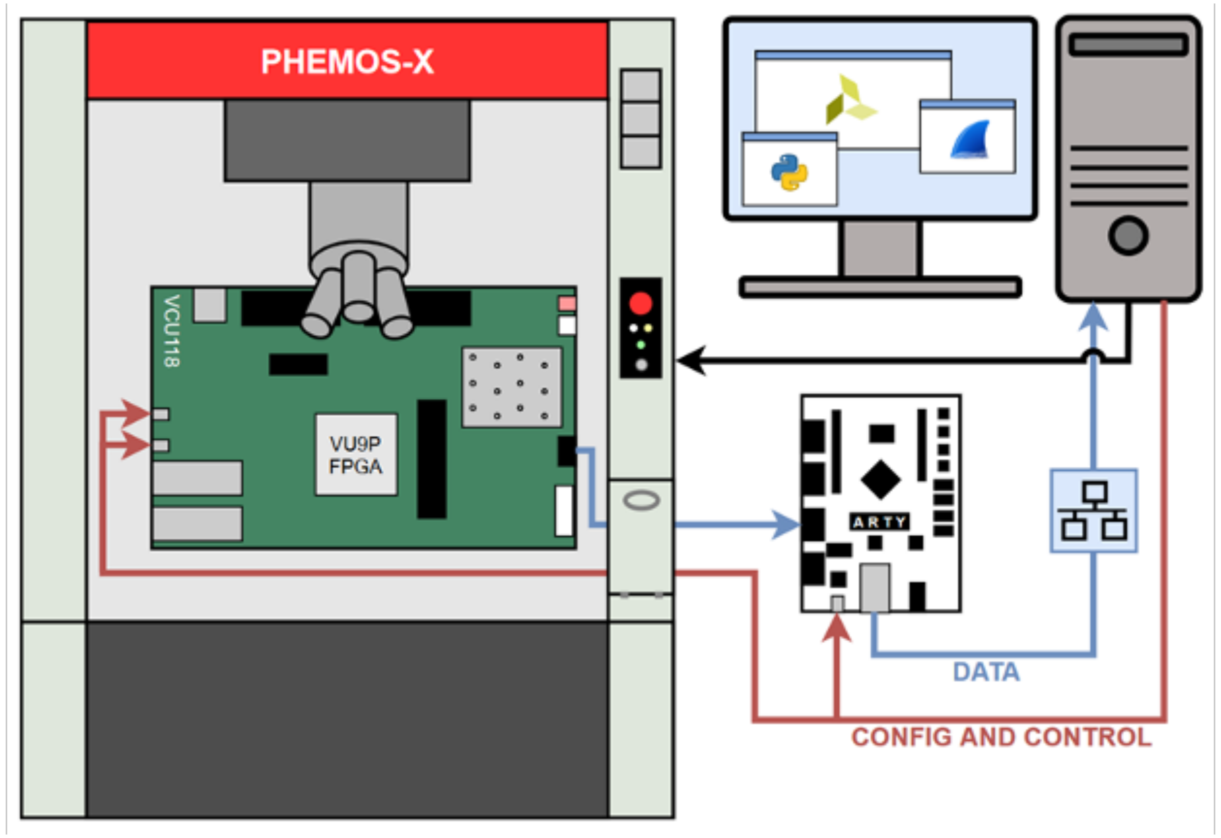




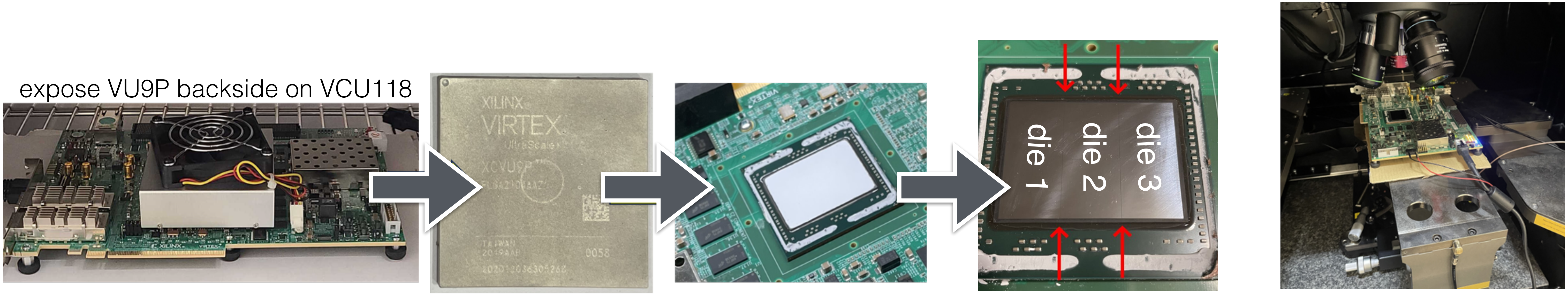
Backup - probing drivers

Testing against Optical Probing

- Hamamatsu PHEMOS-X in Prof. Tajik's lab at WPI
  - Photon emission (activity)
  - Electro-optical frequency mapping (activity+bandpass)
  - Electro-optical probing (waveforms)



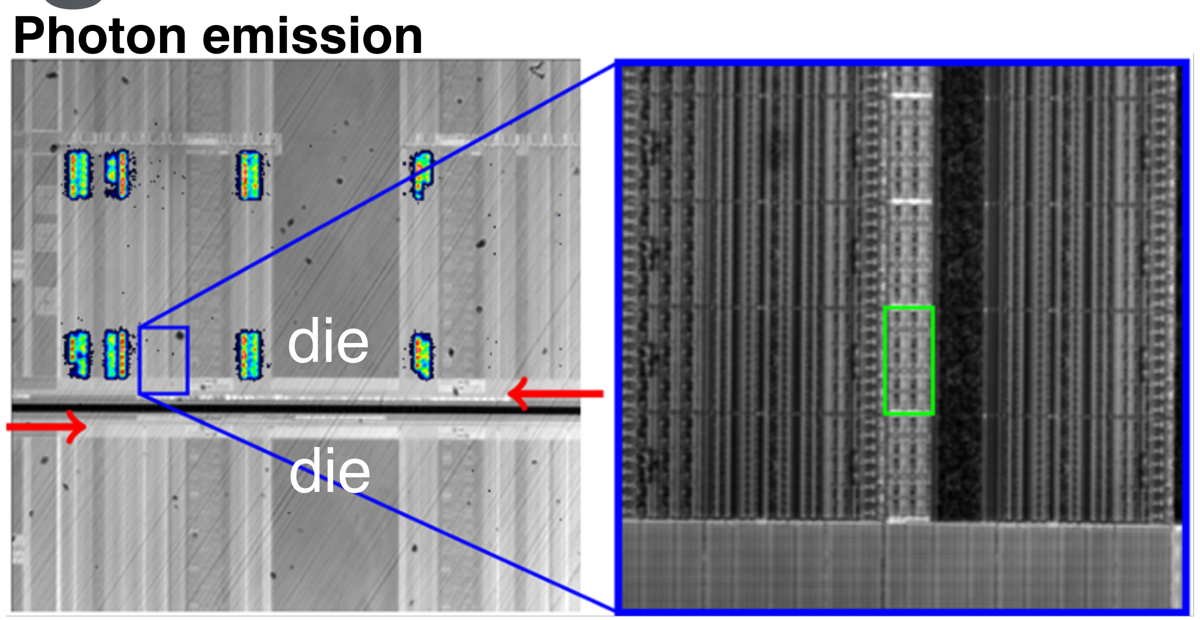
expose VU9P backside on VCU118

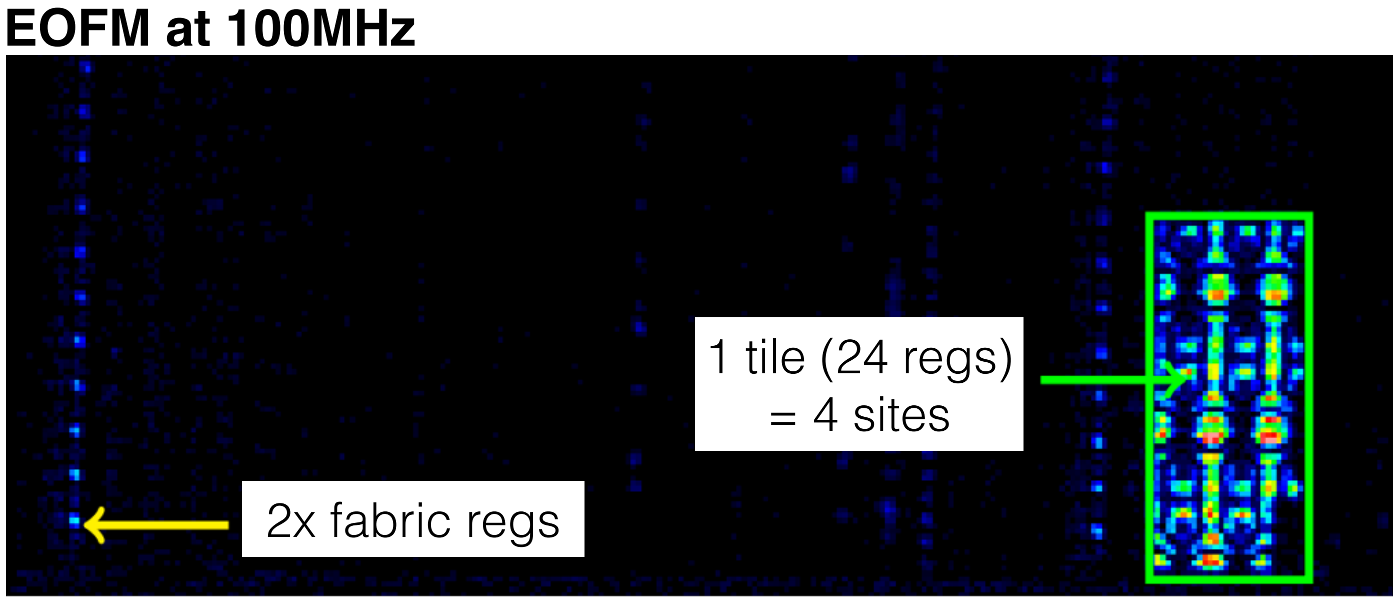


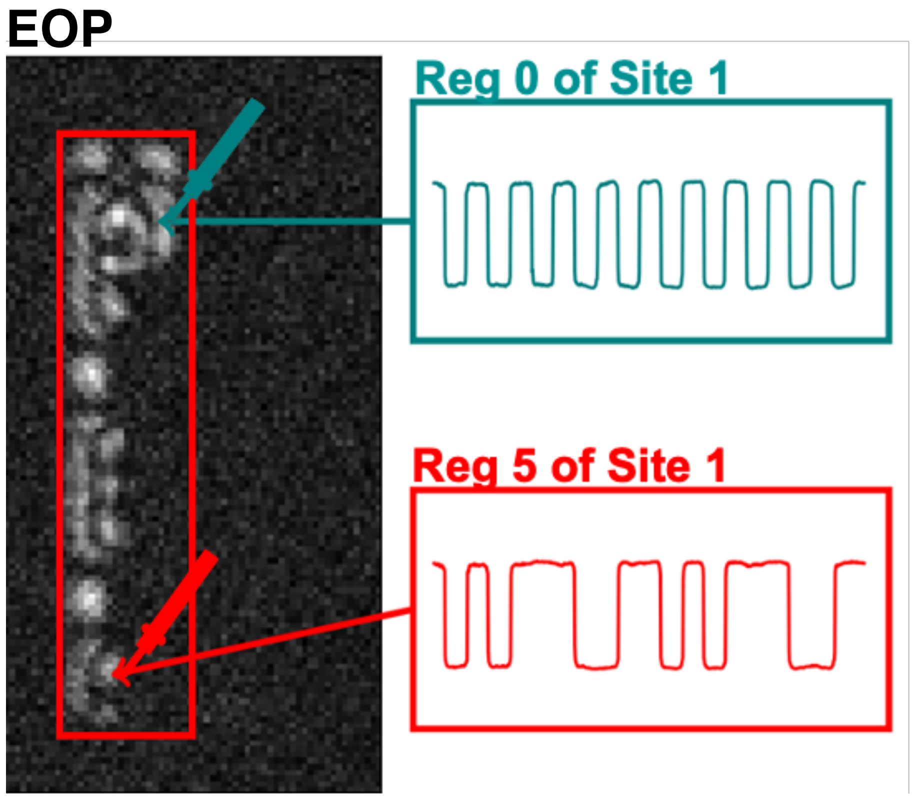
Chiplets vs Backside Probing

- Die-to-die (mm-length) wires driven by wide transistors that should be vulnerable

1. **Photon emission**: find target area
2. **EOFM**: map out drivers for each wires
3. **EOP**: extract waveforms (with repetition)

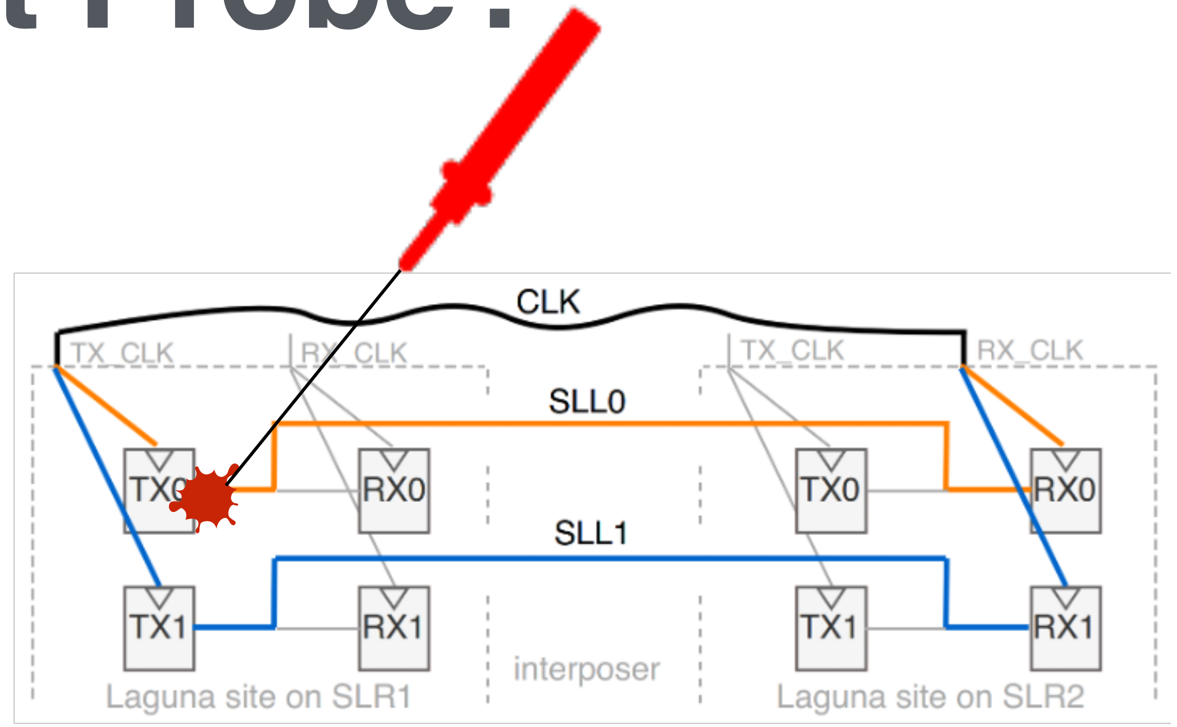


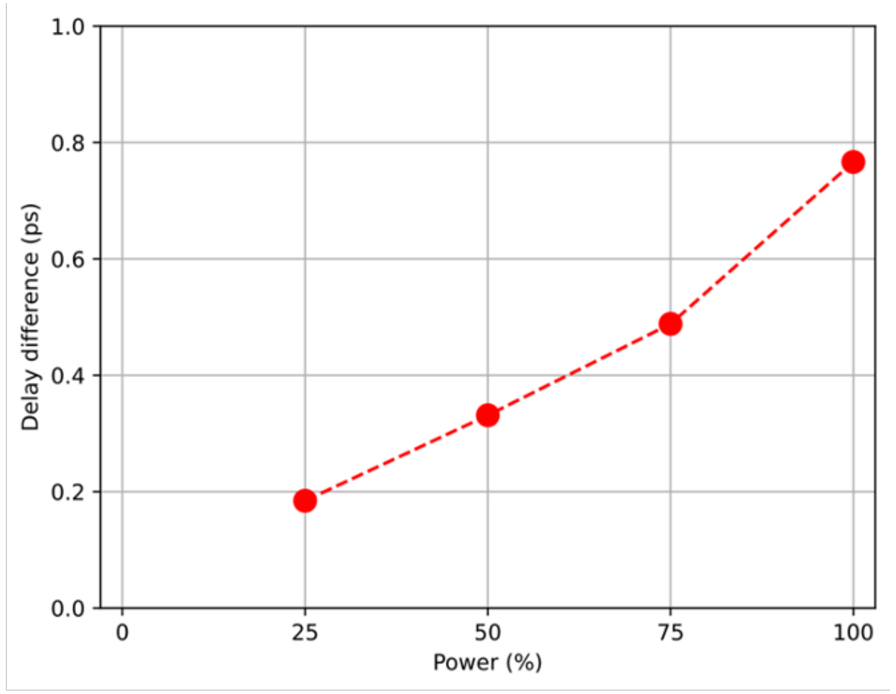


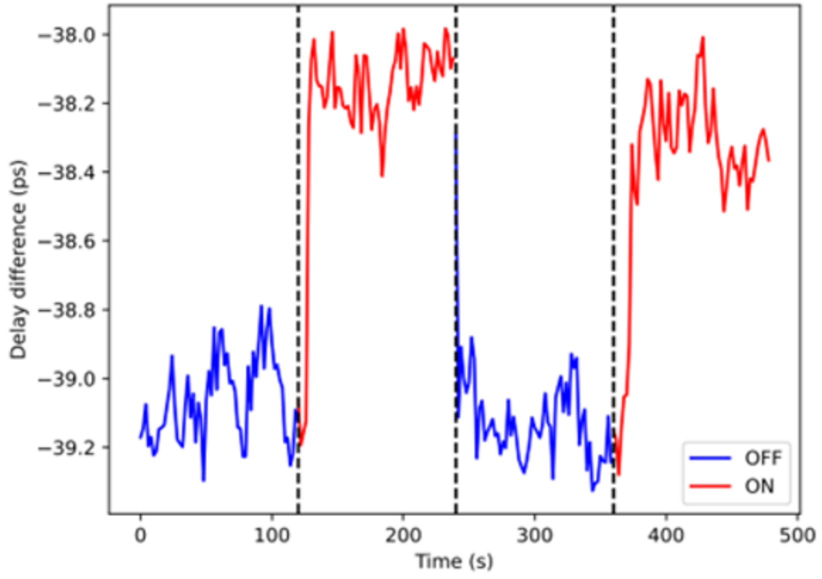


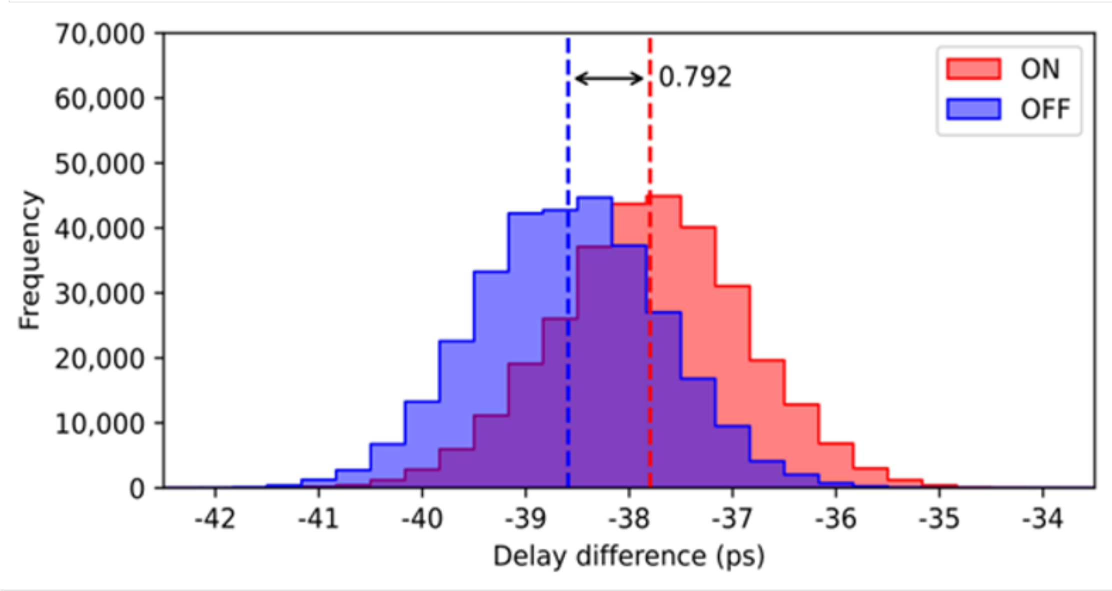
Use PUF as sensor to Detect Probe?

- Laser shifts delay by 0.8ps at full power
- Change of ~0.1% delay cannot easily be distinguished from noise
- Better ways to protect?







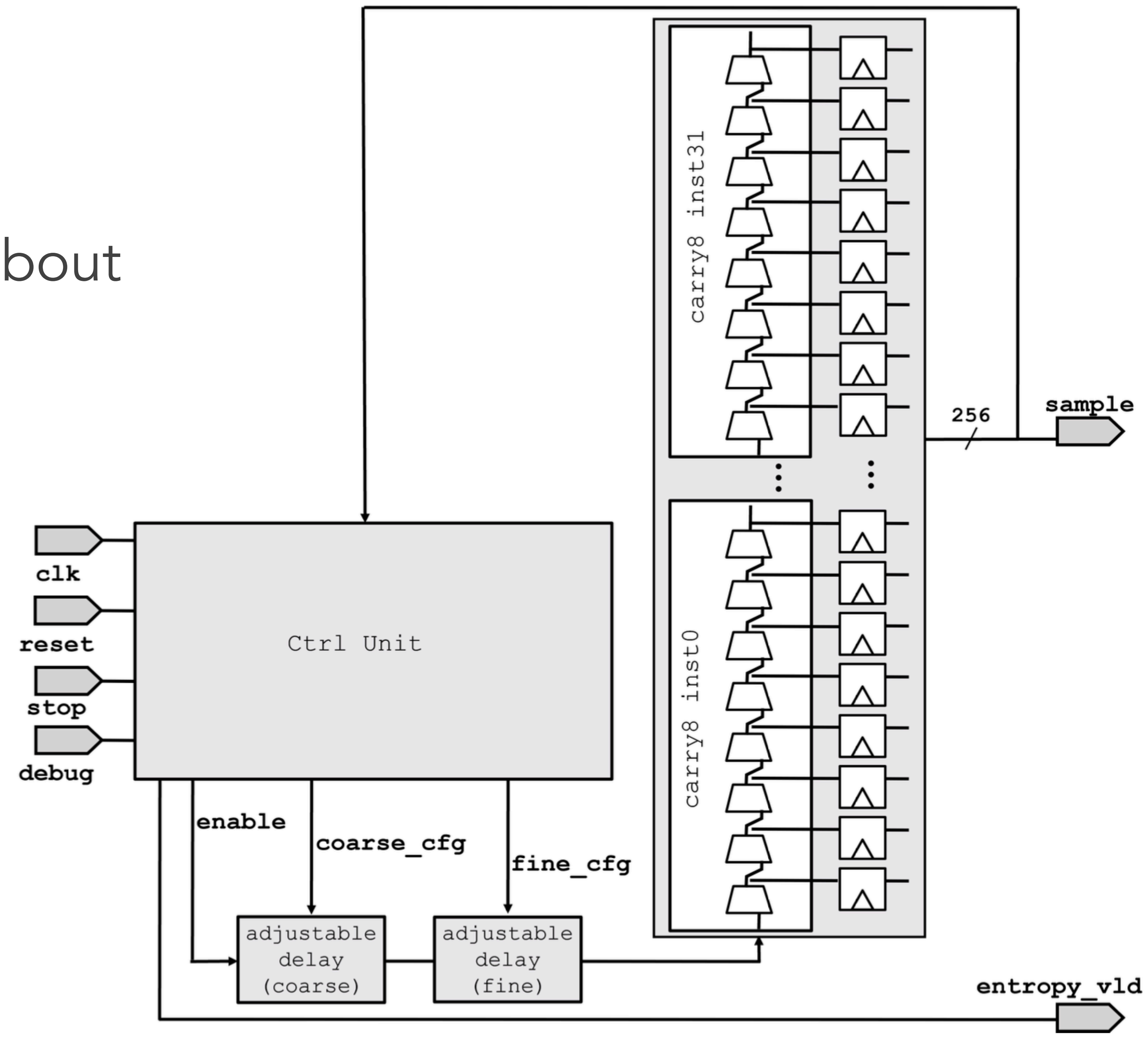


Backup - Sensing jitter for TRNG

Jitter-Based TRNG on AWS EC2 F1 (VU9P)

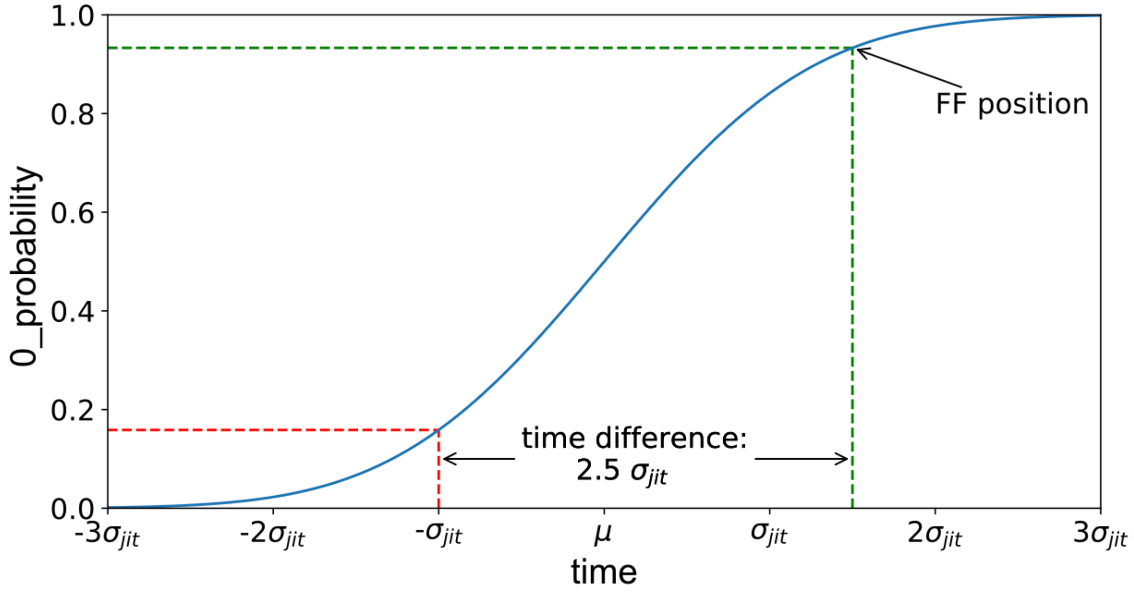
- TDC output fluctuates from jitter
- Physical source for TRNG
- Need stochastic model to reason about worst-case min-entropy

- Don't rely on Xilinx timing models as ground truth
  - Timing models are conservative, not accurate
  - Instance-specific variation



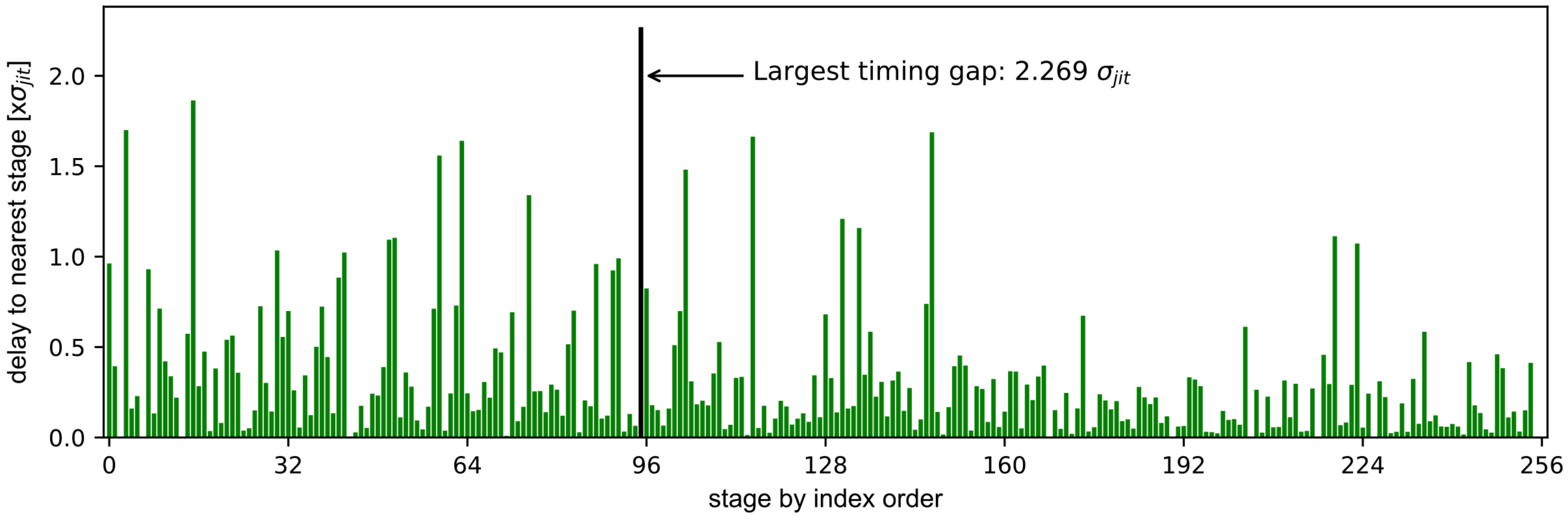
Inferring Timing Model from Samples

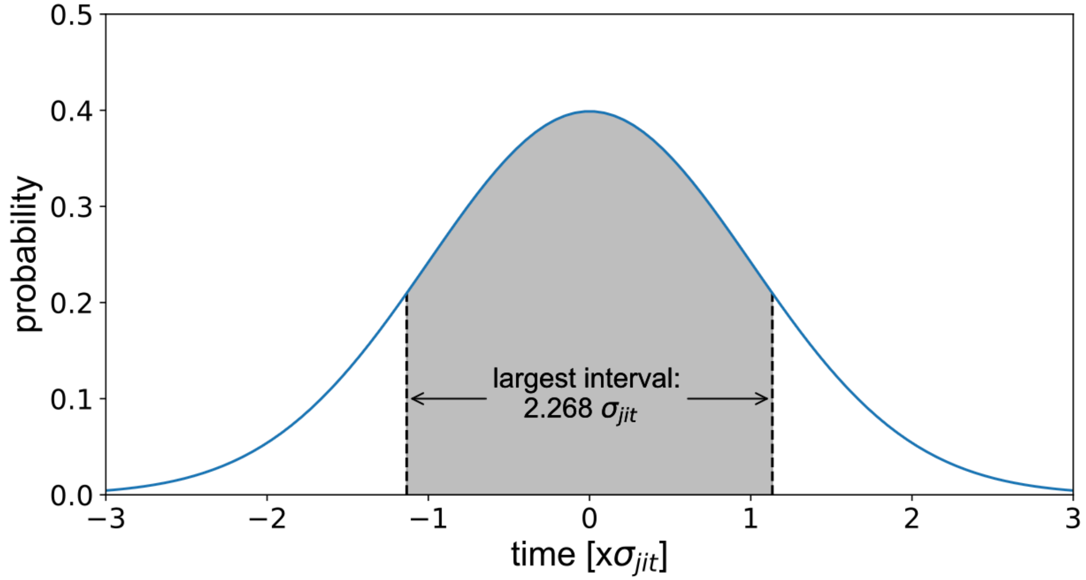
- Assume jitter is IID normally distributed
- Define all delays in terms of σ_{jit}
- Solve set of equations for relative tap delays from dataset of measured Hamming weights
- Largest gap between tap delays causes worst-case min-entropy of TRNG source



$$T_i - T_j = \left(\Phi^{-1}(\widehat{P}_i) - \Phi^{-1}(\widehat{P}_j)\right)\sigma_{jit}$$

$$AT = B\sigma_{jit}$$

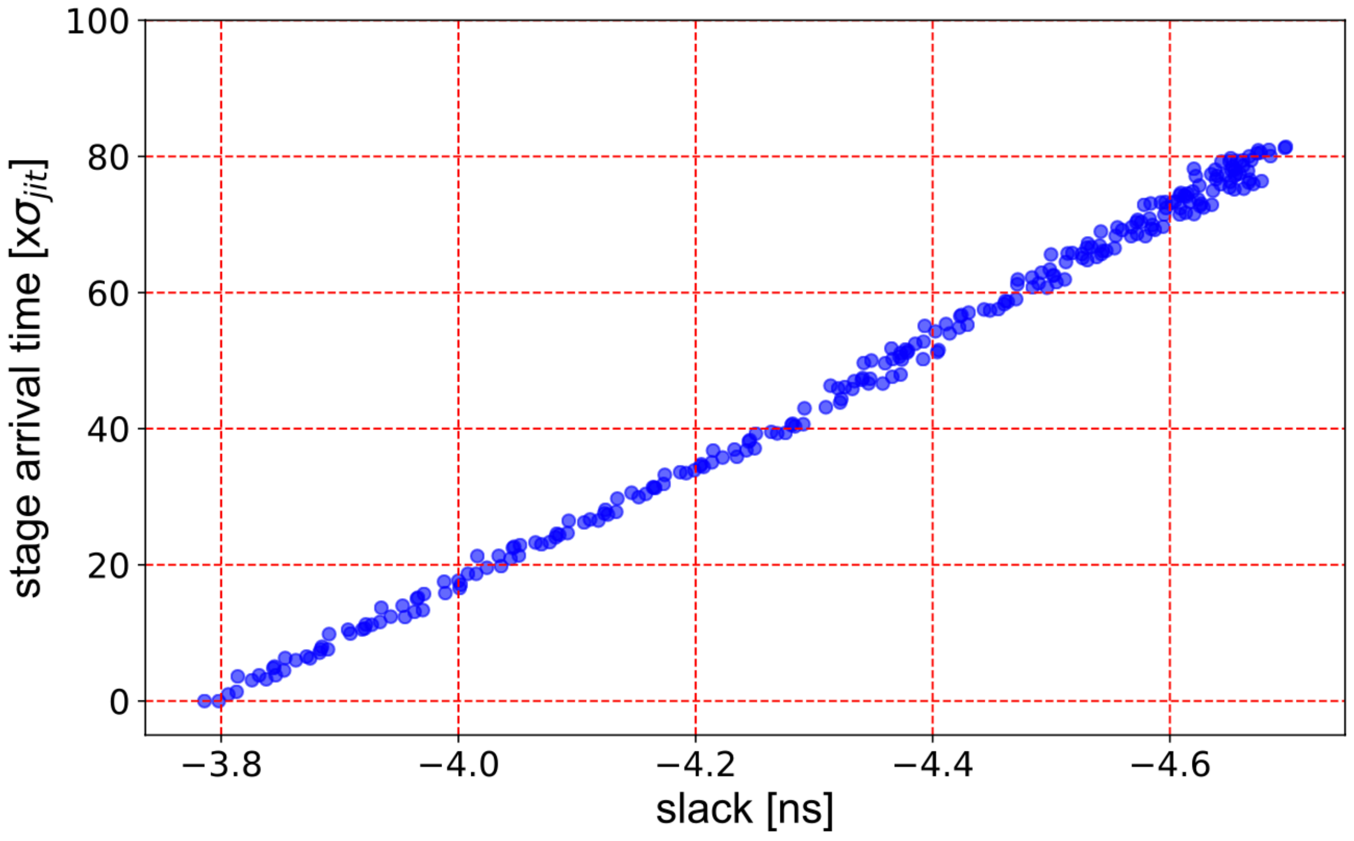




Findings from Inferred Timing Model

- Inferred timing aligns to known details of FPGA
- Supports correctness of analysis/model

Agreement with Xilinx STA



Irregular delays across clock leafs