



# WPI

## Tools and Methods for Pre-silicon Analysis of Secure Hardware

Patrick Schaumont  
Electrical and Computer Engineering

# Vernam Lab @WPI

Koksal Mus

Berk Sunar

Fatemeh Ganji

Shahin Tajik

Yarkin Doroz

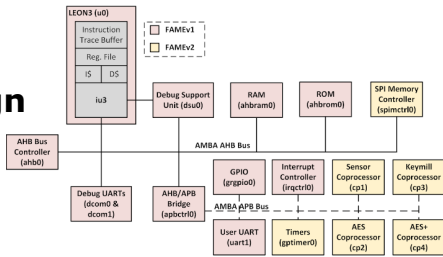
Patrick Schaumont

and our wicked smaht students

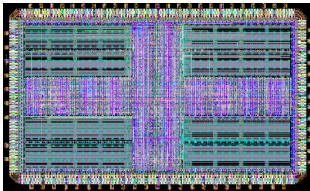


# WPI's Hardware Security Sampler

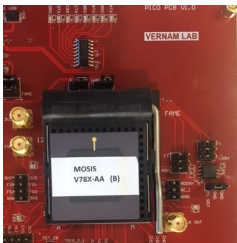
## Design



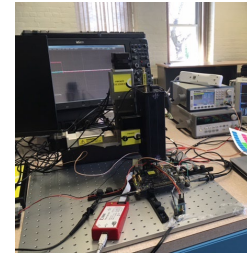
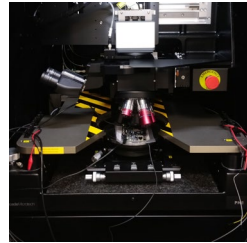
## Chip



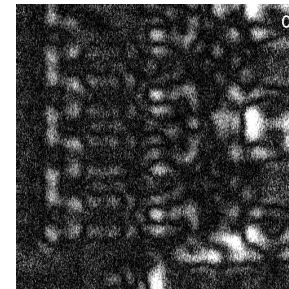
## PCB



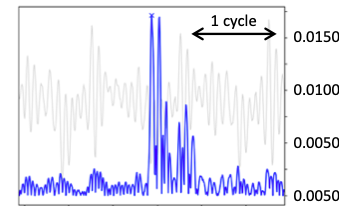
## Security Testing



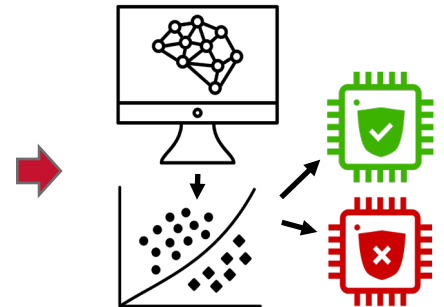
## Physical Data Acquisition



## Correlation on Power Measurement



## Security Assessment using AI



# Outline

---

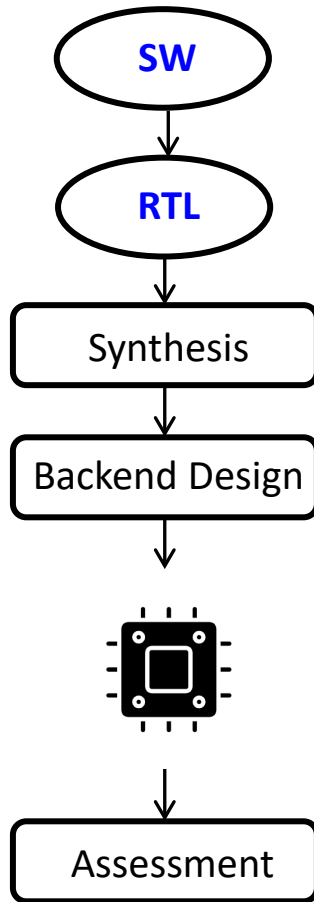
- Pre-silicon Tooling
- Background on Gate-level Power Modeling
- Architecture Correlation Analysis
- ACA Applications
- What's next?

## Acknowledgements:

Students – Pantea Kiaei, Zhenyuan Liu, Yuan Yao, Richa Singh,  
Ramazan Kaan Eren, Dillibabu Shanmugam  
Partners – Riscure, Intrinsix/CEVA, Purdue  
Sponsors – DARPA, National Science Foundation

# The rise of pre-silicon tooling

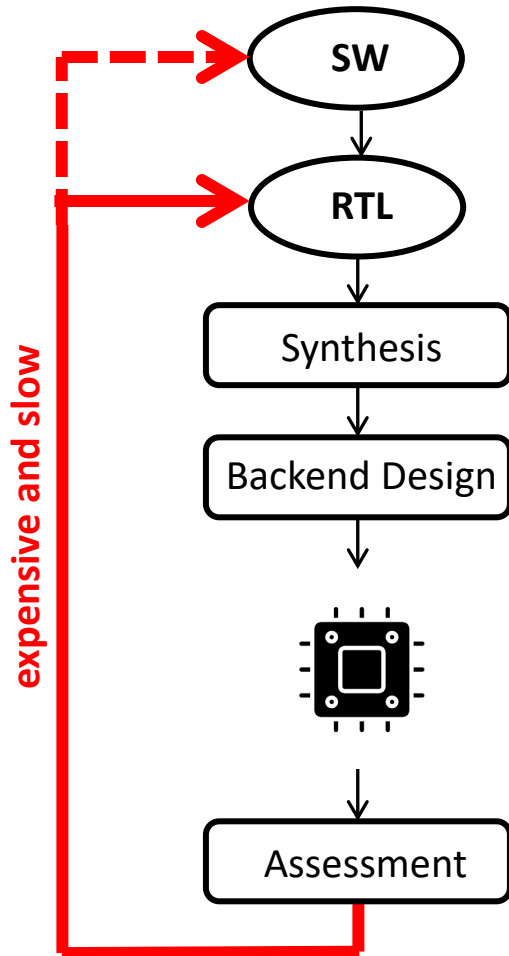
---



*How to design SW/firmware, RTL such that their implementation will have known side-channel leakage and/or fault response properties?*

- EM, Power-based Side-channel Leakage
- Fault Response

# Prototyping essential but slow

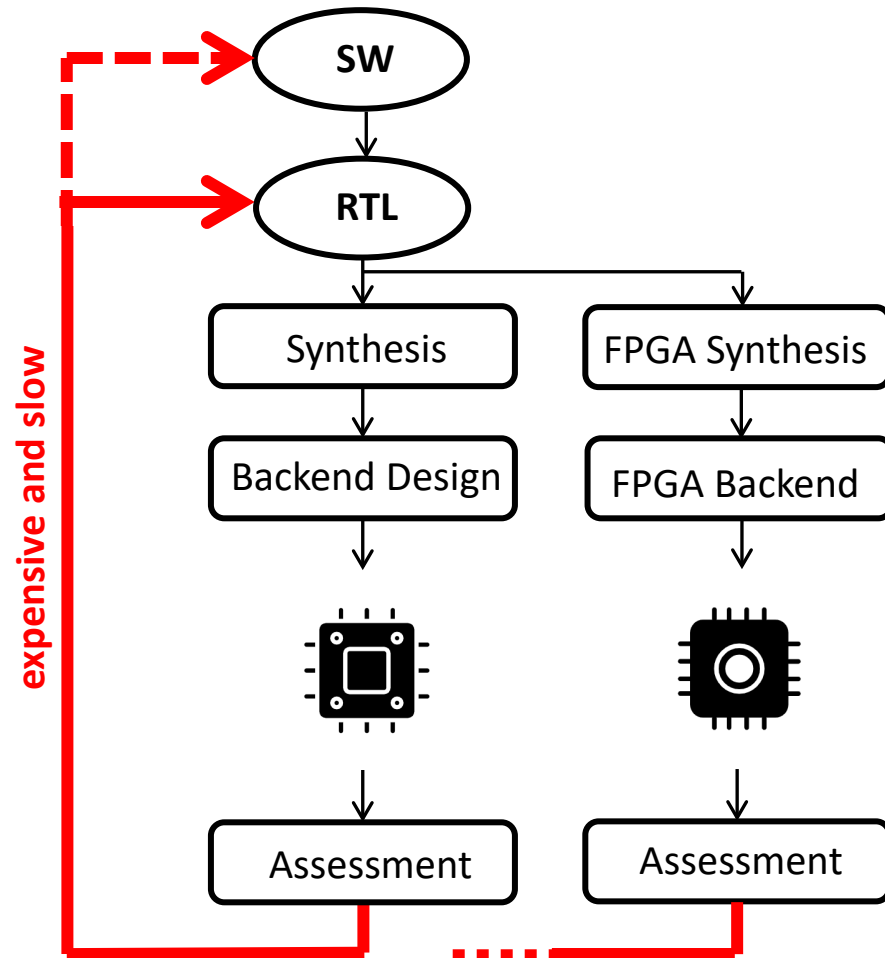


**Fact:** Secure Design requires Detailed Assessment of Side-channel Leakage, Fault Response

1. No 'correct-by-construction' rules exist
2. It's 'turtles all the way down';  
Secure SW requires secure HW requires  
secure cells requires secure layout  
requires ..

- EM, Power-based Side-channel Leakage
- Fault Response

# .. and FPGA are not ASICs

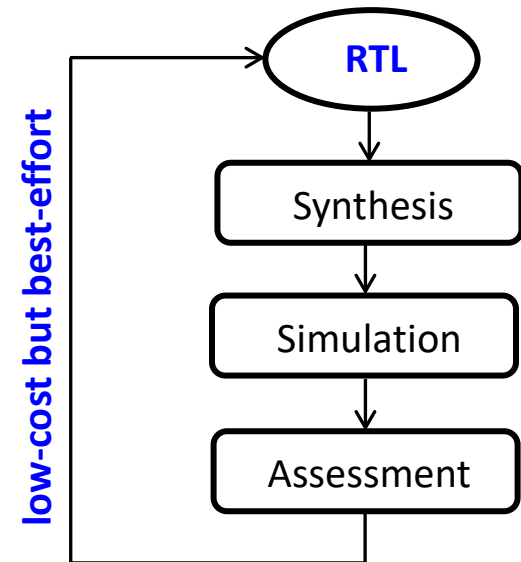
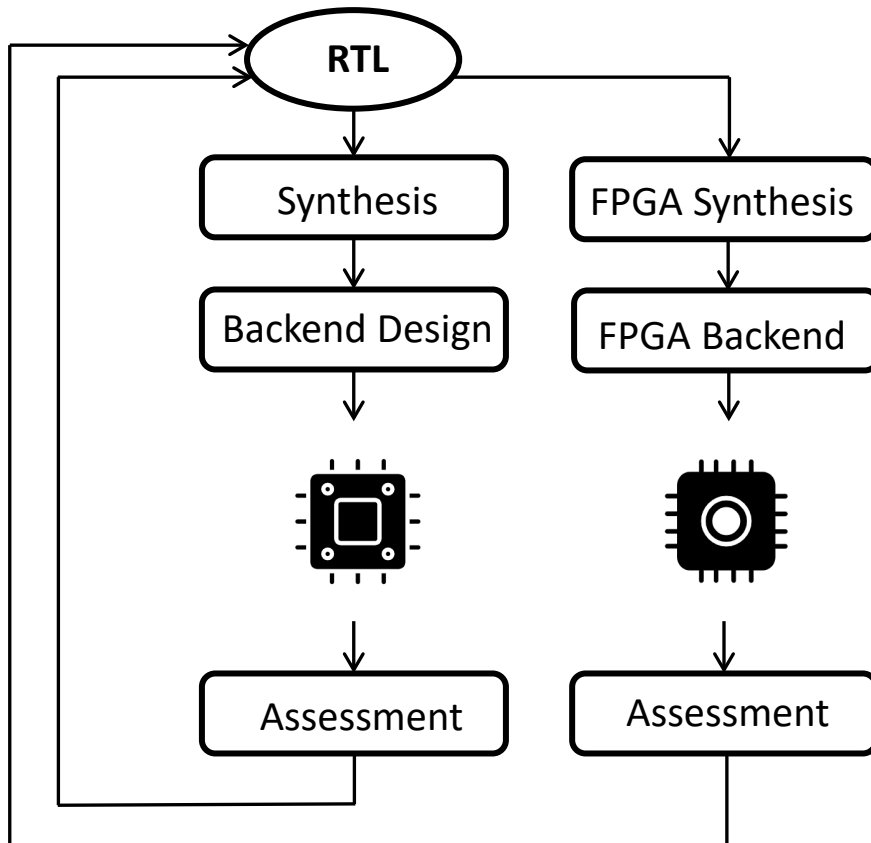


**Fact:** Secure Design requires Detailed Assessment of Side-channel Leakage, Fault Response

1. No 'correct-by-construction' rules exist
2. Turtles all the way down
3. FPGA results not portable

# Enter tools for Pre-silicon ..

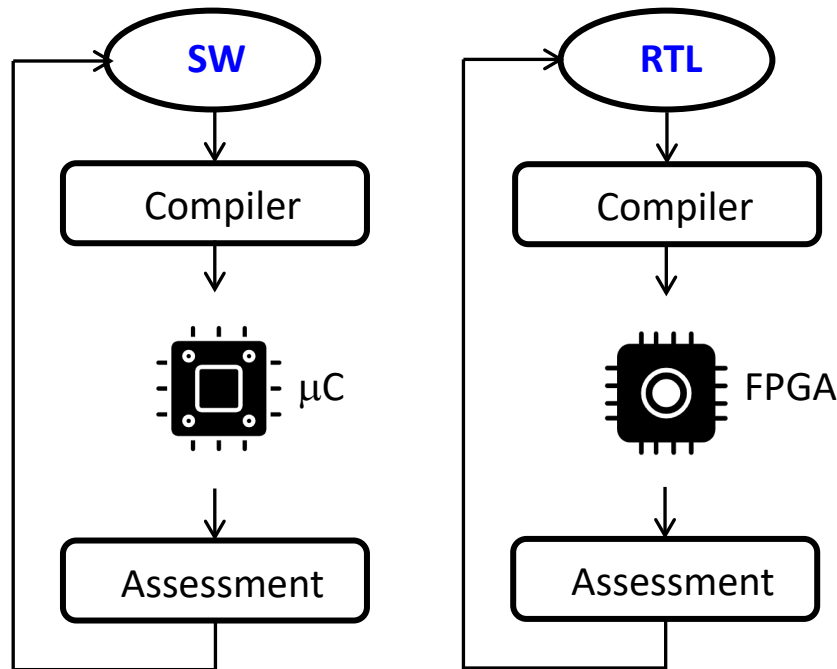
## Pre-silicon Assessment



- Use modeling and simulation to estimate implementation properties
- Common method for every IC performance factor: functionality, timing, power, area, yield, ...

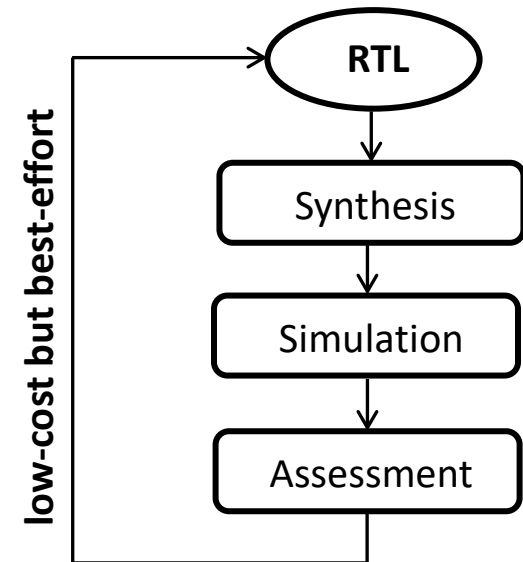
# .. and Post-silicon Assessment

## Post-silicon Assessment



- Post-silicon Assessment is meaningful for black-box scenario

## Pre-silicon Assessment



- Use modeling and simulation to estimate implementation properties
- Common method for every IC performance factor: functionality, timing, power, area, yield, ...

# ileanabuhan.github.io/Tools

ileana Buhan Teaching Tools Who am I? Publications

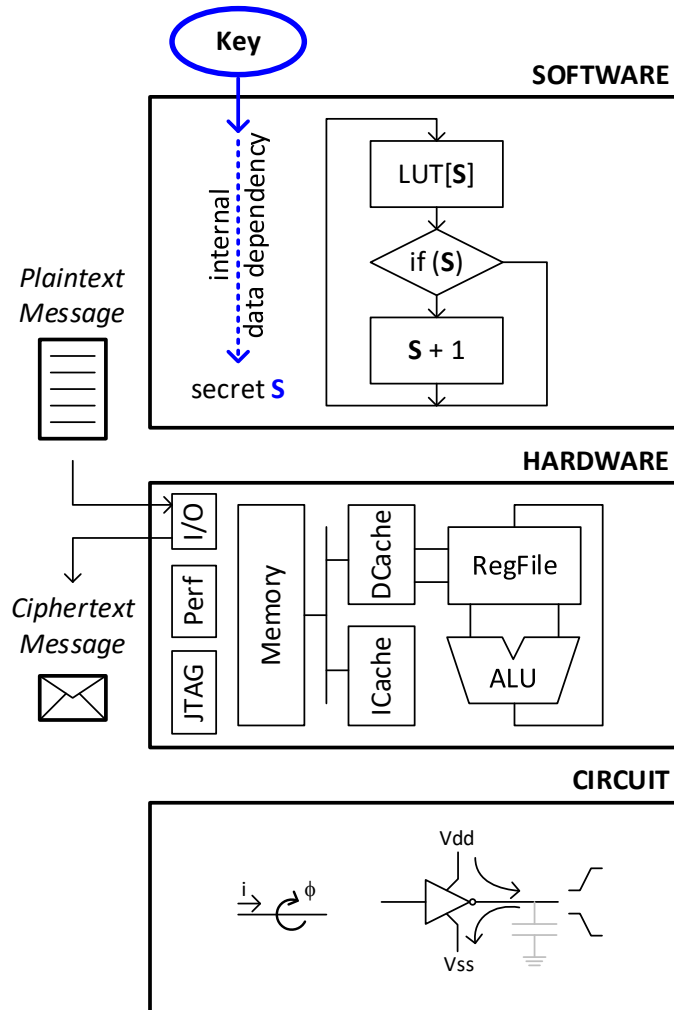
## Tools

A quick overview of available tools for aid in securing cryptographic implementations against *physical side-channel attacks*. A detailed analysis of the tools listed below can be found in our [paper SoK: Design Tools for Side-Channel-Aware Implementations](#) authored with *Lejla Batina, Patrick Schaumont and Yuval Yarom*, which will appear at ASIACCS 2022 [video](#).

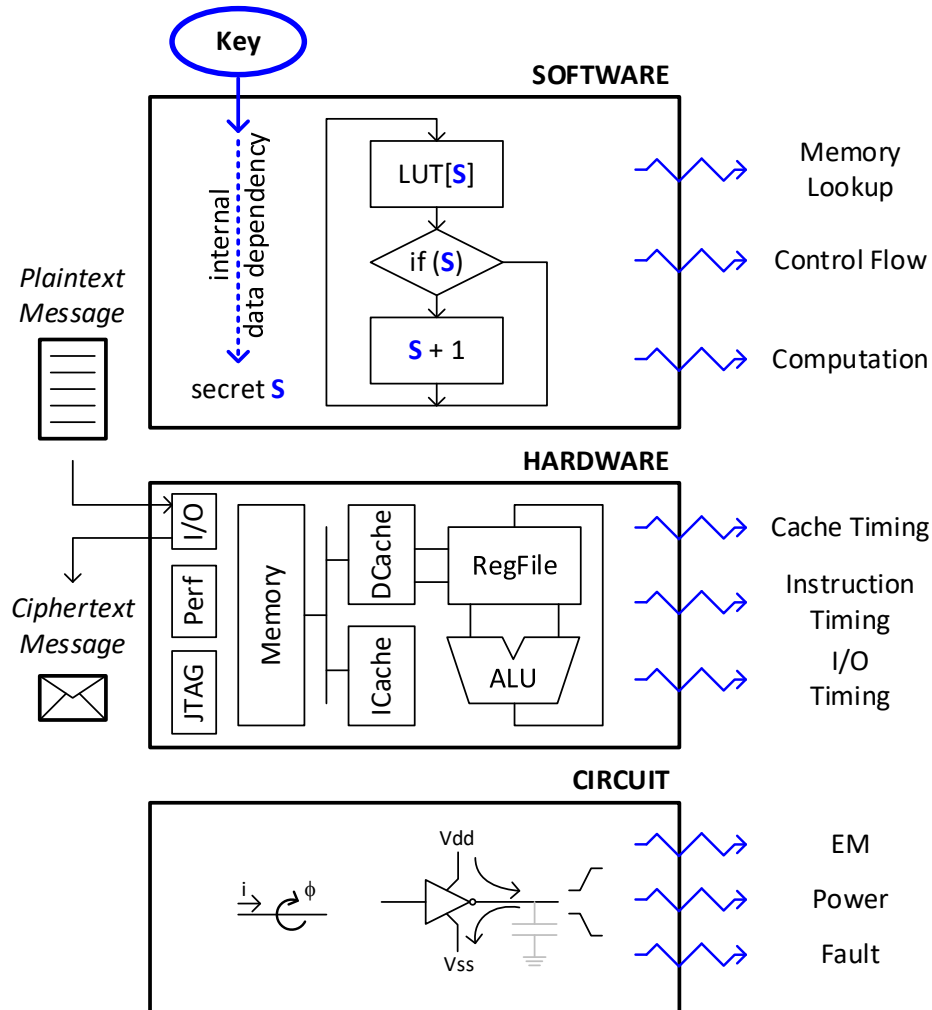
**Post-silicon side-channel leakage emulators (power)** The list is ordered by publication year.

Name	Year	Leakage Model	Target	Function
<a href="#">ARMISTICE</a>	2022	tba	ARM Cortex-M3	tba
<a href="#">ABBY</a>	2022	gray	ARM Cortex-M0	Detect
<a href="#">ROSITA++</a> , <a href="#">repo</a>	2021	gray	ARM Cortex-M0	Mitigate (high order)
<a href="#">ROSITA</a> , <a href="#">repo</a>	2021	gray	ARM Cortex-M0	Mitigate (1-order)

# Side-channel Leakage



# Side-channel Leakage

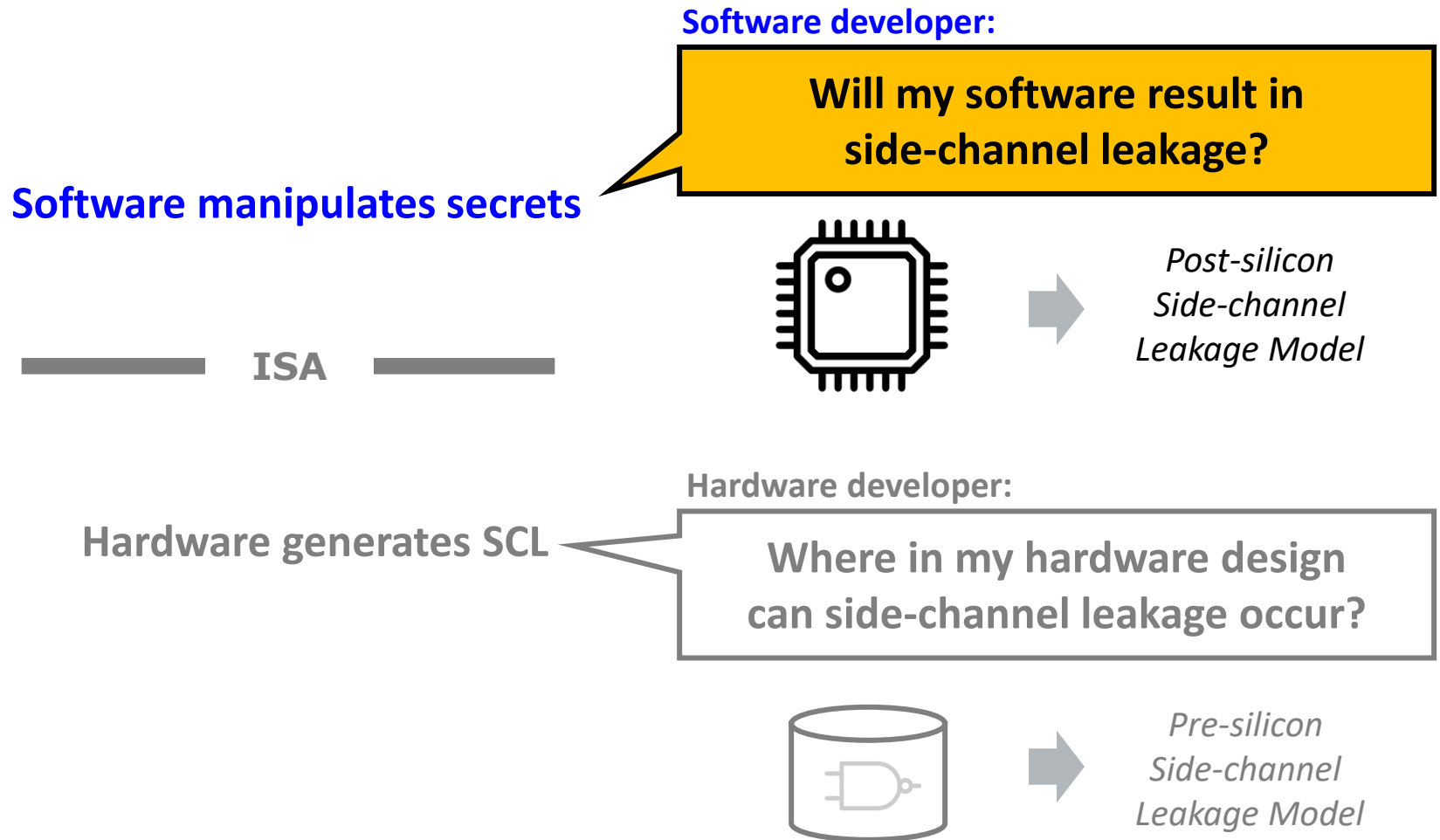


Software manipulates secrets

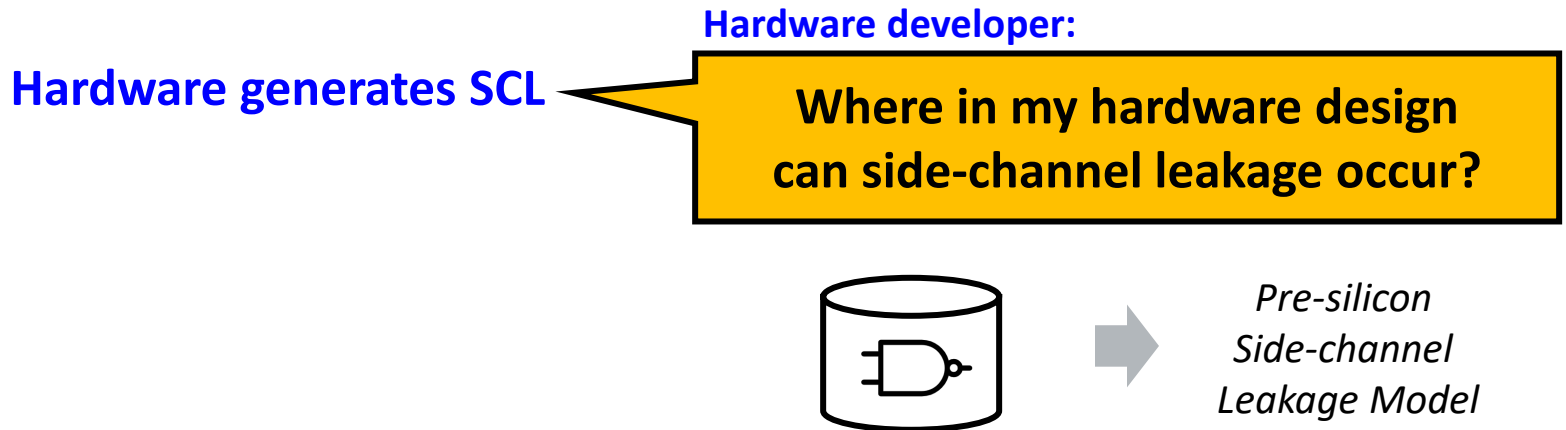
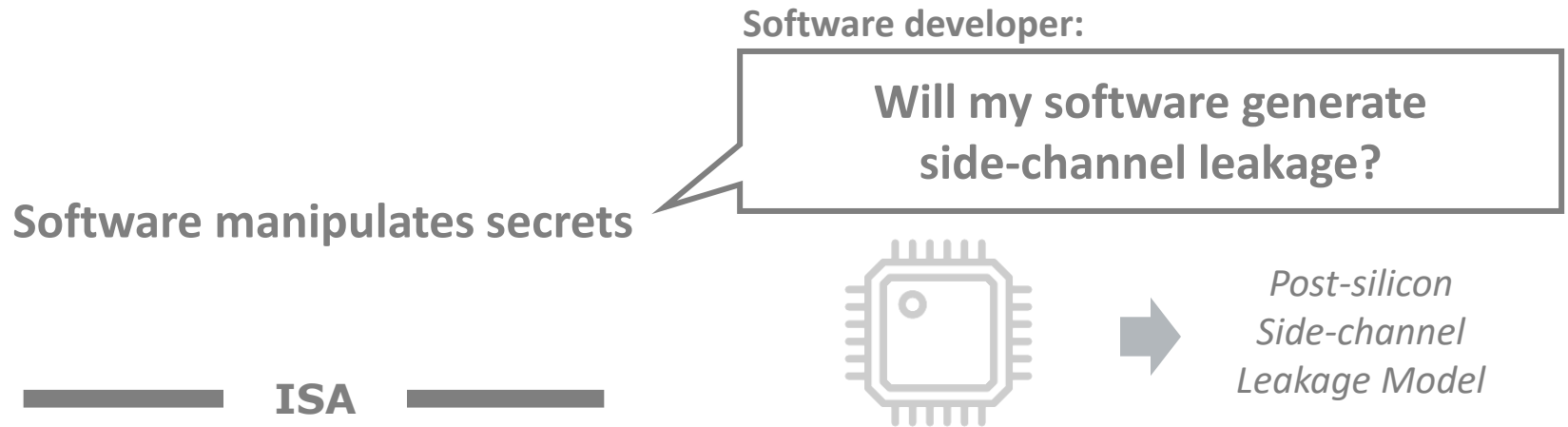
ISA

Hardware generates SCL

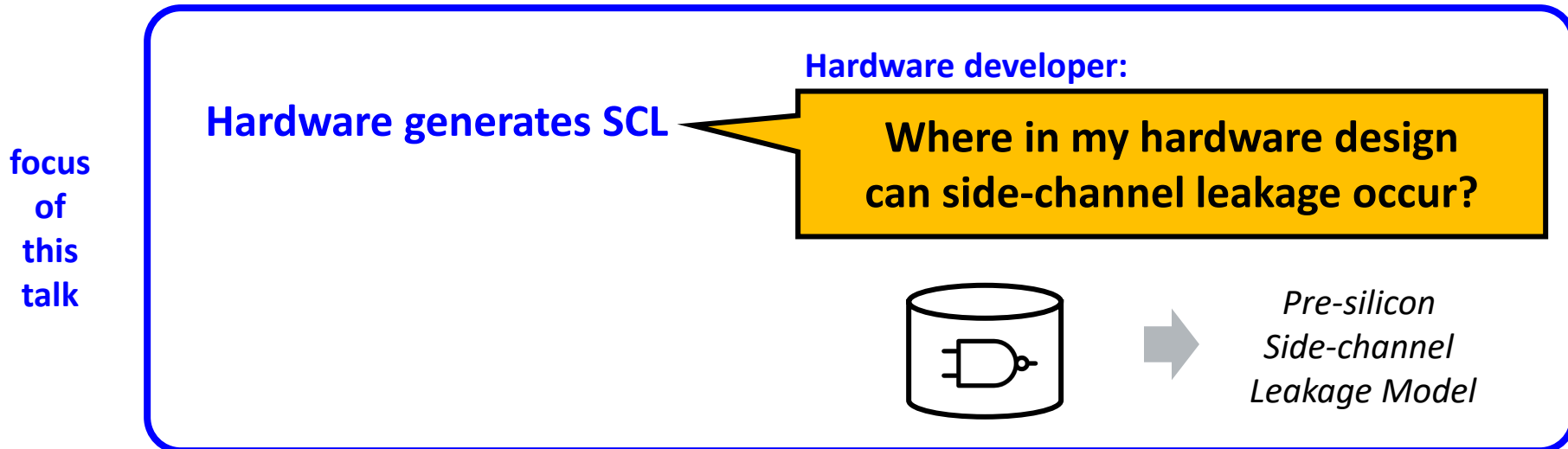
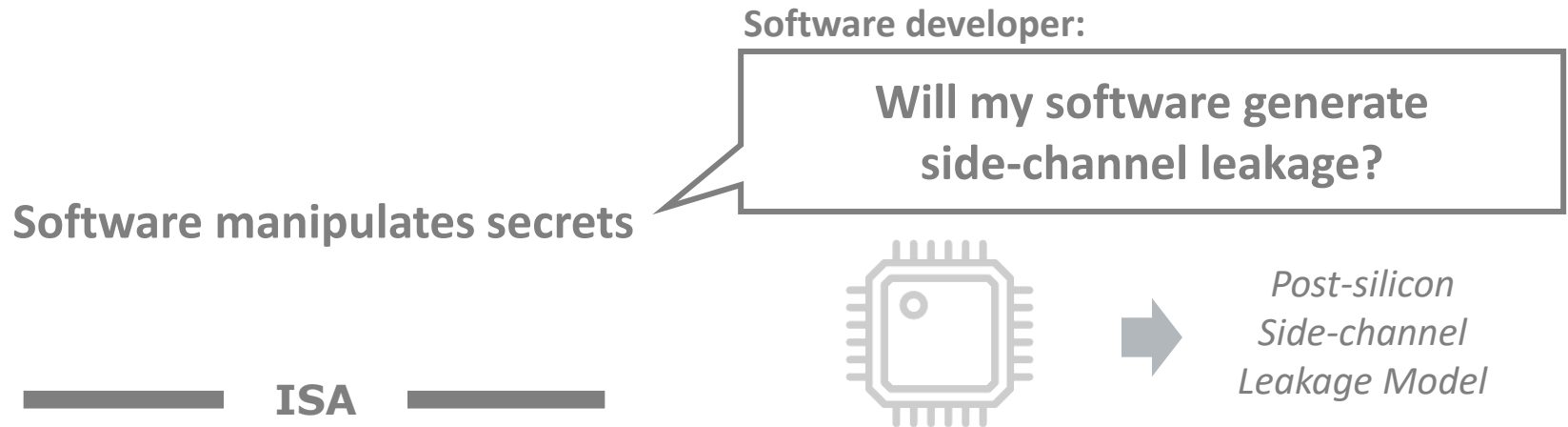
# Cases for SCL Modeling



# Cases for SCL Modeling



# Cases for SCL Modeling



# Hardware SCL Modeling: Choosing the Abstraction Level

---

- The abstraction level decides on
  - Timing granularity: Instructions, Cycles, Events, ...
  - Data granularity: Bytes, Bits, Voltages, ...
- A given abstraction level may demonstrate presence of SCL
  - E.g., HDL event-driven simulation can demonstrate glitches
- No abstraction level can guarantee absence of *any* SCL
  - E.g., HDL event-driven simulation cannot ensure absence of EM
  - False negatives are bound to happen

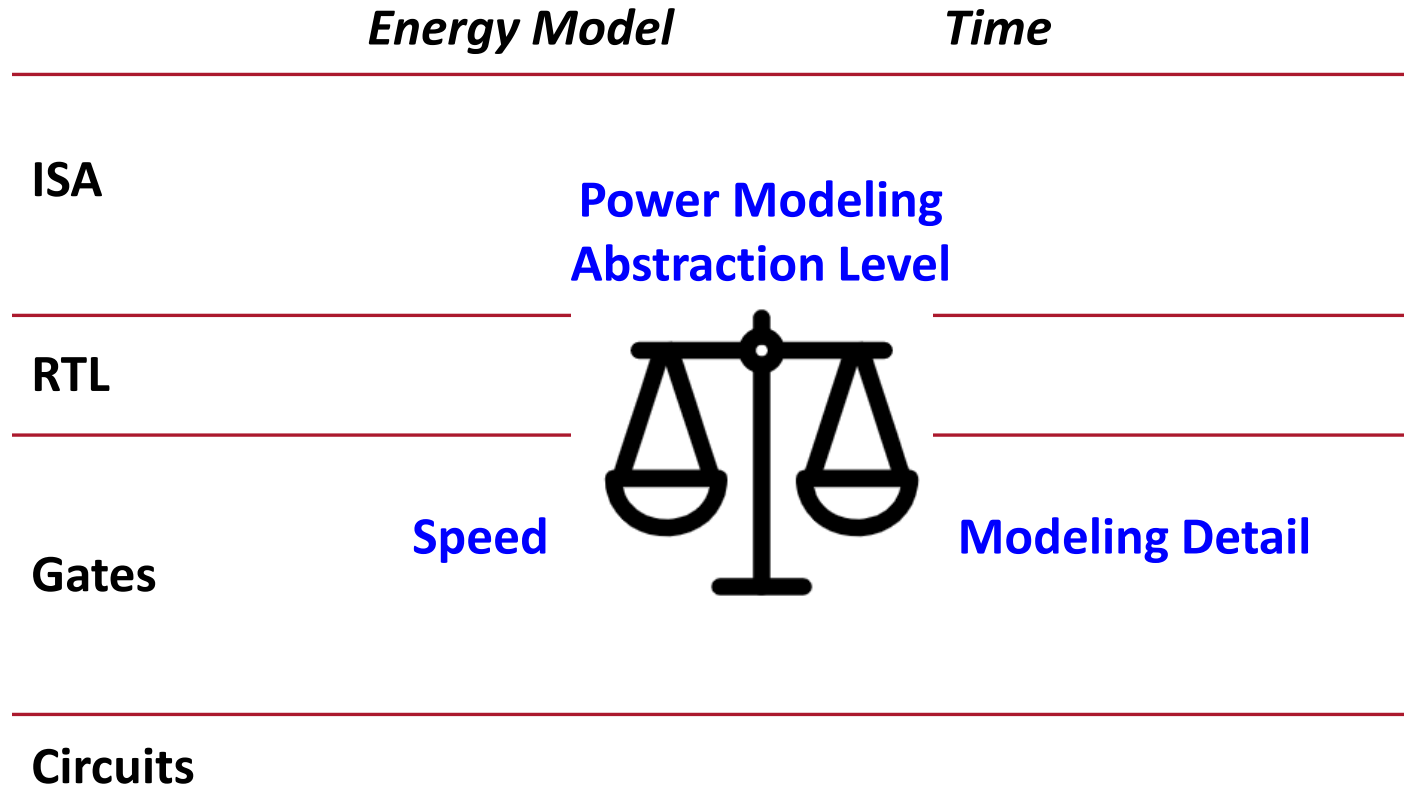


*False positives are bound to happen too!*

SCL may occur outside of the modeled artifact

# Power Modeling Abstraction Levels

---



# Power Modeling Abstraction Levels

---

	<i>Energy Model</i>	<i>Time</i>	
<b>ISA</b>	Full CPU $\Sigma$ $\mu$ Unit	Per Instruction Per $\mu$ Op Per Cycle	<i>Wattch</i> <i>SimplePower</i>
<b>RTL</b>			
<b>Gates</b>			
<b>Circuits</b>			

# Power Modeling Abstraction Levels

---

	<i>Energy Model</i>	<i>Time</i>	
<b>ISA</b>	Full CPU $\Sigma$ $\mu$ Unit	Per Instruction Per $\mu$ Op Per Cycle	
<b>RTL</b>	$\Sigma$ Register Transfer	Per Cycle	<i>Joules</i> <i>SpyGlass</i>
<b>Gates</b>			
<b>Circuits</b>			

# Power Modeling Abstraction Levels

---

	<i>Energy Model</i>	<i>Time</i>	
<b>ISA</b>	Full CPU $\Sigma$ $\mu$ Unit	Per Instruction Per $\mu$ Op Per Cycle	
<b>RTL</b>	$\Sigma$ Register Transfer	Per Cycle	
<b>Gates</b>	$\Sigma$ Weighed Transition + Leakage + Noise	Per Event	<i>PrimeTime</i> <i>Joules</i> <i>RedHawk</i> <i>Voltus</i>
<b>Circuits</b>			

# Power Modeling Abstraction Levels

---

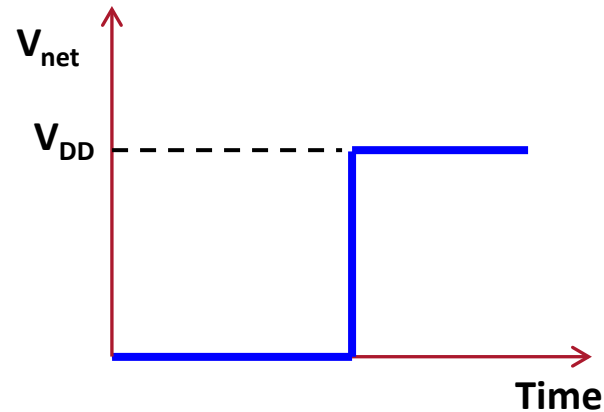
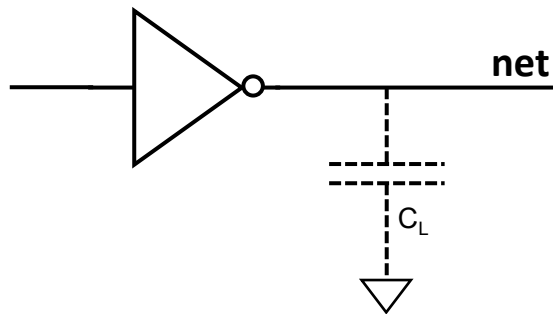
	<i>Energy Model</i>	<i>Time</i>
<b>ISA</b>	Full CPU $\Sigma$ $\mu$ Unit	Per Instruction Per $\mu$ Op Per Cycle
<b>RTL</b>	$\Sigma$ Register Transfer	Per Cycle
<b>Gates</b>	$\Sigma$ Weighed Transition + Leakage + Noise	Per Event
<b>Circuits</b>	I, V modeling	Transient/AC/DC <i>LTSpice</i>

# Outline

---

- Pre-silicon Tooling
- Background on Gate-level Power Modeling
  - NDLM/NDPM power modeling
  - LIB Files and the gate power simulation algorithm
- Architecture Correlation Analysis
- Applications and Tools
- What's next?

# Power Modeling by Net Transition



$$E_C = \frac{C_L V_{DD}^2}{2}$$

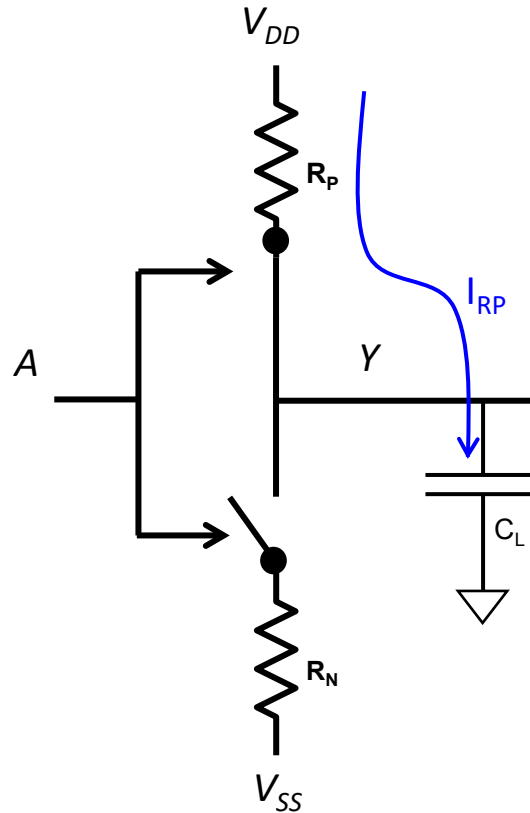
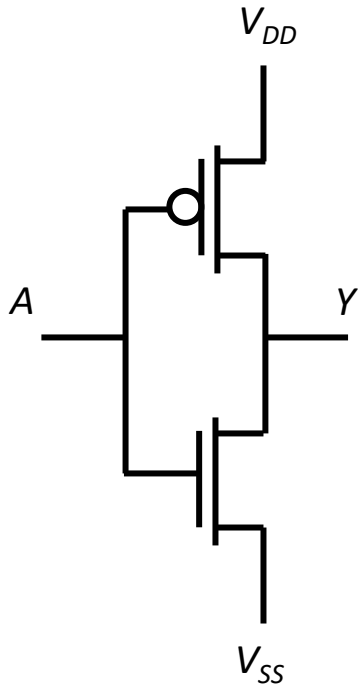
**For one transition**

$C_L = f(\text{gates driven, wire topology})$

$$P_{avg} = \frac{1}{T} \cdot \frac{C_L V_{DD}^2}{2}$$

**Average power for one transition per T seconds**

# Power dissipated by driver gate



$$E_{supply} = C_L V_{DD}^2$$

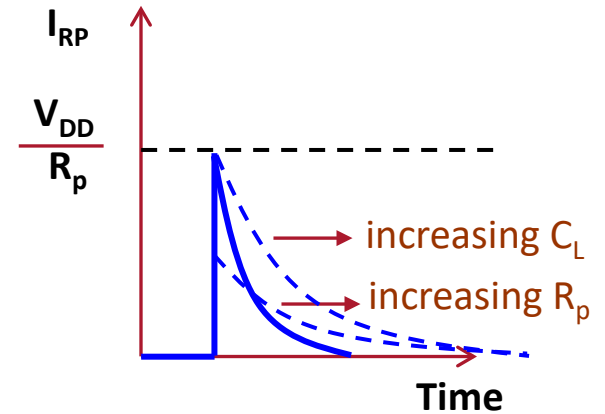
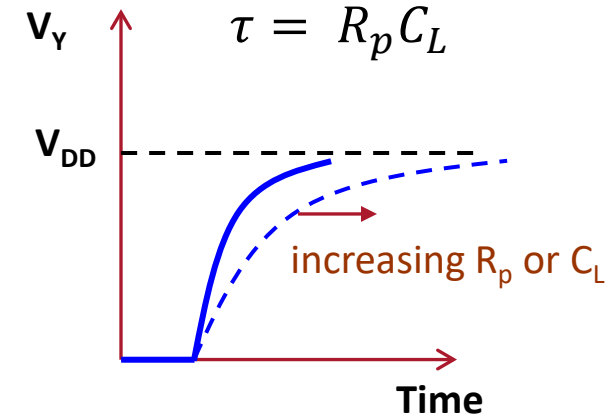
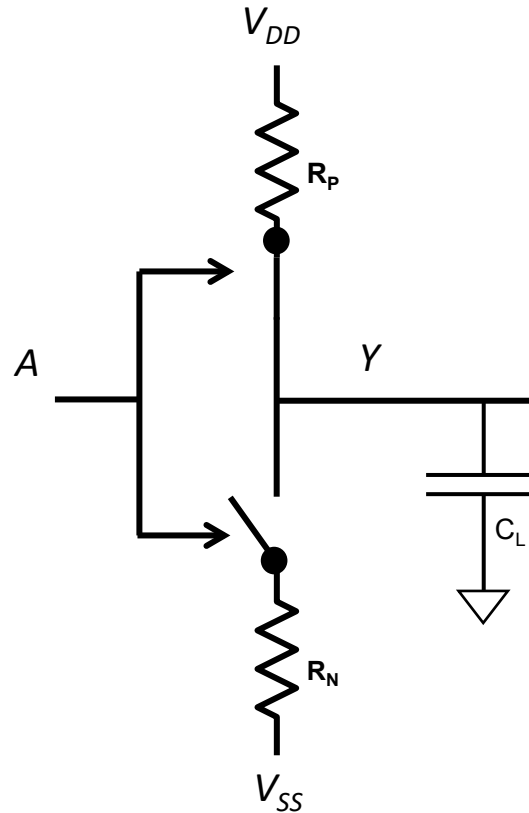
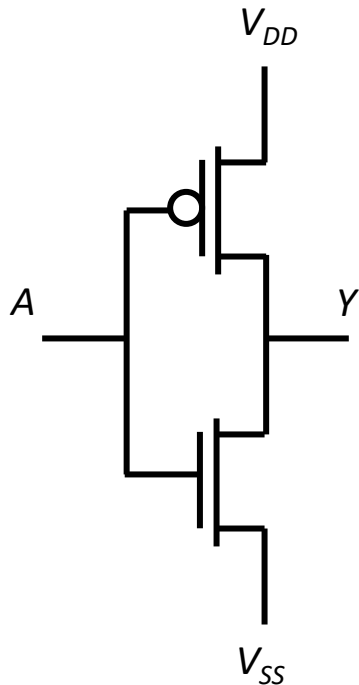


$$E_R = \frac{C_L V_{DD}^2}{2}$$

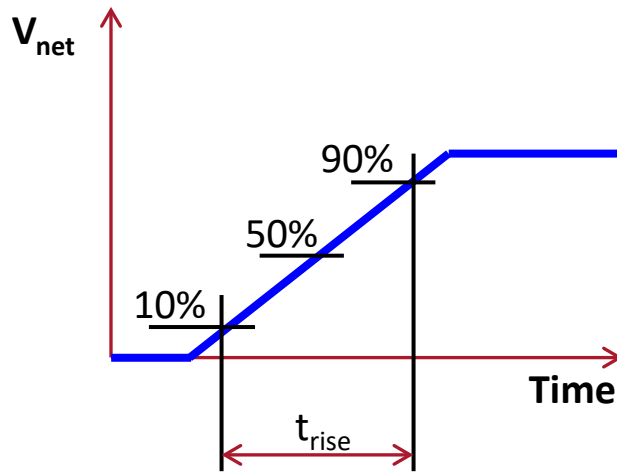


$$E_C = \frac{C_L V_{DD}^2}{2}$$

# $V_{\text{net}}(t)$ depends on gate and load



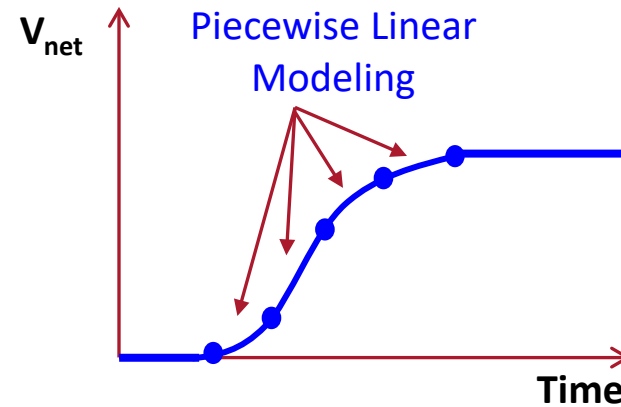
# Net Transition in Gate Simulator



Transition Delay

**NLDM/NLPM**  
( $> 90$  nm)

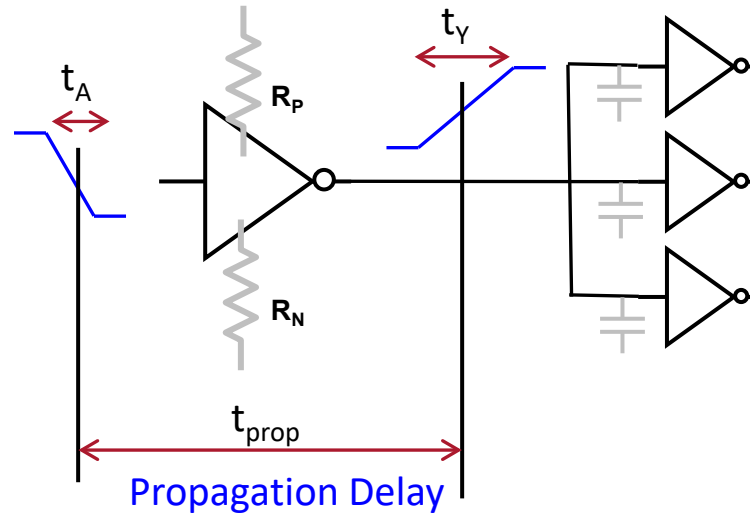
Non-Linear Delay/Power Modeling  
(constant input capacitance)



**CCS, ECSM**  
( $< 90$ nm)

Composite Current Source  
(variable input capacitance)

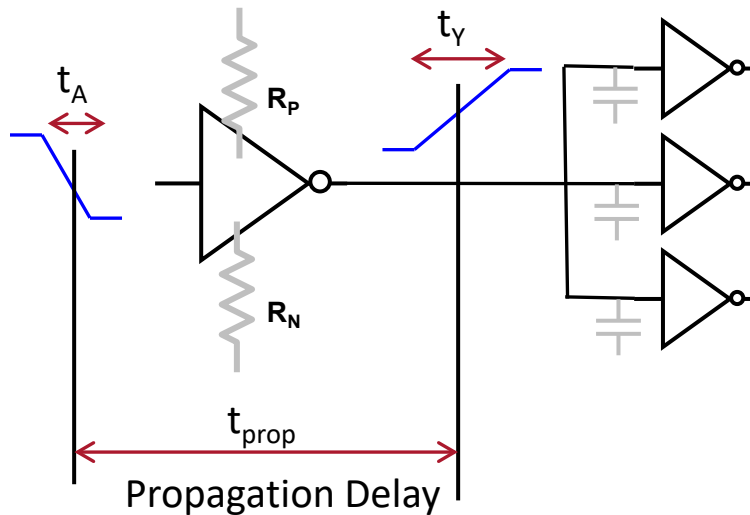
# Gate Timing Simulator computes $t_{\text{fall}}$ , $t_{\text{rise}}$ , $t_{\text{prop}}$



Transition & Propagation Delay depends on

1. Number of gates driven ( $C_L$ )
2. Drive Strength of gate ( $R_N, R_P$ )
3. Input Transition Delay  $t_A$

# Gate Timing Simulator computes $t_{fall}$ , $t_{rise}$ , $t_{prop}$

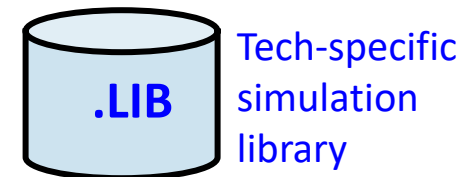


Transition & Propagation Delay depends on

1. Number of gates driven ( $C_L$ )
2. Drive Strength of gate ( $R_N, R_P$ )
3. Input Transition Delay  $t_A$

E.g. Google + SkyWater  
130nm Standard Cell Lib  
sky130\_fd\_sc\_lp

108 inverters  
159 AND/OR/NAND/NOR  
16 XOR/XNOR  
138 AND\_OR\_INV  
132 AND\_OR  
59 ADD/MUX  
92 Flip-flop/Latch  
43 Power Gating



# Delay modeling (.LIB)

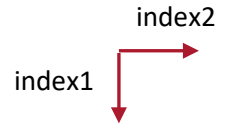
```
cell ("sky130_fd_sc_hs__clkinv_1") {
  pin ("Y") {
    direction : "output";
    function : "(!A)";
  }
  timing () {
    cell_fall {
      index_1("0.01, 1.50");
      index_2("0.00, 0.19");
      values("0.02018, 1.08777", \
            "0.20860, 1.58840");
    }
    cell_rise {
      index_1("0.01, 1.50");
      index_2("0.00, 0.19");
      values("0.01125, 0.56975", \
            "-0.1217, 0.98200");
    }
    fall_transition { ... }
    rise_transition { ... }
  }
}
```

## Inverter gate of SKY130

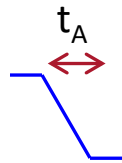
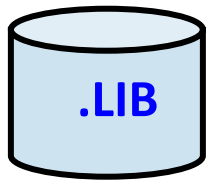
**1->0 Propagation Delay**  
**Input Transition Delay**  
**Loading Capacitance**  
**Delay**

**0->1 Propagation Delay**  
**Input Transition Delay**  
**Loading Capacitance**  
**Delay**

**1 -> 0 Transition Delay**  
**0 -> 1 Transition Delay**



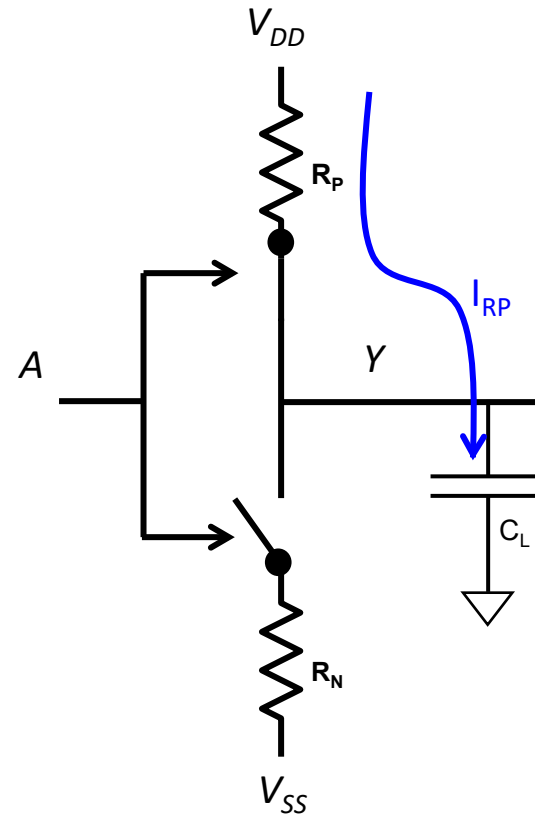
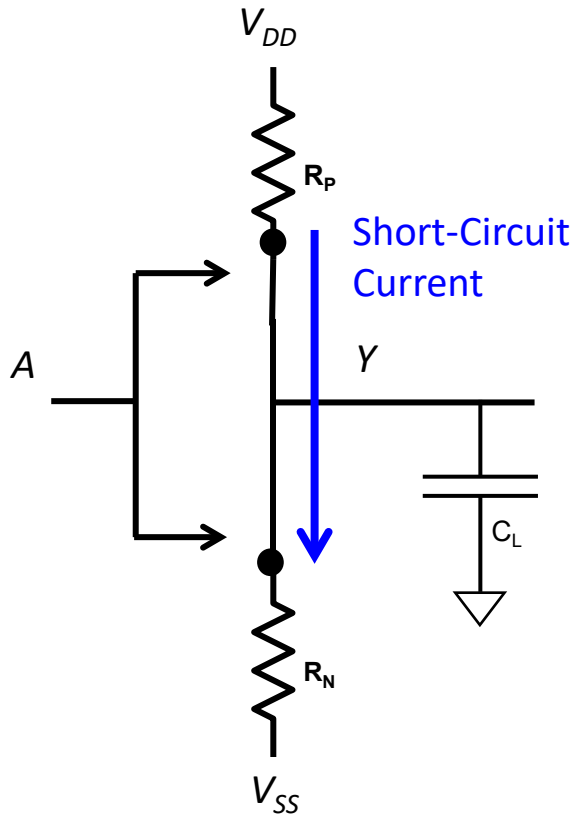
# Energy p. Transition = $f(t_A, R_{P/N}, C_L)$



Short-Circuit Energy

+

Output (Dis)Charging Energy



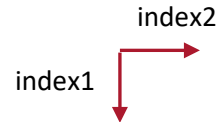
# Energy modeling (.LIB)

```
cell ("sky130_fd_sc_hs__clkinv_1") {  
    ...  
    pin ("Y") {  
        direction : "output";  
        function : "(!A)";  
        internal_power () {  
            related_pin : "A";  
            fall_power {  
                index_1 ("0.01, 1.50");  
                index_2 ("0.00, 0.19");  
                values ("0.00133, -0.31556", \  
                    "0.05087, -0.30703");  
            }  
            rise_power {  
                index_1 ("0.01, 1.50");  
                index_2 ("0.00, 0.19");  
                values ("0.0118, 0.32574", \  
                    "0.0636, 0.34044");  
            }  
        }  
    }  
}
```

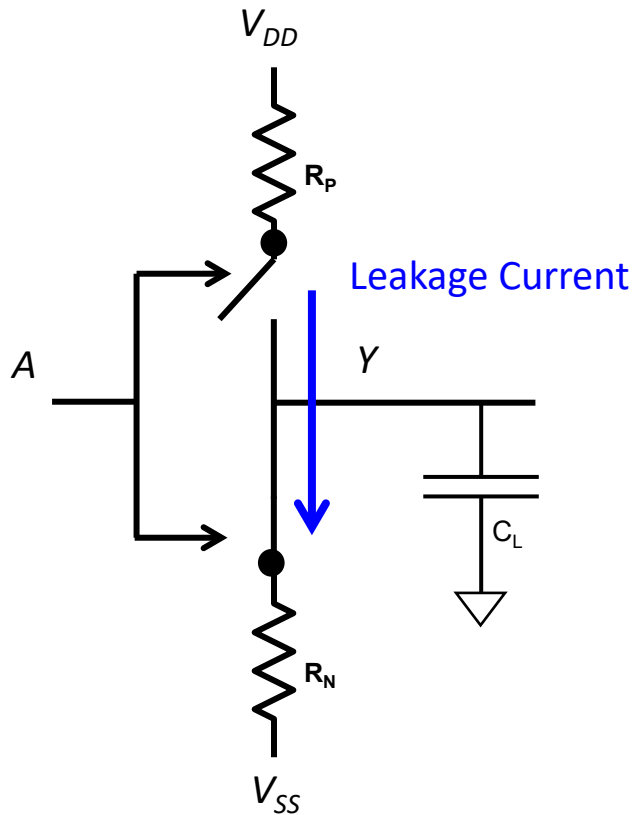
## Inverter gate of SKY130

1->0 transitions on Y  
Input Transition Delay  
Loading Capacitance  
Energy per Transition

0->1 transitions on Y  
Input Transition Delay  
Loading Capacitance  
Energy per Transition



# Leakage Power = f(gate, input state)

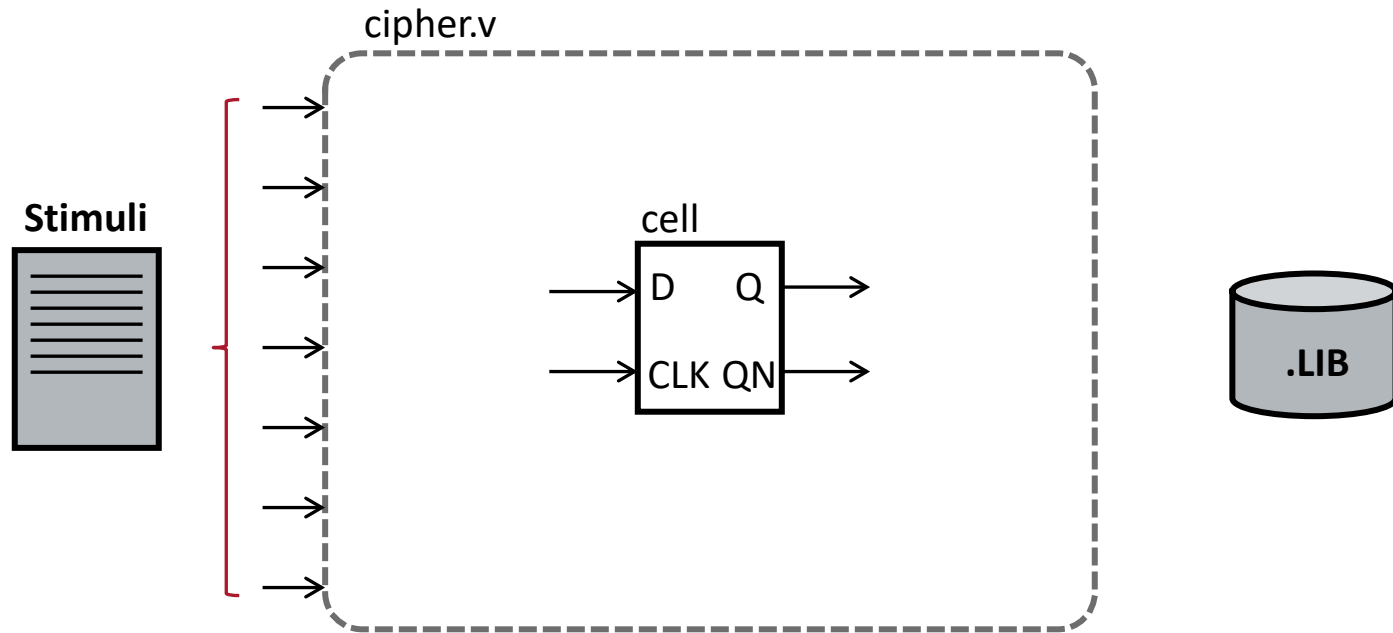


Leakage Power = f(gate, input state)

```
cell ("sky130_fd_sc_hs__clkinv_1") {  
  leakage_power () {  
    value : 0.6800200000;  
    when : "!A";  
  }  
  leakage_power () {  
    value : 188.15526000;  
    when : "A";  
  }  
  ...  
}
```

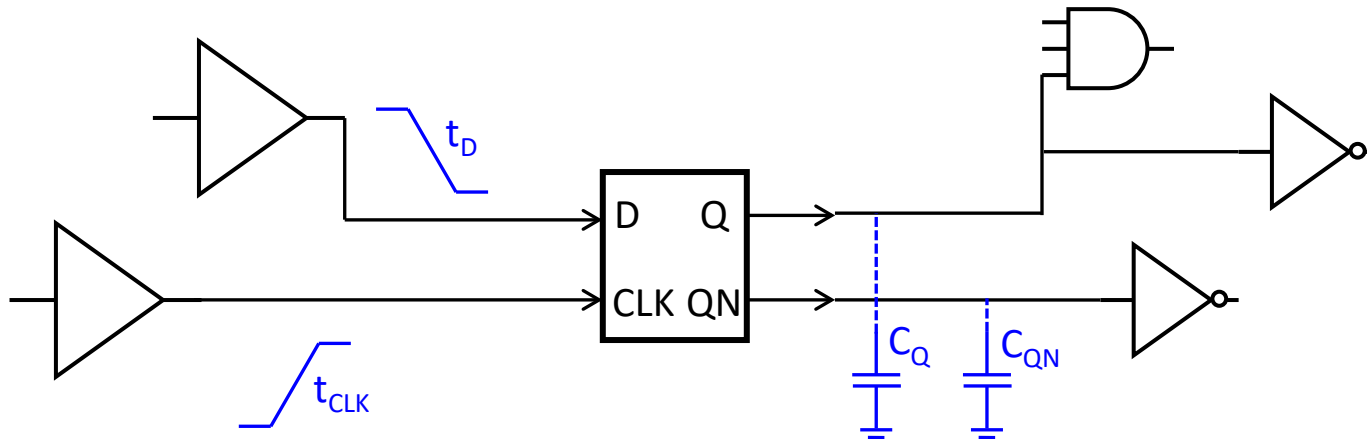


# Overall Gate-Level Power



- Design-level Power  
=  $\sum$  Cell Power  
=  $\sum$  (Internal + Output Switching + Leakage) Power
- Gate-level Power Estimation requires Stimuli, Tech Library

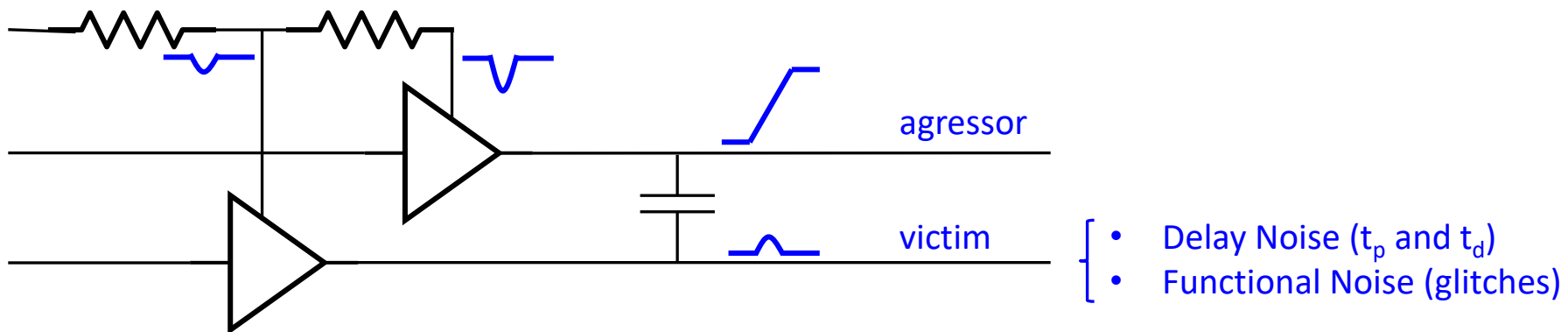
# Computing Overall Gate-Level Power



- For every vector, for every event, for every cell
  - Determine Input Activity and Slew
  - Determine Output Loading Capacity
- Interpolate .LIB to determine leakage power and internal energy
- Use clock frequency or (activity factor) to determine (average) per-cell power

# Advanced Gate-level Power

- Gate-level power simulators are 2x .. 5x slower\* than gate-level simulation
- State-of-the-art gate-level power simulators are within a few % of low-level circuit power simulators (cfr best RTL simulators > 15% error)
- Advanced simulators can also capture noise effects
  - Power integrity
  - Post-layout Parasitic Coupling



\* typical SK crypto HW/SW

# Outline

---

- Pre-silicon Tooling
- Background on Gate-level Power Modeling
- Architecture Correlation Analysis
  - Basic Technique
  - Specific and Non-specific Testing for Leakage
- Applications and Tools
- What's next?

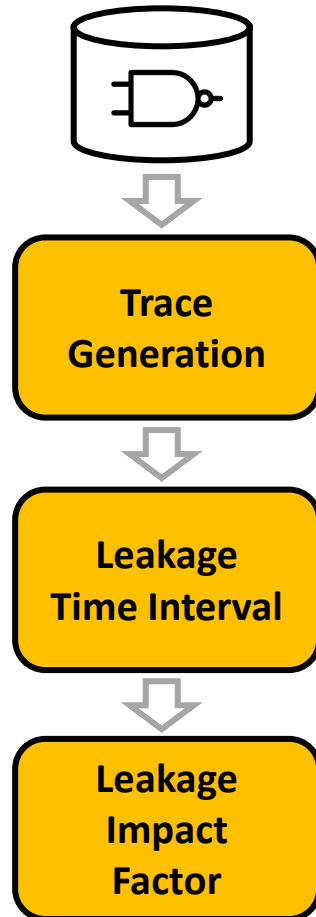
# Architecture Correlation Analysis

---

- **Objective:** Rank the cells in a hardware design according to their (power-based) side-channel leakage.
- Several practical use cases:
  1. Emulating traditional side-channel attack in simulation (noiseless!)
  2. Identifying unexpected sources of side-channel leakage:
    - micro-architecture
    - system integration
    - hardware/software interaction
  3. Determining the need for countermeasures, possibility of local countermeasures

# Architecture Correlation Analysis

---



## Design Description

- Post-Synthesis Netlist

## Collect Power Traces for Selected Stimuli

- Logic Simulation + Power Simulation
- Stimuli depend on testing strategy

## Determine Leaky Points within Trace

- Specific Test
- Non-specific Test

## Quantify Per-cell Leakage

- Activity based (VCD)

# Gate-Level Power Simulation

---

Gate Power =

Switching Power +  
Internal Power +  
Leakage Power

Gate Power =

Output\_Toggle\_Rate  $\times V_{dd}^2 \times C_{Load}$  +  
Input\_Toggle\_Rate  $\times P_{dyn,cell}$  +  
Input\_State  $\times P_{leak,cell}$

Frame-based Power Estimation

$$\text{Toggle\_Rate} = \frac{\# \text{ events}}{\text{frame}}$$

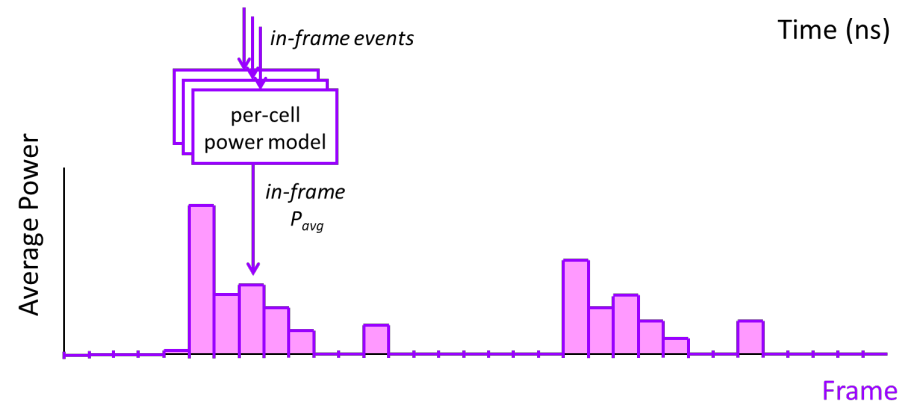
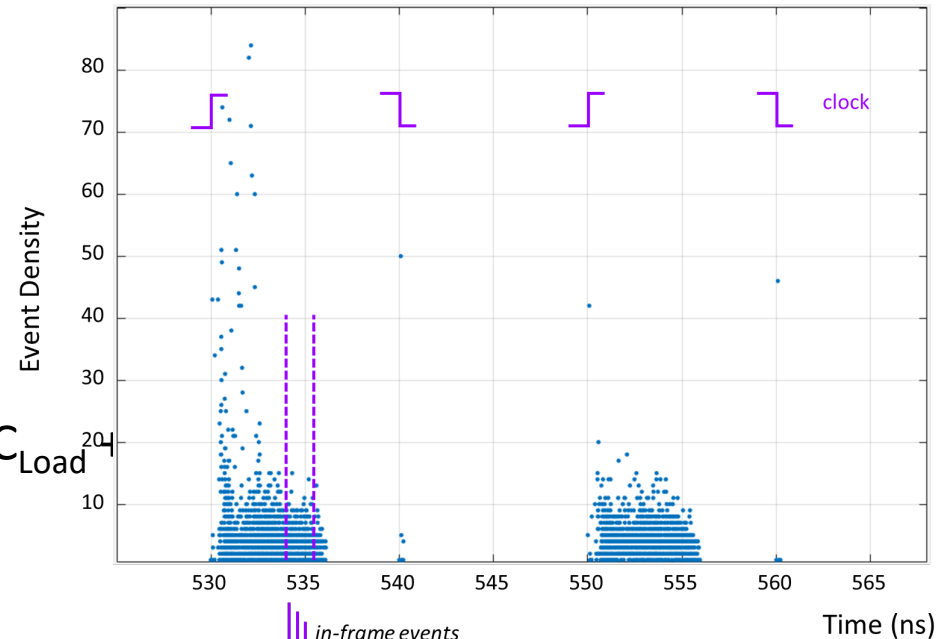
# Gate-Level Power Simulation

Gate Power =  
 Switching Power +  
 Internal Power +  
 Leakage Power

Gate Power =  
 $\text{Output\_Toggle\_Rate} \times V_{dd}^2 \times C_{Load}$   
 $\text{Input\_Toggle\_Rate} \times P_{dyn,cell} +$   
 $\text{Input\_State} \times P_{leak,cell}$

## Frame-based Power Estimation

$$\text{Toggle\_Rate} = \frac{\# \text{ events}}{\text{frame}}$$



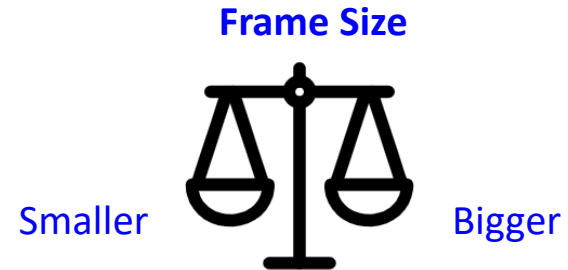
# Gate-Level Power Estimation

Gate Power =  
 Switching Power +  
 Internal Power +  
 Leakage Power

Gate Power =  
 $\text{Output\_Toggle\_Rate} \times V_{dd}^2 \times C_{\text{Load}} +$   
 $\text{Input\_Toggle\_Rate} \times P_{\text{dyn,cell}} +$   
 $\text{Input\_State} \times P_{\text{leak,cell}}$

Frame-based Power Estimation

$$\text{Toggle\_Rate} = \frac{\# \text{ events}}{\text{frame}}$$



*higher resolution*

*faster estimation*

256 cycle Testbench

Normalized Complexity

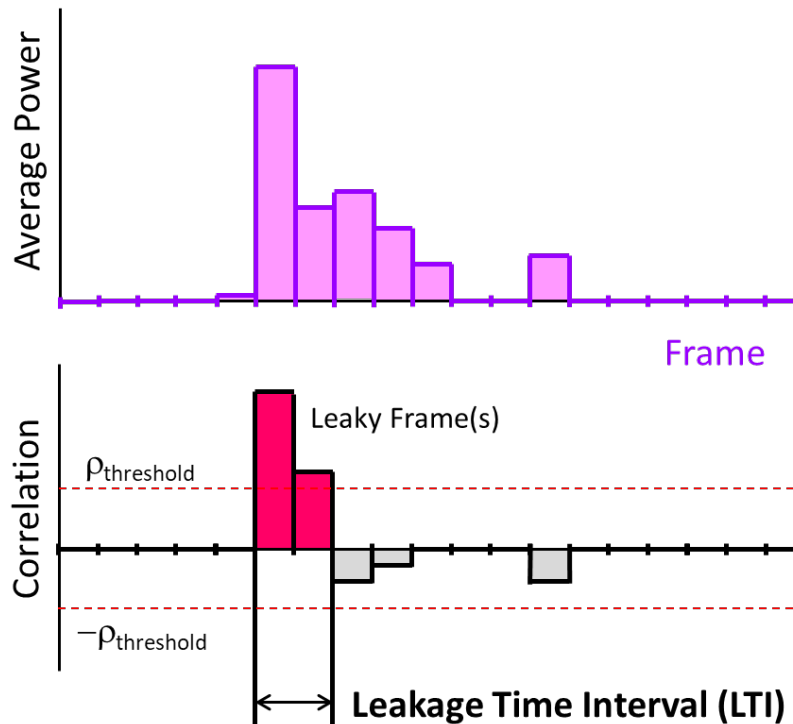
fm/cyc	frames	1 SBOX	16 SBOX
1/256	1	1	4.54
1	256	6.9	93.8
2	512	11.1	149.4

*skywater 130, Cadence Joules*

# Leakage Impact Factor (1)


- **Specific Test**

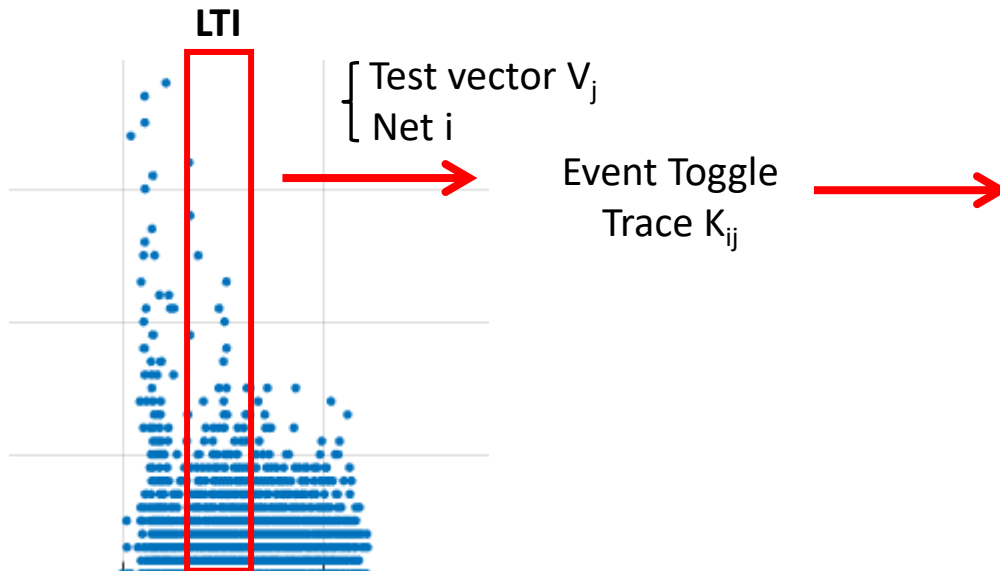
- Select internal variable  $V$ , Leakage model  $L(V)$  (e.g.,  $\text{HammingWeight}(V)$ )
- Compute correlation per frame  $\rho[n] = \text{correlate}(L(V), P[n])$



# Leakage Impact Factor (2)

- Architecture Correlation Analysis

“Toggle Trace” {  or  = +1  
= -1



Architecture Correlation Cell  $C_i$

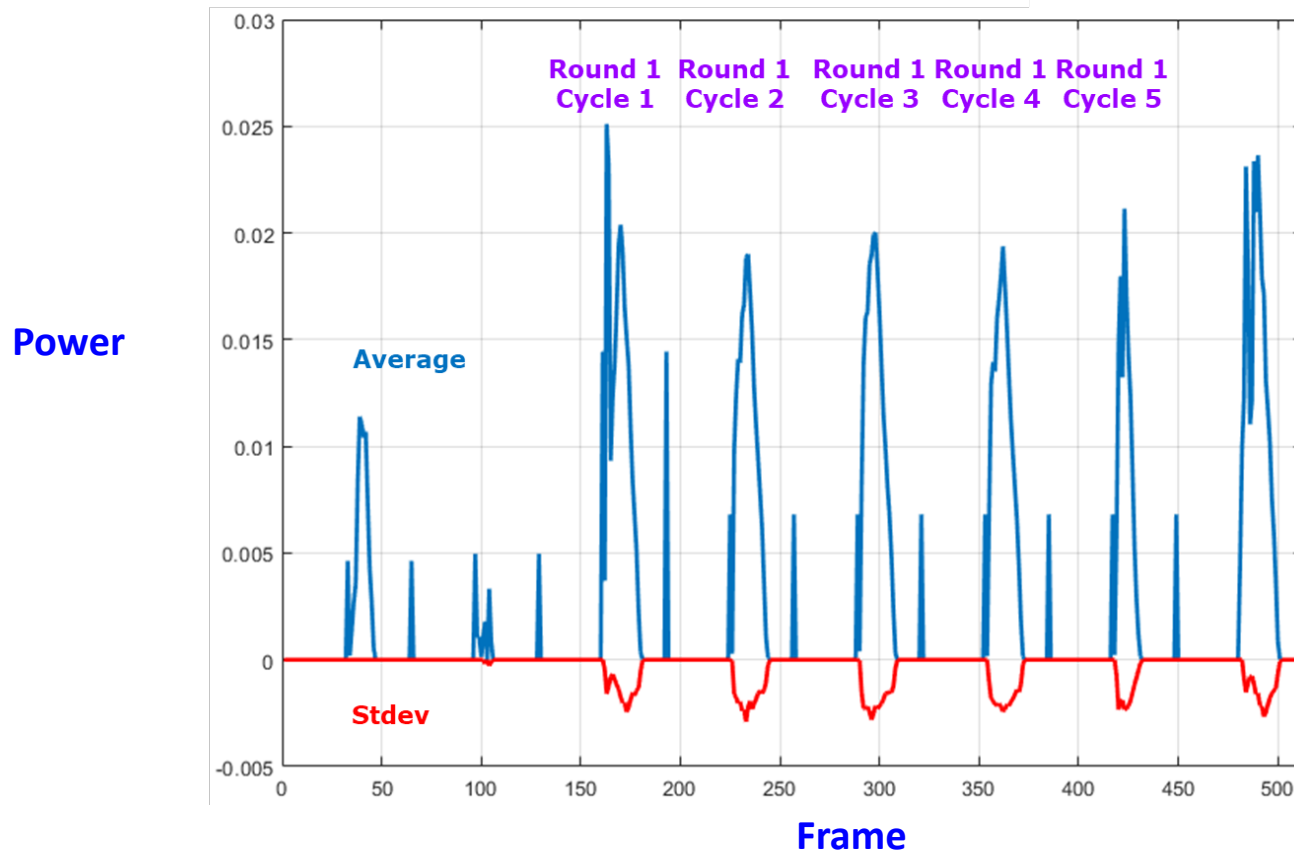
$$C_i = \text{correlate}(L(V_j), K_{ij})$$

Leakage Impact Factor  $F_i$

$$F_i = \frac{C_i \times P_{\text{avg},i}}{P_{\text{avg},T}}$$

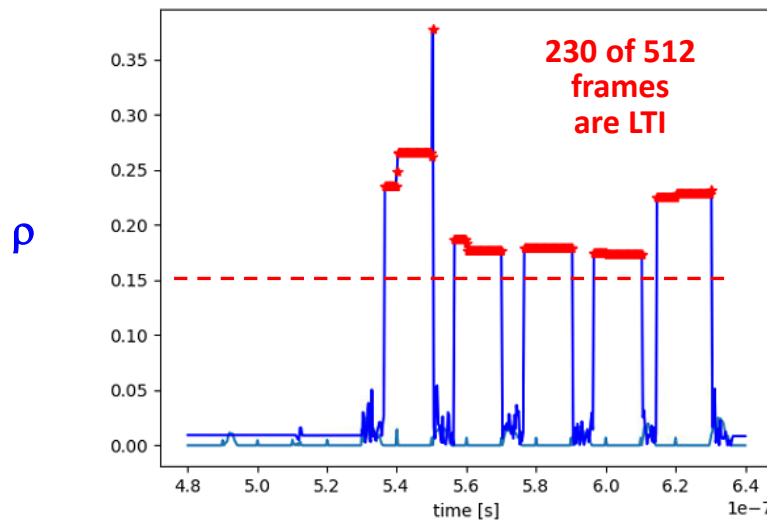
# Example of Specific ACA

- AES on 9,640 cells (sky130), 1024 test vectors,  $L(V) = \text{HW}(\text{SBOX output key byte0})$



# Example of Specific ACA

- AES on **9,640 cells** (sky130), 1024 test vectors,  $L(V) = \text{HW}(\text{SBOX output key byte0})$
- Leakage Time Interval Selection with  $\rho_{\text{threshold}} = 0.15$



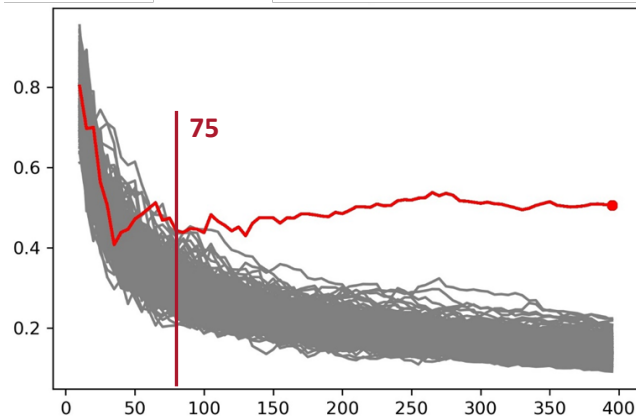
ACA  
➔

**412 cells** (4.3%)  
have non-negligible LIF

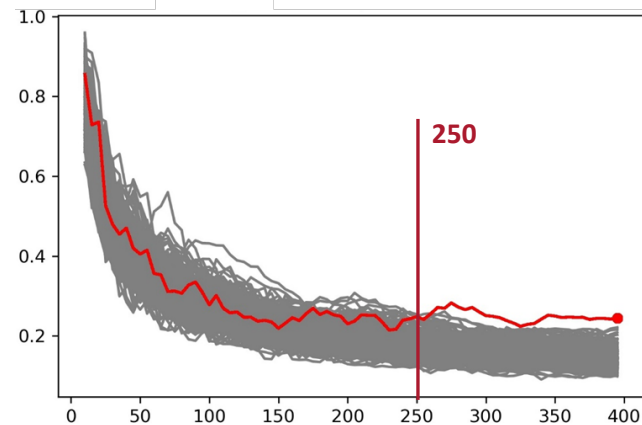
# Example of Specific ACA

- AES on 9,640 cells (sky130), 1024 test vectors,  $L(V) = \text{HW}(\text{SBOX output key byte0})$
- Leakage Time Interval Selection with  $\rho_{\text{threshold}} = 0.15$
- Leaky Gate Analysis: 412 cells leaky

CPA on original AES traces



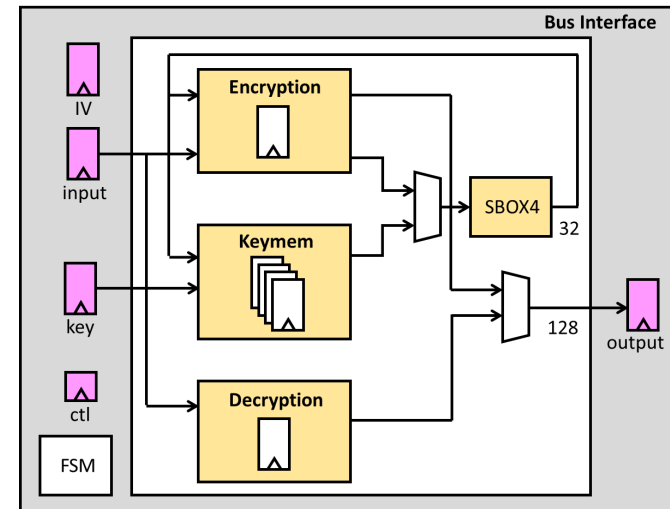
CPA on AES traces with leaky-gate power removed



# Example of Specific ACA

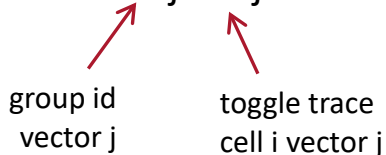
- AES on 9,640 cells (sky130), 1024 test vectors,  $L(V) = HW(SBOX \text{ output key byte0})$
- Leakage Time Interval Selection with  $\rho_{\text{threshold}} = 0.15$
- Leaky Gate Analysis: 412 cells leaky, 47 unique sites in RTL code

Module	Cells	Seq
Top-Level	5	0
Decryption	29	0
Encryption	204	26
Keymem	1	0
SBOX	130	0
Bus Itf	22	10
Unknown	21	0



# Non-specific ACA

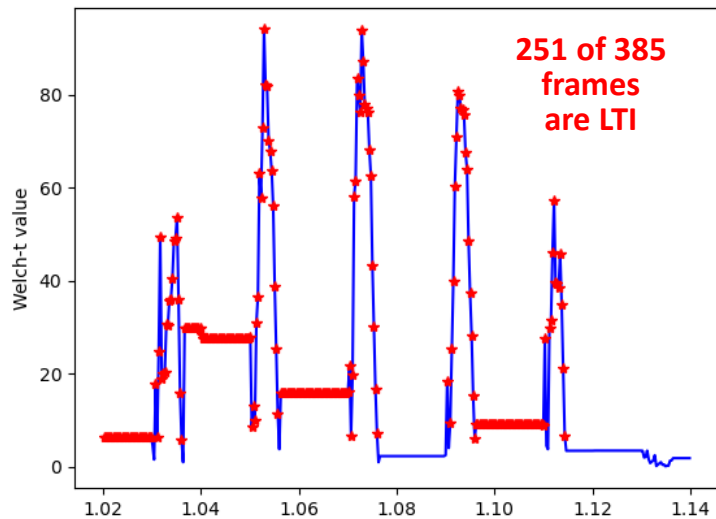
---

- Specific test requires designer to *assume* the attacker
- Extend Leakage Time Interval selection with **non-specific test**
  - Group 1:  $N = -1$ , Test Vectors with property 1 (e.g.  $\text{state}_{\text{AES}}[\text{round6}] = 0$ )
  - Group 2:  $N = +1$ , Test Vectors with property 2 (e.g.  $\text{state}_{\text{AES}}[\text{round6}] = \text{random}$ )
- Compute the LTI using  $\rho[n] = \text{correlate}(N, P)$
- Compute Architecture Correlation of cell  $C_i = \text{correlate}(N_j, K_{ij})$ 

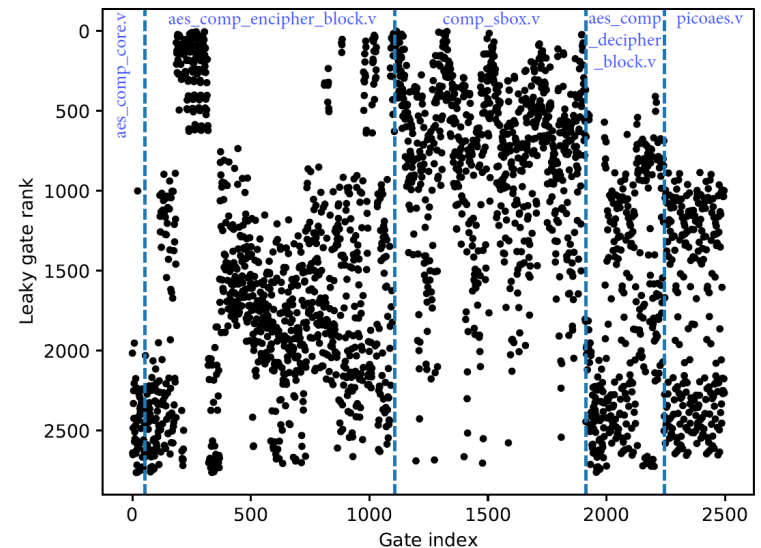
# Example of Non-Specific ACA

- AES on 9,640 cells (sky130), 512+512 test vectors, round 6 state bias
- Leakage Time Interval Selection
- Leaky Gate Analysis: 2,812 cells leaky

Round-6 zero-state LTI



\* strong bias flags *most* cells as leaky



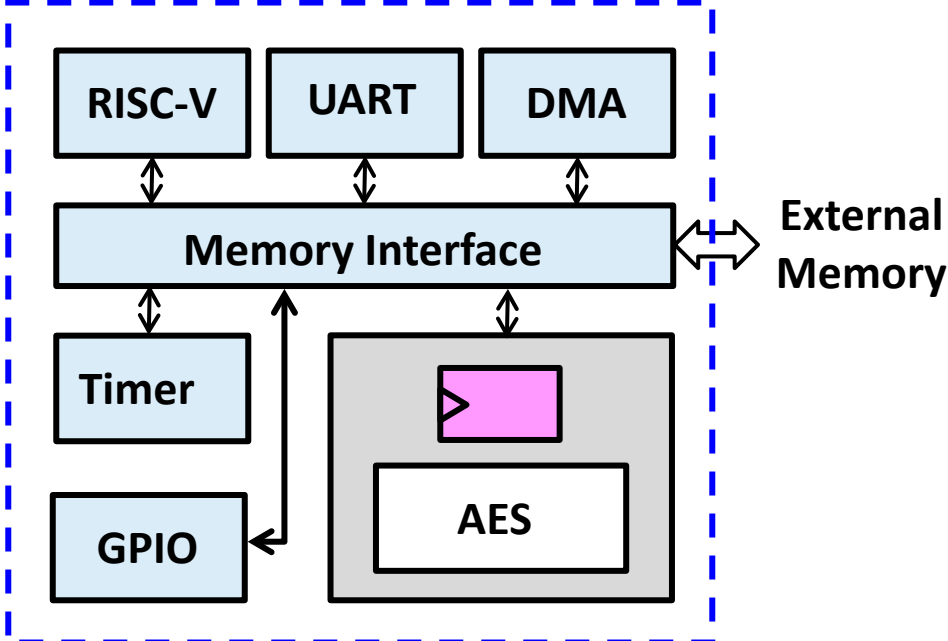
# Outline

---

- Pre-silicon Tooling
- Background on Gate-level Power Modeling
- Architecture Correlation Analysis
- Applications and Tools
  - SoC Analysis
  - Performance Optimization in ACA by Downsampling
- What's next?

# SoC Analysis

modeled in ACA

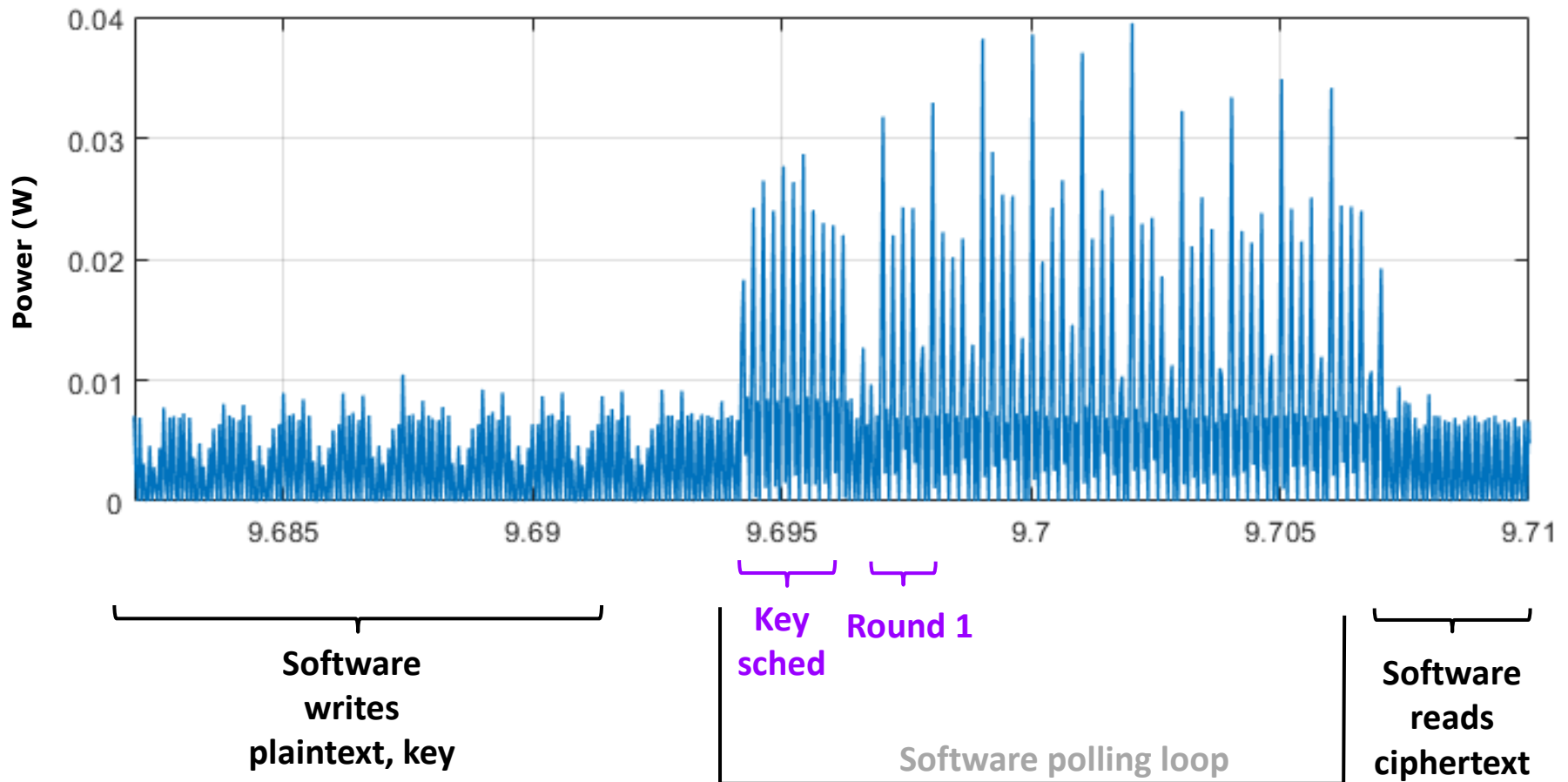


29,575 cells sky130  
50MHz System Clock

```

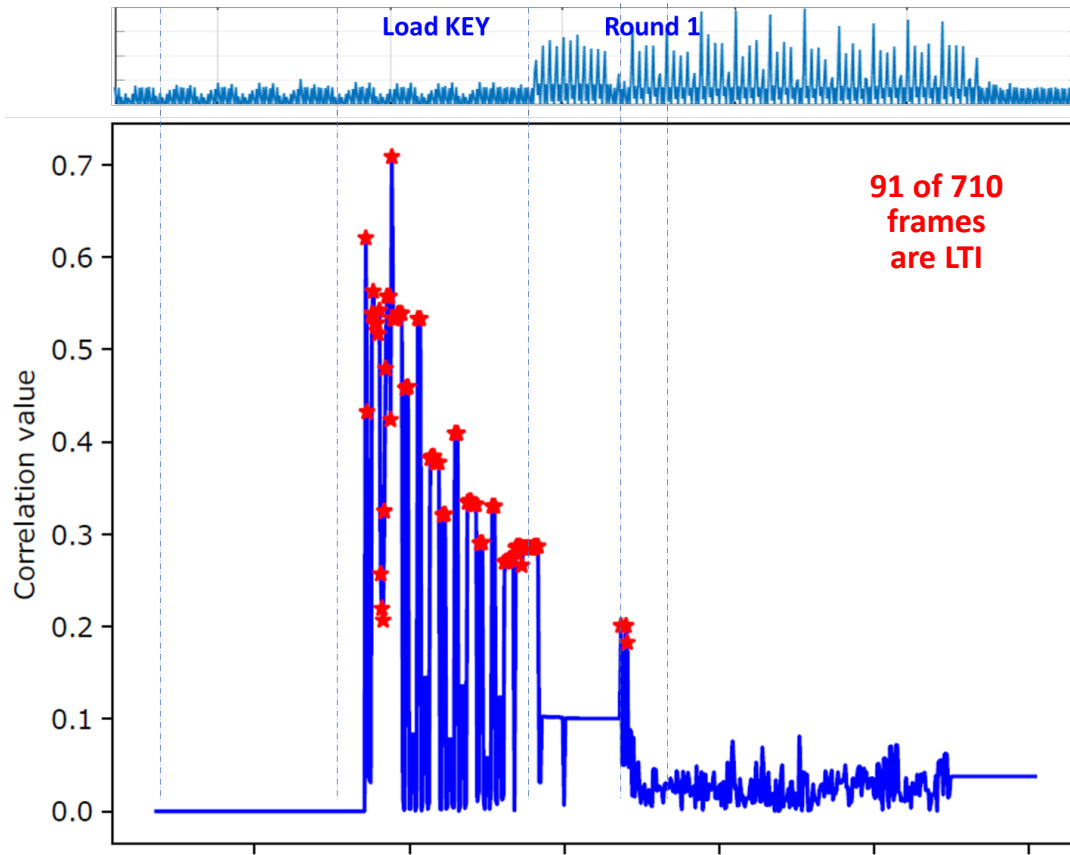
li      a4,8
lw      a3,0(a4)      ; load plaintxt[0]
sw      a3,4(a5)      ; STALL
lw      a3,4(a4)      ; load plaintxt[1]
sw      a3,8(a5)      ; STALL
lw      a3,8(a4)      ; load plaintxt[2]
sw      a3,12(a5)     ; STALL
lw      a4,12(a4)     ; load plaintxt[3]
sw      a4,16(a5)     ; STALL
li      a4,24
lw      a3,0(a4)      ; load key[0]
sw      a3,20(a5)     ; STALL
lw      a3,4(a4)      ; load key[1]
sw      a3,24(a5)     ; STALL
lw      a3,8(a4)      ; load key[2]
sw      a3,28(a5)     ; STALL
lw      a4,12(a4)     ; load key[3]
sw      a4,32(a5)     ; STALL
li      a4,6
sw      a4,0(a5)      ; control
li      a4,4
sw      a4,0(a5)      ; start
li      a3,1
.L121:
lw      a4,68(a5)
bne     a4,a3,.L121
    
```

# Power Trace of Coprocessor Op



# Specific ACA results for SoC

- $L(V) = HW(pt \text{ xor } key)$  at  $\rho_{\text{threshold}} = 0.2$



```

li      a4,8
lw      a3,0(a4)    ; load plaintxt[0]
sw      a3,4(a5)    ; STALL
lw      a3,4(a4)    ; load plaintxt[1]
sw      a3,8(a5)    ; STALL
lw      a3,8(a4)    ; load plaintxt[2]
sw      a3,12(a5)   ; STALL
lw      a4,12(a4)   ; load plaintxt[3]
sw      a4,16(a5)   ; STALL
li      a4,24
lw      a3,0(a4)    ; load key[0]
sw      a3,20(a5)   ; STALL
lw      a3,4(a4)    ; load key[1]
sw      a3,24(a5)   ; STALL
lw      a3,8(a4)    ; load key[2]
sw      a3,28(a5)   ; STALL
lw      a4,12(a4)   ; load key[3]
sw      a4,32(a5)   ; STALL
li      a4,6
sw      a4,0(a5)    ; control
li      a4,4
sw      a4,0(a5)    ; start
li      a3,1
.L121:
lw      a4,68(a5)
bne     a4,a3,.L121
    
```

# Specific ACA results for SoC

---

- 1,298 cells (out of 29,575 cells) flagged as leaky

Module	Cell	Seq
AES Coproc	289	126
RISCV ALU	332	0
RISCV Control	12	0
RISCV Memory	14	0
RISCV Pipeline	66	31
RISCV Regfile	248	248
Peripherals	29	0

# Performance

	AES Coprocessor (9,640 cells)	RISC-V + AES Coprocessor (29,575 cells)
Logic Synthesis	392	1,210
Logic Simulation	2,436	6,996
Power Simulation (64 fpc)	7,862	
Power Simulation (2 fpc)	1,557	
Power Simulation (5 fpc)		31,201
Correlation Analysis	< 60	< 60

Runtime in sections on Xeon Gold 6248 CPU (2.5 GHz, 384G)

# Performance

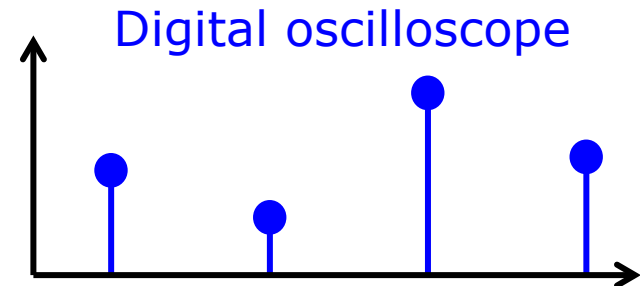
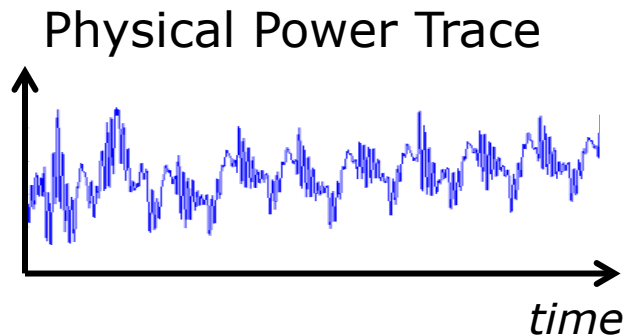
	AES Coprocessor (9,640 cells)	RISC-V + AES Coprocessor (29,575 cells)
Logic S	<b>Increasing Frame Size shortens simulation time</b>	
Logic S	<b>But does it also degrade SCL Assessment?</b>	
Power Simulation (64 fpc)	7,862	
Power Simulation (2 fpc)	1,557	
Power Simulation (5 fpc)		31,201
Correlation Analysis	< 60	< 60

Runtime in sections on Xeon Gold 6248 CPU (2.5 GHz, 384G)

# Downsampling SCA

---

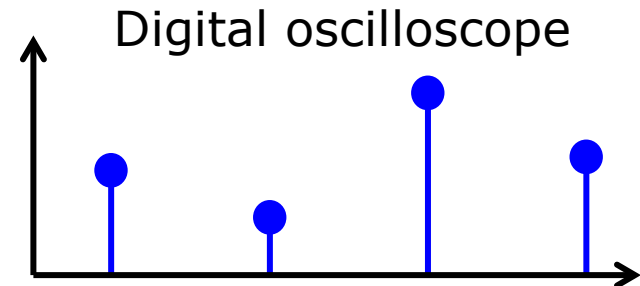
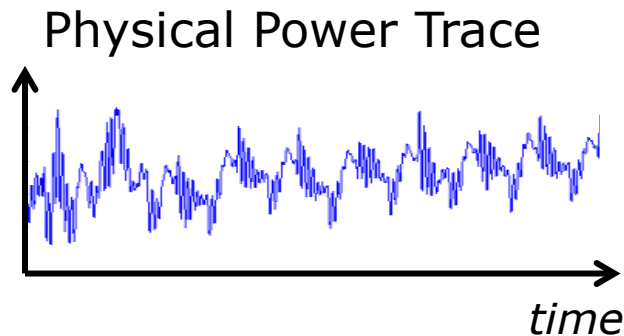
- What is the impact of reducing sample rate?



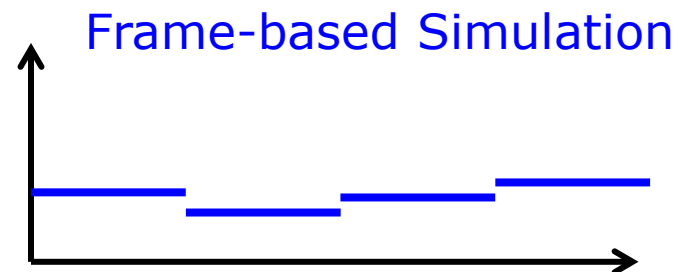
- Aliasing Occurs
- May be resolved using LPF

# Downsampling SCA

- What is the impact of reducing sample rate?



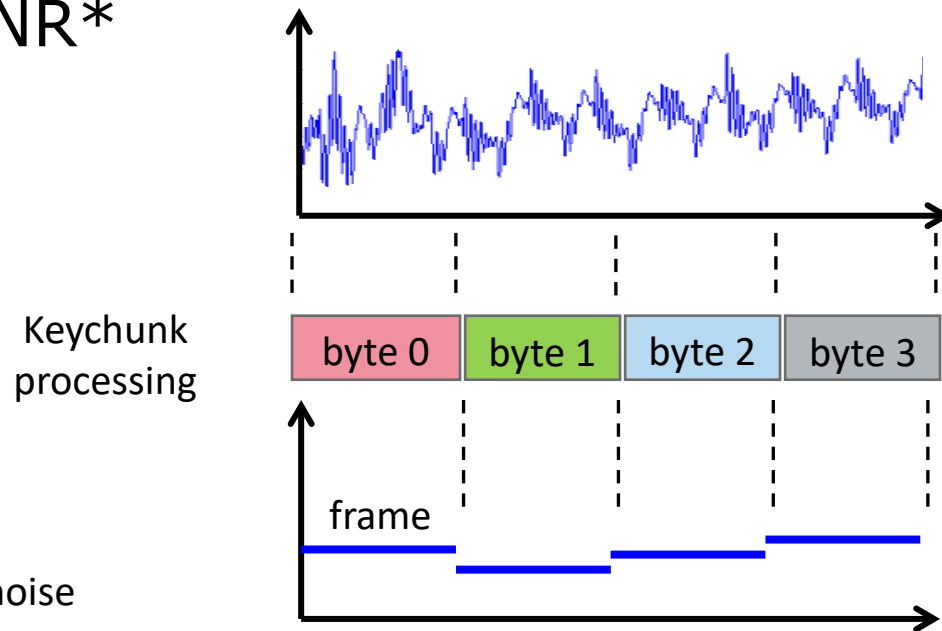
- Aliasing Occurs
- May be resolved using LPF



- Implicit averaging avoid Aliasing

# Constructive Samples

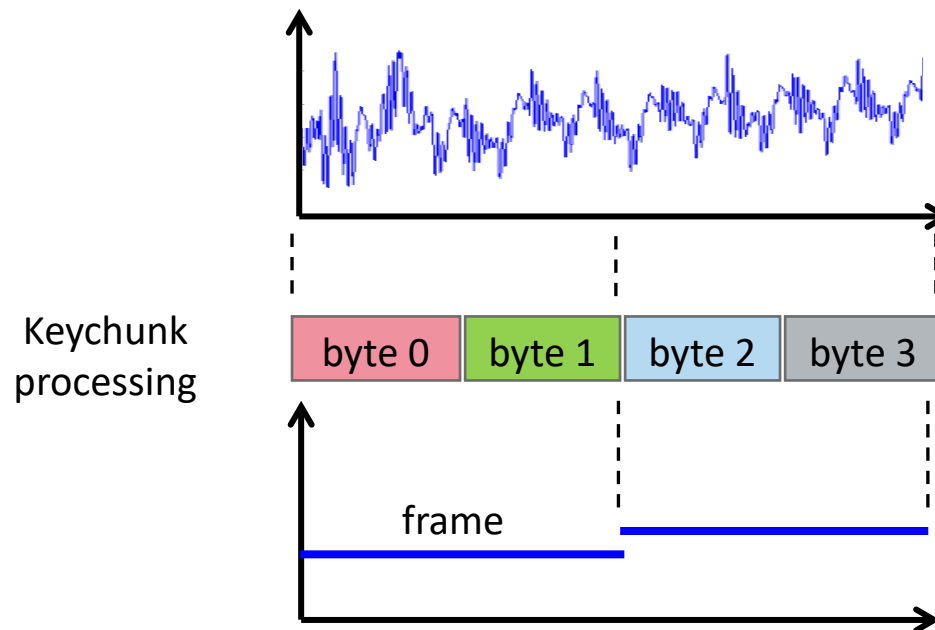
- SCL Assessment is done per *keychunk* (bit, byte, ..)
- Power samples belonging to same keychunk processing are *constructive* samples
- Downsampling of constructive samples does not degrade SNR\*



\* In this SNR,  
N means *algorithmic* noise

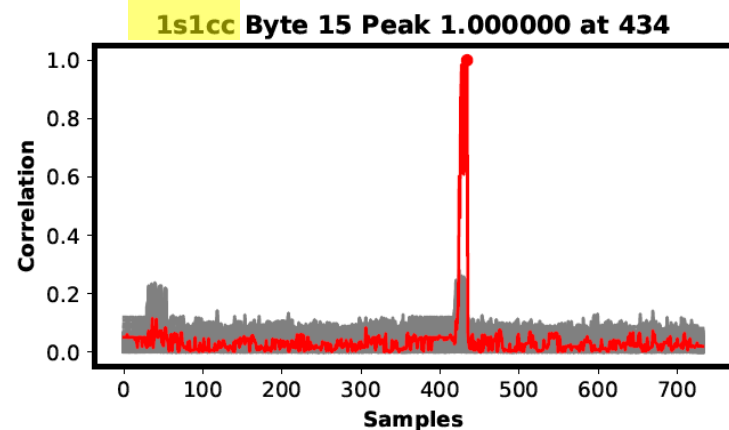
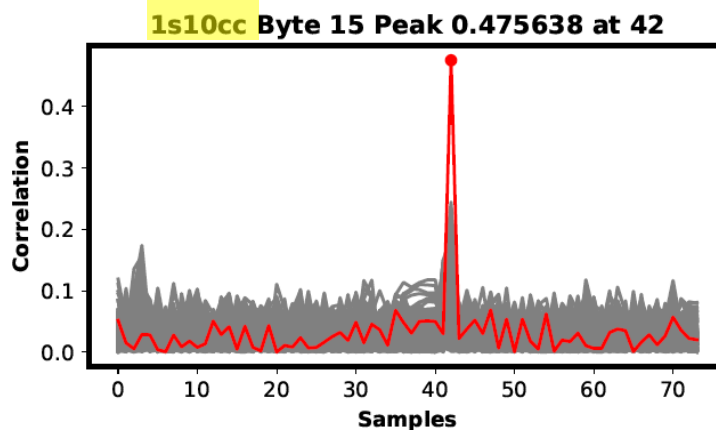
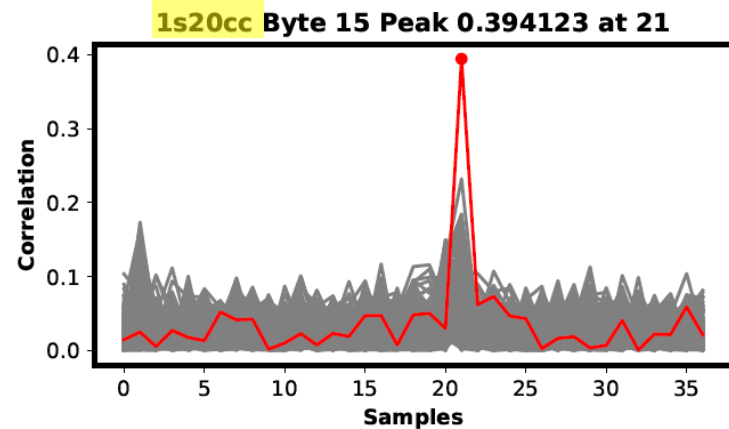
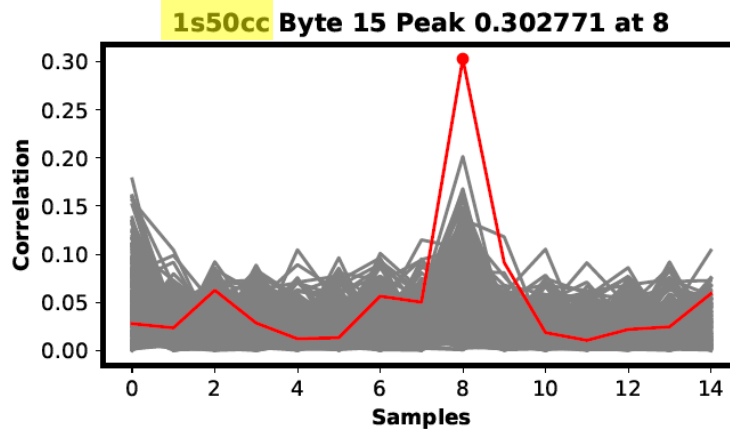
# Destructive Samples

- Power samples belonging to different keychunk processing are *destructive* samples
- Downsampling of destructive samples degrades SNR



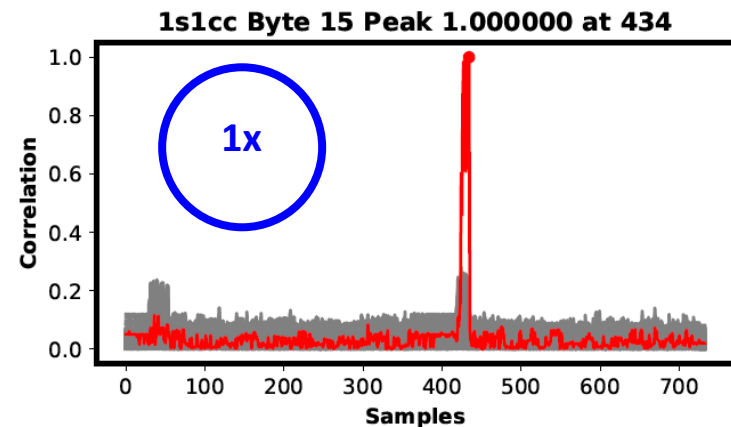
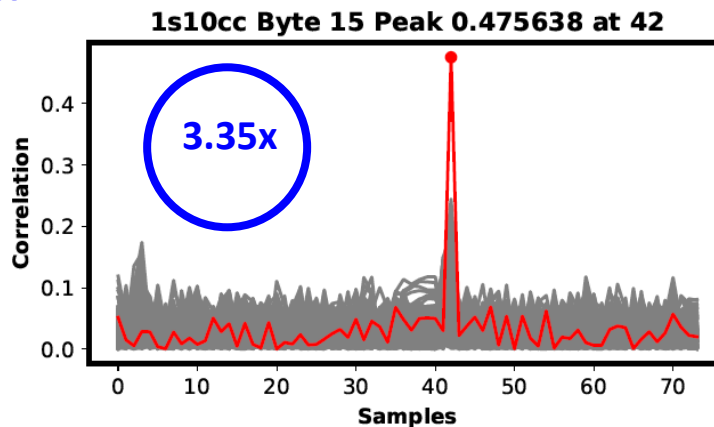
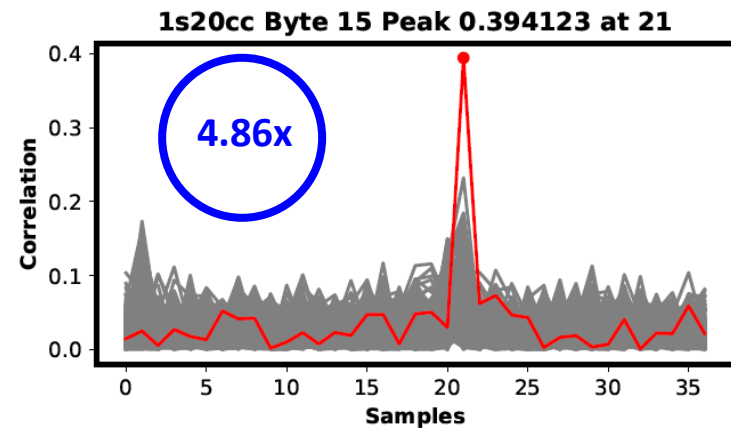
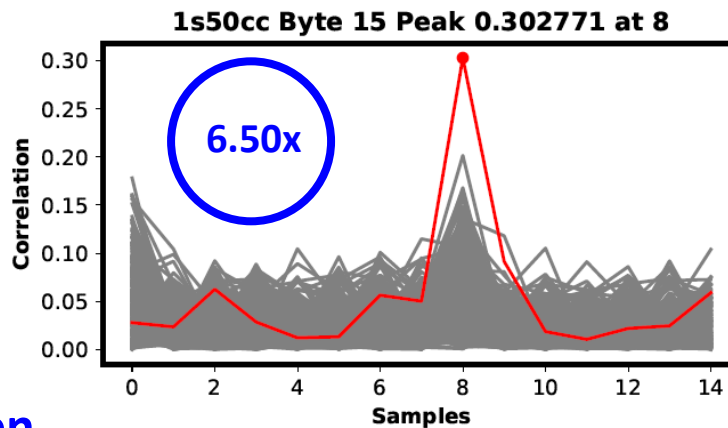
# Constructive Samples Example

- RISC-V 5-stage pipeline, bitwise AES SW, 1024 traces



# Constructive Samples Example

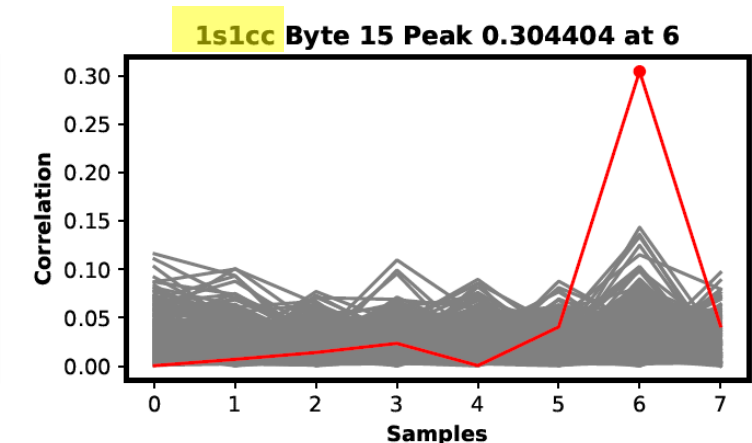
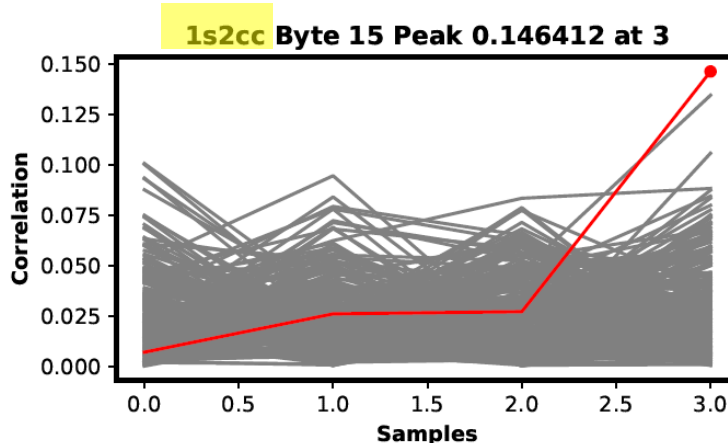
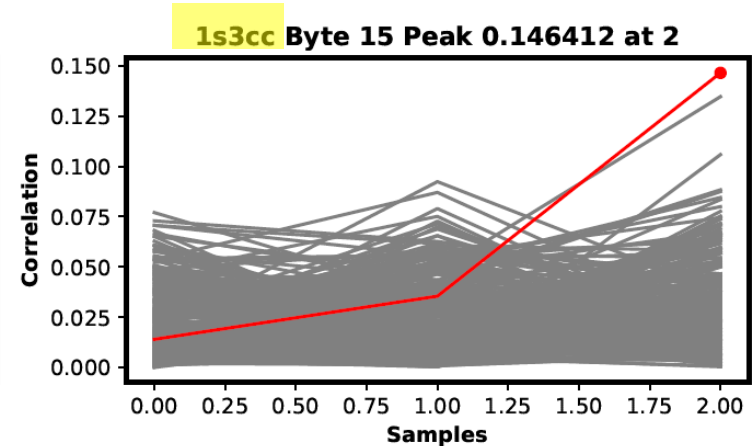
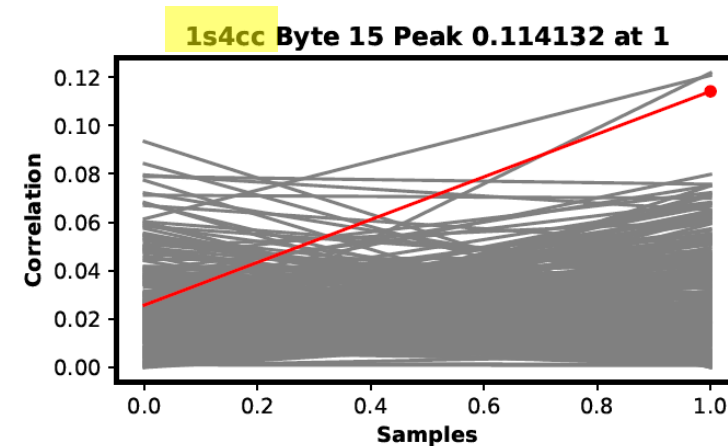
- RISC-V 5-stage pipeline, bitwise AES SW, 1024 traces



Power  
Simulation  
Speedup

# Destructive Samples Example

- Hardware AES, 5 cycles per round, 1024 traces

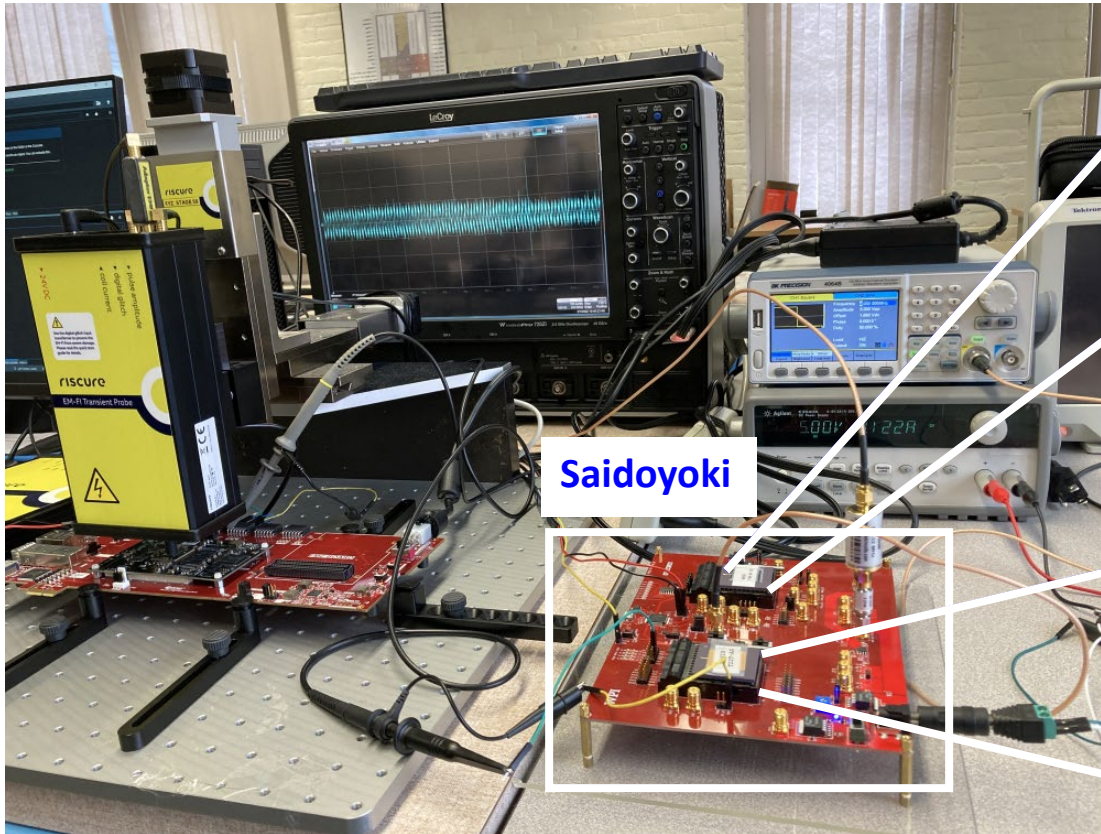


# Outline

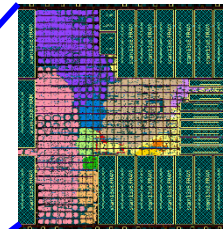
---

- Pre-silicon Tooling
- Background on Gate-level Power Modeling
- Architecture Correlation Analysis
- Applications and Tools
- What's next?

# Validation



Saidoyoki



FAMEv2  
LEON3  
SPARCV  
128K  
AES, Keymill  
Sensors  
TSMC180nm

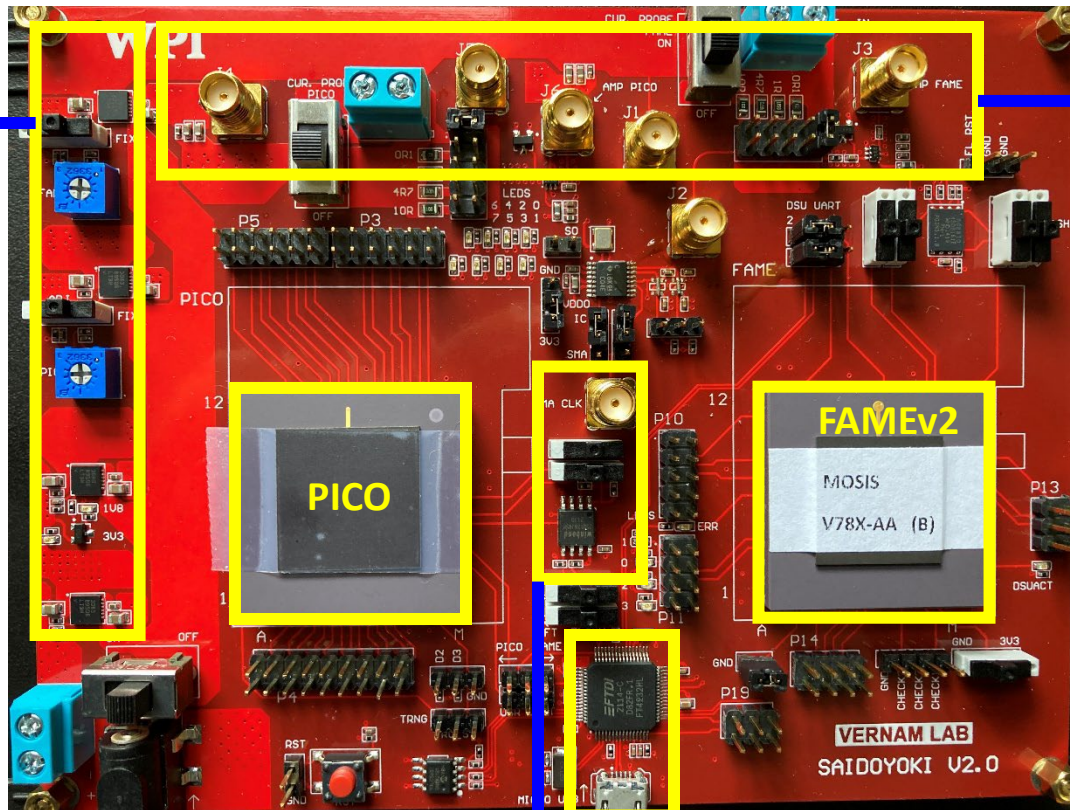


PICORV  
RISCV  
64K  
AES, ASCON,  
Sensors  
TSMC180nmA

# Saidoyoki

Power Supply

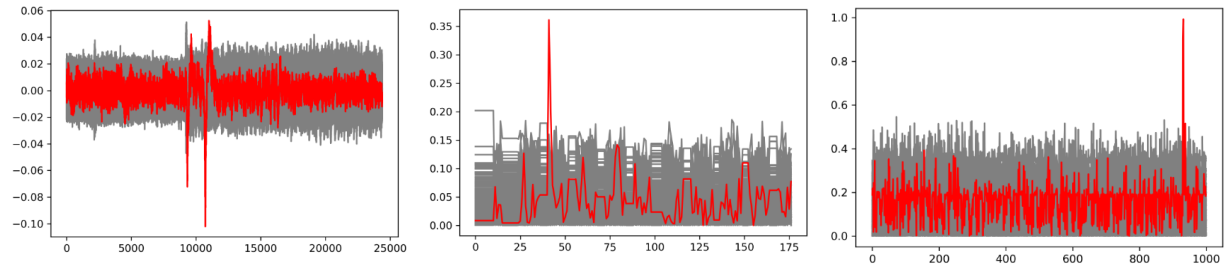
Power Measurement



Clock Generation

User I/O

# Saidoyoki DPA



	Exp 1	Exp 2	Exp 3
Pre/Post	Post	Pre	Pre
Target	FAME AES HW	PICO AES HW	PICO AES SW
Correlation	0.1	0.36	0.99
Traces	25,000	400	60
Samples/Cycle	4	16	1/80
Samples/Trace	24,400	2,00	1,000
Capture Time/Trace	0.06	0.55	260

# Future and Ongoing Research @WPI

---

- DARPA SCATE Flow
  - Design Automation for ACA as comprehensive script
  - Collaborator to Intrinsix/CEVA, Riscure, Purdue
- Back-annotation of leaky gates to software
  - *Explain* side-channel leakage to software programmers
  - Discover micro-architecture leakage
- Countermeasure development and integration
  - Protecting 5% of gates costs less than protecting all gates



# WPI

Thank you for your attention!

Questions, comments, ideas?

Patrick Schaumont  
pschaumont@wpi.edu  
@pschaumont