FPGA-Based Remote Power Side Channel Attacks

By Mark Zhao and G. Edward Suh

Presented by Maitreyi Ashok

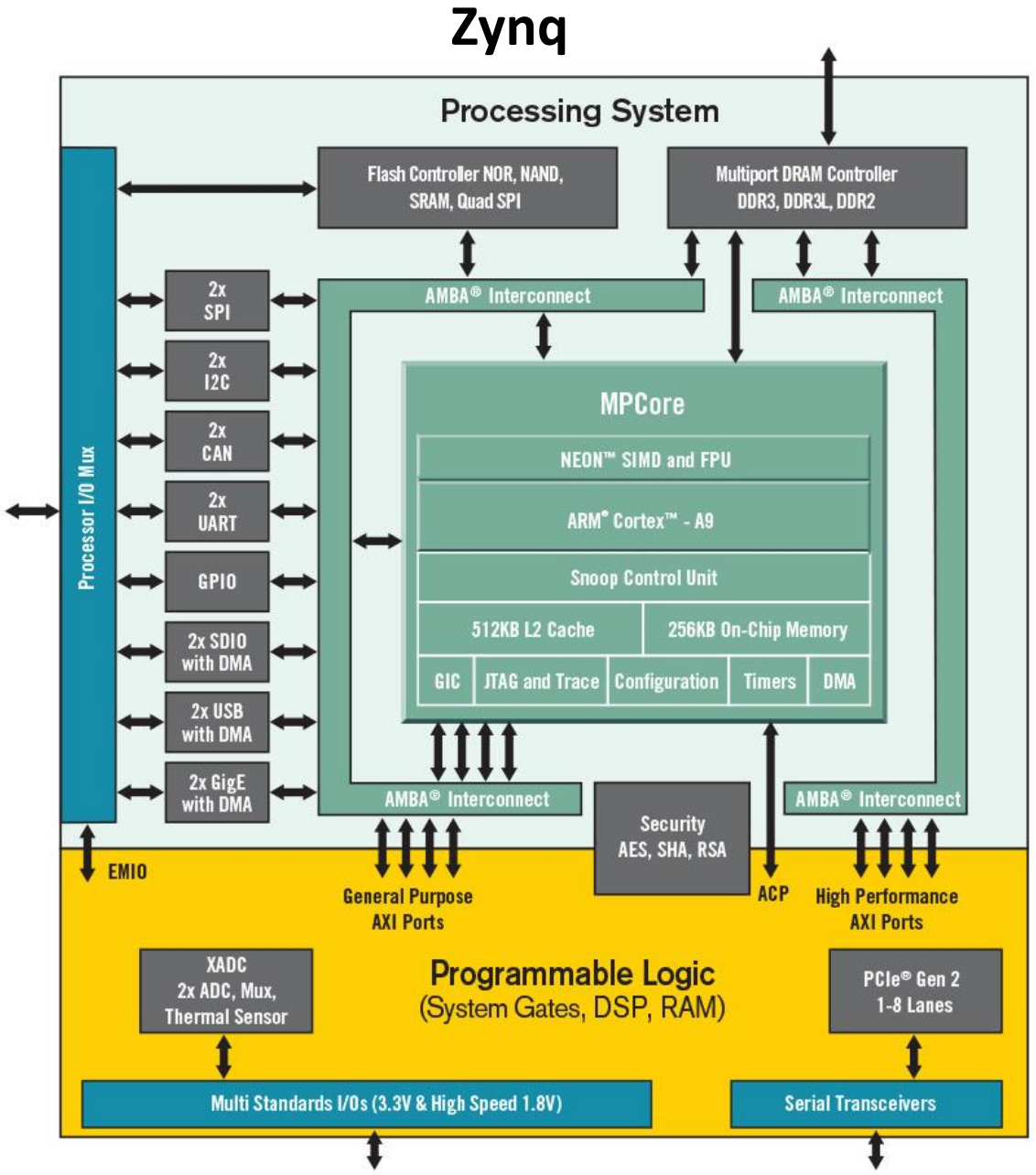
Motivation

- FPGAs are used in most cloud computing environments for hardware acceleration
- The SoCs used for these can have multiple users using different components on the same die

- Can this be used to perform a power side channel attack on other users on the same computing resources?

Threat Model

- Adversary can program a part of the integrated FPGA to implement any circuit
- Victim's hardware designs or programs are not secret (or can be reverse engineered)
- It is not necessarily required for attacker to be able to control place and route constraints
- Attacker and victim resources are physically and logically separated
- Only consider confidentiality attacks (no DoS or integrity)



The General Idea

- Remote power side channel attacks are possible using integrated FPGAs.
- A voltage variation monitor using ROs (introduced for other puposes) can be used to perform the attack.
- Demonstrates FPGA-to-FPGA and FPGA-to-CPU attacks.
- Discusses another possible method of power monitoring and potential countermeasures.

What This Does

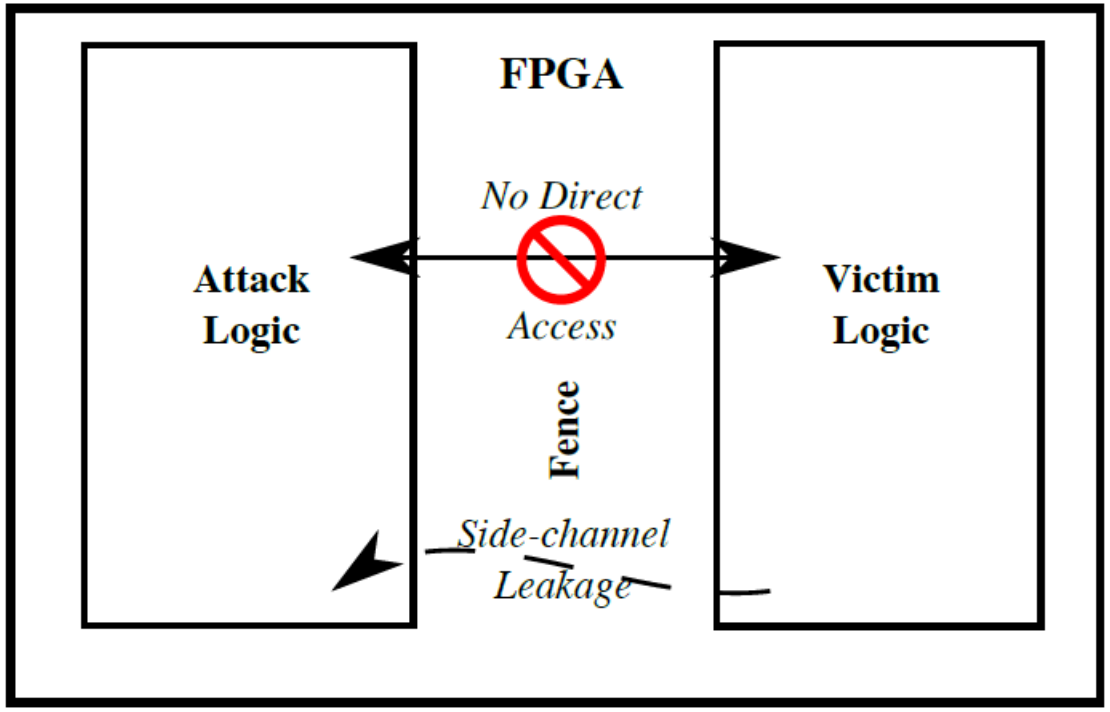
- Renders tamper proof board protections useless
- Blurs the software/hardware boundary since software programmers now have to consider power side channels
- With some more work, this can probably get past traditional SPA and even DPA countermeasures without needing more expensive equipment

Strengths and Weaknesses?

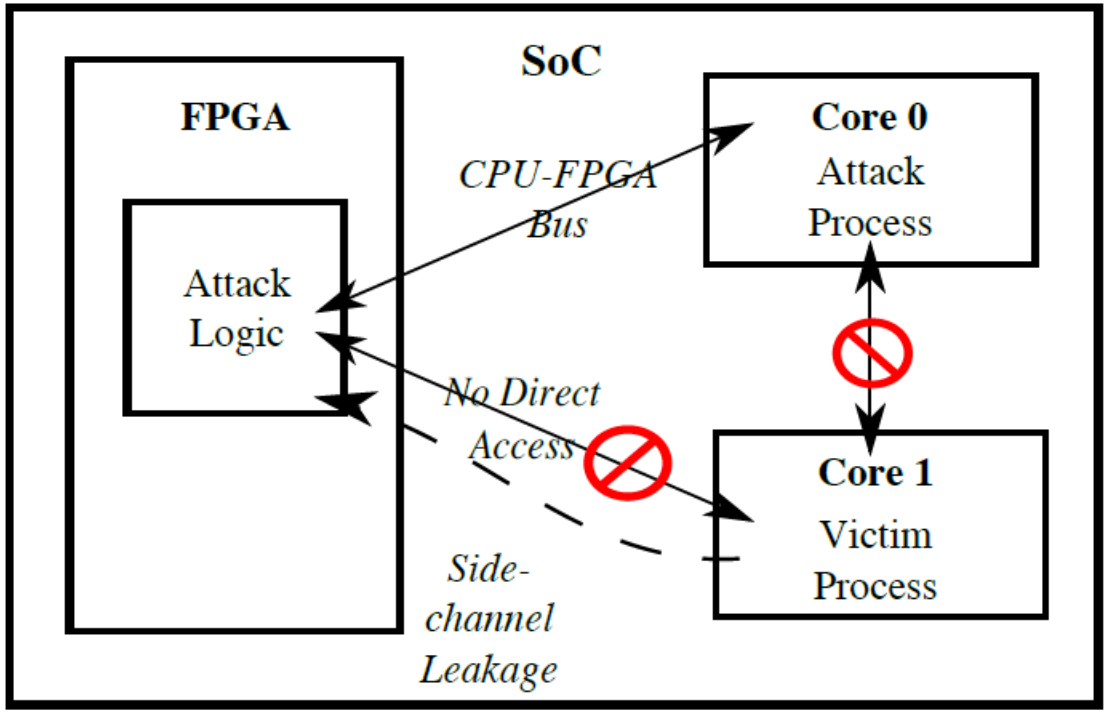
Strengths and Weaknesses?

- Strengths
  - **Remote** physical side channel attack – right now, most of the “remote” attacks are just EM probes
  - Finds a major vulnerability in integrated FPGA systems
  - Attack can be done even without access to P&R constraints
  - This might break some current methods of power side channel protection which equalize supply current at the pin
  - This can extend to other types of attacks pretty easily
- Weaknesses
  - Seems only practical for SPA due to limited voltage and time resolution
  - The noise considered seems pretty restricted
    - Can have other switching activity related to the plaintext/key

Attack Types



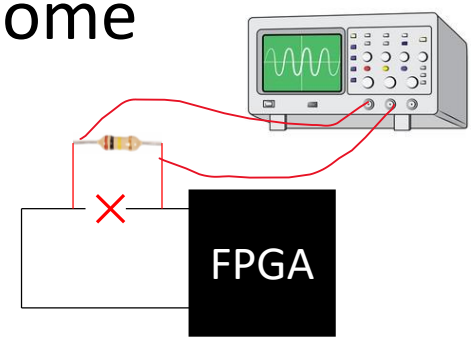
(a) FPGA-to-FPGA attacks on a shared FPGA.



(b) FPGA-to-CPU attacks on a heterogeneous SoC.

Background

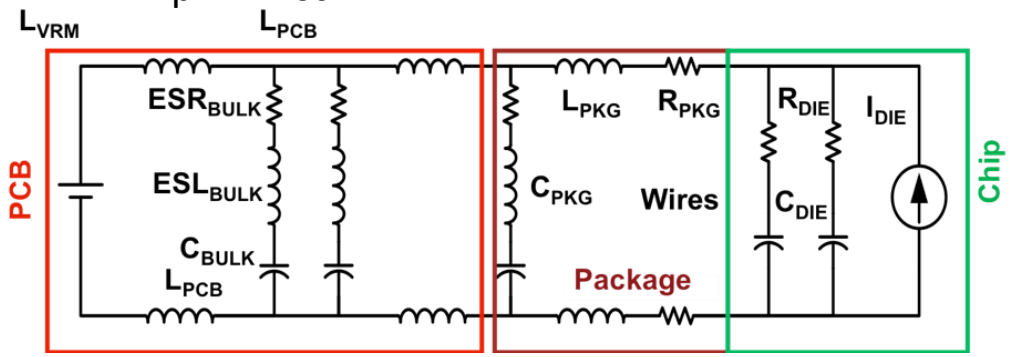
- General power side channels require physical access and some tampering



- Power Delivery Network
  - $P_{dyn} = P_{chrg} + P_{src} = \alpha * f * C_L * V_{DD}^2 + \alpha * f * V_{DD} * I_{pk} * t_{sc}$

$$V_{drop} = IR + L\frac{di}{dt}$$

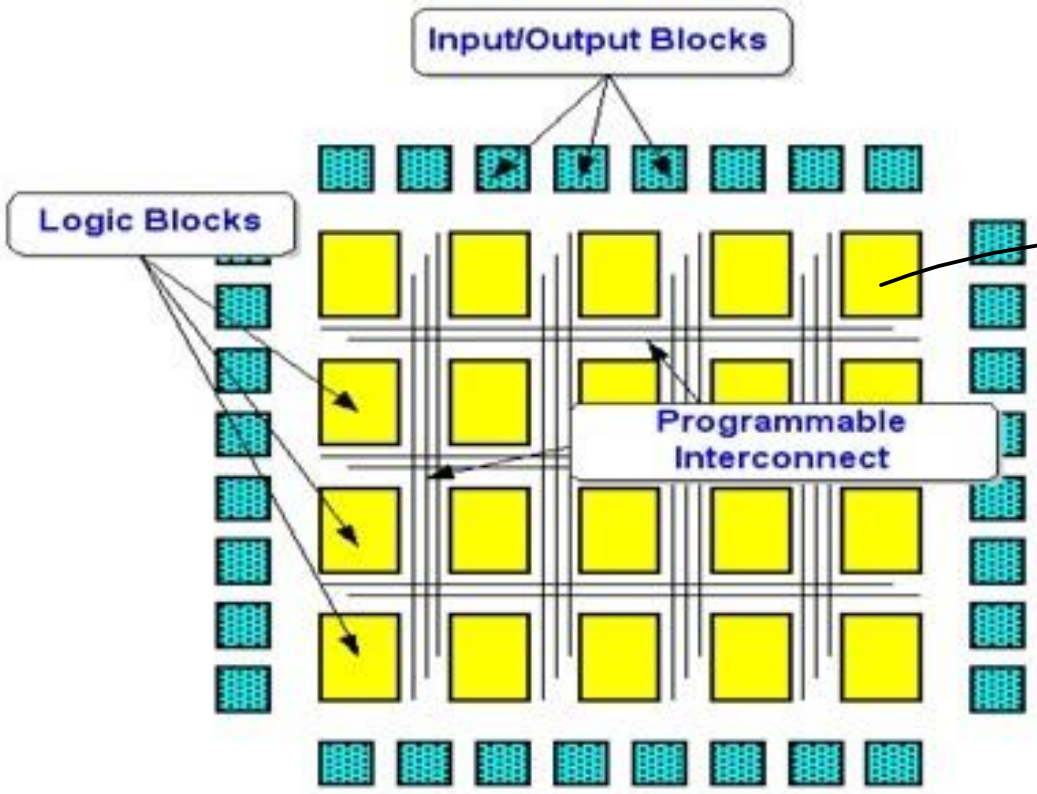
  - Logic delay ∝ 1/Voltage

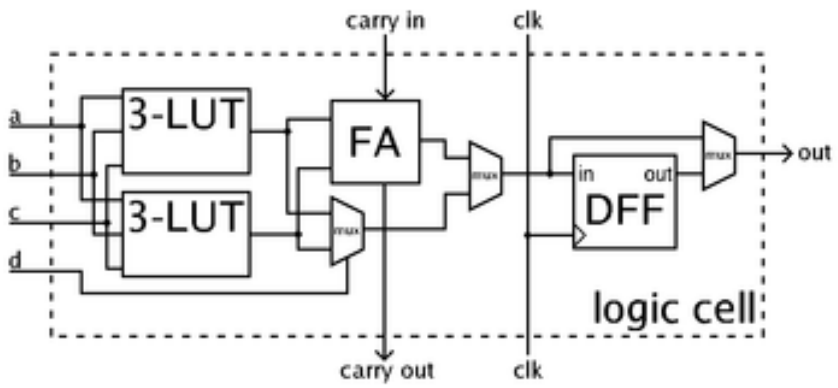


What is a FPGA?

Field Programmable Gate Array

Programmable hardware device that can be configured after it's manufactured

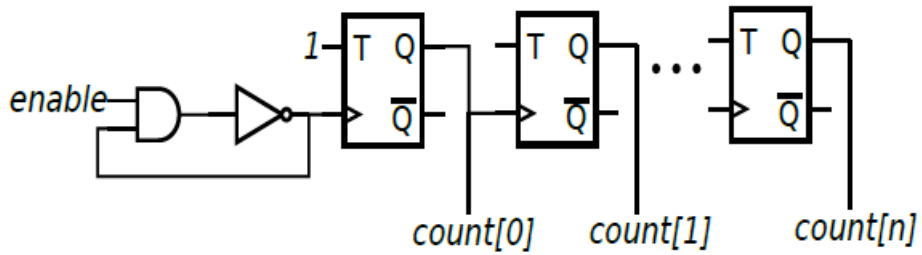




Can specify the interconnects (what gets routed to what) and the LUT equations (combinational logic truth table)

Ring Oscillator as a Power Monitor

- Measure combinational logic delay and use it to estimate power consumption



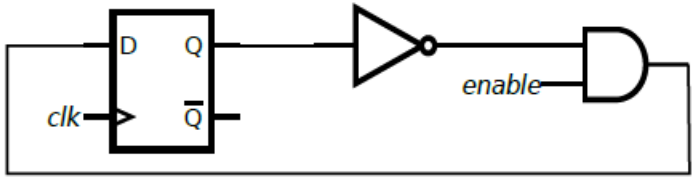
Average 20 counts for higher power resolution

$$f_{RO} = C_{RO} * \frac{f_{Ref}}{C_{Ref}} + \varepsilon$$

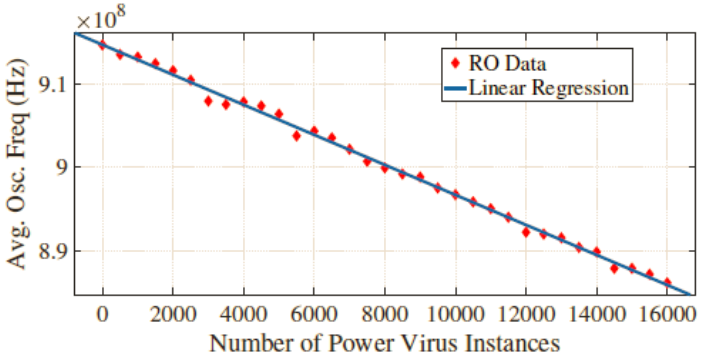
Poll Question

- How do the following parameters change as the sampling period increases (choose one of each row)?
  - Higher/Lower Maximum RO count
  - Better/Worse Power Resolution
  - Better/Worse Time Resolution
  - Higher/Lower Quantization error
  - Higher/Lower RO Frequency

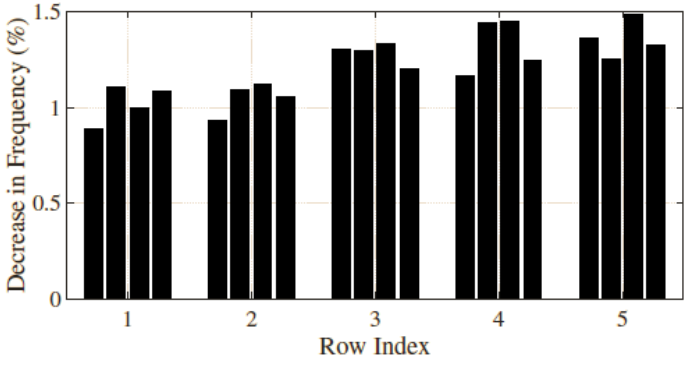
Power Monitor Experiments



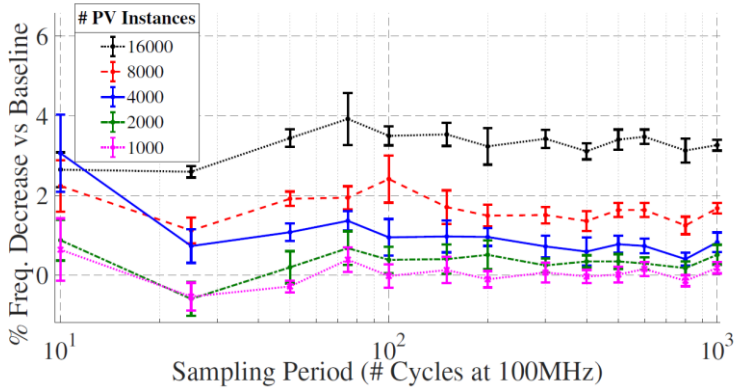
- Experiment 1
  - Use the power monitor to measure at different activity levels
  - **Almost linear relationship**



- Experiment 2
  - Characterize frequency change of ROs with respect to spatial proximity to switching logic
  - **CLB fences are not effective**



- Experiment 3
  - Characterize frequency change for different sampling periods (over various activity levels)
  - **For short periods, the linear relationship doesn't hold and there is more noise**

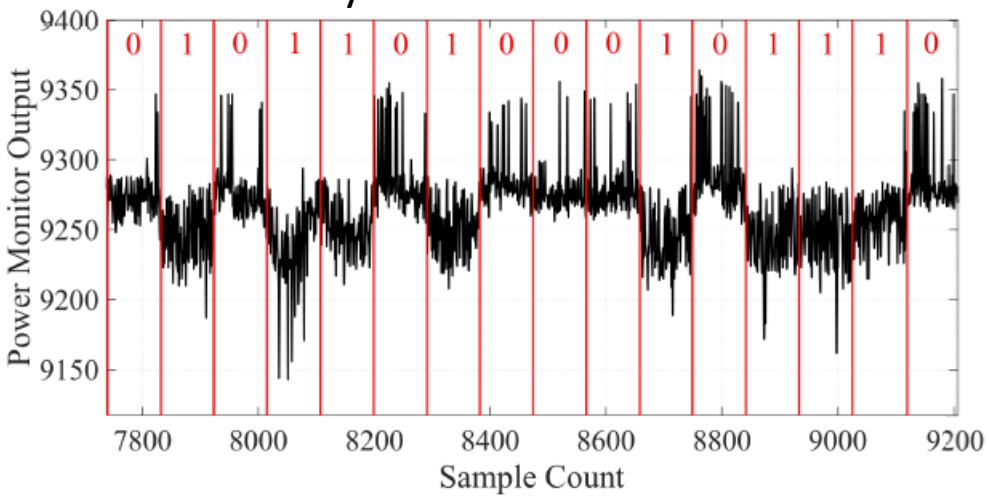


FPGA-to-FPGA RSA Attack

Dedicated modular multiplicand module for each

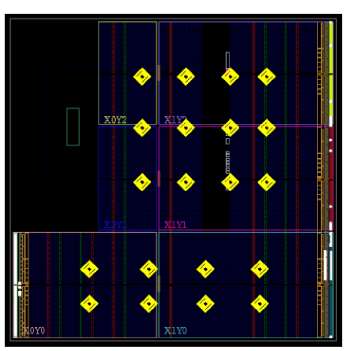
```
mod_exp(M, d, N)
{
  R = 1
  S = M
  for (i = 0 to n-1)
  {
    if (d mod 2 == 1)
      R = R*S mod N
    S = S*S mod N
    d = d >> 1
  }
  return R
}
```

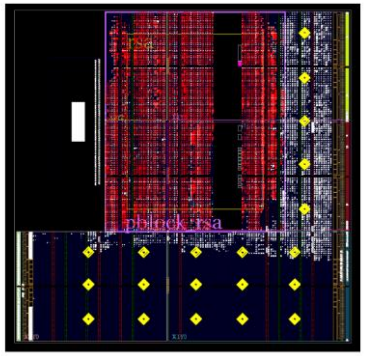
One modular exponentiation every 52.4 ms

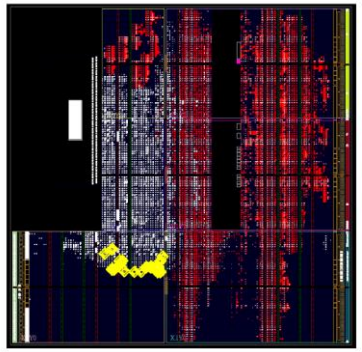


Subtract out static power

FPGA-to-FPGA RSA Attack







Key ID	PR		ISO		NoPR	
	Error	# Traces	Error	# Traces	Error	# Traces
1	3	3	81	27	30	6
2	3	3	178	27	33	6[1]
3	49	3	23	5	30	5
4	7	3	37	8	6	3
5	16	3[1]	44	3	67	18
6	40	4	32	5	76	22
7	49	5	5	3	43	8
8	2	5	30	5	148	22[1,2]
9	2	3	1	3	91	17[3]
10	18	5	1	3	49	7
Average	18.9	3.7	43.2	8.9	57.3	11.4

Effect of Noise

- Background spikes from other components can easily be removed

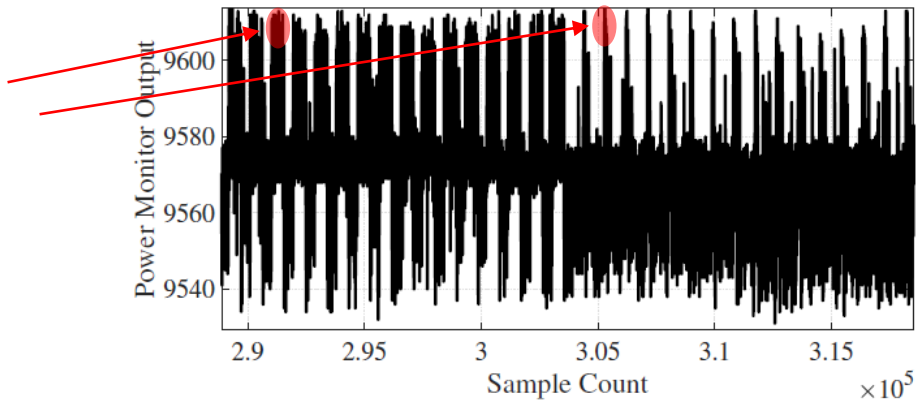
- Considered other FPGA activity with dynamic power from power virus instances near RSA cryptomodule
- Large constant noise can be dealt with since it's just a constant shift in the RO oscillation frequency
- For dynamic background noise, change attack to compute average RO frequency and compare that
  - Need to measure more power monitor traces
- At 8,192 power viruses, the SPA attack is not successful

FPGA-to-CPU Attack

- FPGA and CPU share power supply rails => Voltage drops will be seen across modules

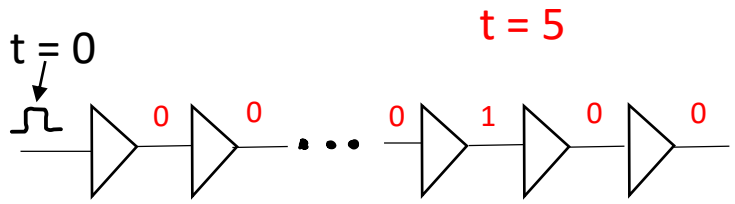
- Can distinguish between long and short strings input to strcmp function
  - Perhaps more interesting, it may be possible to observe non-user privileged operations like memory allocation, etc.
- Standard timing channel protections just delay the external output => Stalling has less power consumption so this technique doesn't work
- SPA on RSA w/ standard timing channel protection can easily be done

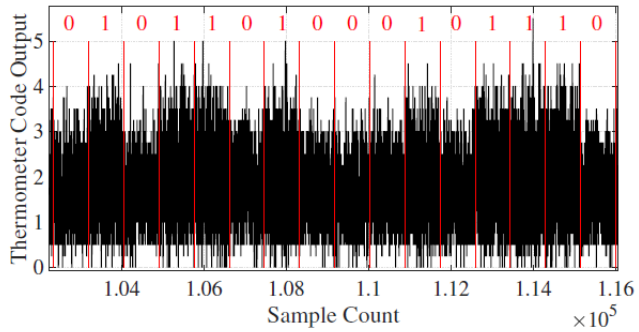
Stalling time larger when bit is 0



Delay Line Power Monitor

Distance along chain ∝ Propagation delay ∝ 1/Voltage





RO	Delay line
Sampling frequency must be low enough to get decent power resolution	Sampling frequency as high as clock frequency
Resolution can be increased by increasing sampling period	Resolution can't be adjusted dynamically
Enough power resolution for RSA SPA	Enough power resolution for RSA SPA
Simpler to implement and requires less customization	More complex to implement and sensitive to placement and routing

Potential Countermeasures

- Victim logic is more resilient to PSCA
  - Random noise, dummy operation, homomorphic encryption, etc.
  - Performance and energy overhead
  - Some can’t be implemented on FPGAs
- Make it difficult to construct power monitoring circuits
  - System admin checks FPGA design, some netlist analysis is done, P&R constraints prevented, etc.
  - Circuits used for power monitor have legitimate uses
  - Attacker can design a monitor to bypass constraints

- *Traditional Power Side Channel Countermeasure*
  - *Equalize power at the pin with either a constant current supply or switched capacitor method*
  - *Can use power monitor circuits to bypass this since the ROs are on the same voltage line as the other circuitry*
- My Countermeasure (maybe too simple)
  - Separate PDN for CPU and FPGA (and maybe even different pblocks of the FPGA that would be allocated to different users)

Discussion Questions (Practicality of the Attack)

- RSA has been repeatedly given as an example of an algorithm that is susceptible to power side channels, but is this a common exploitable property? Does real-world code often have parts that could be taken advantage of by monitoring power?

- How often are cloud SoCs multiplexed between users? This seems impractical, unless maybe taking advantage of partially reconfigurable FPGAs.

- Doesn’t the problem go away if FPGAs/resources are not shared?

Countermeasures

- If SoCs are not multiplexed, the main untrusted party is the cloud provider. Are there designs for FPGA "enclaves" where a verified hash of the bitstream ensures users the provider is not spying?

Discussion Questions (Extending the Attack)

- Can the system bus accessing DRAM as a delay line be used for a similar power monitor based attack?
- Can power side channels be used offensively to momentarily bring voltage rails out of spec and cause glitches in other processes?

- Are there other structures in standard FPGAs outside of delay lines and ring oscillators that could be similarly exploited?
- Could we use "non-suspicious looking" RTL designs (rather than ring oscillators) to perform a similar analysis?