7: Physical Unclonable Functions (PUF)

Yunsi Fei

Northeastern University
ECE Department

Apr. 7th, 2020

What Do We Want to Achieve for Security?

Authentication
no spoofing

Data Integrity
no data alteration

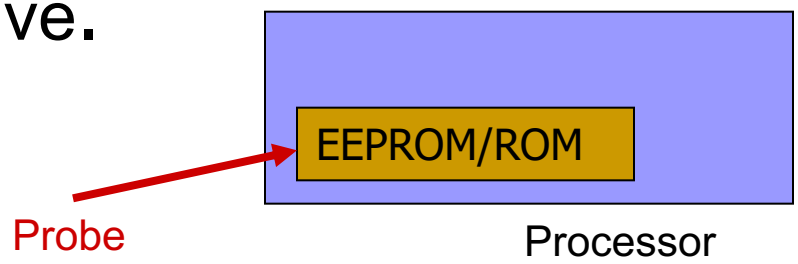
Privacy
no eavesdropping

Secure and reliable identification, authentication, integrity checking, and confidentiality preservation of the systems being protected.

Device Authentication

Secret Digital Keys

Storing secret digital information in a device in a way that is resistant to physical attacks is difficult and expensive.



- The vulnerability lies in the hardware implementation and key storage
  - Non-invasive attacks can extract the secret by side-channel attacks while the processor is on
  - Invasive attacks can physically read keys from EEPROM while the processor is off
    - EEPROM adds additional complexity to manufacturing

Existing Approaches

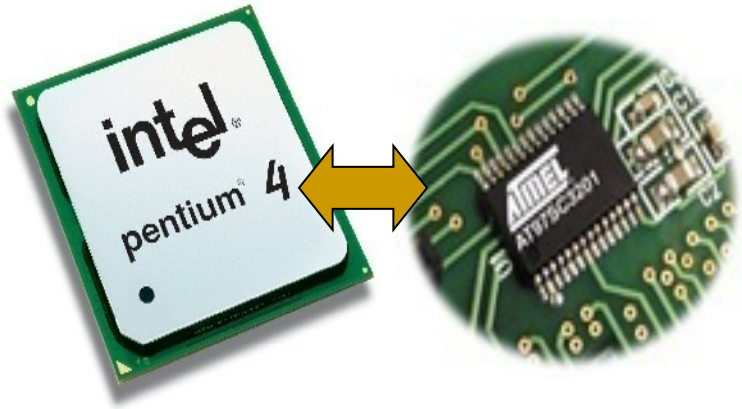
IBM 4764

Tamper-proof package
containing a secure processor
With a secret key and memory
- Tens of sensors, resistance, temperature, voltage, etc.

Continually battery-powered

~ $3000 for a 99 MHz processor and 128MB of memory

Trusted Platform Module (TPM)



Solution – PUF (Physical Unclonable Function)

- Use the chaotic physical structures that are hard to model instead of digital secret

- Physical Unclonable Function
  - Inexpensive to fabricate
  - Prohibitively difficult to duplicate
  - No compact mathematical representation
  - Intrinsically tamper-resistant

Manufacturing Feature

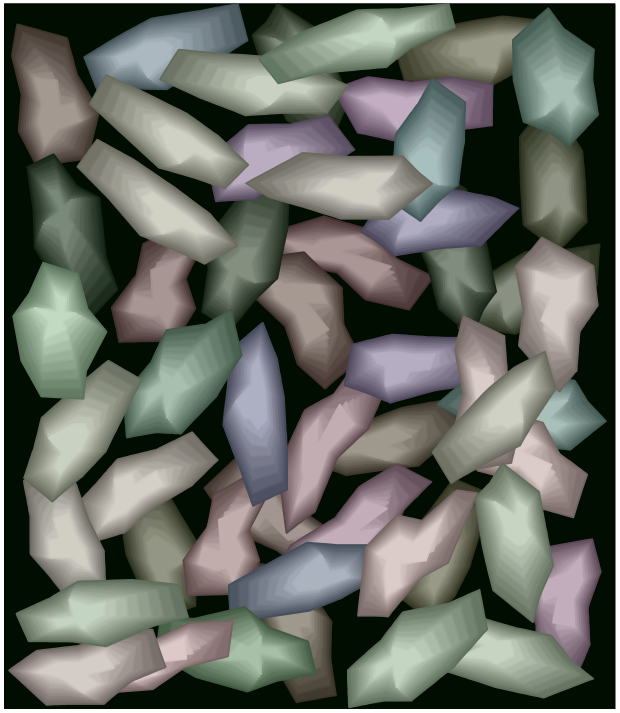
- Process variation
  - oxide thickness variation: non-uniform condition/doping during fabrication
  - circuit dimension (length and width) variation: limited resolution of the lithography process results
- Impact
  - I_{DDT}, I_{DDQ}
  - Circuit performance (delay), leakage power
  - Introduce functional failure
  - Major obstacle to the continued scaling of integrated-circuit technology in the sub-45 nm regime

Solution

Motivation: Process variations can be turned into a feature rather than a problem?

Each IC has unique properties

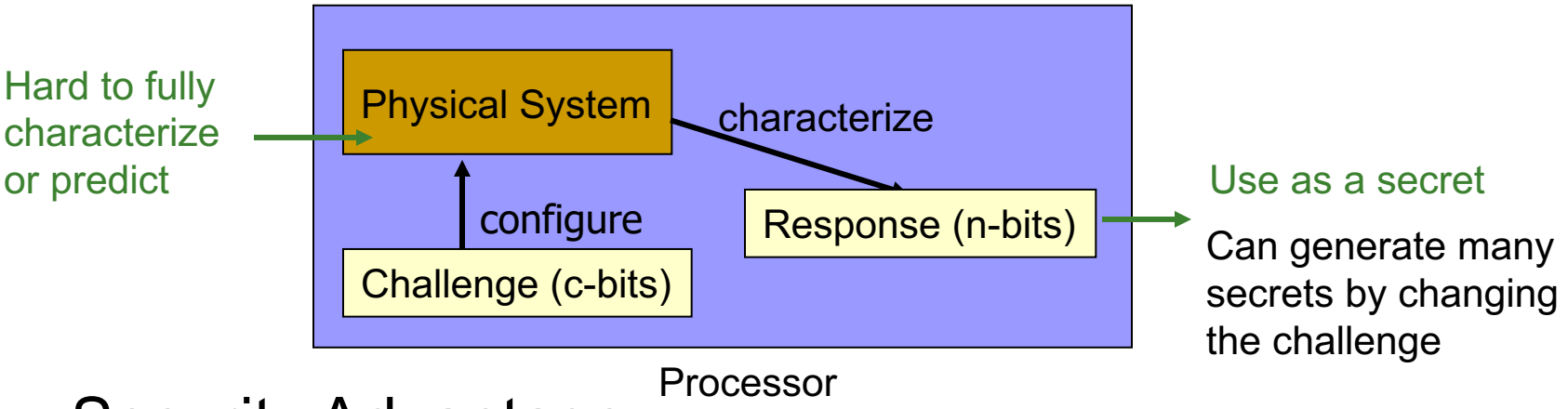
Extract key information from a complex physical system.



Devadas, et. al, DAC02

Physical Random Functions (PRF)

- Generate inherent/independent IDs for different devices
- Generate keys from a complex physical system



- Security Advantage
  - Keys are generated on demand → No non-volatile secrets
  - No need to program the secret
  - Can generate multiple master keys
- What can be hard to predict, but easy to measure?

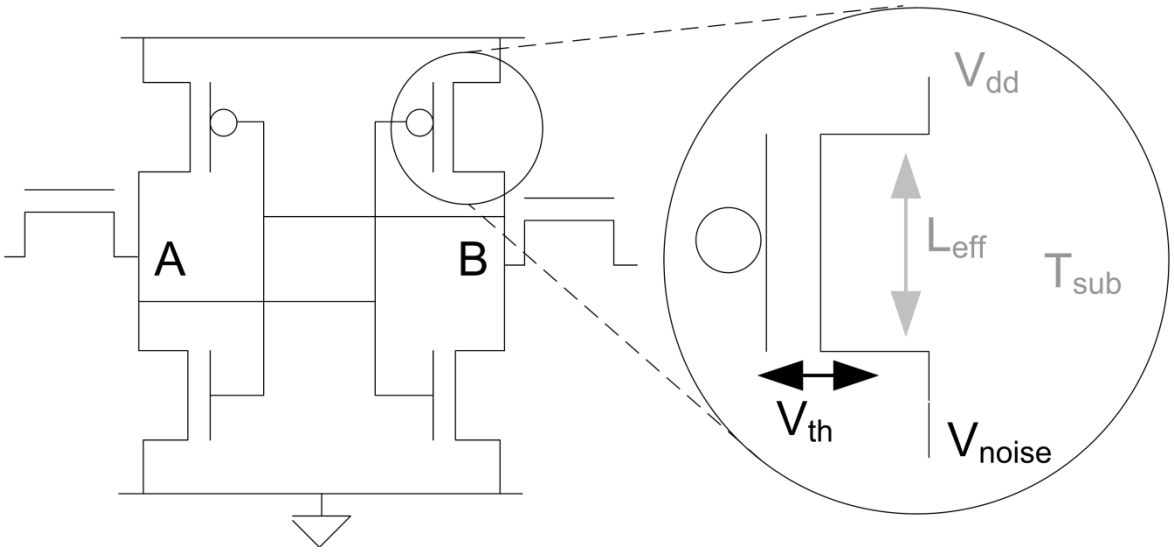
PUF Definition

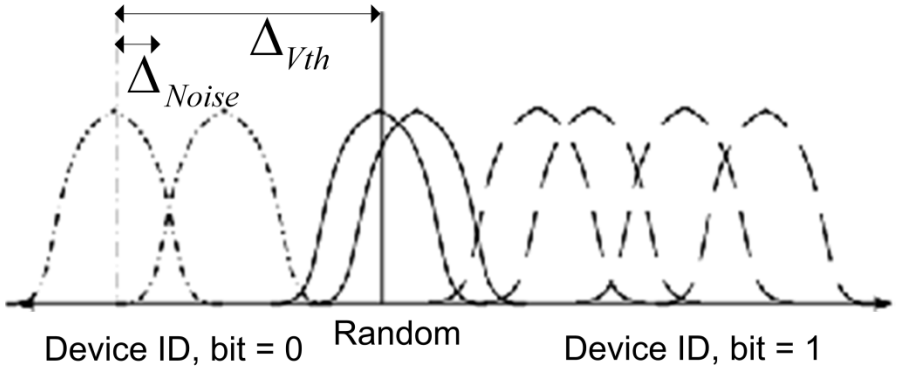
- A Physical Random Function or Physical Unclonable Function (PUF) is a function that is:
  - Based on a physical system
  - **Easy to evaluate (using the physical system)**
  - Its output looks like a random function
  - Unpredictable even for an attacker with physical access
- Categories
  - Weak PUFs – Physical obfuscated keys (POK)
  - Strong PUFs
  - Controlled PUFs

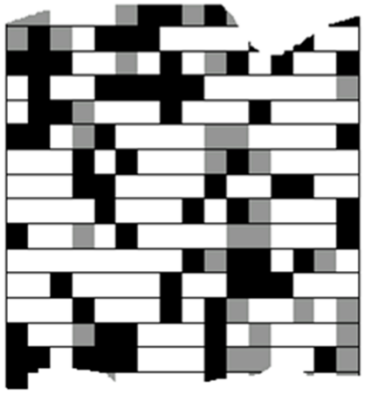
Weak PUF

- Limited number of challenge-response pair, used for physically obfuscated keys
- Examples
  - ICID based on process variation
  - POKs – physically obfuscated key (the response cannot be measured, remains secret)
  - SRAM-based PUF
  - Butterfly PUF
  - Coating PUF (combined with tamper resistance)
  - Resistive PUF (power distribution network)

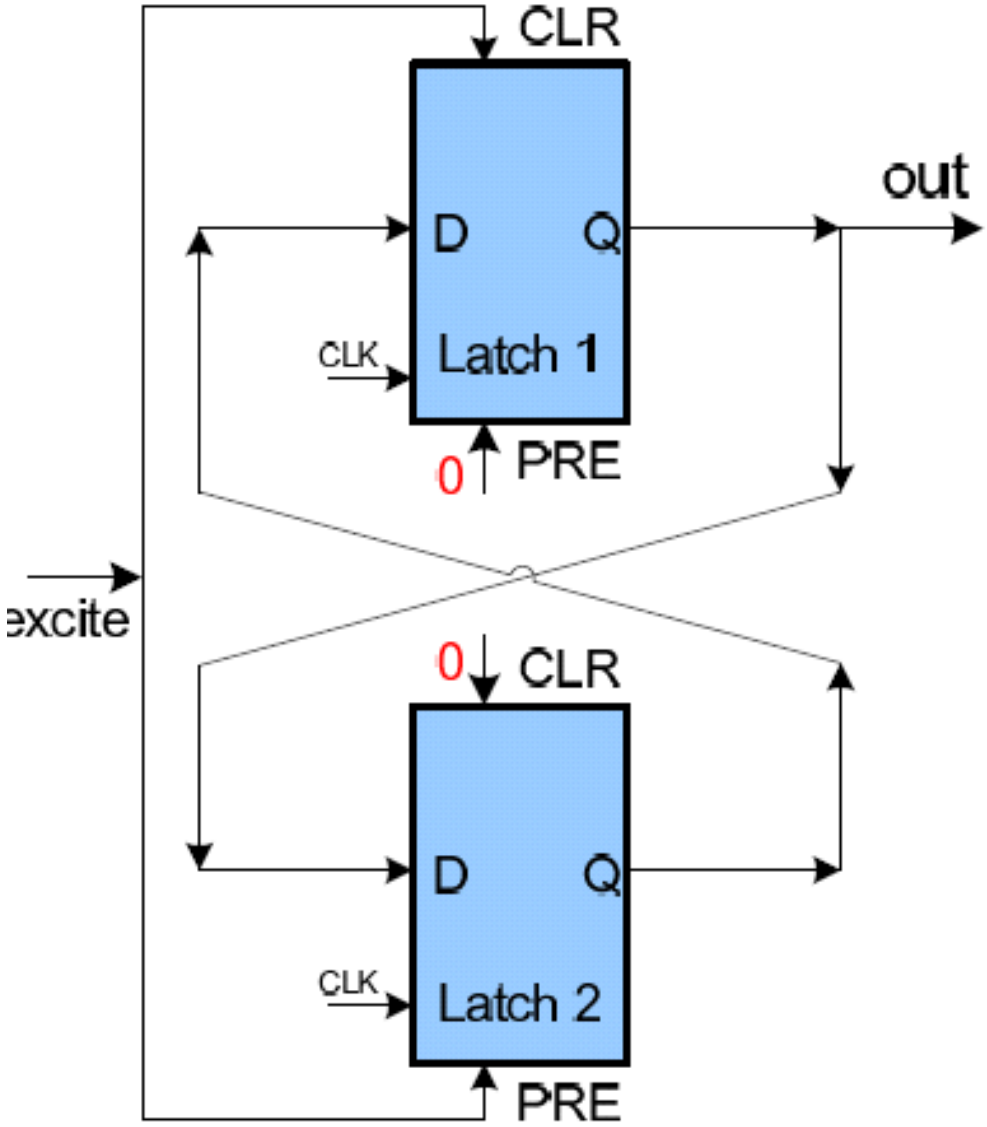
SRAM-based PUF [4]







Butterfly PUF (Any FPGA)



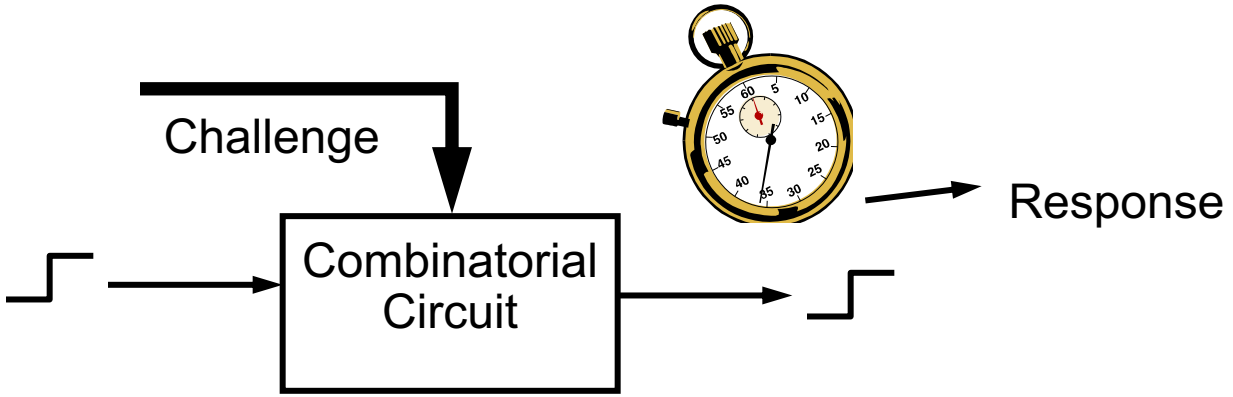
Strong PUFs

- Strong PUFs have many possible challenge-response pairs
- Strong PUFs' responses can be measured and recorded for authentication
- Examples:
  - Optical PUF – physical one-way function, secure but practicality and stability issues
  - Silicon PUF based on arbiter
    - Non-linear variants
  - Silicon PUF based on ring oscillator
  - Analog PUFs

Silicon PUF – Proof of Concept

- Because of process variations, no two Integrated Circuits are identical

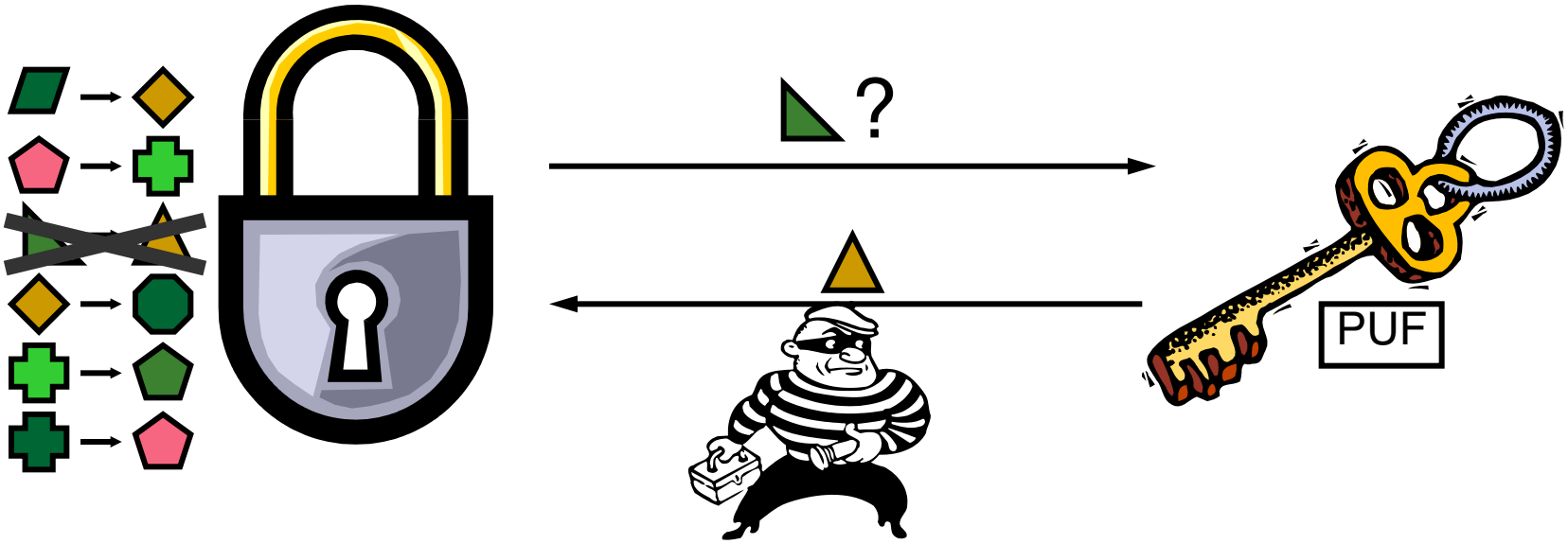
- Experiments in which *identical circuits with identical layouts* were placed on different FPGAs show that path delays vary enough across ICs to use them for identification.



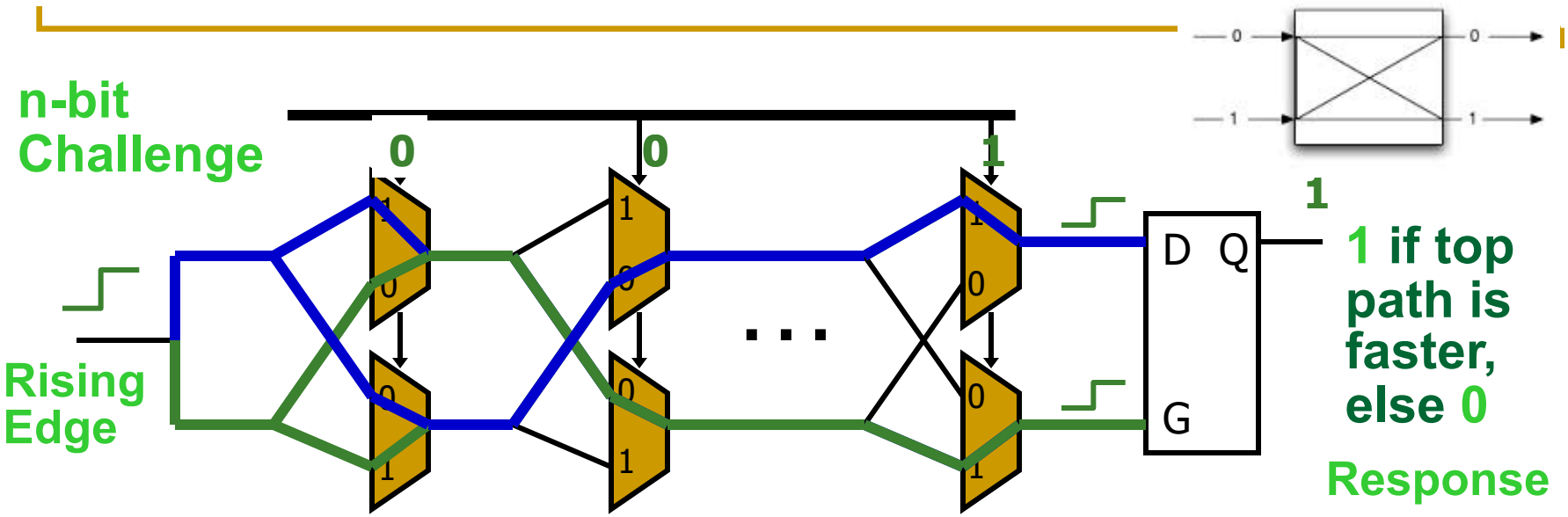
Using a PUF as an Unclonable Key

A Silicon PUF can be used as an unclonable key.

- **The lock has a database of challenge-response pairs.**
- **To open the lock, the key has to show that it knows the response to one or more challenges.**



A Candidate: Silicon PUF [1] [2]



- Compare two paths with an identical delay in design
  - Random process variation determines which path is faster
  - An arbiter outputs 1-bit digital response
- Path delays in an IC are statistically distributed due to random manufacturing variations

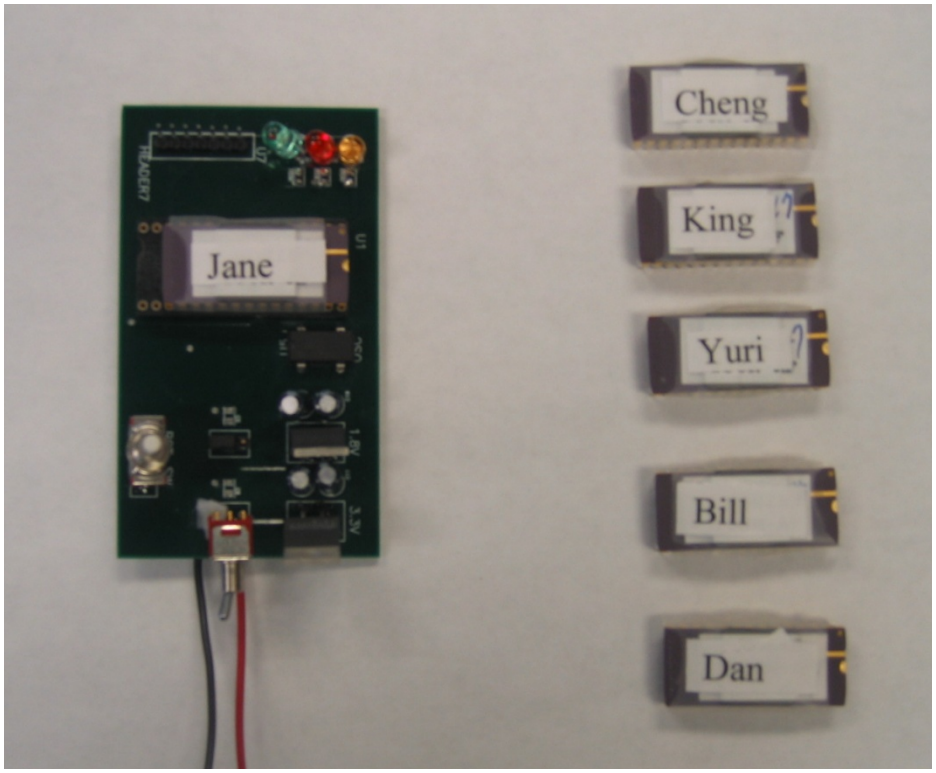
Reliability and Security Metrics

- *Inter-chip variation:* How many PUF output bits are different between PUF A and PUF B? This is a measure of uniqueness. If the PUF produces uniformly distributed independent random bits, the inter-chip variation should be 50% on average.

- *Intra-chip (environmental) variation:* How many PUF output bits change when re-generated again from a single PUF with or without environment changes. This indicates the reproducibility of the PUF outputs. Ideally, the intra-chip variation should be 0%.

PUF Experiments

- Fabricated 200 “identical” chips with PUFs in TSMC 0.18μ on 5 different wafer runs



Security (uniqueness)

– What is the probability that a challenge produces different responses on two different PUFs?

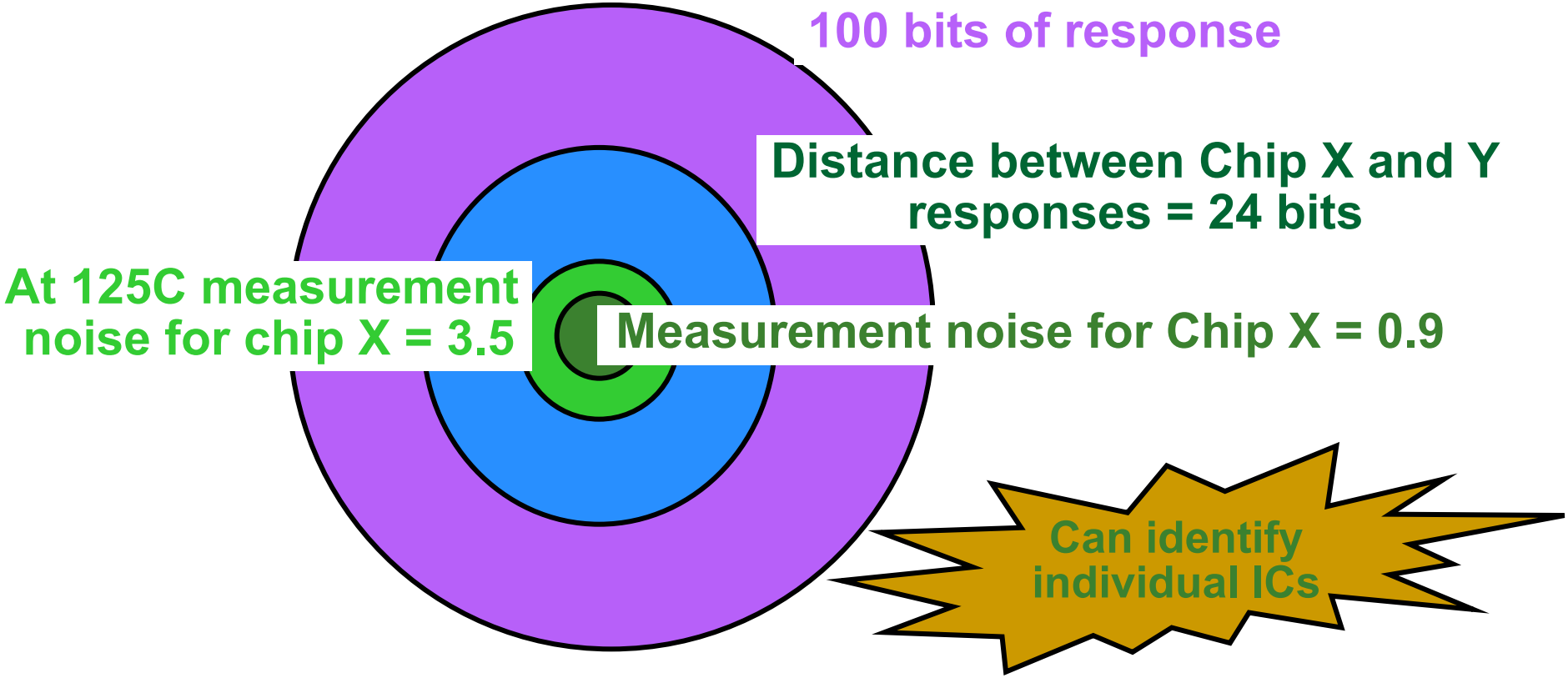
Reliability

– What is the probability that a PUF output for a challenge changes with temperature change?

– With voltage variation?

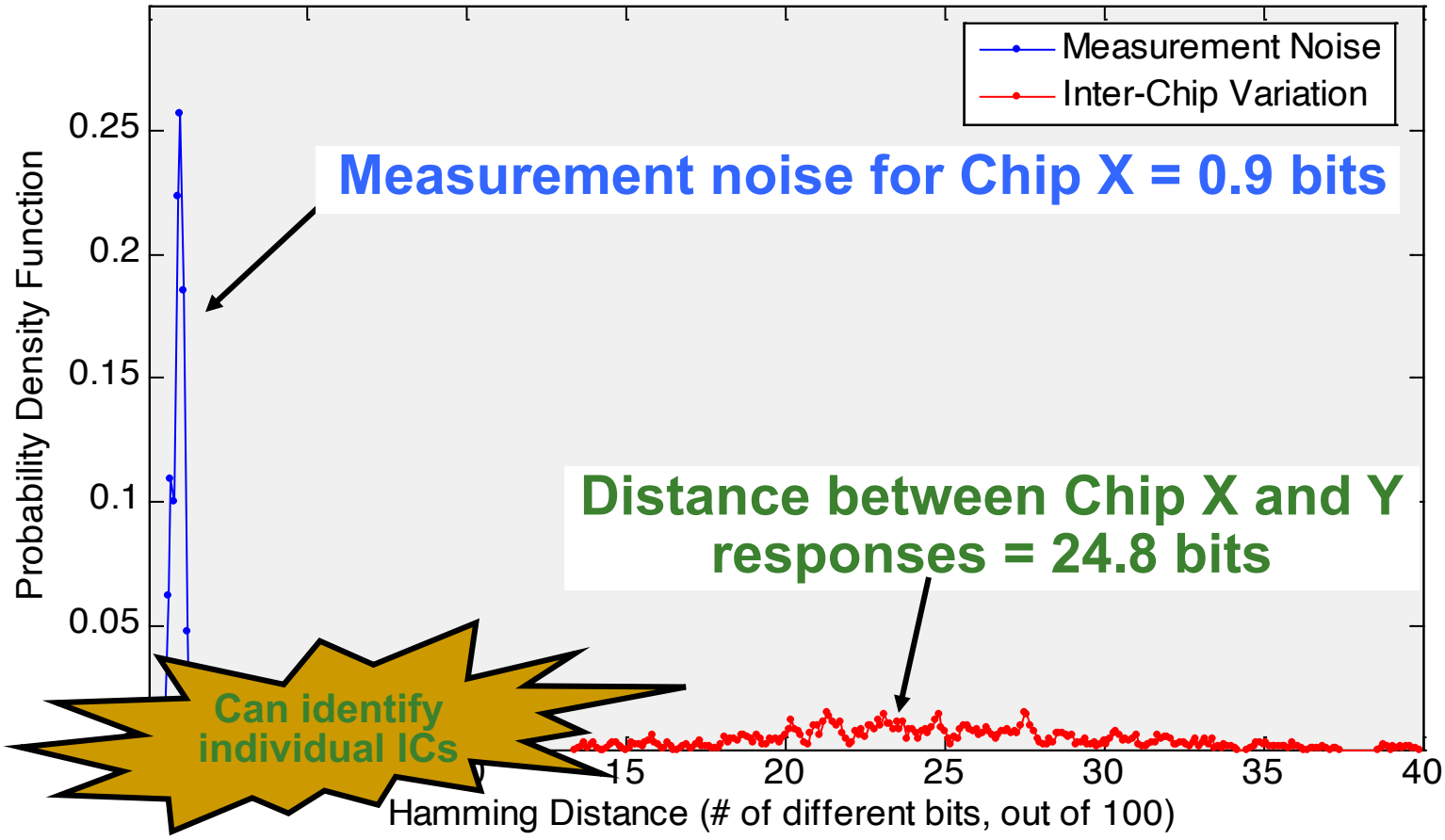
Experiments

- Apply 100 random challenges and observe response



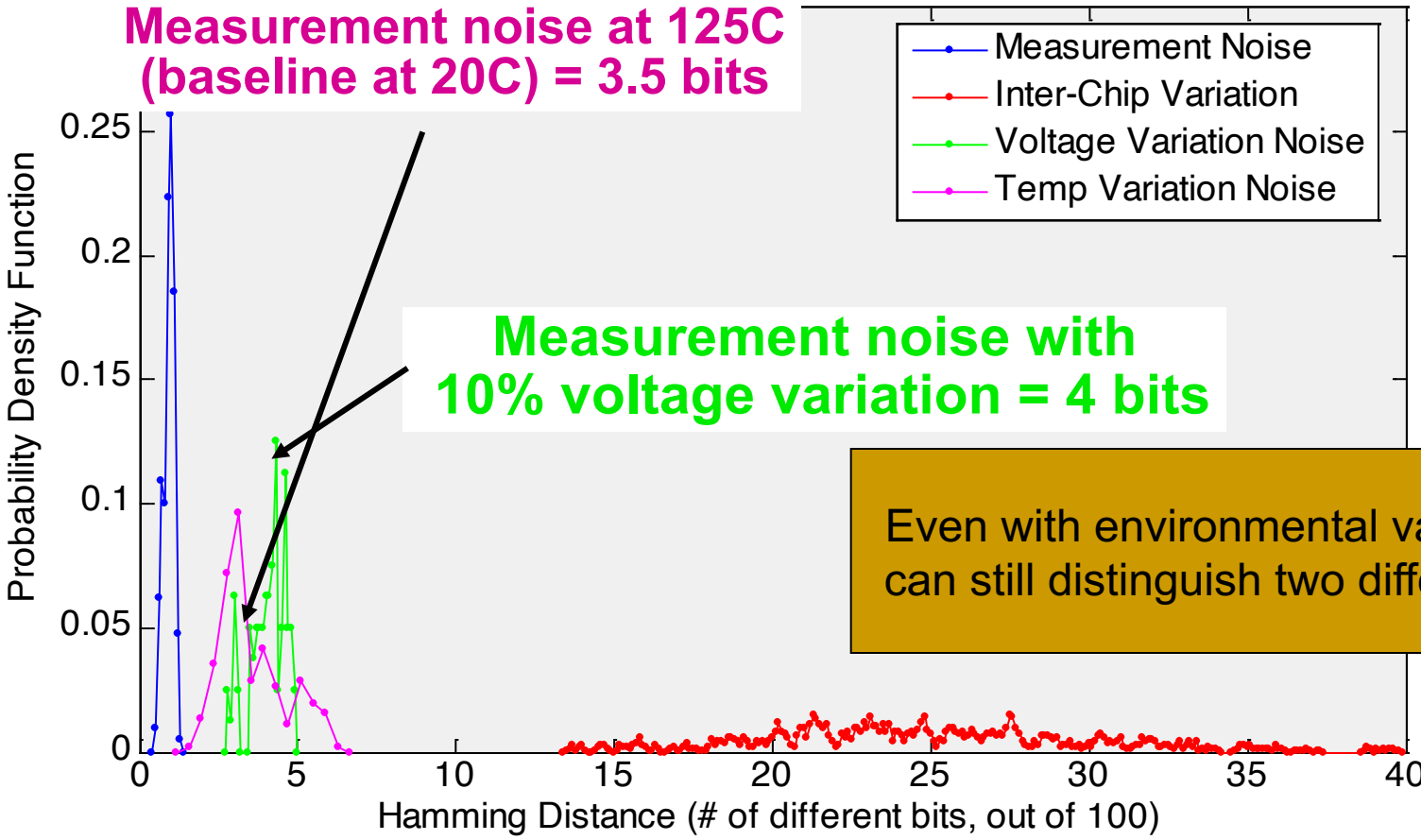
Inter-Chip Variation

- Apply random challenges and observe 100 response bits



Intra-chip Variations (Temporal Environmental)

- What happens if we change voltage and temperature?

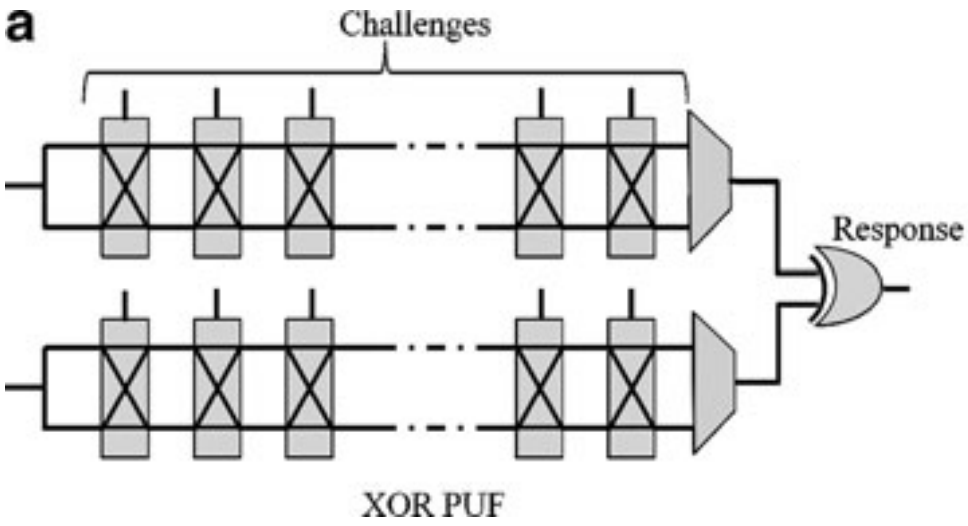


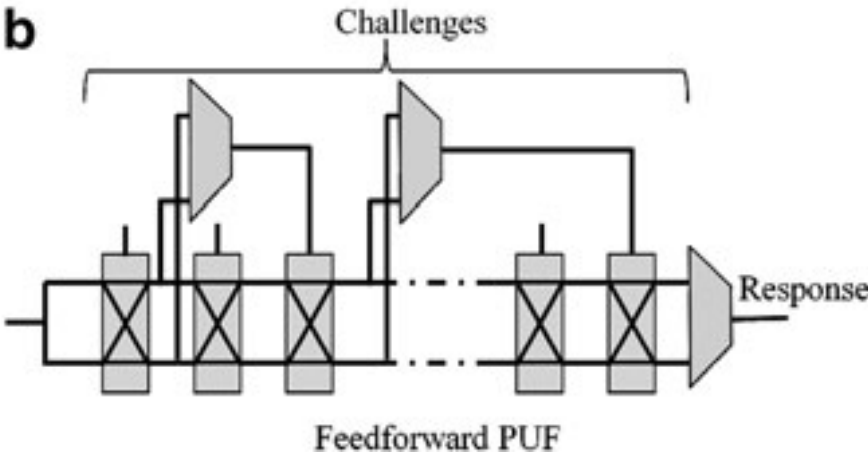
Even with environmental variation, we can still distinguish two different PUFs

Attacks on Silicon PUFs

- Duplication
  - Barrier: due to statistical variation, the adversary has to fabricate a huge number of IC's and precisely characterize each one to discover the counterfeit
- Model building using direct measurement
  - Barrier: Make PUF delays depend on overlaid metal layers and package
    - Invasive attack (e.g., package removal) changes PUF delays and destroys PUF, tamper-evident
- Model building using adaptively-chosen challenge generation
  - Barrier: Creating timing model (10%) accurate within measurement error (0.1%) is difficult
    - Wire delay is not a number but a function of challenge bits and adjacent wire voltages

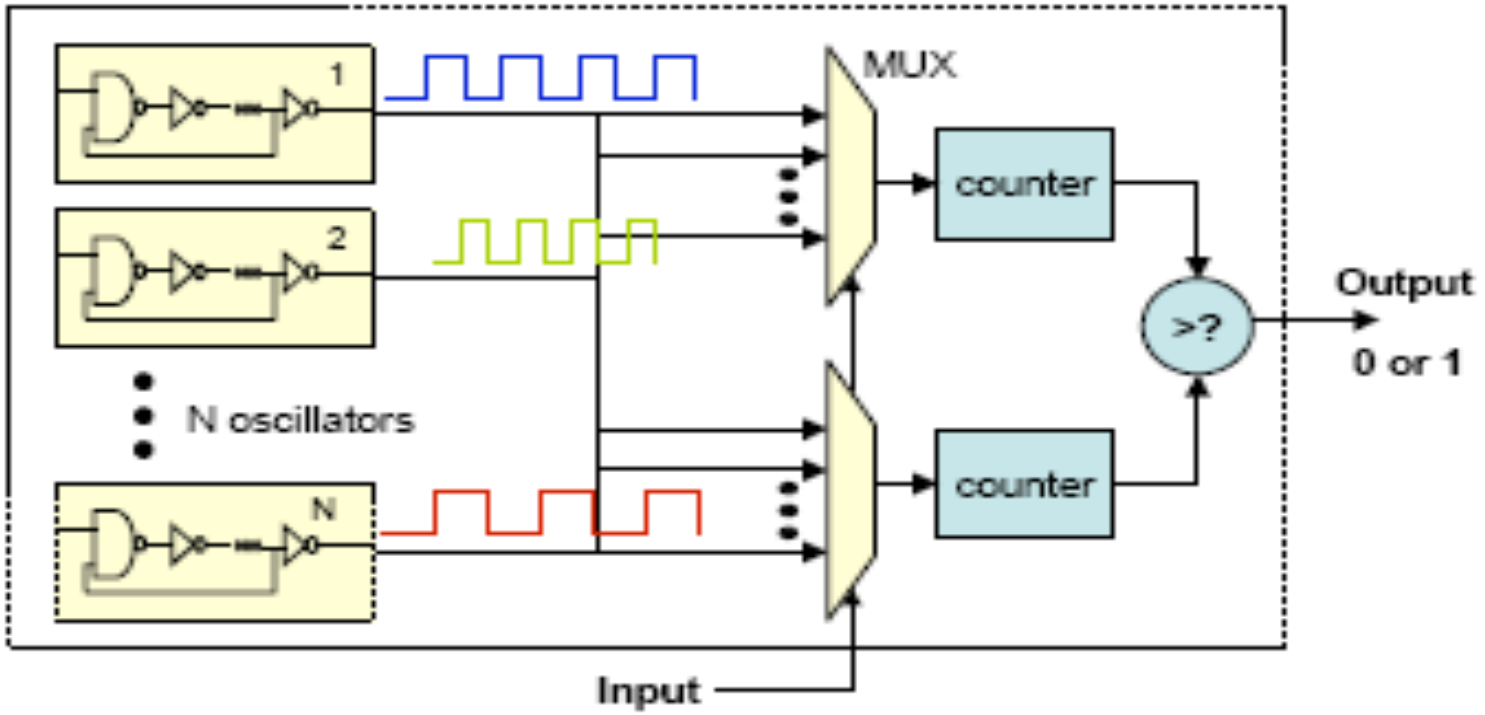
Non-linear Arbiter-based PUFs





Ring-Oscillator (RO) PUF [2]

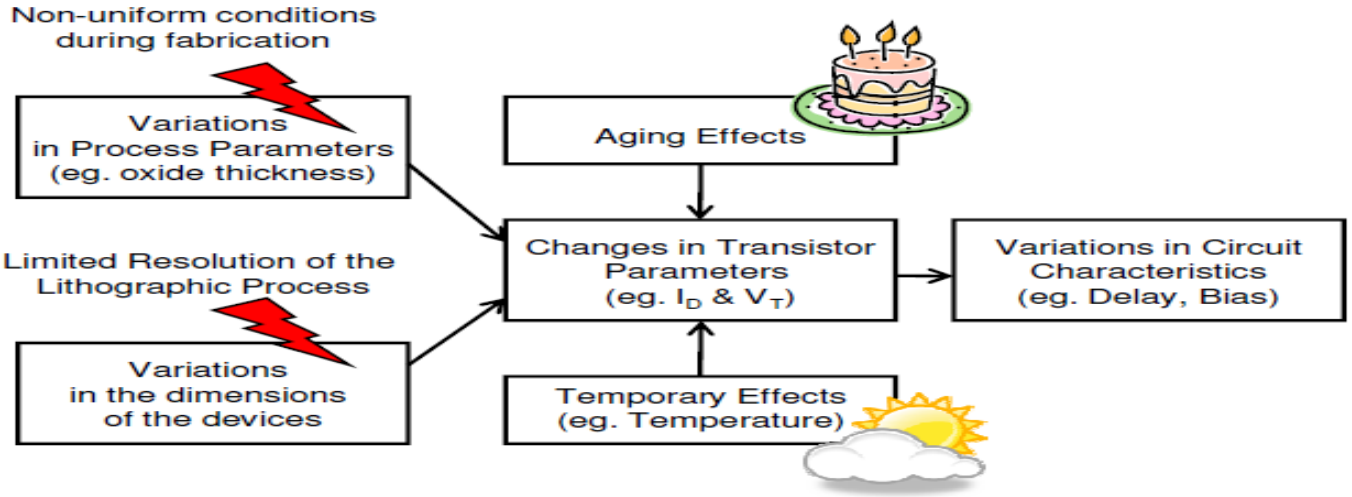
- The structure relies on delay loops and counters instead of MUX and arbiters
- Better results on FPGA – more stable



RO PUFs (cont'd)

- Easy to duplicate a ring oscillator and make sure the oscillators are identical
  - Much easier than ensuring the racing paths with equal path segments
- How many bits can we generate from the scheme in the previous page?
  - There are N(N-1)/2 distinct pairs, but the entropy is significantly smaller because the outputs are correlated: $\log_2(N!)$
  - For example:
    - 35 ROs can produce 133 bits
    - 128 can produce 716 bits
    - 1024 can produce 8769 bits

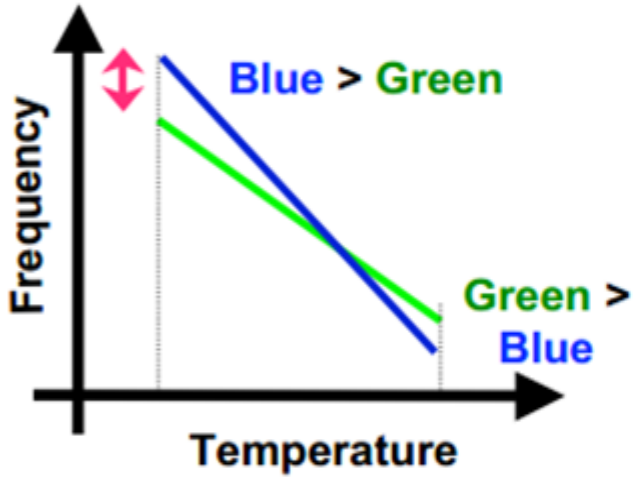
Reliability of RO PUFS [3]



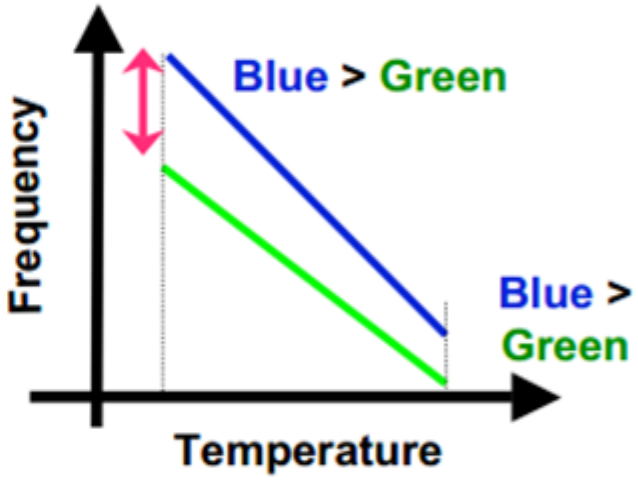
- Process variation: variants in doping and lithography process
- Temporary environment variations
  - Temperature
    - Slows down the device
  - Supply voltage
- Aging:
  - Negative Bias Temperature Instability
  - Hot Carrier Injection (HCI)
  - Temp Dependent Dielectric Breakdown
  - Interconnect Failure

Reliability Enhancement

- Environmental changes have a large impact on the freq. (and even relative ones)



(a) Frequencies are close



(b) Frequencies are far apart

Comparison of Delay-based PUFs

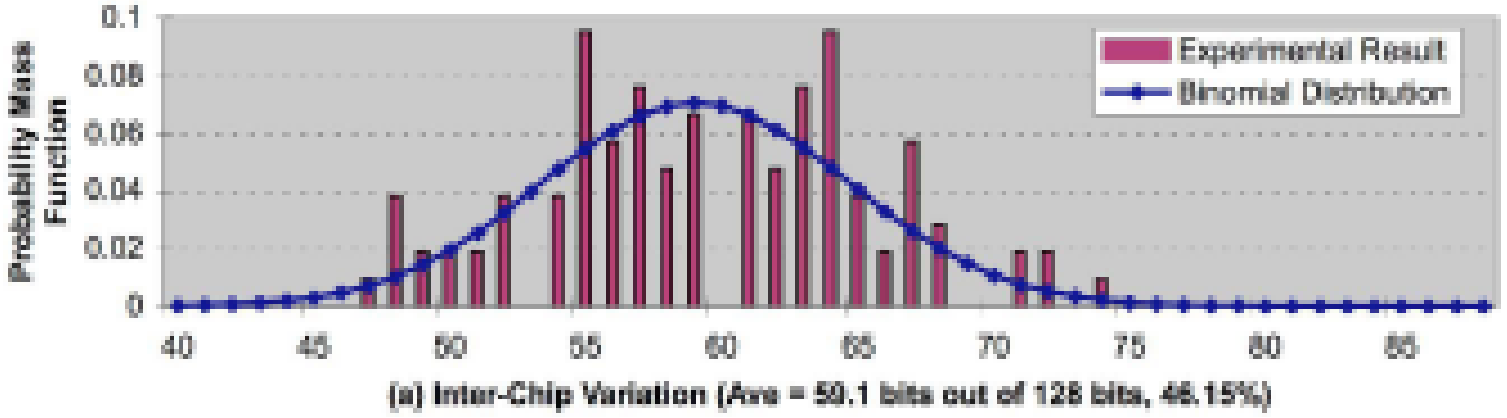
- ROs whose frequencies are far more stable than the ones with closer frequencies
  - Possible advantage: do not use all pairs, but only the stable ones
  - It is easy to watch the distance in the counter and pick the very different ones.
- RO PUF allows an easier implementation for both ASICs and FPGAs.
- The arbiter PUF is appropriate for resource constrained platforms such as RFIDs
- RO PUF is better for use in FPGAs and in secure processor design.

Experiments with RO PUFs

- Experiments done on 15 Xilinx Virtex4 LX25 FPGA (90nm)
- They placed 1024 ROs in each FPGA as a 16-by-64 array
- Each RO consisted of 5 INVs and 1 AND, implemented using look-up tables
- The goal is to know if the PUF outputs are unique (for security) and reproducible (for reliability and security)

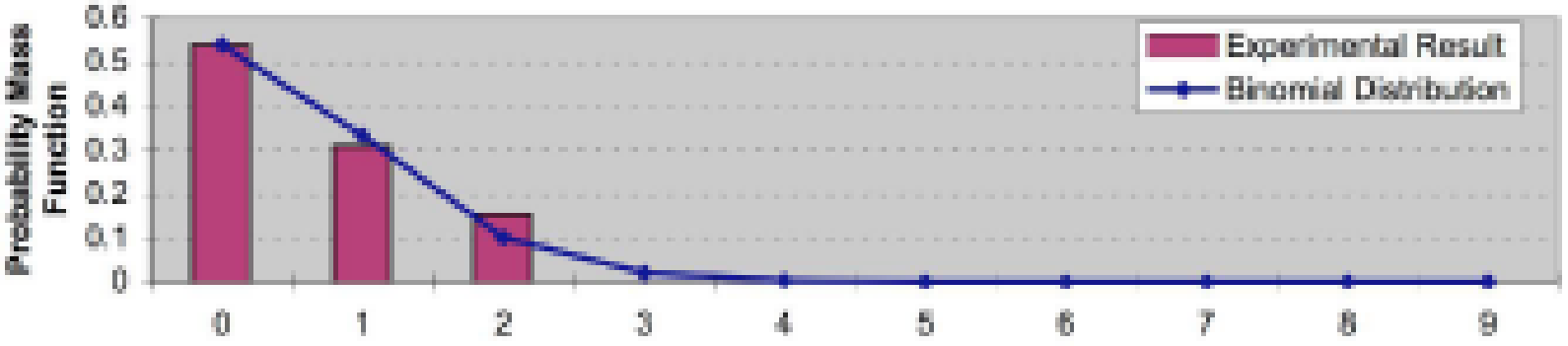
The Probability Distribution for Inter-chip Variations

- 128 bits are produced from each PUF
- x-axis: number of PUF o/p bits different b/w two FPGAs; y-axis: probability
  - Purple bars show the results from 105 pair-wise comparisons
  - Blue lines show a binomial distribution with fitted parameters (n=128, p =0.4615)
- Average inter-chip variations 0.4615 ~ 0.5



The Probability Distribution for Intra-chip Variations

- PUF o/p are generated at two different conditions and compared
- Changing the temperature from 20C to 120C and the core voltage from 1.2 to 1.08 altered the PUF o/p by ~0.6 bits (0.48%)
- Intra-chip variations is much lower than inter-chip – the PUF o/p did not change from small to moderate environmental changes



(b) Intra-Chip Var. under the Worst-Case Env. Change (20C 1.2V vs. 120C 1.08V)
(Ave = 0.61 bits out of 128 bits, 0.48%)

Leakage-based Silicon Analog PUF [5]

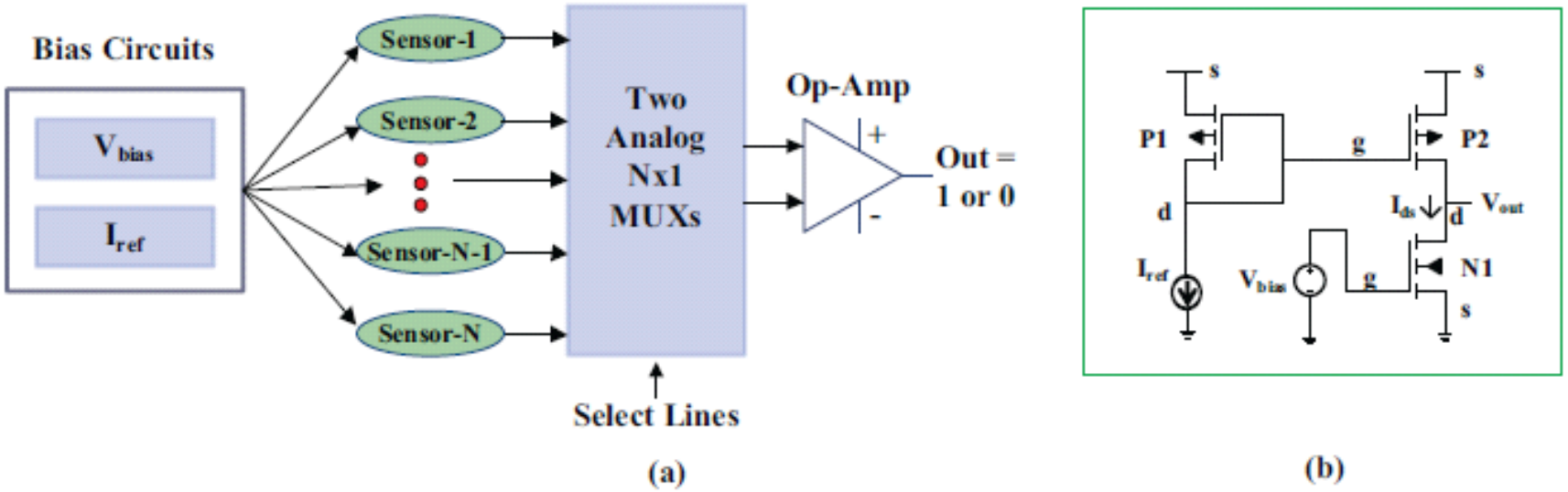


Fig. 2. (a) Architecture of L-PUF. (b) Leakage Sensor.

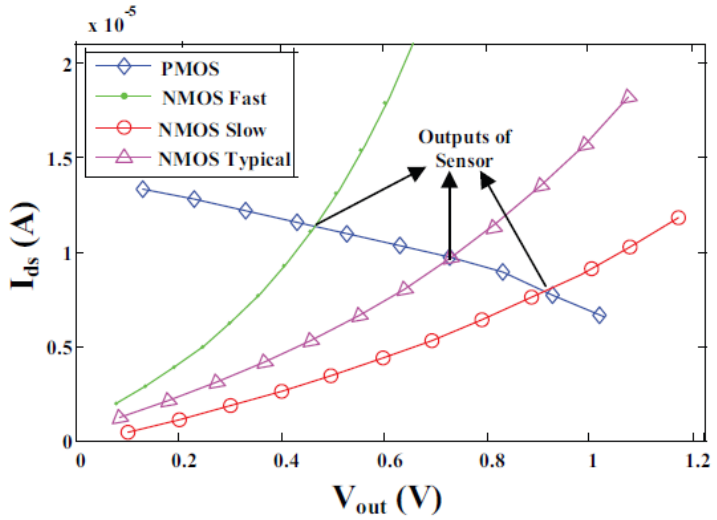


Fig. 3. I-V curves for P2 and N1.

Security Metrics

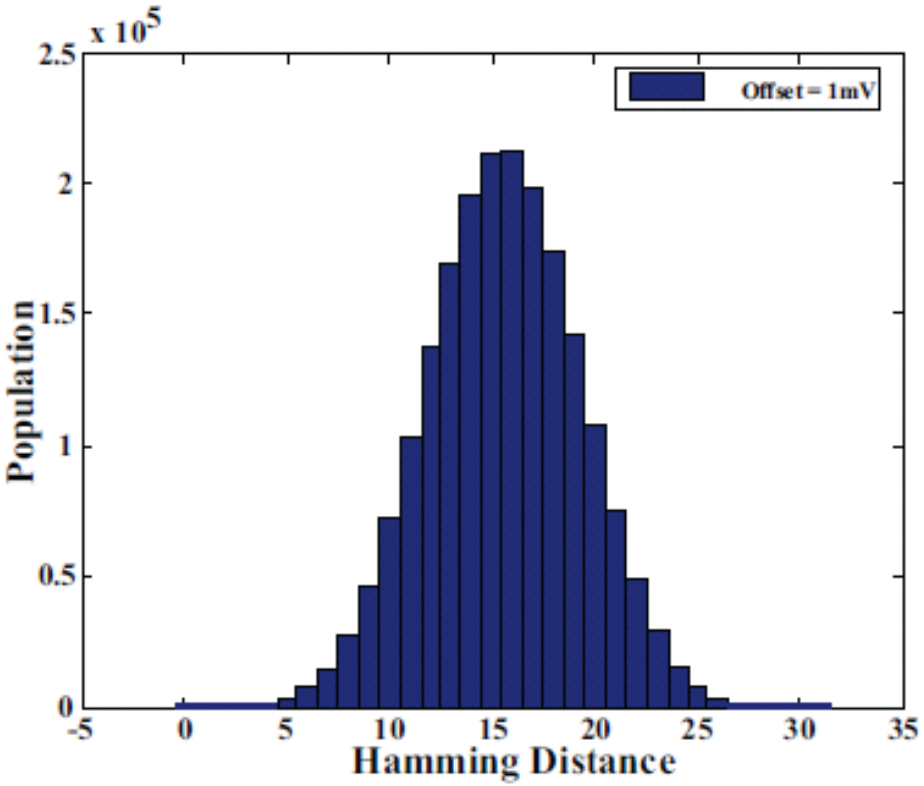


Fig. 4. Inter-chip variation of L-PUF.

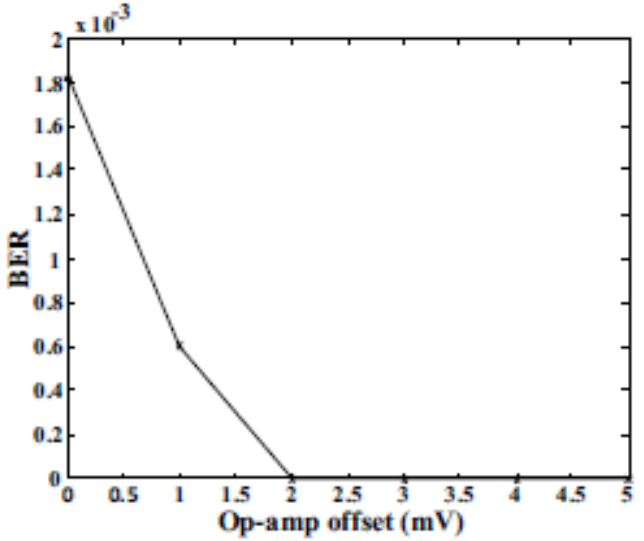


Fig. 5. Bit Error Rate vs. Op-amp offset.

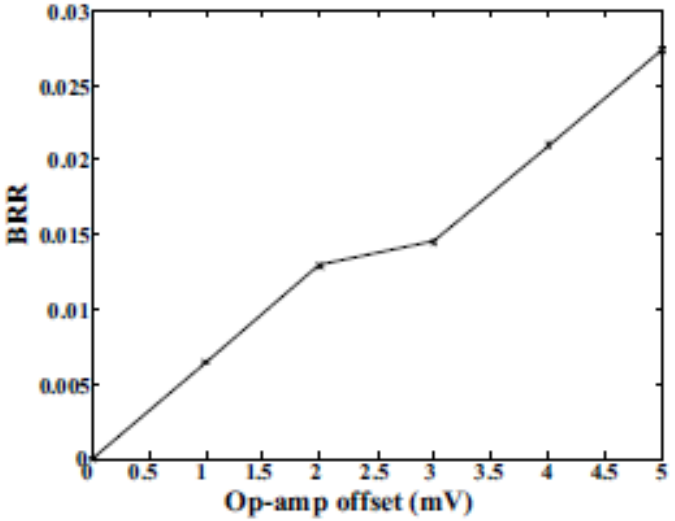


Fig. 6. Bit Rejection Rate vs. Op-amp offset.

References

[1] U. Ruhrmair, S. Devadas, and F. Koushanfar, *Security based on physical unclonability and disorder*, In: Introduction to hardware security and trust, Chapter 4.

[2]G. E. Suh and Srinivas Devasdas, *Physical Unclonable Function for Device Authentication and Secret Key Generation,* IEEE/ACM Design Automation Conference, 2007.

[3] A. Maiti and P. Schaumont, "Improved ring oscillator PUF: An FPGA-friendly secure primitive," J. Cryptology, vol. 24, no. 2, pp. 375–397.,2011.

[4] Daniel E. Holcomb, Wayne P. Burleson, and Kevin Fu, "Power-up SRAM state as an identifying Fingerprint and Source of True Random Numbers for RFID Tags," IEEE Trans. on Computers, vol. 58, no. 9, 2009.

[5] Dinesh Ganta, Vignesh Vivekraja, Kanu Priya and Leyla Nazhandali, *A Highly Stable Leakage-Based Silicon Physical Unclonable Functions*, VLSI 2011

Lab 7

- Design 16-bit arbiter-based PUF
- Simulate the functionality, and synthesize it for FPGA implementation
- Upload the design through virtual lab to one ZedBoard, and receive the response of your PUFs
- Run security and reliability analysis of your PUFs

PUF Design

- What you are given
  - puf_main.v
  - arbiterpuf_16.v
  - puf_main_pins.xdc
  - puf_main_timing.xdc
  - uart_rx.v
  - uart_tx.v
  - uart64_rx.v
  - uart64_tx.v

```
module arbiterpuf_16(input s,
                     input [15:0]C,
                     output [15:0]Q);

    arbiter A0(s, s, C, Q[0]);
    arbiter A1(s, s, C, Q[1]);
    arbiter A2(s, s, C, Q[2]);
    arbiter A3(s, s, C, Q[3]);

    arbiter A4(s, s, C, Q[4]);
    arbiter A5(s, s, C, Q[5]);
    arbiter A6(s, s, C, Q[6]);
    arbiter A7(s, s, C, Q[7]);

    arbiter A8(s, s, C, Q[8]);
    arbiter A9(s, s, C, Q[9]);
    arbiter A10(s, s, C, Q[10]);
    arbiter A11(s, s, C, Q[11]);

    arbiter A12(s, s, C, Q[12]);
    arbiter A13(s, s, C, Q[13]);
    arbiter A14(s, s, C, Q[14]);
    arbiter A15(s, s, C, Q[15]);

endmodule
```