RingRAM: A Unified Hardware Security Primitive for IoT Devices that Gets Better with Age

Michael Moukarzel

mamoukar@vt.edu

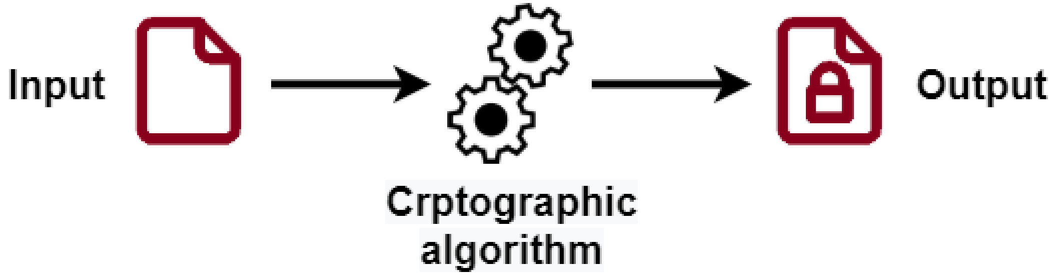
Matthew Hicks

mdhicks2@vt.edu

COMPUTER SCIENCE
VIRGINIA TECH™

Security depends on non-determinism

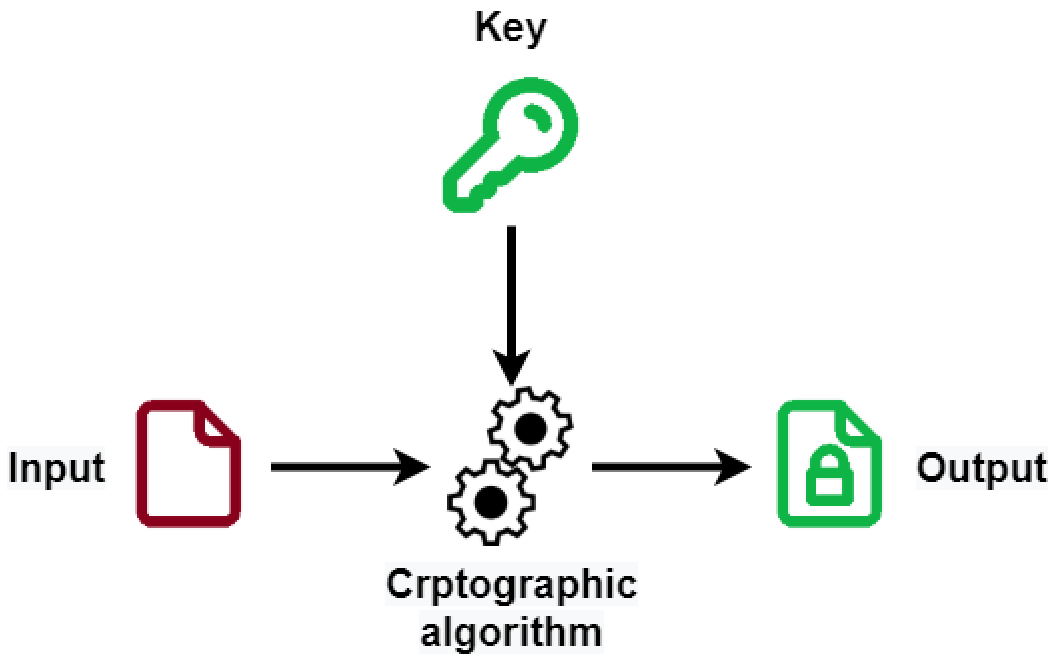
- Cryptographic algorithms
  - Deterministic by design
  - Same inputs = same output



Security depends on non-determinism

- Cryptographic algorithms
  - Deterministic by design
  - Same inputs = same output

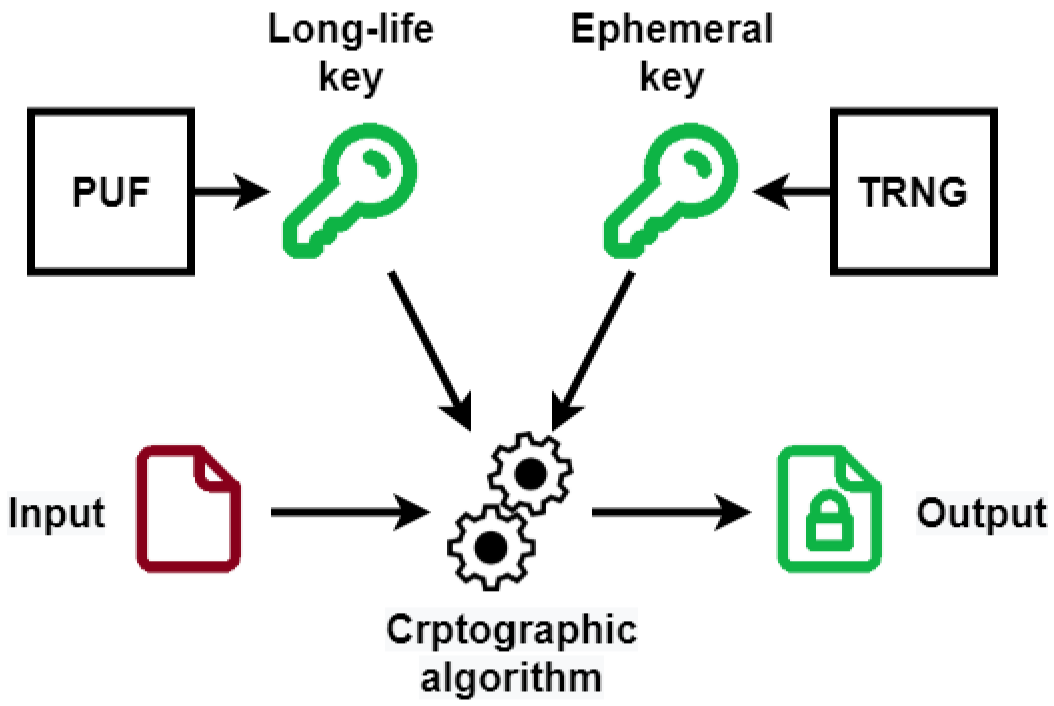
- Key
  - Non-deterministic
  - Adds security



There are two types of keys

- Long-life keys
  - Pre-shared keys
  - Device fingerprint
  - Private key

- Ephemeral keys
  - Generated continuously
  - Key agreement protocol

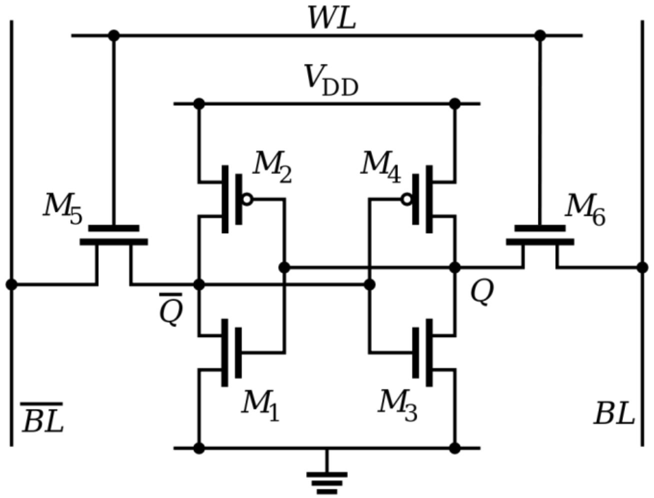


PUFs provide long-life keys

- Fingerprints
  - Device-specific identifier
  - Harnesses manufacturing-time chaos
  - Depends on within-chip variation

- SRAM
  - Leverages power-on state

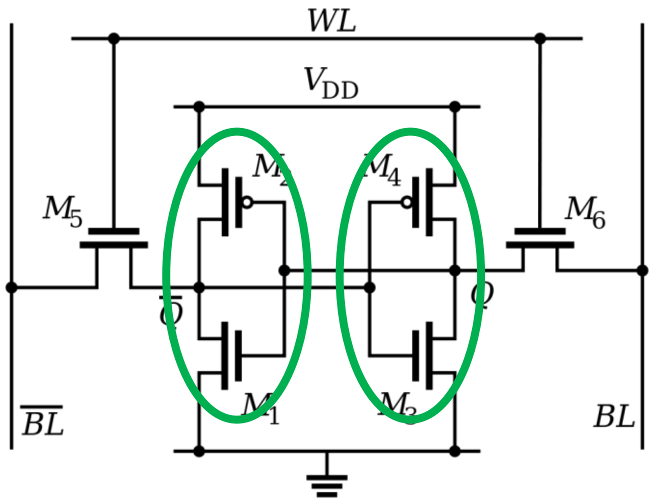
6T SRAM cell



PUFs provide long-life keys

- Fingerprints
  - Device-specific identifier
  - Harnesses manufacturing-time chaos
  - Depends on within-chip variation
- SRAM
  - Leverages power-on state

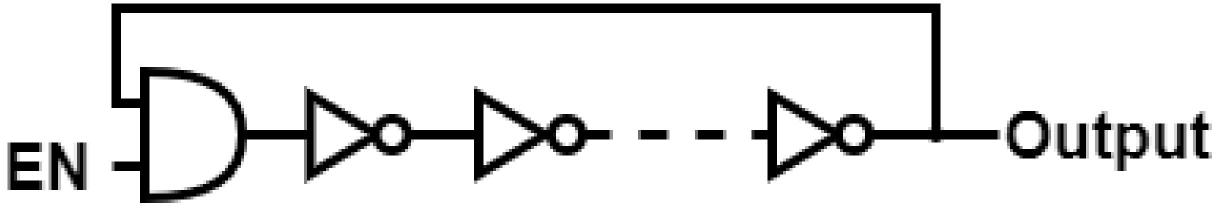
6T SRAM cell



TRNGs provide ephemeral keys

- Entropy
  - Accumulated operational chaos
  - Ideally high-rate and unbounded

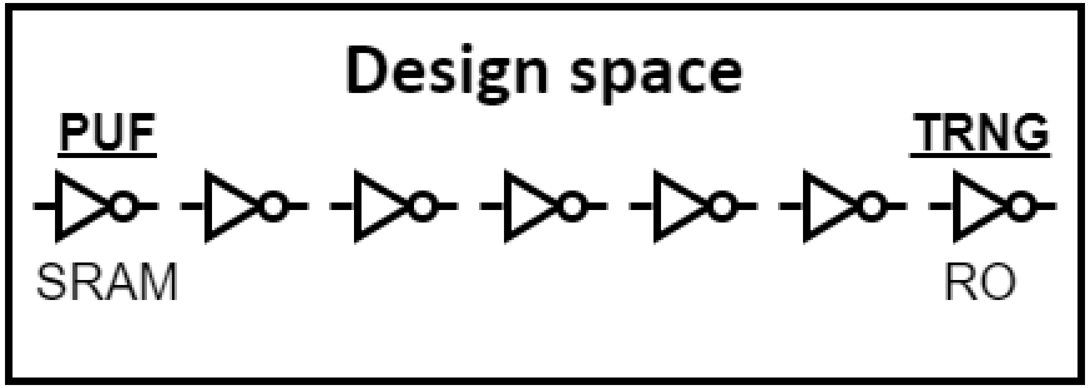
- Ring Oscillators (RO)
  - Frequency variation



IoT demands a unified hardware security primitive

- Advantage
  - Reduced overhead (power, area, cost)
  - Cross integrity validation

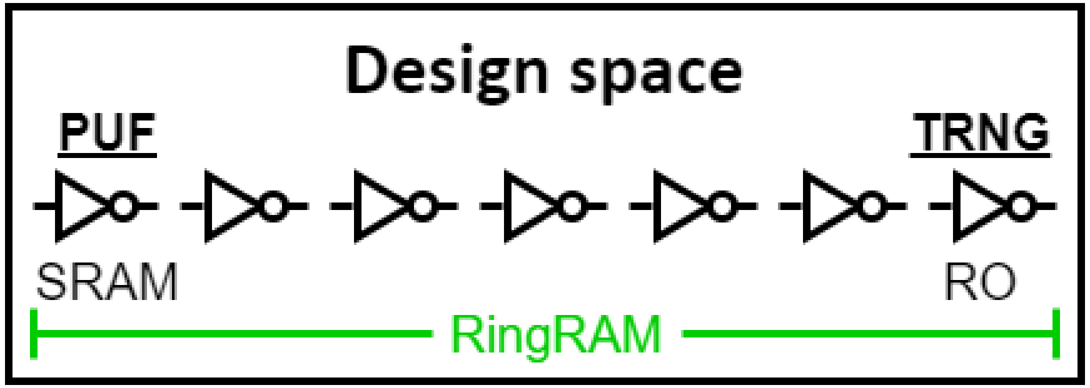
- Trade Space: SRAM vs RO
  - Different forms of chaos
  - Captured using inverters



IoT demands a unified hardware security primitive

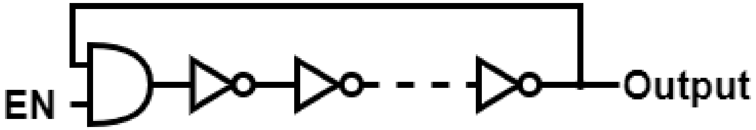
- Advantage
  - Reduced overhead (power, area, cost)
  - Cross integrity validation

- Trade Space: SRAM vs RO
  - Different forms of chaos
  - Captured using inverters



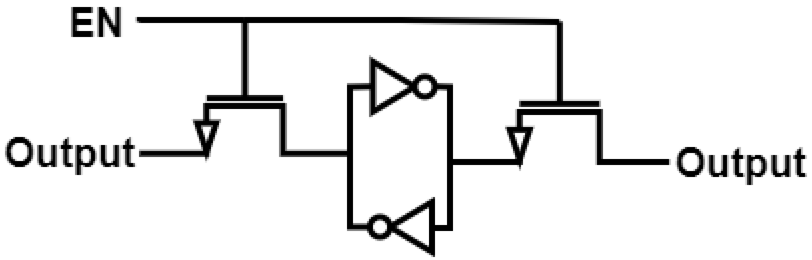
RingRAM is a best-of-breed combination of RO and SRAM

RO



Chaos: operational
Generation: unbound

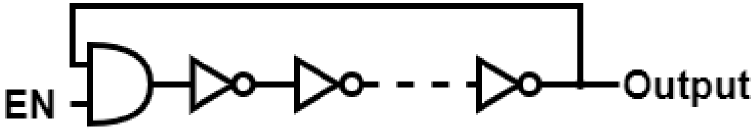
SRAM



Chaos: manufacturing
Generation: bound

RingRAM is a best-of-breed combination of RO and SRAM

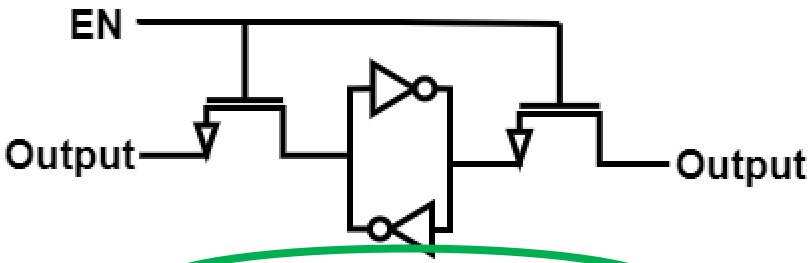
RO



~~Chaos: operational~~

Generation: unbound

SRAM



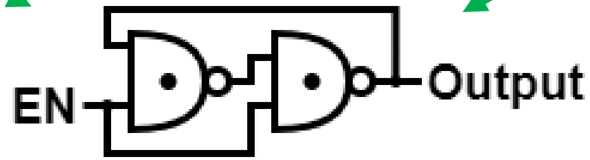
Chaos: manufacturing

~~Generation: bound~~

Unbounded

Chaos

RingRAM



Chaos: manufacturing

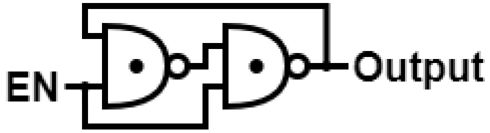
Generation: unbound

Exposing the PUF/TRNG design space

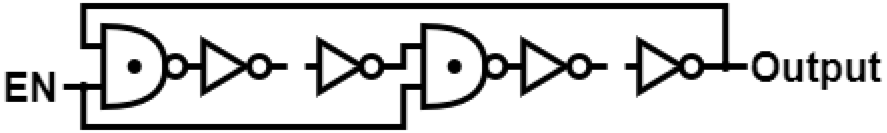
- **Observation**: increasing inverters per chain normalizes propagation delay

- **Result**: longer chains increase proportion of TRNG cells

NAND

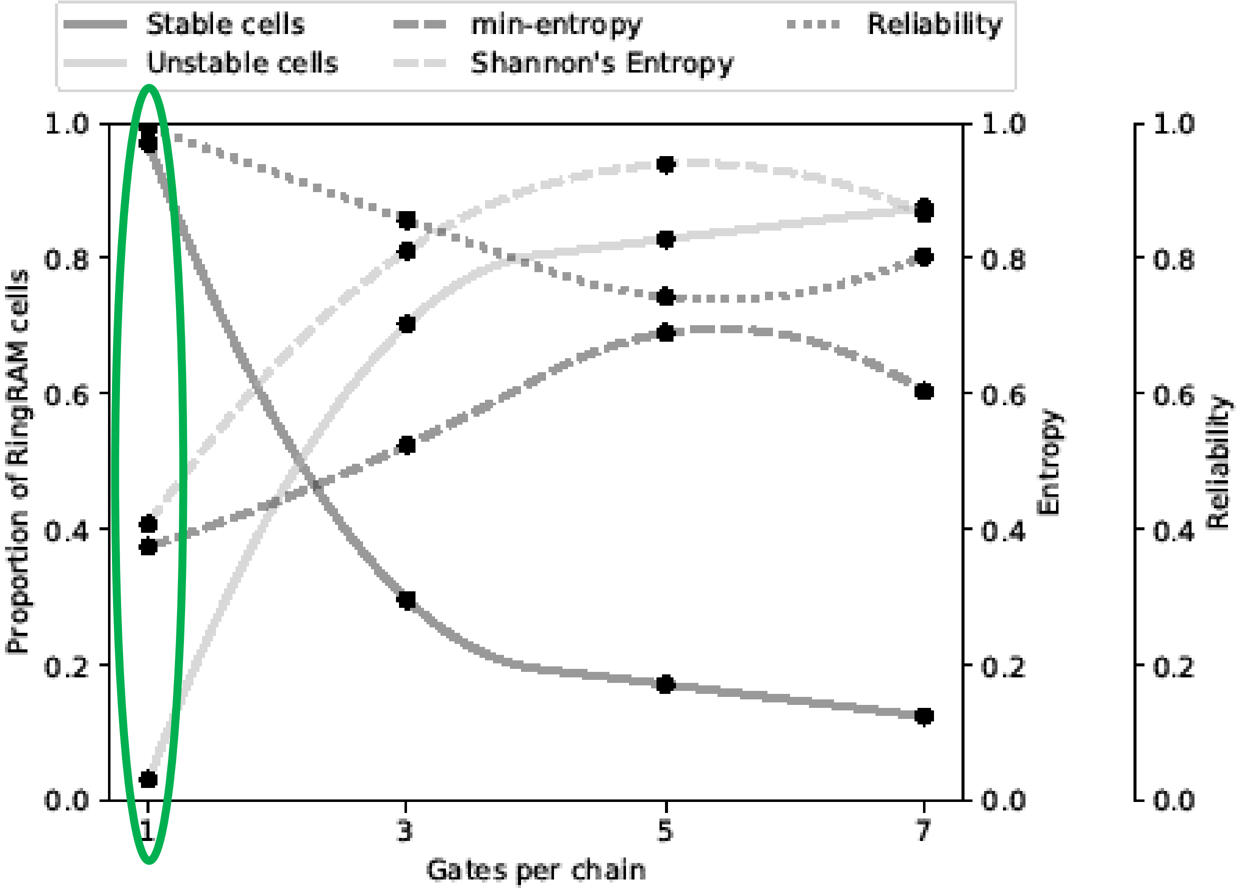


NAND + even number of inverters



Exposing the PUF/TRNG design space

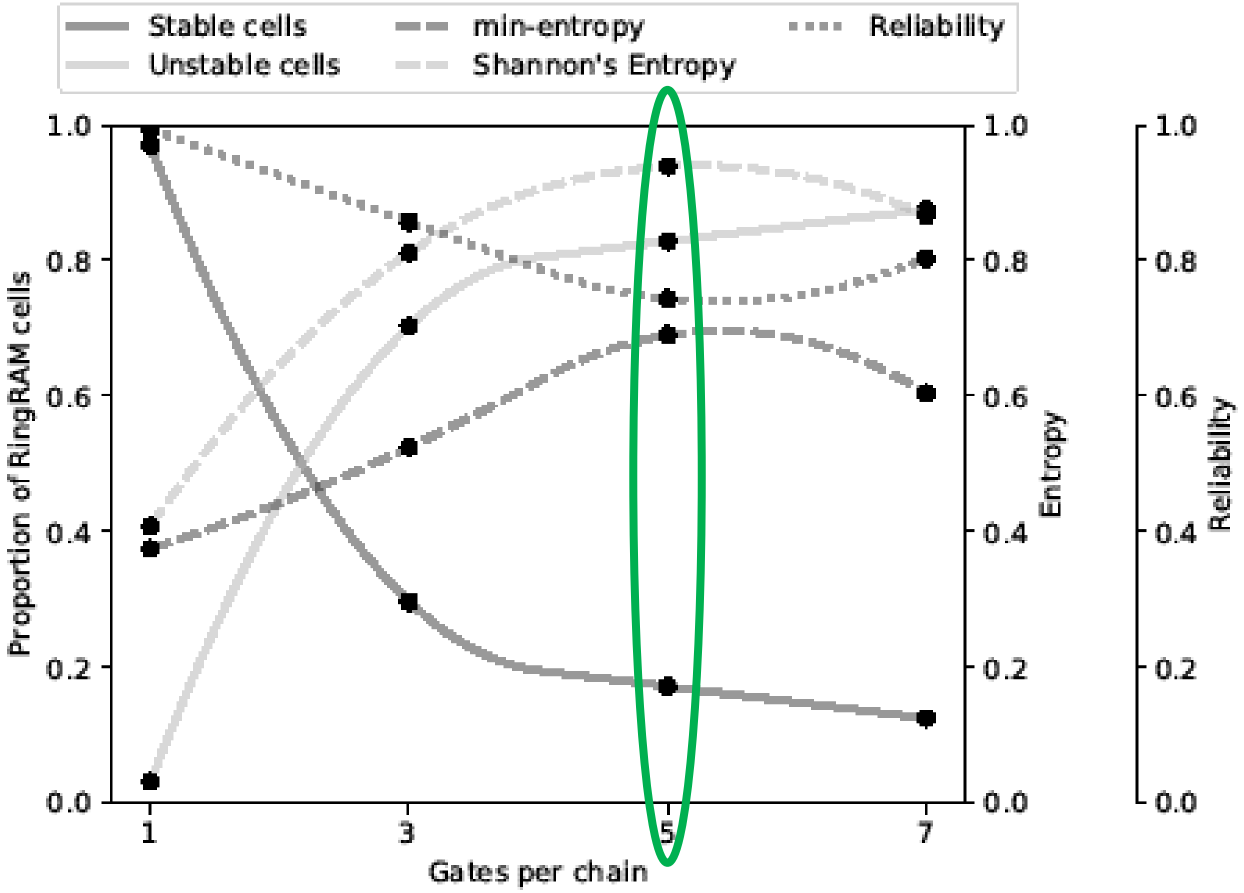
- Controlling composition
  - Number of gates per chain
  - Explore trade-space
- Single NAND gate
  - Highly stable cells
  - Great for PUFs
  - Not great for TRNGs



Exposing the PUF/TRNG design space

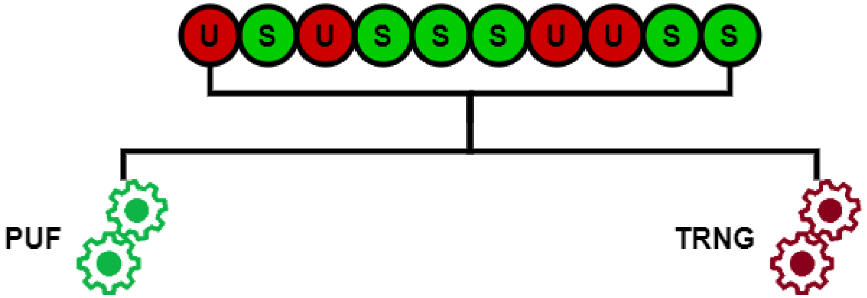
- Controlling composition
  - Number of gates per chain
  - Explore trade-space

- NAND gate + 4 inverters
  - High TRNG entropy
  - Good PUF reliability



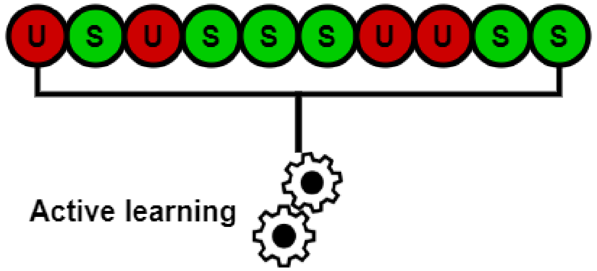
On-chip classification of response bits

- **Issue**:
  - PUFs want less noise
  - TRNGs want more noise

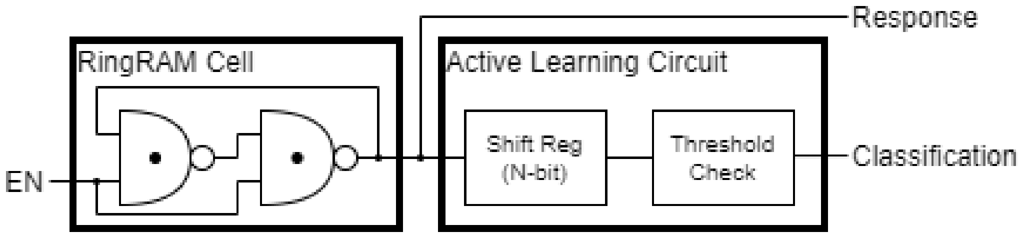


On-chip classification of response bits

- **Issue**:
  - PUFs want less noise
  - TRNGs want more noise



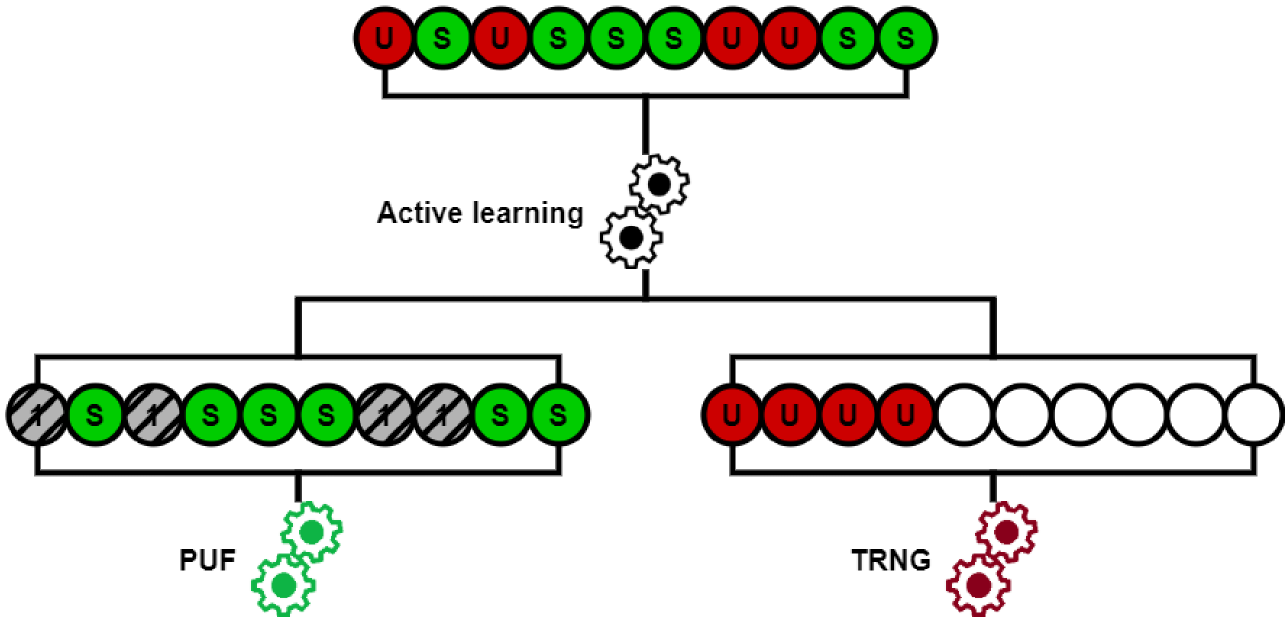
- Active learning:
  - Classifies cells dynamically
  - Shift reg: stores response
  - Threshold check: shift reg bias



On-chip classification of response bits

- **Issue**:
  - PUFs want less noise
  - TRNGs want more noise

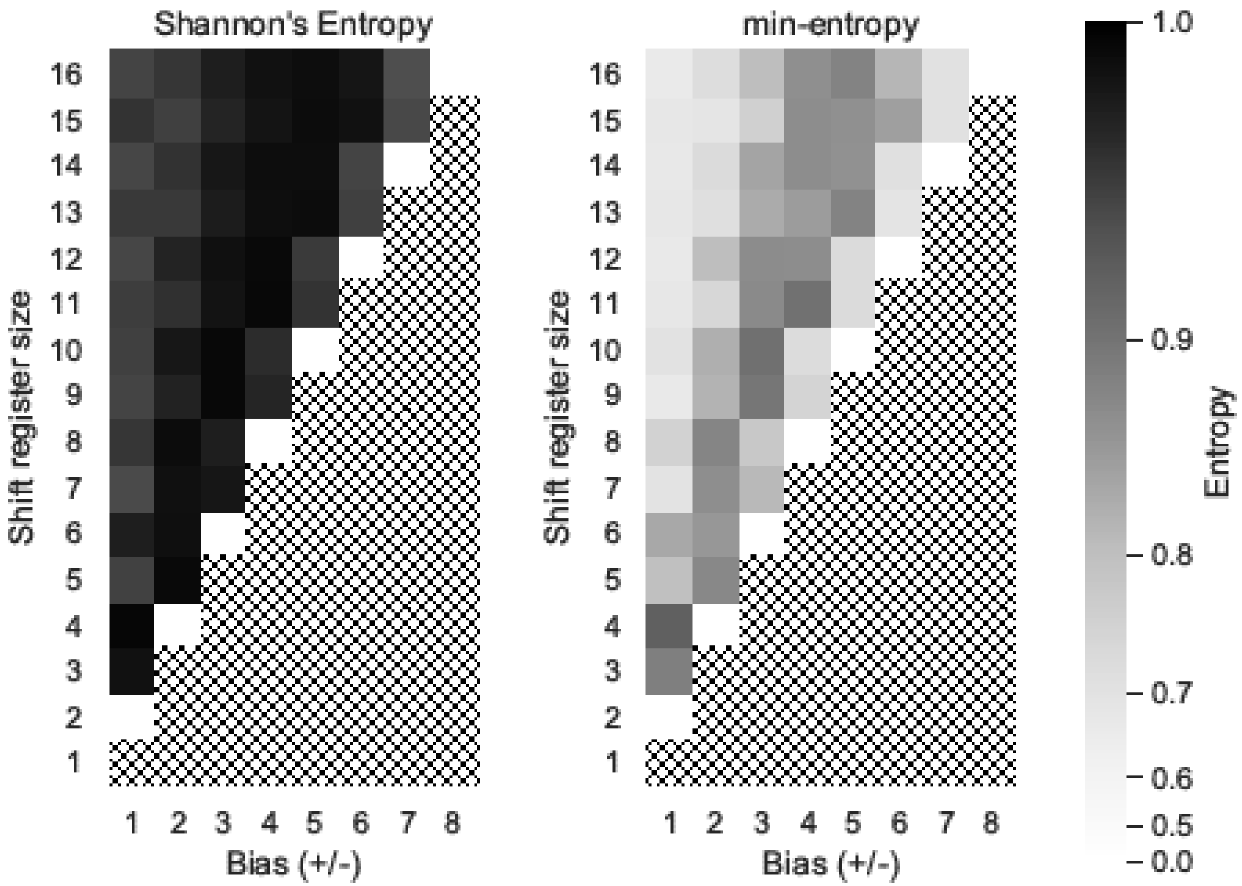
- Active learning:
  - Classifies cells
  - PUFs only output stable
  - TRNGs only output unstable



On-chip classification of response bits

- **Issue**:
  - PUFs want less noise
  - TRNGs want more noise

- Active learning design space exploration
  - Examine entropy
  - Vary bias and shift reg

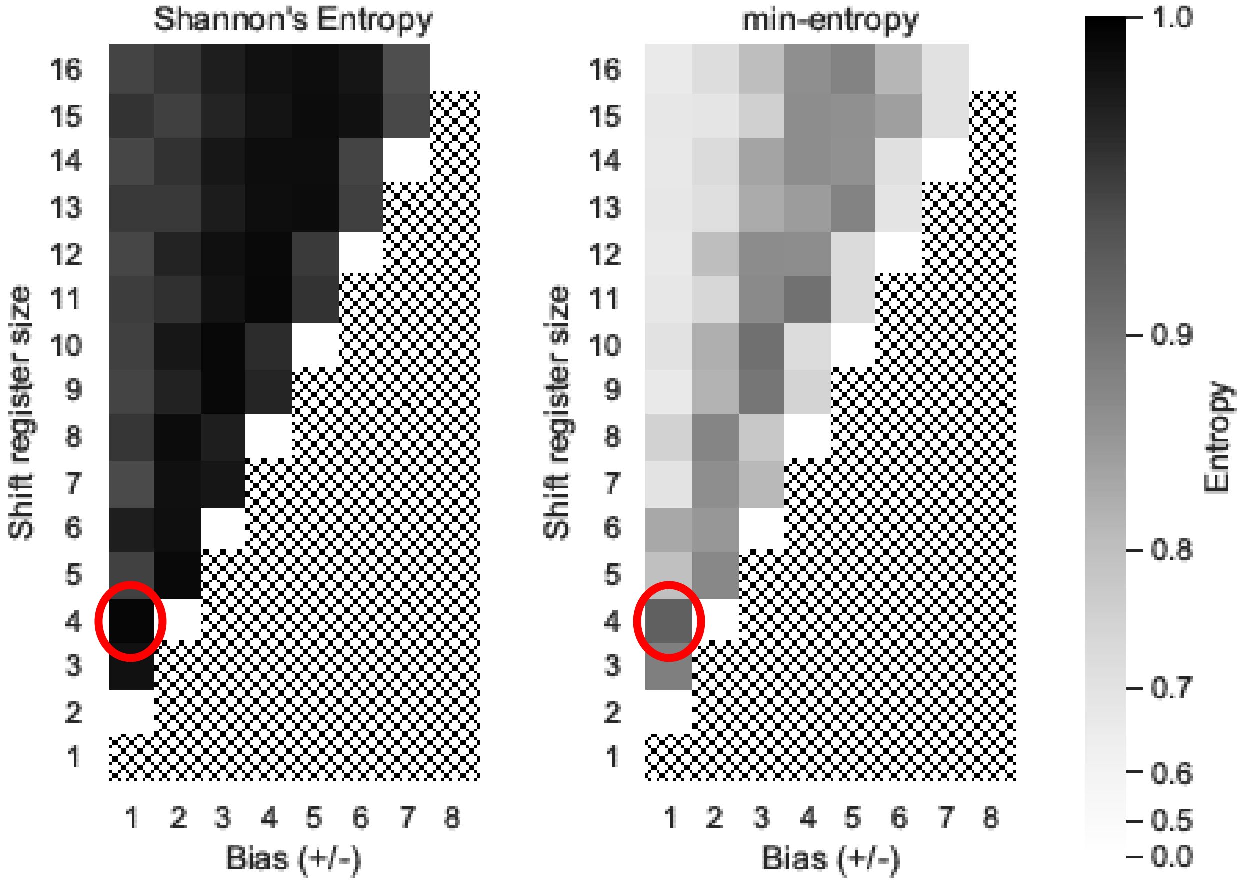


On-chip classification of response bits

- **Issue**:
  - PUFs want less noise
  - TRNGs want more noise

- Active learning design space exploration
  - Examine entropy
  - Vary bias and shift reg

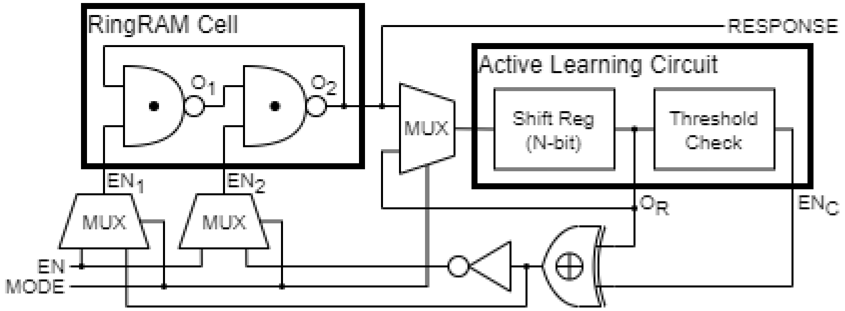
Optimal design: 4 shift reg ±1 bias



Making age a strength

- Aging:
  - All electronic devices age
  - Use dependent

- Active aging:
  - Stable cells – increase bias for PUFs
  - Unstable cells – decrease bias for TRNGs



RingRAM is an effective PUF

- Reliability: produce the same responses

$$100\% - \frac{1}{m}\sum_{i=1}^{m}\frac{HD(R_0, R_i)}{n} \times 100\%$$

- Uniformity: produce balanced responses

$$\frac{1}{m}\sum_{i=1}^{m}\frac{HW(R_i)}{n} \times 100\%$$

- Uniqueness: dependency on placement

$$\frac{2}{c(c-1)}\sum_{i=1}^{c-1}\sum_{j=i+1}^{c}\frac{HD\left(R_i, R_j\right)}{n} \times 100\%$$

Metric	RO	SRAM	RingRAM
Reliability	99.10%	92.20%	98.40%
Uniformity	49.40%	48.70%	48.20%
Uniqueness	47.20%	48.70%	48.40%
Transistors/unified bit	1641.1	99.75	88.5

RingRAM is an effective TRNG

- Min-entropy: worst-case

$$log_2 \frac{1}{P_{MAX}(x)}$$

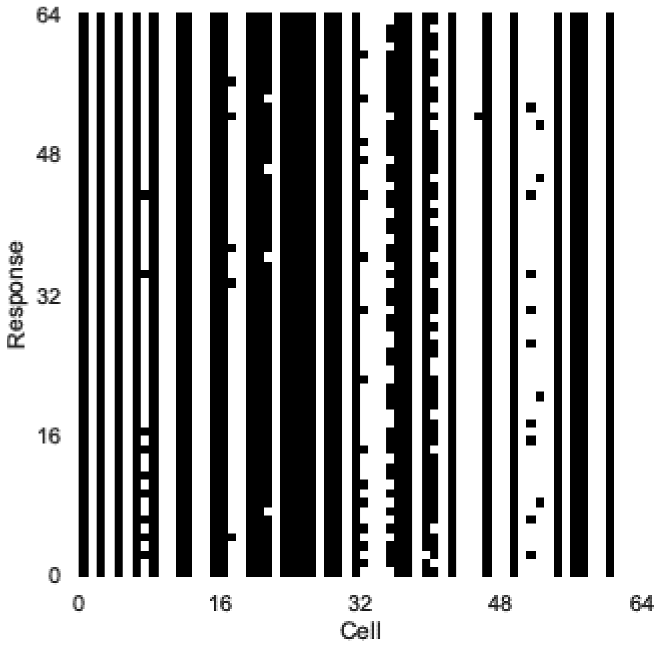
- Shannon’s Entropy: average case

$$-\sum_{i=0}^{n} P(x_i) log_2 P(x_i)$$

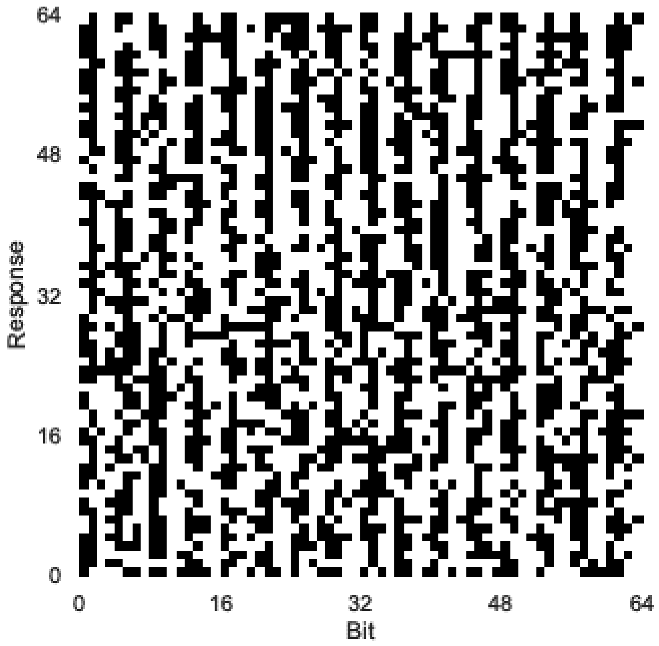
Metric	RO	SRAM	RingRAM
Unbounded	✓	×	✓
Throughput	38M	N/A	228M
min-entropy	0.97	0.031	0.981
Shannon’s Entropy	0.99	0.058	0.999
Transistors/unified bit	1641.1	99.75	88.5

RingRAM is an effective TRNG

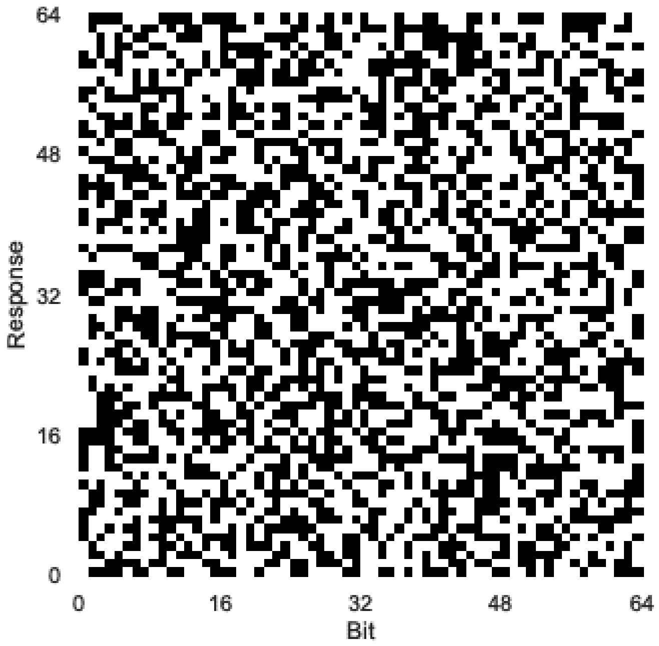
RingRAM



RingRAM + CC



RingRAM + CC + AL



RingRAM is resilient to thermal attacks

- Temperature:
  - Significant source of systematic run-time variation
- RingRAM's cross-coupled design:
  - Sensitive to manufacturing-time chaos
  - Naturally filters out systematic run-time variation
- Controlled thermal operation:
  - Ambient temperature: 0°C, 10°C, 20°C, 30°C, 40°C
  - PUF reliability: ±1.27%
  - TRNG entropy: ± 0.22%

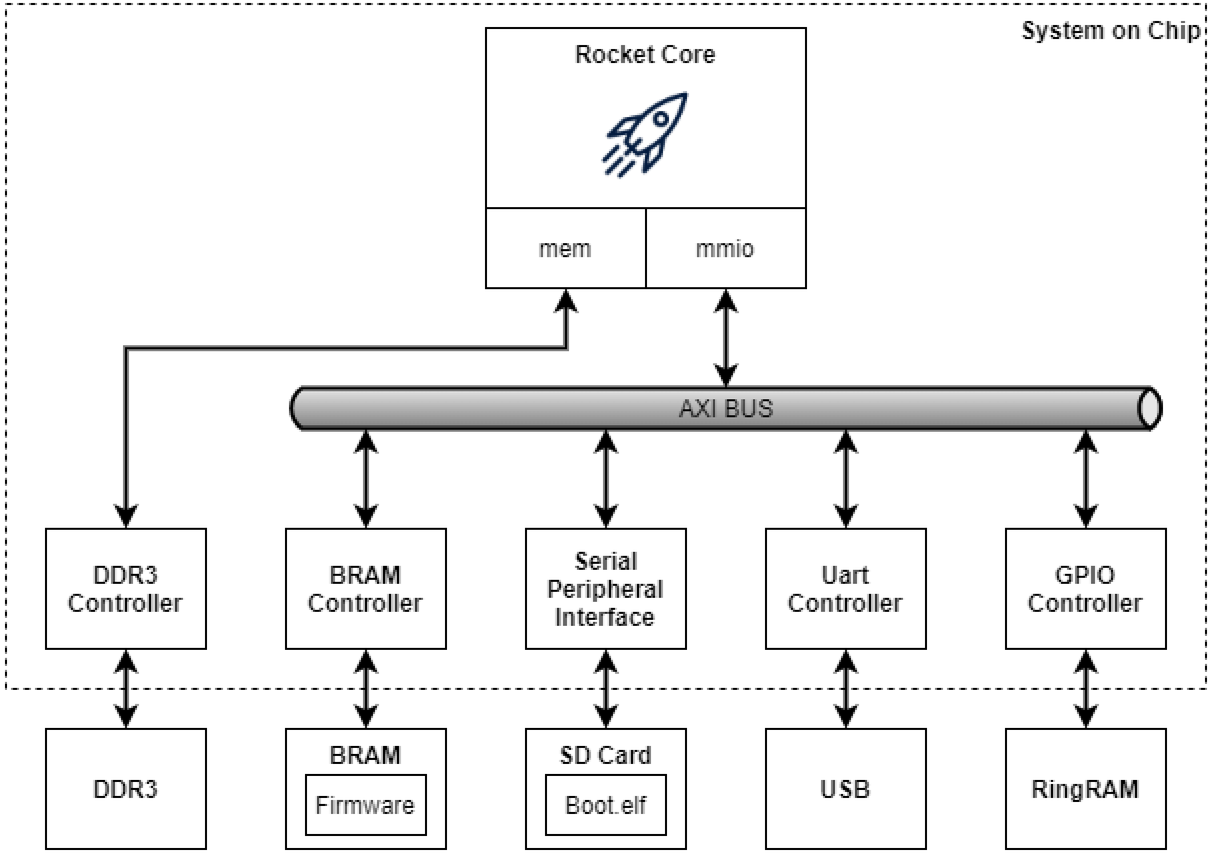
RingRAM is secure

- Single-use
  - Read only to attacker
- Aging resilience
  - Active aging increases performance over time
- Thermal & voltage resilience
  - Tightly packed layout treats these as common-mode noise

Metric	RO	SRAM	RingRAM
Single-use	✓	X	✓
Aging Resilient	✓	X	✓
Thermal Resilient	X	✓	✓
Voltage Resilient	X	✓	✓

RingRAM improves system security

- Core: 64-bit Rocket RISC-V
- OS: Linux 5.5.2
- Speed test: OpenSSL 1.1.1

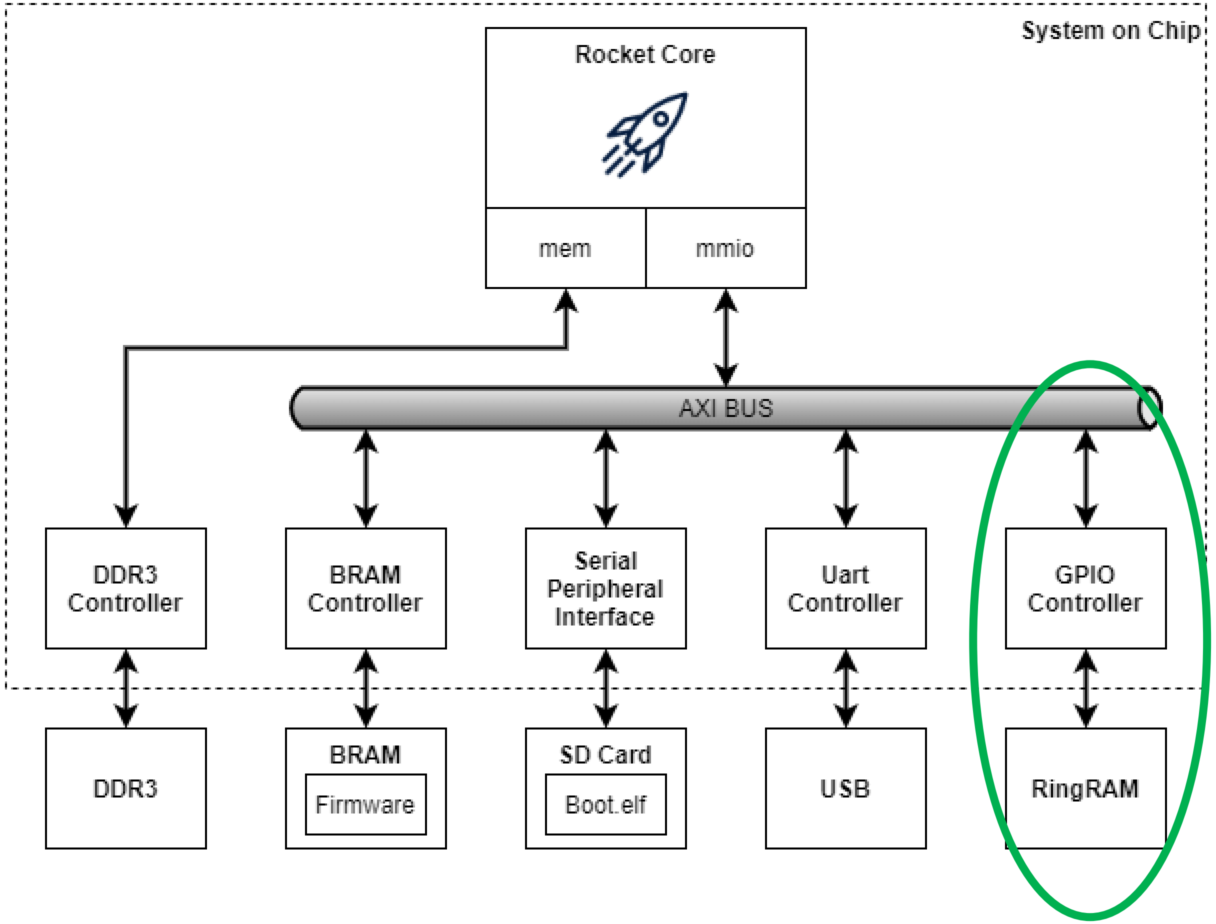


RingRAM improves system security

- Device drivers
    - /dev/random
    - /dev/urandom

- OpenSSL speed test

Test	RingRAM	Linux
sha-512	0.00%	0.13%
aes-192	-0.05%	0.03%
rsa-2048	0.00%	0.00%



Thank you!

Find RingRAM source code and FPGA prototypes at:

https://github.com/FoRTE-Research/RingRAM