Feb 26, 2015 @ Worcester Polytechnic Institute

SRAM-based Physical Unclonable Functions

Daniel E. Holcomb
UMass Amherst

Collaborators for these works:
Wayne P Burleson
Kevin Fu
Amir Rahmati
Uli Ruhrmair
Negin Salajegheh
Xiaolin Xu

Counterfeit Electronics

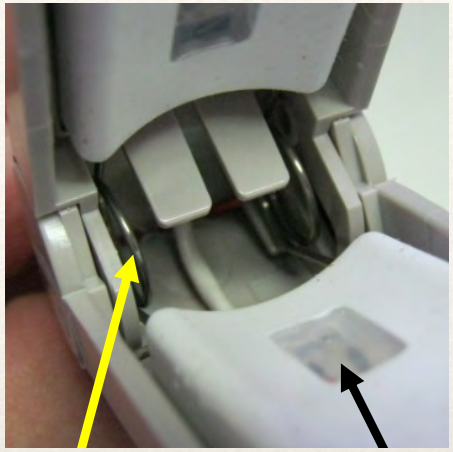
- DoD: 1.8k incidents in 2009/10; 1M parts
- Recycled e-waste
- Test rejects
- Regrading

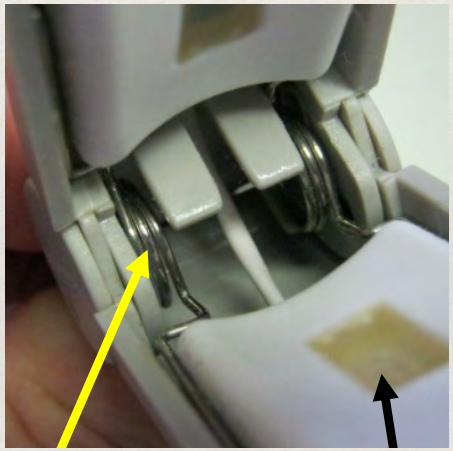
aeri.com

Counterfeit Electronics

- DoD: 1.8k incidents in 2009/10; 1M parts
- Recycled e-waste
- Test rejects
- Regrading

aeri.com



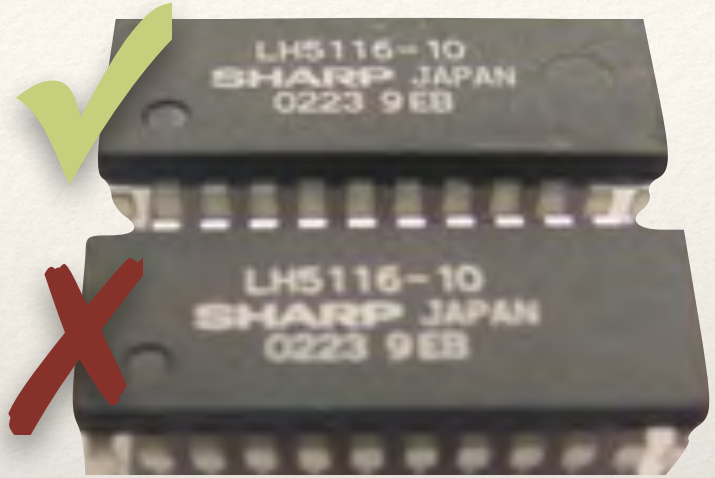

mhra.gov.uk

We do not want a $12 million missile defense interceptor's reliability ***compromised by a $2 counterfeit part.***
-- General Patrick O'Reilly, Director, Missile Defense Agency

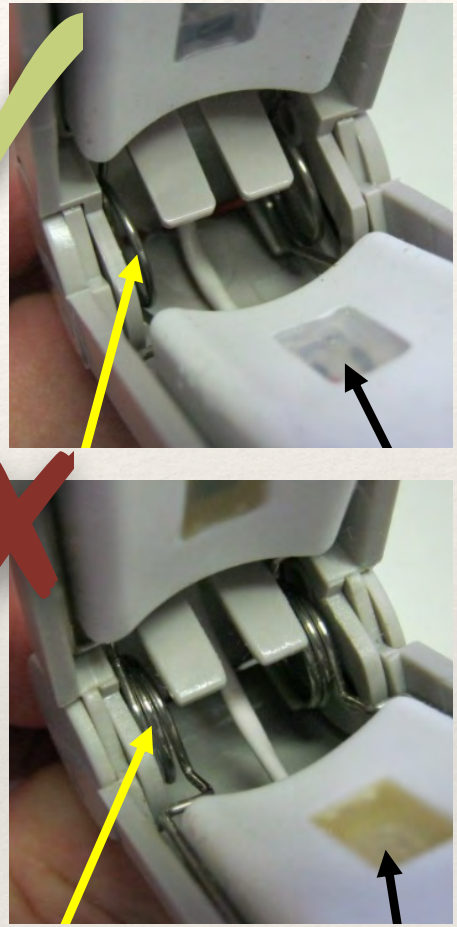
Both components and devices have been counterfeited, and the practice appears to be growing. With a high potential profit, ***counterfeiting medical devices is a huge business.***
-- Unique Identification for Medical Devices -- FDA-sponsored report, 2006

Counterfeit Electronics

- DoD: 1.8k incidents in 2009/10; 1M parts
- Recycled e-waste
- Test rejects
- Regrading



aeri.com



mhra.gov.uk

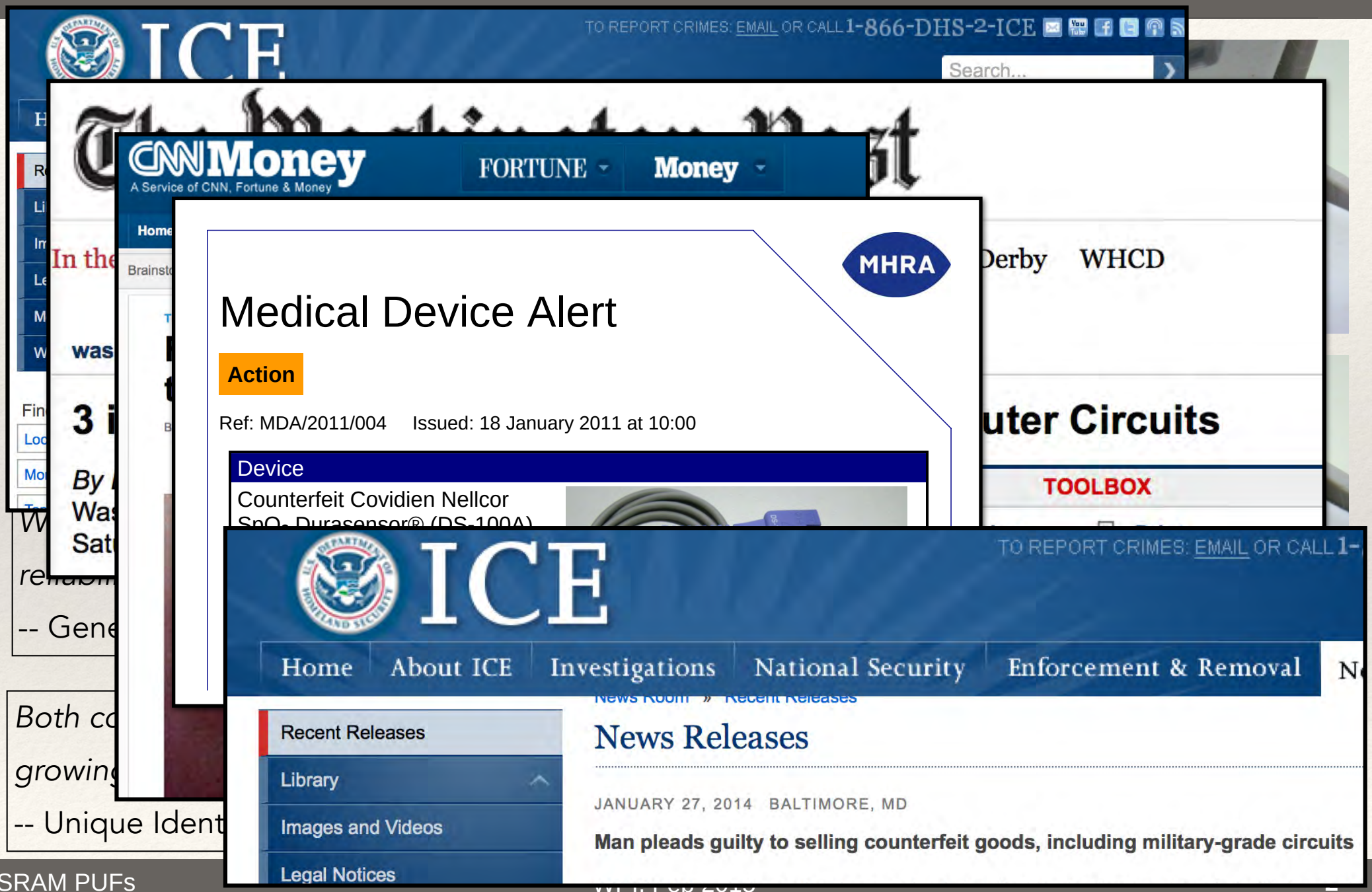
We do not want a $12 million missile defense interceptor's reliability ***compromised by a $2 counterfeit part.***

-- General Patrick O'Reilly, Director, Missile Defense Agency

Both components and devices have been counterfeited, and the practice appears to be growing. With a high potential profit, ***counterfeiting medical devices is a huge business.***

-- Unique Identification for Medical Devices -- FDA-sponsored report, 2006

Counterfeit Electronics



Both co

growing

-- Unique Ident

Unique Features

Abstraction: the act of considering something as a general quality or characteristic, **apart from concrete realities**, specific objects, **or actual instances**. [dictionary.com]

Biometrics: the measurement and analysis of **unique** physical or behavioral **characteristics** especially as a means of **verifying personal identity**. [merriam-webster.com]

[Bertillon, 1893]

Unique Features

Abstraction: the act of considering something as a general quality or characteristic, **apart from concrete realities**, specific objects, **or actual instances**. [dictionary.com]

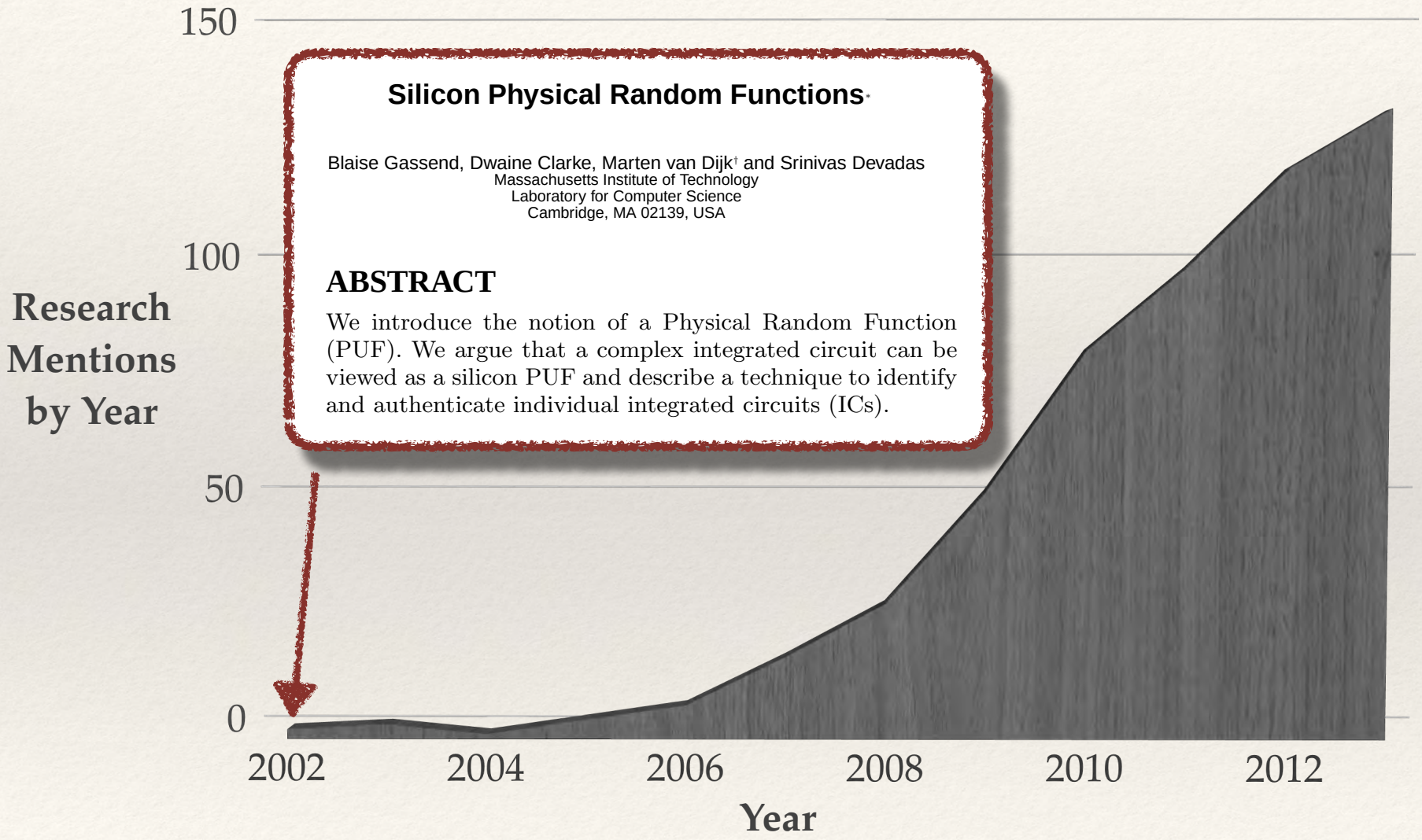
Biometrics: the measurement and analysis of **unique** physical or behavioral **characteristics** especially as a means of **verifying personal identity**. [merriam-webster.com]

Overview

- **Introduction to PUFs**

1. SRAM power-up state as PUF
2. SRAM data retention voltage as PUF
3. Modified SRAM as challenge-response PUF

Physical Unclonable Functions



Silicon Physical Random Functions*

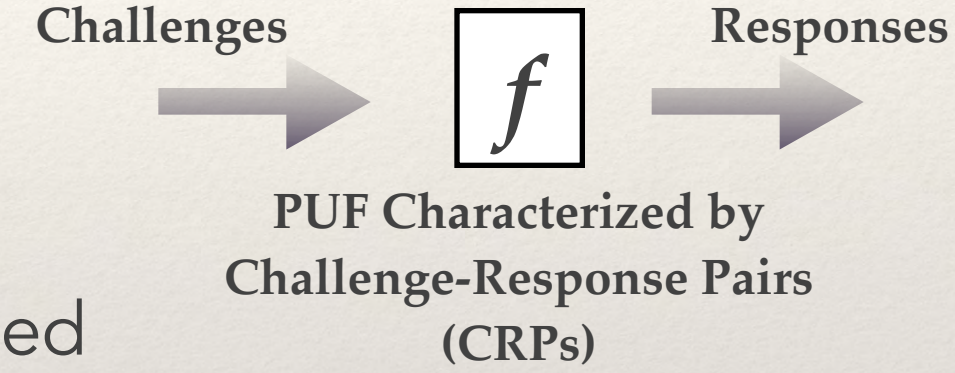
Blaise Gassend, Dwaine Clarke, Marten van Dijk† and Srinivas Devadas
Massachusetts Institute of Technology
Laboratory for Computer Science
Cambridge, MA 02139, USA

ABSTRACT

We introduce the notion of a Physical Random Function (PUF). We argue that a complex integrated circuit can be viewed as a silicon PUF and describe a technique to identify and authenticate individual integrated circuits (ICs).

Physical Unclonable Functions

- Physical
  - Behavior depends on physical variations
- Unclonable
  - No way to predict outputs
  - Behavior cannot be modeled
  - Behavior cannot be observed
- Function
  - Produces responses, possibly from challenges



Design Considerations for Silicon PUFs

- Outputs determined by uncorrelated variation
  - Random dopant fluctuations and small devices
  - Balanced parasitics and wire lengths to avoid bias
- Variation and noise hard to separate
  - Distance-based matching
  - Error correction
- Secure
  - Unreadable by invasive attack

Weak vs Strong PUFs

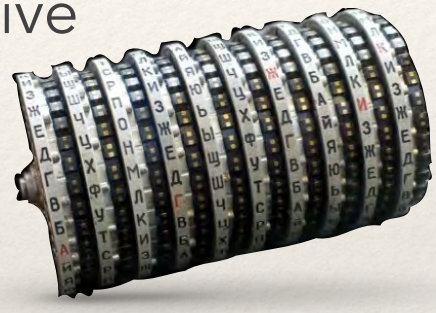
Weak PUFs

- Use cases: New form of key storage

- No challenge, just response
- Responses remain internal
  - Perfect internal error correction
- Attacks: Cloning and invasive reading of responses

Strong PUFs

- Use cases: New cryptographic primitive



- Many challenge-response pairs
- Public CRP interface
  - Error correction outside PUF is possible
- Attacks: Modeling attacks and protocol attacks

Weak vs Strong PUFs

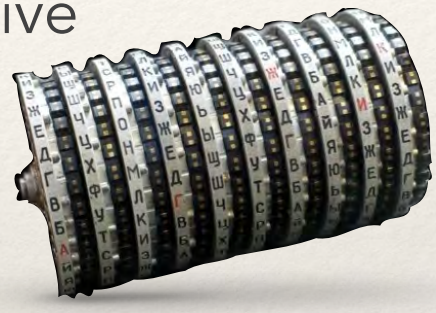
Weak PUFs

- Use cases: New form of key storage

- No challenge, just response

Strong PUFs

- Use cases: New cryptographic primitive

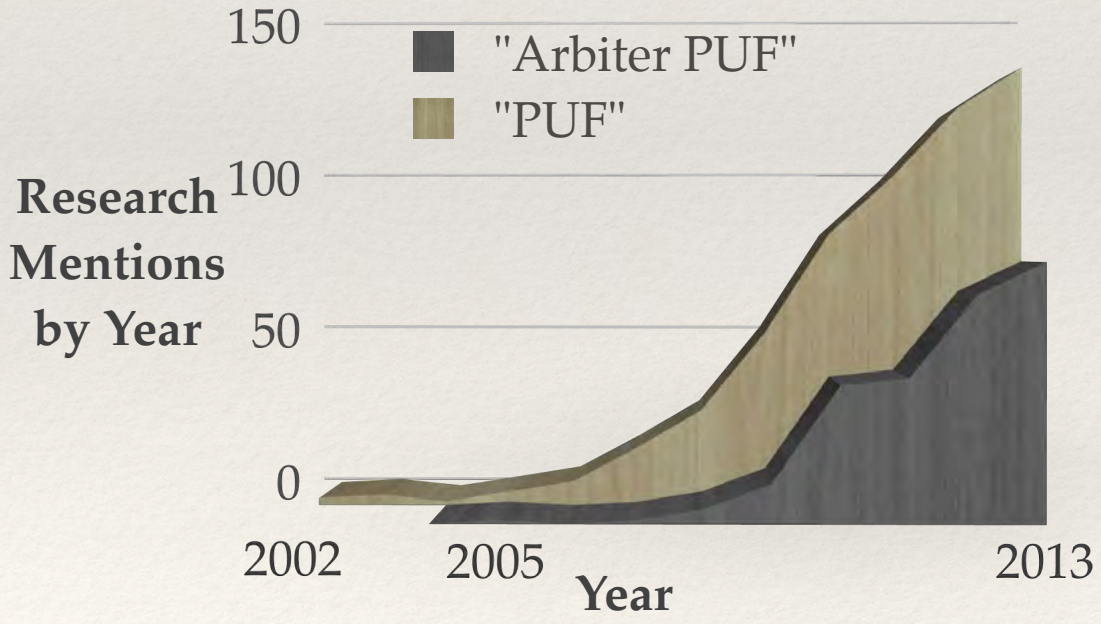


- Many challenge response pairs

- Weak and strong are two PUF subclasses among many
  - Controlled PUFs
  - Public PUFs
  - SIMPL, etc

Examples of Strong PUFs

- Optical PUF [Pappu et al. '02]
- Arbiter PUF [Gassend et al. '02, Lim et al. '05]
- Bistable Ring PUF [Chen et al. '11]
- Low-power current-based PUF [Majzoobi et al. '11]



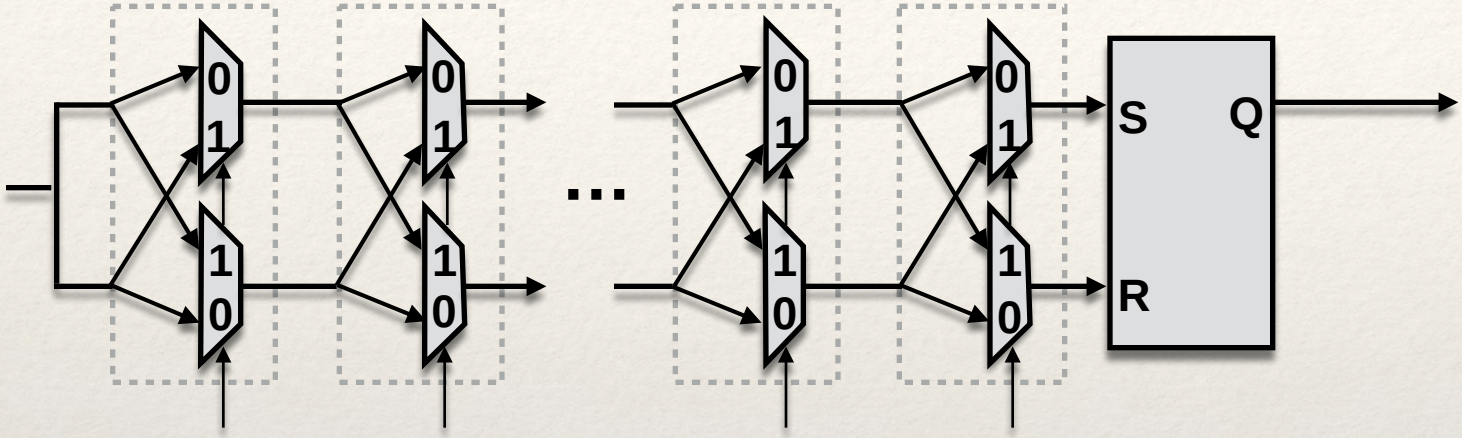
Strong PUF Protocols

- Identification/Authentication (1)
- Key Exchange (2,3)
- Oblivious transfer (4,3,5,6) — enables secure two-party computation
- Bit commitment (3,5,6,7,8) — enables zero-knowledge proofs
- Combined key exchange and authentication (9)

(1) R. Pappu et al, Science 2002
(2) M.v.Dijk, US Patent 2,653,197, 2004
(3) C. Brzuska et al, CRYPTO 2011
(4) U. Rührmair, TRUST 2010
(5,6) U. Rührmair, M.v.Dijk, CHES 2012 and JCEN 2013
(7) U. Rührmair, M.v. Dijk, Cryptology ePrint Archive, 2012
(8) Ostrovsky et al., EUROCRYPT 2013
(9) Tuyls and Skoric, Strong Authentication with Physical Unclonable Functions, Springer 2007

Arbiter PUF

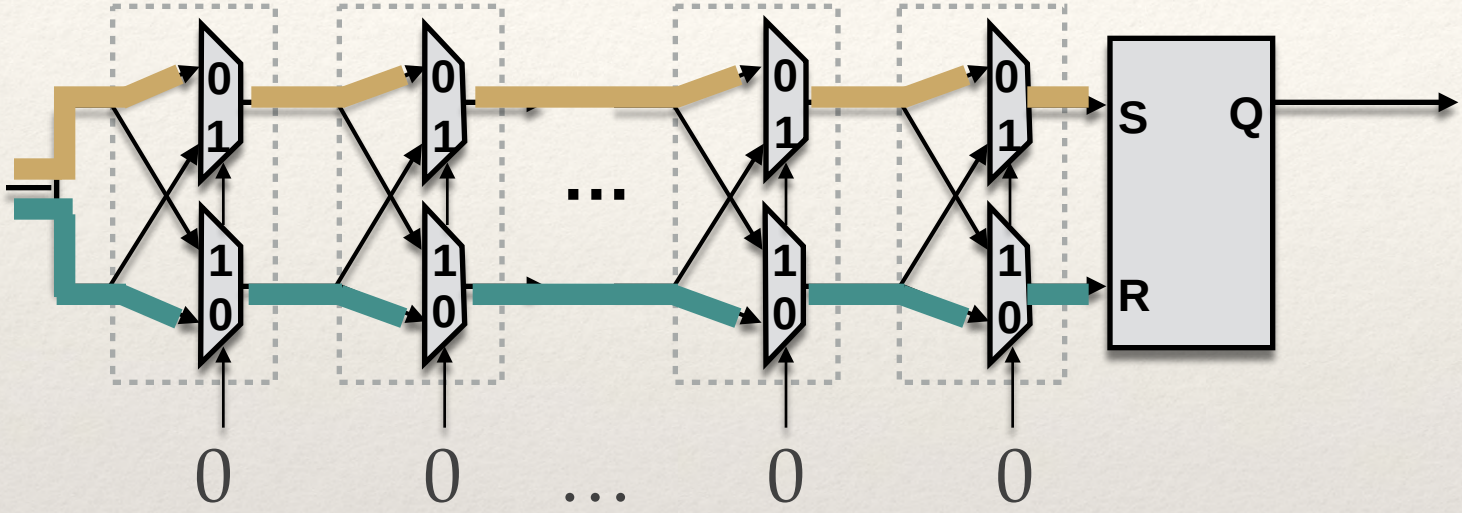
[B Gassend et al., '02] [D. Lim et al., '05]



- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 0,1$
- Uses variations in subcomponent delays

Arbiter PUF

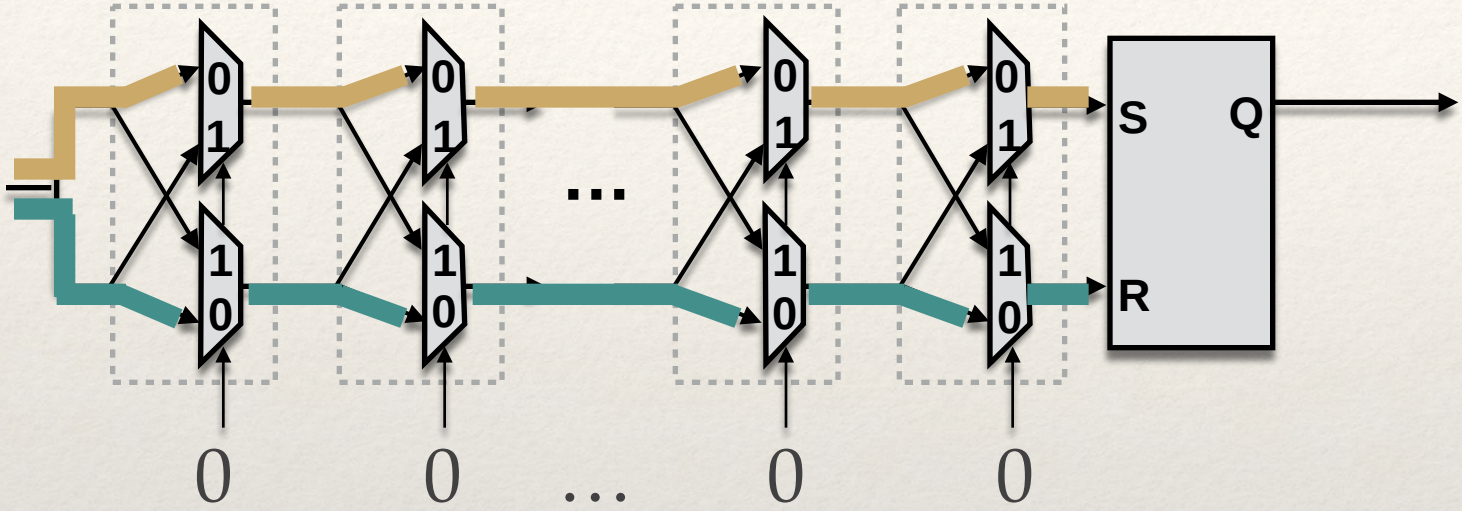
[B Gassend et al., '02] [D. Lim et al., '05]

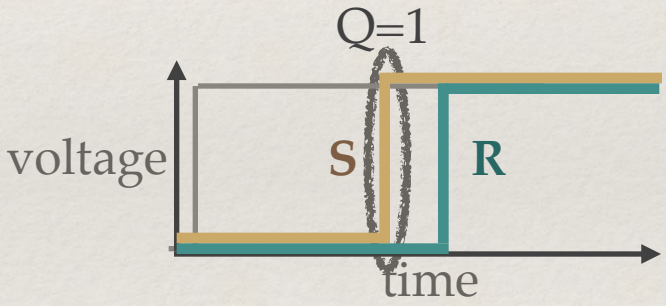


- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 0,1$
- Uses variations in subcomponent delays

Arbiter PUF

[B Gassend et al., '02] [D. Lim et al., '05]

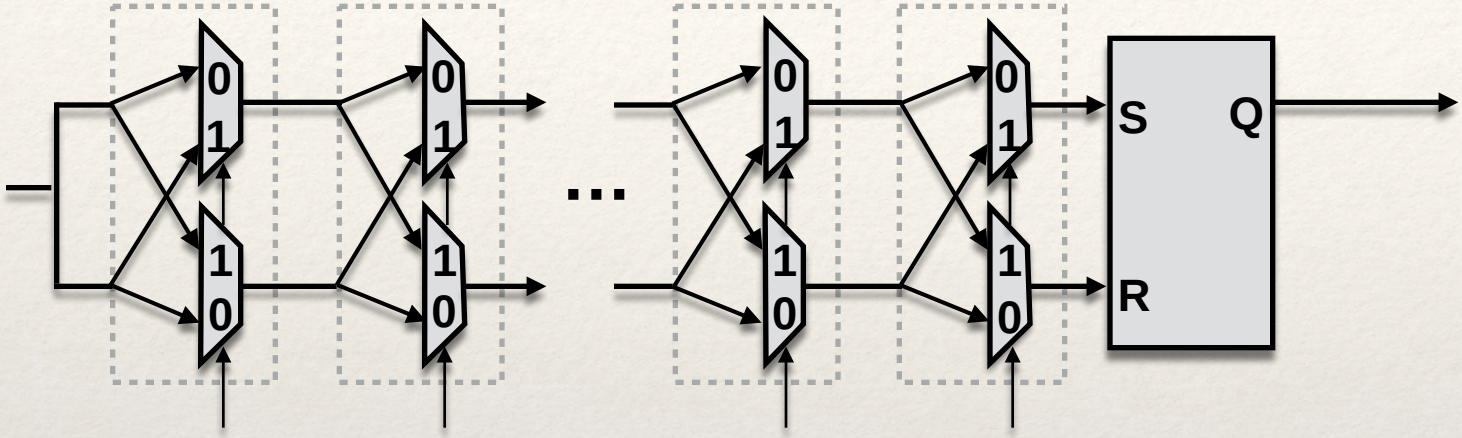


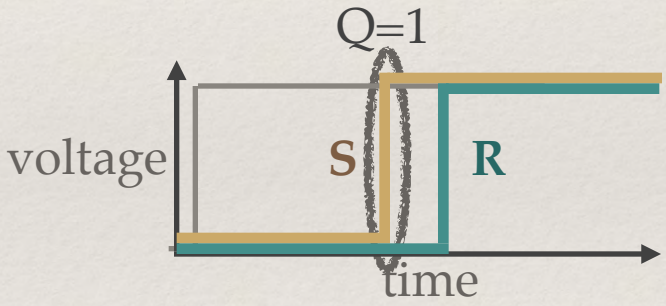


- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 0,1$
- Uses variations in subcomponent delays

Arbiter PUF

[B Gassend et al., '02] [D. Lim et al., '05]

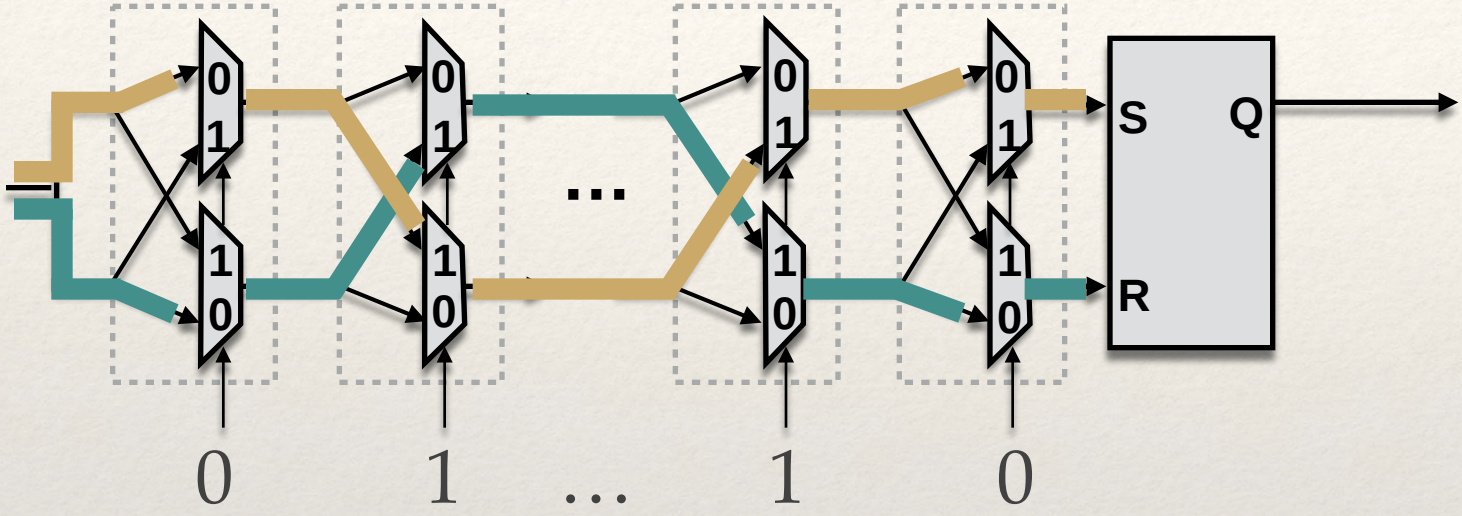


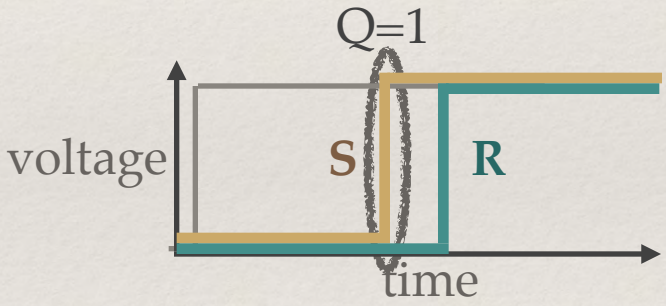


- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 0,1$
- Uses variations in subcomponent delays

Arbiter PUF

[B Gassend et al., '02] [D. Lim et al., '05]

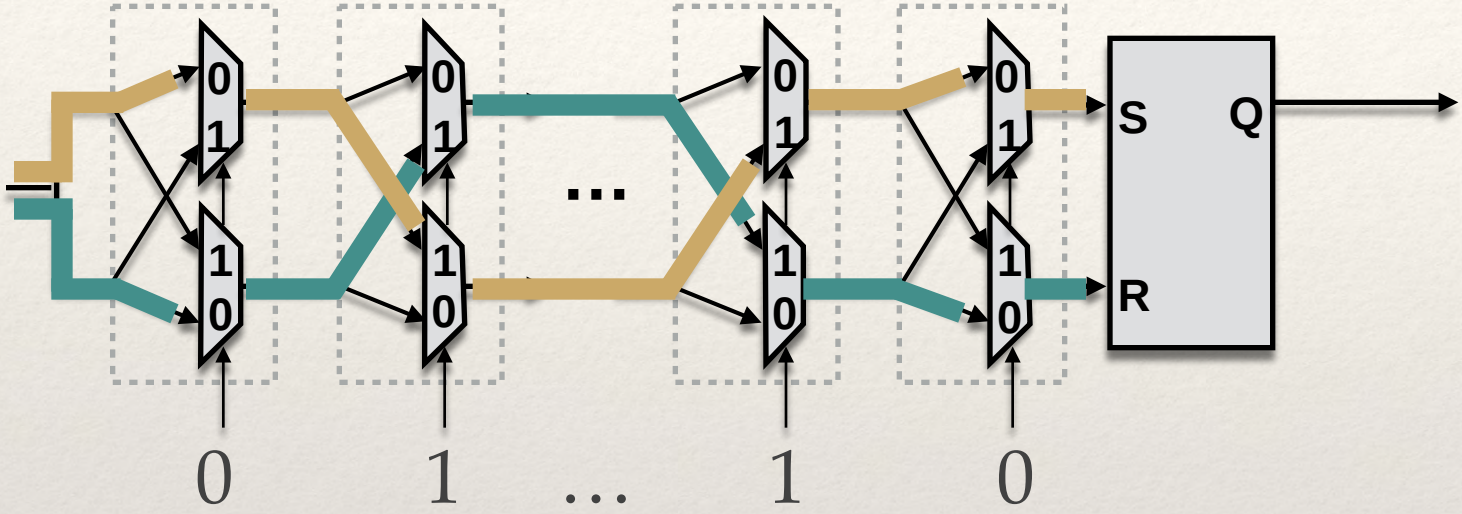




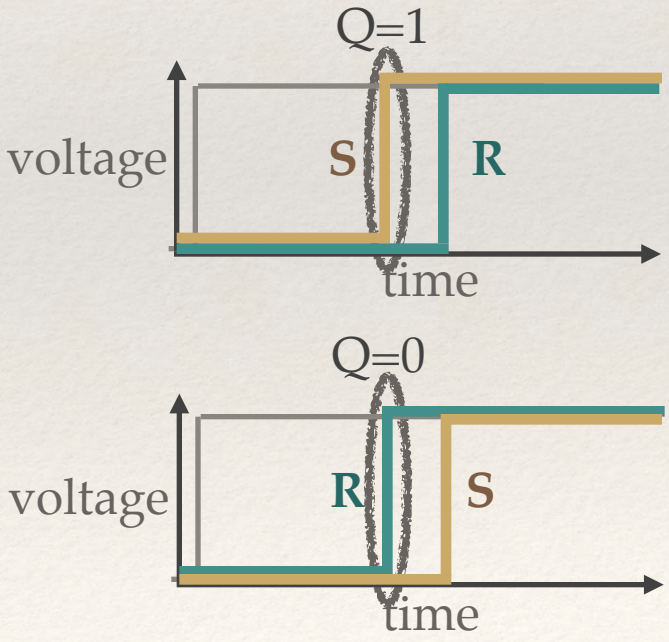
- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 0,1$
- Uses variations in subcomponent delays

Arbiter PUF

[B Gassend et al., '02] [D. Lim et al., '05]

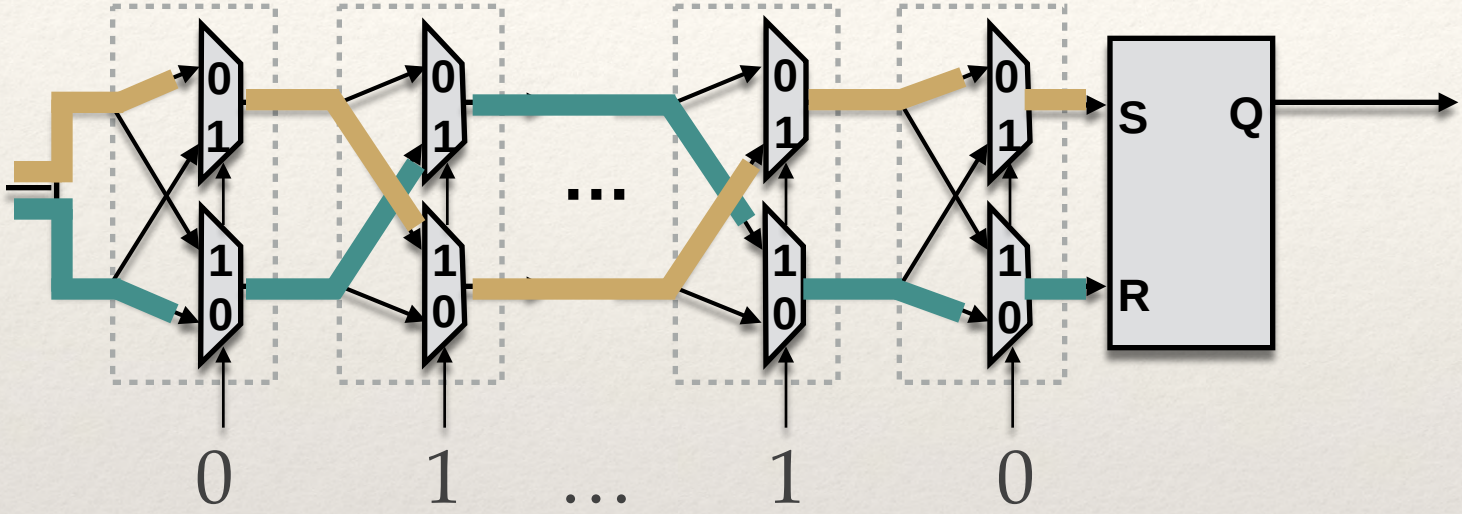


- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 0,1$
- Uses variations in subcomponent delays

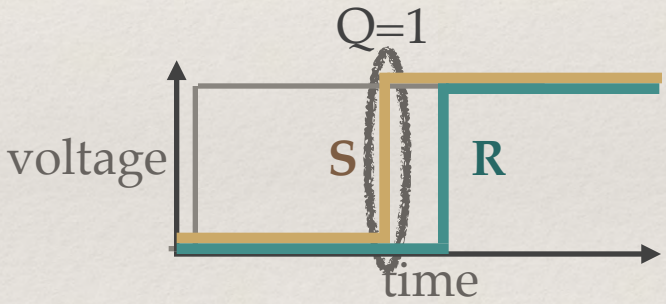


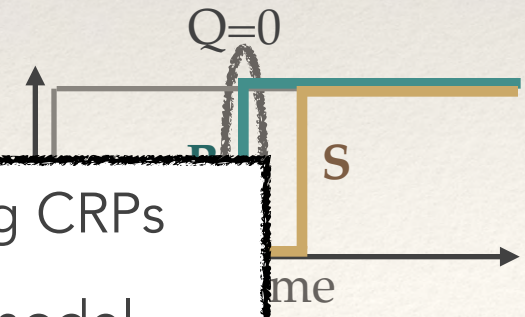
Arbiter PUF

[B Gassend et al., '02] [D. Lim et al., '05]



- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 0,1$
- Uses variations in subcomponent delays



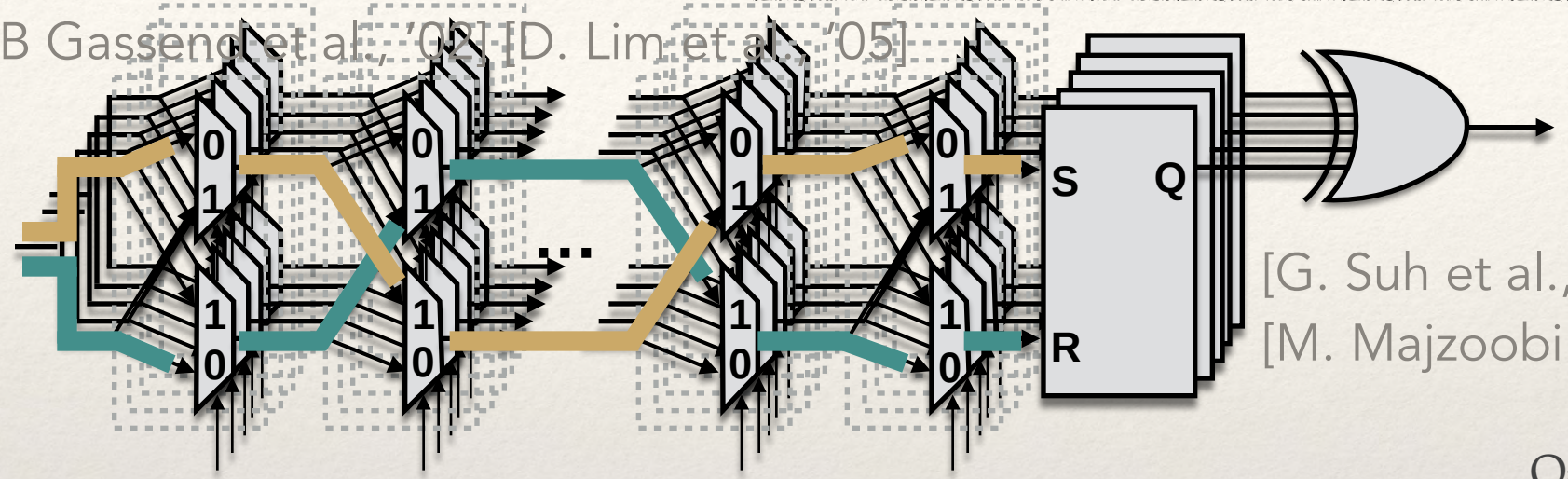


- Assumes that model cannot be created by observing CRPs
- But basic arbiter PUF susceptible to additive delay model

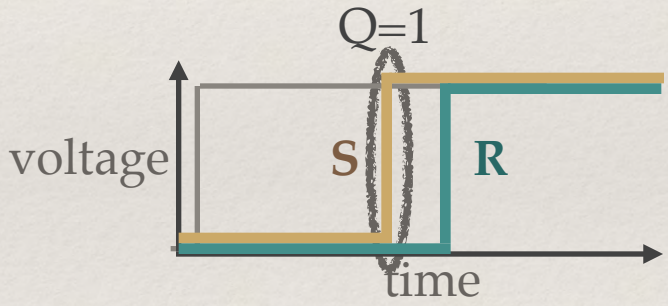
Arbiter PUF

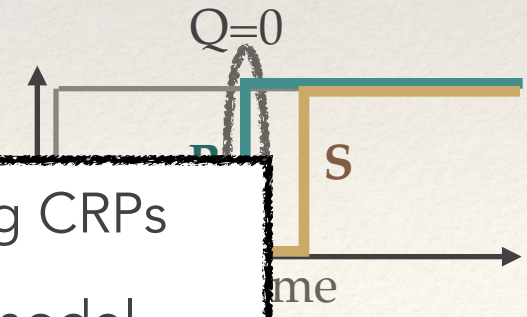
- XOR Arbiter PUF resists additive model

[B Gassend et al., '02] [D. Lim et al., '05]



[G. Suh et al., '07]
[M. Majzoobi et al., '08]





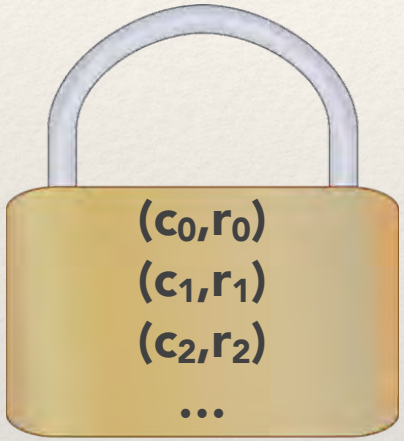
- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 0,1$
- Uses variations in subcomponent delays

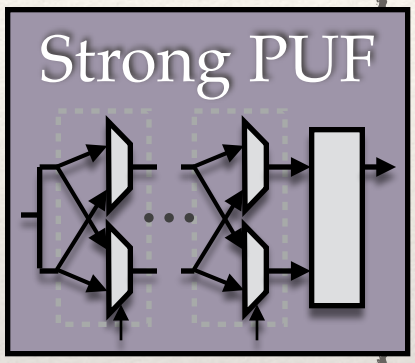
- Assumes that model cannot be created by observing CRPs
- But basic arbiter PUF susceptible to additive delay model

Authentication using Strong PUF

Enroll PUF

- Choose random challenges
- Apply and store private CRPs

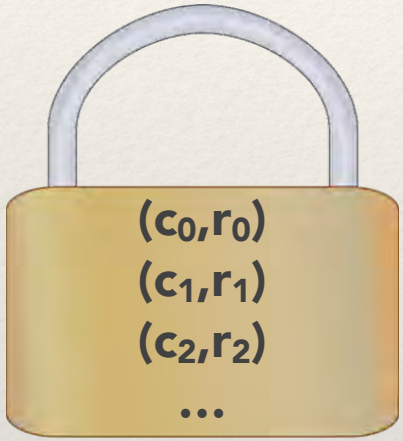




Authentication using Strong PUF

Enroll PUF

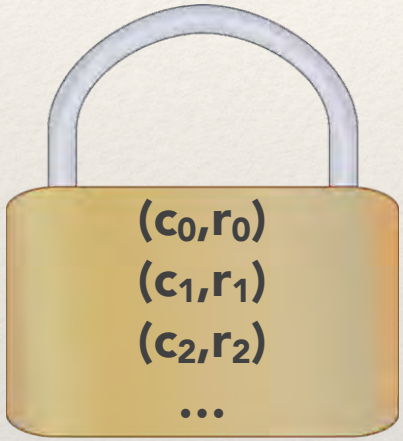
- Choose random challenges
- Apply and store private CRPs

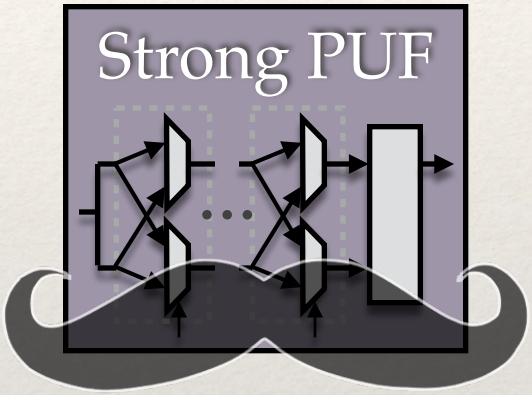


Authentication using Strong PUF

Enroll PUF

- Choose random challenges
- Apply and store private CRPs

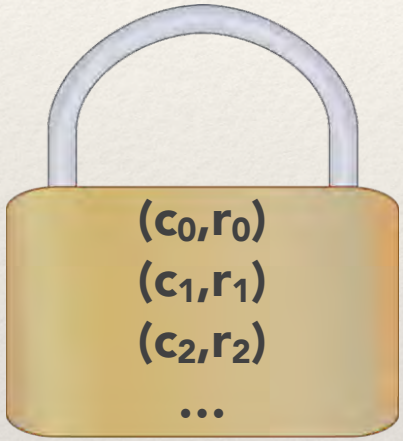




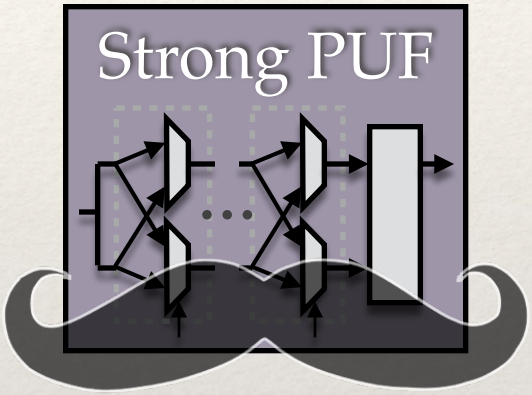
Authentication using Strong PUF

Enroll PUF

- Choose random challenges
- Apply and store private CRPs



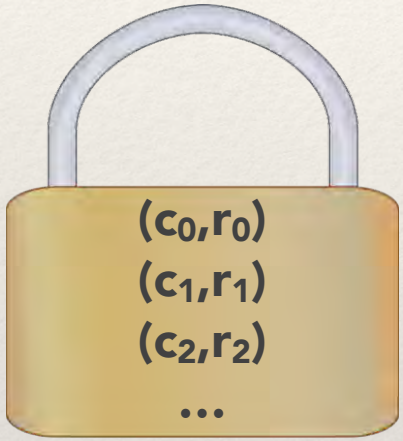
c_0



Authentication using Strong PUF

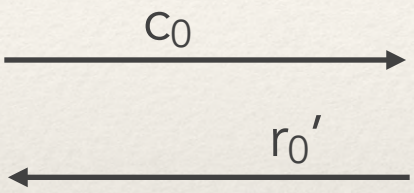
Enroll PUF

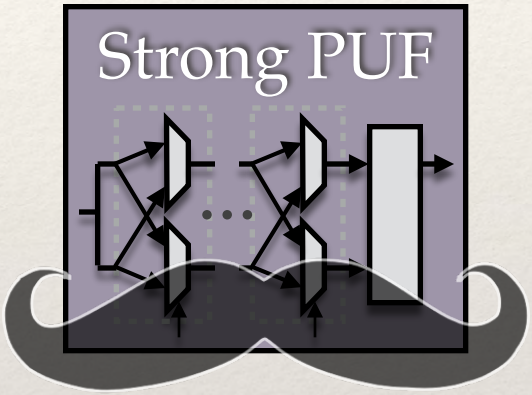
- Choose random challenges
- Apply and store private CRPs



Authenticate

$r_0 \approx r_0' ?$





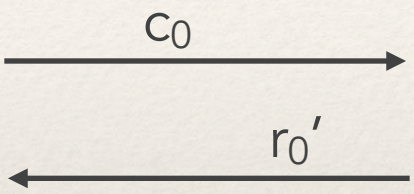
Authentication using Strong PUF

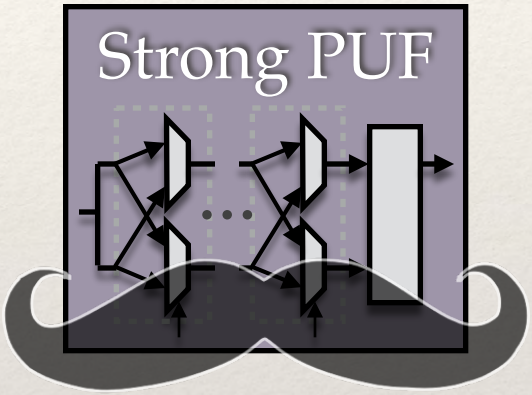
Enroll PUF

- Choose random challenges
- Apply and store private CRPs

Authenticate

$r_0 \approx r_0' ?$





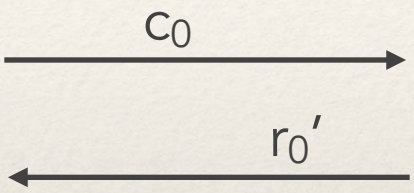
Authentication using Strong PUF

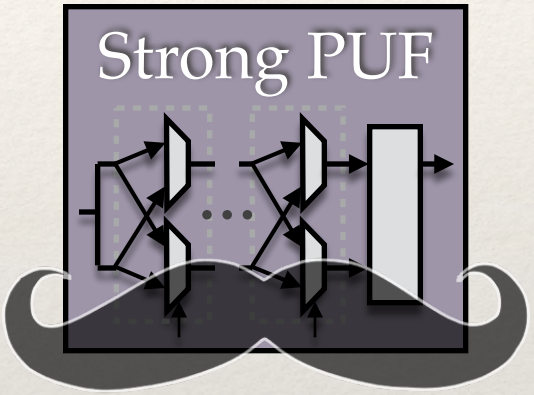
Enroll PUF

- Choose random challenges
- Apply and store private CRPs

Authenticate

$r_0 \approx r_0'$?

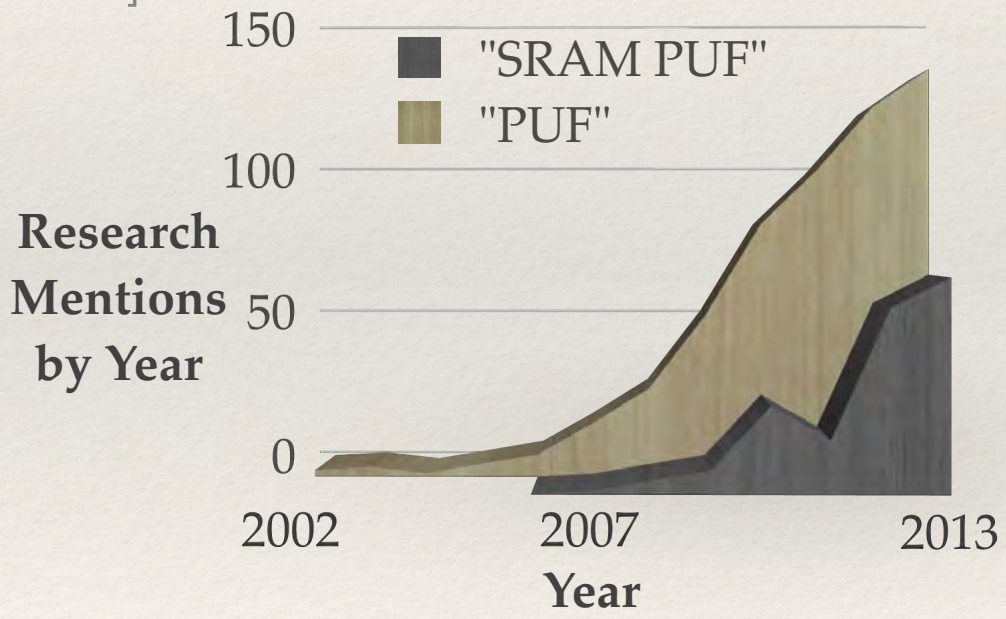




- Responses can be public if PUF resists modeling

Examples of Weak PUFs

- Using custom circuits
  - Drain currents [Lofstrom et al. '02]
  - Capacitive coating PUF [Tuyls et al. '06]
  - Cross-coupled devices [Su et al. '07]
  - Sense amps [Bhargava et al. '10]
- Using existing circuits
  - Clock skew [Yao et al.'13]
  - Flash latency [Prabhu et al. '11]
  - Power-up SRAM state [Guajardo et al. '07, Holcomb et al. '07]



Applications of Weak PUFs

- Identification
- Authentication
- Secret key
- Random number generation

RFID Security 2007
IEEE Transactions on Computer 2009

SRAM Power-up State

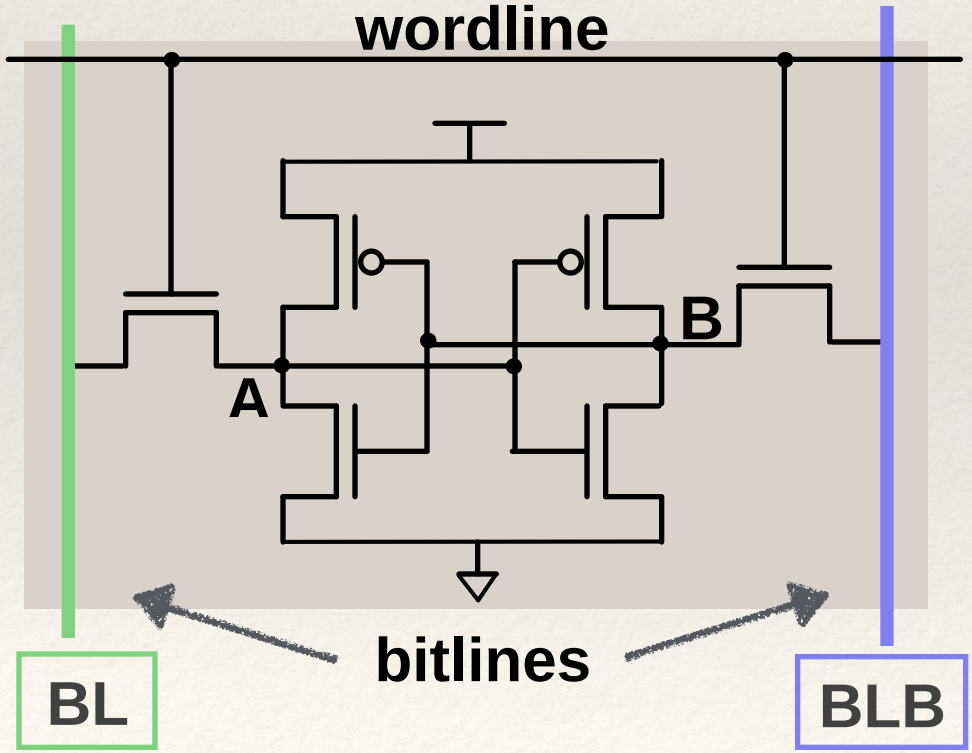
Using Retention voltage of SRAM cells as a signature

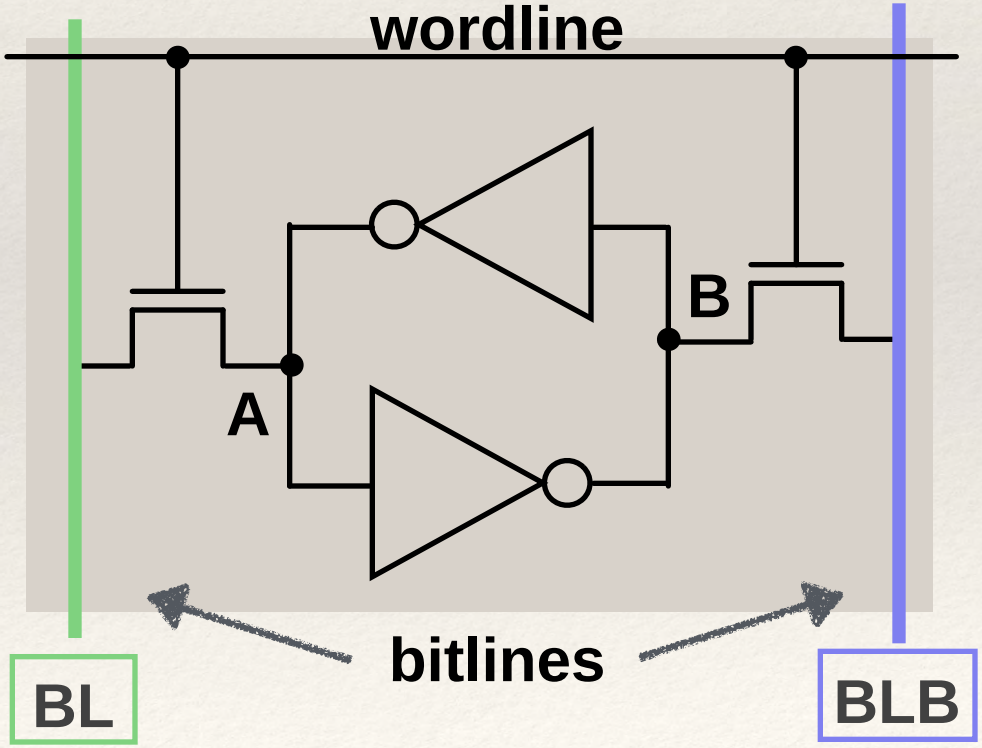
Daniel E. Holcomb
Kevin Fu
Wayne Burleson

See also:
Guajardo et al., CHES'07
Intrinsic ID

6-Transistor SRAM Cell

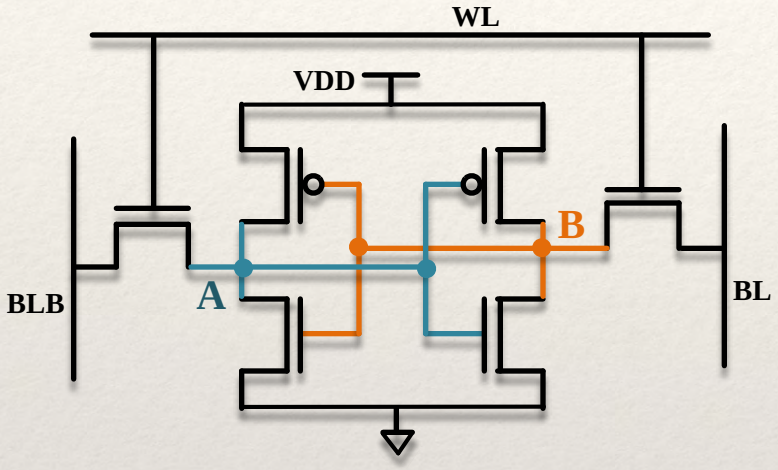
- Ubiquitous memory
- Two stable states: “0” (AB=01) “1” (AB=10)
- **Wordline** selects a cell for reading/writing
- Complementary **bitlines** read/write values to/from selected cells

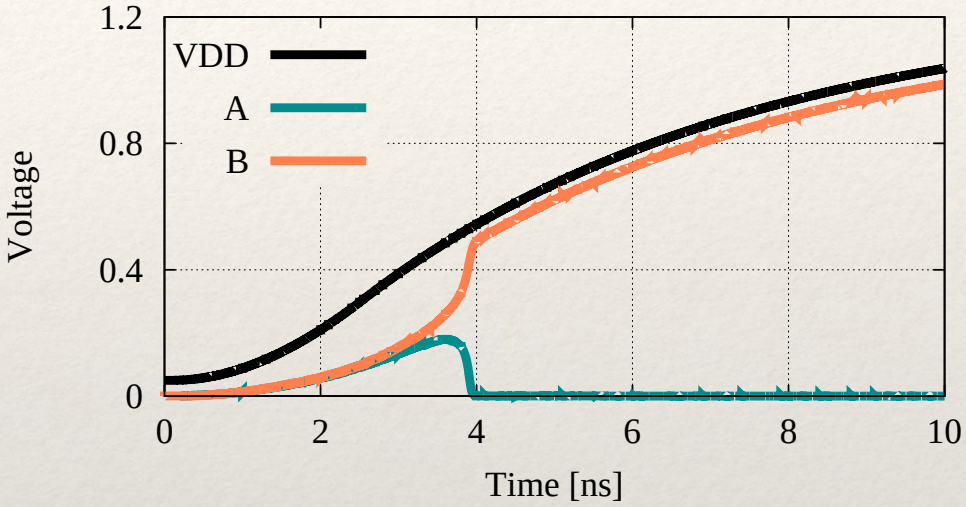




SRAM Power-up State

Utilize inherent power-up bias of each SRAM cell

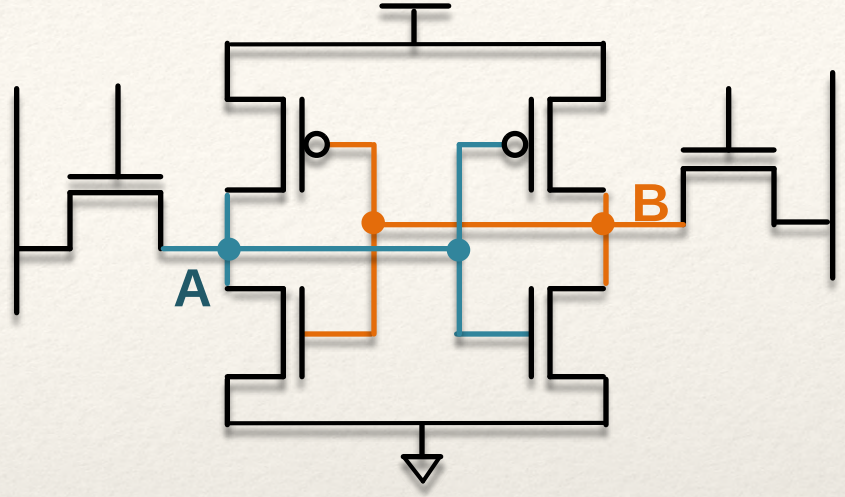


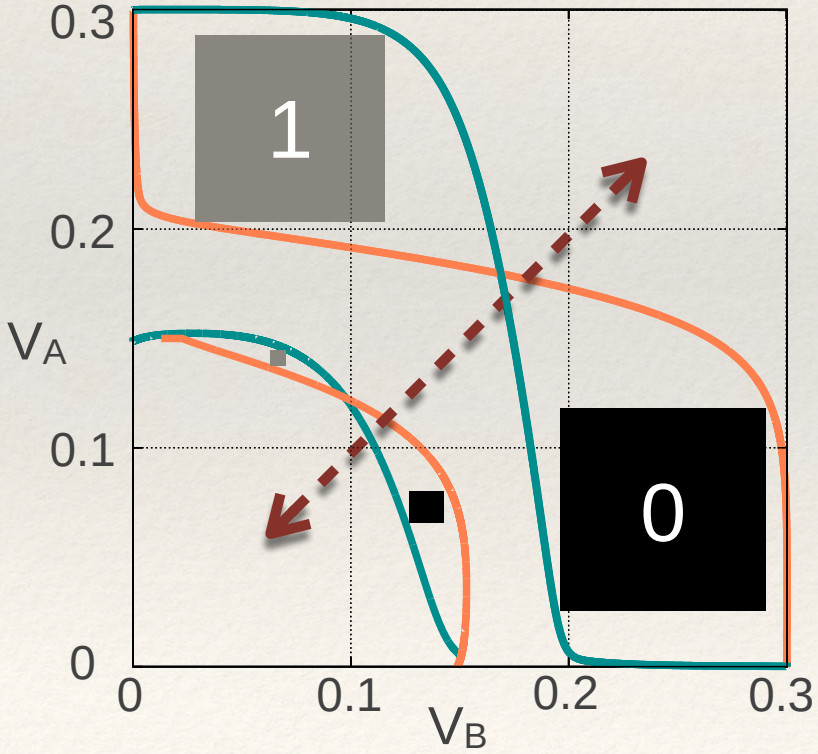


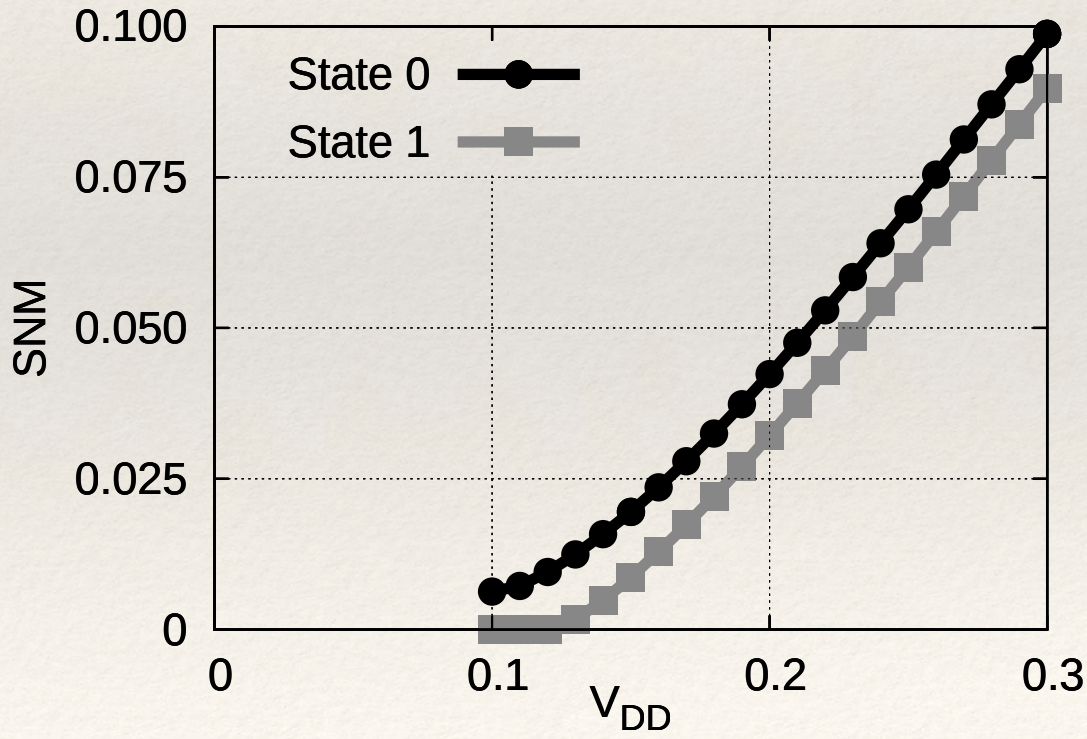
- No challenge other than cell selection
- Responses: $r \in 2^n$ (power-up state of n cells)
- Behavior from threshold variation of transistors in cell

SRAM Variation

- Static noise margin
  [Seevink et al., 1987]
- Sets lower bound on safe V_{DD}

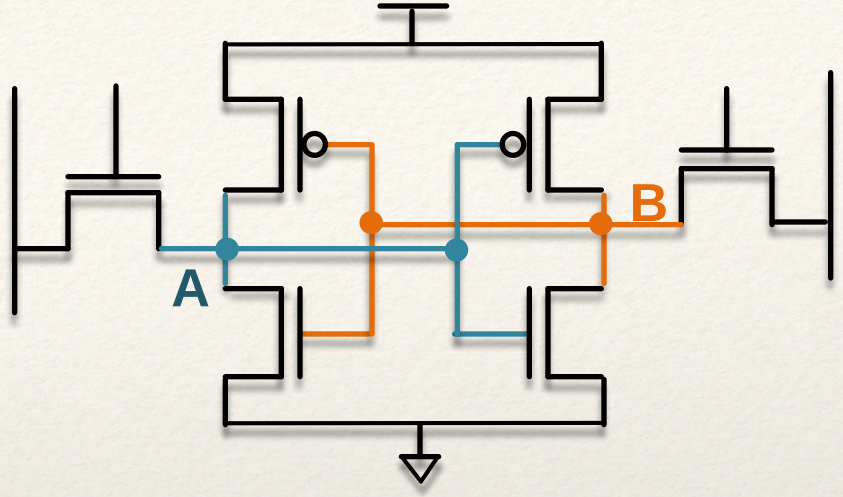


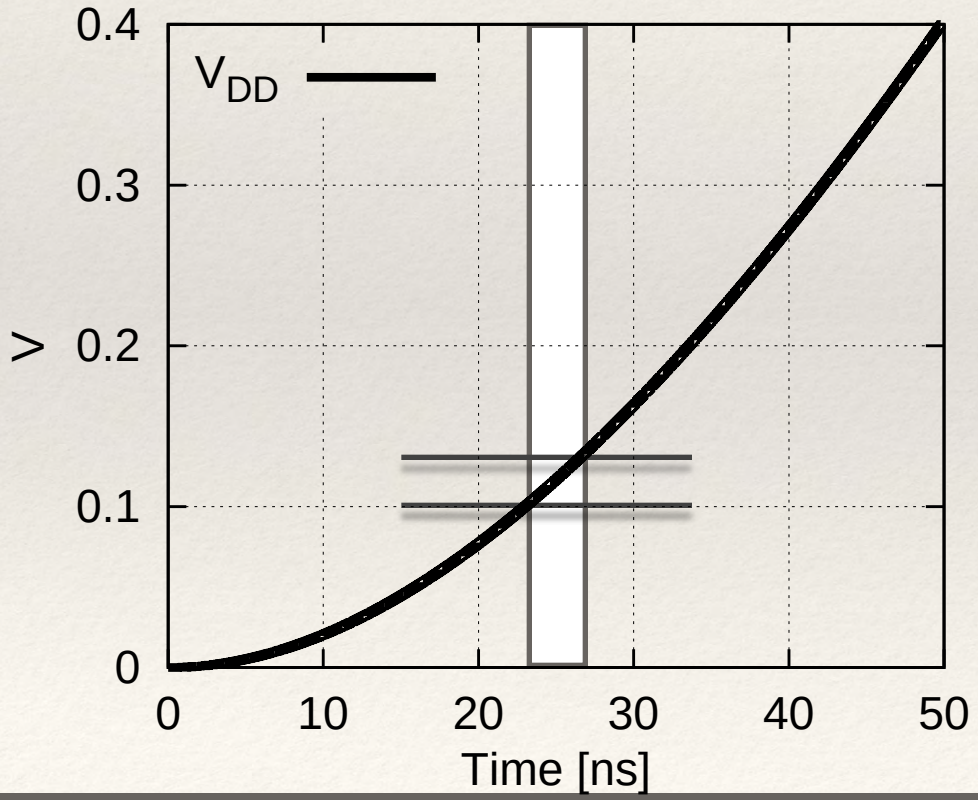


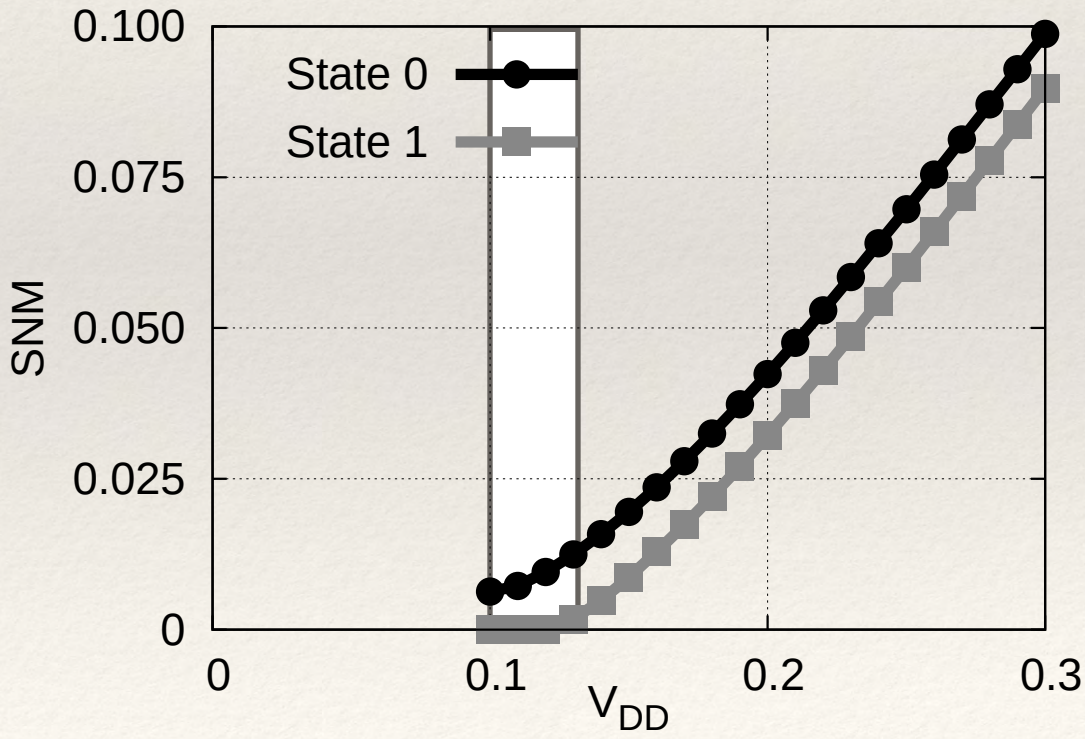


SRAM Power-up

- Power-up sensitive to variations
- Uncorrelated across cells and chips
- Persistent

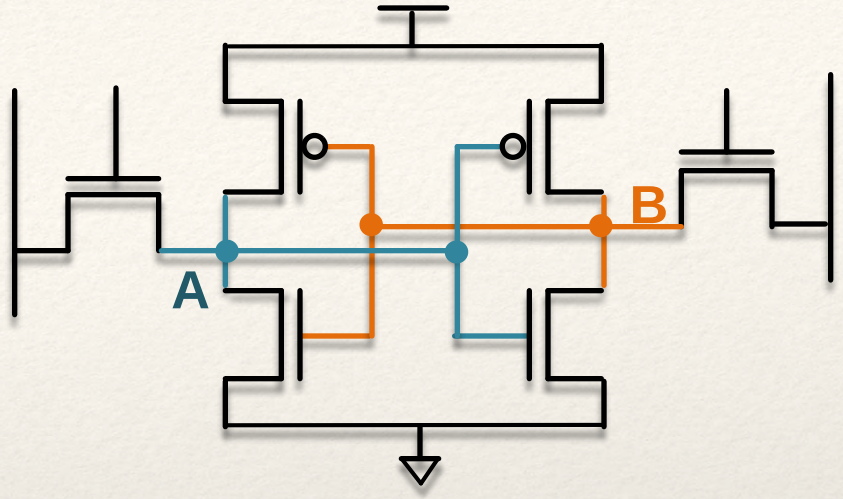


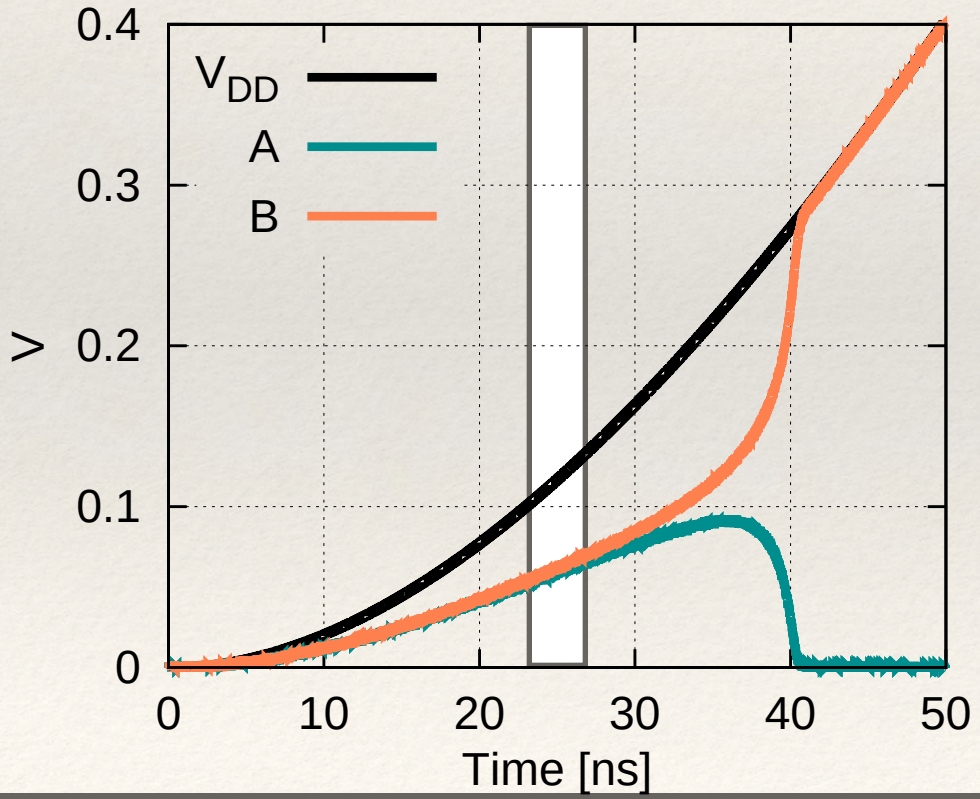


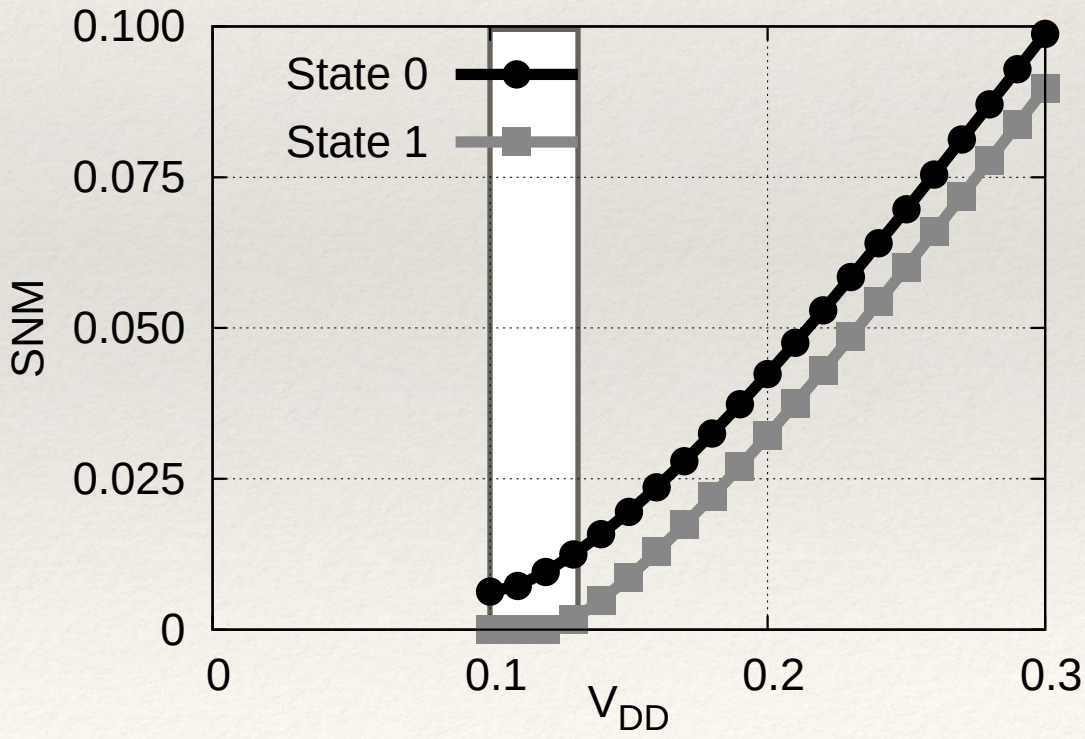


SRAM Power-up

- Power-up sensitive to variations
- Uncorrelated across cells and chips
- Persistent

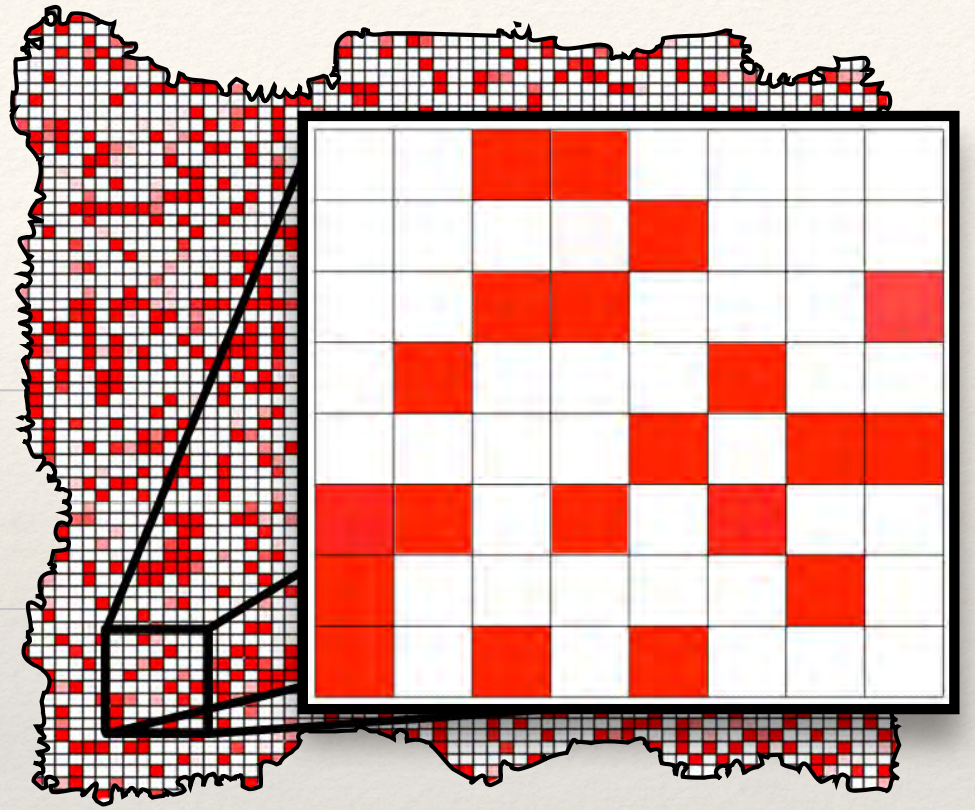




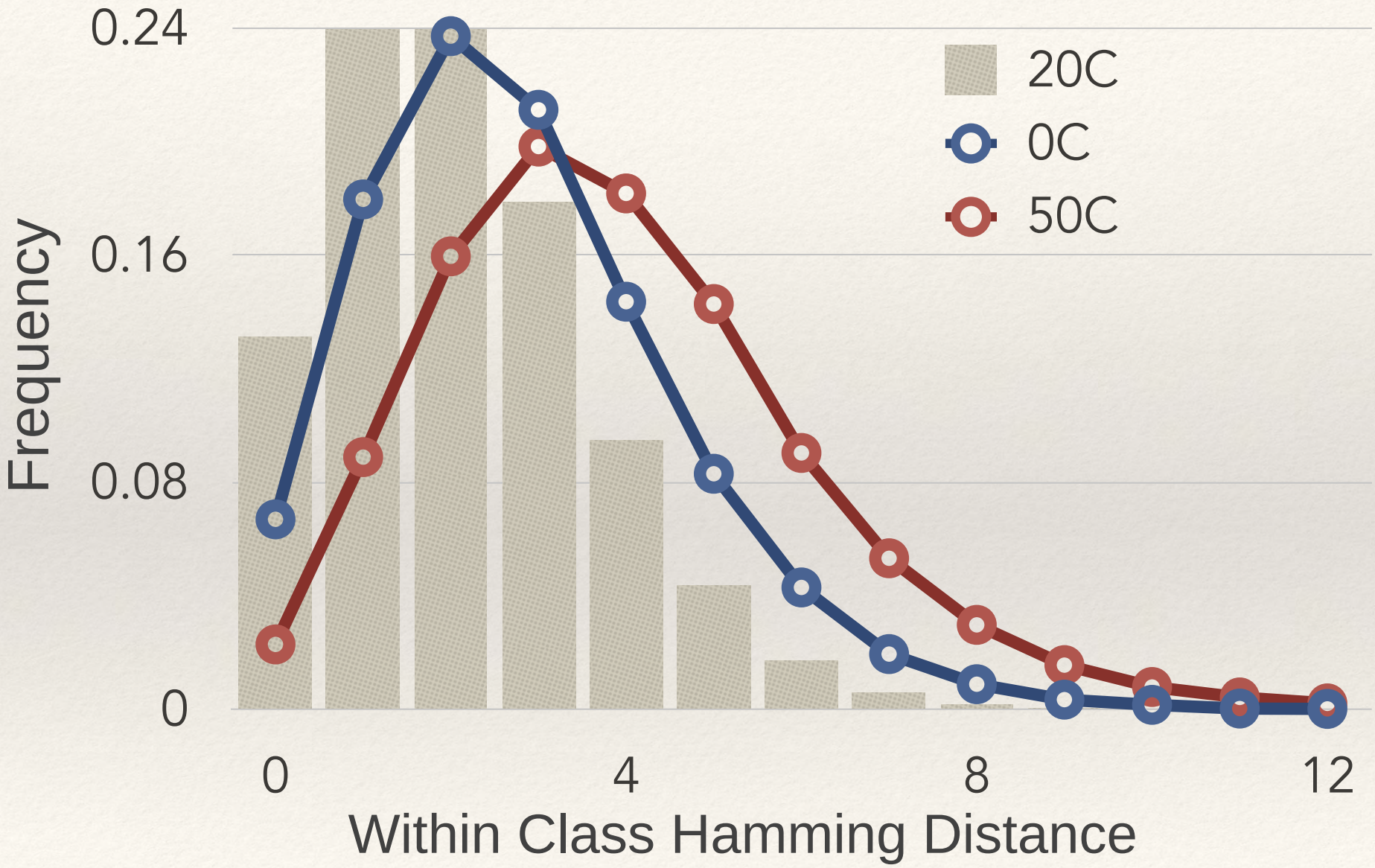


Power-up Fingerprint

- 64-bit fingerprints
- Population size of 5,120

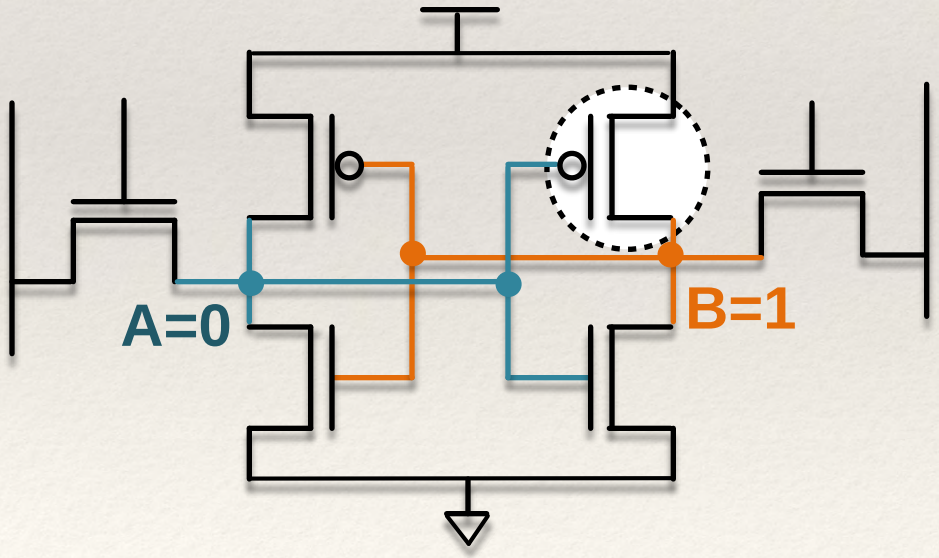


Temperature



NBTI Aging

- Stored state impacts subsequent power-up tendency
- Favors opposite of stored state
- Possible directed attack
- Recovery after stress removed



- Directed aging can improve reliability
- Constructively bias cells away from metastability

  [Bhargava et al. HOST'12]

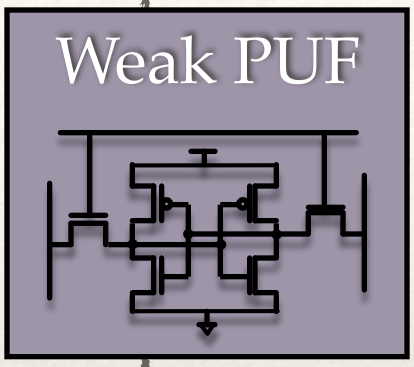
  [Mathew et al. ISSCC'14]

Power-up State PUF as Secret Key

Enroll PUF at Manufacture

- Learn response r
- Choose key k and derive public helper data h:

  h = Encode(k) ⊕ r



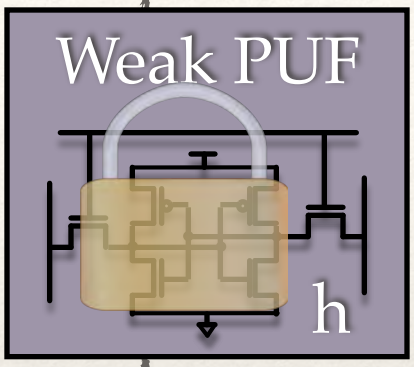
code offset construction
[Dodis et al. '08]

Power-up State PUF as Secret Key

Enroll PUF at Manufacture

- Learn response r
- Choose key k and derive public helper data h:

  h = Encode(k) ⊕ r
- Store h with PUF
- Disable access to response r



code offset construction
[Dodis et al. '08]

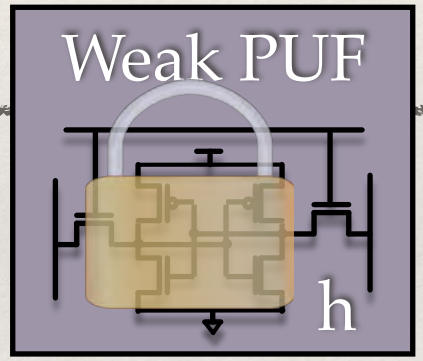
Power-up State PUF as Secret Key

Enroll PUF at Manufacture

- Learn response r
- Choose key k and derive public helper data h:

  $h = \text{Encode}(k) \oplus r$
- Store h with PUF
- Disable access to response r

Generate Key in Field



code offset construction
[Dodis et al. '08]

Power-up State PUF as Secret Key

Enroll PUF at Manufacture

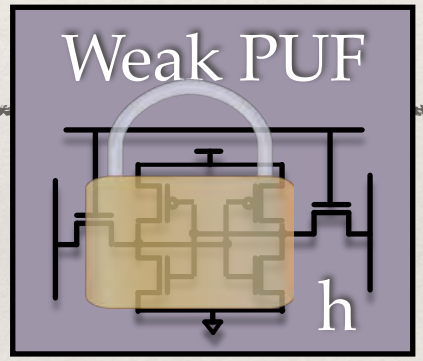
- Learn response r
- Choose key k and derive public helper data h:

  h = Encode(k) ⊕ r
- Store h with PUF
- Disable access to response r

k is reliable key

Generate Key in Field

- Measure r′ ⊕ h
- Key k = Decode(r′ ⊕ h)



code offset construction
[Dodis et al. '08]

Power-up State PUF as Secret Key

Enroll PUF at Manufacture

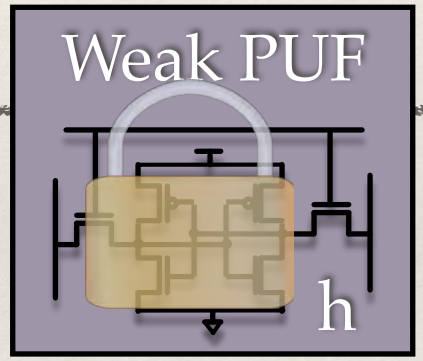
- Learn response r
- Choose key k and derive public helper data h:

  h = Encode(k) ⊕ r
- Store h with PUF
- Disable access to response r

k is reliable key

Generate Key in Field

- Measure r′ ⊕ h
- Key k = Decode(r′ ⊕ h)



- Reliable unclonable key for crypto
- Assumes that r cannot be read in field

code offset construction [Dodis et al. '08]

RFID Security 2012

IEEE Transactions on CAD 2015

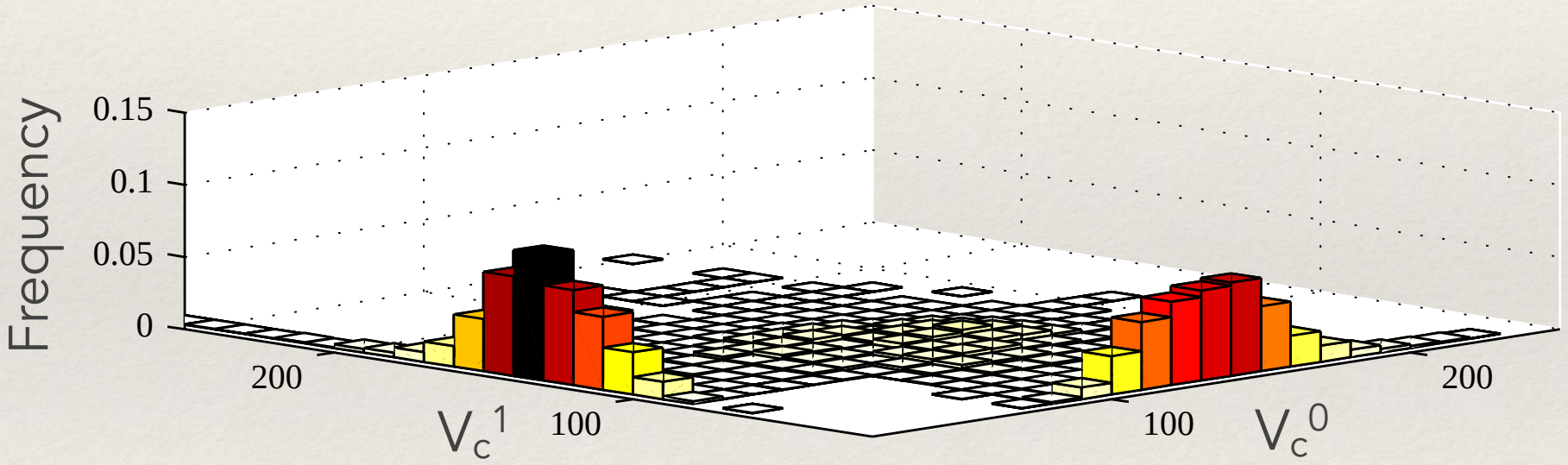
DRV Fingerprinting

Using Retention voltage of SRAM cells as a signature

Daniel Holcomb
Xiaolin Xu
Amir Rahmati
Negin Salajegheh
Kevin Fu
Wayne Burleson

DRV Fingerprint Matching

- Fingerprint of cell is a pair [V_c^0 , V_c^1]
  - V_c^0 : Highest voltage that causes flip from 0 state
  - V_c^1 : Highest voltage that causes flip from 1 state

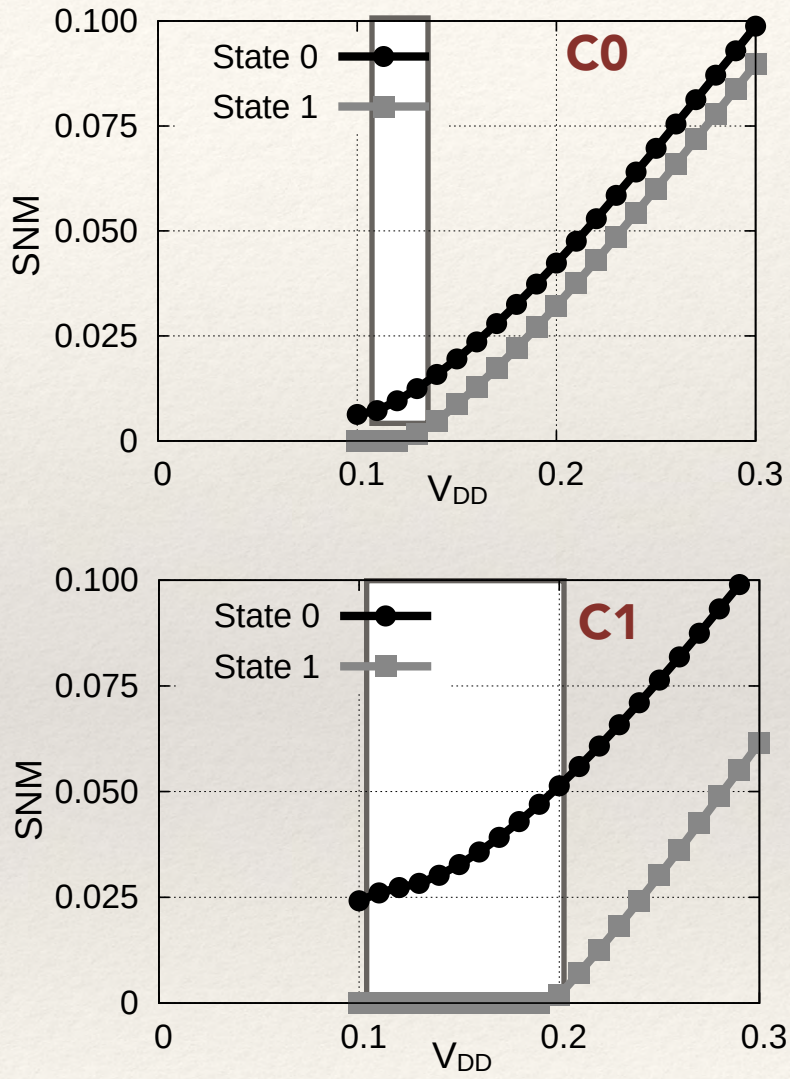


- Identification using Euclidean distance matching

	Unique Correct Match
DRV	99.7%
Power-up	71.7%

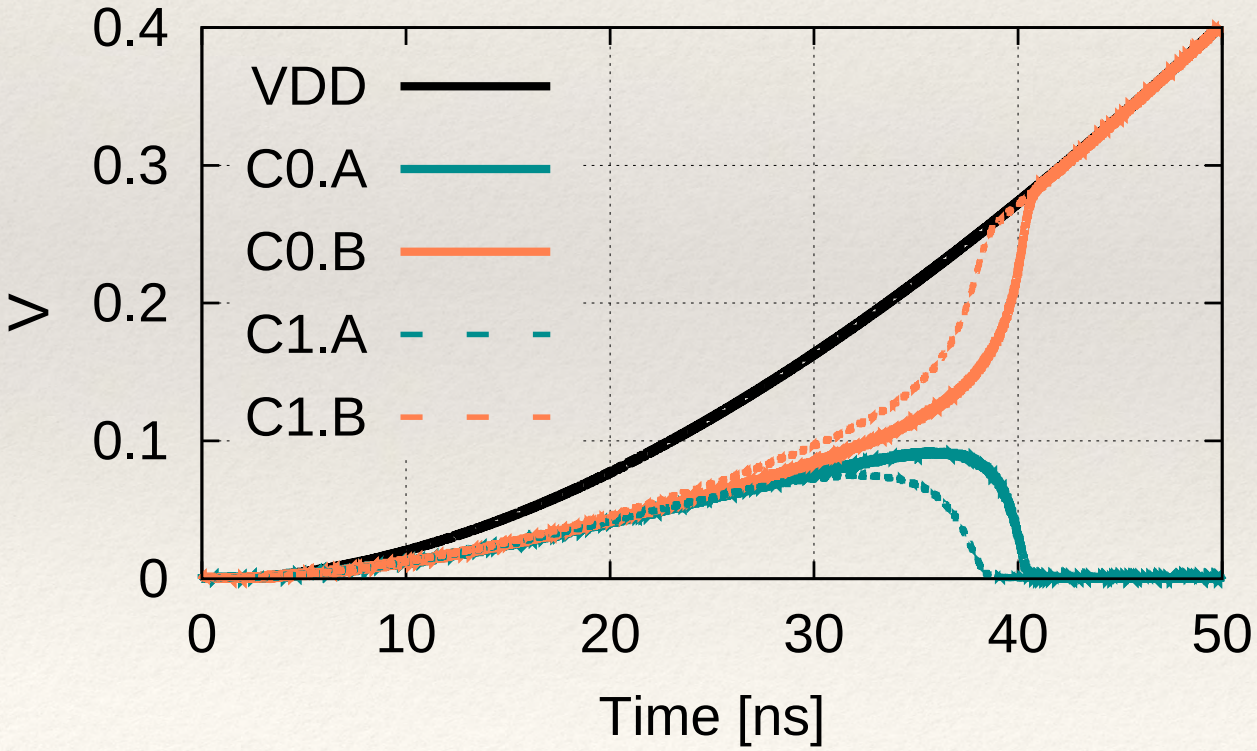
Data Retention Voltage

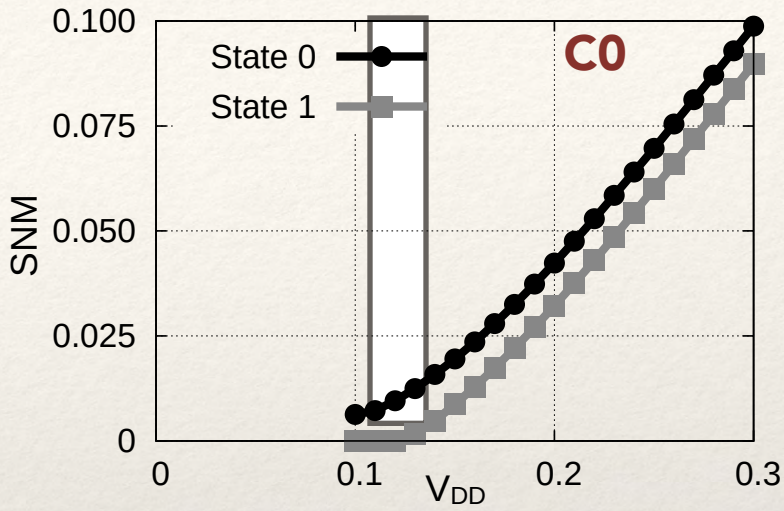
- More informative than power-up state

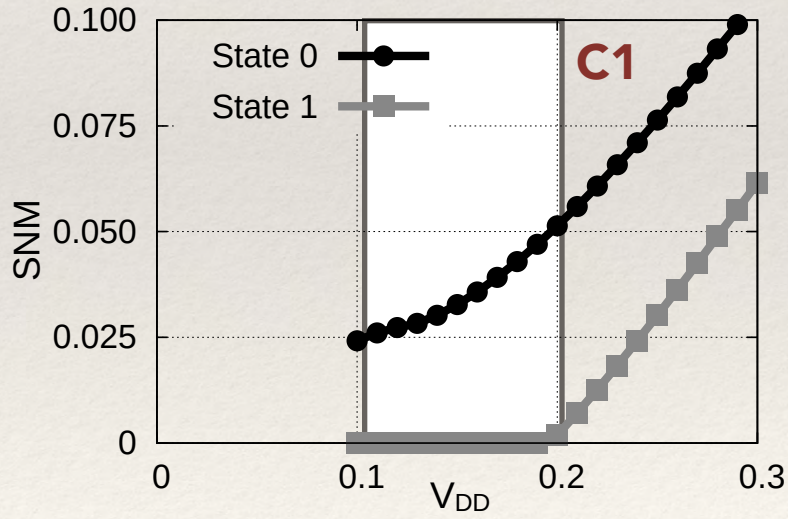


Data Retention Voltage

- More informative than power-up state

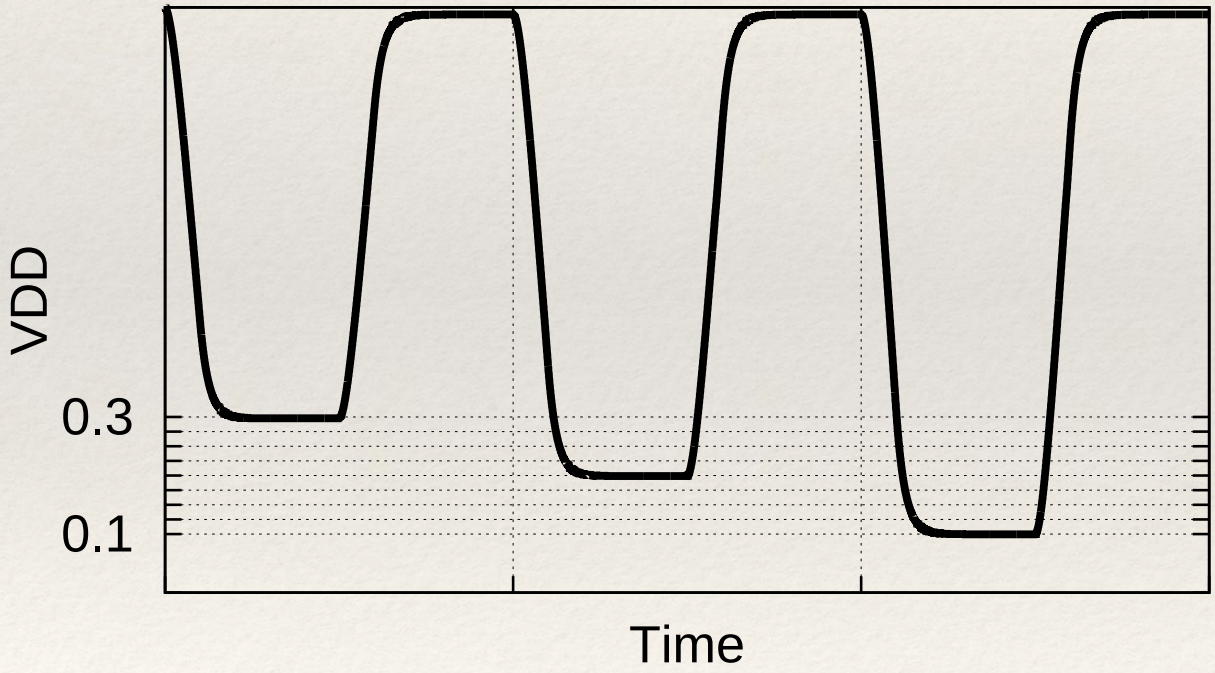


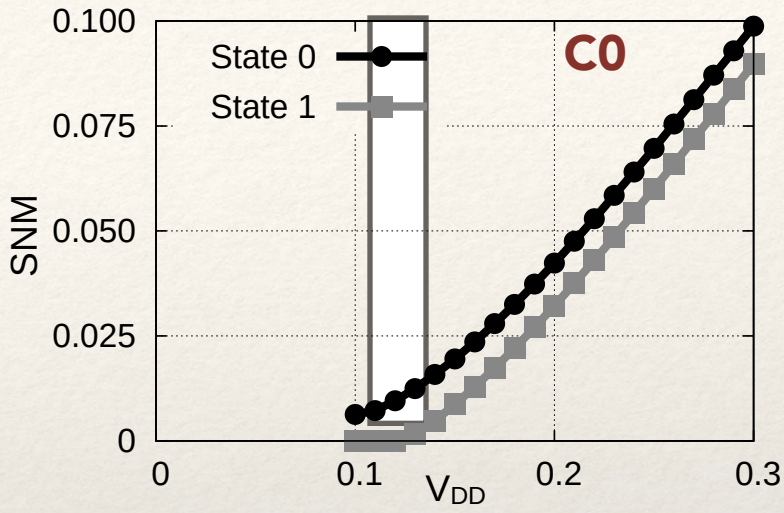


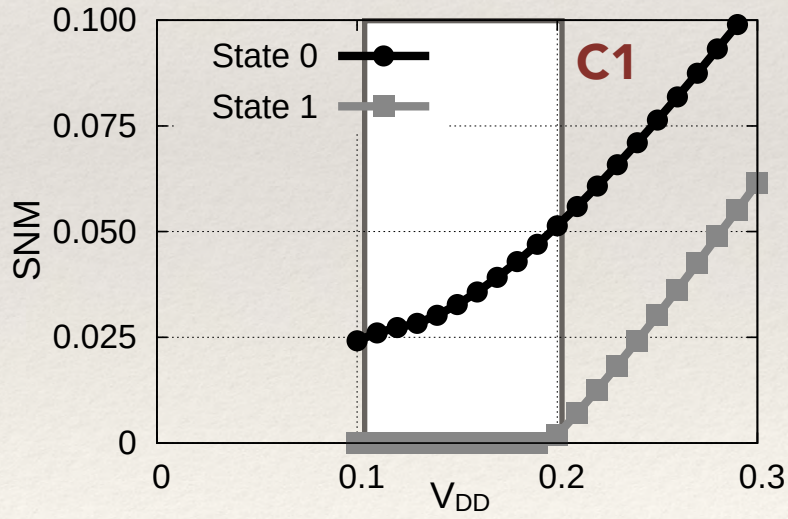


Data Retention Voltage

- More informative than power-up state

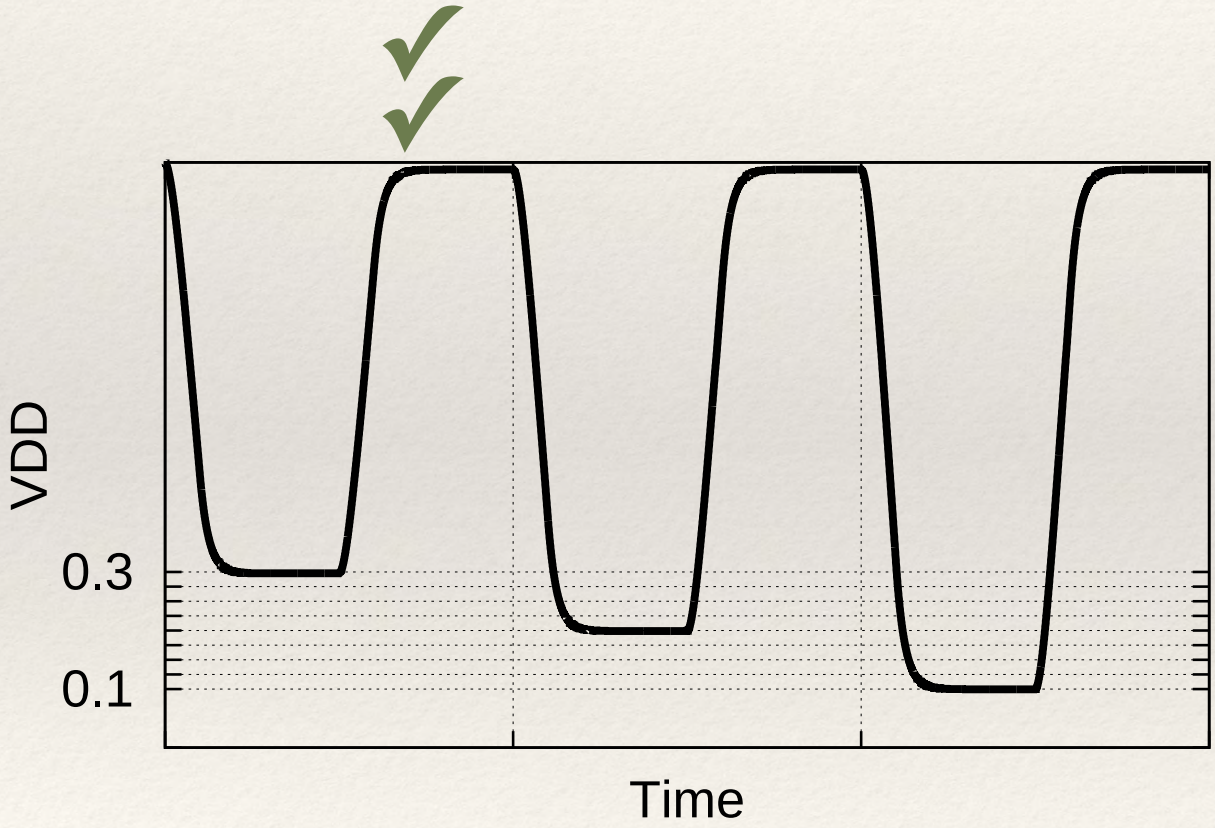


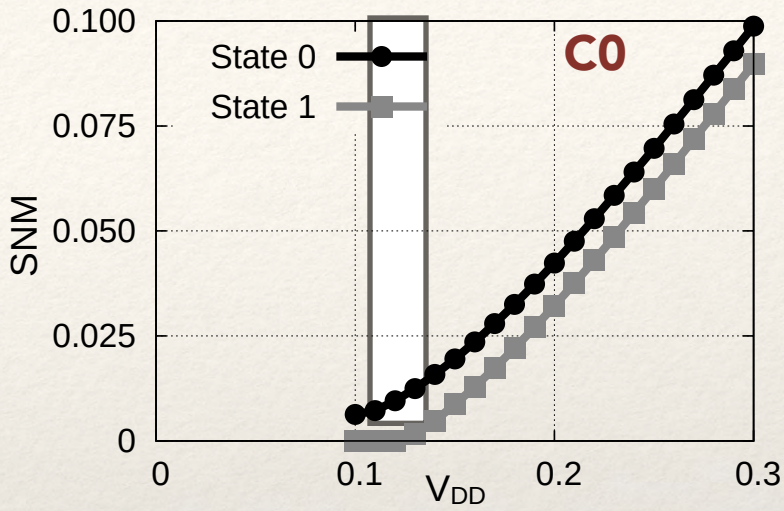


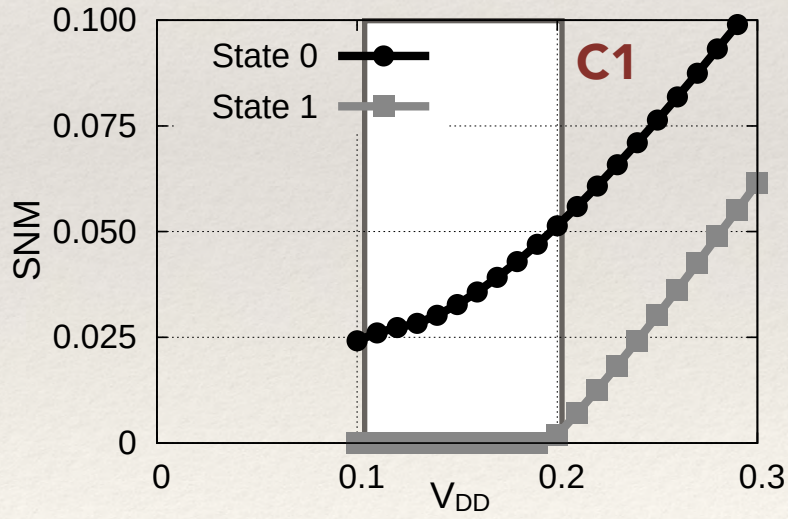


Data Retention Voltage

- More informative than power-up state

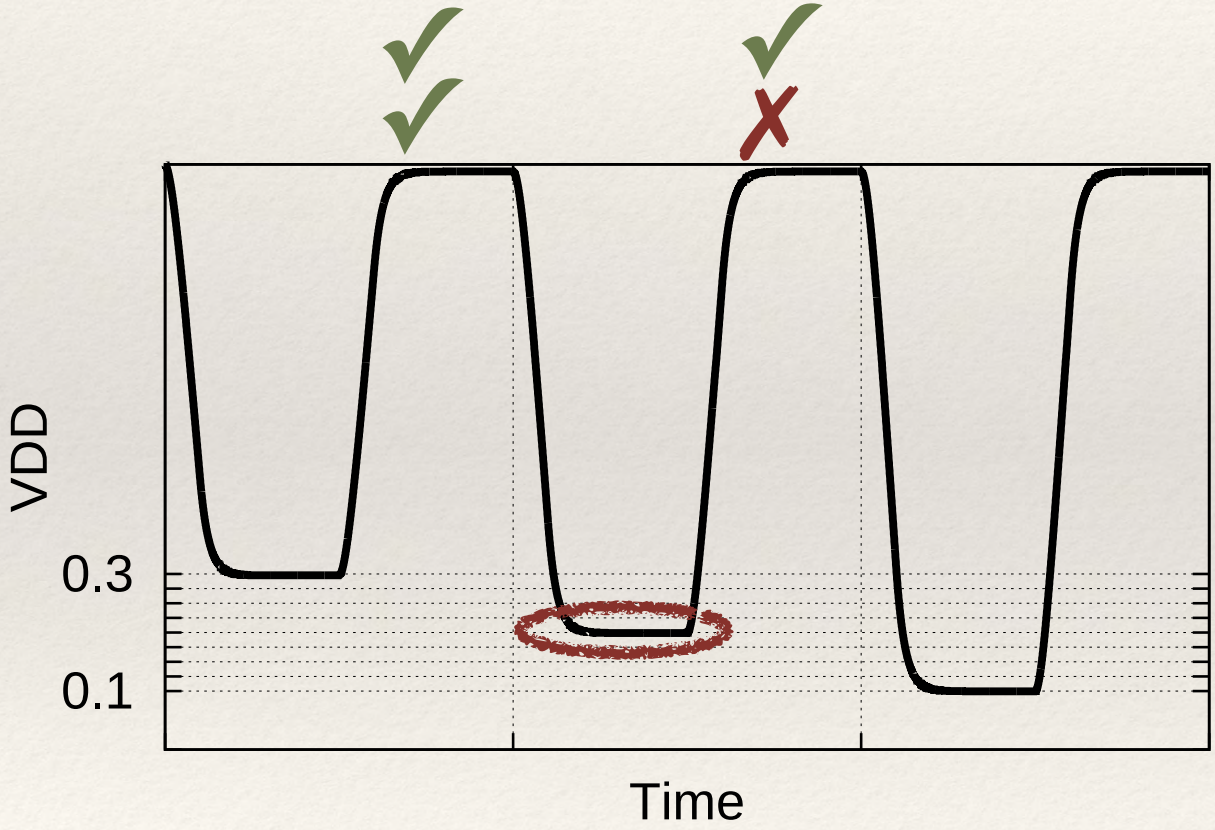


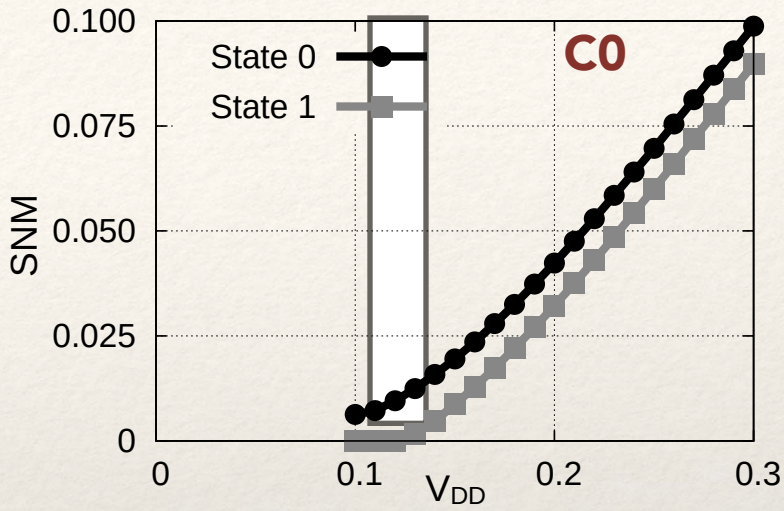


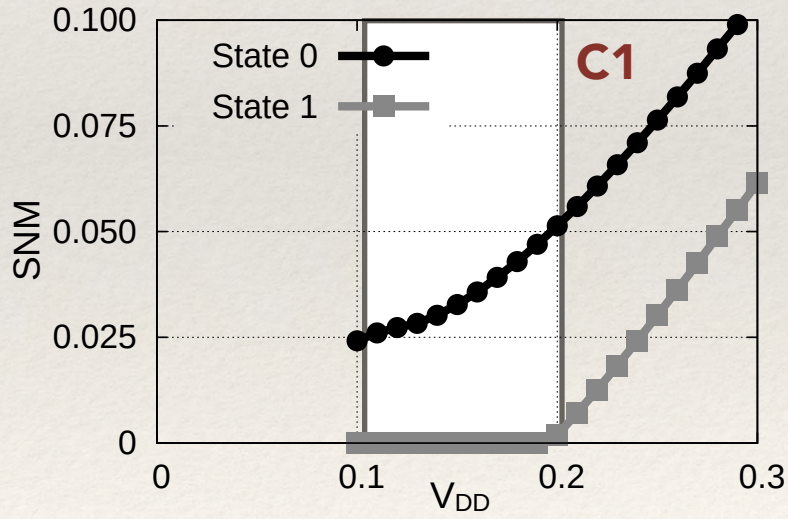


Data Retention Voltage

- More informative than power-up state
- Support from non-volatile storage

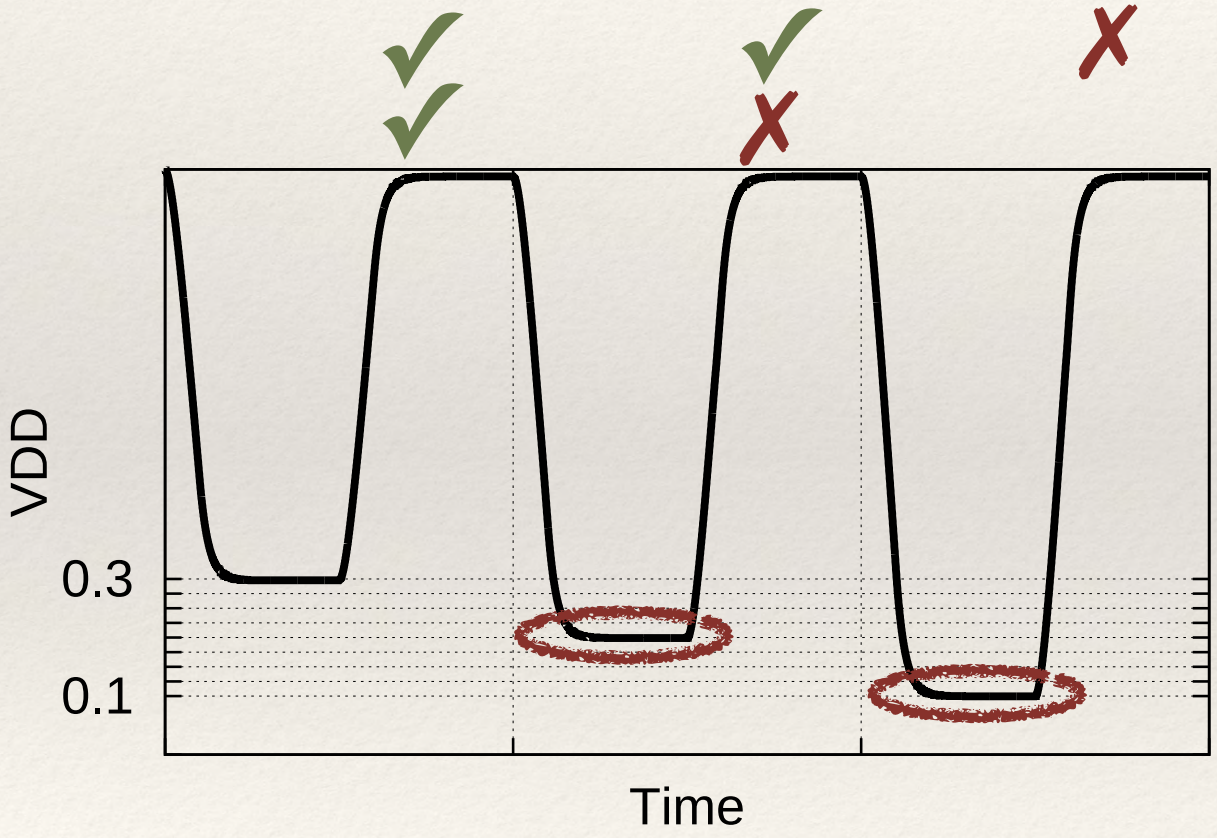


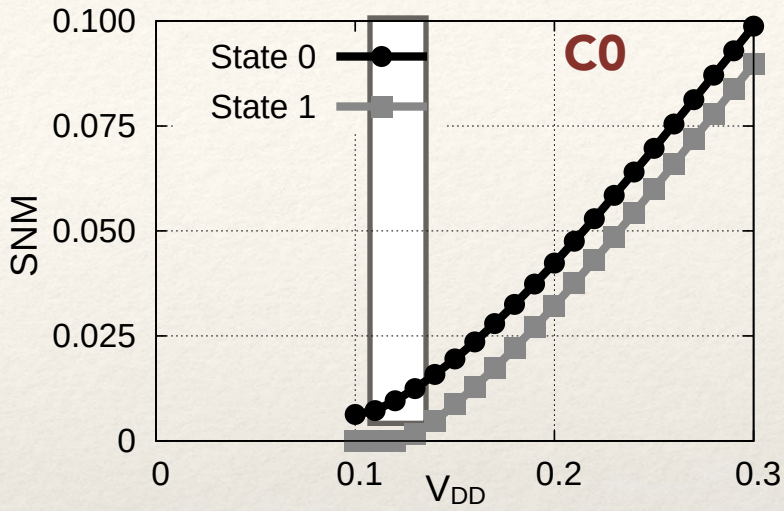


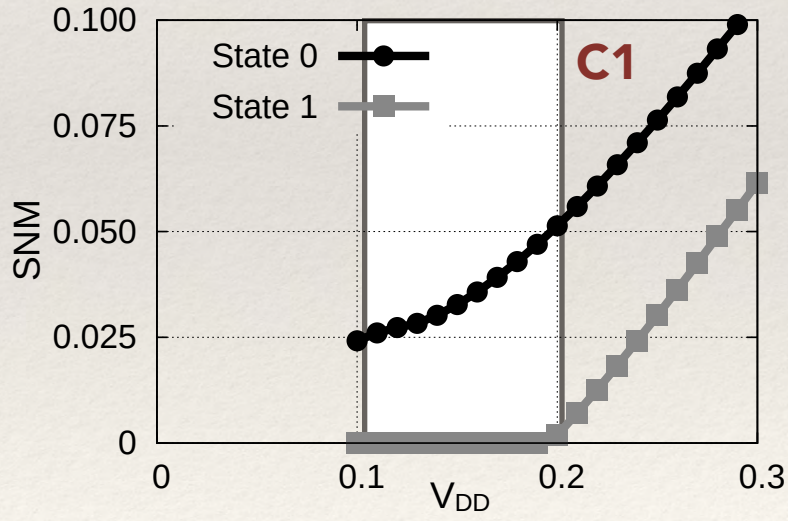


Data Retention Voltage

- More informative than power-up state
- Support from non-volatile storage







DRV PUF as Secret Key

Encode/Decode n-bit key using ≥ 2n-bit SRAM

helper data

Address	DRV
9	300
6	290
5	280
8	270
0	250
2	230
4	210
1	190
3	180
7	170

arbitrary key

index-based syndrome coding [Yu et al. D&TC'10] [Hiller et al. HOST'12]

DRV PUF as Secret Key

Encode/Decode n-bit key using ≥ 2n-bit SRAM

bit i = 1 if first address
in pair i has higher DRV

helper data ← encode ← arbitrary key

1,x,x

Address	DRV
9	300
6	290
5	280
8	270
0	250
2	230
4	210
1	190
3	180
7	170

index-based syndrome coding [Yu et al. D&TC'10] [Hiller et al. HOST'12]

DRV PUF as Secret Key

Encode/Decode n-bit key using ≥ 2n-bit SRAM

bit i = 1 if first address
in pair i has higher DRV

helper data

⟨9,1⟩,⟨x,x⟩,⟨x,x⟩

encode

arbitrary key

1,x,x

Address	DRV
9	300
6	290
5	280
8	270
0	250
2	230
4	210
1	190
3	180
7	170

index-based syndrome coding [Yu et al. D&TC'10] [Hiller et al. HOST'12]

DRV PUF as Secret Key

Encode/Decode n-bit key using ≥ 2n-bit SRAM

bit i = 1 if first address
in pair i has higher DRV

helper data ⟵ encode ⟵ arbitrary key

⟨1,9⟩,⟨x,x⟩,⟨x,x⟩

Address	DRV
9	300
6	290
5	280
8	270
0	250
2	230
4	210
1	190
3	180
7	170

0,x,x

index-based syndrome coding [Yu et al. D&TC'10] [Hiller et al. HOST'12]

DRV PUF as Secret Key

Encode/Decode n-bit key using ≥ 2n-bit SRAM

bit i = 1 if first address
in pair i has higher DRV

helper data

arbitrary key

⟨1,9⟩,⟨x,x⟩,⟨x,x⟩

Address	DRV
9	300
6	290
5	280
8	270
0	250
2	230
4	210
1	190
3	180
7	170

index-based syndrome coding [Yu et al. D&TC'10] [Hiller et al. HOST'12]

DRV PUF as Secret Key

Encode/Decode n-bit key using ≥ 2n-bit SRAM

bit i = 1 if first address
in pair i has higher DRV

helper data → decode → arbitrary key

⟨1,9⟩,⟨x,x⟩,⟨x,x⟩

Address	DRV
9	287
6	305
5	263
8	268
0	251
2	232
4	213
1	203
3	181
7	182

0,x,x

index-based syndrome coding [Yu et al. D&TC'10] [Hiller et al. HOST'12]

DRV PUF as Secret Key

Encode/Decode n-bit key using ≥ 2n-bit SRAM

bit i = 1 if first address
in pair i has higher DRV

helper data → decode → arbitrary key

⟨1,9⟩,⟨x,x⟩,⟨x,x⟩ → 0,x,x

Address	DRV
9	287
6	305
5	263
8	268
0	251
7	182

- 100% reliable key generation using silicon data
- Cost of DRV characterization in field is a limitation

index-based syndrome coding [Yu et al. D&TC'10] [Hiller et al. HOST'12]

Cryptographic Hardware and Embedded Systems 2014

Bitline PUF:

Building Native Challenge-Response PUF Capability into Any SRAM

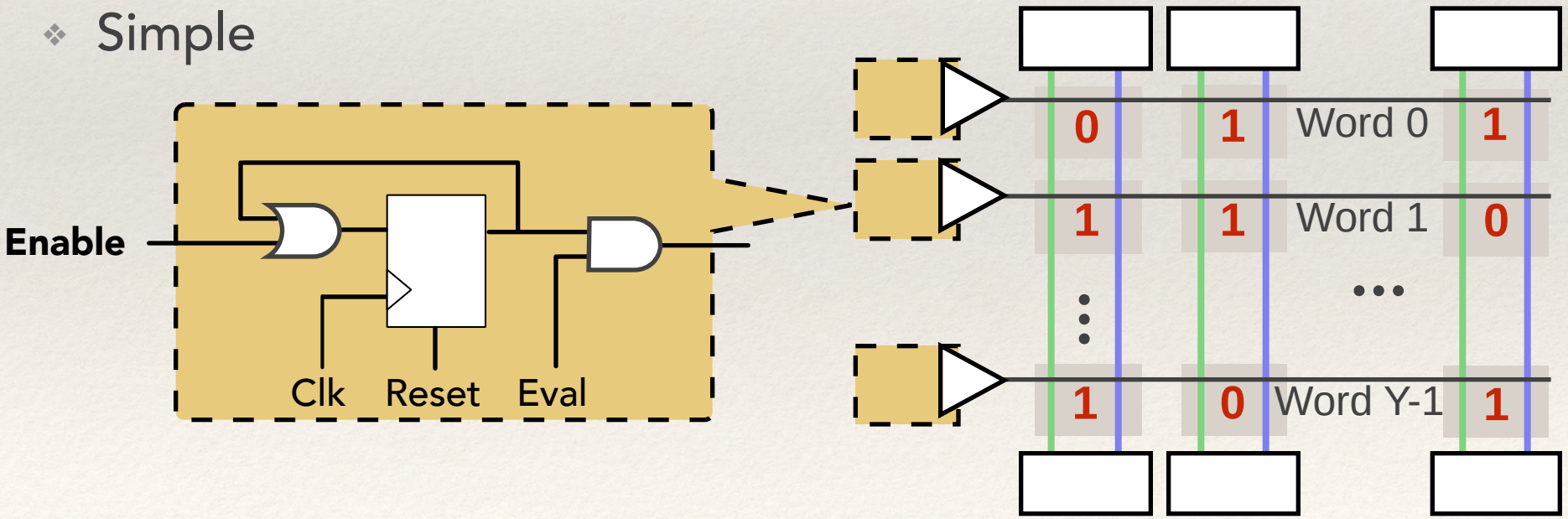
Daniel E. Holcomb

Kevin Fu

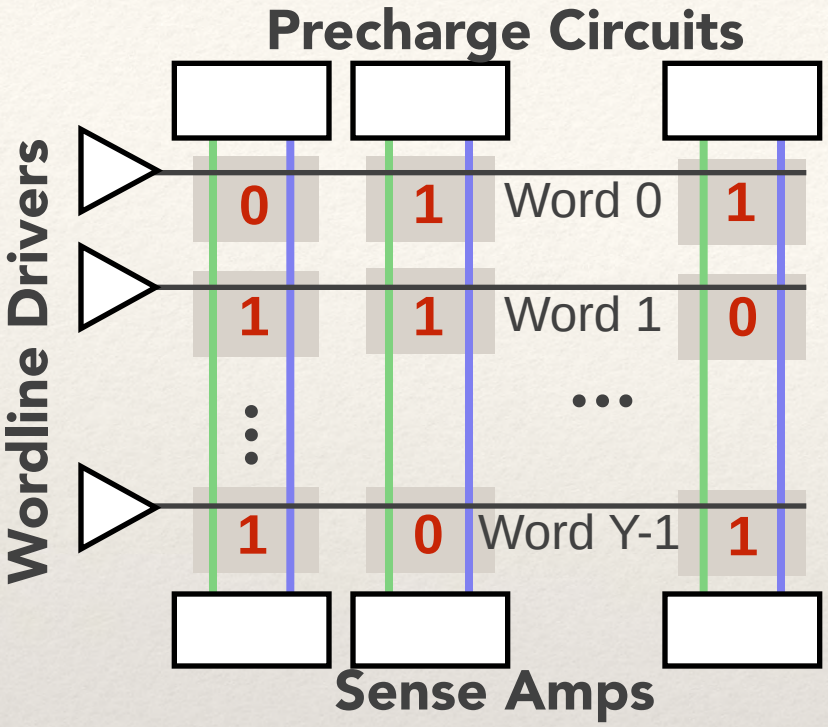
Acknowledgment: This work was supported in part by C-FAR, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA, and by NSF CNS-1331652. Any opinions, findings, conclusions, and recommendations expressed in these materials are those of the authors and do not necessarily reflect the views of the sponsors.

Contributions

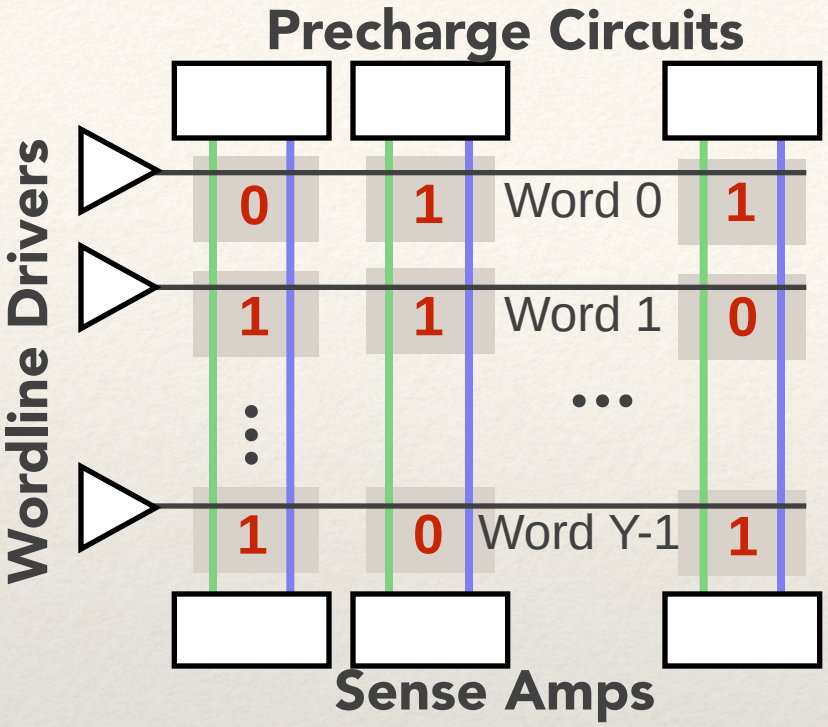
- **Adding a few gates to wordline drivers of SRAM creates a new PUF**
- Bitline PUF
  - Challenge-response operation
  - Low area overhead
  - Simple

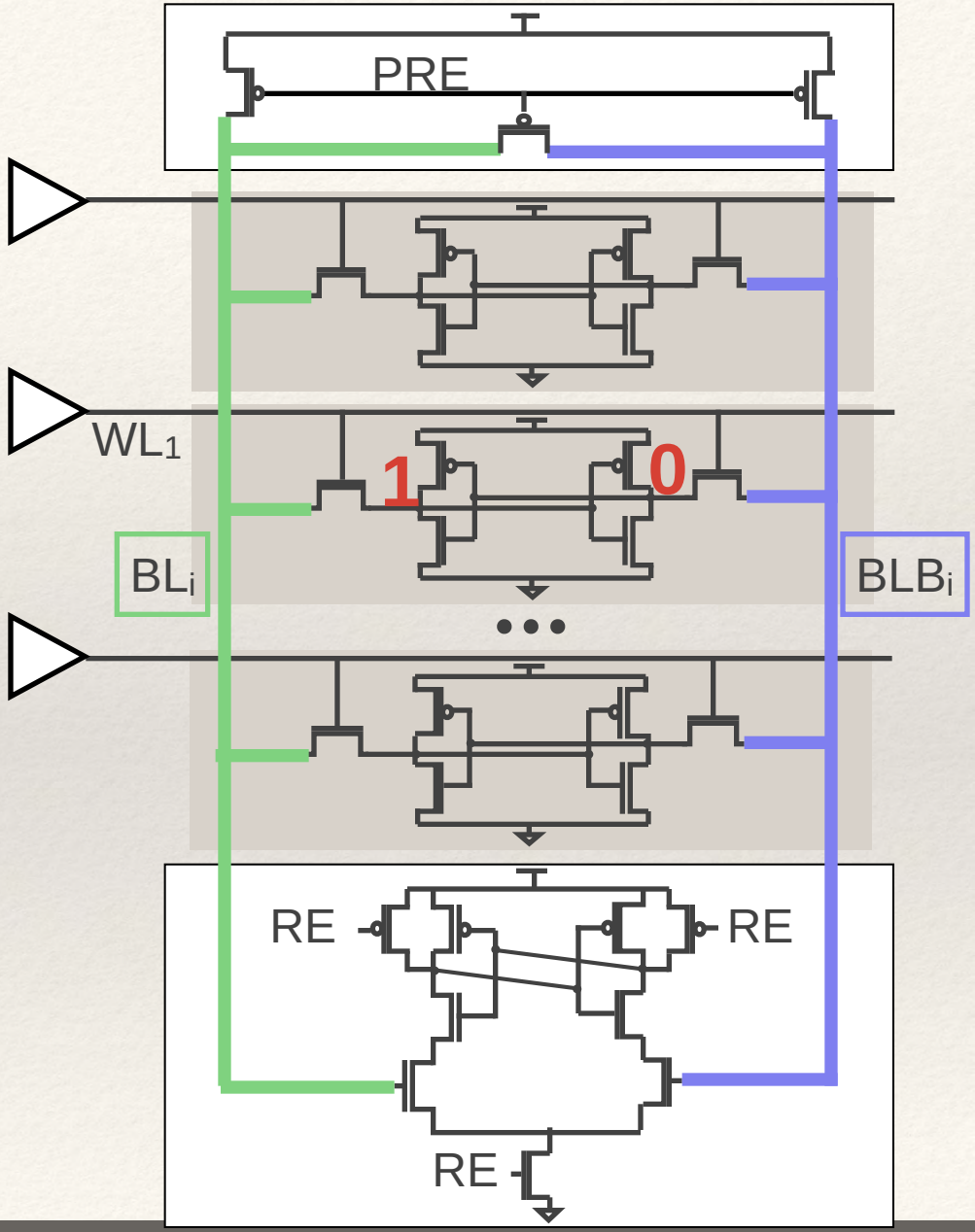


Reading an SRAM Cell

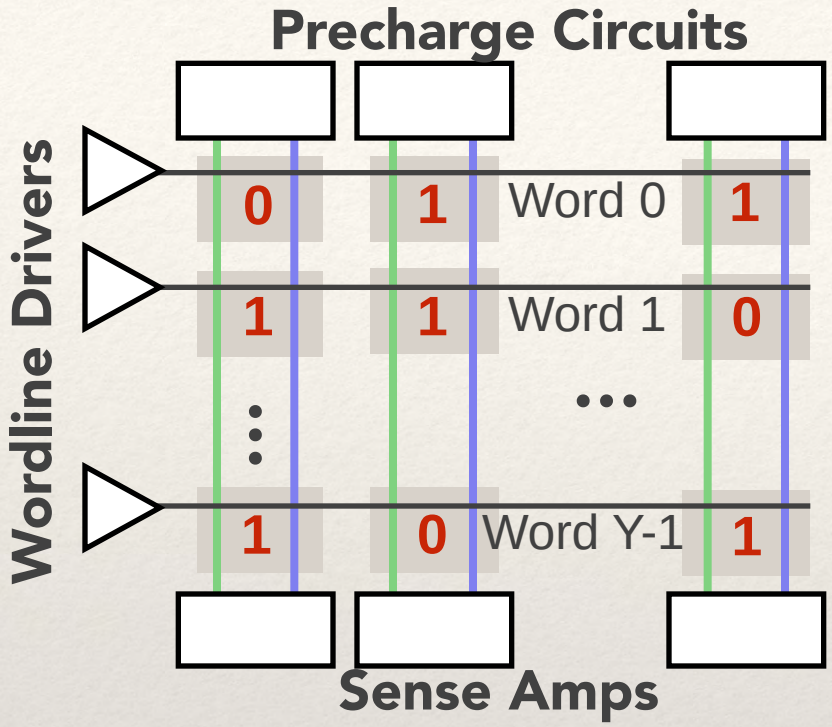


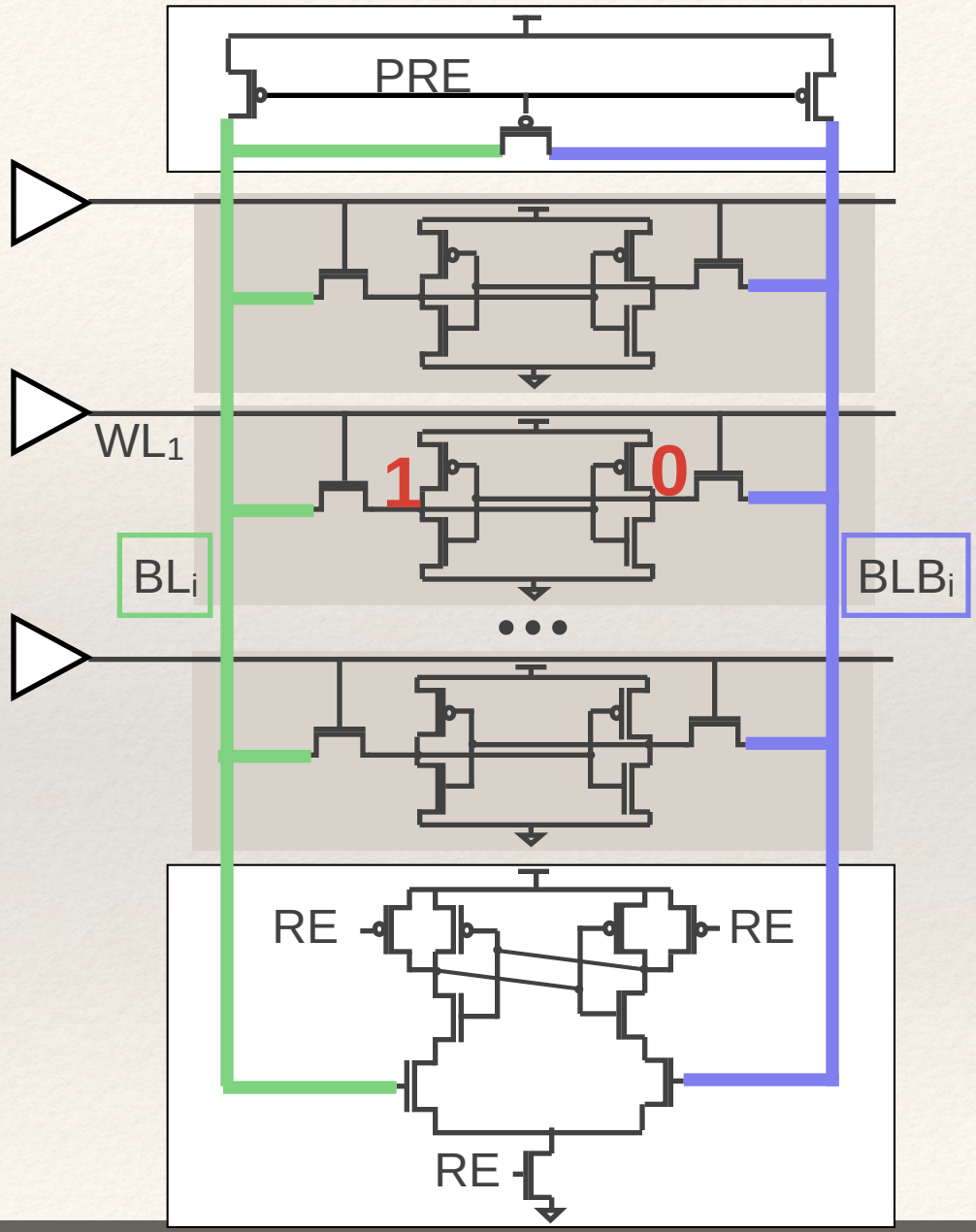
Reading an SRAM Cell

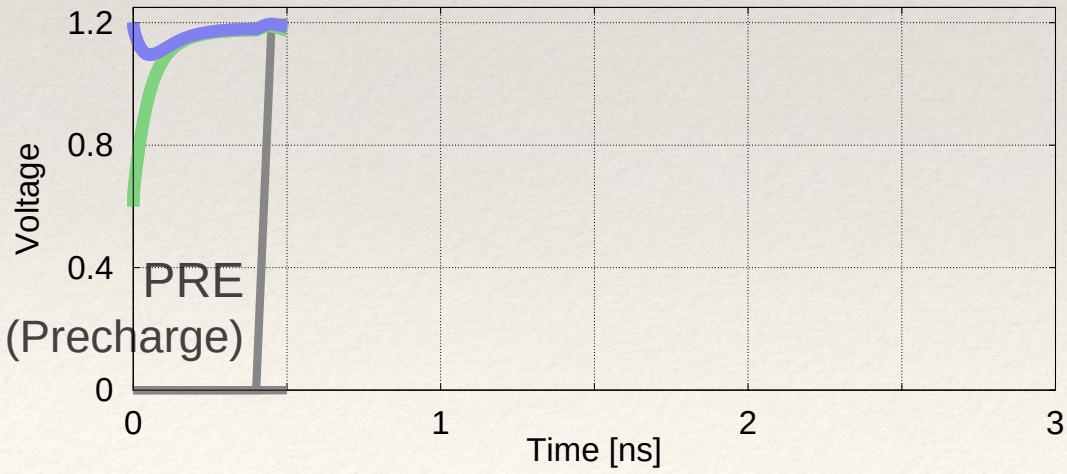




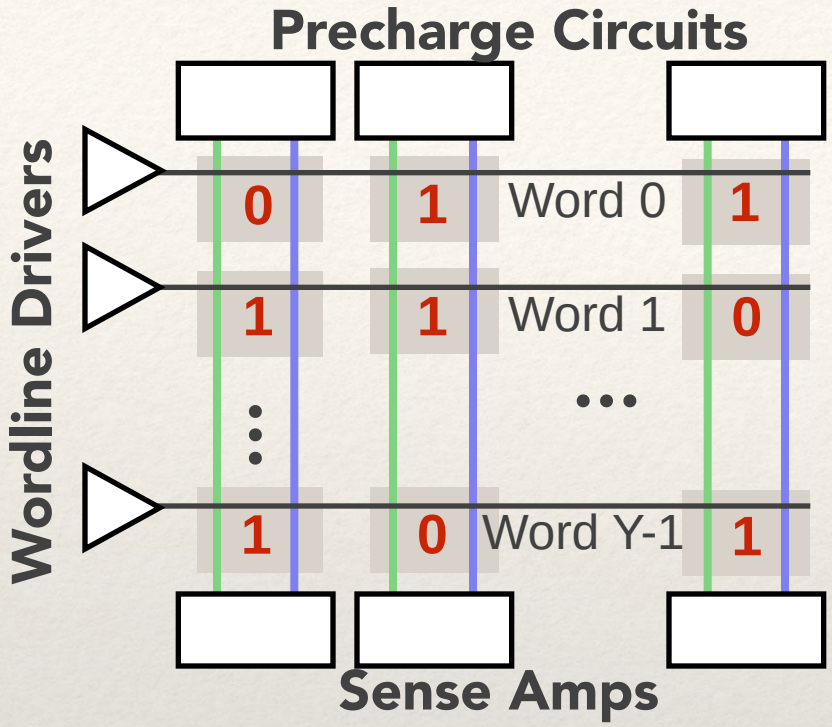
Reading an SRAM Cell

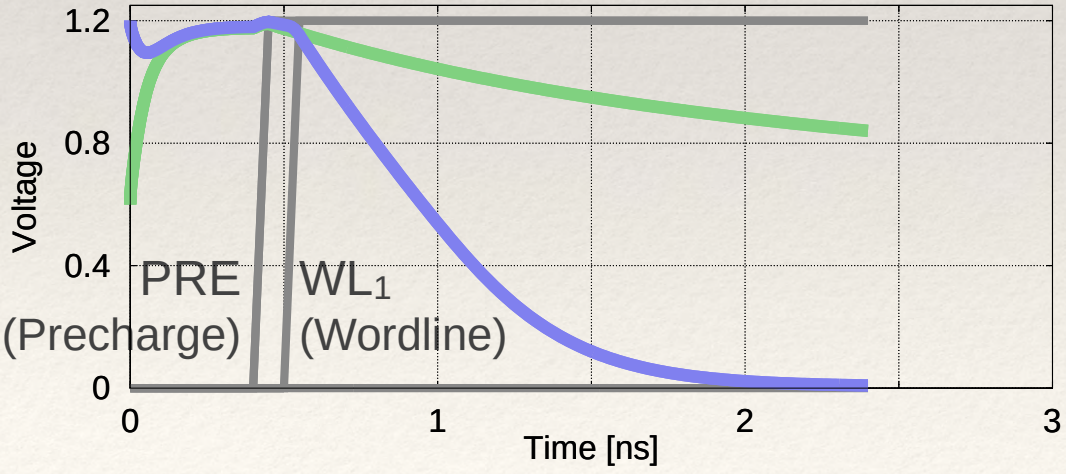


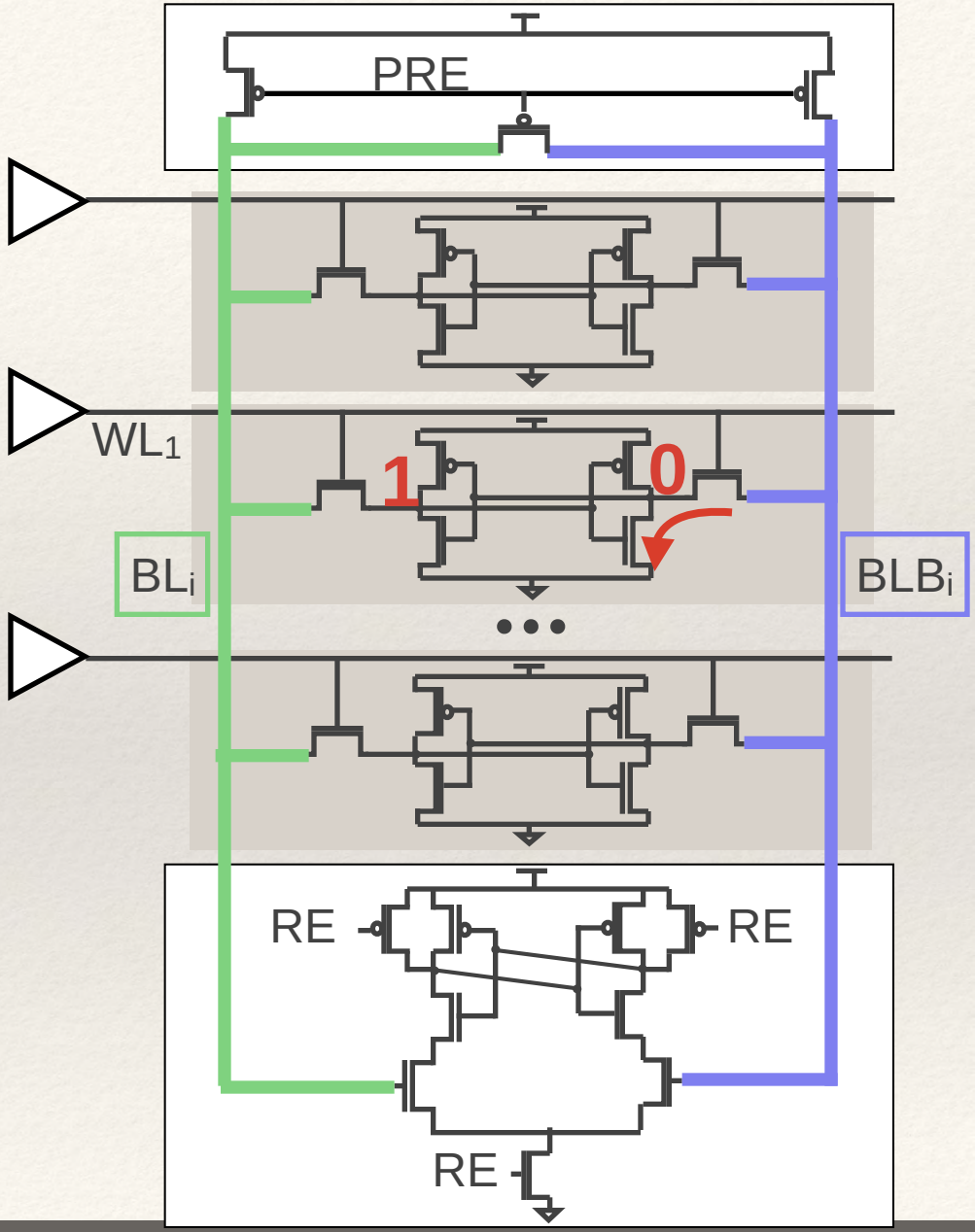




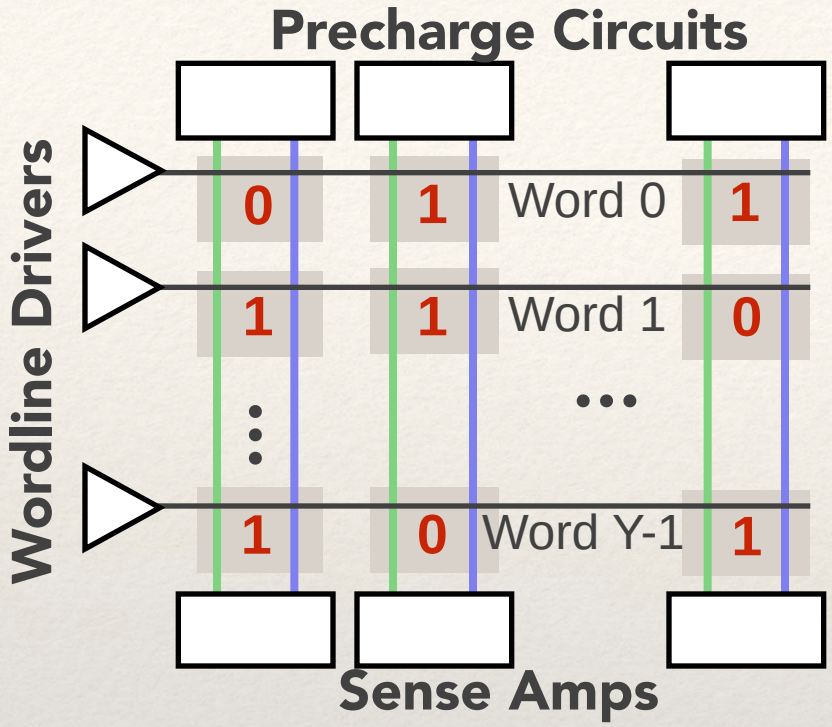
Reading an SRAM Cell

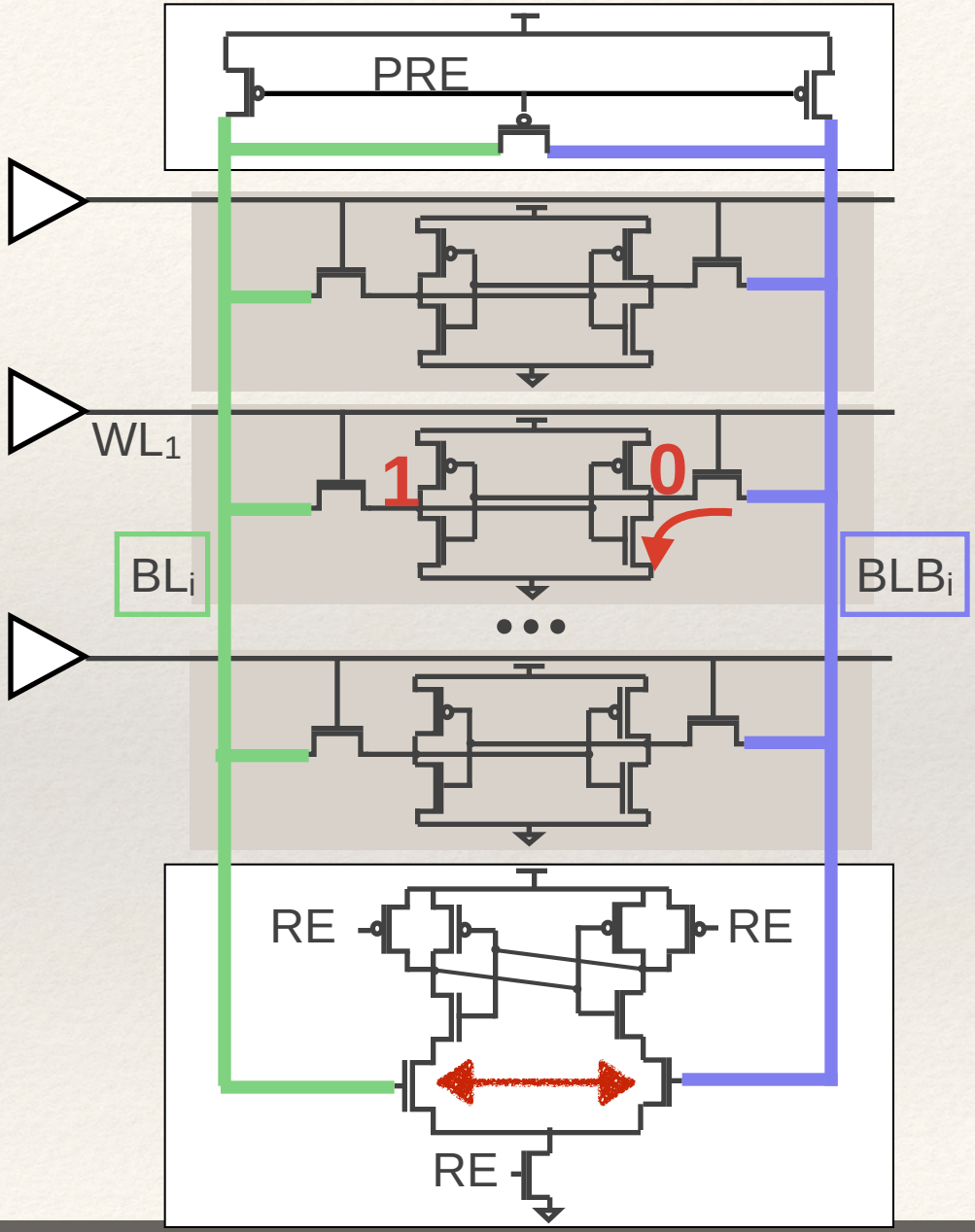


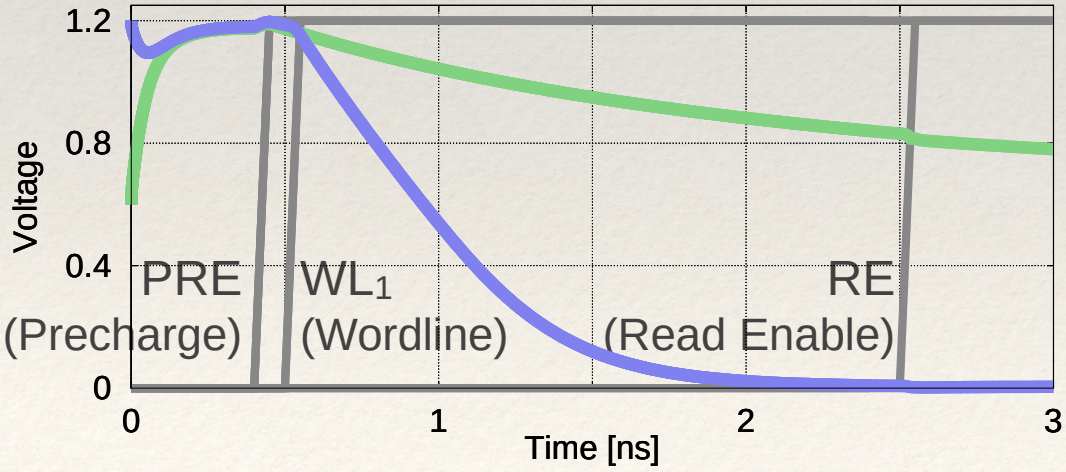




Reading an SRAM Cell

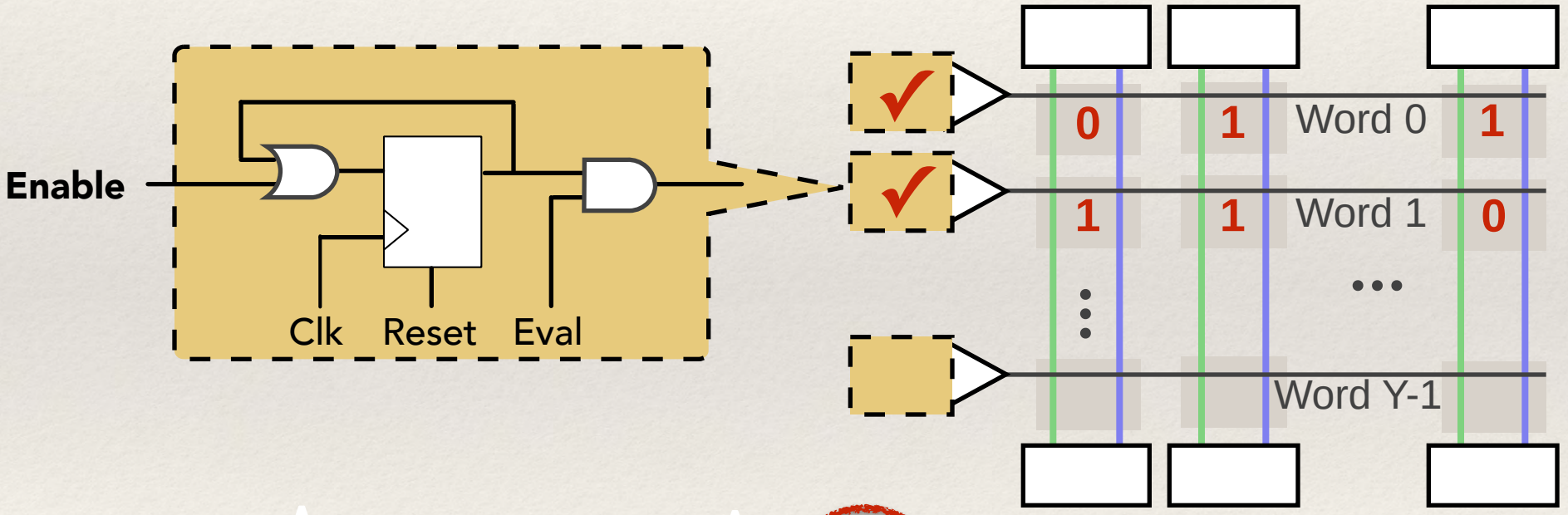


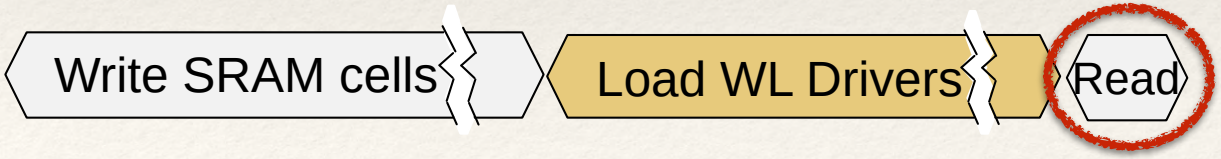




Bitline PUF

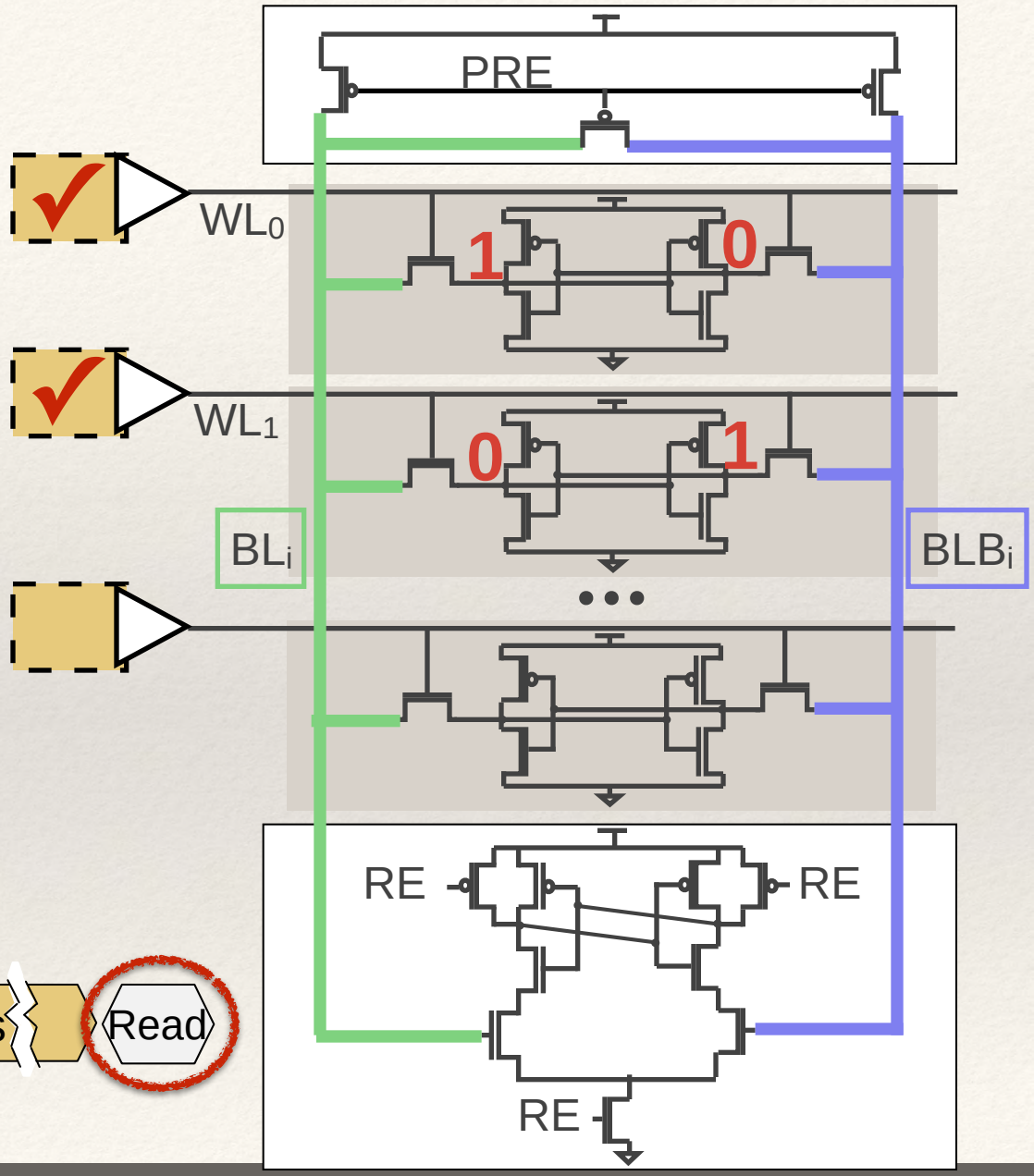
- Accumulate wordline enable signals for **concurrent read**
- Concurrent reading causes contention
- Contention resolves according to variations

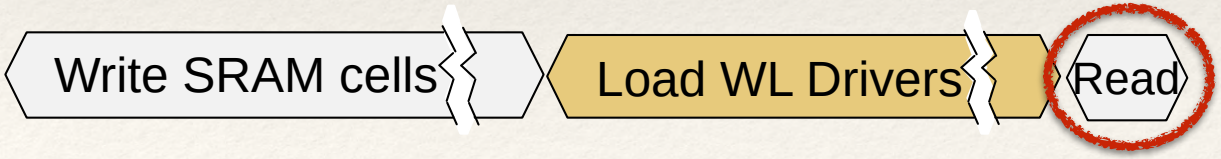




Reading a Bitline PUF

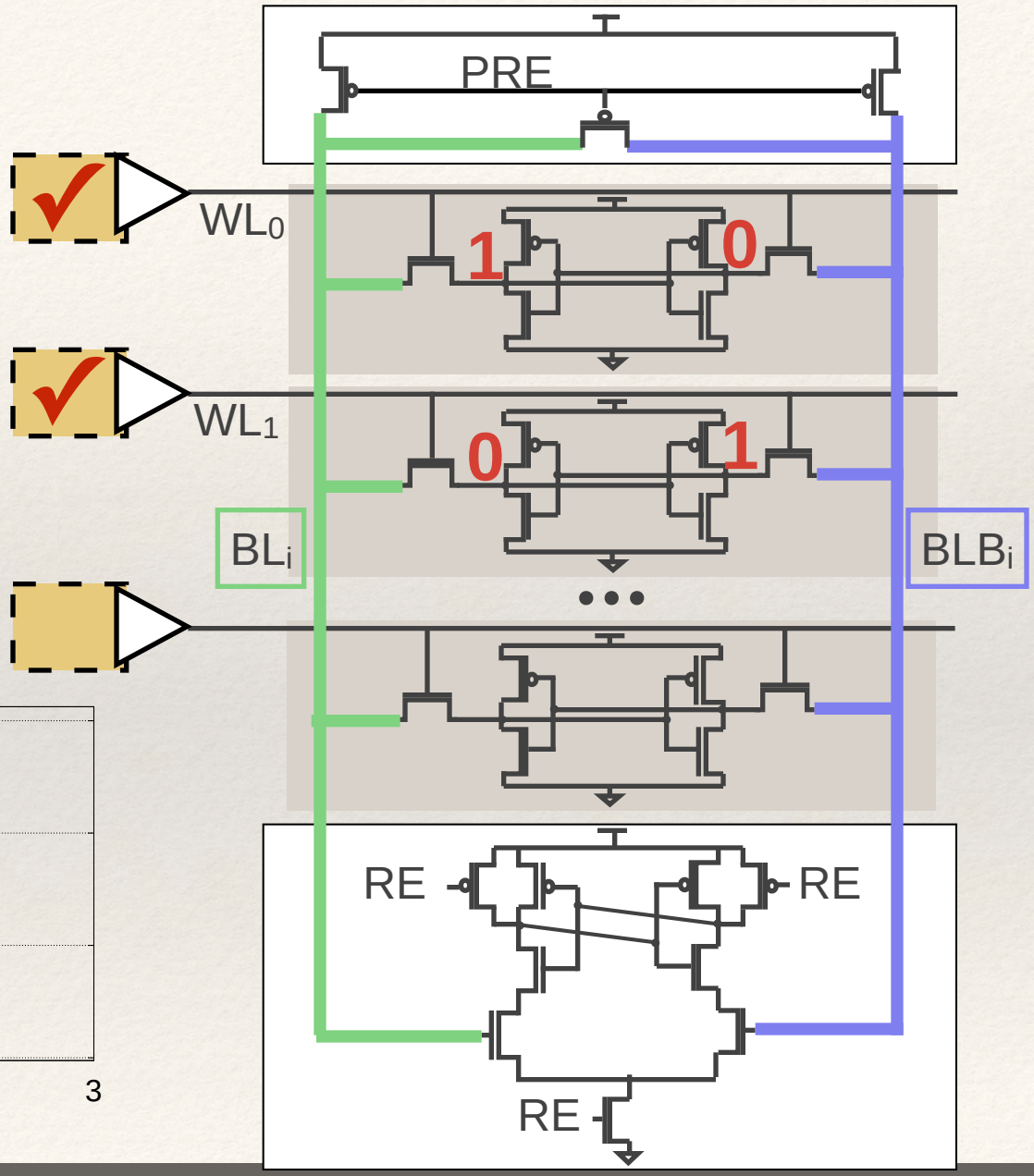
- Read with contention
- Contention resolves according to variation

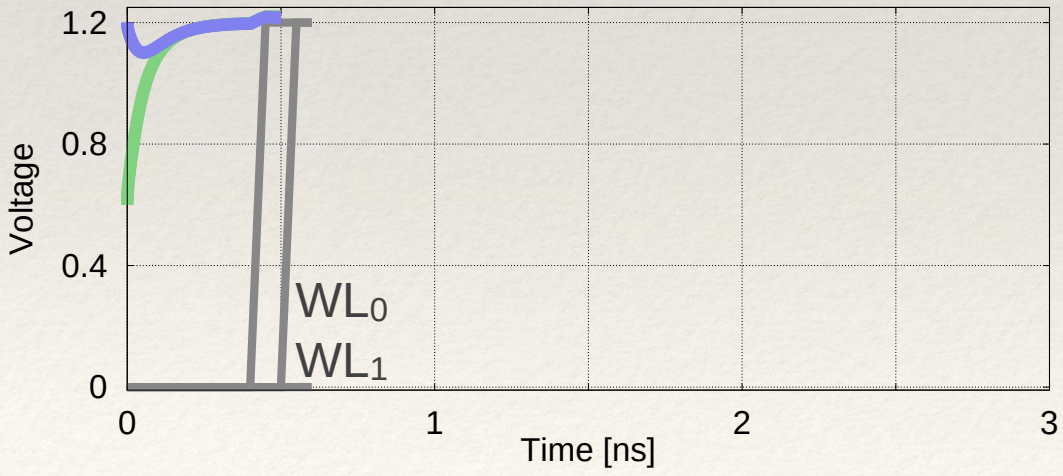




Reading a Bitline PUF

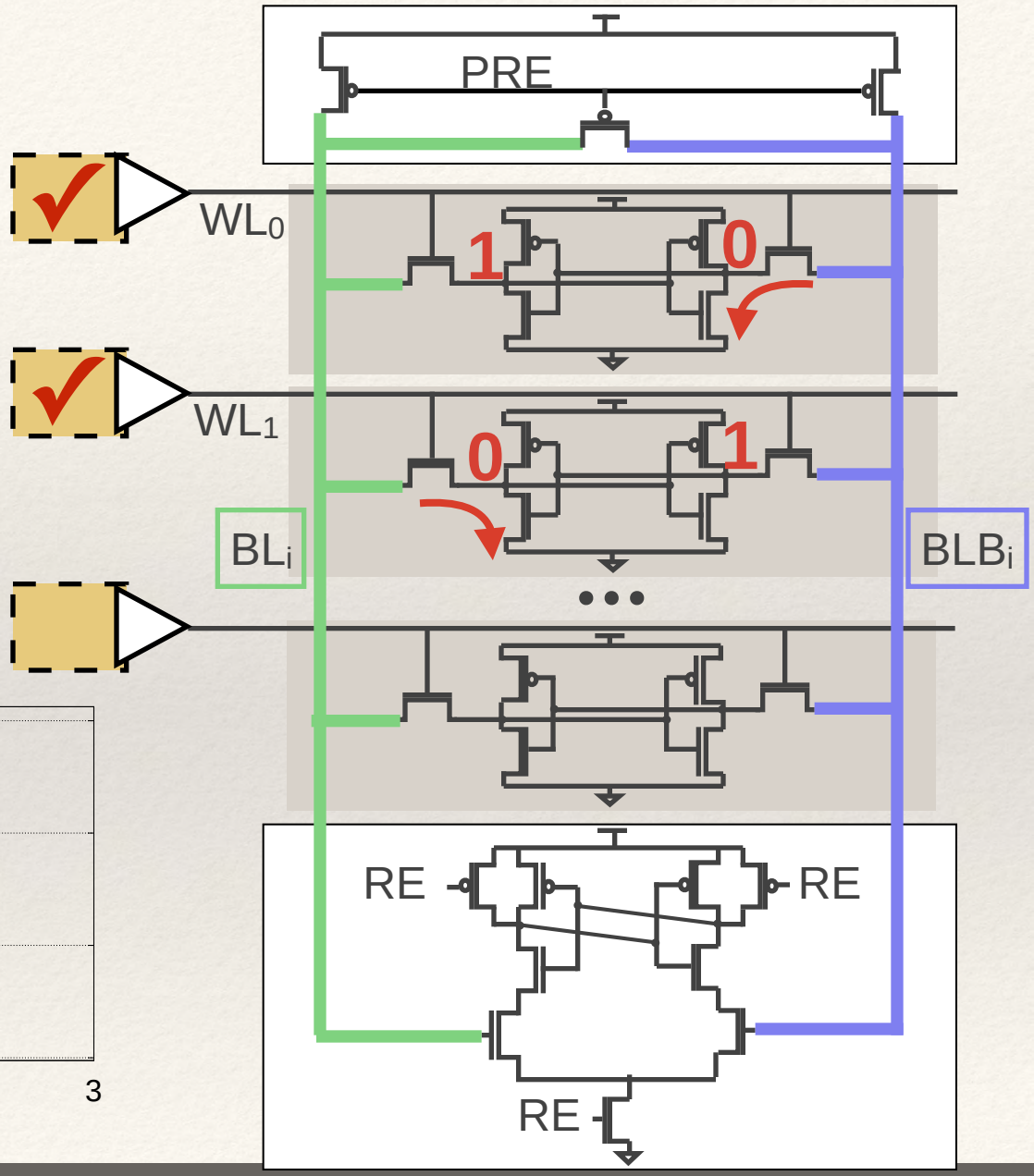
- Read with contention
- Contention resolves according to variation

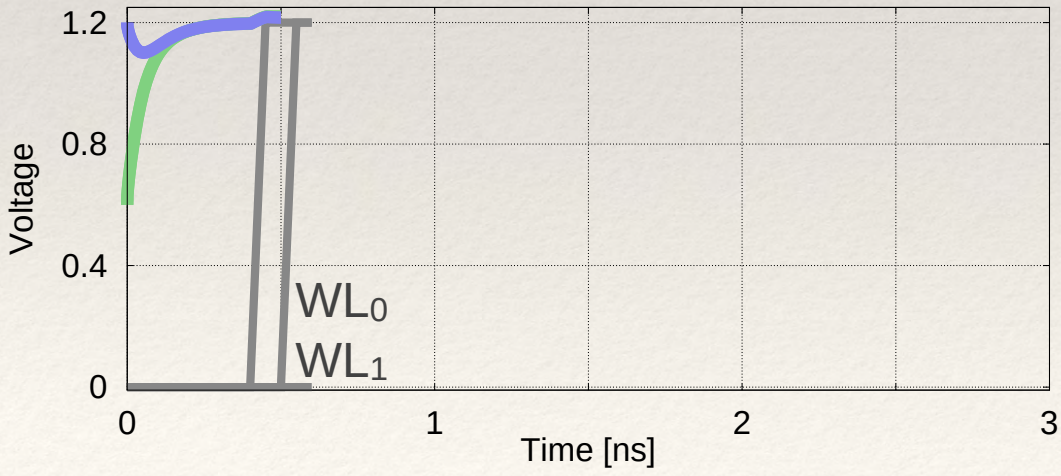




Reading a Bitline PUF

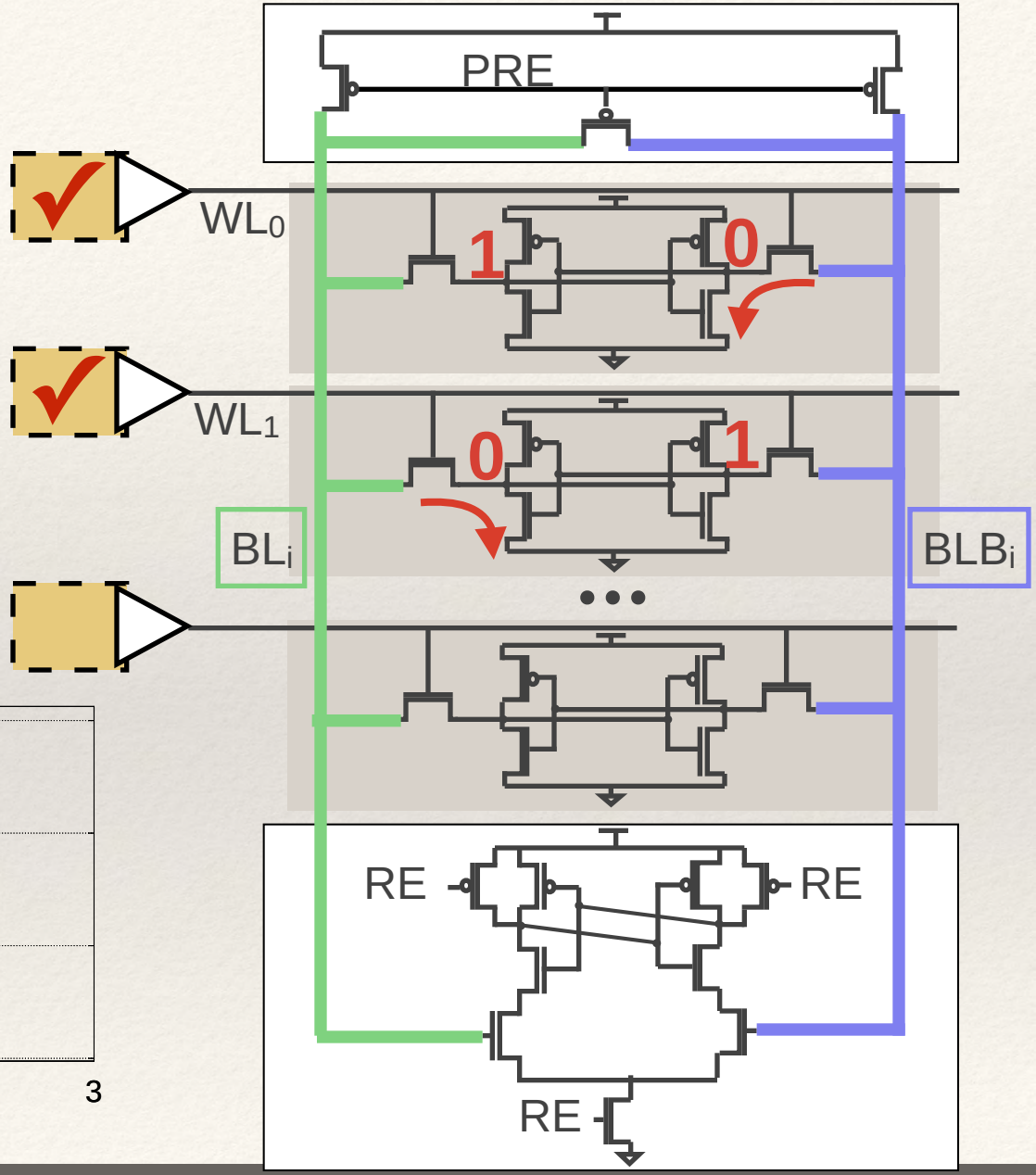
- Read with contention
- Contention resolves according to variation

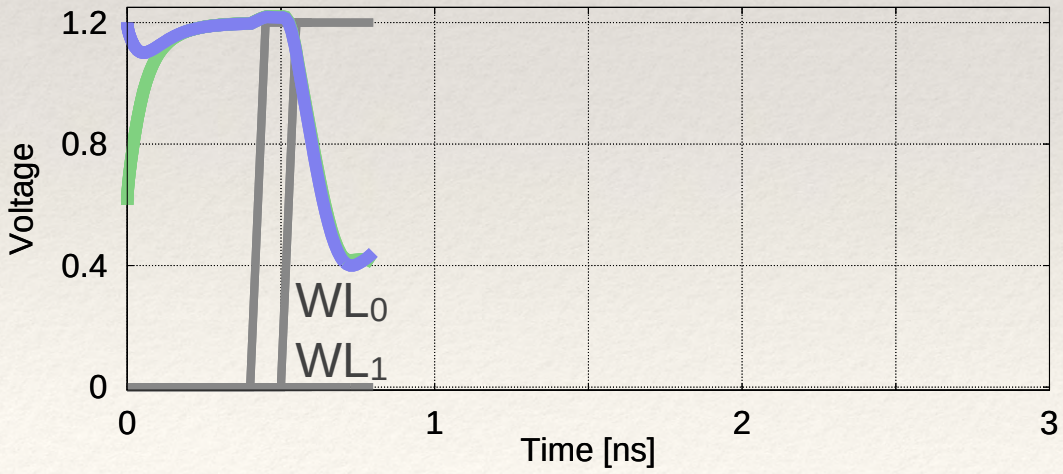




Reading a Bitline PUF

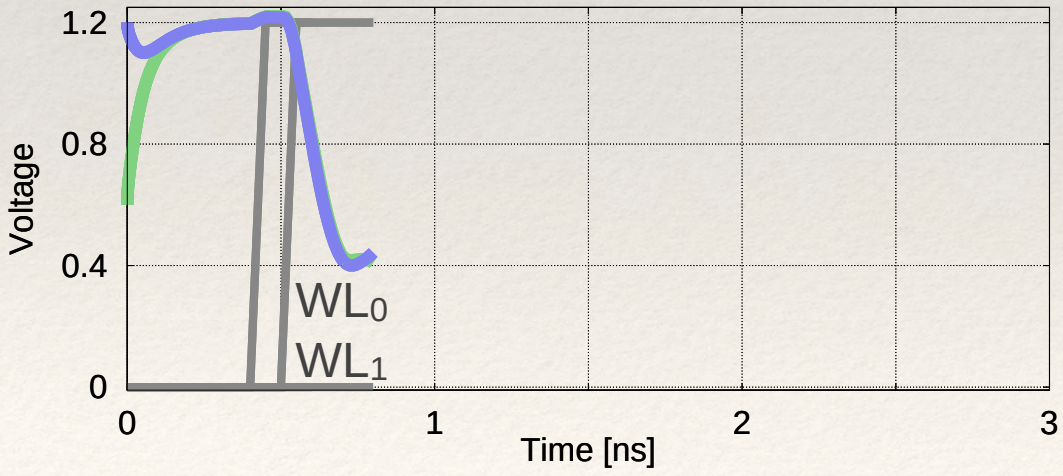
- Read with contention
- Contention resolves according to variation

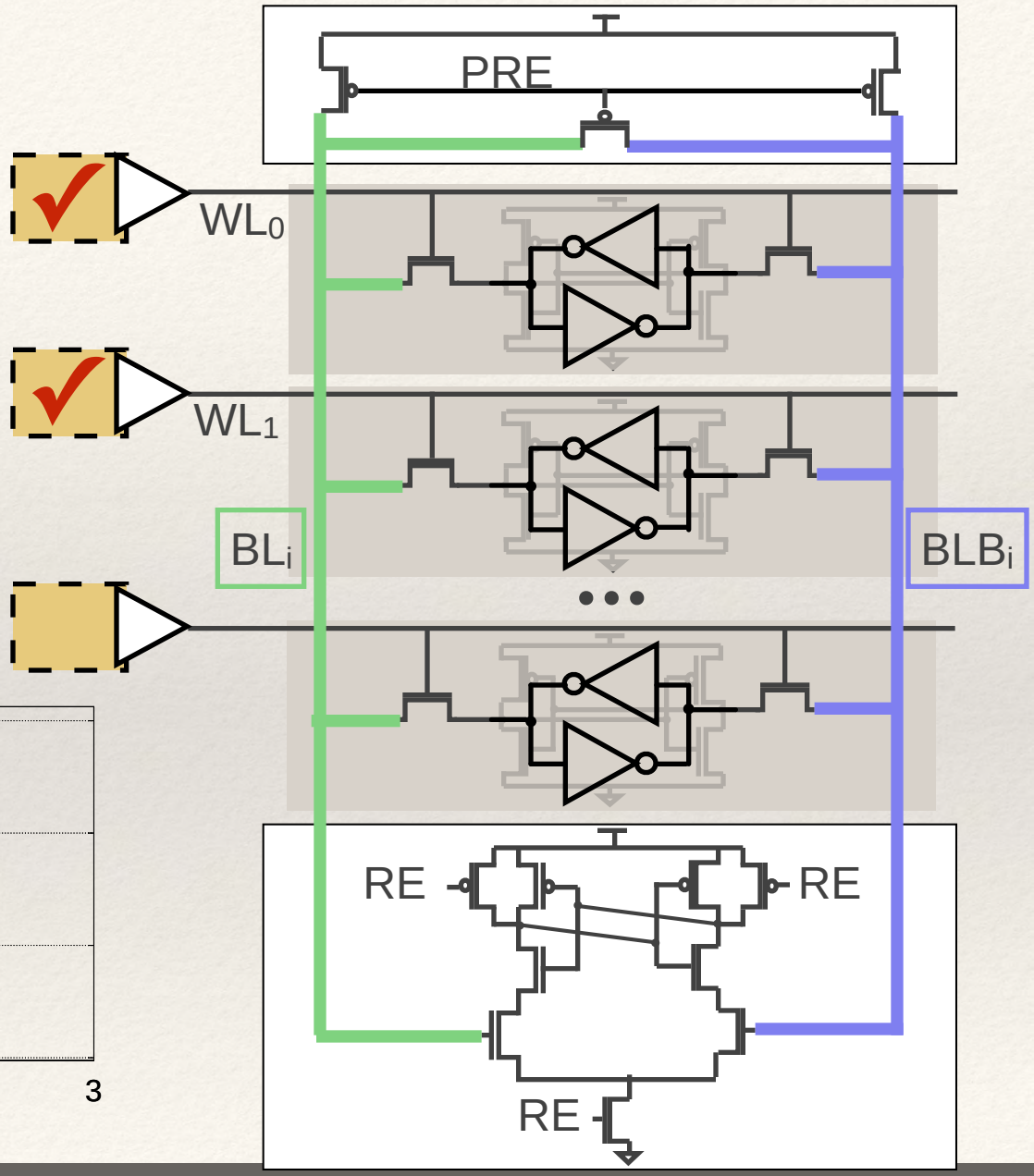




Reading a Bitline PUF

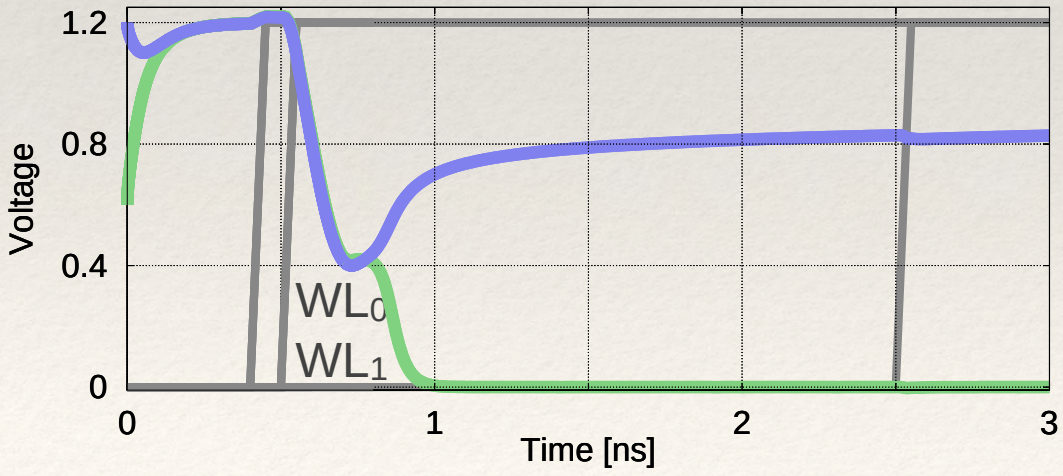
- Read with contention
- Contention resolves according to variation

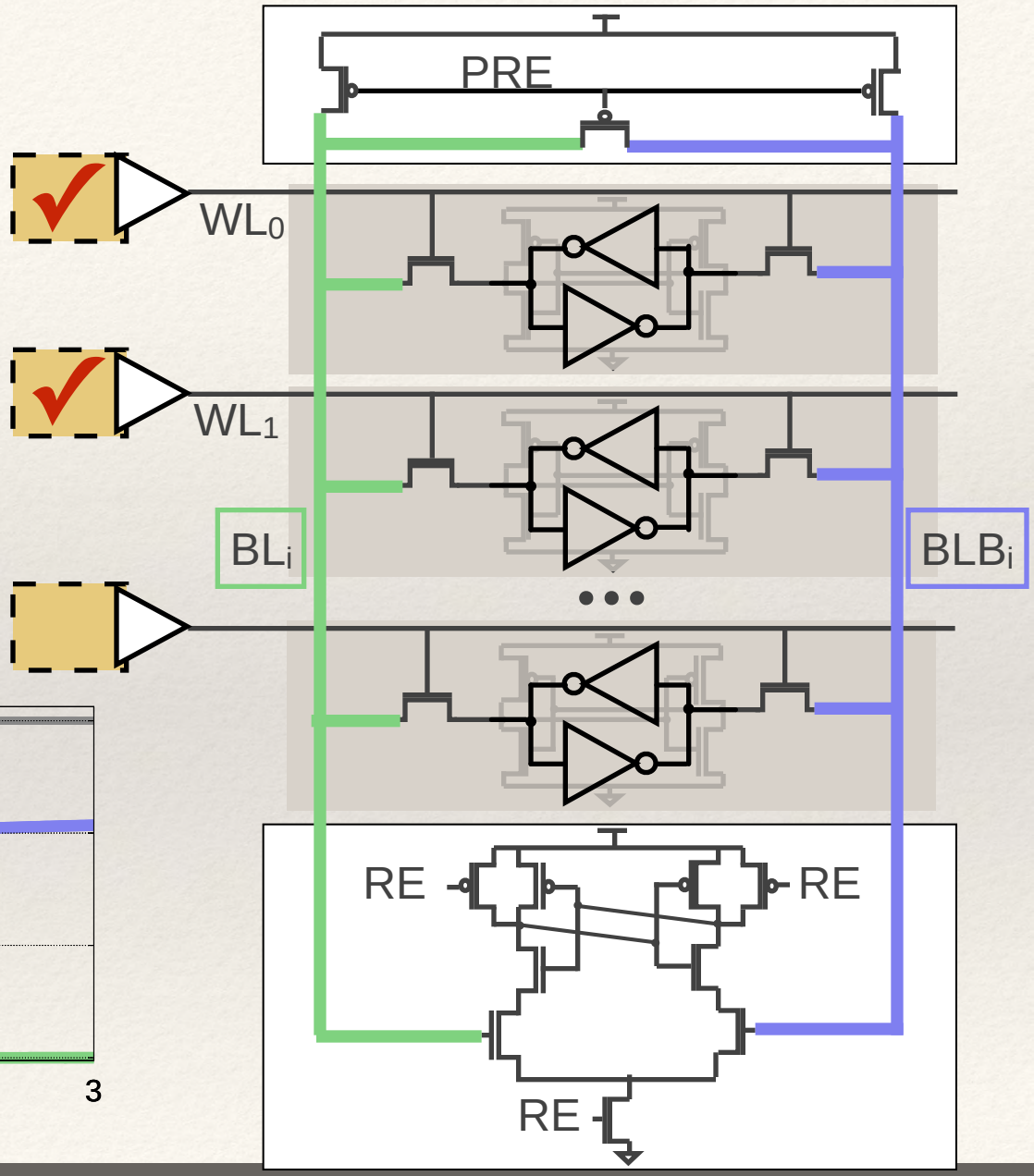




Reading a Bitline PUF

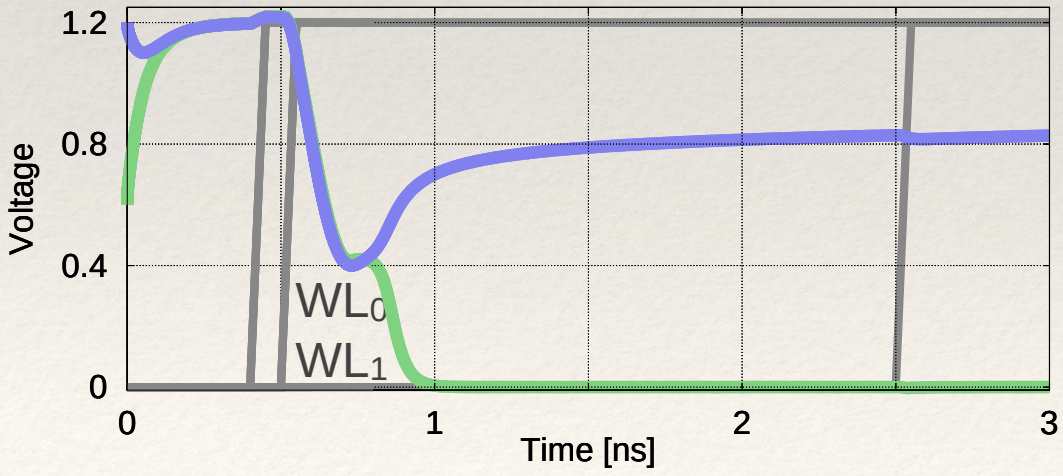
- Read with contention
- Contention resolves according to variation

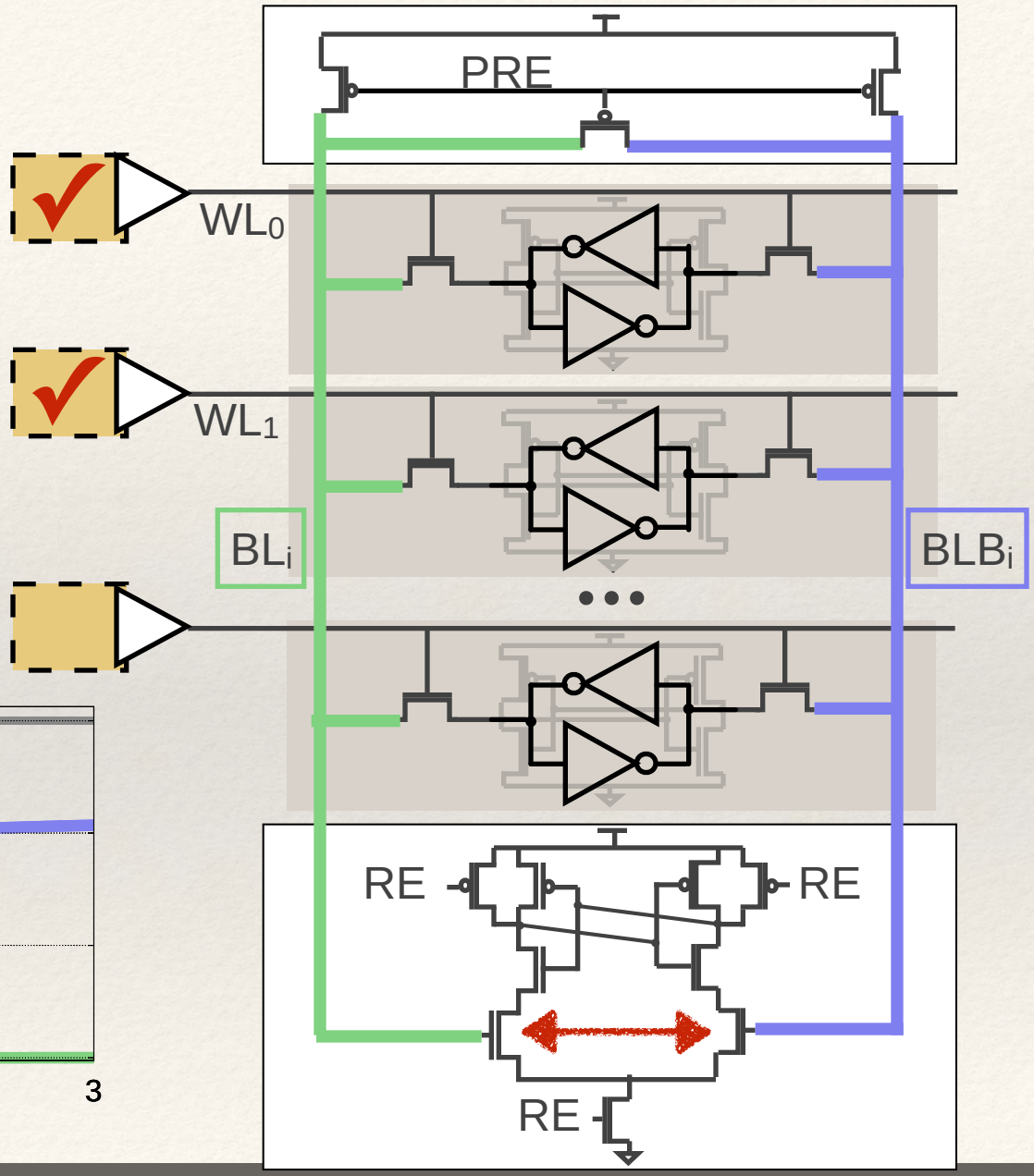




Reading a Bitline PUF

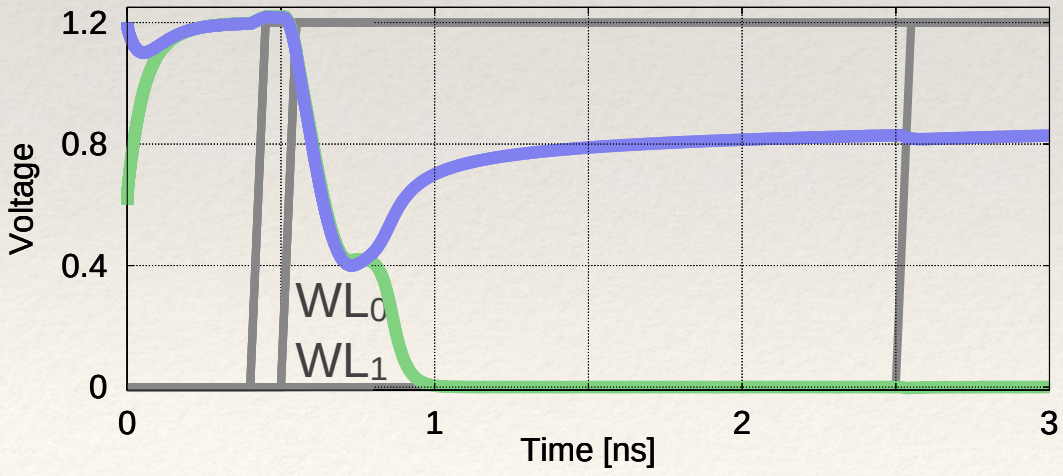
- Read with contention
- Contention resolves according to variation

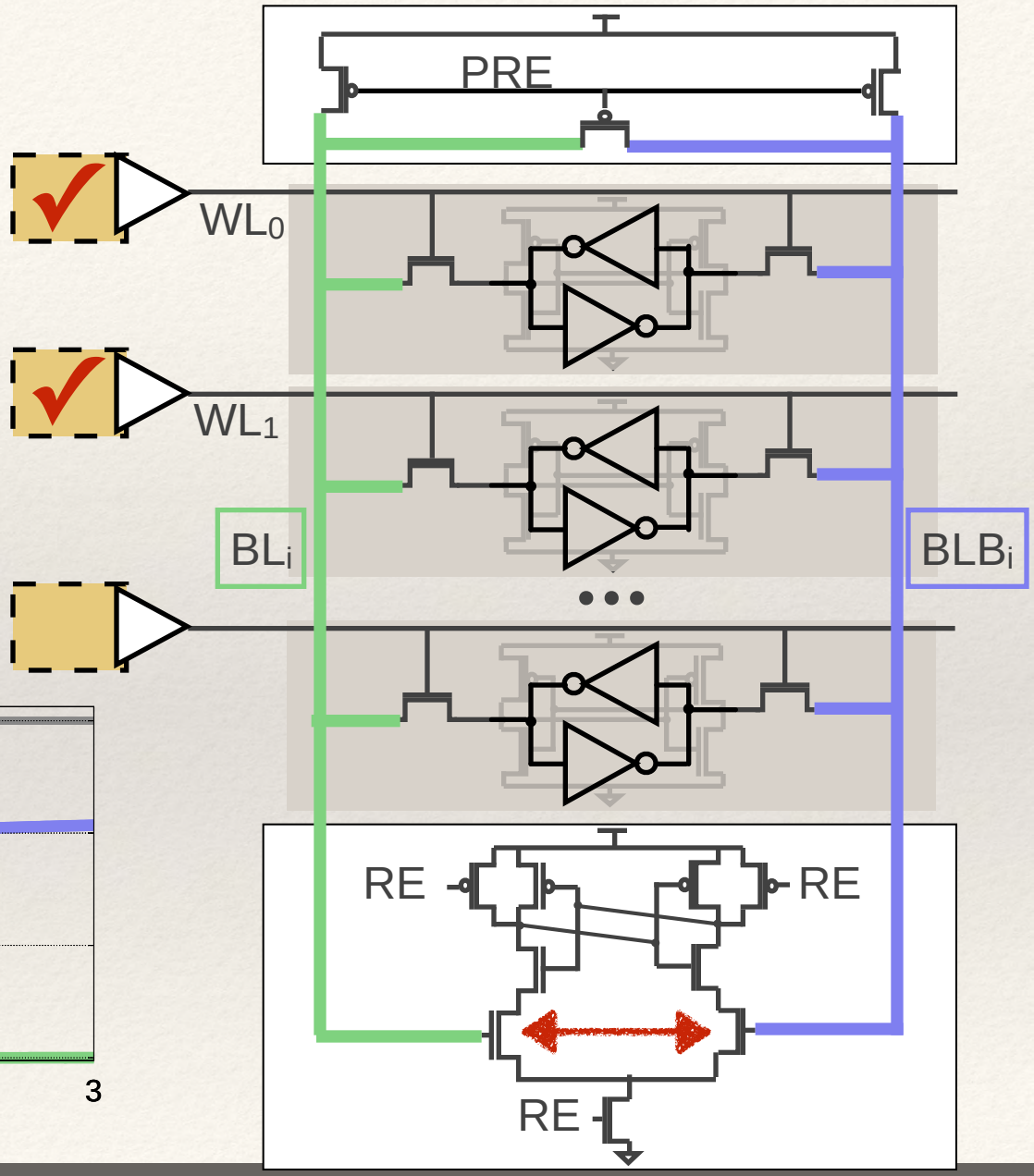




Reading a Bitline PUF

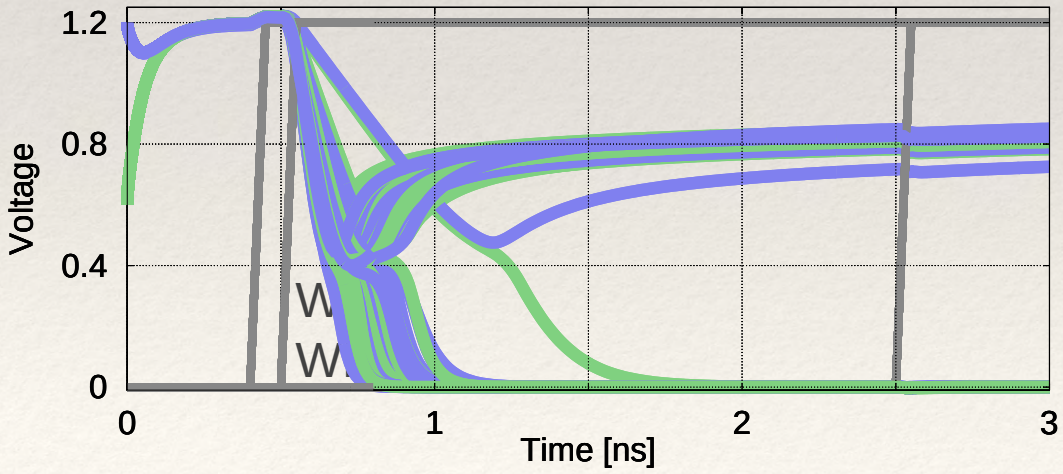
- Read with contention
- Contention resolves according to variation
- Largely consistent over time for given column

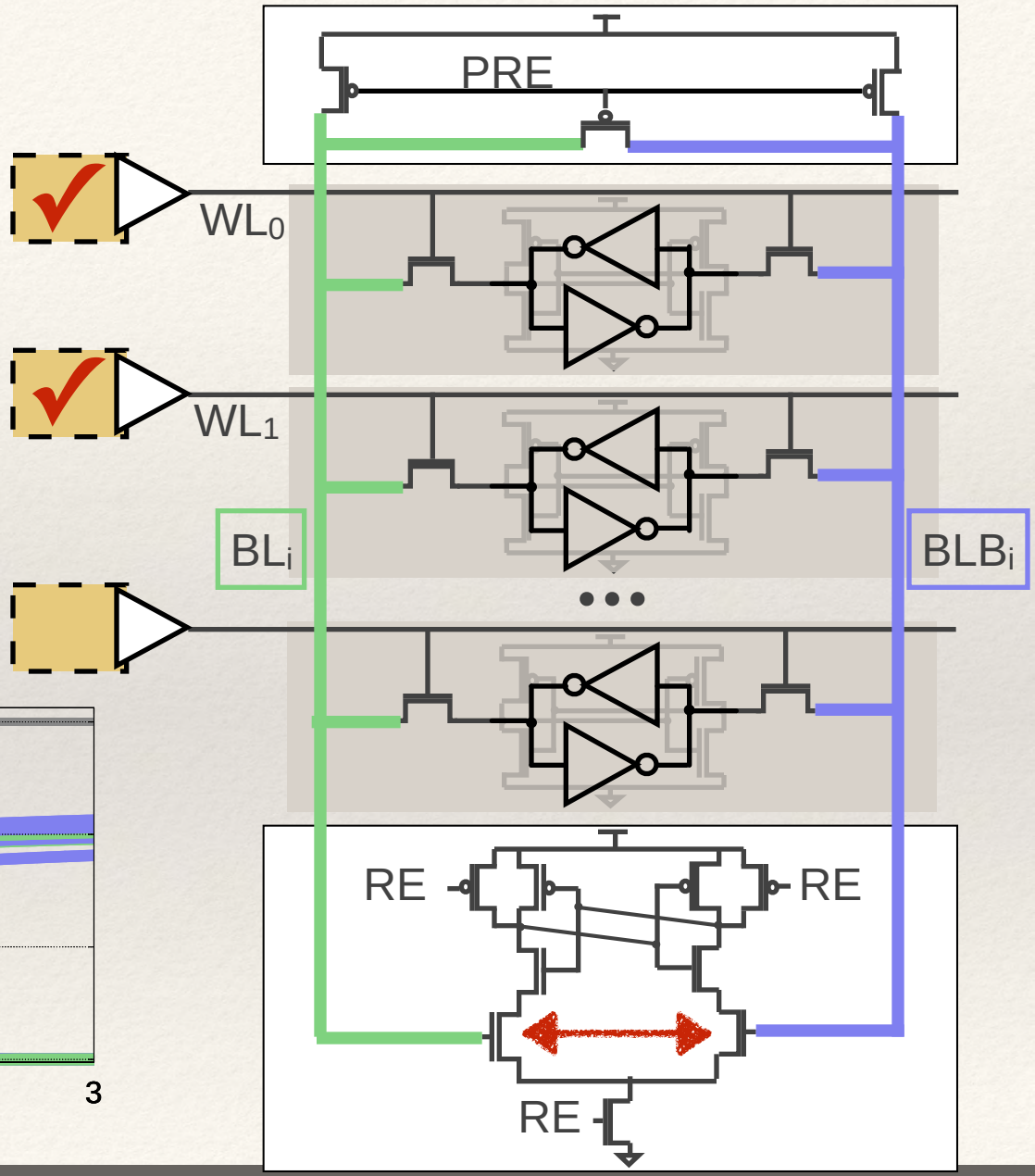




Reading a Bitline PUF

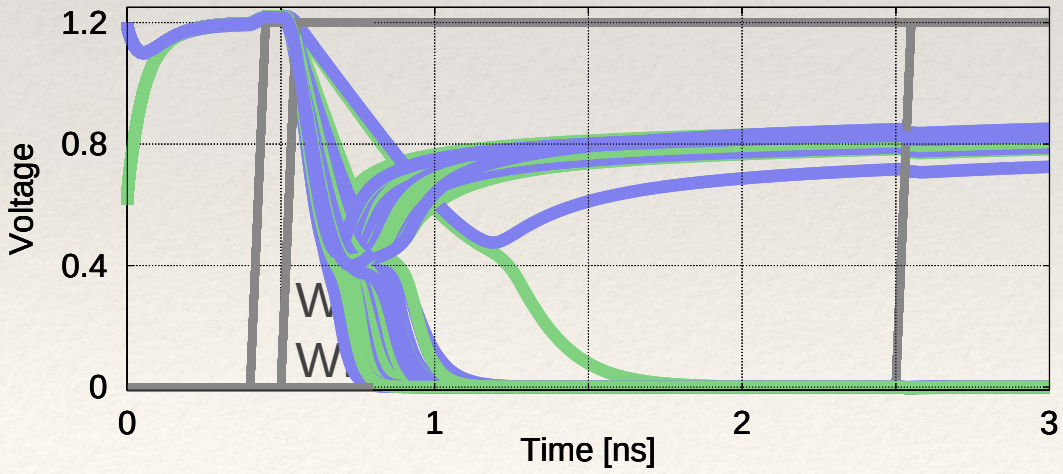
- Read with contention
- Contention resolves according to variation
- Largely consistent over time for given column

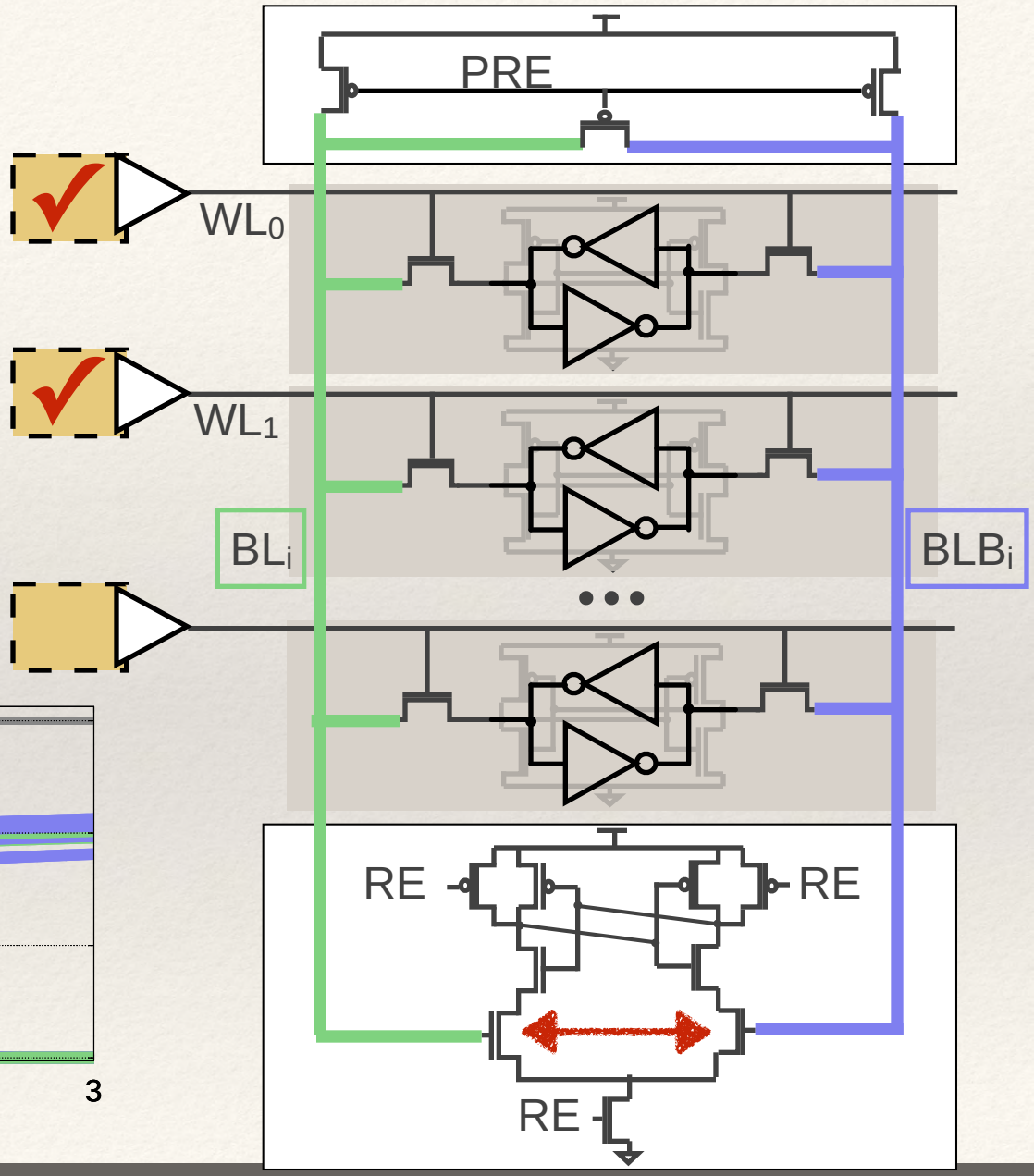




Reading a Bitline PUF

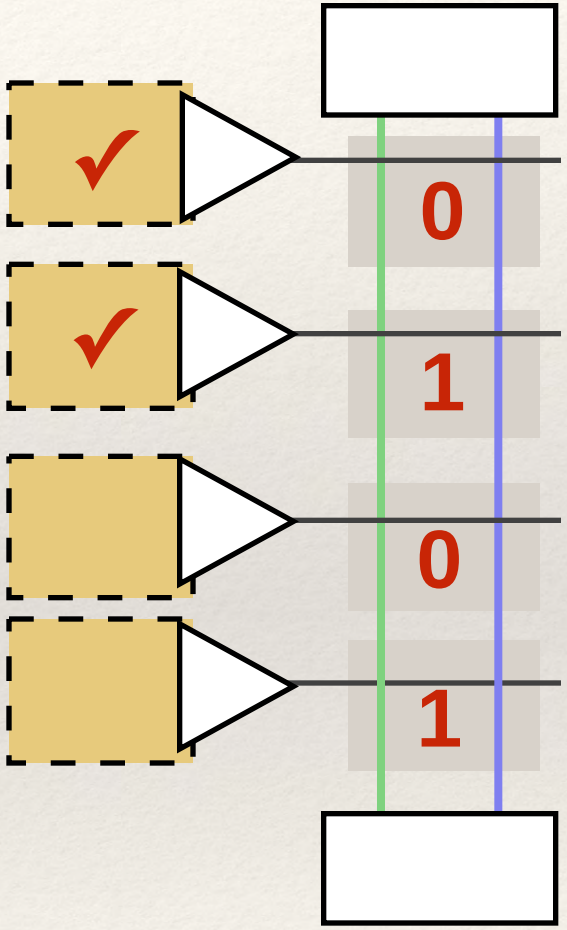
- Read with contention
- Contention resolves according to variation
- Largely consistent over time for given column
- Varies across columns or chips



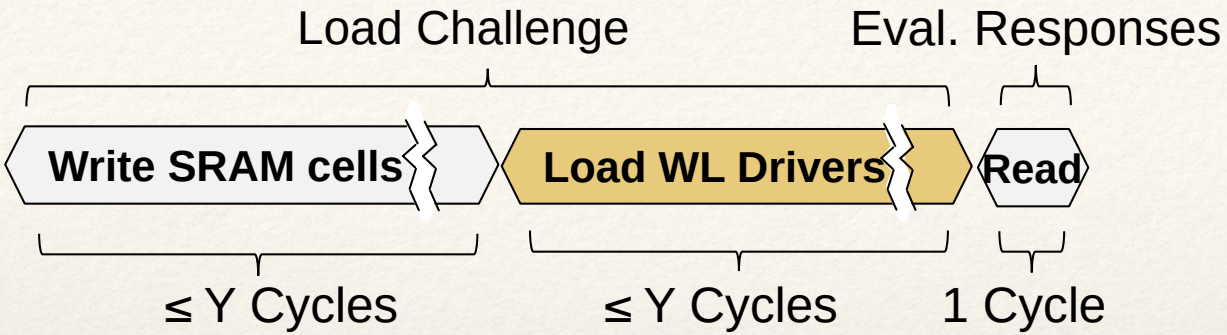


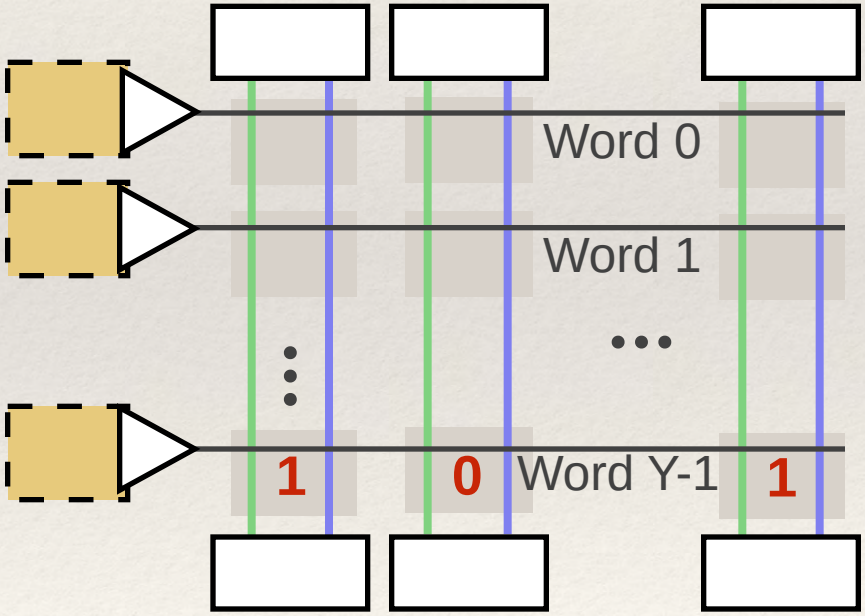
Challenge Response Pairs

- PUF Challenge:
  - 4^{Y} possible challenges (Y = num. rows)
  - For each cell in column:
    1. wordline on, cell value 0
    2. wordline on, cell value 1
    3. wordline off, cell value 0
    4. wordline off, cell value 1
- PUF Response:
  - Value read by sense amp of column(s)

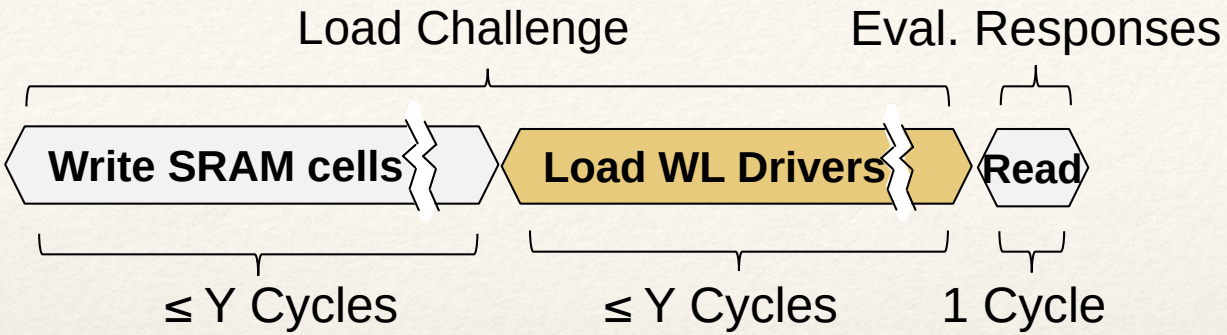


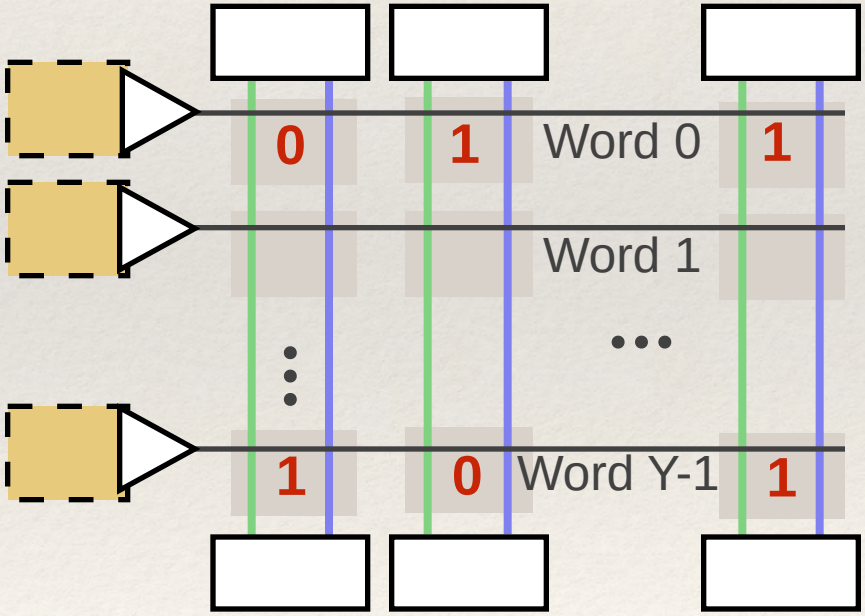
Performance and Overhead



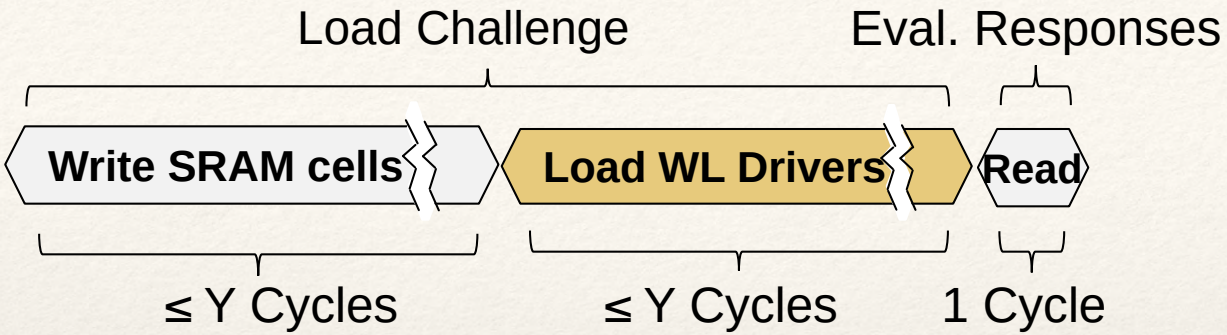


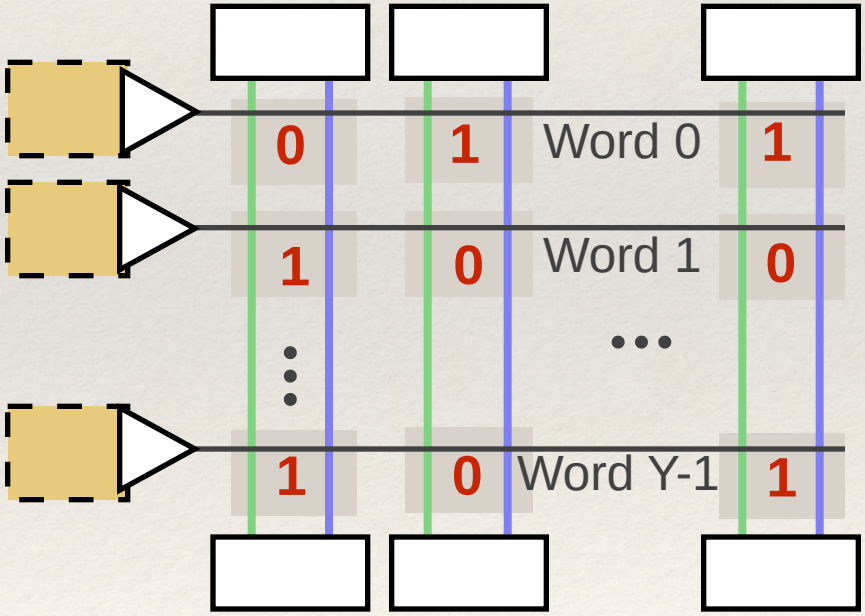
Performance and Overhead



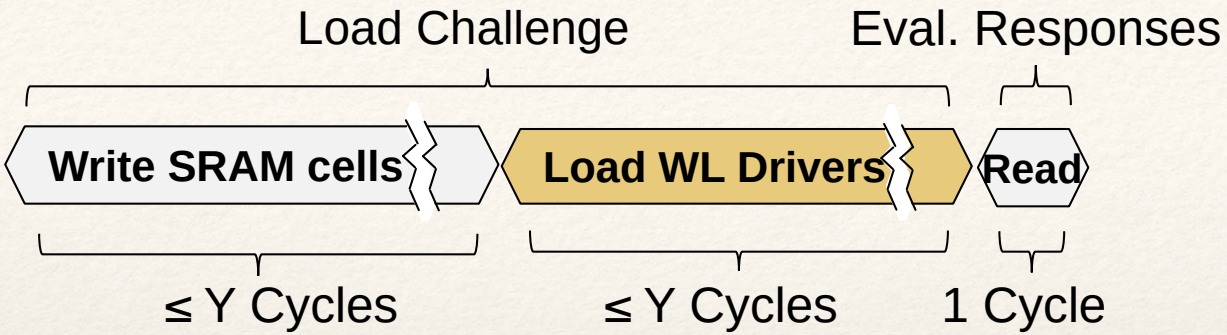


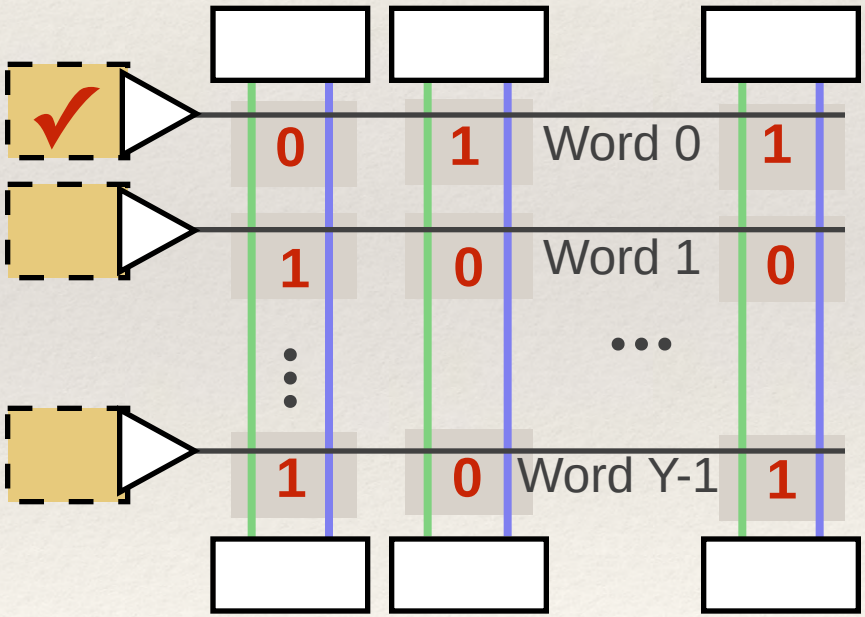
Performance and Overhead



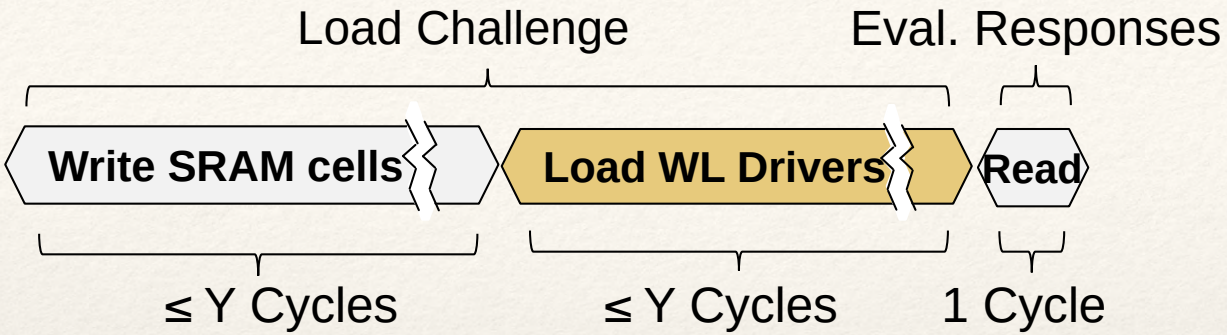


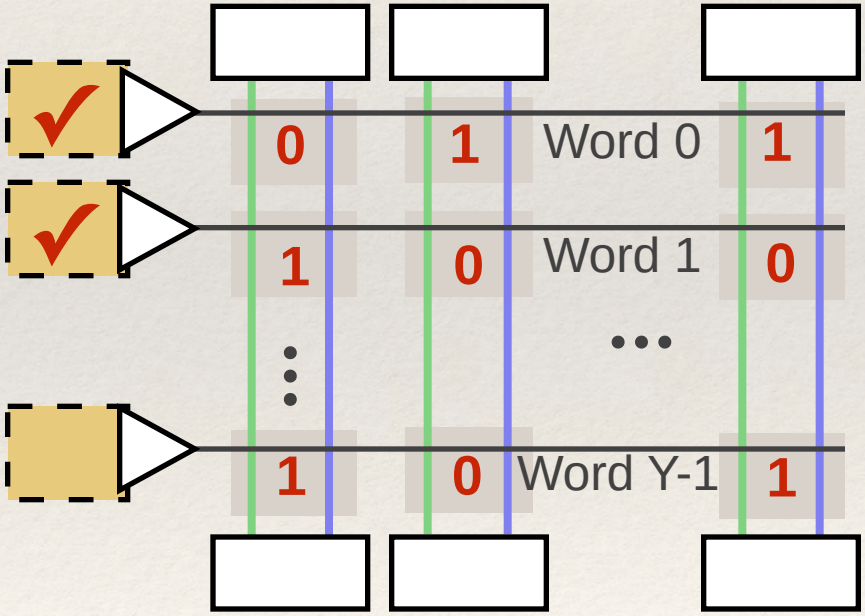
Performance and Overhead



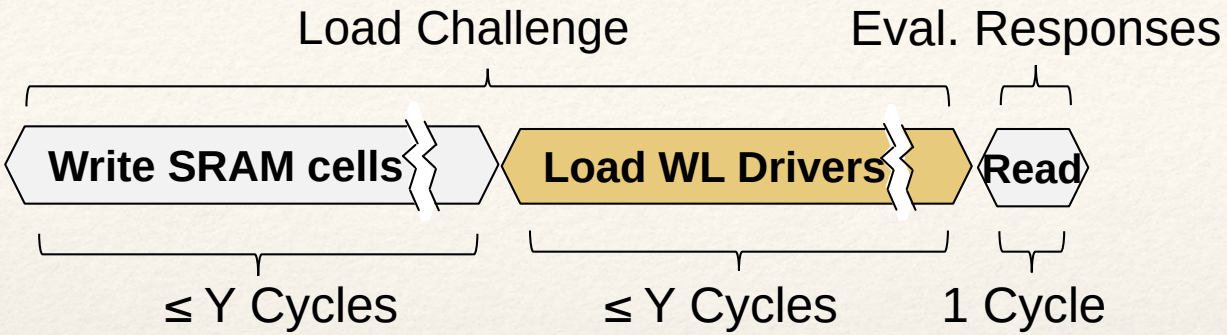


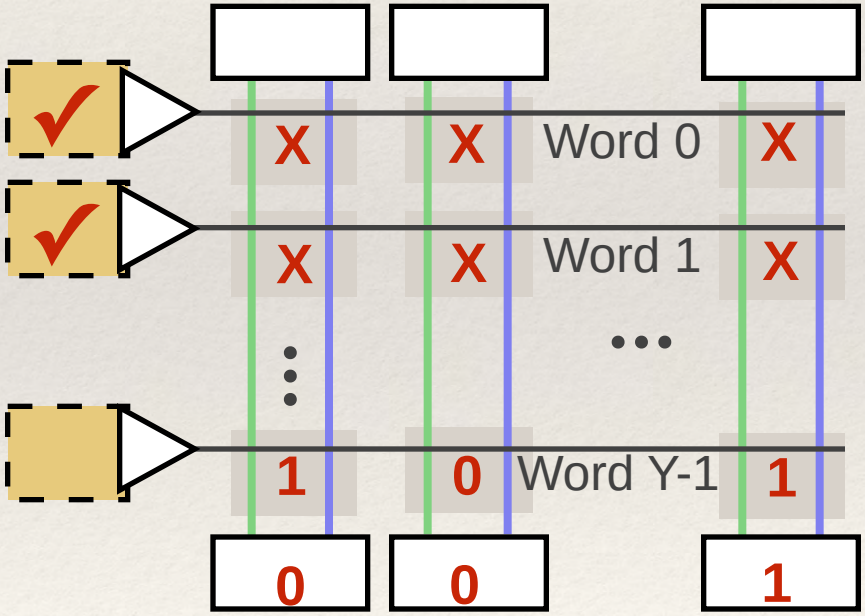
Performance and Overhead



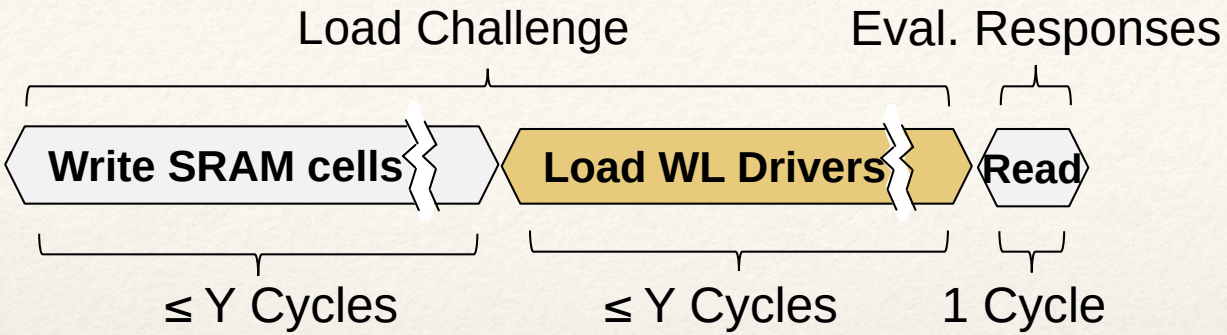


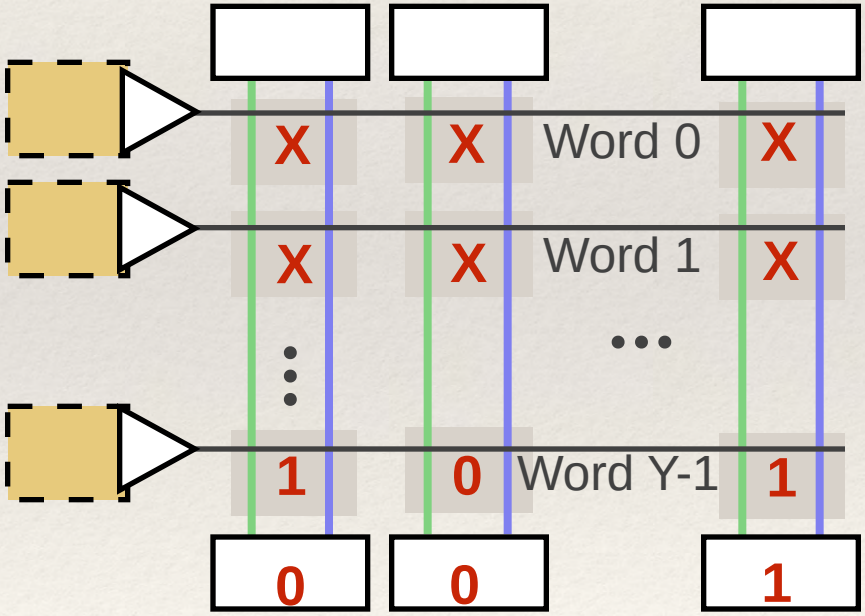
Performance and Overhead



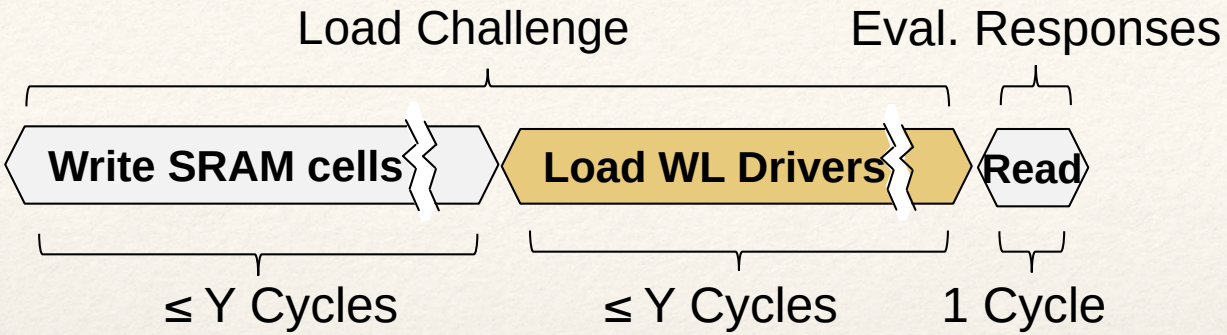


Performance and Overhead

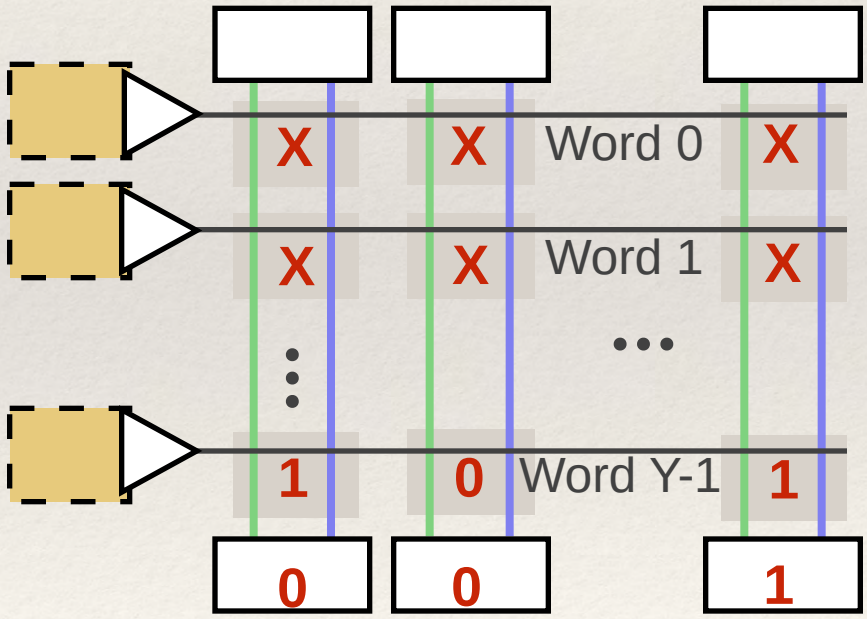




Performance and Overhead

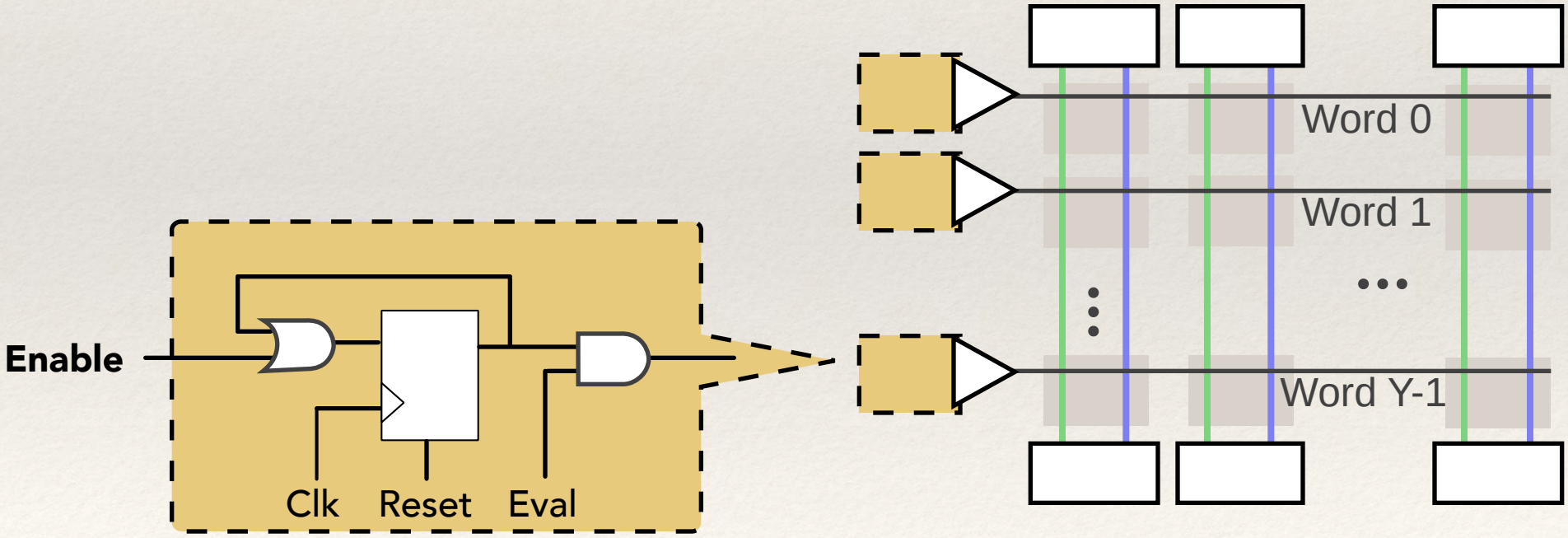


- Word-parallel (e.g. 256 columns)
- Response latency
  - 6 cycles for 256-bit response as shown
  - Depends on number of enabled rows
- Area overhead
  - A few extra gates per SRAM row
  - Don't need to add circuitry on all rows



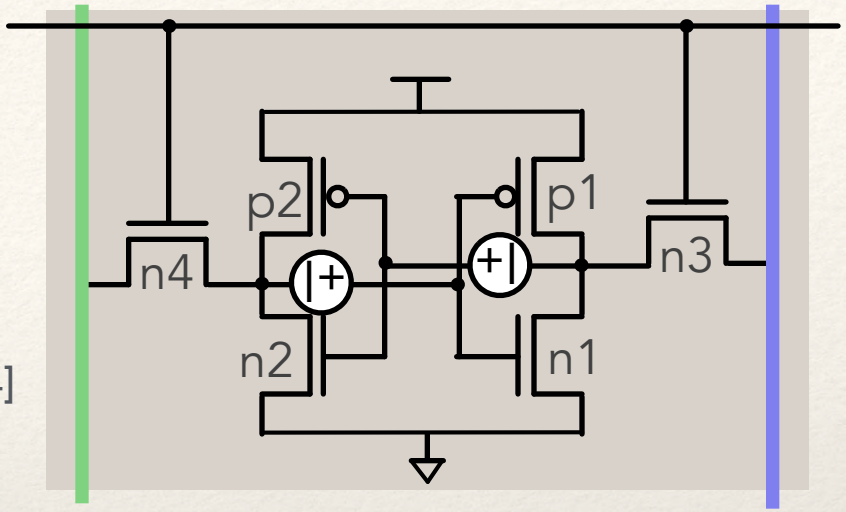
Integration

- Simple digital interface
- No power-cycling required
- Non-exclusive, SRAM rows still usable as memory when not used for PUF
- Does not upset stored data in non-used rows



Methodology

- Circuit simulation using Ngspice
- Devices are 90nm Predictive Technology Model [1]
- Sizing according to Nii et al. [2]
- Variation: threshold voltage and channel length [3,4]
- Noise: between cross-coupled nodes [5]

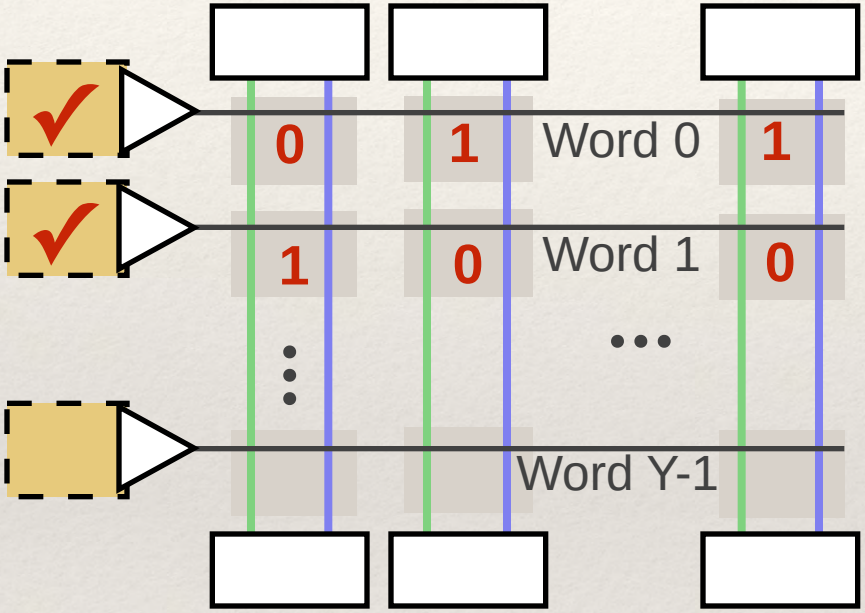


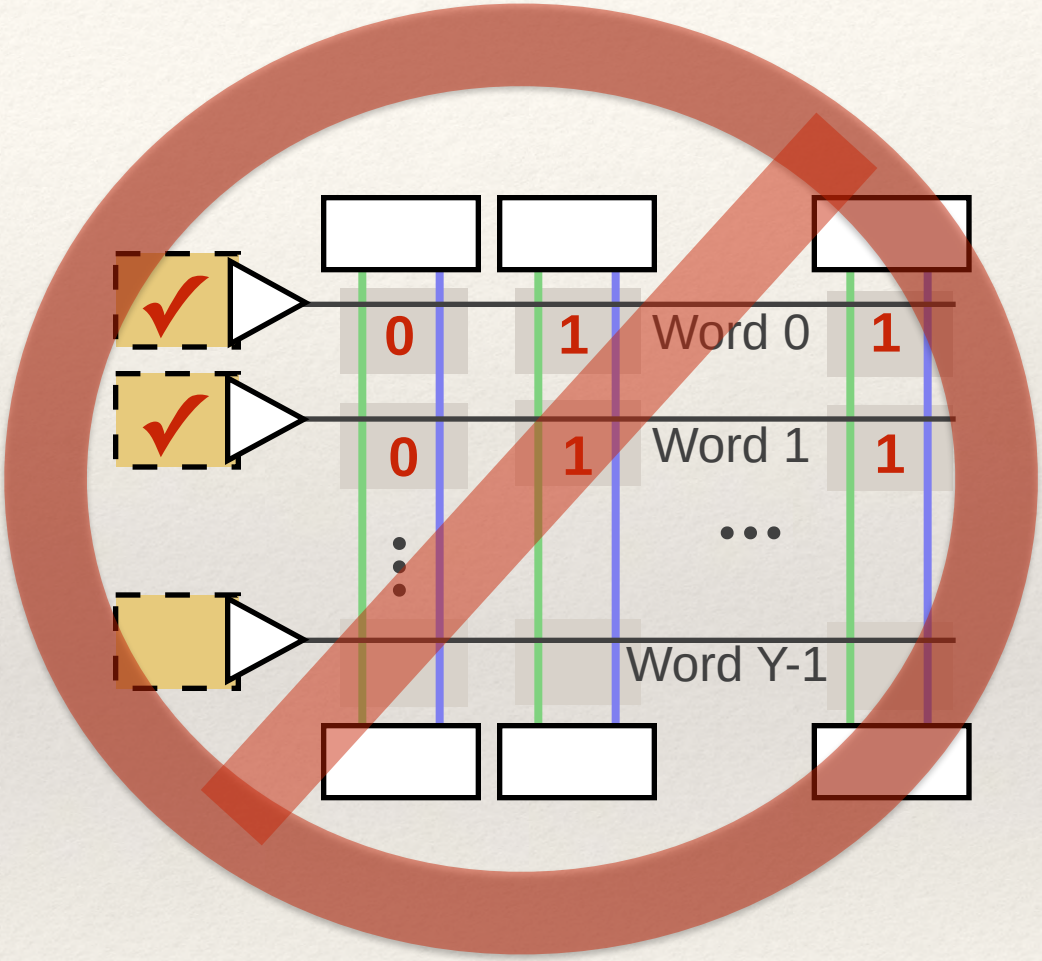
		Sizing		Process Variation			
				vth0 [mV]		lint [nm]	
		W [nm]	L [nm]	μ	σ	μ	σ
SRAM cell	n1,n2	200	90	397	13.4	7.5	3
	n3,n4	140	90	397	16.0	7.5	3
	p1,p2	140	90	-339	16.0	7.5	3
Sense Amp & Precharge	NMOS	1000	90	397	6.0	7.5	3
	PMOS	1000	90	-339	6.0	7.5	3

experiment code available online: https://github.com/danholcomb/bitline-puf

[1] Predictive Technology Model. 90nm NMOS and PMOS BSIM4 Models
[2] Nii et al., IEEE Journal of Solid State Circuits, 2004
[3] Pelgrom et al. IEEE Journal of Solid State Circuits, 1989
[4] Seevinck et al. IEEE Journal of Solid State Circuits, 1987
[5] Anis et al. Workshop on System-on-Chip for Real-Time Applications, 2005

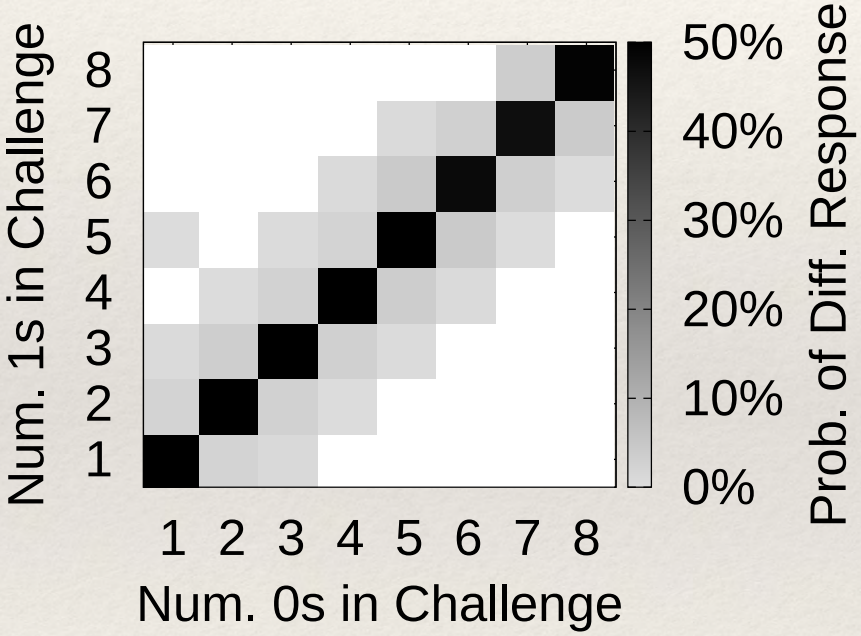
Choosing Useful Challenges

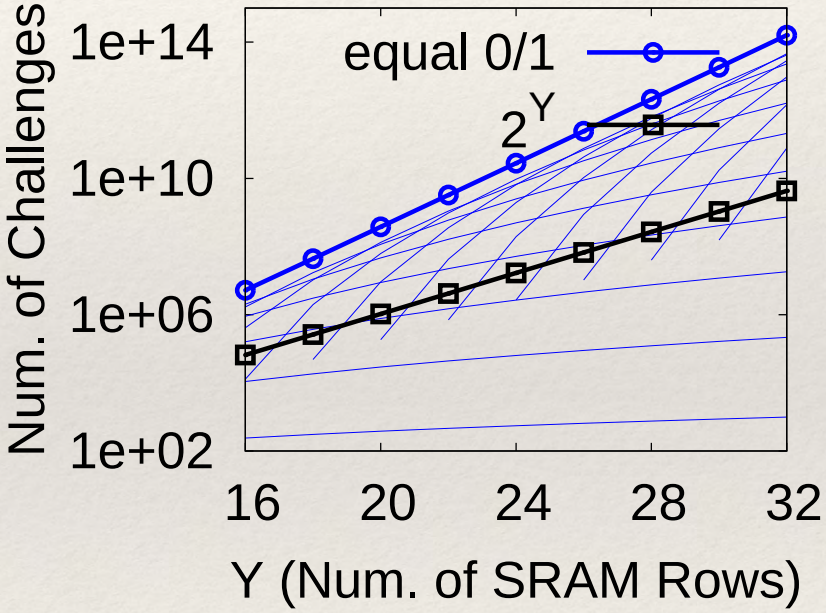




Choosing Useful Challenges

- Useful challenges have equal number of 0s and 1s
- Exponential subset of the 4^Y possible challenges

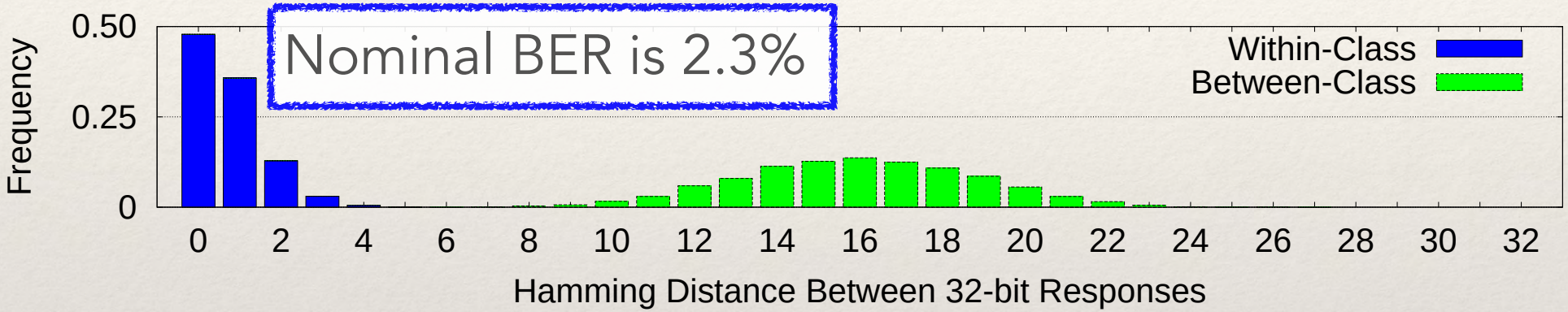




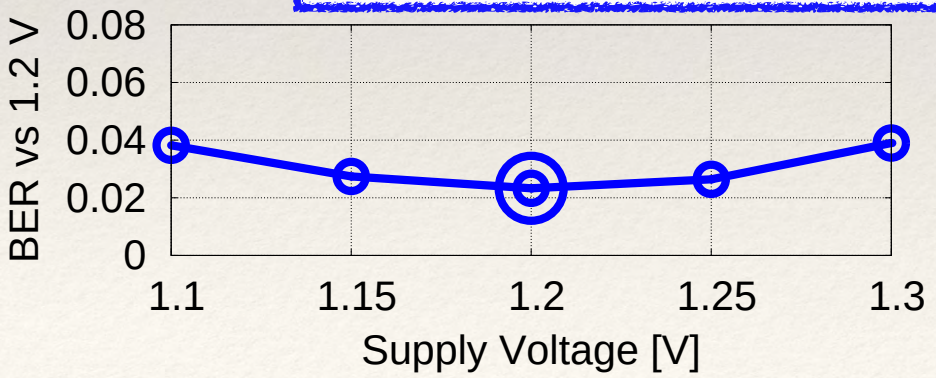
(Asymmetric designs may have different useful challenges)

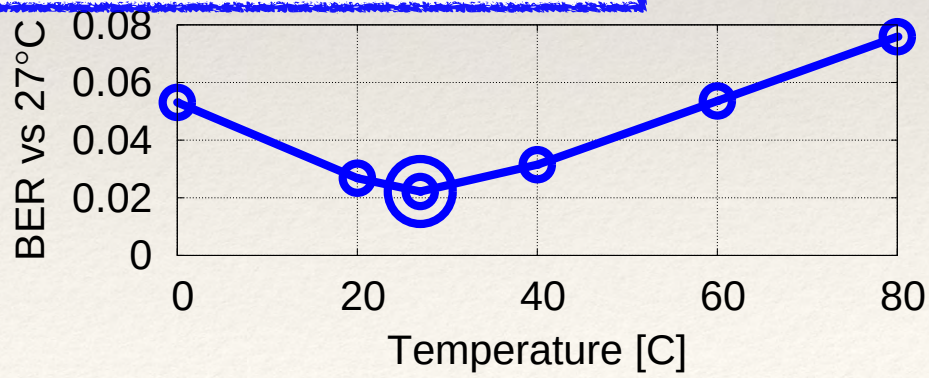
Uniqueness and Reliability

- Applying random challenges with equal number 0s and 1s
- Nominal conditions: 1.2V and 27°C



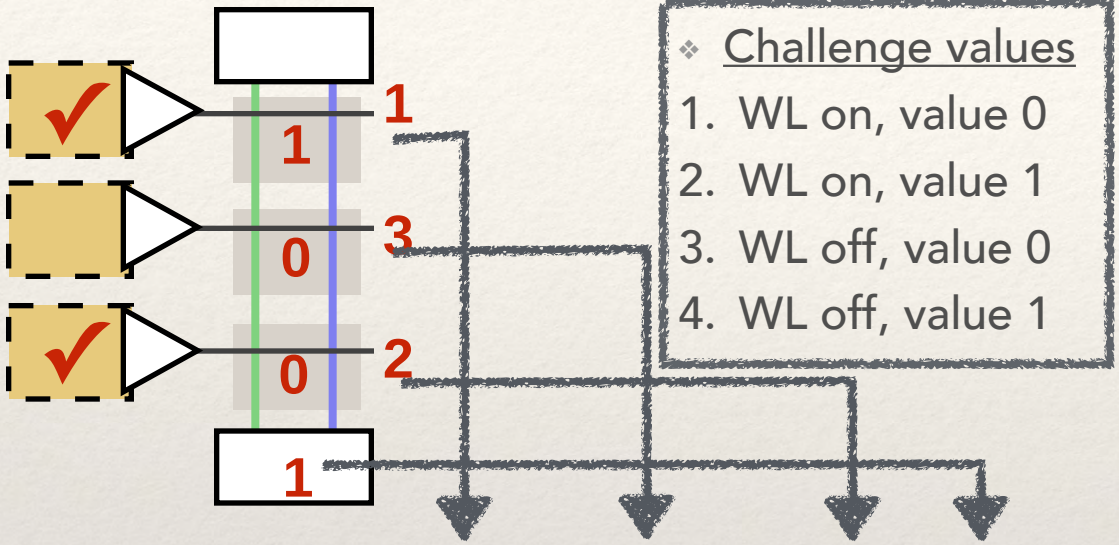
BER ≤ 7.6% across voltage and temperature



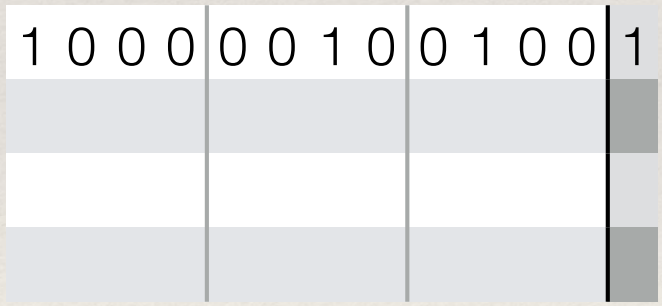


Modeling Attacks

- Can a model predict Bitline PUF's responses? **(Yes)**

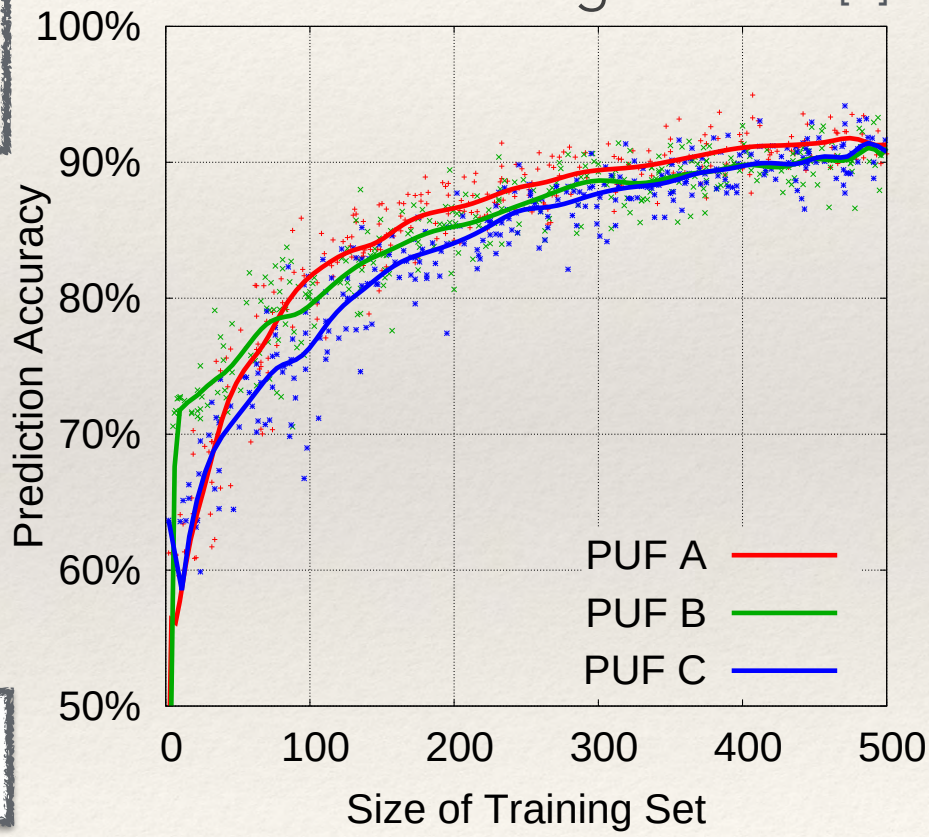


- Challenge values
  1. WL on, value 0
  2. WL on, value 1
  3. WL off, value 0
  4. WL off, value 1



- CRPs must be obfuscated

Classification using SVM^{light} [1]



[1] Joachims. Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, 1999

Summary

- PUFs as a new key storage mechanism

1. **SRAM power-up:** Use initial RAM state as basis for key
2. **DRV fingerprint:** Use minimum data retention voltage as basis for key
3. **Bitline PUF:** Modify SRAM array to enable physical challenge-response hashing

Thank you for your attention.

Questions?