Design Automation and Test in Europe 2014

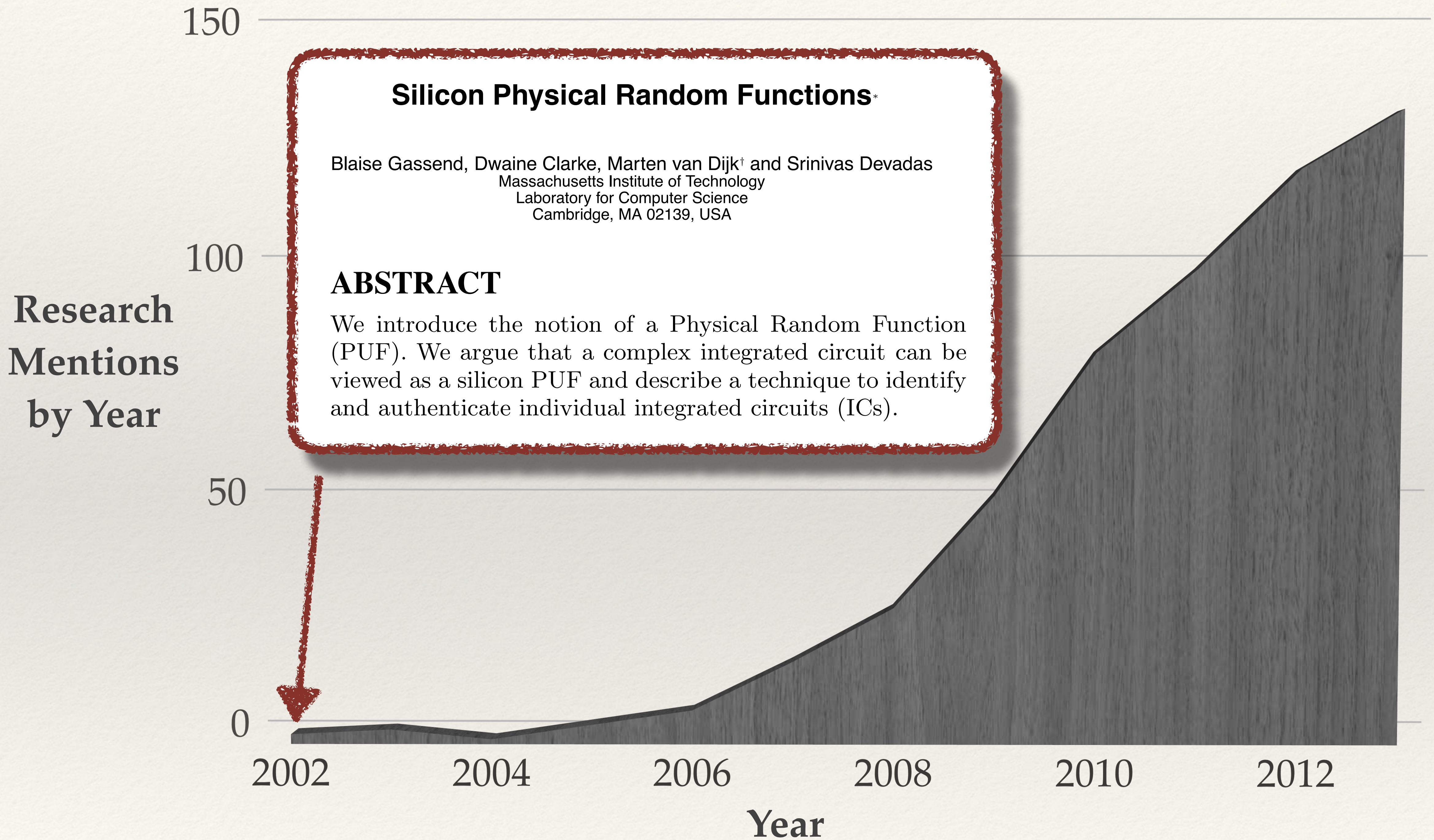
PUFs at a Glance

Ulrich Rührmair
Technische Universität München

Daniel E. Holcomb
University of Michigan

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 NSF CNS-0845874.

Physical Unclonable Functions



Overview

Context and motivation for remainder of session

1. **Brief introduction to PUFs**
2. Weak PUFs and applications
3. Strong PUFs and applications
4. Conclusions

Physical Unclonable Functions

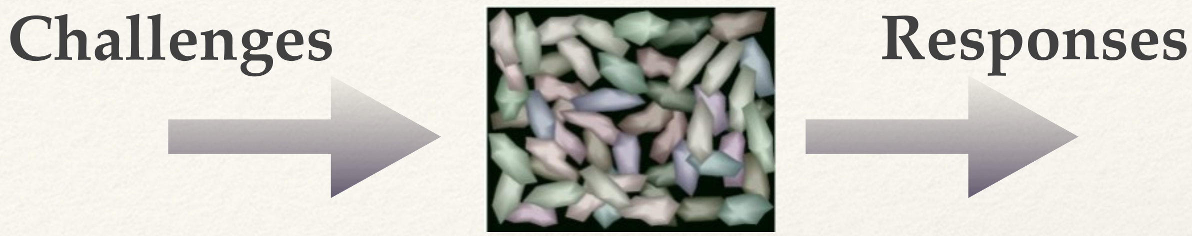
Physical Unclonable Functions

- Function
  - Map challenges to responses

Physical Unclonable Functions

- Function
  - Map challenges to responses
- Physical
  - Mapping depends on physical variations

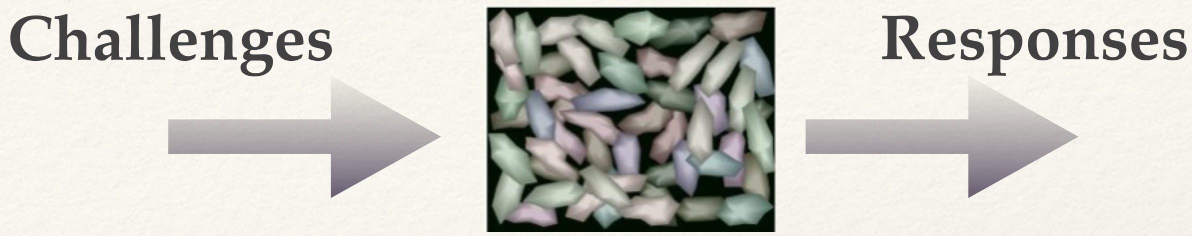
Physical Unclonable Functions



- Function
  - Map challenges to responses
- Physical
  - Mapping depends on physical variations

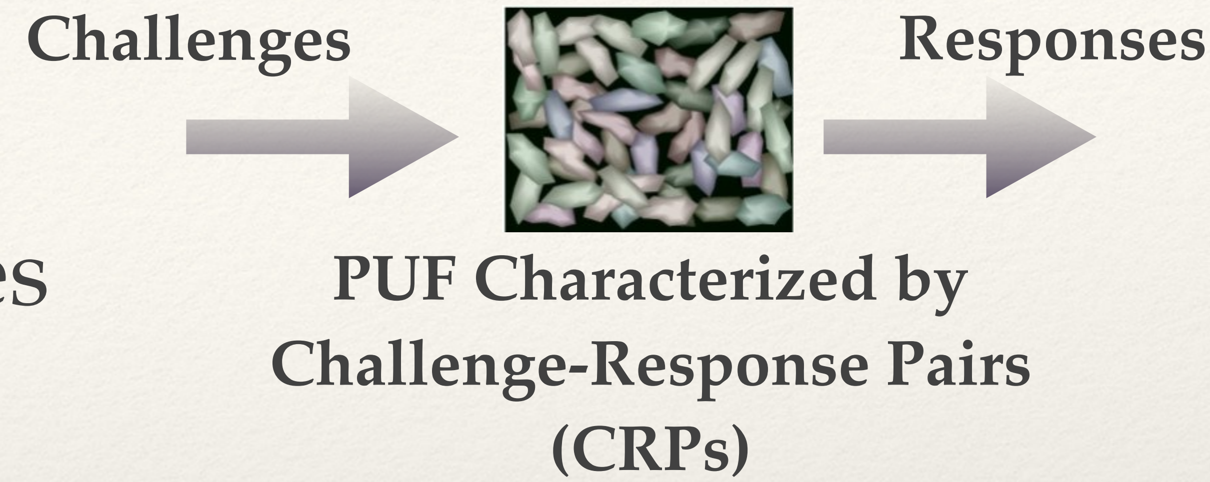
Physical Unclonable Functions

- Function
  - Map challenges to responses
- Physical
  - Mapping depends on physical variations



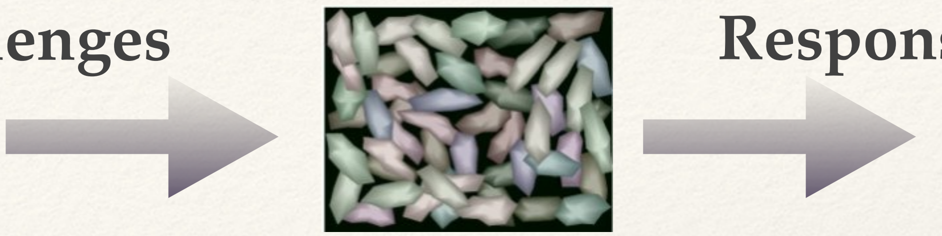
PUF Characterized by Challenge-Response Pairs (CRPs)

Physical Unclonable Functions



- Function
  - Map challenges to responses
- Physical
  - Mapping depends on physical variations
- Unclonable
  - No compact model exists, and CRP space is too large for dictionary

Physical Unclonable Functions



- Function
  - Map challenges to responses
- Physical
  - Mapping depends on physical variations
- Unclonable
  - No compact model exists, and CRP space is too large for dictionary
  - Or, responses kept secret

Design Considerations for Silicon PUFs

- Outputs determined by uncorrelated variation
  - Random dopant fluctuations and small devices
  - Balanced parasitics and wire lengths to avoid bias

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
  - Mask unreliable outputs
  - Majority voting
  - Error correction

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
  - Mask unreliable outputs
  - Majority voting
  - Error correction
- Secure

Security Considerations

- Assumed capabilities of adversary
  - Observe CRPs
  - Measure side channels
  - Disassemble and probe chip

Security Considerations

- Assumed capabilities of adversary
  - Observe CRPs
  - Measure side channels
  - Disassemble and probe chip
- Possible results of attacks
  - DOS by increasing error rate of CRPs
  - Train parametric model to predict responses
  - Clone with another instance of PUF

Security Considerations

- Assumed capabilities of adversary
  - Observe CRPs
  - Measure side channels
  - Disassemble and probe chip
- Possible results of attacks
  - DOS by increasing error rate of CRPs
  - Train parametric model to predict responses
  - Clone with another instance of PUF

2nd talk of session

Security Considerations

- Assumed capabilities of adversary
  - Observe CRPs
  - Measure side channels
  - Disassemble and probe chip
- Possible results of attacks
  - DOS by increasing error rate of CRPs
  - Train parametric model to predict responses
  - Clone with another instance of PUF

3rd talk of session

Security Considerations

- Assumed capabilities of adversary
  - Observe CRPs
  - Measure side channels
  - Disassemble and probe chip
- Possible results of attacks
  - DOS by increasing error rate of CRPs
  - Train parametric model to predict responses
  - Clone with another instance of PUF

4th talk of session

Weak vs Strong PUFs

Weak PUFs

Strong PUFs

Weak vs Strong PUFs

Weak PUFs

- Few/one challenges

Strong PUFs

- Many challenges

Weak vs Strong PUFs

Weak PUFs

- Few/one challenges
- Responses remain internal
  - Perfect internal error correction

Strong PUFs

- Many challenges
- Public CRP interface
  - Error correction outside PUF is possible

Weak vs Strong PUFs

Weak PUFs

- Few/one challenges
- Responses remain internal
  - Perfect internal error correction
- Attacks: Cloning and invasive reading of responses

Strong PUFs

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

Weak vs Strong PUFs

Weak PUFs

- Few/one challenges
- Responses remain internal
  - Perfect internal error correction
- Attacks: Cloning and invasive reading of responses
- Use cases: New form of key storage

Strong PUFs

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

Weak vs Strong PUFs

Weak PUFs

- Few/one challenges
- Responses remain internal
  - Perfect internal error correction
- Attacks: Cloning and invasive reading of responses
- Use cases: New form of key storage

Strong PUFs

- Many challenges
- Public CRP interface
  - Error correction outside PUF is possible
- Attacks: Modeling attacks and protocol attacks
- Use cases: New cryptographic primitive

Weak vs Strong PUFs

Weak PUFs

Strong PUFs

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

Overview

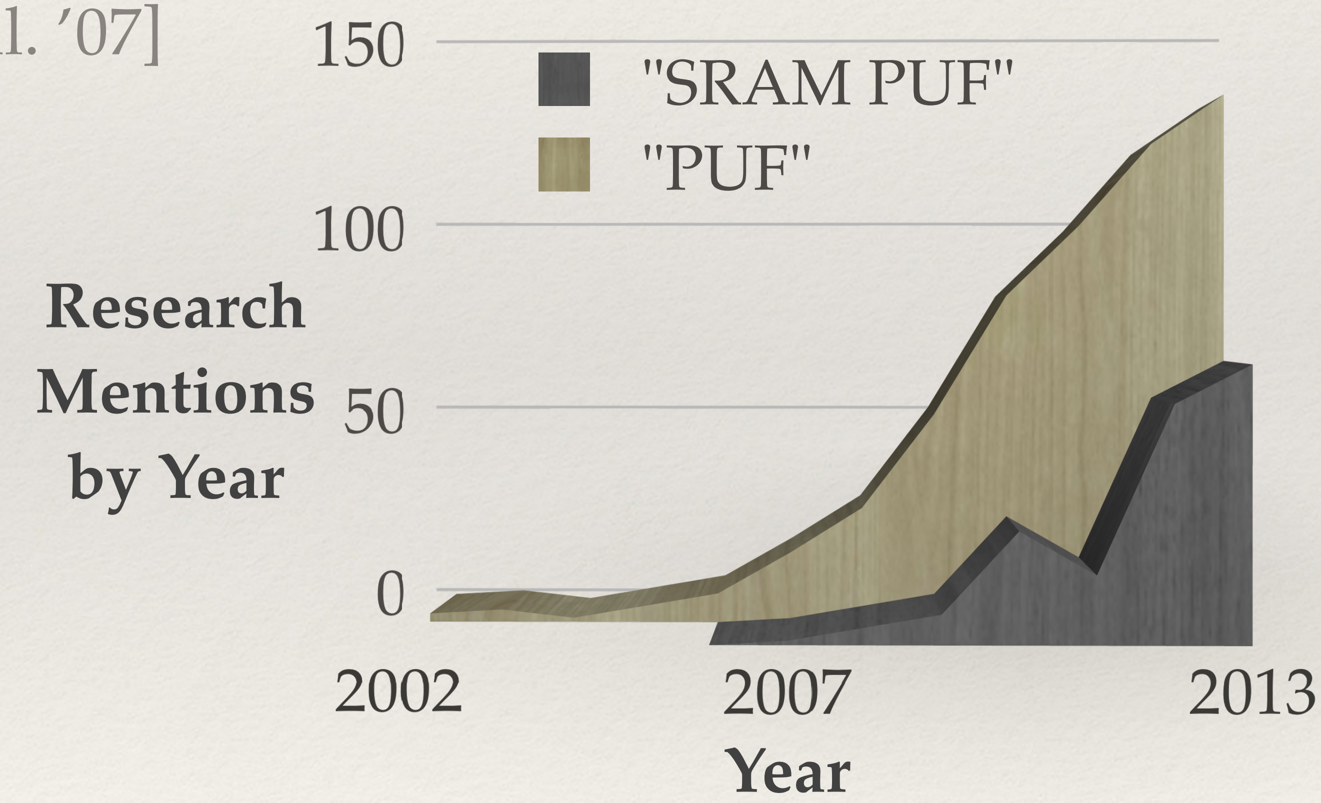
1. Brief introduction to PUFs
2. **Weak PUFs and applications**
3. Strong PUFs and applications
4. Conclusions

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]

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

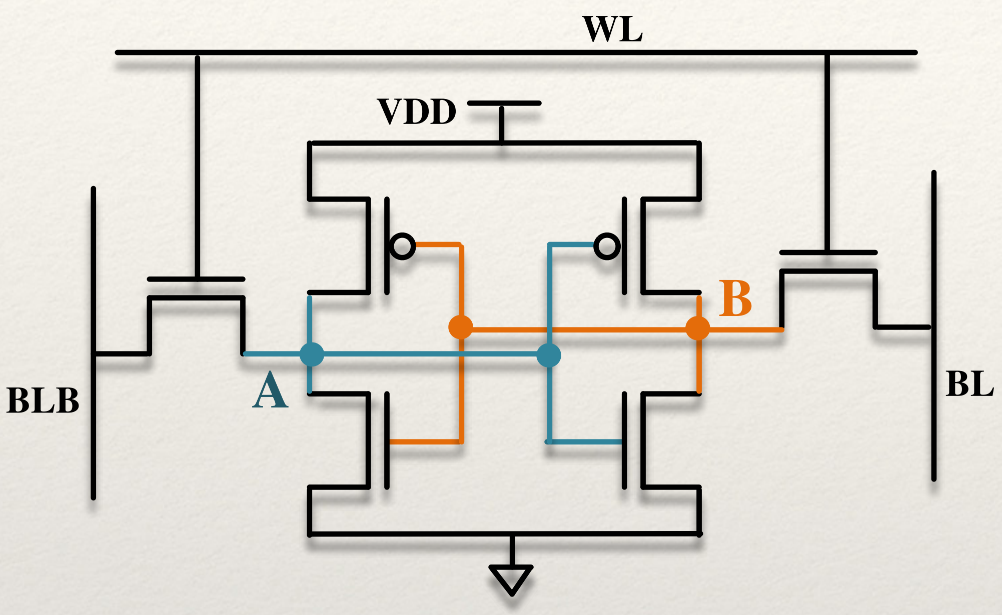
- Identification
- Authentication
- Secret key
- Random number generation

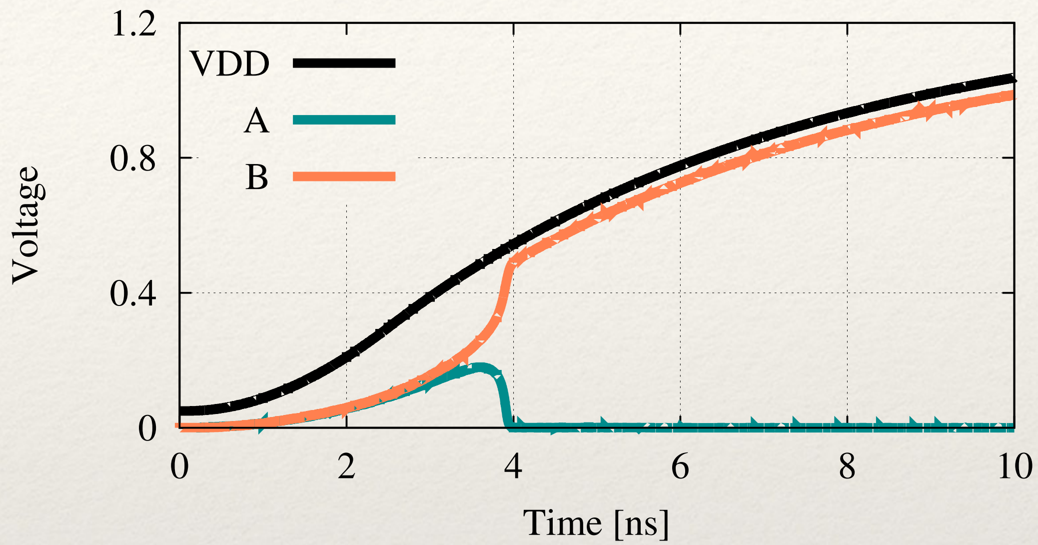
Applications of Weak PUFs

- Identification
- Authentication
- Secret key
- Random number generation

SRAM Power-up State

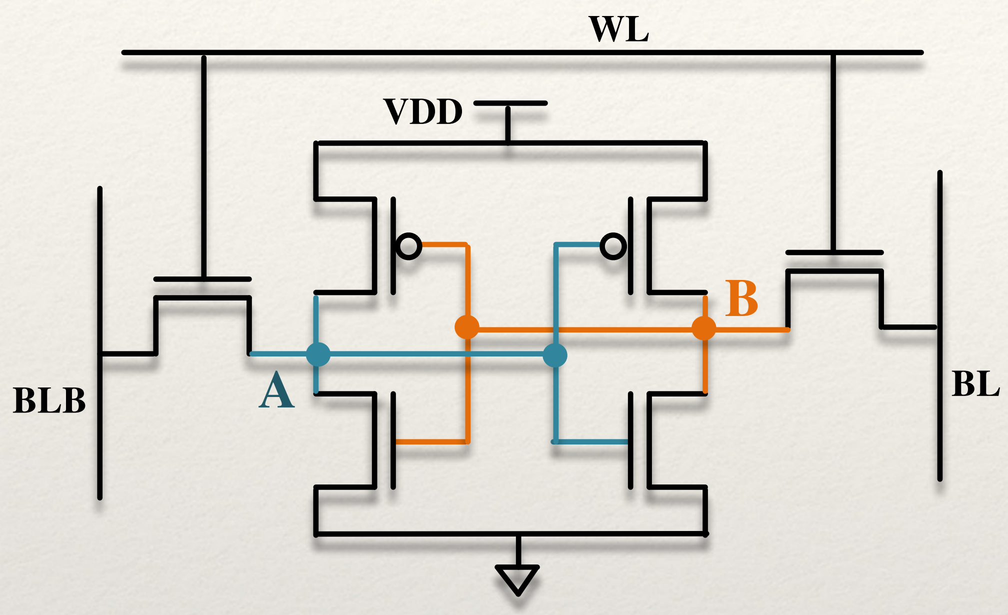
Utilize inherent power-up bias of each SRAM cell

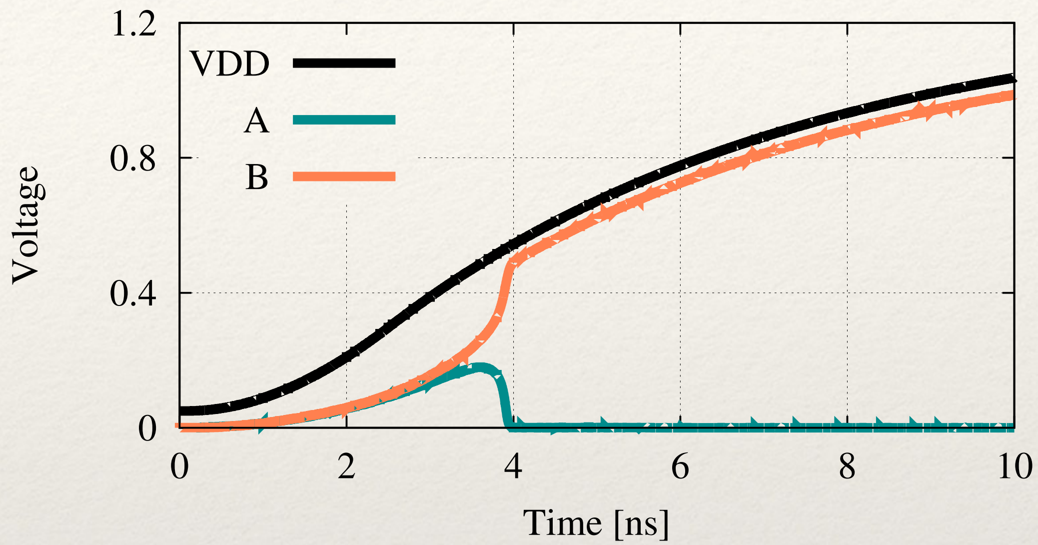




SRAM Power-up State

Utilize inherent power-up bias of each SRAM cell

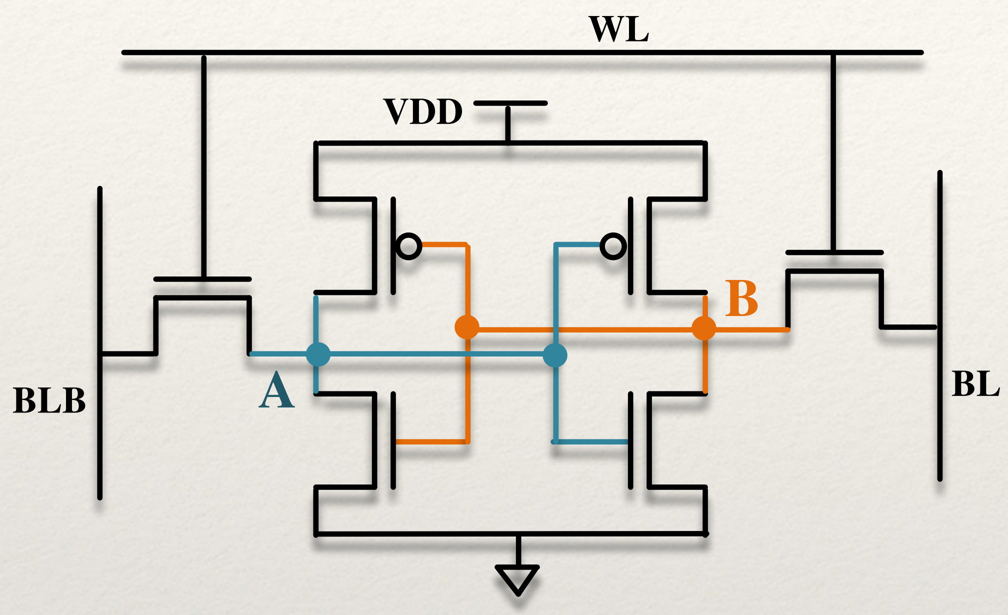


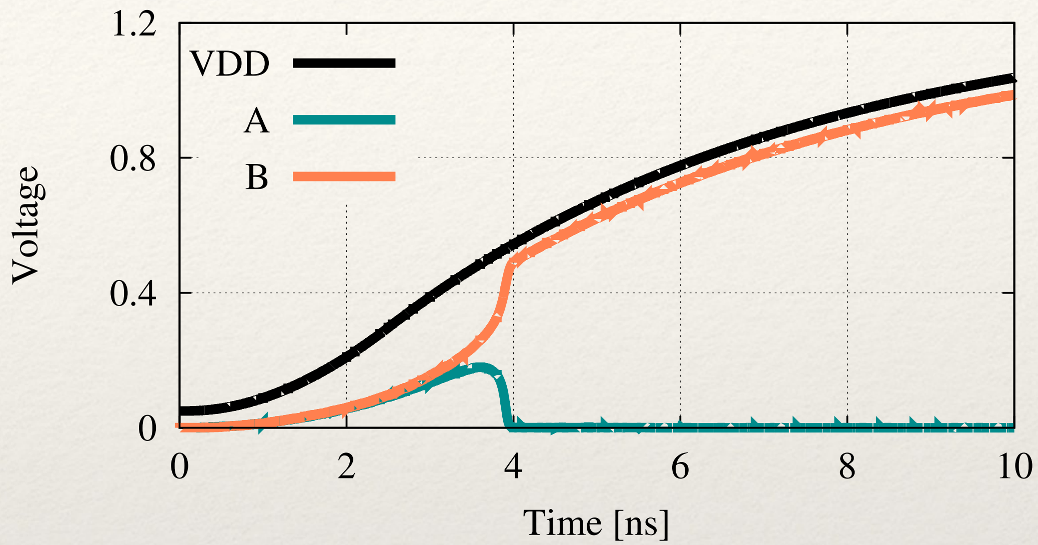


- Challenge: c (selects n cells)

SRAM Power-up State

Utilize inherent power-up bias of each SRAM cell

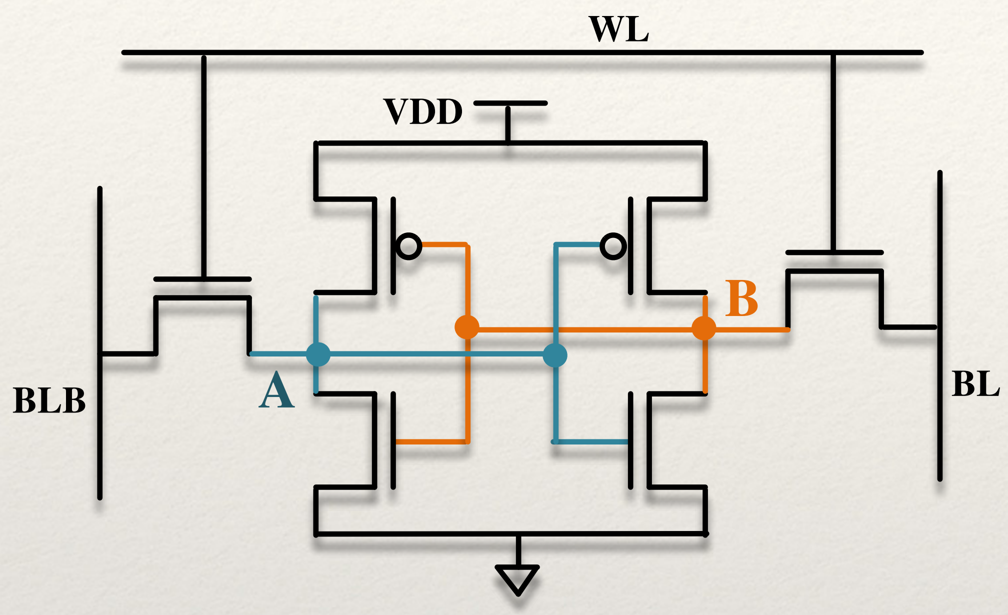


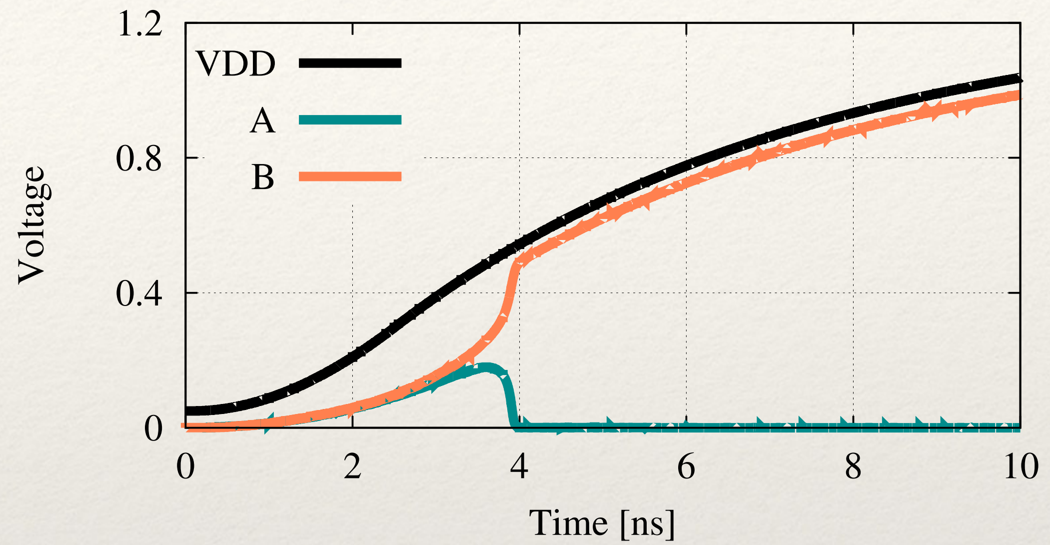


- Challenge: c (selects n cells)
- Responses: $r \in 2^n$ (power-up state of n cells)

SRAM Power-up State

Utilize inherent power-up bias of each SRAM cell

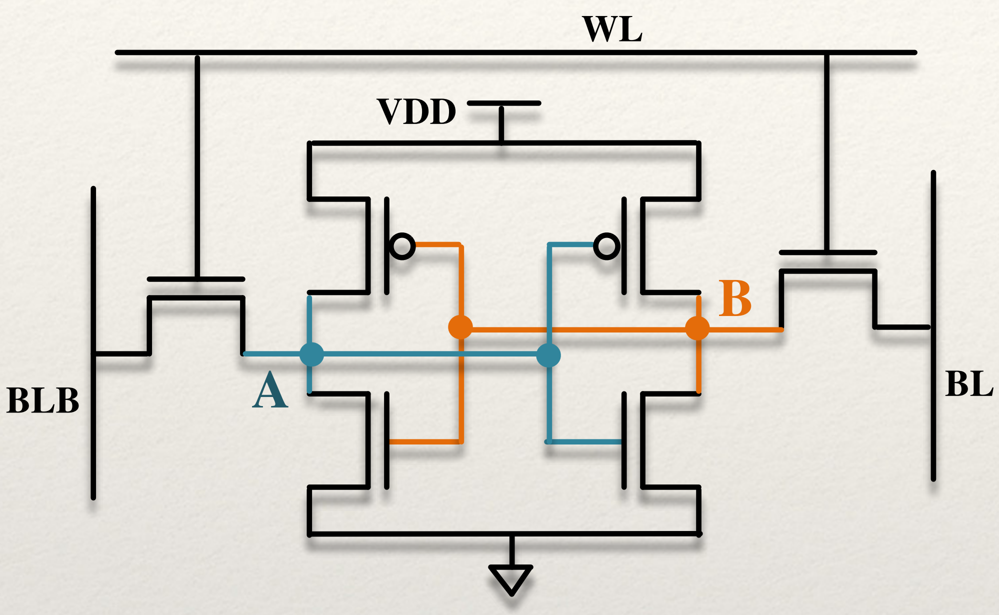


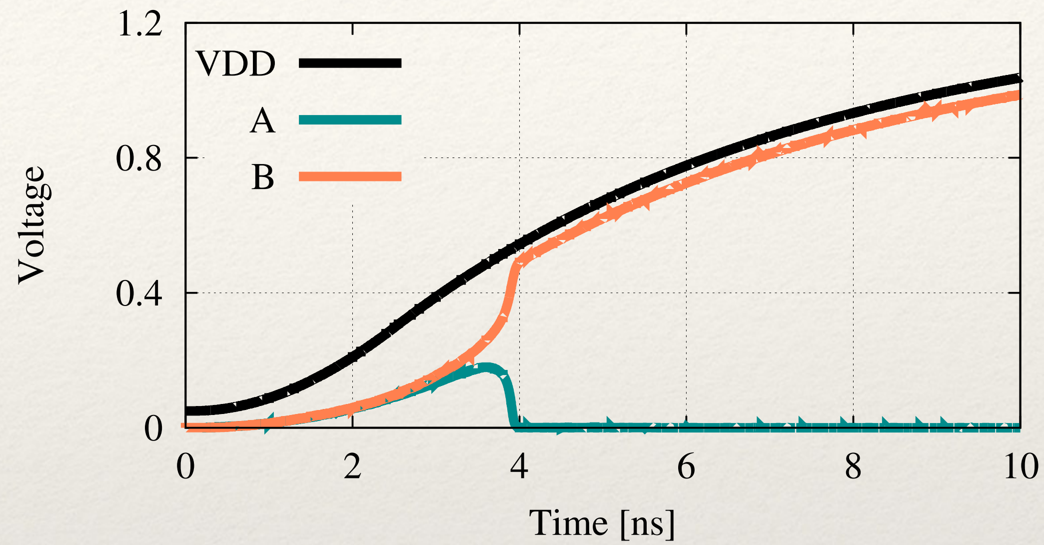


- Challenge: c (selects n cells)
- Responses: $r \in 2^n$
  (power-up state of n cells)
- Disorder/randomness: Threshold variation of transistors in SRAM cell

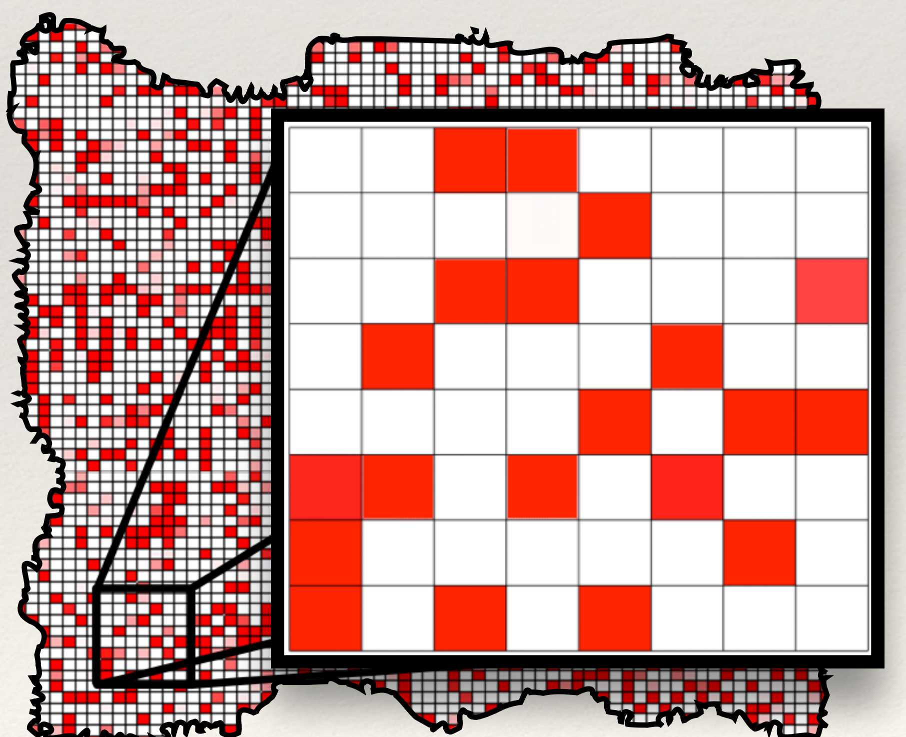
SRAM Power-up State

Utilize inherent power-up bias of each SRAM cell





- Challenge: c (selects n cells)
- Responses: $r \in 2^n$
  (power-up state of n cells)
- Disorder/randomness: Threshold variation of transistors in SRAM cell

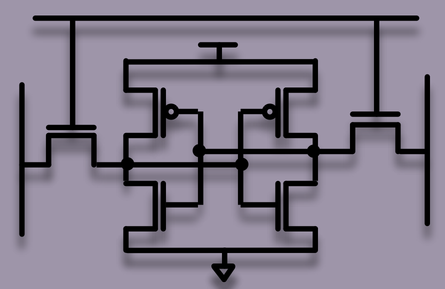


[Holcomb et al., '07]

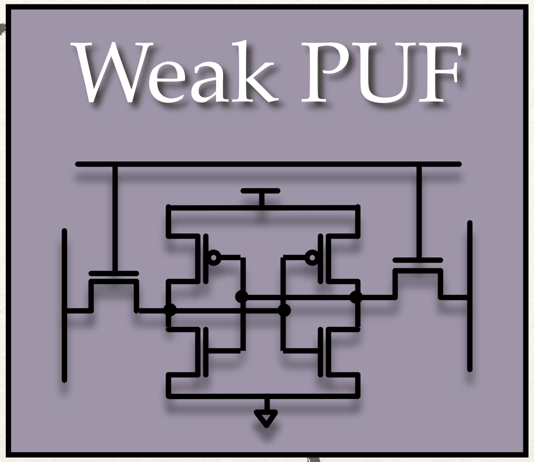
Weak PUF as Secret Key

Enroll PUF

Weak PUF



Weak PUF as Secret Key



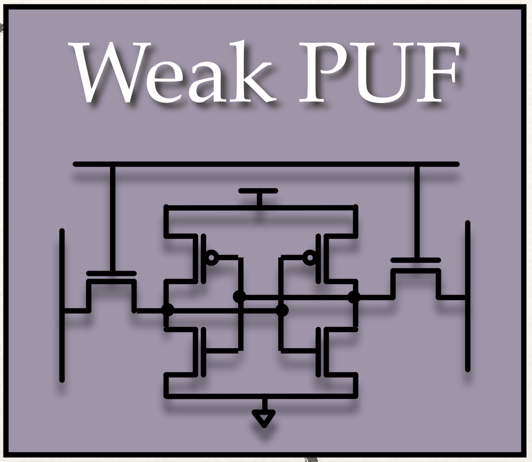
Enroll PUF

- Learn CRP (c,r)

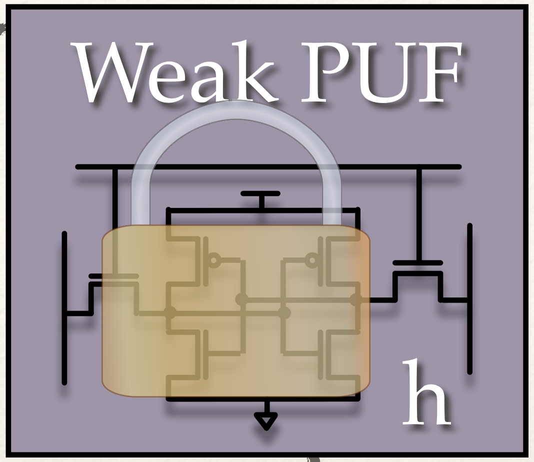
Weak PUF as Secret Key

<u>Enroll PUF</u>

- Learn CRP (c,r)
- Derive public error correcting data h for r
- Key k = Decode(r ⊕ h)



Weak PUF as Secret Key



Enroll PUF

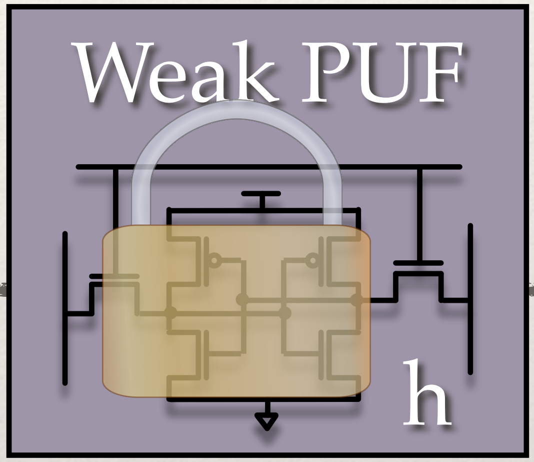
- Learn CRP (c,r)
- Derive public error correcting data h for r
- Key k = Decode(r ⊕ h)
- Store h with PUF
- Disable access to response r

Weak PUF as Secret Key

Enroll PUF

- Learn CRP (c,r)
- Derive public error correcting data h for r
- Key k = Decode(r $\oplus$ h)
- Store h with PUF
- Disable access to response r

Generate Key in Field

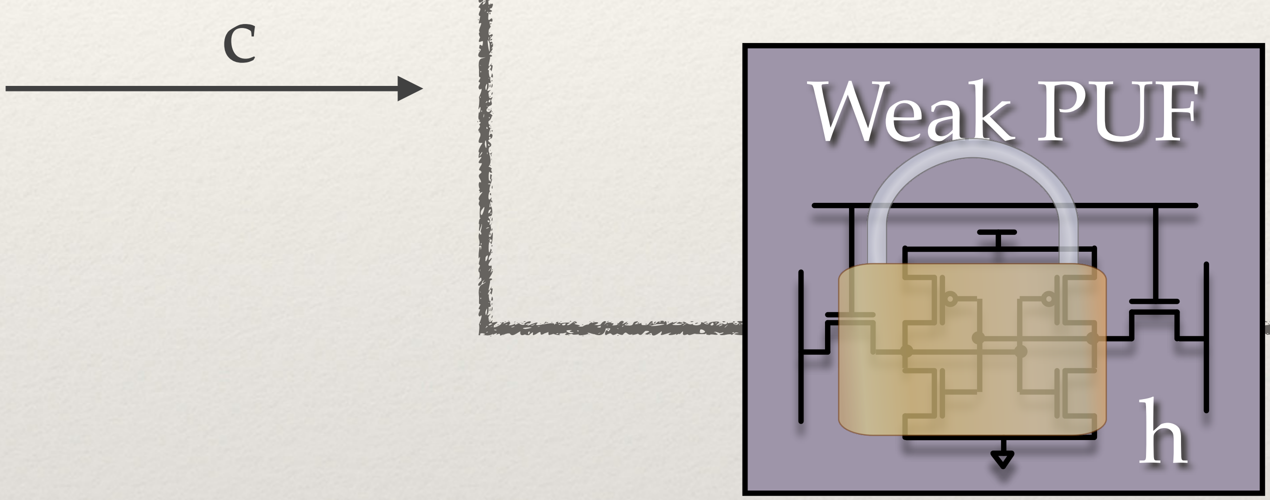


Weak PUF as Secret Key

Enroll PUF

- Learn CRP (c,r)
- Derive public error correcting data h for r
- Key k = Decode(r ⊕ h)
- Store h with PUF
- Disable access to response r

Generate Key in Field



Weak PUF as Secret Key

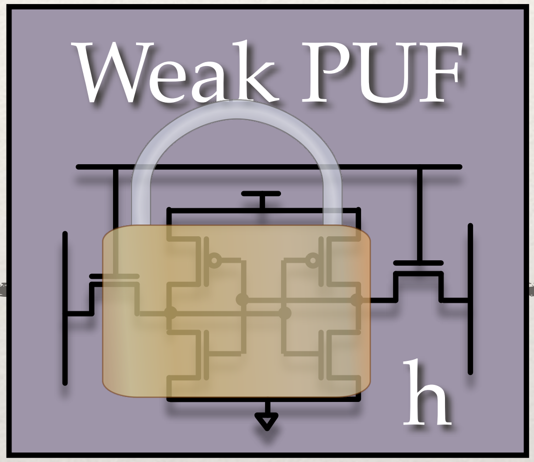
Enroll PUF

- Learn CRP (c,r)
- Derive public error correcting data h for r
- Key k = Decode(r ⊕ h)
- Store h with PUF
- Disable access to response r

c →

Generate Key in Field

- Apply c, obtain r′ ⊕ h
- Key k = Decode(r′ ⊕ h)



Weak PUF as Secret Key

Enroll PUF

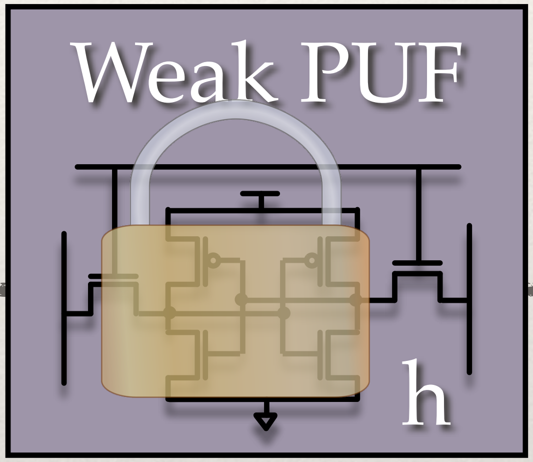
- Learn CRP (c,r)
- Derive public error correcting data h for r
- Key k = Decode(r ⊕ h)
- Store h with PUF
- Disable access to response r

c

k is reliable key

Generate Key in Field

- Apply c, obtain r′ ⊕ h
- Key k = Decode(r′ ⊕ h)



Weak PUF as Secret Key

Enroll PUF

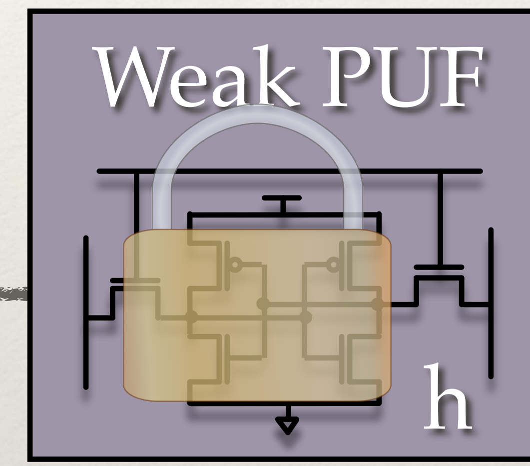
- Learn CRP (c,r)
- Derive public error correcting data h for r
- Key k = Decode(r ⊕ h)
- Store h with PUF
- Disable access to response r

c

k is reliable key

Generate Key in Field

- Apply c, obtain r′ ⊕ h
- Key k = Decode(r′ ⊕ h)



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

Overview

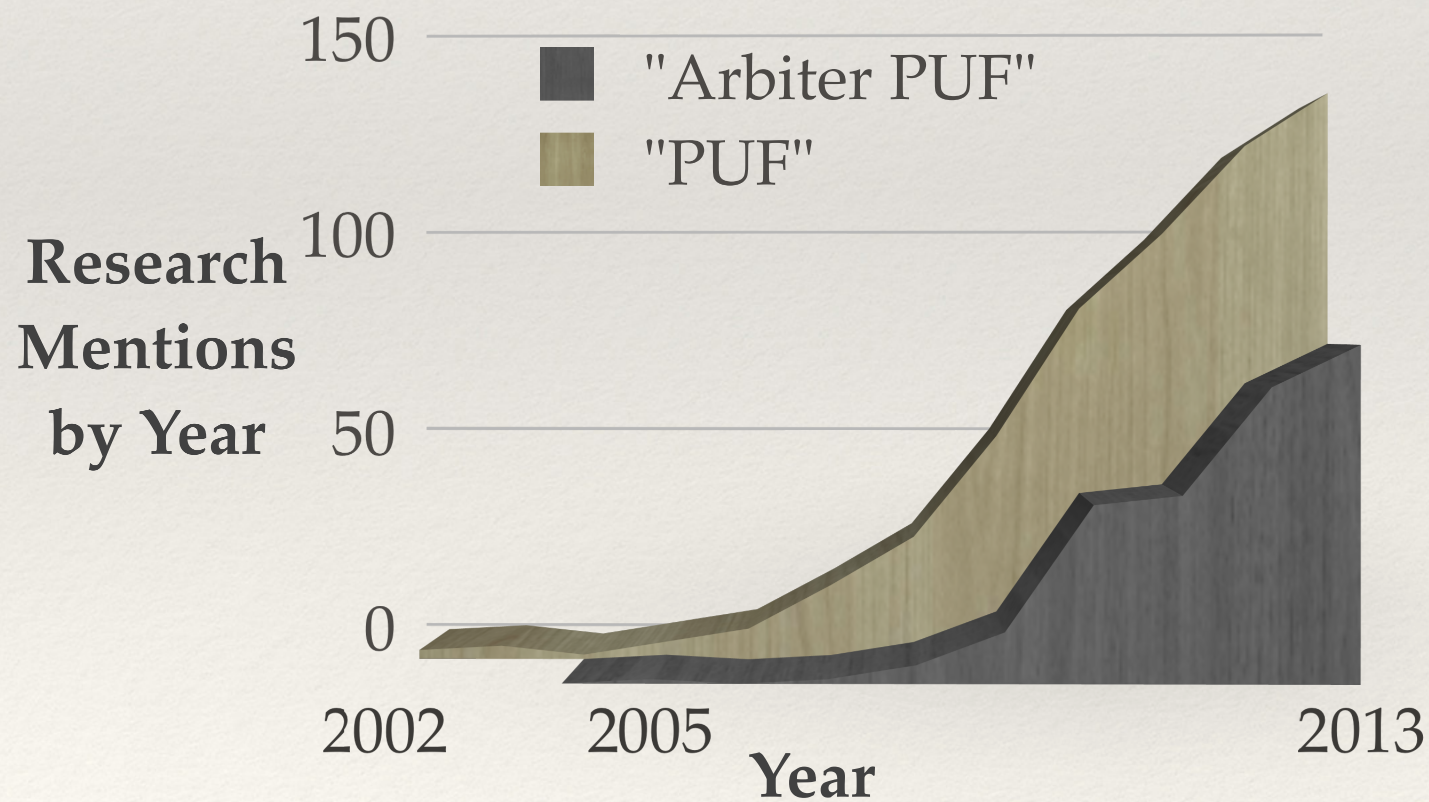
1. Brief introduction to PUFs
2. Weak PUFs and applications
3. **Strong PUFs and applications**
4. Conclusions

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]

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

Strong PUF Protocols

- Identification / Authentication (1)
- Key Exchange (2,3) ← 5th talk of session
- 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

Strong PUF Protocols

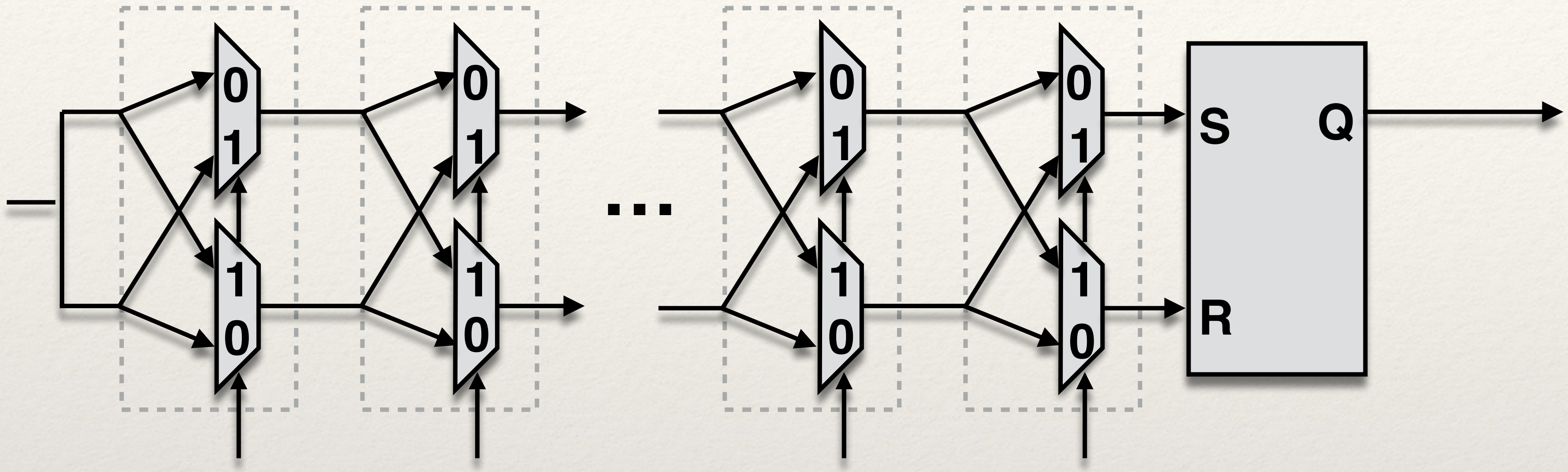
- Identification / Authentication (1)
- Key Exchange (2,3)
- Oblivious transfer (4,3,5,6) — enables secure two-party computation
- Bit commitment (3,5,6,7,8) — enables zero-knowledge proofs
- Combined key exchange and authentication (9)

5th talk of session

(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

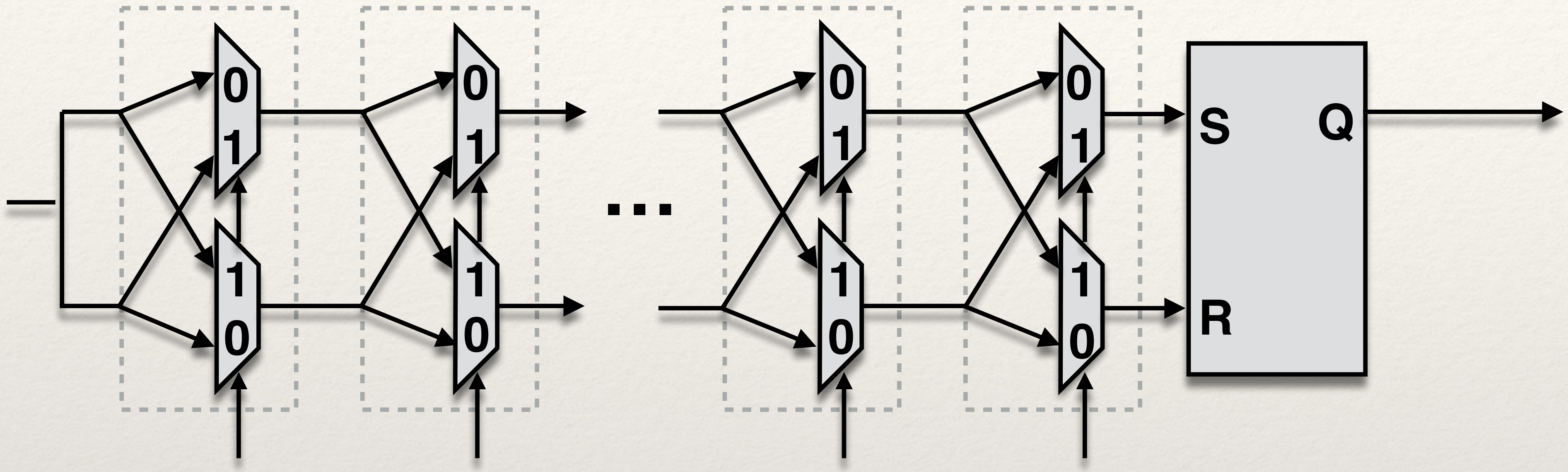
Arbiter PUF

[D. Lim et al., ’05]



Arbiter PUF

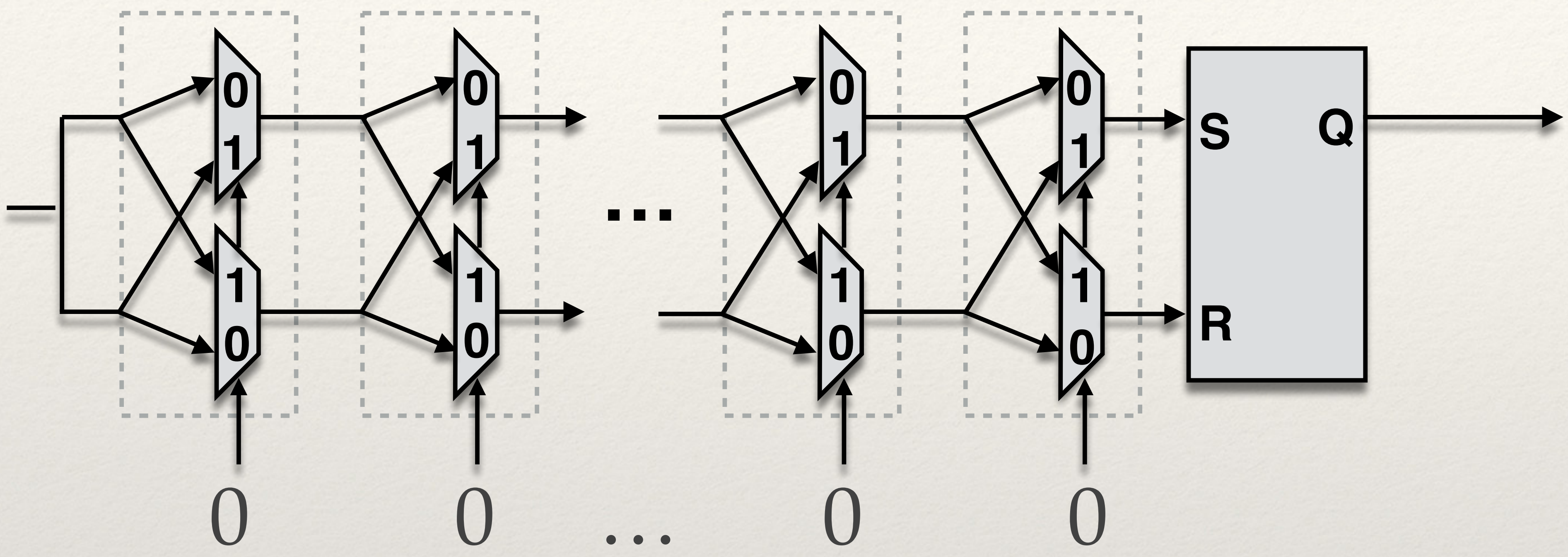
[D. Lim et al., '05]



- Challenges: $c_i \in 2^m$ (m= num stages)

Arbiter PUF

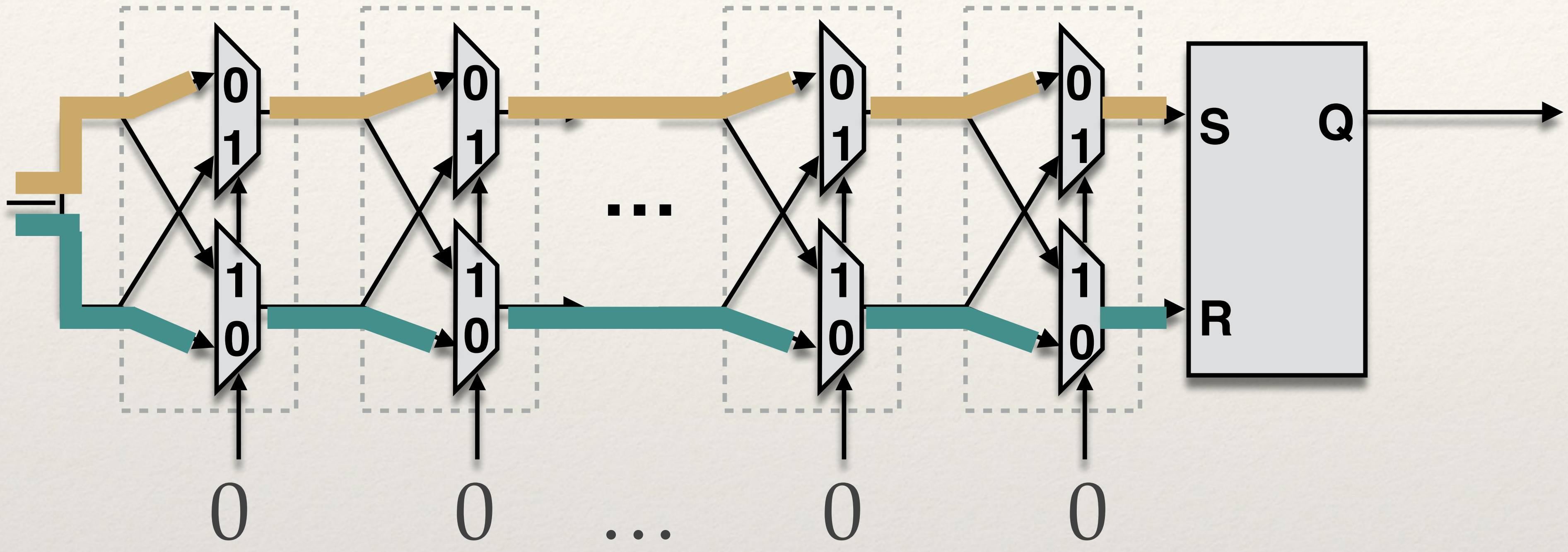
[D. Lim et al., ’05]



- Challenges: $c_i \in 2^m$ (m= num stages)

Arbiter PUF

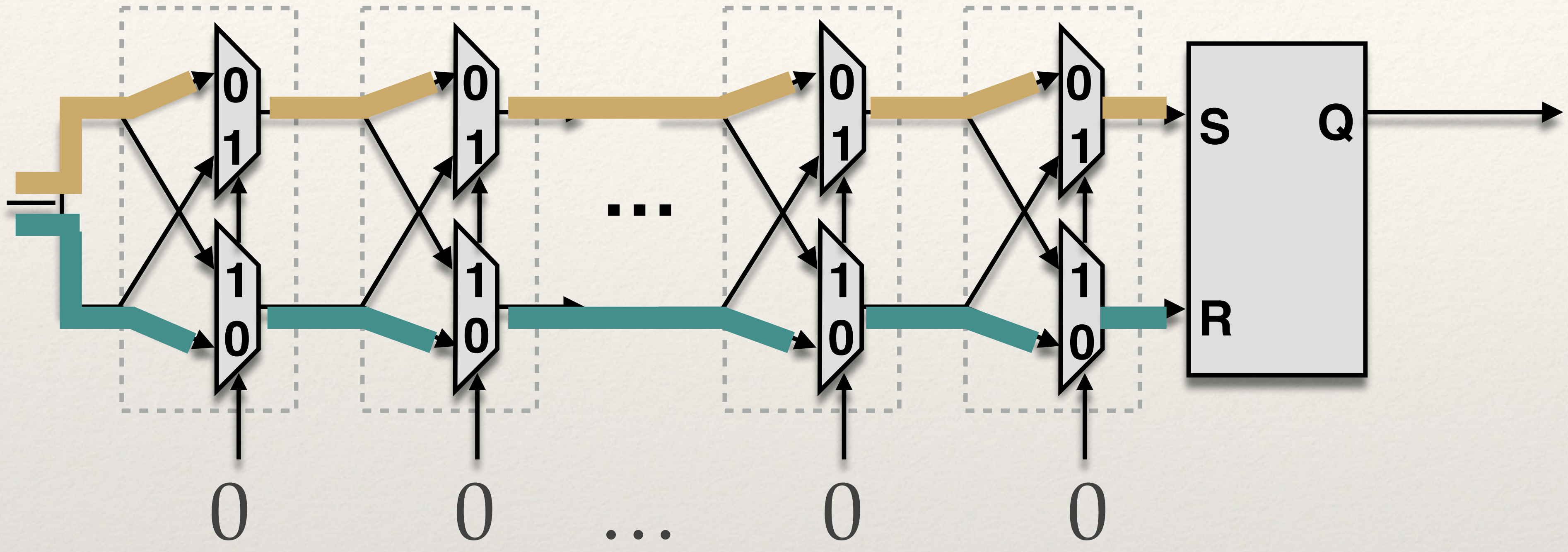
[D. Lim et al., '05]



- Challenges: $c_i \in 2^m$ (m= num stages)

Arbiter PUF

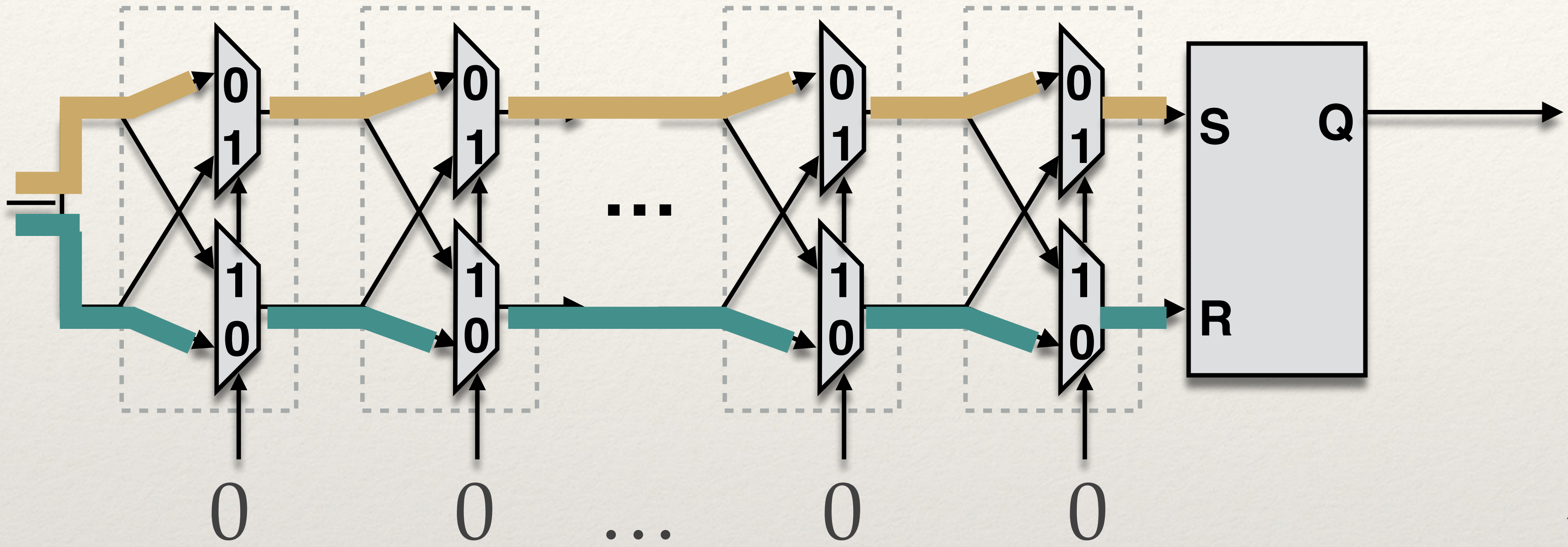
[D. Lim et al., ’05]

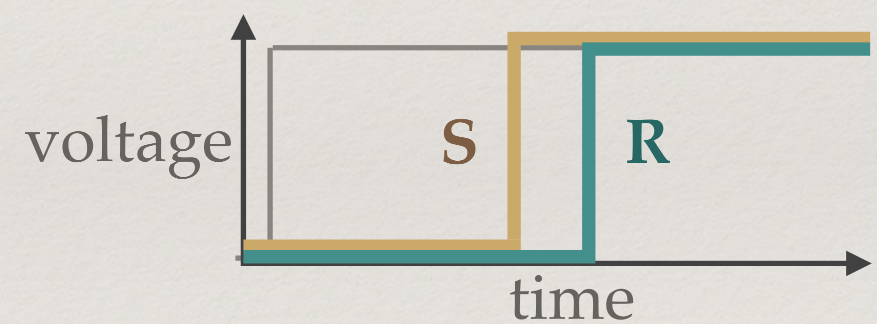


- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 2^n$ (n=1 shown)

Arbiter PUF

[D. Lim et al., ’05]

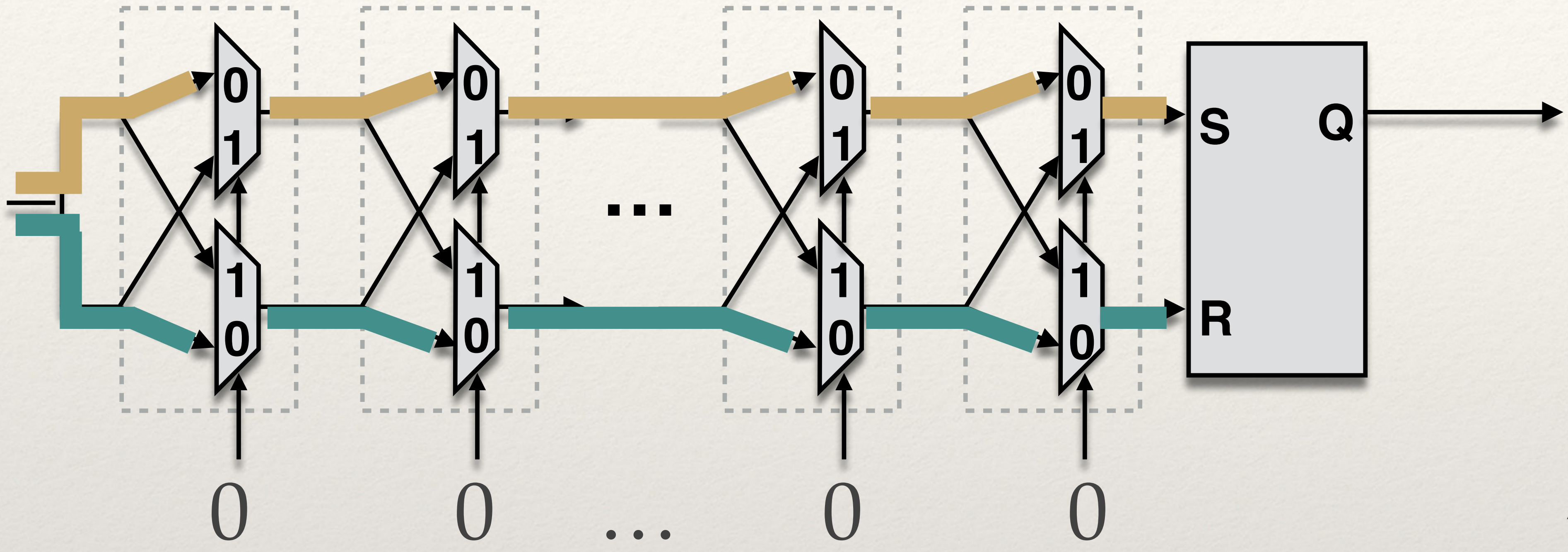


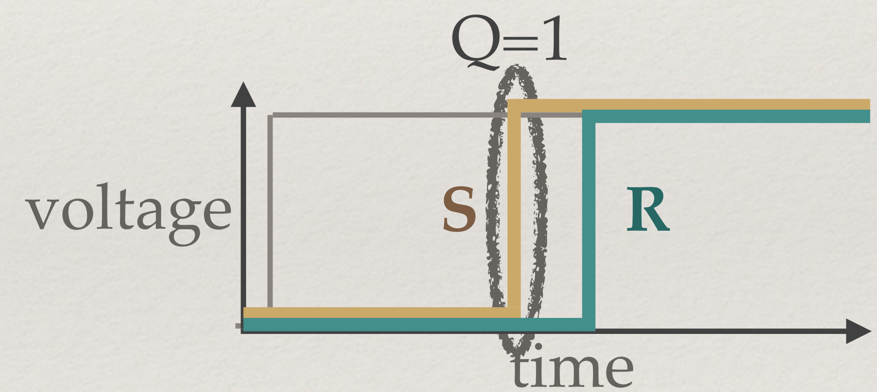


- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 2^n$ (n=1 shown)

Arbiter PUF

[D. Lim et al., ’05]

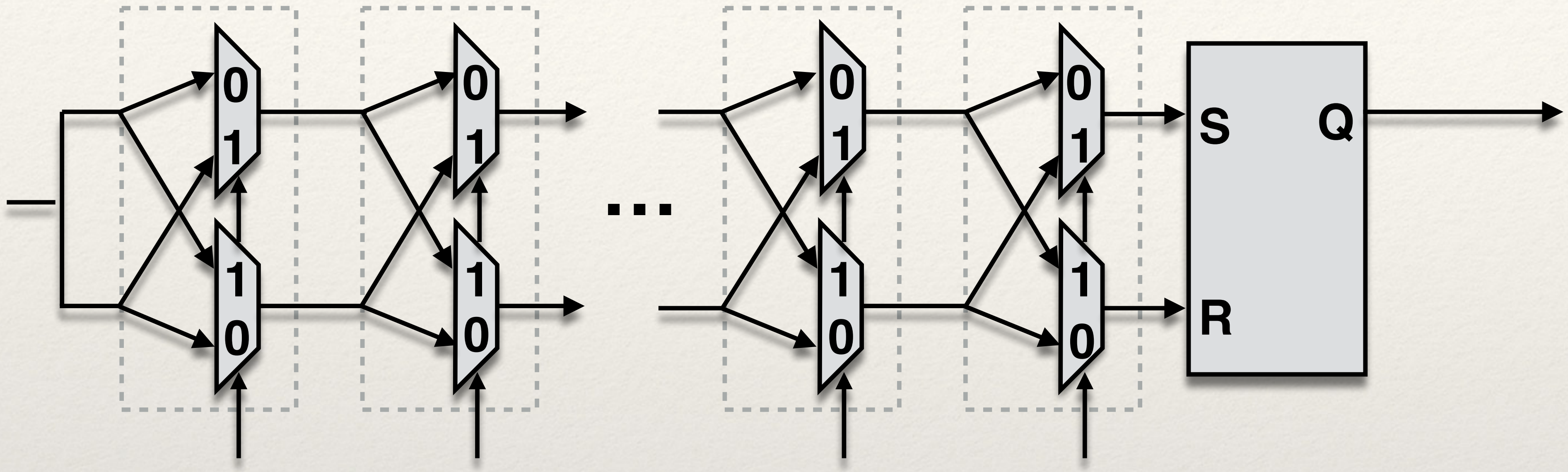


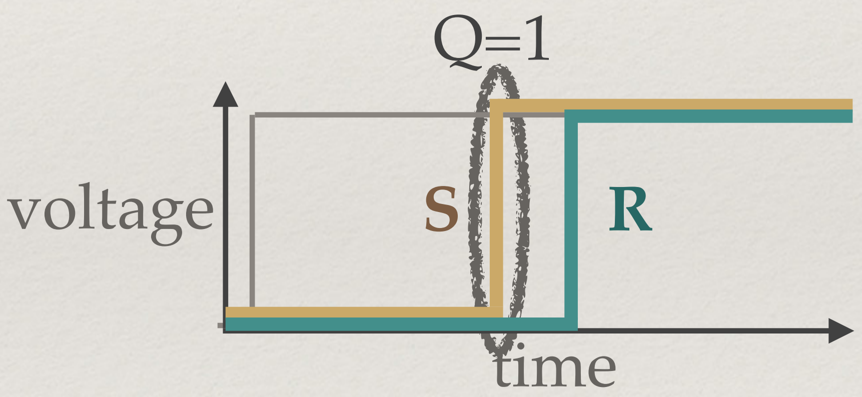


- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 2^n$ (n=1 shown)

Arbiter PUF

[D. Lim et al., ’05]

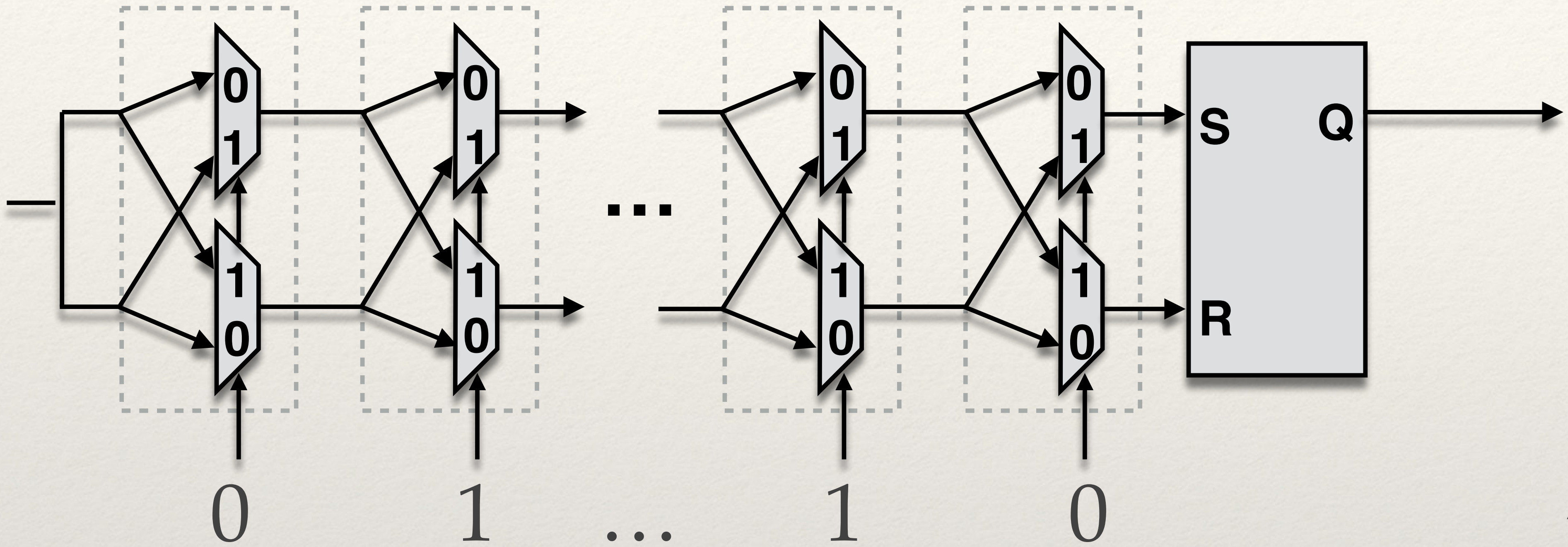


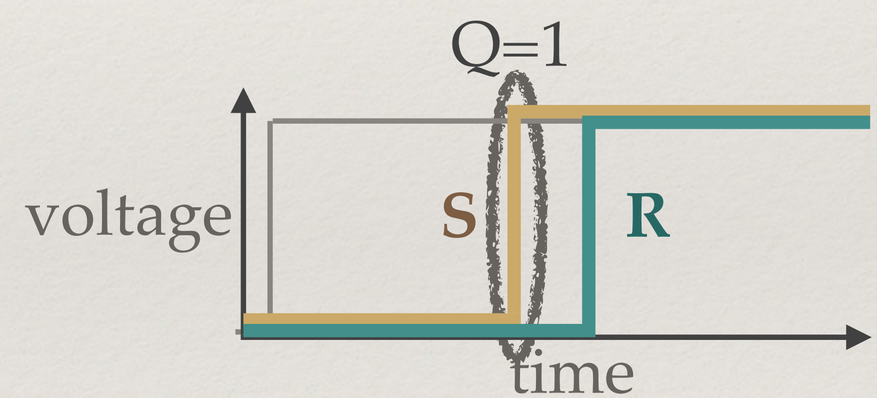


- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 2^n$ (n=1 shown)

Arbiter PUF

[D. Lim et al., ’05]

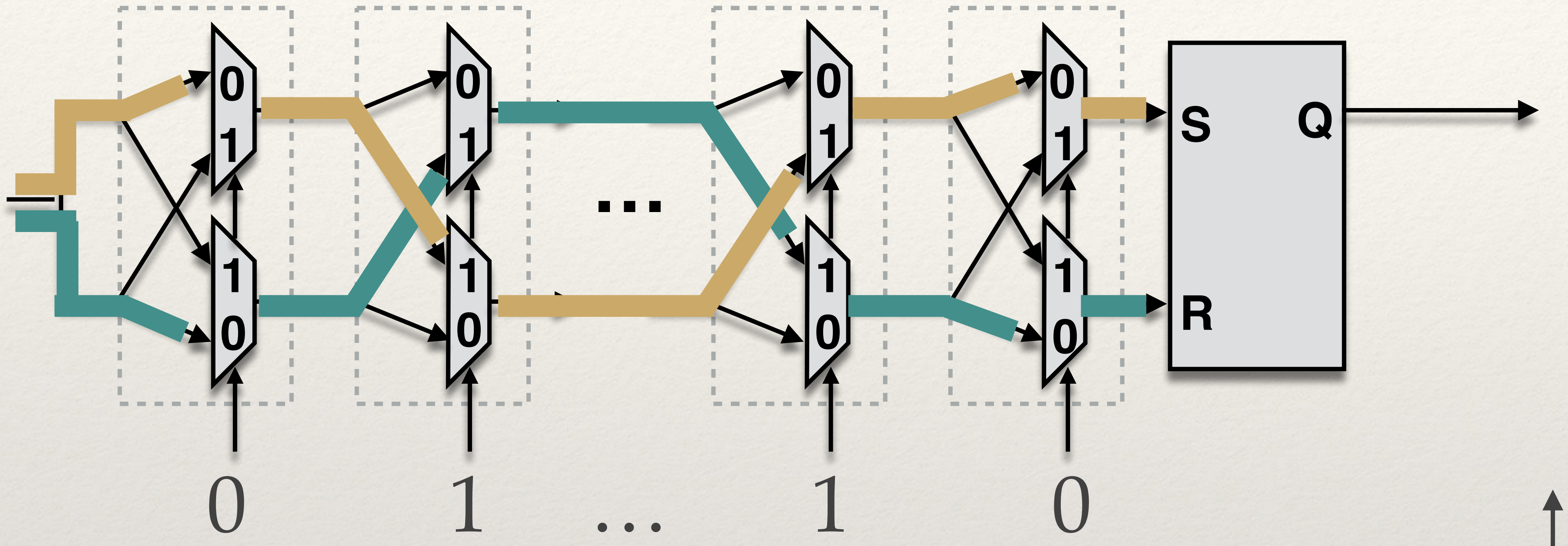


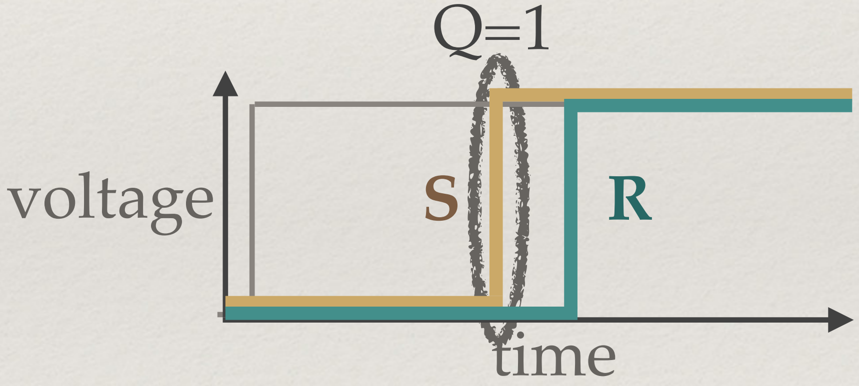


- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 2^n$ (n=1 shown)

Arbiter PUF

[D. Lim et al., ’05]

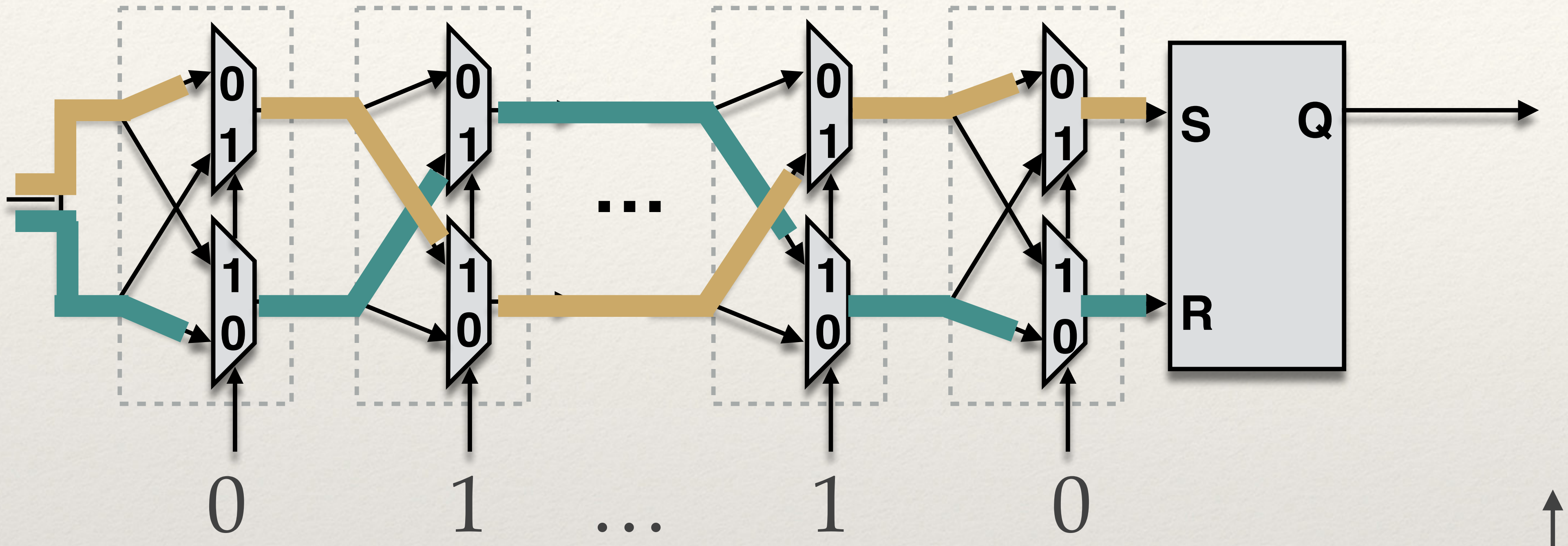


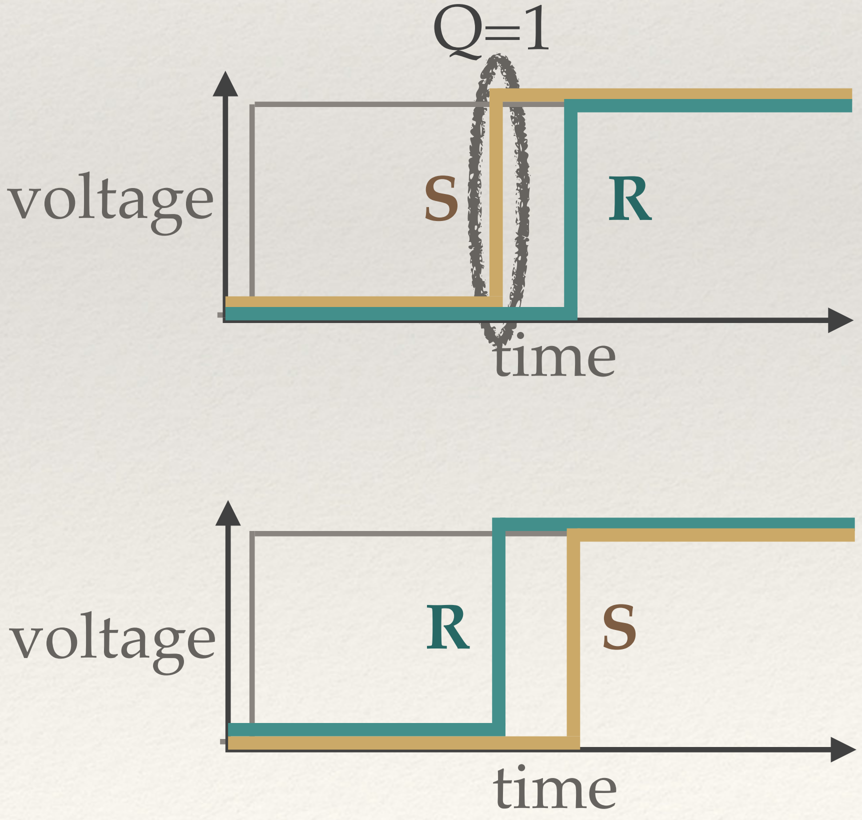


- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 2^n$ (n=1 shown)

Arbiter PUF

[D. Lim et al., '05]

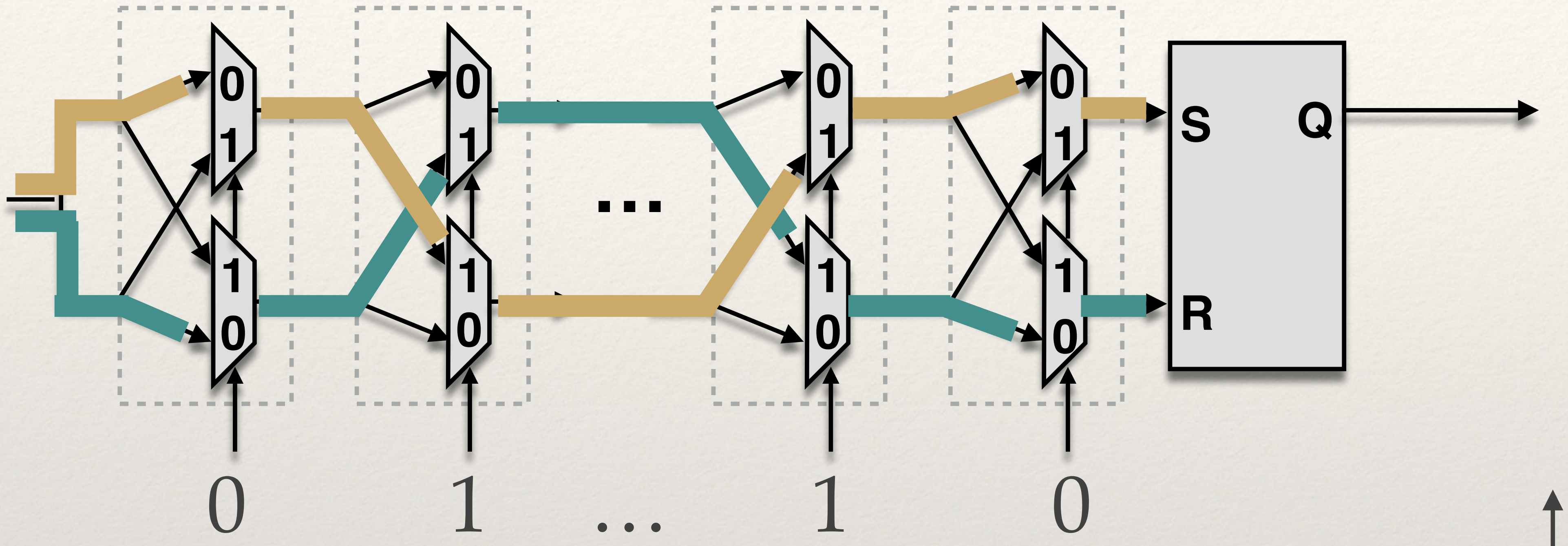




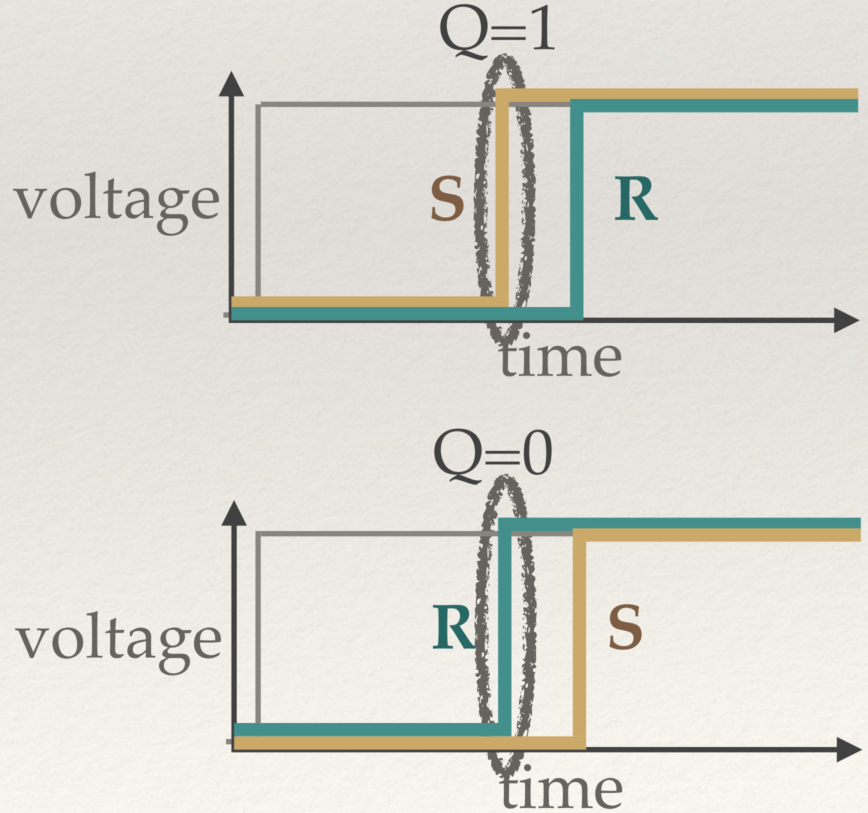
- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 2^n$ (n=1 shown)

Arbiter PUF

[D. Lim et al., '05]

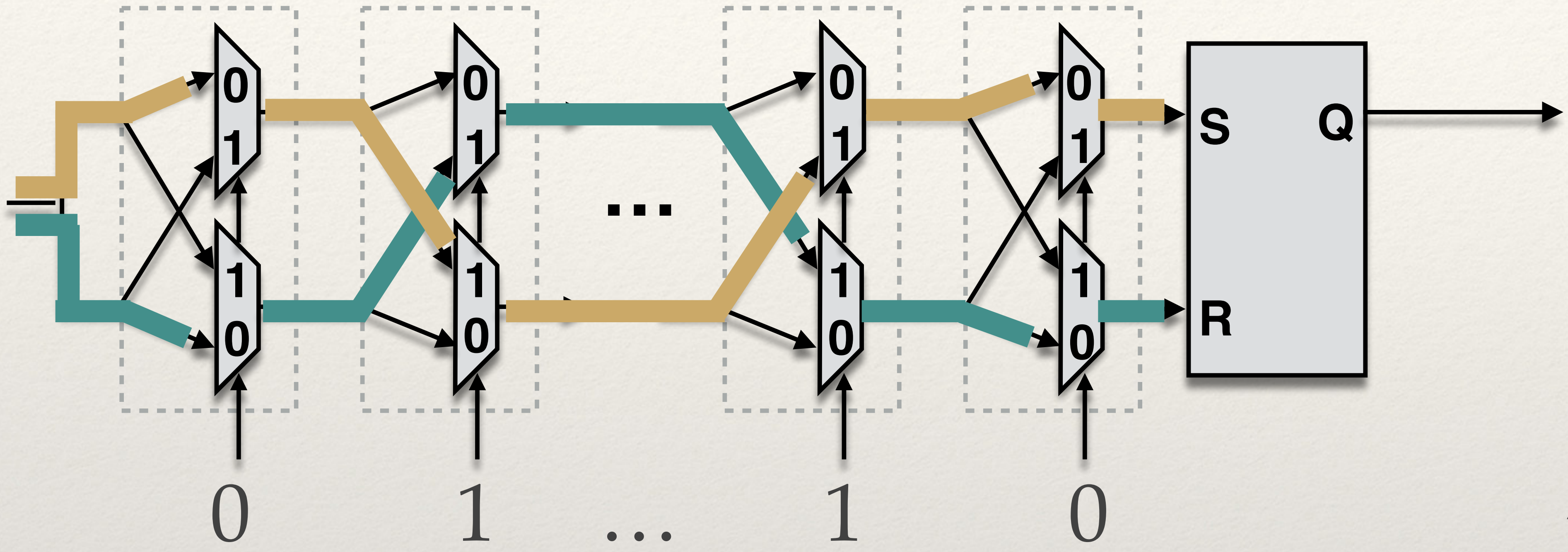


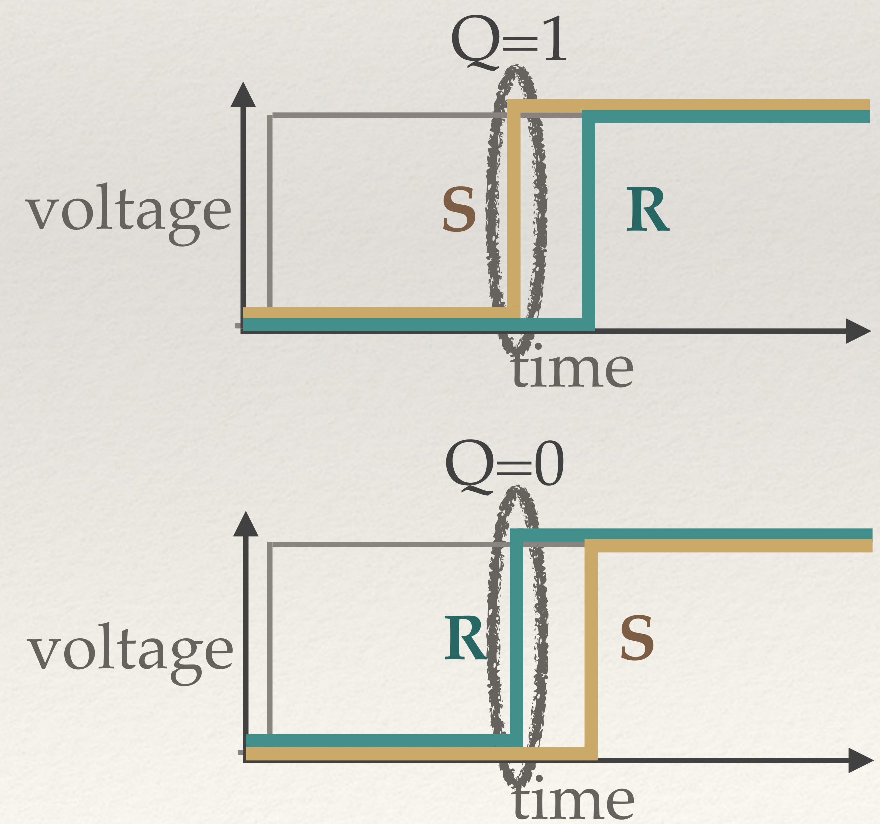
- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 2^n$ (n=1 shown)



Arbiter PUF

[D. Lim et al., '05]

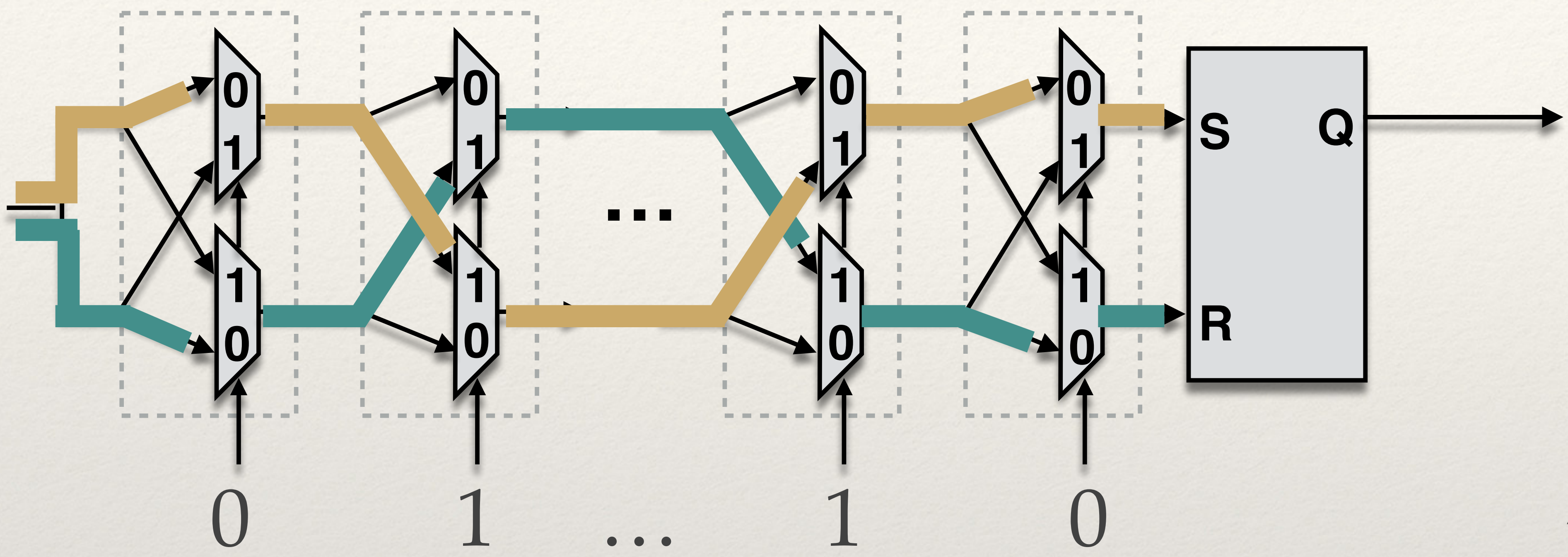


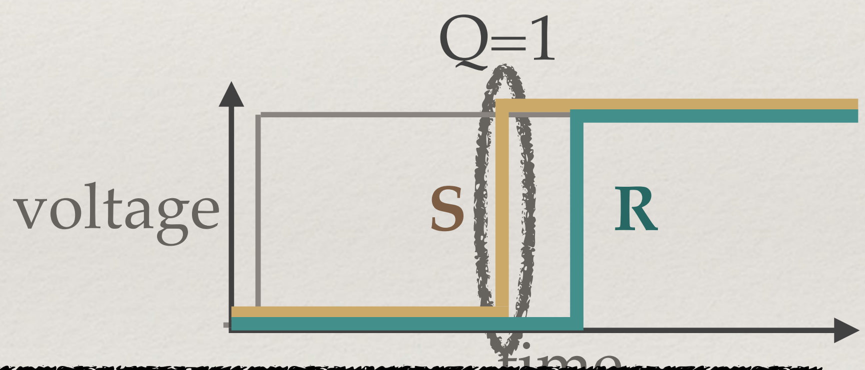


- Challenges: $c_i \in 2^m$ (m= num stages)
- Responses: $r_i \in 2^n$ (n=1 shown)
- Disorder / randomness: Delays in the subcomponents

Arbiter PUF

[D. Lim et al., ’05]





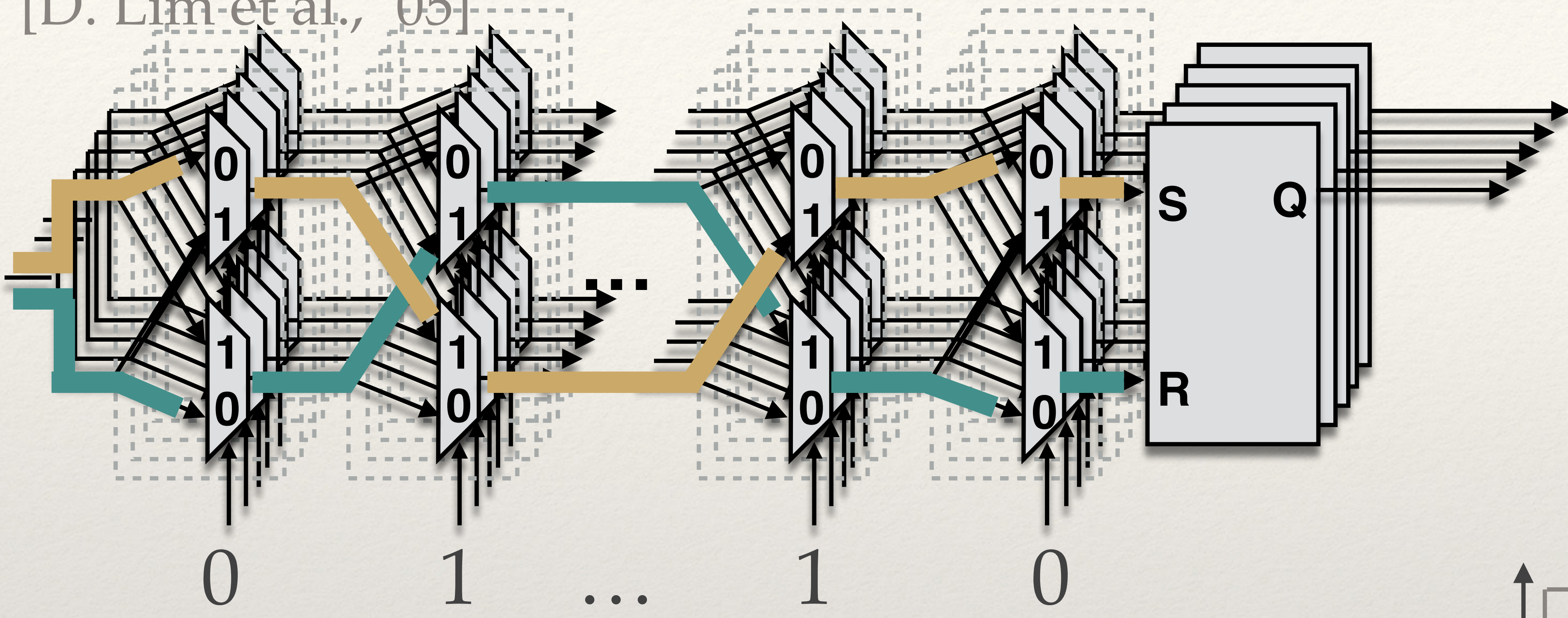
- Challenges: $c_i \in 2^m$ (m= num stages)

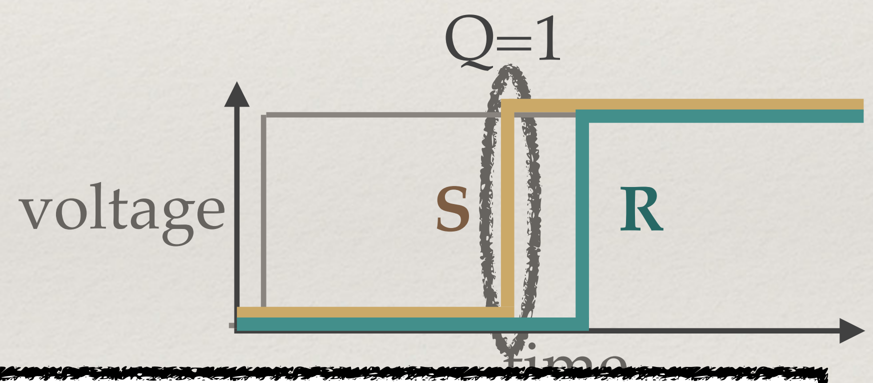
- Assumes that model cannot be created by observing CRPs
- But basic arbiter PUF susceptible to additive delay model

time

Arbiter PUF

[D. Lim et al., '05]



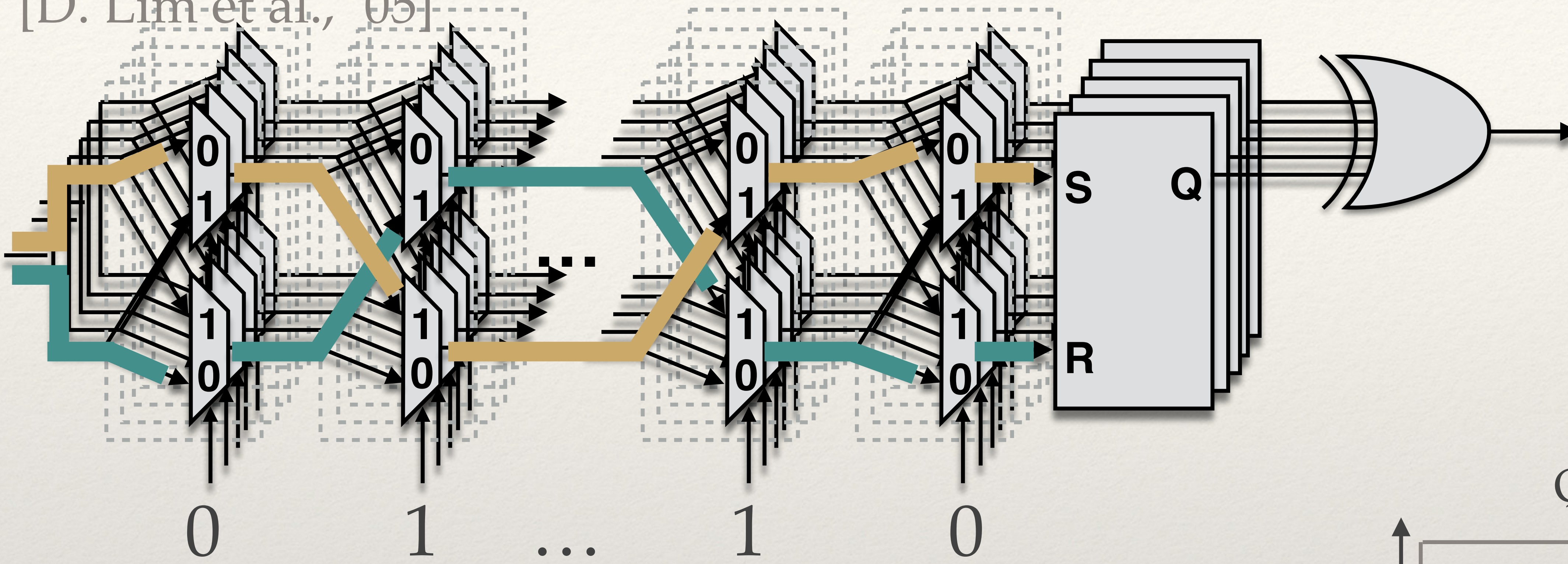


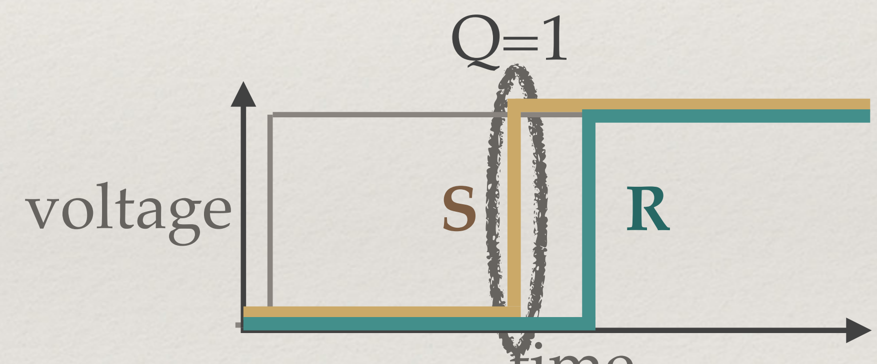
- Challenges: $c_i \in 2^m$ (m= num stages)

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

Arbiter PUF

[D. Lim et al., '05]



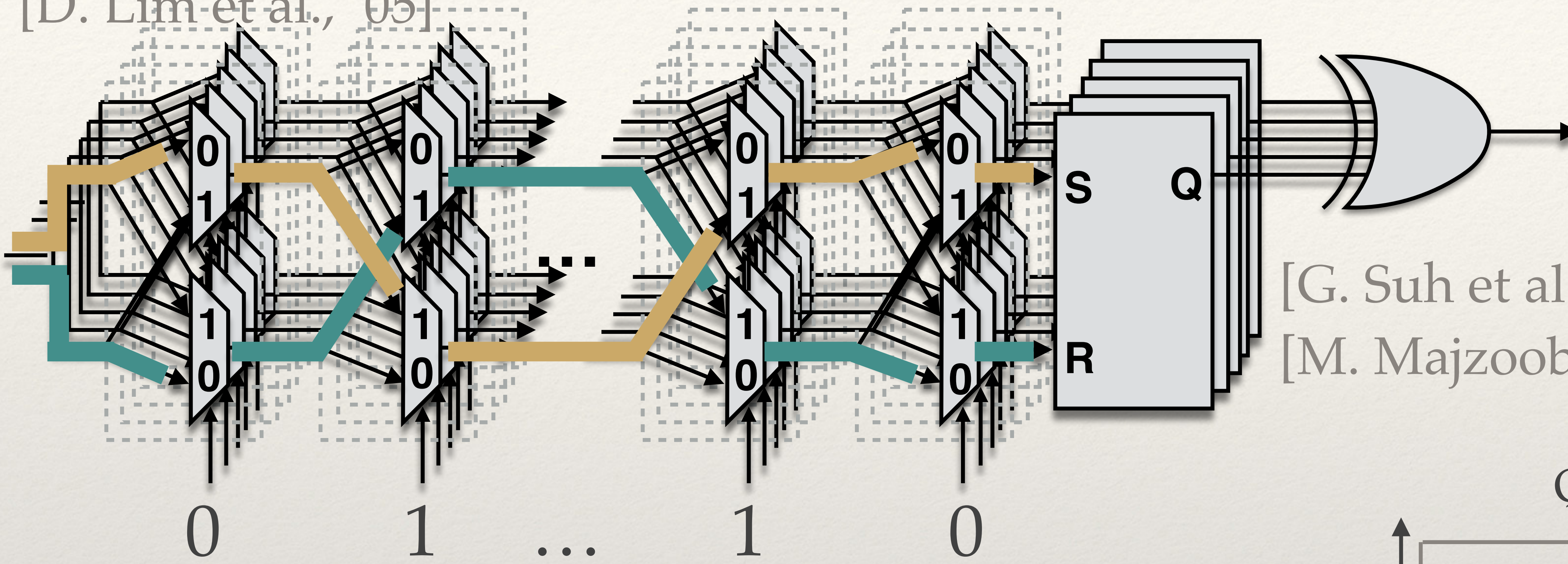


- Challenges: $c_i \in 2^m$ (m= num stages)

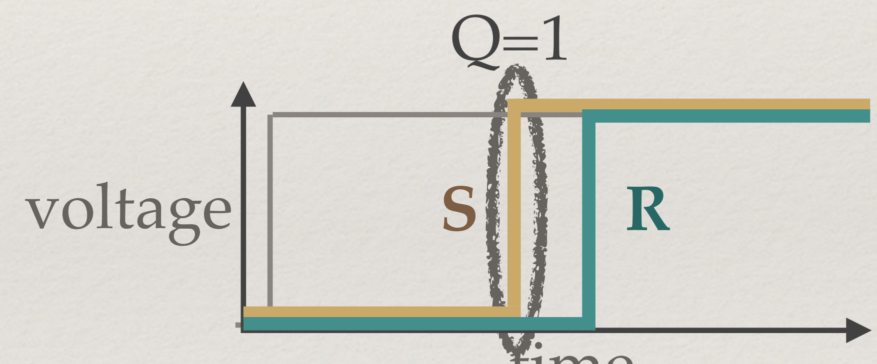
- Assumes that model cannot be created by observing CRPs
- But basic arbiter PUF susceptible to additive delay model

Arbiter PUF

[D. Lim et al., '05]



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

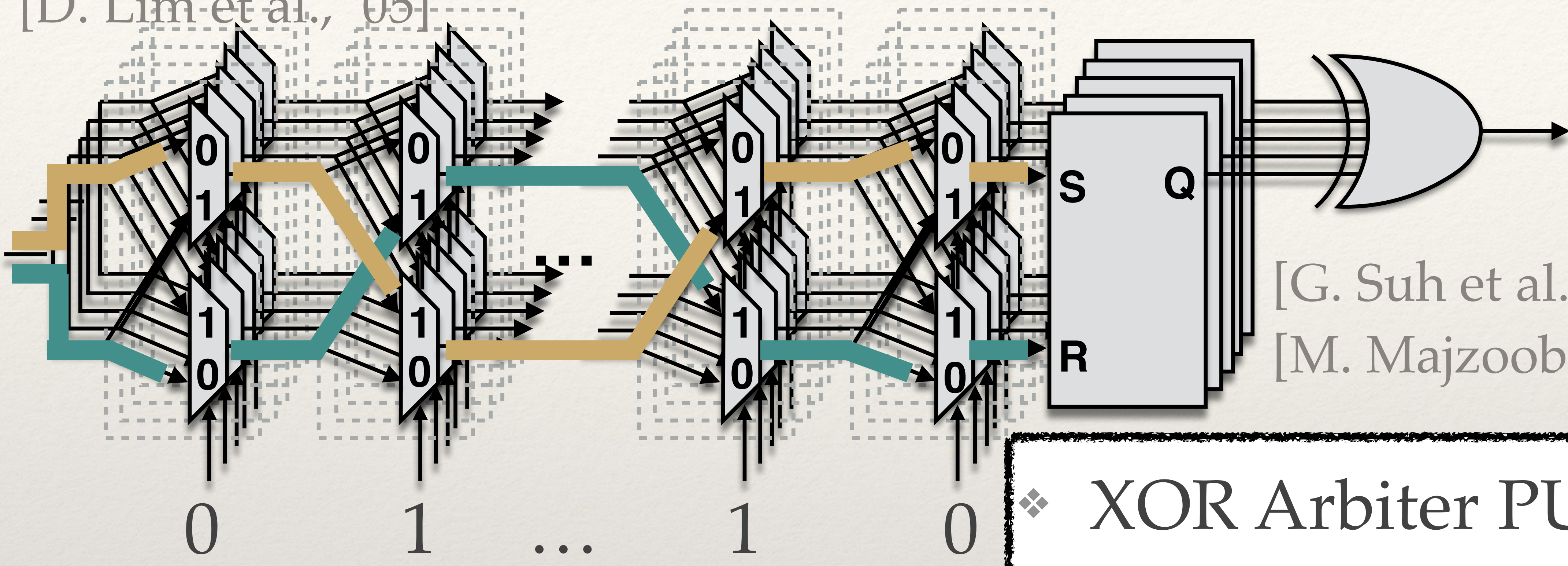


- Challenges: $c_i \in 2^m$ (m= num stages)

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

Arbiter PUF

[D. Lim et al., ’05]



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

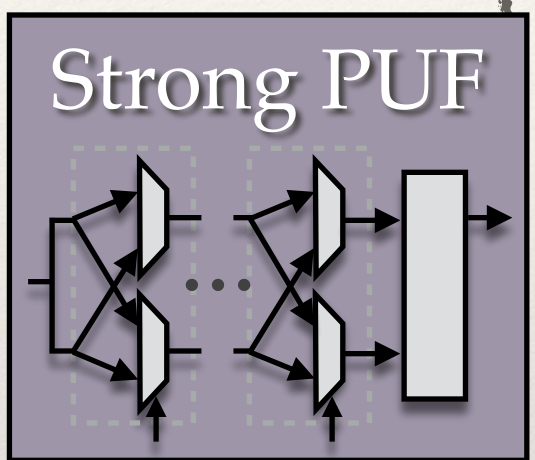
- XOR Arbiter PUF resists additive model

- Challenges: $c_i \in 2^m$ (m= num stages)

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

Authentication using Strong PUF

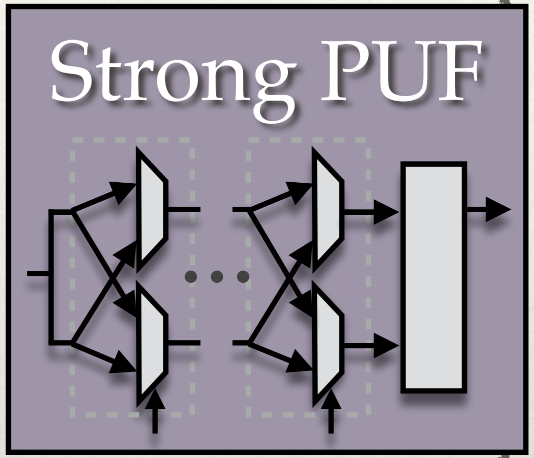
Enroll PUF



Authentication using Strong PUF

Enroll PUF

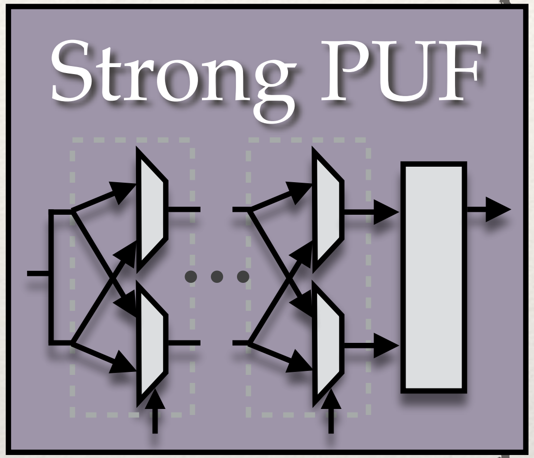
- Choose random challenges



Authentication using Strong PUF

Enroll PUF

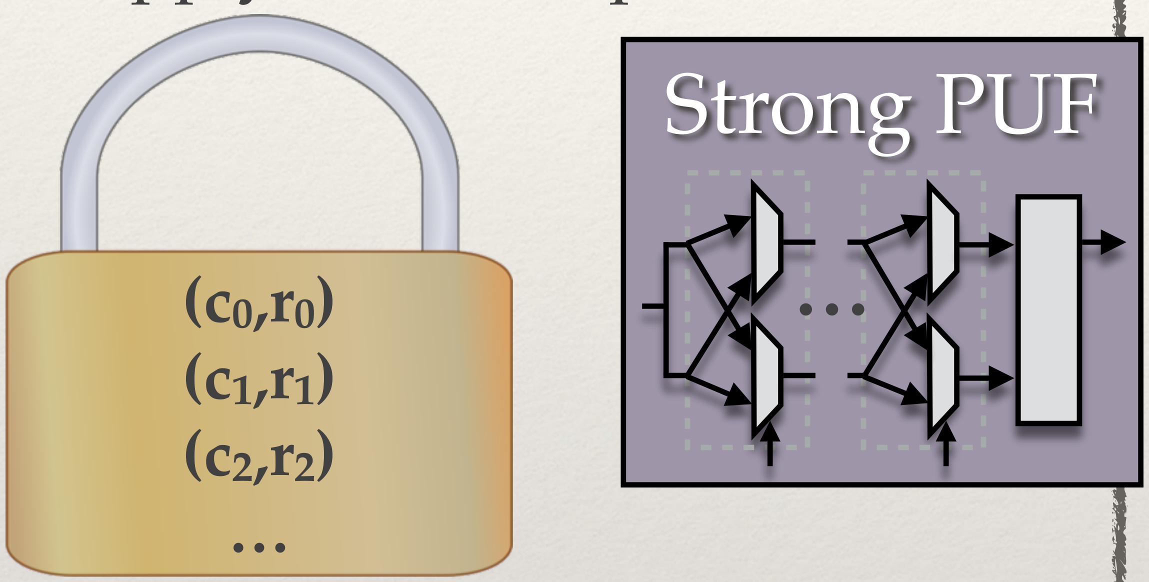
- Choose random challenges
- Apply and store private CRPs



Authentication using Strong PUF

Enroll PUF

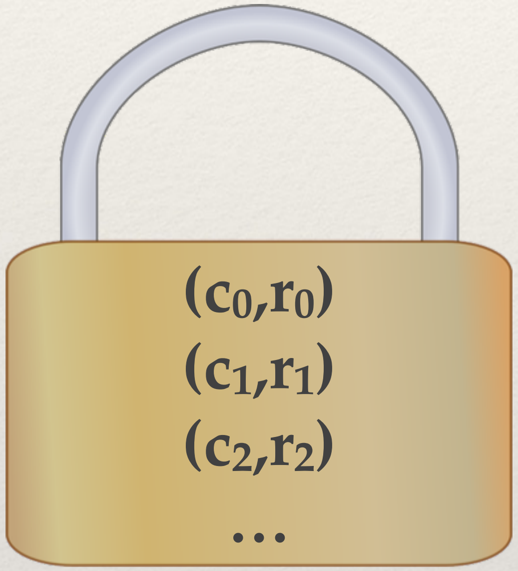
- Choose random challenges
- Apply and store private CRPs



Authentication using Strong PUF

Enroll PUF

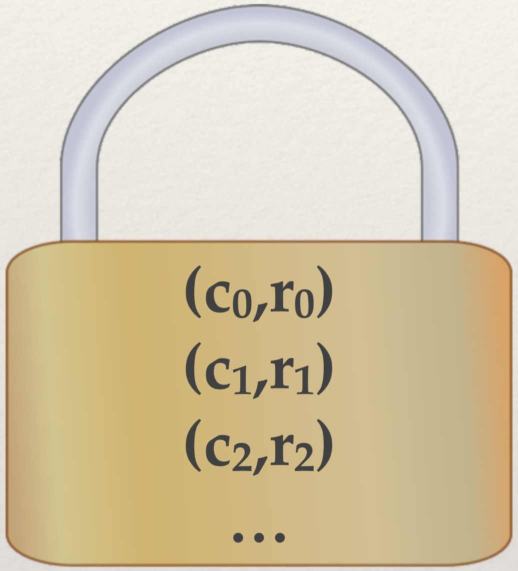
- Choose random challenges
- Apply and store private CRPs

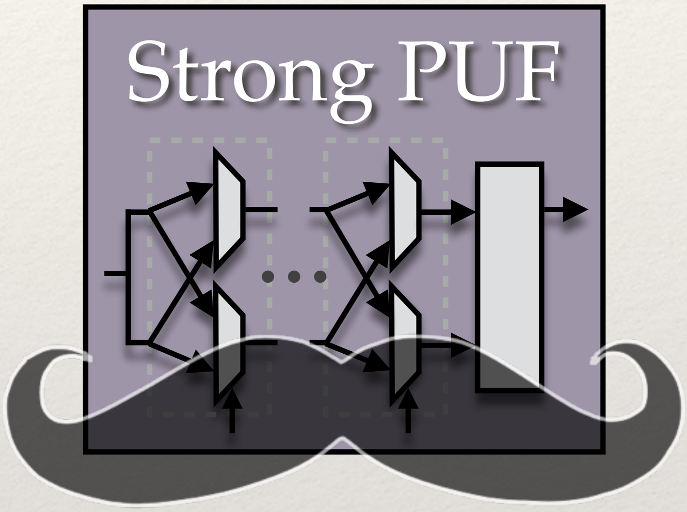


Authentication using Strong PUF

Enroll PUF

- Choose random challenges
- Apply and store private CRPs

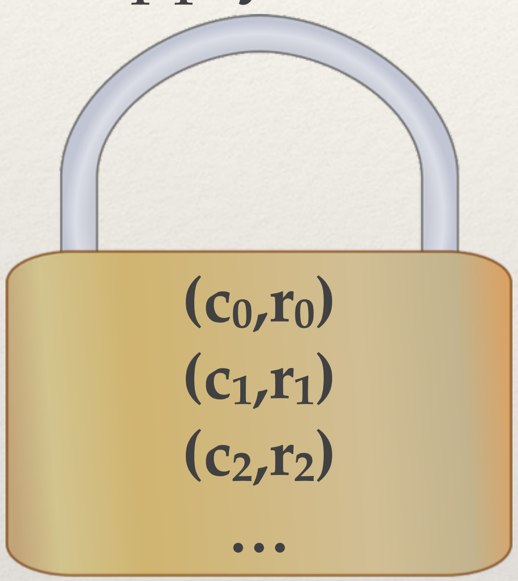




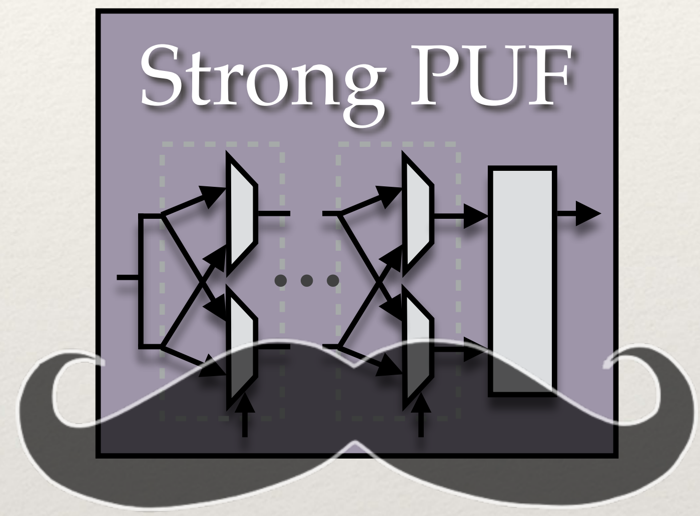
Authentication using Strong PUF

Enroll PUF

- Choose random challenges
- Apply and store private CRPs



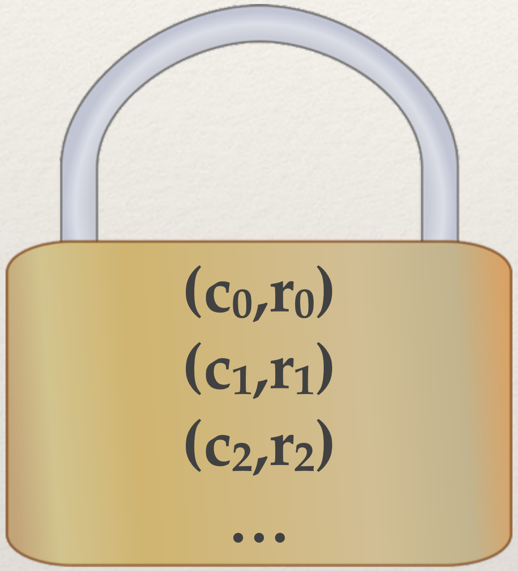
c_0

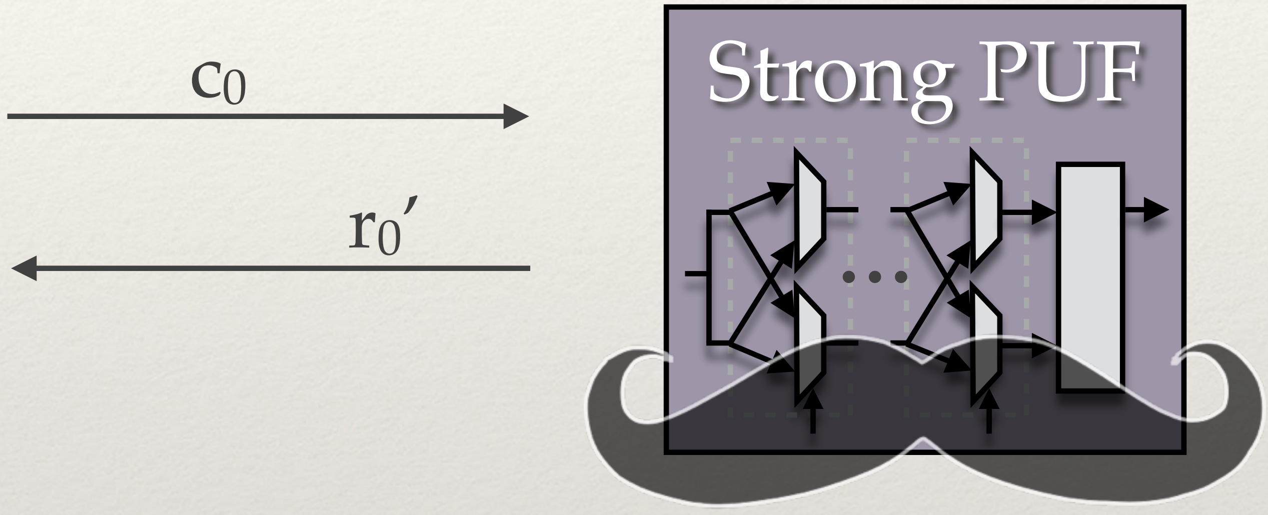


Authentication using Strong PUF

Enroll PUF

- Choose random challenges
- Apply and store private CRPs

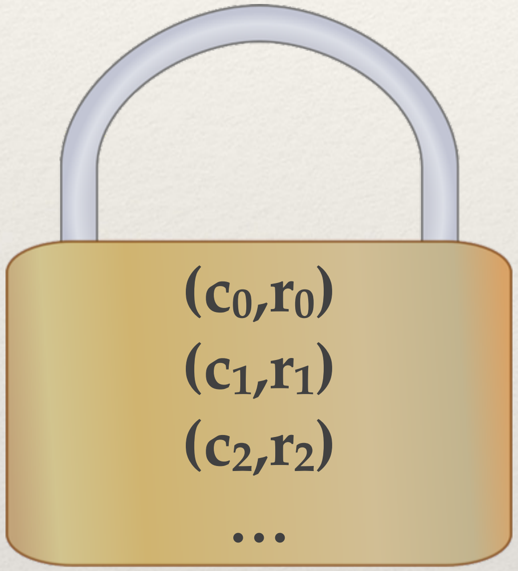




Authentication using Strong PUF

Enroll PUF

- Choose random challenges
- Apply and store private CRPs

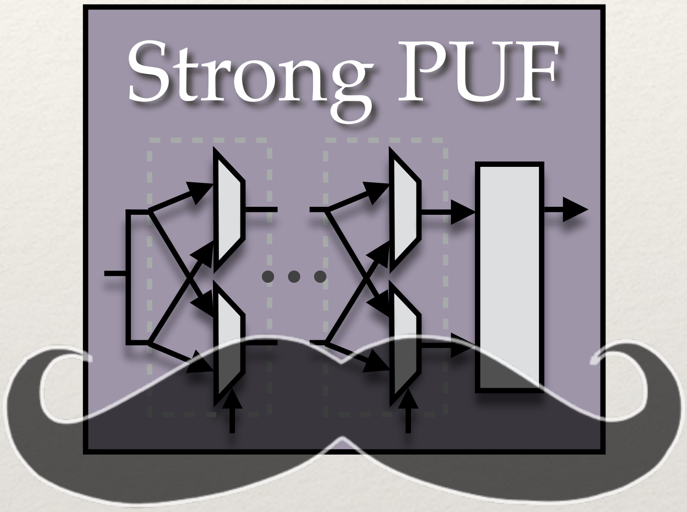


Authenticate

$r_0 \approx r_0'$?

c_0 →

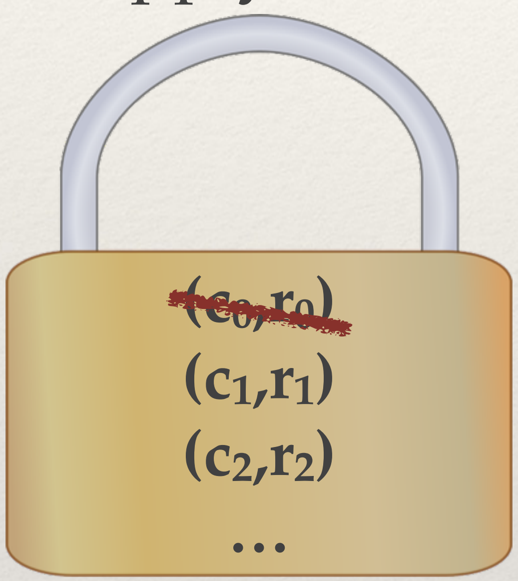
← r_0'



Authentication using Strong PUF

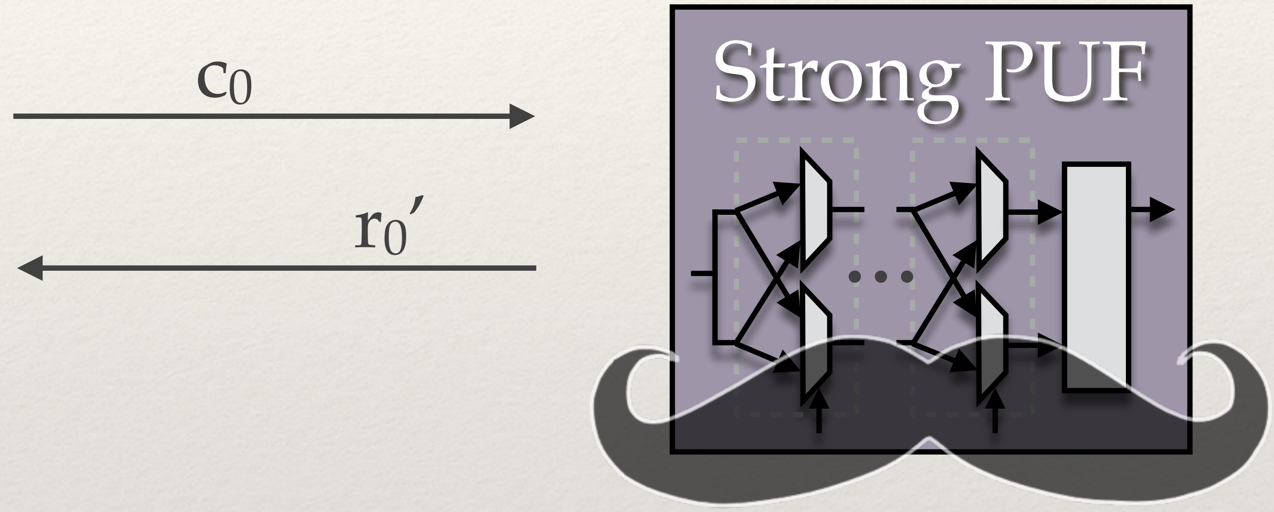
Enroll PUF

- Choose random challenges
- Apply and store private CRPs



Authenticate

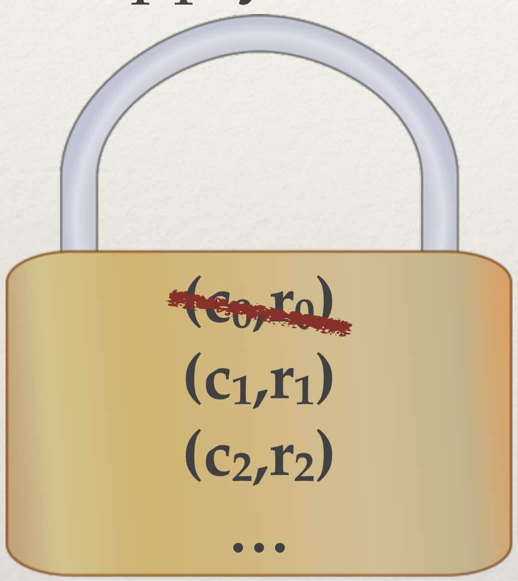
$r_0 \approx r_0'$?



Authentication using Strong PUF

Enroll PUF

- Choose random challenges
- Apply and store private CRPs

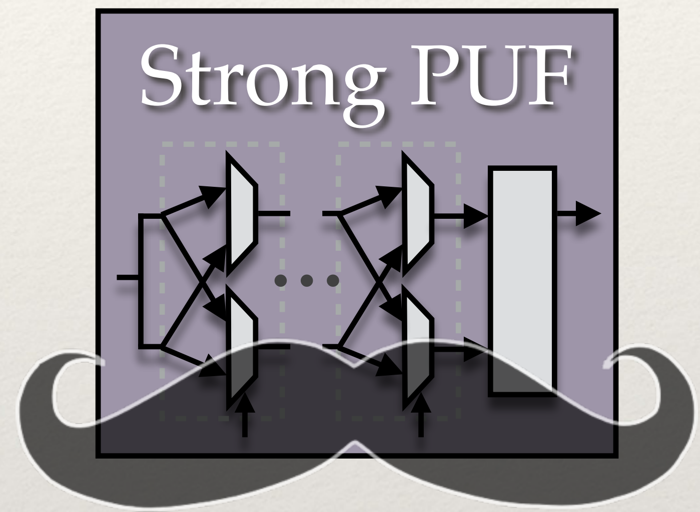


Authenticate

$r_0 \approx r_0'$?

c_0 →

← r_0'



- No need to hide responses if PUF cannot be modeled

Overview

1. Brief introduction to PUFs
2. Weak PUFs and applications
3. Strong PUFs and applications
4. **Conclusions**

Review

- PUFs are exciting new security primitive based on physical disorder
  - Desirable properties but also limitations
  - Arms race between designing and breaking

PReview

- PUFs are exciting new security primitive based on physical disorder

1. ~~PUFs at a Glance~~
2. Modeling attacks
3. Modeling attacks using side-channel information
4. Invasive attacks
5. Requirements for secure PUF protocols
6. Forward-looking trends and challenges