RUHR-UNIVERSITÄT BOCHUM

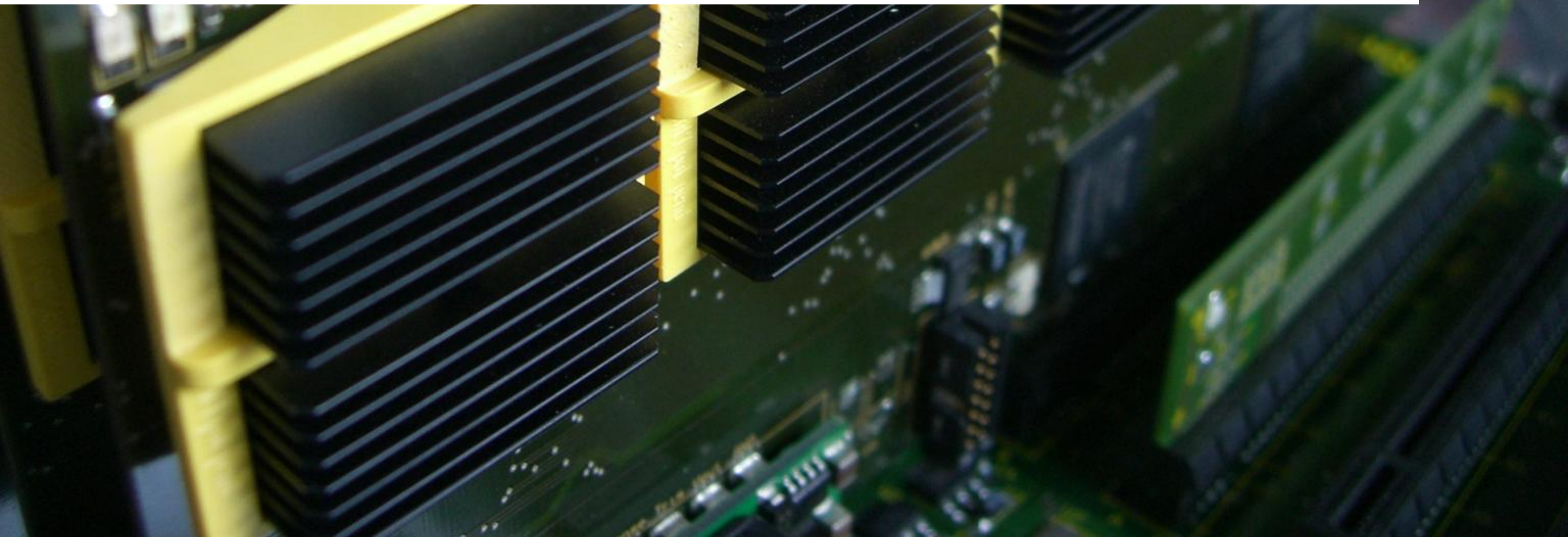
RUB

Generic Side-Channel Countermeasures for Reconfigurable Devices

CHES 2011, Nara, Japan

Tim Güneysu, Amir Moradi
Horst Görtz Institute for IT-Security, Ruhr-University Bochum, Germany

29/09/2011



Agenda

- Introduction and Motivation
- Design Proposals for FPGAs
    - Noise Generation
    - Clock Disalignment
    - Memory Masking
- Evaluations
- Conclusions

Agenda

- **Introduction and Motivation**
- Design Proposals for FPGAs
  - Noise Generation
  - Clock Disalignment
  - Memory Masking
- Evaluations
- Conclusions

Introduction and Motivation

- **All cryptographic implementations need countermeasures (CM) against side-channel attacks**

- **Designing and deploying a CM on a dedicated platform is costly**
  - Development time (e.g., balanced routing for logic styles)
  - Execution time (e.g., additional time for random dummy cycles)
  - Physical resources (e.g., more logic for masked data paths/S-boxes)

- For strong protection, **several CMs need to be combined** (which are even often very cipher-dependant)

- **Ideally**: **Given a set of generic and efficient CMs to establish (basic) SCA protection on a specific processing platform**

Introduction and Motivation

- **This talk: proposing countermeasures for FPGAs**
    - Generically usable with most (symmetric) cryptosystems
    - Applicable to many (Xilinx) FPGA devices
    - Predesigned as (hard) macros just to be added to an application

- **Portfolio of countermeasures:**
    - FPGA-specific noise generators (using registers, memories, short circuits)
    - Clock disalignment using Digital Clock Managers (DCM)
    - Memory masking in dual-ported memories
    - Detector for input clock manipulations (prevent down-clocking)

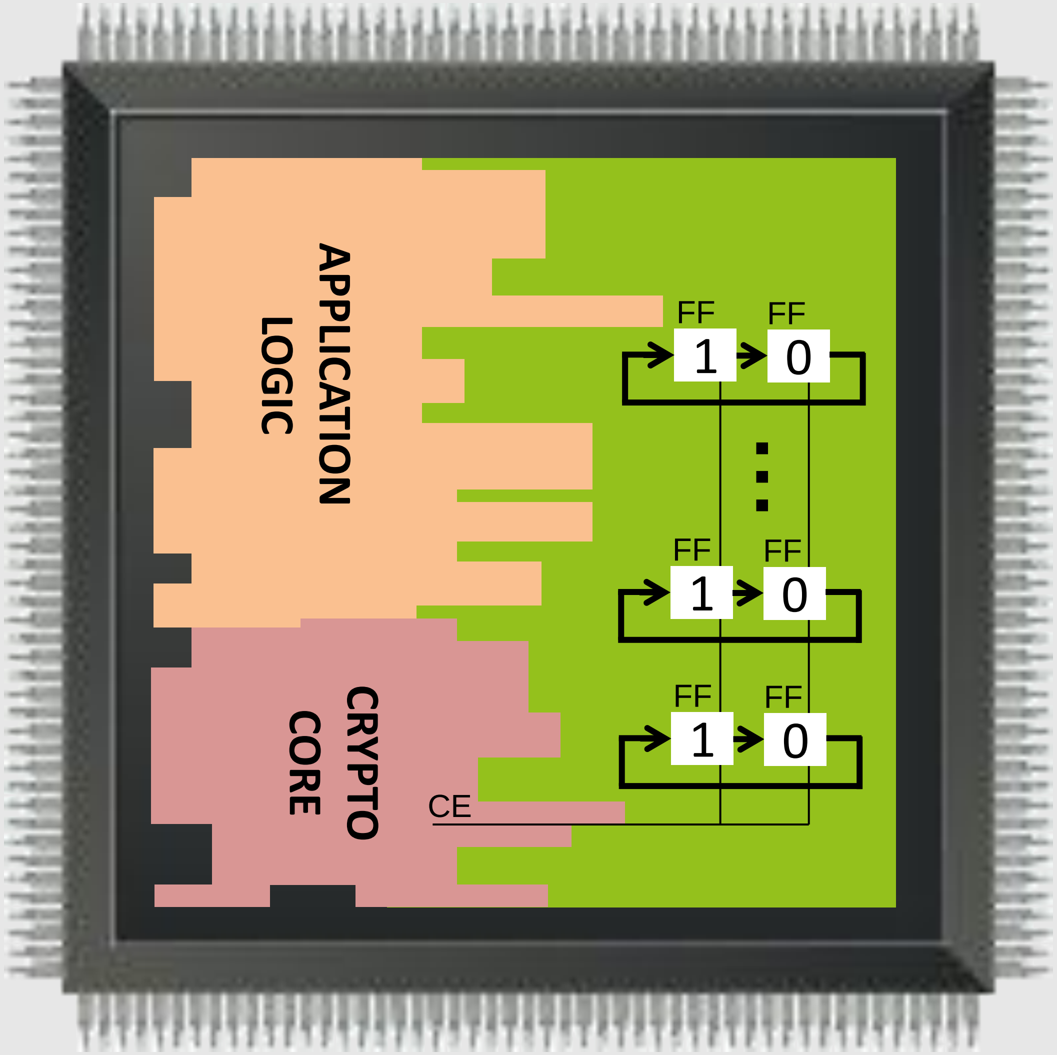
Agenda

- Introduction and Motivation
- **Design Proposals for FPGAs**
  - **Noise Generation**
  - Clock Disalignment
  - Memory Masking
- Evaluations
- Conclusions

Implementing Noise Generators in FPGAs

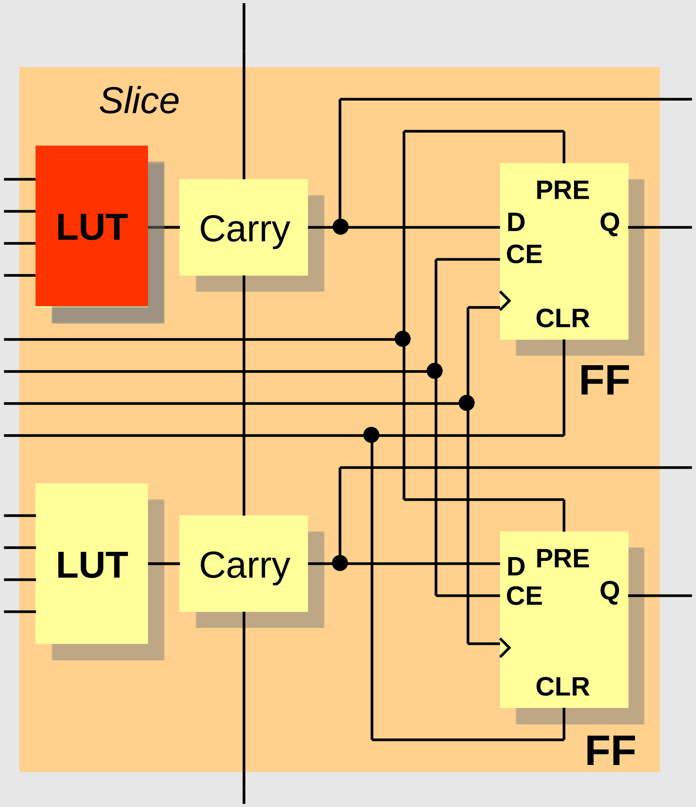
- **Common design**: application including cryptographic core

- **Noise generation strategy**
  - Configure remaining, routable slices (flip-flops) as cyclic shift registers
  - Preload sequence „01“ into shift registers
  - Run noise generator in synch with crypto core



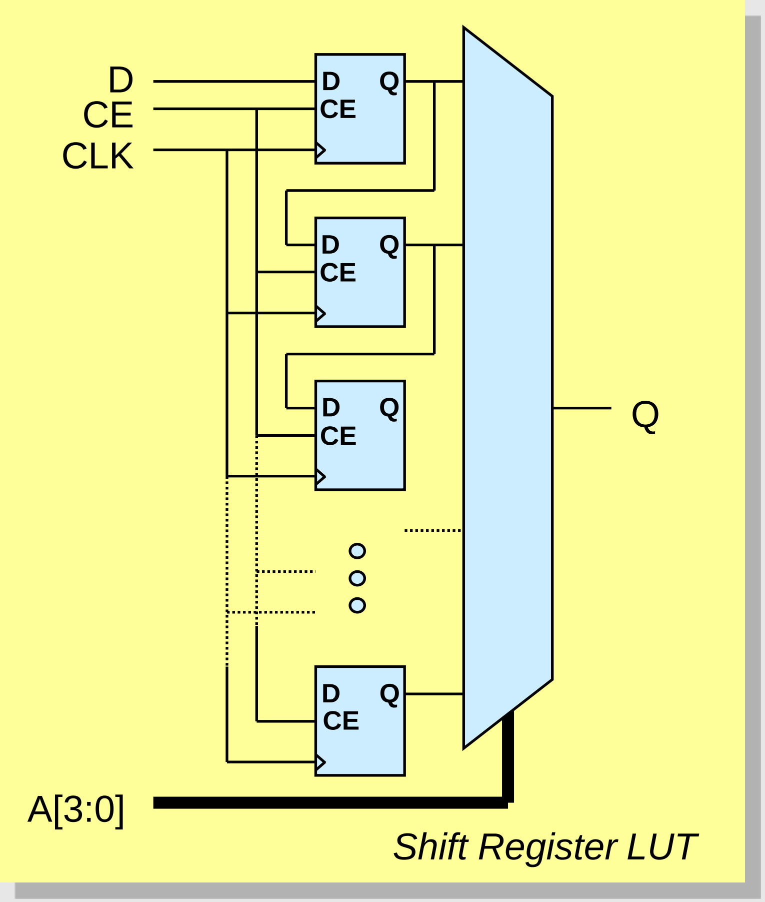
Proposal #1: Using Shift Register LUTs (SRL)

- Logic elements consist of LUTs and FFs
- Special (alternative) LUT function:
  Shift Register LUT (SRL)
  - *n*-bit register length (*n=16* or *n=64*)



Proposal #1: Using Shift Register LUTs (SRL)

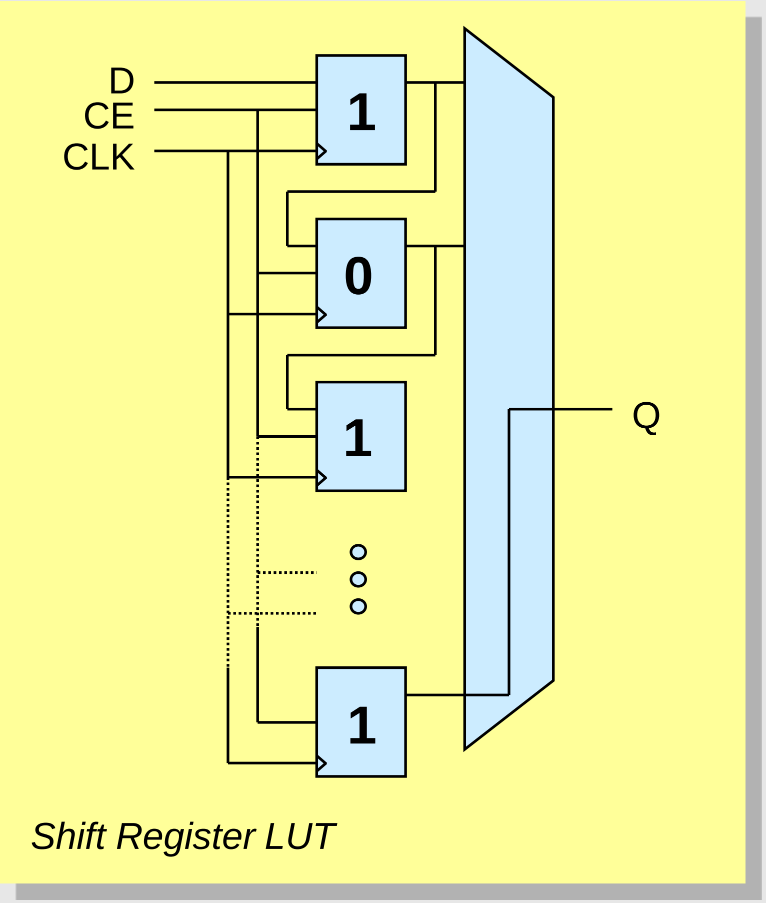
- Logic elements consist of LUTs and FFs
- Special (alternative) LUT function:
  Shift Register LUT (SRL)
  - *n*-bit register length (*n=16* or *n=64*)



Shift Register LUT

Proposal #1: Using Shift Register LUTs (SRL)

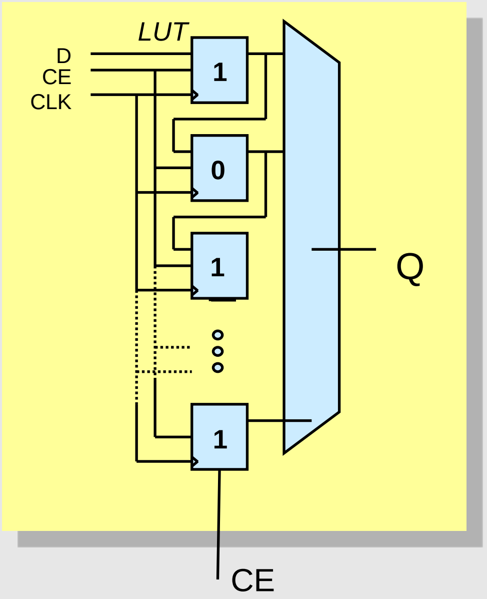
- Logic elements consist of LUTs and FFs
- Special (alternative) LUT function:
  Shift Register LUT (SRL)
  - *n*-bit register length (*n=16* or *n=64*)
  - Preload SRL with „01“ combination

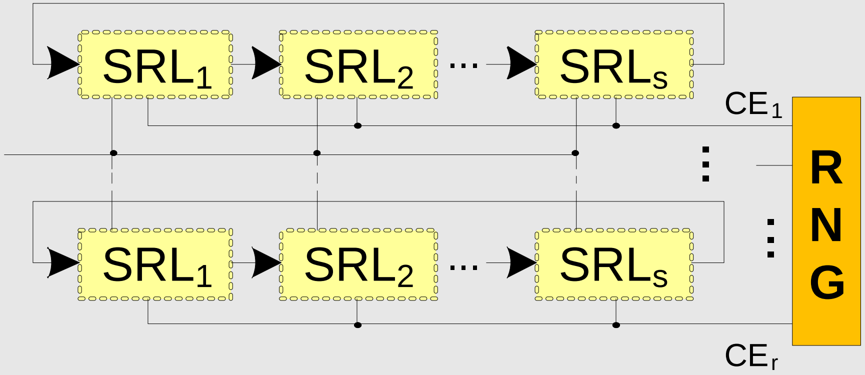


Shift Register LUT

Proposal #1: Using Shift Register LUTs (SRL)

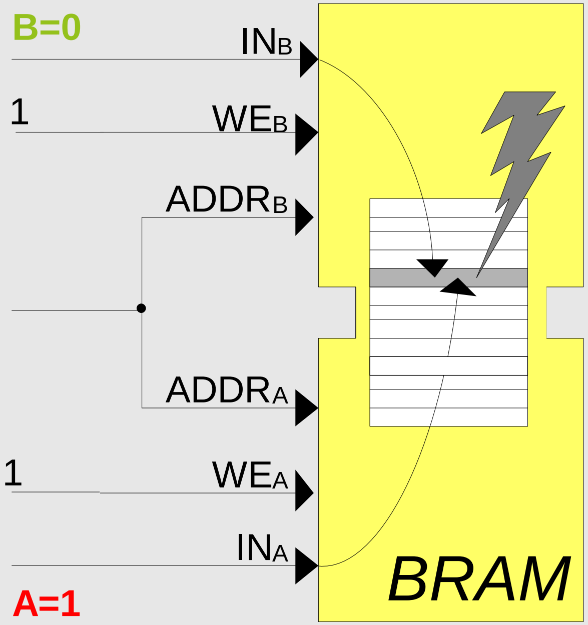
- Logic elements consist of LUTs and FFs
- Special (alternative) LUT function:
  Shift Register LUT (SRL)
  - *n*-bit register length (*n=16* or *n=64*)
  - Preload SRL with „01“ combination
- Create *r* cyclic rings using *s* cascaded SRLs
- SRLs are clocked according to free-running RNG

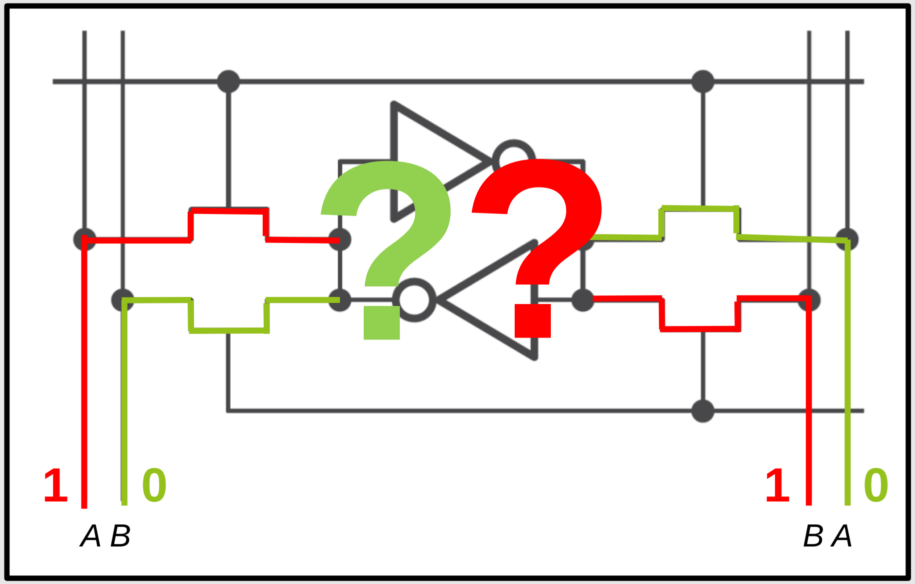




Proposal #2: Write Collisions in BRAMs

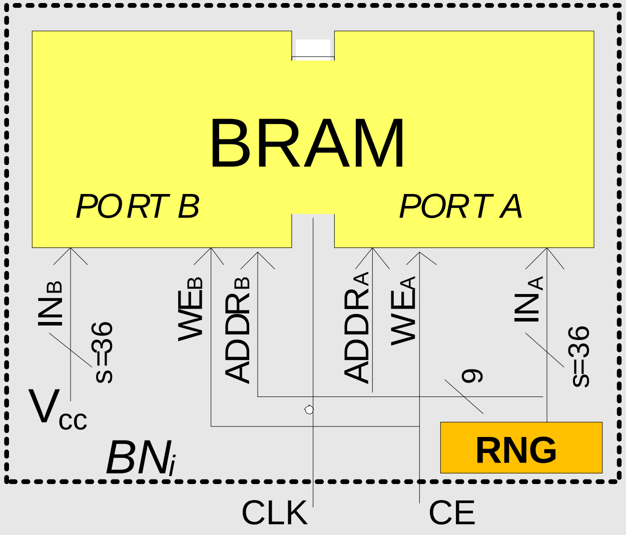
- Write collision when concurrently writing data to the same address of dual-ported memories (BRAM)
- Opposite driving directions in inverter pair result in uncertain outcome [GP09,G10]

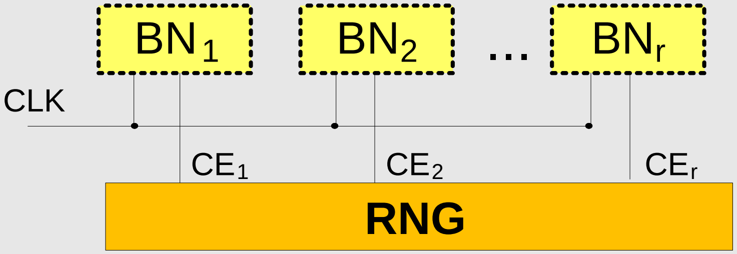




Proposal #2: Write Collisions in BRAMs

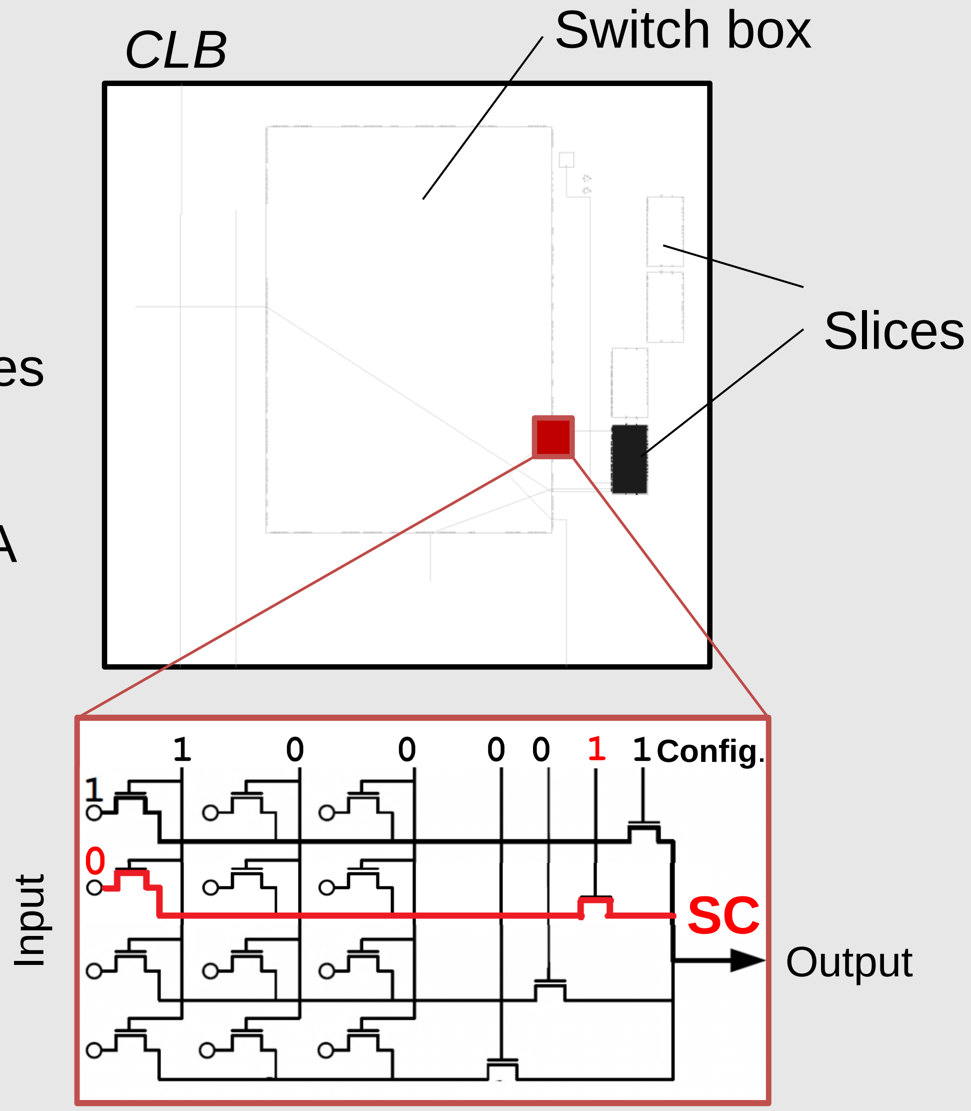
- Write collision when concurrently writing data to the same address of dual-ported memories (BRAM)
- Opposite driving directions in inverter pair result in uncertain outcome [GP09,G10]
- Likely to exhibit higher power consumption
- **Idea for noise generation:**
  - Create a write collision generator
  - Create collisions according to output of an RNG





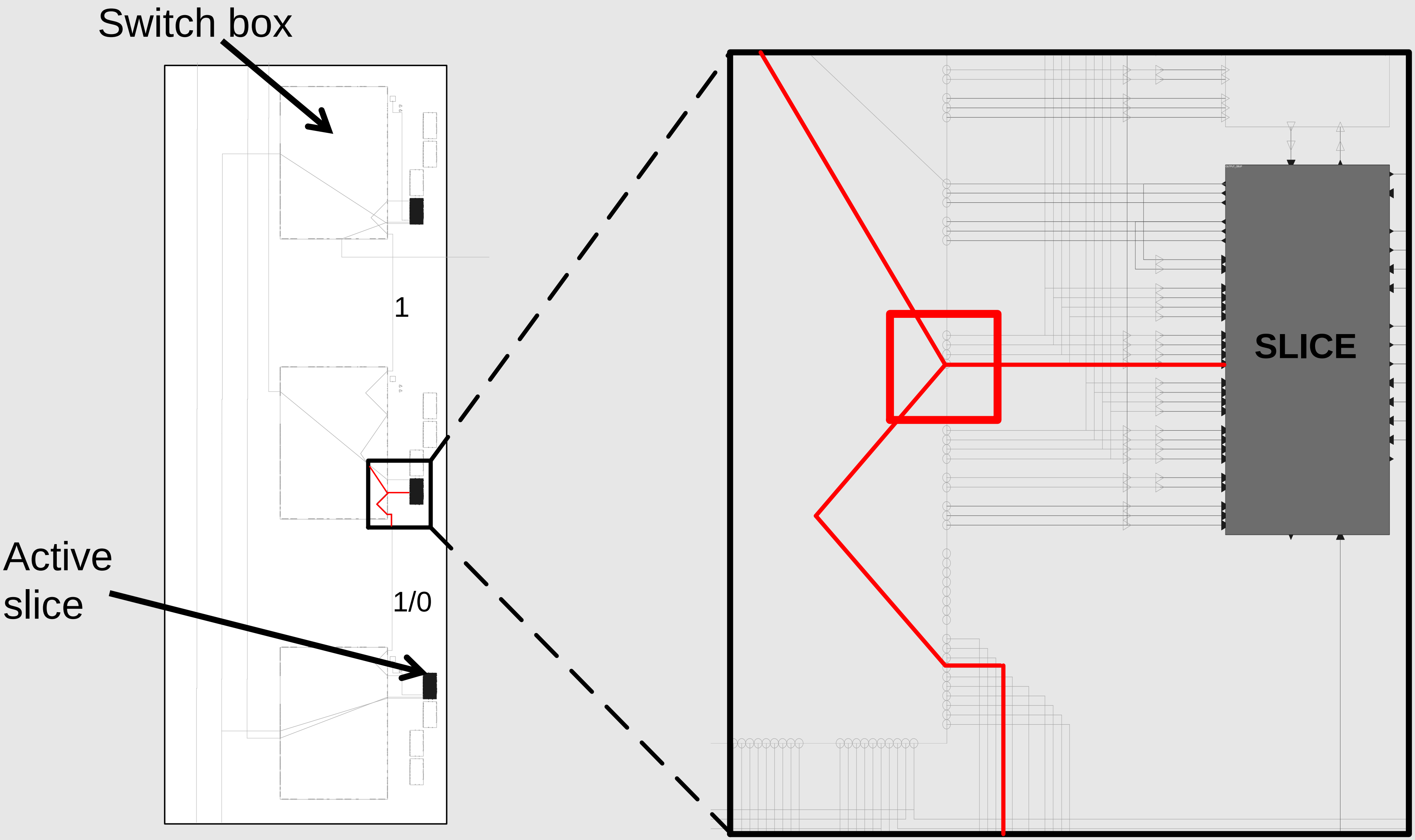
Proposal #3: Short Circuits in FPGAs

- Short circuits (SC) can be created in the FPGA‘s routing network [BKT10]
- SCs in output multiplexers of switch boxes
- Power restriction limits currents < 100 µA
- Establishing controlled SCs requires manual routing (via XDL)



Output multiplexer

Proposal #3: Short Circuits in FPGAs

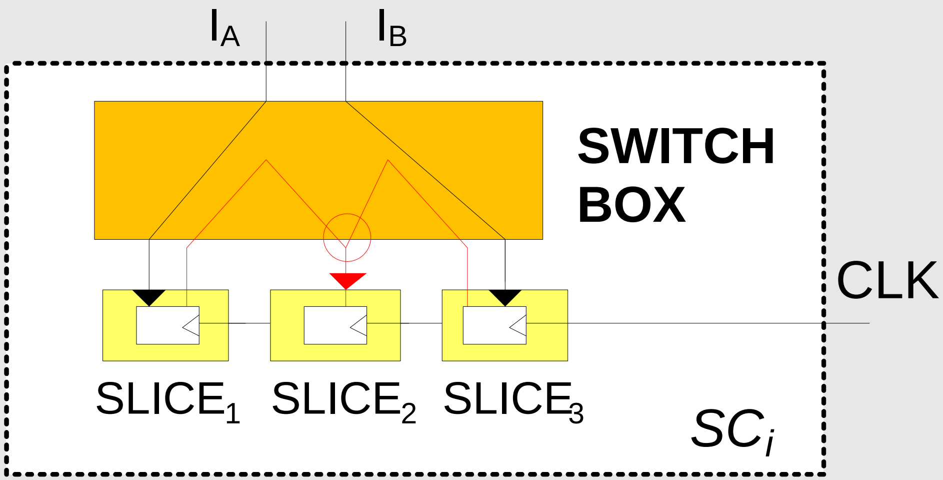


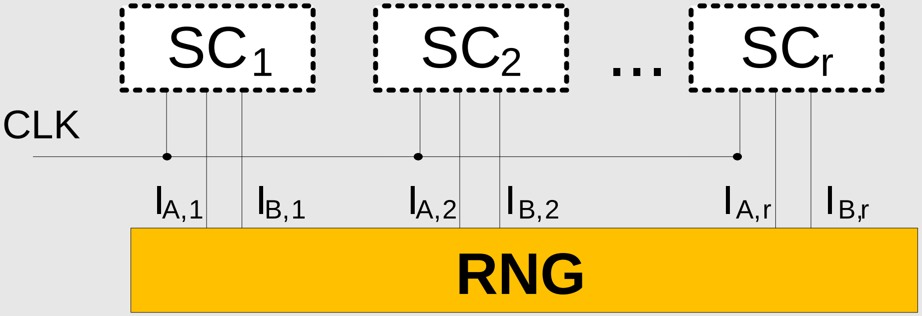
Controlled SC with 3 CLBs

Design of a Controlled Short Circuit

Proposal #3: Short Circuits in FPGAs

- Package controlled SC into hard macro
- Instantiate *r* controlled SC units on FPGA
- Distribute SCs among different power domains to distribute load



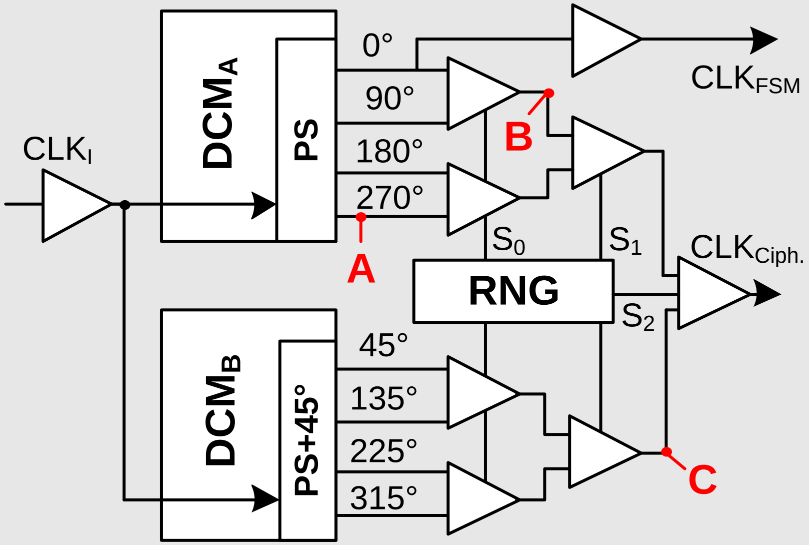


Agenda

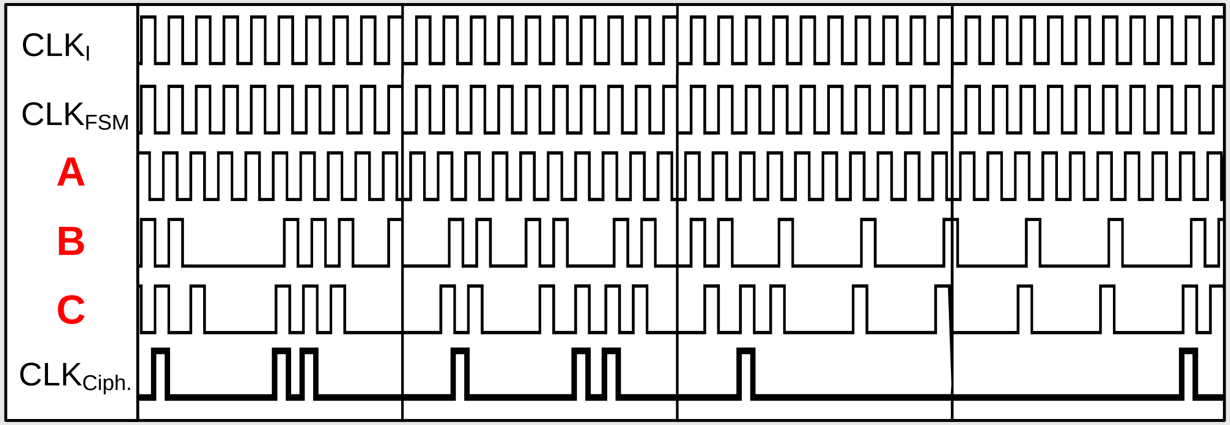
- Introduction and Motivation
- **Design Proposals for FPGAs**
  - Noise Generation
  - **Clock Disalignment**
  - Memory Masking
- Evaluations
- Conclusions

Proposal #4: Clock Disalignment using DCMs

- Digital Clock Managers (DCM) support concurrent phase-shift channels
- Clock buffers can be configured as glitch-free clock multiplexers
- Cascading clock muxes result in a randomly delayed, phase-shifted clock



Clock Output Waveform

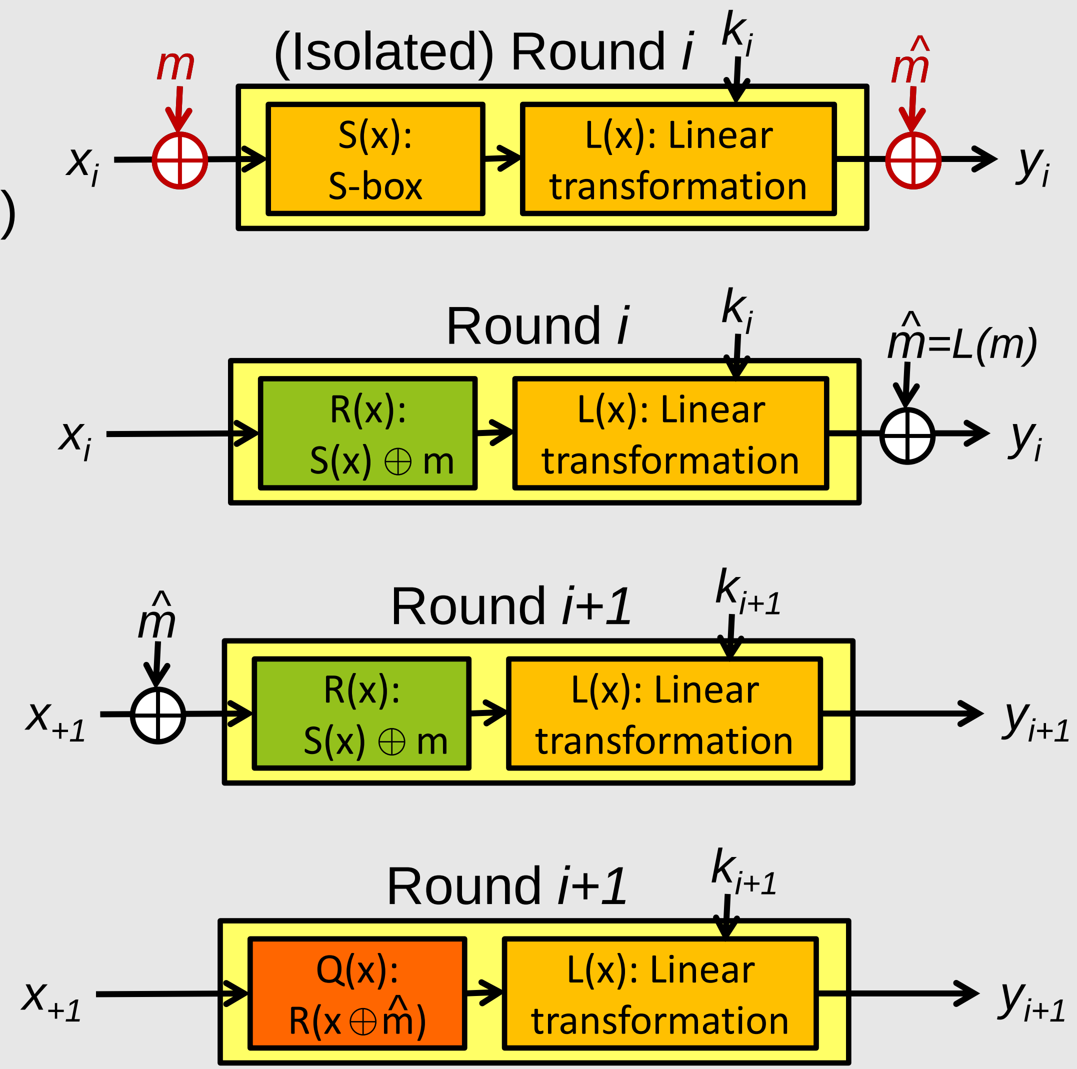


Agenda

- Introduction and Motivation
- **Design Proposals for FPGAs**
  - Noise Generation
  - Clock Disalignment
  - **Memory Masking**
- Evaluations
- Conclusions

Proposal #5: Data Masking with BRAMs

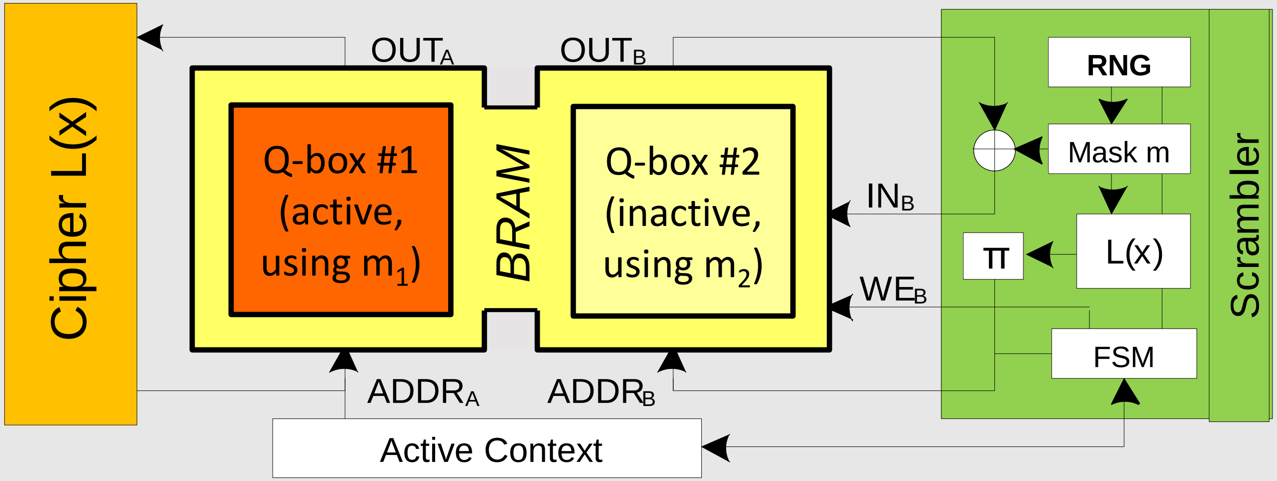
- Round functions often have linear and non-linear part (S-box in memory)
- CM: implement masking on data path
- **Implementation idea:**
  - Push masking scheme into dual-ported memory (S-box)
  - Perform mask update by concurrent process
- **Simplification**: use same random mask for (few) consecutive rounds ➔ (first-order SCA-resistant only!)



➔ **S-box im memory: *Q(x) = S(x⊕L(m))⊕m***

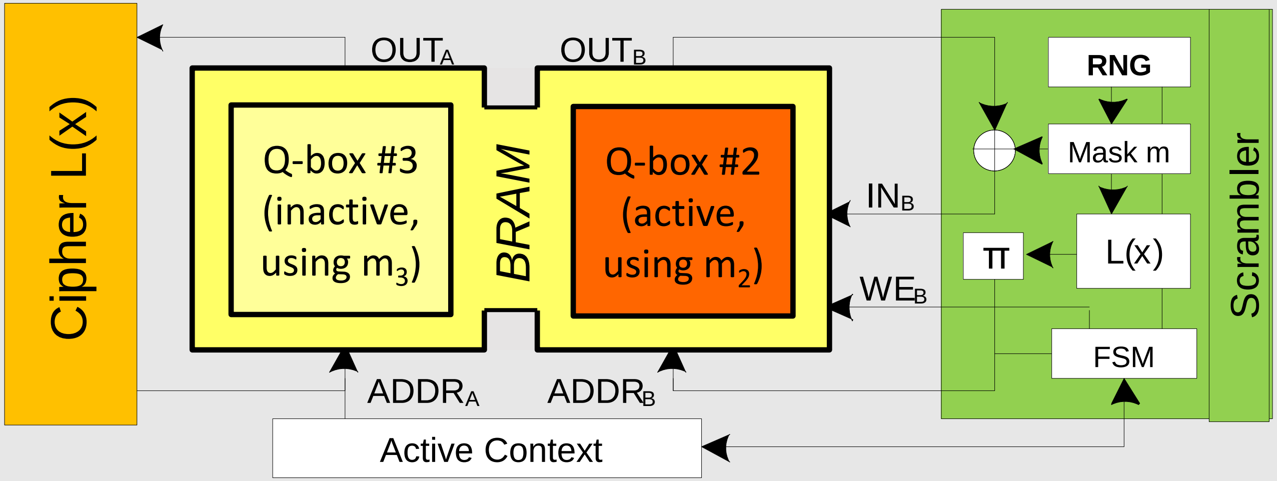
Proposal #5: Data Masking with BRAMs

- Dual-ported BRAM allows simultaneous use and mask update of Q-box
  - Active context (Q-box #1) used by cipher operation
  - Inactive context (Q-box #2) updates mask by concurrent process
  - Context switch after update and cipher process are finished



Proposal #5: Data Masking with BRAMs

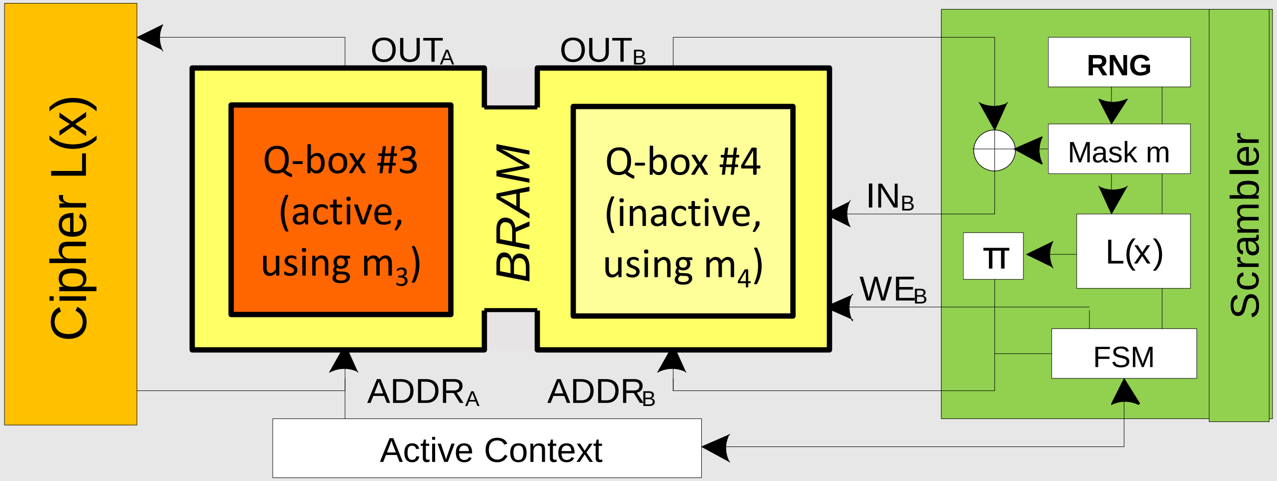
- Dual-ported BRAM allow simultaneous access and mask update in Q-box
  - Active context (Q-box #1) used by cipher operation
  - Inactive context (Q-box #2) updates mask by concurrent process
  - Context switch after update and cipher process are finished



Current Mask m:
F439AD0B8C…

Proposal #5: Data Masking with BRAMs

- Dual-ported BRAM allow simultaneous access and mask update in Q-box
  - Active context (Q-box #1) used by cipher operation
  - Inactive context (Q-box #2) updates mask by concurrent process
  - Context switch after update and cipher process are finished

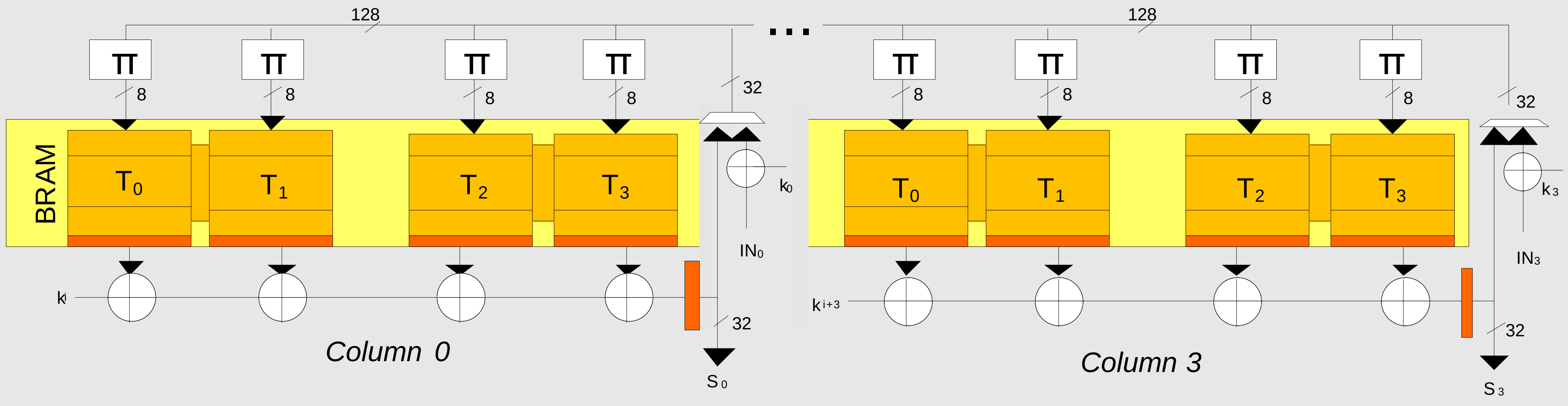


Current Mask m:
4E9A25C321…

Agenda

- Introduction and Motivation
- Design Proposals for FPGAs
  - Noise Generation
  - Clock Disalignment
  - Memory Masking
- **Evaluations**
- Conclusions

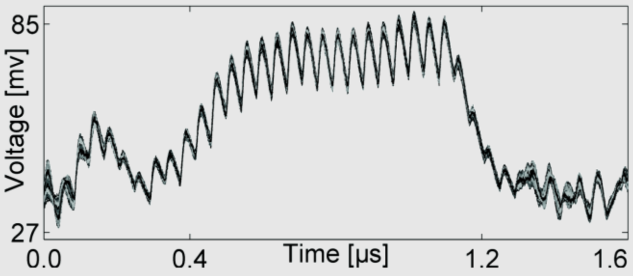
Evaluation based on AES T-Table Implementation

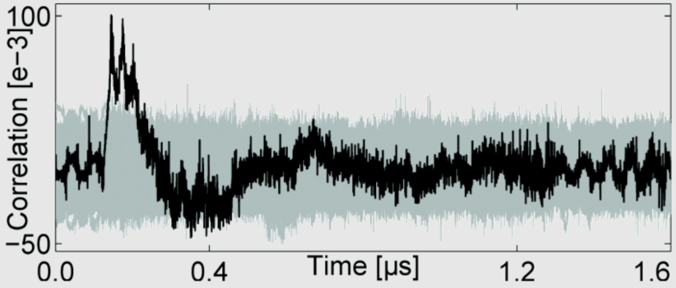


- AES-128 T-Table implementation/128-bit data path (16 T-Tables, 21 cycles)
- SASEBO board populated with Xilinx Virtex-II Pro FPGA (xc3vp7)
- Measuring setup: Diff. Probe at LeCroy WP715Zi 1.5 GHz@2GS/s
- Correlation Power Analysis (CPA) using Hamming Weight (HW) model

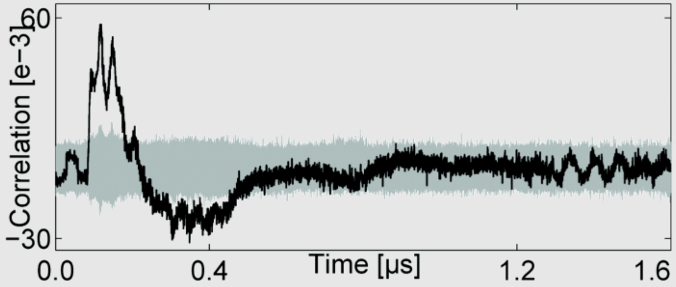
Evaluations: CPA on individual CMs

Plain AES-128@24Mh:
10^4 measurements
→ 3,000 traces req.

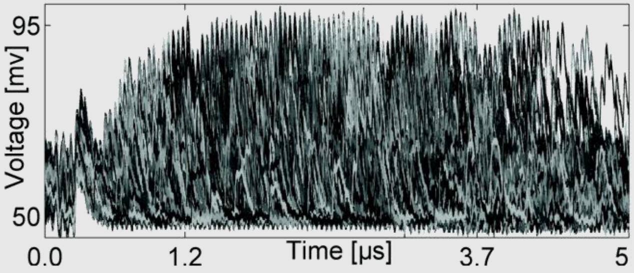


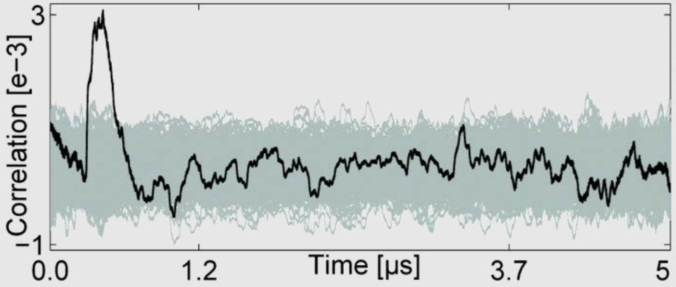


Individual/all noise generators combined:
Parameters used: *r=16 (instances), s=36 (width)*
$5x10^4$ measurements
→ 8,000 traces req.

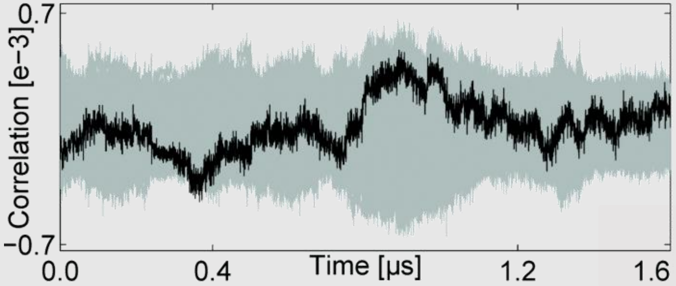


Clock disalignment
8 phase shift steps
10^7 measurements
→ 3,000,000 traces req.



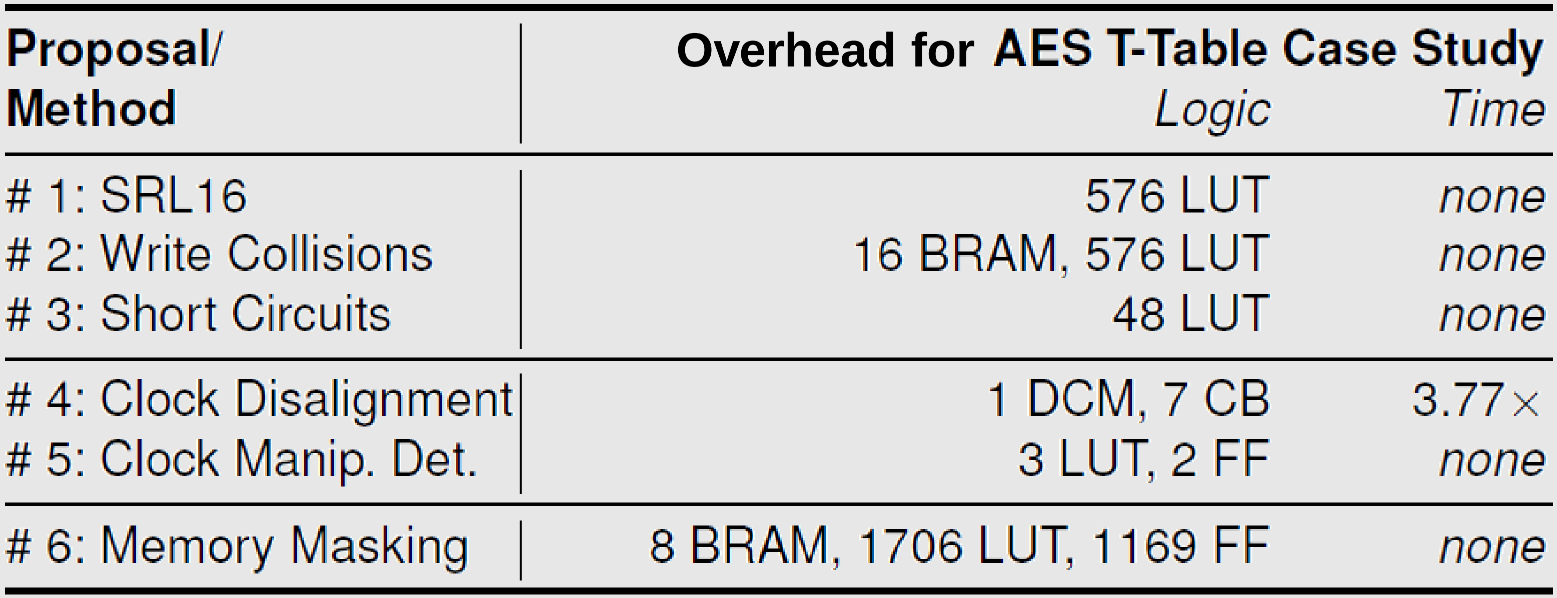


Memory masking with dual-ported BRAMs
10^8 measurements
→ Not successful (using first-order attack)



Evaluations: Efficiency and Resources

- To achieve higher SCA protection, combine several countermeasures
- CMs are quite efficient (parameters used: *s=16 (instances), r=36 (unit width))*



Proposal/ Method	Overhead for AES T-Table Case Study	
	Logic	*Time*
# 1: SRL16	576 LUT	*none*
# 2: Write Collisions	16 BRAM, 576 LUT	*none*
# 3: Short Circuits	48 LUT	*none*
# 4: Clock Disalignment	1 DCM, 7 CB	3.77×
# 5: Clock Manip. Det.	3 LUT, 2 FF	*none*
# 6: Memory Masking	8 BRAM, 1706 LUT, 1169 FF	*none*

(FF = Flip-Flop, LUT = Look-Up-Table, CB = Clock Buffer,
DCM=Digital Clock Manager, BRAM = Block RAM)

Agenda

- Introduction and Motivation
- Design Proposals for FPGAs
  - Noise Generation
  - Clock Disalignment
  - Memory Masking
- Evaluations
- **Conclusions**

Conclusions

- Proposed five generic countermeasures specific for FPGAs (implemented using resources usually wasted otherwise)
  - Noise generators
  - Clock disalignment and manipulation detection
  - Memory masking using dual-ported BRAMs
- Memory masking method provides solid protection against first-order attacks
- Combining countermeasures might also provide protection against higher-order attacks (→ still needs to be evaluated!)
- For third-party evaluation, PROM files for SASEBO are provided
  → available after next week at http://www.emsec.rub.de/research/publications

Ende. Thank you!