Trojan Side Channels

Lightweight Hardware Trojans through Side Channel Engineering

Lang Lin[1] Markus Kasper[2] Tim Güneysu[2]
Christof Paar[1,2] Wayne Burleson[1]

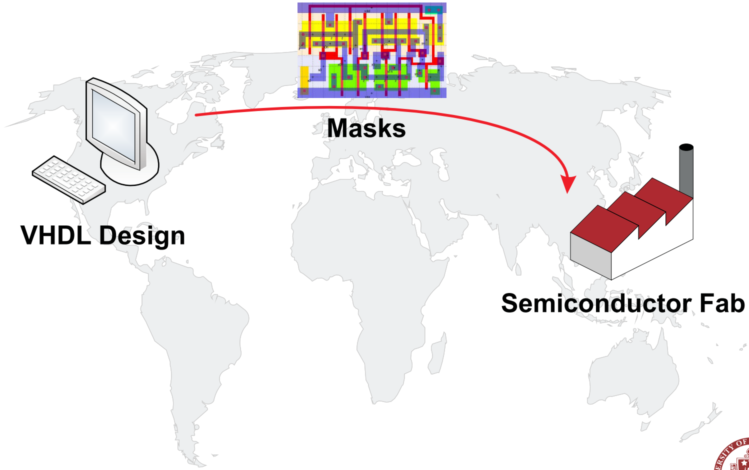
[1]VLSI Circuits and Systems Group
University of Massachusetts Amherst

[2]Embedded Security Group
Ruhr-Universität Bochum

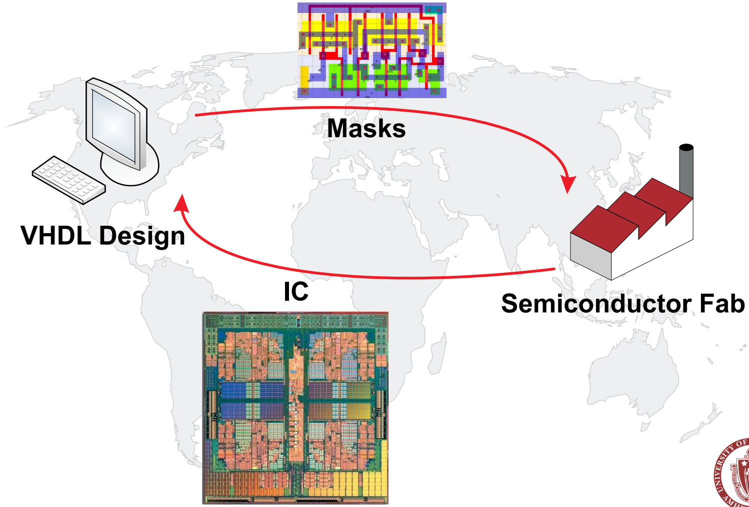
CHES Workshop 2009

A Semiconductor Manufacturing Flow

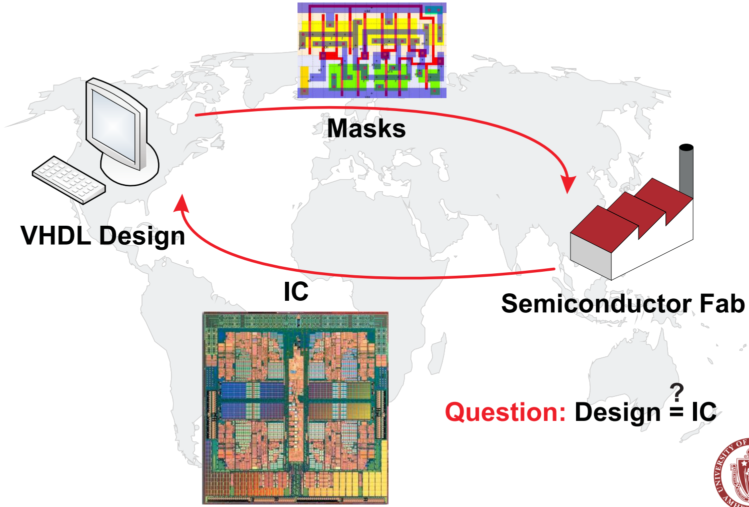
A Semiconductor Manufacturing Flow



A Semiconductor Manufacturing Flow



A Semiconductor Manufacturing Flow

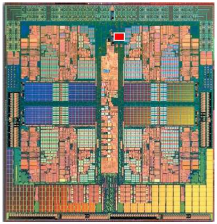


Question: Design $\overset{?}{=}$ IC

Trojan Hardware

Action: Functionality of the malicious hardware

- Modify functionality
- Leak secret information



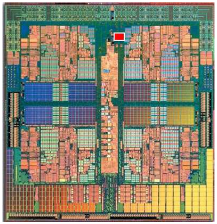
Trojan Hardware

Action: Functionality of the malicious hardware

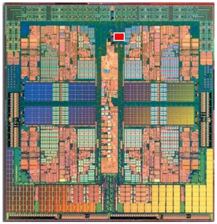
- Modify functionality
- Leak secret information

Trigger: Mechanism to activate the Trojan

- "Always on"
- Triggered by timer/counter
- Event controlled
  - Sequence
  - State



Trojan Hardware



Action: Functionality of the malicious hardware

- Modify functionality
- Leak secret information

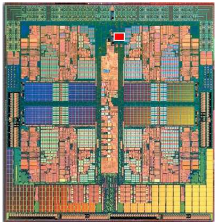
Trigger: Mechanism to activate the Trojan

- "Always on"
- Triggered by timer/counter
- Event controlled
    - Sequence
    - State

Channel: Transmission of information

- Via I/O pins

Trojan Hardware



Action: Functionality of the malicious hardware

- Modify functionality
- Leak secret information

Trigger: Mechanism to activate the Trojan

- "Always on"
- Triggered by timer/counter
- Event controlled
  - Sequence
  - State

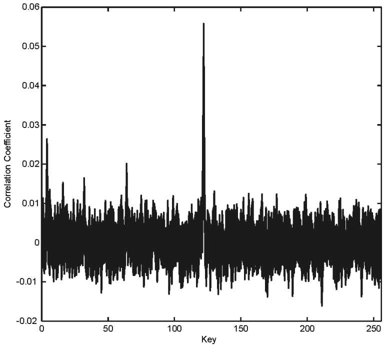
Channel: Transmission of information

- Via I/O pins
- Via physical side channels

Combining two Concepts

Ingredients:

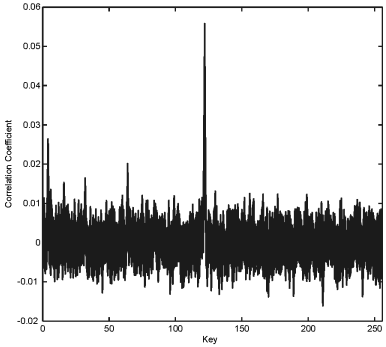
Side Channel Analysis



Combining two Concepts

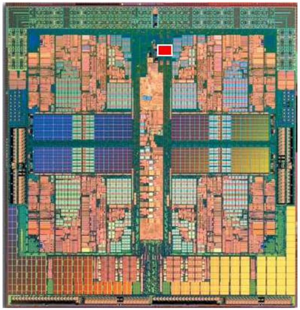
Ingredients:

Side Channel Analysis

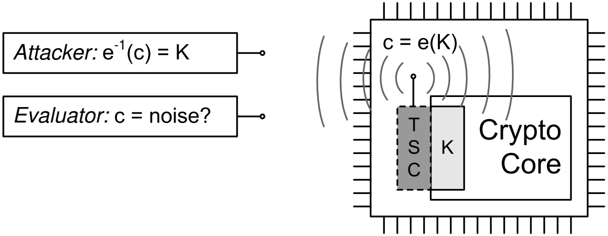


+

Trojan Hardware



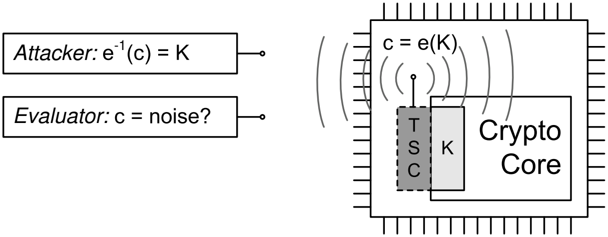
Our Model of an Trojan Side Channel Scenario



The Concept of Trojan Side Channels:

- Use subtle side-channels to leak information
  $\Rightarrow$ Replacing leakage via I/O pins

Our Model of an Trojan Side Channel Scenario



The Concept of Trojan Side Channels:

- Use subtle side-channels to leak information
  $\Rightarrow$ Replacing leakage via I/O pins
- "Encrypt" the Trojan
  $\Rightarrow$ Only the implementer may access the information

Assumptions

Assumptions:

- Target device provides a high level of side channel resistance
  $\Rightarrow$ Side channel attacks on the targeted secret are infeasible

Assumptions

Assumptions:

- Target device provides a high level of side channel resistance
  $\Rightarrow$ Side channel attacks on the targeted secret are infeasible
- Target device uses reverse engineering countermeasures

How to Design a Trojan Side Channel

Resulting Design Goals

How to Design a Trojan Side Channel

Resulting Design Goals

Detectability:

- **Size:** Low gate count or parametric changes only
- **Side Channel Leakage:** Must not be detected by power analyses
- **Trigger:** "Always on"

How to Design a Trojan Side Channel

Resulting Design Goals

Detectability:

- **Size:** Low gate count or parametric changes only
- **Side Channel Leakage:** Must not be detected by power analyses
- **Trigger:** "Always on"

Usability:

Encryption property: TSC is only accessible by its implementer

Components of a Trojan Side Channel

Components of a Trojan Side Channel

1. An internal state

- Inherently existing or artificially introduced
- Used to add variance to the secret information

Components of a Trojan Side Channel

1. An internal state

- Inherently existing or artificially introduced
- Used to add variance to the secret information

2. A combination function

- Encodes secret information using the internal state

Components of a Trojan Side Channel

1. An internal state

- Inherently existing or artificially introduced
- Used to add variance to the secret information

2. A combination function

- Encodes secret information using the internal state

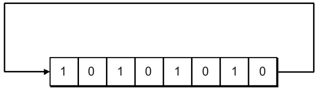
3. A leakage circuit

- Maps output of the combination function to physical leakage
- Can sometimes be realized within the combination function

The Leakage Circuit

For FPGAs:

- Look-up table as circular shift register
- Toggling flip-flops
- Other glitching logic
- ...

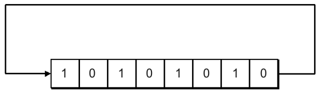


$$\text{clocked} = \begin{cases} \text{yes} & \Rightarrow \text{leakage} \\ \text{no} & \Rightarrow \text{no leakage} \end{cases}$$

The Leakage Circuit

For FPGAs:

- Look-up table as circular shift register
- Toggling flip-flops
- Other glitching logic
- ...



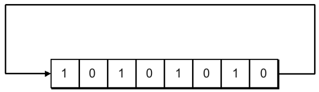
$$\text{clocked} = \begin{cases} \text{yes} & \Rightarrow \text{leakage} \\ \text{no} & \Rightarrow \text{no leakage} \end{cases}$$

Note: The amount of generated leakage is part of the design space

The Leakage Circuit

For FPGAs:

- Look-up table as circular shift register
- Toggling flip-flops
- Other glitching logic
- ...



$$\text{clocked} = \begin{cases} \text{yes} & \Rightarrow \text{leakage} \\ \text{no} & \Rightarrow \text{no leakage} \end{cases}$$

Note: The amount of generated leakage is part of the design space

But: More leakage makes detection easier for everyone

First Practical Results

We demonstrate two examples of Trojan Side Channels:

1. Spread Spectrum Trojan Side Channel

First Practical Results

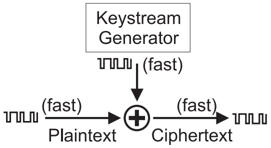
We demonstrate two examples of Trojan Side Channels:

1. Spread Spectrum Trojan Side Channel
2. Input Modulated Trojan Side Channel

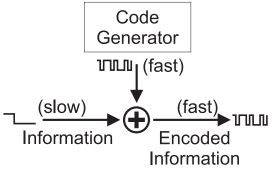
Spread Spectrum Theory on One Slide

Encoding: Very similar to a stream cipher

Stream Cipher:

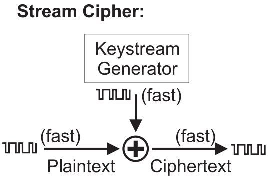


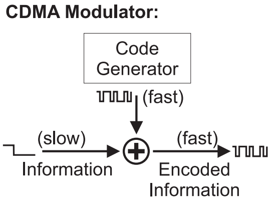
CDMA Modulator:



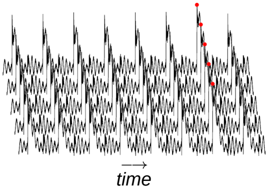
Spread Spectrum Theory on One Slide

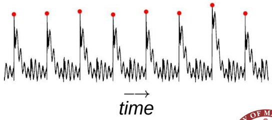
Encoding: Very similar to a stream cipher



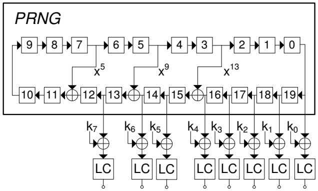


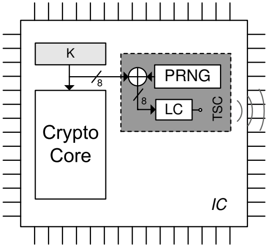
Decoding: Very similar to a correlation power analysis





Spread Spectrum TSC





Components:

Internal state: Introduced linear feedback shift register (LFSR)

Combination function: Bitwise XOR with LFSR outputs

Leakage circuit: 8 parallel flip-flops per bit

Encryption property: Unknown PRNG sequence

Detection of the CDMA TSC

Method to detect this kind of Trojan:

1. Use fixed key and average over many traces

Detection of the CDMA TSC

Method to detect this kind of Trojan:

1. Use fixed key and average over many traces
2. Flip a single key bit and average over many traces

Detection of the CDMA TSC

Method to detect this kind of Trojan:

1. Use fixed key and average over many traces
2. Flip a single key bit and average over many traces
3. Calculate the difference of means

Detection of the CDMA TSC

Method to detect this kind of Trojan:

1. Use fixed key and average over many traces
2. Flip a single key bit and average over many traces
3. Calculate the difference of means
4. Find code sequence

Detection of the CDMA TSC

Method to detect this kind of Trojan:

1. Use fixed key and average over many traces
2. Flip a single key bit and average over many traces
3. Calculate the difference of means
4. Find code sequence
5. Reconstruct the LFSR

Detection of the CDMA TSC

Method to detect this kind of Trojan:

1. Use fixed key and average over many traces
2. Flip a single key bit and average over many traces
3. Calculate the difference of means
4. Find code sequence
5. Reconstruct the LFSR
6. Use the Trojan Side Channel

Detection of the CDMA TSC

Method to detect this kind of Trojan:

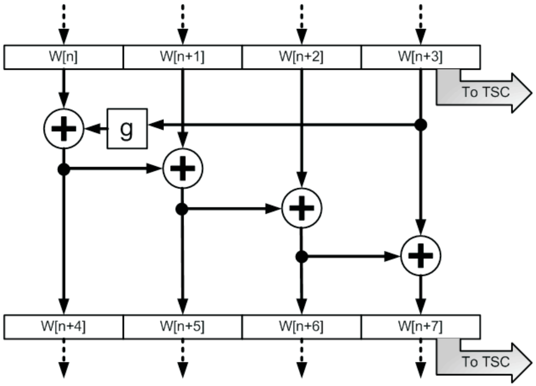
1. Use fixed key and average over many traces
2. Flip a single key bit and average over many traces
3. Calculate the difference of means
4. Find code sequence
5. Reconstruct the LFSR
6. Use the Trojan Side Channel

How to improve this TSC:

- Transfer only nonlinear combinations of bits (see second TSC)
- Use input to initialize the code generator

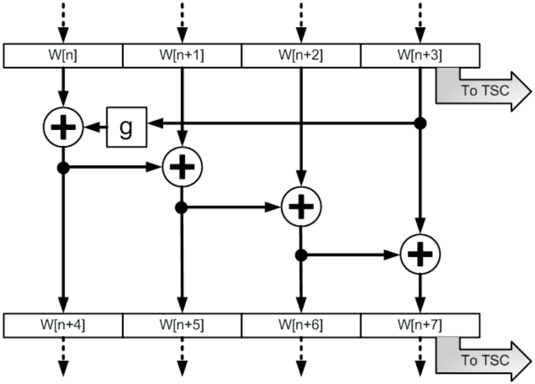
The Input Modulated Trojan Side Channel

Scenario:
TSC attached to AES-128 key schedule



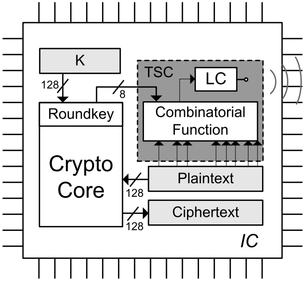
The Input Modulated Trojan Side Channel

Scenario:
TSC attached to AES-128 key schedule



Background:
16 bits of the schedule sufficient to attack the key

The Input Modulated Trojan Side Channel

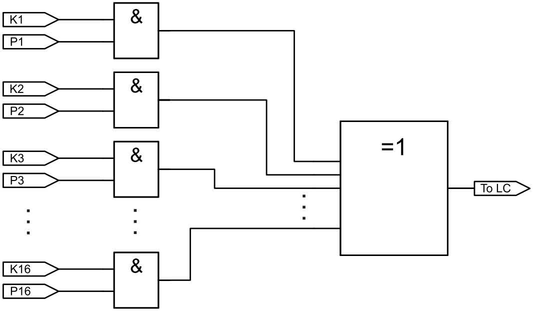


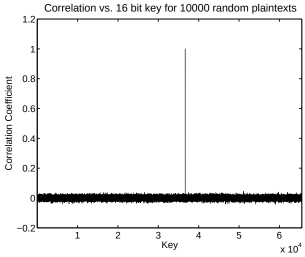
Components:

- Internal State: 16 bits of plaintext
- Combination Function: next slide
- Leakage Circuit: Look-up table as circular shift register

The Combination Function

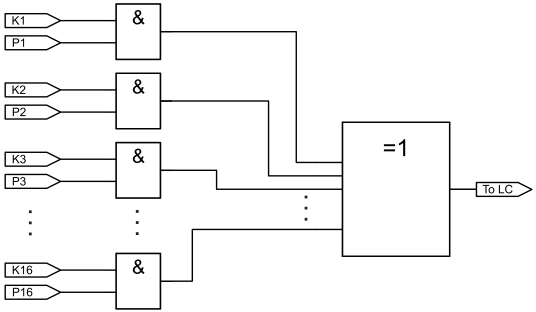
Combination function used here:

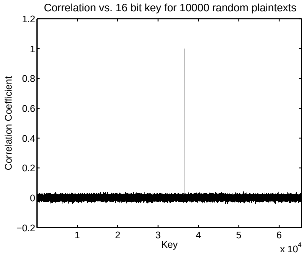




The Combination Function

Combination function used here:





Design Criteria:

Good discrimination properties in DPAs

Uses only few input bits and logic gates

Leakage hard to detect during SCA evaluation

Encryption Property

Question:
Where is the encryption property?

Encryption Property

Question:
Where is the encryption property?

Evaluator has to predict selection of plaintext bits

Encryption Property

Question:
Where is the encryption property?

Evaluator has to predict selection of plaintext bits

- The choice of used plaintext bits is not known

Encryption Property

Question:
Where is the encryption property?

Evaluator has to predict selection of plaintext bits

- The choice of used plaintext bits is not known
- Their order is also a secret

Encryption Property

Question:
Where is the encryption property?

Evaluator has to predict selection of plaintext bits

- The choice of used plaintext bits is not known
- Their order is also a secret
- $\approx 2^{55}$ possible choices for an 8 bit TSC

Encryption Property

Question:
Where is the encryption property?

Evaluator has to predict selection of plaintext bits

- The choice of used plaintext bits is not known
- Their order is also a secret
- $\approx 2^{55}$ possible choices for an 8 bit TSC
- $\approx 2^{110}$ possible choices for an 16 bit TSC

Experimental Setup

Implemented: AES key schedule with input modulated TSC

Side-channel Attack
Standard Evaluation BOard
SASEBO

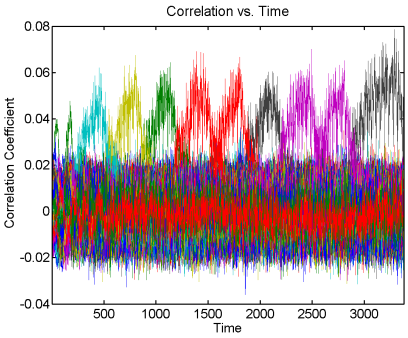
Device: Xilinx Virtex-2 PRO XC2VP7-5 FPGA @ 24MHz

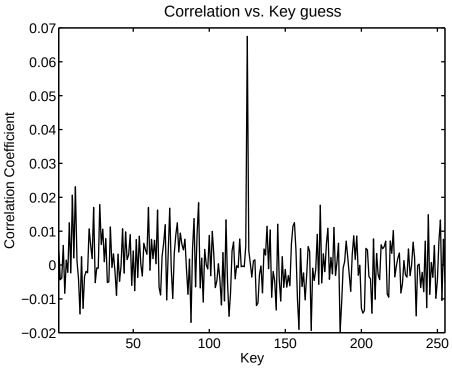
For more info on the SASEBO project:

Contact Akashi Satoh or visit the SASEBO website:

http://www.rcis.aist.go.jp/special/SASEBO/index-en.html

Experimental Results





Figures generated using 20.000 power traces

Summary

- New concept: Use side channels as building blocks
- Introduced the very flexible approach of Trojan Side Channels
- First practical results demonstrate feasibility of the concept

Trojan Side Channels

Lightweight Hardware Trojans through Side Channel Engineering

Lang Lin, Markus Kasper, Tim Güneysu, Wayne Burleson, Christof Paar

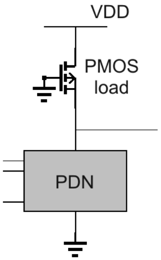
Thanks For Your Attention!

Any Questions?

The Leakage Circuit

For ASICs:

- Ratioed Logic
  - Pseudo-NMOS
  - Pseudo-PMOS
  - Other resistive load
- Precharge Logic (dynamic logic)
- ...



Note:
Gates don’t have to be sized to be functional, they only need to modify the power consumption!

Detection Methods

Four methods to detect malicious hardware:

1. Detection during regular testing (functional)

Detection Methods

Four methods to detect malicious hardware:

1 Detection during regular testing (functional)

$\rightarrow$ Limited by trigger

Detection Methods

Four methods to detect malicious hardware:

1. Detection during regular testing (functional)

   $\rightarrow$ Limited by trigger
2. Optical scanning of the chip-layout (geometry)

Detection Methods

Four methods to detect malicious hardware:

1. Detection during regular testing (functional)
   $\rightarrow$ Limited by trigger
2. Optical scanning of the chip-layout (geometry)
   $\rightarrow$ Limited by size

Detection Methods

Four methods to detect malicious hardware:

1. Detection during regular testing (functional)

   $\rightarrow$ Limited by trigger
2. Optical scanning of the chip-layout (geometry)

   $\rightarrow$ Limited by size
3. Verification of physical properties (stochastic methods)

Detection Methods

Four methods to detect malicious hardware:

1. Detection during regular testing (functional)
   $\rightarrow$ Limited by trigger
2. Optical scanning of the chip-layout (geometry)
   $\rightarrow$ Limited by size
3. Verification of physical properties (stochastic methods)
   $\rightarrow$ Limited by process variations

Detection Methods

Four methods to detect malicious hardware:

1. Detection during regular testing (functional)

   $\rightarrow$ Limited by trigger
2. Optical scanning of the chip-layout (geometry)

   $\rightarrow$ Limited by size
3. Verification of physical properties (stochastic methods)

   $\rightarrow$ Limited by process variations
4. Built-in self test (design for integrity)

Detection Methods

Four methods to detect malicious hardware:

1. Detection during regular testing (functional)

   $\rightarrow$ Limited by trigger
2. Optical scanning of the chip-layout (geometry)

   $\rightarrow$ Limited by size
3. Verification of physical properties (stochastic methods)

   $\rightarrow$ Limited by process variations
4. Built-in self test (design for integrity)

   $\rightarrow$ Limited by the integrity of the additional logic

Further Work

There are still many things to be analyzed:

- Detectability needs to be verified experimentally

Further Work

There are still many things to be analyzed:

- Detectability needs to be verified experimentally
- Different combination and leakage circuits

Further Work

There are still many things to be analyzed:

- Detectability needs to be verified experimentally
- Different combination and leakage circuits
- ASIC implementations

Further Work

There are still many things to be analyzed:

- Detectability needs to be verified experimentally
- Different combination and leakage circuits
- ASIC implementations
- TSCs with parametric modifications (like doping or geometry)