TU Graz

SCIENCE
PASSION
TECHNOLOGY

SYNFI: Pre-Silicon Fault Analysis of an Open-Source Secure Element

P. Nasahl[1,3], M. Osorio[1], P. Vogel[2], M. Schaffner[1], T. Trippel[1], D. Rizzo[1], S. Mangard[3,4]
[1]Google, [2]lowRISC, [3]Graz University of Technology, [4]Lamarr Security Research

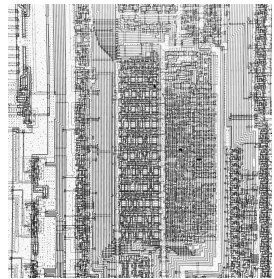
CHES, 20.09.2022

Fault Attacks

- Active, physical attacks
- Manipulation of device or its environment
- Local or remote attacks

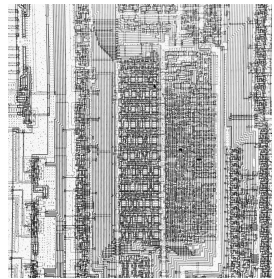
Fault Effects

- Effects are the electrical level are manifold
  - Timing violations
  - Transient voltage and current changes
  - ...
- Interested in higher levels of abstractions
  - Transient effects (bit flips)
  - Permanent effects (stuck-at)



Fault Effects

- Effects are the electrical level are manifold
  - Timing violations
  - Transient voltage and current changes
  - ...
- Interested in higher levels of abstractions
  - Transient effects (bit flips)
  - Permanent effects (stuck-at)



Fault Exploitation

- Manipulate control- or data-flow
- Generic:
  - Finite-state machines
  - Handshake signals
- CPU:
  - Manipulate PC, interrupt vectors
  - Manipulate and skip instructions

Fault Exploitation

- Manipulate control- or data-flow
- Generic:
  - Finite-state machines
  - Handshake signals
- CPU:
  - Manipulate PC, interrupt vectors
  - Manipulate and skip instructions

Fault Exploitation

- Manipulate control- or data-flow
- Generic:
  - Finite-state machines
  - Handshake signals
- CPU:
  - Manipulate PC, interrupt vectors
  - Manipulate and skip instructions

Fault Countermeasures

- Spatial Redundancy
- Temporal Redundancy
- Encoding

Fault Countermeasures

- Spatial Redundancy
- Temporal Redundancy
- Encoding
- Needed for security-critical devices

OpenTitan

- Open-source[1] RTL, design verification, firmware, software, and tooling
- Acts as root-of-trust
- Dedicated HW-based SCA & fault countermeasures

[1] https://github.com/lowRISC/opentitan

SYNFI

SYNFI Motivation

→ **We need to assure that the countermeasures work**

SYNFI Motivation

→ **We need to assure that the countermeasures work**

- Testing at RTL model
  - Verification results only valid at this level of abstraction
  - Synthesis consists of several optimization phases
    - redundancy-based countermeasures

SYNFI Motivation

→ **We need to assure that the countermeasures work**

- Testing at RTL model
  - Verification results only valid at this level of abstraction
  - Synthesis consists of several optimization phases
    - redundancy-based countermeasures
- Experimental verification
  - FPGA: different technology
  - ASIC: costly and time-consuming

SYNFI Motivation

→ **We need to assure that the countermeasures work**

- Testing at RTL model
  - Verification results only valid at this level of abstraction
  - Synthesis consists of several optimization phases
    - redundancy-based countermeasures
- Experimental verification
  - FPGA: different technology
  - ASIC: costly and time-consuming

FI countermeasure verification at the netlist

Why SYNFI?

- Manual FI in testbench
  - Gate/wire names are mangled

Why SYNFI?

- Manual FI in testbench
  - Gate/wire names are mangled
- Related work
  - Limited set of cells supported
  - No support for common design patterns (cycles)
  - Only fully flattened netlists

Why SYNFI?

- Manual FI in testbench
  - Gate/wire names are mangled
- Related work
  - Limited set of cells supported
  - No support for common design patterns (cycles)
  - Only fully flattened netlists

  $\rightarrow$ **Impose requirements to netlist**

Design Flow Adaptions

- Design flows need to work
- Interface for different stakeholders
- Responsible for the success of the project

→ **Use design flows that are provably working!**

→ **Tool that imposes requirements to netlist cannot be used!**

Design Flow Adaptions

- Design flows need to work
- Interface for different stakeholders
- Responsible for the success of the project

→ **Use design flows that are provably working!**

→ **Tool that imposes requirements to netlist cannot be used!**

Design Flow Adaptions

- Design flows need to work
- Interface for different stakeholders
- Responsible for the success of the project

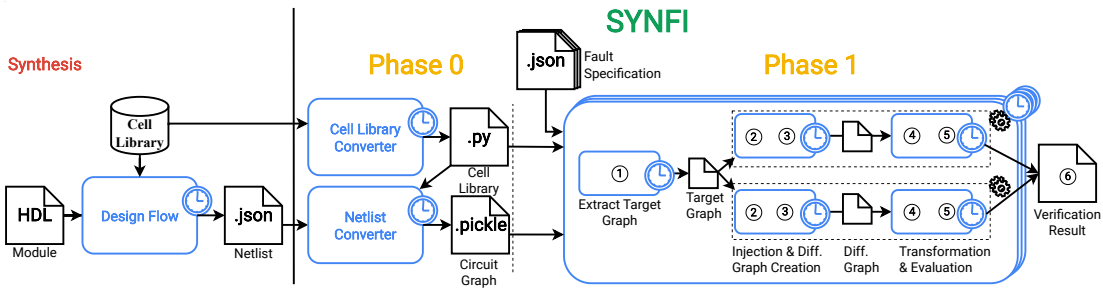
→ **Use design flows that are provably working!**

→ **Tool that imposes requirements to netlist cannot be used!**

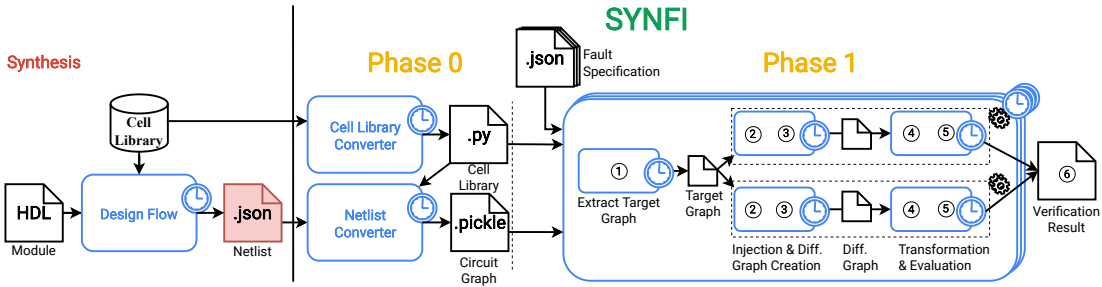
SYNFI

- Imposes no requirements to netlist:
  - Open or proprietary synthesis tools & standard cell libraries
  - Support common design patterns
- Analysis:
  - Fault countermeasures provide expected security

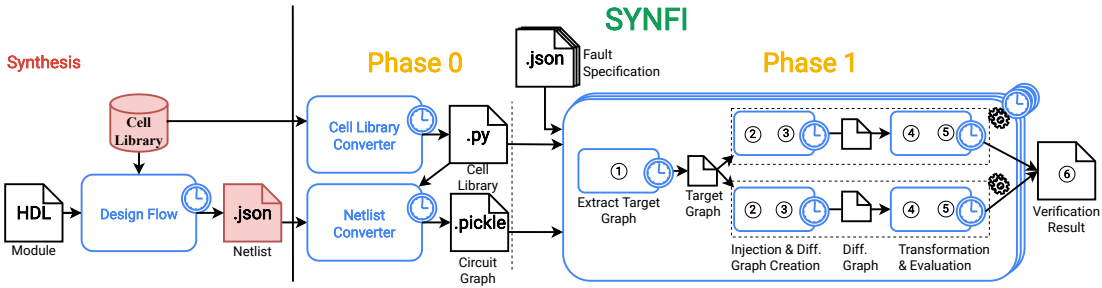
SYNFI Overview



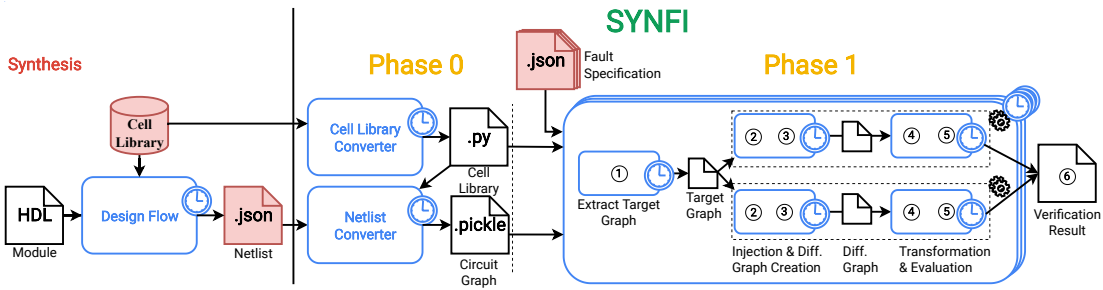
SYNFI Inputs



SYNFI Inputs



SYNFI Inputs

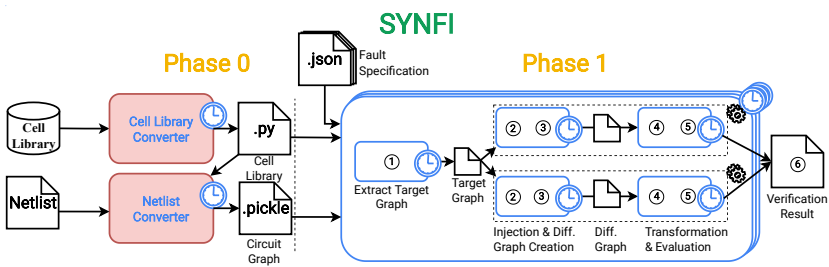


Fault Specification

- Fault Target
- Attacker Model:
  - Number of simultaneous faults
  - Fault locations
  - Fault effects

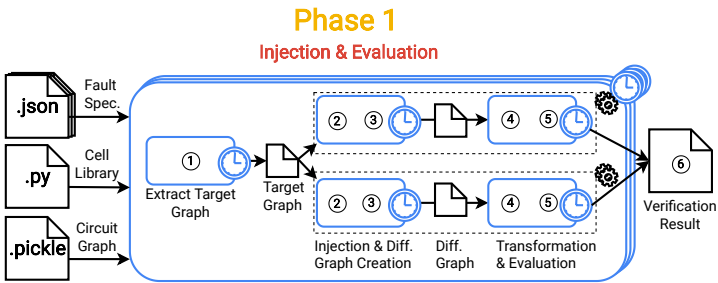
Phase 0: Preparation

- Cell Library Conversion
- Netlist Conversion
  - **MultiDiGraph**



Phase 1: Injection & Evaluation

- Target graph extraction
- Fault injection
- Differential graph
- Transformation & evaluation



Evaluation

- Arbitrary Fault Effects:
  - Change in output & countermeasure not triggered
- Specific Fault Effects:
  - Target state reached & countermeasure not triggered

Evaluation

- Arbitrary Fault Effects:
  - Change in output & countermeasure not triggered
- Specific Fault Effects:
  - Target state reached & countermeasure not triggered

Analysis of OpenTitan

OpenTitan Analysis

→ **Check whether fault countermeasures work**

OpenTitan Analysis

→ **Check whether fault countermeasures work**

- Dedicated threat model for each module
- Analyzed AES, Life Cycle Controller, Ibex, Generic Primitives
- Synthesized with proprietary & open-source hardware design flow

AES: Handshake Signals

- **Overview:**
  - Internal signals driven by FSMs
- **Attacker goal:**
  - Manipulate handshake signals
  - Influence data- and control-flow of the encryption
- **Attacker target:**
  - Next-state logic, state registers, output logic
  - Control signals
  - Output signal

AES: Handshake Signals

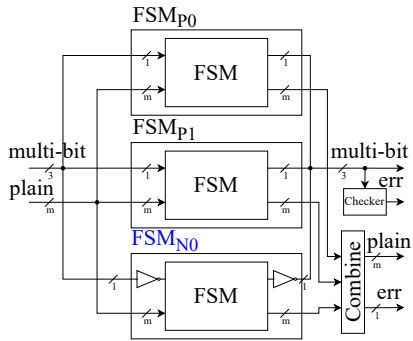
- Multi-Rail FSM

AES: Handshake Signals

- Multi-Rail FSM
- Encode input & output signals
  - $1 \rightarrow 011, 0 \rightarrow 100$

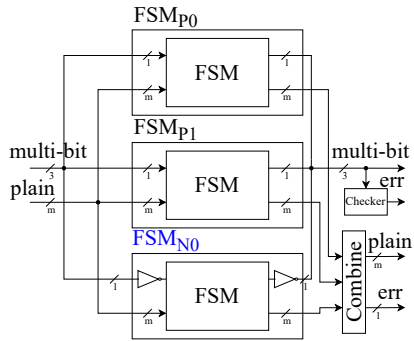
AES: Handshake Signals

- Multi-Rail FSM
- Encode input & output signals
  - $1 \rightarrow 011, 0 \rightarrow 100$
- Redundantly instantiate FSMs
  - Operate on positive and negative rail



AES: Handshake Signals

- Multi-Rail FSM
- Encode input & output signals
  - $1 \rightarrow 011, 0 \rightarrow 100$
- Redundantly instantiate FSMs
  - Operate on positive and negative rail
- Combine output signals & check encoding



AES: Handshake Signals

- **SYNFI evaluation:**
  - 3 simultaneous faults
  - Manipulate handshake signal
  - Expected protection level: 3

AES: Handshake Signals

- **SYNFI evaluation:**
  - 3 simultaneous faults
  - Manipulate handshake signal
  - Expected protection level: 3
- **SYNFI result:**
  - Verified protection level: 2

AES: Handshake Signals

- **SYNFI evaluation:**
  - 3 simultaneous faults
  - Manipulate handshake signal
  - Expected protection level: 3
- **SYNFI result:**
  - Verified protection level: 2
  - Synthesis tool reduced 3-bit encoded signal to 2-bits

AES: Handshake Signals

- **SYNFI evaluation:**
  - 3 simultaneous faults
  - Manipulate handshake signal
  - Expected protection level: 3
- **SYNFI result:**
  - Verified protection level: 2
  - Synthesis tool reduced 3-bit encoded signal to 2-bits

→ **Attribute register**

AES: Handshake Signals

- **SYNFI evaluation:**
  - 3 simultaneous faults
  - Manipulate handshake signal
  - Expected protection level: 3
- **SYNFI result:**
  - Verified protection level: 2
  - Synthesis tool reduced 3-bit encoded signal to 2-bits

→ **Attribute register**

→ **Expected protection level confirmed**

AES: FSM State Encoding

- **Overview:**
  - Encode states with Hamming distance of 3
- **Attacker goal:**
  - Enter a different state
- **Attacker target:**
  - State registers

AES: FSM State Encoding

- **SYNFI evaluation:**
  - 3 simultaneous faults
  - Enter a different state
  - Expected protection level: 3

AES: FSM State Encoding

- **SYNFI evaluation:**
  - 3 simultaneous faults
  - Enter a different state
  - Expected protection level: 3
- **SYNFI result:**
  - Verified protection level: 2

AES: FSM State Encoding

- **SYNFI evaluation:**
  - 3 simultaneous faults
  - Enter a different state
  - Expected protection level: 3
- **SYNFI result:**
  - Verified protection level: 2
  - Yosys FSM optimizations

AES: FSM State Encoding

- **SYNFI evaluation:**
  - 3 simultaneous faults
  - Enter a different state
  - Expected protection level: 3
- **SYNFI result:**
  - Verified protection level: 2
  - Yosys FSM optimizations

→ **Disable FSM optimizations**

AES: FSM State Encoding

- **SYNFI evaluation:**
  - 3 simultaneous faults
  - Enter a different state
  - Expected protection level: 3
- **SYNFI result:**
  - Verified protection level: 2
  - Yosys FSM optimizations

→ **Disable FSM optimizations**

→ **Area vs. security trade-off**

Life Cycle Controller

- **Overview:**
  - Transfers OpenTitan into different operational states
- **Attacker goal:**
  - Enter security-sensitive state
- **Attacker target:**
  - Bypass token check
  - Hijack FSM

Life Cycle Controller

- **SYNFI evaluation:**
  - 7 simultaneous faults
  - Enter debug state from production state
  - Expected protection level: at least 3

Life Cycle Controller

- **SYNFI evaluation:**
  - 7 simultaneous faults
  - Enter debug state from production state
  - Expected protection level: at least 3
- **SYNFI result:**
  - Faulting token comparisons
  - Faulting state registers to hijack FSM

Life Cycle Controller

- **SYNFI evaluation:**
  - 7 simultaneous faults
  - Enter debug state from production state
  - Expected protection level: at least 3
- **SYNFI result:**
  - Faulting token comparisons
  - Faulting state registers to hijack FSM
  - Verified protection level: ≥ 3

Ibex: Program Counter

- **Overview:**
  - 32-bit RISC-V CPU
- **Attacker goal:**
  - Manipulate program counter
  - Redirect control-flow
- **Attacker target:**
  - Instruction fetch pipeline stage

Ibex: Program Counter

- **SYNFI evaluation:**
  - Single core
  - Dual core lockstep
    - Expected protection level: 2

Ibex: Program Counter

- **SYNFI evaluation:**
  - Single core
  - Dual core lockstep
    - Expected protection level: 2
- **SYNFI result:**

  $\rightarrow$ **Expected protection level confirmed**

Conclusion

- **SYNFI:**
  - Analyze resilience of fault countermeasures
  - Netlists synthesized with open & proprietary synthesis tools
- **Evaluation:**
  - Identified several weaknesses in the AES IP
  - Confirmed security of other modules
- **Improved Designs:**
  - New countermeasures
  - Reassessed security, contributed to OpenTitan project

Conclusion

- **SYNFI:**
  - Analyze resilience of fault countermeasures
  - Netlists synthesized with open & proprietary synthesis tools
- **Evaluation:**
  - Identified several weaknesses in the AES IP
  - Confirmed security of other modules
- **Improved Designs:**
  - New countermeasures
  - Reassessed security, contributed to OpenTitan project

Conclusion

- **SYNFI:**
  - Analyze resilience of fault countermeasures
  - Netlists synthesized with open & proprietary synthesis tools
- **Evaluation:**
  - Identified several weaknesses in the AES IP
  - Confirmed security of other modules
- **Improved Designs:**
  - New countermeasures
  - Reassessed security, contributed to OpenTitan project

TU Graz

SCIENCE
PASSION
TECHNOLOGY

SYNFI: Pre-Silicon Fault Analysis of an Open-Source Secure Element

P. Nasahl[1,3], M. Osorio[1], P. Vogel[2], M. Schaffner[1], T. Trippel[1], D. Rizzo[1], S. Mangard[3,4]
[1]Google, [2]lowRISC, [3]Graz University of Technology, [4]Lamarr Security Research

CHES, 20.09.2022