Exploration of Fault Effects on Formal RISC-V Microarchitecture Models*

Simon Tollec[1], Mihail Asavoae[1], Damien Couroussé[2], Karine Heydemann[3] and Mathieu Jan[1]

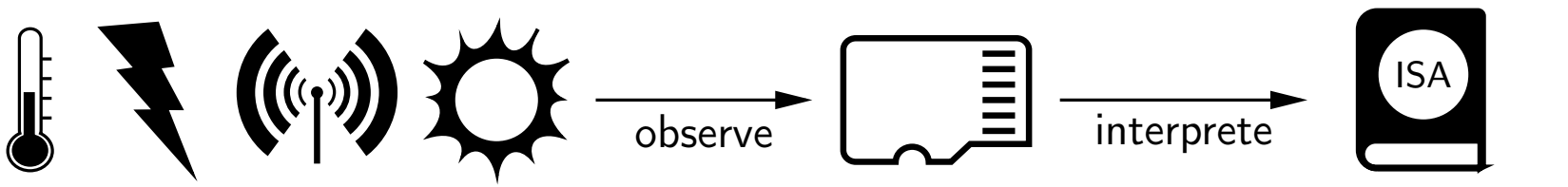
[1]Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France
[2]Univ. Grenoble Alpes, CEA, List, F-38000 Grenoble, France
[3]Sorbonne Univ., CNRS, LIP6, F-75005, Paris, France

JAIF, November 9, 2022

*Published in *2022 Workshop on Fault Diagnosis and Tolerance in Cryptography*

Fault Effects Characterization

Experimental characterization



Most common observed effects

- Instruction skips [Riviere, 2015] [Menu, 2020]
- Instruction replacement [Moro, 2013] [Trouchkine, 2020]
- Data corruption

Microarchitectural Fault Effects

Recent work

- Some effects cannot be *explained* at the ISA level
  - *Magic edges* [Proy, 2019]
- Some effects cannot be *captured* at the ISA level
  - *Forwarding* [Laurent, 2018]

➔ We need an analysis that combines both the HW and the SW

Bridging the Gap between the HW and the SW Level

Existing fault analysis methods

- At the circuit level
  - Check if an adversary can enter a specific HW state
  - Test if HW countermeasures correctly detect fault injections

  ➔ *The running software is not considered*

- At the ISA level
  - Encompass a broad set of effects and abstract the implementation details
  - ISA fault models may incorrectly capture the real effects

  ➔ *Not sufficient to fully understand potential of fault injection*

➔ We need an automated method to bridge the gap between SW/HW

Contributions

Automated formal modeling of HW and SW

➔ For exploring microarchitectural fault effects on SW security

➔ For analyzing the robustness of HW or SW countermeasures

Why using formal methods, e.g., model checking?

- Give counterexamples or a proof
- Verification process guided toward counter-examples

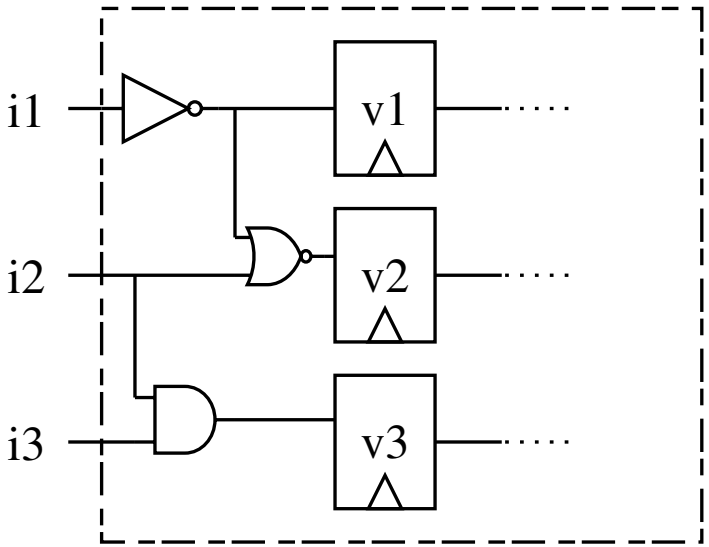
Modeling Synchronous Circuits

Transition System, a tuple $\langle X, I, T \rangle$ where

- X is a set of the state variables composing the system,
- $I(X)$ is the formula constraining the initial state,
- $T(X, X')$ is the transition relation between X and the next state X'

Modeling Synchronous Circuits as Transition Systems

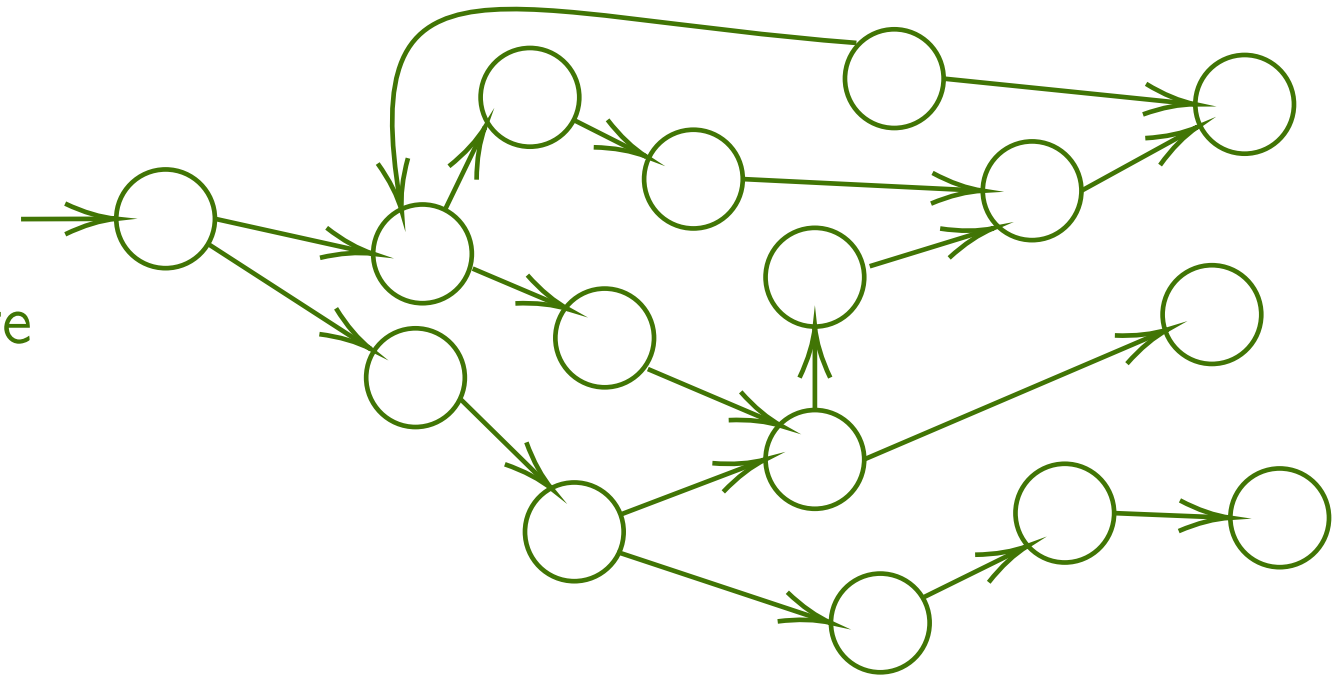
- $X = (i_1, i_2, \ldots, v_1, v_2, \ldots)$ where i_N are the inputs and, v_N are the state-holding elements
- $I(X) =$ initial state (e.g., imposed by a reset)
- $T(X, X')$, state-holding elements update via combinatorial logic



Modeling HW/SW co-design with Fault

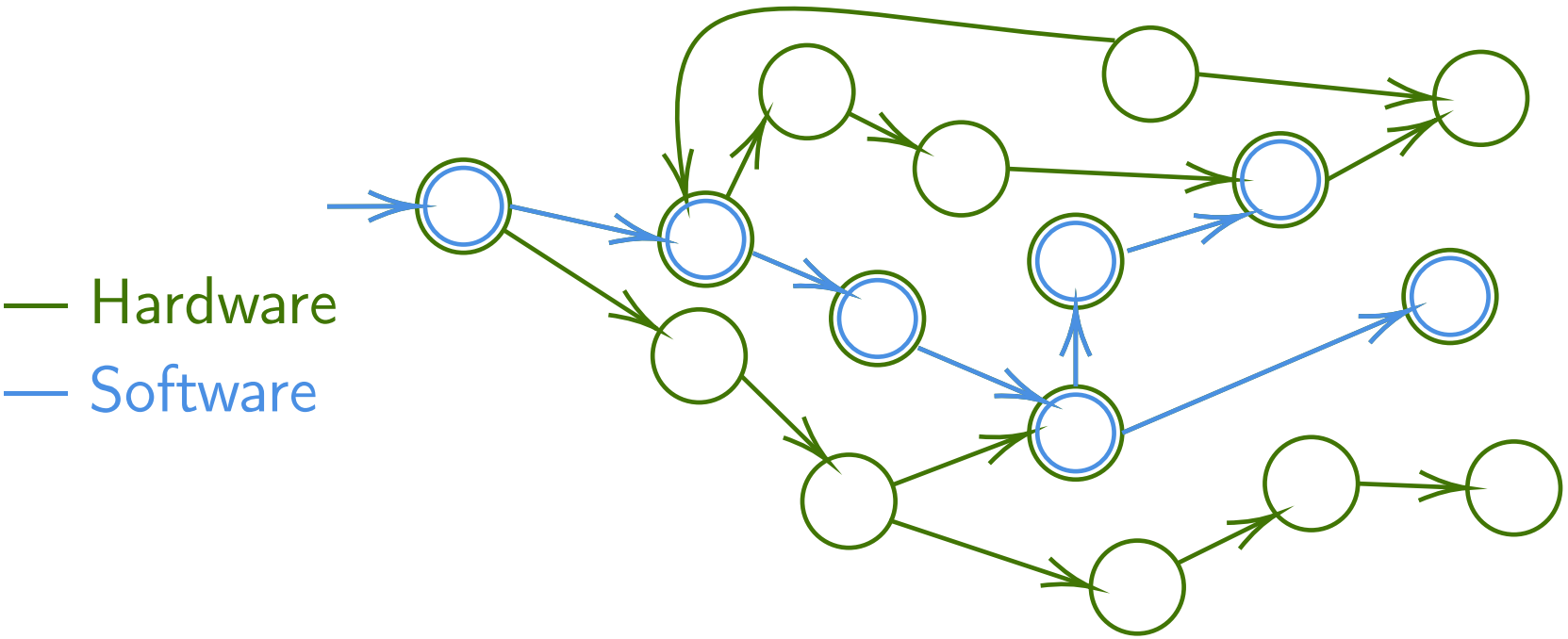
Modeling Steps

— Hardware



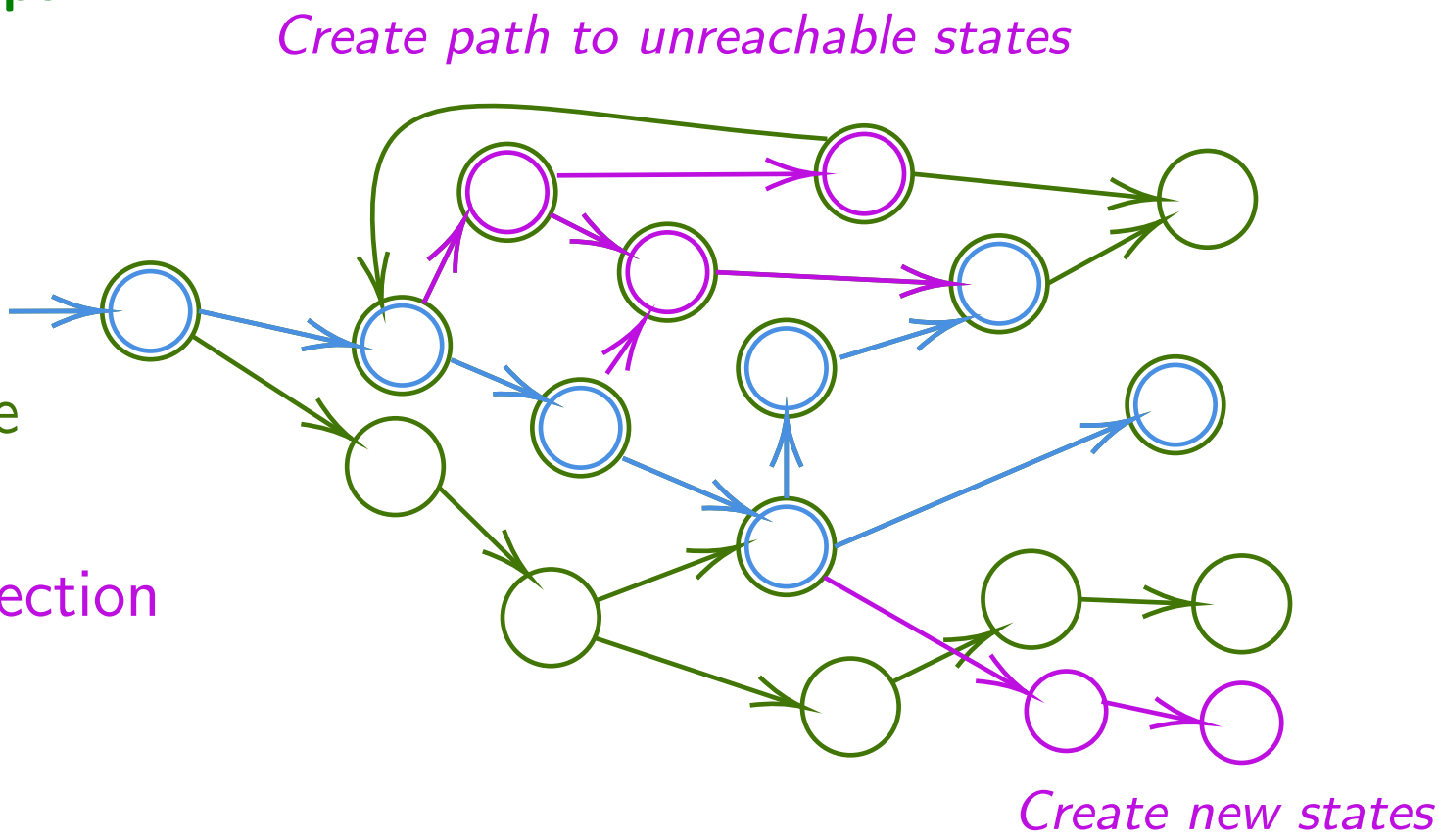
Modeling HW/SW co-design with Fault

Modeling Steps



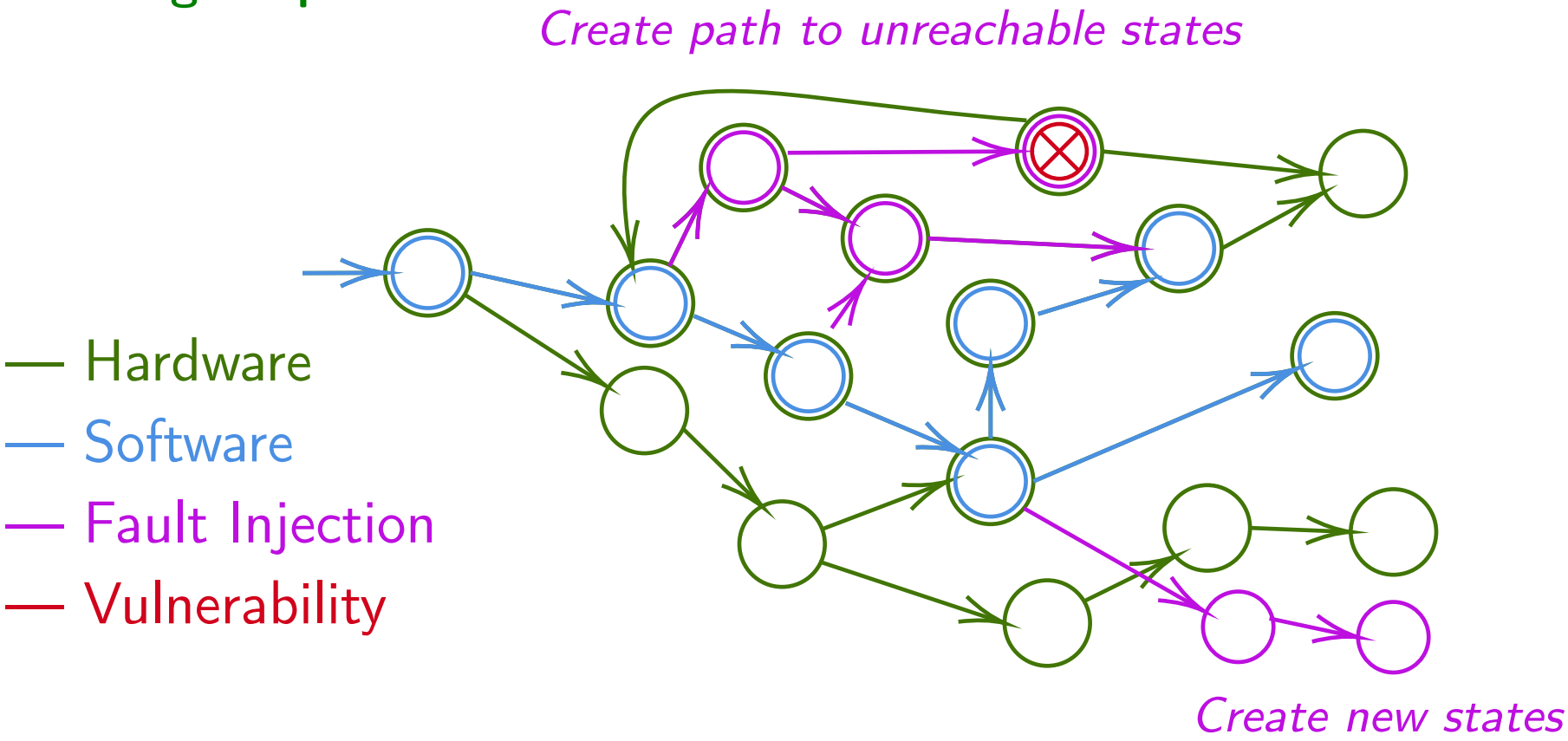
Modeling HW/SW co-design with Fault

Modeling Steps



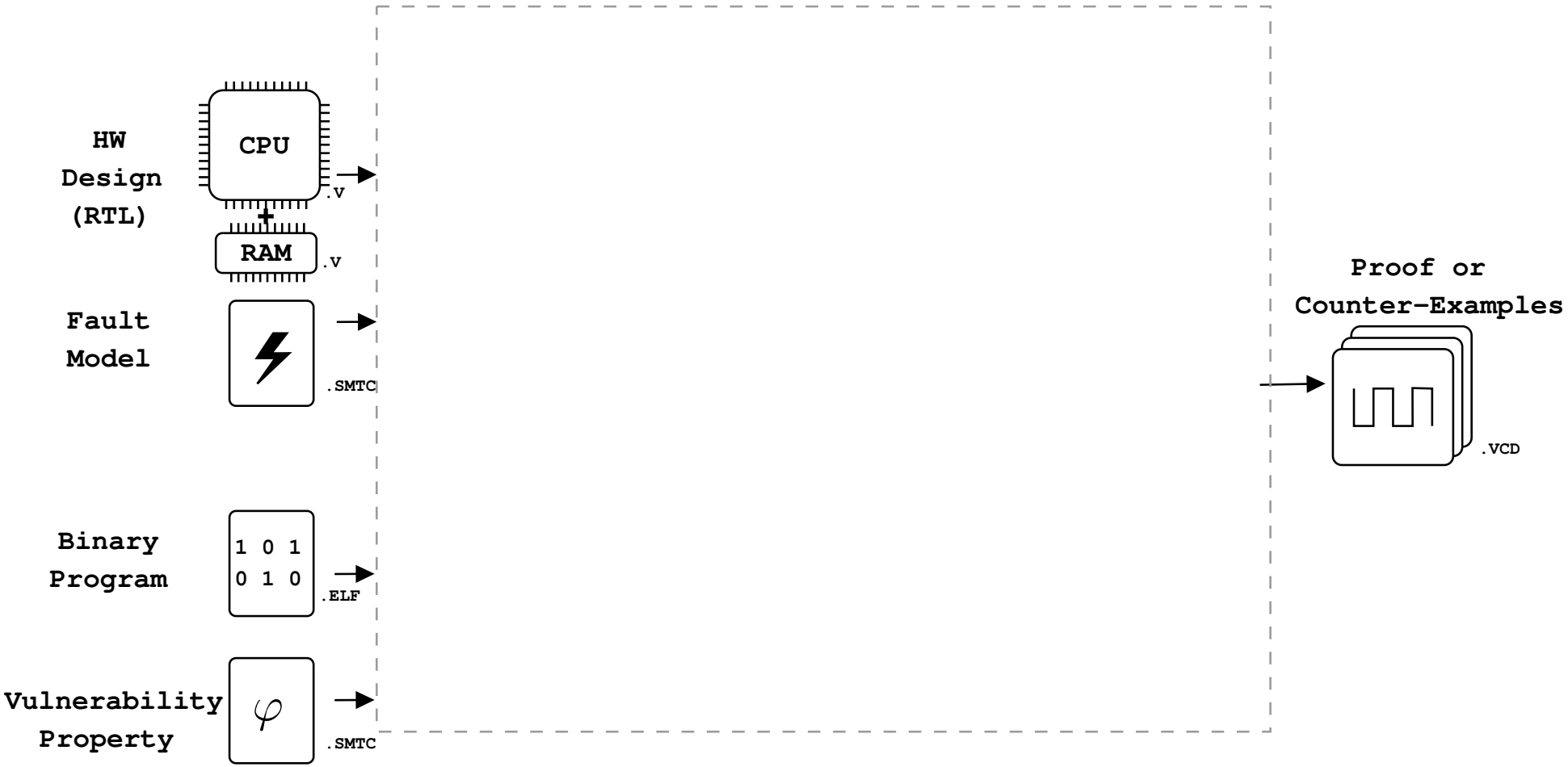
Modeling HW/SW co-design with Fault

Modeling Steps



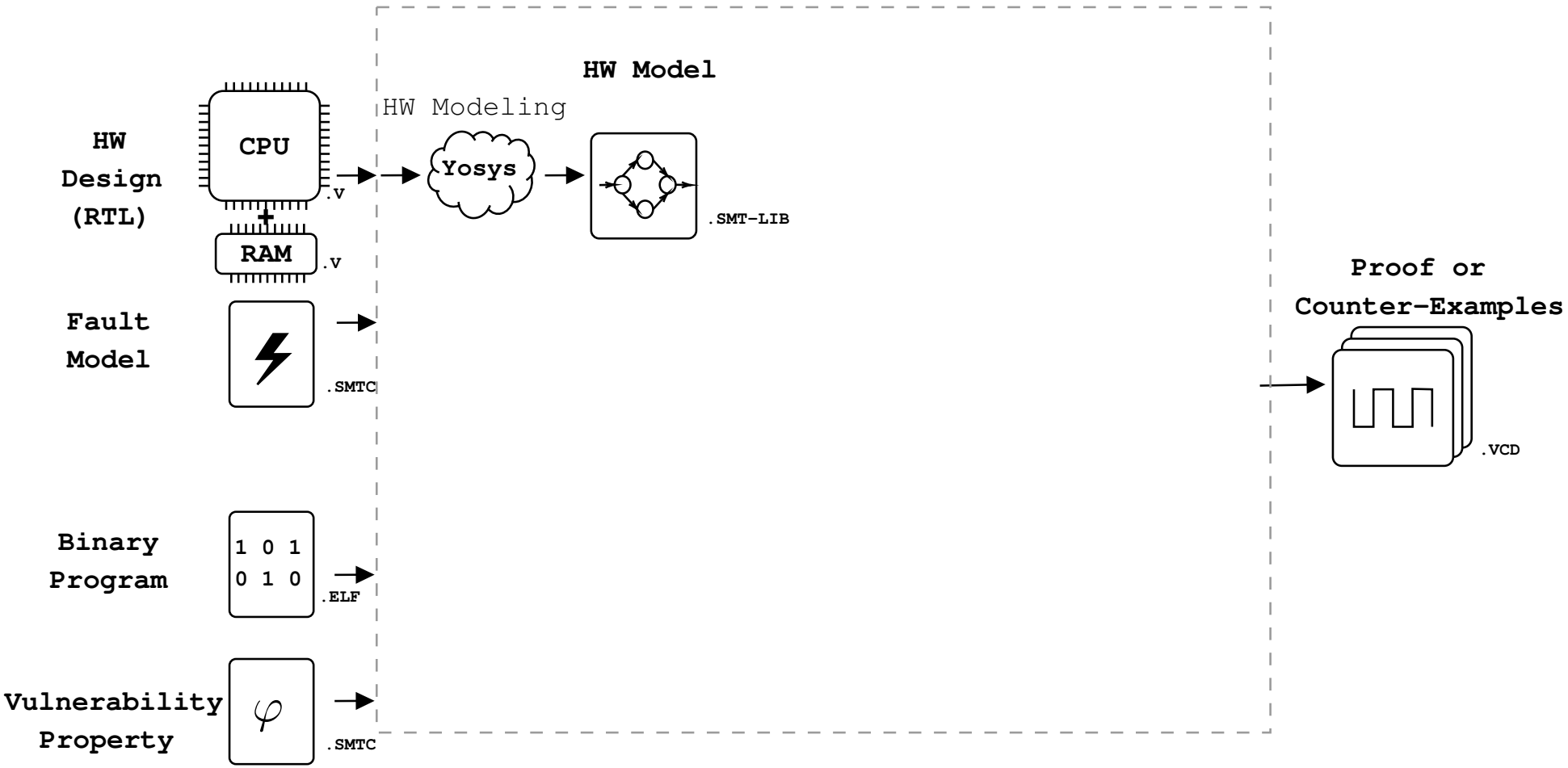
Workflow: Modeling Steps

Inputs / Outputs



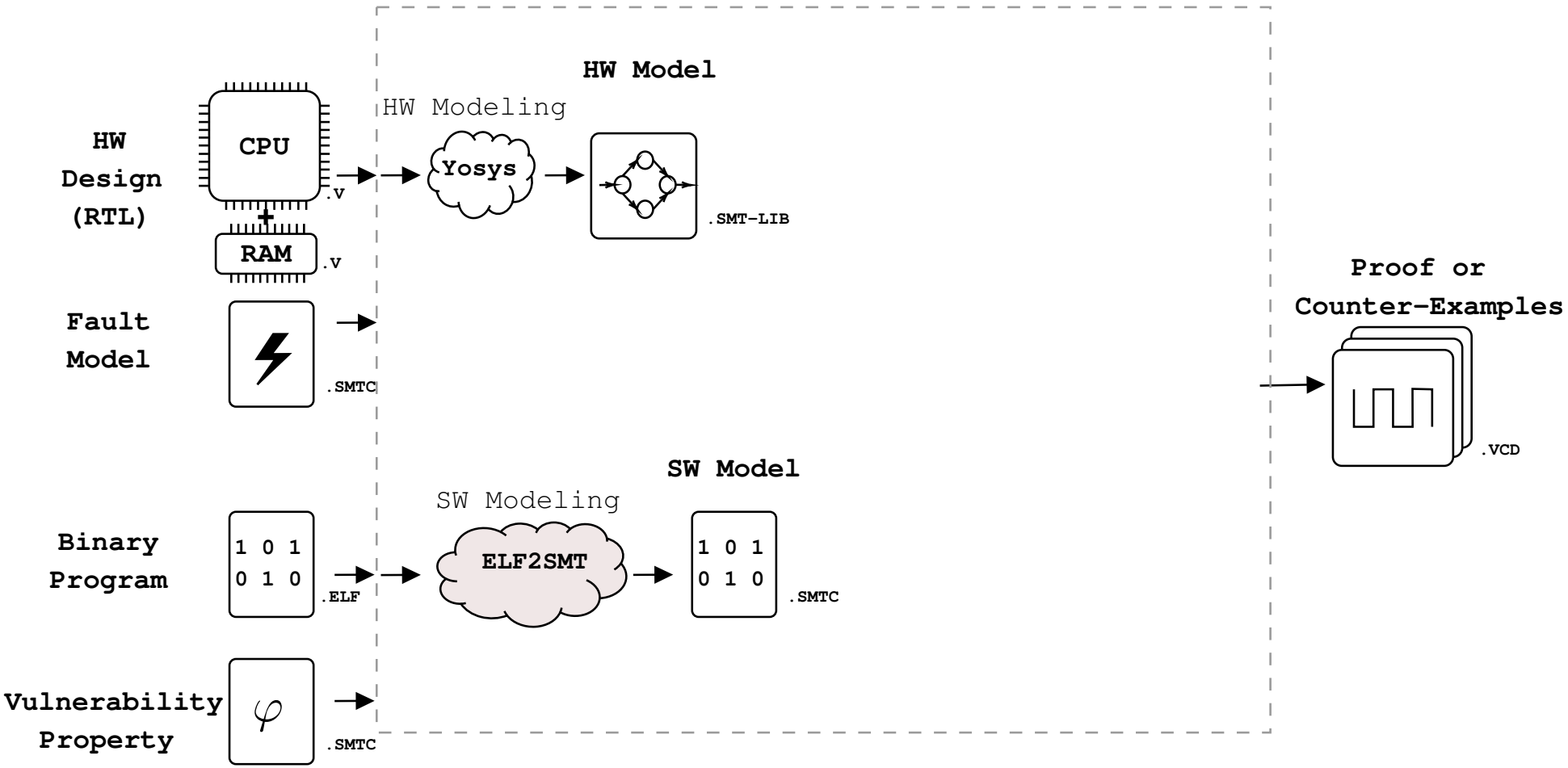
Workflow: Modeling Steps

Hardware Modeling



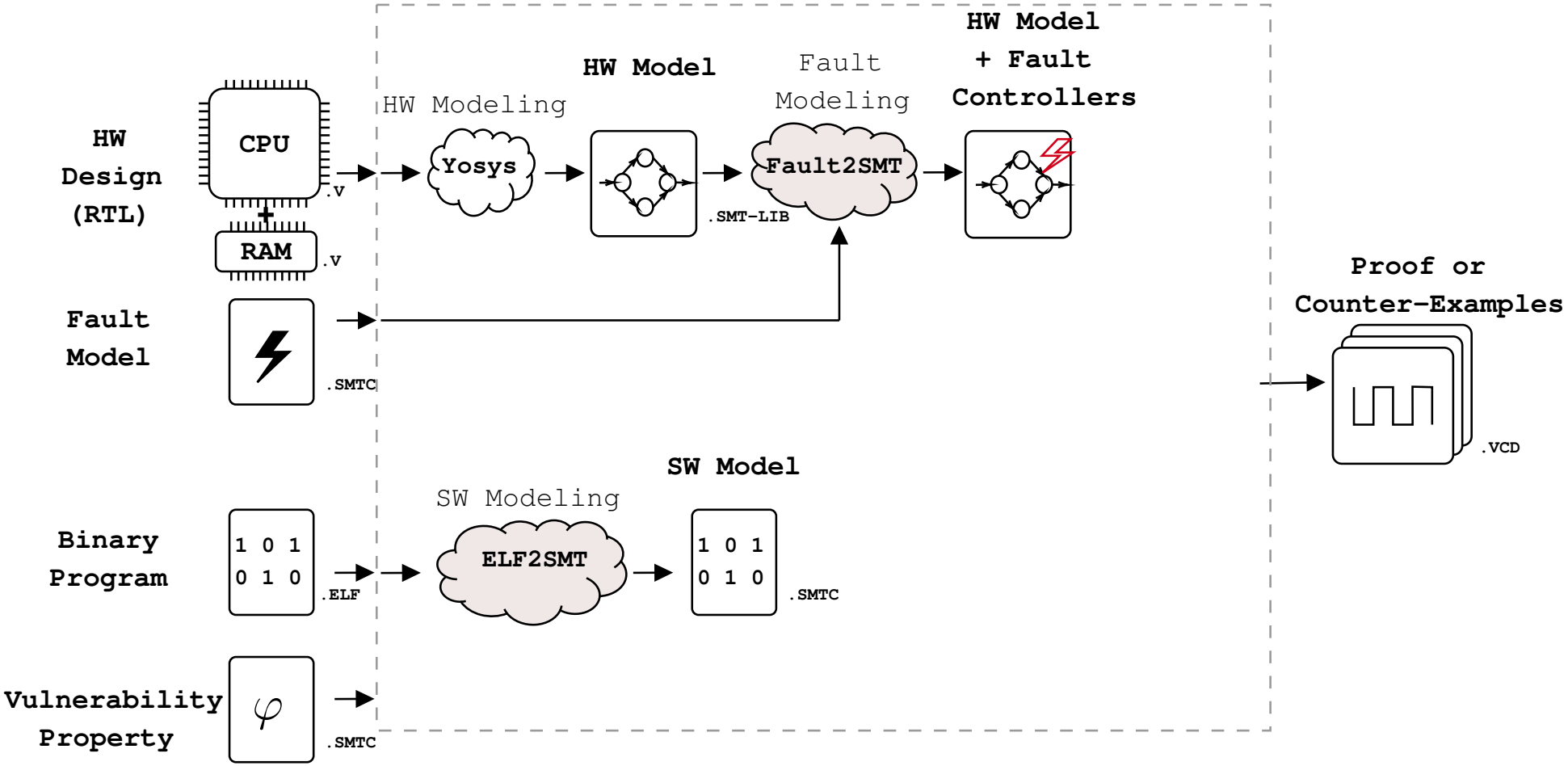
Workflow: Modeling Steps

Software Modeling



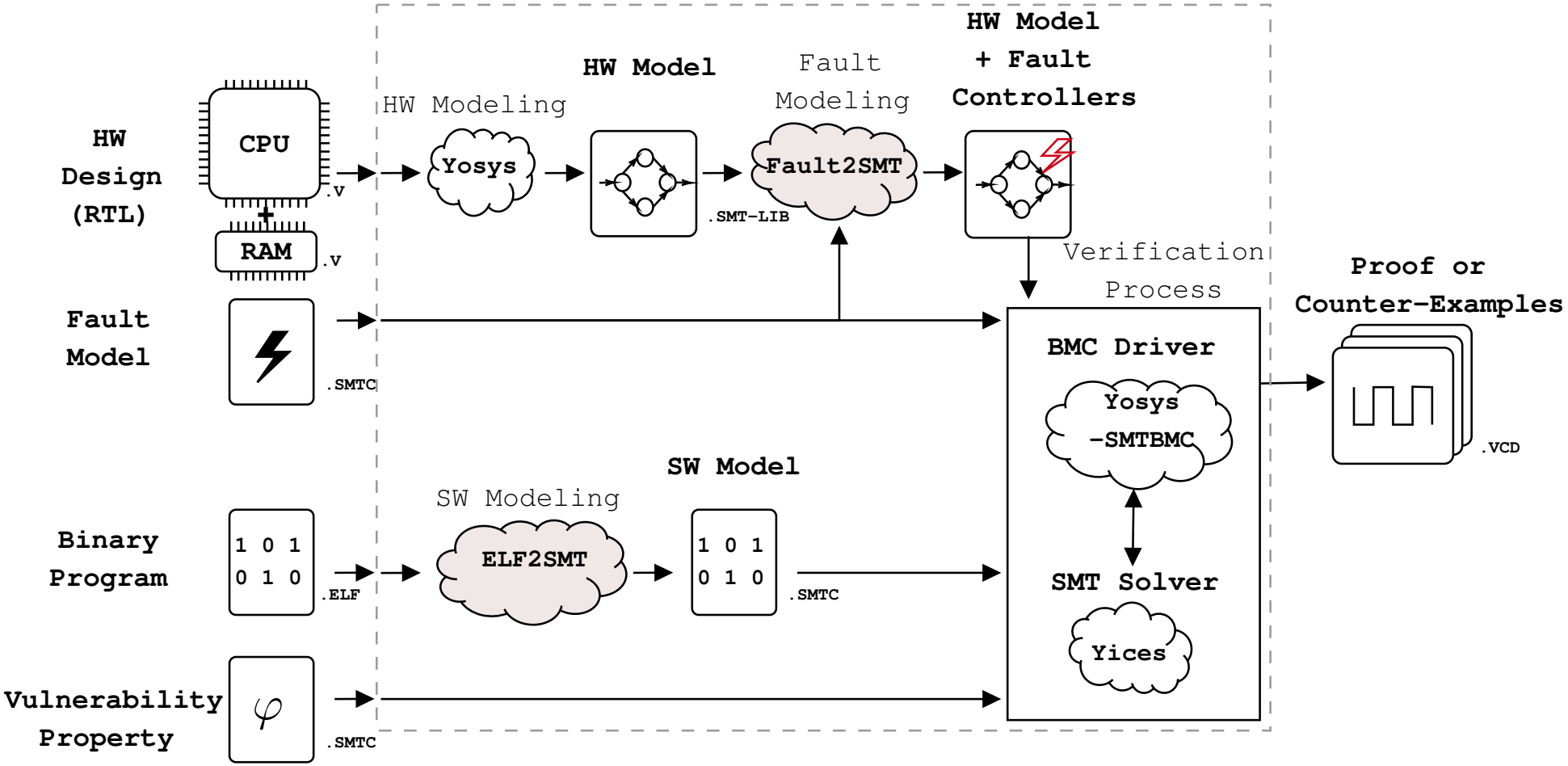
Workflow: Modeling Steps

Fault Modeling



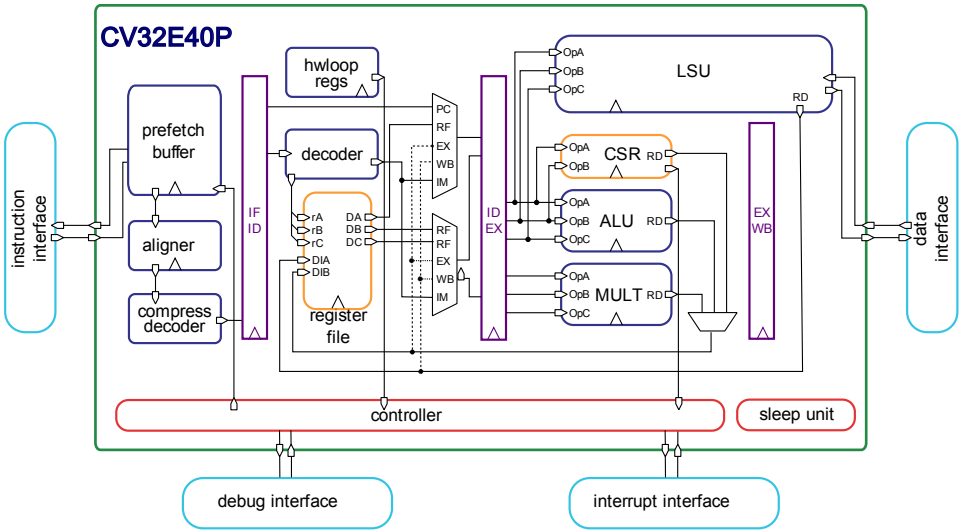
Workflow: Modeling Steps

Property Specification



Hardware Part

CV32E40P



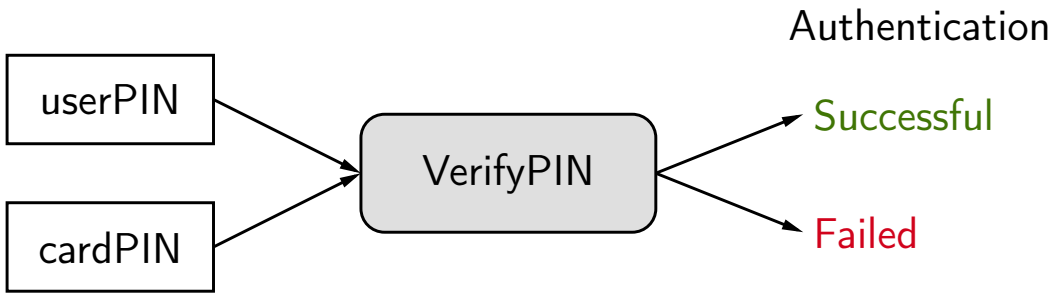
- Standard version [CV32E40P]
- Hardened version [Chamelot, 2022]
  - Control flow integrity
  - Code integrity
  - Execution integrity

Microarchitectural Fault Model

- Single fault injection
- During the whole program
- Everywhere in the circuit
- Symbolic fault effect

Software Part

VerifyPIN




```
Compare {
   for ( i = 0 ; i < 4 ; i ++) {
      if(userPIN[i] != cardPIN[i])
         return false;
   return true;
}

VerifyPIN {
   authentification = false;
   if (tries > 0) {
      if(Compare()) {
         tries = 3 ;
         authentification = true;
      } else {
         tries -- ;
      }
   }
}
```

- Standard version [Dureuil (FISSC), 2016]
- Versions implementing SW countermeasures
  - Constant iteration number loop
  - Inline function calls
  - Duplication of critical tests

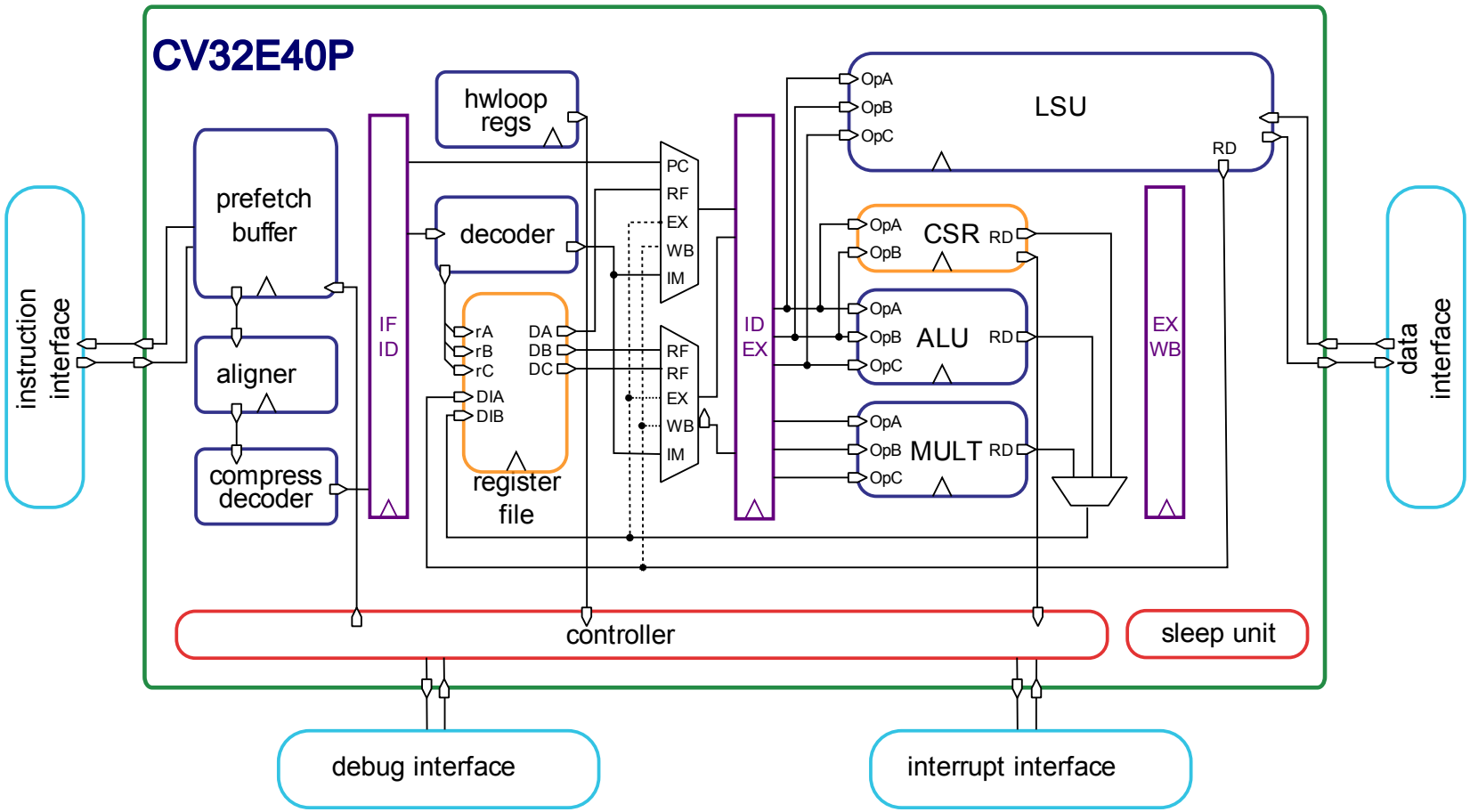
Security Property

- $\texttt{userPIN} \neq \texttt{cardPIN} \implies \neg\, \texttt{authenticated}\ (\vee\ \texttt{detected\_attack})$

Fault Effects Exploration Results

The forwarding mechanism (known attack [Laurent, 2019])

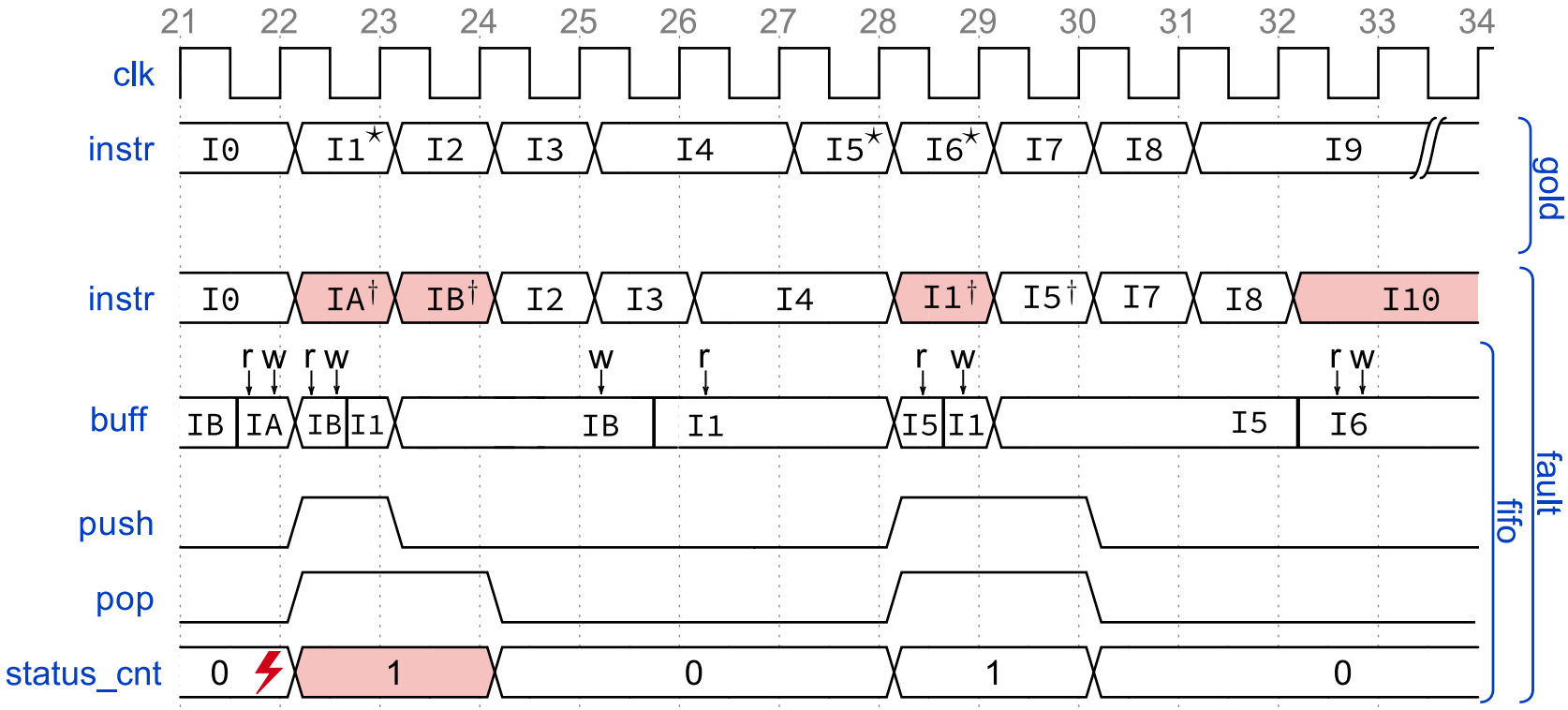
- Retrieve sensitive last-read data from the memory
- Invert conditional branches



Fault Effects Exploration Results

Fault in the Prefetch Buffer

- **Immediate one-time effect**, e.g., replay the Prefetch Buffer instructions
- **Immediate recurring effect**, e.g., incorrect order of the (replayed) instructions
- **Long-term effect**, e.g., corruption of the next branch target



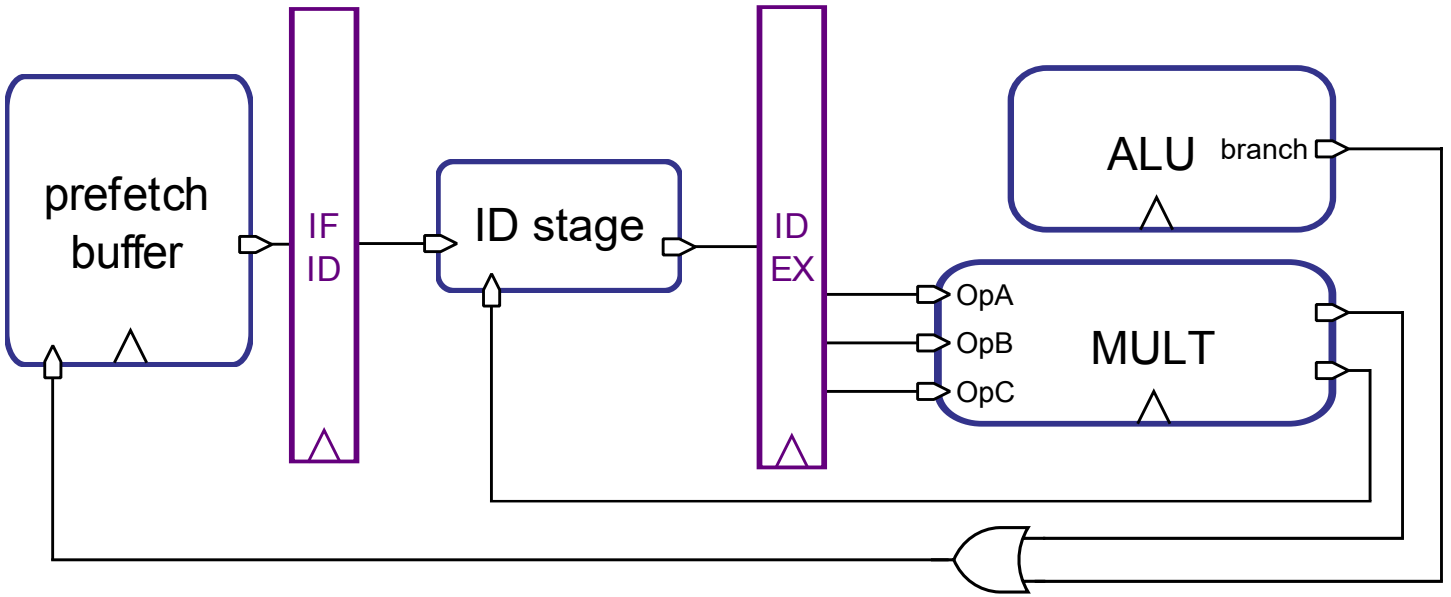
➔ *Fault effects depend on the microarchitectural details and the execution context*

Fault Effects Exploration Results

Fault in the Multiplier

- When a multi-cycle multiplication is in progress, other stages are stalled
- When a branch address is calculated in the ALU, the IF stage cannot be stalled by the EX stage

➔ Activating the ALU and MULT at the same time will result in instructions being ignored



Robustness Analysis Results

Baseline CV32E40P + VerifyPIN with the most countermeasures

- No fault injection permits bypassing the secure authentication were detected

Hardened CV32E40P + unprotected VerifyPIN

ϕ_0 VerifyPIN authentication succeeds without triggering any software countermeasures.

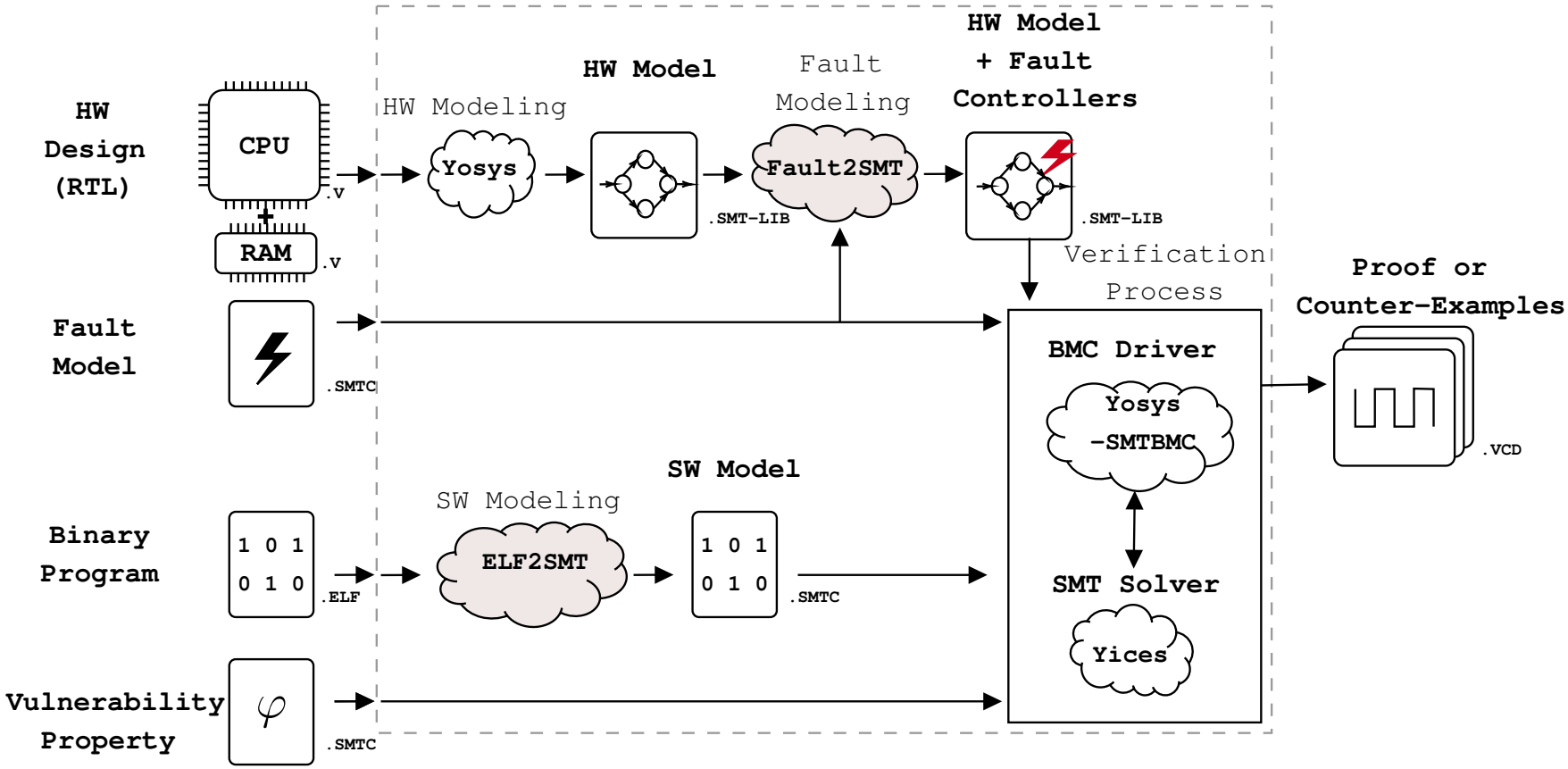
ϕ_1 Faults applied upstream from the pipeline state lead to an alteration of the pipeline state.

ϕ_2 Faults applied downstream from the pipeline state are detected by the redundancy mechanism, i.e., raise the alarm signal.

- No fault injection permits bypassing the secure authentication
- The hardware countermeasure is effective

Use Case	Overall Run Time (h)	# Fault Injections	Fault Effects	Program Length	userPIN & cardPIN (32 bits)
`Baseline CV32E40P`	12.9	15240	Symbolic	70 instr	Symbolic
`Hardened CV32E40P`	25.0	22640	Symbolic	120 instr	Symbolic

Questions ?


HW Design (RTL)
CPU
.v
RAM
.v
HW Modeling
Yosys
HW Model
.SMT-LIB
Fault Modeling
Fault2SMT
HW Model + Fault Controllers
.SMT-LIB
Fault Model
.SMTC
Verification Process
BMC Driver
Yosys -SMTBMC
SMT Solver
Yices
Proof or Counter-Examples
.VCD
Binary Program
1 0 1
0 1 0
.ELF
SW Modeling
ELF2SMT
SW Model
1 0 1
0 1 0
.SMTC
Vulnerability Property
φ
.SMTC

References I

[Tollec, FDTC 2022] S. Tollec et al. (2022, to appear)
Exploration of Fault Effects on Formal RISC-V Microarchitecture Models
Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)

[Riviere, 2015] L. Riviere et al. (2015)
High precision fault injections on the instruction cache of ARMv7-M architectures
IEEE International Symposium on Hardware Oriented Security and Trust (HOST)

[Moro, 2013] N. Moro et al. (2013)
Electromagnetic fault injection: towards a fault model on a 32-bit microcontroller
Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)

[Menu, 2020] A. Menu et al. (2020)
Experimental analysis of the electromagnetic instruction skip fault model
15th Design & Technology of Integrated Systems in Nanoscale Era (DTIS)

[Trouchkine, 2020] T. Trouchkine et al. (2020)
Fault Injection Characterization on Modern CPUs
Information Security Theory and Practice

References II

[Laurent, 2018] J. Laurent et al. (2018)
On the importance of analysing microarchitecture for accurate software fault models
21st Euromicro Conference on Digital System Design (DSD)

[Laurent, 2019] J. Laurent et al. (2019)
Fault Injection on Hidden Registers in a RISC-V Rocket Processor and Software Countermeasures
Design, Automation & Test in Europe Conference & Exhibition (DATE)

[Proy, 2019] J. Proy et al. (2019)
A first ISA-level characterization of EM pulse effects on superscalar microarchitectures: a secure software perspective
Proceedings of the 14th International Conference on Availability, Reliability and Security

[CV32E40P] OpenHW group
CORE-V CV32E40P User Manual
`https://cv32e40p.readthedocs.io/en/latest/intro/`

[Chamelot, 2022] T. Chamelot et al. (2022)
SCI-FI: control signal, code, and control flow integrity against fault injection attacks
Design, Automation & Test in Europe Conference & Exhibition (DATE)

References III

[Dureuil (FISSC), 2016] L. Dureuil et al. (2016)
FISSC: A fault injection and simulation secure collection
International Conference on Computer Safety, Reliability, and Security

[Hoffmann, (ARMORY) 2021] M. Hoffmann et al. (2021)
ARMORY: Fully Automated and Exhaustive Fault Simulation on ARM-M Binaries
IEEE Transactions on Information Forensics and Security

[Nasahl (SYNFI), 2022] P. Nasahl et al. (2022)
SYNFI: Pre-Silicon Fault Analysis of an Open-Source Secure Element
IACR Transactions on Cryptographic Hardware and Embedded Systems (CHES)