Design of Access Control Mechanisms in SoCs with Formal Integrity Guarantees

USENIX Security Symposium ’23

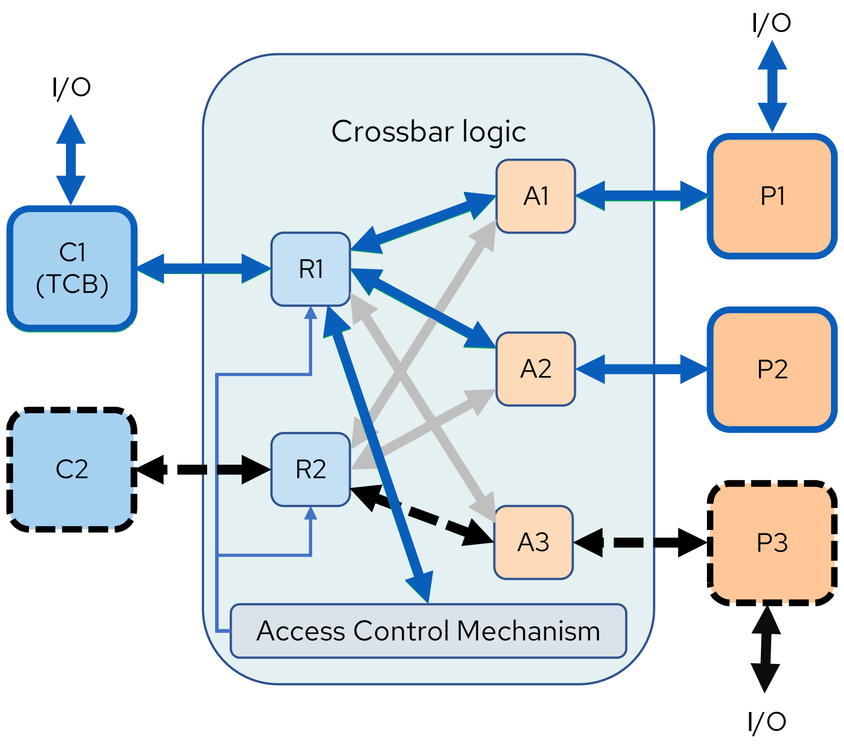
2023-08-10

Dino Mehmedagić, Mohammad Rahmani Fadiheh, Johannes Müller, Anna Lena Duque Antón, Dominik Stoffel, Wolfgang Kunz

Rheinland-Pfälzische Technische Universität Kaiserslautern Landau

Threat Model

- Increasing need for SoCs with diversified hardware
- Third-party IPs → trust issues ☹
- SoC Access Control Mechanism
  - Domains: High-security vs low-security
  - Access control ensures that communication between domains doesn't endanger security



RPTU

Security Target

> **Operation integrity:**

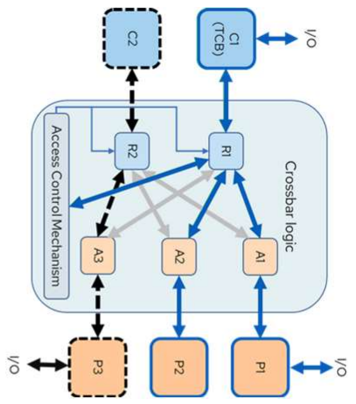
> Forbidden information flow: low security domain → high security domain outputs

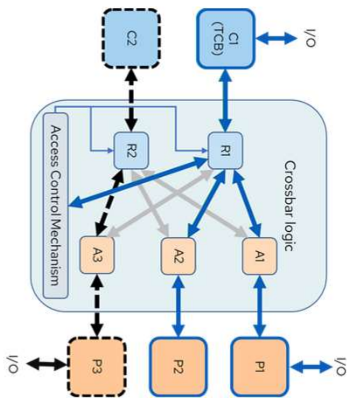
> **UPEC:**

> *Exhaustive* verification of information flow restrictions at the RTL

> Interval Property Checking (IPC)

> 2-instance (miter) model

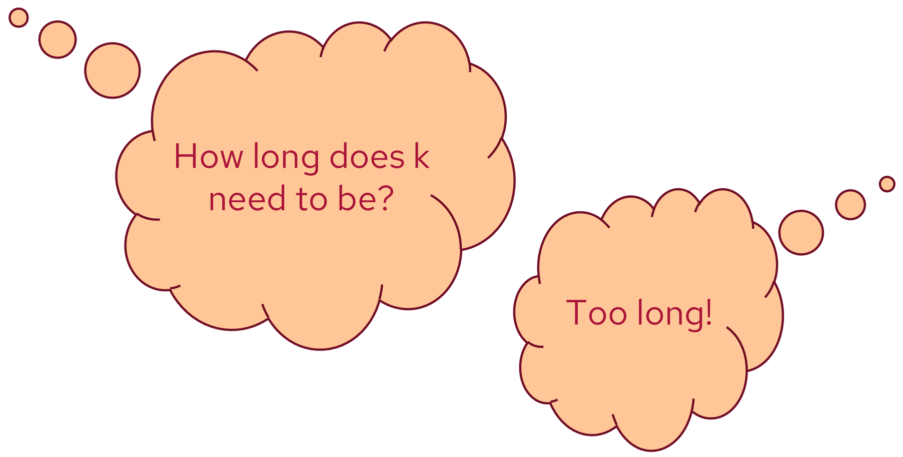


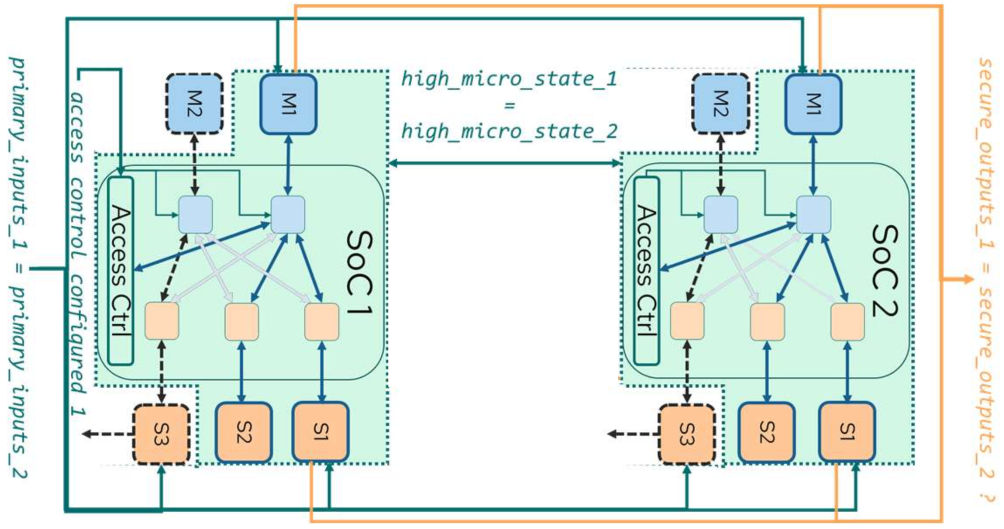


RPTU

UPEC for Operation Integrity

```
UPEC-OI:
assume
 t₀     high_micro_state₁ = high_micro_state₂;
 t₀..tₖ primary_inputs₁ = primary_inputs₂;
 t₀..tₖ access_ctrl_configured₁;
prove
 tₖ     secure_outputs₁ = secure_outputs₂
```

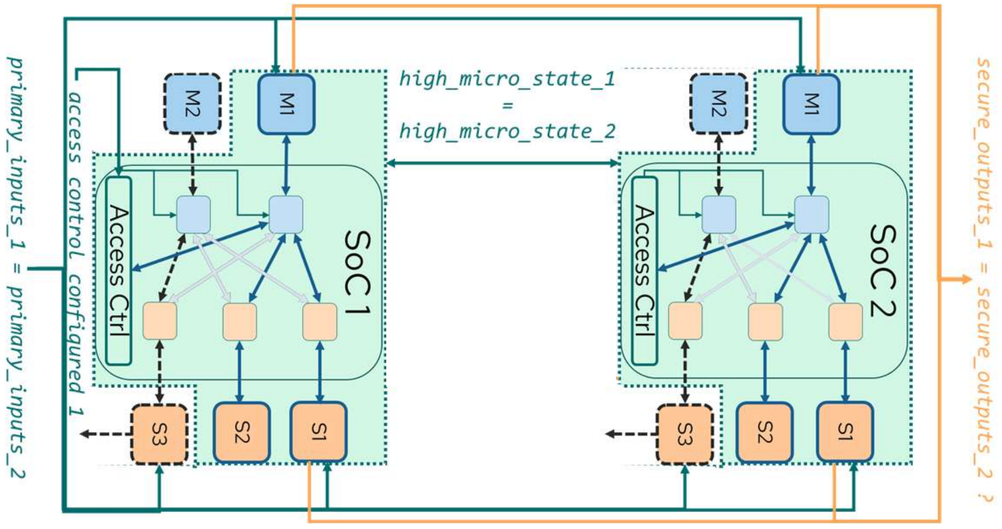



primary_inputs_1 = primary_inputs_2
access_control_configured_1
high_micro_state_1 = high_micro_state_2
secure_outputs_1 = secure_outputs_2 ?
M1
M2
S1
S2
S3
Access Ctrl
SoC 1
SoC 2

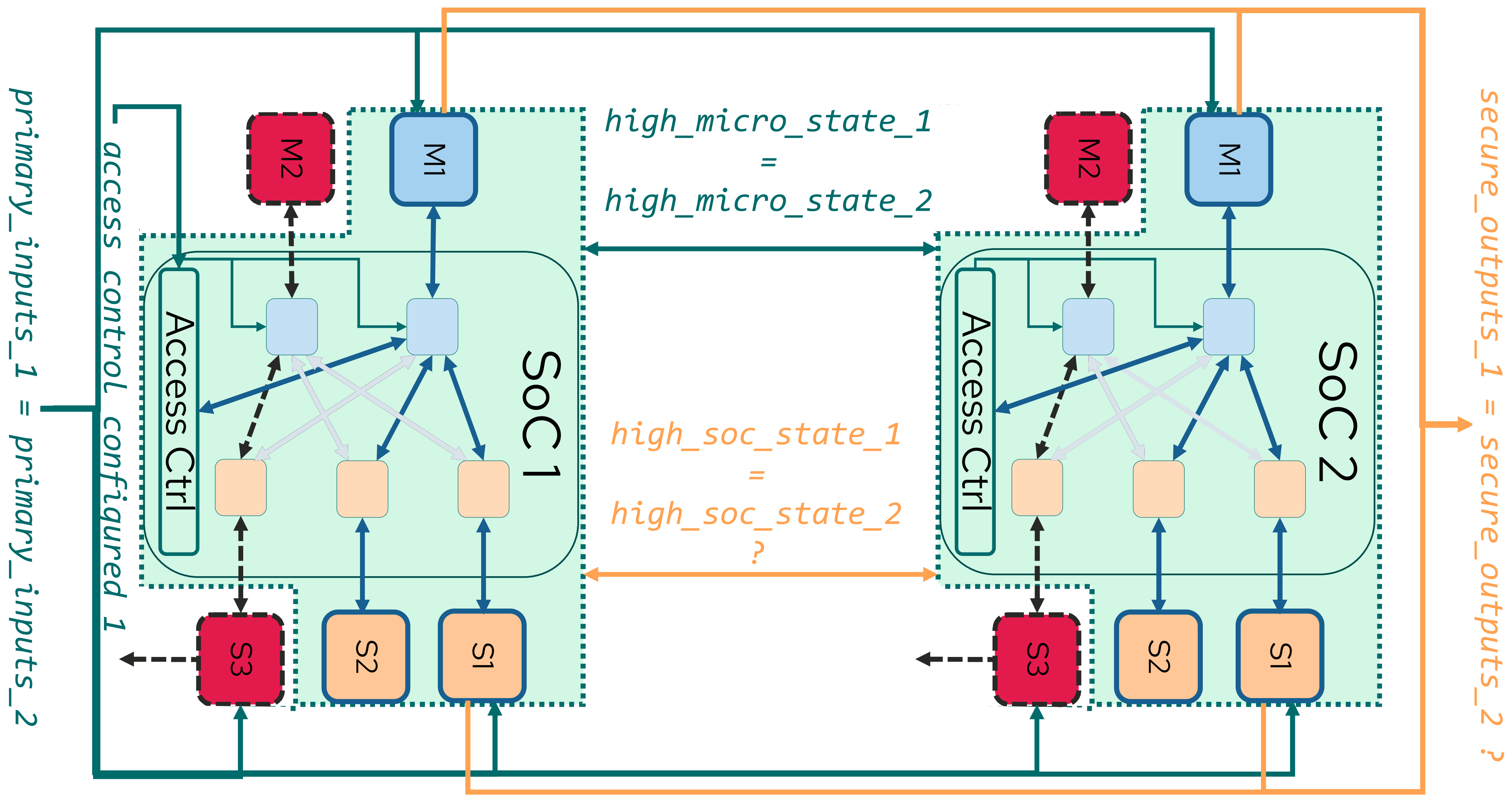
RPTU

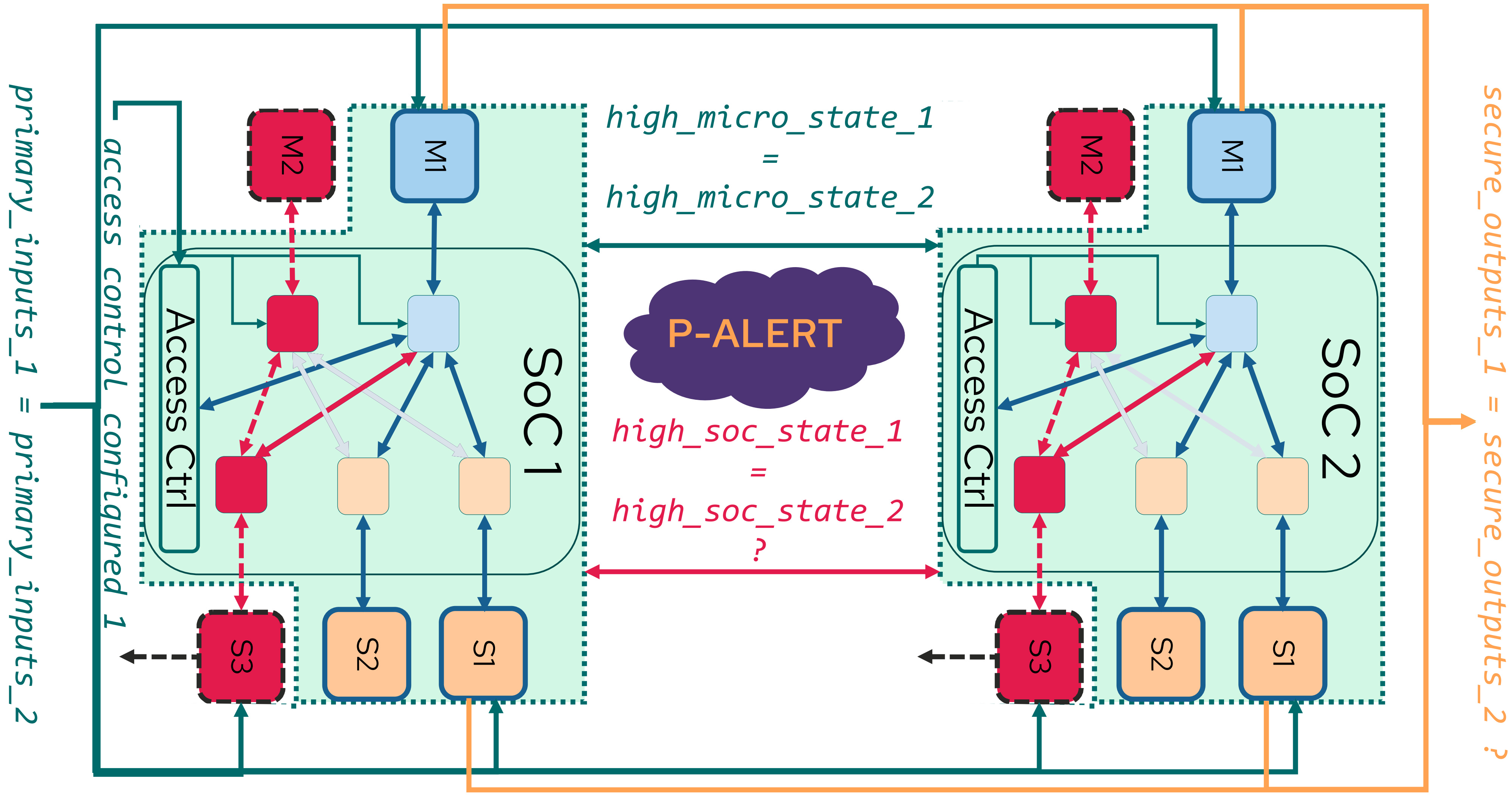
Decomposing the Proof

```
UPEC-OI:
assume
 t₀     high_micro_state₁ = high_micro_state₂;
 t₀..tₖ primary_inputs₁ = primary_inputs₂;
 t₀..tₖ access_ctrl_configured₁;
prove
 tₖ     secure_outputs₁ = secure_outputs₂
 tₖ     high_soc_state₁ = high_soc_state₂;
```

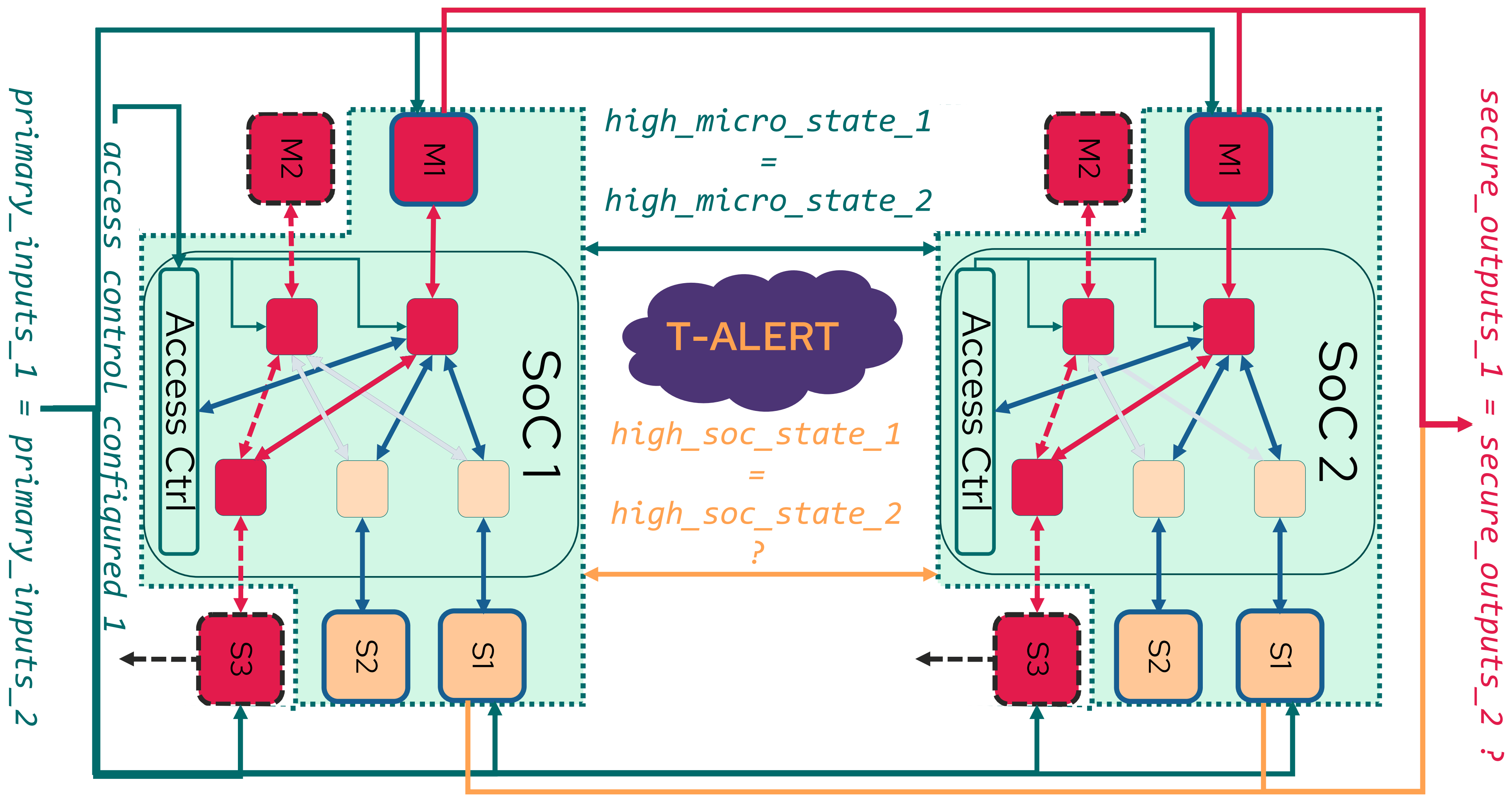

primary_inputs_1 = primary_inputs_2
access control configured 1
Access Ctrl
M2
M1
SoC 1
S3
S2
S1
high_micro_state_1
=
high_micro_state_2
Access Ctrl
M2
M1
SoC 2
S3
S2
S1
secure_outputs_1 = secure_outputs_2 ?

RPTU


primary_inputs_1 = primary_inputs_2
access control configured 1
M2
M1
Access Ctrl
SoC 1
S3
S2
S1
high_micro_state_1
=
high_micro_state_2
high_soc_state_1
=
high_soc_state_2
?
M2
M1
Access Ctrl
SoC 2
S3
S2
S1
secure_outputs_1 = secure_outputs_2 ?


primary_inputs_1 = primary_inputs_2
access control configured 1
M2
M1
Access Ctrl
SoC 1
S3
S2
S1
high_micro_state_1
=
high_micro_state_2
P-ALERT
high_soc_state_1
=
high_soc_state_2
?
M2
M1
Access Ctrl
SoC 2
S3
S2
S1
secure_outputs_1 = secure_outputs_2 ?

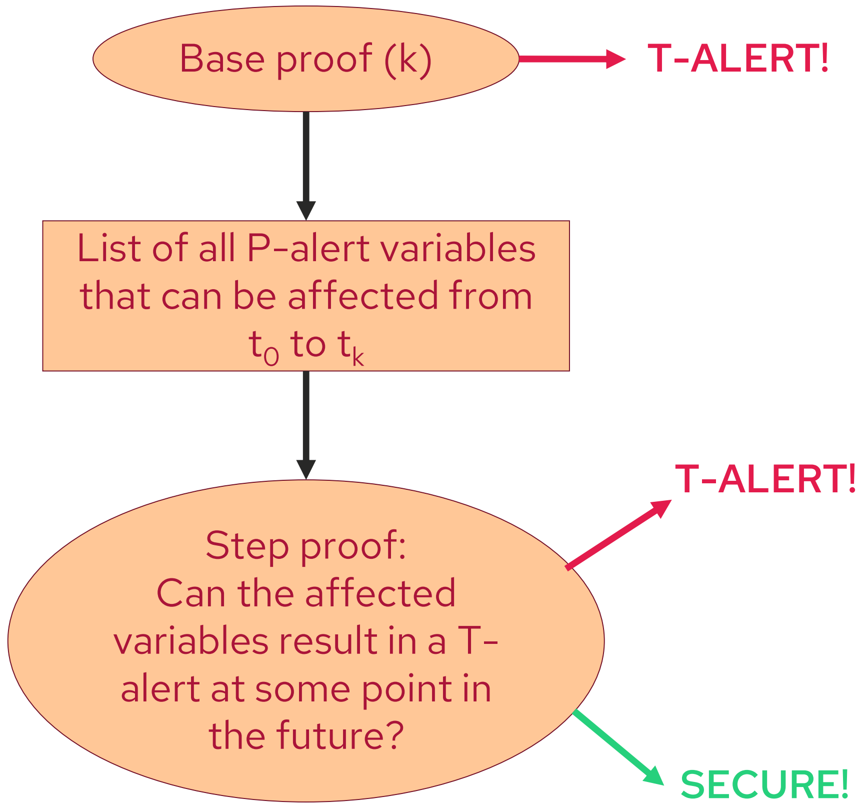
RPTU



primary_inputs_1 = primary_inputs_2
access control configured 1
M2
M1
Access Ctrl
SoC 1
S3
S2
S1
high_micro_state_1
=
high_micro_state_2
T-ALERT
high_soc_state_1
=
high_soc_state_2
?
M2
M1
Access Ctrl
SoC 2
S3
S2
S1
secure_outputs_1 = secure_outputs_2 ?

UPEC-OI Verification Methodology

- Induction-based approach to completely verify operation integrity
  - Base proof: Find all P-alerts and verify OI for a bounded time window *k*
  - Step proof: Use IPC's symbolic initial state to fast forward to any future time point in which a T-alert can occur
- Additional optimizations
  - Sound blackboxing
  - Spatial, temporal decomposition, T-alert trigger expansion…


Base proof (k)
T-ALERT!
List of all P-alert variables that can be affected from t_0 to t_k
Step proof: Can the affected variables result in a T-alert at some point in the future?
T-ALERT!
SECURE!

RPTU

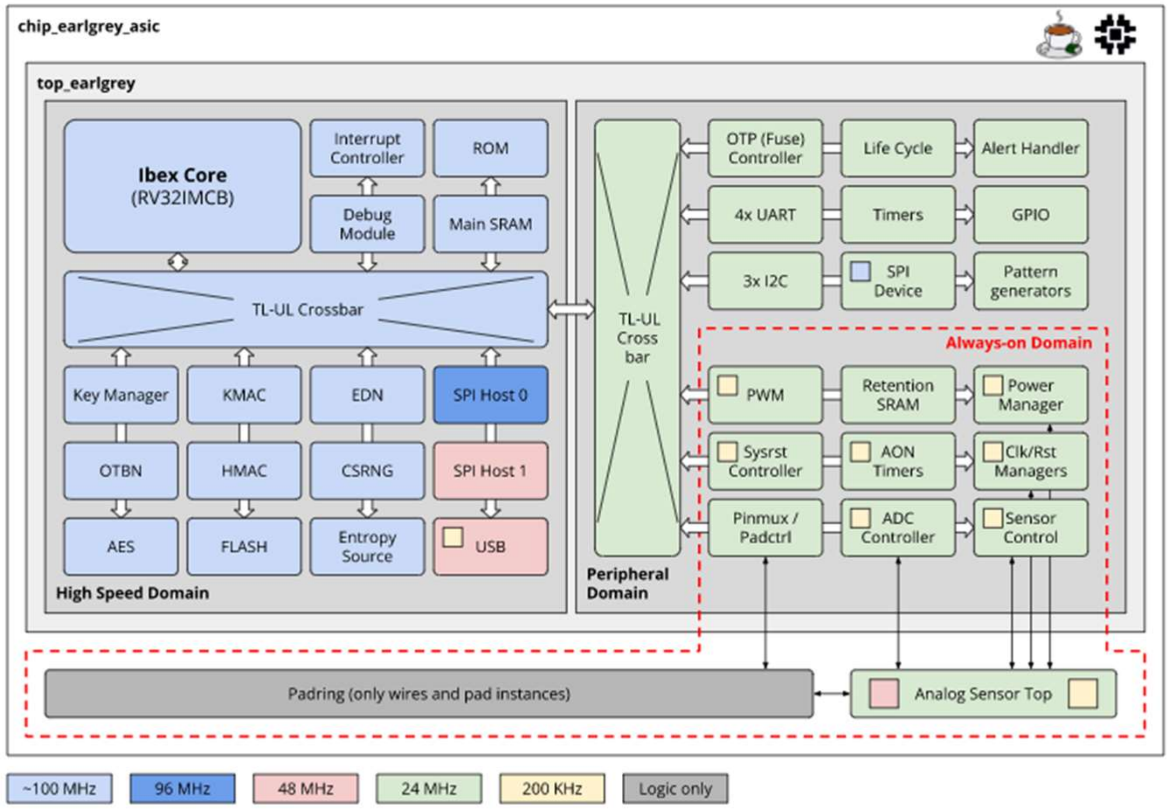
Case Study: OpenTitan

UPEC-Driven Design of Access Control

> Add malicious IPs to model the threat

> Equip SoC with access control mechanism in the interconnect

> Refine the access control mechanism through a UPEC-OI-driven design flow



RPTU

Case Study: Results

UPEC-Driven Design of Access Control

Overall design process	3 person-months
Number of verify-patch iterations	19
Average property check time	~5 minutes
Longest UPEC-OI check time	11 hours
Peak memory consumption	25 GB
Design size	14 million state bits

Conclusion

- Developed a methodology to formally verify operation integrity:
  - Property formulation
  - Proof decomposition
  - Scalability and usability optimizations
- Case study shows: UPEC-OI is feasible for realistic SoCs

More details in the paper “Design of Access Control Mechanisms in Systems-on-Chip with Formal Integrity Guarantees” – available as a preprint on https://www.usenix.org/conference/usenixsecurity23/presentation/mehmedagic

Thank you!

Questions?

Contact me at:
dino.mehmedagic@edu.rptu.de