



RUHR-UNIVERSITÄT BOCHUM

Pre-Silicon Security

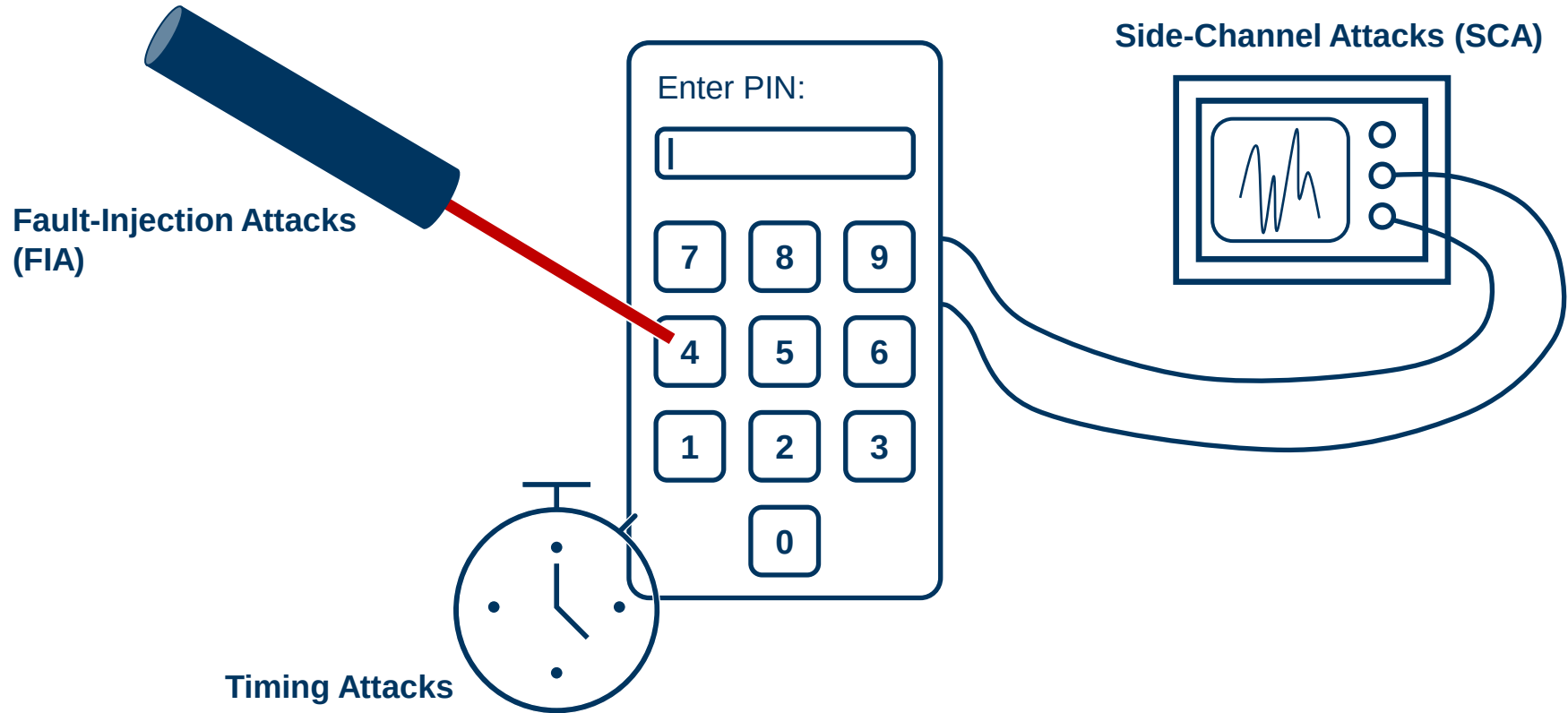
From Automated Design to Countermeasure Verification

Prof. Dr.-Ing. Tim Güneysu

7. Oktober 2026

Chair for Security Engineering
Faculty of Computer Science
Ruhr University Bochum

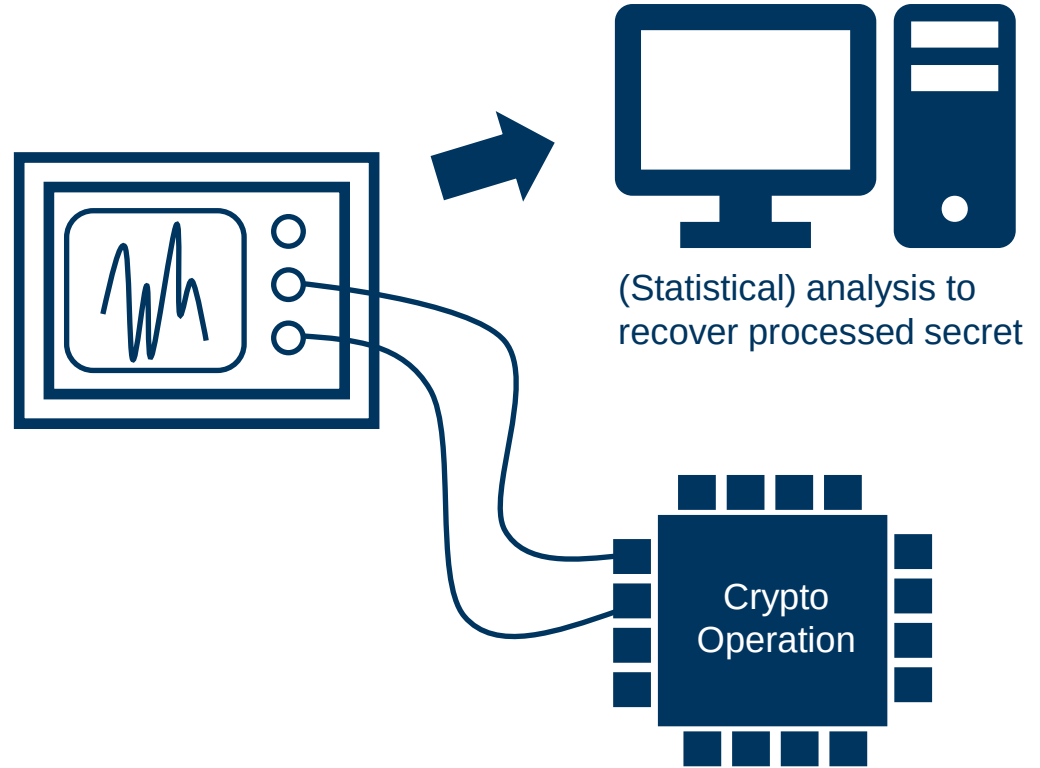
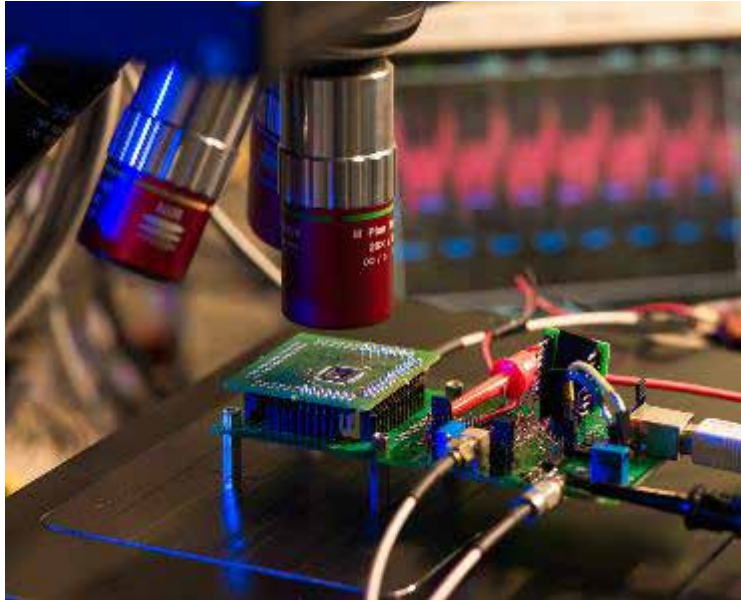




SIDE-CHANNEL ANALYSIS

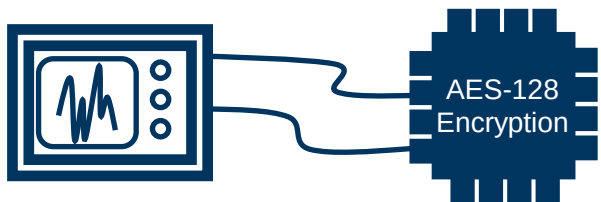


Side-Channel Attacks (Timing, Power, EM, ...)



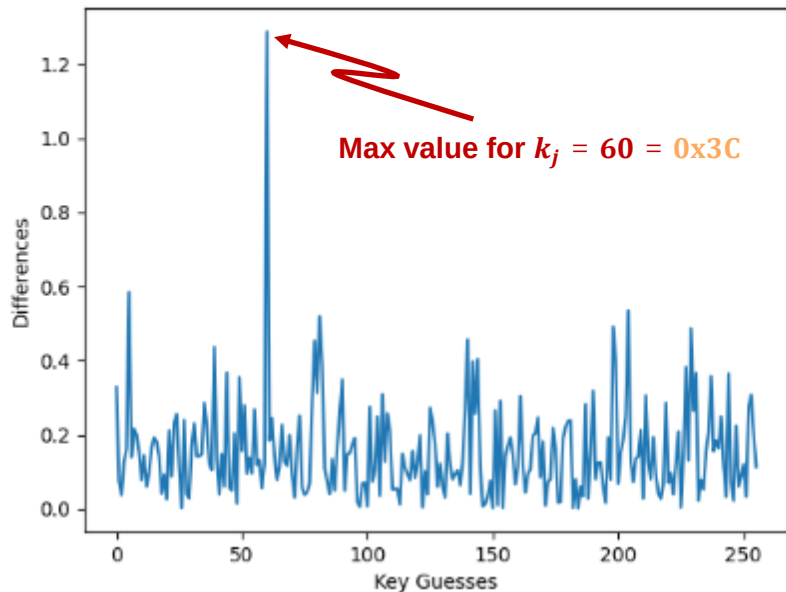
Measure power consumption during ongoing operation

Exemplary Byte-wise SCA Attack on 128-bit AES Key

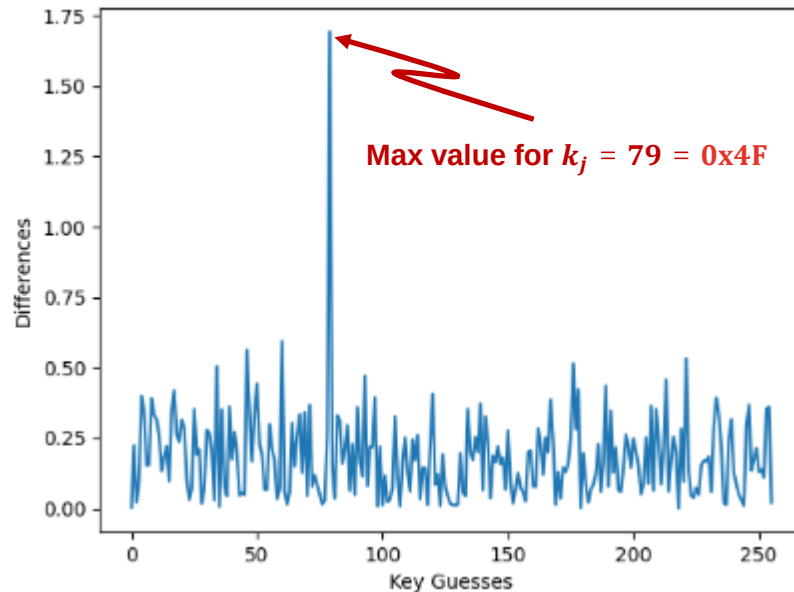


Correct Key: 0x2b7e151628aed2a6abf7158809cf4f3c

Attack on First Byte

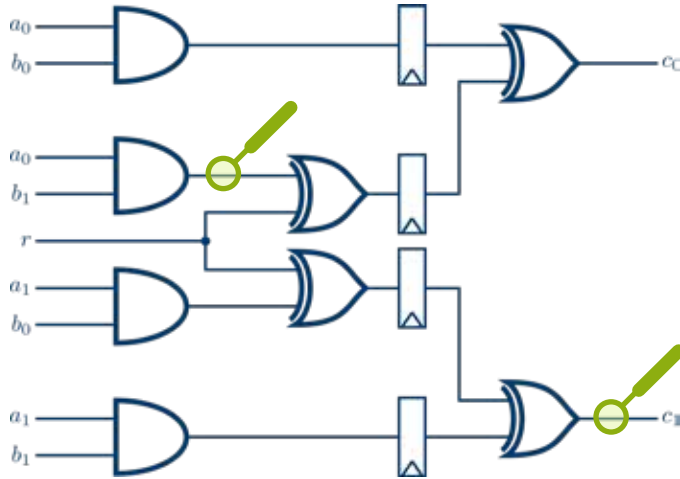


Attack on Second Byte



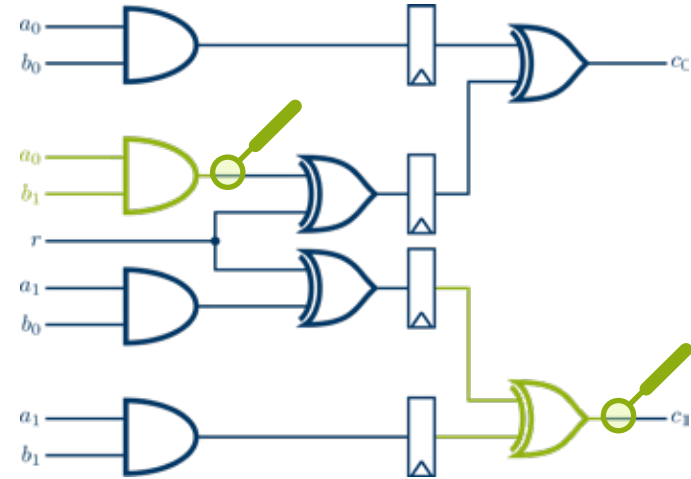
Modeling Side-Channel Attacks on Cryptographic Circuits

d -probing model [ISW03]



An adversary is given the exact values of up to d wires of a circuit C .

More elaborated models, e.g. Glitch-extended d -probing model [FGP+18]

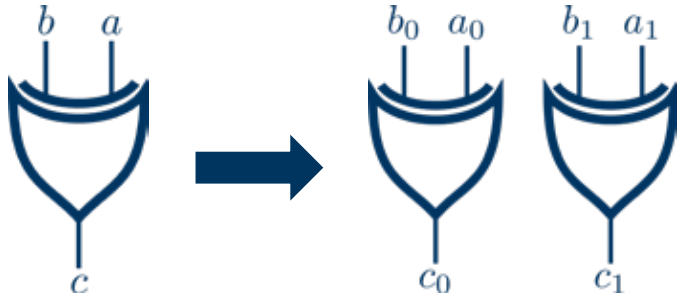


An adversary is given the exact values of all synchronization points influencing up to d wires of a circuit C .

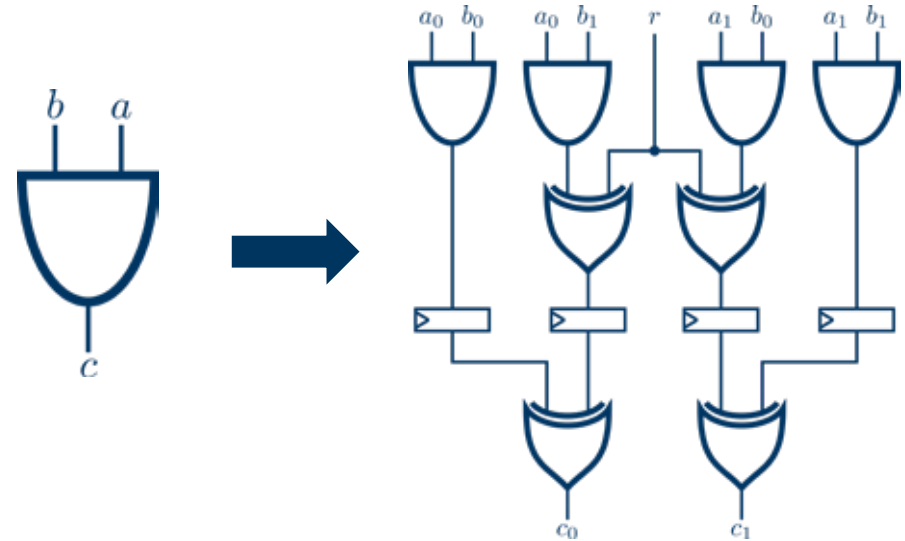
Countermeasures against Side-Channel Attacks

Masking circuits by secret sharing provides formal guarantees: $x = x_0 \oplus x_1 \oplus \dots \oplus x_{s-1}$

Linear Functions

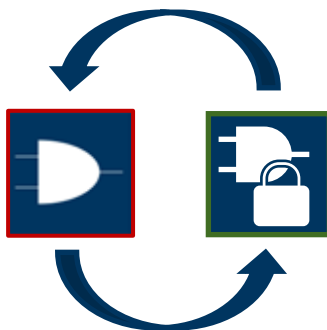
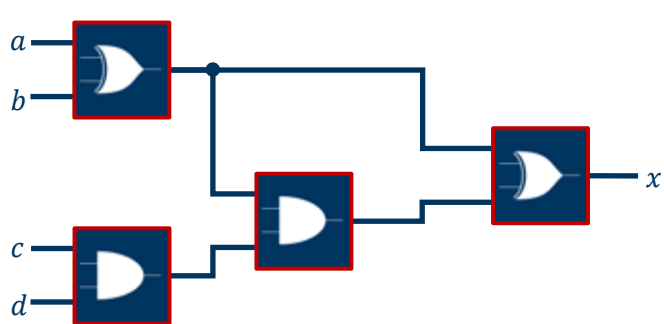


Non-linear Functions

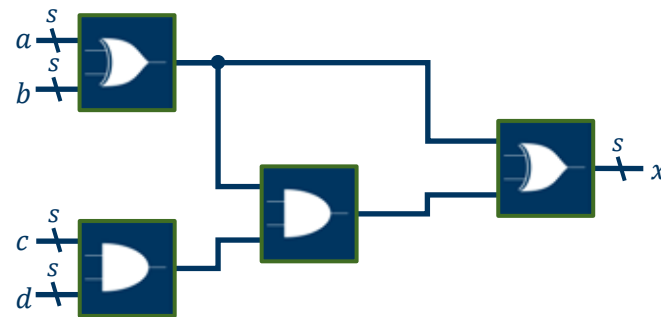


Protection by Secure Gadgets

Insecure Circuit



Protected Circuit



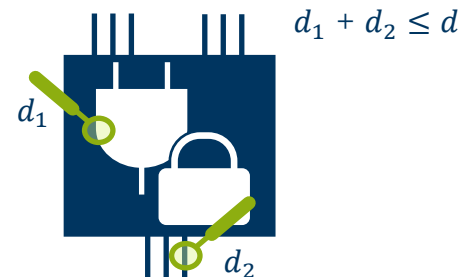
Protection: PINI [CS20]

Probe-Isolating Non-Interference

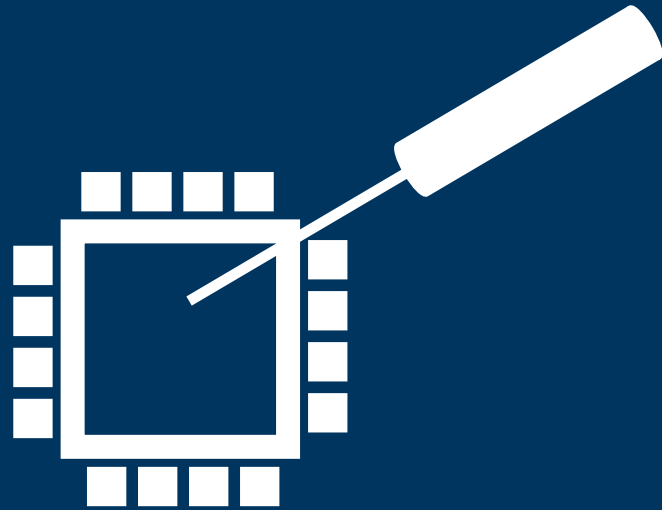
Replace insecure gates by
secure gadgets

Share inputs and outputs

Maintain timing (pipelining)

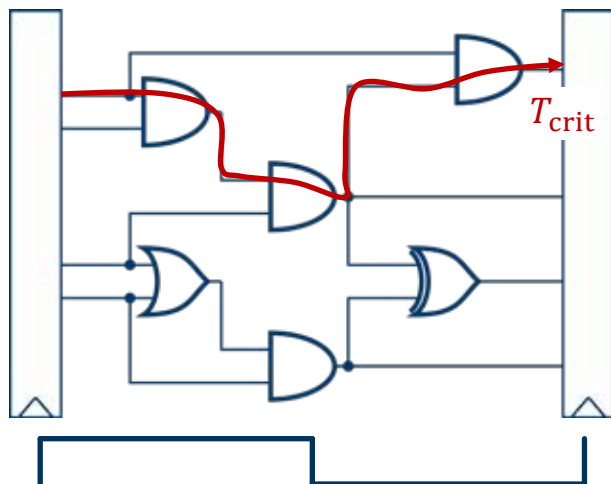


FAULT-INJECTION ATTACKS

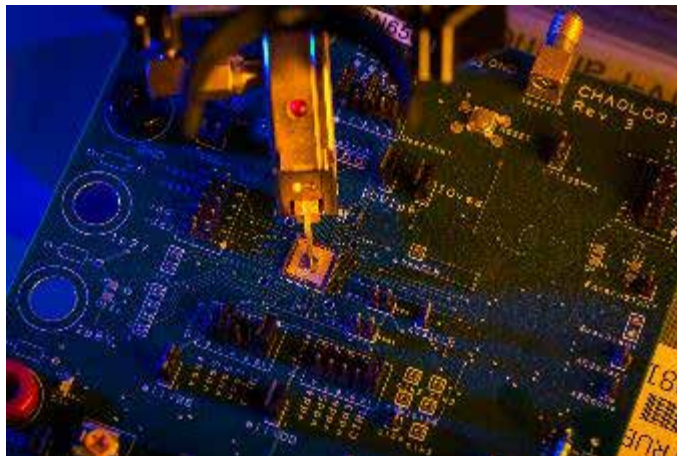
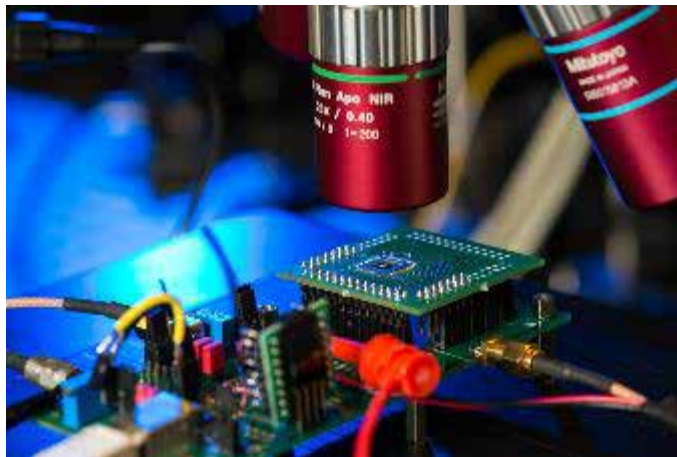


Fault-Injection Attacks [RBSG21]

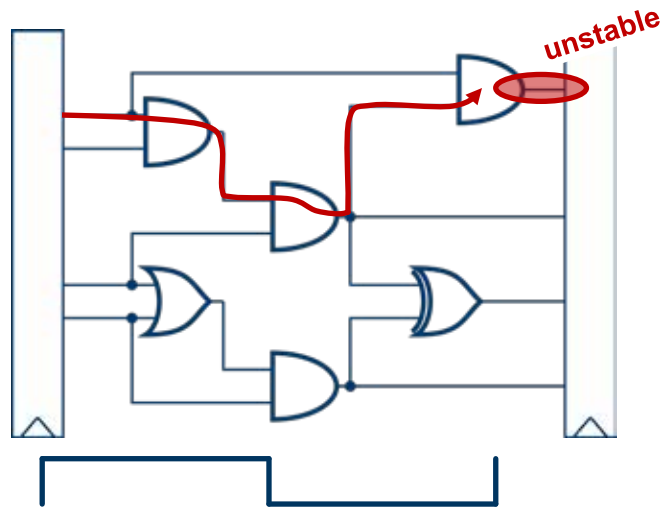
Clock Glitches



$$T_{\text{clk}} \geq T_{\text{crit}} + t_{\text{clkq}} + t_{\text{setup}} - \delta$$

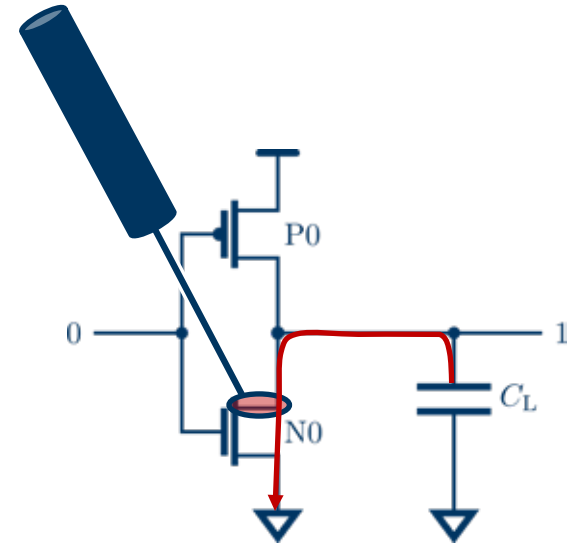


Clock Glitches

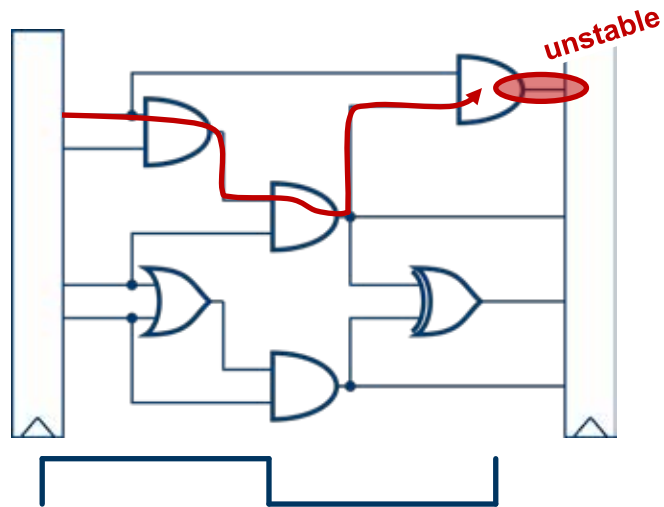


$$T'_{\text{clk}} < T_{\text{crit}} + t_{\text{clkq}} + t_{\text{setup}} - \delta$$

Laser Fault Injection

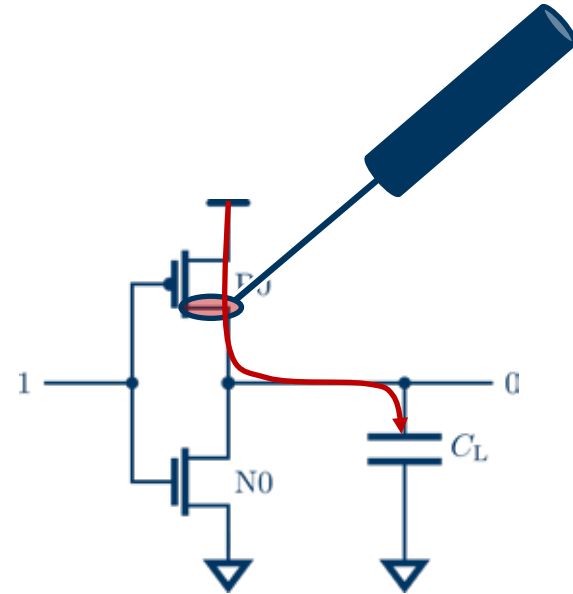


Clock Glitches



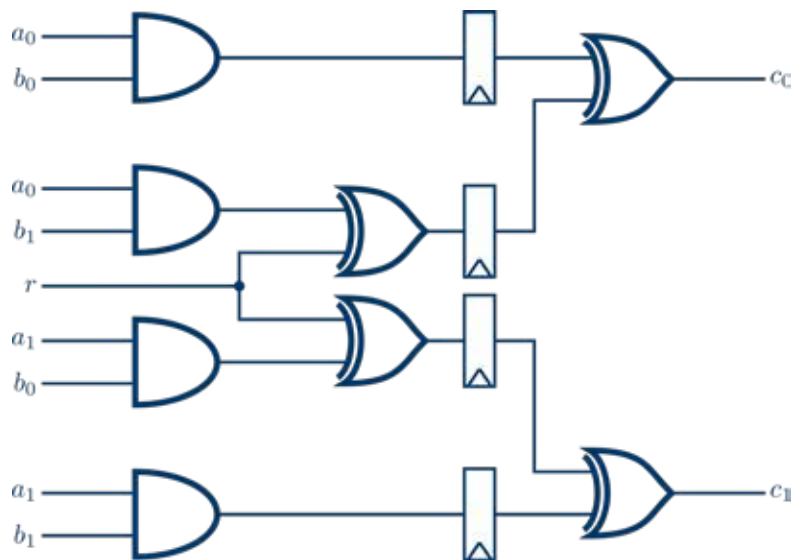
$$T'_{\text{clk}} < T_{\text{crit}} + t_{\text{clkq}} + t_{\text{setup}} - \delta$$

Laser Fault Injection

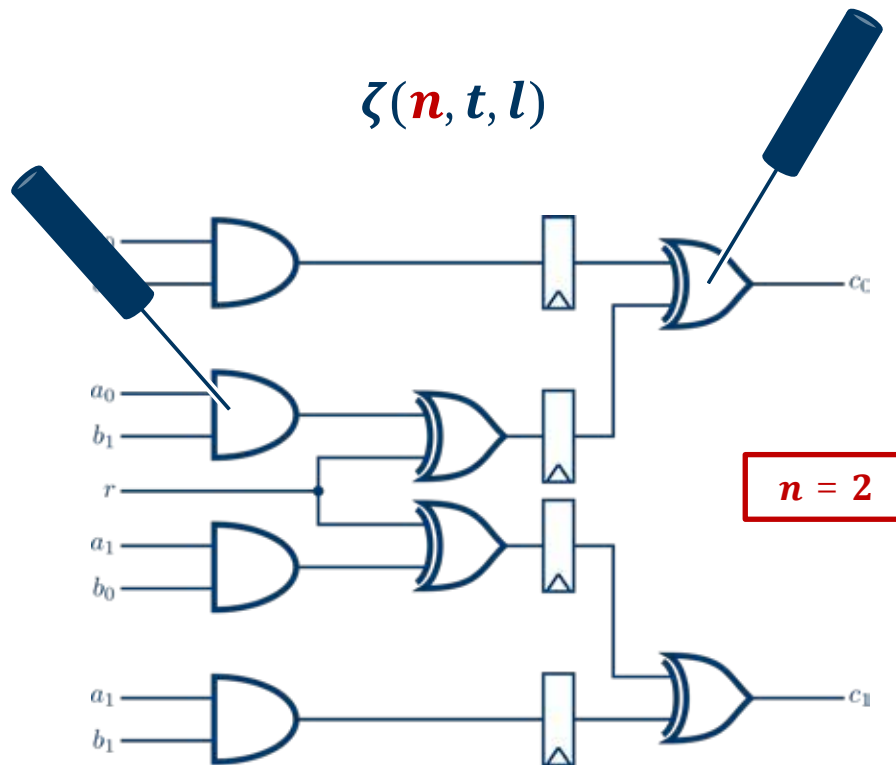


Modeling Fault-Injection Attacks [RBSG21]

$$\zeta(n, t, l)$$

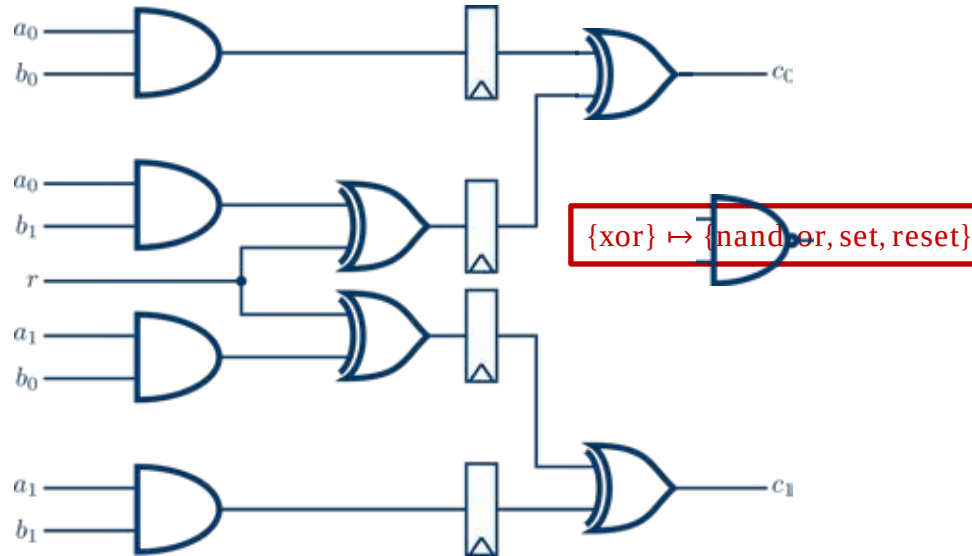


Modeling Fault-Injection Attacks [RBSG21]

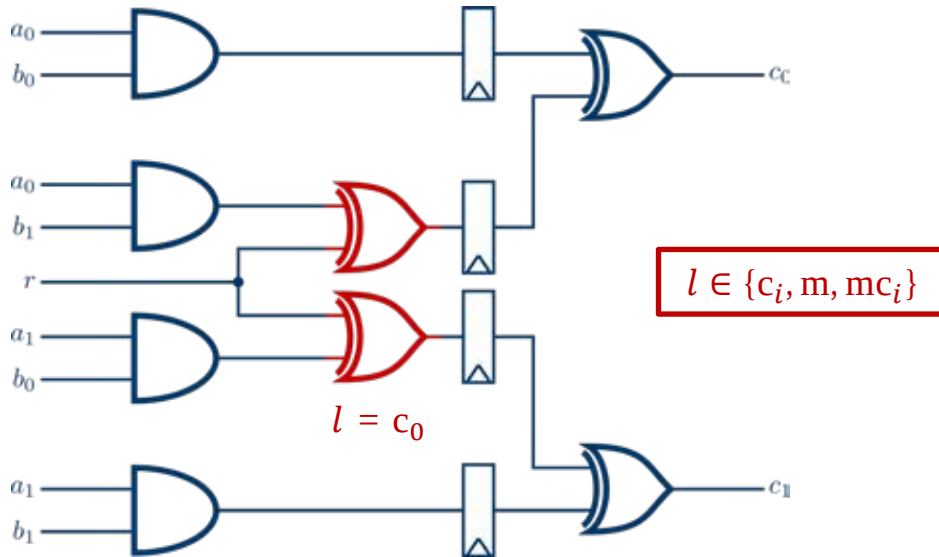


Modeling Fault-Injection Attacks [RBSG21]

$$\zeta(n, \mathbf{t}, l)$$

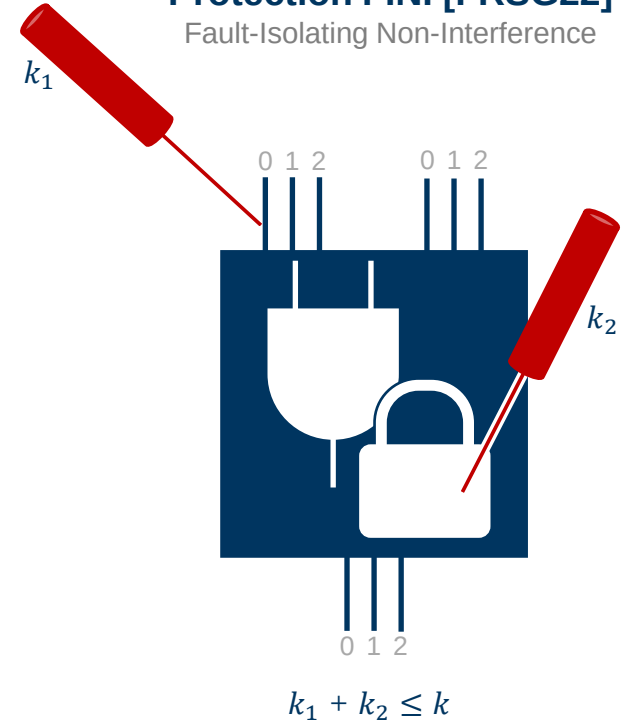


Modeling Fault-Injection Attacks [RBSG21]

 $\zeta(n, t, l)$


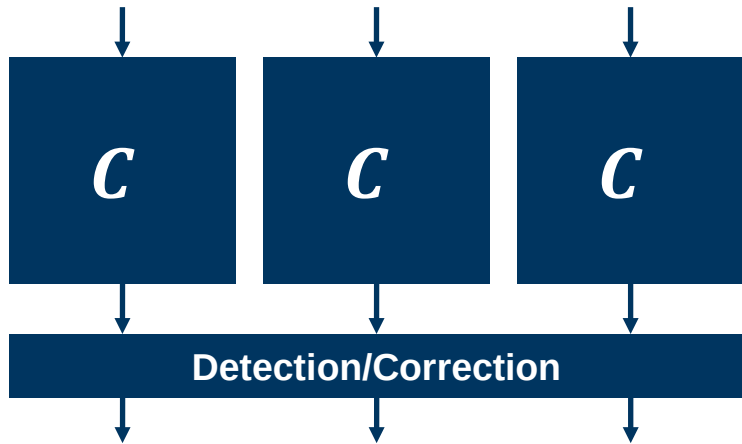
Protection FINI [FRSG22]

Fault-Isolating Non-Interference



Countermeasures against Fault-Injection Attacks

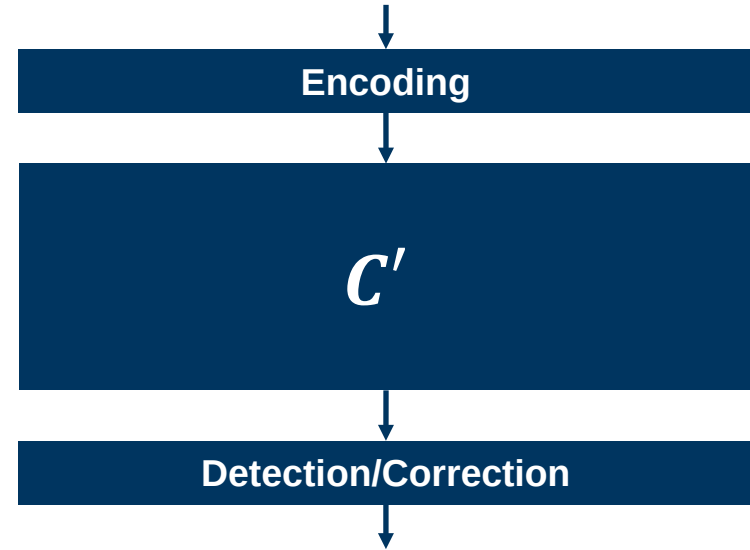
Duplication



Detection: $k + 1$ instantiations

Correction: $2k + 1$ instantiations

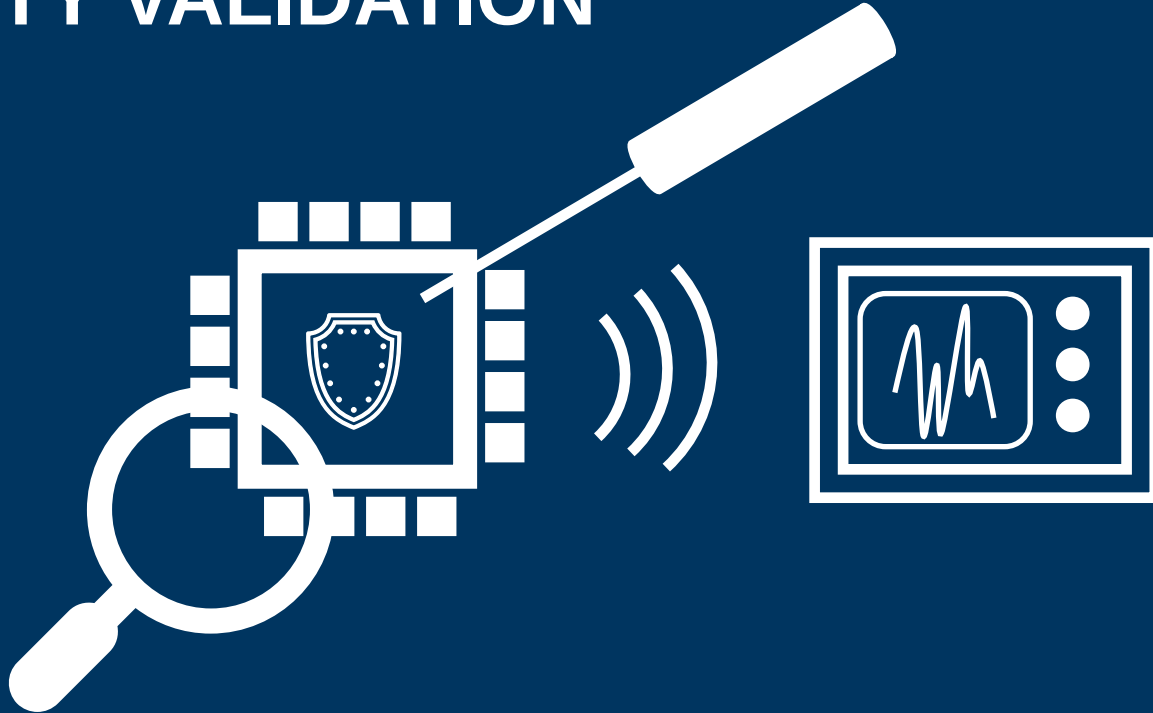
Linear Error Correcting Codes



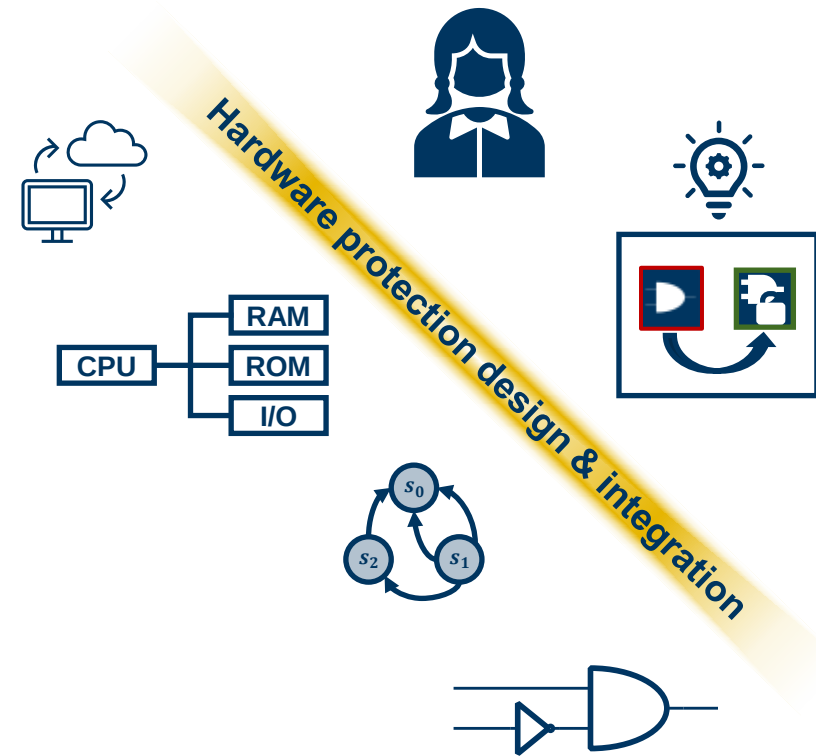
Detection: $d_{\min} - 1$

Correction: $\left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$

COUNTERMEASURE INTEGRATION & SECURITY VALIDATION



Integration and Validation of Countermeasures in Cryptography



- § **Most (security-related) countermeasures require manual integration and placement**
 - § Locations selected by criticality analysis
 - § Protections chosen by individual coverage assumption
- § **Security-oriented computer-aided design tools** can support the designer in an early stage of the design process
- § Frameworks for **automated exploration** and protection integration according to adversary model / assumptions

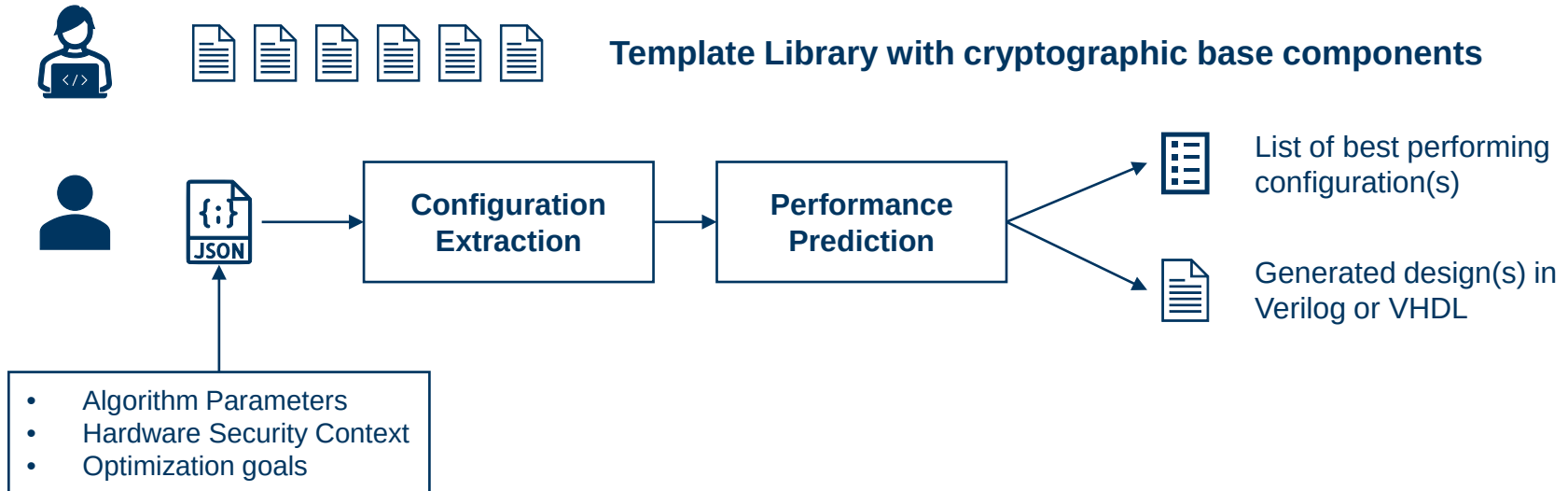
The HADES Framework

Automated Generation of Cryptographic Hardware



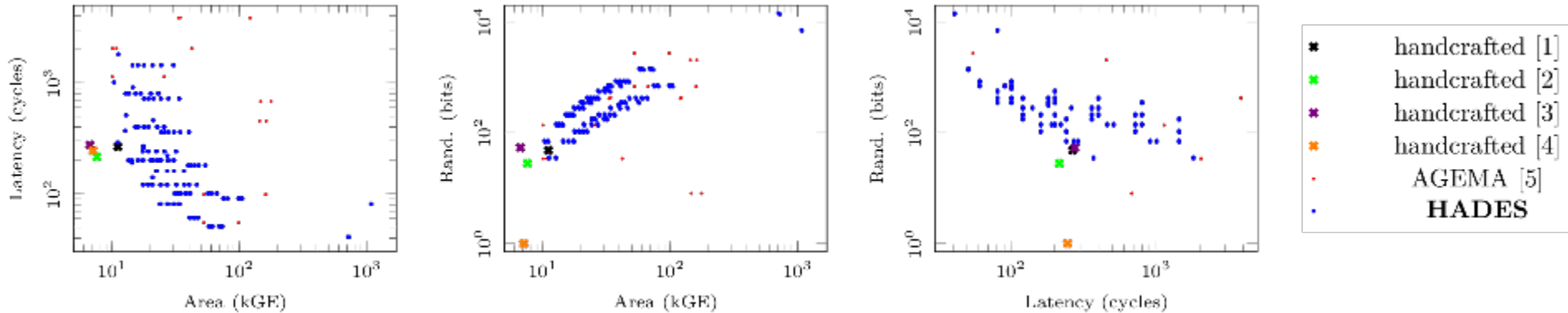
HADES

- § **Full design exploration** for standardized cryptographic units (AES, SHA-3, ML-KEM)
- § **Automated integration of SCA countermeasures** based on secure gadgets
- § **Flexible but competitive results** while eliminating errors from manual integration



HADES Framework Case Study: First-order Masked AES-128

Comparison with Previous (Manually-Crafted) Works

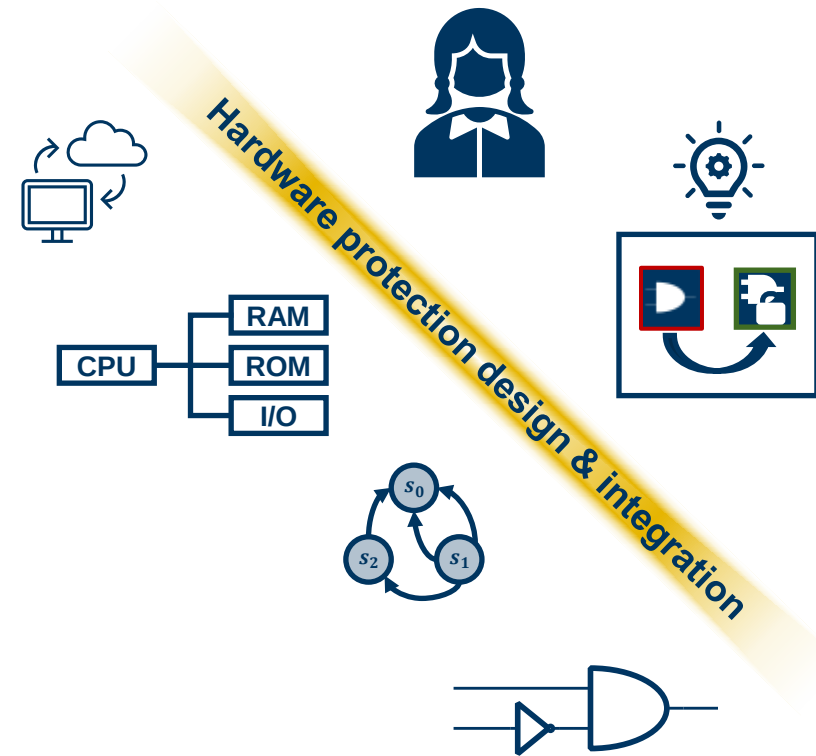


- [1] Pushing the limits: A very compact and a threshold implementation of AES. Moradi et al., Eurocrypt '11
- [2] Domain-Oriented Masking: Compact Masked Hardware Impl. with Arbitrary Protection Order. Groß et al., TIS '16
- [3] Masking AES with $d+1$ Shares in Hardware. De Cnudde et al., CCS '16
- [4] Re-consolidating first-order Masking Schemes. Shahmirzadi and Moradi, CHES '21
- [5] Automated Generation of Masked Hardware. Knichel et al., CHES '22



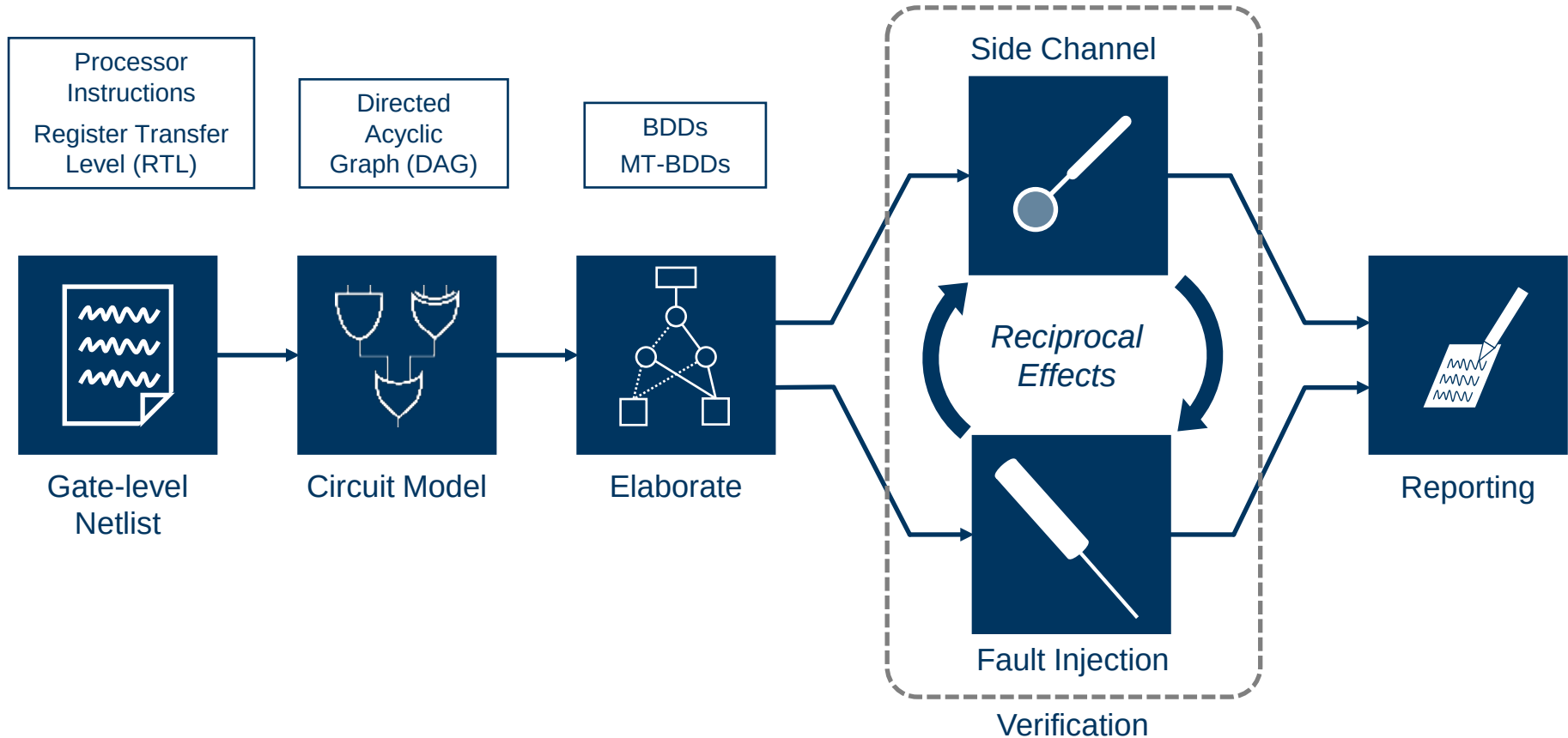
HADES

Integration and Validation of Countermeasures in Cryptography



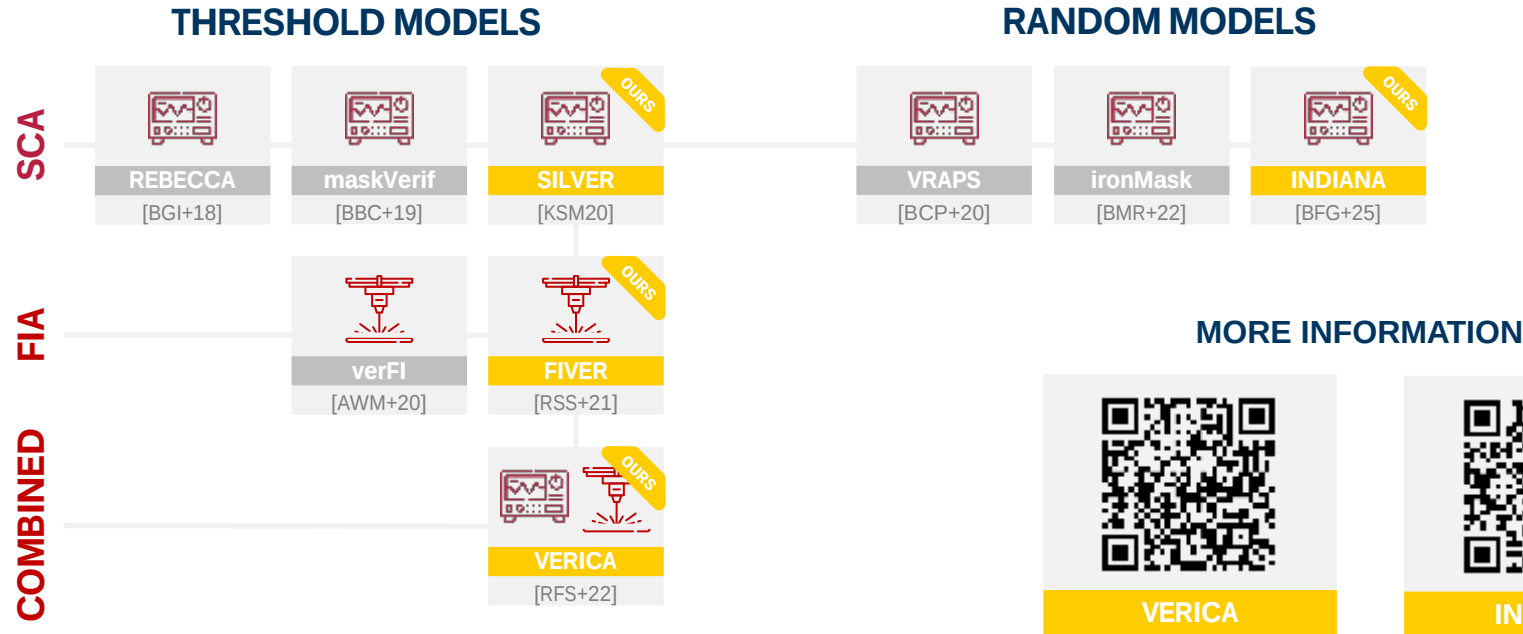
- § **Most (security-related) countermeasures require manual integration and placement**
 - § Locations selected by criticality analysis
 - § Protections chosen by individual coverage assumption
- § **Security-oriented computer-aided design tools** can support the designer in an early stage of the design process
- § Frameworks for **automated exploration** and protection integration according to adversary model / assumptions
- § **Pre-silicon validation of protection countermeasures**
 - § Validation based on abstract model either by
 - § Formal verification (limited circuit complexity)
 - § Simulation (limited coverage/)

Formal Verification: Netlist-Based Approach

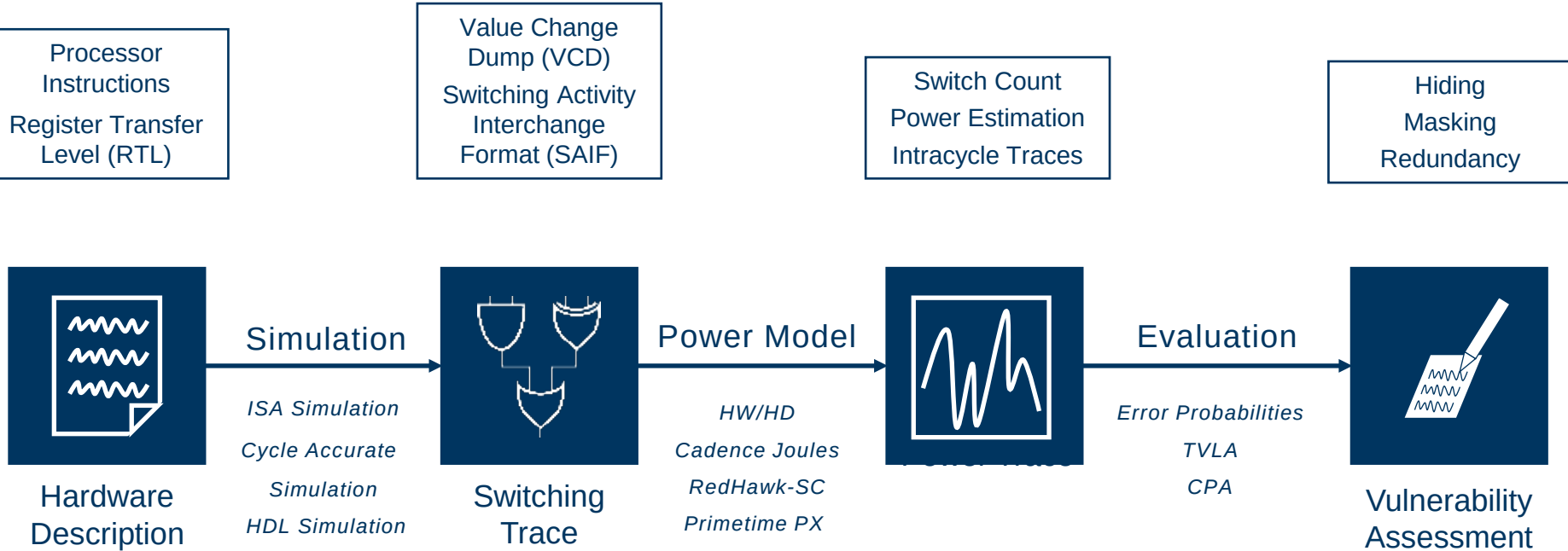


Formal Verification of Physical Security Properties

- § Exhaustive verification of non-interference of security properties **over all circuit components**
- § **High computational complexity** with respect to input circuit size + security models

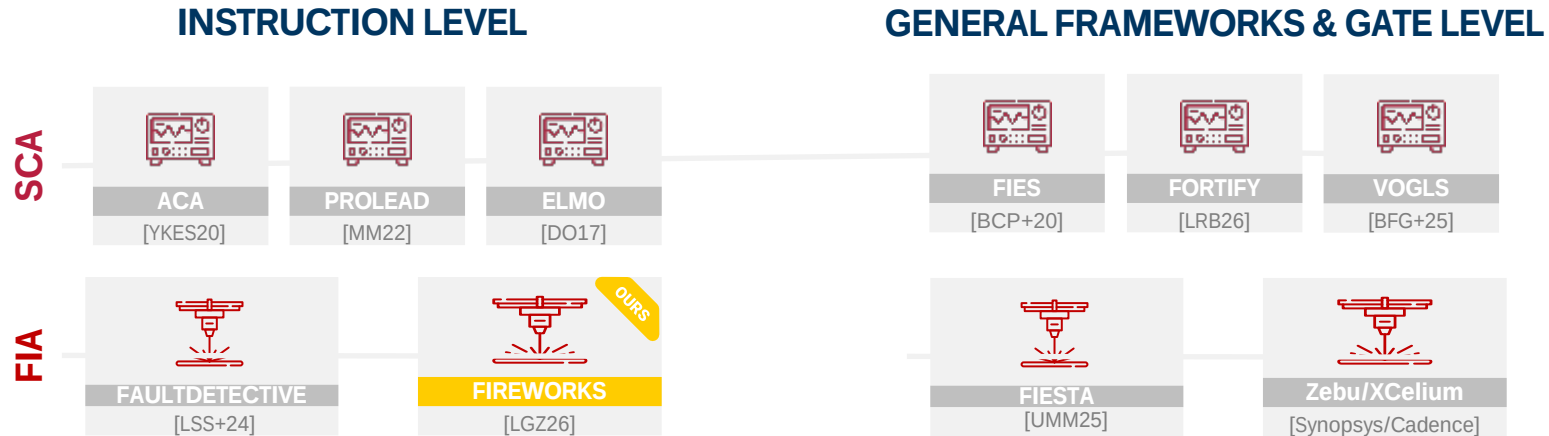


Simulation Tools of Leakage (and Fault Effects)



Formal Verification of Physical Security Properties

- § Assessment quality directly related to simulation scope and model accuracy
- § Underlying use of (worst-case) power model tooling



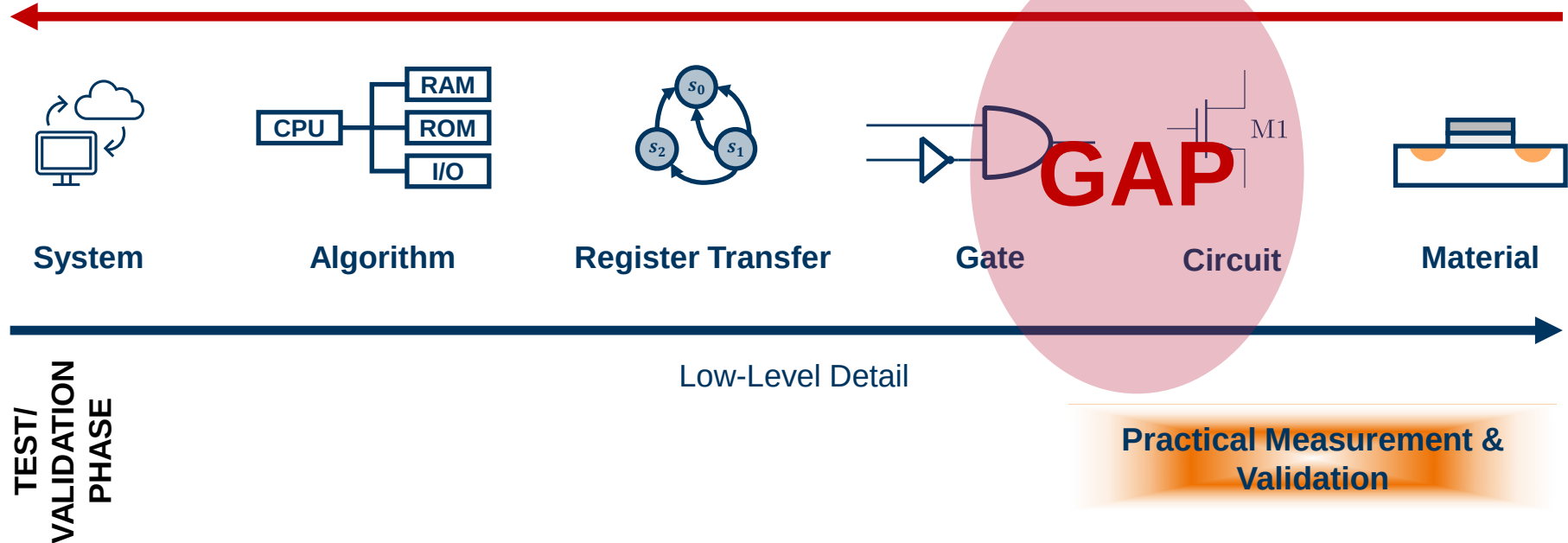
Gap between Integration & Validation of Countermeasures

Countermeasure design & integration

Side-Channel + Fault Modelling

Abstraction

DESIGN
PHASE



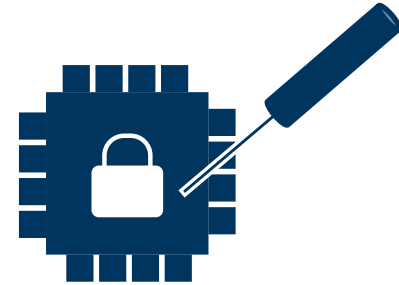
Practical Measurement & Validation

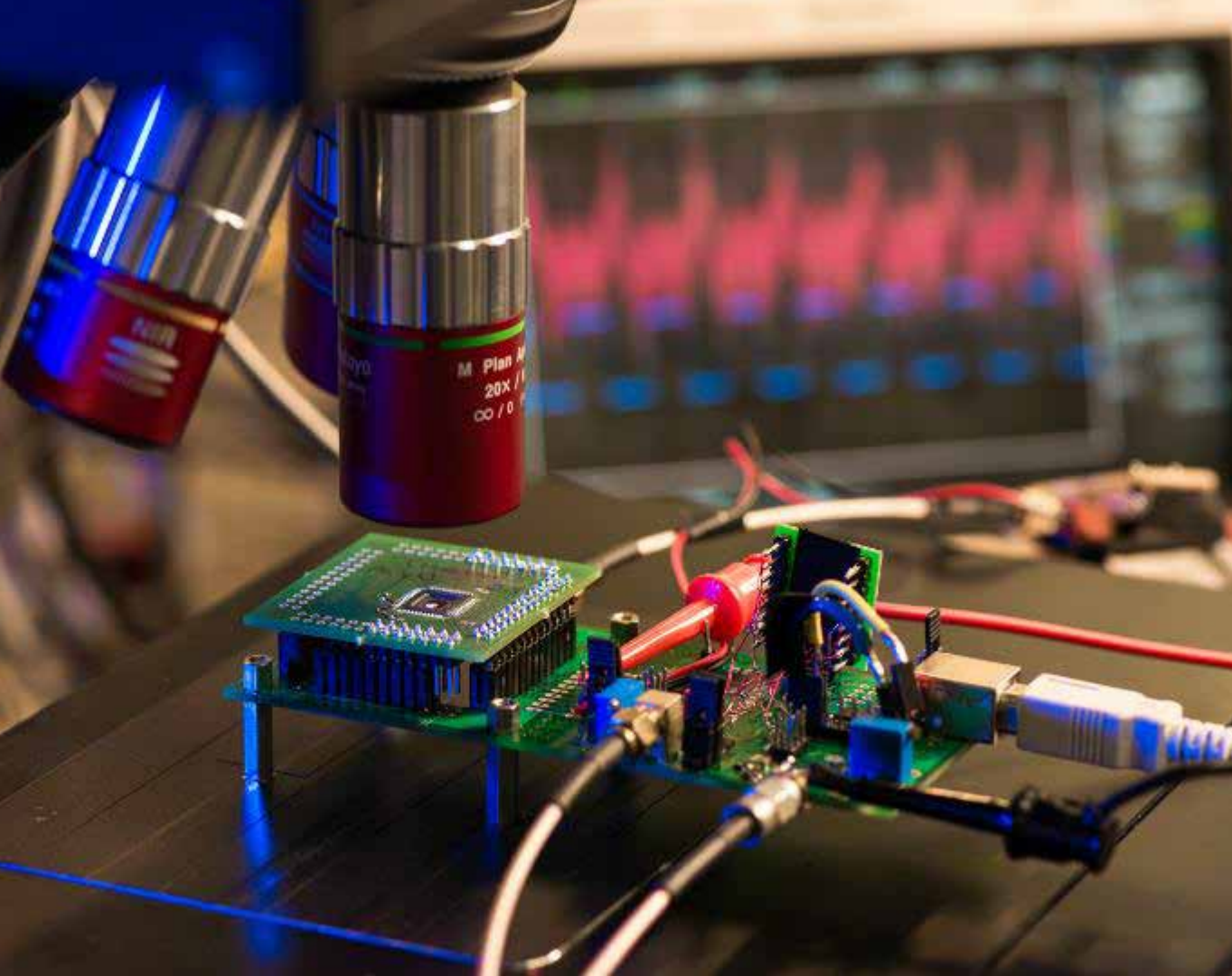
CONCLUSIONS

Conclusions & Lessons Learned

Presilicon security assurance is an active field of research...

- **Develop abstract but sufficiently accurate (low-level) models**
 - Better understanding of modelling+practical evaluation gap
 - Physical security models for SCA, FIA and combined analysis
 - Reproducible results with different evaluation setups/reports
- **Develop structural countermeasures for integration in EDA tools**
 - Composability is a trade-off between generic application but higher cost due to genericity of approach
 - Complex countermeasure interactions still hard to achieve





Thank you!

Tim Güneysu
Security Engineering
tim.gueneyasu@rub.de

References

- [BFG+24] Belaid, Feldtkeller, Güneysu, Guinet, Richter-B., Rivain, Sasdrich, Taleb, Formal Definition and Verification for Combined Random Fault [...], ASIACRYPT 2024
- [AWMN20] Arribas, Wegener, Moradi, Nikova, Cryptographic Fault Diagnosis using VerFI, HOST 2020
- [RFSG22] Richter-Brockmann, Feldtkeller, Sasdrich, Güneysu: VERICA - Verification of Combined Attacks Automated formal verification of security [...], CHES, 2022
- [LSS24] Liu, Shanmugam, Schaumont, FaultDetective Explainable to a Fault, from the Design Layout to the Software, CHES, 2024
- [UMM25] Uhle, Müller, Moradi, Formal Definition and Verification for Combined Random Fault and Random Probing Security, ASIACRYPT 2024
- [ISW03] Ishai et al., Private Circuits: Securing Hardware against Probing Attacks, CRYPTO, 2003
- [FD18] Faust et al., Composable Masking Schemes in the Presence of Physical Defaults & the Robust Probing Model, CHES, 2018