# Sécurité des moyens de test des SoC

Mathieu da Silva, Marie-Lise Flottes, Giorgio Di Natale, Bruno Rouzeyre




# SÉCURITÉ DES MOYENS DE TEST DES SOC

Mathieu Da Silva, Marie-Lise Flottes, Giorgio Di Natale, Bruno Rouzeyre

Journée thématique des GDR SoC² et Sécurité Informatique

*Sécurité des SoC complexes hétérogènes – de la TEE au matériel*

# PROJET TEEVA

- Travaux réalisés dans le cadre du projet TEEVA: Trusted Environment Execution eVAluation

- Partenaires

gemalto security to be free TRUSTONIC PHONESEC LA MAÎTRISE DES RISQUES NUMÉRIQUES MINES Saint-Étienne LABORATOIRE HUBERT CURIEN UMR • CNRS • 5516 • SAINT-ETIENNE LIRMM

# Summary

1) Context of testing
2) Threats related to the test infrastructures
3) Proposed countermeasures: Scan Encryption
4) Application of the proposed countermeasures
5) Conclusion

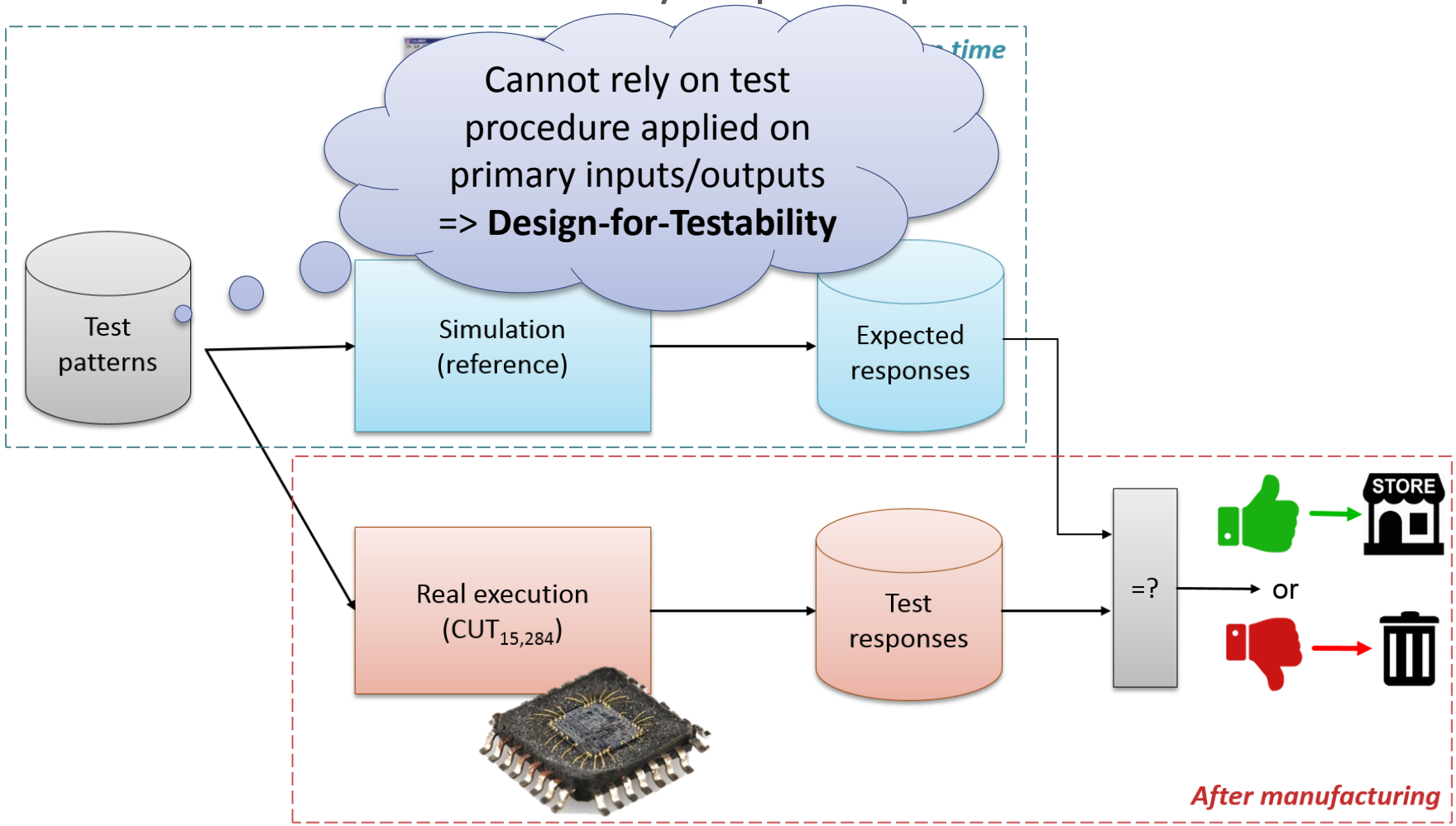
# Summary

1) **Context of testing**
   - Design-for-Testability (DfT)
   - Test standards
2) Threats related to the test infrastructures
3) Proposed countermeasures: Scan Encryption
4) Application of the proposed countermeasures
5) Conclusion

# Context of testing

- Design-for-Testability (DfT)
- Test standards

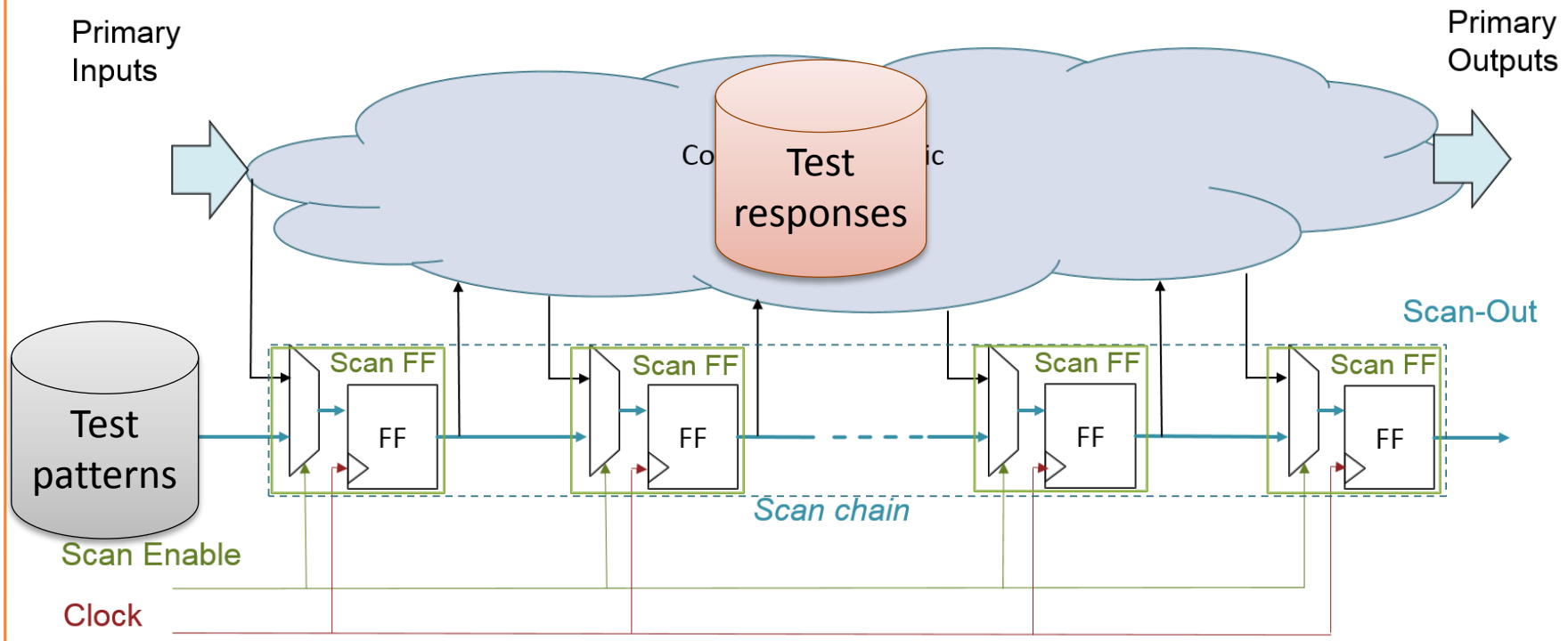
- Test of circuit is a mandatory step in IC production



# Scan chains

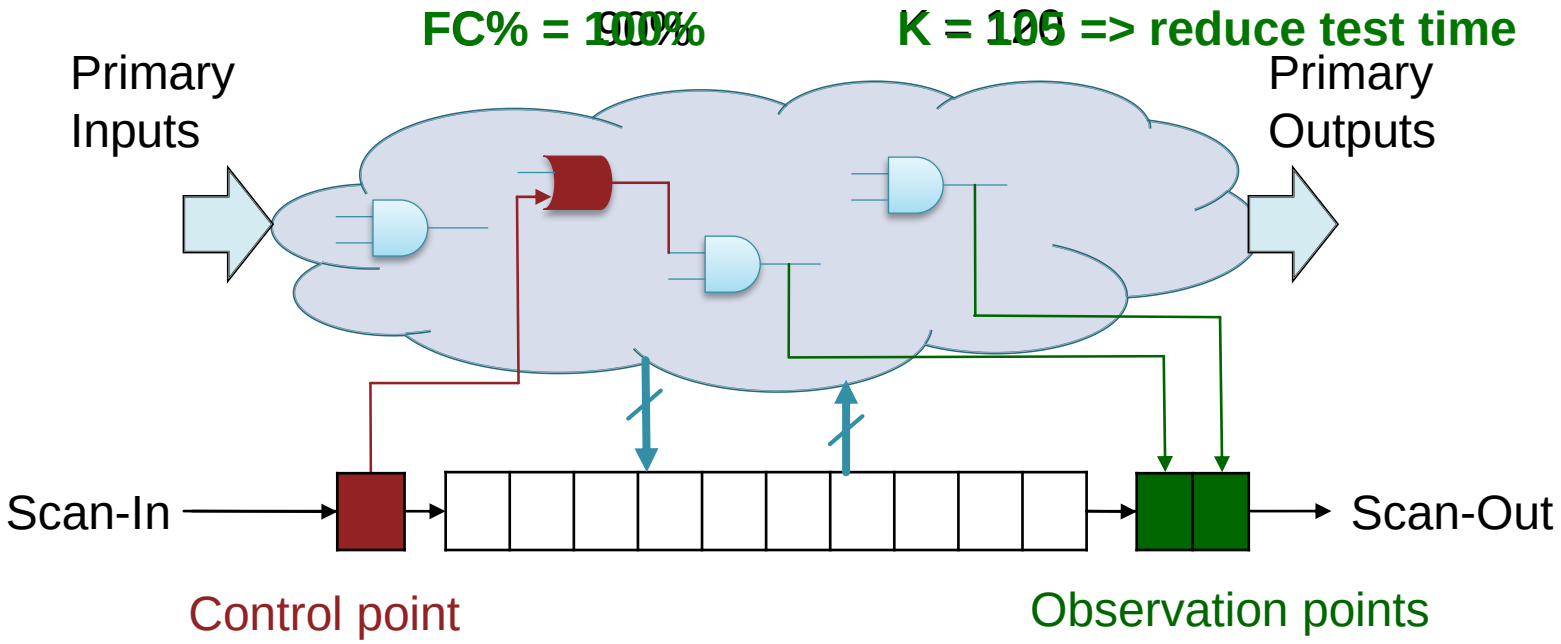


- Most popular method for Design-for-Test = Scan chains
  - Replace original FF by Scan FF connected serially together
  - Extra port « Scan-In » => total control on internal states
  - Extra port « Scan-Out » => total observation on internal states



# INSERTION OF TEST POINTS

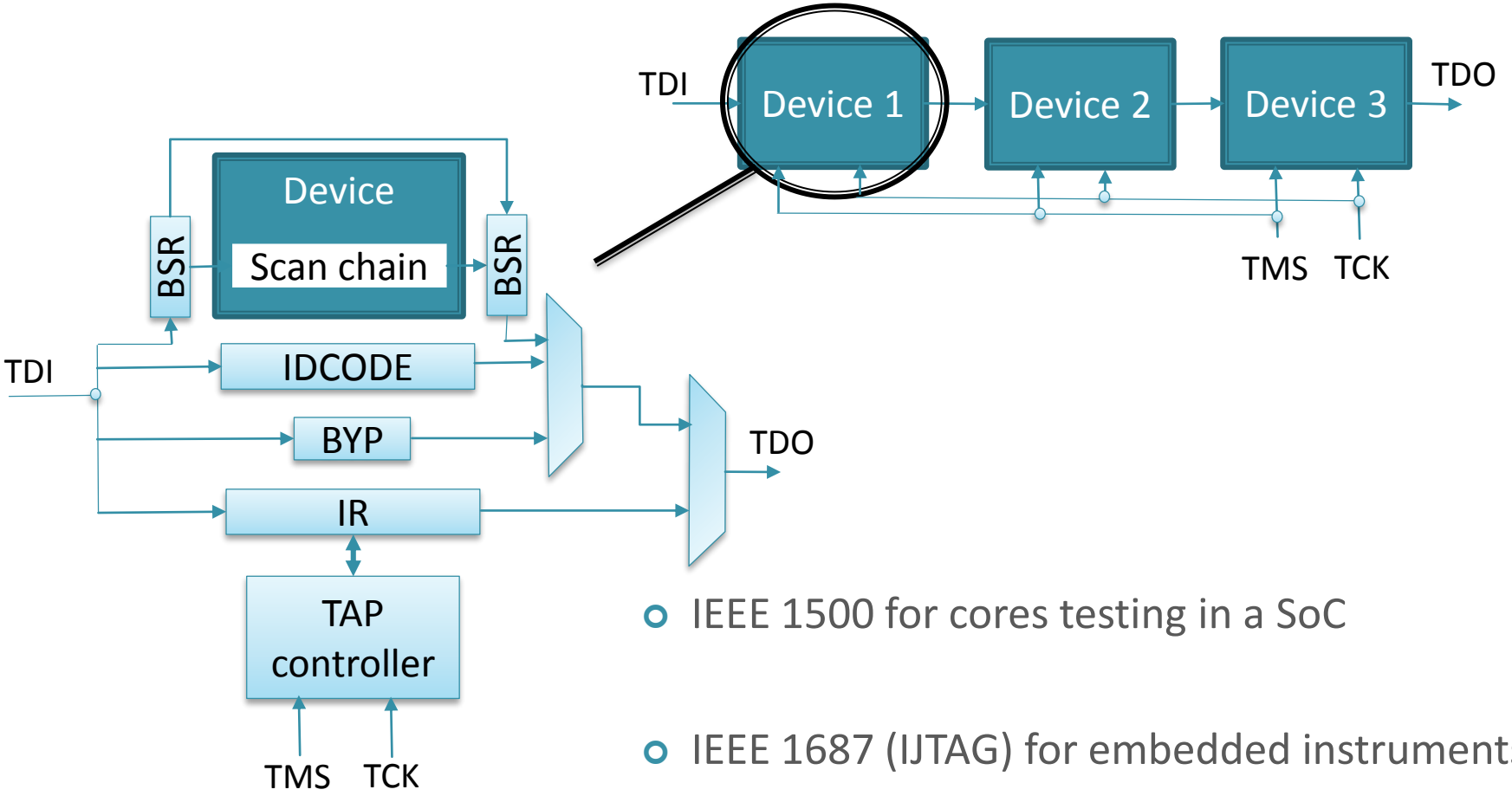
- Extra-DfT: insertion of test points
- Goal: increase the fault coverage FC% and/or
  reduce the number of patterns K



# TEST STANDARDS



- IEEE 1149 (JTAG) for board testing + diagnosis & debug facilities




- IEEE 1500 for cores testing in a SoC
- IEEE 1687 (IJTAG) for embedded instruments

# Summary

1) Context of testing

2) **Threats related to the test infrastructures**

   - Overview of the threats
   - Scan attacks
   - Security analysis on TEE

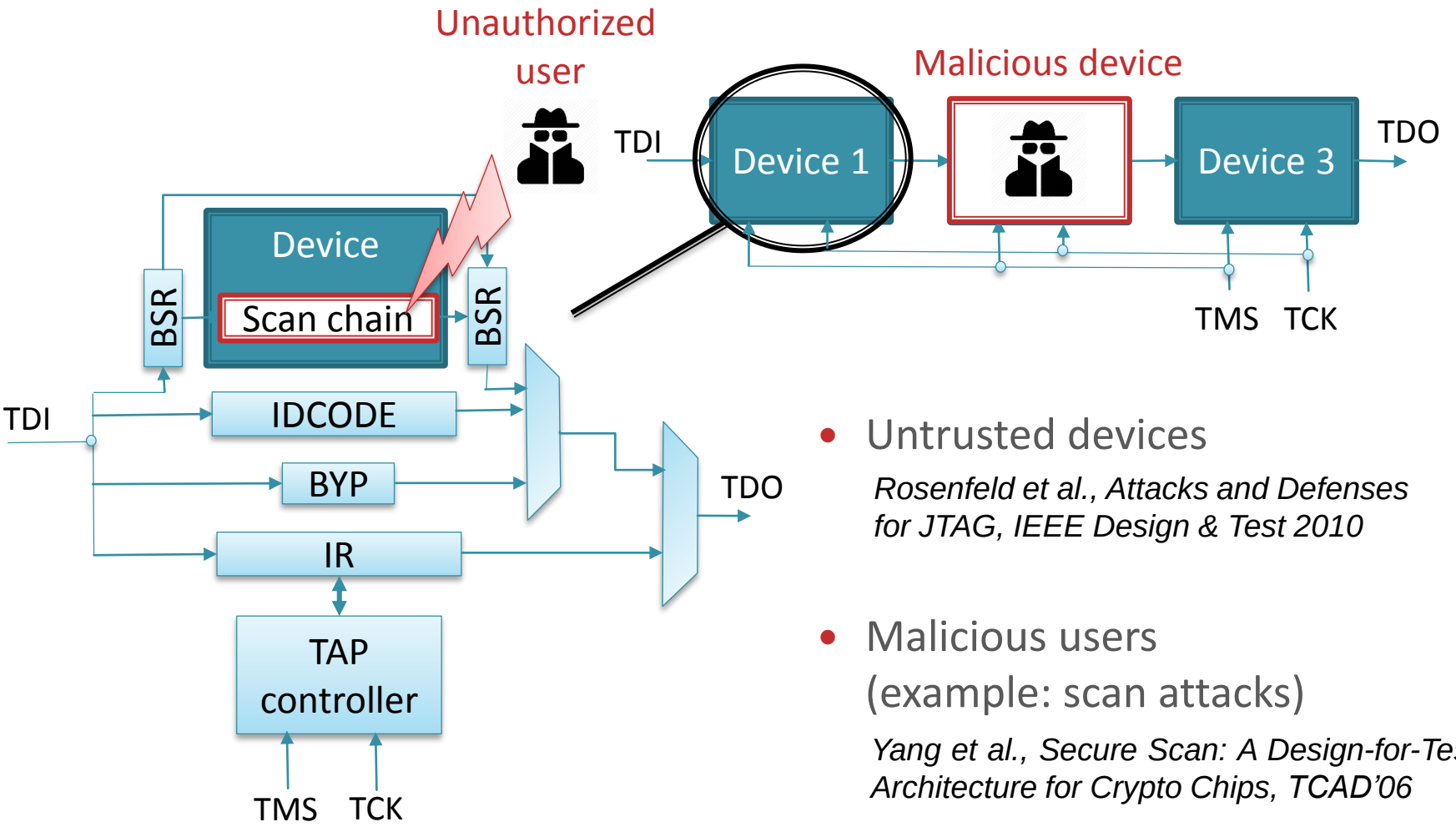
3) Proposed countermeasures: Scan Encryption

4) Pros and cons of the proposed countermeasures

5) Conclusion

# THREATS






- Untrusted devices

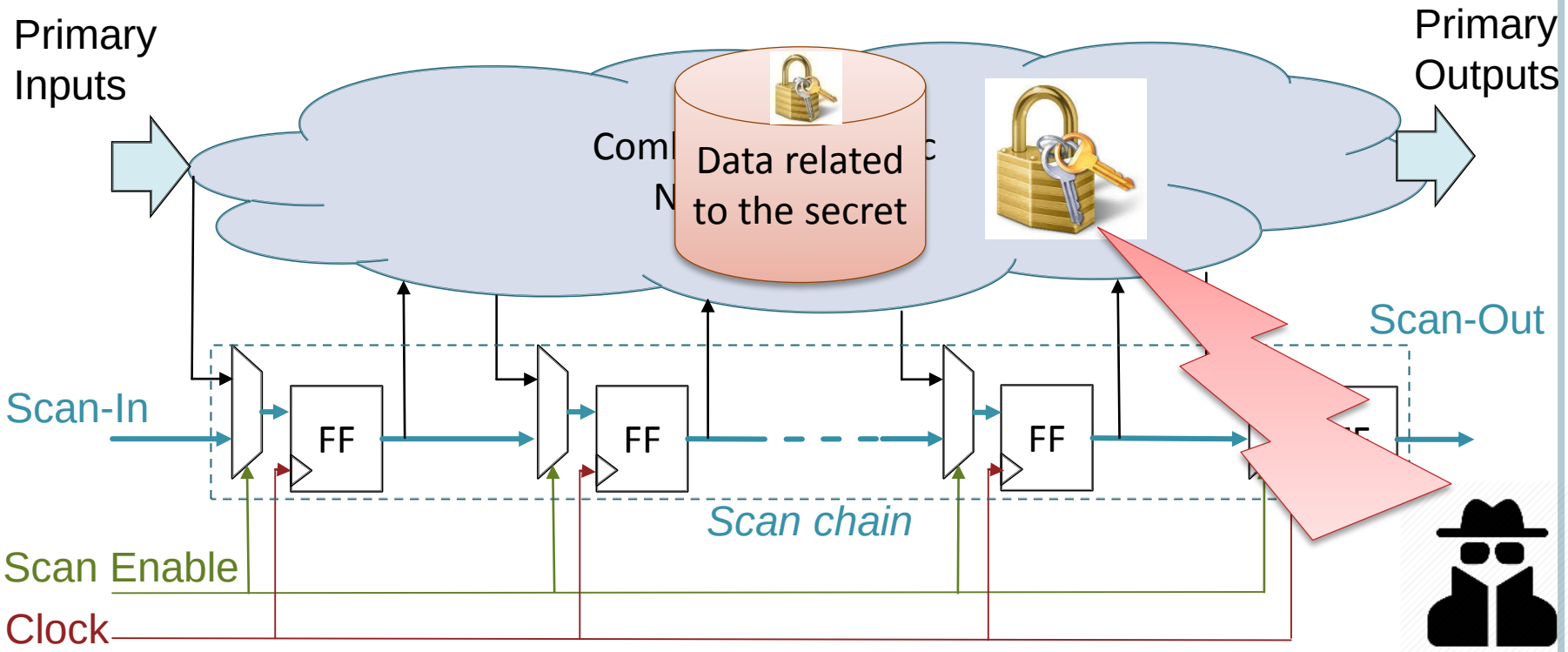
  *Rosenfeld et al., Attacks and Defenses for JTAG, IEEE Design & Test 2010*

- Malicious users (example: scan attacks)

  *Yang et al., Secure Scan: A Design-for-Test Architecture for Crypto Chips, TCAD'06*

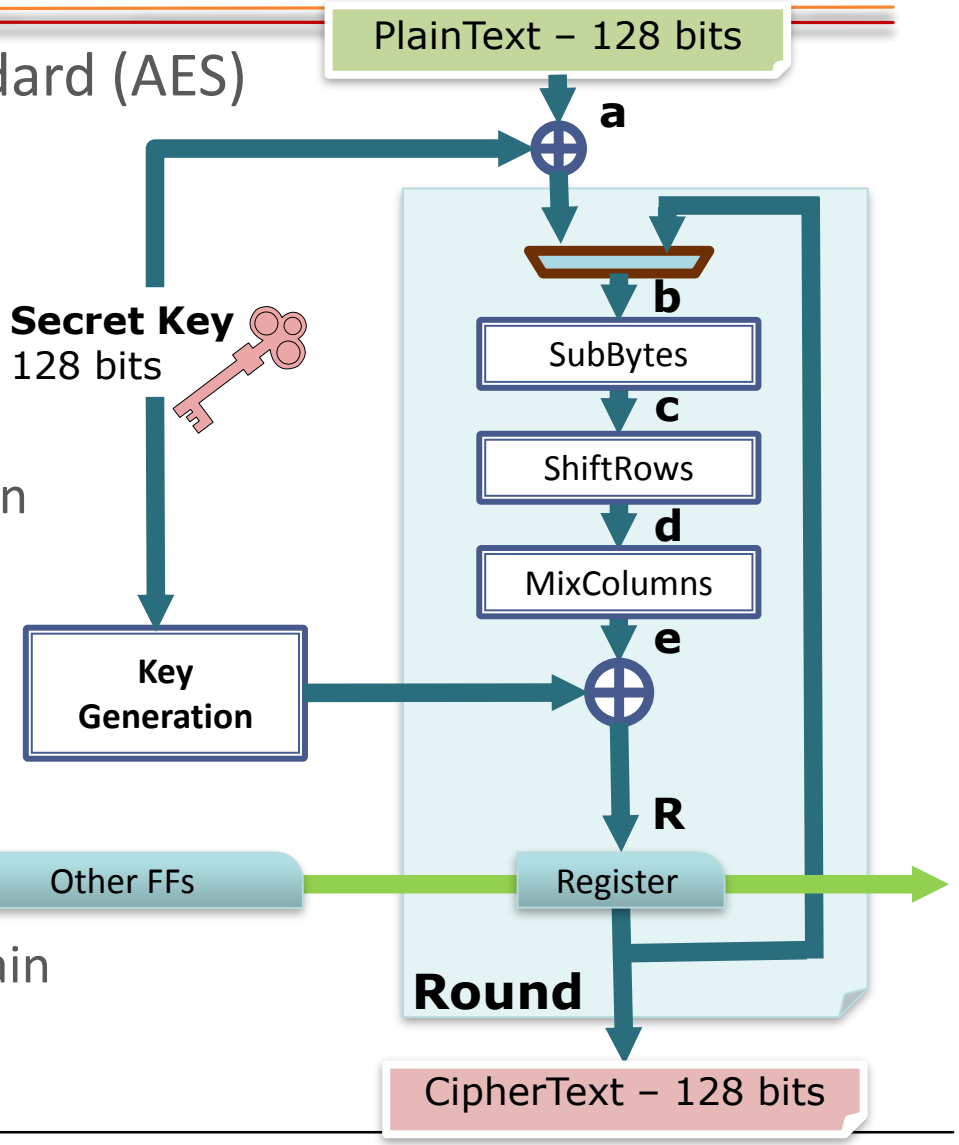
# Scan attack principle

- Exploit the scan chain by an attacker => Scan attacks
  - Goal: Retrieve embedded secret data
  - Exploit observability or controllability offered by scan chains
  - Principle: switch between functional and scan modes



# Scan attack on AES

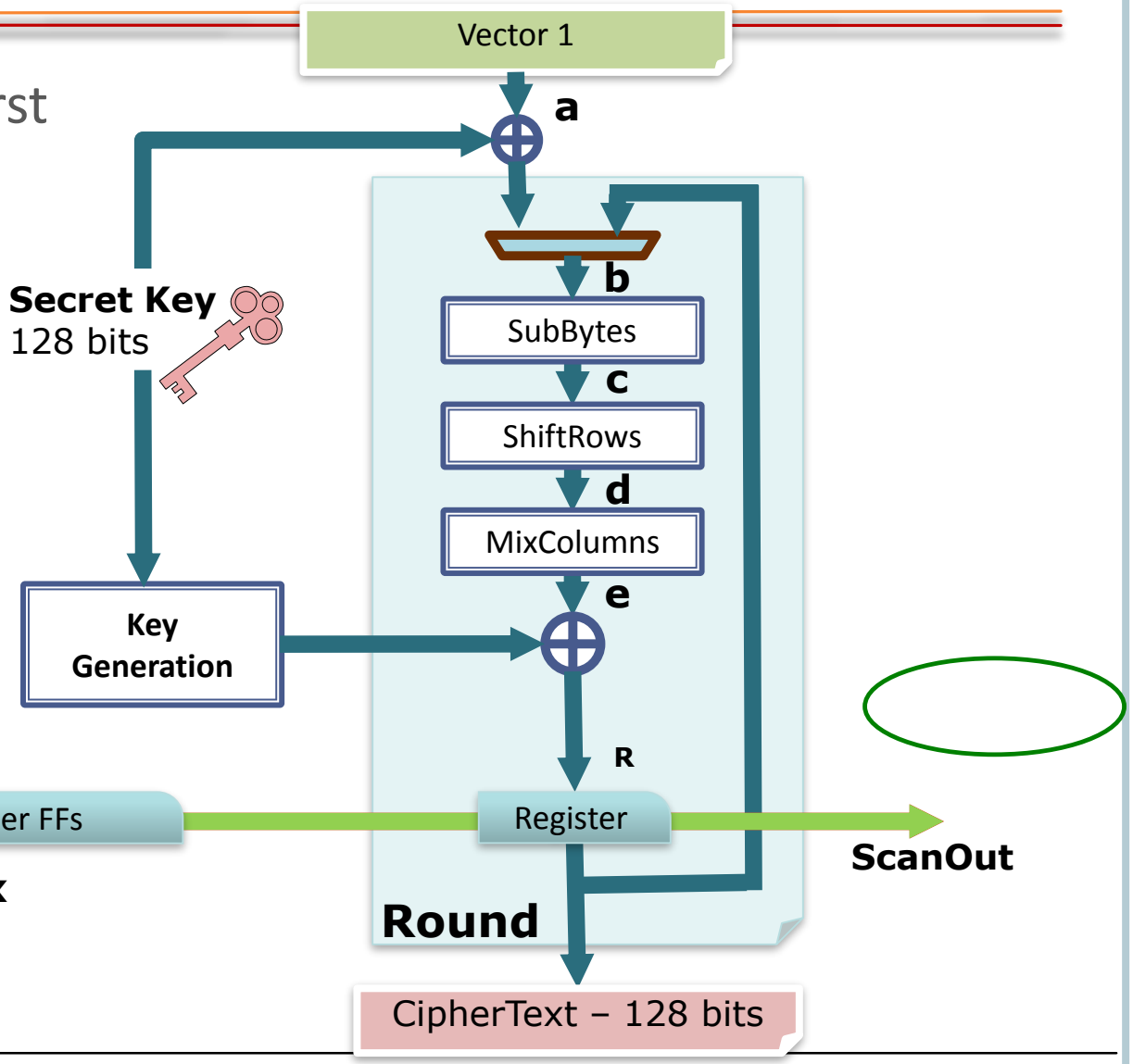
- Advanced Encryption Standard (AES)
  - Ciphertext after 10 rounds
  - Not secure after 1 round
- Attack pre-requisites
  - Attacker can switch between functional and test modes
  - Scan chain includes FFs of the round register
- Attack principle
  - Observation of the scan chain after 1 round



# Differential attack

- Application of a first vector

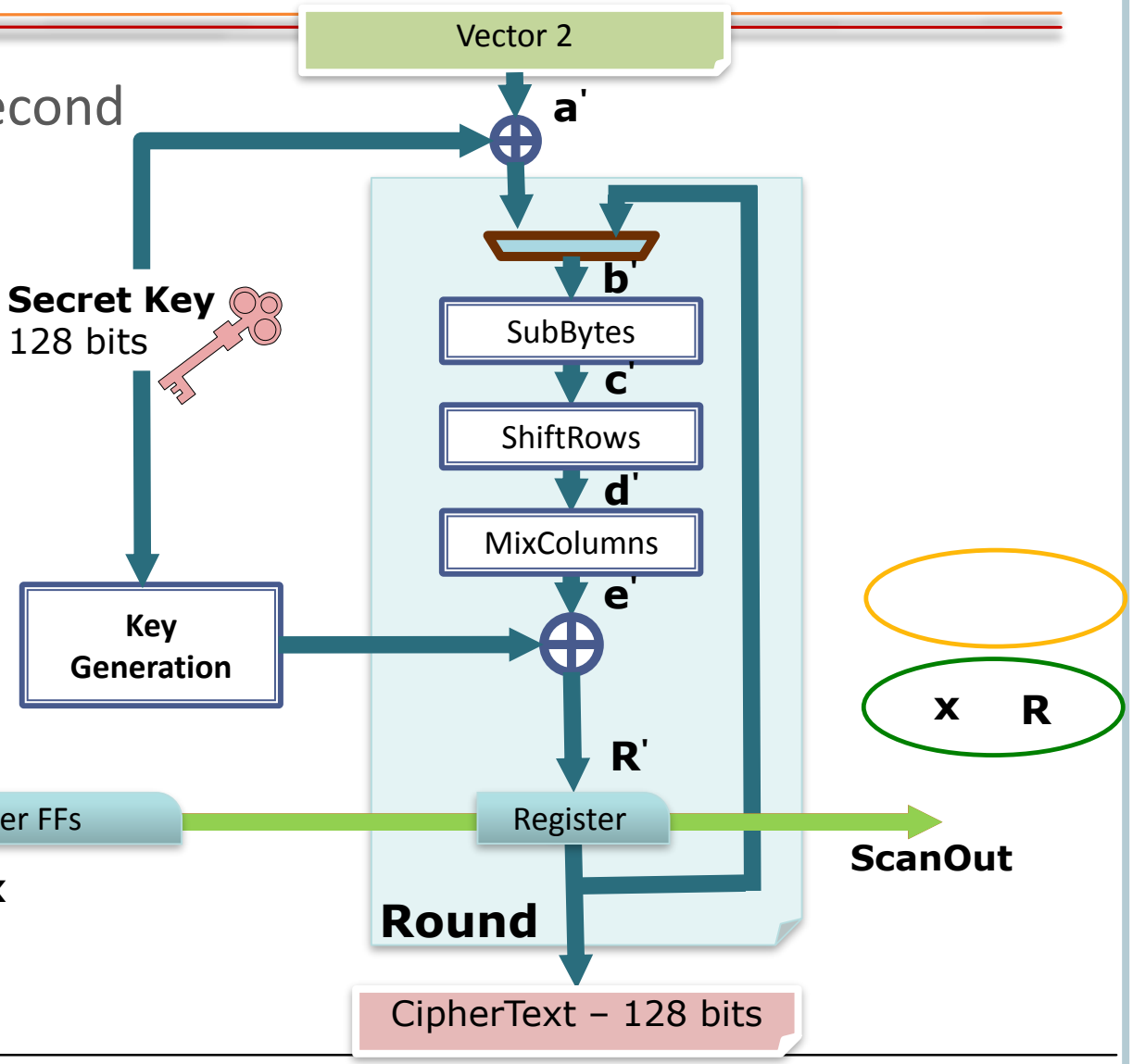
1) Reset
2) Normal mode
   - 1 AES round
3) Test mode
   - Scan out the round register content



# Differential attack

- Application of a second vector

1) Reset
2) Normal mode
   - 1 AES round
3) Test mode
   - Scan out the round register content



# Differential attack

- Hamming distance

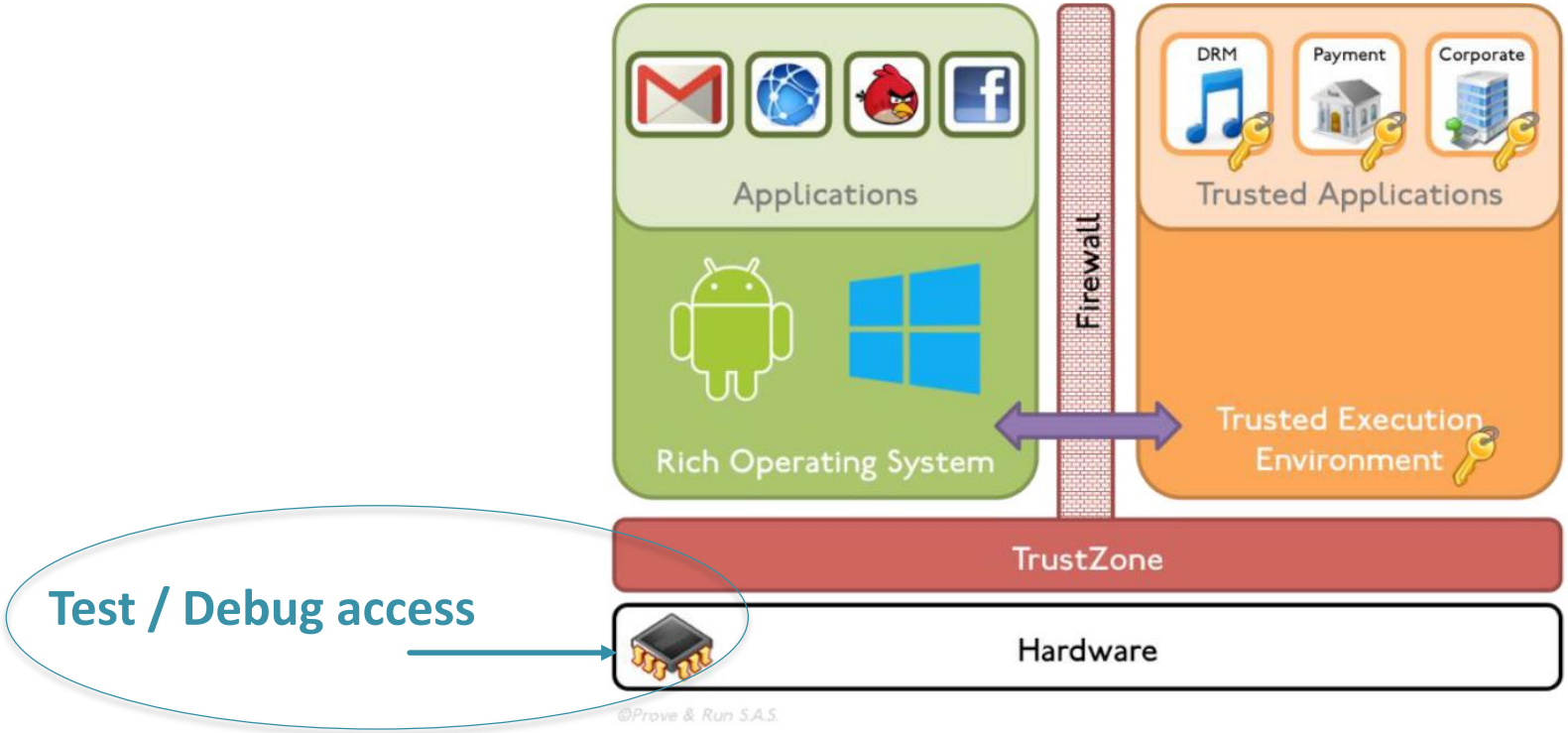

  - Attacker applies pairs of input values until hamming distance equal to specific values => key byte revealed

- On average, 32 trials

⇒ 512 trials to retrieve the whole 128-bit key

# Threats on TEE?

- Accessing the scan chains => no differentiation between data processed and saved in Non Secure and Secure world
- Test & Debug access = an open door for attacks



# Security analysis on TEE

- Industrial practice to ensure protection: **disconnection of the access to the scan chains**
- **Disadvantages:**
  - In-field diagnosis and debug impossible
  - Probing on disconnected access

  ⇒ Circumvent the countermeasure


Test / Debug access

# Summary

1) Context of testing

2) Threats related to the test infrastructures

3) **Proposed countermeasures: Scan Encryption**

   - Principle of Scan Encryption
   - Implementation with block cipher
   - Implementation with stream cipher

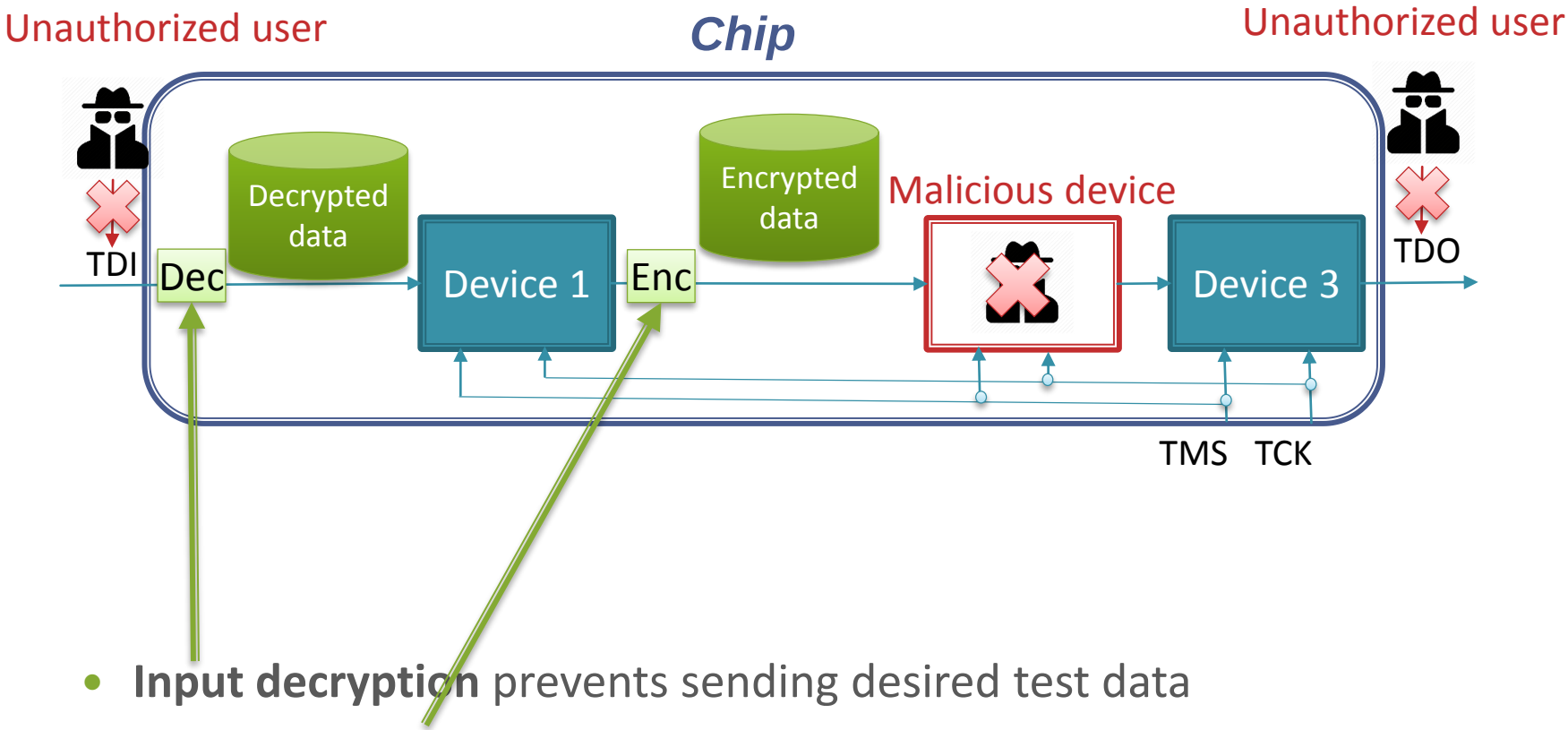
4) Application of the proposed countermeasures

5) Conclusion

# Scan encryption



- Solution: test communication encryption

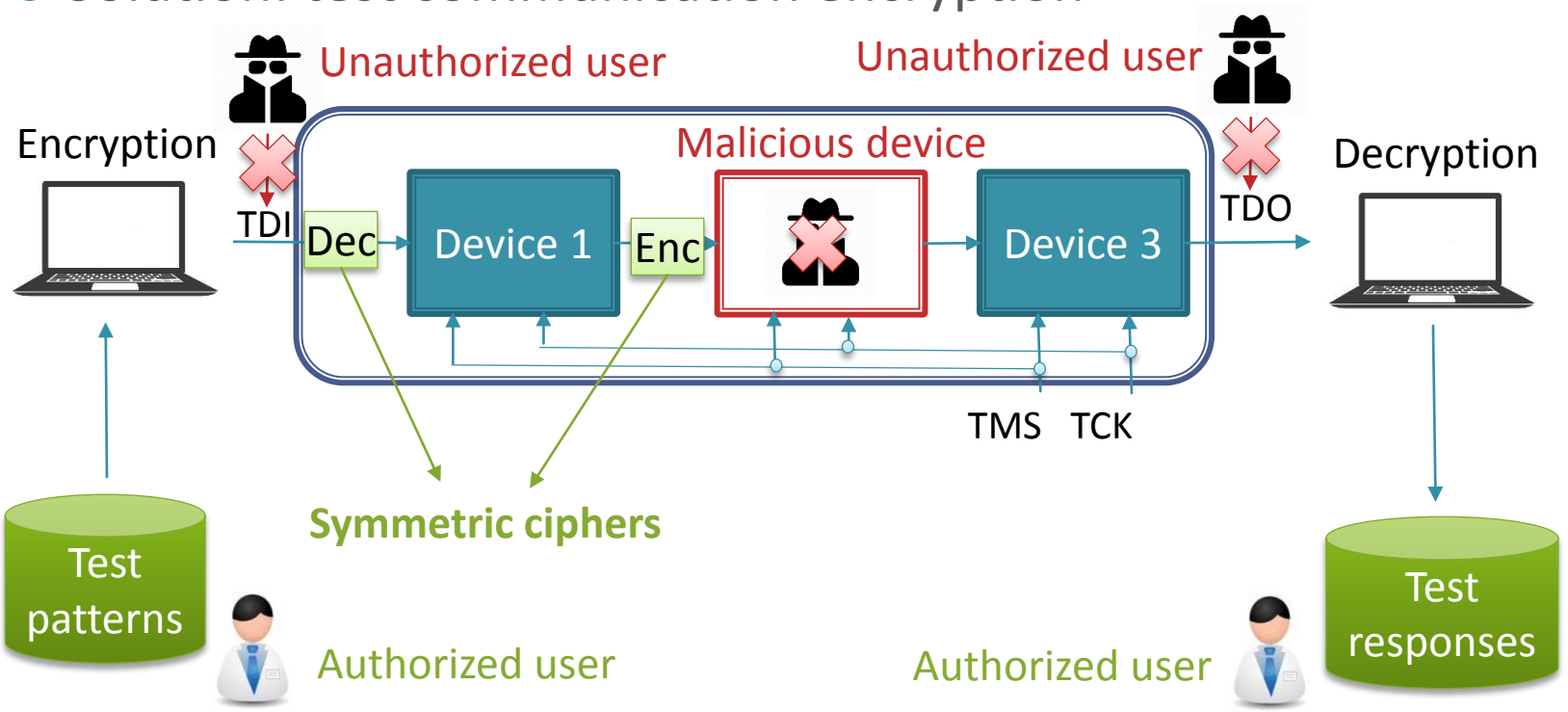



- **Input decryption** prevents sending desired test data
- **Output encryption** prevents reading plain test responses

# Scan encryption



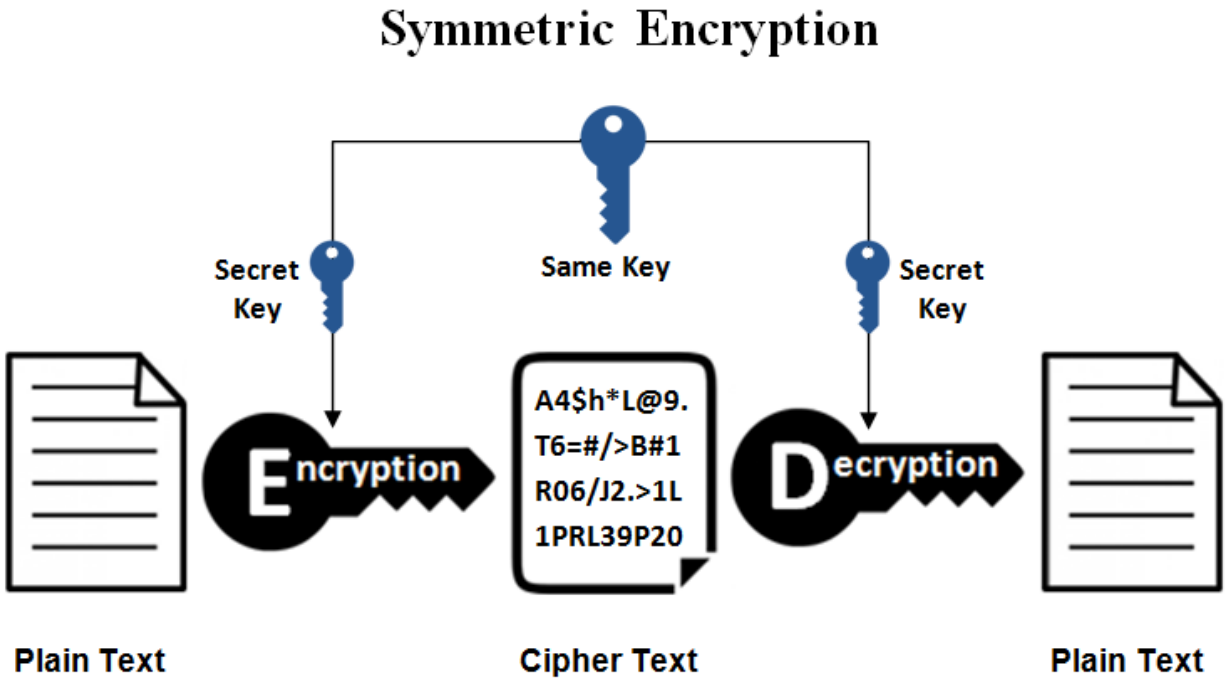
- Solution: test communication encryption




- Input decryption prevents sending desired test data
- Output encryption prevents reading plain test responses
- Test/debug only possible by authorized user knowing the secret key

# SYMMETRIC CIPHER

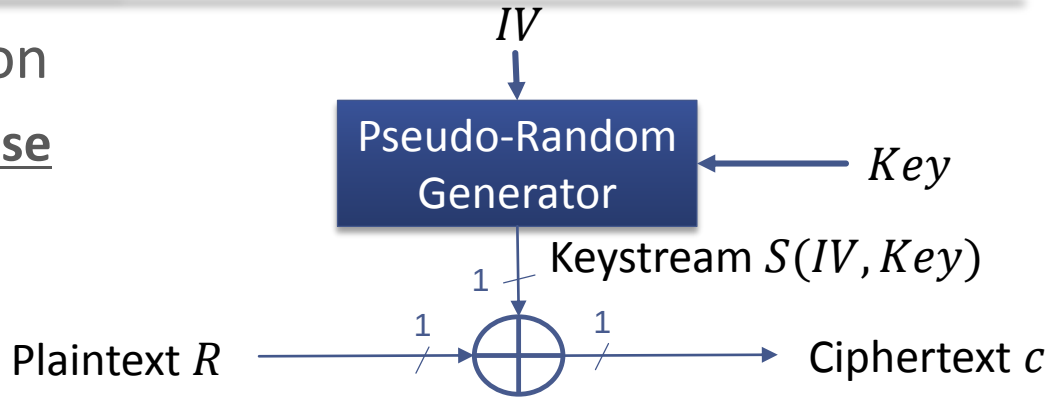





- 2 types of symmetric cipher: stream and block ciphers

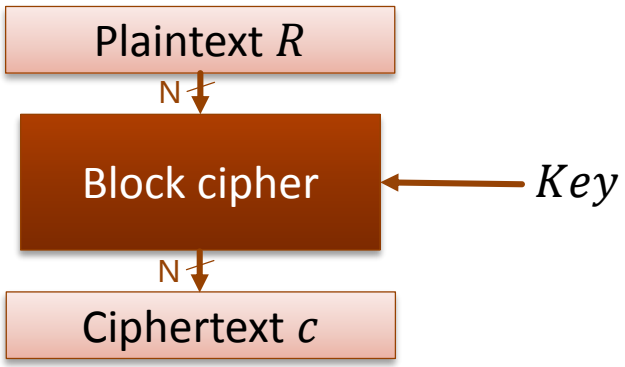
# Stream cipher / Block cipher

- Stream cipher encryption
  - Keystream XORed **bitwise** with the plaintext




+ "Naturally" adapted to serial test communication (JTAG, IEEE 1500, IJTAG)
+ Smaller area footprint compared to block ciphers
- But security?

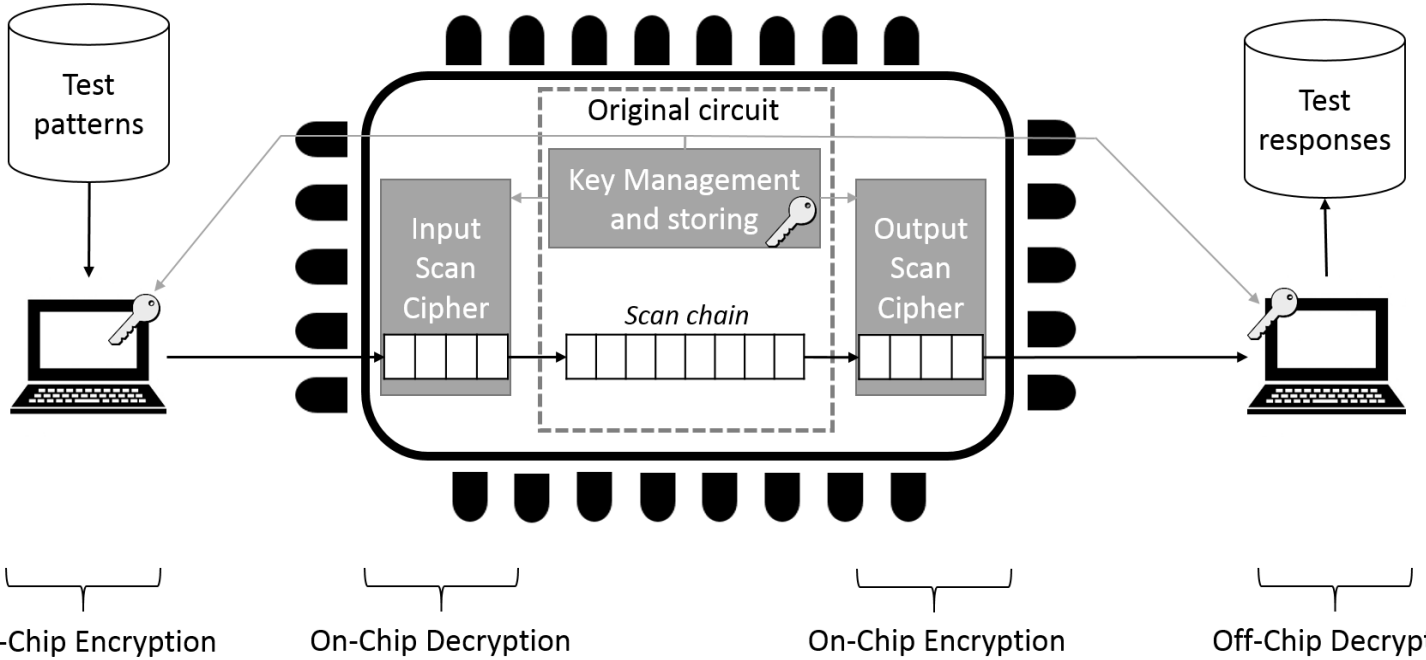
- Block cipher encryption
  - Confusion and diffusion on a **block** of plaintext




+ Strong security
- But cost?

# BASIC SCHEME

- Study of both solutions (block cipher and stream cipher)

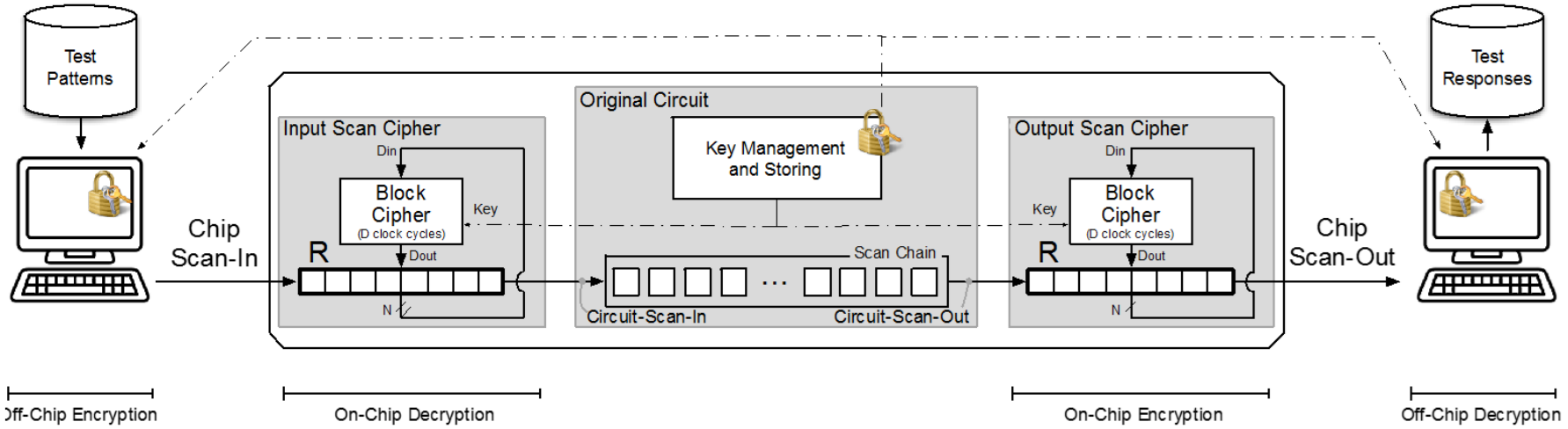



- Assumption: original circuit embedded a crypto-core with its key management and storing
- Scan chain encryption solution shares the key management and storing already implemented

# Block Cipher-based solution

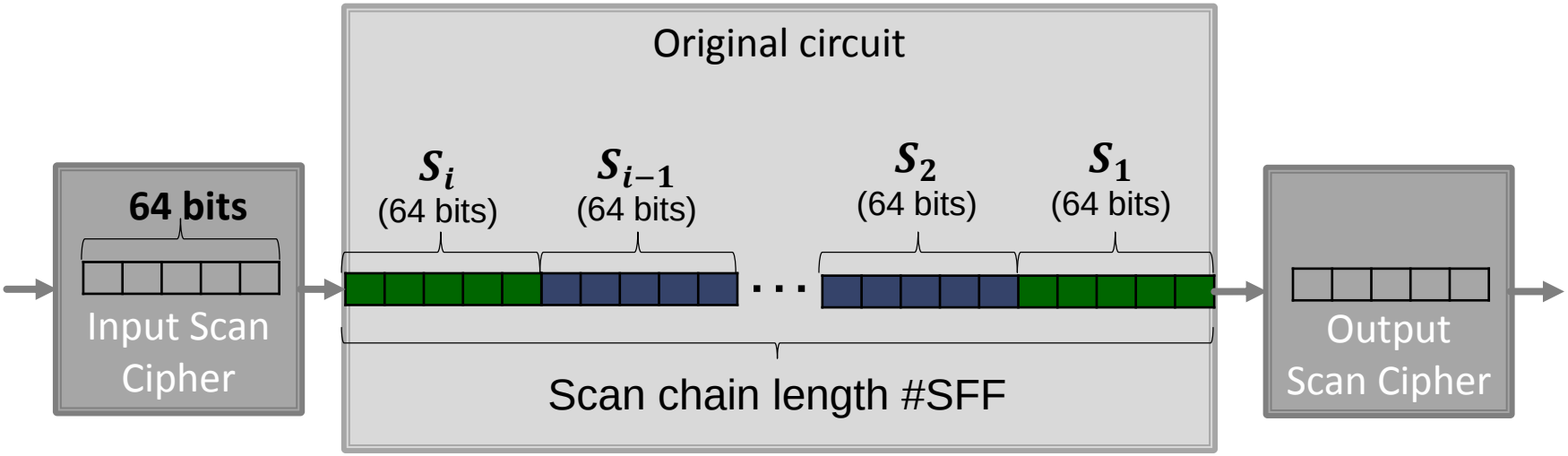


- Implementation on scan chain with 2 PRESENT block ciphers:
  - Lightweight (1 PRESENT = 2 139 GE)
  - Encryption by 64-bits block size



# Mode of operations

- 64 bits encrypted every 32 clock cycles

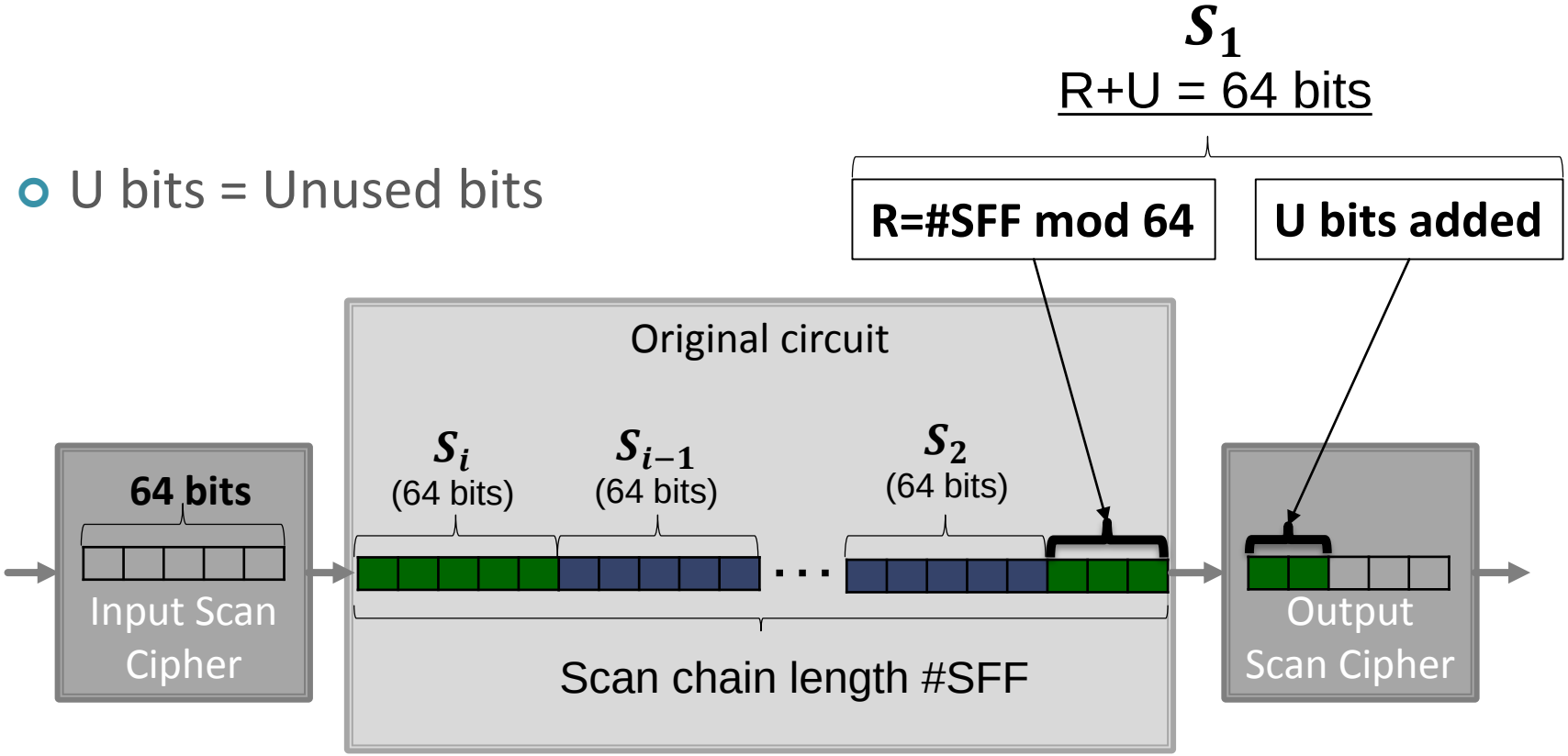



⇒ **#SFF = Px64**

⇒ **No test time overhead on each pattern**

# Mode of operations

- U bits = Unused bits

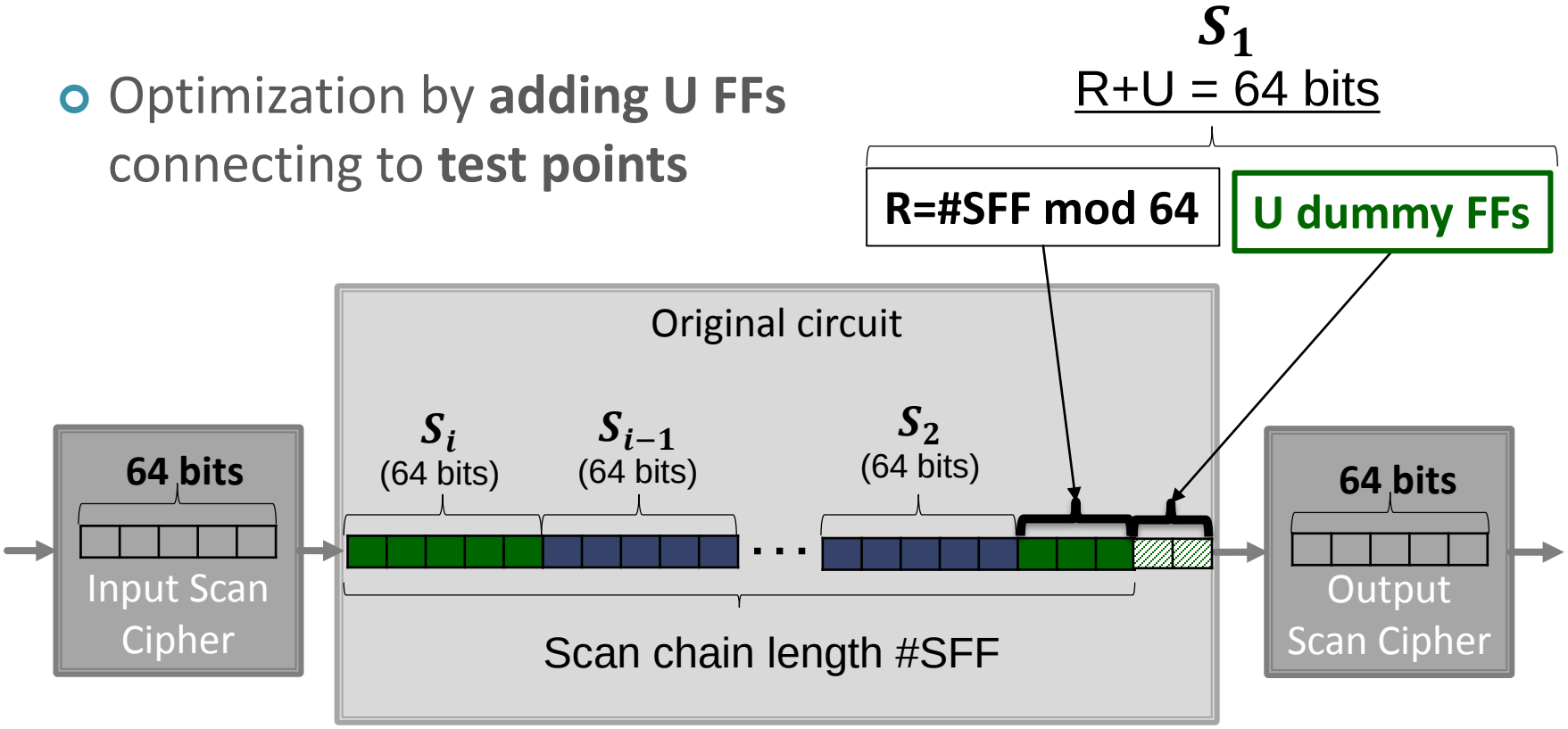



⇒ **#SFF = Px64 + R**

⇒ **Loss of U clock cycles per pattern**

# Test time optimization

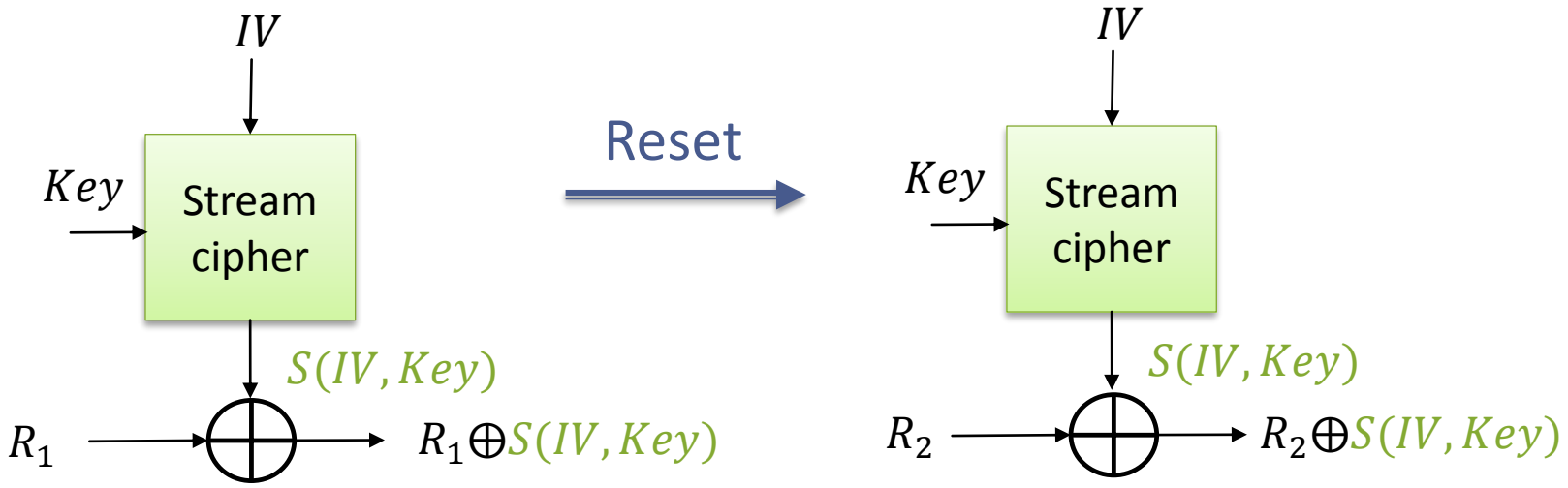
- Optimization by **adding U FFs** connecting to **test points**




⇒ **Reduce test time overhead**

# Stream cipher security?

- **Two-times pad**: same key and *IV* re-used => same keystream generated to encrypt different data




⇒ Possible to carry out attacks if requirement is not fit

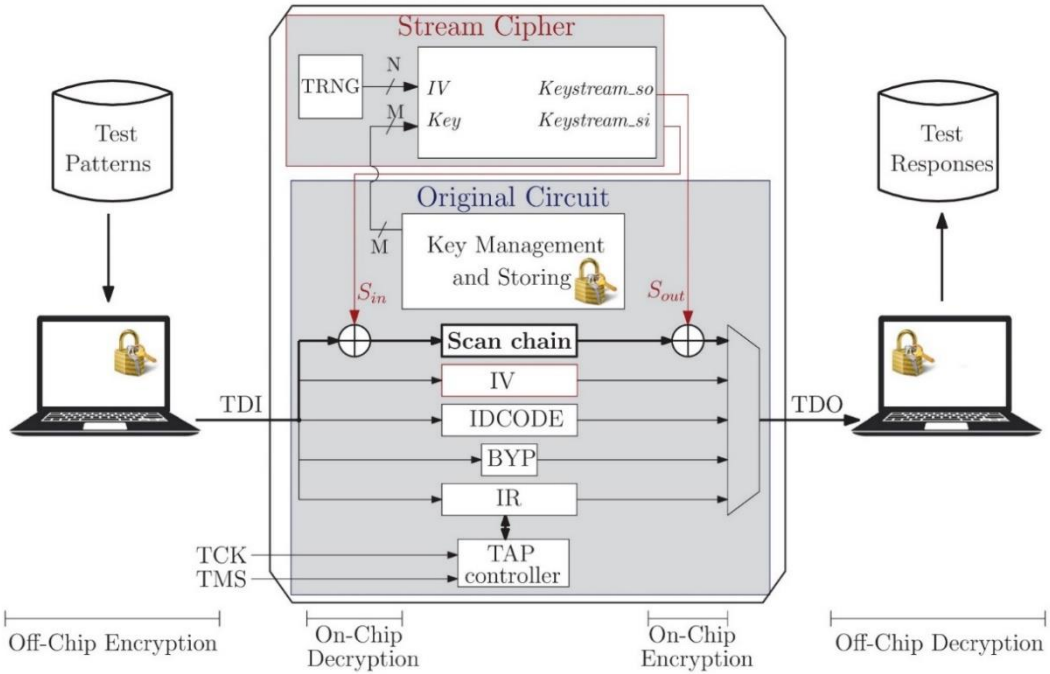
$\oplus$

⇒ Solution: *IV* generated randomly at each circuit reset

$$R1 \oplus S(IV_1, Key) \oplus R2 \oplus S(IV_2, Key)$$

# Stream cipher-based solution

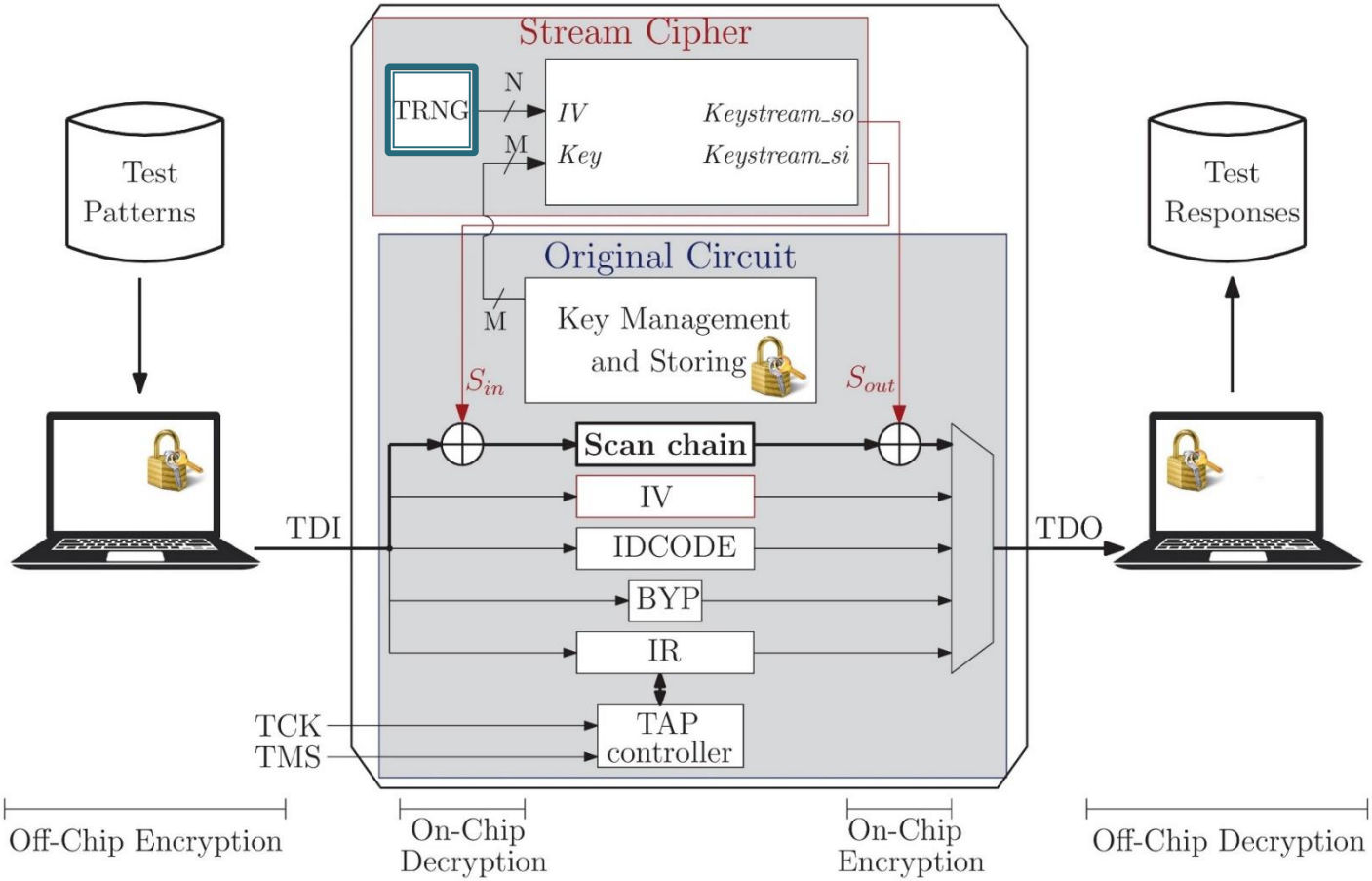
- 1 TRIVIUM stream cipher (2 016 GE)
  - 2 Keystreams
  - True Random Number Generator (TRNG) to generate random *IV*
- E.g. on JTAG, new instruction *GetIV* with a test data register IV




- Mode of operations in 2 phases: initialization and encryption

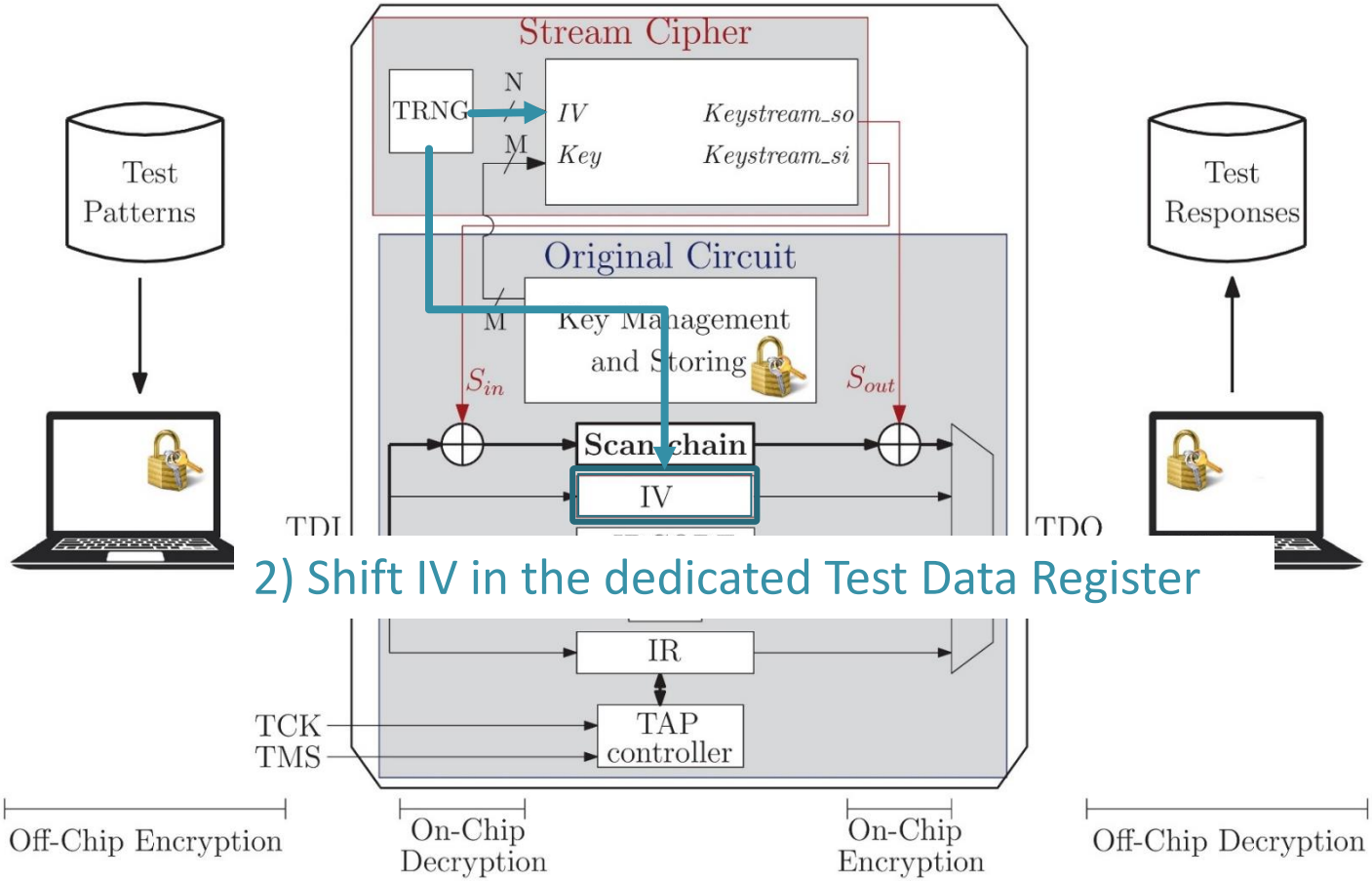
# Initialization phase

## 1) TRNG initialization: reach sufficient entropy to generate random number



# INITIALIZATION PHASE



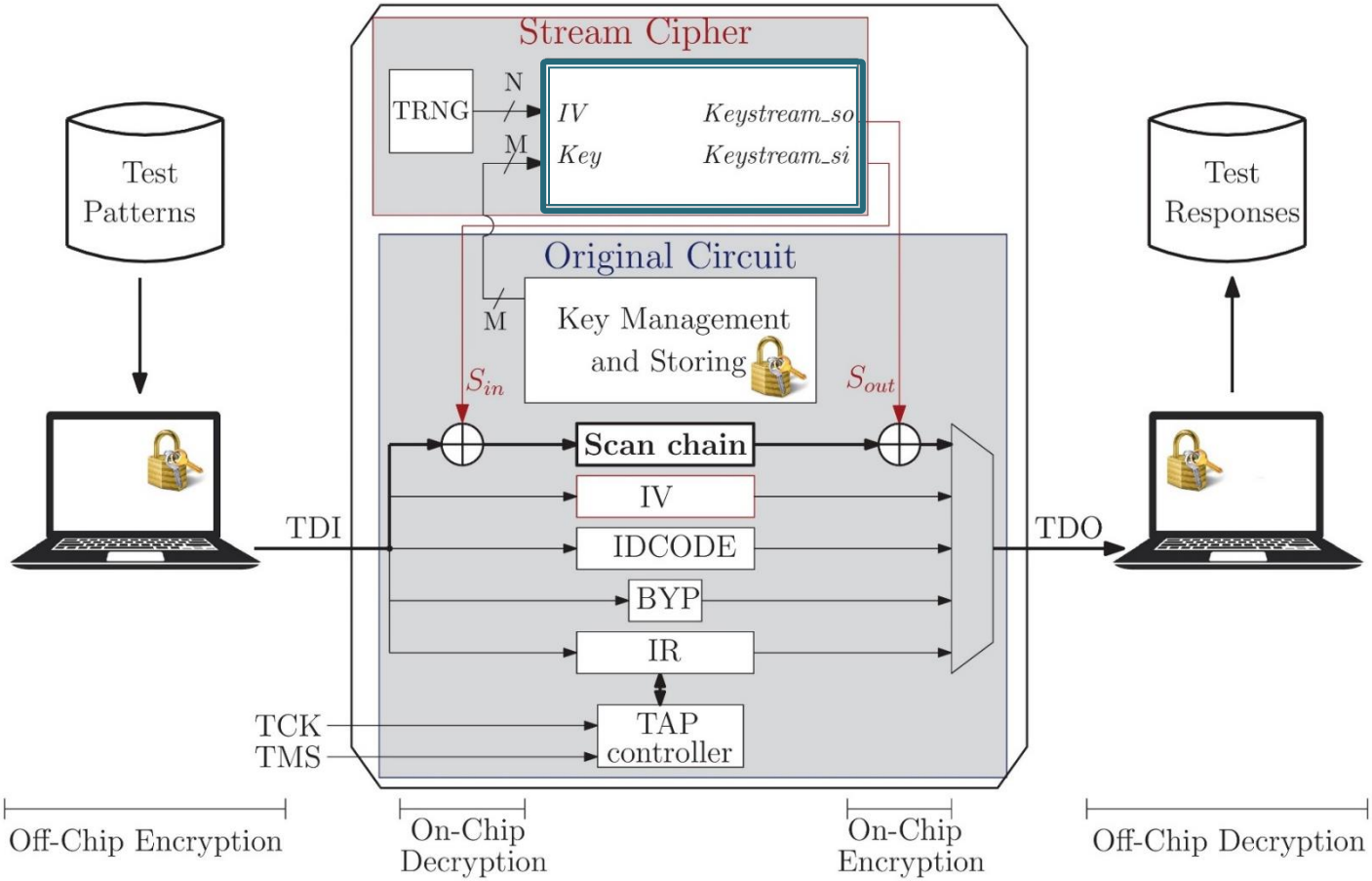



2) Shift IV in the dedicated Test Data Register

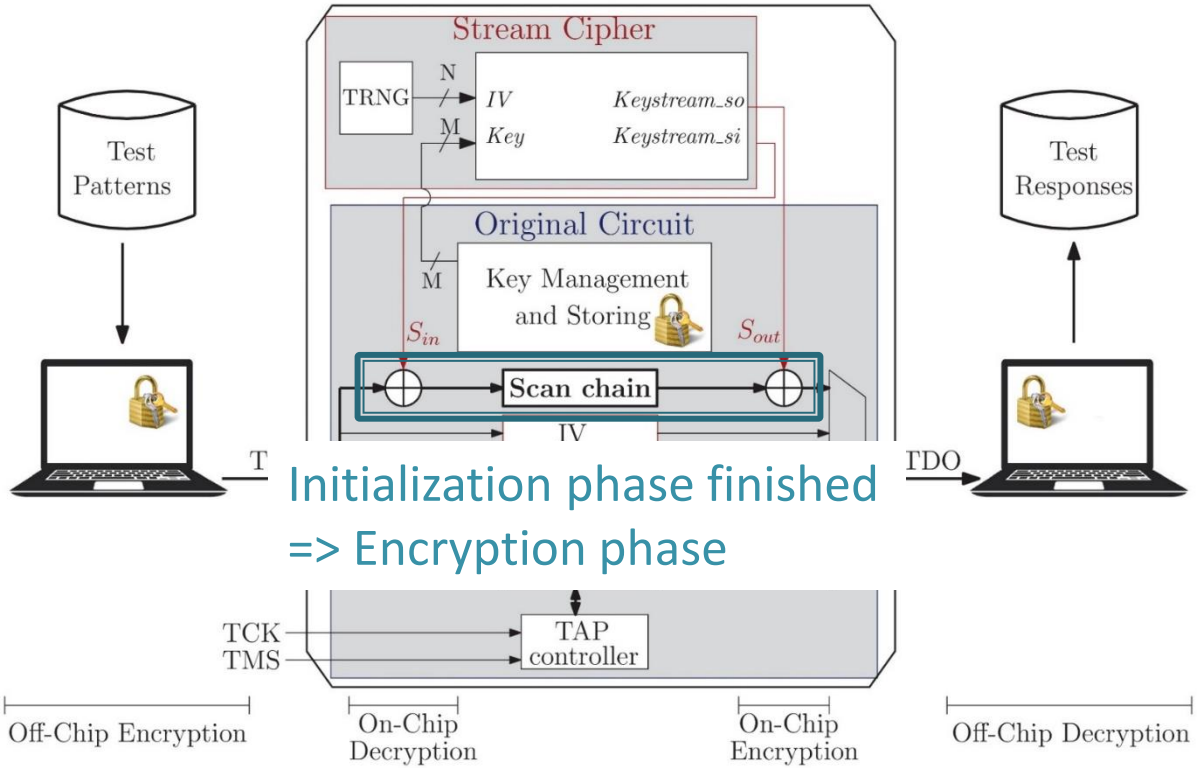
# INITIALIZATION PHASE



## 3) Stream cipher setup



# Initialization phase

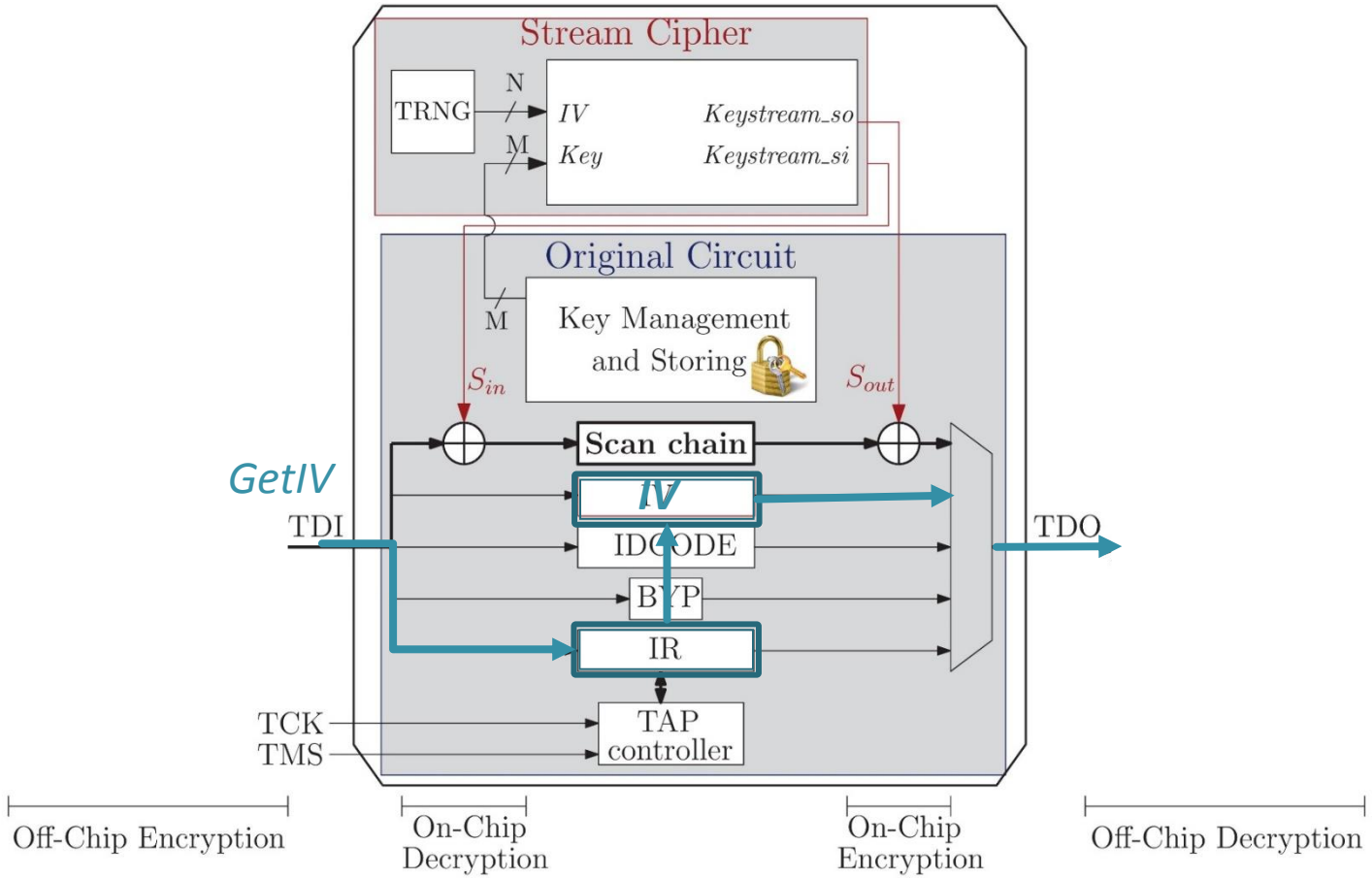



- Test time overhead:
  - $T_{TRNG\_init}$ to initialize the TRNG
  - 80 clock cycles to shift the *IV* in the register
  - 1 152 clock cycles for the stream cipher setup

# Encryption phase

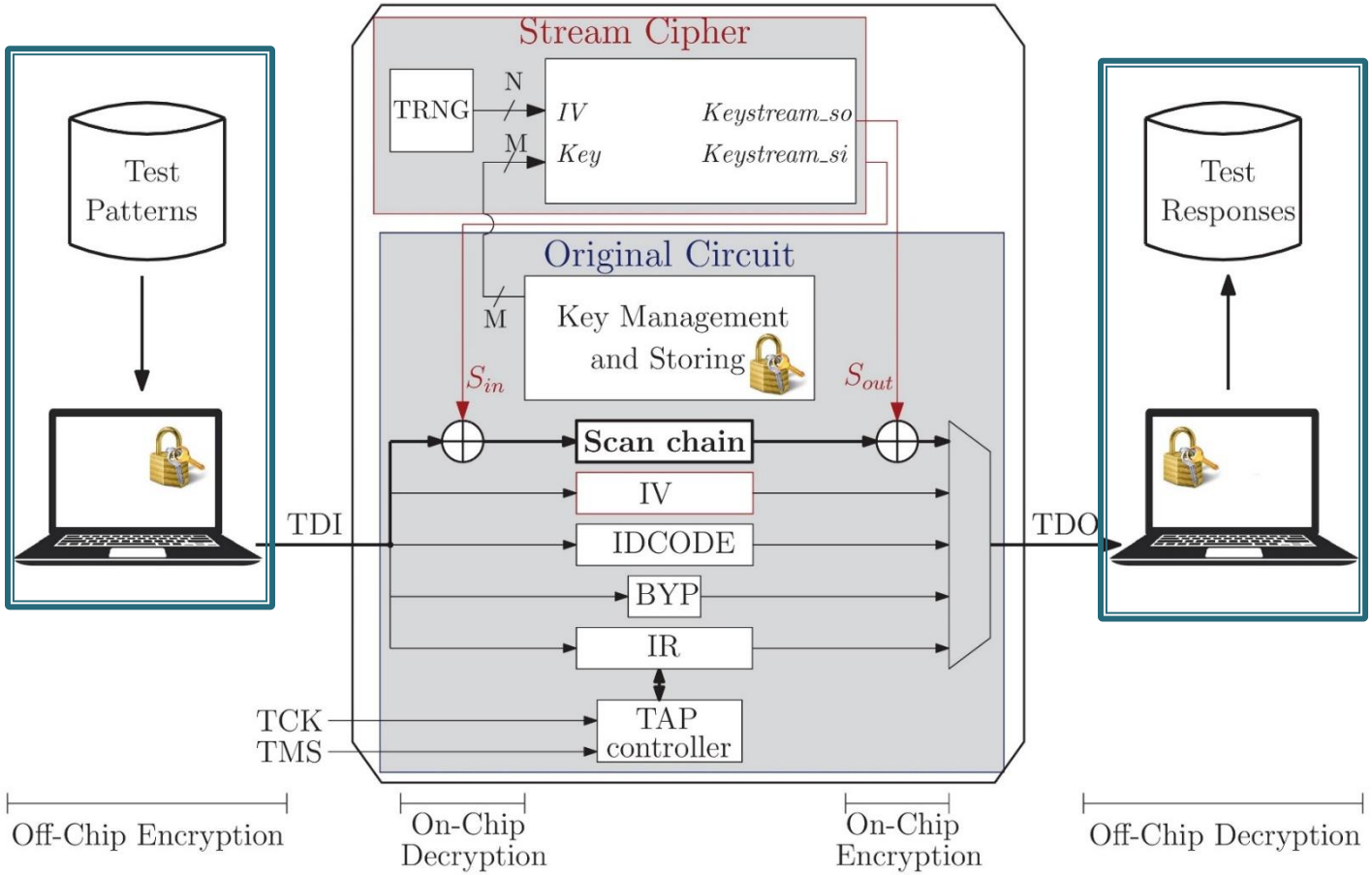
- Send *GETIV* instruction

⇒ Shift the content of the IV register out the circuit



# Encryption phase

- User can encrypt and decrypt test data with the **obtained *IV*** and the **shared secret key**

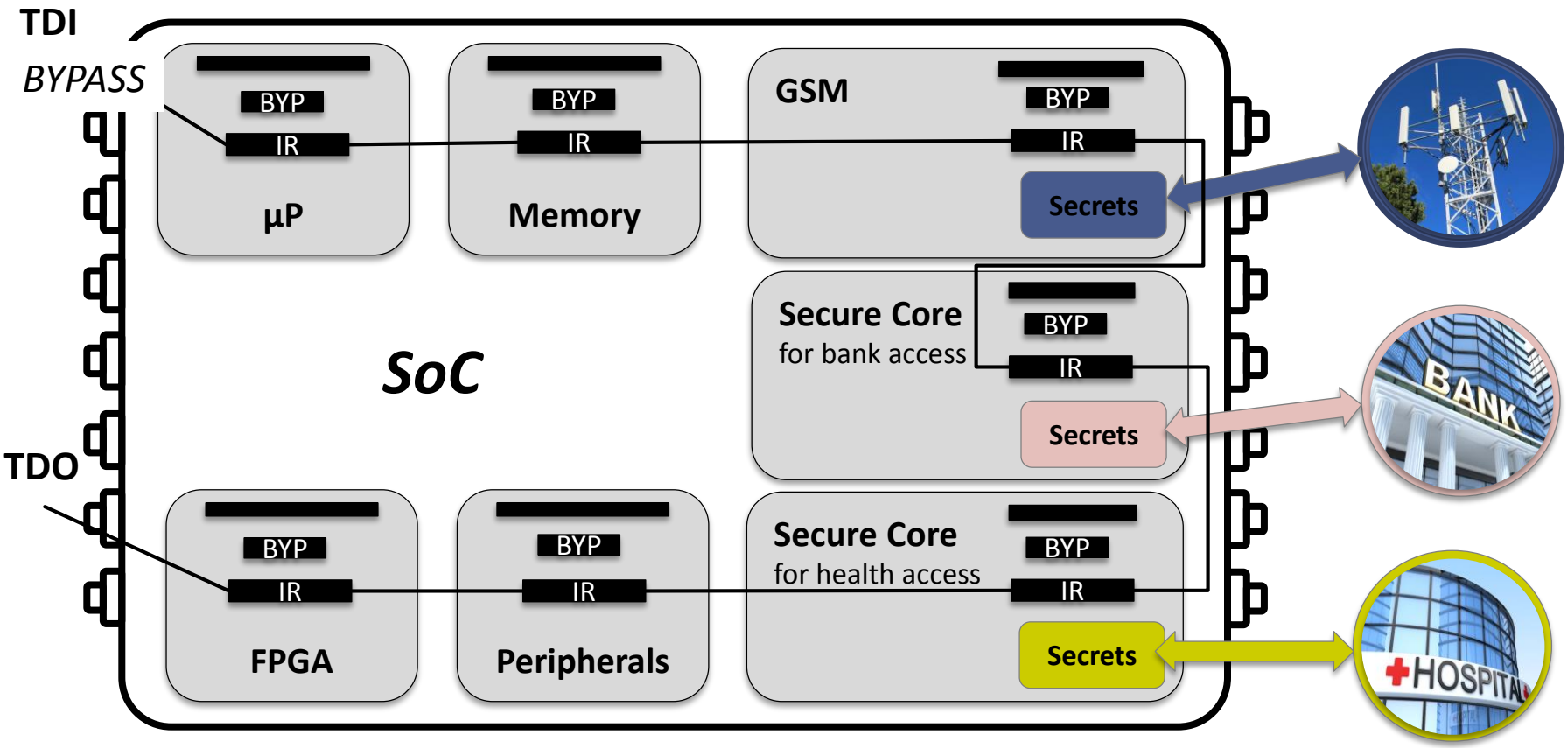


# SUMMARY

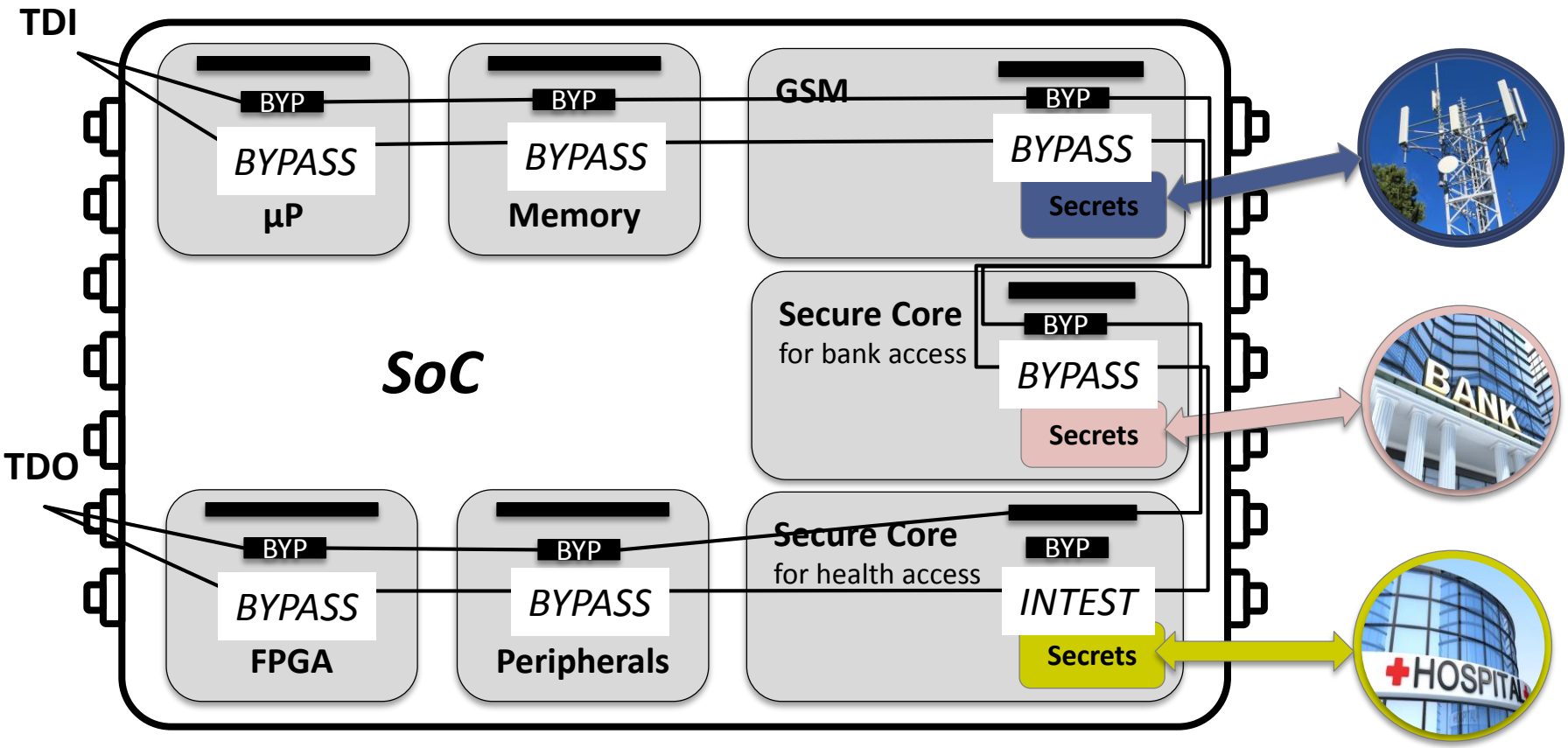
1) Context of testing
2) Threats related to the test infrastructures
3) Proposed countermeasures: Scan Encryption
4) **Application of the proposed countermeasures**
   - Integration in a SoC design
   - General advantages
   - Comparison between both implementations
5) Conclusion

# Example of SoC design

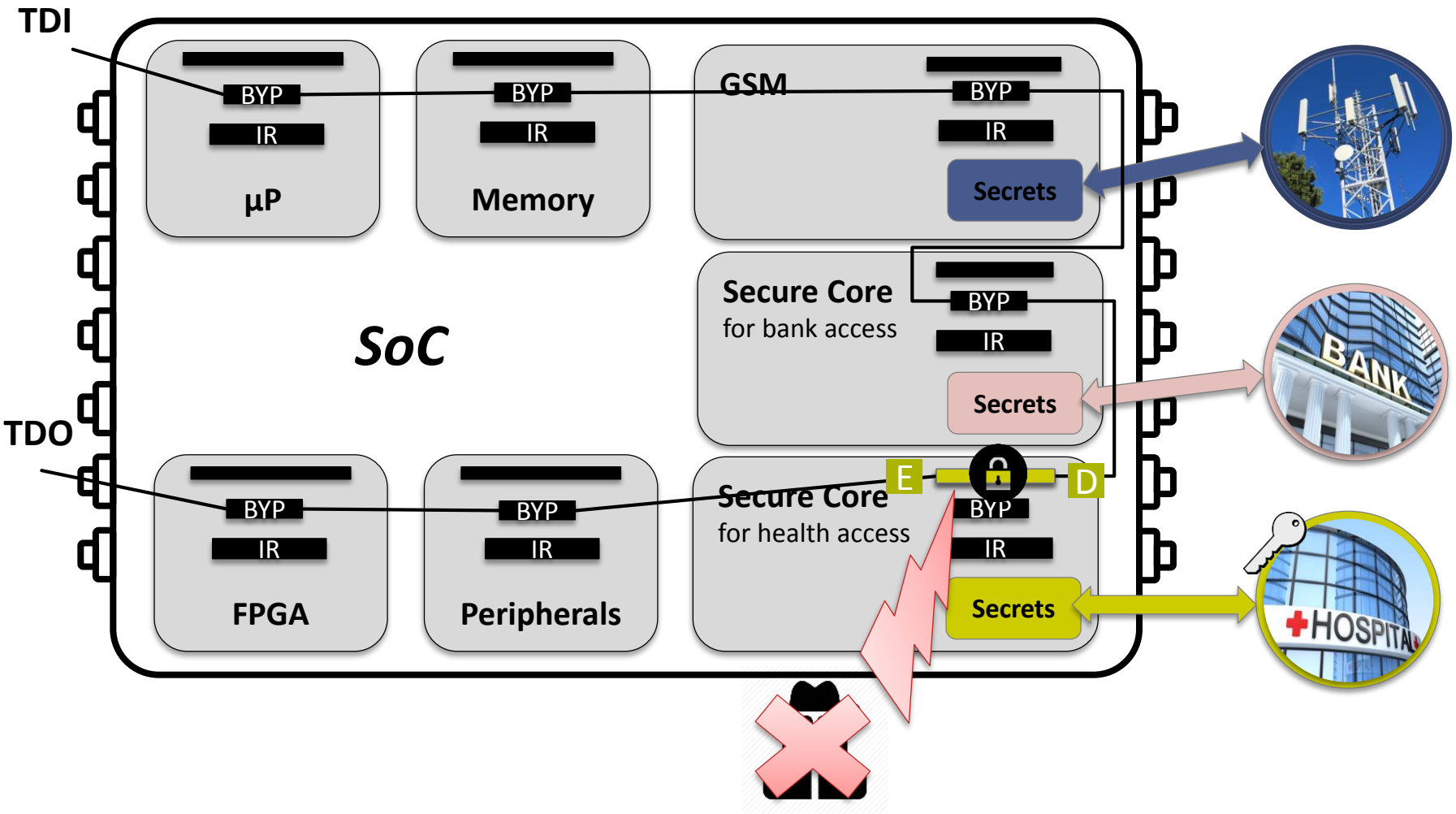




# Instructions shifted in IR registers

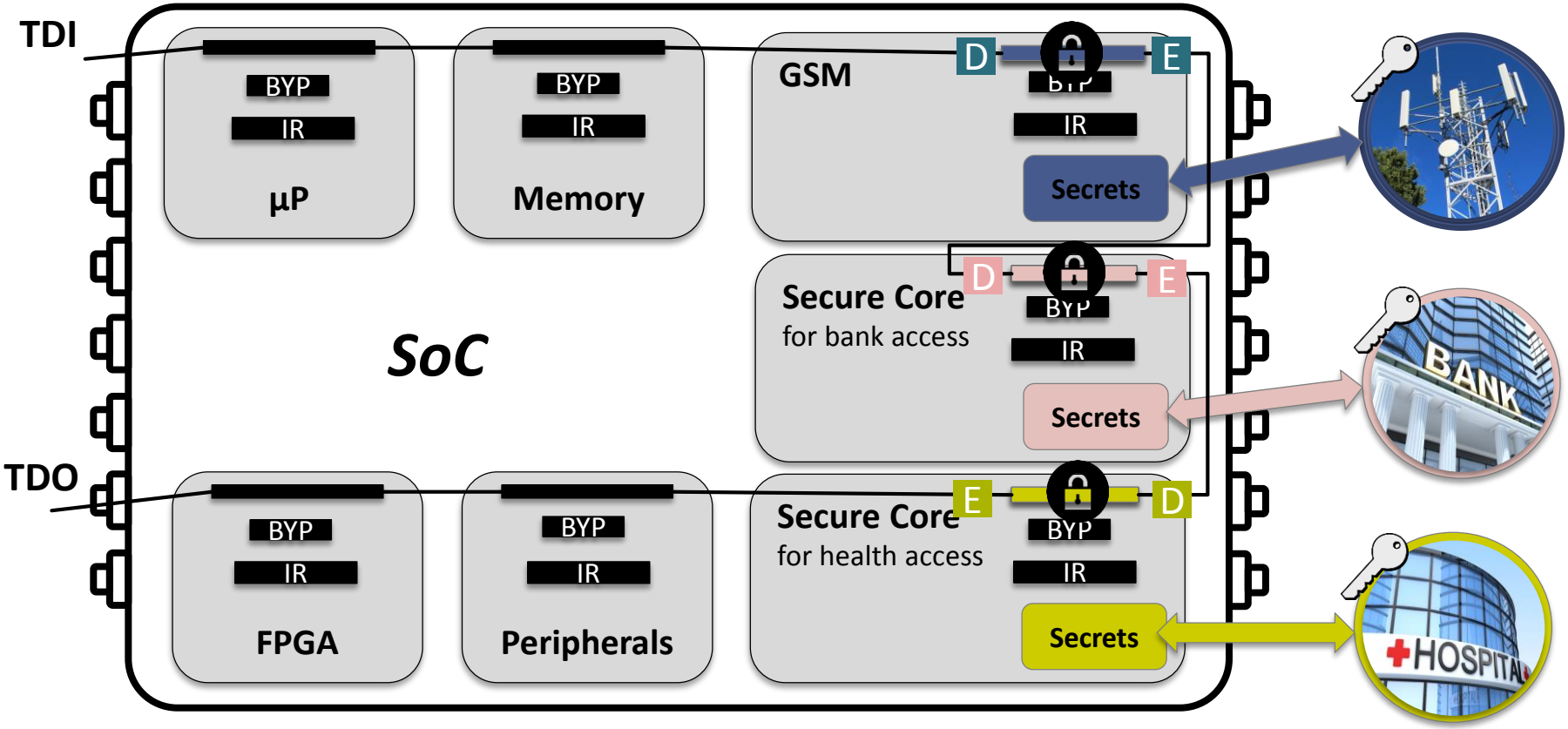


# Integration of scan encryption



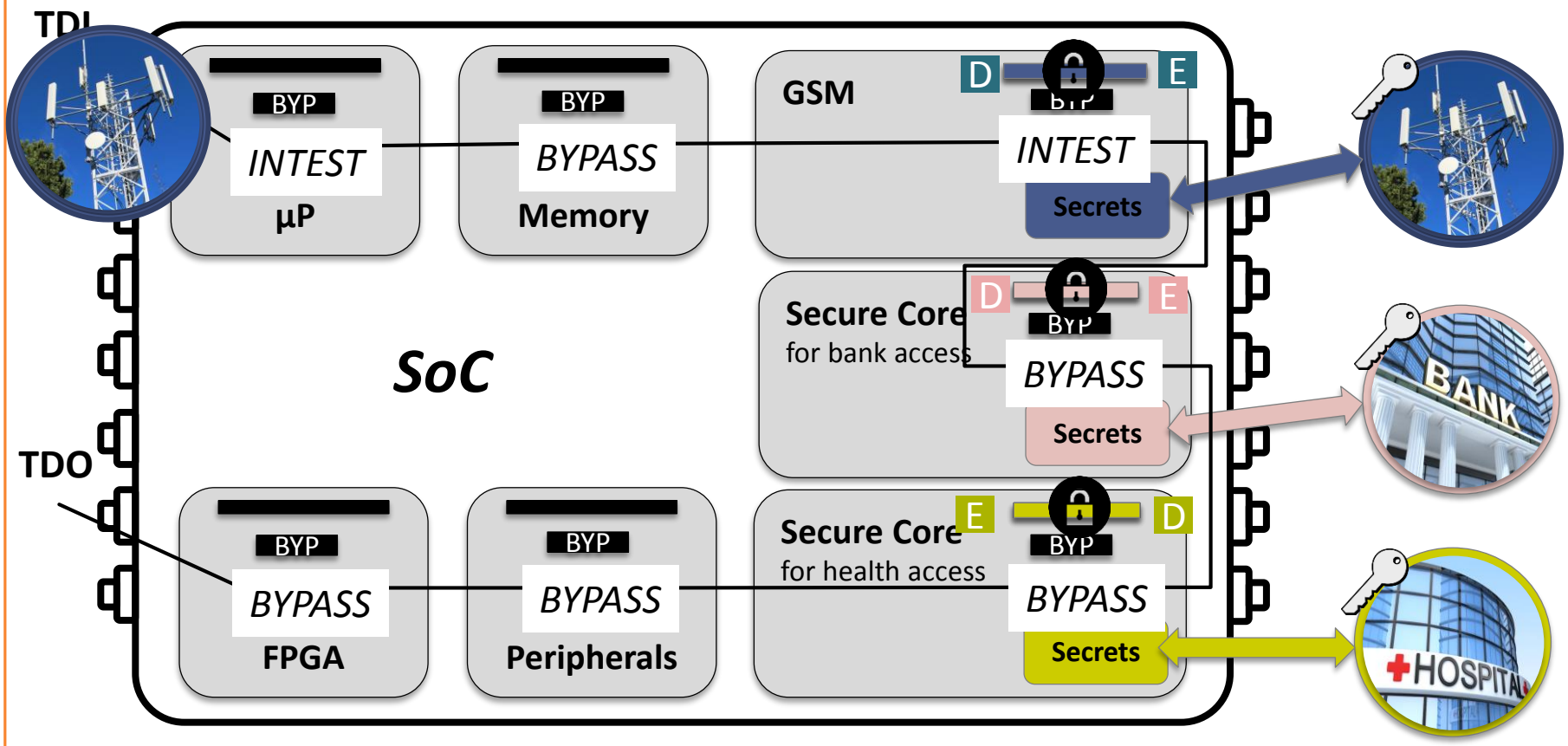
# Fine-grained access

- Allow to distinguish between different group of users



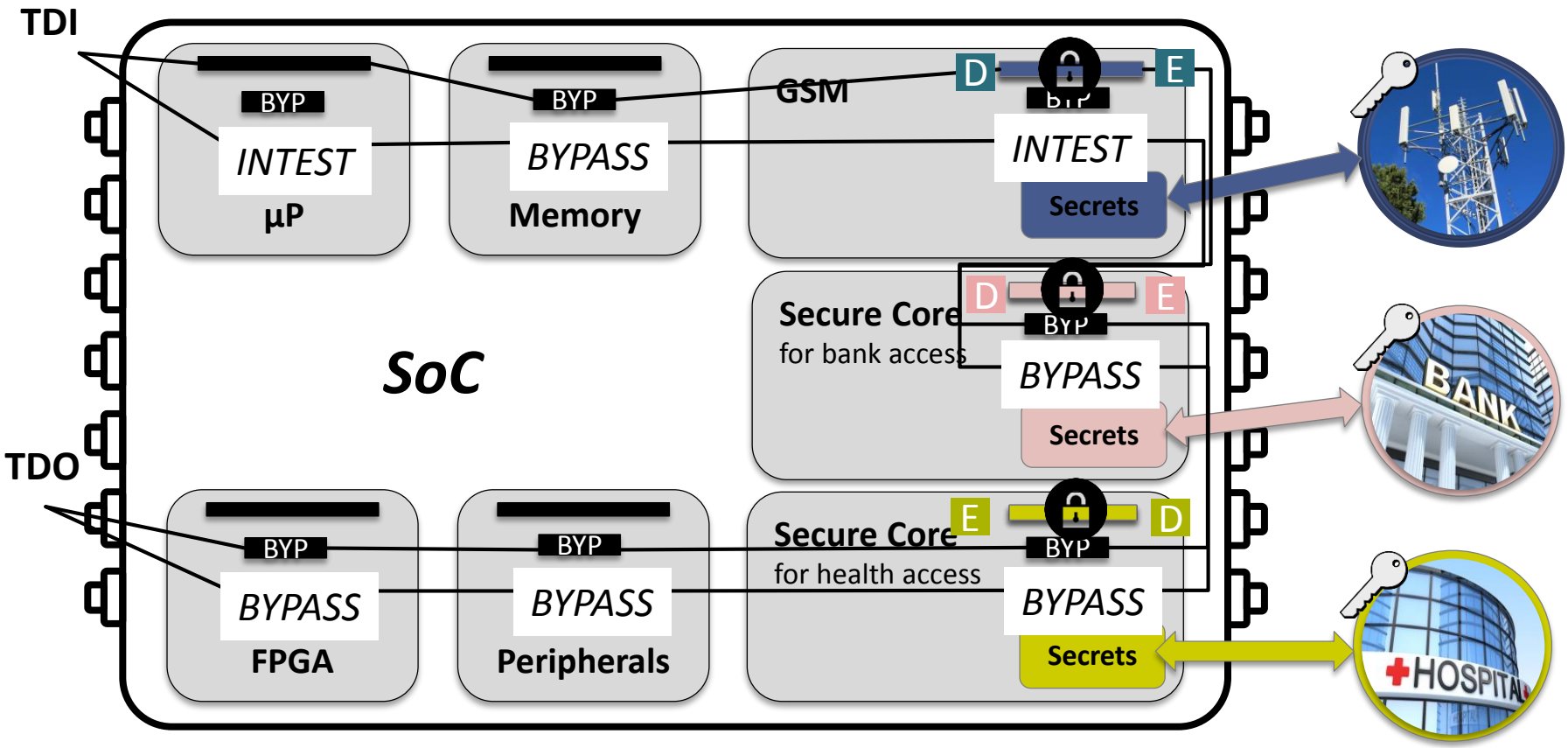
# Example

- Test in the SoC of **µP** and **GSM module** by **GSM operator**



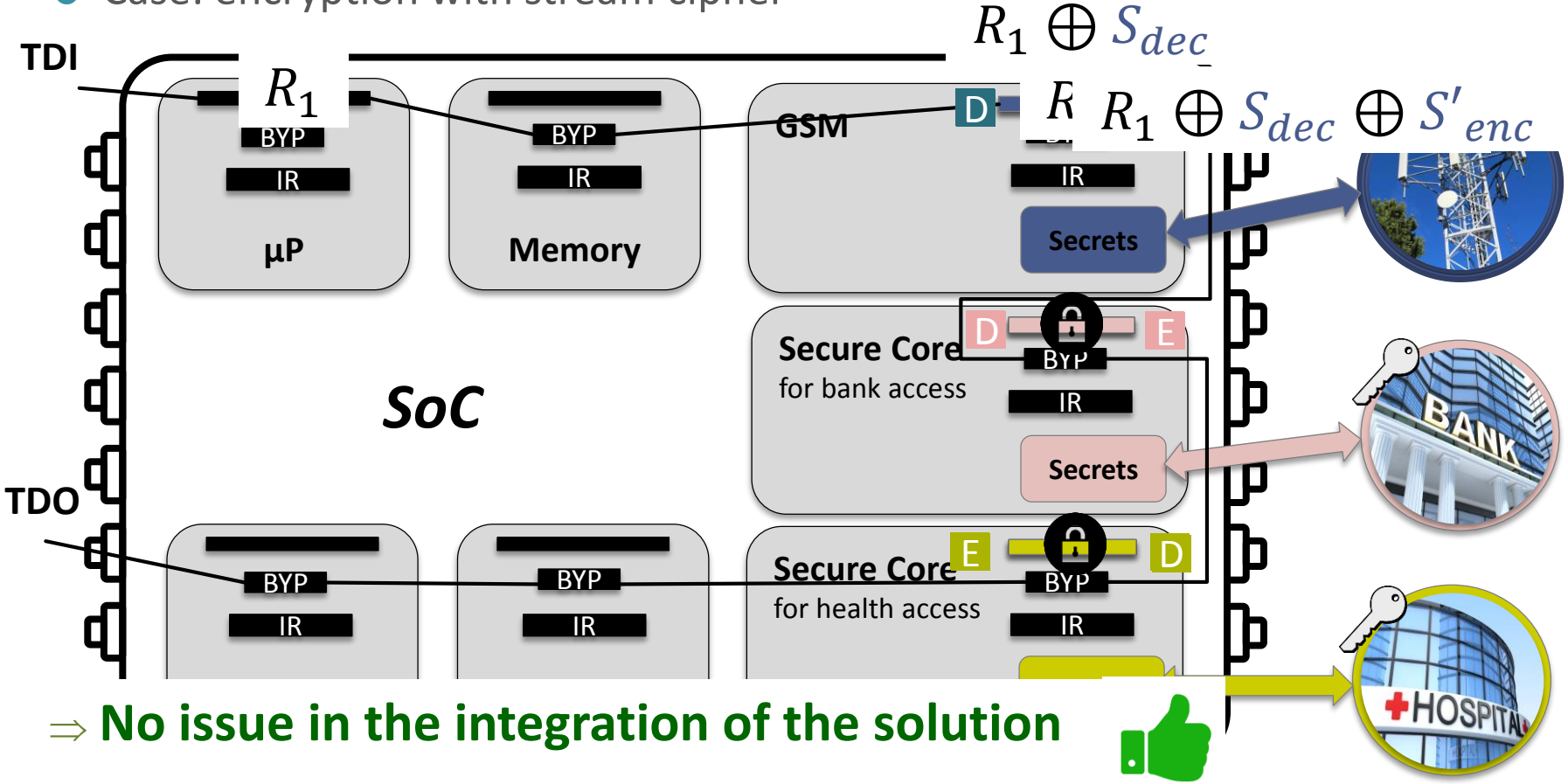
# Example

- Test in the SoC of **µP** and **GSM module** by **GSM operator**



# Example

- Test in the SoC of **µP** and **GSM module** by **GSM operator**
- Case: encryption with stream cipher




⇒ **No issue in the integration of the solution**

# General advantages



- Advantages of scan encryption solutions (both stream and block encryption):

+ **Security**
  - Protected against scan attacks
  - Protected against malicious core

+ **Diagnosis and debug preserved**
  - Still possible in-field

+ **Key management**
  - Re-use key management already implemented

+ **Integration in a SoC design**
  - No issue

# Comparison

- Block cipher vs stream cipher

| | Stream cipher-based solution | | Block cipher-based solution | |
|---|---|---|---|---|
| Conditions on the original circuit | TRNG already implemented | No TRNG implemented | Scan chain length not multiple of 64 | Scan chain multiple of 64 (insertion of test points) |
| ***Cost*** | | | | |
| - Area |  |  |  |  |
| - Test time |  |  |  |  |

# Summary

1) Context of testing
2) Threats related to the test infrastructures
3) Proposed countermeasures: Scan Encryption
4) Application of the proposed countermeasures
5) **Conclusion**

# Conclusion

- Need a protection on the test infrastructures (even with TEE)
  - ⇒ Data saved and processed in Secure world can be controlled and observed through the scan chains
- Solution consisting in disconnecting test accesses
  - ⇒ Important issues with in-field diagnosis and debug
  - ⇒ Security threats with probing attacks
- Proposition of Scan Encryption countermeasures
  - ⇒ Preserve diagnosis and debug only for authorized users
  - ⇒ Prevents both external and internal attacks exploiting test infrastructures
  - ⇒ Study of two implementations (block cipher and stream cipher)


Thank You