



HAL
open science

Scan chain encryption, a countermeasure against scan attacks

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

► To cite this version:

Mathieu da Silva, Giorgio Di Natale, Marie-Lise Flottes, Bruno Rouzeyre. Scan chain encryption, a countermeasure against scan attacks. PHISIC: Practical Hardware Innovations in Security Implementation and Characterization, May 2018, Gardanne, France. , Workshop on Practical Hardware Innovations in Security Implementation and Characterization, 2018. lirmm-01882565v2

HAL Id: lirmm-01882565

<https://hal-lirmm.ccsd.cnrs.fr/lirmm-01882565v2>

Submitted on 10 Oct 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



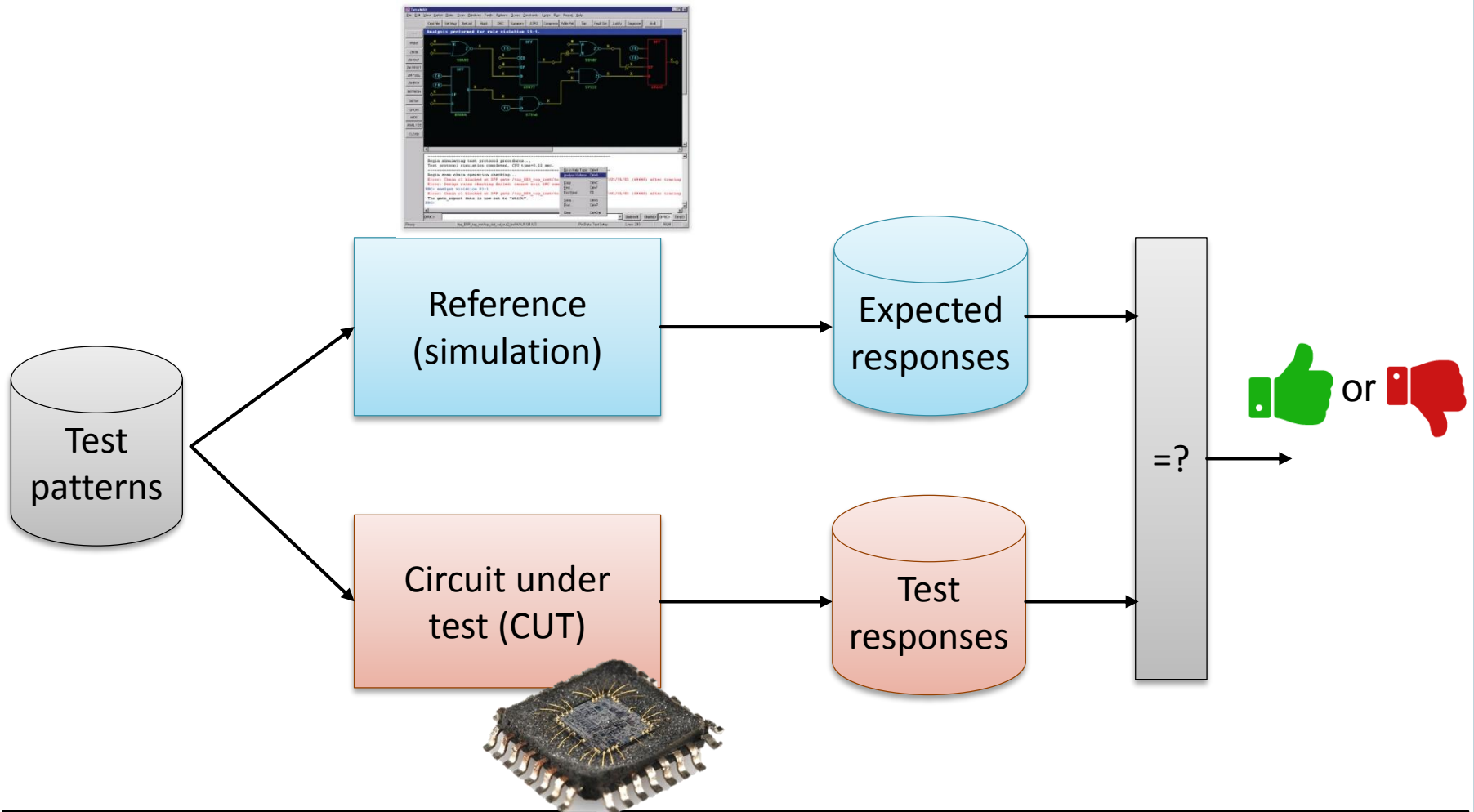
SCAN CHAIN ENCRYPTION, A COUNTERMEASURE AGAINST SCAN ATTACKS

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

PHISIC 2018

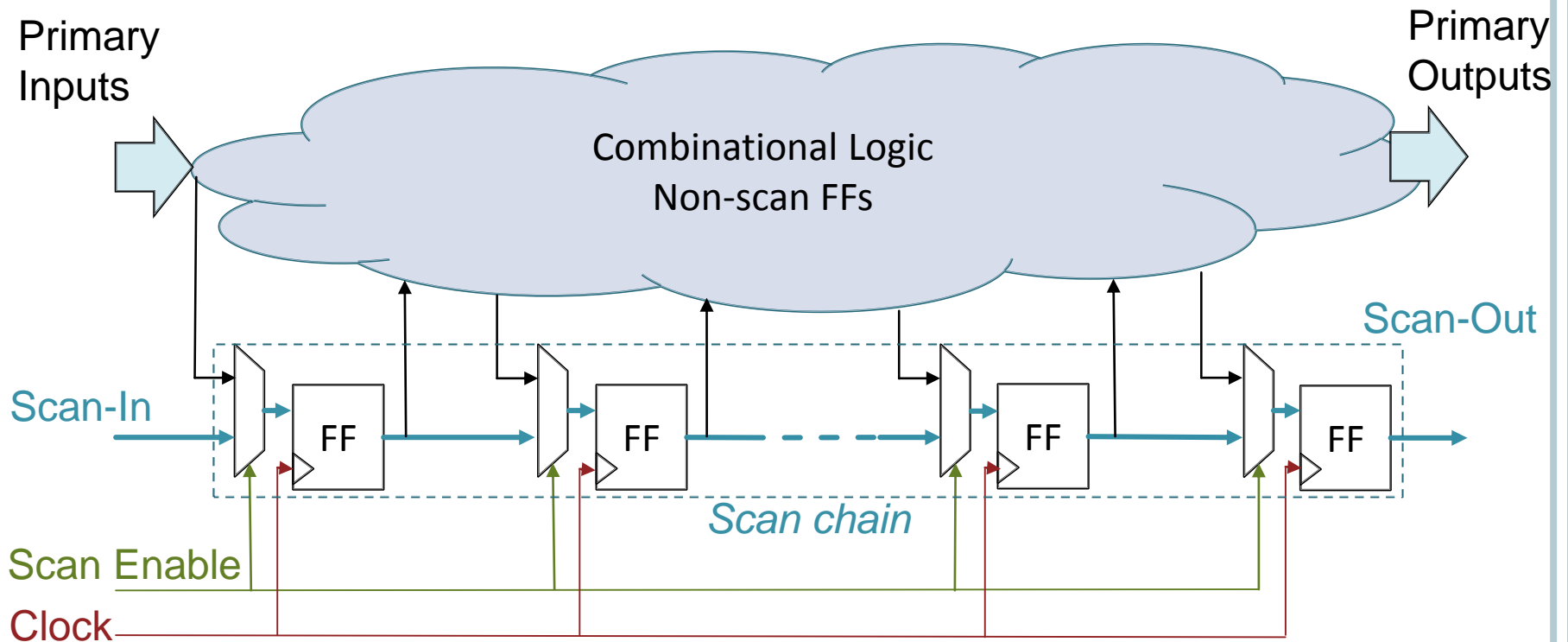
CONTEXT

- Test of circuit is a mandatory step in IC production



CONTEXT

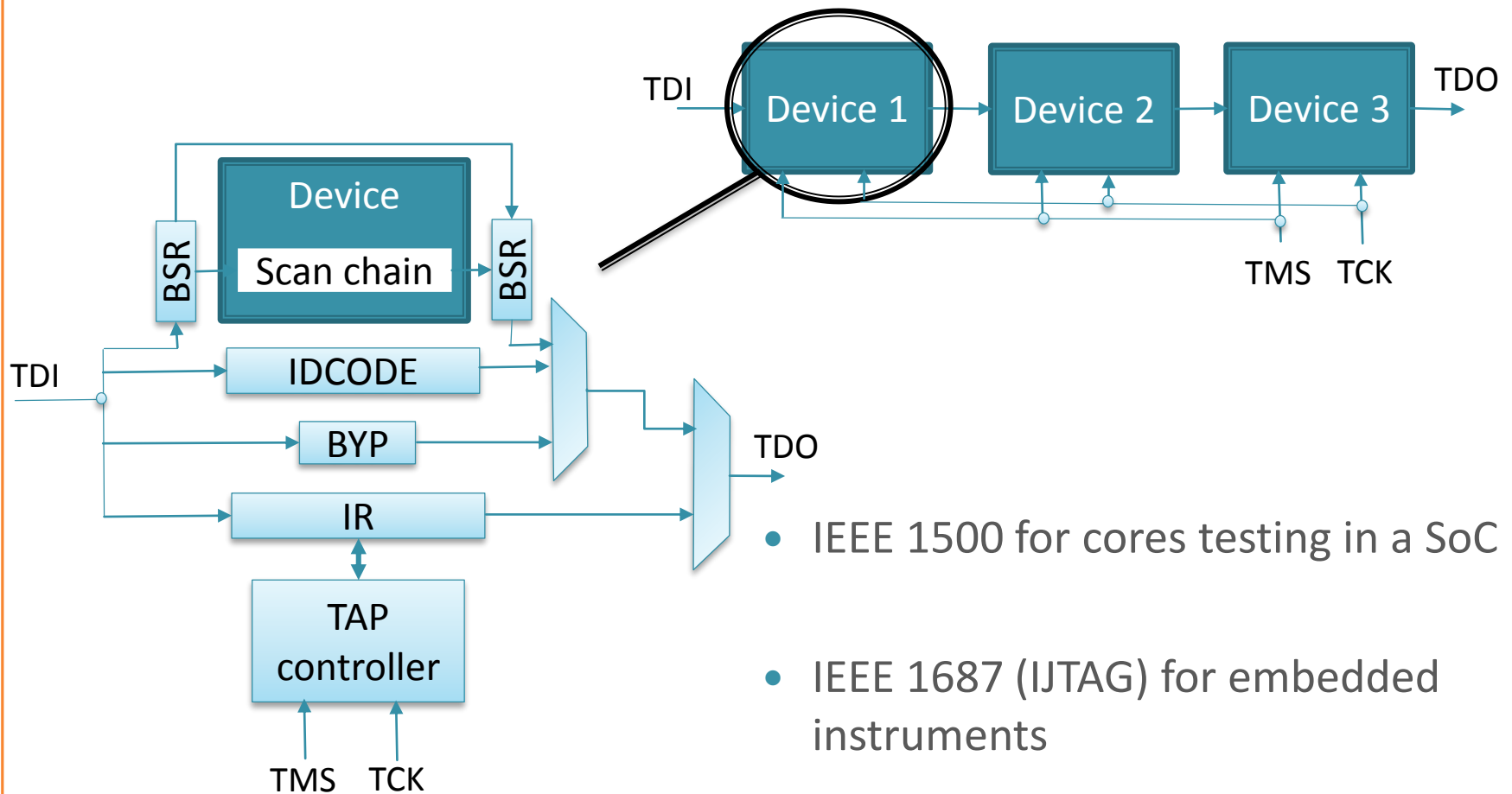
- Most popular method for Design-for-Test = Scan chains
 - Replace original FF by Scan FF connected serially together
 - Extra port « Scan-In » => controllability on internal states
 - Extra port « Scan-Out » => observability on internal states



CONTEXT

- Test standards

- IEEE 1149 (JTAG) for board testing

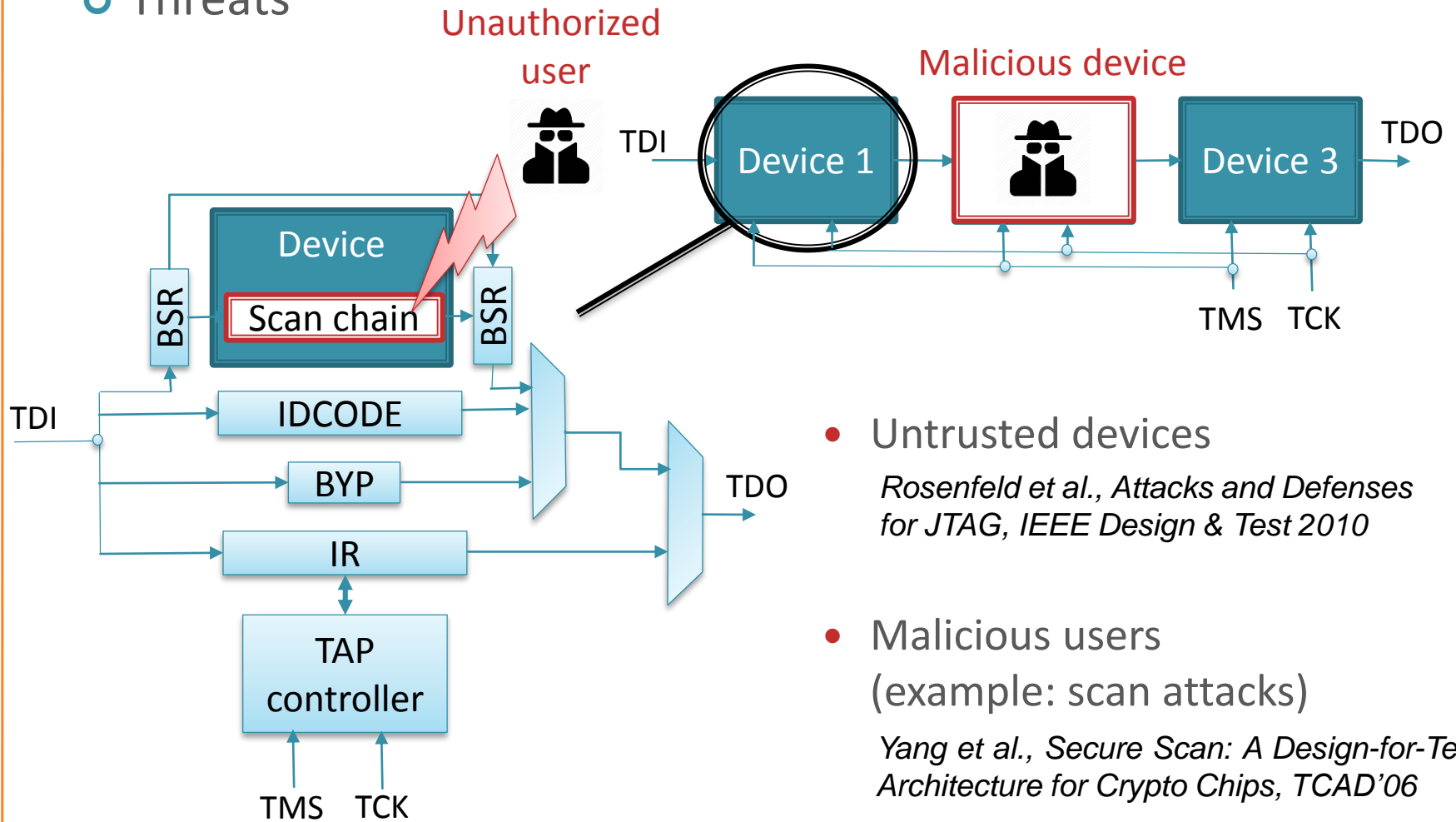


- IEEE 1500 for cores testing in a SoC
- IEEE 1687 (IJTAG) for embedded instruments



CONTEXT

○ 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



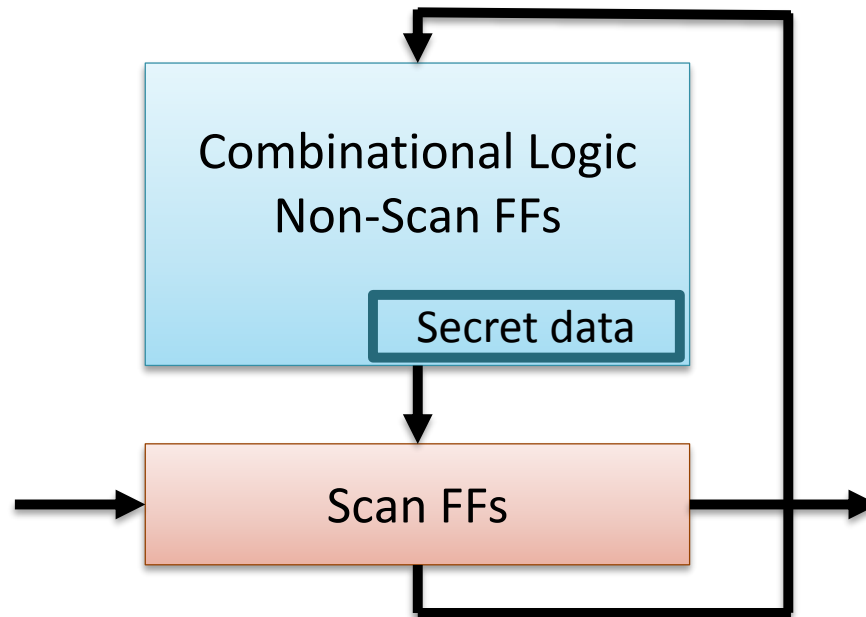
SUMMARY

- 1) **Scan attacks**
- 2) A new countermeasure: Scan chain encryption
- 3) Implementation with block cipher
- 4) Implementation with stream cipher
- 5) Conclusion



SCAN ATTACK PRINCIPLE

- Goal: Retrieve embedded secret data
- Exploit observability or controllability offered by scan chains
- Principle: switch between functional and scan modes
- Main target: secret key of crypto-processors (example: AES)



SCAN ATTACK ON AES

- Advanced Encryption Standard (AES)

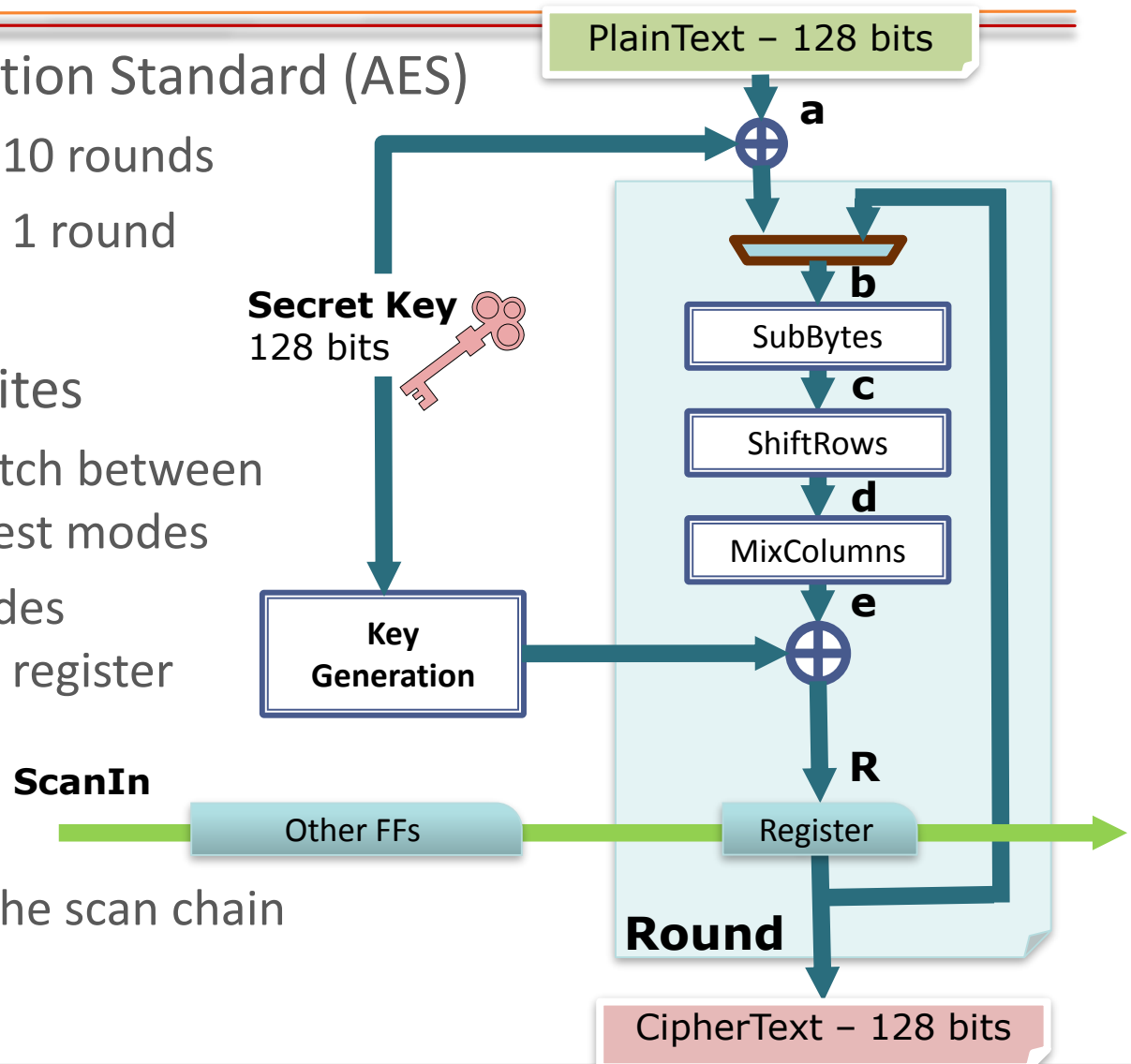
- Ciphertext after 10 rounds
- Not secure after 1 round

- Attack pre-requisites

- Attacker can switch between functional and test modes
- Scan chain includes FFs of the round register

- Attack principle

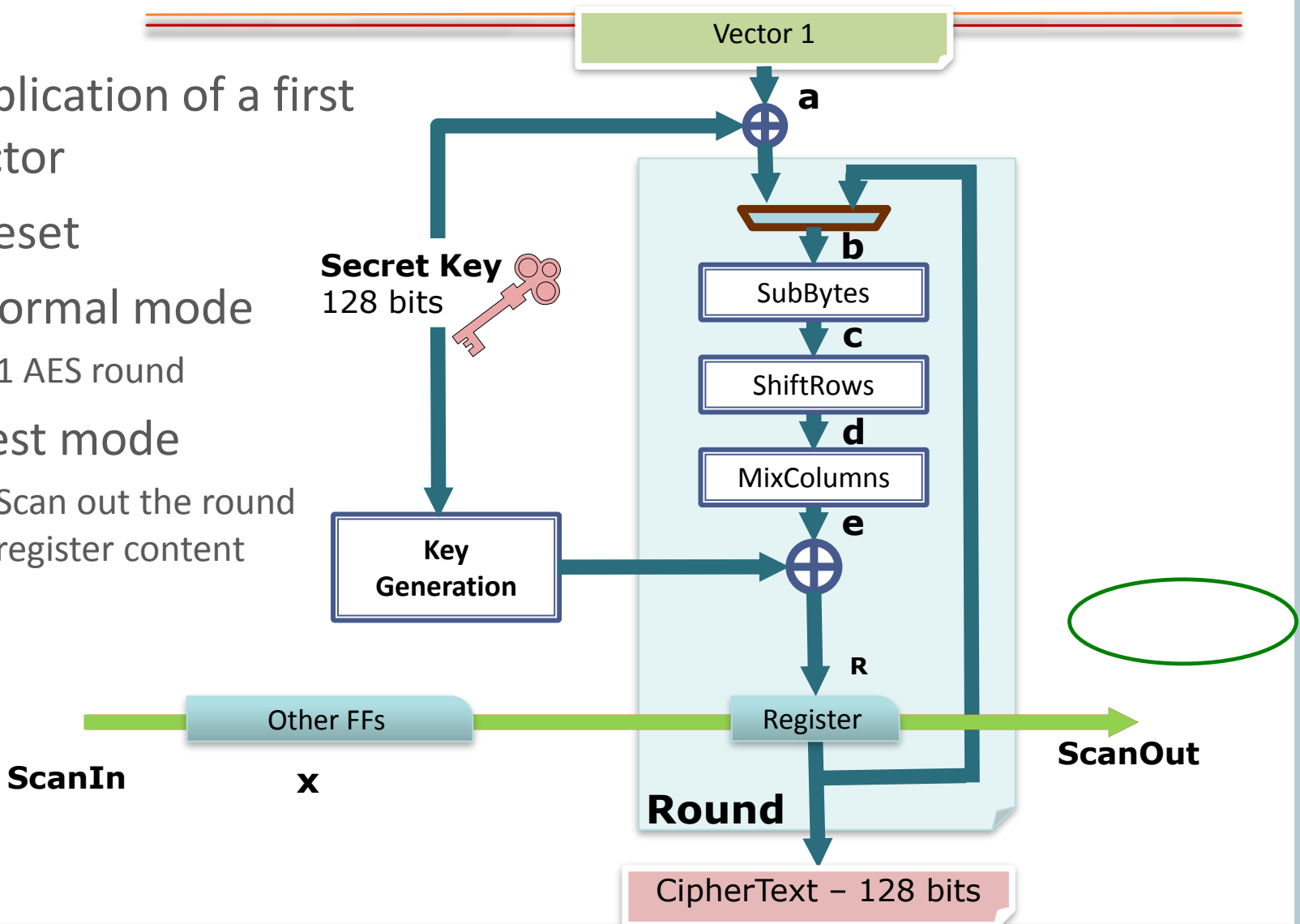
- Observation of the scan chain after 1 round



DIFFERENTIAL ATTACK

○ Application of a first vector

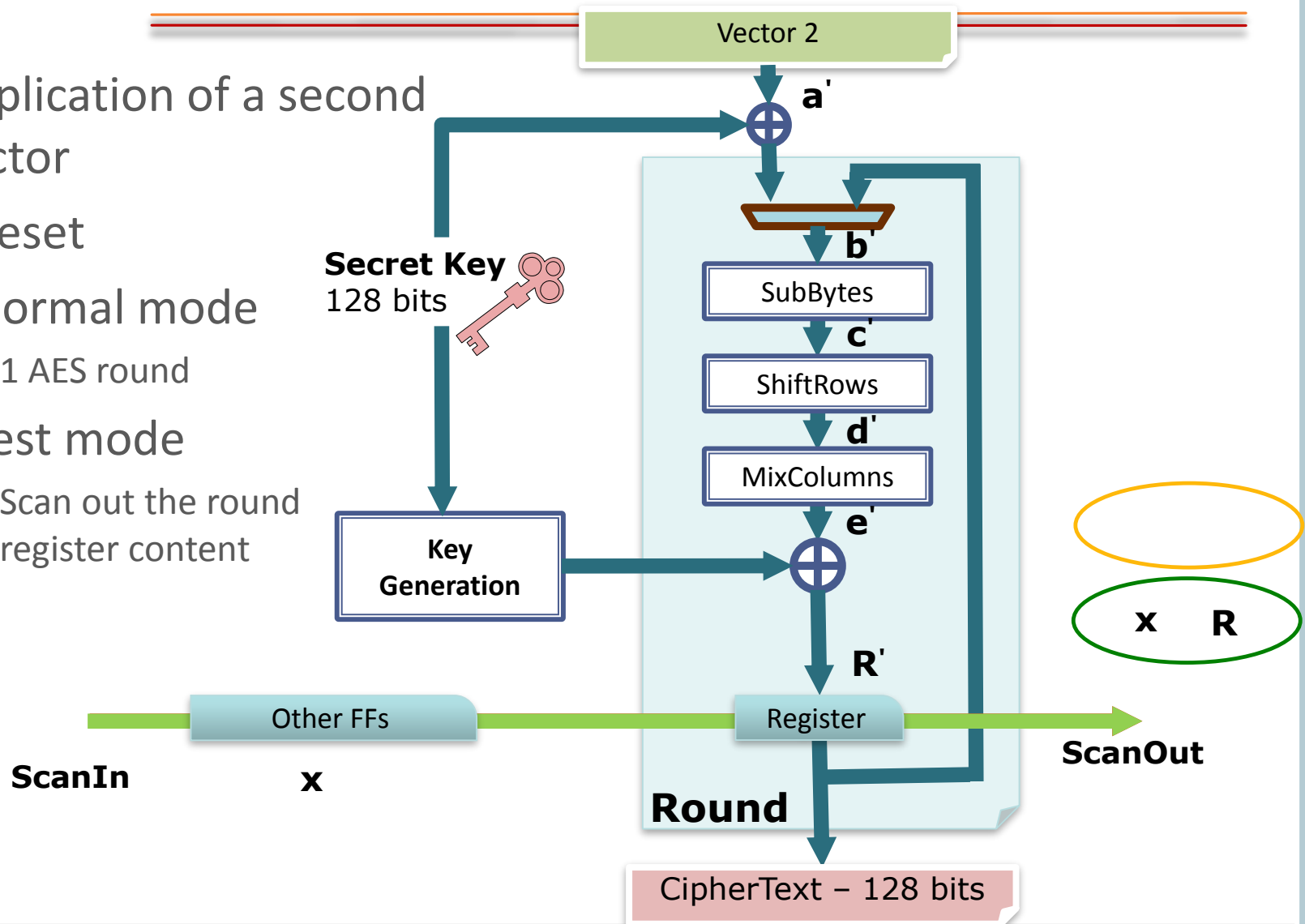
- 1) Reset
- 2) Normal mode
 - 1 AES round
- 3) Test mode
 - Scan out the round register content



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



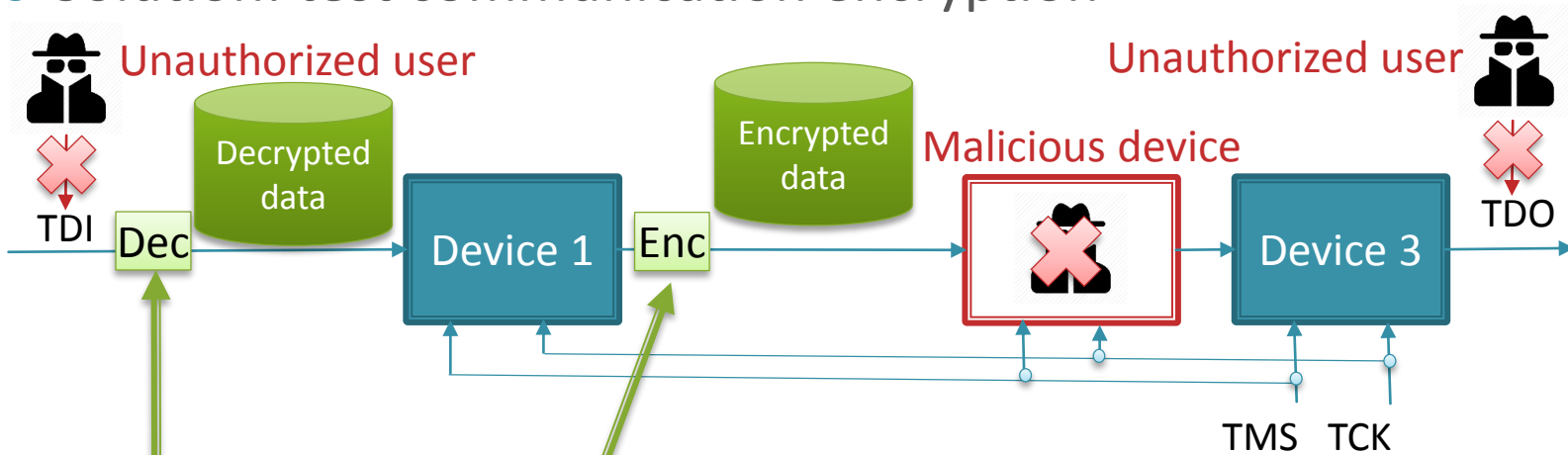
SUMMARY

- 1) Scan attacks
- 2) A new countermeasure: Scan chain encryption**
- 3) Implementation with block cipher
- 4) Implementation with stream cipher
- 5) Conclusion



SCAN CHAIN ENCRYPTION

- Solution: test communication encryption

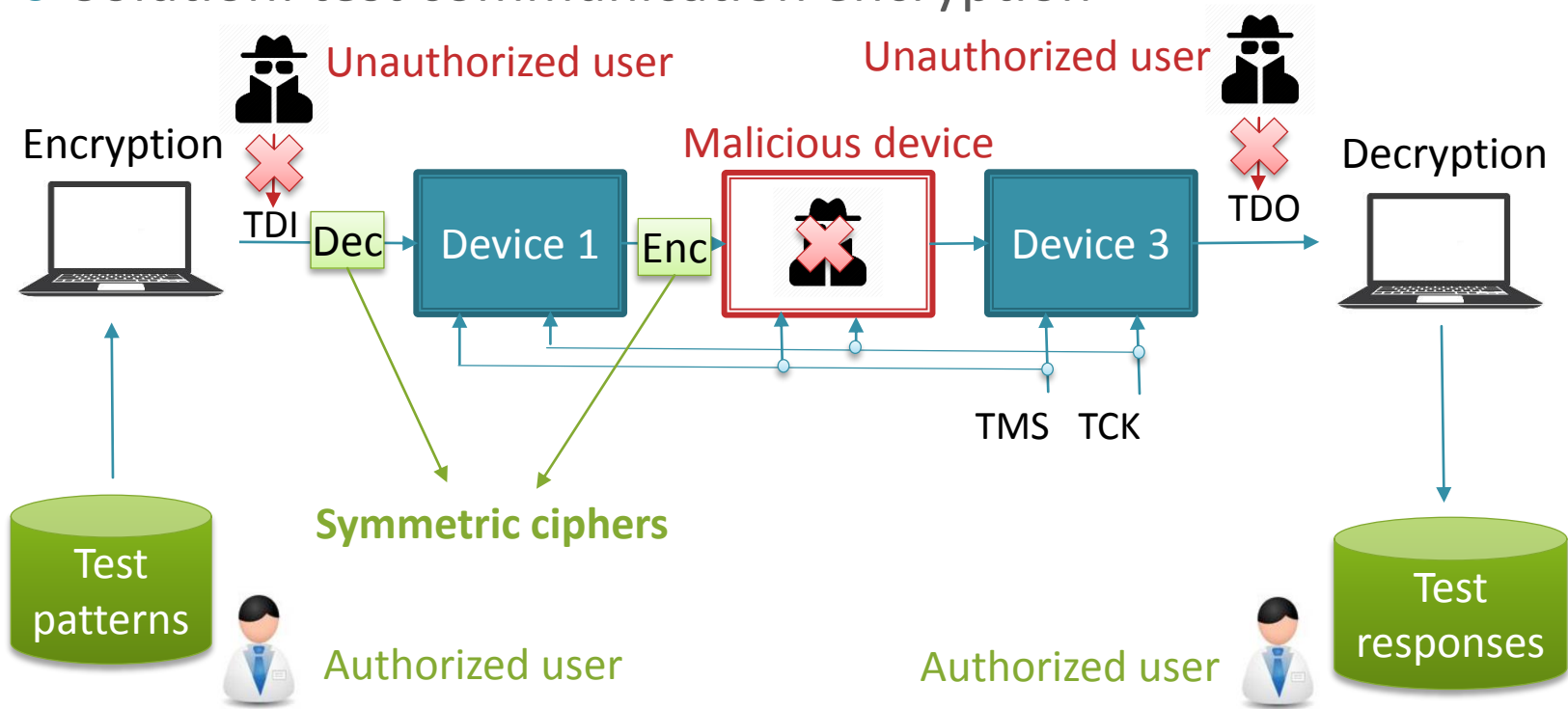


- **Input decryption** prevents sending desired test data
- **Output encryption** prevents reading plain test responses



CONTEXT

○ Solution: test communication encryption

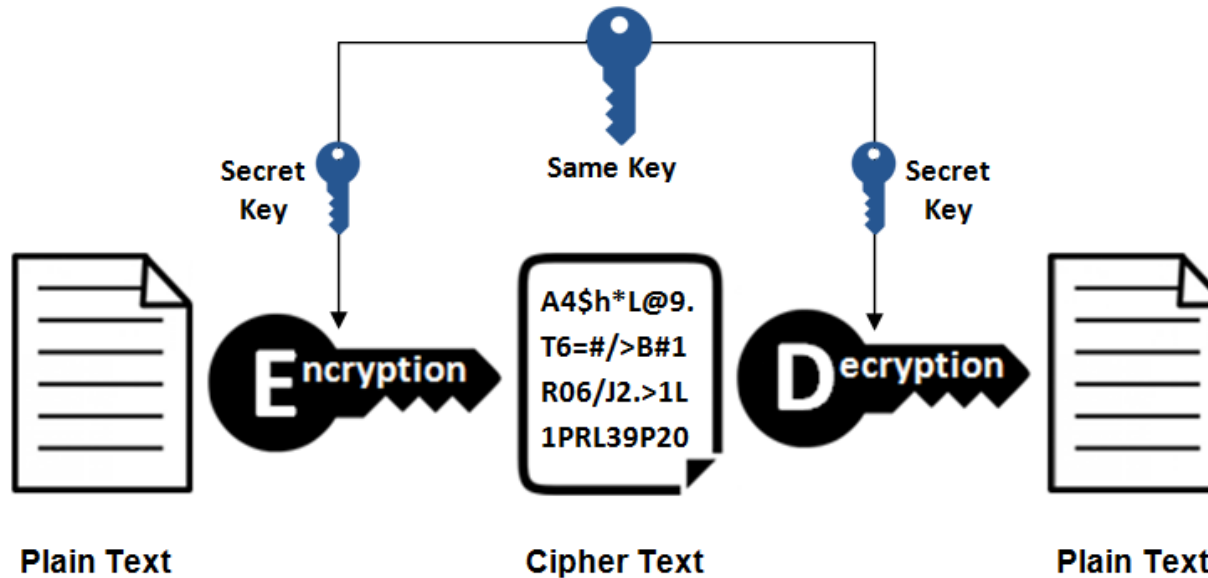


- Input decryption prevents sending desired test data
- Output encryption prevents reading plain test responses
- Test/debug only possible by authorized user knowing the secret key



SYMMETRIC CIPHER

Symmetric Encryption



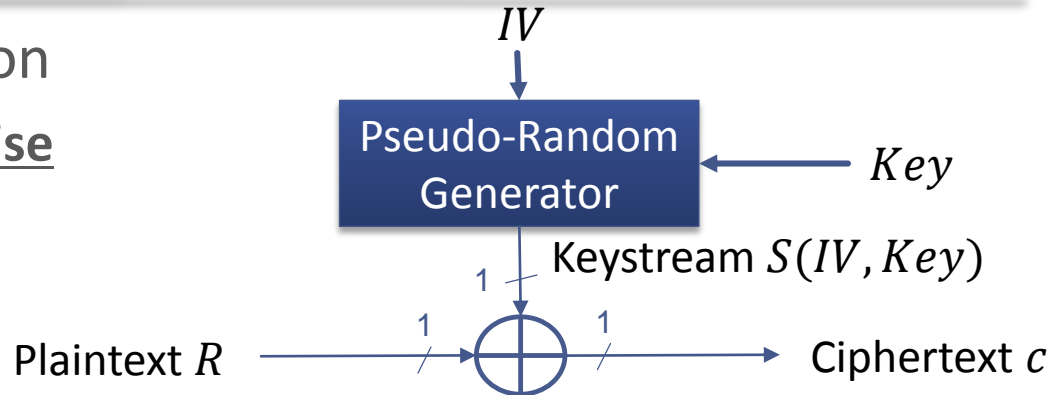
- 2 types of symmetric cipher: stream and block ciphers



STREAM CIPHER / BLOCK CIPHER

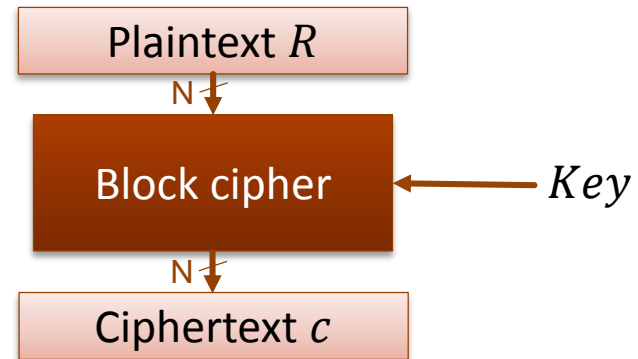
Stream cipher encryption

- Keystream XORed **bitwise** with the plaintext



Block cipher encryption

- Confusion and diffusion on a **block** of plaintext



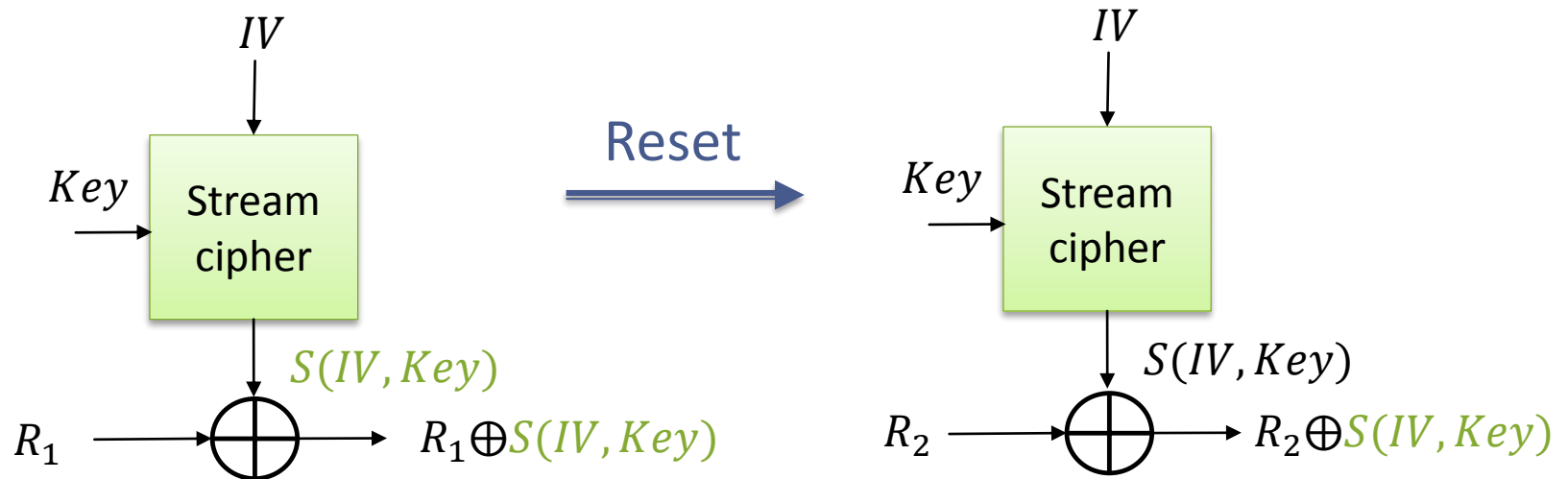
Preference for stream ciphers

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



TWO-TIMES PAD: STREAM CIPHER REQUIREMENT

- Two-times pad: same key and IV re-used \Rightarrow same keystream generated to encrypt different data



\Rightarrow Possible to carry out attacks if requirement is not fit

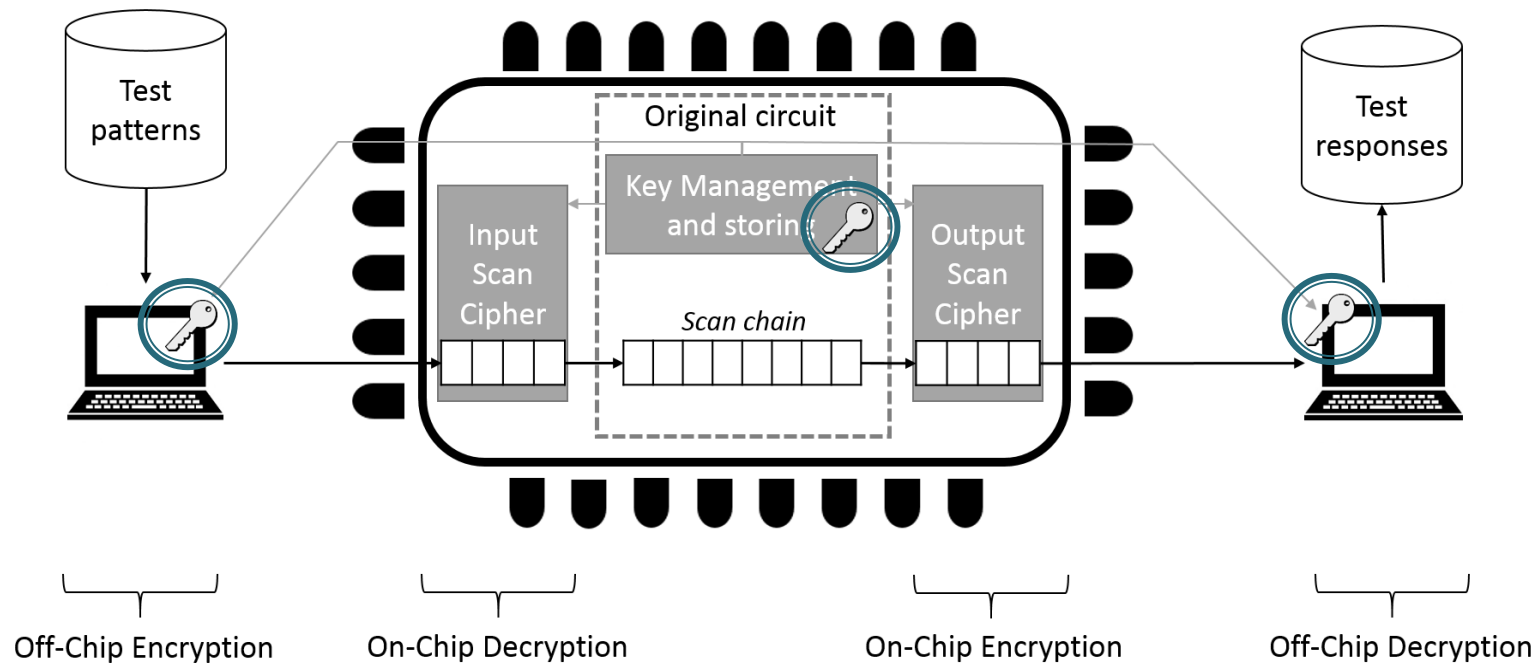
$$R_1 \oplus \cancel{S(IV, Key)} \oplus R_2 \oplus \cancel{S(IV, Key)}$$

\Rightarrow Solution: IV generated randomly at each circuit reset

$$R_1 \oplus S(IV_1, Key) \oplus R_2 \oplus S'(IV_2, Key)$$



BASIC SCHEME



- 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



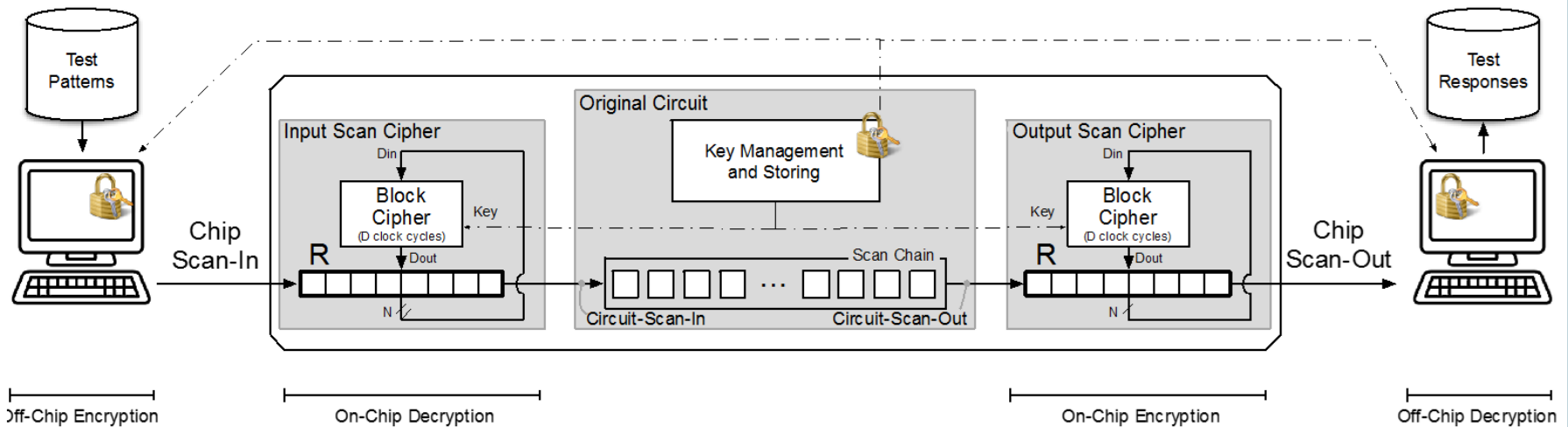
SUMMARY

- 1) Scan attacks
- 2) A new countermeasure: Scan chain encryption
- 3) Implementation with block cipher**
- 4) Implementation with stream cipher
- 5) Conclusion



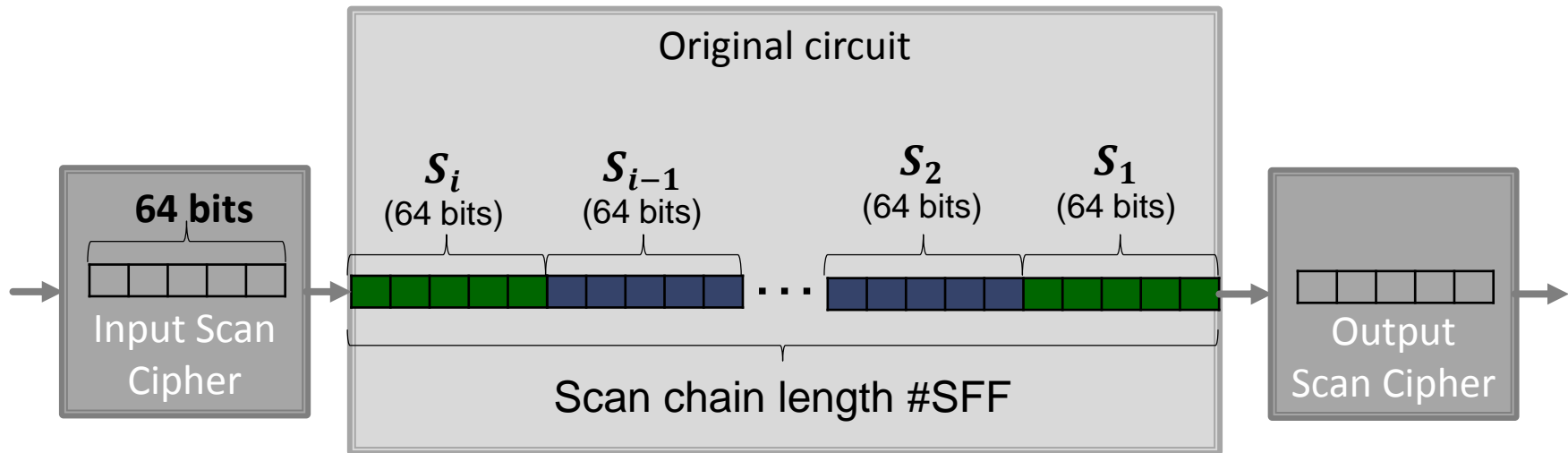
BLOCK CIPHER-BASED SCAN ENCRYPTION

- 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



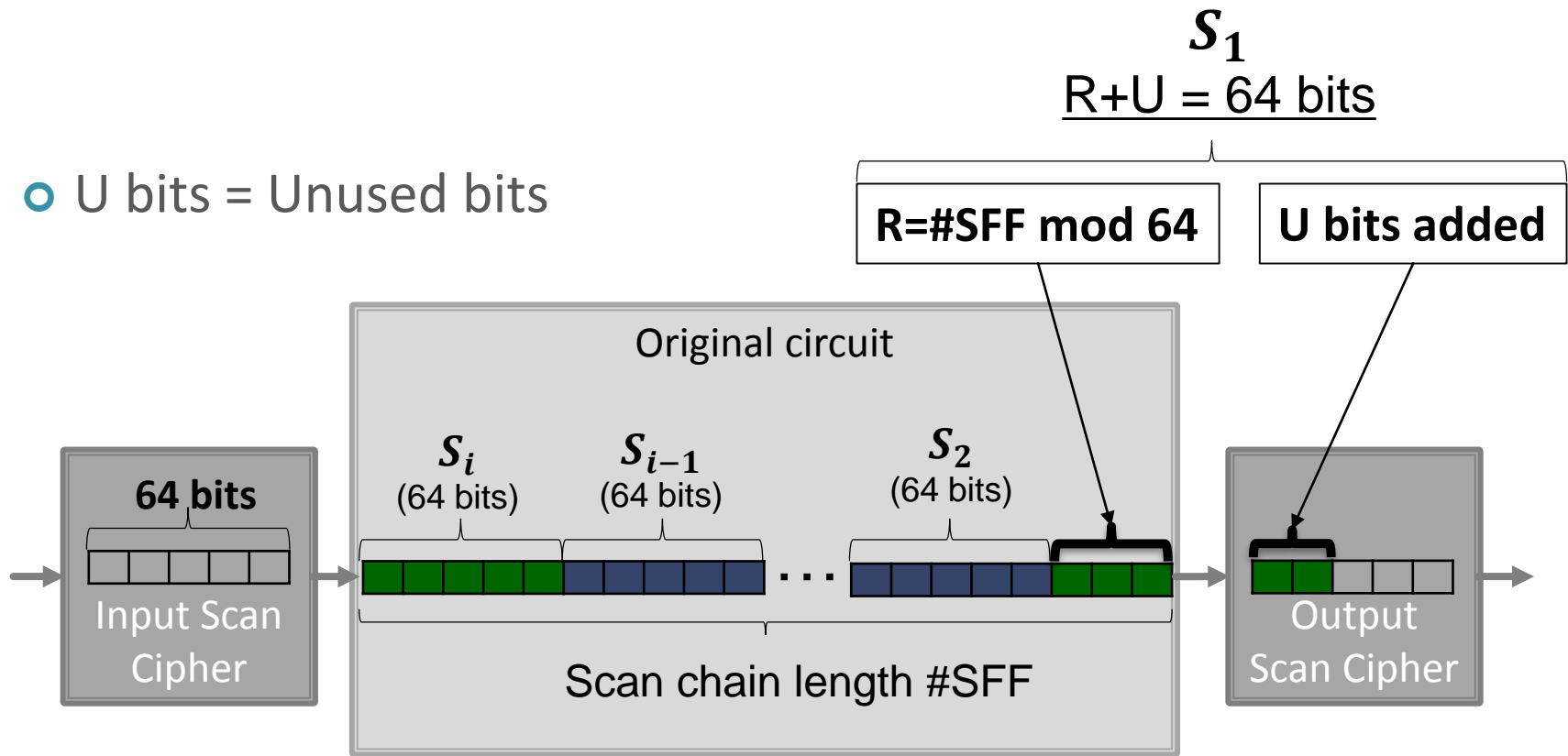
⇒ **#SFF = P x 64**

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



MODE OF OPERATIONS

- U bits = Unused bits



⇒ **#SFF = P x 64 + R**

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



SUMMARY

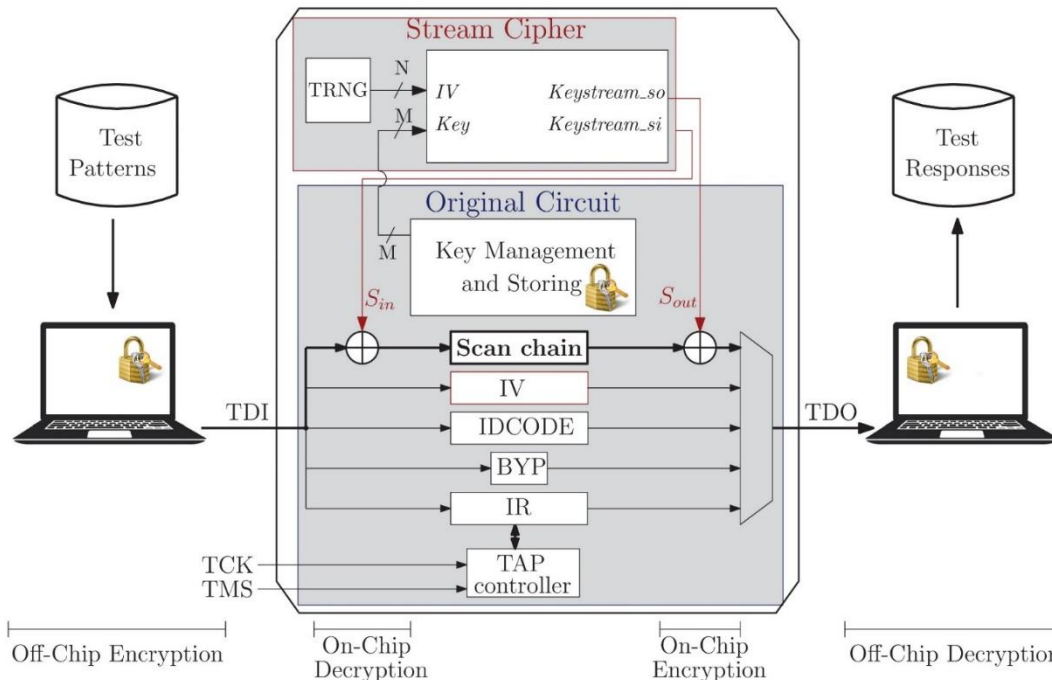
- 1) Scan attacks
- 2) A new countermeasure: Scan chain encryption
- 3) Implementation with block cipher
- 4) Implementation with stream cipher**
- 5) Conclusion



STREAM CIPHER-BASED SCAN ENCRYPTION

Implementation on JTAG:

- 1 TRIVIUM stream cipher (2 016 GE)
- TRNG to generate random IV
- New instruction *GetIV* with a test data register IV

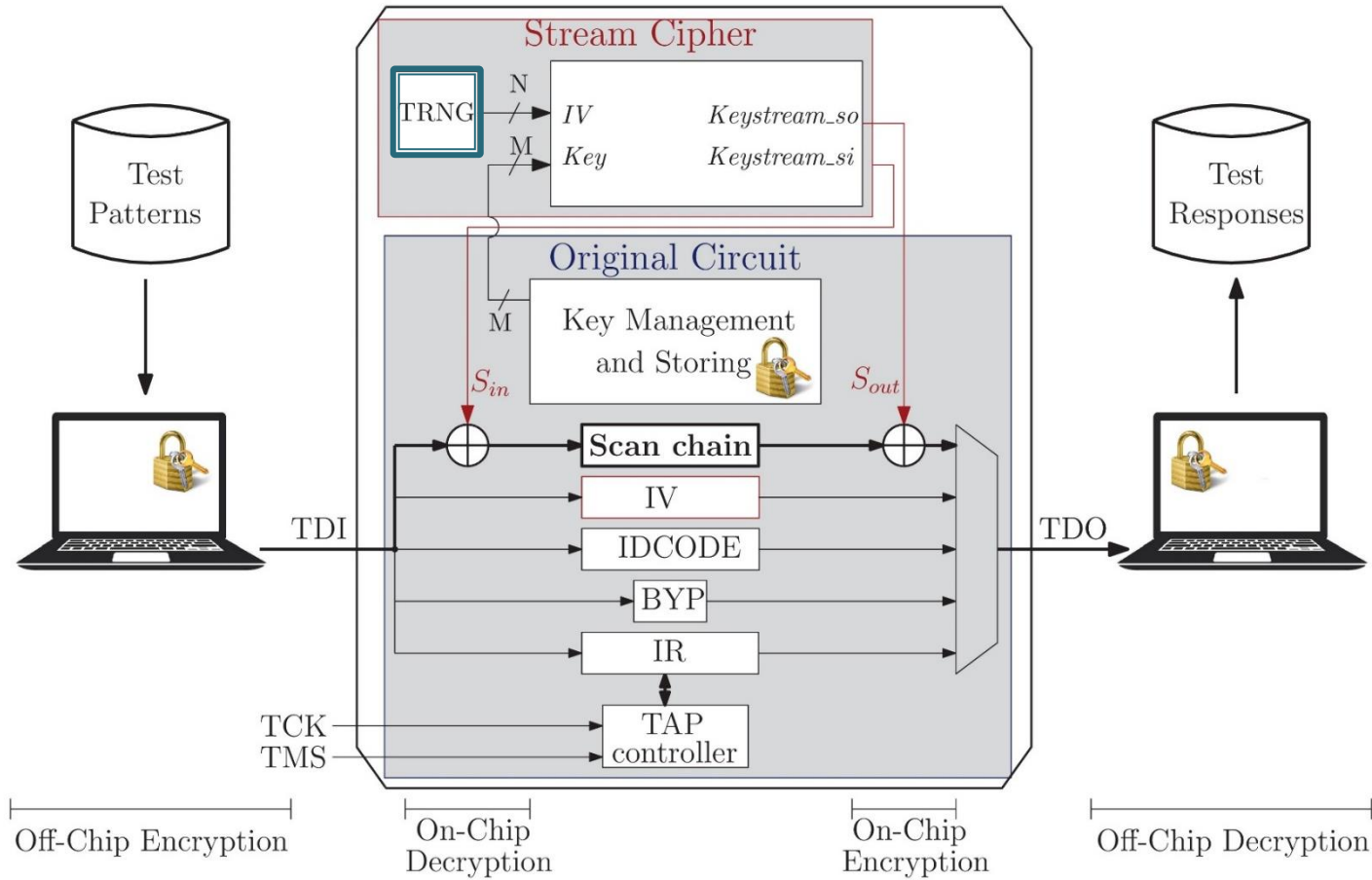


Mode of operations in 2 phases: initialization and encryption

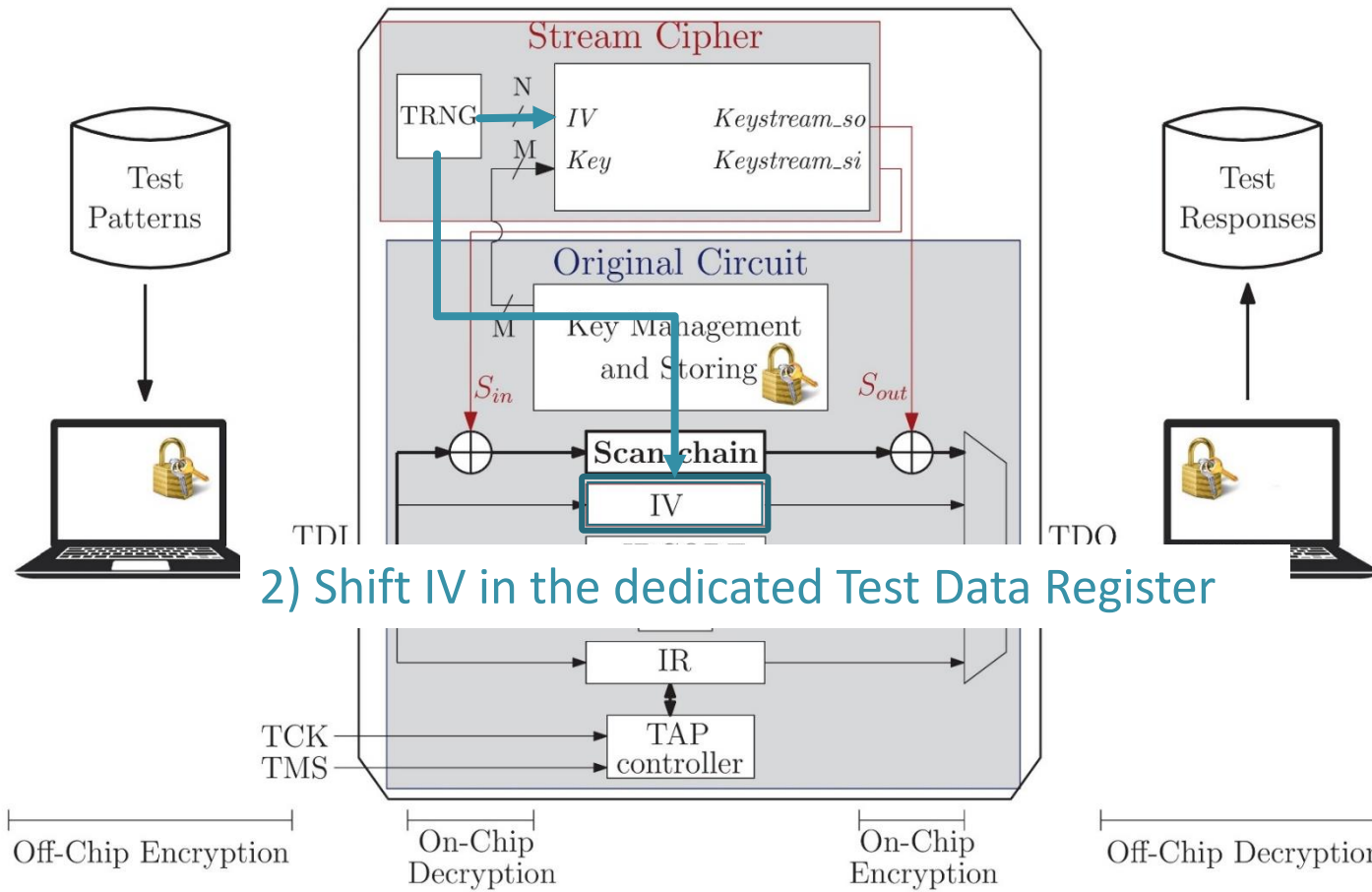


INITIALIZATION PHASE

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



INITIALIZATION PHASE

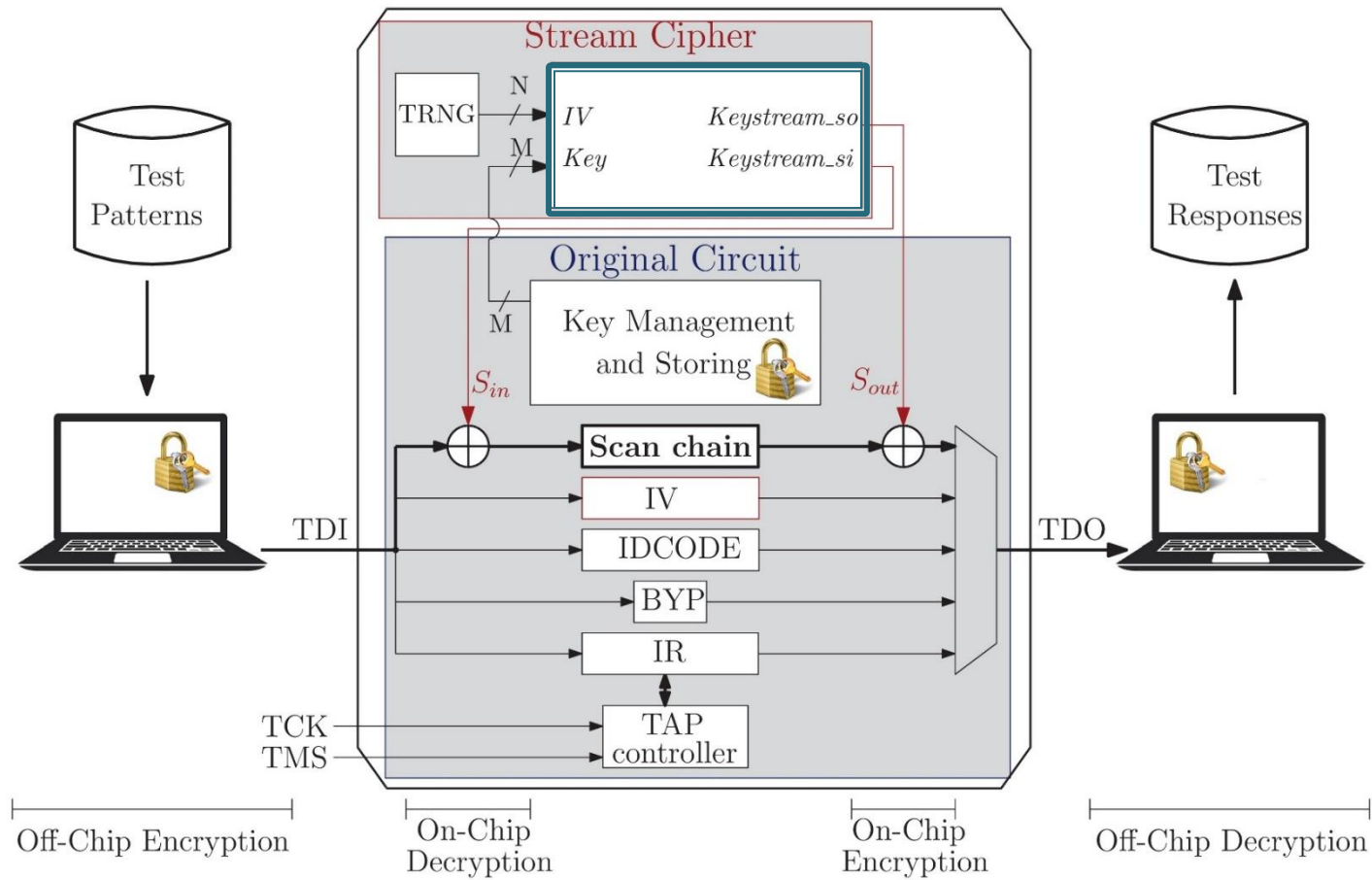


2) Shift IV in the dedicated Test Data Register

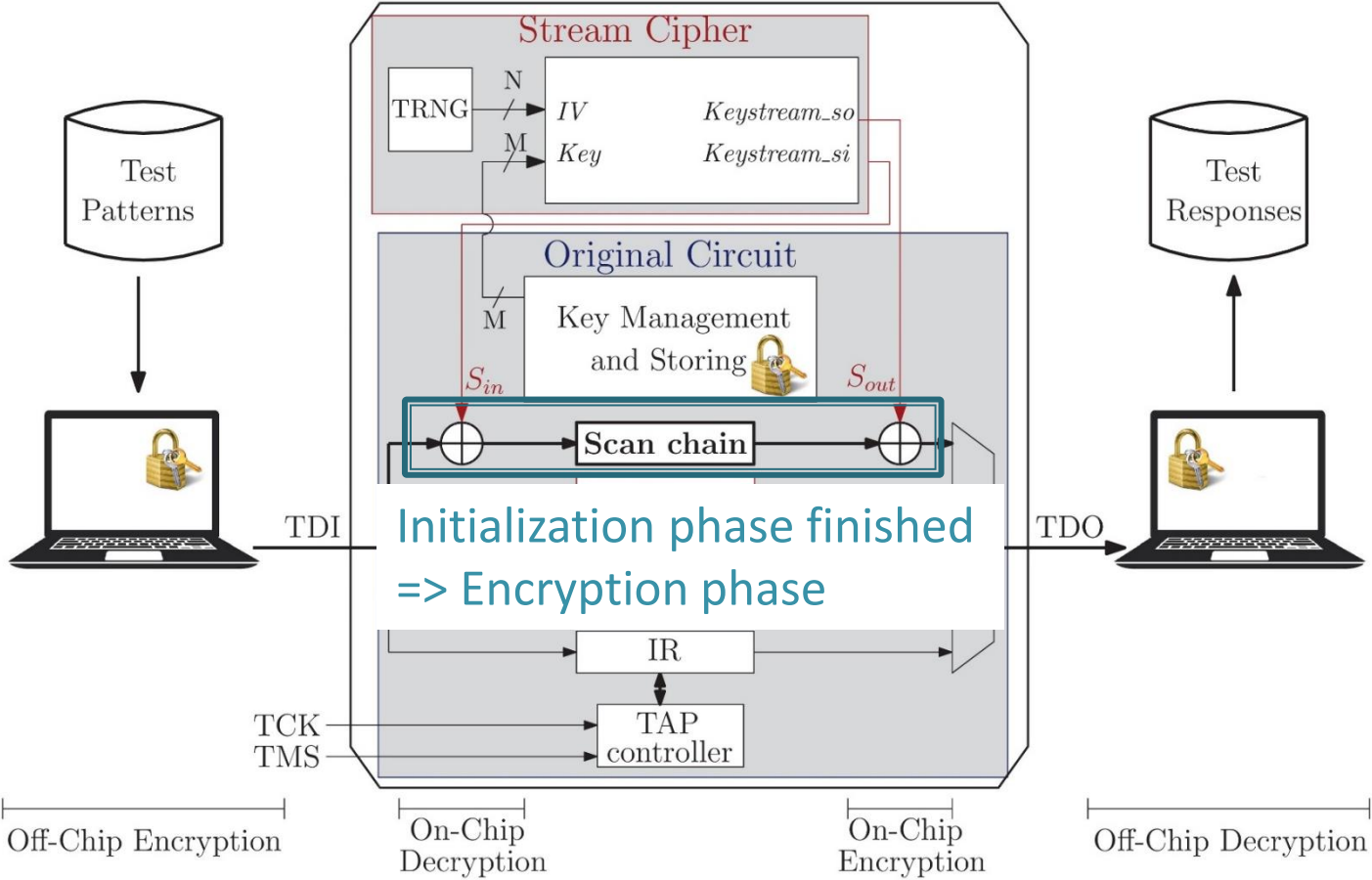


INITIALIZATION PHASE

3) Stream cipher setup

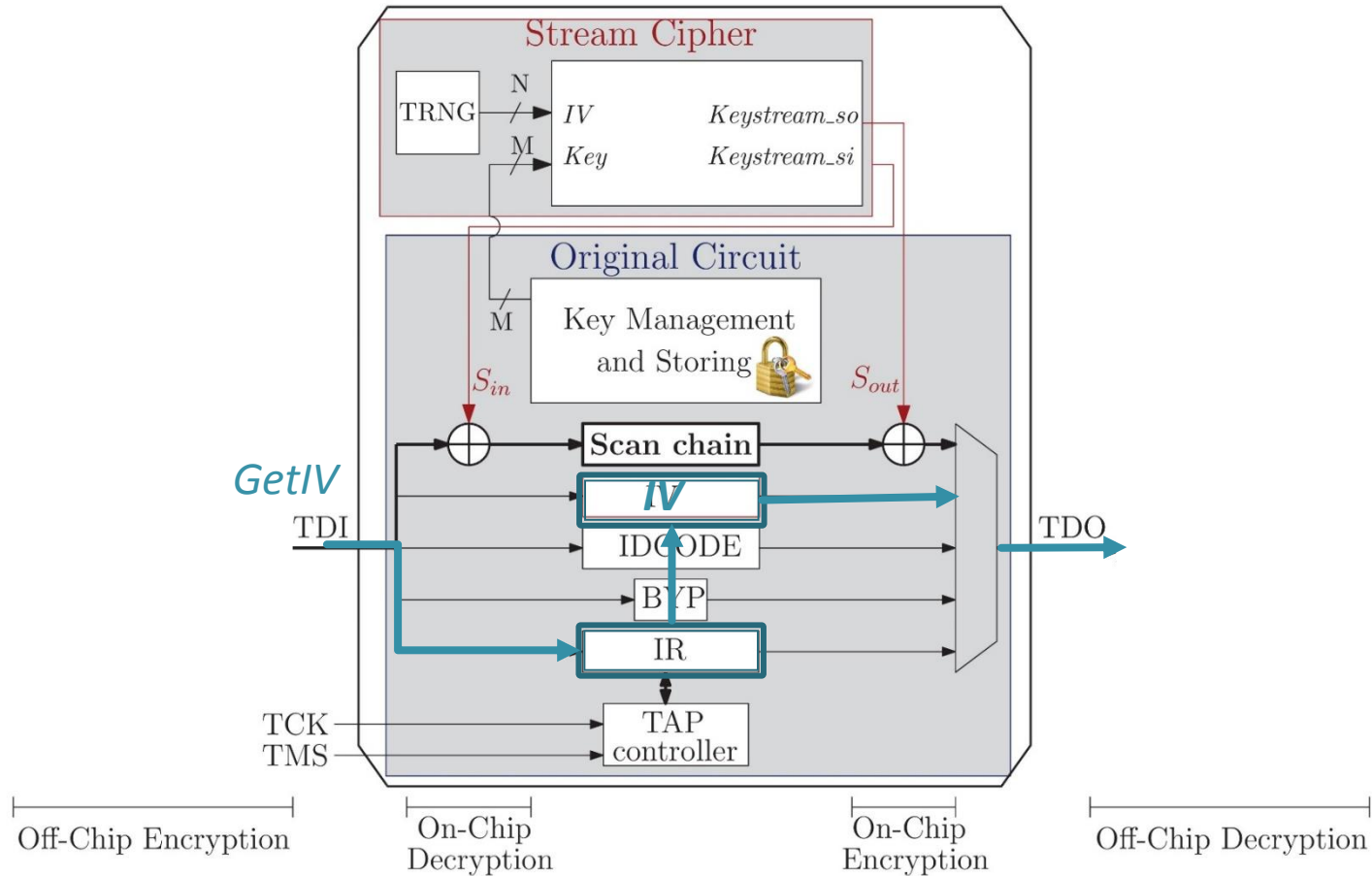


INITIALIZATION PHASE



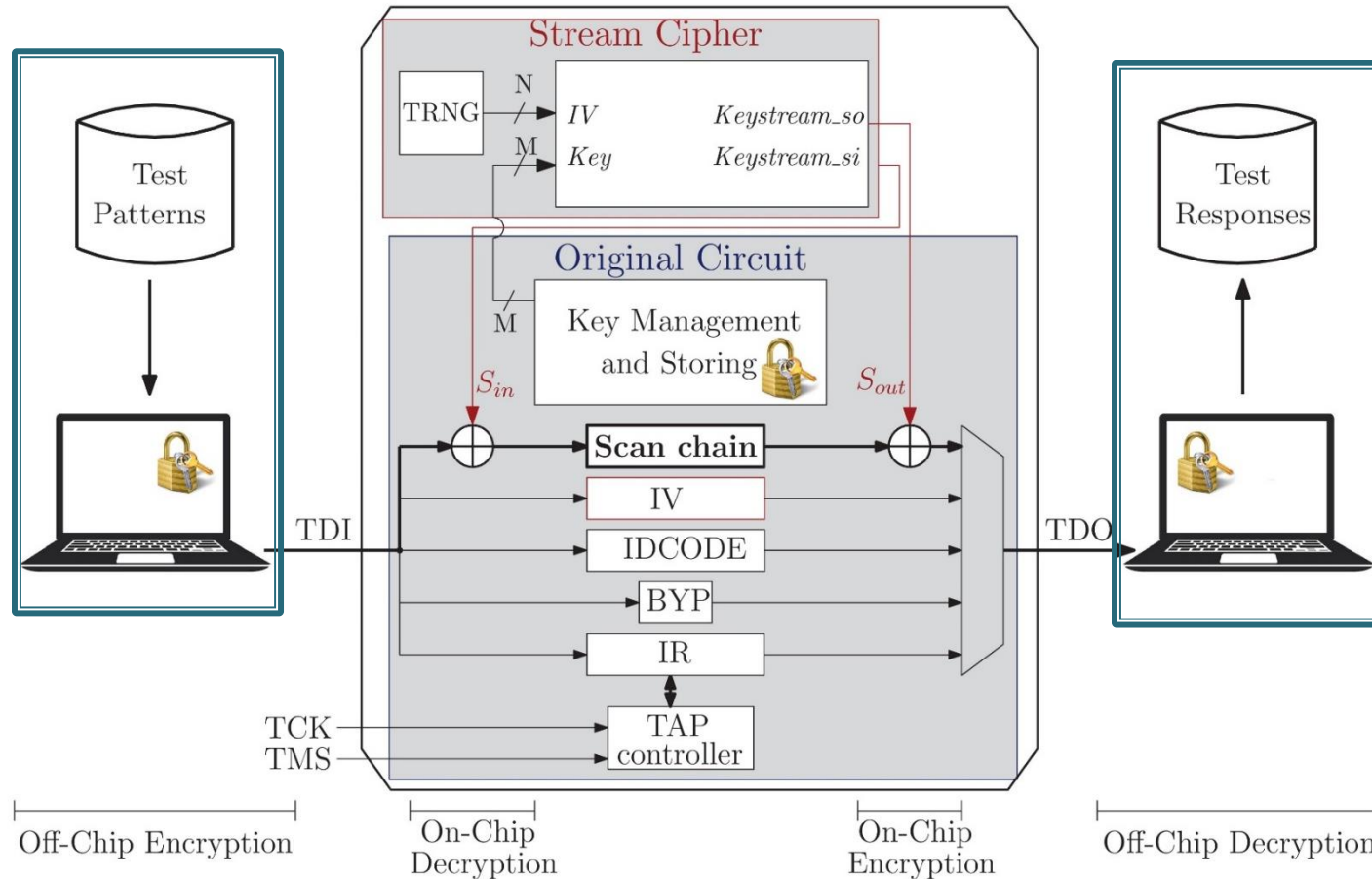
ENCRYPTION PHASE

- 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**



TIME FOR THE INITIALIZATION PROCESS

- 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

Original circuit	Triple-DES	Pipelined AES-128	Pipelined AES-256	RSA 1024	LEON3
Test time* (clock cycles)	687 101	1 944 877	4 559 845	39 405 239	11 612 051
<i>Test time overhead</i>					
Block-based solution (%)	+0.31	+0.81	+0.006	+0.33	+0.004
Stream-based solution (%)**	+0.18	+0.06	+0.03	+0.003	+0.01

*: Test time considered for a fault coverage of 100%, except for LEON3 where it reaches 70%

** : test time overhead without the initialization of the TRNG



SUMMARY

- 1) Scan attacks
- 2) A new countermeasure: Scan chain encryption
- 3) Implementation with block cipher
- 4) Implementation with stream cipher
- 5) **Conclusion**



COMPARISON BETWEEN BOTH SOLUTIONS

	Block cipher-based solution (PRESENT)	Stream cipher-based solution (TRIVIUM)
Security		
- Scan attacks	Protected	Protected (two times pad not possible)
- Malicious core	Protected	Protected
Cost		
- Area	10 658.96 μm^2	5 408.52 μm^2 (+ 31 200 μm^2 for TRNG)
- Test time	Depends on the scan length (multiple or not of the block size)	Clock cycles required for the initialization phase
Integration		
- Diagnosis & debug	Still possible in-field	
- Key management	Re-use key management already implemented	
- Integration in test daisy-chain	Possible issue with the padding of test data	No issue



Thank You



ACKNOWLEDGEMENTS

- FUI#20 TEEVA Project

- Partners

