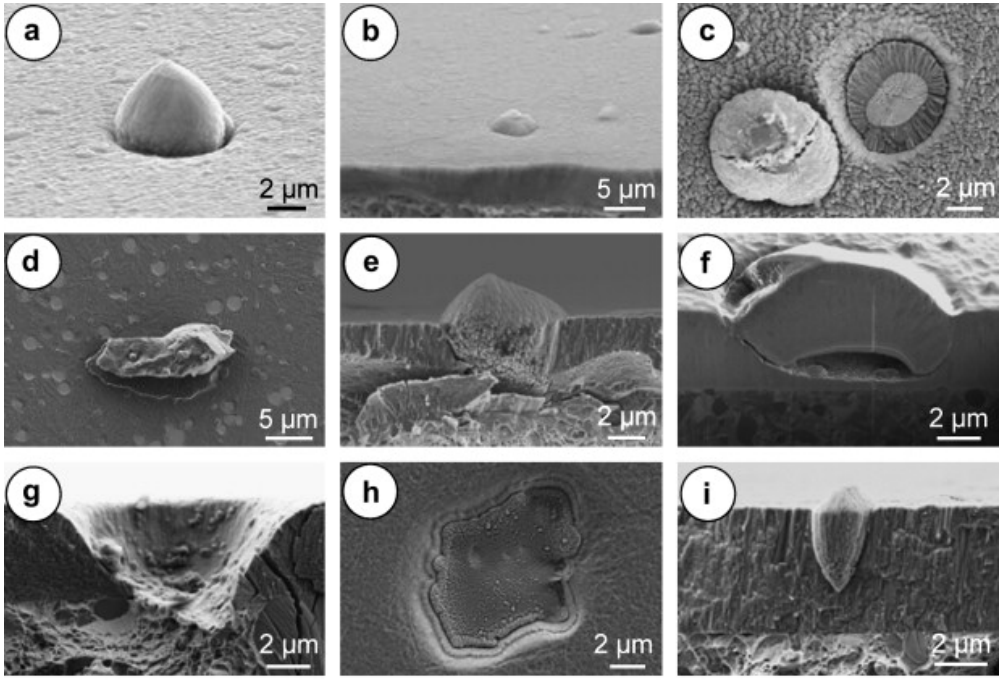
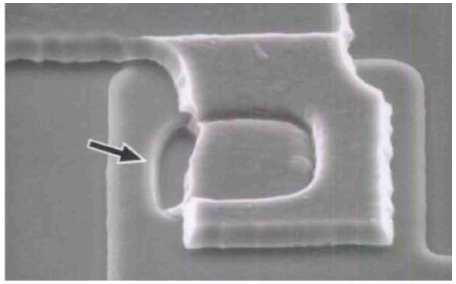
SCAN-CHAIN ATTACK

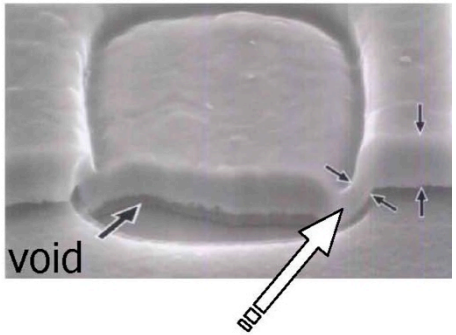
Process de fabrication

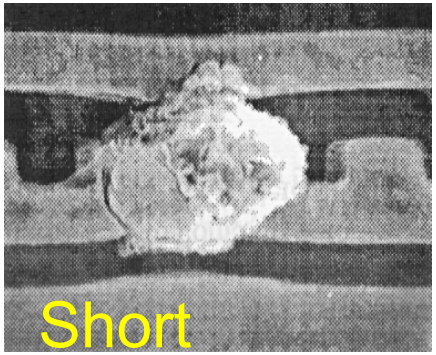
- Le process de fabrication des circuits intégrés pas totalement contrôlé:
  - Poussières, mécanismes physiques, défauts locaux (circuits ouverts, court-circuits,…)
  - Variabilité du process (épaisseurs d'oxyde, tension de seuil, variations des paramètres électriques, …)
  - Défauts d'assemblage (interconnexions entre circuit et carte, chip et packaging, chip et chip)

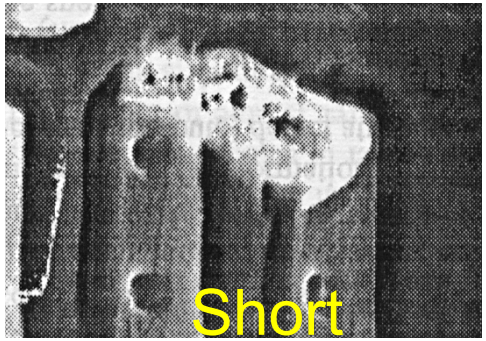
Exemples de défauts

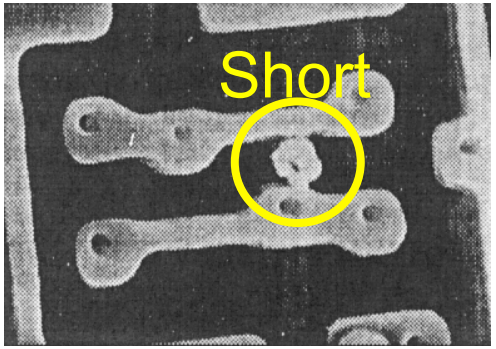




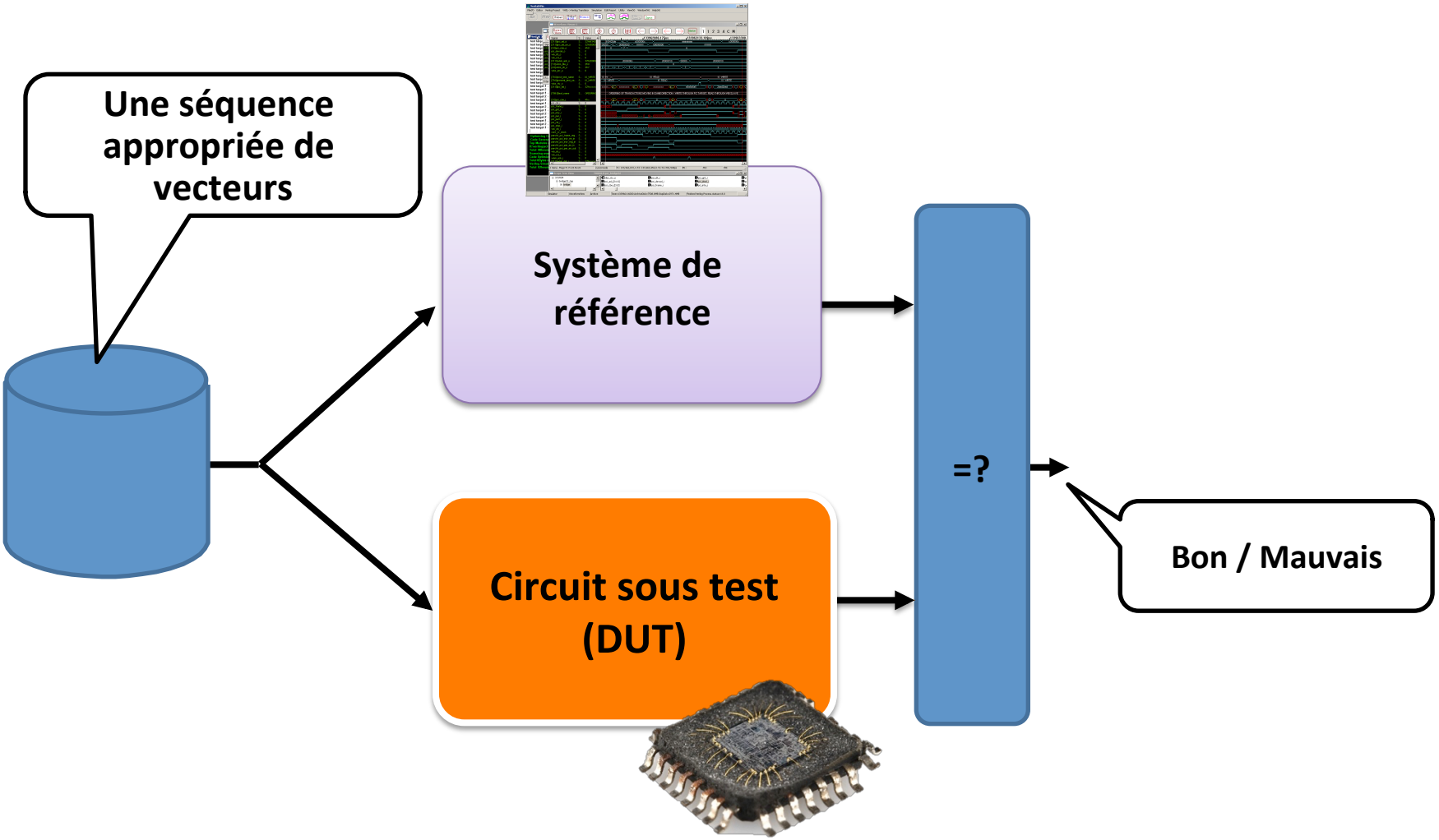








Le test : approche de base



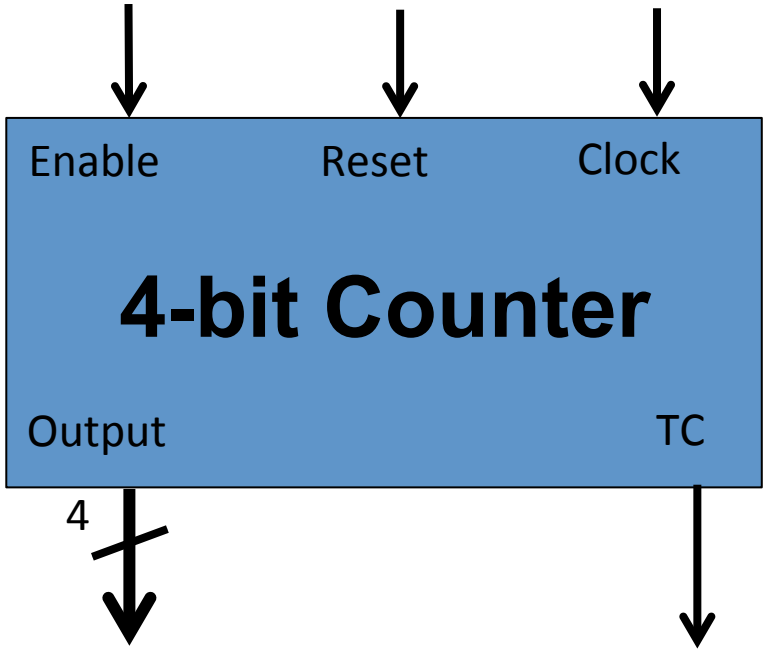
Evolution historique

- Depuis “First design then test”…
- … jusqu'à “Design & Test”
  - Conception en vue du test (CVT ou DFT en anglais), i.e., modifier la logique pour la rendre plus facilement testable.

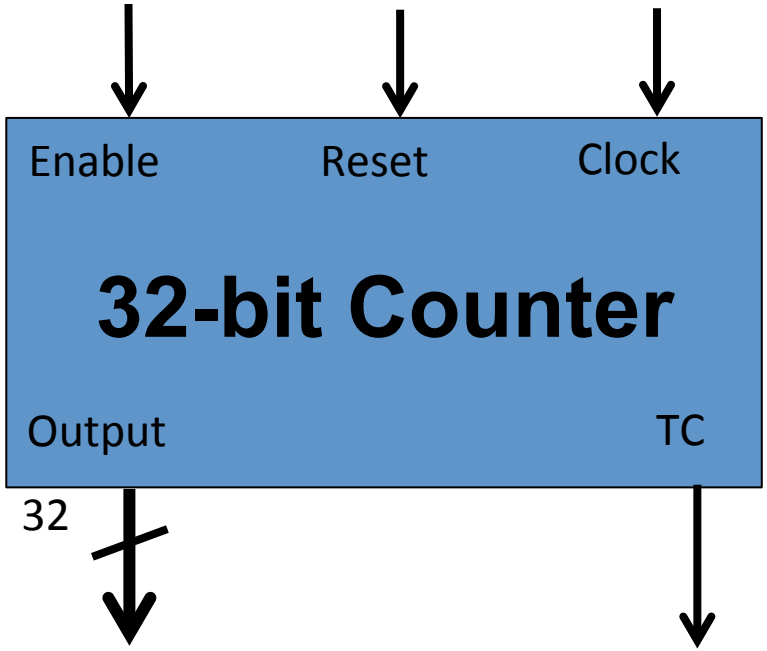
Quelques définitions

- Vecteur de test
  - Ensemble de valeurs, qui à un instant donné, sont appliquées au DUT
- Séquence de test (ou Test Pattern)
  - Séquences de vecteurs de test à appliquer au DUT

Du test fonctionnel …



…sur des circuits plus gros

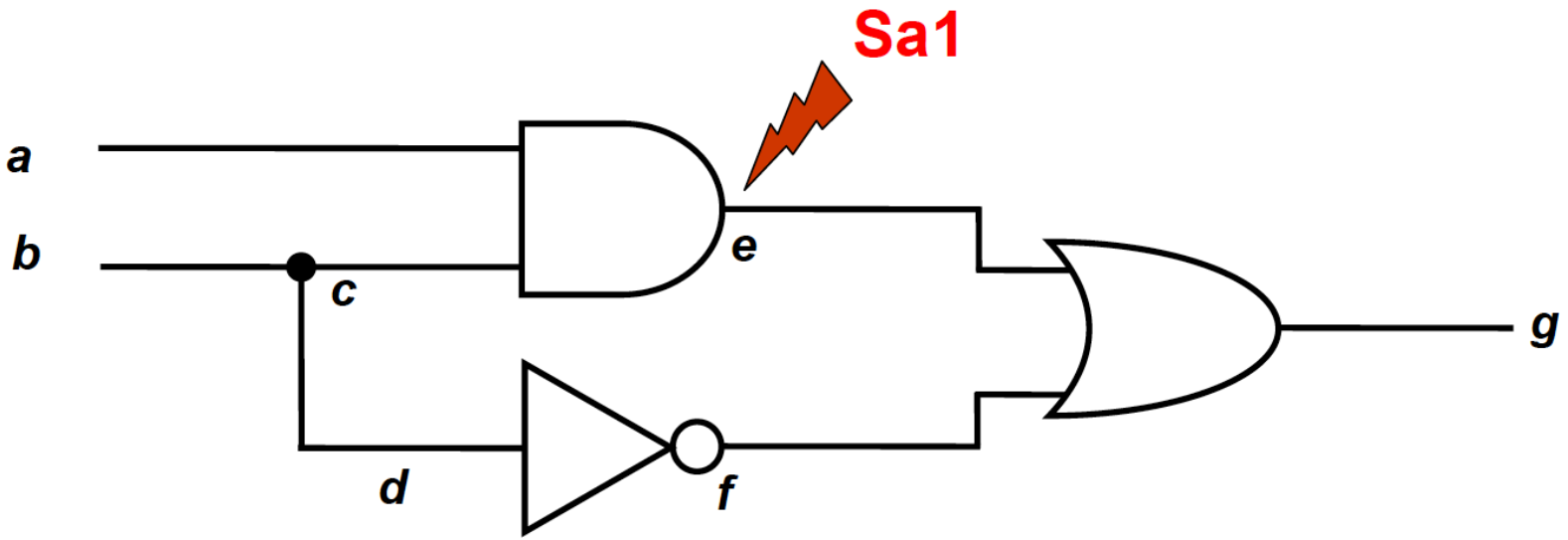


Design for Testability (DfT) conception en vue du test

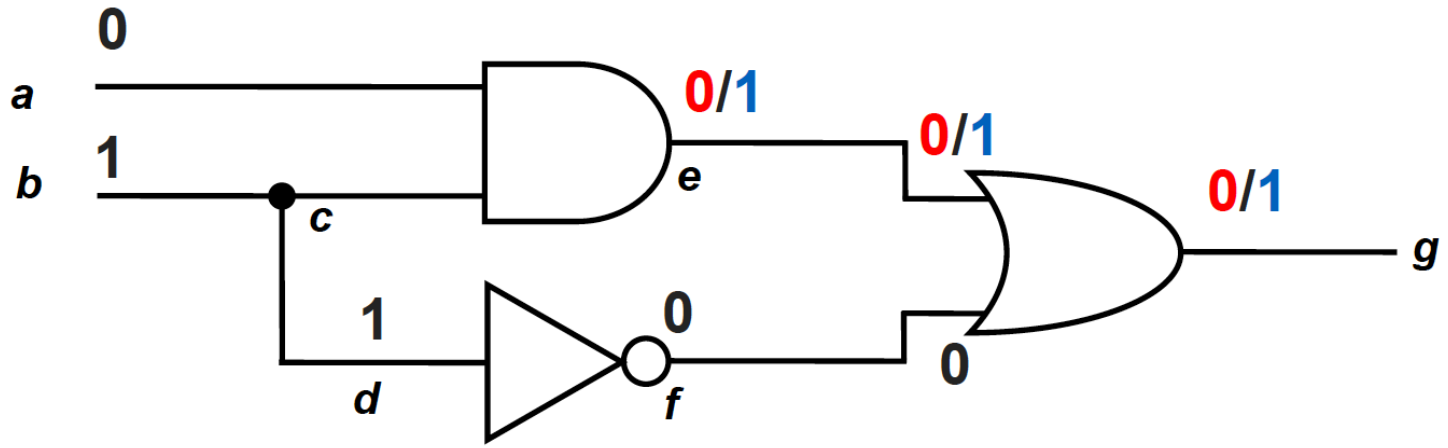
- Ajout de structures pour faciliter le test des circuits (non utilisées en mode fonctionnel)

- La plus commune (99% des circuits) : chaines de scan

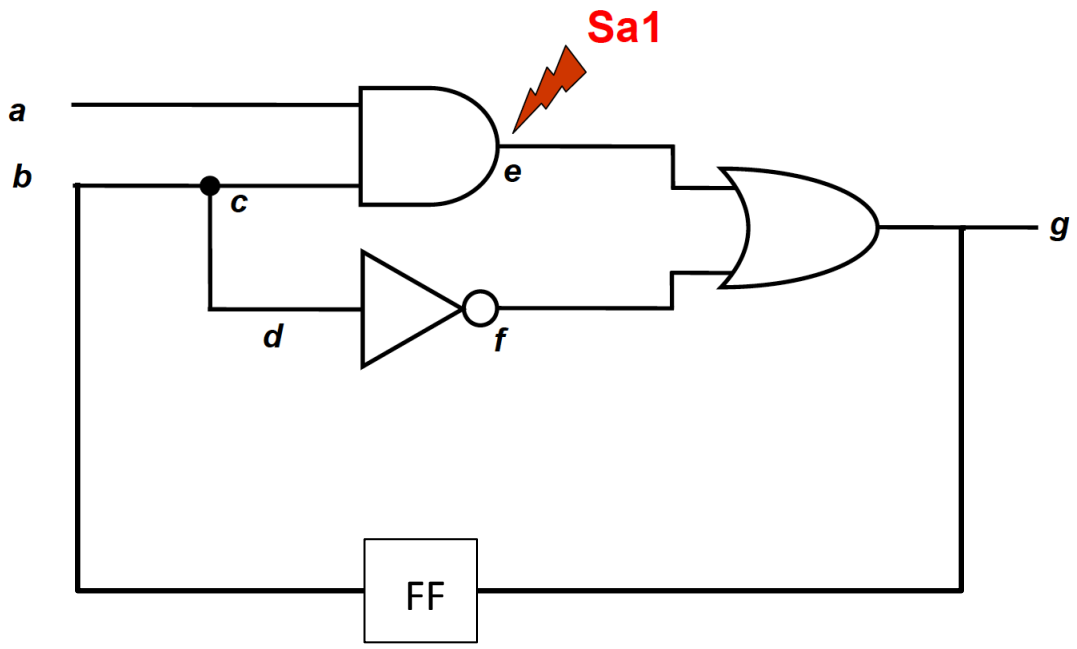
Fautes et génération de vecteurs de test



Génération de vecteurs de test



Circuits séquentiels

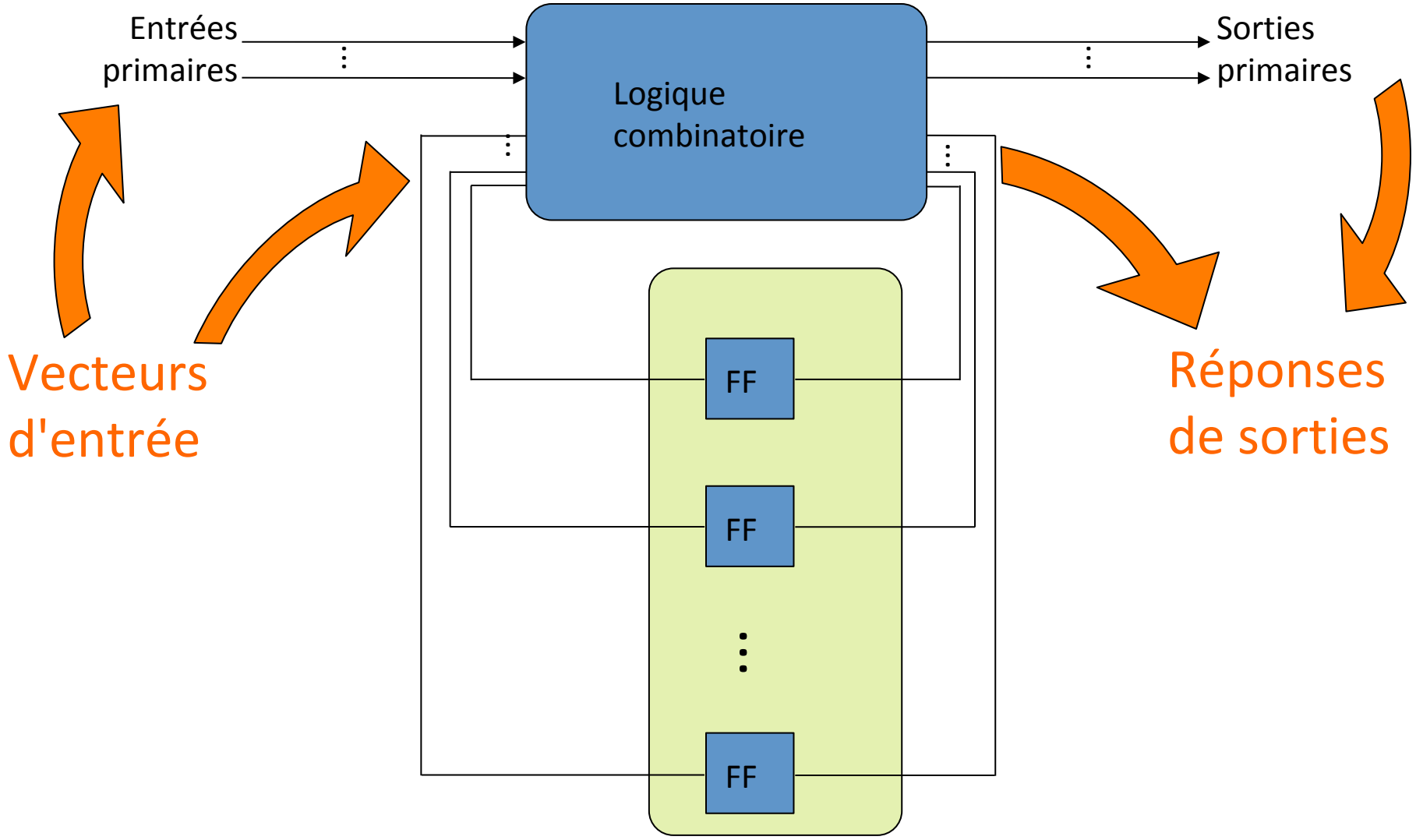


=> Séquence de vecteurs

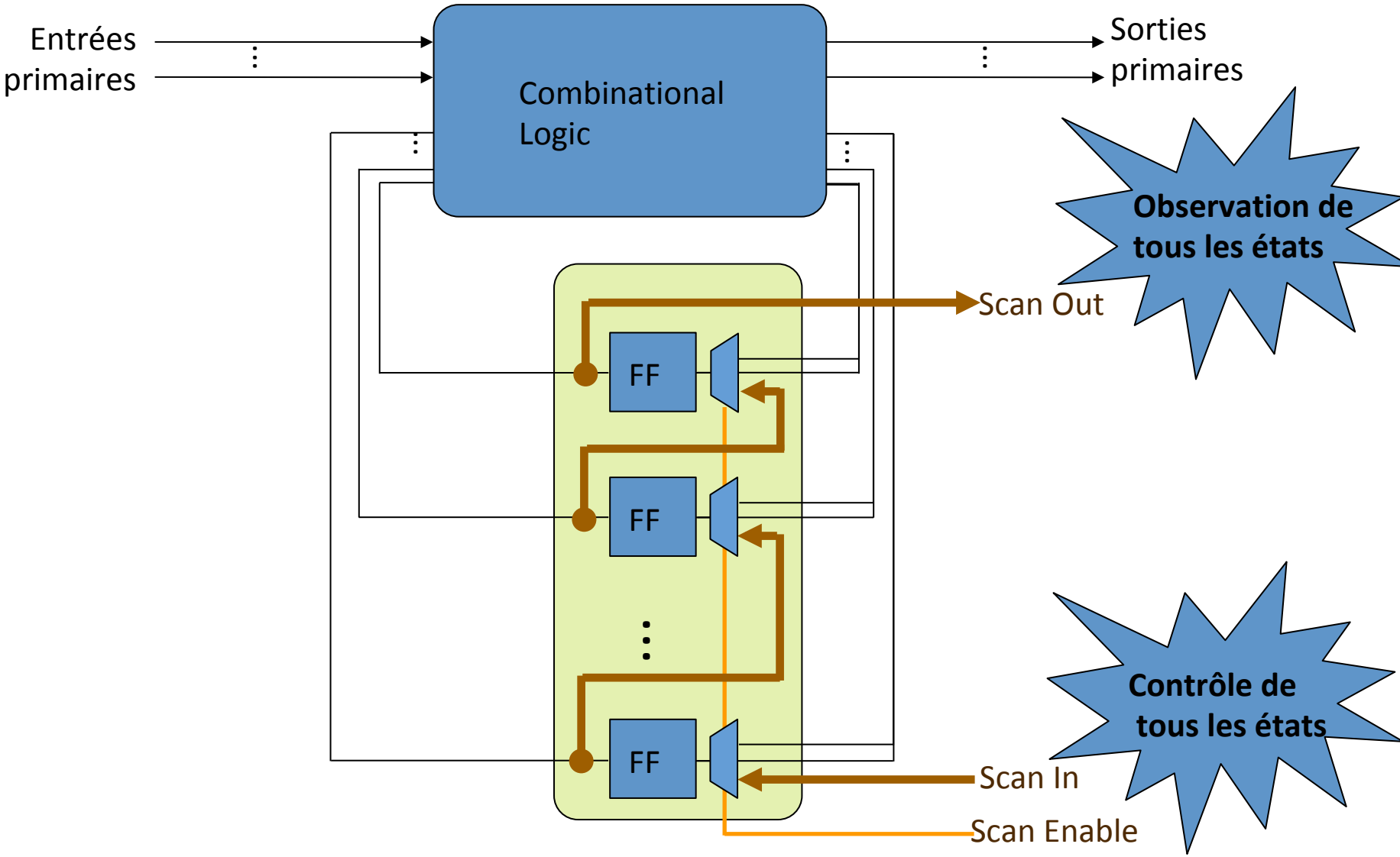
Génération de vecteurs de test

- Facile pour les circuits combinatoires
- Beaucoup (beaucoup) plus difficile pour les circuits séquentiels
  - Justification des états
  - Propagation de l'erreur

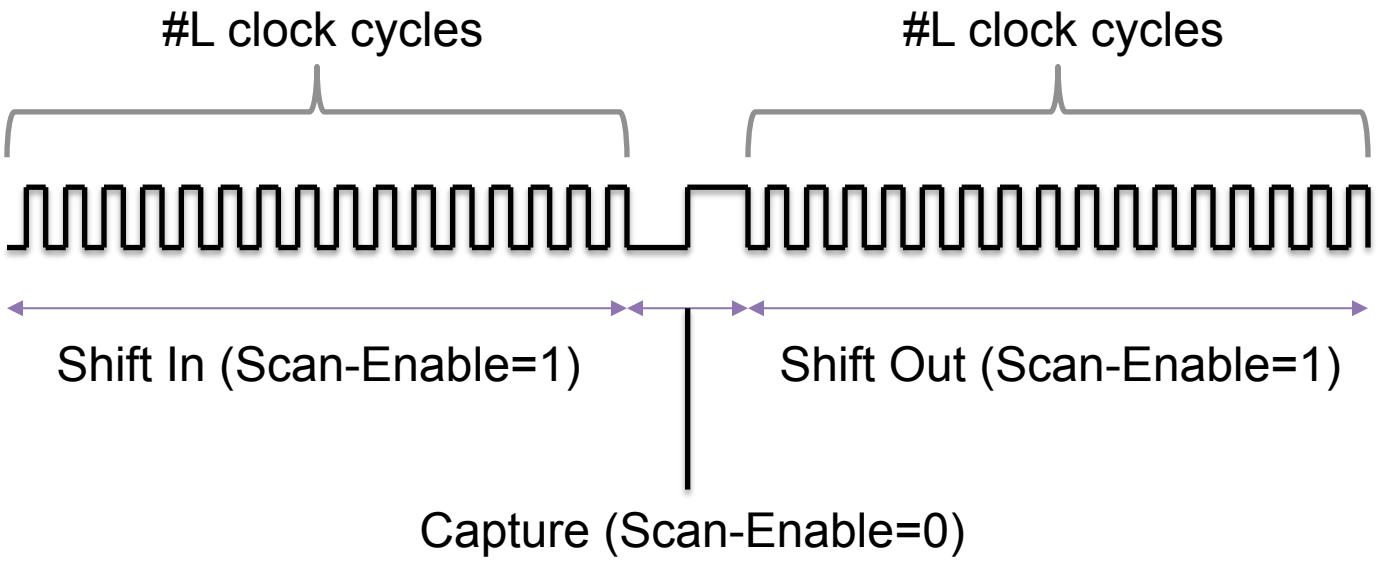
Scan-based Design



Scan-based Design (2)

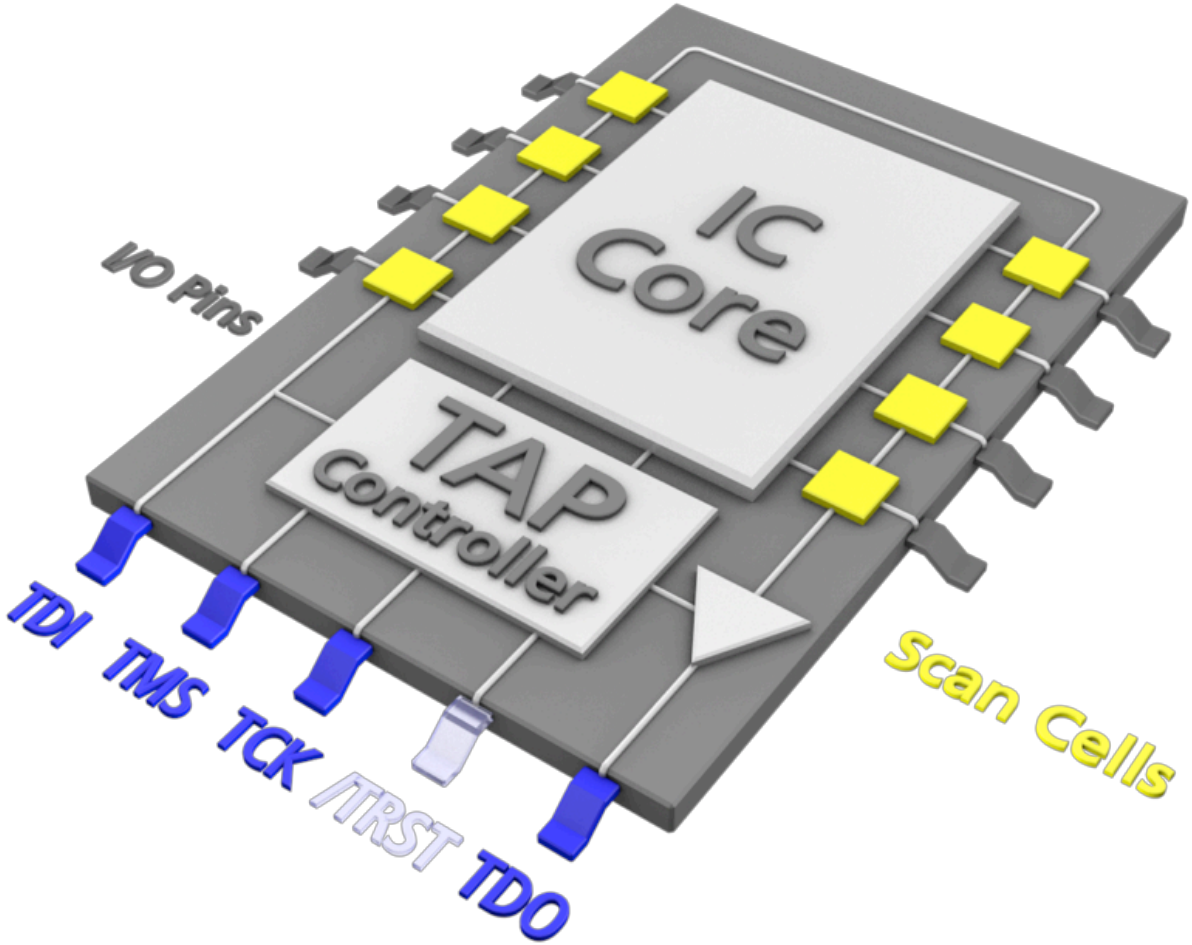


Procedure de test en scan



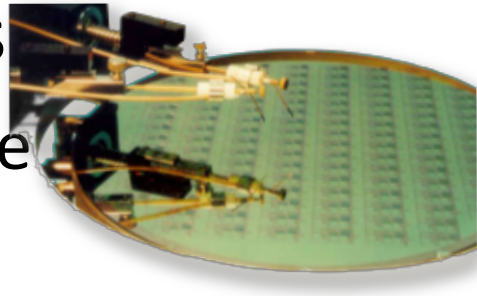
Clock frequency is not realistic. Scan operations @ ≈ 50MHz

JTAG


I/O Pins
IC Core
TAP Controller
Scan Cells
TDI
TMS
TCK
/TRST
TDO

Test vs Securité

- Le test des circuits est nécessaire pour garantir la bonne qualité des circuits
  - Accroissement de la contrôlabilité et de l'observabilité



- Inversement, la sécurité craint la testabilité
  - Infrastructures de test utilisables pour des attaques

Attaque par scan

- Principe
  - Utiliser la chaine de scan pour observer les données internes au circuit à n'importe quel moment

- Méthode (ex: retrouver les données sensibles)

  -1- Trouver les FFs parmi toutes les FFS qui mémorisent le chiffré.

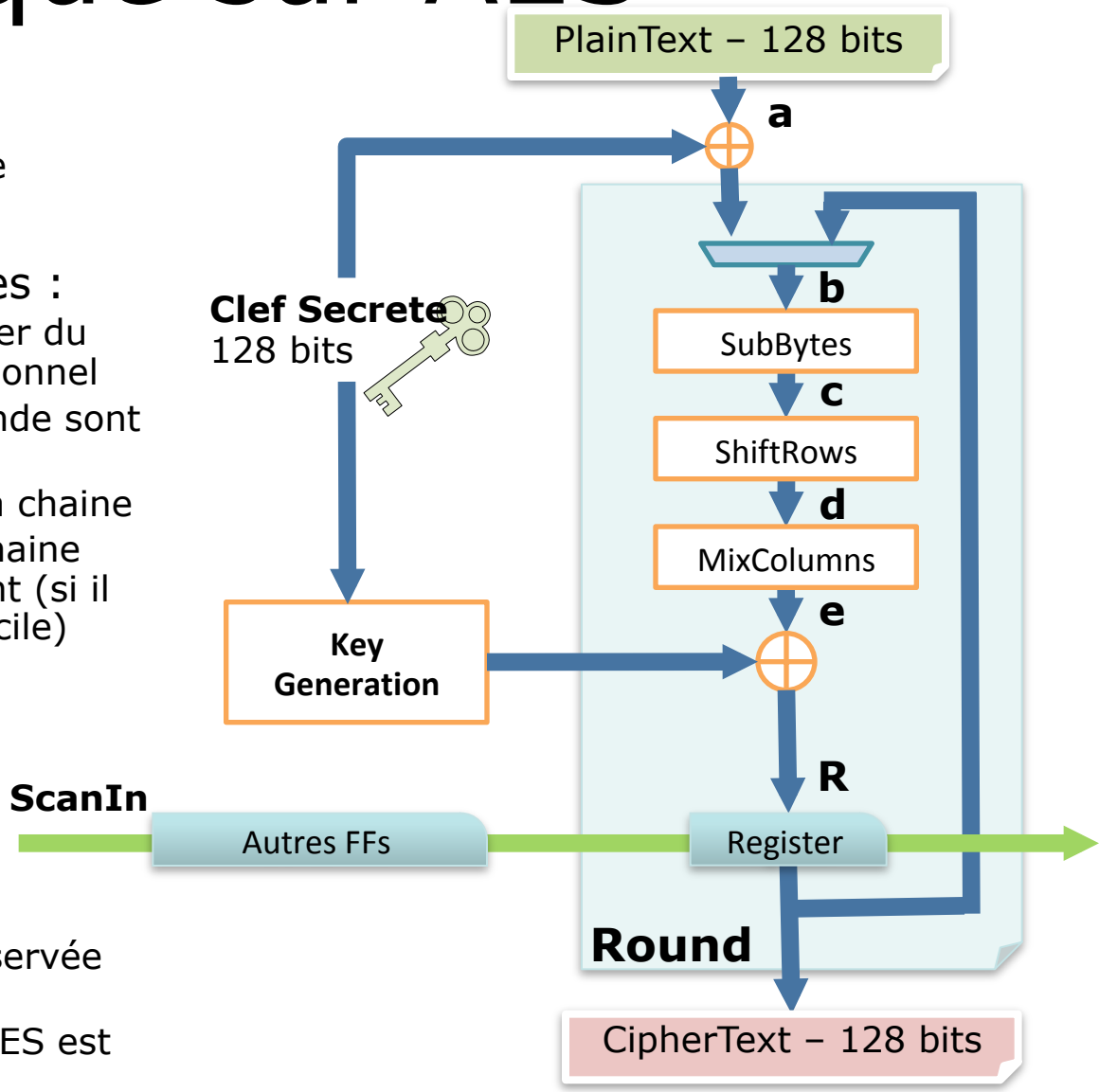
  -2- Connaissant le message clair et le message partiellement chiffré (en utilisant le mode test), en déduire la clef.

Attaque sur AES

- AES
  - Sûr après 10 rondes
  - Pas du tout après 1 ronde

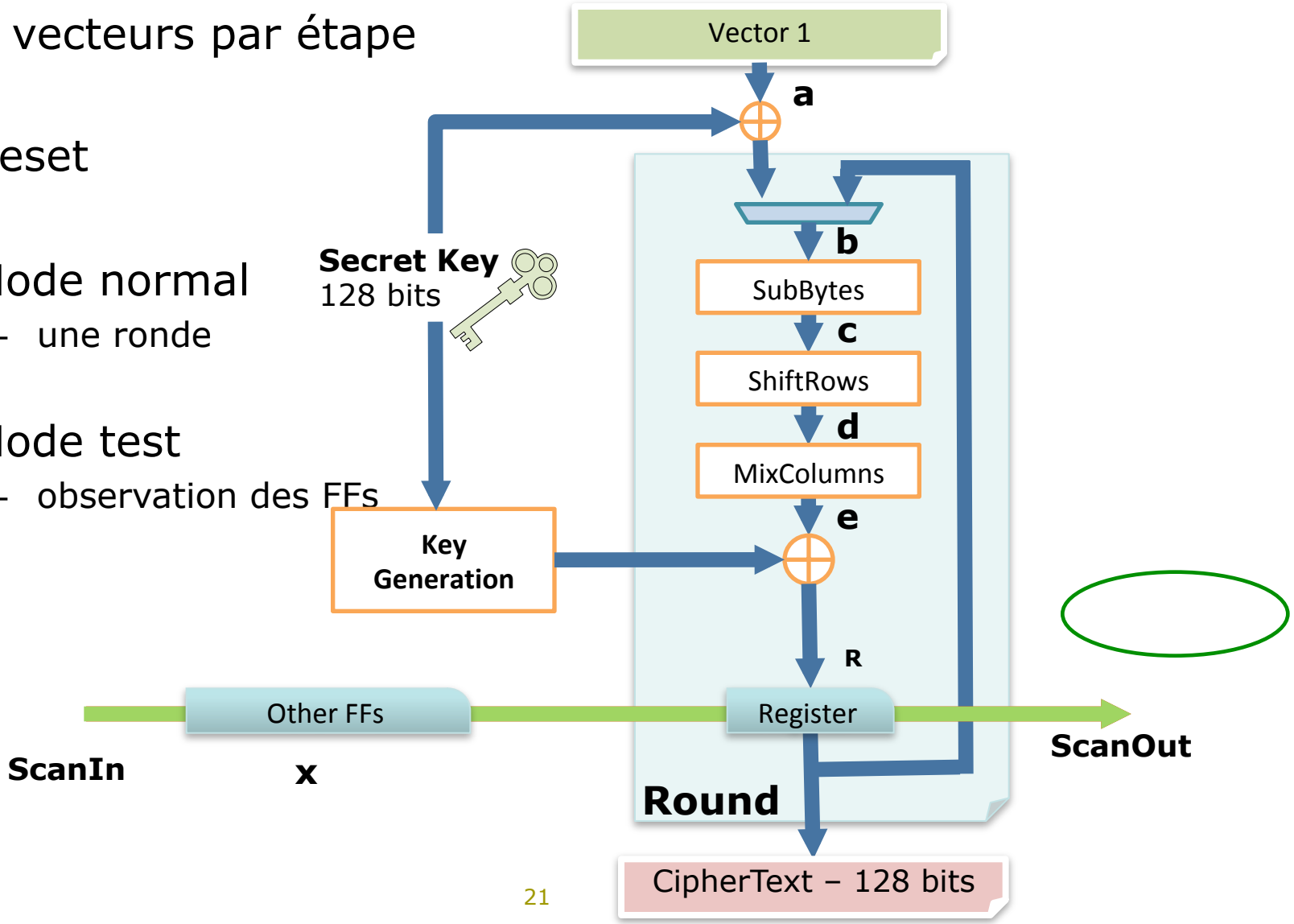
- Contraintes et hypothèses :
  - L'attaquant peut commuter du mode test au mode fonctionnel
  - Les FFs du registre de ronde sont dans la chaine de scan
  - Il y a d'autres FFs dans la chaine
  - L'ordre des FFs dans la chaine est inconnu de l'attaquant (si il le connaît, encore plus facile)

- Attaque:
  - La chaine de scan est observée après la première ronde.
  - Après une seule ronde l'AES est facilement attaquable.



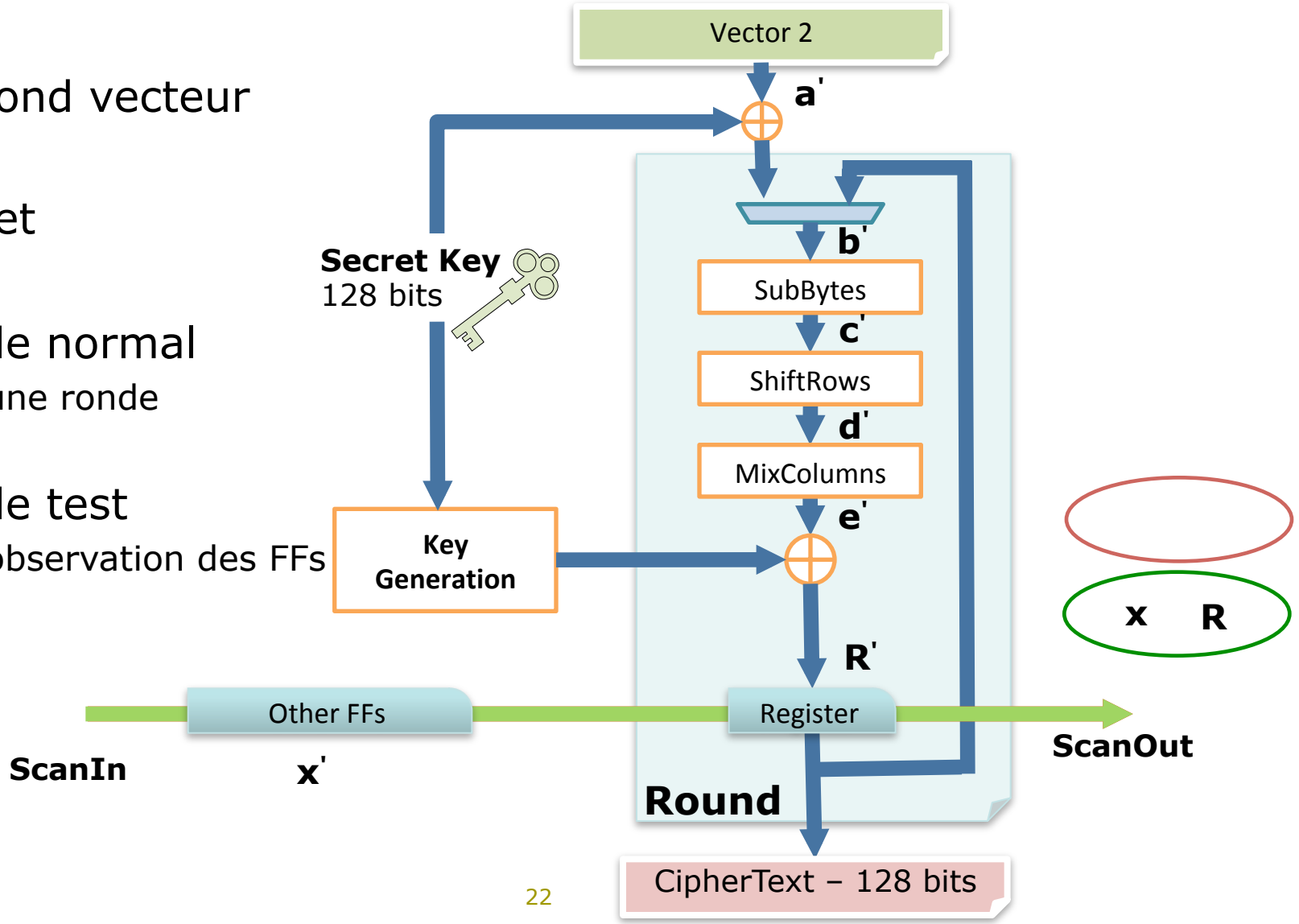
Attaque différentielle

- 2 vecteurs par étape
- Reset
- Mode normal
  - une ronde
- Mode test
  - observation des FFs



Attaque différentielle

- Second vecteur
- Reset
- Mode normal
  - une ronde
- Mode test
  - observation des FFs



Elimination des FFs non sensibles

- Hypothèse:
  Les autres FFs sont indépendantes du message clair
  => après une ronde, elles ont même valeur : x=x'

- Distance de Hamming

On observe 128 bits
Mais ils dépendent de 128 bits de clef
L'ordre des bits n'est pas connu

Problématique

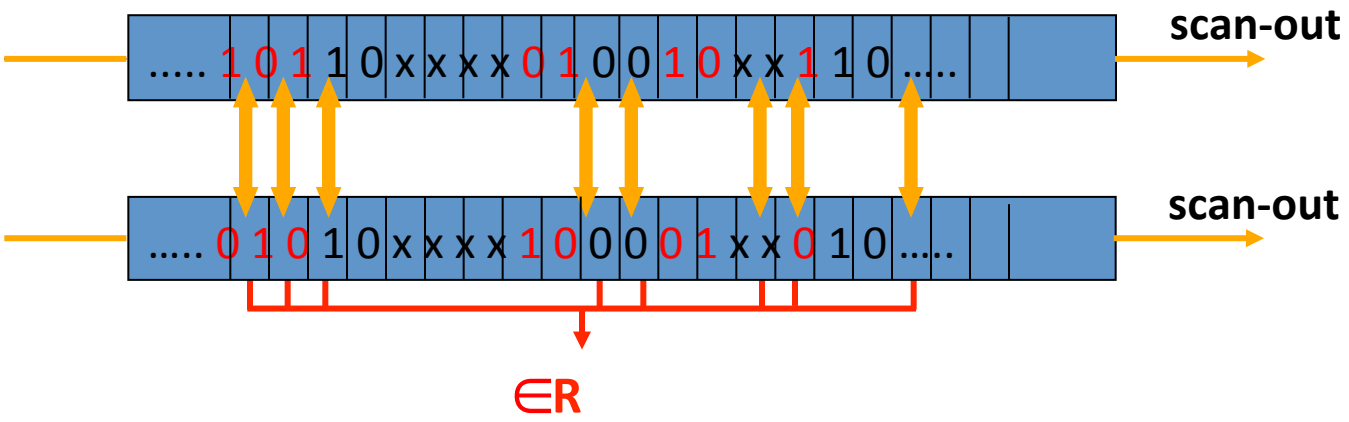
- Comment reconnaître les données intéressantes dans le flots de bits obtenus sur le Scan-out ?

- Comment réduire la complexité de l'analyse des données ?

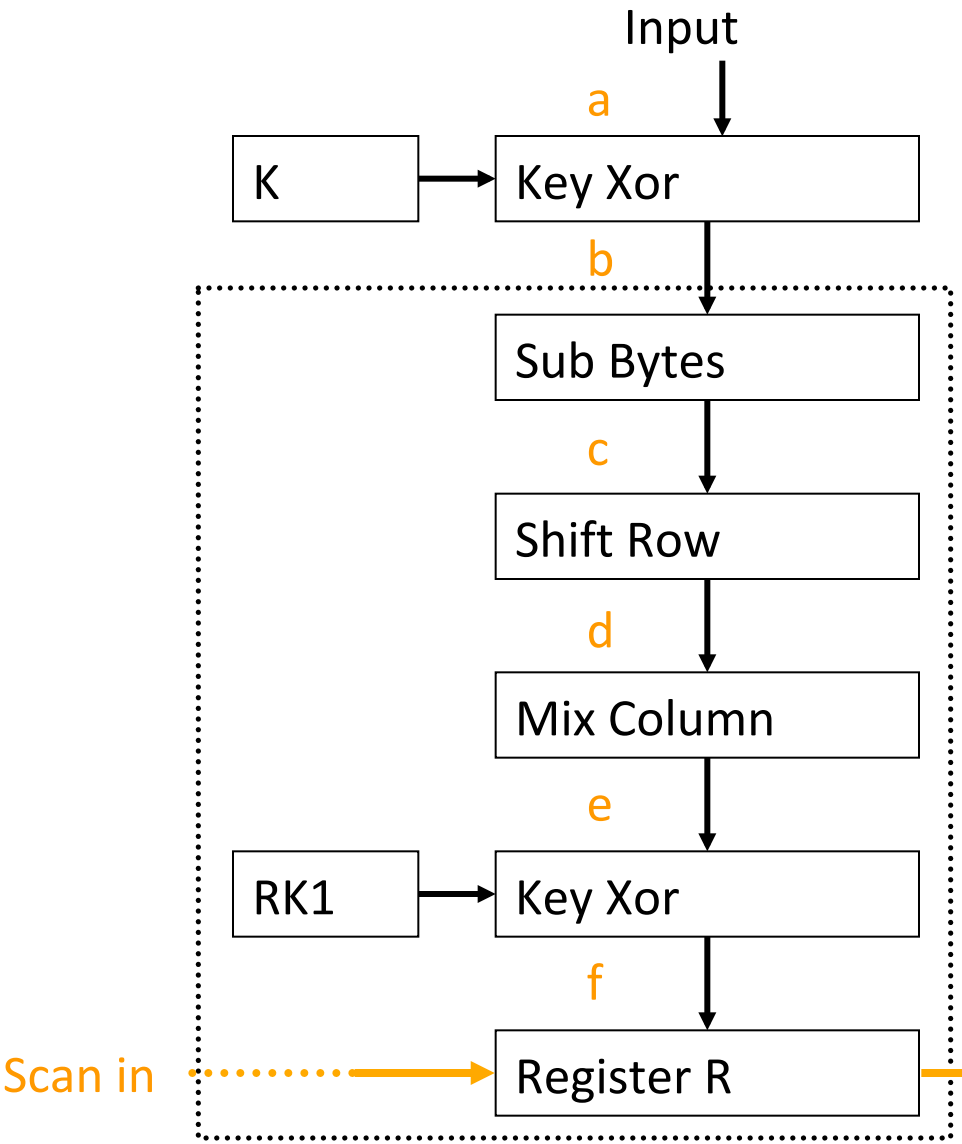
Attaque scan

-1- Identification des FFs du registre de ronde (en fait non nécessaire pour attaque sur AES)

Appliquer plusieurs vecteurs a



Attaque scan



- AES implementation
  - Les 128 bits sont partitionnés en 16 octets

a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}
a_{30}	a_{31}	a_{32}	a_{33}

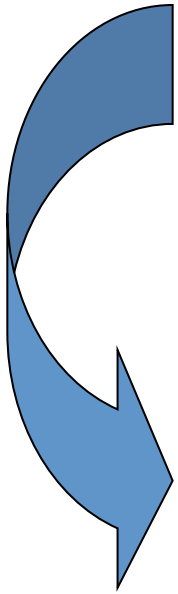
  - Les 128-bits du registre de ronde R dans la chaîne de scan

Attaque scan

-2- Réduction de la complexité

Changement du clair a en a’ => changement des données dans R ?

- Changement d'un seul octet de a
  - a11 changé en a’11



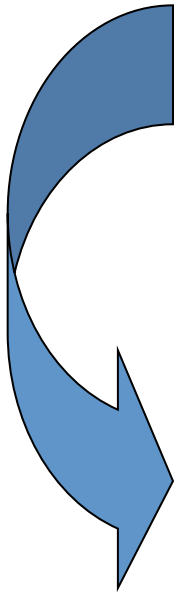
a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}
a_{30}	a_{31}	a_{32}	a_{33}

a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a'_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}
a_{30}	a_{31}	a_{32}	a_{33}

Attaque scan

-2- Réduction de la complexité

- Note: changement du clair a en a’ => changement des données dans R
- Changement d'un seul octet de a
- Opération : Xor Clef
  - a11 changé en a’11
  - b11 changé en b’11



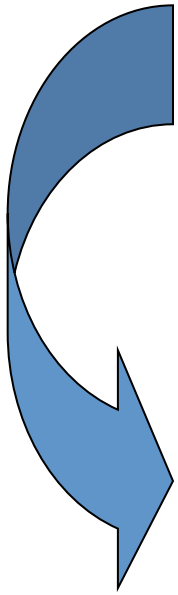
b_{00}	b_{01}	b_{02}	b_{03}
b_{10}	b_{11}	b_{12}	b_{13}
b_{20}	b_{21}	b_{22}	b_{23}
b_{30}	b_{31}	b_{32}	b_{33}

b_{00}	b_{01}	b_{02}	b_{03}
b_{10}	b'_{11}	b_{12}	b_{13}
b_{20}	b_{21}	b_{22}	b_{23}
b_{30}	b_{31}	b_{32}	b_{33}

Attaque scan

-2- Réduction de la complexité

- Note: changement du clair a en a’ => changement des données dans R
- Changement d'un seul octet de a
- Opération : SubByte
  - a11 changé en a’11
  - b11 changé en b’11
  - c11 changé en c’11



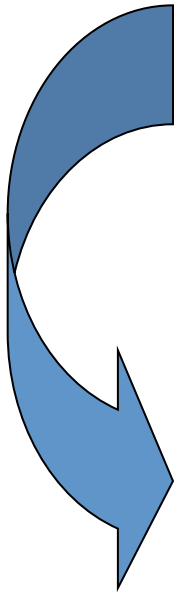
c_{00}	c_{01}	c_{02}	c_{03}
c_{10}	c_{11}	c_{12}	c_{13}
c_{20}	c_{21}	c_{22}	c_{23}
c_{30}	c_{31}	c_{32}	c_{33}

c_{00}	c_{01}	c_{02}	c_{03}
c_{10}	c'_{11}	c_{12}	c_{13}
c_{20}	c_{21}	c_{22}	c_{23}
c_{30}	c_{31}	c_{32}	c_{33}

Attaque scan

-2- Réduction de la complexité

- Note: changement du clair a en a' => changement des données dans R
- Changement d'un seul octet de a
- Opération : ShiftRow
  - a11 changé en a'11
  - b11 changé en b'11
  - c11 changé en c'11
  - d10 changé en d'10



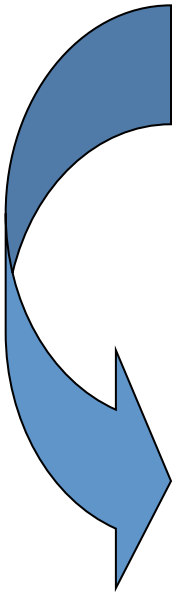
d_{00}	d_{01}	d_{02}	d_{03}
d_{10}	d_{11}	d_{12}	d_{13}
d_{20}	d_{21}	d_{22}	d_{23}
d_{30}	d_{31}	d_{32}	d_{33}

d_{00}	d_{01}	d_{02}	d_{03}
d'_{10}	d_{11}	d_{12}	d_{13}
d_{20}	d_{21}	d_{22}	d_{23}
d_{30}	d_{31}	d_{32}	d_{33}

Attaque scan

-2- Réduction de la complexité

- Note: changement du clair a en a' => changement des données dans R
- Changement d'un seul octet de a
- Opération : MixColumn
  - a11 changé en a'11
  - b11 changé en b'11
  - c11 changé en c'11
  - d10 changé en d'10
  - ei0 changé en e'i0, I = 1..4, 4 octets



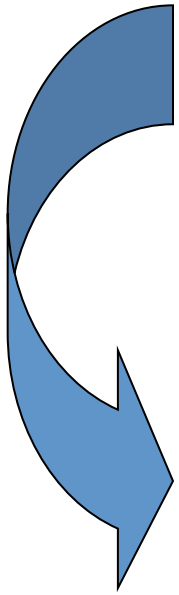
e_{00}	e_{01}	e_{02}	e_{03}
e_{10}	e_{11}	e_{12}	e_{13}
e_{20}	e_{21}	e_{22}	e_{23}
e_{30}	e_{31}	e_{32}	e_{33}

e'_{00}	e_{01}	e_{02}	e_{03}
e'_{10}	e_{11}	e_{12}	e_{13}
e'_{20}	e_{21}	e_{22}	e_{23}
e'_{30}	e_{31}	e_{32}	e_{33}

Attaque scan

-2- Réduction de la complexité

- Note: changement du clair a en a' => changement des données dans R
- Changement d'un seul octet de a
- Opération : AddRoudKey
  - a11 changé en a'11 b11 changé en b'11
  - c11 changé en c'11
  - d10 changé en d'10
  - ei0 changé en e'i0, I = 1..4
  - fi0 changé en f'i0 (4 octets dans R)



f_{00}	f_{01}	f_{02}	f_{03}
f_{10}	f_{11}	f_{12}	f_{13}
f_{20}	f_{21}	f_{22}	f_{23}
f_{30}	f_{31}	f_{32}	f_{33}

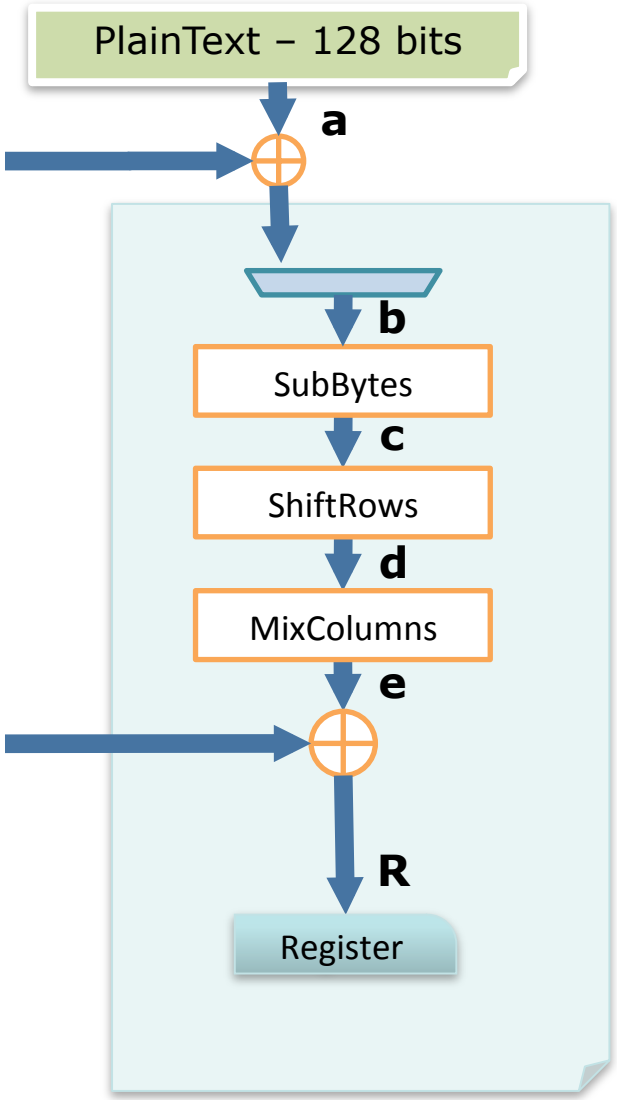
f'_{00}	f_{01}	f_{02}	f_{03}
f'_{10}	f_{11}	f_{12}	f_{13}
f'_{20}	f_{21}	f_{22}	f_{23}
f'_{30}	f_{31}	f_{32}	f_{33}

Réduction de la complexité

- Conclusions :
  - Le changement d'un seul octet de message clair entraine la modification de seulement 4 octets dans le registre de ronde.
  - La distance de Hamming (le ouex) entre les réponses ne dépend que d'un seul octet de la clef (primaire, l'effet de la clef de ronde est supprimé par la ouex entre les 2 valeurs intermédiaires observées grâce au scan)

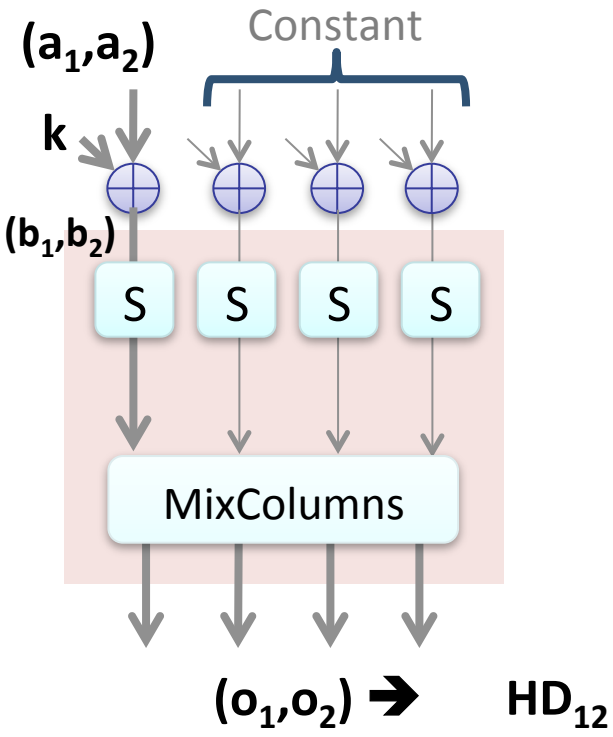
Propriétés

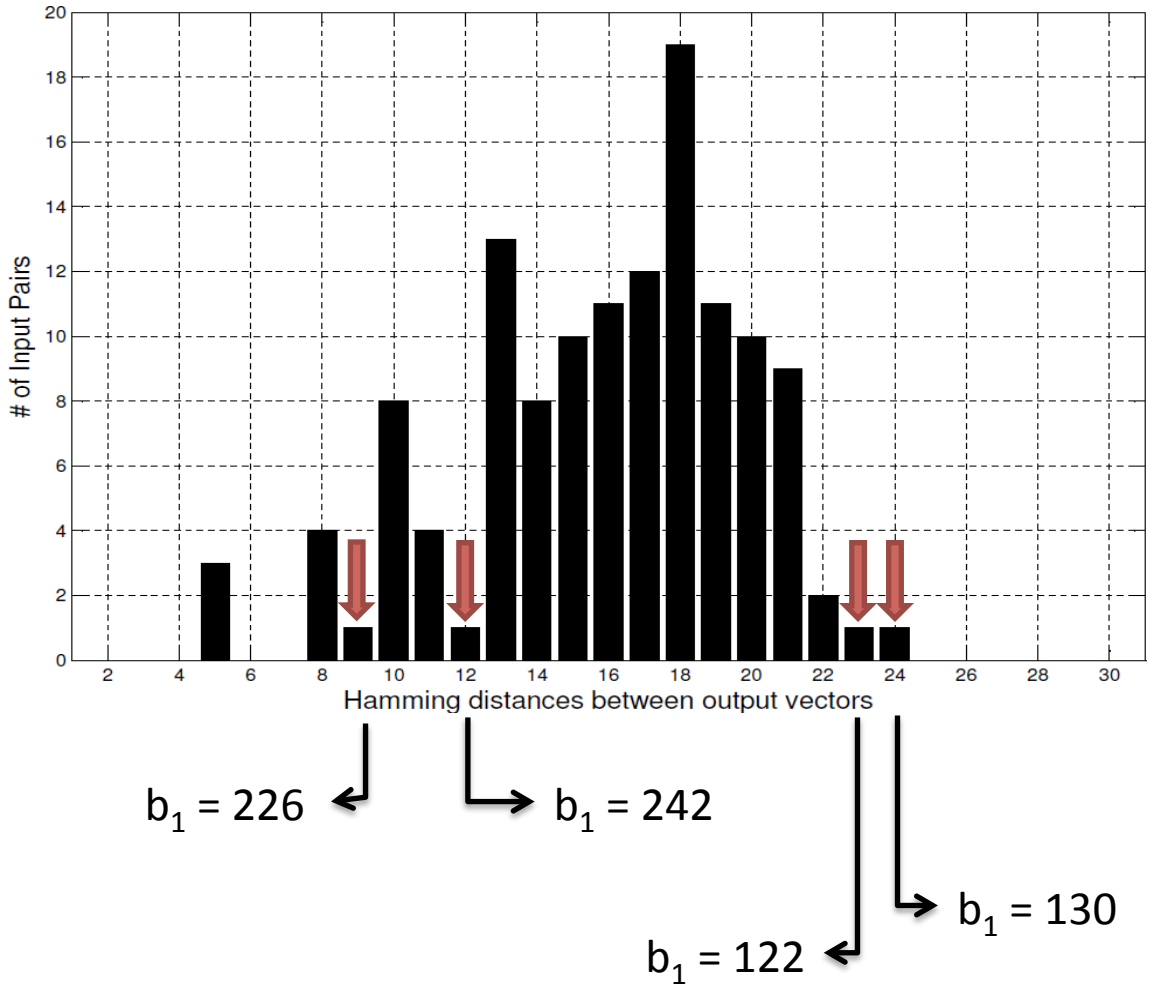
- P1 : Après le changement du bit de poids faible dans un octet de a, la distance de Hamming entre R et R' vaut entre 5 et 27
- P2 : Les valeurs de la distance de Hamming 9, 12, 23 ou 24 correspondent à une unique valeur de b (octet) : 226, 242, 122, 130 (resp.)



Hamming Distance

$(b_1, b_1 \text{ XOR } 1)$





Attaque

- Principe de l'attaque :

  On applique des paires (a,a') de la forme (2m,2m+1)

  - on observe en scan les résultats après la première ronde
  - on calcule la distance de Hamming entre les 2 réponses

  jusqu'à obtenir une distance de Hamming égale à 9, 12, 23 ou 24 (au pire on fait 128 essais)

  On en déduit la valeur de l'octet $b_{i,j}$

  Or $b_{ij} = K_{ij} \oplus a_{ij}$ donc $K_{ij} = a_{ij} \oplus b_{ij}$

  On a un octet de clef

  On recommence pour les autres octets

Attaque scan : exemple

On applique a avec $a_{ij} = (00100010)_2 = 34$
On applique a' avec $a'_{ij} = (00100011)_2 = 35$

On compte le nombre de 1 dans $f \oplus f' = 9$ par exemple

On déduit que $b_{ij} = 226 = (1100\ 0010)_2$ ou $b_{ij}' = 226$

Donc $K_{ij} = a_{ij} \oplus b_{ij} = (00100010)_2 \oplus (1100\ 0010)_2$
$= (1110\ 0000)_2 = 224$

ou bien $K_{ij} = a'_{ij} \oplus b'_{ij} = (00100011)_2 \oplus (1100\ 0010)_2$
$= (1110\ 0001)_2 = 225$

La suite

- 2 possibilités par octet de clef
- On essaie d'autres paires (au pire 127) => Une seule possibilité

- En tout pour les 16 octets de clef :

  128*16 = 2048 essais

- A comparer à une attaque exhaustive :

  $2^{128} \approx 3.10^{38}$ essais

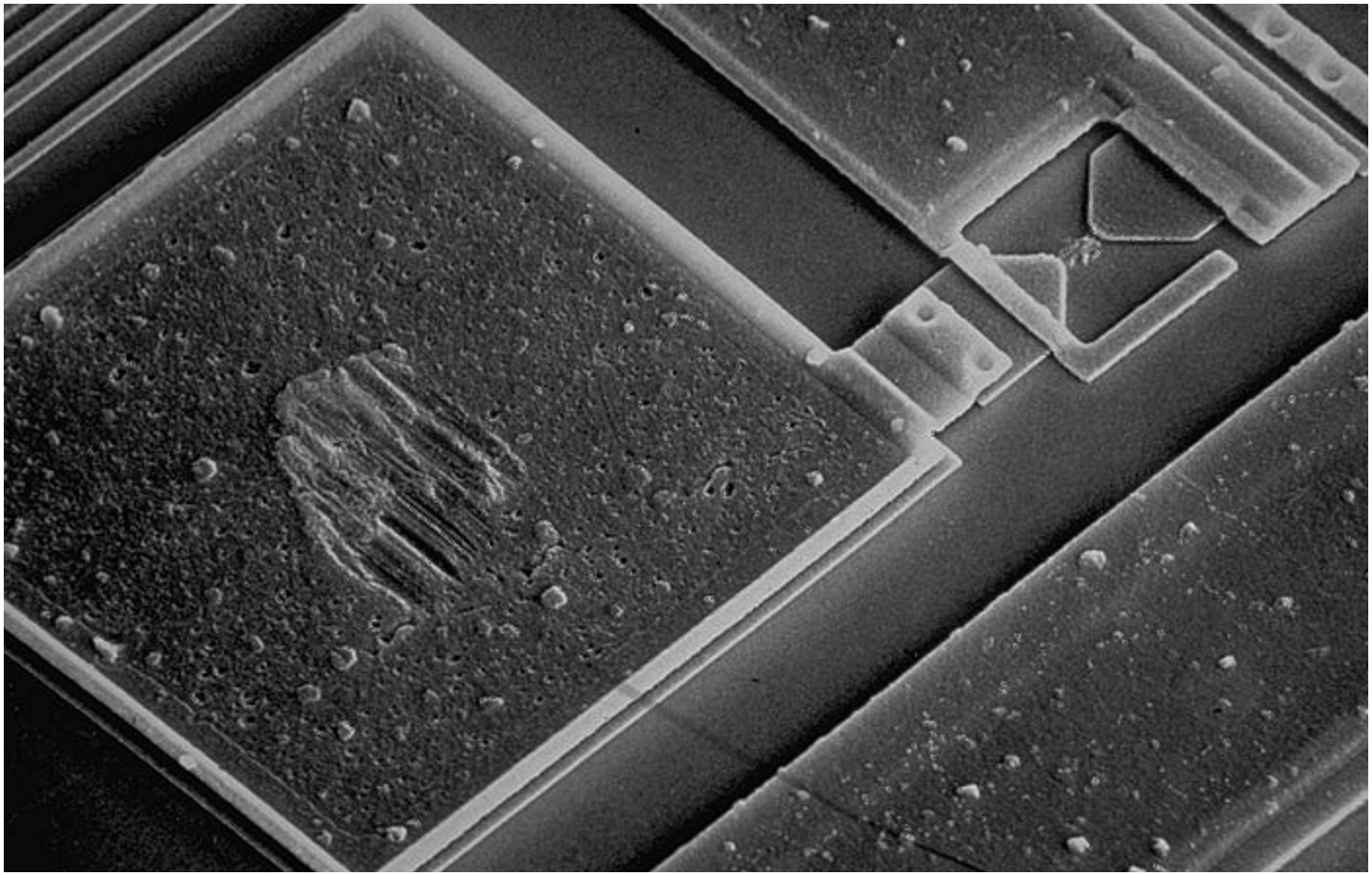
Countermeasures

- Leave the scan chain unbound (fuses)
- Built-In Self-Test
- On-chip test comparison
- Secure Test Access Mechanism
- Scan Chain Encryption
- Industrial solutions

Countermeasures

- **Leave the scan chain unbound (fuses)**
- Built-In Self-Test
- On-chip test comparison
- Secure Test Access Mechanism
- Scan Chain Encryption
- Industrial solutions

Fuses



Countermeasures

- Leave the scan chain unbound (fuses)
- **Built-In Self-Test**
- On-chip test comparison
- Secure Test Access Mechanism
- Scan Chain Encryption
- Industrial solutions

Built-In Self-Test (BIST)

- Motivation:
  - Avoid scan-based testing
  - Allow at-speed testing
  - Reduced ATE cost
- But:
  - Area overhead ?
  - Fault coverage ?

Crypto-Algorithms

- Confusion
  - making the relationship between the key and the ciphertext as complex and involved as possible
- Diffusion
  - a change in a single bit of the plaintext should result in changing the value of many ciphertext bits

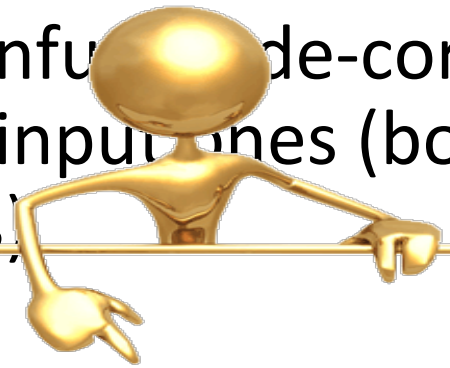
Testability

- Diffusion vs testability:
  - every input bit influences many output bits (i.e. every input bit is in the logic cone of many output bits)
    - the circuit is very observable
  - the input logic cone of every output contains many inputs
    - each fault is highly controllable

- **Therefore these circuits are highly testable by nature no matter the implementations**

Randomness

- Diffusion and confu de-correlate the output values from the inpu nes (both at encryption and round levels)

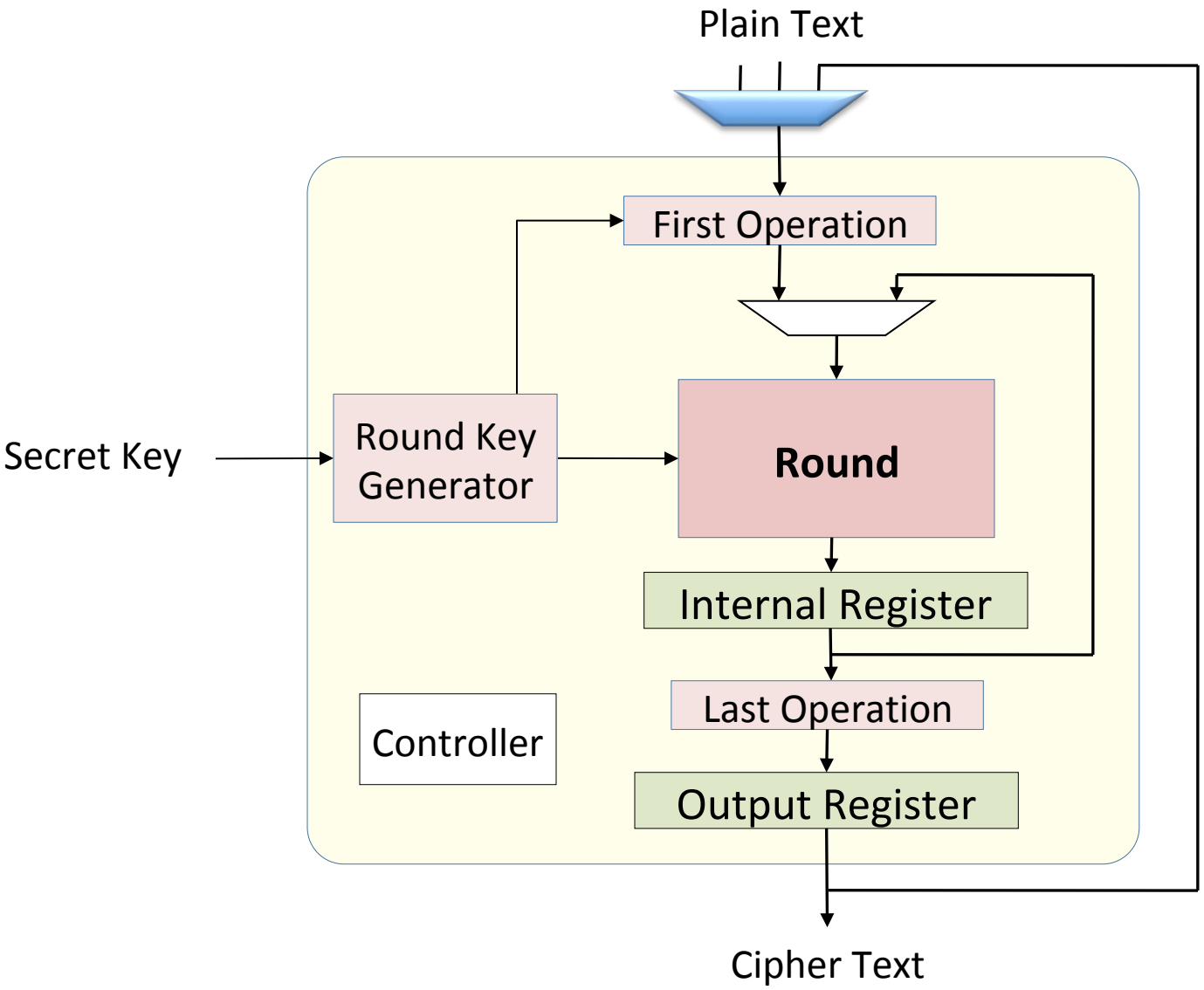


- It has been demonstrated that by feeding back the output to the input, the generated output sequence has random properties

[B. Schneier, Applied Cryptography, Second Ed., John Wiley and Sons, 1996]

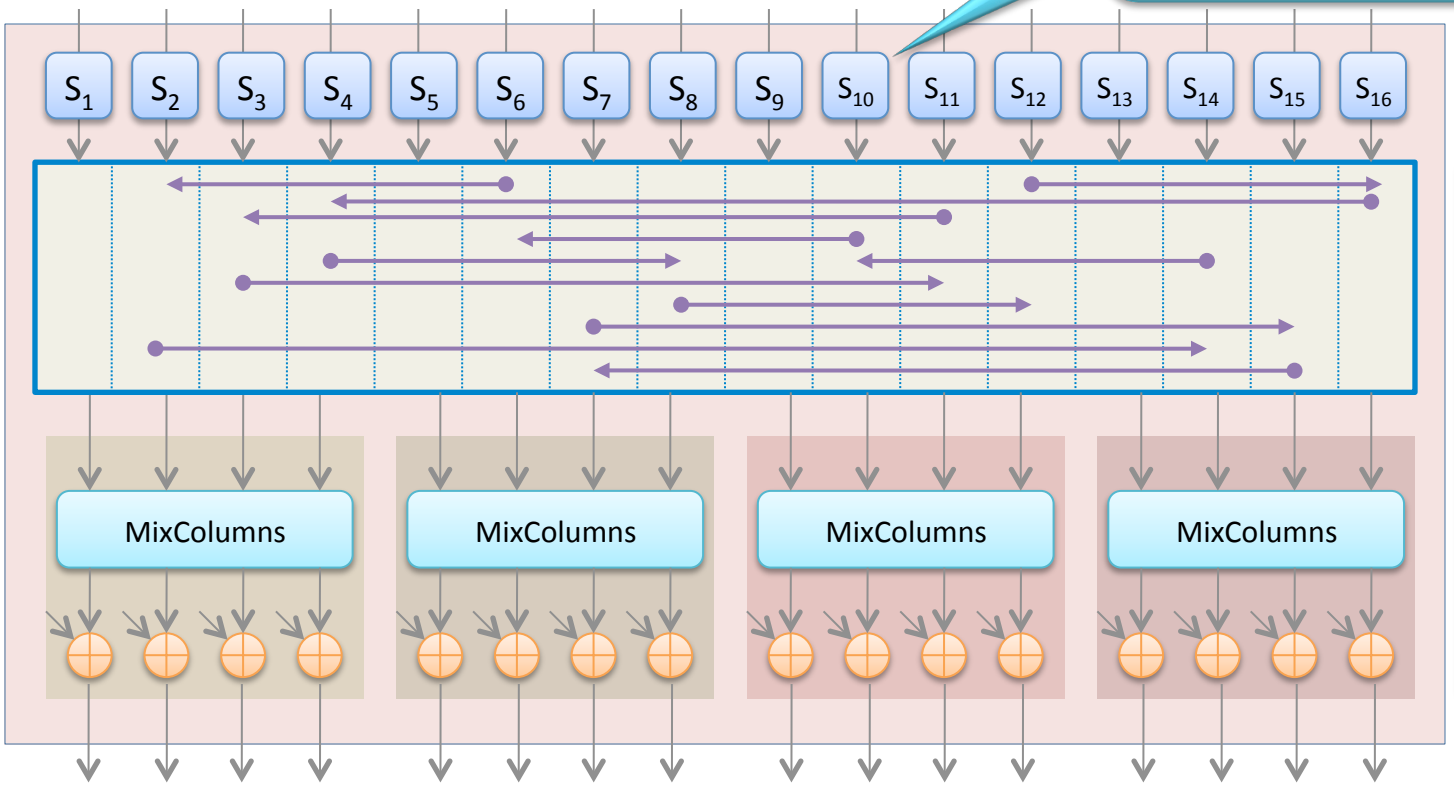
- AES/DES better than LFSR (proved by NIST tests)

BIST Architecture



AES Round Testability

Sbox Implementations:
- ROM → 256 patterns
- Logic → 200 – 220 patterns



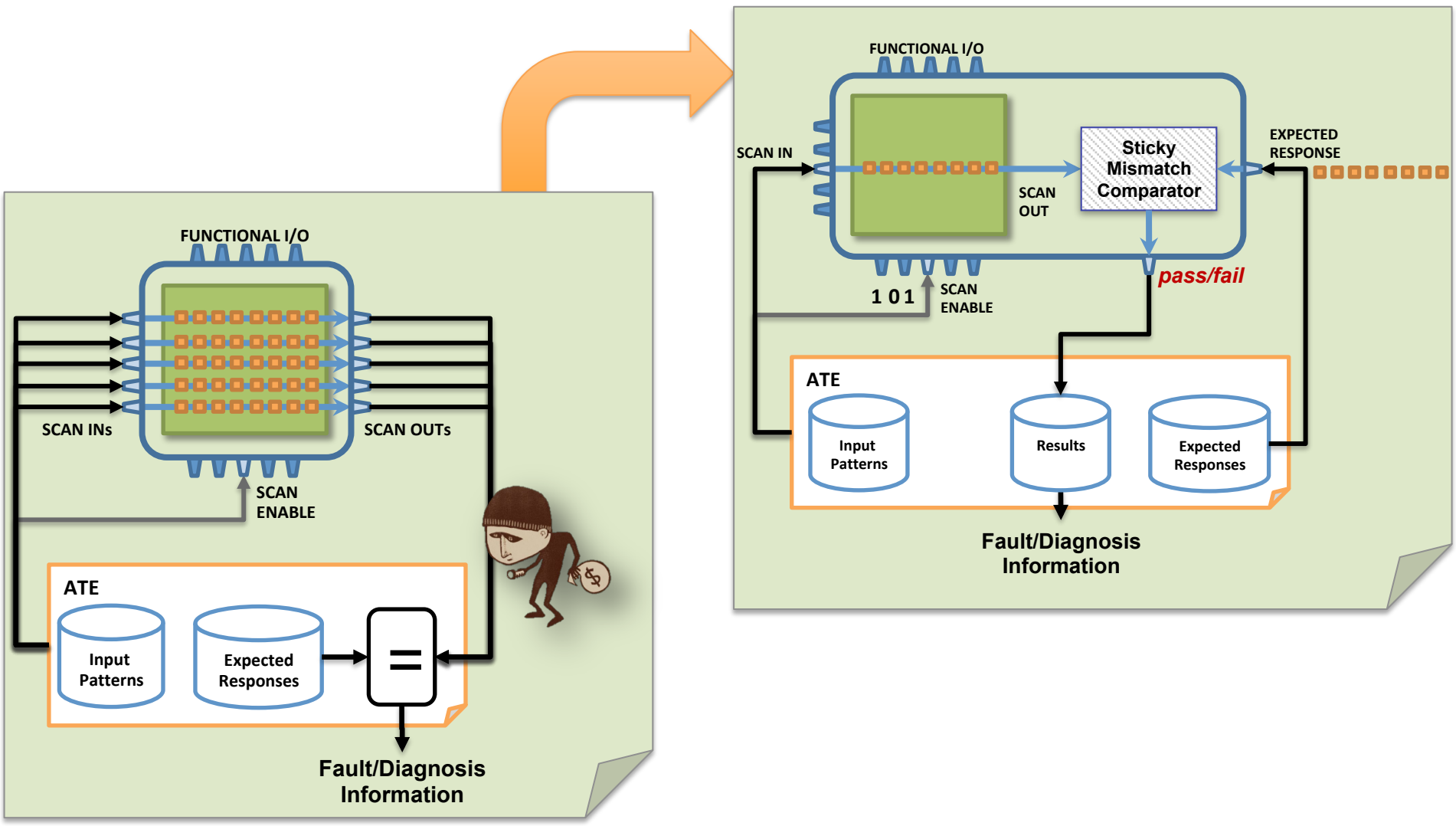
Faults in ShiftRows, MixColumns, Addkey, and Register are also tested by the 200 patterns

Fault coverage: 100% always before 2400 clock cycles
(several keys, several plaintexts)

Countermeasures

- Leave the scan chain unbound (fuses)
- Built-In Self-Test
- **On-chip test comparison**
- Secure Test Access Mechanism
- Scan Chain Encryption
- Industrial solutions

On-Chip Test Comparison



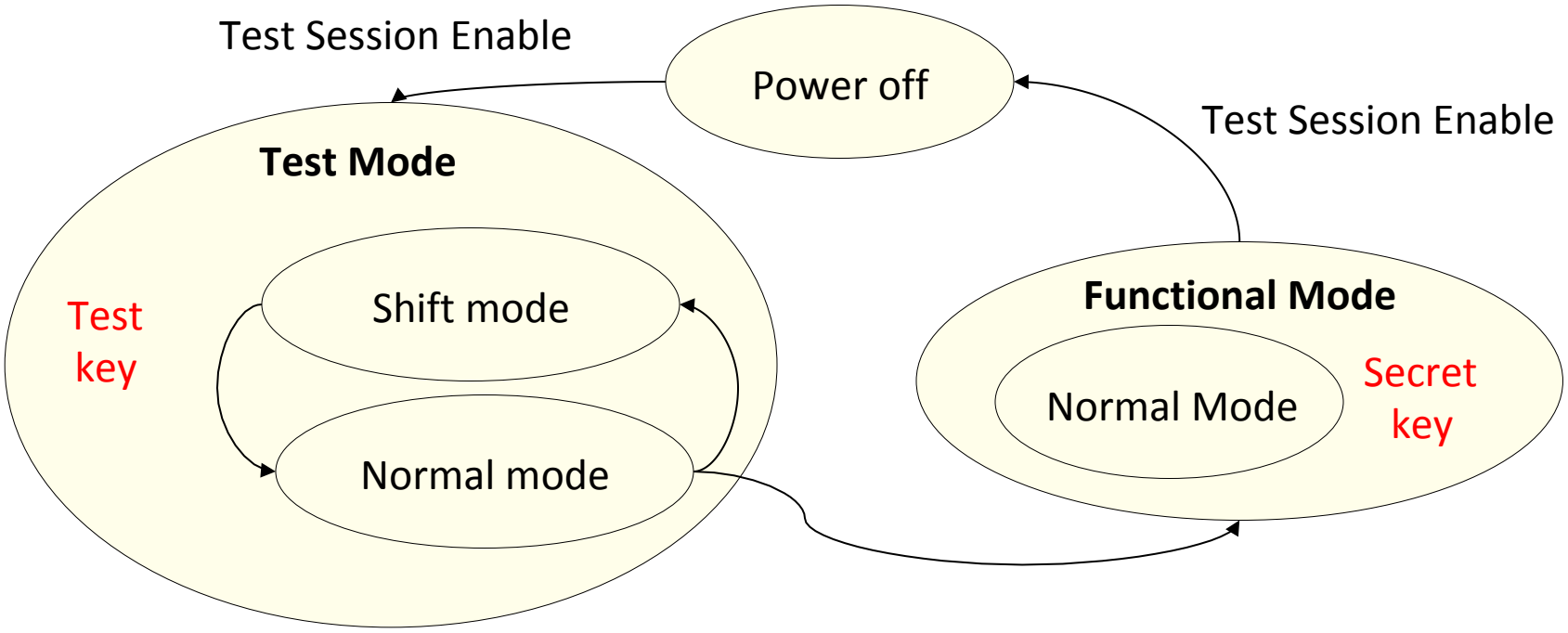
Countermeasures

- Leave the scan chain unbound (fuses)
- Built-In Self-Test
- On-chip test comparison
- **Secure Test Access Mechanism**
- Scan Chain Encryption
- Industrial solutions

Secure Test Access Mechanism

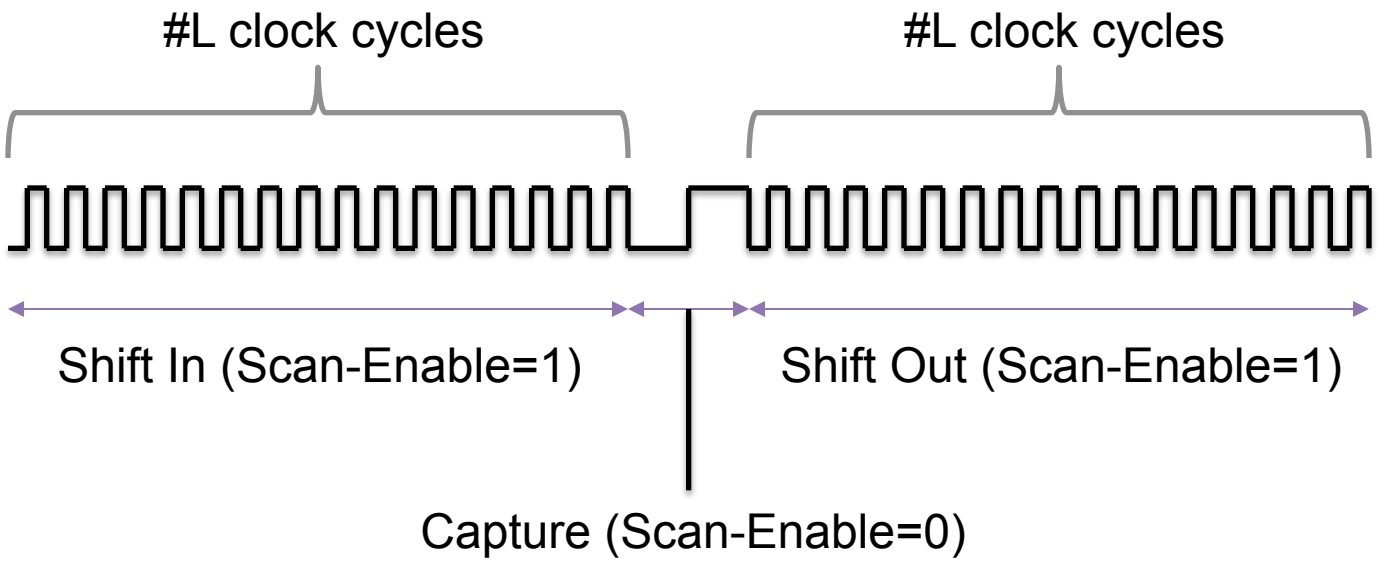
- How to start a test session
  - Additional signal
  - Power on
  - Authentication (public keys, …)
- What to do in functional mode
  - e.g., scrambling, avoiding shift mode, spy Flip Flops
- What to do when switching to test mode
  - e.g., reset FFs
- What to do in test mode
  - e.g., test key instead of actual secret key

Control



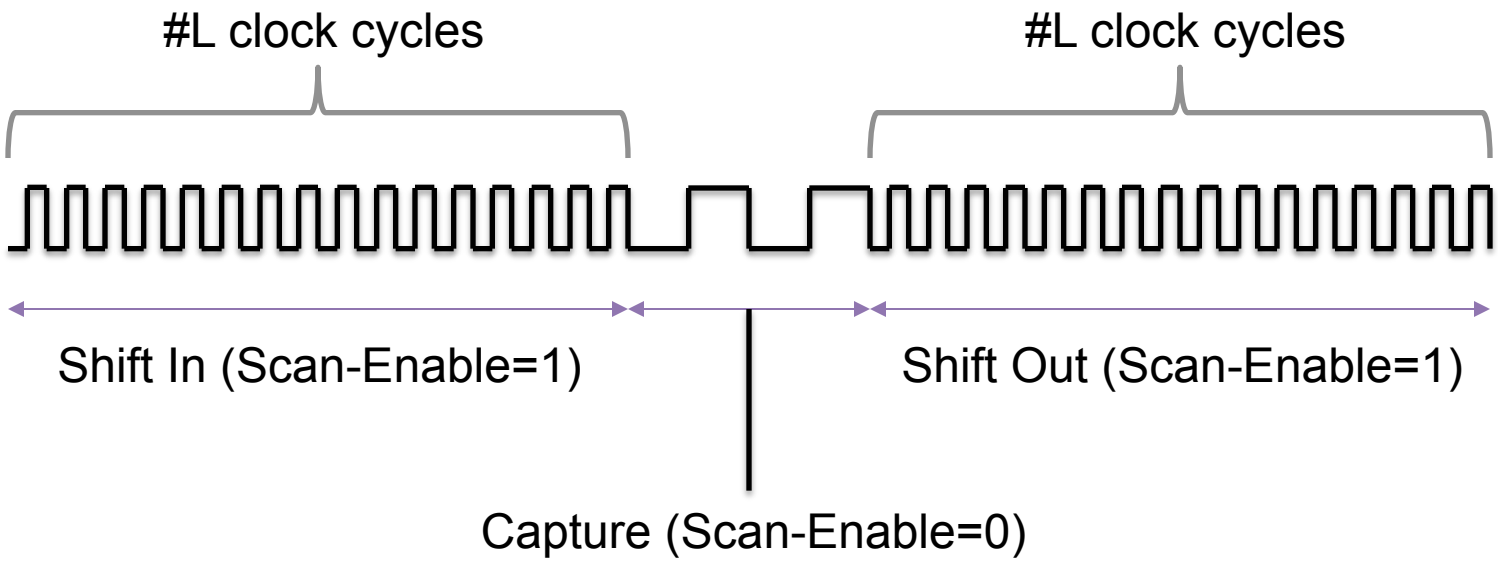
[B. Yang et al., IEEE TRANS. ON CAD, oct. 2006]

Scan Chain Testing

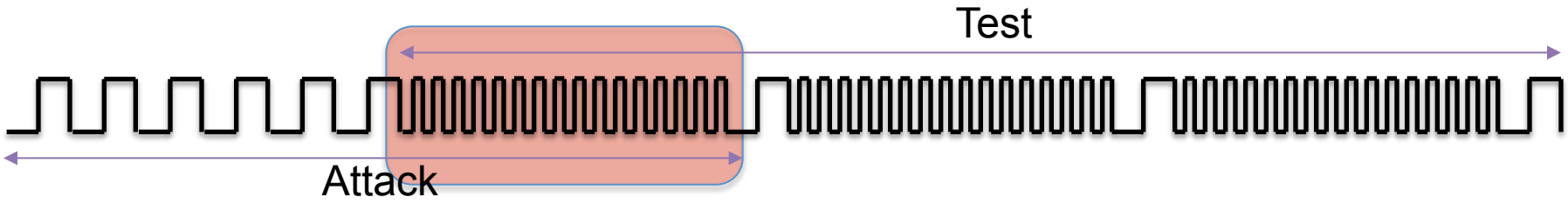


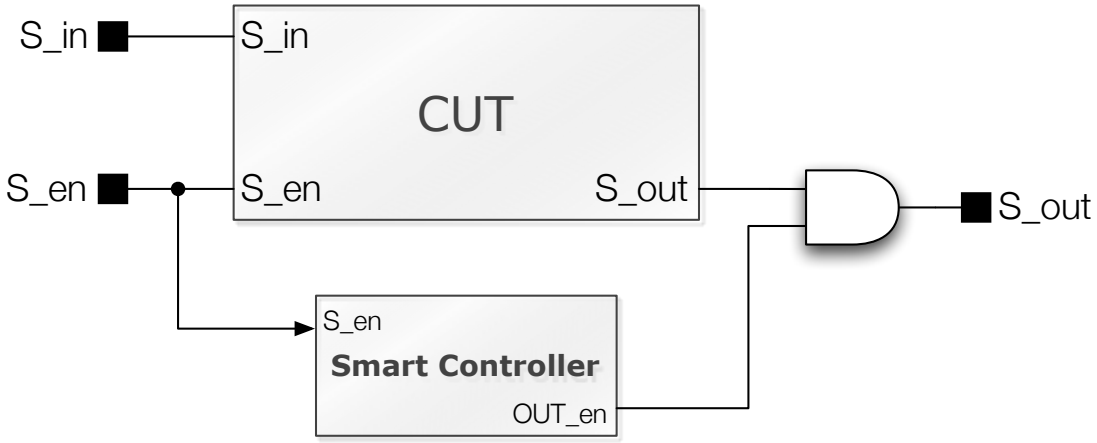
Scan Chain Testing

(delay fault testing with Launch On Capture)



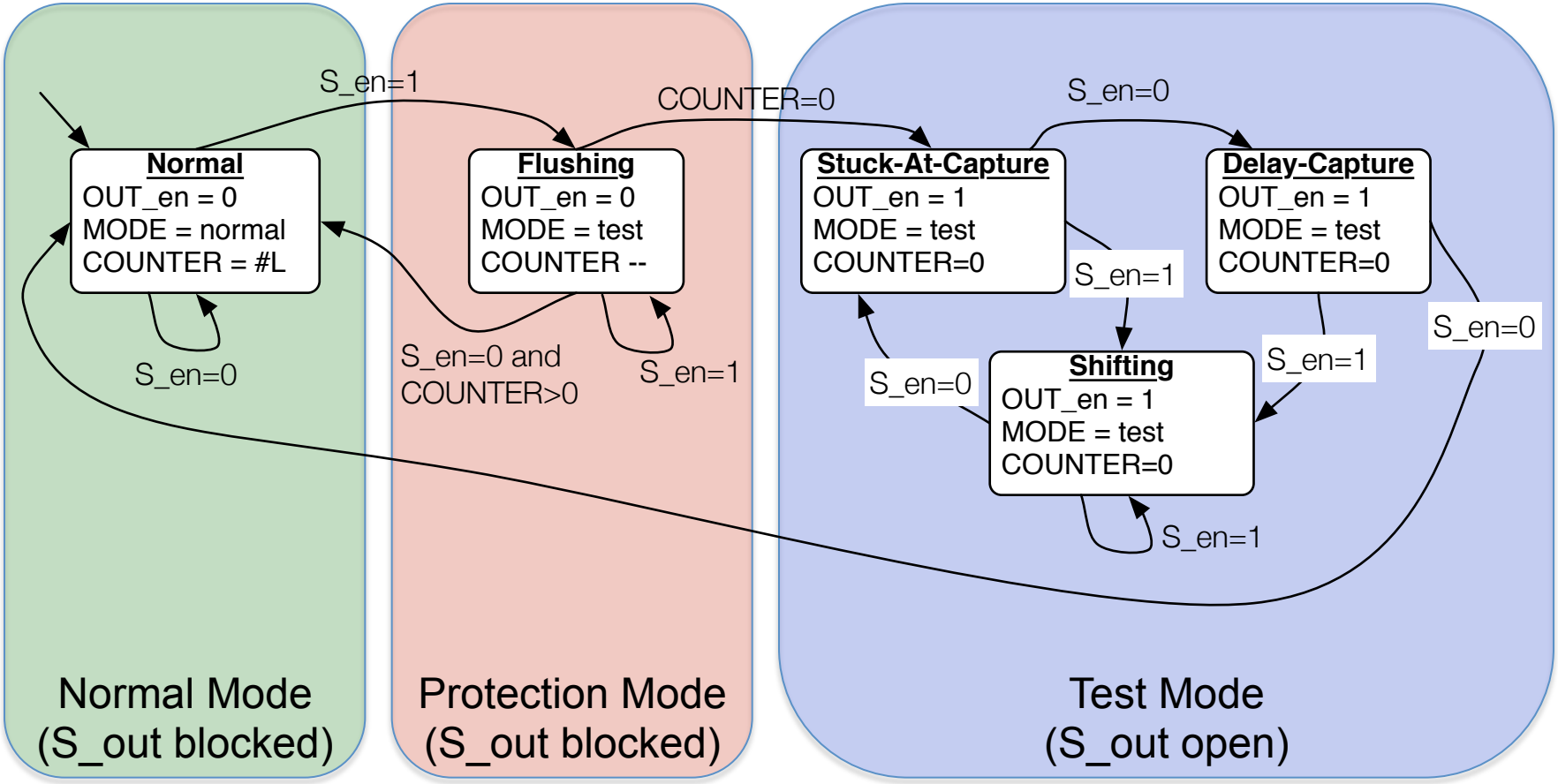
The smart test controller



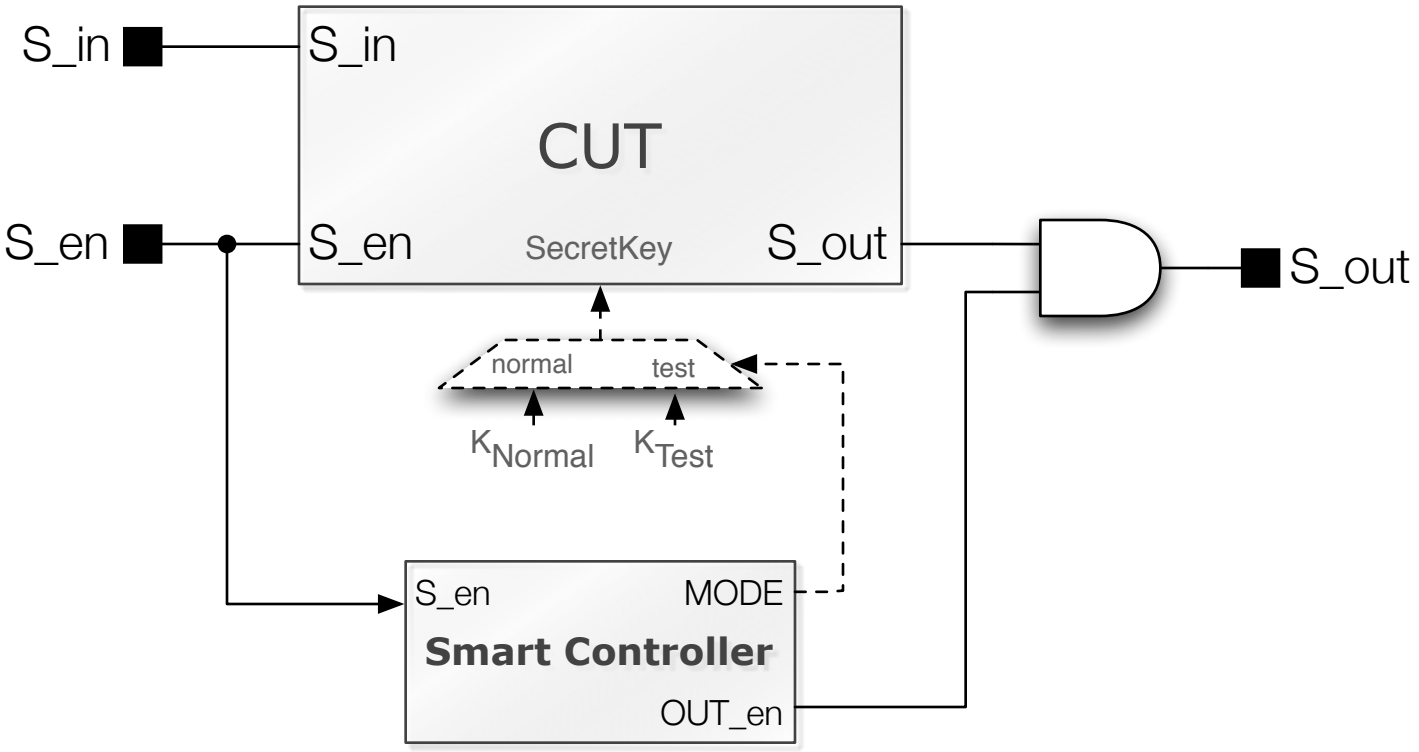


[J. Darolt et al., IEEE IOLTS, 2013]

Detection of the test mode



The smart test controller



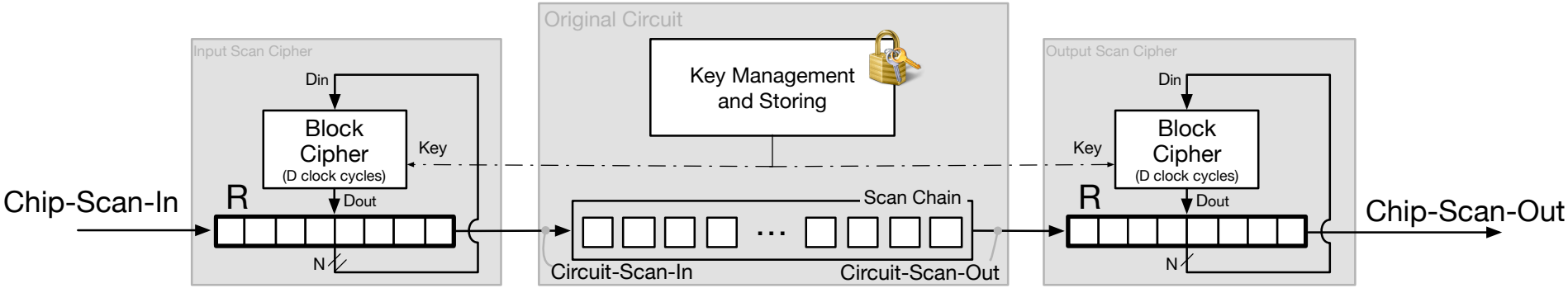
Countermeasures

- Leave the scan chain unbound (fuses)
- Built-In Self-Test
- On-chip test comparison
- Secure Test Access Mechanism
- **Scan Chain Encryption**
- Industrial solutions

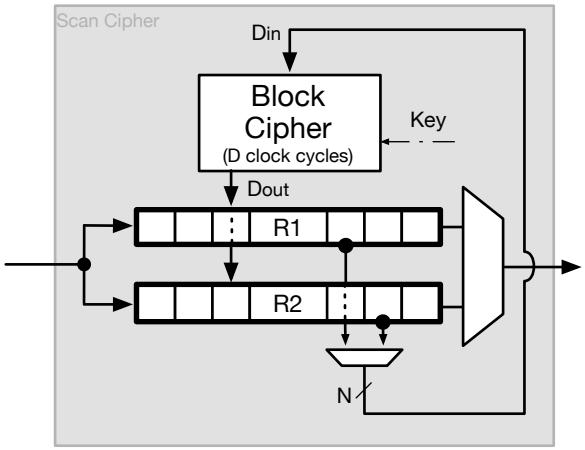
Scan Chain Encryption

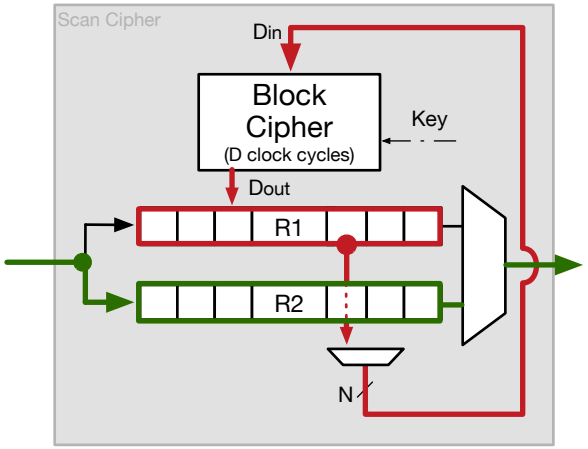
- Encrypting scan chain content with a secret key
- Controllability and the observability:
  - Untouched if the secret key is known
  - Impossible to control or observe otherwise
- Constraints:
  - To modify the test vector and response offline

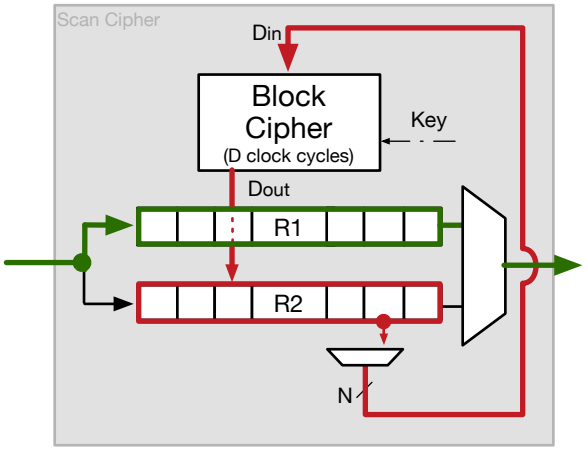
Basic idea


Original Circuit
Key Management
and Storing
Input Scan Cipher
Din
Block
Cipher
(D clock cycles)
Key
R
Dout
N
Chip-Scan-In
Scan Chain
Circuit-Scan-In
Circuit-Scan-Out
Output Scan Cipher
Din
Block
Cipher
(D clock cycles)
Key
R
Dout
N
Chip-Scan-Out

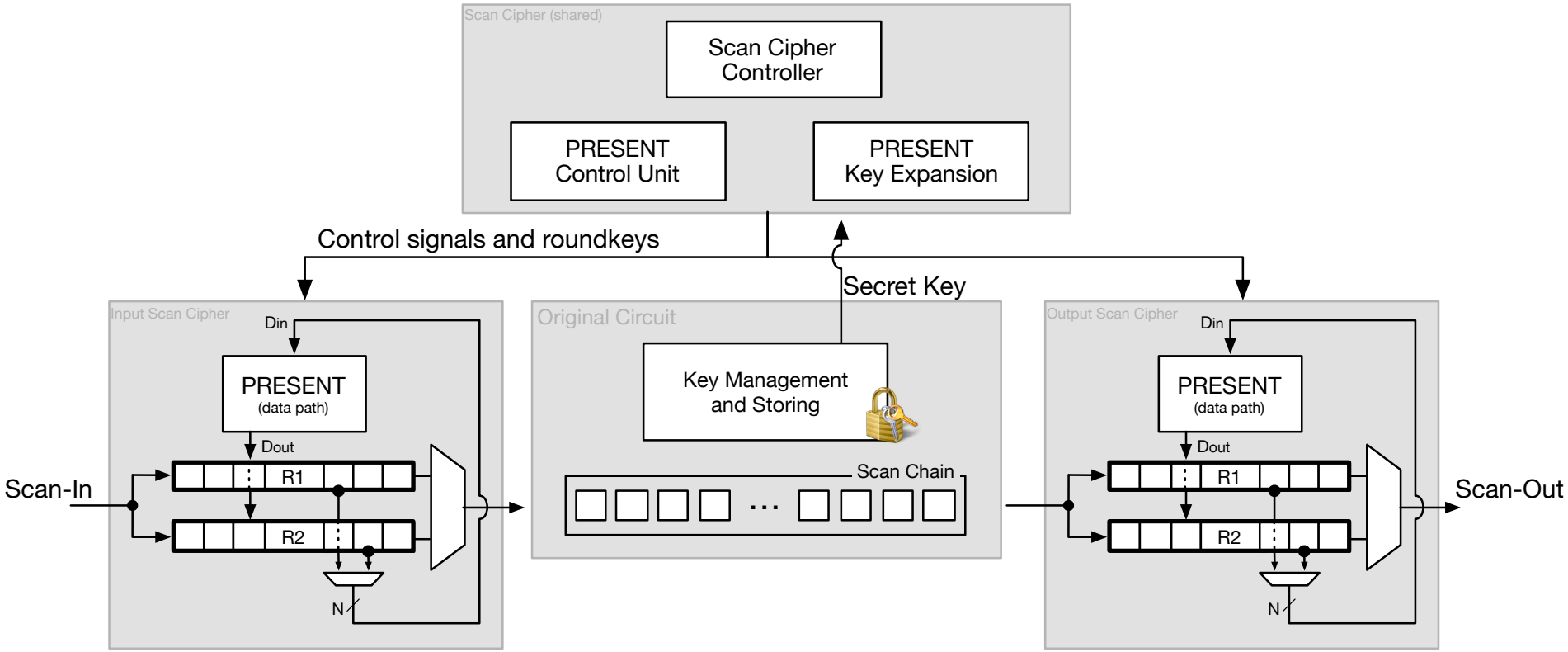
Solution


Scan Cipher
Din
Block Cipher
(D clock cycles)
Key
Dout
R1
R2
N


Scan Cipher
Din
Block Cipher
(D clock cycles)
Key
Dout
R1
R2
N


Scan Cipher
Din
Block Cipher
(D clock cycles)
Key
Dout
R1
R2
N

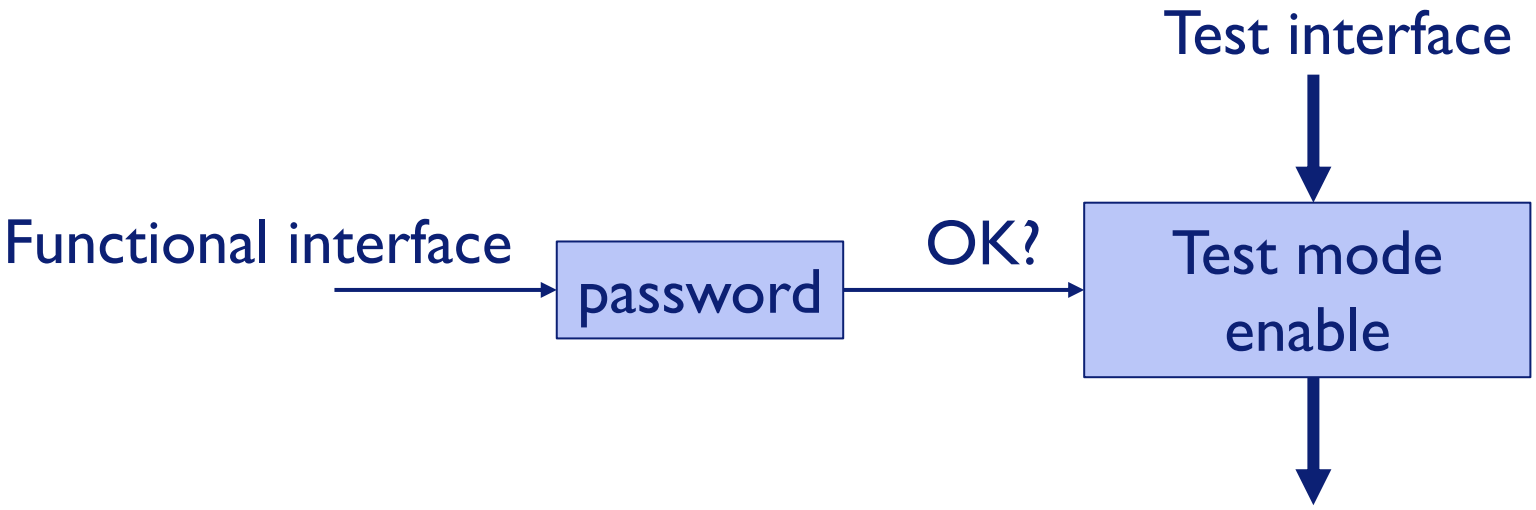
Solution


Scan Cipher (shared)
Scan Cipher Controller
PRESENT Control Unit
PRESENT Key Expansion
Control signals and roundkeys
Secret Key
Input Scan Cipher
Din
PRESENT (data path)
Dout
R1
R2
N
Original Circuit
Key Management and Storing
Scan Chain
Output Scan Cipher
Din
PRESENT (data path)
Dout
R1
R2
N
Scan-In
Scan-Out

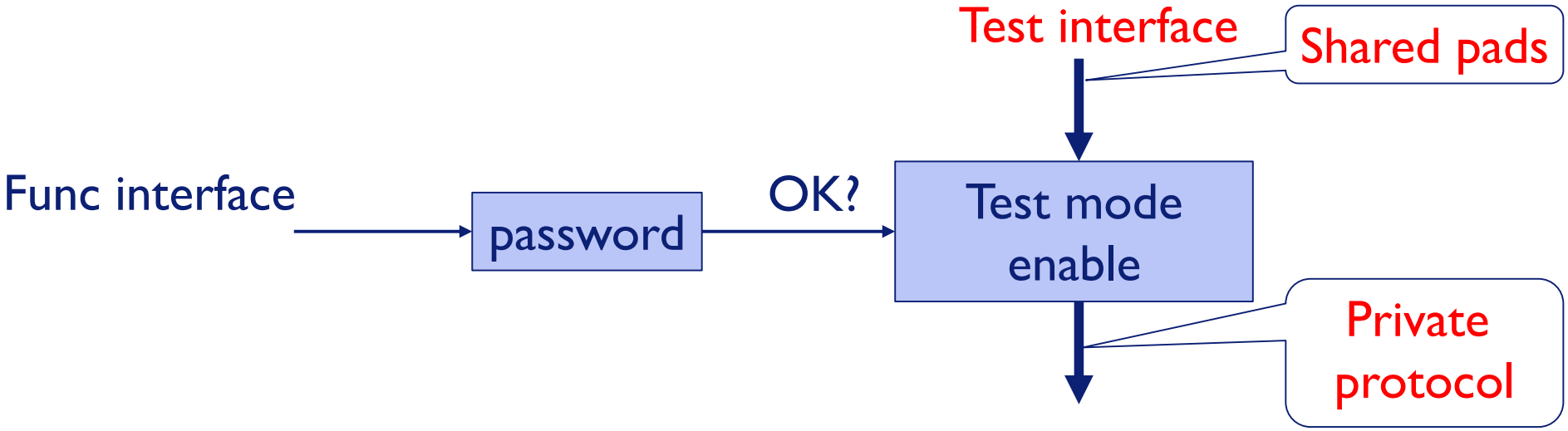
Countermeasures

- Leave the scan chain unbound (fuses)
- Built-In Self-Test
- On-chip test comparison
- Secure Test Access Mechanism
- Scan Chain Encryption
- **Industrial solutions**

Password

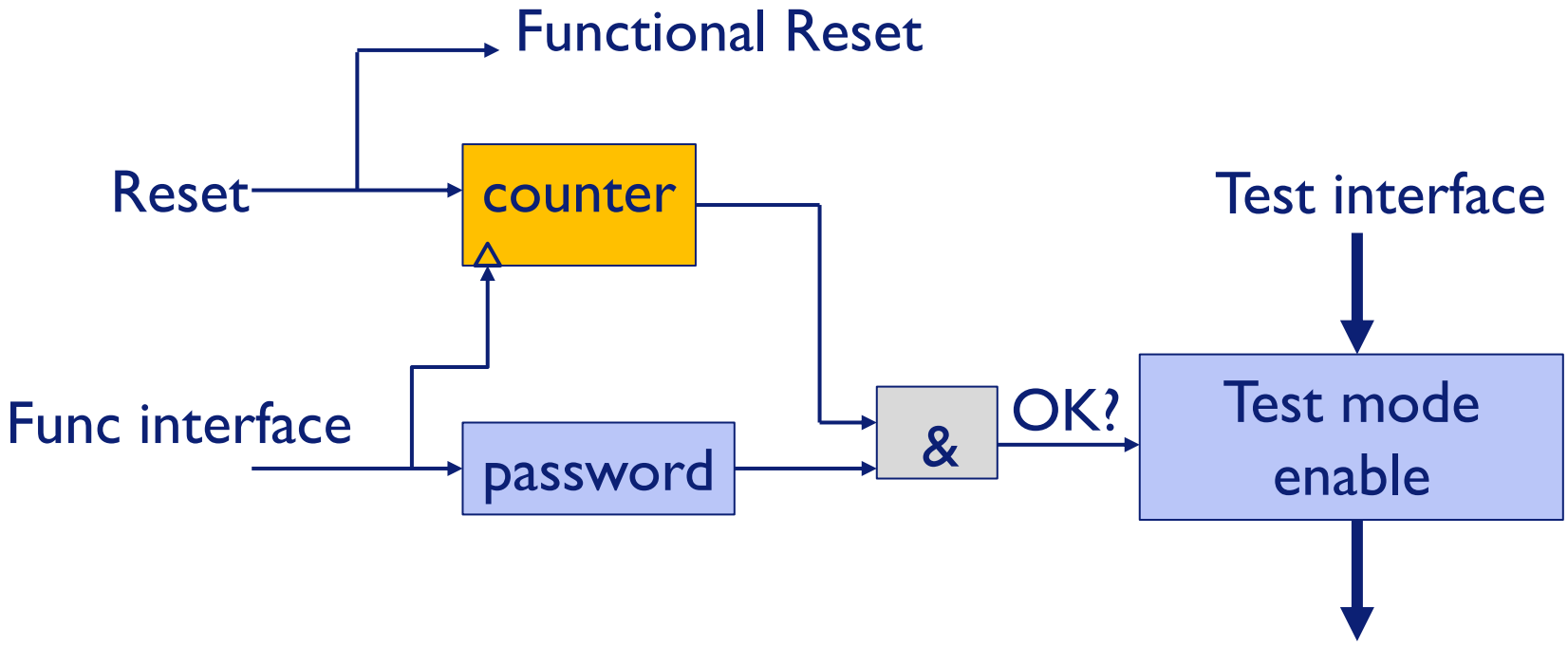

Test interface
Functional interface
password
OK?
Test mode
enable

Custom test protocol



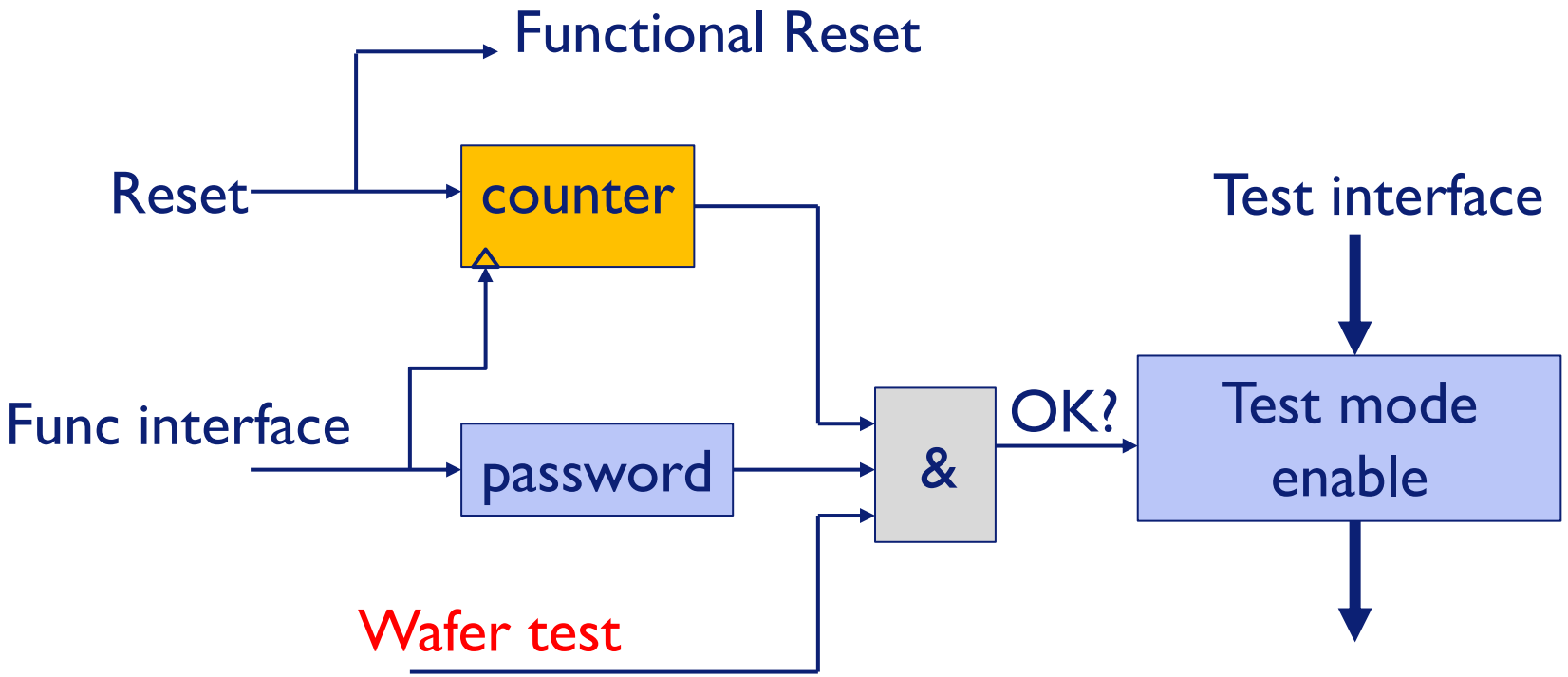
Make it complex

- Password only available few clock cycles after or during RESET



Make it complex (2)

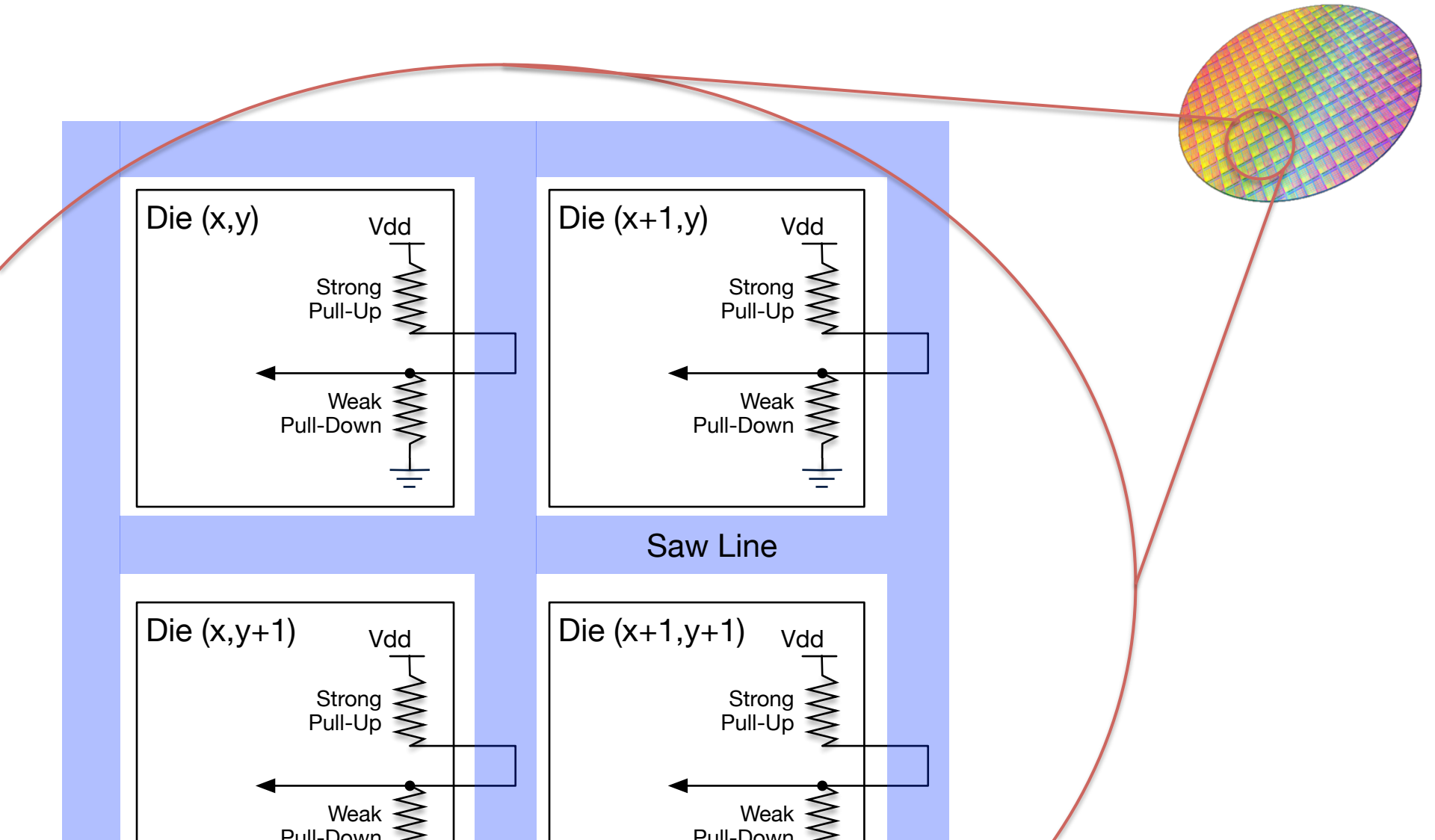
- Password available only at wafer level



Wafer Test Detection

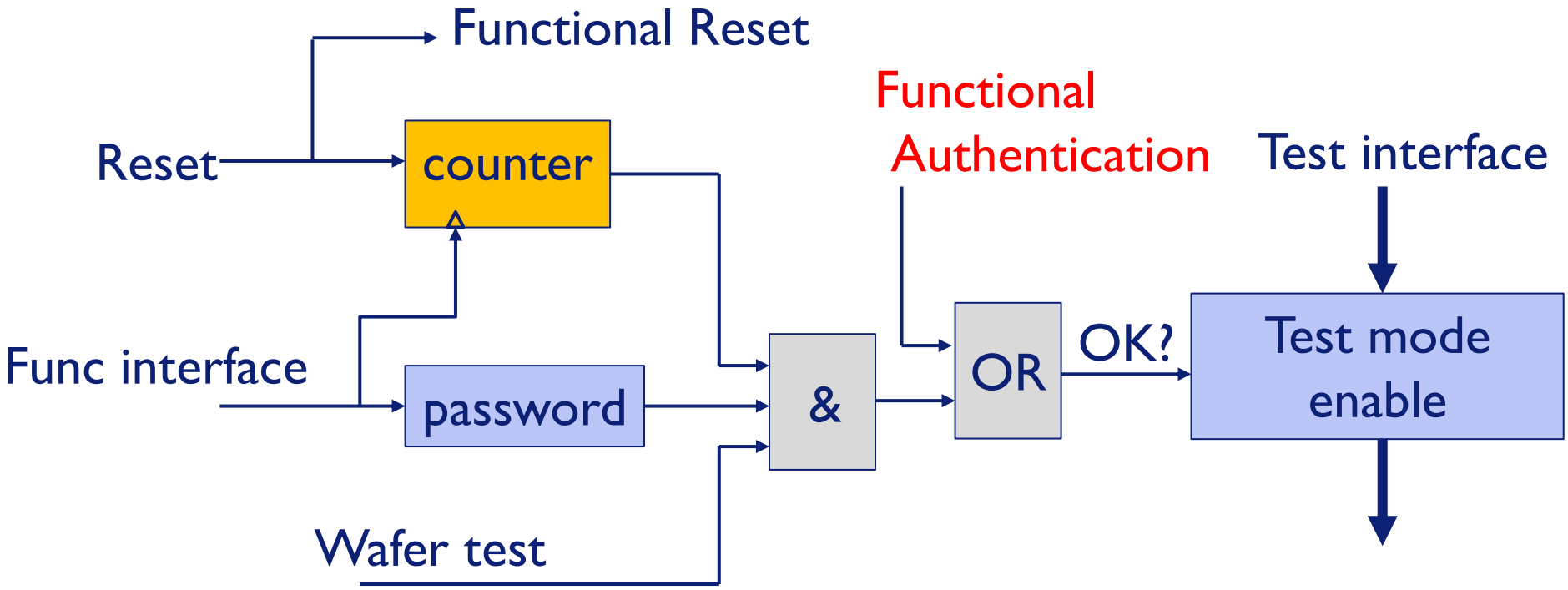
- OPT (One Time Programmable), Poly and Laser Fuses
  - Important area overhead, easy to localized and repair!
- MTP (Multi time Programmable), Non volatile memory
  - Need to be protected to avoid abnormal write access
- Saw bow, physical implementation across saw line
  - Must be compatible with the production flow/technology

Traits de scie



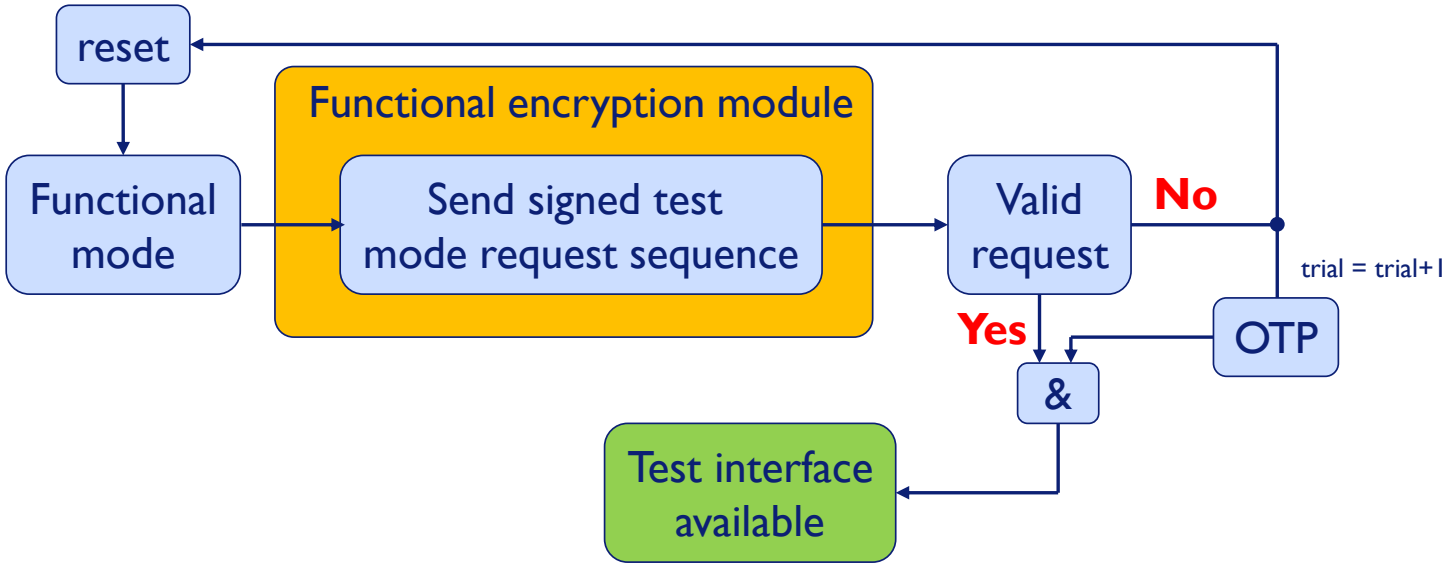
Make it complex (3)

- Password available only at wafer level, then function authentication is required



Authentification

1. Identique pour tous les circuits
   - Même séquence pour tous les circuits
   - Utilisation d'un module d'encryption embarqué
   - Utilisation d'OTP pour limiter le nombre d'essais.



Authentification

2. Authentification unique pour chaque circuit

Secure back box on tester

Constraints

- Test entry implementation have to be adapted to:
  - Available hardware (embedded encryption modules)
  - Test time constraints (symmetric or asymmetric encryption?)
  - Test environment (secure area?)
  - Expected level of security
  - Transparent to the final user

Conclusions on Testing

- Test of Secure devices is critical
- Test of Secure devices is possible (at higher costs)

- Nothing is perfect, solutions should be improved based on coming attacks!