TELECOM ParisTech

Une école de l'IMT

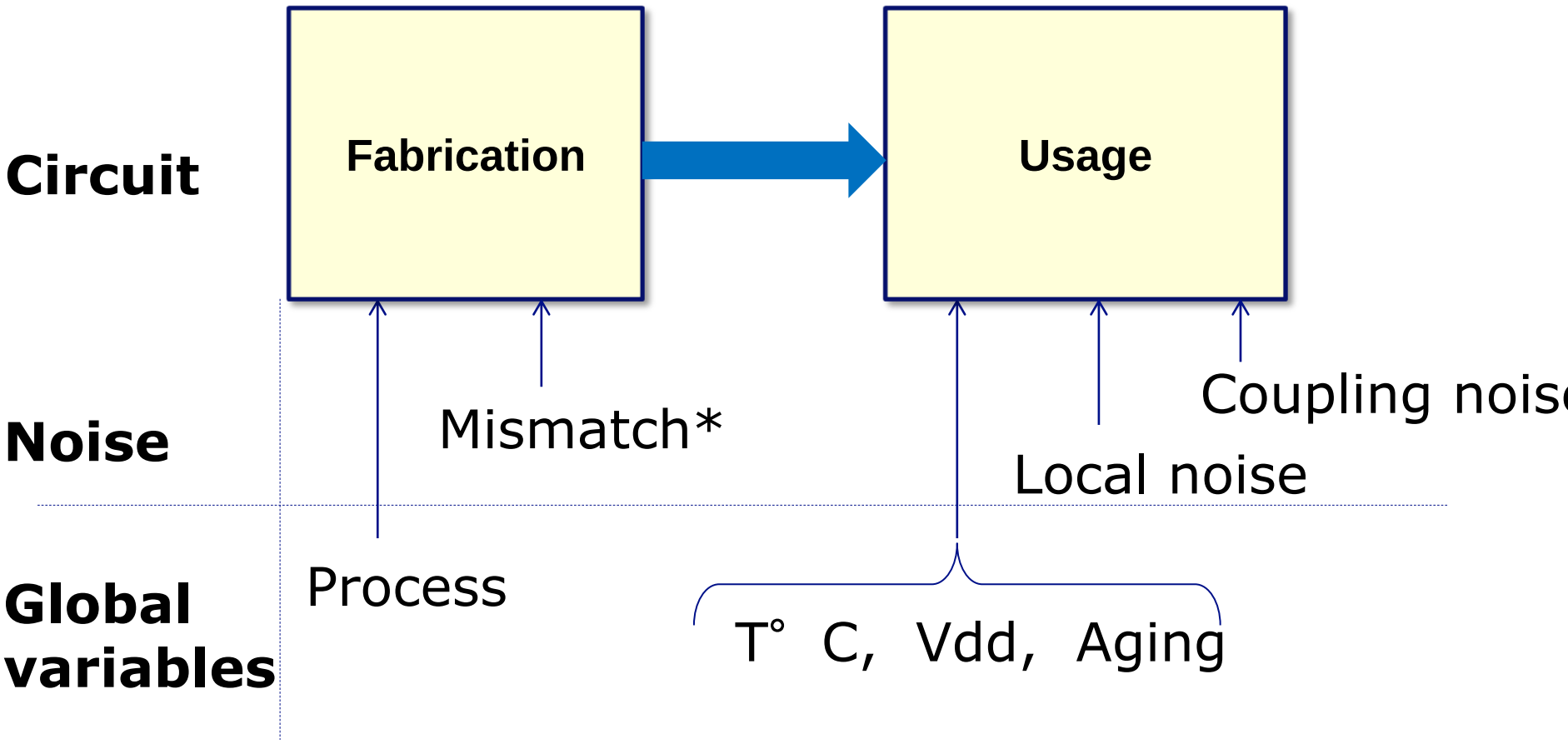
Institut Mines-Télécom

SE304

PUF and TRNG primitives

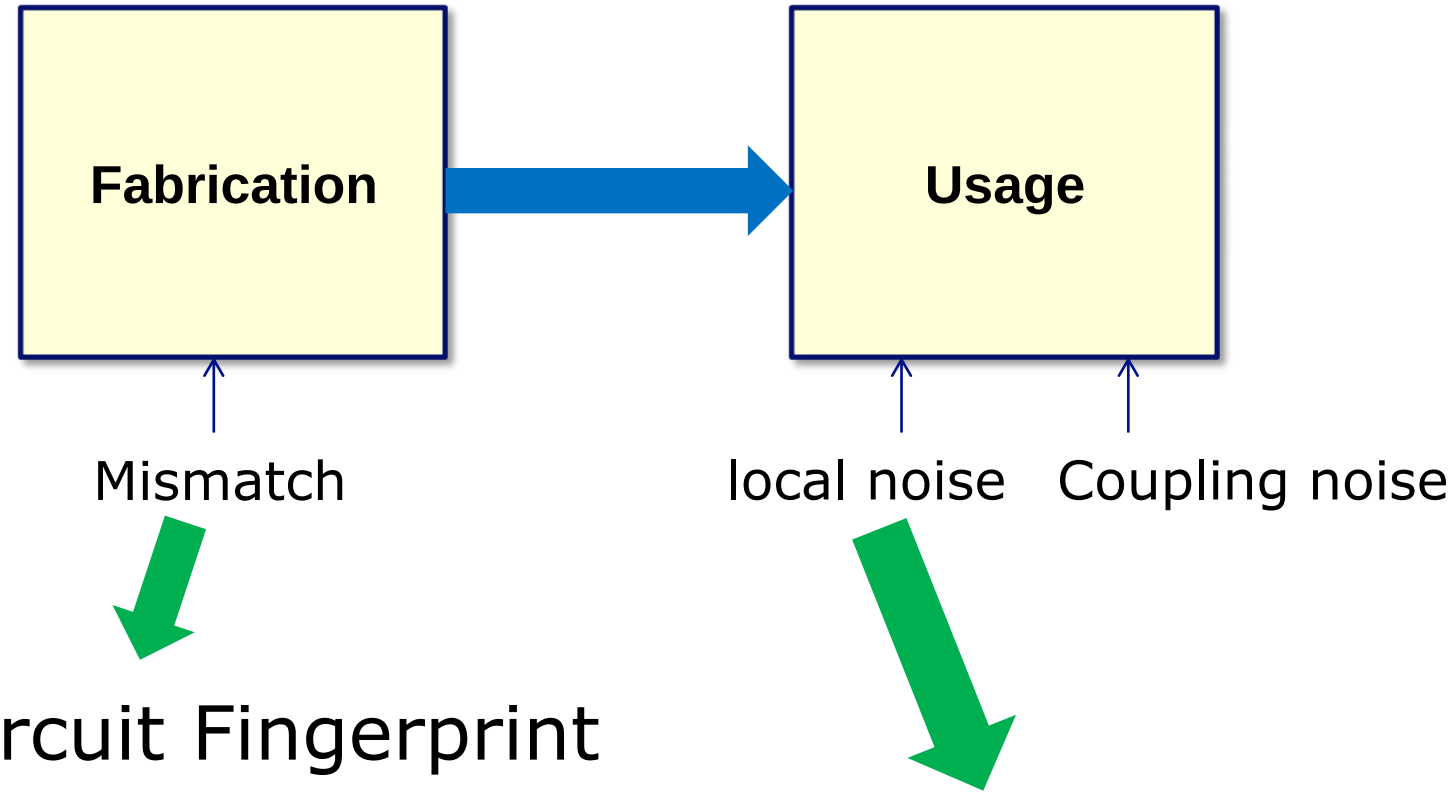
Jean-Luc DANGER

Variability impacting a digital CMOS circuit

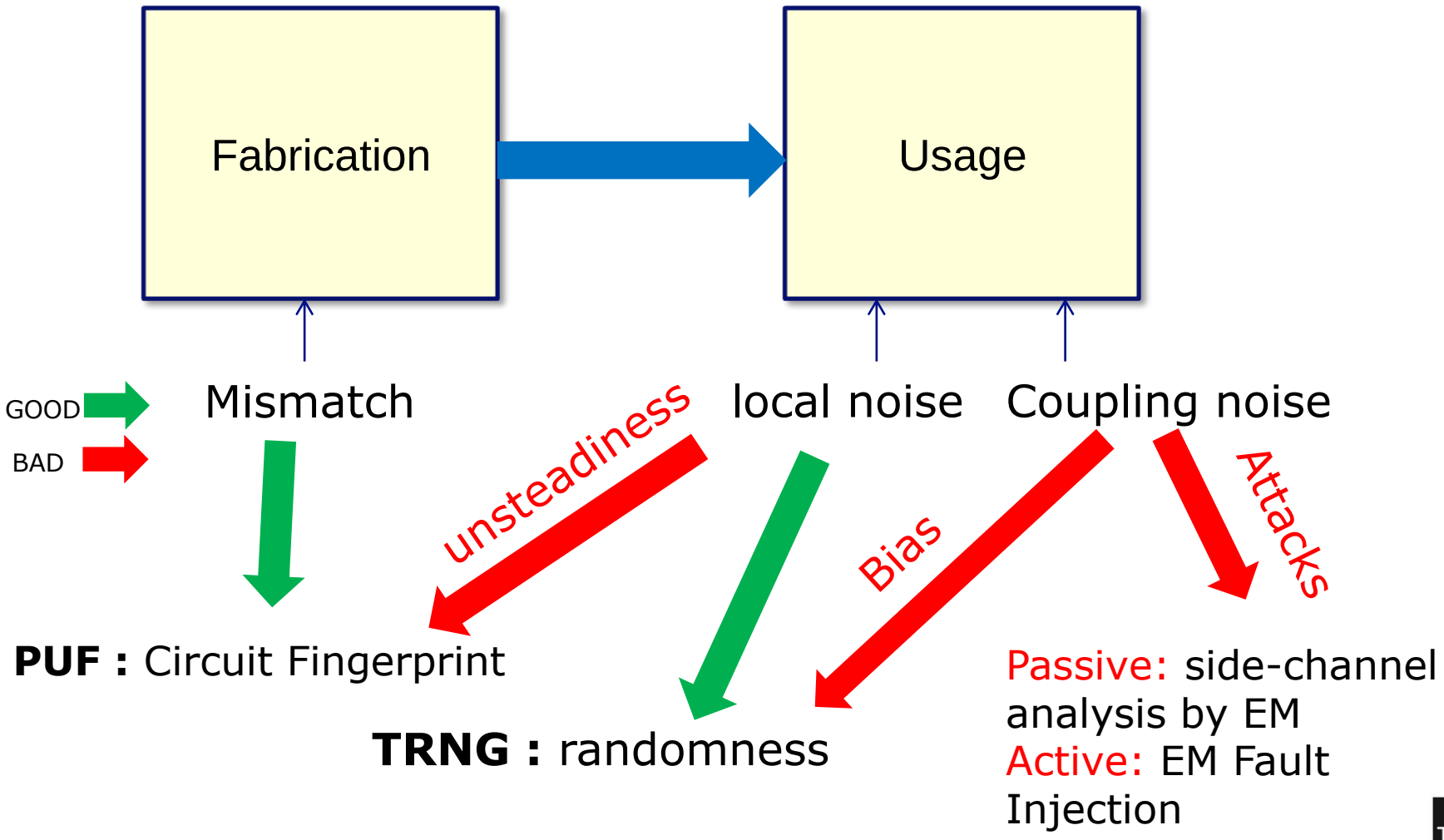


*becomes static after fabrication

The noise can be an advantage for security



But noise creates security weaknesses

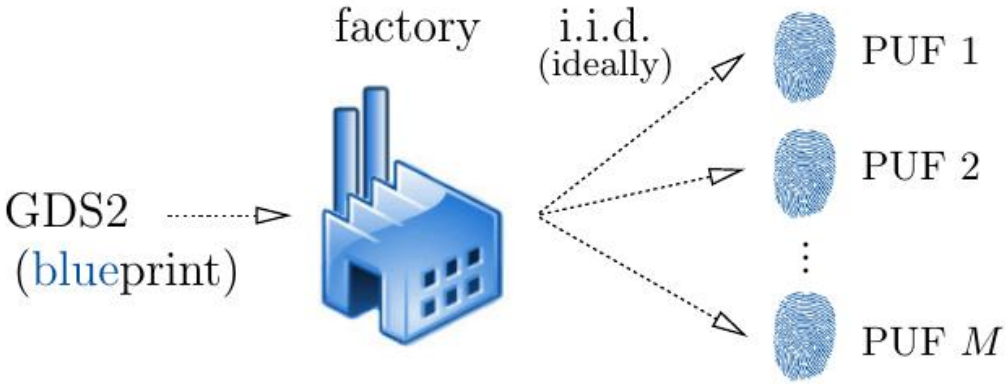


Outline

- **PUF**
  - Architectures
  - Properties
- **TRNG**
  - Architectures
  - Properties
- **Conclusions**

PUF

- **Function returning the fingerprint of the device**
  - Physical function,
  - which exploits material randomness,
  - and is unclonable: same structure for each device



PUFs are instanciations of blueprints by a fab plant

The **fingerprint** can be:

• a unique identifier
for **cryptographic Key**

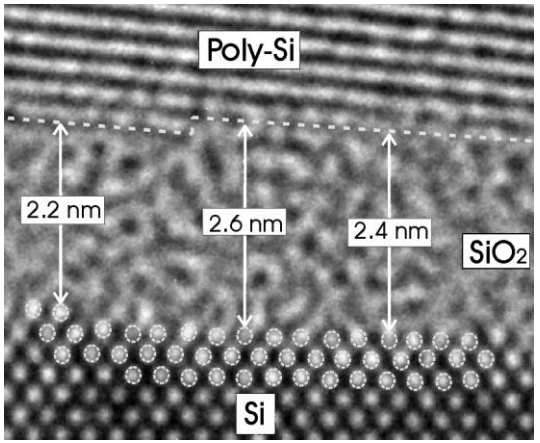
• a list of responses for a given set of challenges
Simple **CRP** authentication protocol

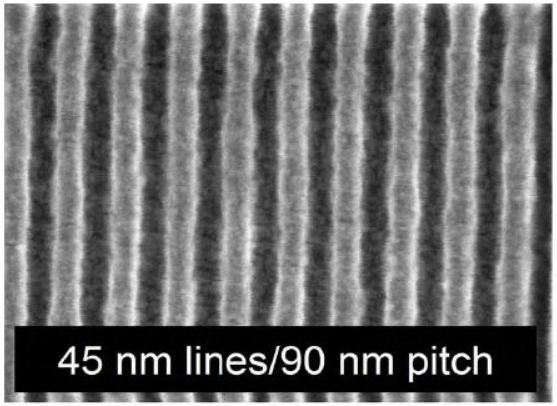
K. Lofstrom, W.R. Daasch, and D. Taylor, Ic identification circuit using device mismatch, Solid-State Circuits Conference, 2000. Di-gest of Technical Papers. ISSCC. 2000 IEEE International, 2000, pp. 372–373.

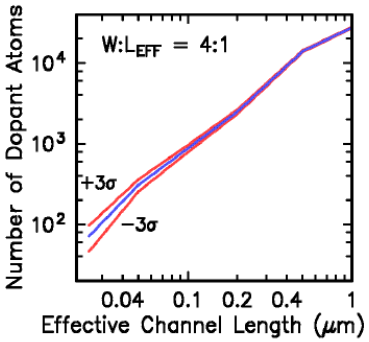
Optical PUF : Ravikanth S. Pappu. Physical One-Way Functions. PhD thesis, Massachusetts Institute of Technology, March 2001.

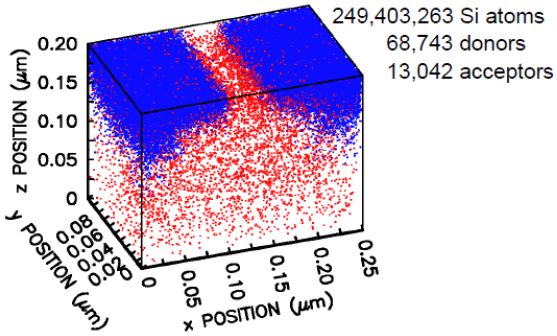
Local mismatch: CMOS process variation

- **Examples**
  - Oxide thickness
  - Metal line edge roughness
  - Random dopant fluctuation









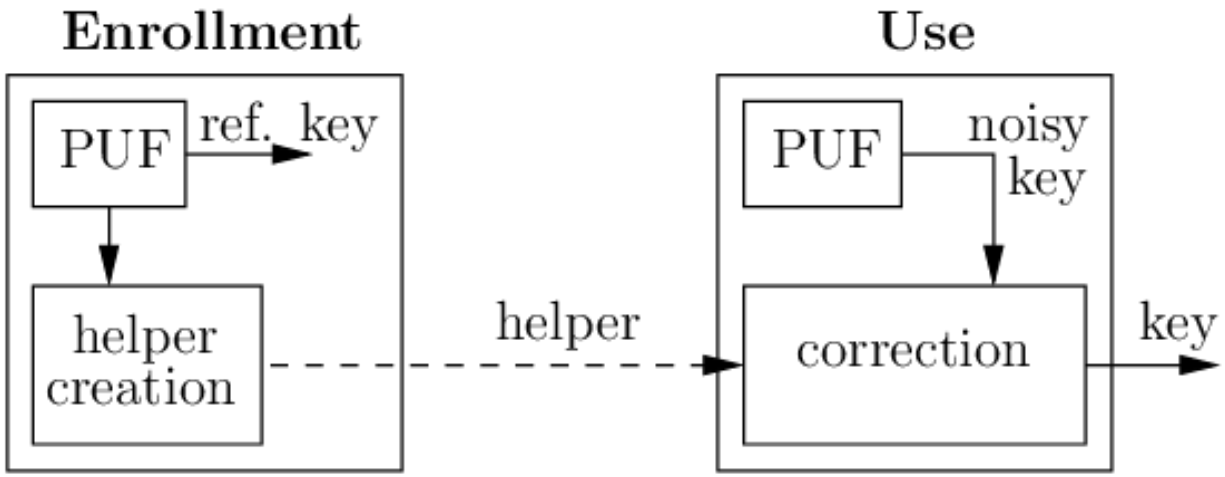
[D. J. Frank, et al., 1999 Symp. VLSI Tech.]

PUF Applications and benefits

- **Applications**
  - IP protection
    - The IP can run only on the authorized device
  - Secure Boot
    - The OS is loaded and deciphered only on the authorized device
  - Safe guard
    - The data are ciphered before being stored in an untrusted device
  - RFID / NFC tag
    - A product can be authenticated and traced (anti-counterfeiting)
- **Benefits**
  - No programmation, contrary to NVM
  - Not clonable, as same blueprint
  - Standard CMOS process: silicon PUF

Two phases of use

- **1. Enrollment**
  - To do once after manufacturing
  - In order to get a "reference PUF" and possibly a "helper data" to rebuild it
- **2. Usage (or reconstruction)**
  - To obtain the reference back when using the PUF
  - Can use a "helper data" for greater reliability

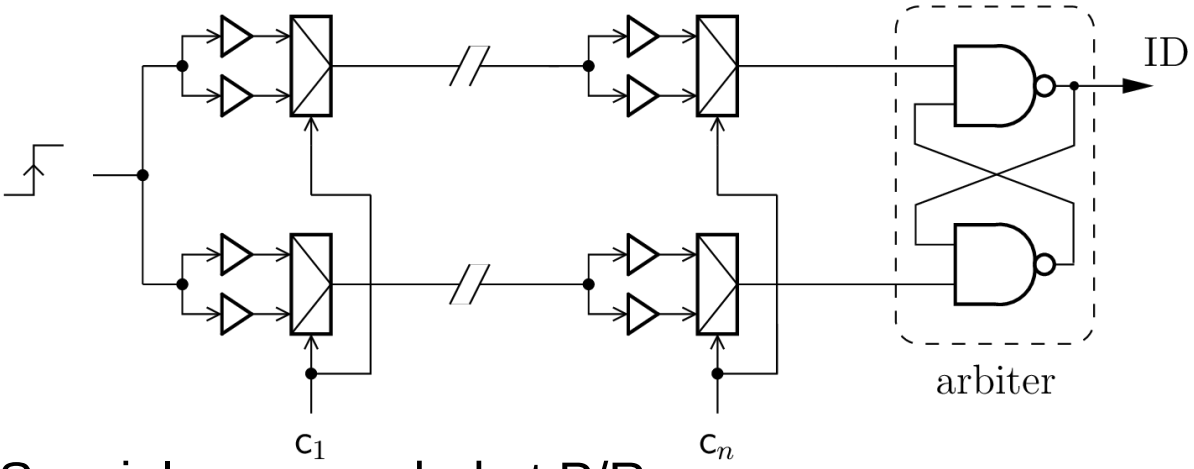


Outline

- **PUF**
  - Architectures
  - Properties
- **TRNG**
  - Architectures
  - Properties
- **Conclusions**

Delay PUF: Arbiter-PUF

- **Race between two identical pathes:**

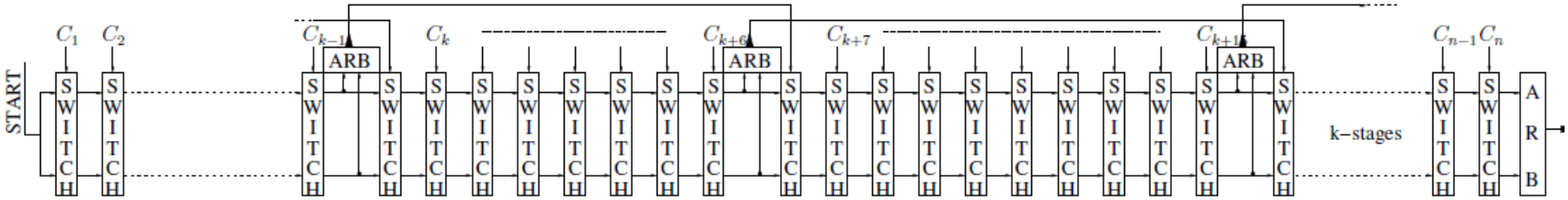


- Special care needed at P/R
- Strong but sensitive to **Modeling attacks** .
- Many derivatives to avoid Modeling attacks
  - XOR , Feed-Forward-PUF…

B. Gassend, D. Lim, D. Clarke, M. Van Dijk, and S. Devadas. Identification and authentication of integrated circuits. Concurrency and Computation: Practice & Experience, 16(11):1077–1098, 2004

Feed-Forward PUF

- **Proposed by Gassend et al. to fight ML attacks**
  - Some challenge bits are generated by intermediate arbiters

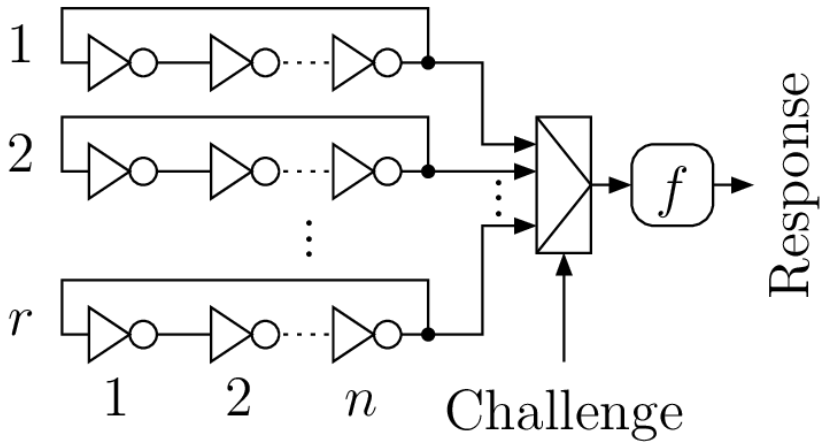


B. Gassend, D. Lim, D. Clarke, M. Van Dijk, and S. Devadas. "Identification and authentication of integrated circuits". Concurrency and Computation: Practice & Experience, 16(11):1077–1098, 2004.

Delay PUF: RO-PUF

- **Comparison between the frequency of identical two Ring Oscillators**

Ring-oscillator PUF (RO-PUF):
(r rings of n inverters)



Rationale:

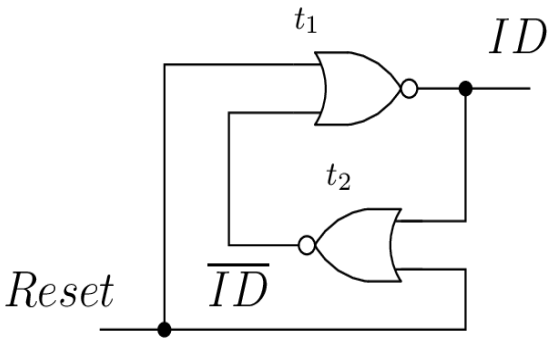
Challenge selects a pair i, j, $1 \leq i \neq j \leq r$.

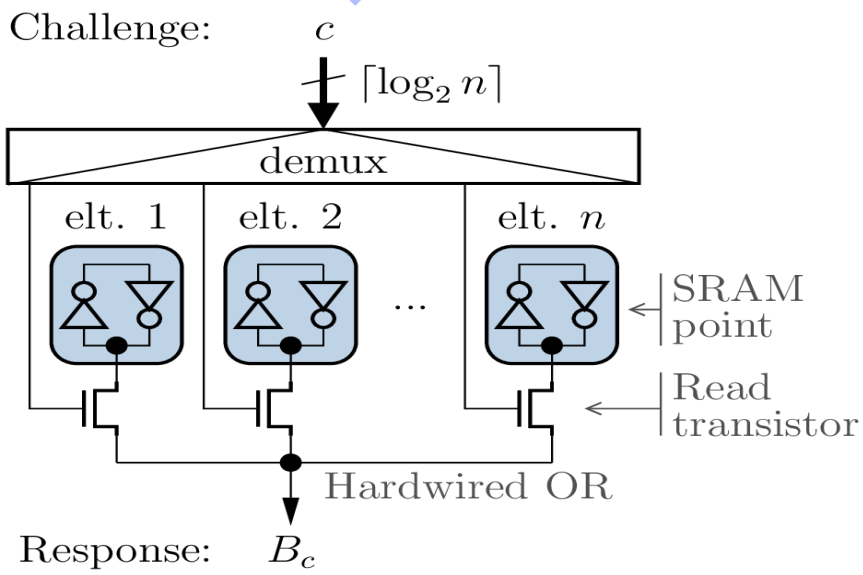
Response is 1 if RO_i rotates $f$aster than RO_j, and 0 otherwise.

G. Edward Suh and Srinivas Devadas. "Physical unclonable functions for device authentication and secret key generation". In DAC, pages 9–14, 2007.

Memory PUF

- **Imbalance between two elements of a latch or SRAM**
  - Metastable latch [1]
  - Power-on [2] : **SRAM-PUF**





1. Su Y., Holleman J., Otis B.: "A 1.6pj/bit 96% stable chip-id generating circuit using process variations". In:. ISSCC 2007. Digest of Technical Papers. IEEE, pp. 406-611, 2007

2. Guajardo, J., Kumar, S. S., Schrijen, G. J., & Tuyls, P. (2007). *FPGA intrinsic PUFs and their use for IP protection* (pp. 63-80). Springer Berlin Heidelberg.

Outline

- **PUF**
  - Architectures
  - Properties
- **TRNG**
  - Architectures
  - Properties
- **Conclusions**

Main Properties to meet

- **Entropy**
  - **Uniqueness** : Each device must have a unique fingerprint (inter-entropy
  - **Randomness:** as many bits at 0 as 1
- **Steadiness, or Reliability**
  - The PUF response should not be sensitive to:
    - Noise
    - Environmental change T°C, Vdd
    - Aging
- **Security**
  - Robustness against **physical** attacks: SCA, FIA, RE
    - No possible RE for delay PUF
  - Robustness against **modeling** attacks
    - Only for delay PUF and CRP application

Need for standard tests and/or stochastic model

Active discussion at ISO sub-committee 27: (**ISO 20897**)

ISO IEC

ISO/IEC JTC 1/SC 27/WG 3 N1233

REPLACES:

ISO/IEC JTC 1/SC 27/WG 3

Information technology - Security techniques - Security evaluation, testing and specification

Convenorship: AENOR, Spain, Vice-convenorship: JISC, Japan

DOC TYPE: working draft

TITLE: **Text for ISO/IEC 1st WD 20897 — Information technology — Security requirements and test methods for physically unclonable functions for generating non-stored security parameters**

SECURE IC

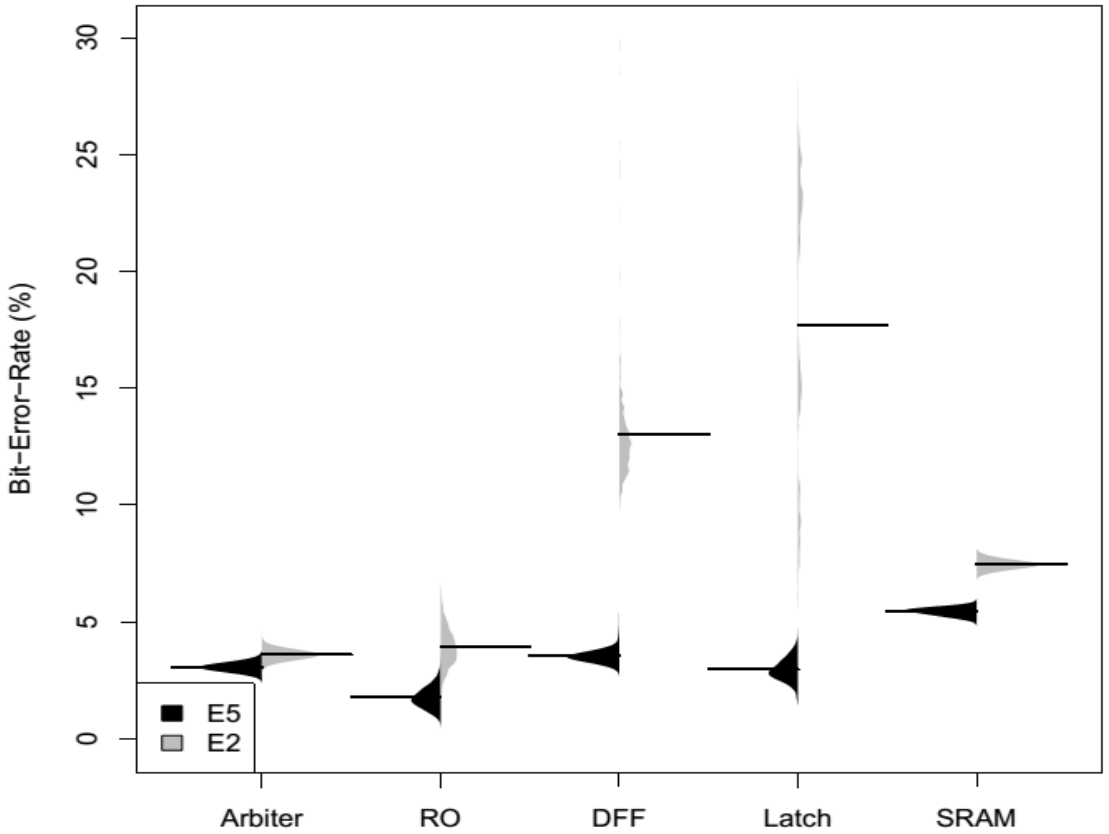
TELECOM ParisTech

Steadiness estimate

- **Results from UNIQUE* project**

* http://www.uniqueproject.eu/publications

BER = 1- steadiness

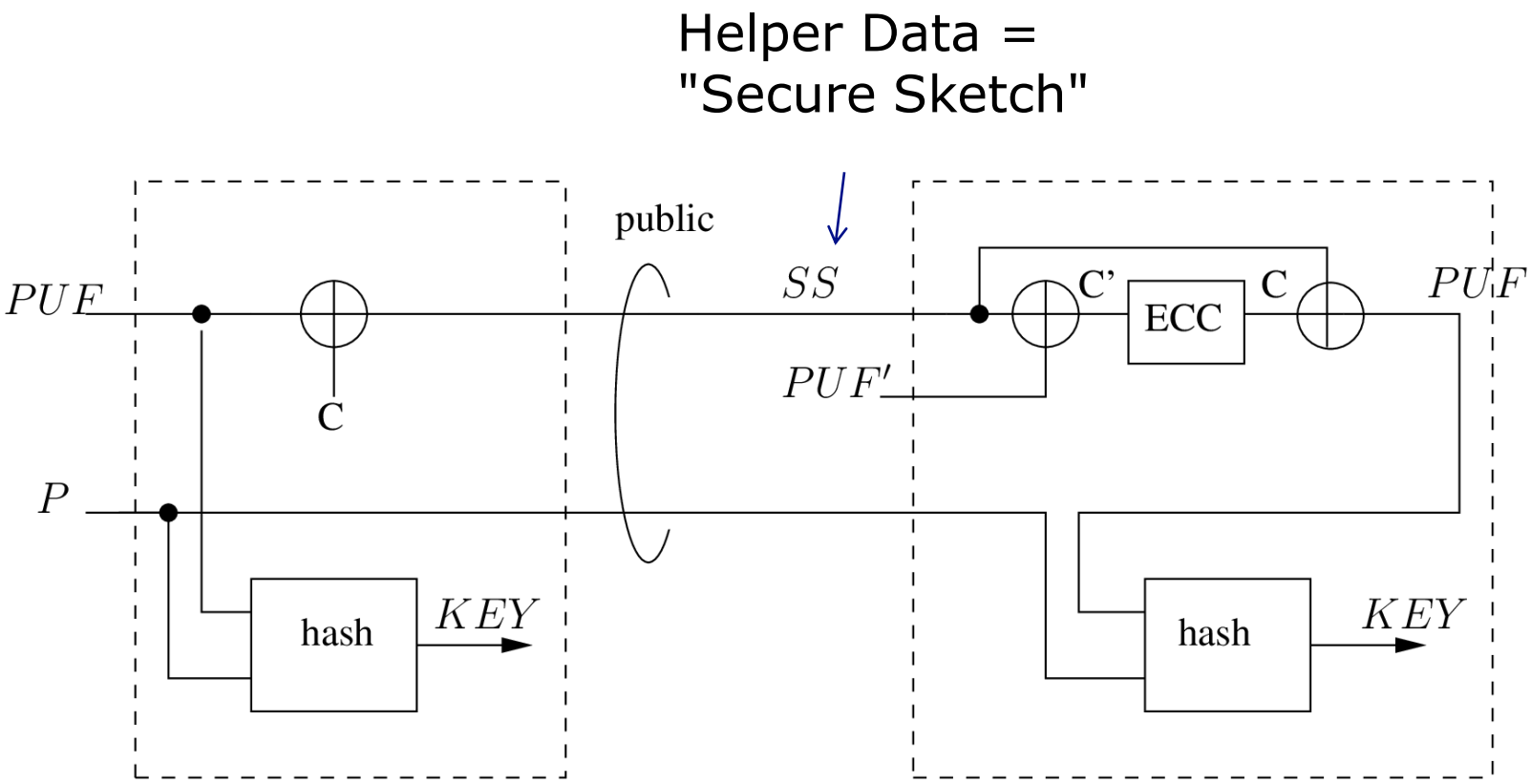


Black: 25° C
Grey: -40C

Lack of Reliability For trusted application !

Katzenbeisser, S., Kocabas, Ü., Rožić, V., Sadeghi, A.R., Verbauwhede,I.,Wachsmann, C.: PUFs: Myth, fact or busted? A security evaluation of physically unclonable functions (PUFs) cast in silicon. In: CHES 2012

Necssity to enhance the Steadiness to generate a key



Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. "Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data". SIAM J. Comput., 38(1):97–139, 2008.

Metrics Examples

L=nb bits
K=nb PUF/device
N=nb devices
T=nb tests
R or b=response

Maïti et al

Randomness

$$\text{Rand}_k = \frac{1}{L}\sum_{l=1}^{L} b_l \times 100\%.$$

Should be 50%

Steadiness

$$\text{Stead}_n = \frac{1}{T}\sum_{t=1}^{T} \frac{HD(R_n^{ref}, R_n^t)}{L} \times 100\%.$$

Should be 0%

Uniqueness

$$\text{Uniq} = \frac{2}{N(N-1)}\sum_{u=1}^{N-1}\sum_{v=u+1}^{N} \frac{HD(R_u, R_v)}{L} \times 100\%.$$

Should be 50%

Hori et al

$$p_n = \frac{1}{K.T.L}\sum_{k=1}^{K}\sum_{t=1}^{T}\sum_{l=1}^{L} b_{n,k,t,l}, \quad \text{then,} \quad \text{Rand}_n = -\log_2 \max(p_n, 1-p_n).$$

Min-entropy, Should be 100%

$$p_{n,k,l} = \frac{1}{T}\sum_{t=1}^{T} b_{n,k,l,t}, \quad \text{then,} \quad \text{Stead}_{n,k,l} = 1 + \log_2 \max(p_{n,k,l}, 1-p_{n,k,l}).$$

$$\text{Stead}_n = \frac{1}{K.L}\sum_{k=1}^{K}\sum_{l=1}^{L} S_{n,k,l}.$$

Should be 100%

$$\text{Uniq} = \frac{4}{K.L.N^2}\sum_{k=1}^{K}\sum_{l=1}^{L}\sum_{u=1}^{N-1}\sum_{v=u+1}^{N} (b_{u,k,l} \oplus b_{v,k,l}).$$

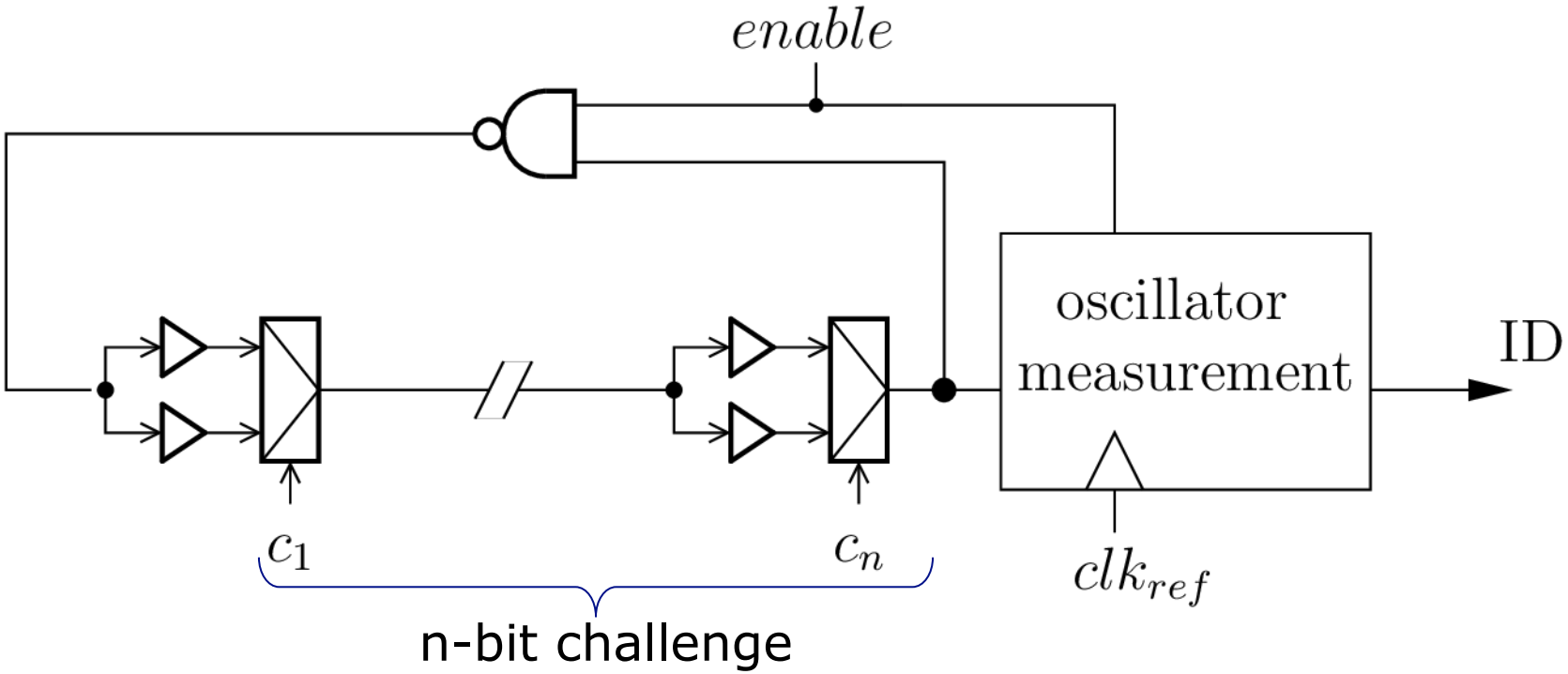
Should be 100%

Maiti, A., Casarona, J., McHale, L., & Schaumont, P. (2010, June). A large scale characterization of RO-PUF. In Hardware-Oriented Security and Trust (HOST), 2010 IEEE International Symposium on (pp. 94-99). IEEE.

Hori, Y., Yoshida, T., Katashita, T., & Satoh, A. (2010, December). Quantitative and statistical performance evaluation of arbiter physical unclonable functions on FPGAs. In Reconfigurable Computing and FPGAs (ReConFig), 2010

Entropy and steadiness metrics on the Loop-PUF

- **Comparison between the frequency of Ring Oscillator controlled by two complementary challenges**



Cherif, Z., Danger, J. L., Guilley, S., & Bossuet, L. (2012, September). An easy-to-design PUF based on a single oscillator: the loop PUF. In *Digital System Design (DSD), 2012 15th Euromicro Conference on* (pp. 156-162). IEEE.

Operating Mode

Algorithm 2.1 Operating Mode with 2 complementary challenges

Input: Challenge C (a word of n bits)
Output: Response Δ (a signed integer whose sign is B)

1: Set challenge C
2: Measure $d_C \leftarrow \lfloor L \sum_{i=1}^{n} d(c_i) \rfloor$
3: Set challenge $\neg C$
4: Measure $d_{\neg C} \leftarrow \lfloor L \sum_{i=1}^{n} d(\neg c_i) \rfloor$
5: Compute $\Delta = d_C - d_{\neg C}$
6: Return Δ with $B = \text{sign}(\Delta) \in \{\pm 1\}$

The information delivered for each challenge pair is Δ

Δ is a high precision integer

Sign(Δ) = B = **Identifier** Module(Δ) = **Reliability**

Entropy of a n-delay Loop-PUF

- **Entropy = n bits if :**
  - A set of n n-bit challenges is chosen from **Hadamard codewords**
  - The delay elements are i.i.d

Proof in (1): a Hamming distance of n/2 creates no dependency between the Δ

For instance for $n = 12$, the n by n Hadamard Matrix is:

$$\mathcal{C}_{12} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 \\ 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 \\ 1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & 1 & -1 \\ 1 & -1 & 1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 \end{pmatrix}$$

(1) Rioul, O., Solé, P., Guilley, S., & Danger, J. L. (2016, July). On the Entropy of Physically Unclonable Functions. In *Information Theory (ISIT), 2016 IEEE International Symposium on* (pp. 2928-2932). IEEE.

LPUF reliability expressed in BER

Average BER

$$\widehat{BER} = \int_{-\infty}^{+\infty} p(\Delta) \cdot BER(\Delta) \cdot d\Delta$$

$$\widehat{BER} = \frac{1}{2} - \frac{1}{\pi} \arctan(\sqrt{SNR})$$

With $SNR = \frac{\Sigma^2}{\sigma^2}$

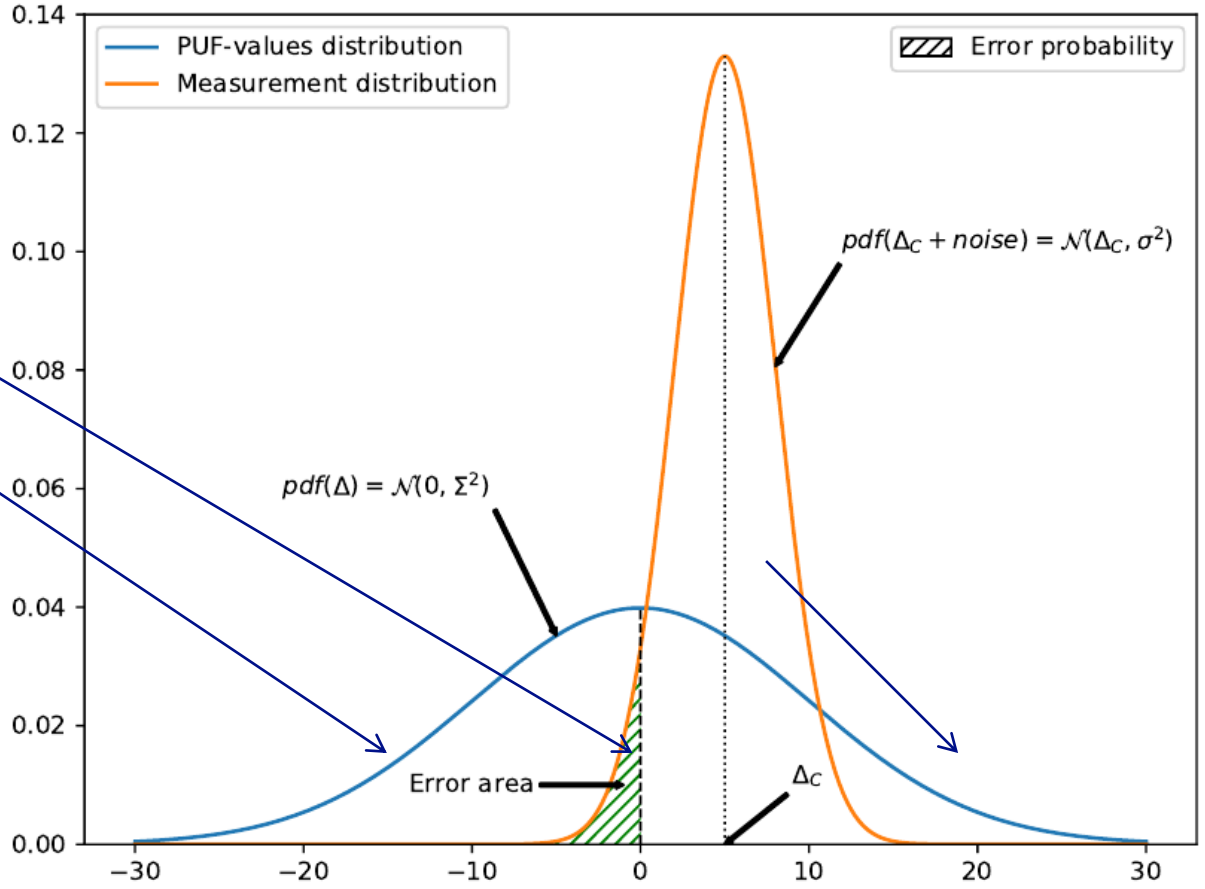
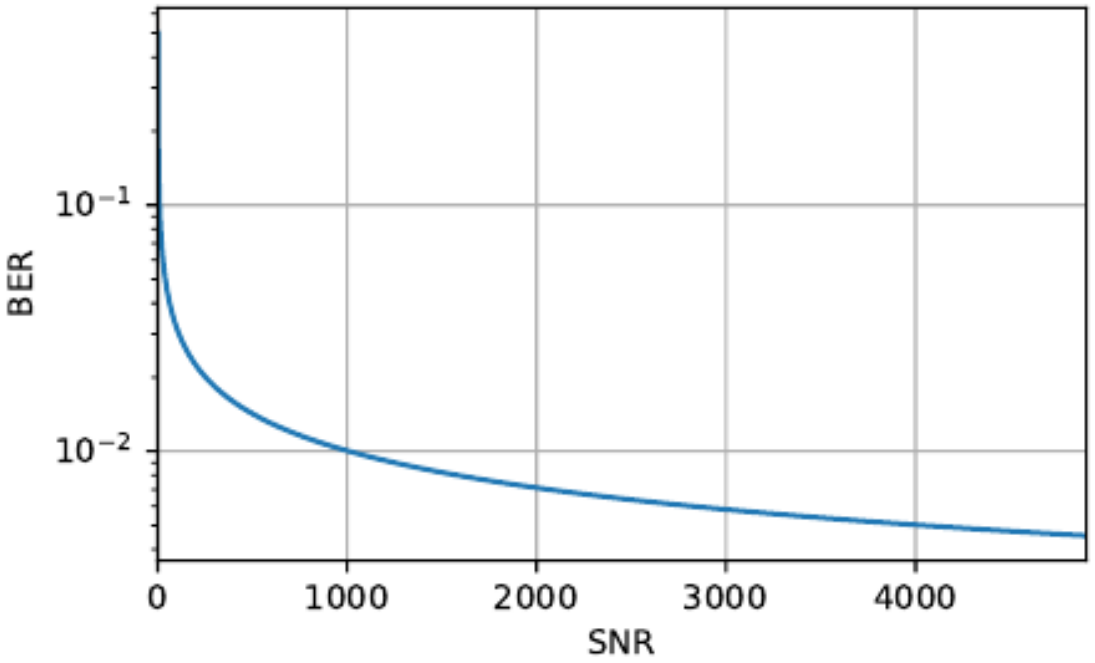


Figure 5: pdf of Δ and noise for a given challenge C

LPUF reliability



$$\widehat{BER} = \frac{1}{2} - \frac{1}{\pi}\arctan(\sqrt{SNR})$$

Figure 6: Expected BER as a function of the SNR

The Reliability is not enough as BER>10^{-3} even with high SNR
=> Needs of Secure Sketch: Error Correcting codes and Helper data

Reliability enhancement by delay knowledge

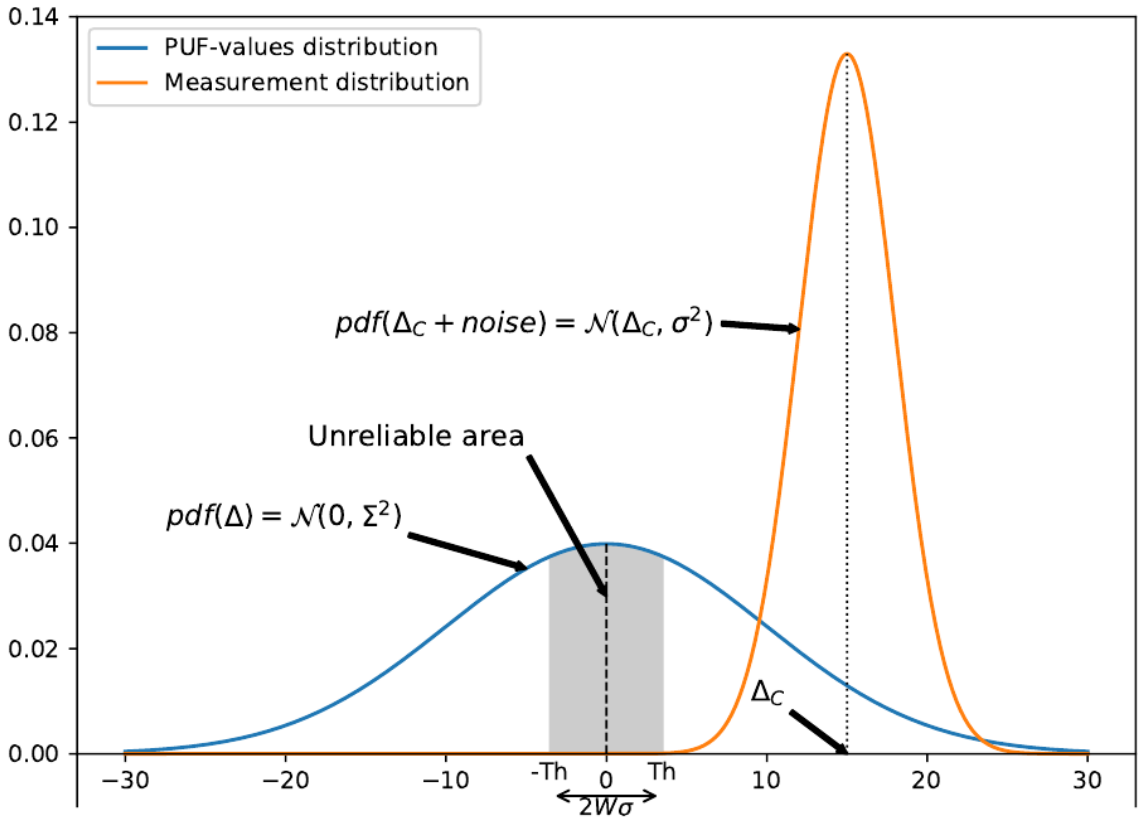


Figure 7: Unreliable area vs Distributions of Δ and noise

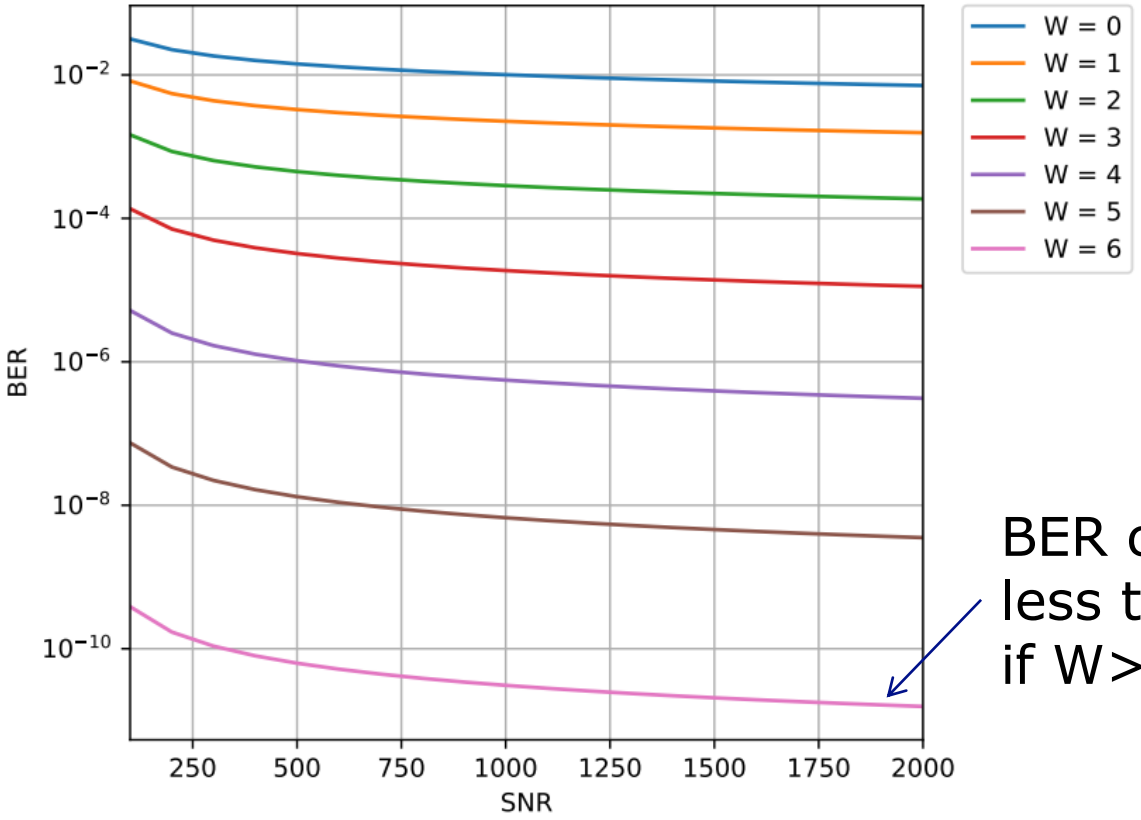
Bit **unreliable** ⇔ |delay| < Th Th= Wσ

The bits in the unreliable area are **discarded**

The helper data indicates the unreliable bits, but gives **no information** on the bit value

New BER with filtered bits

$$\widehat{BER_{filt}} = \frac{2}{\mathrm{erfc}(\frac{W}{\sqrt{2}\sqrt{SNR}})}(T(W, \frac{1}{\sqrt{SNR}}) + \frac{1}{2}\,\mathrm{erf}(\frac{W}{\sqrt{2}\cdot\sqrt{SNR}})(\mathrm{erf}(\frac{W}{\sqrt{2}}) - 1))$$

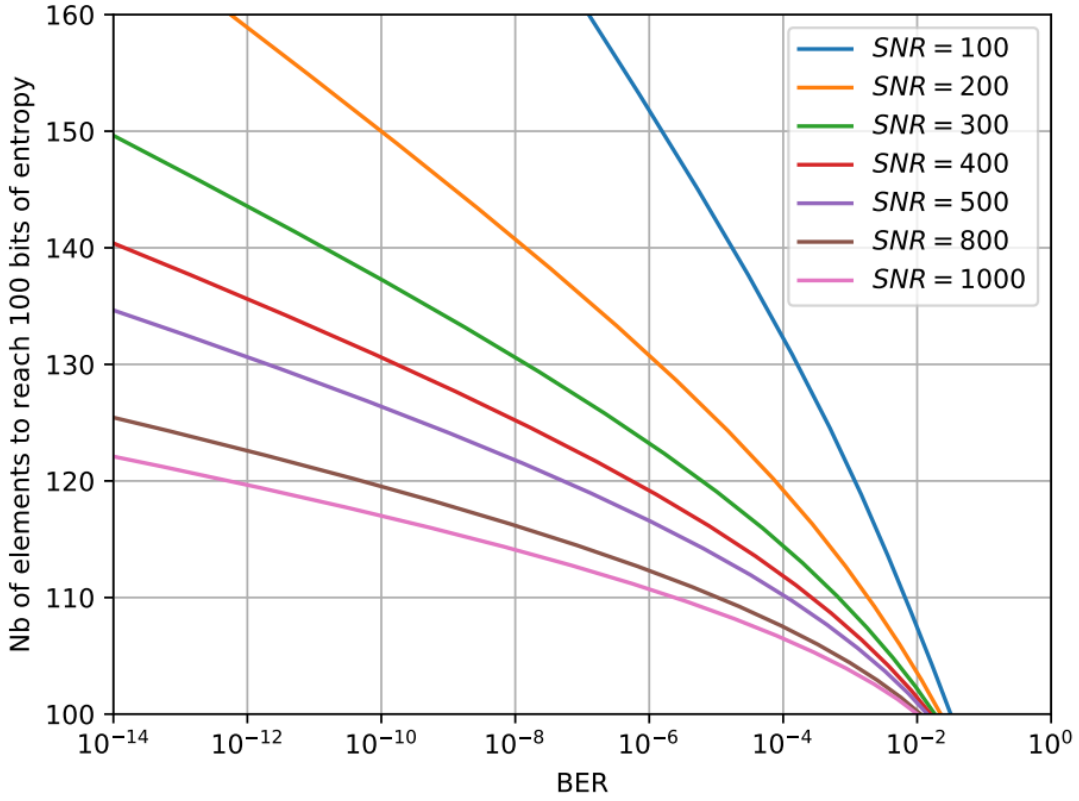


BER can be less than 10^{-10} if W>6

Entropy after bit filtering

Number of delay elements to reach n bits of entropy with Hadamard codes

$$n' = \frac{n}{1 - \mathbb{P}(Bit\ unreliable)} = \frac{n}{\mathrm{erfc}(\frac{W}{\sqrt{2SNR}})}$$



PUF Security

- **Reverse Engineering Attack**
  - Almost impossible: this is the strength of PUF
- **Brute force**
  - To save every Challenge/Response (CRP)
  - Impossible in a reasonnable time
- **Replay**
  - Sniffing CRPs and play them back
  - Can be countered at protocole level (authentication)
- **Mathematical**
  - Reconstruct the PUF model: Modeling Attack
- **Physical attack**
  - Side-channel
  - Faults

Main threats

No metrics , just countermeasuures

Modeling Attacks

- **Based on Machine Learning algorithms**
  - Take advantage of relationships between the challenge / response
  - Very powerful to attack delay-based PUF
- **Countermeasures**
  - Hide the Challenge or Response
    - Cipher the challenge or the response
  - Do not use a CRP protocole for authentication
  - For example:
    - the server sends a Nonce
    - PUF sends Cipher(K_{PUF},Nonce) to the server
    - the server checks it is OK

* Ulrich Rührmair, Frank Sehnke, Jan Sölter, Gideon Dror, Srinivas Devadas, and Jürgen Schmidhuber. "Modeling attacks on physical unclonable functions". In Proceedings of the 17th ACM

PUF side-channel attack

- **Observation of raw oscillating frequency**
  - Applies to RO-PUF and Loop PUF
  - CM : Use sequential measurement (as the Loop PUF) with a random variable to indicate the order.
- **Attack on the Fuzzy extractor [2]**
  - Simple Power Analysis has been carried out on a FE
  - Template attacks have been implemented on ECC
  - CM: those used for cryptographic blocs

1. Merli, D., Schuster, D., Stumpf, F., & Sigl, G. (2011, October). Semi-invasive EM attack on FPGA RO PUFs and countermeasures. In *Proceedings of the Workshop on Embedded Systems Security* (p. 2). ACM.
2. Karakoyunlu, D.; Sunar, B.; , "Differential template attacks on PUF enabled cryptographic devices," *Information Forensics and Security (WIFS), 2010 IEEE International Workshop on* , vol., no., pp.1-6, 12-15 Dec. 2010

Enhanced SCA attack

- **Combination with ML**
  - Use of noise distribution of the arbiter PUF [1]
  - Use unsupervised ML- techniques [2]
    - SCA is performed first
    - The ML technique proposes a model for classification (like for instance the "k-means" algorithm).

1. Delvaux, J., & Verbauwhede, I. (2013, June). Side channel modeling attacks on 65nm arbiter PUFs exploiting CMOS device noise. In *Hardware-Oriented Security and Trust (HOST), 2013 IEEE International Symposium on* (pp. 137-142). IEEE.
2. Becker, G. T., & Kumar, R. (2014). Active and Passive Side-Channel Attacks on Delay Based PUF Designs. *IACR Cryptology ePrint Archive*, *2014*, 287.

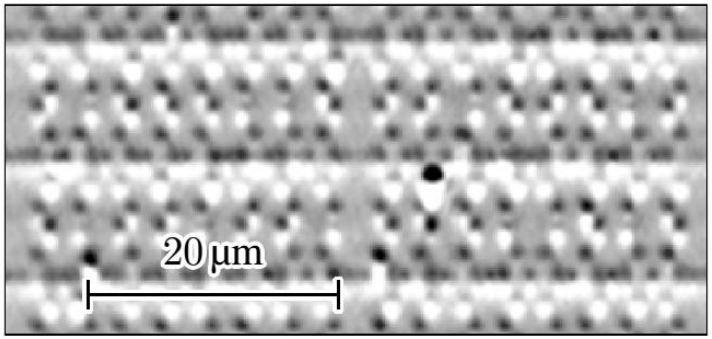
PUF fault injection attack

- **Potential attacks**
  - Pulse attack (laser, EMI,…)
    - The PUF output is forced
  - Harmonics attack*
    - The PUF frequency can be locked on external EM carrier injection (Fault injection attack)
- **Countermeasures**
  - Detection
    - Measure online the entropy of the PUF response

* Poucheret, F., Tobich, K., Lisarty, M., Chusseau, L., Robisson, B., & Maurine, P. (2011, September). Local and direct em injection of power into cmos integrated circuits. In *Fault Diagnosis and Tolerance in Cryptography (FDTC), 2011 Workshop on* (pp. 100-104). IEEE.

PUF invasive attack

- **Read out SRAM PUF**
  - Laser stimulation techniques exploiting the Seebeck effect
    - the off-transistor becomes to conduct under laser shot
    - Provides a current increase
  - Attack performed on AVR microcontrollers



SRAM content read out

Nedospasov, D., Seifert, J. P., Helfmeier, C., & Boit, C. (2013, August). Invasive PUF analysis. In *Fault Diagnosis and Tolerance in Cryptography (FDTC), 2013 Workshop on* (pp. 30-38). IEEE.

Outline

- **PUF**
  - Architectures
  - Properties
- **TRNG**
  - Architectures
  - Properties
- **Conclusions**

RNG

- **Pseudo Random number (PRNG)**
  - Deterministic
  - Periodic
  - Adapted for stream ciphering
- **True Random Number generator (TRNG)**
  - Unpredictable
  - Use physical sources
- **Hybrid Random Number generator (HRNG)**
  - TRNG+PRNG: TRNG used as a seed of PRNG

RNG Applications

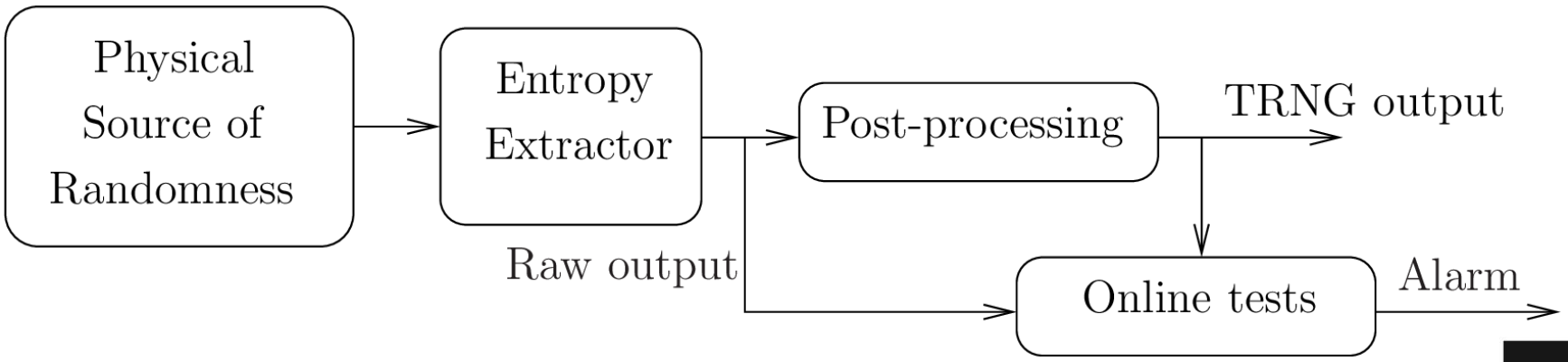
- **Cryptography**
  - Key generation
    - Symmetric/asymetric crypto
    - Stream ciphering (if deterministic RNG)
  - Initialization vector
    - For cryptographic operating modes
  - Authentication
    - Challenges
    - Nonce
  - Protection
    - Masks to thwart Side-Channel Attacks
- **System validation**
  - Monte-Carlo simulation
  - Statistical analysis
  - Channel emulation
- **Games, gambling**

Outline

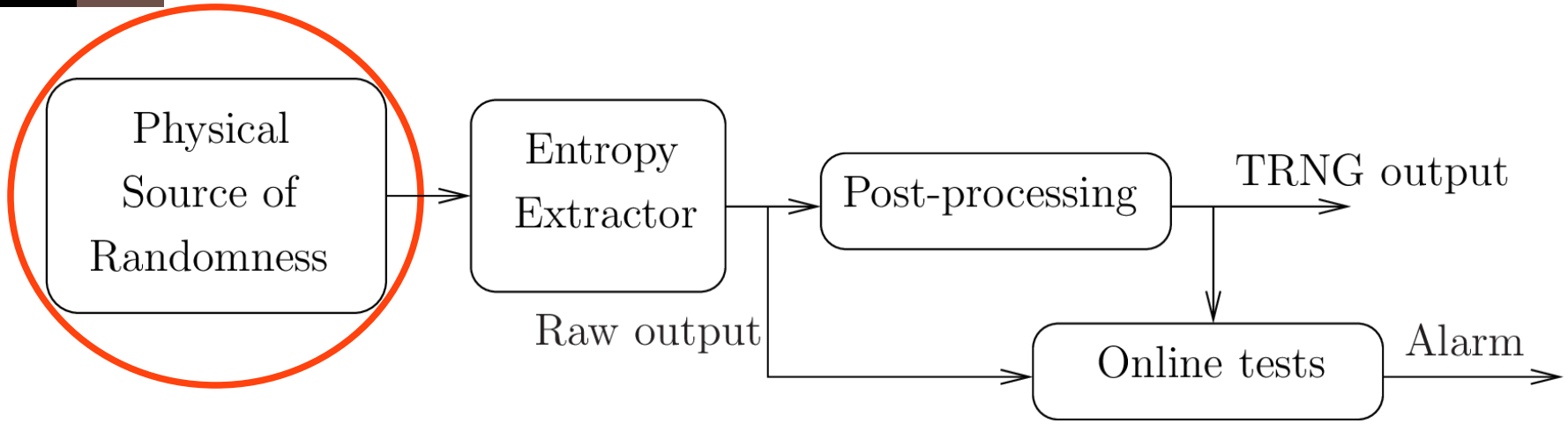
- **PUF**
  - Architectures
  - Properties
- **TRNG**
  - Architectures
  - Properties
- **Conclusions**

TRNG Architecture

- **2 main blocks**
  - Entropy Source
  - Entropy extractor
- **2 optional blocks**
  - Postprocessing
  - Embedded Tests



Source of randomness



Two types of randomness source:

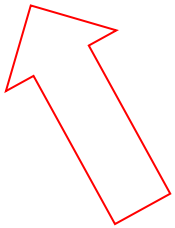
- Non Physical noise
  - Keyboard and mouse activity, Processor load, network data rate,…
- Physical noise
  - Noise in electronics circuits
  - Others: desintegration of radioactive atomic kernel,…

Noise in electronics circuit

- **Noise = Sum of different phenomenon:**
  - Thermal noise
  - 1/F noise
  - Shot noise
  - Popcorn noise

Local noise

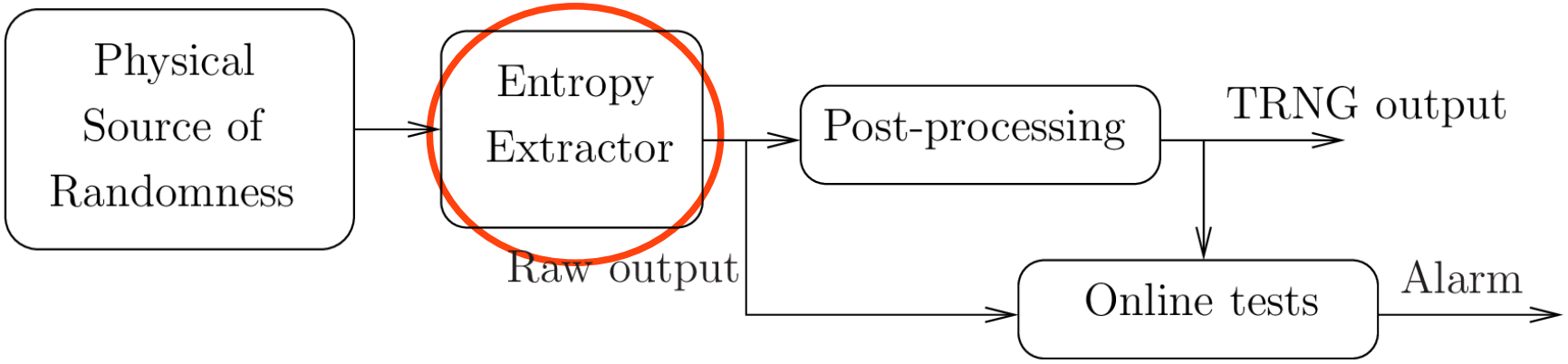
  - Crosstalk
  - Interference



Can be source of attacks !

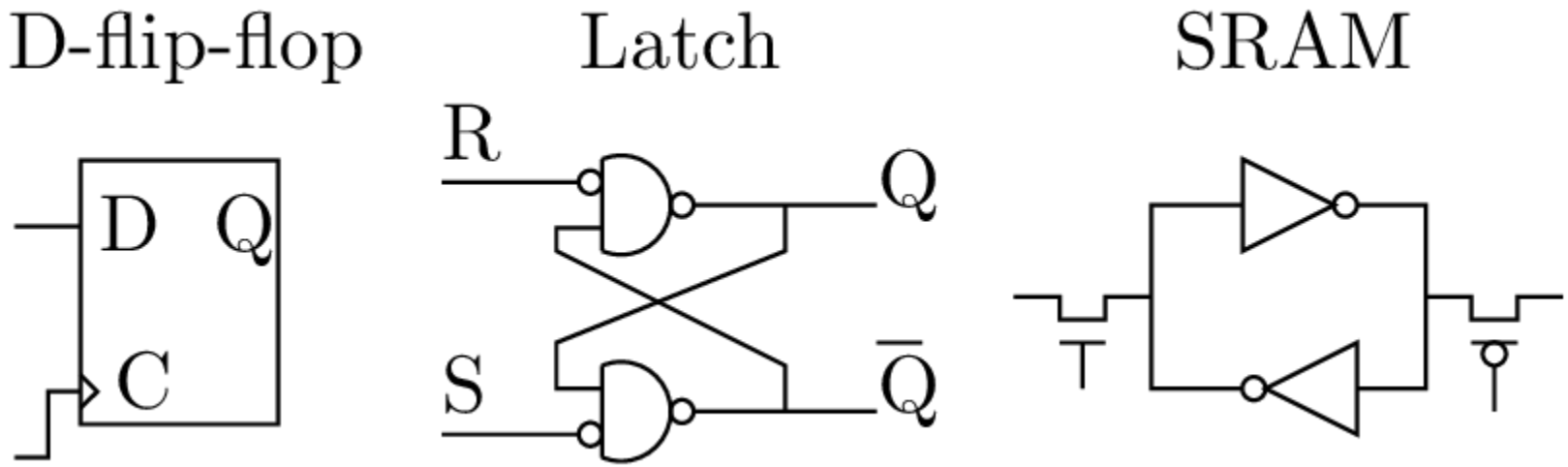
Randomness extraction

- **To sample the noise at source level**
  - Use of a sample/hold circuit, or memory element



Randomness extraction

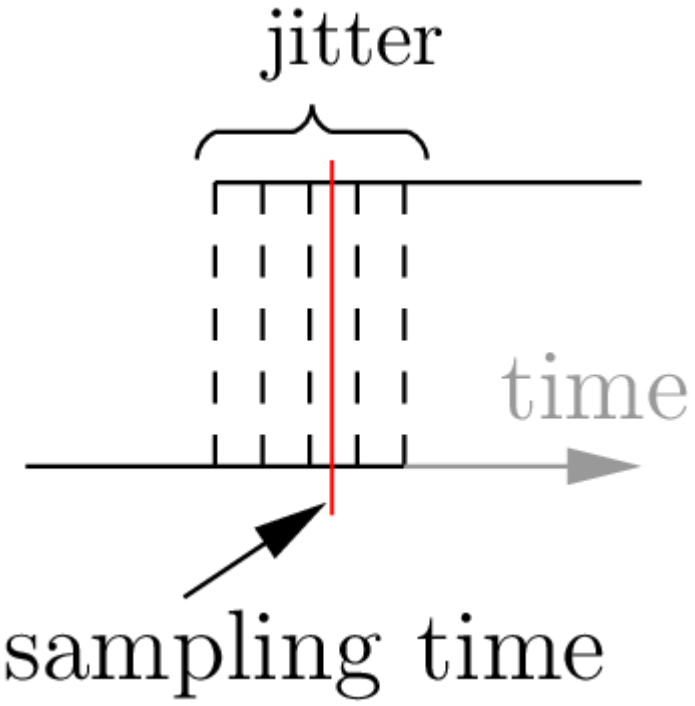
- **Memory elements**



- **Noise Axis**
  - Horizontal: **phase** noise (frequency) or **jitter** (time)
  - Vertical: **amplitude** noise

Phase noise

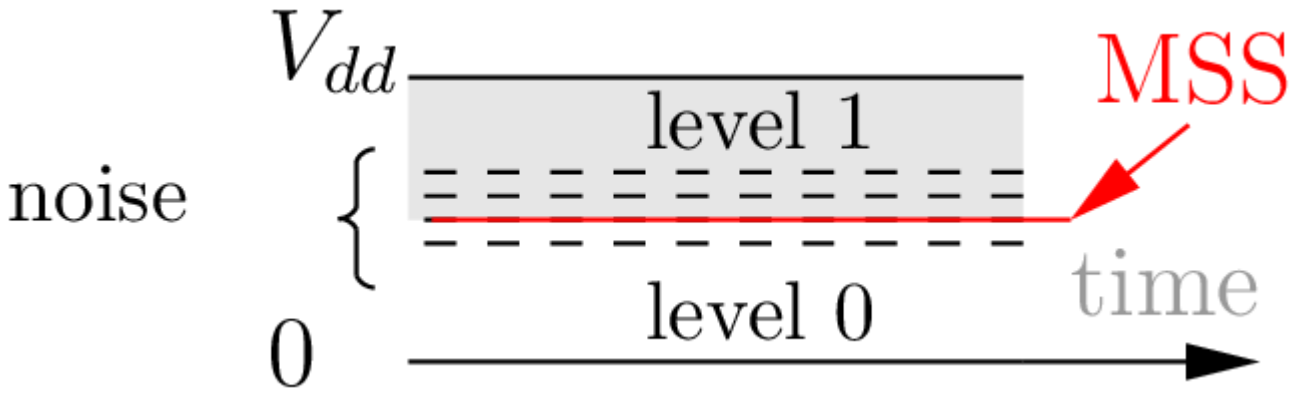
- **Combination of random jitter and deterministic jitter (unwanted)**



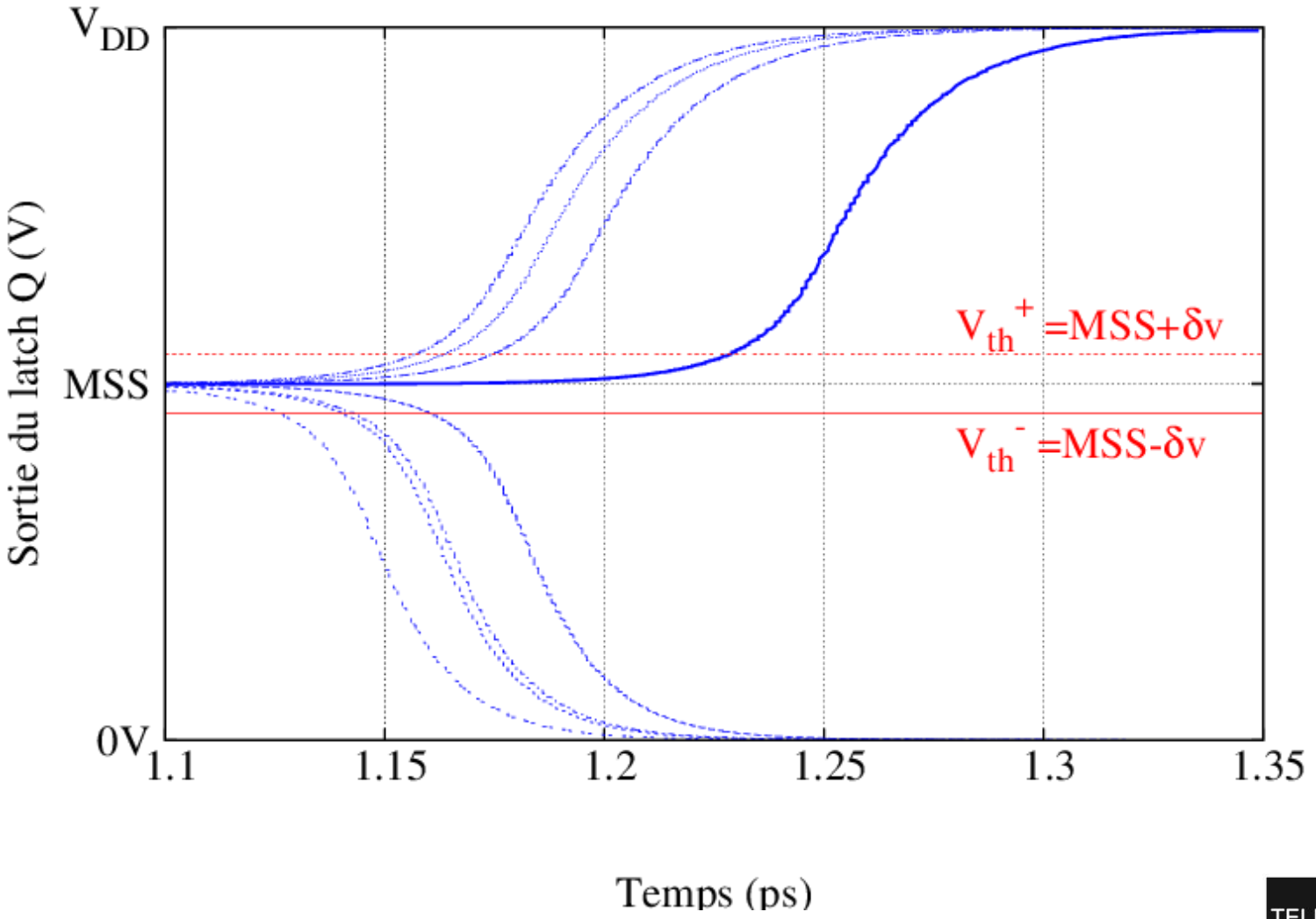
Amplitude noise

Also called vertical noise

The signal is at the boundary of 0 and 1 level, around the "**metastable** state" MSS (~Vdd/2)



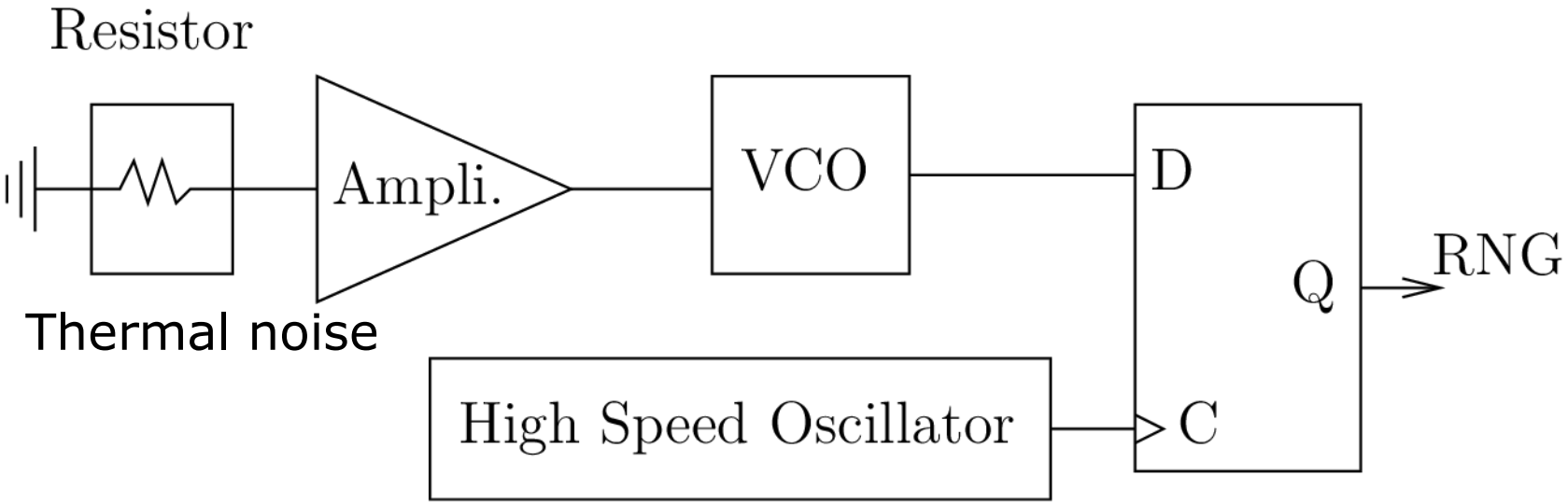
Metastability phenomenon



TRNG types

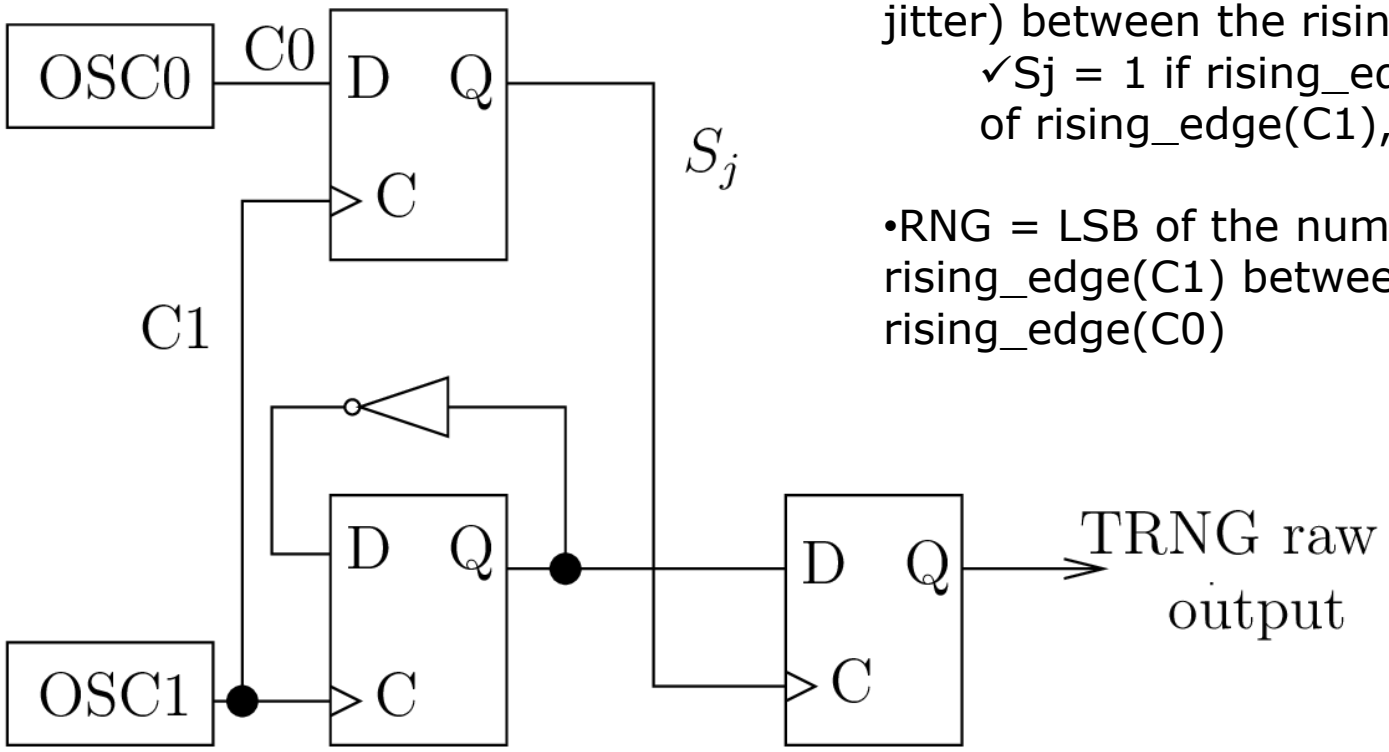
- **Jitter-based TRNG**
  - One jittery clock samples another jittery clock
- **Metastability-based TRNG**
  - The bi-stable is placed in the MSS state, its output depends on the vertical noise
- **Mixed TRNG**
  - Exploits both phenomemon

Jitter-based TRNG: Intel first generation



Benjamin Jun and Paul Kocher. The Intel Random Number Generator, 1999. http://www.cryptography.com/intelRNG.pdf.

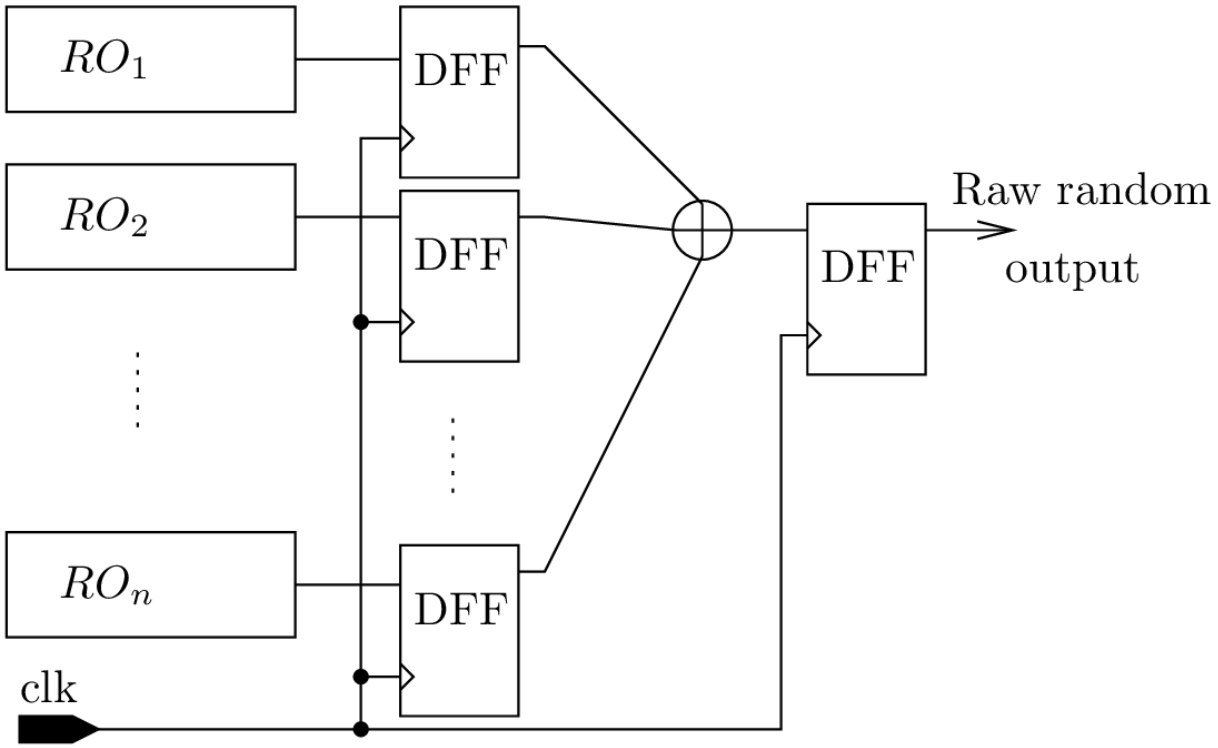
Two-Ring oscillator TRNG Kohlbrenner et al.



- There is a race (which depends on the jitter) between the rising edges:
  - ✓Sj = 1 if rising_edge(C0) is ahead of rising_edge(C1), else 0
- RNG = LSB of the number of rising_edge(C1) between two rising_edge(C0)

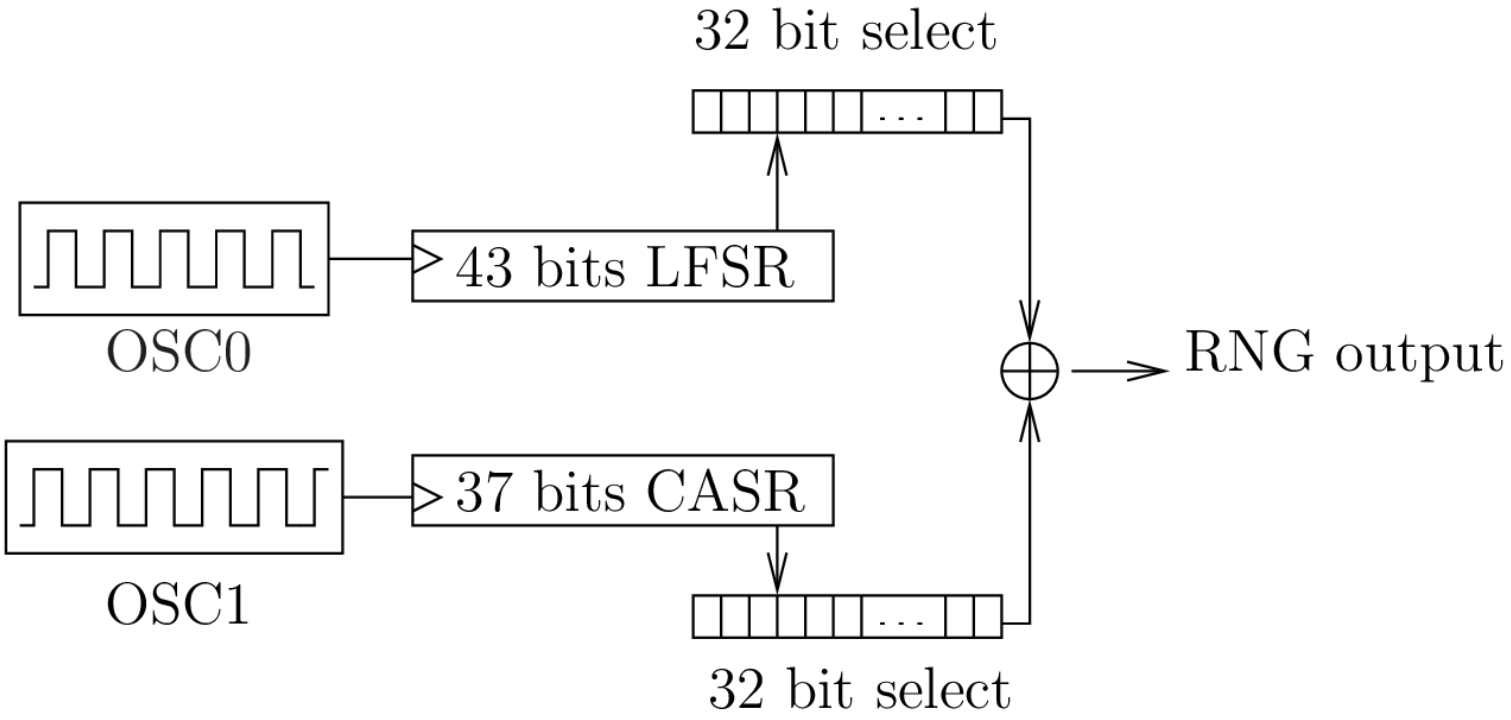
Paul Kohlbrenner and Kris Gaj. An embedded true random number generator for FPGAs. In Proceedings of the 2004 ACM/SIGDA 12th international symposium on Field programmable gate arrays, 2004.

Multiple ROs TRNG
Wold et al.



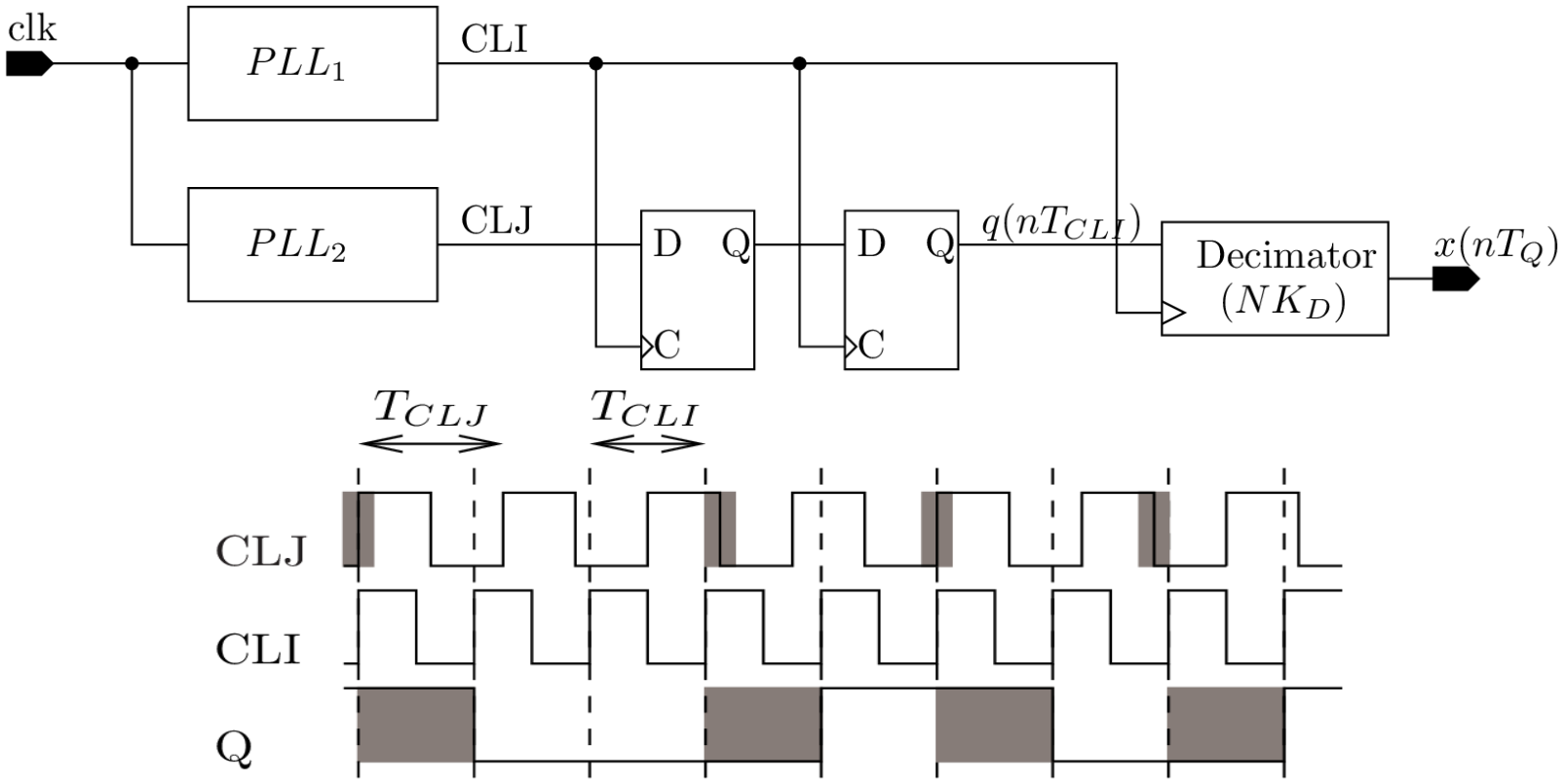
Knut Wold and Chik How Tan. Analysis and enhancement of random number generator in fpga based on oscillator rings. In ReConFig, pages 385–390, 2008.

Two-Ring oscillator with LFSR Tcacik



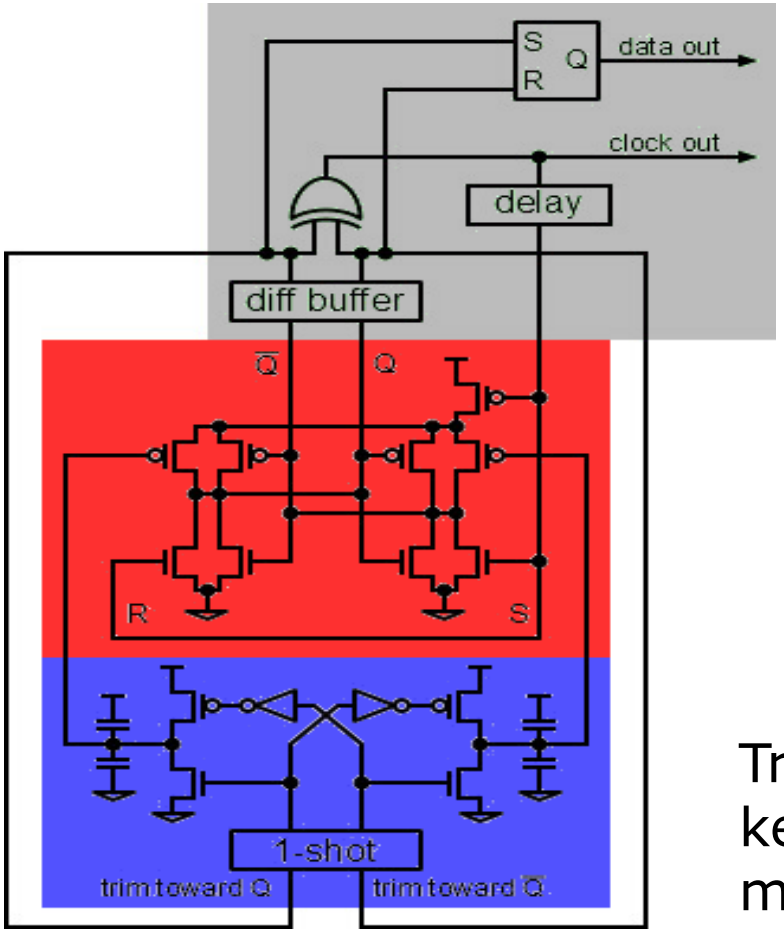
Thomas Tkacik. A hardware random number generator. In BurtonS. Kaliski, ÇetinK. Koç, and Christof Paar, editors, Cryptographic Hardware and Embedded Systems - CHES 2002, volume 2523 of Lecture Notes in Computer Science, pages 450–453. Springer Berlin Heidelberg, 2002.

Jitter from PLL-based TRNG



Viktor Fischer, Milos Drutarovský, Martin Simka, and Nathalie Bochard. High performance true random number generator in altera stratix fplds. In Field Programmable Logic and Application, volume 3203 of Lecture Notes in Computer Science, pages 555–564. Springer 2004.

Metastability_based TRNG: Intel Ivy bridge



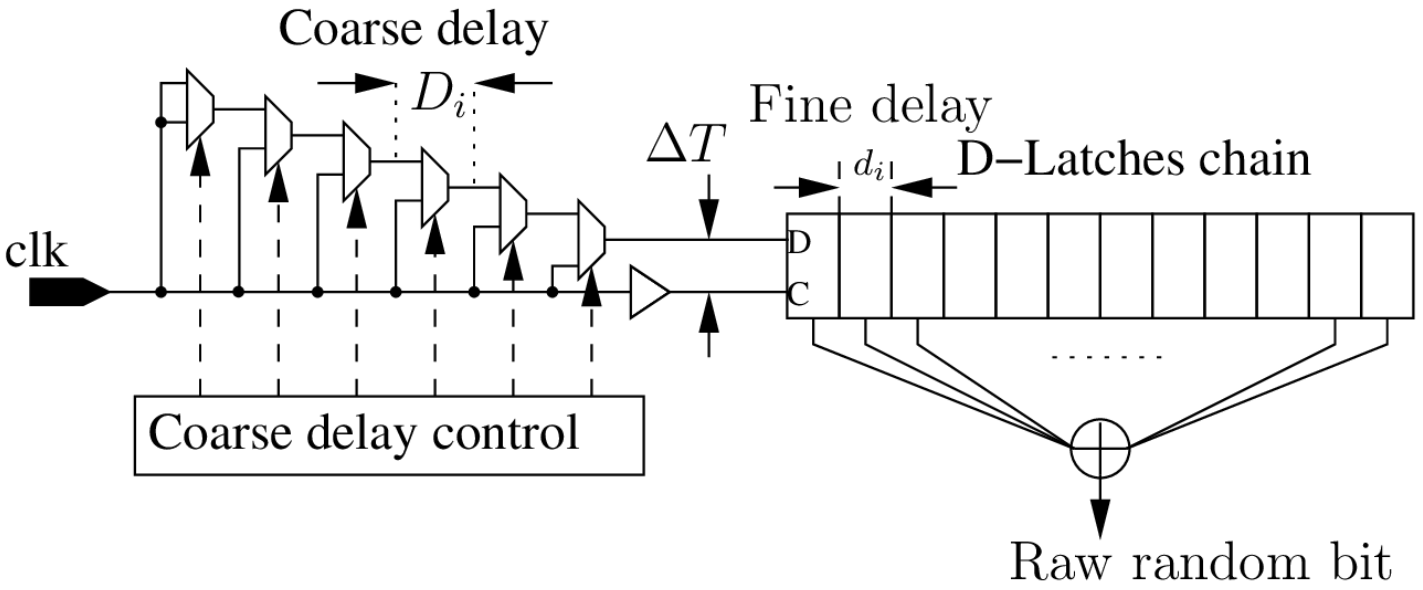
Bistable cell

Trimming circuit to keep the metastable state

Mike Hamburg, Paul Kocher, and Mark E. Marson. Analysis of intel's ivy bridge digital random number generator, March, 12 2012.

Mixed TRNG: Open Loop

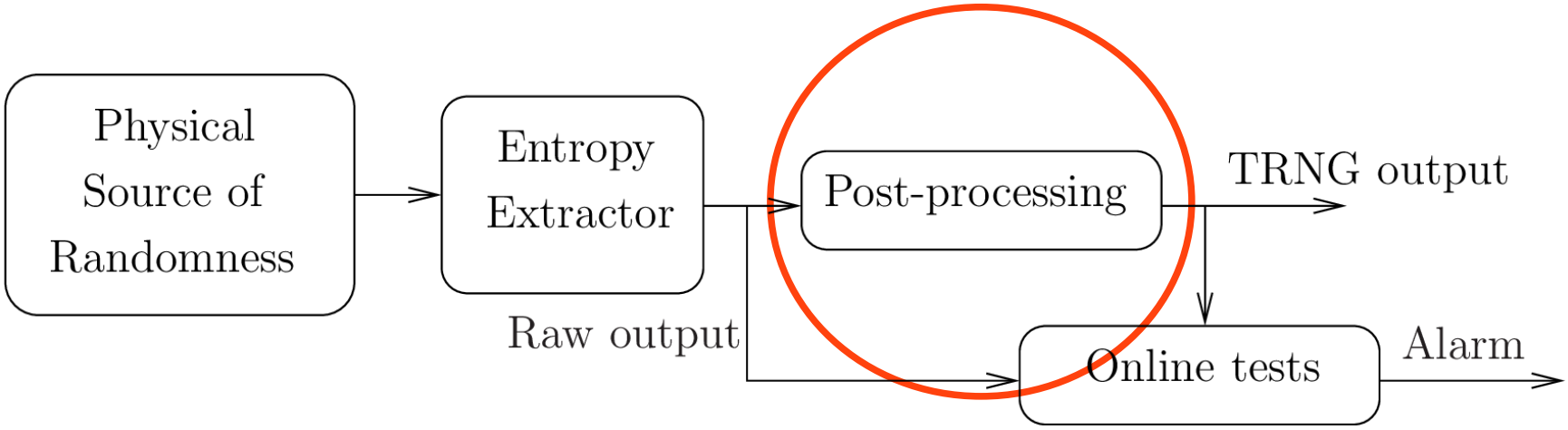
- **A chain of latches with D = C**



Danger, J. L., Guilley, S., & Hoogvorst, P. (2009). High speed true random number generator based on open loop structures in FPGAs. *Microelectronics journal*, *40*(11), 1650-1656.

Post processing

- **Mandatory process:**
  - The source of entropy is biased by the environment
  - The extraction block is not efficient enough
  - The samples are correlated



Post processing for better entropy/bit

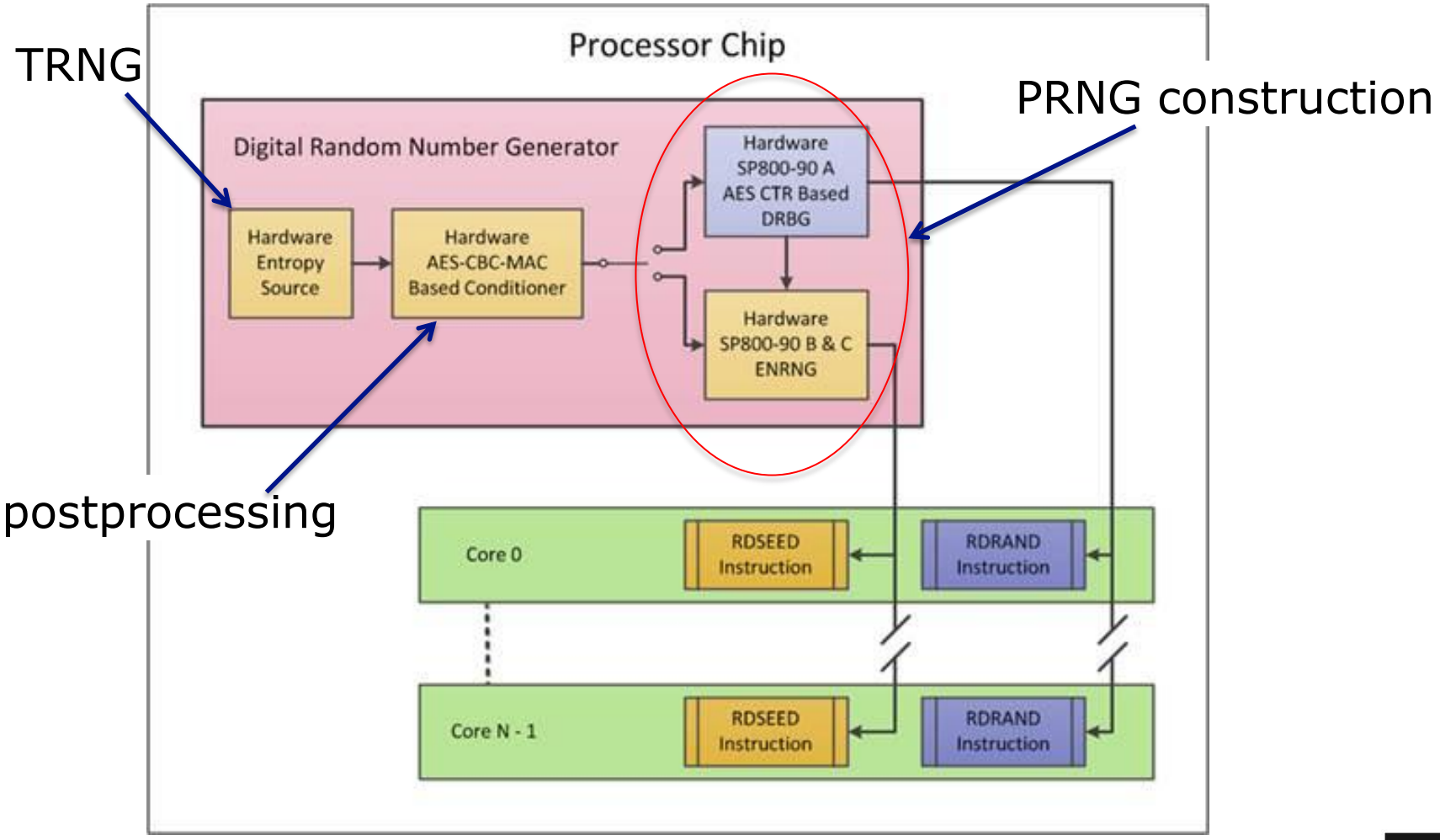
- **Goal: To increase the entropy/bit**
  - by reducing the Bit rate
  - Or data compression
- **Classical methods:**
  - XOR corrector
    - N bits are XORed to get one output bit
    - Constant Bit rate reduction (N)
    - XOR construction proposed by M. Dichtl*
  - Von Neumann corrector
    - 2 identical bits: freeze
    - 2 different bits: 1 if "10" , 0 if "01"
    - Variable Bit rate reduction (at least 4)

* Dichtl, M. (2007, January). Bad and good ways of post-processing biased physical random numbers. In Fast Software Encryption (pp. 137-152). Springer Berlin Heidelberg

Cryptographic post processing

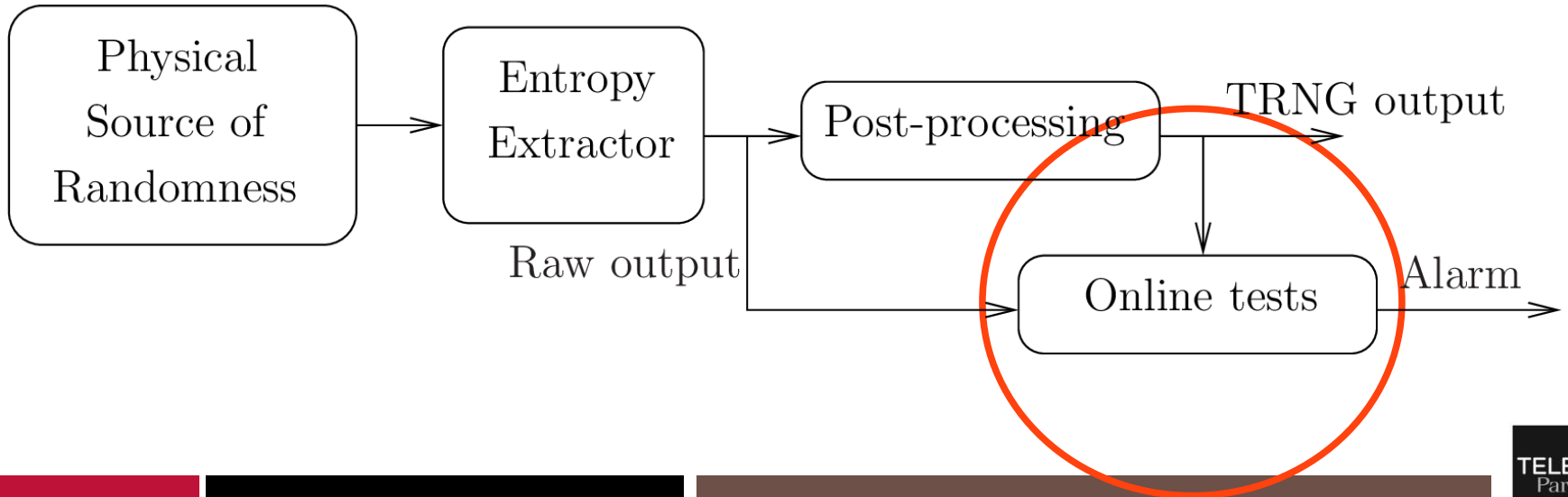
- **Goal: To increase unpredictibility**
  - Exploits non-linearity properties
  - The entropy is the same
- **Hash function**
  - Unpredictability ensures by One-way function
- **Symmetric cryptography cipher**
  - Could be the same as the crypto block.

Intel TRNG postprocessing



On-Line tests

- **Embedded tests = health tests**
  - To ensure dynamically that the output bits have good randomness property :
    - Under environmental variations
    - Under attacks



Embedded test

- **Custom sanity check**

Intel's Ivy bridge

Bit pattern	Allowable number of occurrences per 256-bit sample
1	$109 < n < 165$
01	$46 < n < 84$
010	$8 < n < 58$
0110	$2 < n < 35$
101	$8 < n < 58$
1001	$2 < n < 35$

- **Use of standardized statistical tests**
  - Can be costly => partial tests
- **2nd order effect:**
  - the noise generated by the test impacts the TRNG which passes the test more easily

Outline

- **PUF**
  - Architectures
  - Properties
- **TRNG**
  - Architectures
  - Properties
- **Conclusions**

Properties

- **In theory:**
  - Entropy
    - Shannon
    - Min-Entropy
  - Unpredictability: Conditionnal Entropy
    - Stochastic process
      - H(state_{i} | state_{i-1}) ?
- **In practice**
  - Randomness Assesment Methods
    - Statistical tests
    - Stochastic model
    - On-line tests
  - Robustness against physical attacks

$$H(x) = -\sum p(x)\log_2(p(x))$$

x is a bit vector

$$H(x) = -\log_2 \max(p_i(x))$$

Statistical tests

- **Off-line tests**
  - FIPS140-2
  - NIST SP800-22
  - DIEHARD
  - AIS31
- **On-line tests**
  - Custom tests (as Intel)
  - Partial statistical tests

FIPS-140-2, first version

- **4 tests applied on 20000-bit sequences**
- **Can be embedded in the circuit:**
  - Monobit (pr(bit=0))
  - Poker : uniform distribution of 4-bit groups
  - Runs : check the number of run sequences of ‘0’ and ‘1’ of length between 1 and 6
  - Long runs : no run of ‘0’ or ‘1’ with a length equal or greater than 26 should occur

NIST FIPS (Federal Information Processing Standards). Security Requirements for Cryptographic Modules publication 140-2, May 25 2001.
http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf.

NIST test suite SP800-22

- List of 16 statistical tests, some requiring 1Gbit for all the tests
- Applies to random number generators for cryptographic applications
  - Usually unfit for testing raw physical entropy sources
  - Fit for testing post-processed or “ready-to-use” outputs

Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangel, David Banks, Alan Heckert, James Dray, and San Vo. A statistical test suite for the validation of random number generators and pseudo random number generators for cryptographic applications, april 2010.

NIST test suite SP800-22

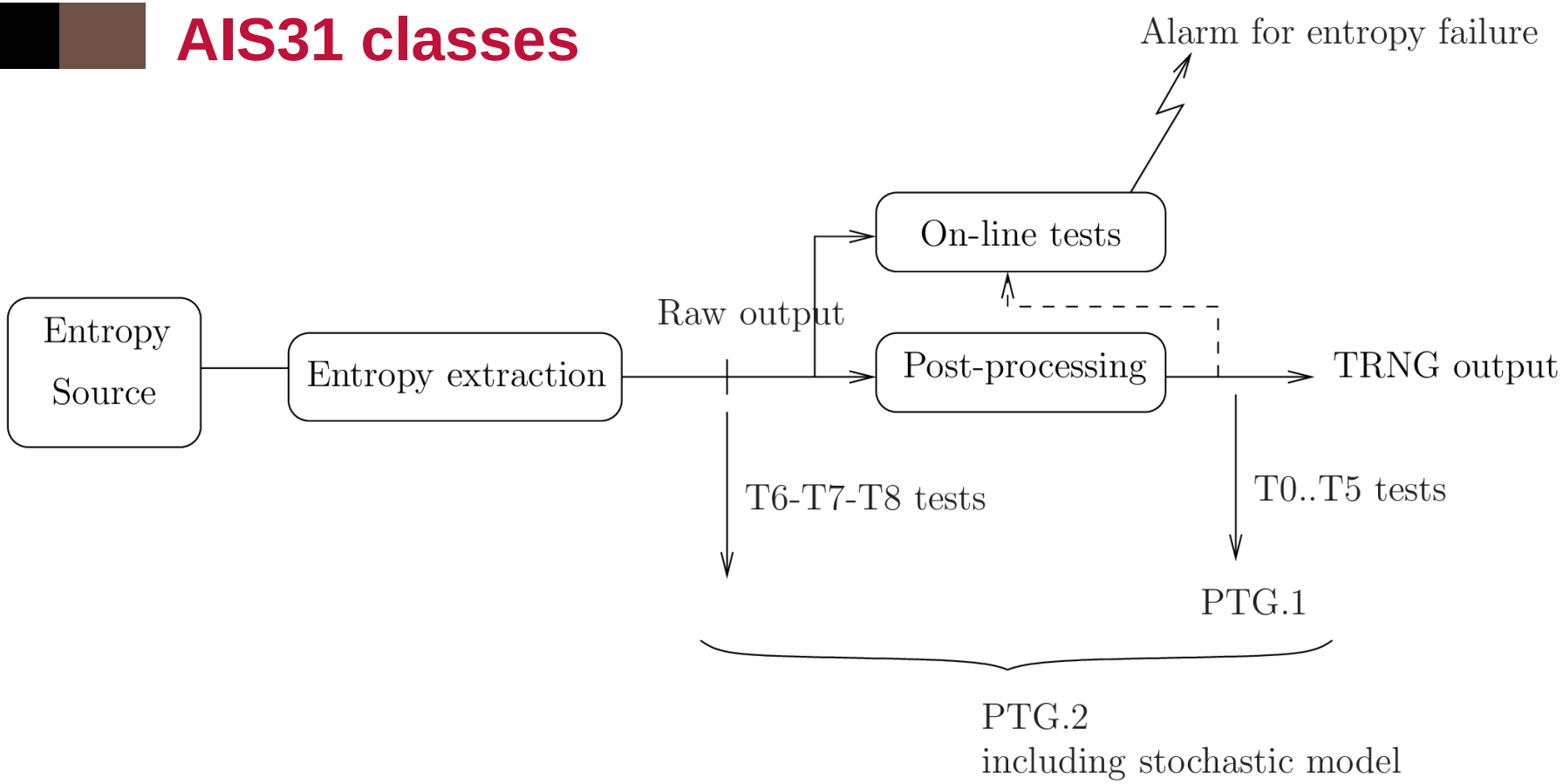
- **Among the list of tests:**
  - Test similar to what was first included in FIPS140-2
  - Random Binary Matrix Rank : check for linear dependency among fixed substrings
  - Check for periodicity (DFT, template matching tests (2 variants))
  - Compression test (Maurer's "universal statistical" test, similar to a Lempel-Ziv compression)
  - Random walk (Cumulative sum, random excursion tests (2 variants))

AIS 31

Bundesamt für Sicherheit in der Informationstechnik

- **BSI's approach:**
  - Strongly common criteria oriented
  - New approach: testing TRNG output as well as the randomness source
  - 9 statistical tests
  - Stochastic model= pdf of raw entropy source
  - Recommends using on-line tests,
    - Entropy source failure tests
    - Statistical defect tests on the raw random sequence
    - Statistical defect tests on the post-processed sequence
  - Defines TRNG classes PTG.[1-3]

AIS31 classes



•PTG.1 represents the lower-end class and PTG.2 and PTG.3 the higher-end classes
•PTG.2 adds minimum entropy requirements for post-processed bits and introduces raw data online tests and entropy source stochastic model
•PTG.3 requires cryptographic post-processing (class DRG.3 or DRG.4)

NIST SP-800 90x

- **Drafted SP-800 90A, 90B and 90C**
  - SP-800 90A considers DRNG
  - SP-800 90B considers entropy source design and validation
  - SP-800 90C considers design of hybrid RNGs using SP-800 90B + SP-800 90A
  - SP-800 90B defines requirements similar to BSI's AIS31
  - Two categories defined:
    - Entropy source
    - Full entropy source
  - Minimal online health tests defined
    - Lightweight runs detection test (repetition count test)
    - Lightweight uniformity test (adaptive proportion test)

TRNG Technology
TRNG validation methodologies

- **ISO/IEC: a standard for TRNG validation**
  - Working Group in progress
  - ISO 20543
  - Goal of standardization
    - Define RNG classes (example: deterministic, non-deterministic, hybrid) following the ISO/IEC 18031
    - Define evaluation methods for the various RNG classes

TRNG robustness against physical attacks

- **Passive attacks**
  - Observation of the TRNG activity
    - Frequency of RO TRNG
    - Register loads
  - Allows the attacker **to locate** the TRNG
- **Reverse Engineering**
  - Allows the attacker **to locate** the TRNG
- **Active attacks**
  - **Bias** generated by fault injection
  - **Force** the TRNG output or the on-line test alarm
- **Trojans**
  - Bias generated at design or manufacturing stage

PUF conclusions

- **Characteristics to assess:**
  - Steadiness (the most critical)
    - Needs correction by means of Helper data
    - Ideally expressed in a BER closed-form with SNR parameter
  - Entropy
    - Uniqueness, Randomness ,
    - Ideally expressed in closed-form by Shannon or min-entropy
  - Security
    - PUF can be attacked physically and mathematically (only with CRP)
  - ISO Standard under study
- **Design needs to consider:**
  - Post-processing to enhance the reliability
  - Protections against attacks
- **ISO standard for PUF validation in preparation**

TRNG Conclusions

- **Characteristics to assess:**
  - Entropy and Unpredictability
    - Statistical tests
    - Possibly Stochastic Model (AIS31 PTG2)
  - Robustness against attacks
- **Designing a TRNG is challenging**
  - Many design challenges:
    - The randomness quality depends on Technology and Noise
    - Need a **Test Chip** for validation
  - Necessity to embed Online tests
    - To detect abnormal behaviour
- **ISO standard for TRNG validation in preparation**

THANK YOU FOR YOUR ATTENTION !