KU LEUVEN

COSIC

Search for randomness: essential for security

Ingrid Verbauwhede
KU Leuven, ESAT - COSIC

erc

Croatia Summer School, June 14, 2022

Slides credit: Miloš Grujić, Jeroen Delvaux, Kent Chuang, Adriaan Peetermans, Roel Maes and ***ALL*** past and present PhD students

Outline

- Next generation systems
- Definition of trust
- Hardware roots of trust
- Physically Unclonable Functions
- Random Number Generation
- Conclusions

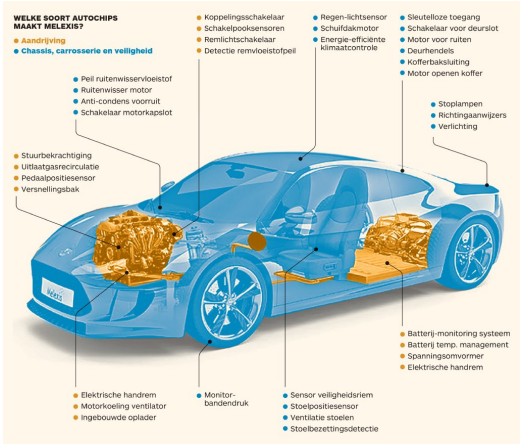
KU LEUVEN

NEXT GENERATION EMBEDDED SYSTEMS

KU LEUVEN

Automotive

"Networked embedded systems interacting with the environment"



Today 58 Melexis chips in TESLA Model Y,
170 Melexis chips in Mercedes EQS

[De Tijd, February 2, **2022**]

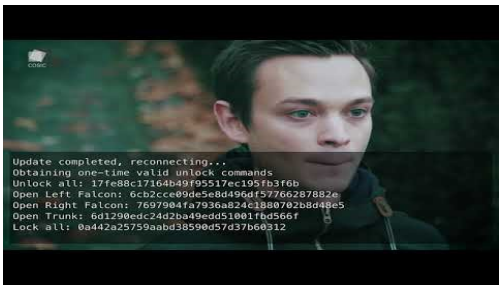
ANDY GREENBERG SECURITY 07.21.15 06:00 AM

HACKERS REMOTELY KILL A JEEP ON THE HIGHWAY—WITH ME IN IT

- Networked ➔ secure, authenticated communication, low latency
- Embedded ➔ compact (no external memory), cheap, no batch processing
- Interacting with environment ➔
  - LOW latency
  - Compact
- Resistant to attacks

KU LEUVEN

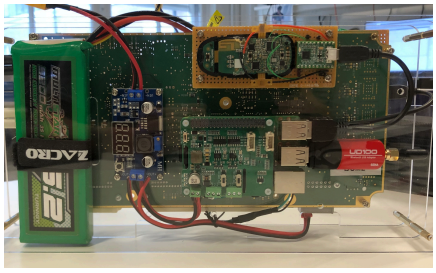
How to evaluate security? Where to start?



Tesla Model X key fob (2020)
https://youtu.be/clrNuBb3myE

Tesla Model S key fob (2018)
https://youtu.be/aVlYuPzmJoY

[Lennert Wouters, COSIC]

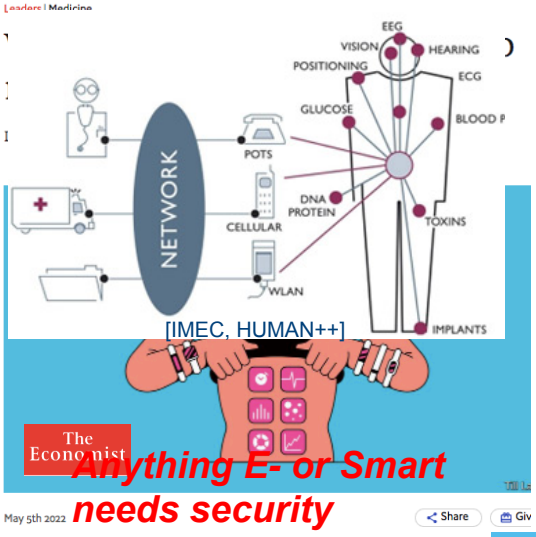


Passive Keyless Entry and Start System:
- Wireless challenge response system
- No Mutual authentication (model S)
- Weak crypto (model S)
- Secure element, but problems with protocol (model X)
- Off the shelf radios and components

KU LEUVEN

Internet of Everything – IOT – Industry4.0 - E-…

- Internet of things
- E-health, e-commerce
- E-voting, e-…
- Smart grid
- Big data



[IMEC, HUMAN++]

Anything E- or Smart needs security

KU LEUVEN

Trust Definition

Trust (R. Anderson in “Security Engineering”, after NSA):

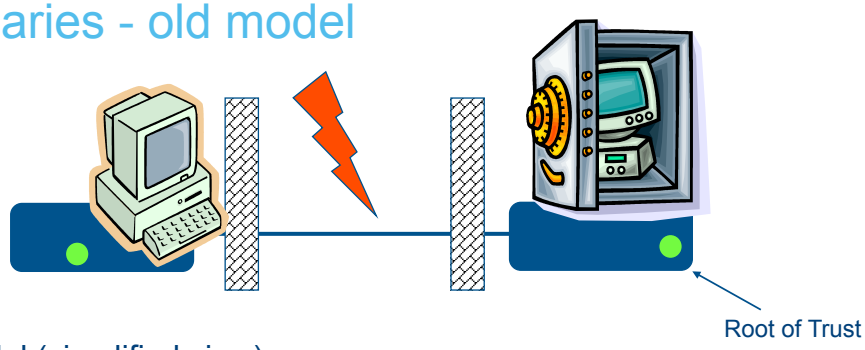
- “Trusted system or component is one whose failure can break the security policy, while a trustworthy system or component is one that won’t fail.”

Trust (Trusted Computing Group):

- “An entity can be trusted if it always behaves in the expected manner for the intended purpose.”
- Loosely stated: if trusted system or component fails, then bad things can happen.
- Goal of security: minimize what needs to be trusted

KU LEUVEN

Trust boundaries - old model



Old attack model (simplified view):

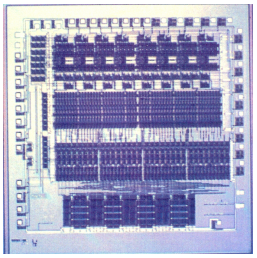
- Attack on channel between communicating parties
- Encryption and cryptographic operations in **black** boxes
- Protection by strong mathematic algorithms and protocols

Focus on performance

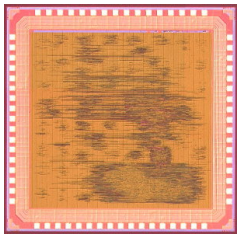
KU LEUVEN

DES, AES, ECC, SABER dedicated ASICs

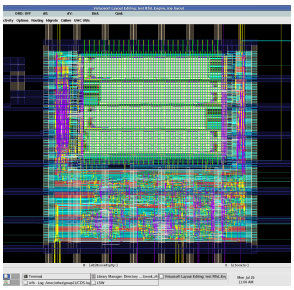
- Performance: what is feasible, throughput, latency, power (cooling), energy (battery lifetime) etc.



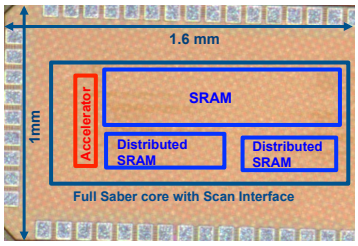
DES



Rijndael



ECC

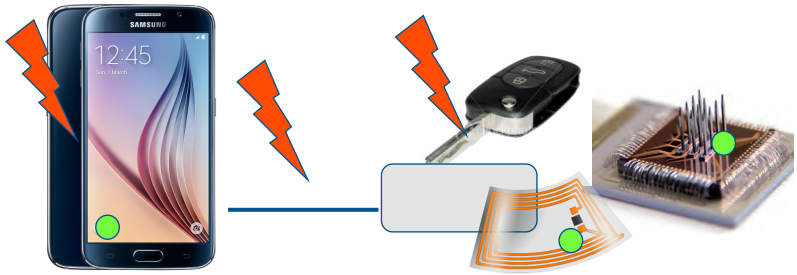


Saber

- Next: light weight crypto, FHE, …

KU LEUVEN

Trust boundaries – current model



Current attack model (simplified view):

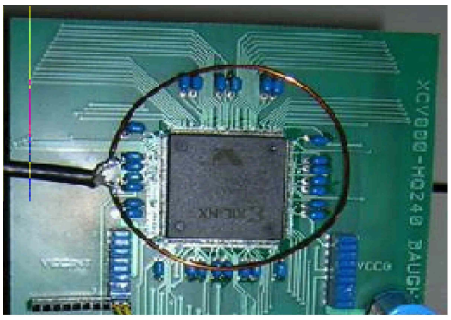
- Attack on channel between communicating parties and on components!
- Encryption and cryptographic operations in **gray** boxes
- Protect by strong mathematic algorithms and protocols
- Protect by secure implementations

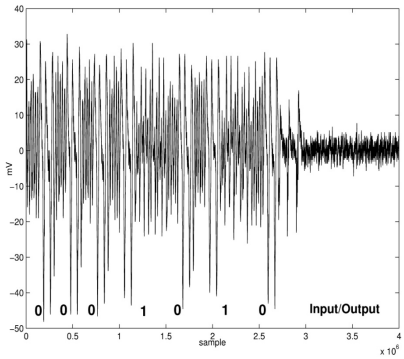
Focus on performance and security!

KU LEUVEN

Design for performance *AND* security

SEMA attack: Simple Electromagnetic Attack on Elliptic Curve
Public Key implementation – point double and add



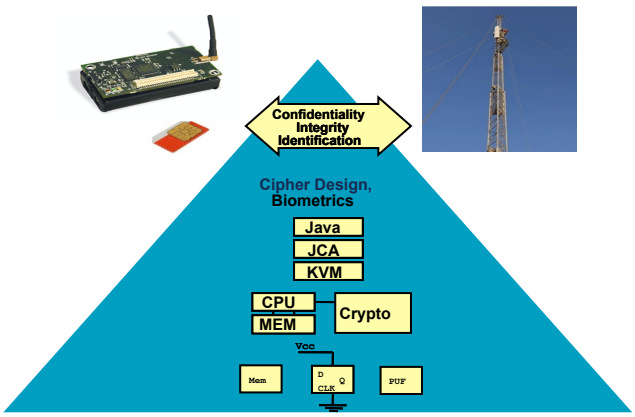


[E. Demulder EUROCON 2005]

KU LEUVEN

HOW: DESIGN METHOD

DECOMPOSE IN COMPONENTS



- Application: secure communication
- Algorithms: public key, secret key, post-quantum
- Architecture: Hardware/Software platform, Sancus
- Micro-architecture: crypto co-processors, instruction set extension,
- Logic circuits and (secure) memory
- TRNGs and PUFs
- Technology

[DATE2007]

*"A root of trust is a component at a **lower** abstraction layer, upon which the system relies for its security."*

KU LEUVEN

RANDOMNESS

TRUE RANDOM NUMBER GENERATION
PHYSICALLY UNCLONABLE FUNCTIONS (PUFS)

KU LEUVEN

Which number is more random?

0123456789 5051433441092 314159265359

KU LEUVEN

2

Which number is more random?

0123456789

5051433441092

50° 51' 43.3" N 4° 41' 09.2" E

314159265359

π

KU LEUVEN

2

What is a Random Number?

Is 4 a random number?

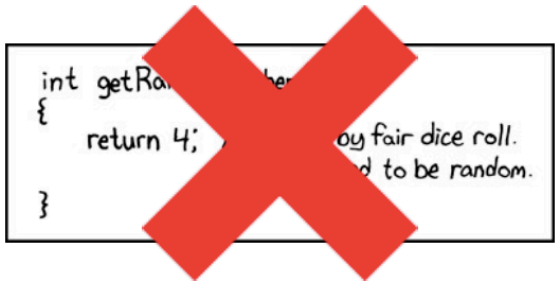
```
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
               // guaranteed to be random.
}
```

KU LEUVEN

3

What is a Random Number?

Is 4 a random number?



Random Numbers = Randomly Generated Numbers

KU LEUVEN

3

Randomness is not a statistical property
Randomness models the unpredictability by the attacker

KU LEUVEN

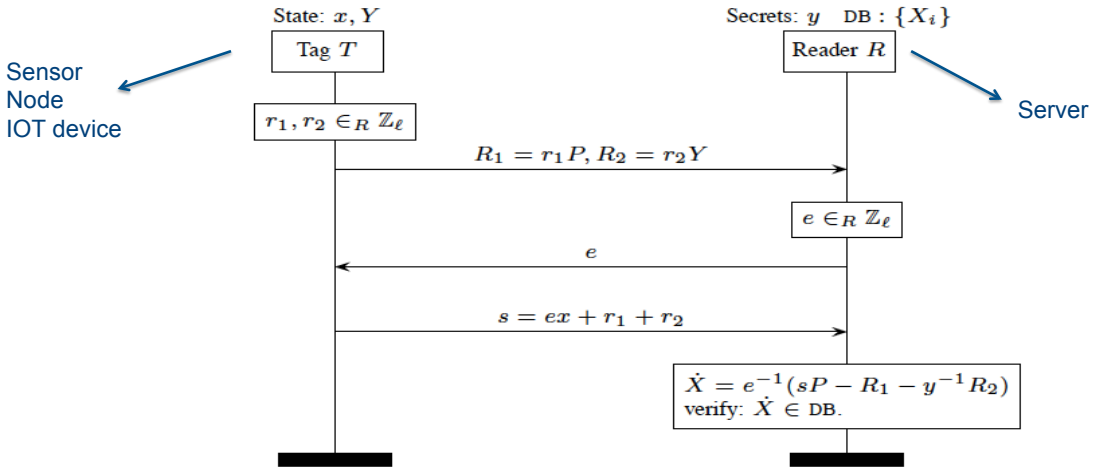
4

If the RNG fails, security is lost
even if strong cryptography is used

KU LEUVEN

11

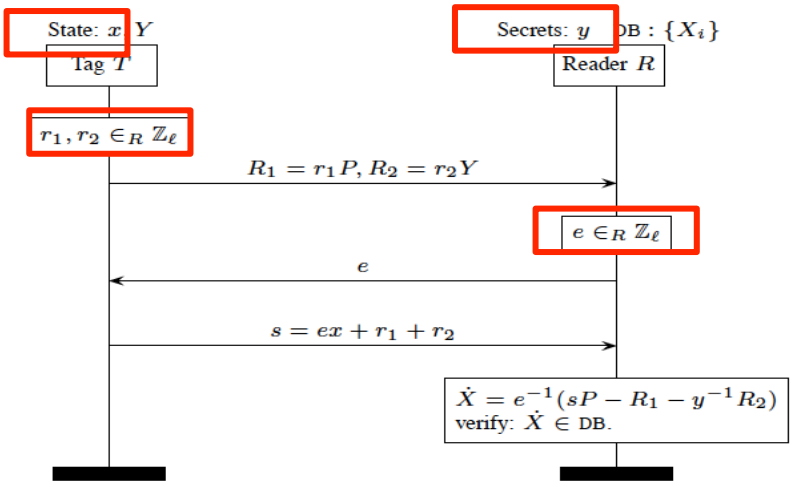
How the crypto protocol paper sees it:



Source: J.Hermans, et al., "Proper RFID Privacy: Model and Protocols," IEEE Trans on Mobile computing, 2014

KU LEUVEN

Protocol relies on secrets and random numbers

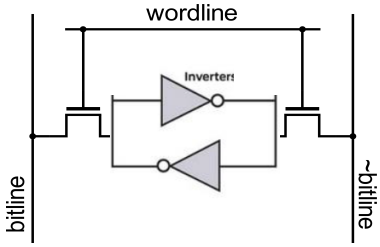


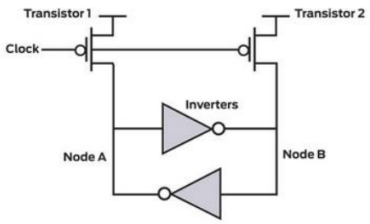
Source: J.Hermans, et al., "Proper RFID Privacy: Model and Protocols," IEEE Trans on Mobile computing, 2014

KU LEUVEN

Two ends of randomness

- Fixed randomness, stable over time
- Randomness changing over time





Physically Unclonable Function

Used: secret key
Challenge: time varying noise

True Random Number Generation

Used: freshness
Challenge: fixed noise

KU LEUVEN

RANDOMNESS

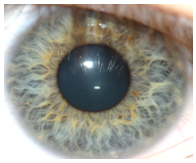
PUFS

KU LEUVEN

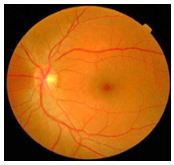
Biometrics

- Every human is unique

fingerprint

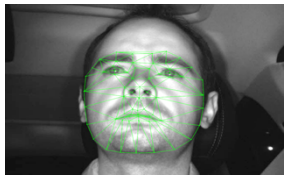


iris

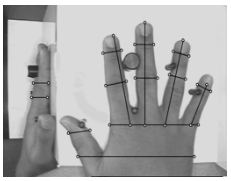


retina

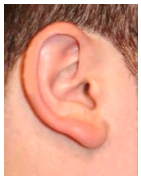
gait



face



hand

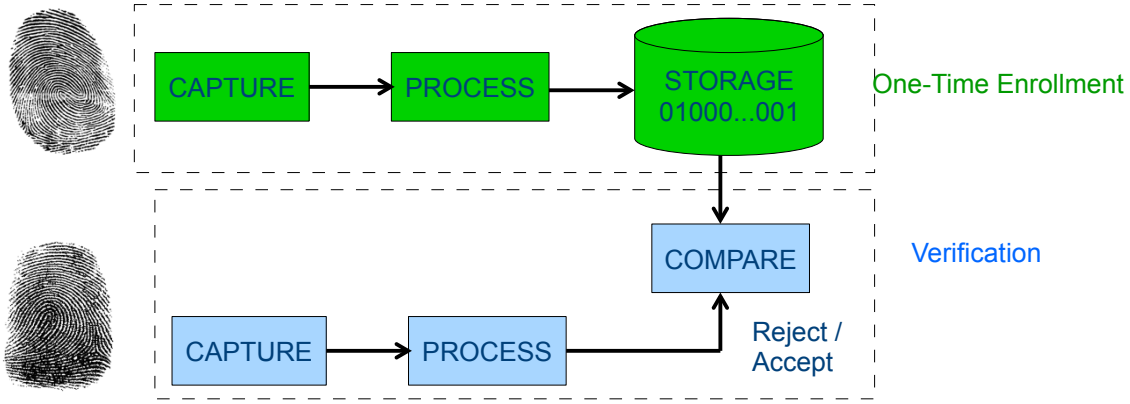


ear

KU LEUVEN

Biometrics

- User authentication (prove identity): two phases



KU LEUVEN

Fixed randomness - Silicon PUFs

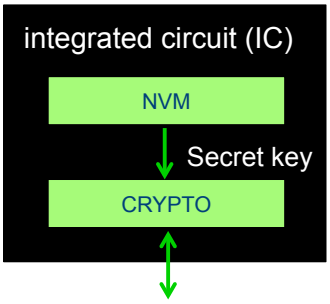
Purpose: CHEAP unique key or ID generation

Replacement for more expensive secure non-volatile memory (NVM)

- EEPROM/Flash
- Fuses
- Battery-backed SRAM

Embedded context:

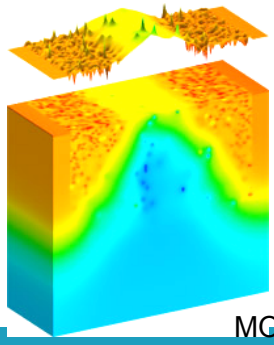
- Physical attacks
- Constraints on cost and resources



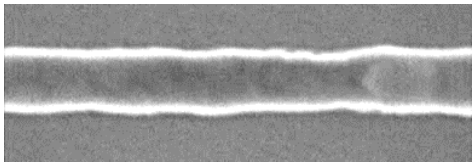
KU LEUVEN

Variability is inherently presented in ICs

- Variability in transistors and interconnect
- In general undesired – except for PUFs
- Random dopant fluctuation
- Line edge/width roughness



MOSFET

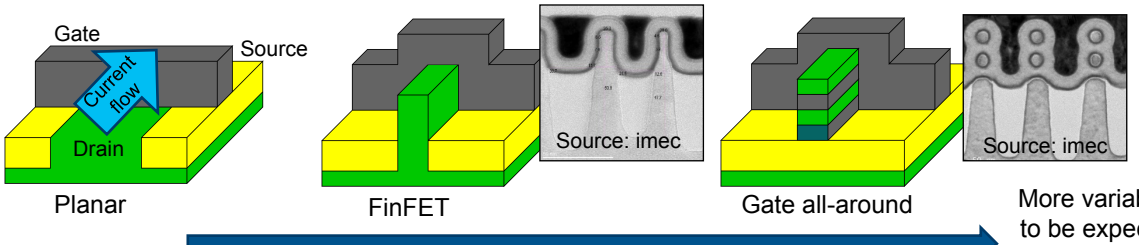


Interconnect

KU LEUVEN

More opportunities brought by scaling

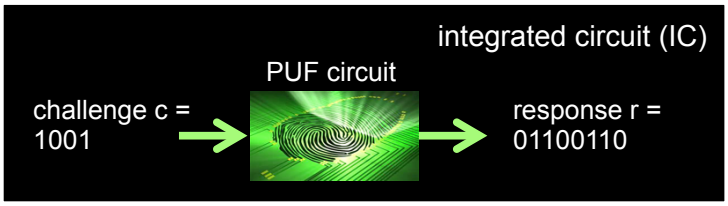
- Even more challenging to manufacture identical devices in scaled technologies
  - Moore's Law
  - 40nm → 28nm →16nm → 7nm → ...
- More variability comes from:
  - More processing steps, new materials
  - Decreased size (e.g. 2nm difference → **5%** in 40nm and **30%** in 7nm)



KU LEUVEN

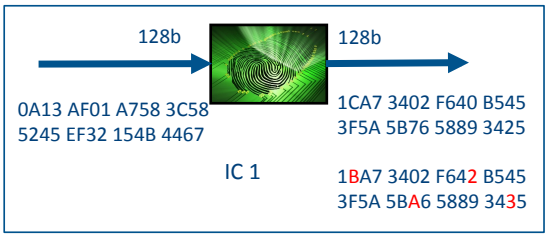
Model: PUF (F = Function)

- Binary input & binary output
  - Therefore, easy of integration with other systems
  - Input = challenge; output = response
  - **C**hallenge-**R**esponse **P**airs (CRPs)
  - Easy to evaluate



KU LEUVEN

The almost ideal PUF

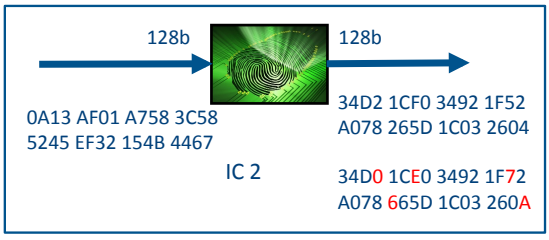


Chip-dependent binary function with noisy output

Evaluation 1

≈ 1-15% noise

Evaluation 2



Evaluation 1

Evaluation 2 ≈ 1-15% noise

Unique
Unpredictable
Unclonable
Intrinsic
Tamperproof
Stable
…

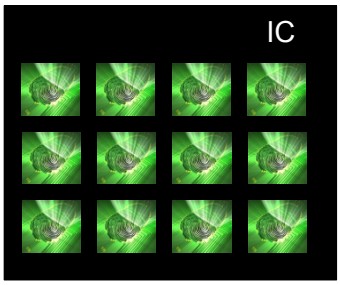
IDEAL PUF is without noise

KU LEUVEN

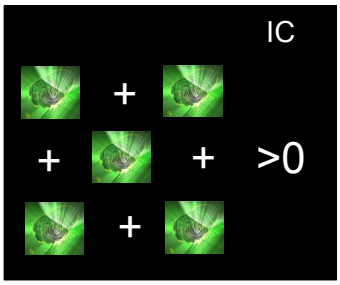
PUF (F = Function)

- **Dream 1: IDEAL PUFS don‘t exist...**
- Two design methodologies for PUF circuits

Key PUF



Authentication PUF



KU LEUVEN

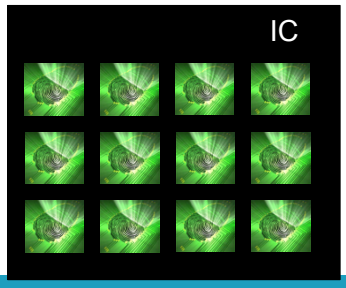
Key Generation PUF (aka weak) = 1 output

- An array of identically designed circuit elements
- Each producing 1 (or a few) response bit(s)
- High-quality response bits, i.e., high entropy
- Limited number of bits, e.g., a few 1000s
- E.g., SRAM PUF

BADLY chosen terminology

- Typical application: key generation

  E.g. 128-bit AES



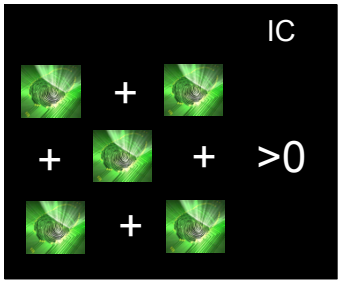
KU LEUVEN

Authentication PUF (aka Strong)

- Mathematical operations
- E.g., sum of delays, currents, voltages, frequences etc.
- Can produce a gazillion of response bits (2^{128})
- Low-quality bits: highly correlated, low-entropy, mostly broken

BADLY chosen terminology

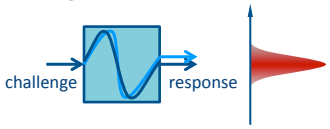
- E.g., arbiter PUF
- Typical application:
- IC authentication



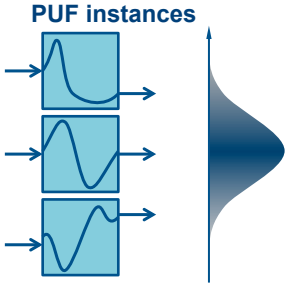
KU LEUVEN

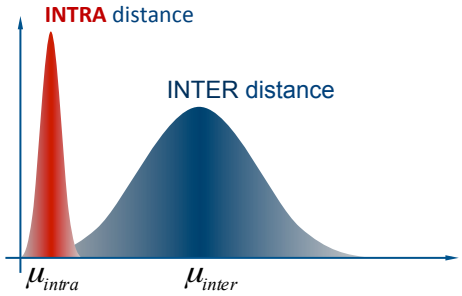
Uniqueness

Single PUF instance



Multiple "identically manufactured" PUF instances



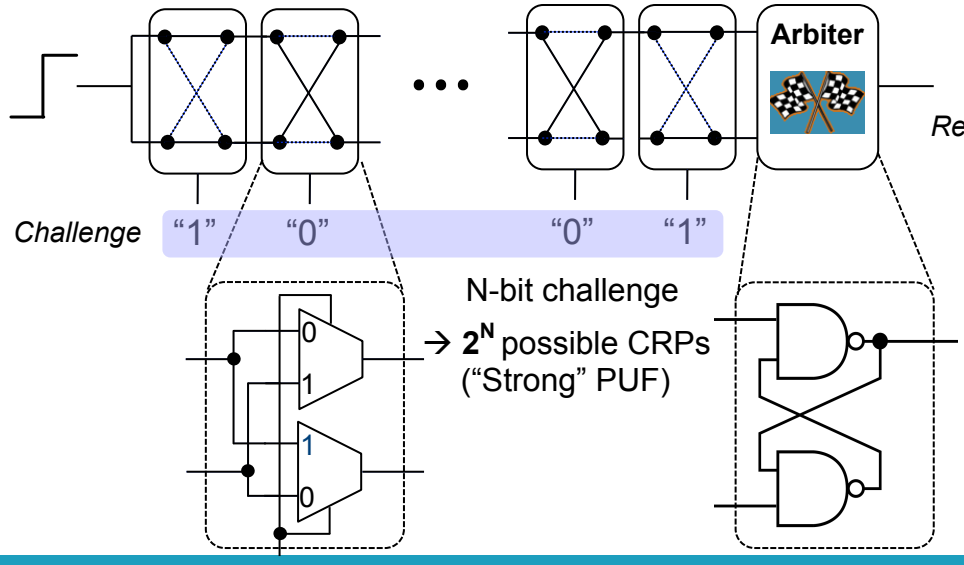


Basic PUF property:

$\mu_{inter} >> \mu_{intra}$

KU LEUVEN

Authentication PUF: Arbiter PUF – based on timing differences

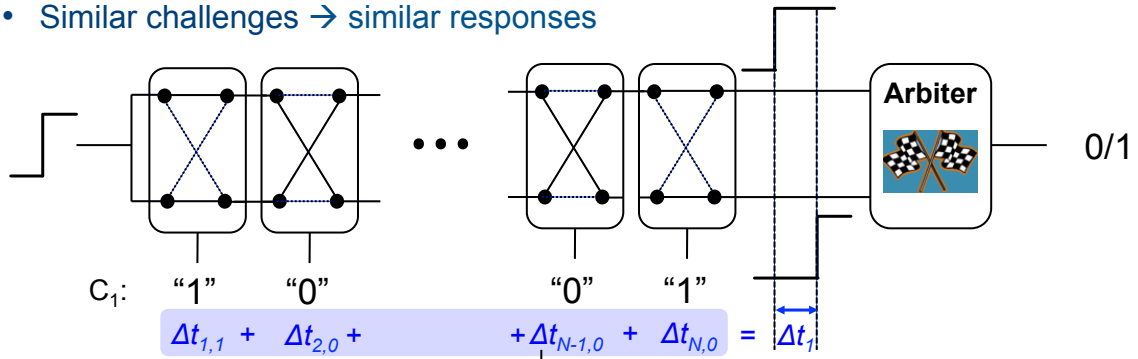


[Lee, VLSIC 2004]
[Gassend, 2004]

KU LEUVEN

Arbiter PUF is not an ideal strong PUF

- *Linear* additive structure: sum of delays
- Similar challenges → similar responses

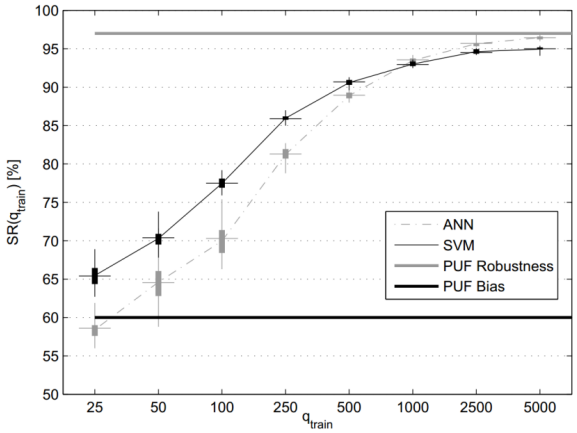


KU LEUVEN

Arbiter PUFs: Modeling attacks

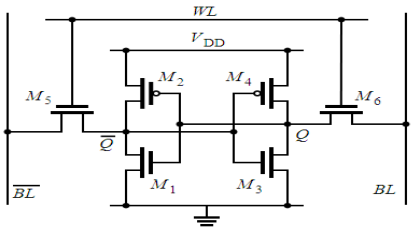
Arbiter PUF problem: **Modeling Attacks**

- Circuit delay = additive
  - ⇨ Arbiter response = linear function of unknown delay parameters
  - ⇨ Solve unknown parameters from observed challenge/response pairs
- Solving: linear regression, neural networks (ANN), support vector machines (SVM), ...

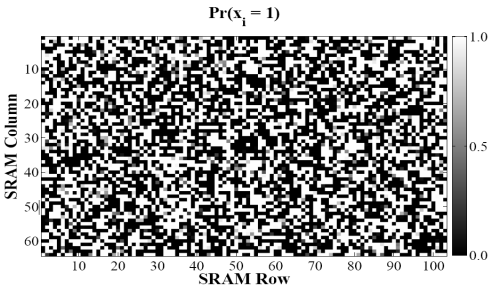


KU LEUVEN

Key (Weak) SRAM PUF: Basics

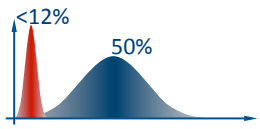


6T CMOS SRAM Cell



Statistics:

- INTER distance between different PUFs
- INTRA distance between multiple readings same PUF



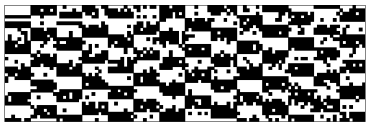
Guajardo et al. 2007, FPGA SRAM temp./volt. var.

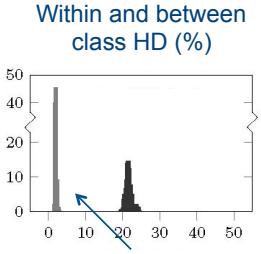
KU LEUVEN

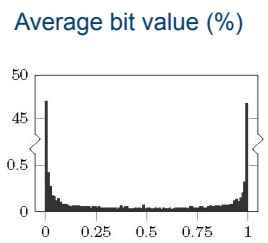
PUF behavior in SRAM of commodity micro-controller

Black box approach (off the shelf micro-controllers)

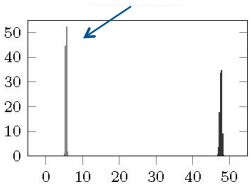
- PIC16F1825

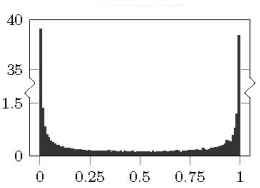






- STM32F100R8





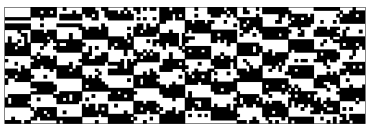
[PhD thesis Anthony VH, PUFFIN]

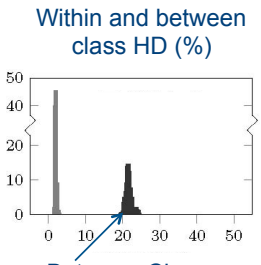
KU LEUVEN

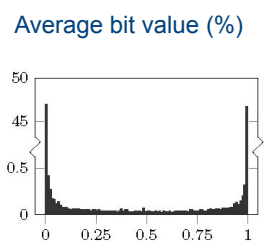
PUF behavior in SRAM of commodity micro-controller

Black box approach (off the shelf micro-controllers)

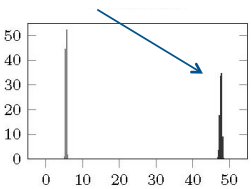
- PIC16F1825

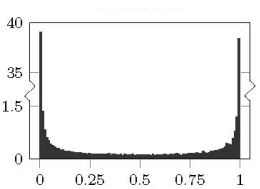






- STM32F100R8





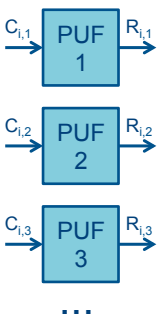
Not yet useful: needs post-processing to create ID or key!

KU LEUVEN

PUF Usage 1: Authentication

KU LEUVEN

Challenge/Response (C/R) Identification
Step 1: Enrollment



1. Generate random challenges $C_{i,j}$ and apply to PUF population
2. Store responses $R_{i,j}$ in CRP database

PUF ID	Challenge	Response
1	$C_{1,1}$	$R_{1,1}$
1	$C_{2,1}$	$R_{2,1}$
1	$C_{3,1}$	$R_{3,1}$
2	$C_{1,2}$	$R_{1,2}$
2	$C_{2,2}$	$R_{2,2}$
2	$C_{3,2}$	$R_{3,2}$
3	$C_{1,3}$	$R_{1,3}$
3	$C_{2,3}$	$R_{2,3}$
3	$C_{3,3}$	$R_{3,3}$
...	...	...

KU LEUVEN

Challenge/Response (C/R) Identification Step 2: (Authenticated Identification)

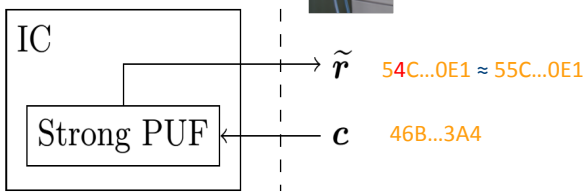
1. Send ID = 2
2. Pick random stored challenge $C_{i,2}$ for ID=2 and apply to PUF
3. If measured response $R'_{i,2}$ is "close enough" to stored response $R_{i,2}$ then accept
4. Scratch used CRP from database

PUF ID	Challenge	Response
1	$C_{1,1}$	$R_{1,1}$
1	$C_{2,1}$	$R_{2,1}$
1	$C_{3,1}$	$R_{3,1}$
2	$C_{1,2}$	$R_{1,2}$
~~2~~	~~$C_{2,2}$~~	~~$R_{2,2}$~~
2	$C_{3,2}$	$R_{3,2}$
3	$C_{1,3}$	$R_{1,3}$
3	$C_{2,3}$	$R_{2,3}$
3	$C_{3,3}$	$R_{3,3}$
...	...	...

IDEAL: but what is the reality?? [CHES 2014 Jeroen Delvaux]

KU LEUVEN

Secure light-weight Entity Authentication with strong PUFs: Mission Impossible? CHES 2014



Prerecorded I/O pairs

1F2…51F	C73…467
46B…3A4	55C…0E1
E03…127	0A1…245

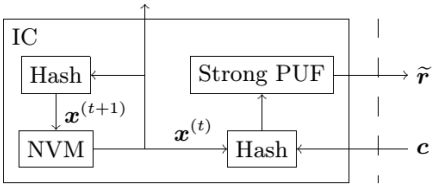
3 authentications, discard pair after use to avoid replay

No secure instantiation: PUF modeling attacks
- Mathematical clone: learn full I/O behavior given a small training set (machine learning)
- No PUF has valid claim to be resistant and lightweight

KU LEUVEN

Mission impossible?

- All “light-weight” protocols need extra building blocks: NVM, hash, TRNG, error correction code, etc.
- Error tolerant, error correction?
- Crypto post-processing: hash?
- NVM required? Write-secure or read/write secure?
- Scalability? (Size of server?)



(g) Logically reconfigurable PUF [11].

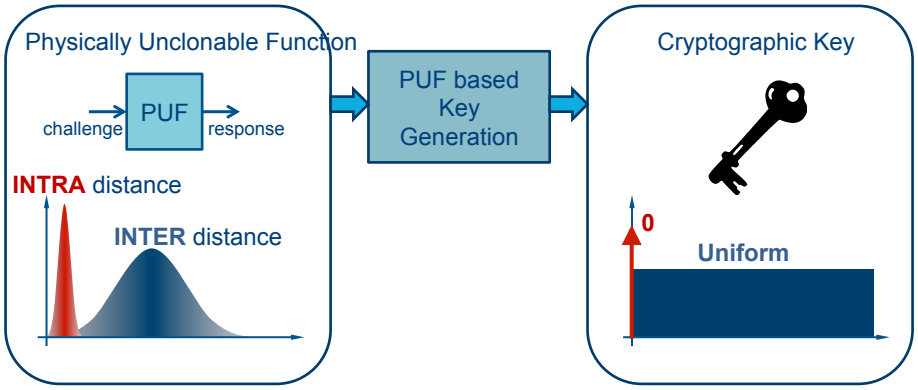
[CHES 2014 Jeroen Delvaux]

KU LEUVEN

PUF Usage 2: Key Generation

KU LEUVEN

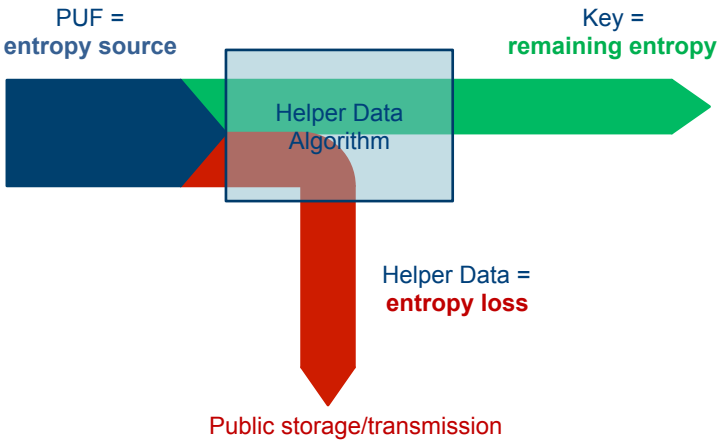
Cryptographic Key Generation: Basic Goals



KU LEUVEN

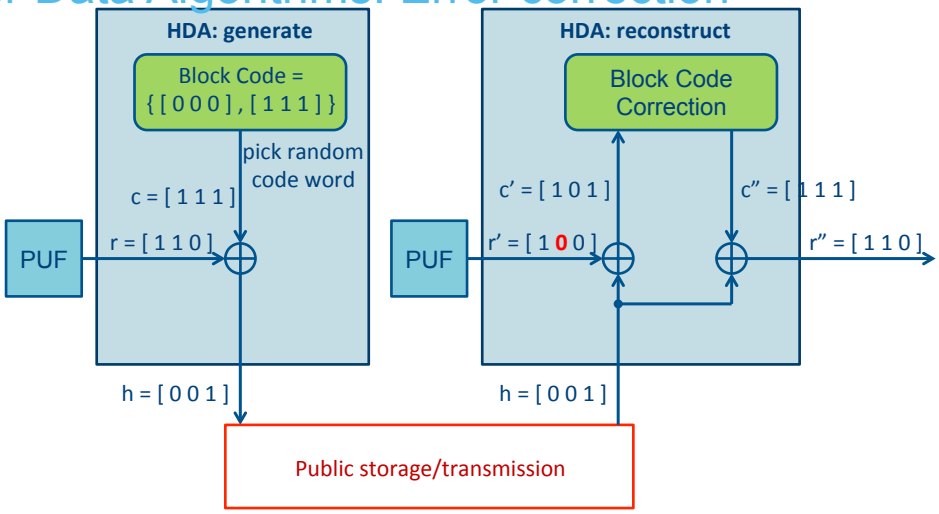
Cryptographic Key Generation: Entropy flow

- Entropy is a measure for the "secrecy" of a value
  i.e., the uncertainty an adversary has about the value



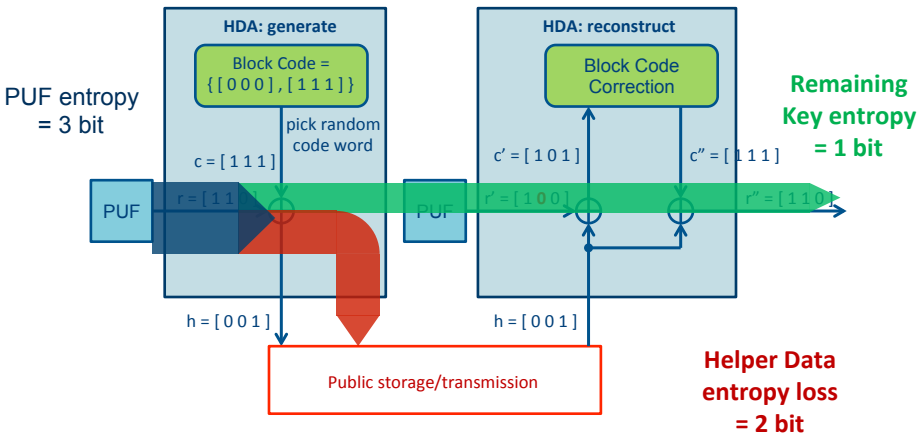
KU LEUVEN

Cryptographic Key Generation: Helper Data Algorithms: Error correction



KU LEUVEN

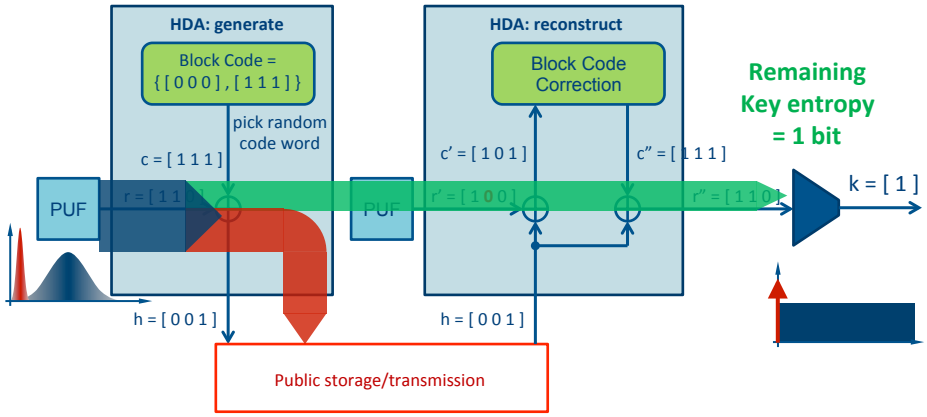
Cryptographic Key Generation: Helper Data Algorithms: Entropy flow



KU LEUVEN

Cryptographic Key Generation: Helper Data Algorithms: Entropy extraction

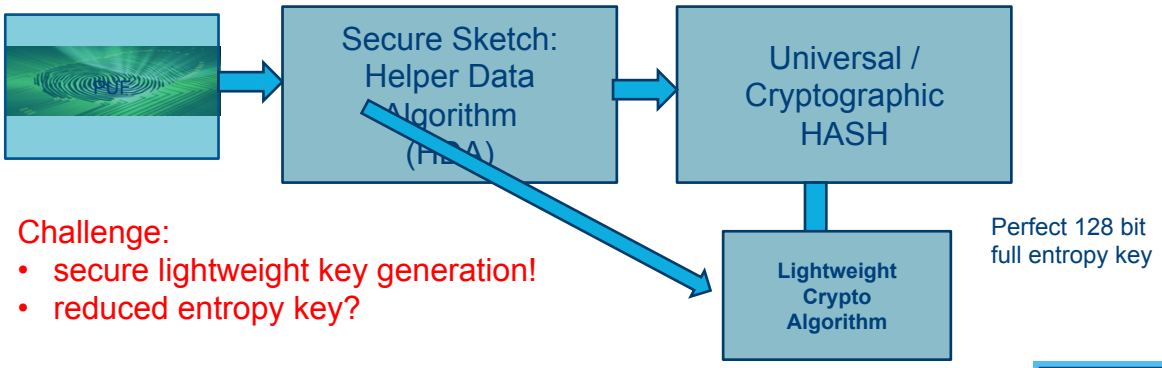
- Still 1 bit of entropy in output of length 3 bit ⇨ not uniform
- Apply entropy-extracting compression function
  - e.g. a universal hash function



KU LEUVEN

Light weight solution

- PUF as 'light-weight' key generation for IOT, RFID tags, etc.
- Key generation is larger than lightweight algorithm??



Challenge:
- secure lightweight key generation!
- reduced entropy key?

KU LEUVEN

Pseudo Random Number Generation or Deterministic Random Number Generator

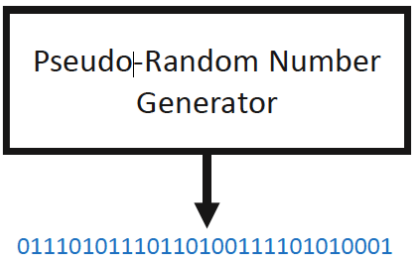
KU LEUVEN

Pseudo-Random Number Generators

- Also called “Deterministic Random Number Generators” or DRNGs
- “Random-*looking*” but not random
- Expand a short random sequence called the ‘seed’ into a longer sequence
  - The seed has to be generated by a True Random Number Generator
  - DRNG can be periodically re-seeded

KU LEUVEN

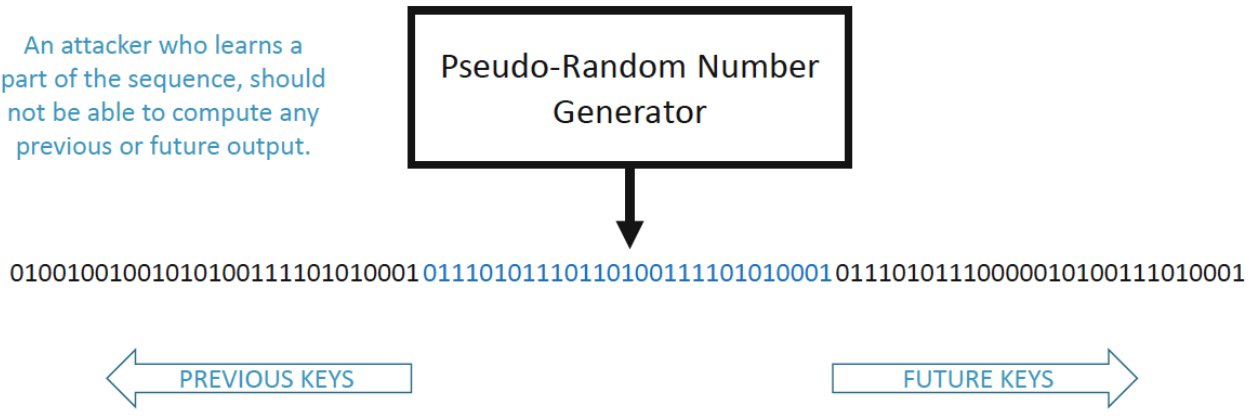
Requirements



KU LEUVEN

Requirements

An attacker who learns a part of the sequence, should not be able to compute any previous or future output.



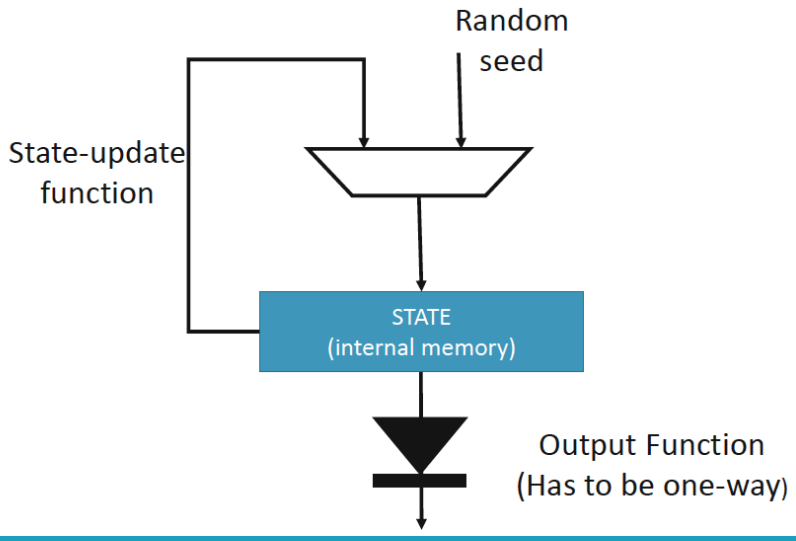
KU LEUVEN

Requirements

- Forward Secrecy
  - The assurance that subsequent (future) output values cannot be determined from the current or previous output values.
- Backward secrecy
  - The assurance that previous output values cannot be determined from the current or future output values.

KU LEUVEN

Generic architecture



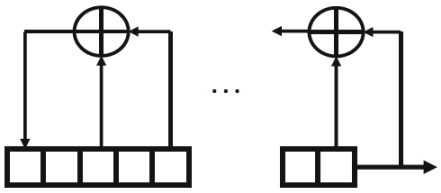
KU LEUVEN

Enhanced requirements

- Enhanced Forward Secrecy
  - The assurance that subsequent (future) output values cannot be determined from the current internal state, or from current or previous output values.
- Enhanced Backward secrecy
  - The assurance that previous output values cannot be determined from the current internal state, or from the current or future output values.

KU LEUVEN

What about LFSRs?



- Long periodic sequence 2^n-1
- Balanced 0s and 1s
- Balanced patterns
- Any state size
- However, …
  - Distinguishable from ideal RNG
    - Using Linear Complexity Test
  - No Forward Secrecy!
  - No Backward Secrecy!

KU LEUVEN

AIS-31 DRNG classes – German BSI

- DRNG1
  - Output indistinguishable from ideal RNG
  - Forward secrecy
- DRNG2
  - + Backward secrecy
- DRNG3
  - + Enhanced backward secrecy
- DRNG4
  - + Enhanced forward secrecy

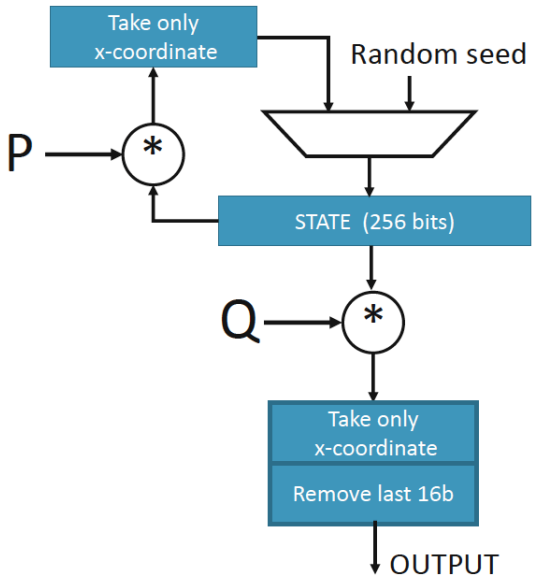
KU LEUVEN

DUAL – EC – NIST SP800-90A

- Based on elliptic curve cryptography
  - Operations on the points of the elliptic curve
  - Each point P(x,y) has two coordinates of 256 bits
  - It is easy to compute kP = P + P + P + … P for any scalar k
  - … but it is difficult to compute k if you know P and kP
- Designed by NSA and standardized by NIST in SP 800-90A
- Later revealed to have a back door
- Recalled

KU LEUVEN

DUAL - EC



KU LEUVEN

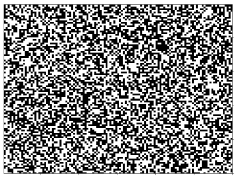
DUAL - EC

- P and Q are constants selected by the designer
- What if P = d * Q, where d is known by the designer?
  - The attacker can easily reconstruct the next state as d * Q
- No forward secrecy
  - Attack effort is effectively reduced to 2^{16}

KU LEUVEN

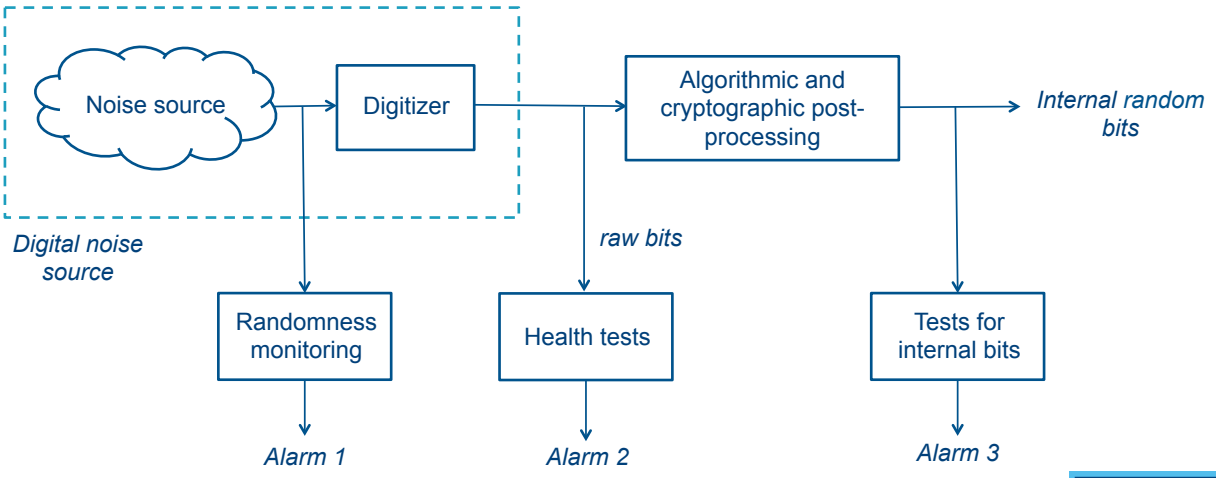
TRUE RANDOM NUMBERS

DESIGN AND ATTACKS ON TRNGS



KU LEUVEN

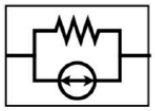
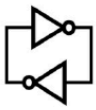
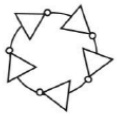
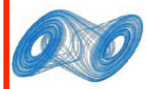
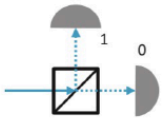
Architecture: True Random Number Generator



KU LEUVEN

Entropy sources

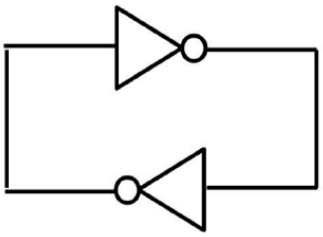
Thermal noise | Metastability | Timing jitter | Chaos circuit | Quantum effect

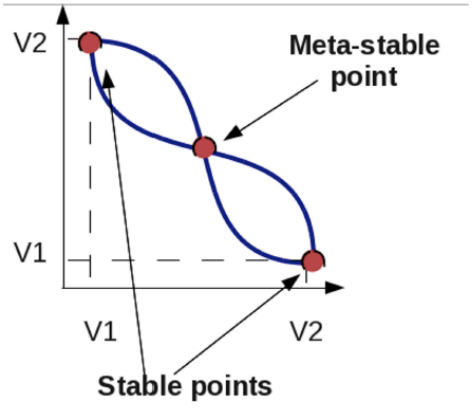






KU LEUVEN

Terminal

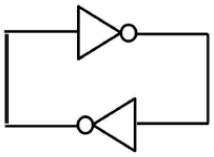
Metastability

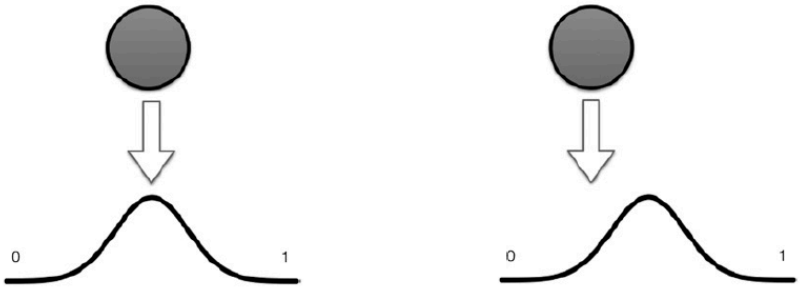




KU LEUVEN

Metastability

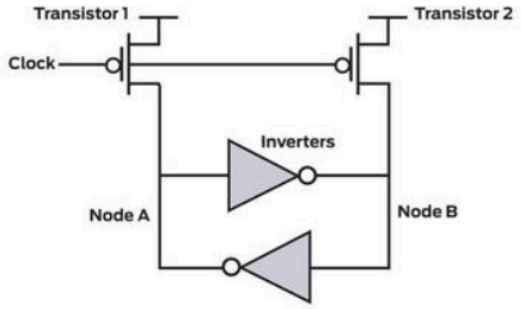


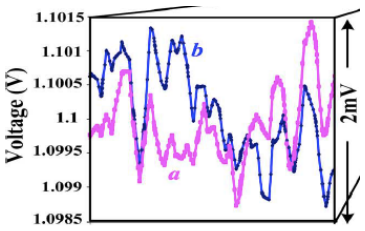


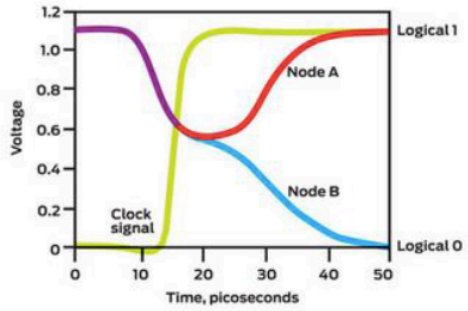
KU LEUVEN

28

Metastability source: ASIC





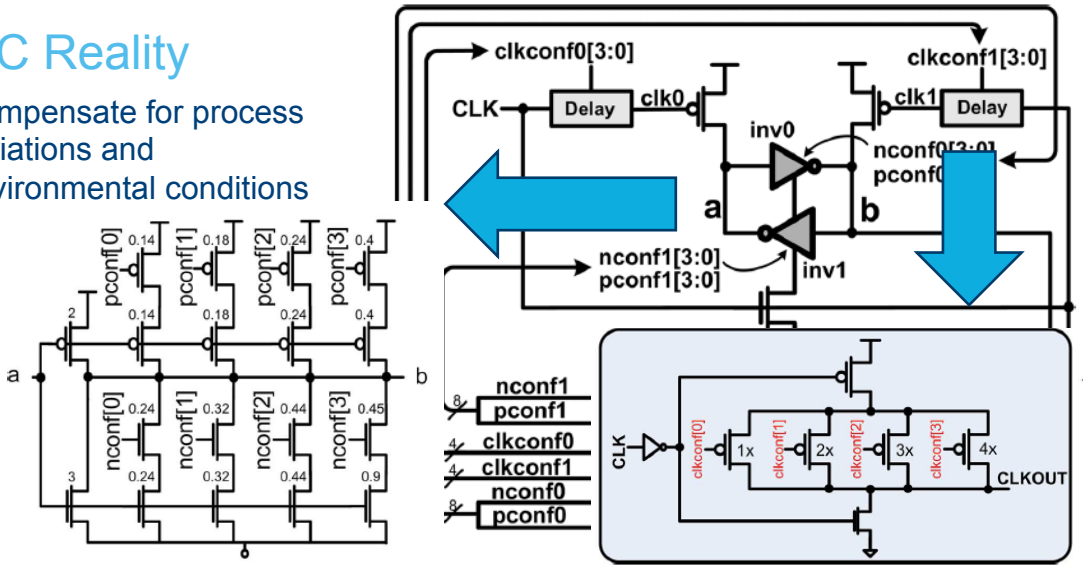


R. Parker, DAC 2019, S. Mathew JSSC 2012 –
https://software.intel.com/en-us/articles/intel-digital-random-number-generator-drng-software-implementation-guide

KU LEUVEN

ASIC Reality

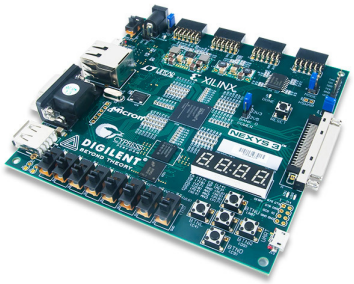
- Compensate for process variations and environmental conditions



"2.4 Gbps, 7 mW All-Digital PVT-Variation Tolerant True Random Number Generator for 45 nm CMOS High-Performance Microprocessors," S. Mathew et al, IEEE JSSC, Nov 2012

KU LEUVEN

TRNGs on FPGAs

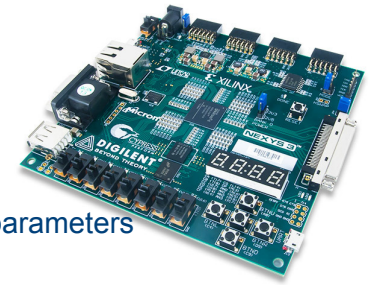


Challenges:

- FPGAs developed to behave in deterministic digital manner
- Vendors' goals: low noise & stable digital behavior → decrease variations of non-deterministic processes → limited number of available noise sources
- TRNG simulations not supported by tools and implementations require special constraints ("set_dont_touch")

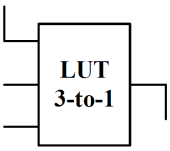
KU LEUVEN

TRNGs on FPGAs

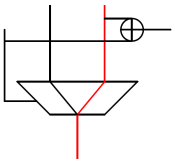


Challenges:

- Portability: different FPGAs → different structure & physical parameters
- Maintain claimed security level on different platforms
- Unrealistic noise assumptions



Fast carry structure on Cyclone IV Intel

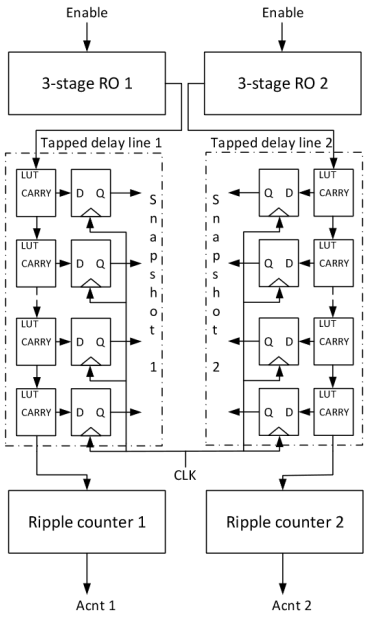


Fast carry structure on Spartan 6 Xilinx

KU LEUVEN

Jitter based

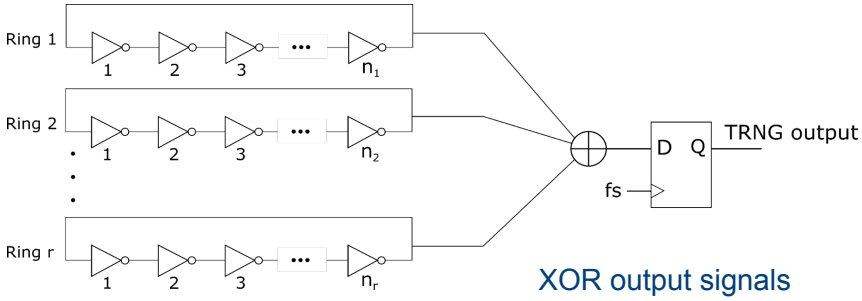
- FPGA Delay line ring oscillators
- On-chip differential jitter measurement
- Isolate contribution of the white noise:
  - differential setup → reduce global noise
  - short measurement time → reduce flicker noise
- Lower bound on jitter strength → conservative entropy estimate



[Yang et al., AsianHOST'17]

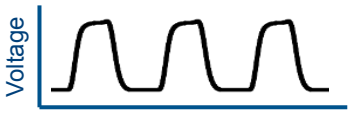
KU LEUVEN

Simple Jitter based Ring Oscillator TRNG



XOR output signals
sampled at arbitrary frequency

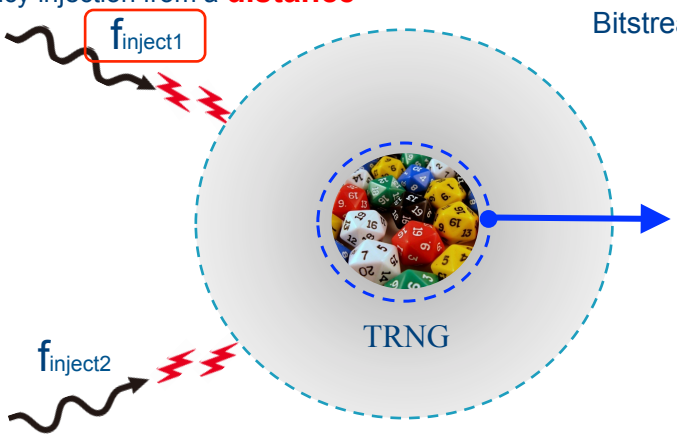
The output signal of an RO



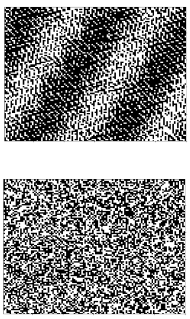
KU LEUVEN

IEMI: Intentional Electro Magnetic Injection

frequency injection from a **distance**



Bitstream generated by TRNG



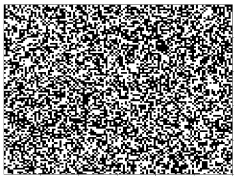
Goal: suppress jitter of Ring Oscillators

KU LEUVEN

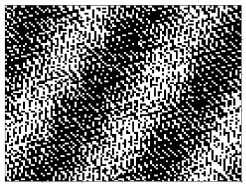
Suppression of ROs' jitter

Jitter : variations in the transition timing of an RO in the time domain

Normal TRNG output



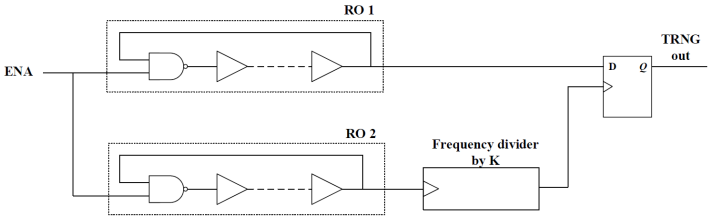
introduce CW to reduce randomness



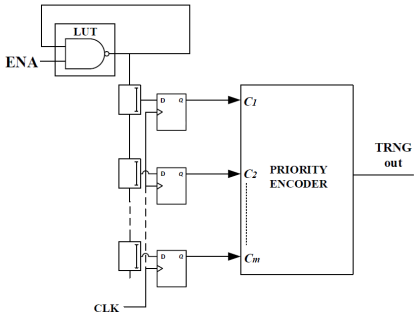
KU LEUVEN

Architectures suitable for FPGAs

- Entropy source: timing jitter
- Time deviation from a periodic signal due to random noise - accumulates *very slowly*
- Relatively easy to model
- Susceptible to temperature and voltage variations



Elementary oscillator based TRNG [BLMT11]

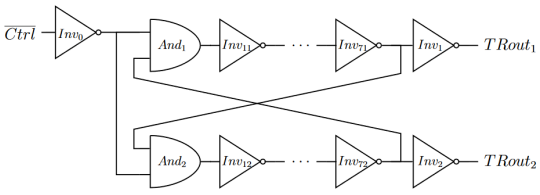


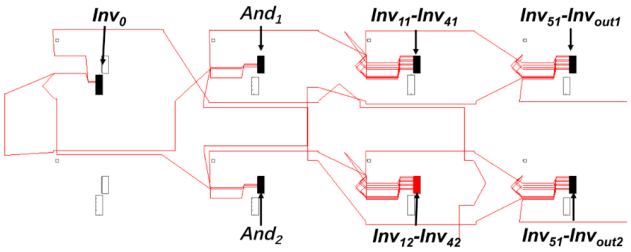
Delay chain TRNG [Rožić et al., DAC'15]

KU LEUVEN

Architectures suitable for FPGAs

- ➢ Transition Effect Ring Oscillator (TERO) TRNG
- entropy source: oscillator metastability
- digitizer: asynchronous counter → LSB used as random bit
- manual placement and routing of LUTs
- high throughput (1.3 Mb/s)



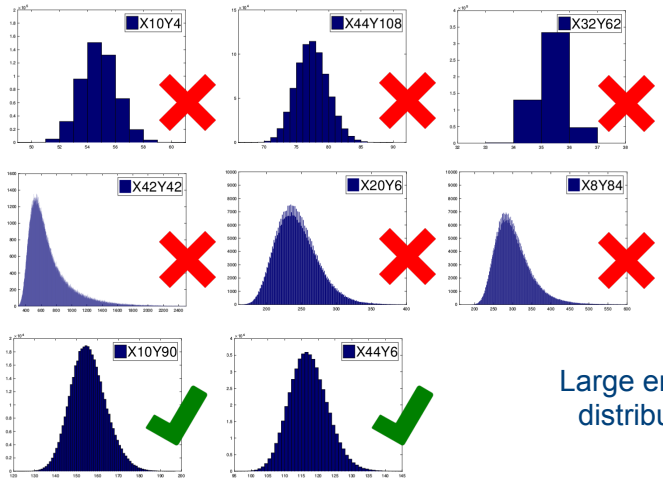


[VD10] *[Cao et al., MWSCAS'16]*

KU LEUVEN

TERO TRNG on FPGAs

- Influence of FPGA location on oscillation occurrence



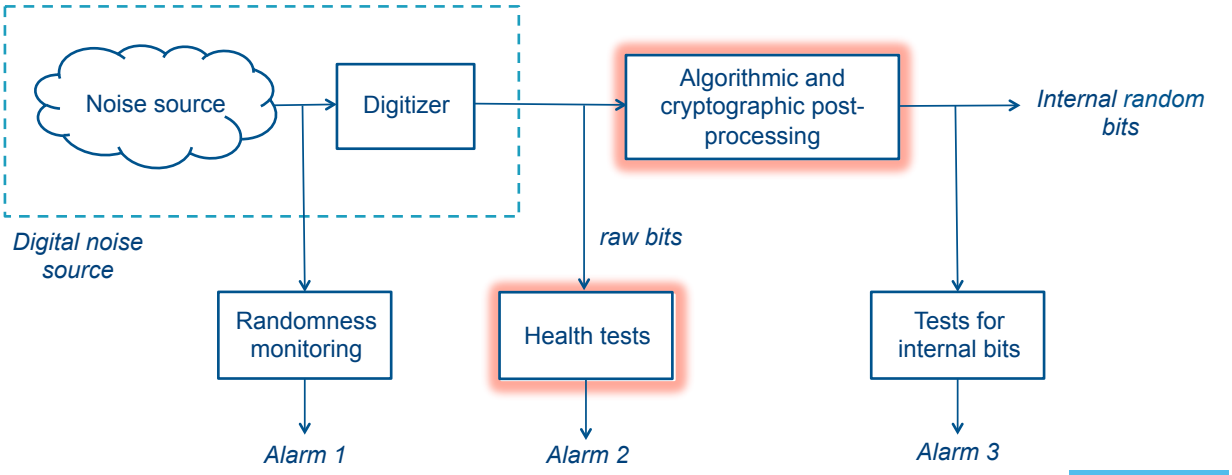
Very few transitions

Statistical defects

Large enough number of transitions (> 100), distribution close to Gaussian, pass NIST SP800-22

KU LEUVEN

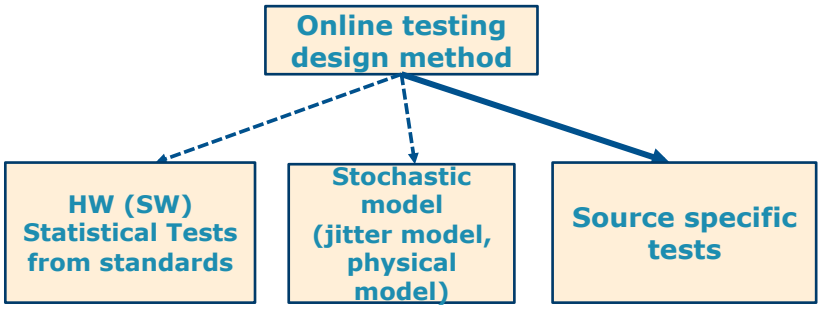
TRNG Tests and post processing



GERMAN BSI, AIS-31 or US NIST SP800-B/C

KU LEUVEN

Online tests: options



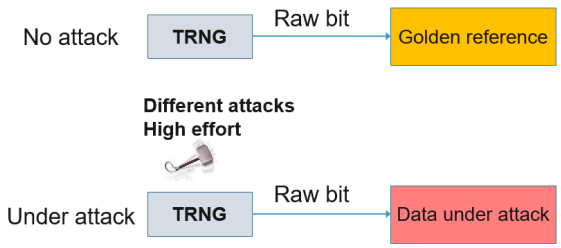
Challenges:

- Low latency
- Release data before it has been checked? Health bit?
- Memory requirements?

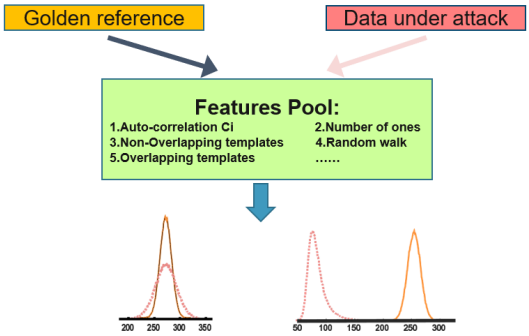
KU LEUVEN

Online tests: TOTAL methodology

- Step 1: Data collection



- Step 2: Selection of useful features



KU LEUVEN

Online tests: TOTAL methodology

- Step 3: Feature verification

➢ Collect more data to verify the bounds
➢ Test the robustness of the selected features

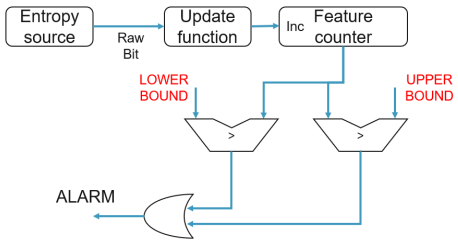
- Step 4: Attack impact analysis

➢ Check the usefulness under different attack efforts

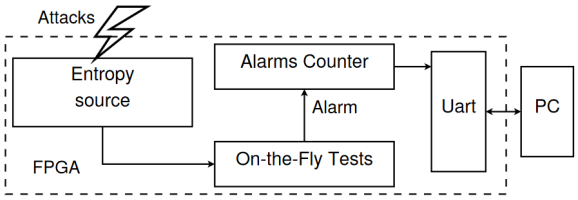
KU LEUVEN

Online tests: TOTAL methodology

- Step 5: HW implementation



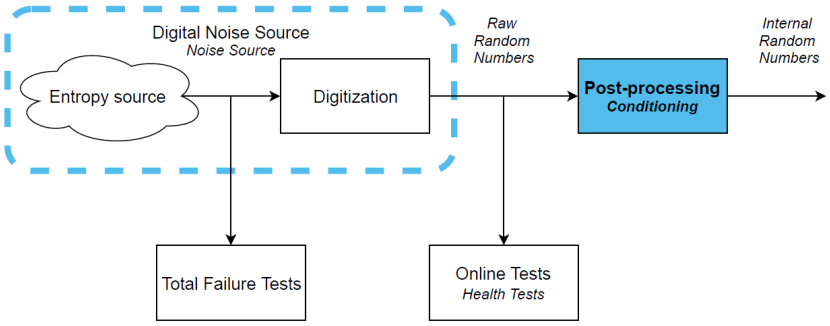
- Step 6: HW verification



	Spartan 6 FPGA		
	Slices	LUTs	FFs
Sensitive test	9	28	25
Robust test	10	26	22
Combined test	14	42	35

KU LEUVEN

Post-processing

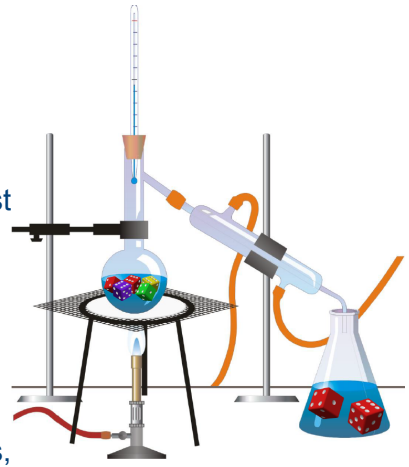


TRNG architecture

KU LEUVEN

Post-processing

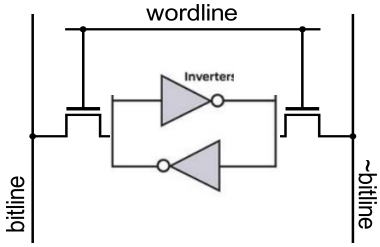
- Cryptographic post-processing
  - High implementation costs i.r.t. TRNG
  - Block ciphers & hash functions
  - Required by AIS-31 standard for the highest security level
- Arithmetic post-processing
  - Low implementation costs
  - Low latency
  - XOR, Von-Neumann debiasing, linear codes, strong blenders



KU LEUVEN

Conclusion: two ends of randomness

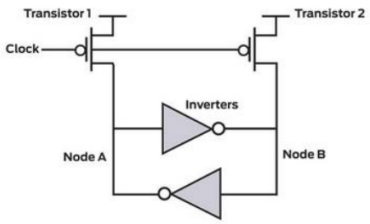
- Fixed randomness, stable over time



Physically Unclonable Function

Used: secret key
Challenge: time varying noise

- Randomness changing over time



True Random Number Generation

Used: freshness
Challenge: fixed noise

KU LEUVEN

Tenure Track or Full Professor in hardware security or applied cryptography

- KU Leuven, COSIC research group
- Research professor position
- a motivation letter, a brief CV (2 pages), a research plan (2 pages) and a publication list
- by Monday June 20 2022 (not strict deadline)
- to Saartje Verheyen (firstname.lastname@kuleuven.be).

- More information:
  - Ingrid Verbauwhede
  - Bart Preneel
  - Or any other COSIC here

KU LEUVEN

Join us

KU LEUVEN

KU LEUVEN

THANK YOU!

QUESTIONS?

Acknowledgements: All current and past PhD students,

KU LEUVEN

References

[YRG+17] B. Yang, V. Rožić, M. Grujić, N. Mentens, and I. Verbauwhede, “On-chip Jitter Measurement for True Random Number Generators, ” *AsianHOST,* 2017.

[BLMT11] M. Baudet, D. Lubicz, J. Micolod, and A. Tassiaux, “On the security of oscillator-based random number generators,” *Journal of Cryptology*, 2011.

[CFAF13] A. Cherkaoui, V. Fischer, A. Aubert, and L. Fesquet, “A self-timed ring based true random number generator,” *ASYNC*, 2013.

[FD02] V. Fischer and M. Drutarovsky, “True random number generator embedded in reconfigurable hardware,” *CHES*, 2002.

[PRV19] A. Peetermans, V. Rožić, and I. Verbauwhede, “A Highly-Portable True Random Number Generator based on Coherent Sampling,” *FPL,* 2019.

[VHKK08] I. Vasyltsov, E. Hambardzumyan, Y.S. Kim, B. Karpinskyy, “Fast Digital TRNG Based on Metastable Ring Oscillator,” *CHES*, 2008.

[CRY+16] Y. Cao, V. Rožić, B. Yang, J. Balasch, and I. Verbauwhede, "Exploring Active Manipulation Attacks on the TERO Random Number Generator," *MWSCAS*, 2016.

[VD10] M. Varchola and M. Drutarovsky, “New high entropy element for FPGA based true random number generators,” *CHES*, 2010.

[KG04] P. Kohlbrenner and K. Gaj, “An embedded true random number generator for FPGAs,” *12th International Symposium on Field Programmable Gate Arrays*, 2004.

[YRM+16] B. Yang, V. Rožić, N. Mentens, W. Dehaene, and I. Verbauwhede, “TOTAL: TRNG on-the-fly testing for attack detection using Lightweight hardware,” *DATE*, 2016.

KU LEUVEN