Random Number Generators for Cryptography

Design and Evaluation

Viktor FISCHER

Laboratoire Hubert Curien, UMR 5516 CNRS
Jean Monnet University, Member of University of Lyon
Saint-Etienne, France

fischer@univ-st-etienne.fr

Summer School on Design and Security of Cryptographic Algorithms and Devices,
Šibenik, Croatia, June 2014

Random Numbers in Cryptography

- Random numbers are crucial for cryptography, they are used as:
  - Cryptographic keys
  - Initialization vectors, nonces, padding values, ...
  - Masks in countermeasures against side channel attacks
- Since the era of Kerckhoff, confidentiality is based on cryptographic keys – algorithms and their implementation can be known by adversaries
- Consequently, cryptographic keys must fulfill stringent security requirements
  - Perfect statistical parameters
  - Unpredictability

LABORATOIRE Hubert CURIEN
UMR • CNRS • 5516 • ST-ETIENNE

Basic RNG Classes

- Deterministic (Pseudo-) random number generators (PRNG)
  - Algorithmic generators
  - Usually faster, with good statistical properties
  - Must be computationally secure, i. e. it should be computationally difficult to guess the next or previous values
  - Their period must be very long
- Physical (True-) random number generators (TRNG)
  - Using some physical source of randomness
  - Unpredictable, usually having suboptimal statistical characteristics
  - Usually slower
- Hybrid random number generators (HRNG)
  - Deterministic RNG seeded repeatedly by a physical random number generator
  - True RNG with algorithmic (e. g. cryptographic) post-processing

LABORATOIRE Hubert CURIEN
UMR • CNRS • 5516 • ST-ETIENNE

RNGs in Logic Devices

- RNGs – usually a part of a Cryptographic SoC $\Rightarrow$ in logic devices
- Logic devices (ASICs or FPGAs)
  - Aimed at implementation of deterministic systems
  - Designed so that the deterministic behavior dominates
  - Some analog blocks are sometimes available (PLL, RC-oscillator, A/D and D/A converters, etc.)

Challenge #1

Implementation of PRNGs in logic devices is straightforward ... but ...

... finding and exploiting correctly a robust physical source of randomness is a challenging task

LABORATOIRE Hubert CURIEN UMR • CNRS • 5516 • ST-ETIENNE

TRNG for Cryptography – Classical Design Strategy

- Classical TRNG design
  - Proposition of the physical principle for generating digital noise
    - Simple – occupying small area
    - Giving high bit-rate (if possible)
    - Having low power consumption
  - Enhancement of statistical parameters of the generated bitstream using arithmetic post-processing
    - Bias
    - Correlation
    - Entropy per bit
  - Evaluation of the quality by common statistical tests
    - FIPS 140-1 or FIPS 140-2 [1]
    - NIST SP 800-22
    - DIEHARD

[1] Only the first, original version of FIPS 140-2, which is not valid any more

Classical versus Modern TRNG Design Approach

- Two main security requirements on RNGs:
  - R1: Good statistical properties of the output bitstream
  - R2: Output unpredictability

- Classical approach:
  - Assess both requirements using statistical tests – difficult
- Modern ways of assessing security:
  - Evaluate statistical parameters using statistical tests
  - Evaluate entropy using entropy estimator (stochastic model)
  - Test online the source of entropy using dedicated statistical tests

Objective of the course

To show on practical examples

- Why the thorough security assessment is so important
- How the strict security requirements can be satisfied

Motto

It is quite easy to design a "TRNG" that will pass the statistical tests ...

☺

...but it is much more difficult to know where the "randomness" comes from and how much true randomness there is... [1]

☹

[1] Knowing that only the true randomness cannot be guessed or manipulated

Outline

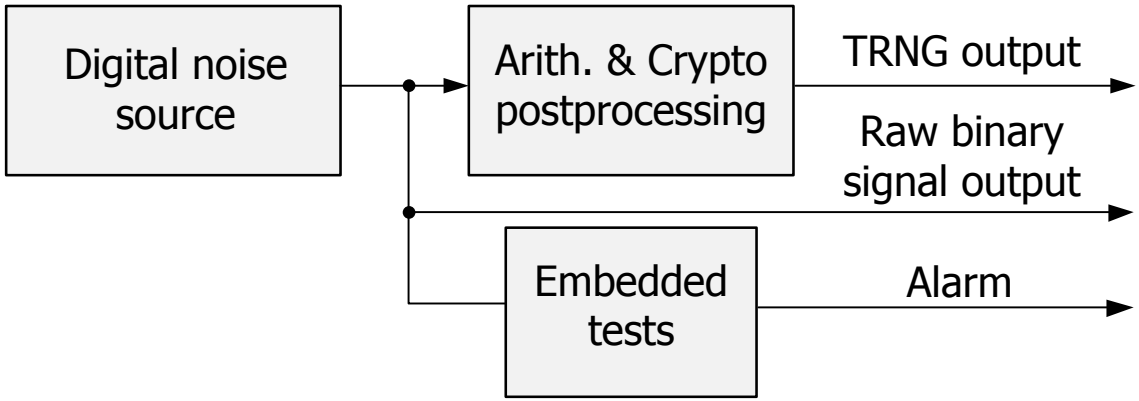
1. Contemporary TRNG design
   - Sources of randomness and entropy extraction methods
   - Post-processing methods
   - Stochastic models and entropy estimators
   - Classical and new methodology of TRNG testing
   - TRNG design and security evaluation
2. Main TRNG Classes
   - "Maximum entropy" TRNGs
   - TRNGs making entropy estimation difficult or impossible
   - TRNGs suitable for entropy estimation
3. Conclusions

LABORATOIRE Hubert CURIEN UMR • CNRS • 5516 • ST-ETIENNE

Outline

1. Contemporary TRNG design
   - Sources of randomness and entropy extraction methods
   - Post-processing methods
   - Stochastic models and entropy estimators
   - Classical and new methodology of TRNG testing
   - TRNG design and security evaluation
2. Main TRNG Classes
   - "Maximum entropy" TRNGs
   - TRNGs making entropy estimation difficult or impossible
   - TRNGs suitable for entropy estimation
3. Conclusions

TRNG Design – Recommendations AIS 31



- Source of randomness and entropy extractor
  - Should give as much entropy per bit as possible
  - Should enable sufficient bit-rate
  - Shouldn't be manipulable (robustness)
- Post-processing
  - Algorithmic – enhances statistics without reducing the entropy
  - Cryptographic – for unpredictability when source of entropy fails
- Embedded tests
  - Fast total failure test
  - Online tests detecting intolerable weaknesses

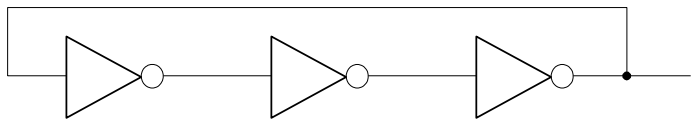
Sources of Randomness in Logic Devices

- All sources are related to some physical process
  - **Clock jitter**: short-term variation of an event from its ideal position
  - **Metastability**: ability of an unstable equilibrium electronic state to persist for an indefinite period in a digital system (rare)
  - **Chaos**: stochastic behavior of a deterministic system which exhibits sensitive dependence on initial conditions (needs analog blocks)
  - **Thermal noise**: noise developed in a resistor (or a passive component), even without electric current (needs analog blocks)

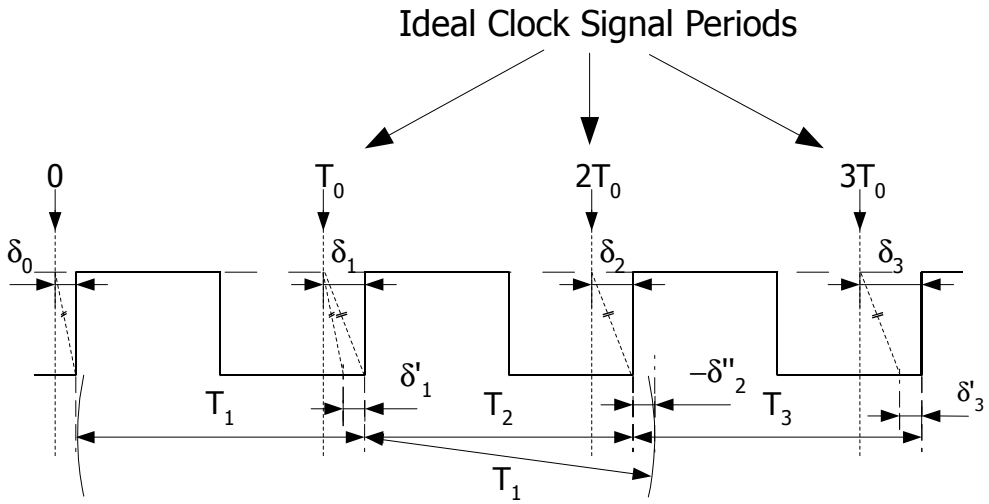
Sources of Randomness: Jittery Clock Signals 1/2

- **Clock signal**: Periodic rectangular-waveform signal controlling the timing in digital systems
- Its period varies over time, this variation can be seen as:
  - Phase noise (in frequency domain)
  - Timing jitter (in time domain) - used in digital electronics
- Common sources of the clock signal in logic devices:
  - RC oscillator (suitable for digital ICs) – unbounded jitter
  - Ring oscillator (ideal for digital ICs) – unbounded jitter
  - Voltage-controlled oscillator (limited use in digital ICs) – jitter bounded by a phase-locked loop (PLL) control
- Ring oscillator – odd number of inverters connected in a ring generating clock signal with the mean period $T = 2 \times N \times d_{inv}$

Three-element ring oscillator ($N = 3$)



Sources of Randomness: Jittery Clock Signals 2/2

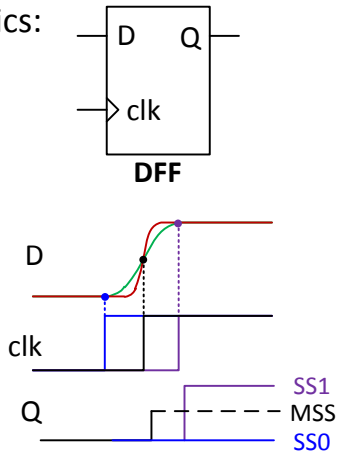


- Clock jitter – unwanted and reduced in recent digital technologies
- Measurements
  - Phase jitter - $\delta_n = t_n - nT_0$
  - Period jitter - $\delta'_n = (t_n - t_{n-1}) - T_0 = \delta_n - \delta_{n-1}$
  - Cycle-to-cycle jitter - $\delta''_n = (t_n - t_{n-1}) - (t_{n-1} - t_{n-2}) = \delta'_n - \delta'_{n-1}$
- Composition
  - Random jitter – obeys the central limit theorem (Gaussian PDF)
  - Deterministic jitter – dangerous (can potentially be manipulated)

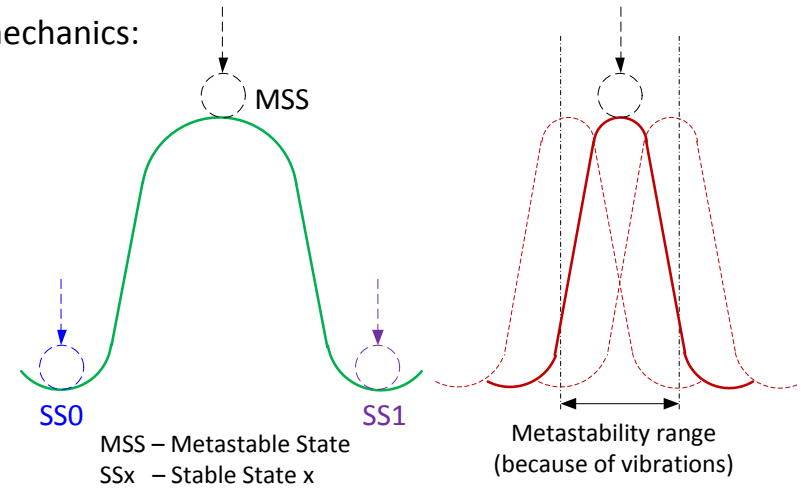
Sources of Randomness: Metastability?

Metastability:

In electronics:



In mechanics:



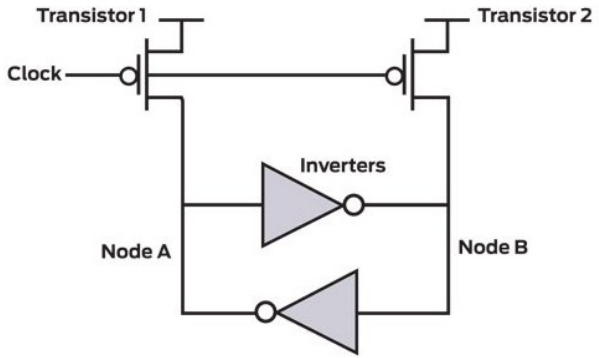
- **Definition**: Randomly lasting equilibrium of a complex system
- Dangerous in logic devices – achieved when a binary signal is sampled during its rising or falling edge
- Characterized by the mean time between failures (MTBF) $\approx$ tens of years in current IC technologies
- **Surprisingly, some TRNG designs claimed to use metastability obtain an output bitrate of several Mbits/s ...**[1]

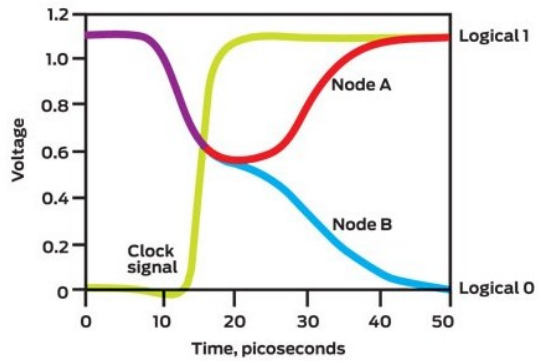
[1] M. Majzoobi et al.: FPGA-Based True Random Number Generation Using Circuit Metastability with Adaptive Feedback Control, CHES 2011

Other Sources of Randomness in Digital Devices

- Initialization of a bi-stable circuit to a random state [1]

  Intel's hardware random number generator





- Randomness in two concurrent writings to RAM memory blocks [2]
- Transitional oscillations in rings of inverters [3]

---

[1] G. Taylor, G. Cox: Behind Intel's New Random-Number Generator, http://spectrum.ieee.org

[2] T. Guneysu: True Random Number Generation in Block Memories of Reconfigurable Devices, FPT 2010

[3] M. Varchola and M. Drutarovsky: New High Entropy Element for FPGA Based True Random Number Generators, CHES 2010

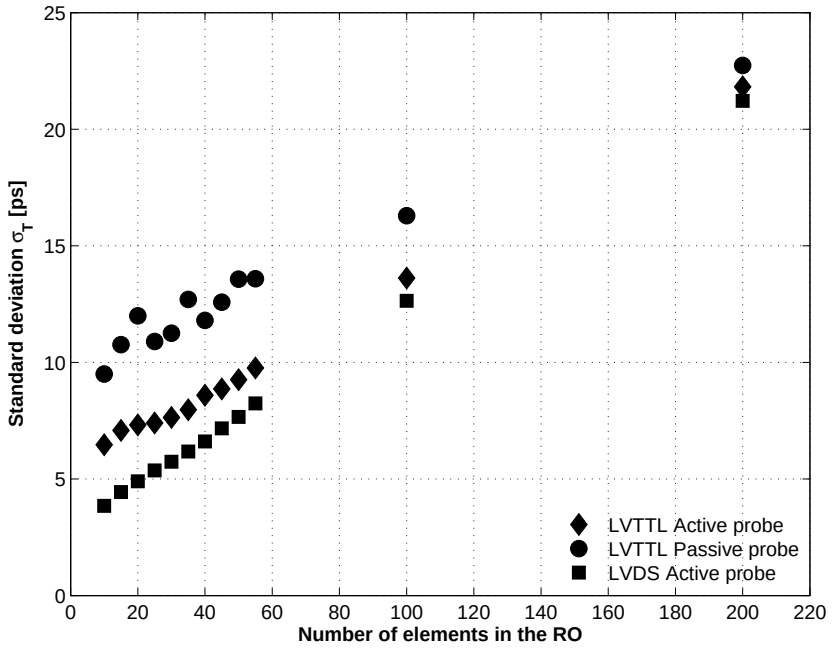
Choice of the Source of Randomness

- The source of randomness must be clearly defined and well quantified with respect to the entropy extraction method
- Perfect example of what should be avoided:
  While claiming to use metastability, the designer uses some other, uncharacterized source of entropy

Challenge #2

To define and characterize the physical process that is INDEED used as a source of randomness

External Methods of Randomness Quantification – A Pitfall



Measurement setup

- Oscilloscope LeCroy WavePro 7300
- Standard passive 500 MHz and differential active 3.5 GHz probes
- Standard and LVDS outputs used

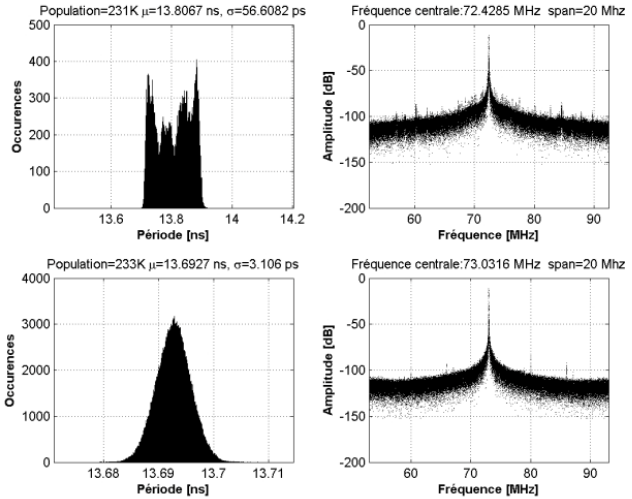
Results and conclusion

- Jitter measured using LVDS outputs and differential probe – two times smaller than that using common IOs and probes (!)
- Jitter measured using standard outputs and probes is significantly overestimated

Question

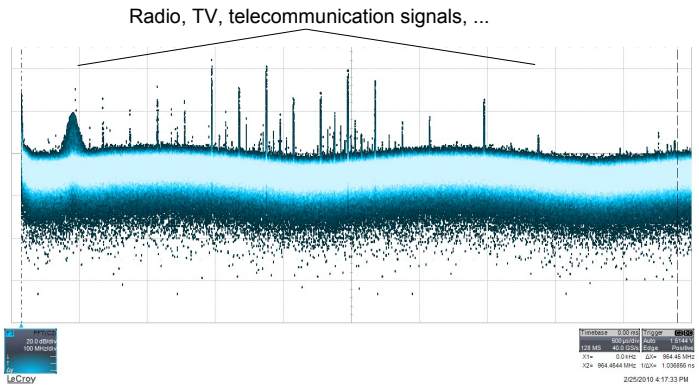
- What is the real jitter inside the device?

External and Internal Signals Affect Randomness Sources



RO clock period histogram and clock spectrum

- Upper panel:
  - RO near AES cipher
- Lower panel:
  - RO alone in the same chip



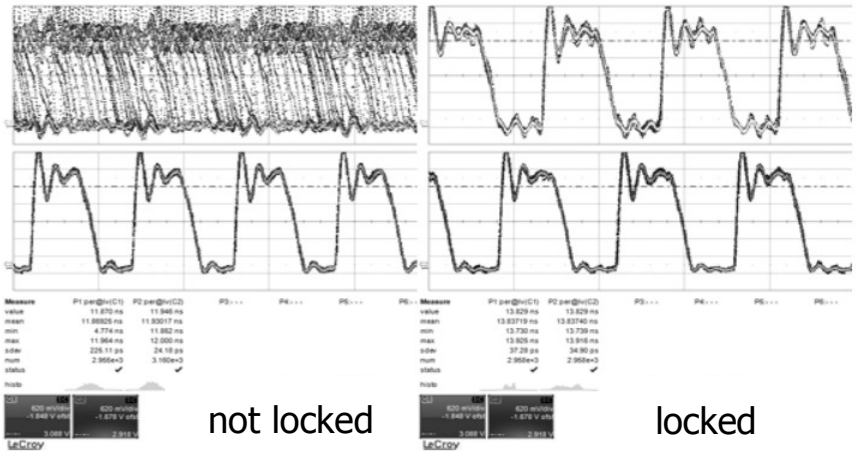
DC Power Supply Spectrum

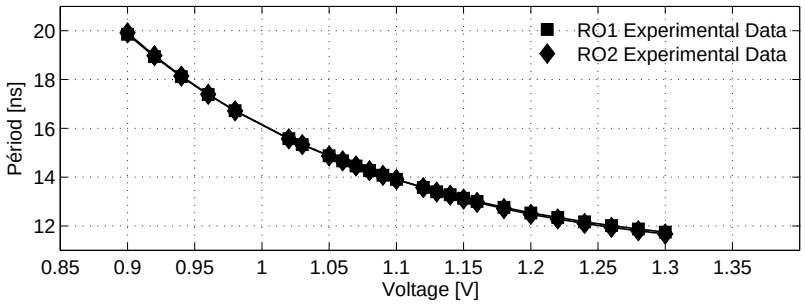
- Electromagnetic waves captured by hardware increase electric noise

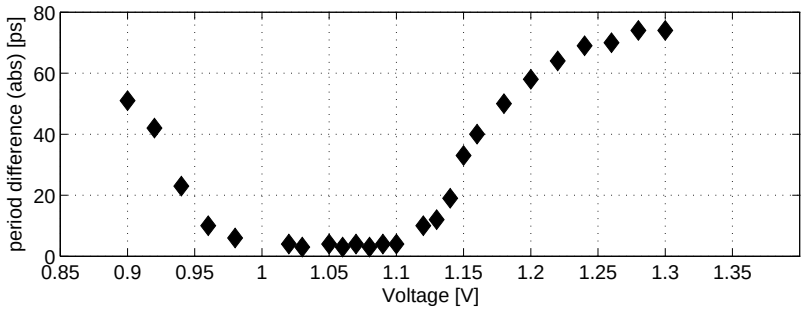
Challenge #3

- Estimate and reduce impact of the environment on the generator

Mutual Dependence of Ring Oscillator Frequencies







Testing conditions

- Two similar ROs are implemented inside the FPGA,
- Frequencies are measured outside the FPGA,
- The power supply varies between 0.9 and 1.3 V.

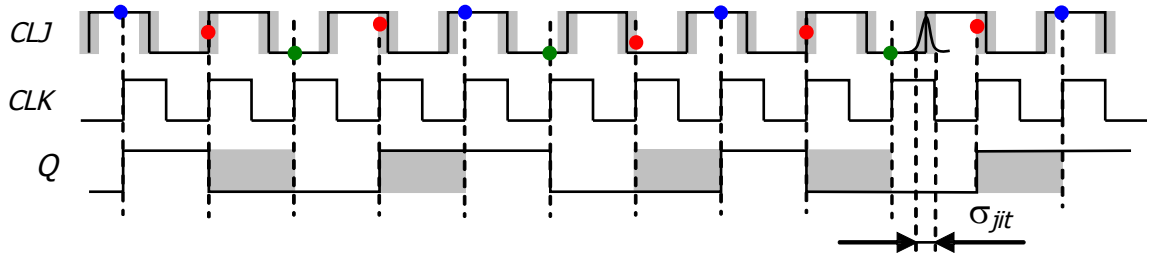
Results

- Frequencies approach and lock to the same value during some voltage interval.

Randomness Extraction from the Clock Jitter

Principle:

Sampling of a jittery clock signal (CLJ) on the rising edge of the reference clock signal (CLK) using DFFs or latches



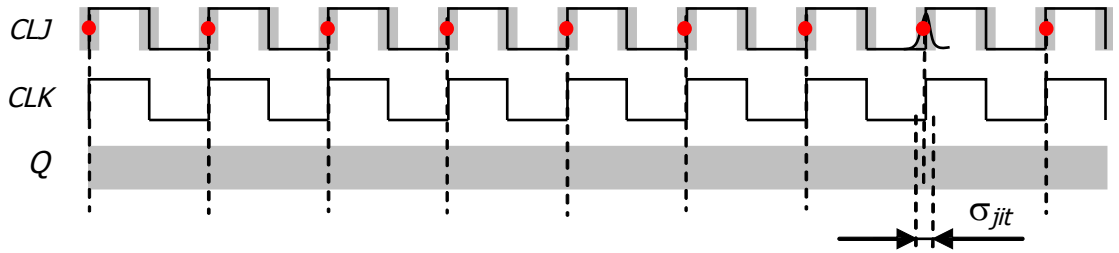
- Depending on the frequency and phase relationship, some samples (signal Q) can be:
  - Equal to one (blue samples) or zero (green samples)
  - Equal to one or to zero depending on the jitter (red samples)
- **Number of red samples determines the output entropy**

Challenge #4

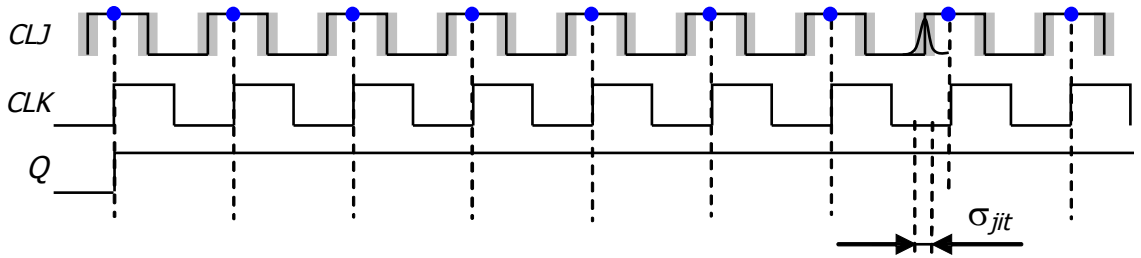
- To find a RELIABLE method for extracting maximum entropy from the existing source

Extreme Cases in Entropy Extraction by Clock Sampling

- The entropy depends
  - On the size of the (random) jitter
  - On the spectrum of the jitter
  - On the clock frequencies and their initial phase
- Maximum entropy – equal frequencies and zero phase difference
  => Each sample is influenced by the jitter



- Minimum entropy – equal frequencies and phase difference bigger than the jitter size => No sample is influenced by the jitter!



Post-processing Methods

- Enhance statistical and security characteristics of the TRNG
- Main statistical parameters
  - Bias of the probability of ones (from the ideal value – 1/2)
  - Auto-correlation of the TRNG output
  - Entropy per bit (can be increased when reducing the bit rate)
- Main security objectives
  - Even if the source of randomness fails, next and previous values should not be guessable
  - Internal memory of the post-processing algorithm should maintain some entropy, before the total failure test will trigger alarm

Remarks

- The statistical post-processing method shouldn't decrease entropy per bit
- The cryptographic post-processing method must be cryptographically sure

Stochastic Models – Objectives

- Main objectives – characterize:
  - Probability of ones: $P(X = 1)$
  - Probability of an n-bit pattern: $P(X_1 = x_1, X_2 = x_2, ..., X_n = x_n,)$
  - Entropy and so-called conditional entropy
- Bias of the output bit-stream: $P(X = 1) - 0.5$
  - AIS31: smaller than 0.0173 for the raw binary signal
  - Can be easily reduced for uncorrelated random variables (post-processing)
- Entropy – gives the uncertainty contained in an information unit
  - Shannon entropy for "iid" random variables from a finite set Ω:

$$H(X) = -\sum_{x \in \Omega} P(X = x) log_2 P(X = x)$$

  - The entropy per bit of a TRNG should be close to 1 (according to AIS31, $H(X) > 0.997$)
  - High entropy rate guarantees that the preceding or succeeding bits cannot be guessed with a probability different from 0.5
  - Property of random variables and not of observed realizations - it cannot be measured, just **estimated using the model**

Evaluation of the TRNG Using General Statistical Tests

- Classical approach: various general-purpose statistical tests are applied on the generator output
- FIPS140-1 and FIPS140-2 tests [1]
  - 4 tests (Monobit, Poker, Runs, Long runs) applied on bit-streams of 20000 bits
  - The thresholds are different in FIPS 140-1 and FIPS 140-2
  - Tests not included in the latest version of the standard FIPS 140-2
- NIST 800-22 tests [2]
  - 15 statistical tests with given testing strategy
  - About 1 Gbit of random data needed
- DIEHARD tests [3]
  - 15 statistical tests with testing strategy similar to NIST tests
  - At least 80 million bits needed

[1] Federal Information Processing Standard FIPS140-2: Security Requirements for Cryptographic Modules, NIST 2001

[2] A. Rukhin et al.: A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, NIST Special Publication 800-22 rev1a, 2010

[3] G. Marsaglia, DIEHARD: Battery of Tests of Randomness, 1996

AIS31 Testing Methodology Adapted for Physical RNG

- Eight statistical tests have been proposed to be used at different levels of the TRNG evaluation
  - Tests applied on generated random numbers
    - T0 – Disjointness test (2^{16} 48-bit random blocks must be different), rejection probability for an ideal random source: 10^{-17}
    - T1 – T4 – Four tests from FIPS140-1 (not from FIPS140-2!) with rejection probability limit 10^{-6}
    - T5 – Autocorrelation test
  - Tests applied on the raw binary signal in class PTG.2 and PTG.3 (some weaknesses are tolerable)
    - T6 – Uniform distribution test
    - T7 – Comparative test for multinomial distribution
    - T8 – Coron's entropy test [1]
- AIS 31 testing strategy is clearly defined (how much data, how many test repetitions, how many rejections allowed)

[1] J.-S. Coron: On the Security of Random Sources, Gemplus, Technical Report IT02-1998

Security Threat in TRNG Testing

- Paradox of embedded tests
  - Paradox: implementation of embedded tests (FIPS, NIST, etc.) inside the device, as in [1] and [2]
  - Problem: authors DO NOT consider the impact of the tests on the TRNG
  - Consequences:
    - Tests generate a digital noise – the TRNG output passes tests more easily
    - During the normal operation (testing is stopped), the effective noise could be much smaller and the TRNG would not pass the tests
- Solutions:
  - Authors should ensure that the tests do not have ANY impact on the generator – difficult
  - ... tests should never stop running!

[1] R. Santoro et al.: On-line Monitoring of Random Number Generators for Embedded Security, ISCAS 2009

[2] F. Veljkovic et al.: Low-Cost Implementations of On-the-Fly Tests for Random Number Generators, DATE 2012

TRNG Design Evaluation Criteria

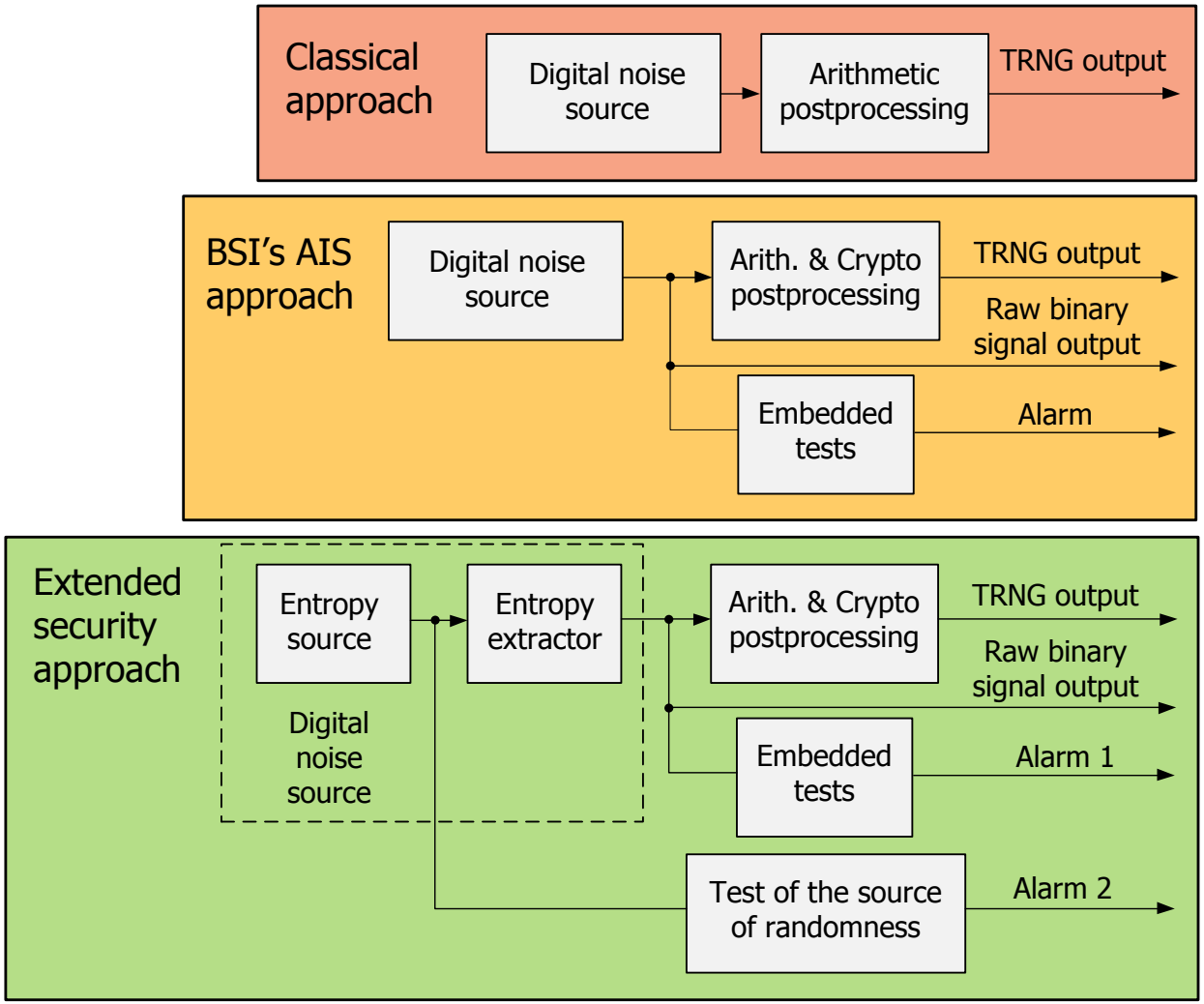
- Resource usage
  - Type and quantity of necessary resources
  - FPGA technology is more restrictive than ASIC
- Speed
  - Bit-rate
  - Regularity of the speed
- Power consumption
  - Depending on the principle and the clock frequency
  - Possibility of stopping the generator
- Feasibility in selected technology
  - Available logic and routing resources
- Design automation
  - Manual intervention (P/R) is needed for each device individually
  - Manual intervention is needed for each device package and/or family
  - Completely automated – no manual intervention is needed

TRNG Security Evaluation Criteria

- Robustness, resistance against attacks
  - No way to decrease **entropy** under a given **minimum bound**
  - Three possibilities exist
    - A proof of robustness against ALL attacks exist
    - Neither proof nor attack exist
    - Some attack on a particular generator has been reported
- Existence of a statistical model
  - **Stochastic model**: quantifies lower entropy bound depending on
    - Random input variables (source of randomness)
    - Generator principle (randomness extraction)
  - Stochastic models are different from physical models describing the origin of a physical phenomenon
  - The stochastic models must describe only the random process that is actually used as a source of randomness
- Inner testability
  - **Inner testability**: The raw binary signal must be available
  - **Absolute inner testability**: The raw binary signal must be available and must not contain a pseudo random pattern

TRNG Design – Conclusion

- TRNG designs should continue to evolve towards security:



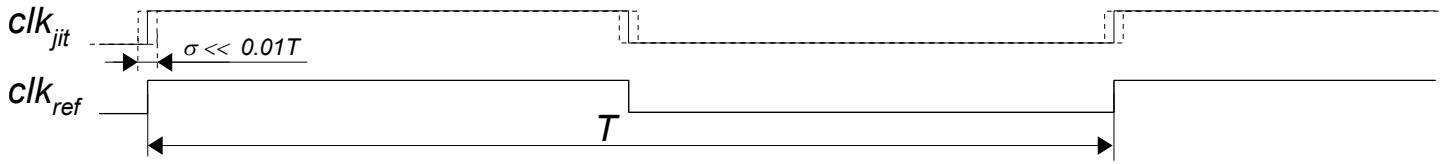
Outline

1. Contemporary TRNG design
   - Sources of randomness and entropy extraction methods
   - Post-processing methods
   - Stochastic models and entropy estimators
   - Classical and new methodology of TRNG testing
   - TRNG design and security evaluation
2. Main TRNG Classes
   - "Maximum entropy" TRNGs
   - TRNGs making entropy estimation difficult or impossible
   - TRNGs suitable for entropy estimation
3. Conclusions

"Maximum Entropy" True Random Number Generators

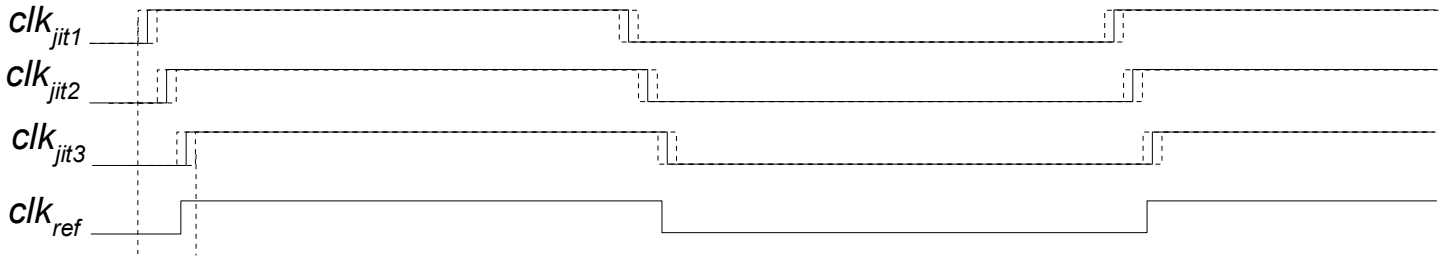
- Principle:

  Two clocks: the same frequency, "zero" phase difference

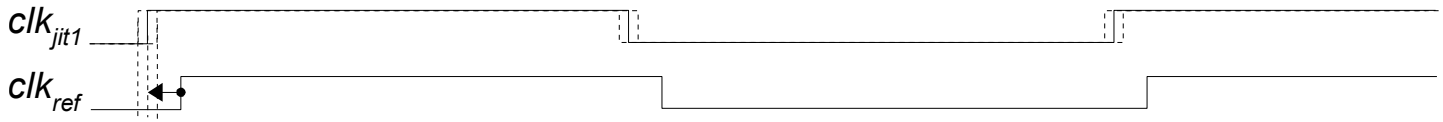


- Tolerance to a "non-zero" phase difference can be obtained in two ways:

  Several slightly delayed jittery clock signals are used

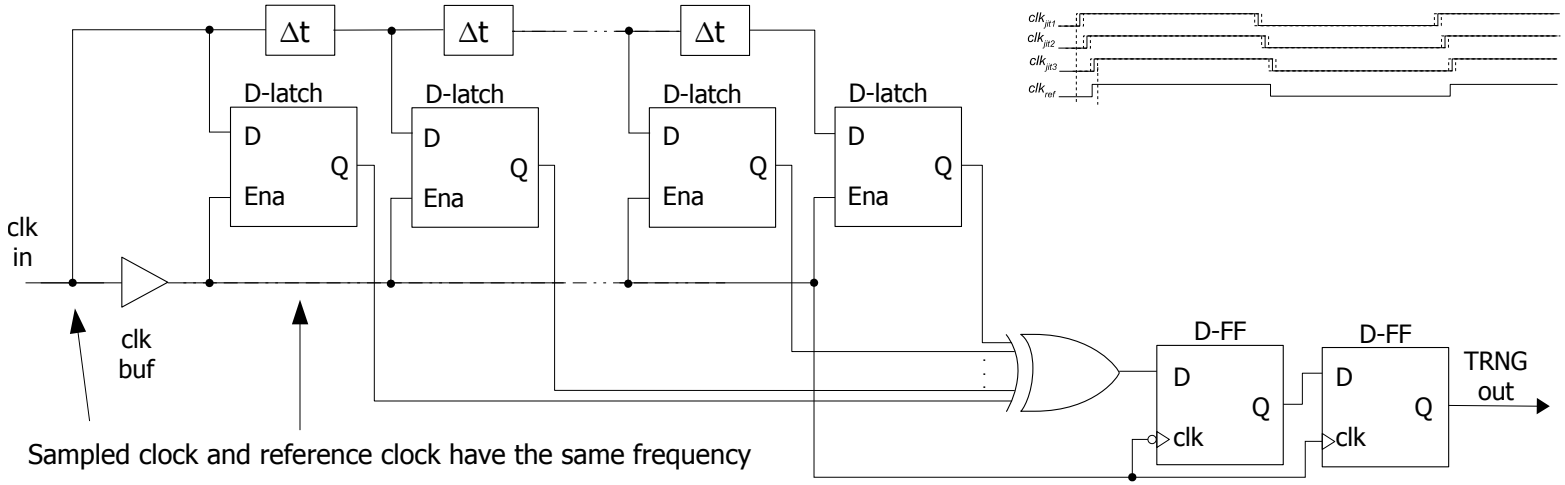


  The mutual phase is adjusted dynamically to zero



Example 1: Open-loop TRNG – "OLOOP-TRNG"

- Generator claimed to use metastability [1]
- Many slightly delayed signals are used



- Delays must be smaller than the jitter (overlapped jittery zones)
- Jittery clocks are sampled using latches and not flip-flops!

[1] J.-L. Danger, S. Guilley, P. Hoogvorst: High Speed True Random Number Generator Based on Open Loop Structures in FPGAs, Elsevier, Microelectronics Journal, 2009

OLOOP-TRNG – Assessment

- Resource usage
  - Small area ($\approx$ 120 FPGA logic cells)
  - Common elements: LUTs, latches and DFFs
  - Critical point: Delay elements (featuring very small delays, $\approx$ ps)
- Speed
  - High and regular speed ($\approx$ 20 Mb/s)
- Power consumption
  - Considering the speed, could be relatively low (not given)
- Feasibility in logic devices
  - Not feasible if delays cannot be sufficiently small
- Design automation
  - Per family (if feasible)

Security Assessment

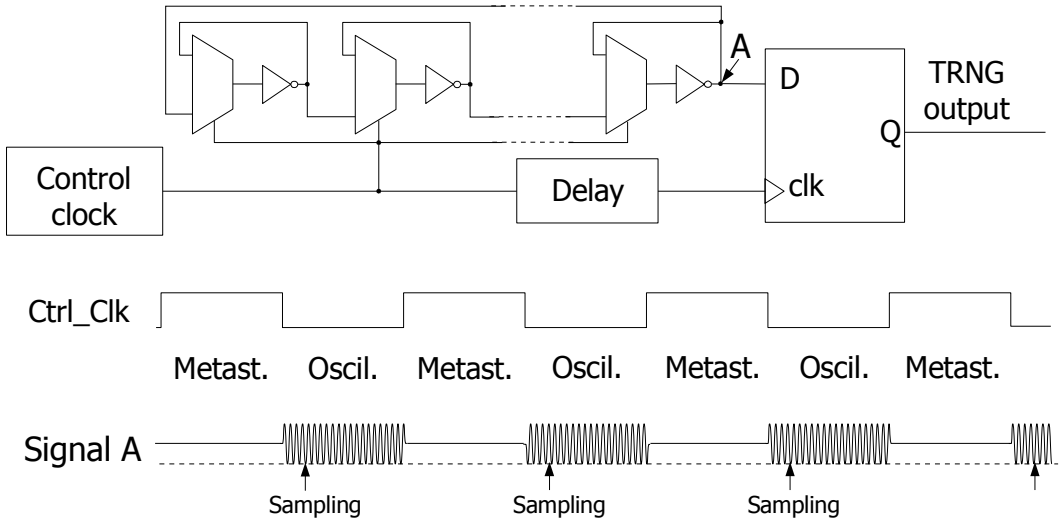
- Difficult to create a model (unknown delays)
- Impossible to test in real time (too many signals)
- Critical point: delays depend on the temperature

TRNGs Making Entropy Estimation Difficult or Impossible

- Generators using randomness in initialization of flip-flops, memories, "metastable" structures, etc.
- Group of generators mixing pseudo-randomness and true randomness before entropy extraction

Example 2: TRNG Using Metastable RO – "MERO-TRNG"

- Yet another generator claimed to use metastability [1]
- Inverters of the ring oscillator (RO) are put periodically to a "metastable" state
  - The phase after the "metastable" state is unknown (randomness)



- In reality, the metastable state is very difficult to obtain

[1] I. Vasyltsov, E. Hambardzumyan, Y .S. Kim, B. Karpinskyy: Fast Digital TRNG Based on Metastable Ring Oscillator, CHES 2008

MERO-TRNG – Assessment

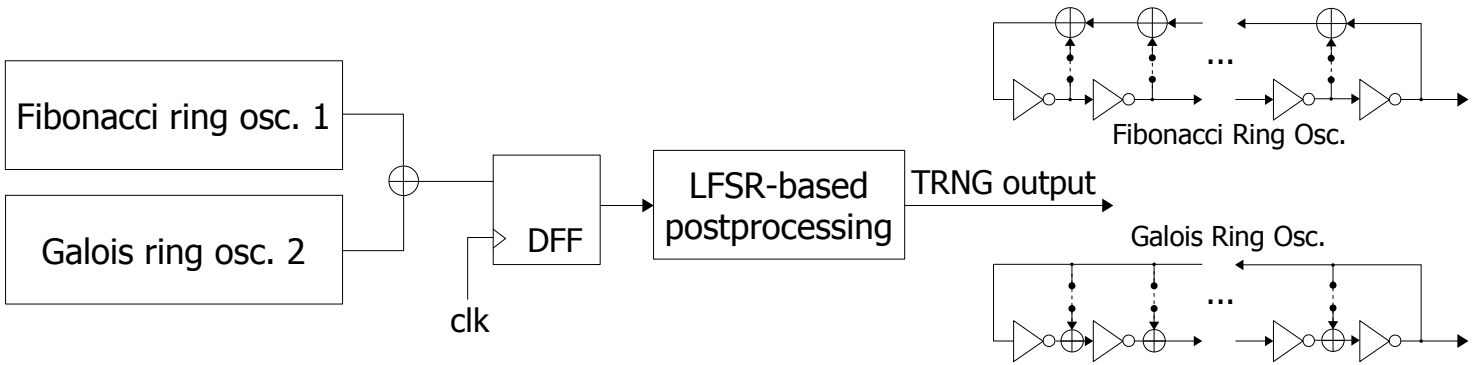
- Resource usage
  - Small area ($\approx$ tens of FPGA logic cells)
  - Common elements: MUX, inverters, control logic
  - Critical point: Setting up inverters into metastable states
- Speed
  - Relatively high and regular speed ($\approx$ 10 Mb/s)
- Power consumption
  - Could be relatively low (not given)
- Feasibility in logic devices
  - Should be feasible in logic devices, but more difficult in FPGAs
- Design automation
  - Per family (if feasible)

Security Assessment

- Impossible to create a model (unknown distribution of initial states)
- Impossible to test initial states in real time
- Critical point: initial states can (will) depend on the temperature

Example 3: TRNG Using Fibonacci and Galois RO – "FIGARO-TRNG"

- Original idea: replace registers in Fibonacci and Galois LFSR by inverters [1]
- The two ring oscillators should give noisy signals having a uniform spectrum (white noise)



- In reality, some frequencies dominate
- Another problem observed: the generator sometimes stalls

[1] J. Golic: New Methods for Digital Generation and Post-processing of Random Data. IEEE TC 55(10), 2006

FIGARO-TRNG – Assessment

- Resource usage
  - Small area ($\approx$ hundreds of FPGA logic cells)
  - Common elements: XOR gates, inverters, registers
  - Manual routing of both ring oscillators is necessary
- Speed
  - High speed depending on the noisy signal spectrum ($\approx$ 10 Mb/s)
- Power consumption
  - Relatively high and local (not given)
- Feasibility in logic devices
  - Should be feasible in logic devices, but more difficult in FPGAs
- Design automation
  - Needs manual routing for each device family

Security Assessment

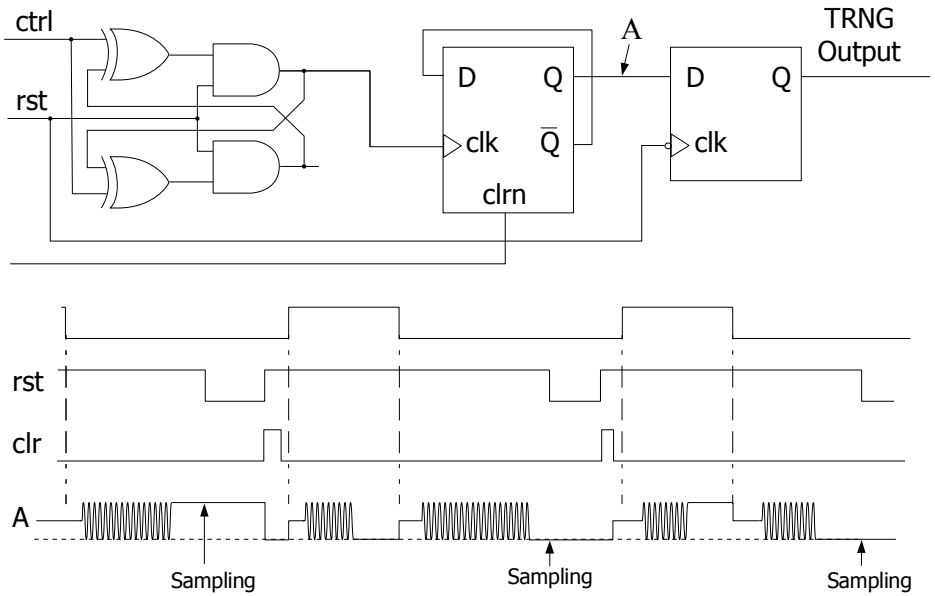
- Impossible to create a model (pseudo-randomness)
- Impossible to test (some modification proposed by Dichtl)
- Critical point: they can generate patterns and stall

TRNGs Suitable for Entropy Estimation

- Generators with transitional oscillatory state
- Multiphase sampling TRNGs with minimum entropy estimation
  - TRNGs with randomly distributed clock phases
  - TRNGs with periodically occurring clock phases (coherent sampling)

Ex. 4: Transition Effect RO-based TRNG – "TERO TRNG"

- Original idea: a bi-stable logic structure can be initialized into an oscillatory state of random duration [1] (similar patented by Dichtl)
- Duration of oscillations depends on the symmetry of the structure
- The noise dynamically changes the delays



- Difficulty: oscillations shouldn't be too short (small entropy) nor too long (no entropy)
- Problem: some cells oscillate infinitely without explication ...

[1] M. Varchola, M. Drutarovsky: New High Entropy Element for FPGA Based True Random Number Generators, CHES 2010

TERO-TRNG – Assessment

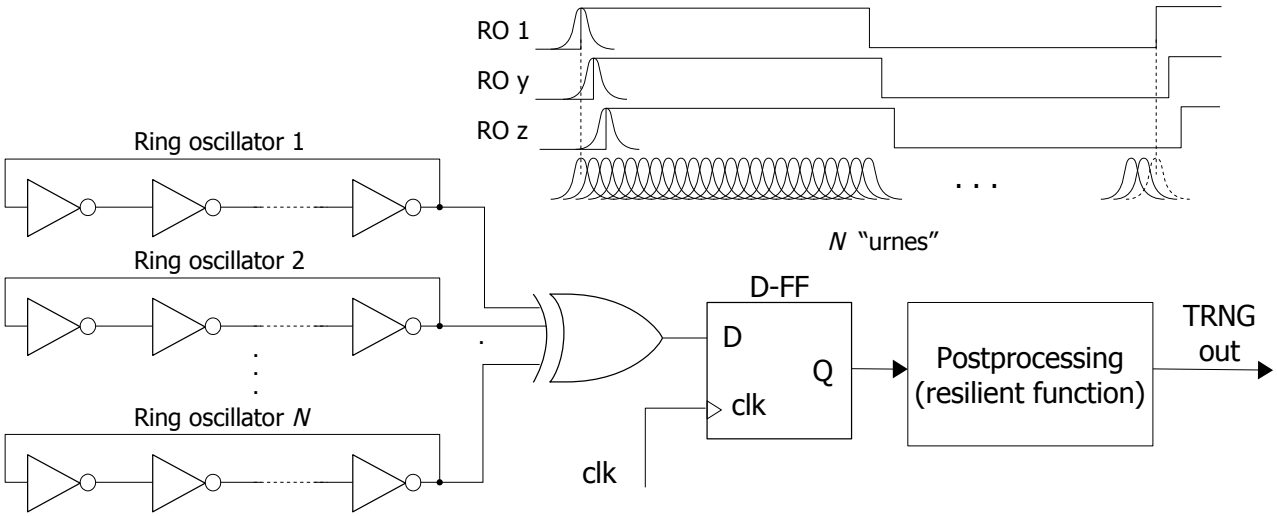
- Resource usage
  - Very small area (only a few FPGA logic cells per TERO core)
  - Common elements: XOR and AND gates, registers
- Speed
  - Very high speed depending on number of TERO cells ($\approx$ 250 kb/s per one TERO cell)
- Power consumption
  - Relatively high and local (not given)
- Feasibility in logic devices
  - Feasible in logic devices including FPGAs
- Design automation
  - Needs manual routing for each device family

Security Assessment

- Statistical model can be easily created
- TRNG-specific tests can be easily implemented
- Critical point: unknown reason for infinite oscillations

Example 5: Multiple Ring Oscillator TRNG – "MURO TRNG"

- Source of randomness – jitter of clocks generated in multiple ROs
- Generated clock period T divided to N "urns" depending on jitter size – N rings are needed (114 in a given example)
- Rings are supposed to be independent – urns are distributed uniformly across T



- The generator has been "proven to be secure" [1] for N sufficiently large

[1] B. Sunar, W. J. Martin, D. R. Stinson: A Provably Secure True Random Number Generator with Built-in Tolerance to Active Attacks, IEEE TC 2007

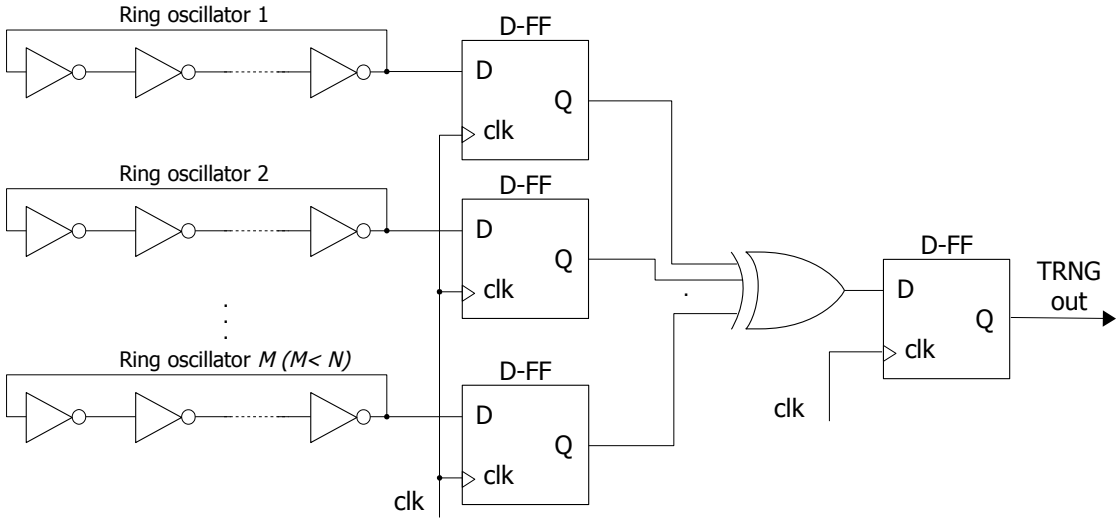
Sunar *et al.*'s Approach

- Good approach...
  1. Mathematical model (Urn model)
  2. Entropy estimators based on jitter size
  3. Post-processing using resilient functions

- But... unrealistic assumptions (Dichtl & Golic, Wold & Tan, . . .):
  1. Jitter size determined by external measurements
  2. Too many transitions in the XOR tree
  3. Setup and Hold time violation in the D-Flip Flop
  4. (In)dependence between ROs (coupling).

Improvement of Sunar *et al.*'s Principle

- Wold and Tan added flip-flops at outputs of ROs[1]



- Problem with transitions in the XOR tree solved ⇒ undeniable improvement!
- Conclusions of Wold and Tan:
  1. 114 ROs are not needed because TRNG output passes statistical tests for configurations with 50 and even with only 25 ROs
  2. Post-processing not necessary anymore
  3. Lower cost and power consumption, because less ROs are used

[1] K. Wold, C. H. Tan: Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings, IJRC 2009

Important Remarks Concerning MURO-TRNG

- Wold and Tan: number of ROs reduced from 114 down to 50 or 25 because outputs passed the tests
- Mathematical problem: according to the urn model of Sunar, not enough entropy
- Our experimental result: simulation outputs WITHOUT jitter (= WITHOUT randomness) pass tests starting from 18 rings

Remark 1

Sunar's original principle (and Wold's improvements too) produce a huge amount of pseudo-randomness that can be predicted (mathematical equation) or manipulated from outside the chip (see last attacks of Markettos *et al.* and Bayon *et al.*)

Remark 2

Reducing the number of ROs (as proposed by Wold and Tan) represents a security-critical attempt for cryptographic applications and should be certainly avoided

MURO-TRNG – Assessment

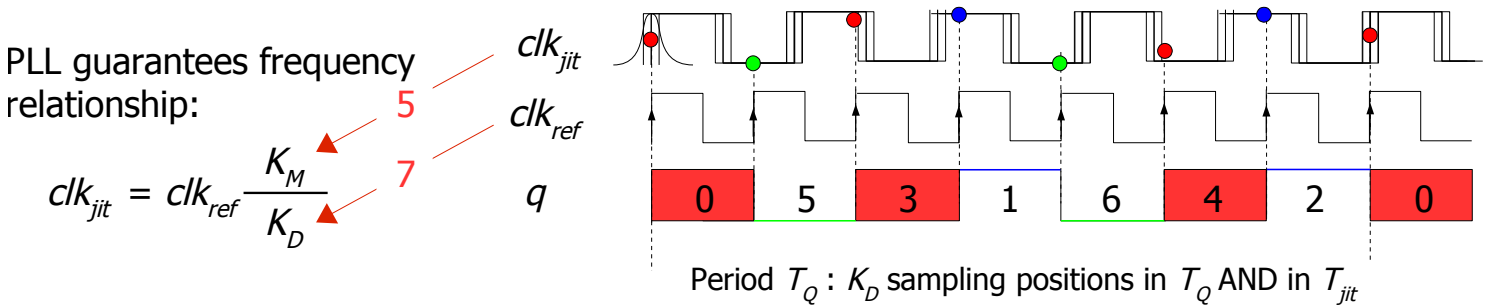
- Resource usage
  - Relatively big area (many urns)
  - Common elements: inverters (for ROs), registers
- Speed
  - Medium speed (after necessary post-processing)
- Power consumption
  - Relatively high (not given)
- Feasibility in logic devices
  - Feasible in logic devices in general (including FPGAs)
- Design automation
  - Needs manual routing in order to avoid locking of ROs

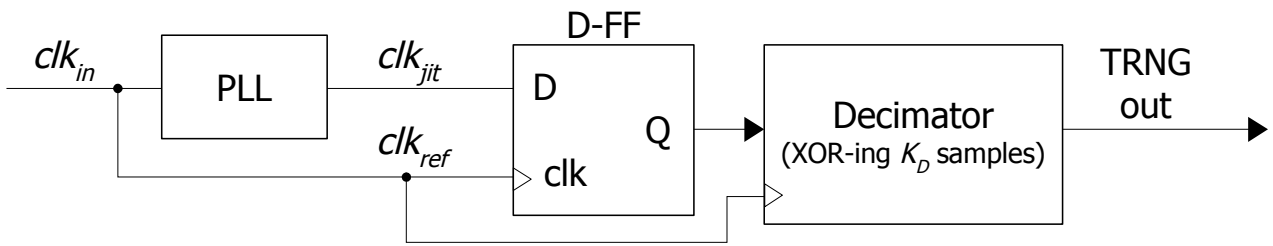
Security Assessment

- Statistical model assumptions must be verified
- TRNG-specific tests cannot be implemented
- Critical point: oscillators can lock and reduce entropy to zero!

Example 6: PLL-based TRNG – "PLL-TRNG" 1/2

- Principle[1]: PLL-based coherent sampling
- Source of randomness: tracking jitter of the PLL (bounded)





- K_M and K_D must be relatively prime, K_D should be odd

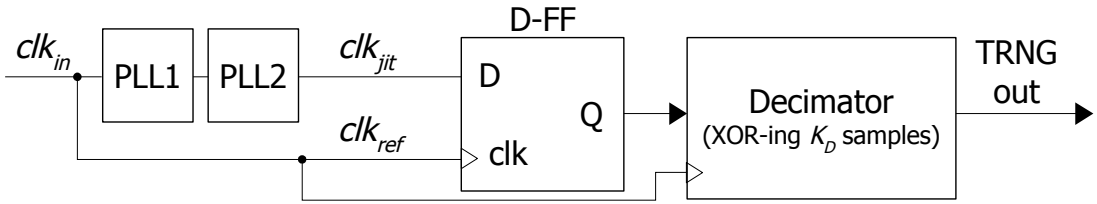
[1] V. Fischer and M. Drutarovsky: True Random Number Generator Embedded in Reconfigurable Hardware, CHES 2002

PLL-based TRNG – "PLL-TRNG" 2/2

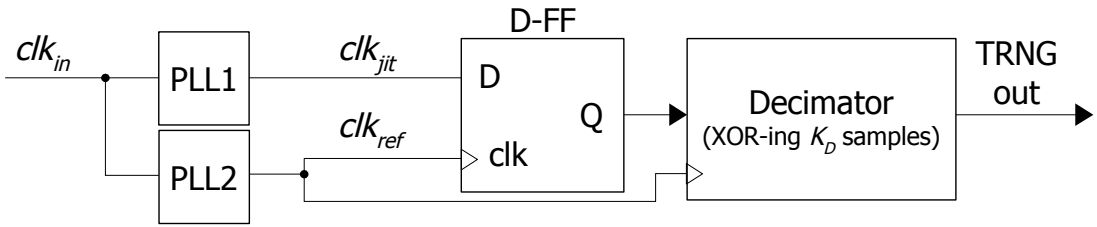
- TRNG output bitrate: $R = T_Q^{-1} = f_{ref}/K_D$
- Sensitivity to jitter: $S = \Delta^{-1} = K_D/T_{jit}$
- Conclusions:
  - For increasing R and S, f_{ref} should be as high as possible
  - For increasing R, K_D should be as small as possible
  - For increasing S, K_M should be as big as possible
- Two PLLs can be used for increasing the bitrate and sensitivity to jitter:

$$clk_{jit} = clk_{ref}\frac{K_M}{K_D}$$

$$K_M = K_{M1} \cdot K_{M2}$$
$$K_D = K_{D1} \cdot K_{D2}$$



$$K_M = K_{M1} \cdot K_{D2}$$
$$K_D = K_{D1} \cdot K_{M2}$$



PLL-TRNG – Assessment

- Resource usage
  - Small area ($\approx$ tens of FPGA logic cells)
  - PLLs + Common elements: XOR gates, registers, counters
  - Critical point: PLLs not available in all technologies
- Speed
  - Relatively high speed depending on PLL parameters ($\approx$ 1 Mb/s)
- Power consumption
  - Essentially given by PLL (can be stopped in Actel, not in Altera)
- Feasibility in logic devices
  - If PLL available, no problems in many configurations
- Design automation
  - PLL settings must be done manually, routing fully automatic

Security Assessment

- Easy to model
- Easy to test (absolutely internally testable)
- PLL often physically isolated from the rest of device – advantage

Outline

1 Contemporary TRNG design
- Sources of randomness and entropy extraction methods
- Post-processing methods
- Stochastic models and entropy estimators
- Classical and new methodology of TRNG testing
- TRNG design and security evaluation

2 Main TRNG Classes
- "Maximum entropy" TRNGs
- TRNGs making entropy estimation difficult or impossible
- TRNGs suitable for entropy estimation

3 Conclusions

Conclusions

- Designing robust generators giving high-quality true random numbers in logic devices **remains a challenge**
- We explained that security parameters like robustness, availability of a stochastic model, testability, etc. **always take priority** in a data security system
- **Statistical tests** – necessary BUT insufficient
- **Entropy** cannot be measured, only estimated from the model
- Testing the source of entropy before entropy extraction **increases security**

Random Number Generators for Cryptography

Design and Evaluation

Viktor FISCHER

Laboratoire Hubert Curien, UMR 5516 CNRS
Jean Monnet University, Member of University of Lyon
Saint-Etienne, France

fischer@univ-st-etienne.fr

Summer School on Design and Security of Cryptographic Algorithms and Devices,
Šibenik, Croatia, June 2014