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

Implementation of True Random Number Generators in (Reconfigurable) Logic Devices

State of the Art

Viktor Fischer, Alain Aubert, Nathalie Bochard
(fischer,alain.aubert, Nathalie.Bochard)@univ-st-etienne.fr
Laboratoire Hubert Curien
UMR 5516 CNRS Université Jean Monnet, Saint-Etienne, France

Outline

- Introduction
- Classification of TRNGs
- Analysis and comparison of basic TRNG principles
- Jitter (in ring oscillators) as a source of randomness – some important remarks
- Conclusions

- Appendix – beware of ring oscillators in FPGAs

Introduction (1/2)

- Use of RNGs in cryptography
  - Generation of cryptographic keys (symmetric, public, private) – special security requirements
  - Generation of initialization vectors
  - Generation of nonces
  - Generation of padding values
  - Counter-measures against side-channel attacks
- Required characteristics of RNGs in cryptography
  - Good statistical parameters of the output numbers
  - Unpredictability of the output
  - Testability
  - (Provable) security – robustness, resistance against attacks

Introduction (2/2)

- Basic RNG classes used in cryptography
  - Deterministic (pseudo-) random number generators (PRNG)
    - Algorithmic generators
    - Usually faster, with good statistical parameters
    - Must be computationally secure (e.g. encrypted binary sequences)
  - Physical random number generators (TRNG)
    - Using some physical source of randomness
    - Unpredictable, usually worth statistical parameters
    - Usually slower
  - Hybrid random number generators
    - Deterministic RNG seeded repeatedly by a physical random number generator output

Characterization of TRNGs (1/6)

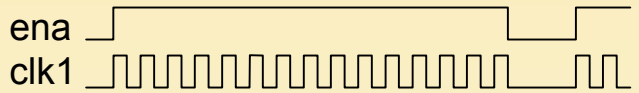
- Source of randomness employed
- Randomness extraction principle used
- Post-processing method applied (optional)
- Output rate
- Power consumption
- Testability
- Existence of a mathematical model
- Security (robustness, resistance against attacks)
- Feasibility in a (Cryptographic) System-on-Chip
  - Mixed-signal ICs
  - Logic devices
  - FPGAs

Characterization of TRNGs (2/6)

- Source of randomness – determines the available entropy
- Sources in digital devices
  - Frequency/phase instability of the clock signals
    - Free-running oscillators (RC oscillators)
    - PLL- and DLL-based clock generators
  - Delay instability of logic elements
    - Ring oscillators
    - Delay lines
  - "Analogue" features of flip-flops
    - Metastability

Characterization of TRNGs (3/6)

- Randomness extraction
  - Using some mechanism that collects as much entropy as possible to give a digital output
  - Needs justification of the method employed
- In logic devices - output sampling of:
  - Free-running oscillators
  - Gated free-running oscillators



  - Frequency-related clocks $f_2 = k \cdot f_1$

Characterization of TRNGs (4/6)

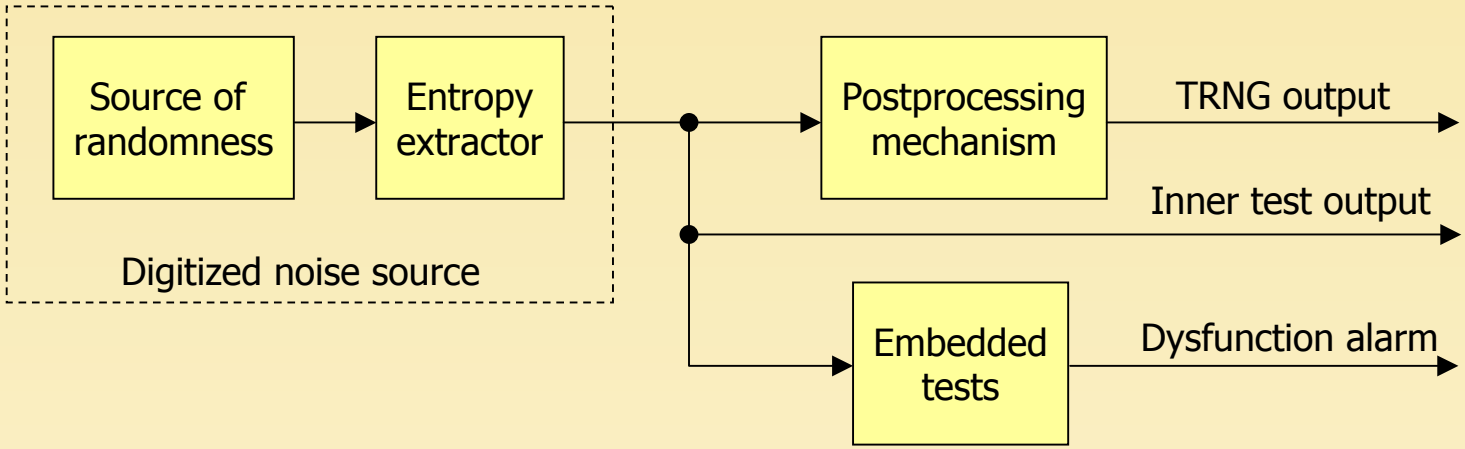
- Post-processing - optional
  - Used to mask imperfections (if any) in the entropy source or in the entropy extraction mechanism
  - May provide resistance to environmental variations and tampering
- Examples of methods commonly used
  - XOR corrector
  - Von Neumann's corrector
  - PRNGs (LFSRs)
  - Hashing functions (SHA-1)
  - Encryption of the generator output
  - Resilient functions

Characterization of TRNGs (5/6)

- Testing TRNG output
  - Tests designed for PRNGs (NIST Test Suite, DIEHARD) are commonly used → the entropy is not evaluated
  - If these tests do not pass, the sampling frequency can be reduced or post-processing methods can be used
  - New tests were proposed specifically for TRNGs by BSI (AIS31) – entropy testing
  - New paradigm: the designers should provide
    - Mathematical model of the principle (if it exists) and/or
    - Generator-specific test (before post-processing) and testing methodology

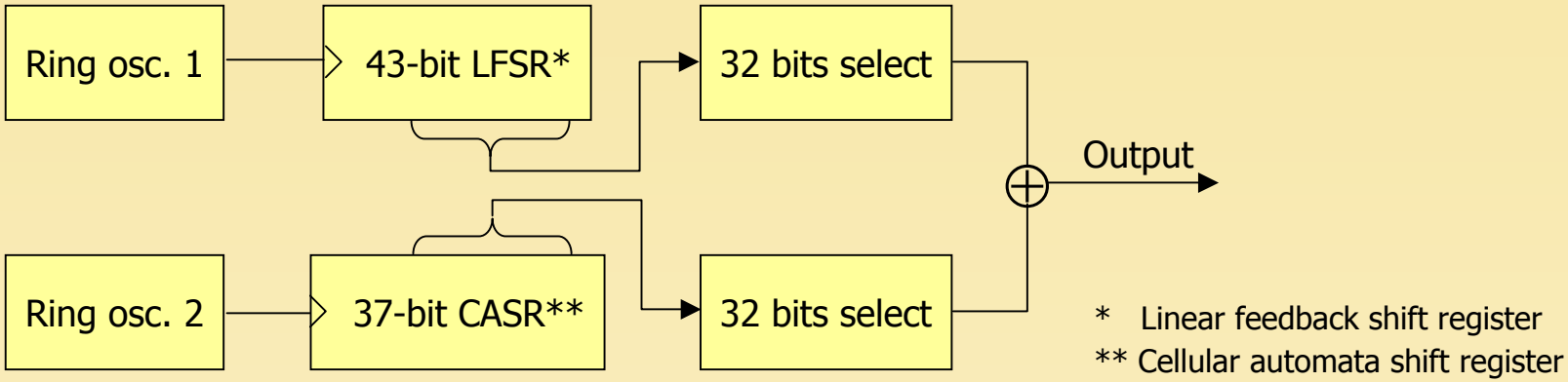
Characterization of TRNGs (6/6)

- Inner testability
  - Possibility to test the source of randomness (entropy) before post-processing



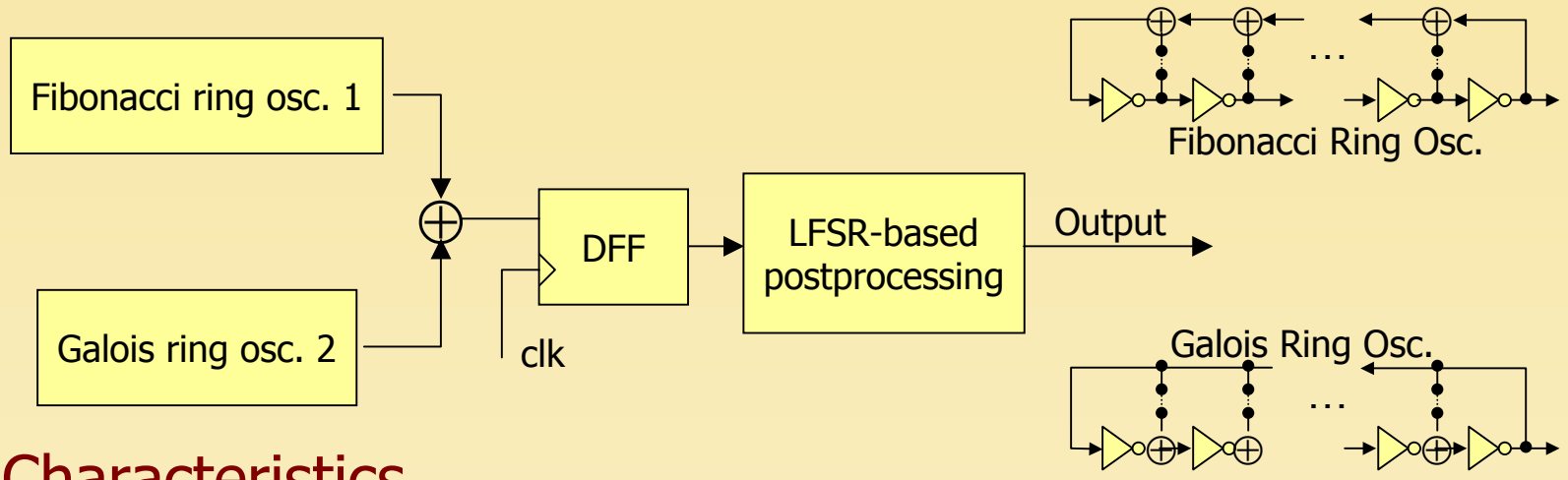
- Unconditional inner testability
  - If the inner test output does not contain any entropy, it should be equal to zero

TRNG of Tkacik [CHES 2002]



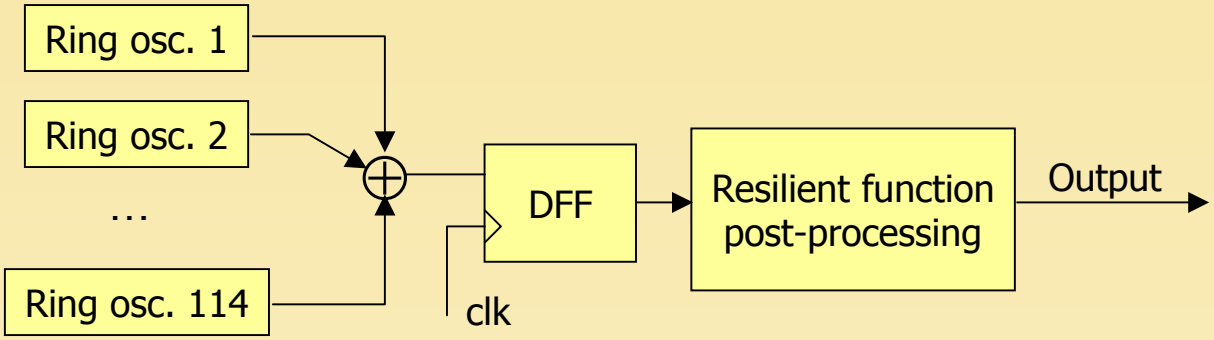
- Characteristics
  - Randomness comes from two ring oscillators, pseudo-randomness is added by two PRNGs
  - Entropy extraction and post-processing are joined together
  - Inner tests not feasible
  - Easy to implement in FPGAs
  - Attacked by Dichtl (CHES2003)

TRNG of Golić [IEEE TC 2006]



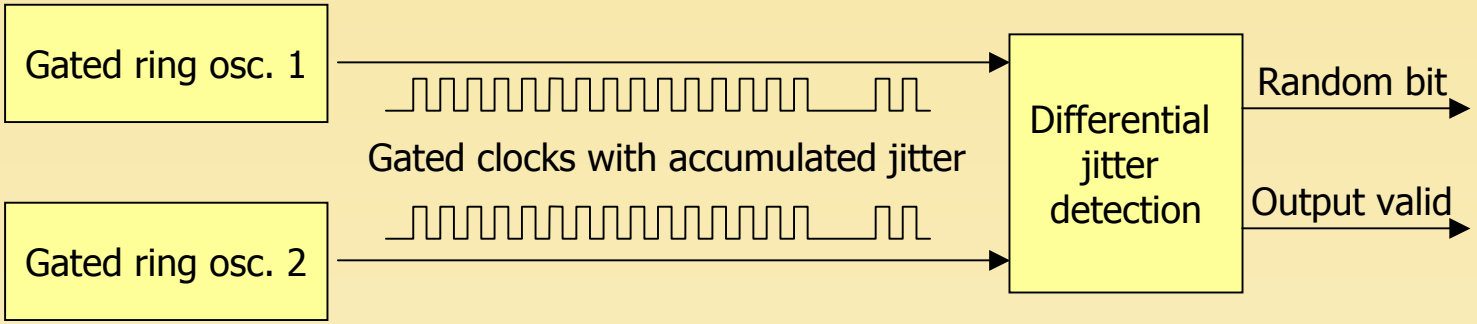
- Characteristics
  - Randomness comes from two generalized ring oscillators, pseudo-randomness is added by their pseudo-random behavior (feedbacks)
  - Entropy extraction by a DFF clocked by a reference clock
  - Simple post-processing
  - Easy to implement in FPGAs
  - Inner tests feasible (but not unconditional inner tests)

TRNG of Sunar et al. [IEEE TC 2007]



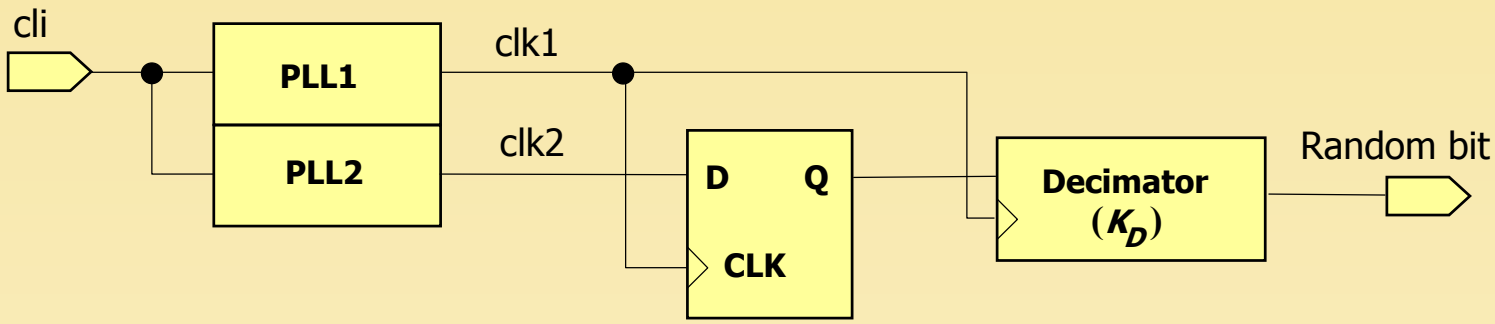
- Characteristics
  - Randomness comes from many independent ring oscillators
  - "Provable security" based on deep probability analysis
  - Entropy extraction by a DFF clocked by a reference clock
  - Sophisticated post-processing – fault resilient
  - Aimed for FPGAs
  - Inner tests feasible (but not unconditional inner tests)

TRNG of Bucci et al. [ISCAS 2006]



- Characteristics
  - Two ring oscillators share in time four delay elements (in 4 functional phases)
  - Randomness comes from differential delay of delay elements
  - Entropy extraction by a latch with an accumulated jitter detection
  - Can be implemented in FPGAs (manual P/R necessary)
  - Unconditional inner tests feasible

TRNG of Fischer & Drutarovsky [CHES 2002]



- Characteristics
  - One or two PLLs generate frequency-related clocks $f_2 = f_1 \cdot K_M/K_D$
  - Randomness comes from the tracking jitter of PLLs
  - Entropy extraction by DFF and decimator – one random bit per K_D periods of clk1
  - Easy to implement in FPGAs (containing PLLs)
  - Unconditional inner tests feasible

Jitter as a source of randomness (1/4)

- Clock jitter – deviation from the ideal timing of an event (e.g. zero crossing) on a clock signal
- Two jitter components (hard to differentiate)
  - Random jitter caused by a
    - Thermal (white) noise (Gaussian PDF)
    - Shot noise
    - "Pink" flicker noise (1/f)
  - Deterministic jitter
    - Duty cycle distortion
    - Data dependent jitter
    - Sinusoidal jitter
    - Uncorrelated bounded jitter
    - Etc.

Jitter as a source of randomness (2/4)

- Clock jitter classification
  - Period jitter
    - Change in a clocks output transition from its ideal position over consecutive clock edges (long-term jitter)
    - Easy to measure using a wide-range oscilloscope
  - Cycle-to-cycle jitter
    - Difference in a clocks period from one cycle to the next one (short-term jitter)
    - More difficult to measure
  - Tracking jitter
    - Represents how closely the PLL (or DLL) output tracks the reference (input) clock
    - Standard measurement methods not available

Jitter as a source of randomness (3/4)

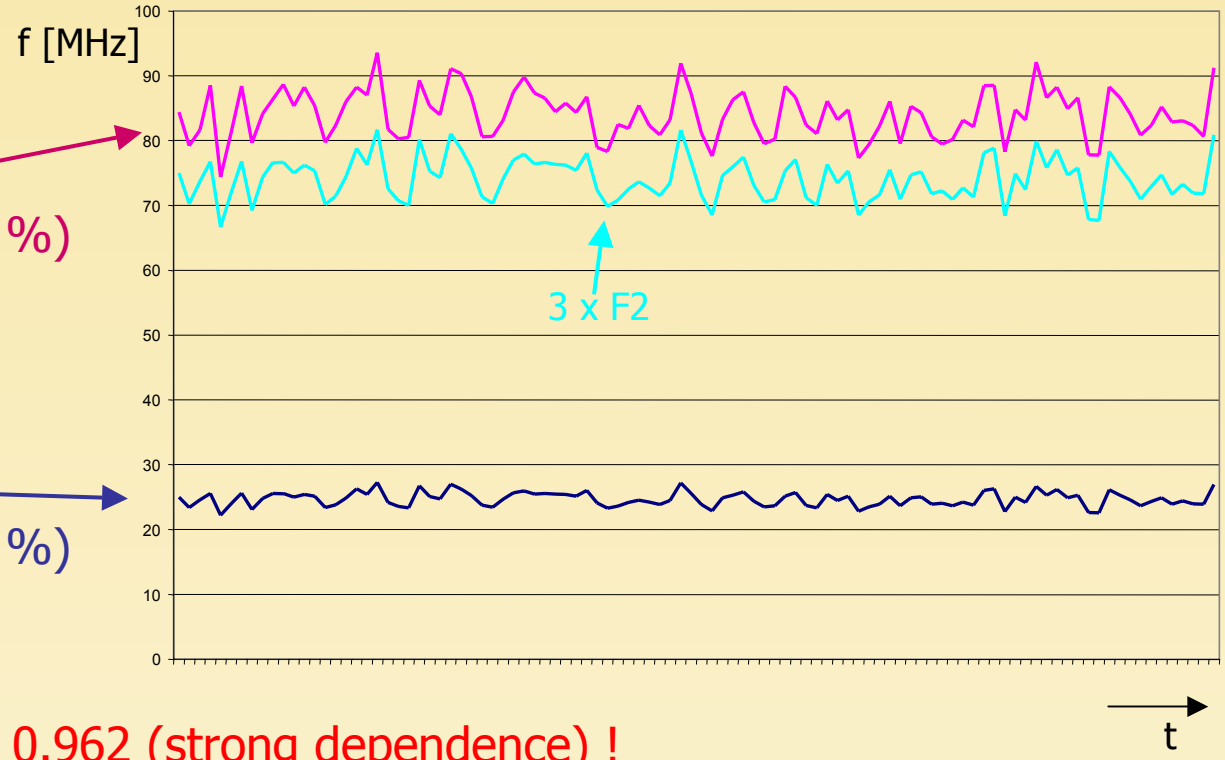
- Clock jitter measurement in FPGA applications
  - Problem
    - Random number generators employ (fast) internal processes
    - Jitter is measured outside the device (measurement influenced by the I/O circuitry)
  - Example
    - Period jitter measurement using digital oscilloscope
  - Attention
    - What we measure is not exactly what happens inside!

Jitter as a source of randomness (4/4)

Output frequency measurement of two (independent?) ROs in Altera Stratix II Evaluation Board using Tektronix TDS 3052 digital oscilloscope

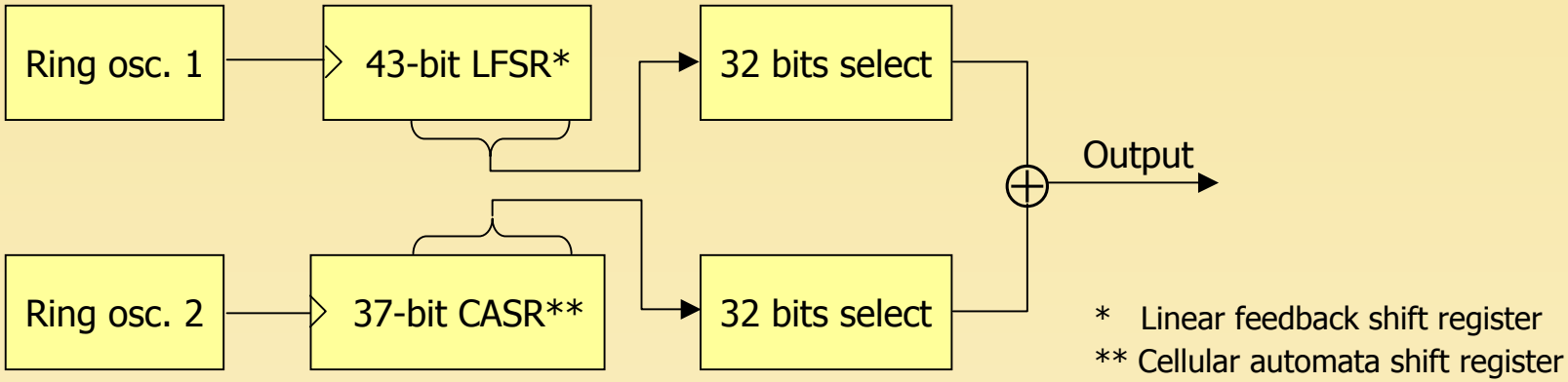
Osc1: 5 Des
F1_int: 336.3 MHz
T1_int: 3.0 ns
Jitter: 135 ps (4.51 %)
ND: 298 ps

Osc2: 15 DEs
F2_int: 98.8 MHz
T2_int: 10.1 ns
Jitter: 452 ps (4.46 %)
ND: 338 ps



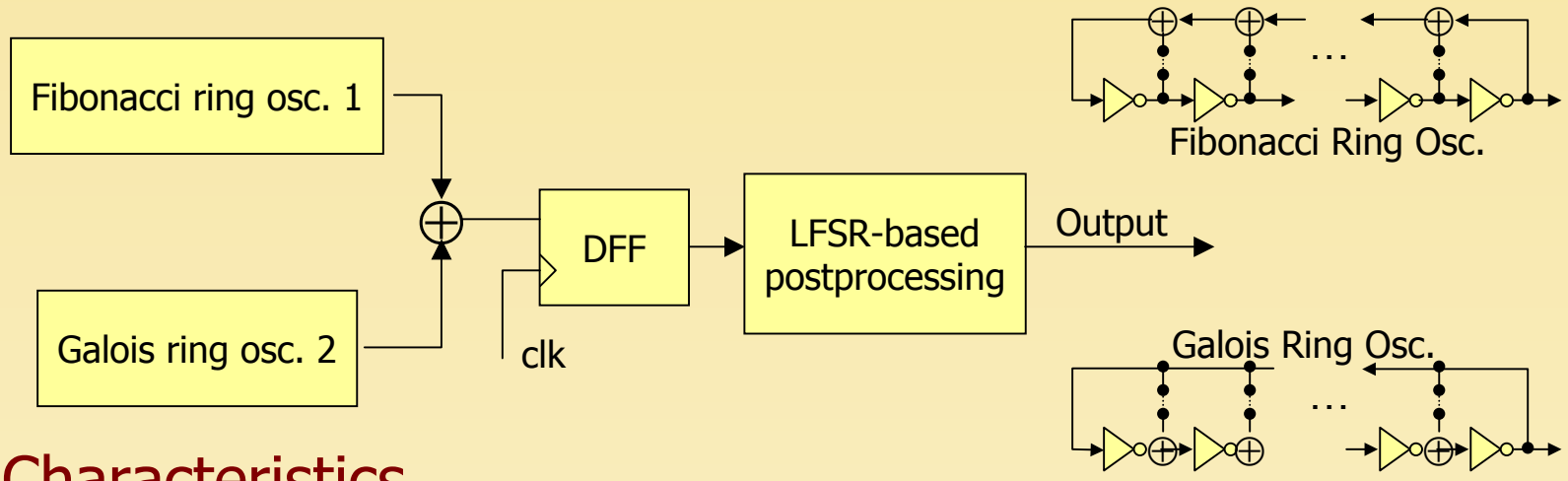
Covariance F1 and F2: 0.962 (strong dependence) !

TRNG of Tkacik reconsideration



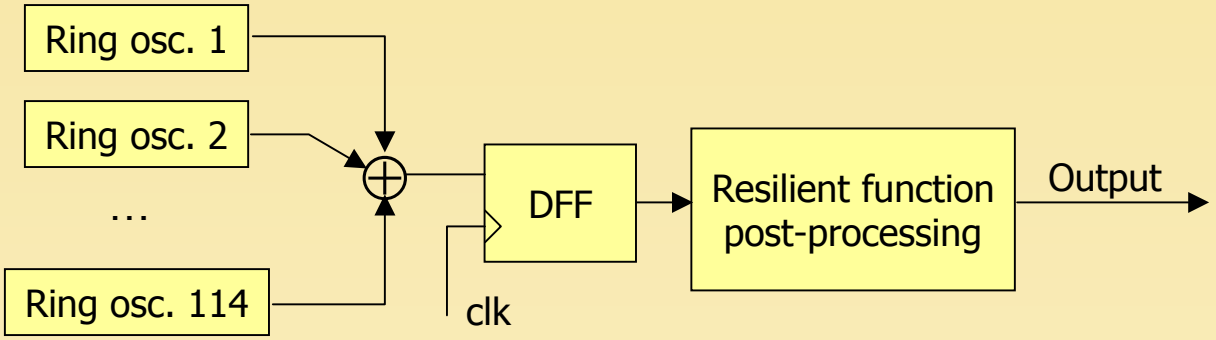
- Characteristics
  - Randomness comes from two ring oscillators, pseudo-randomness is added by two PRNGs
  - Entropy extraction and post-processing are joined together
  - Inner tests not feasible
  - Easy to implement in FPGAs, but oscillators will be closely coupled
  - Attacked by Dichtl (CHES2003)

TRNG of Golić reconsideration



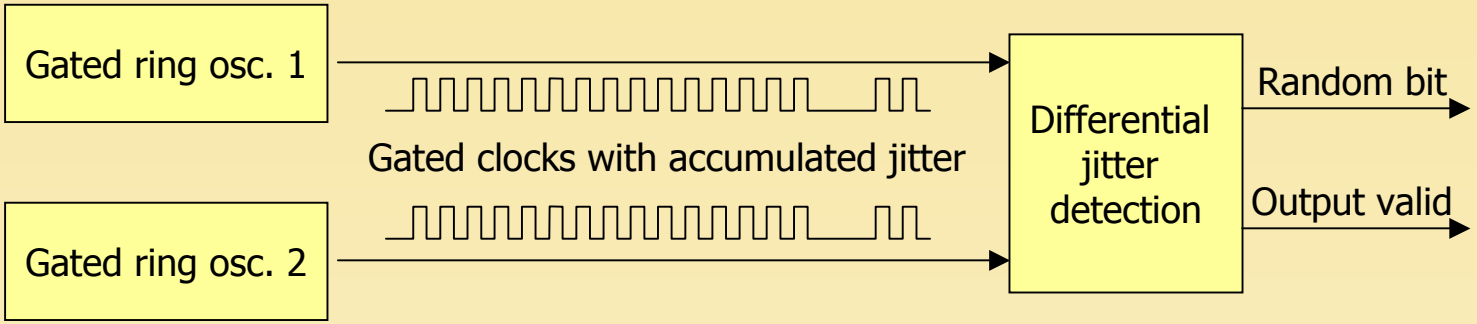
- Characteristics
  - Randomness comes from two generalized ring oscillators, pseudo-randomness is added by their pseudo-random behavior (feedbacks)
  - Entropy extraction by a DFF clocked by a reference clock
  - Simple post-processing
  - Easy to implement in FPGAs, but oscillators will be closely coupled
  - Inner tests feasible (but not unconditional inner tests)

TRNG of Sunar et al. reconsideration



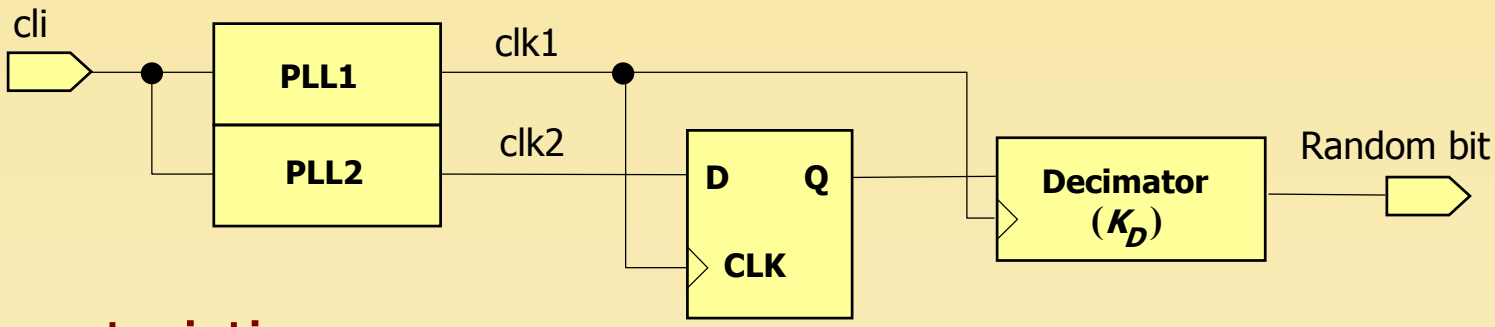
- Characteristics
  - Randomness comes from many independent ring oscillators
  - "Provable security" based on deep probability analysis
  - Entropy extraction by a DFF clocked by a reference clock
  - Sophisticated post-processing – fault resilient
  - Aimed for FPGAs, but oscillators are certainly not independent!
  - Inner tests feasible (but not unconditional inner tests)

TRNG of Bucci et al. reconsideration



- Characteristics
  - Two ring oscillators share in time four delay elements (in 4 functional phases)
  - Randomness comes from differential delay of delay elements
  - Entropy extraction by a latch with an accumulated jitter detection
  - Can be implemented in FPGAs (manual P/R necessary), perhaps the best ring oscillator-based principle for FPGAs
  - Unconditional inner tests feasible

TRNG of Fischer & Drutarovsky reconsideration



- Characteristics
  - One or two PLLs generate frequency-related clocks $f_2 = f_1 \cdot K_M / K_D$
  - Randomness comes from the tracking jitter of PLLs
  - Entropy extraction by DFF and decimator – one random bit per K_D periods of clk1
  - Easy to implement in FPGAs (containing PLLs), separated PLL and logic ground and the PLL negative feed-back reduces dependence
  - Unconditional inner tests feasible

Conclusions

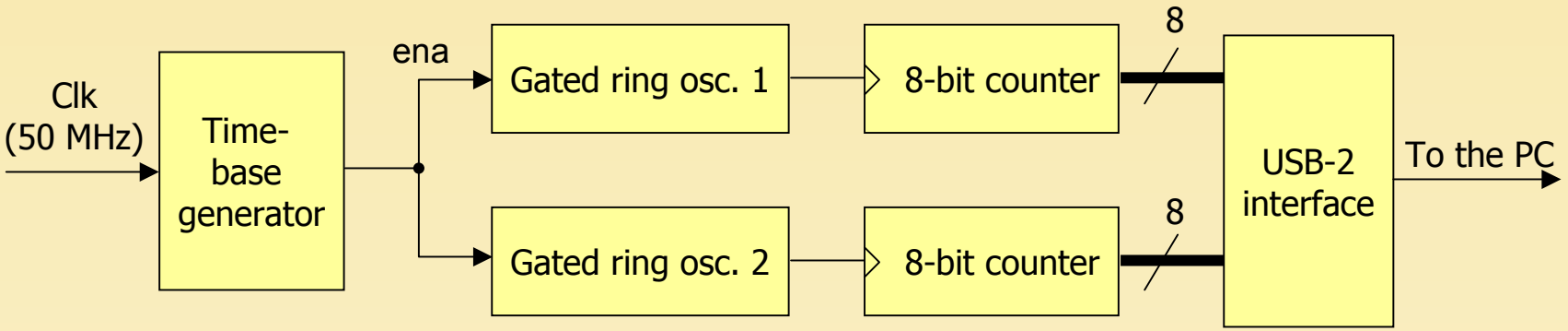
- Clock jitter has to be used with precaution
- Ring oscillators cannot be "independent" in FPGAs
- Before employing the jitter, analyze the kind of jitter you will use and measure it
- Take attention what do you measure
- Some proposed generators are not really provably secure
- Some proposed TRNG generators are not really "true random"

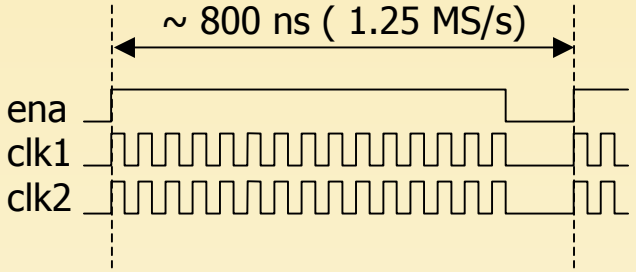
Appendix (1/6)
Beware of ring oscillators in FPGAs

- Problem N° 1:
  - Oscillator jitter measurement across slow I/O circuitry
- Problem N° 2:
  - I/O circuitry adds an extra jitter to that of the oscillator
- Problem N° 3:
  - Significant differences between jitter in Altera Stratix II and Actel Fusion evaluation boards – yes, but why???
- Solution:
  - Jitter measurement inside FPGA!
  - Yes, but how???

Appendix (2/6)
Beware of ring oscillators in FPGAs

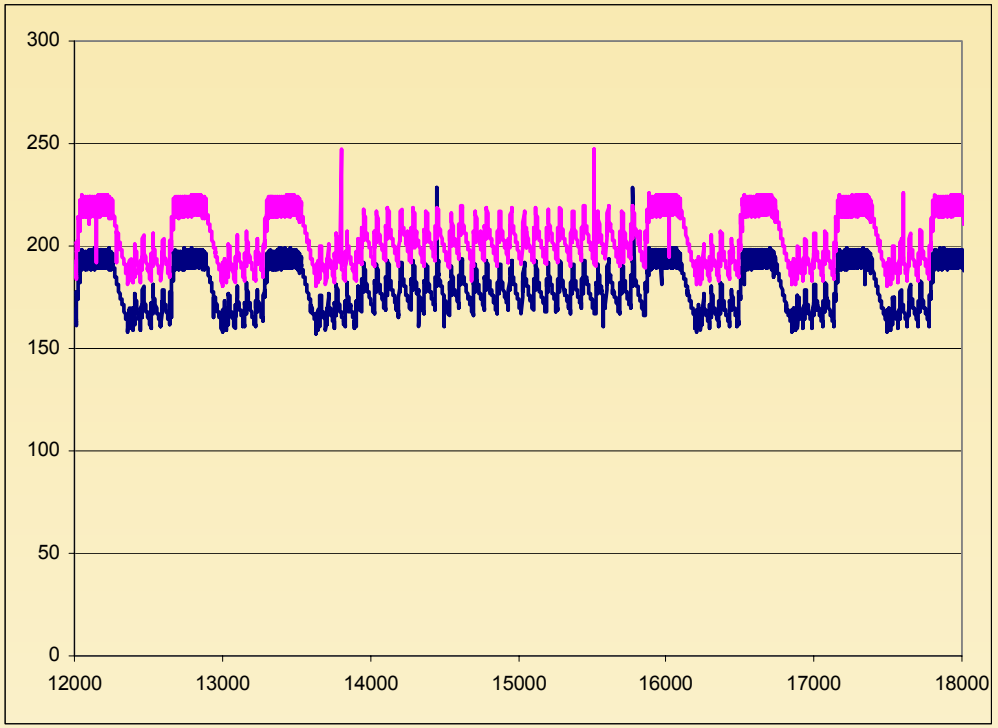
- The method





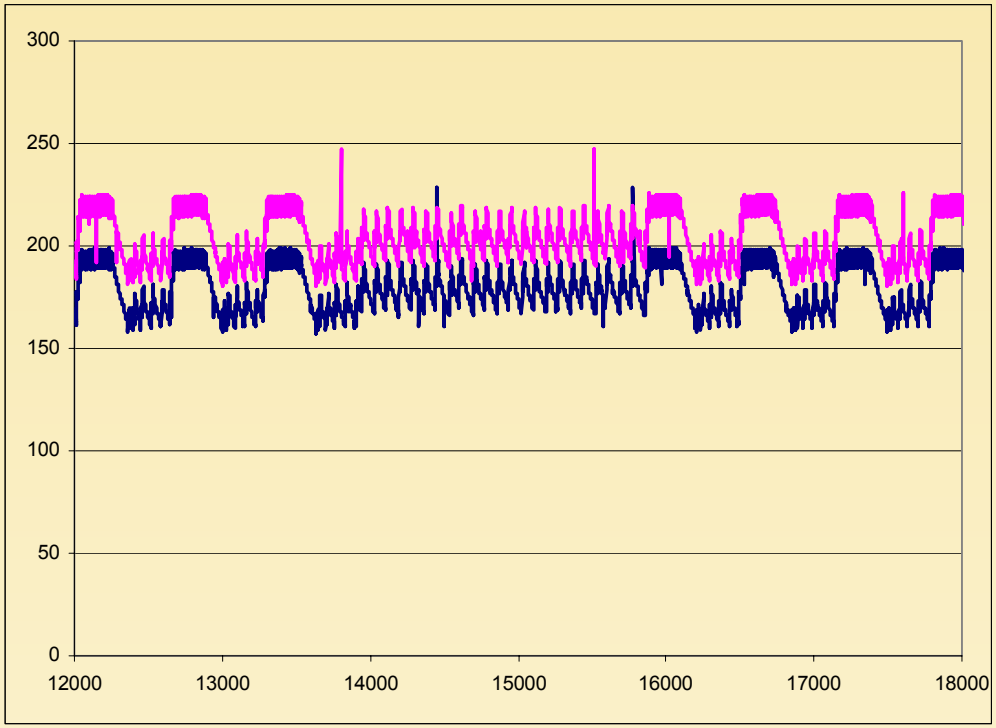
Appendix (3/6)
Beware of ring oscillators in FPGAs

Oops, something goes wrong!!!



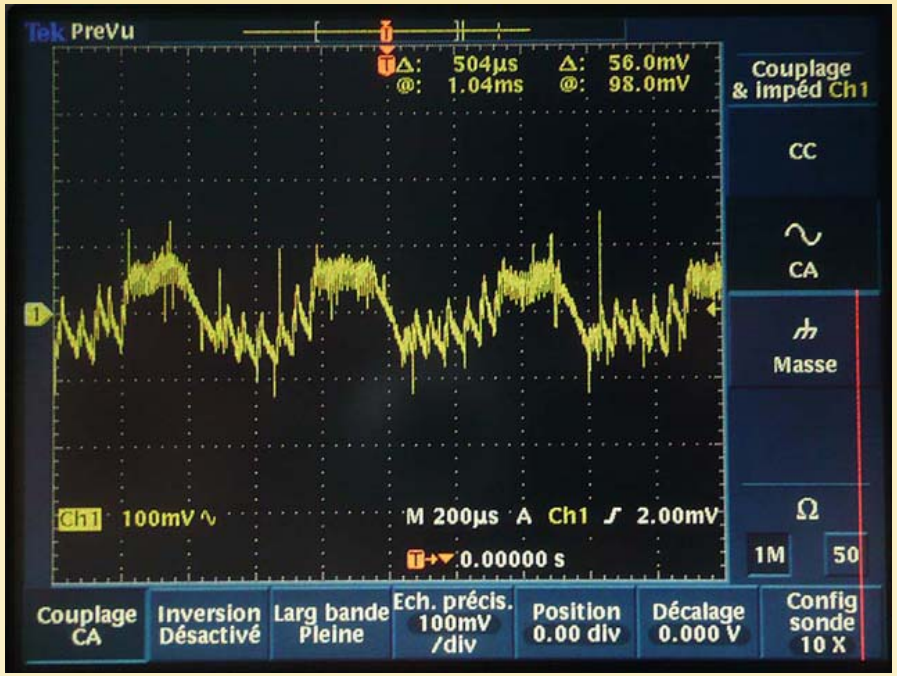
Appendix (4/6)
Beware of ring oscillators in FPGAs

Is THIS really a random signal ?!



Appendix (5/6)
Beware of ring oscillators in FPGAs

Oscillators' frequencies depend perfectly on power supply variations!



Appendix (6/6)
Beware of ring oscillators in FPGAs

Conclusions

- Stay optimist - random jitter still exists
- The method just has to take into account that
  - The deterministic jitter is much bigger and can be (easily) controlled
  - The ring oscillators inside FPGA cannot be independent
- This is not taken into account in (most) of above-mentioned methods