True Random Number Generation with Logic Gates Only

Jovan Golić

Security Innovation, Telecom Italia

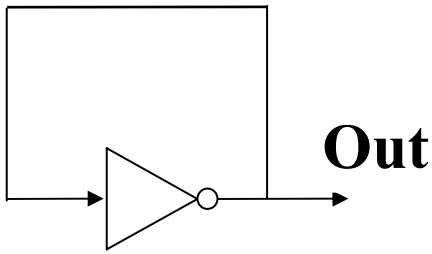
Winter School on Information Security, Finse 2008, Norway

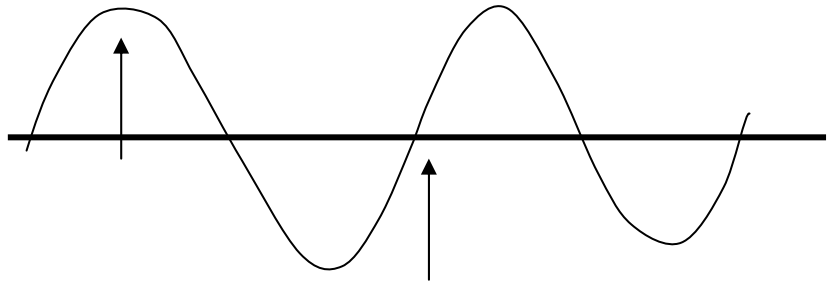
Digital Random Number Generation

- *True random numbers are needed for*
  - seeding pseudorandom number generators
  - generating cryptographic keys (e.g., one-time pad, symmetric keys, asymmetric keys)
  - generating random nonces and salts
  - protection against side-channel attacks
- *Digital random number generator (RNG) uses digital elements – logic gates only*
  - suitable for implementation on digital chips
  - cost effective

Common Digital RNGs

- ***Ring oscillators (ROs) exploit digital jitter***
  - random delays and transition times of logic gates





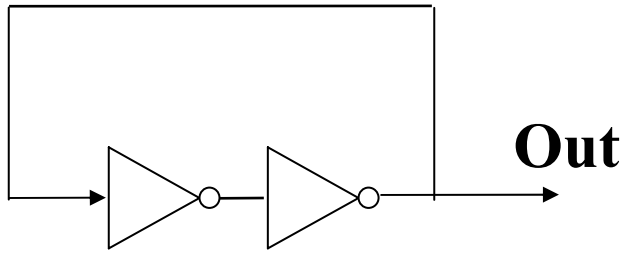
  - A slow oscillator samples a fast ring oscillator
  - Edge-triggered D-type flip-flop is used for sampling, with clock and data inputs provided by slow and fast ring oscillators, resp.

Common Digital RNGs (2)

- Mutual coupling reduces relative phase jitter
- *Sensitivity to jitter is higher near the edges of oscillating signal, but this happens rarely*
- *Regular oscillating waveform is not suitable for extraction of true randomness by sampling*
- Low entropy rate
- ***Can we transform randomness caused by jitter into a form more suitable for fast sampling?***

Common Digital RNGs (3)

- ***RS latches and edge-triggered flip-flops exploit metastability events***

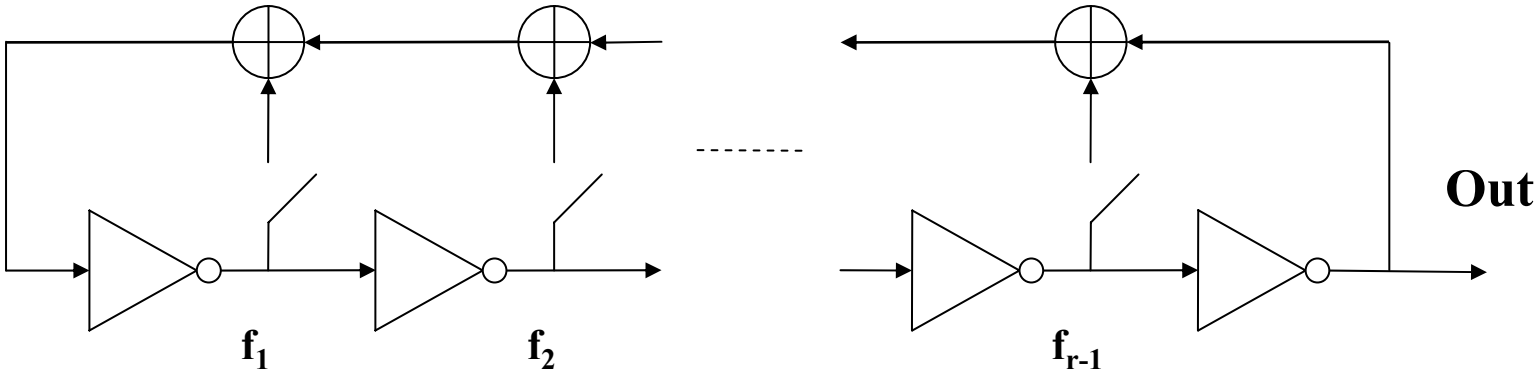


  - States (0,0) and (1,1) are metastable
  - High sensitivity to manufacturing variations and changes in temperature and voltage
  - Low entropy rate

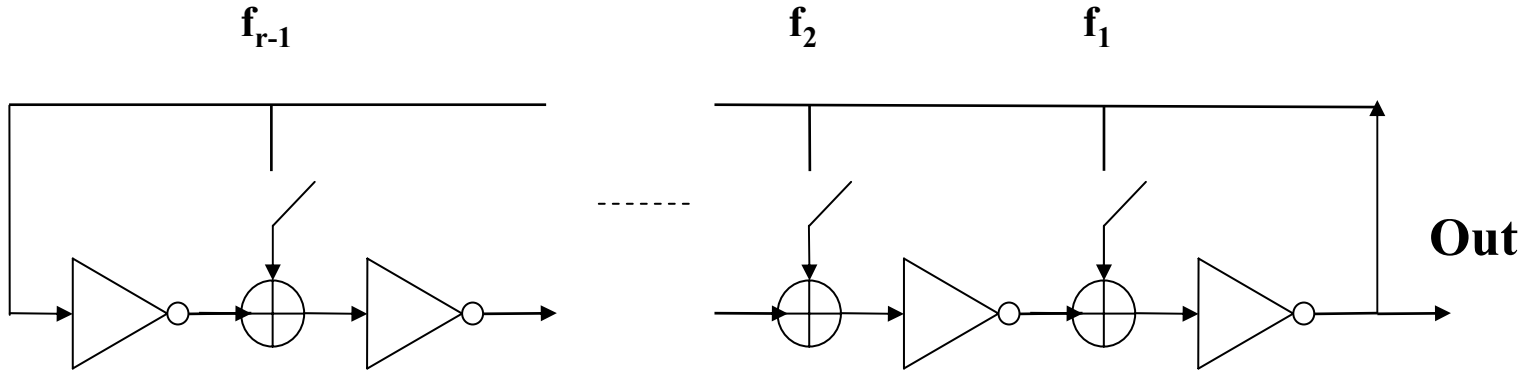
New Paradigm – FIROs & GAROs

- *Golić proposed to make feedback in a RO-like design more complex and, hence, transform the randomness caused by jitter*
- J. Dj. Golić , "New Methods for Digital Generation and Postprocessing of Random Data," IEEE Trans. Computers, vol. 55(10), pp. 1217-1229, Oct. 2006
- Two different circuits are suggested:
  - FIROs (Fibonacci Ring Oscillators)
  - GAROs (Galois Ring Oscillators)

Fibonacci Ring Oscillator (FIRO)



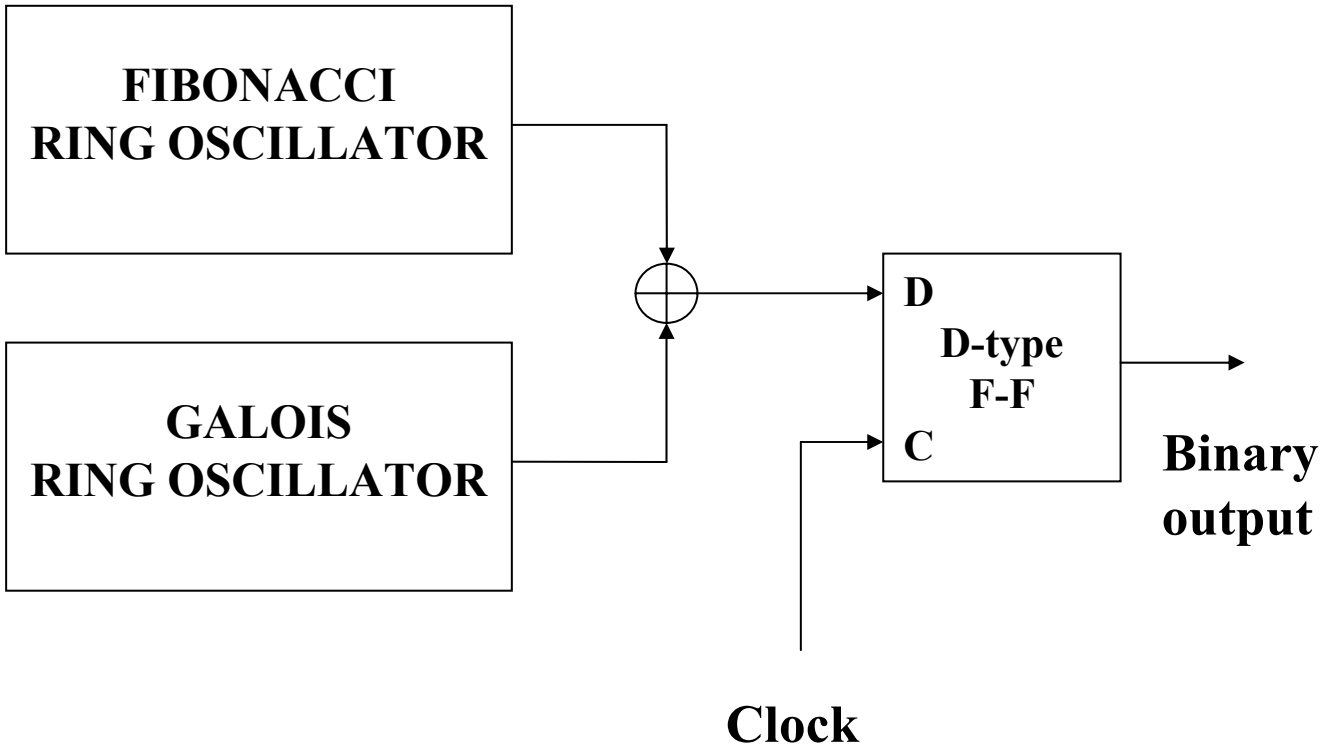
Galois Ring Oscillator (GARO)



Basic Design Criteria for FIROs & GAROs

- **Avoid fixed points** by choosing the feedback polynomial $f(x) = \sum_{i=0}^{r} f_i x^i$ appropriately
- ***Characterization***: $f(x) = (1+x)h(x) \Leftrightarrow f(1) = 0$, where $h(1) = 1$, for FIRO, and r odd, for GARO
- If $h(x)$ is primitive, then synchronous state-transition diagram contains a long cycle of length $2^r - 2$ and a short cycle of length 2, which is metastable in asynchronous operation

Combined Oscillator – FIGARO



Advantages

- **High-speed, noise-like irregular oscillating signal, with random, pseudorandom, and chaotic properties on analog/digital level**
- ***Unlike RO, total jitter increases with number of inverters, as switching frequency does not decrease***
- ***Sensivity to jitter significantly increases, as jitter is quickly propagated and transformed through feedback, resulting in oscillating waveform more suitable for extraction of true randomness by sampling***

Advantages (2)

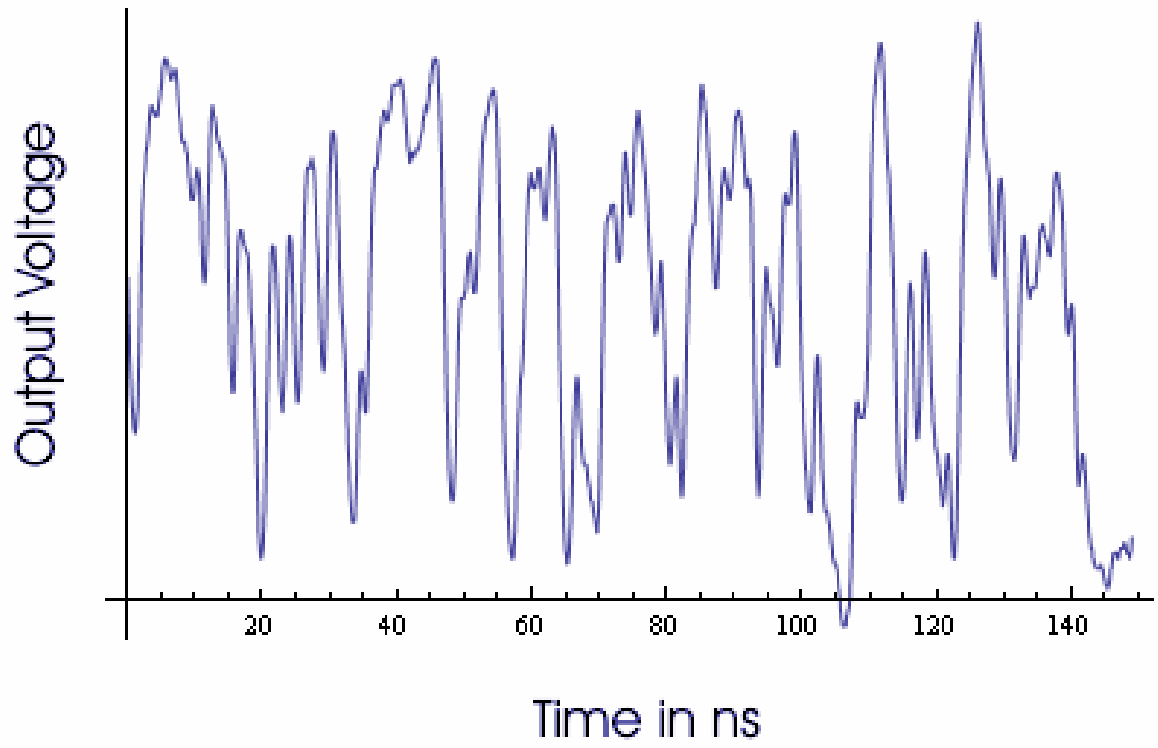
- *Mutual coupling/interlocking reduced considerably*
- *More robustness of randomness properties*
- *Easy for implementation, also in FPGA technology*
- *Internal metastability events in oscillator*
- *Sampling metastability events in sampling circuit, such as D-type flip-flop, due to noise-like irregular oscillating signal*
- ***As a consequence, much higher entropy rate***

FPGA Experiments

- *Joint work with Markus Dichtl (CHES 2007)*
- Xilinx Spartan-3 Starter Kit based on Xilinx FPGA XC3S200-4FT256C
- Each logic inverter is implemented as 1 inverter logic gate
- It is easy to find feedback polynomials yielding good randomness; for very short oscillators, in some cases, periodicity effects are observed
- A FIRO of length 15 and a GARO of length 31 are used in reported experiments

FPGA Experiments (2)

- An example of FIRO output signal



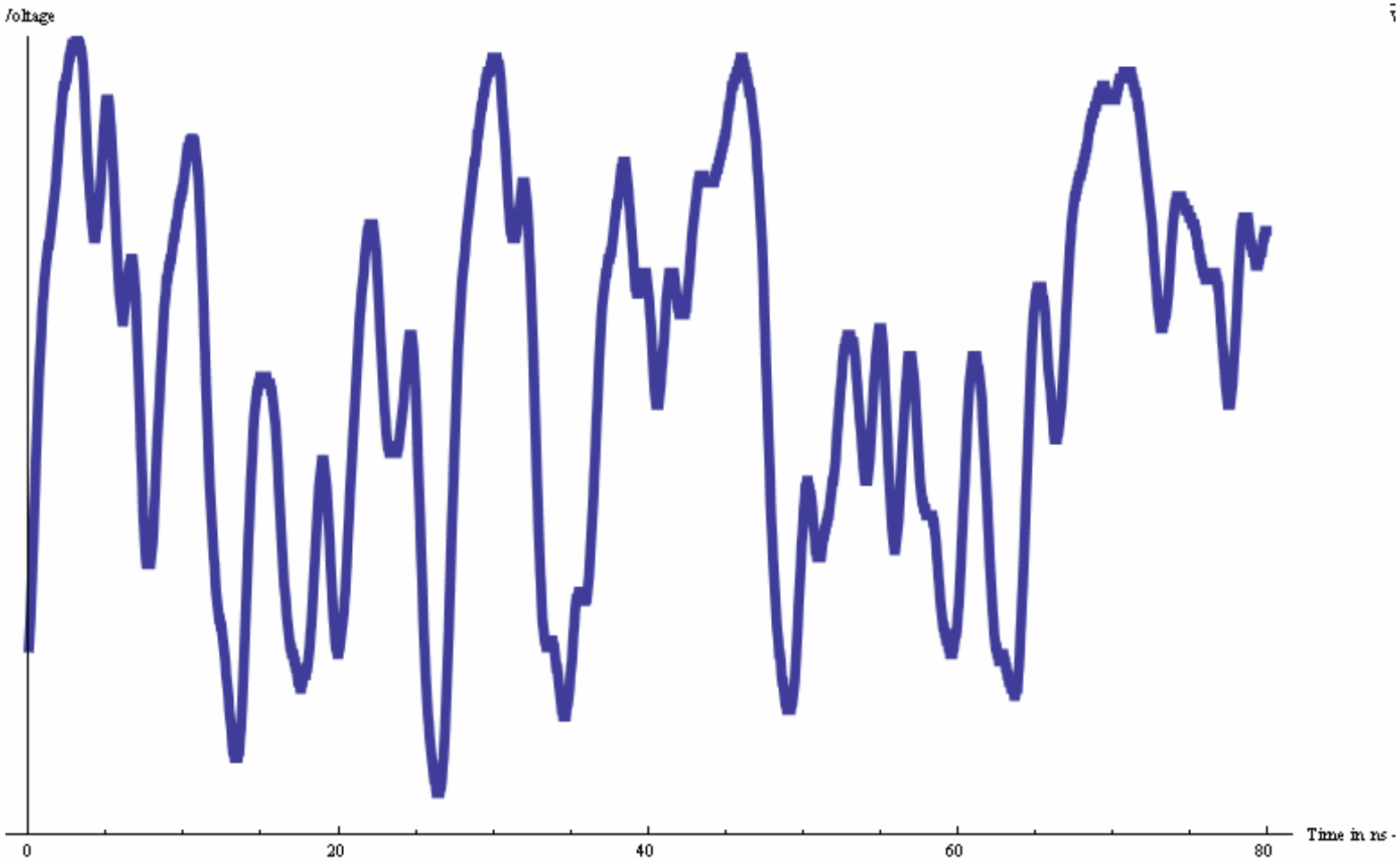
Distinguishing between True and Pseudo Randomness

- Usually, randomness is measured by statistical test suits; however, good pseudorandom sequences also satisfy these tests
- **How to distinguish between true and pseudo randomness in a FIRO or GARO?**
- ***If we use restarting from the same conditions, then changes in the output signal at any given time are due to randomness*** *(CHES 2007)*

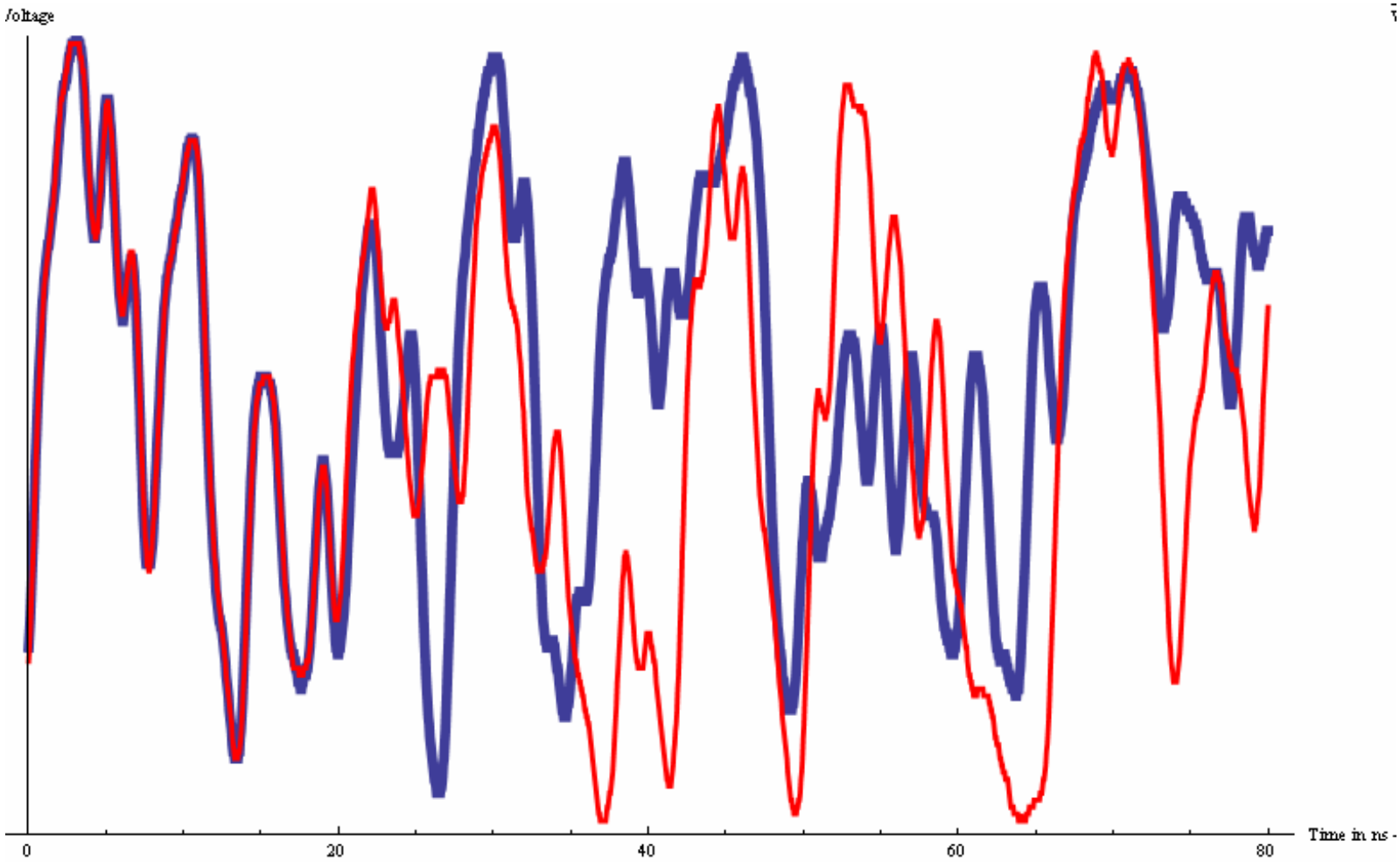
Distinguishing between True and Pseudo Randomness (2)

- Bucci & Luzzi, at CHES 2005, proposed to restart RNGs in order to produce statistically independent outputs
- Restarting can be performed by resetting each inverter to a fixed state (e.g., by using NAND gates) and by allowing the outputs of XOR gates to stabilize
  - In testing, controllable disturbances should be eliminated (e.g., quartz clock for sampling should follow the same state sequence, for each restart)

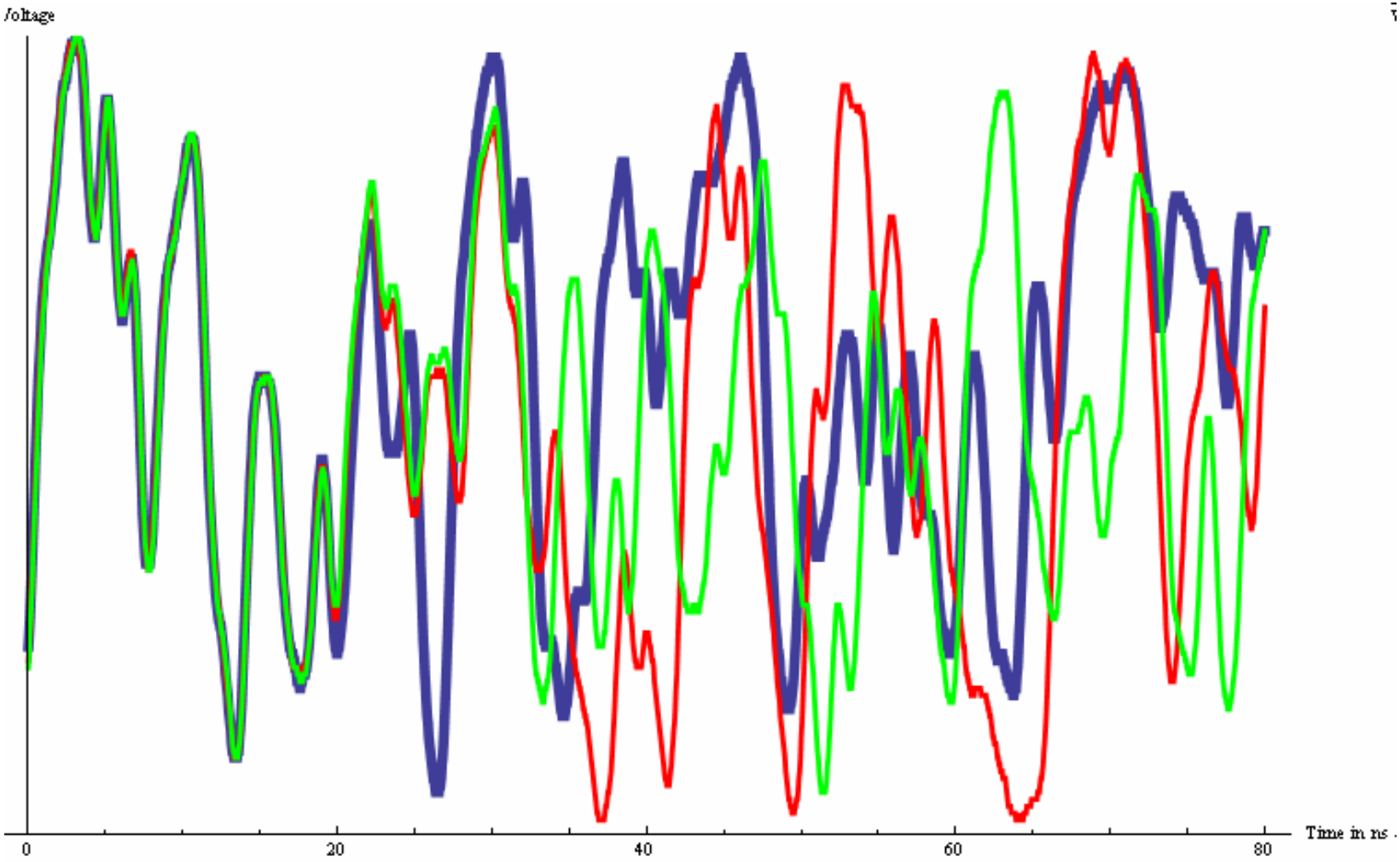
FIRO Restarts from Identical States (I)



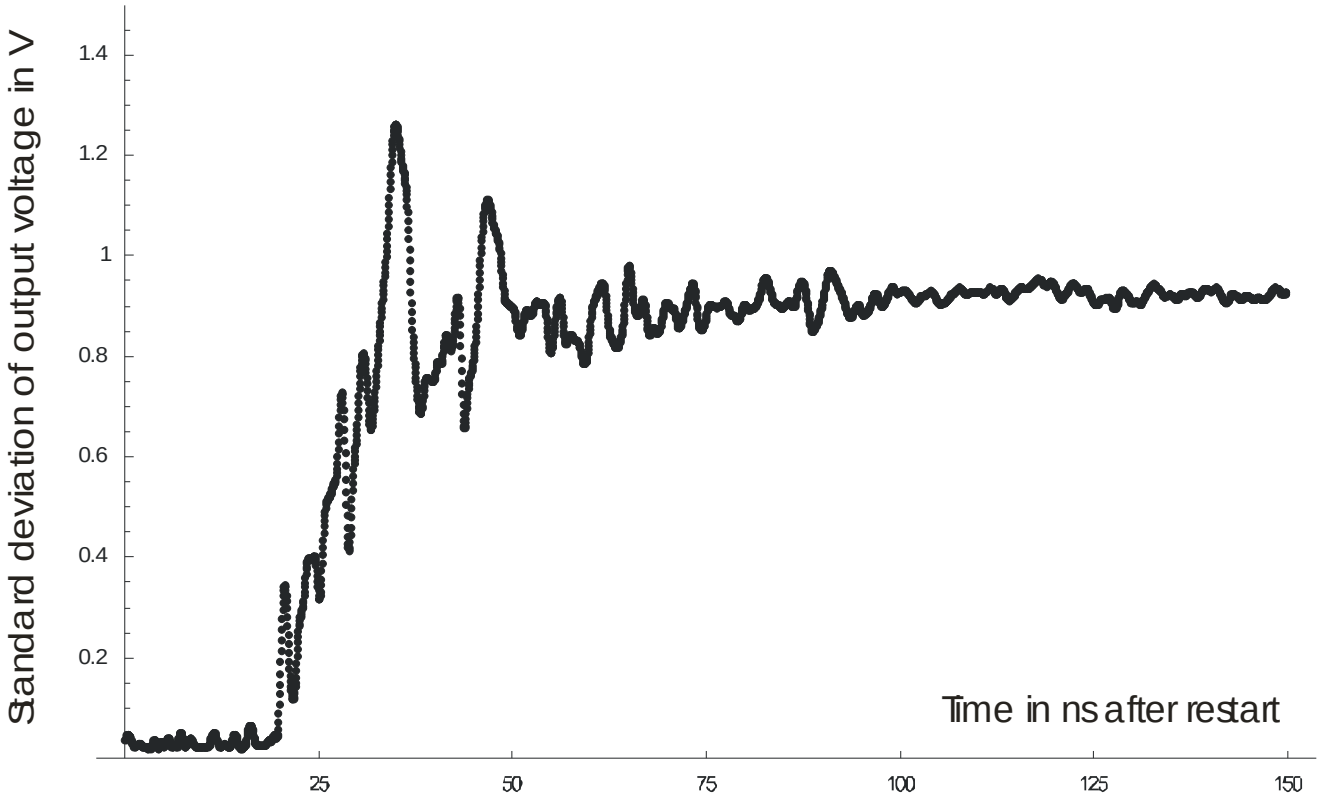
FIRO Restarts from Identical States (II)



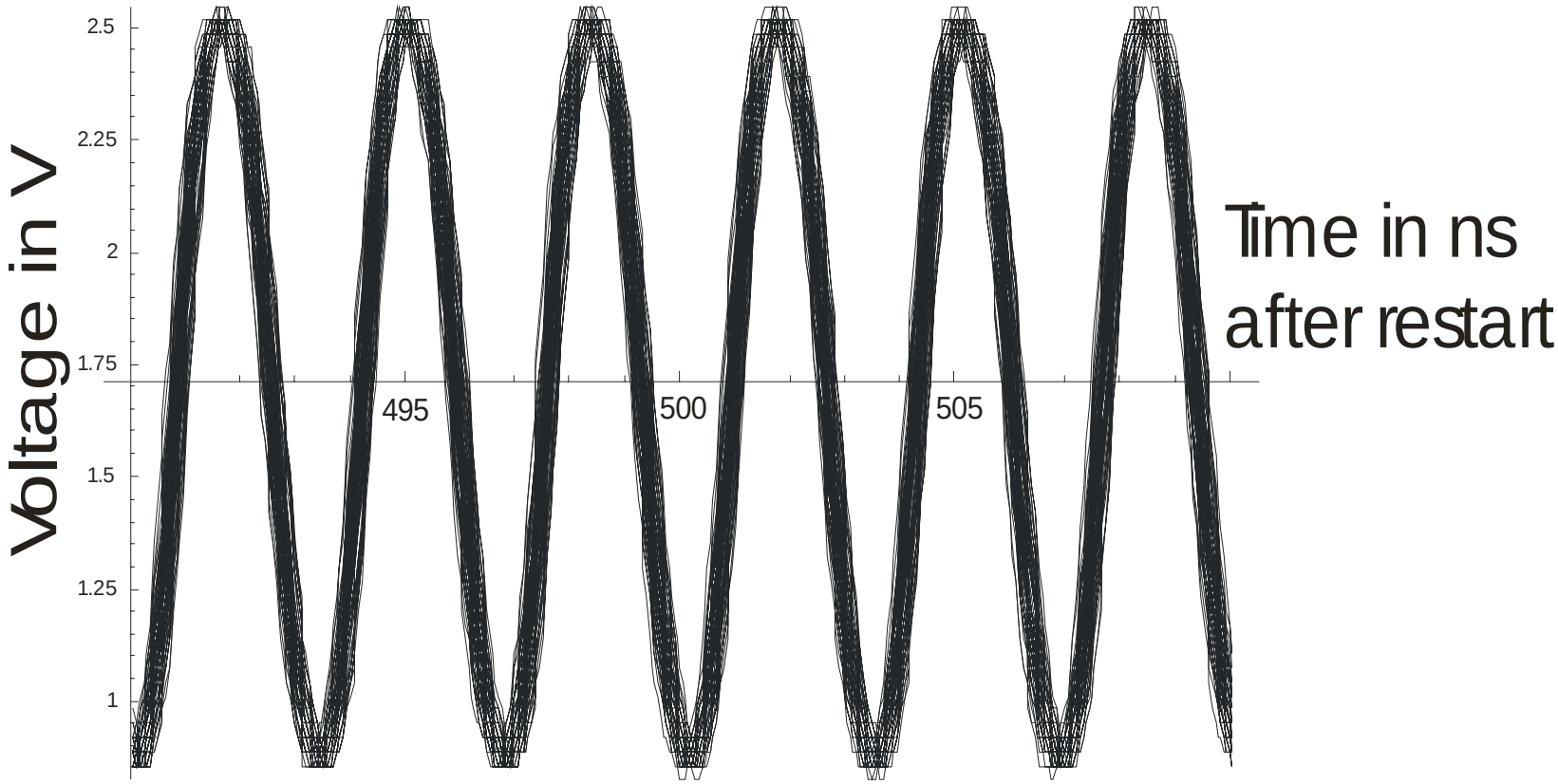
FIRO Restarts from Identical States (III)



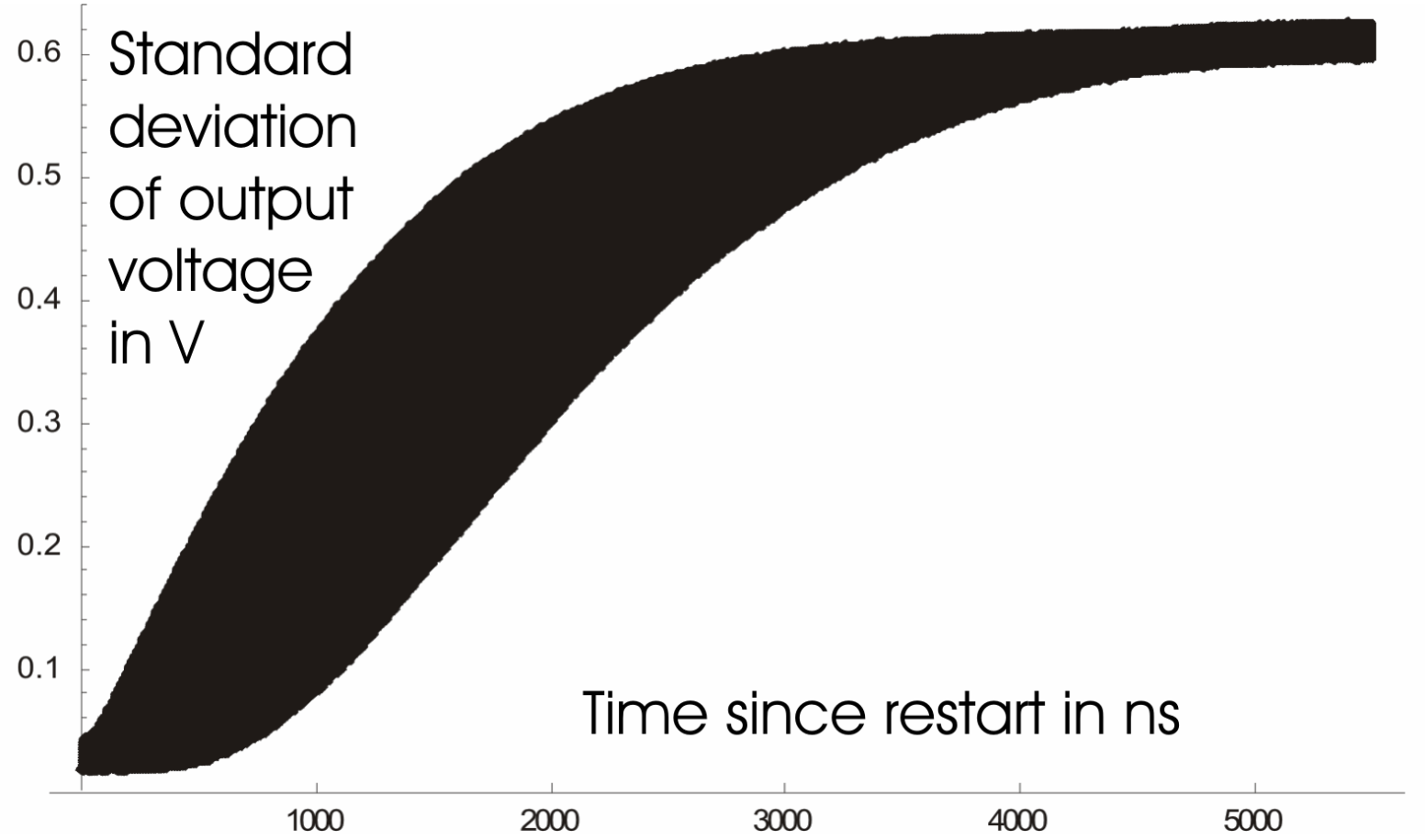
Standard Deviation of 1000 FIRO Restarts



Restarting a RO, of length 3, 100 Times

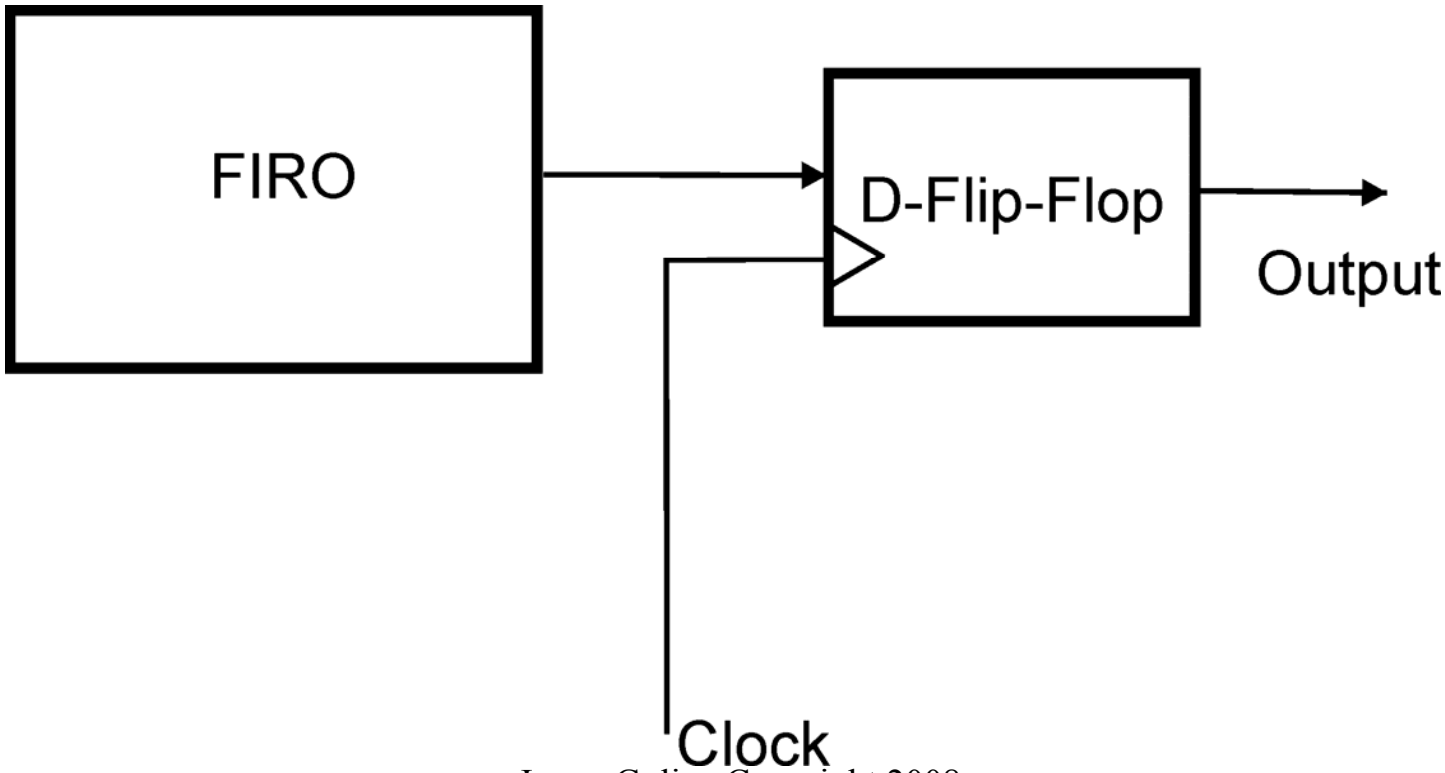


Standard Deviation of 1000 RO Restarts



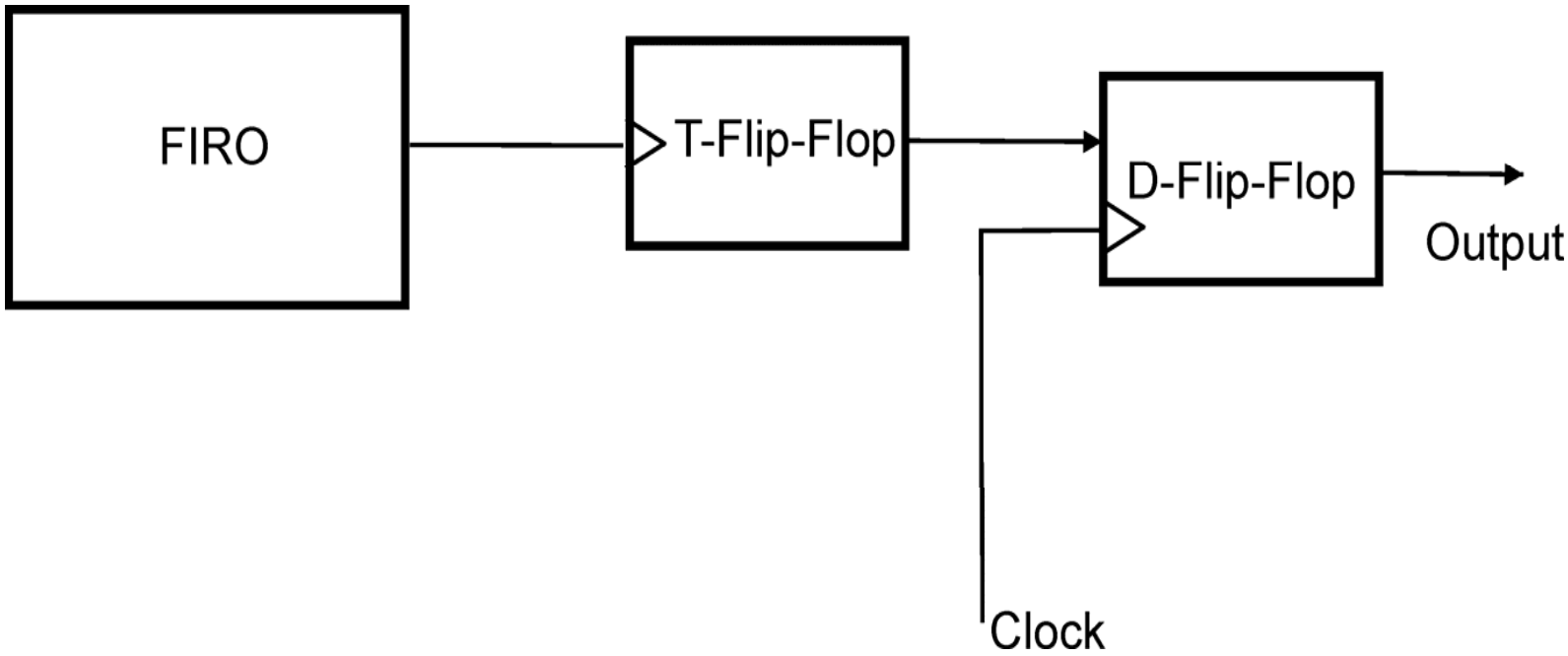
Extraction of Bits by Sampling

- ***Direct sampling***



Extraction of Bits by Sampling (2)

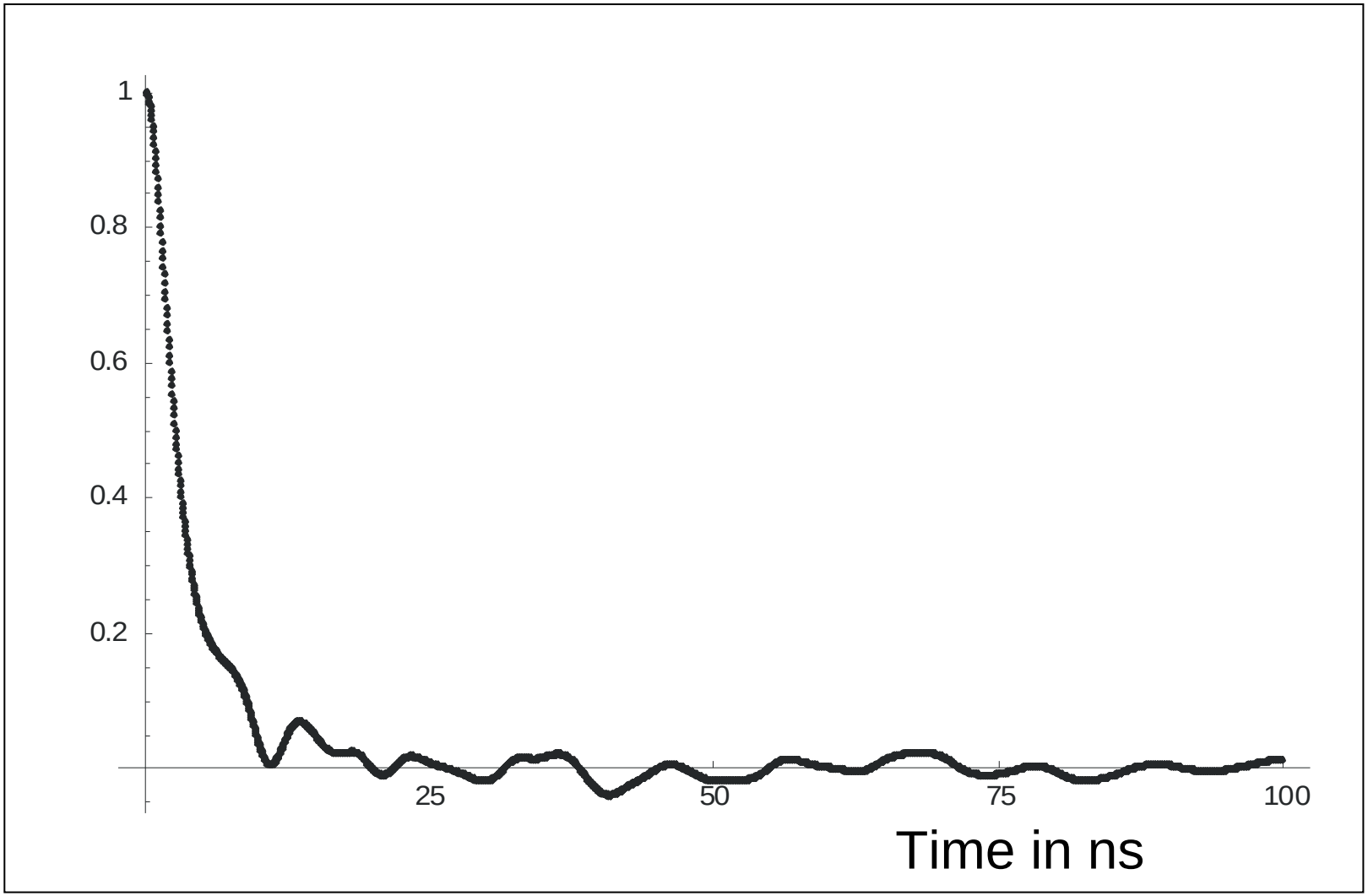
- ***Transition sampling*** with intermediate edge-triggered T-type flip-flop, reduces bias of bits



Restarting versus Continuous Operation

- ***Restarting mode:*** One bit generated at a time, needs time for transitory voltages to settle down, output bits are statistically independent and, hence, postprocessing is easy *(high-security applications)*
- ***Continuous mode:*** As many bits as needed generated at a time (restarting from a fixed state), independence plausible for higher sampling rates, but pseudo randomness is not ideally separated *(high-speed applications)*

Autocorrelation for Continuous Mode of FIRO



Data Rates Achieved

- ***FIRO Restarting mode,*** run for 60 ns, stop for 40 ns, transition sampling:

  7.14 Mbit/s (probability of 1: 51.62 %)

- ***FIRO Continuous mode,*** transition sampling, passing chi-square statistical independence test for 4-tuples:

  12.5 Mbit/s (probability of 1: 51.92 %)

Doubling Entropy Rate

- *Simultaneous direct and transition sampling doubles (raw) data rate, e.g., from 7.14 to 14.28 Mbits/s*
- Two bits from one run are weakly dependent, but the pairs from different runs are independent
- Suitable postprocessing can yield almost all the Shannon entropy, which was 1.933 per pair, in the considered example with restarting
- Achieved output entropy rate is thus 13.8 Mbits/s

Power Consumption

- Theoretically, FIRO or GARO power consumption could increase linearly with length, as average inverter gate switching frequency does not decrease with length, and more power consumption means more primary randomness due to jitter
- For FIRO of length 15 on CMOS ICs 74HCTXX, measured power consumption was 3 to 4 times higher than for a RO (depending also on feedback)
- ***FIRO entropy rate is orders of magnitude higher***

Generalizations

- Instead of FIRO or GARO, other autonomous asynchronous logic circuits with feedback, without fixed points, may be used
- *Next-state function of associated (synchronous) finite-state machine (FSM) should satisfy:*
  - Loops should not exist (no fixed points)
  - Cycles of length two (states) should be metastable in asynchronous operation
- In particular, (programmable linear) cellular automata may also be used

Digital Postprocessing

- **RNG** generates a raw binary sequence, possibly biased and correlated, where, typically, correlations may extend over a small number of consecutive bits
- The bias and correlations are usually difficult to quantify and should, hence, be considered as unknown
- The objective of postprocessing is to obtain a purely random binary output sequence, without using auxiliary purely random bits *(unlike what is known as randomness extraction)*

Digital Postprocessing (2)

- *If the raw binary sequence is not correlated (i.e., is a sequence of statistically independent, possibly biased bits, such as in the restarting mode of operation), then one may apply theoretical algorithms*
  - von Neumann algorithm, treating pairs of consecutive bits, but inefficient in terms of entropy rate achieved
  - Juels, Jakobbson, Shriver, Hillyer [JJSH2000] algorithm "How to turn loaded dice into fair coins", treating *n*-tuples of consecutive bits
  - For any given *n*, [JJSH2000] algorithm is provably optimal and, asymptotically in *n,* is able of extracting the full Shannon entropy

Digital Postprocessing (3)

- *If the raw binary sequence is possibly correlated (e.g., as in the continuous mode of operation), then one may apply heuristic algorithms*
  - Data rate has to be reduced
  - Bias and correlations need to be diffused among output bits
  - Synchronous nonautonomous FSM with one input (raw data) and one output, which implements a sequential transformation
  - Input can be introduced into the next-state function one symbol/bit at a time by using a latin-square/XOR operation
  - Output sequence can be irregularly decimated for speed reduction

Digital Postprocessing (4)

- ***Theoretical criterion:*** if input sequence is purely random, then output sequence is also purely random
  - e.g., reversible sequential transformation
  - in particular, a current input bit can be XOR-ed with a current output bit of autonomous FSM and also with one or more state bits to influence the next state; FSM initial state can be fixed
- ***Heuristic criteria:***
  - Computational distinguishibility from purely random sequence, for any (or zero) input sequence
  - A change of the first input bit induces a computationally unpredictable change of subsequent output sequence (propagation effect)

Digital Postprocessing (5)

- For example, one may use a self-clock-controlled linear feedback shift register (LFSR) in Galois configuration