High-Speed True Random Number Generation with Logic Gates Only

Markus Dichtl

Jovan Golić

Being One Year Late

You should have heard this talk at CHES 2006, but …

Why We Need Random Numbers from Logic Gates

Many cryptographic protocols need random numbers (key generation, seeding pseudo random number generators, random nonces, protection against side channel attacks ...)

Analogue components are cumbersome on digital chips, so random number generators using digital logic only are preferable

Most Popular Random Number Generators Based on Logic Gates

- Ring oscillators (ROs)
- Devices exploiting metastability of digital circuits like flip-flops

Jovan Golić's Great Invention

Jovan Golić found that making the feedback in a RO-like design more complex made the behaviour of the oscillators much more complex

He suggested two different circuits:

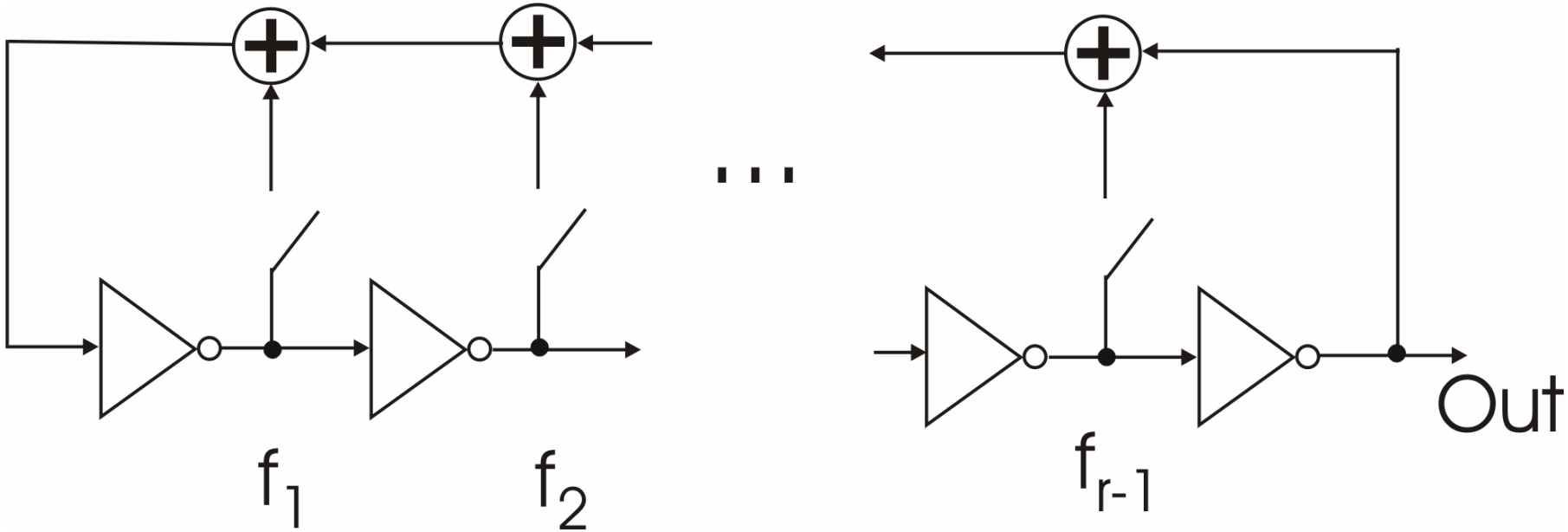
- FIROs (Fibonacci Ring Oscillators)
- GAROs (Galois Ring Oscillators)

and specified certain requirements for them (e. g., how to avoid fixed points)

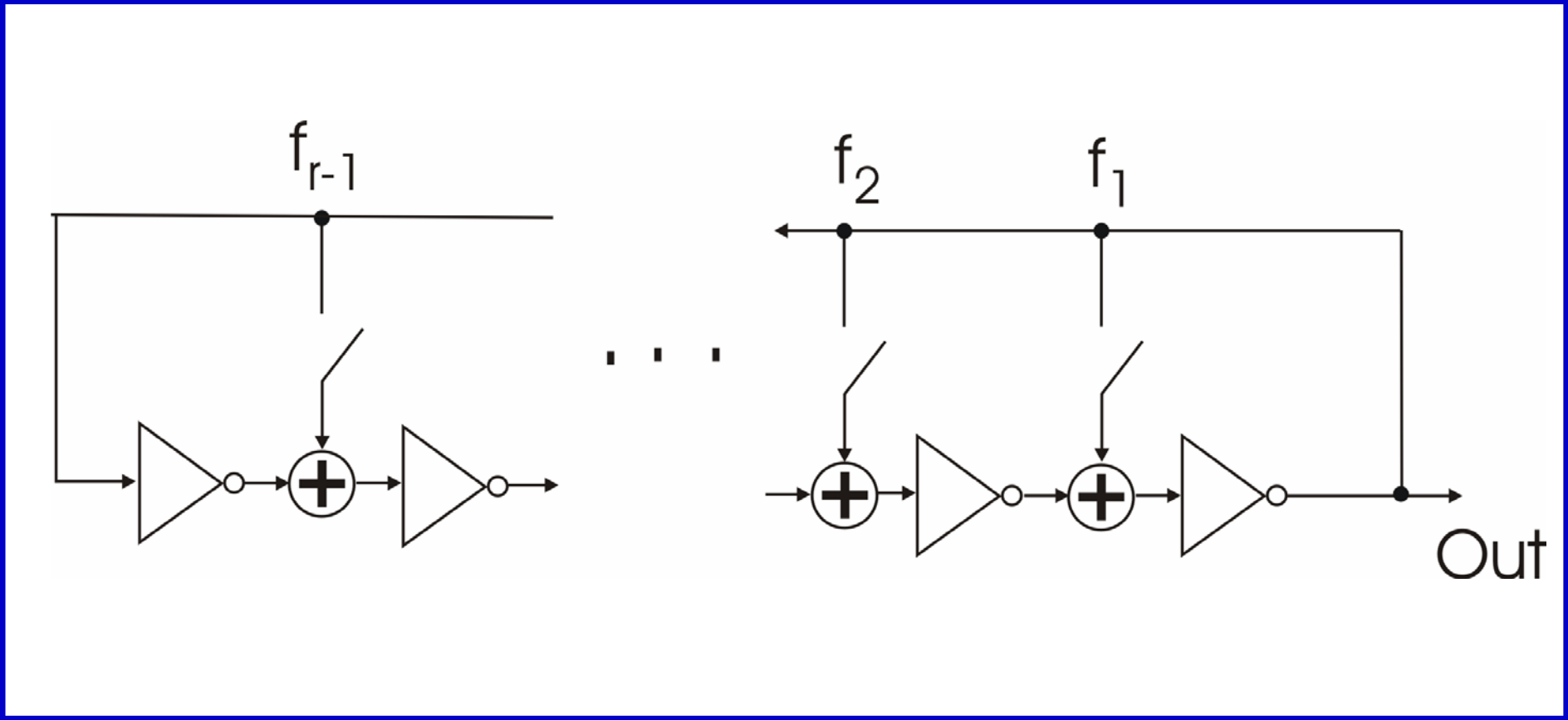
Reference for FIROs and GAROs

J. Dj. Golić , “New Methods for Digital Generation and Postprocessing of Random Data,” IEEE Trans. Computers, vol. 55(10), pp. 1217-1229, Oct. 2006

FIRO



GARO



Hardware Used for Experiments

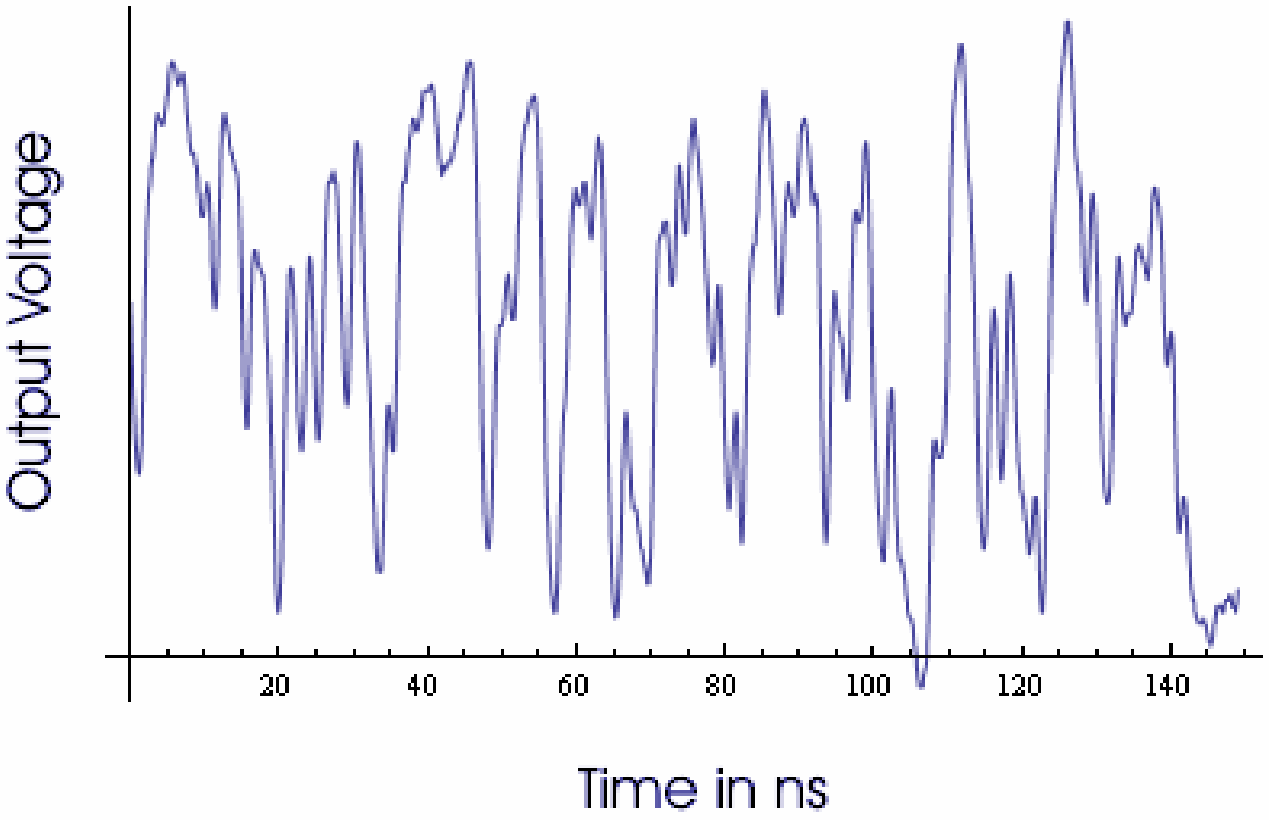
Most experiments:

Xilinx Spartan-3 Starter Kit based on Xilinx FPGA XC3S200-4FT256C

Some experiments with CMOS chips 74HCTXX

Output voltages were recorded by a LeCroy Wavepro 7200 oscilloscope

Example of FIRO Output



(Feedback polynomial $x^{15}+x^{14}+x^7+x^6+x^5+x^4+x^2+1$)

The Problem to Solve

As FIROs and GAROs are similar to LFSRs, although they operate asynchronously, and LFSRs have good pseudo random properties, the question is:

Is the complex behaviour truly random or pseudo random?

How can we distinguish that?

The Solution

To solve the problem of distinguishing between pseudo randomness and true randomness, we go back to the definition:

When restarting many times from the same conditions, pseudo randomness leads to identical behaviour for each restart, true randomness does not!

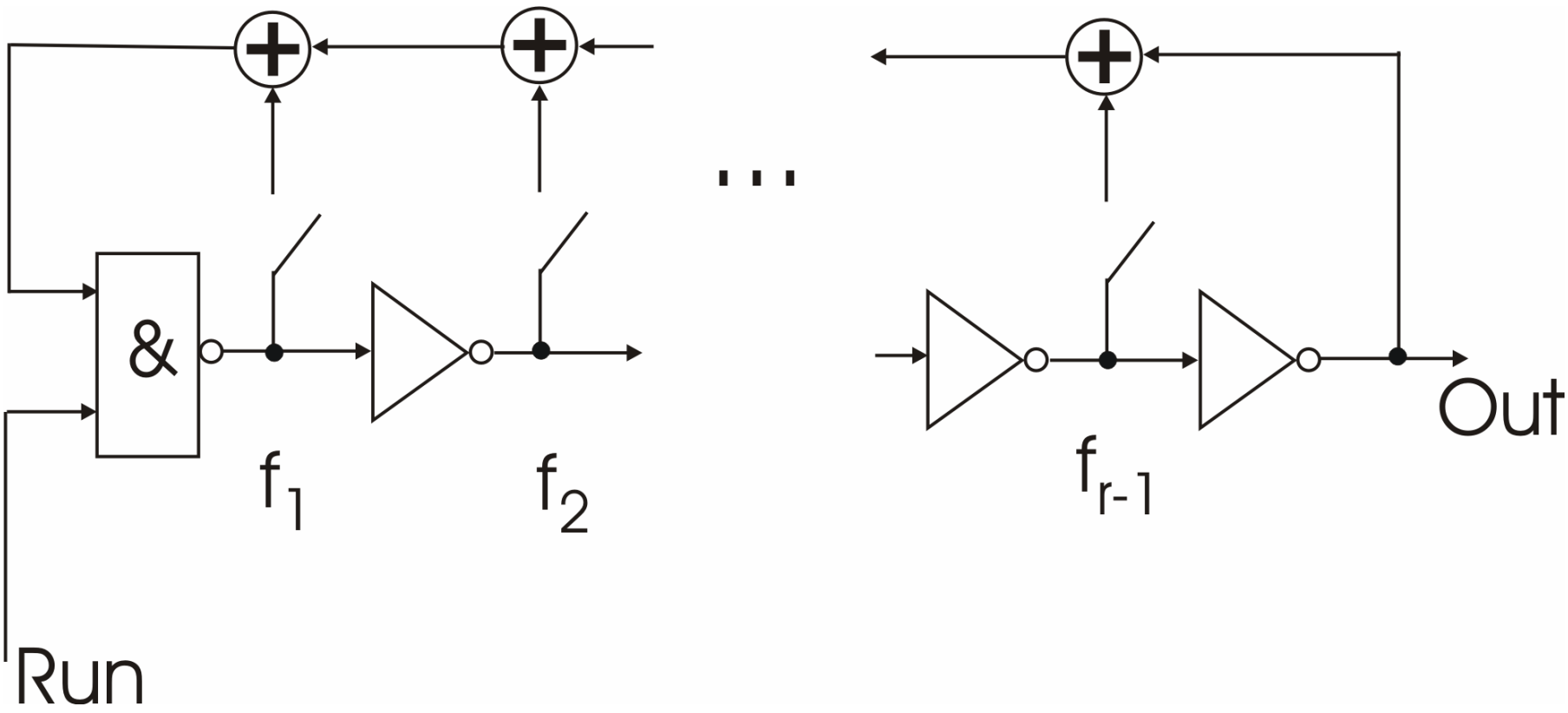
A Restarting TRNG

H. Bock, M. Bucci, R. Luzzi, “An Offset-Compensated Oscillator-based Random Bit Source for Security Applications”, CHES 2004

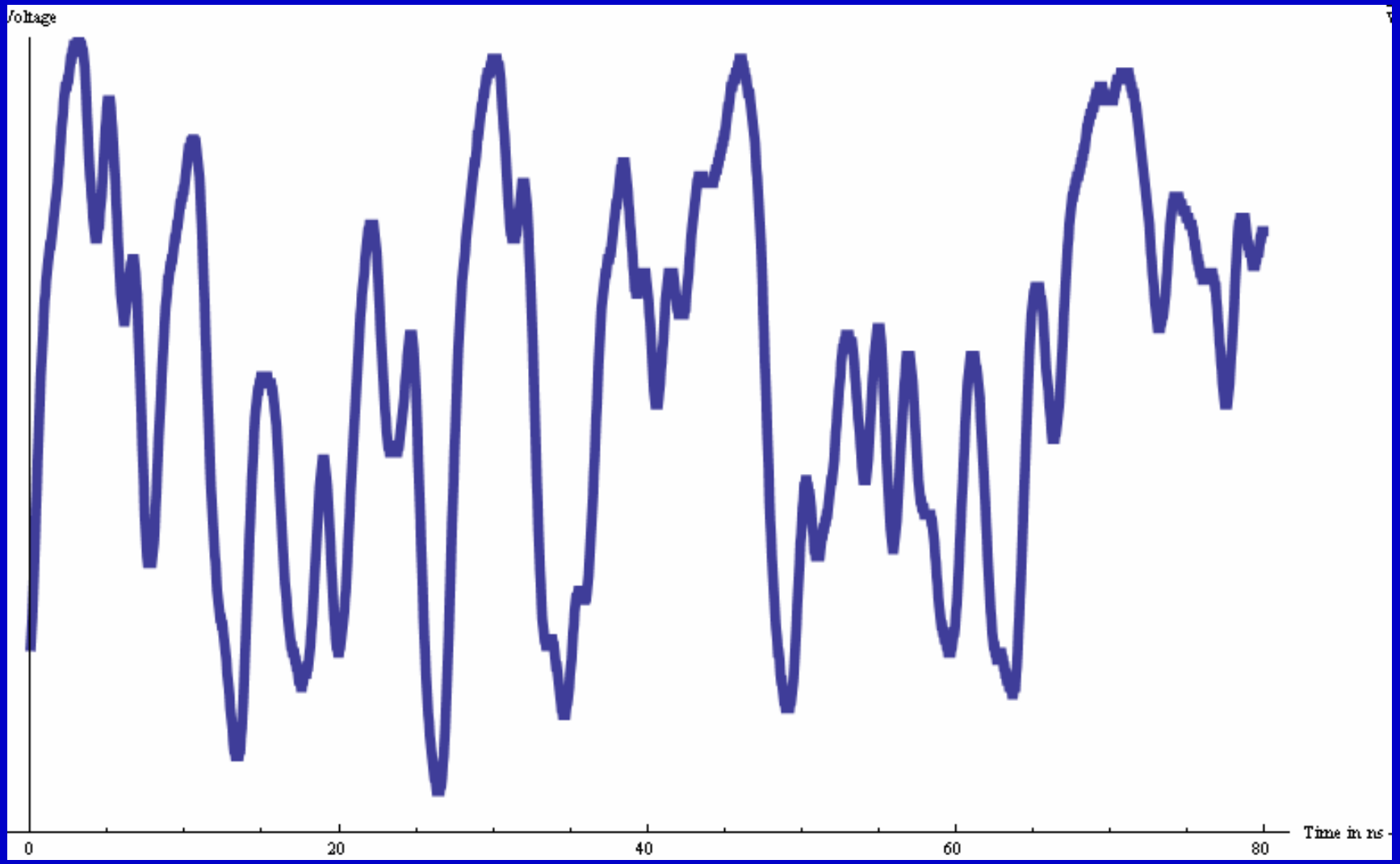
M. Bucci, R. Luzzi, “Design of Testable Random Bit Generators,” CHES 2005

proposed restarting a RO to avoid a complicated deterministic beating pattern between fast and slow frequencies in a RO and restarting a random number generator to obtain statistically independent output bits, respectively

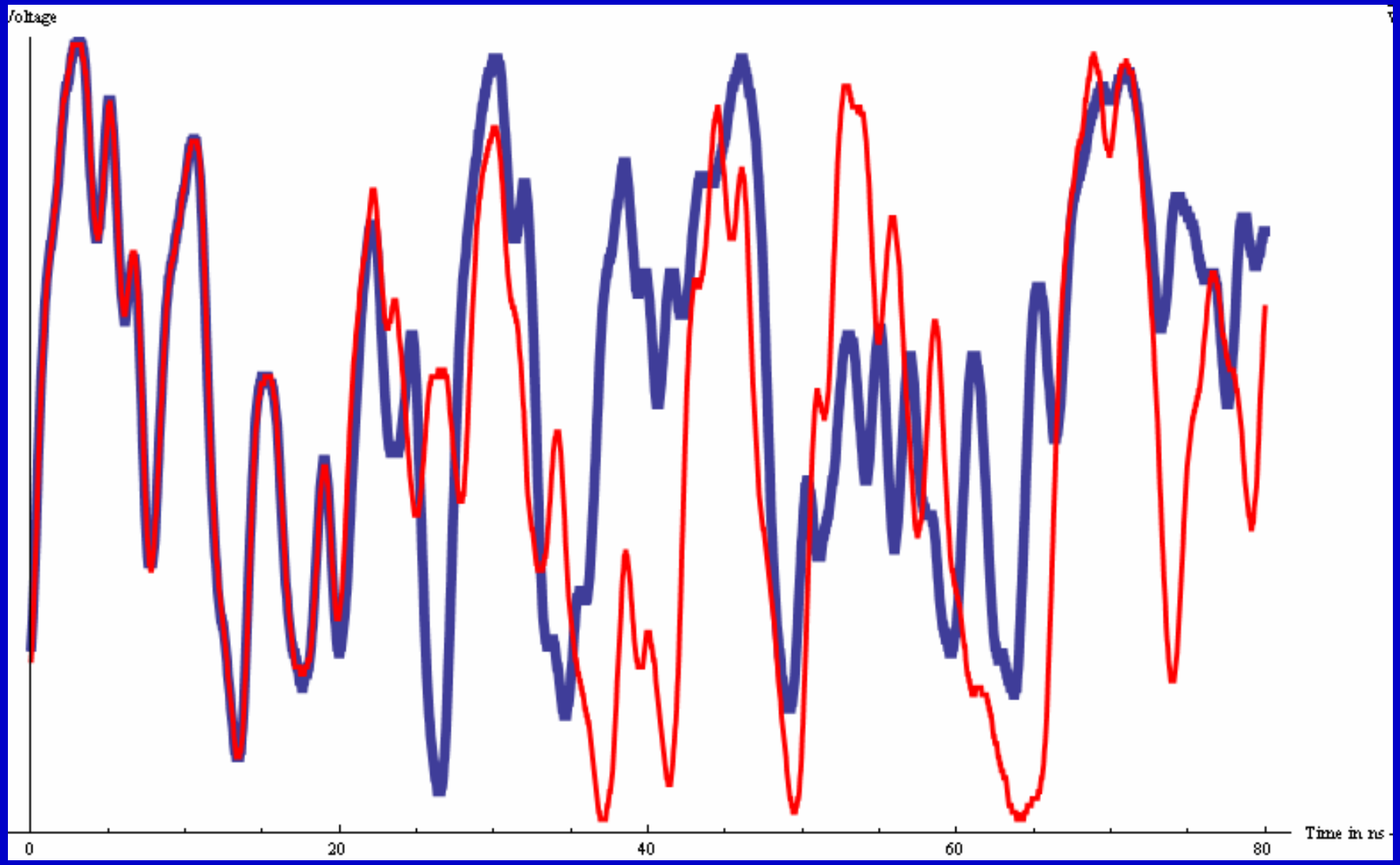
A Restarting FIRO



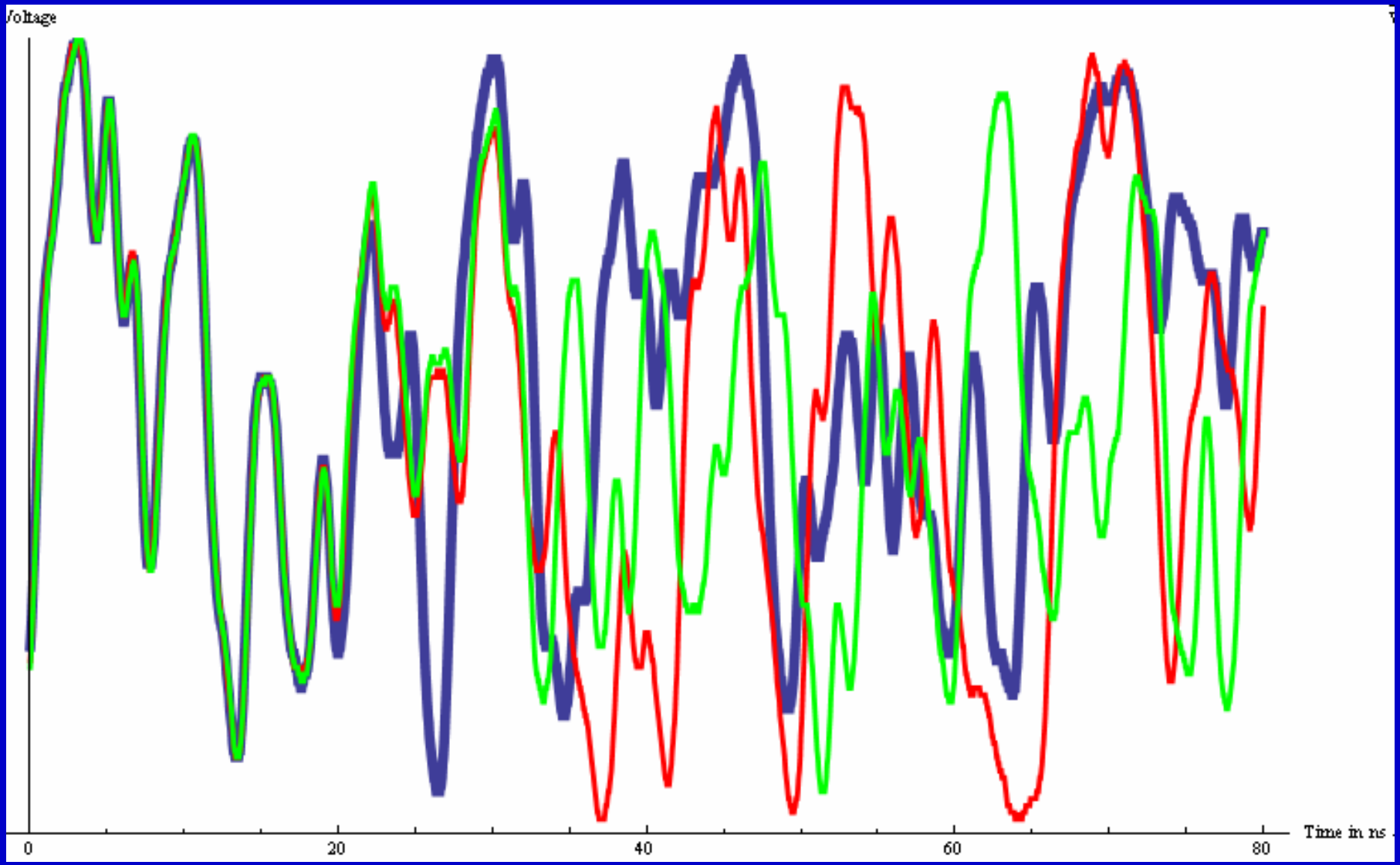
FIRO Restarts from Identical States (I)



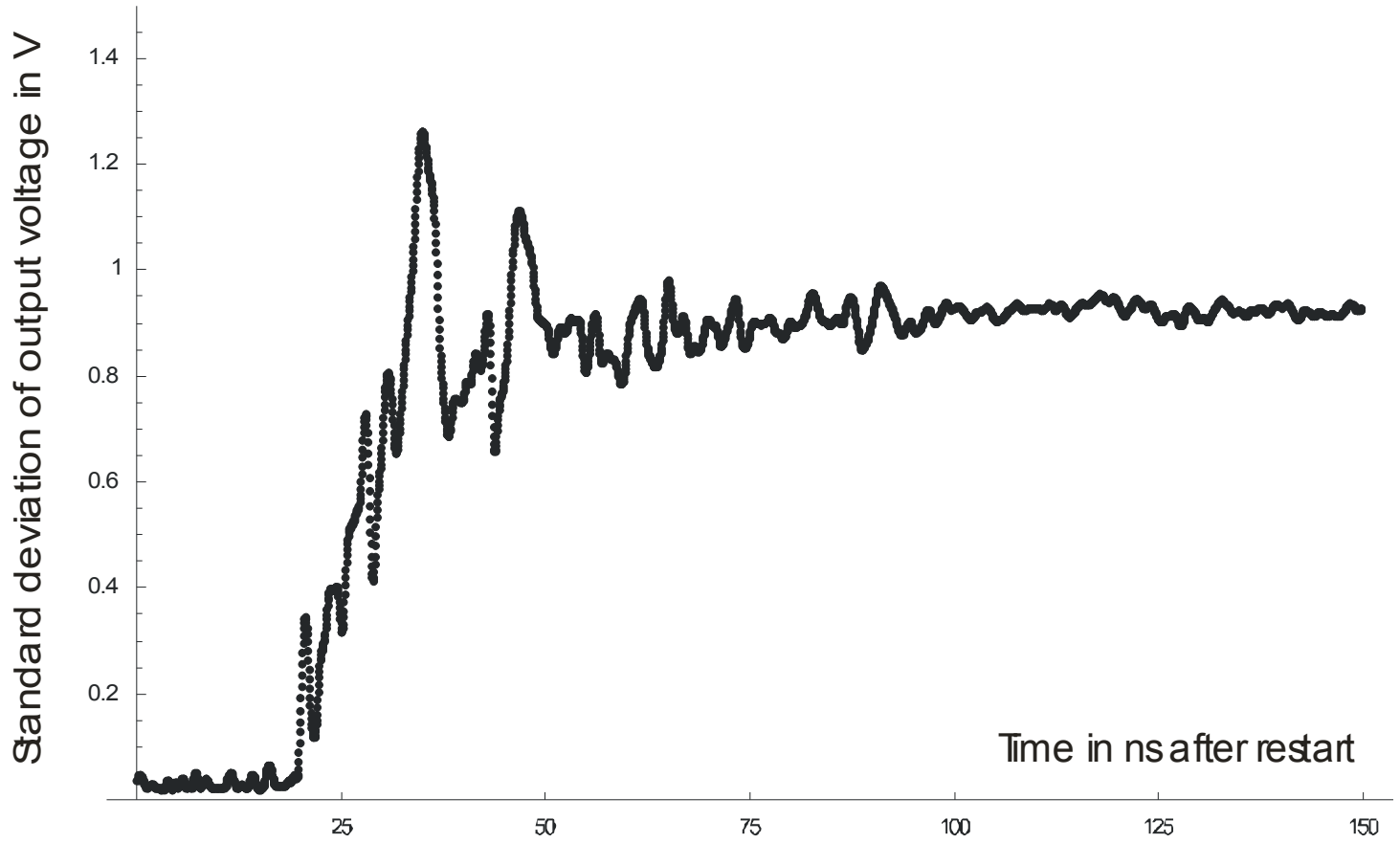
FIRO Restarts from Identical States (II)



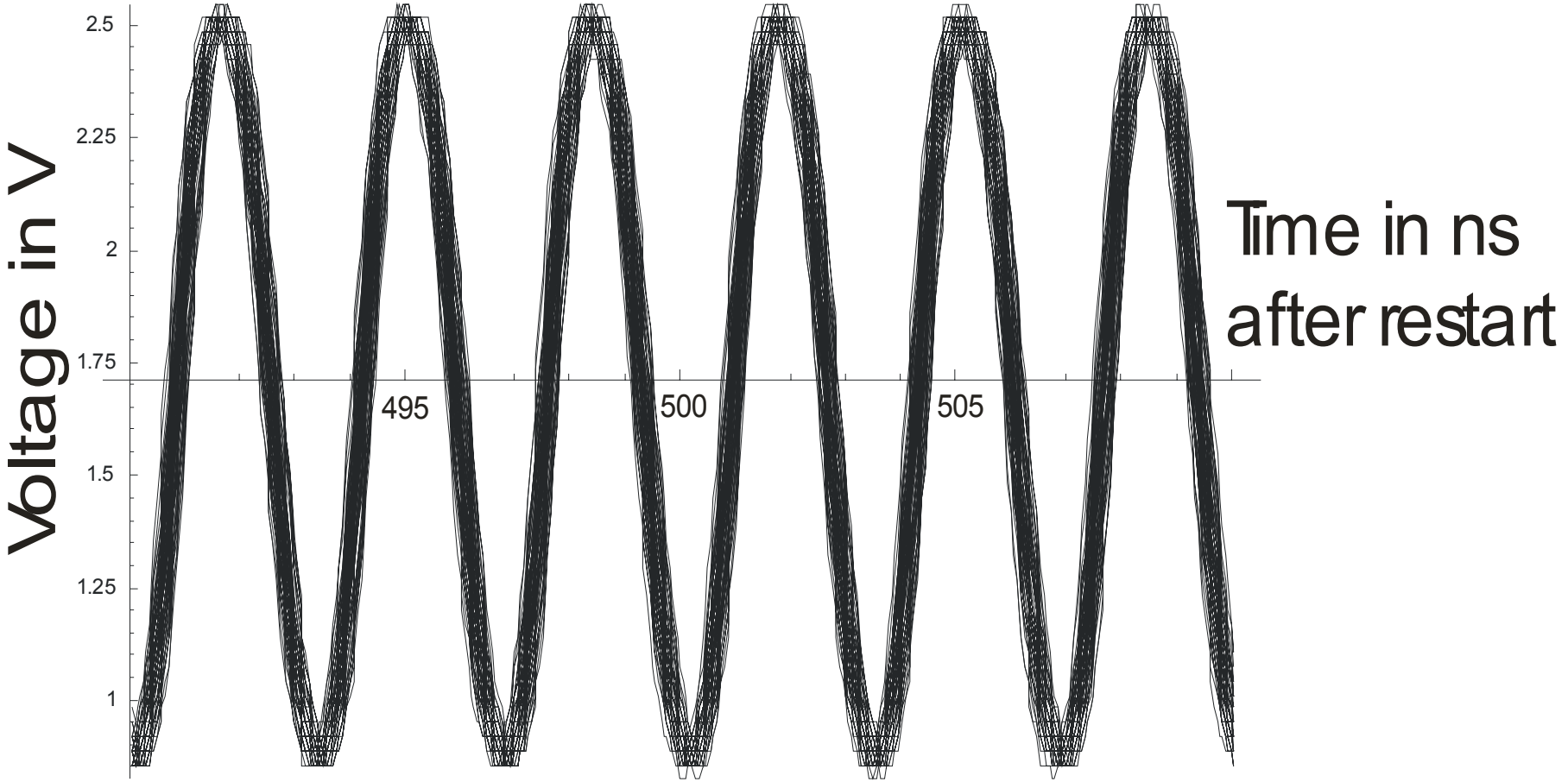
FIRO Restarts from Identical States (III)



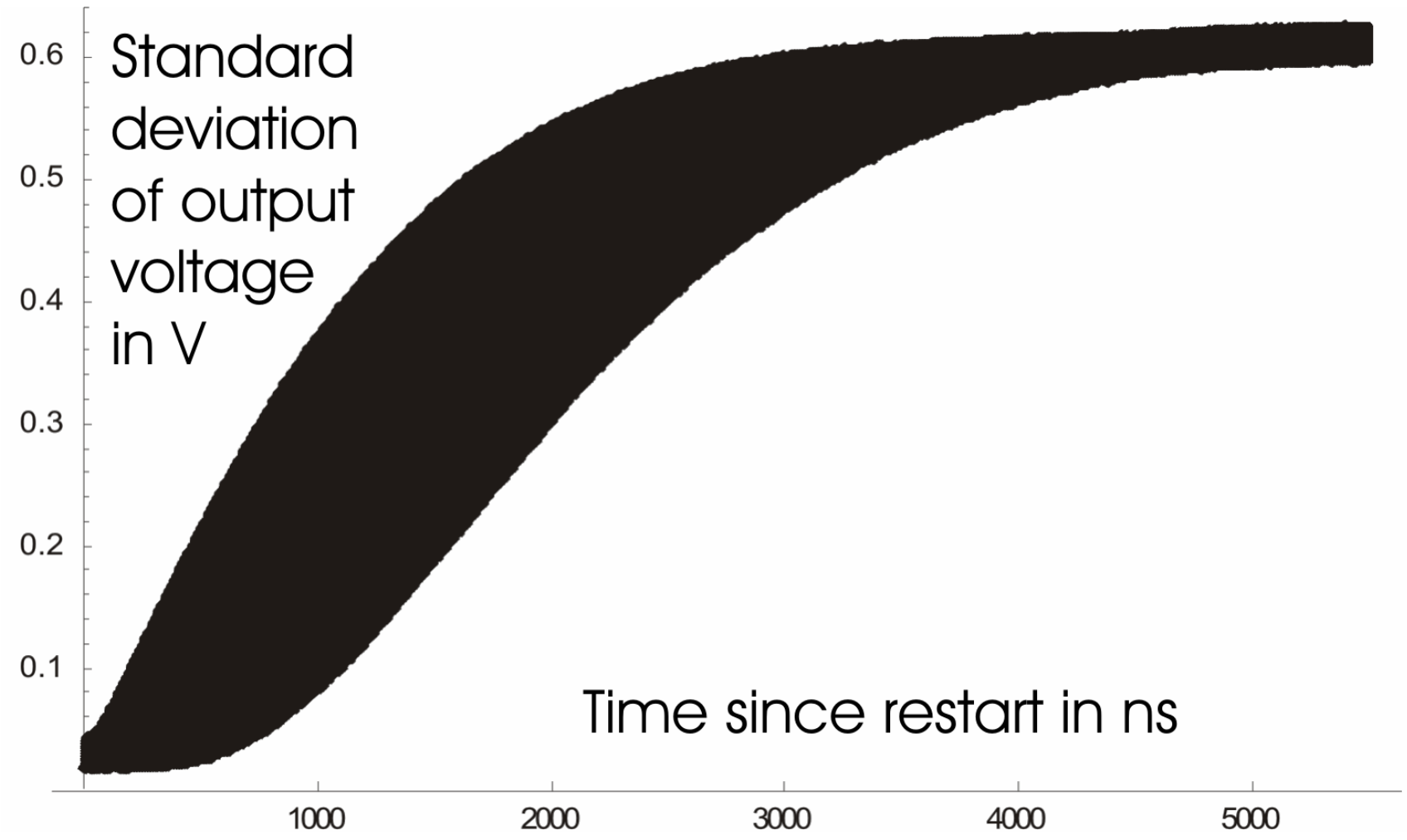
Standard Deviation of 1000 FIRO Restarts



Restarting a RO (length 3) 100 Times



Standard Deviation of 1000 RO Restarts



Extracting Random Bits

Up to now we just considered “analogue“ random voltages, but we want random bits

How to extract them?

A RO Based TRNG Considered “State of the Art“ by one CHES 2007 Reviewer

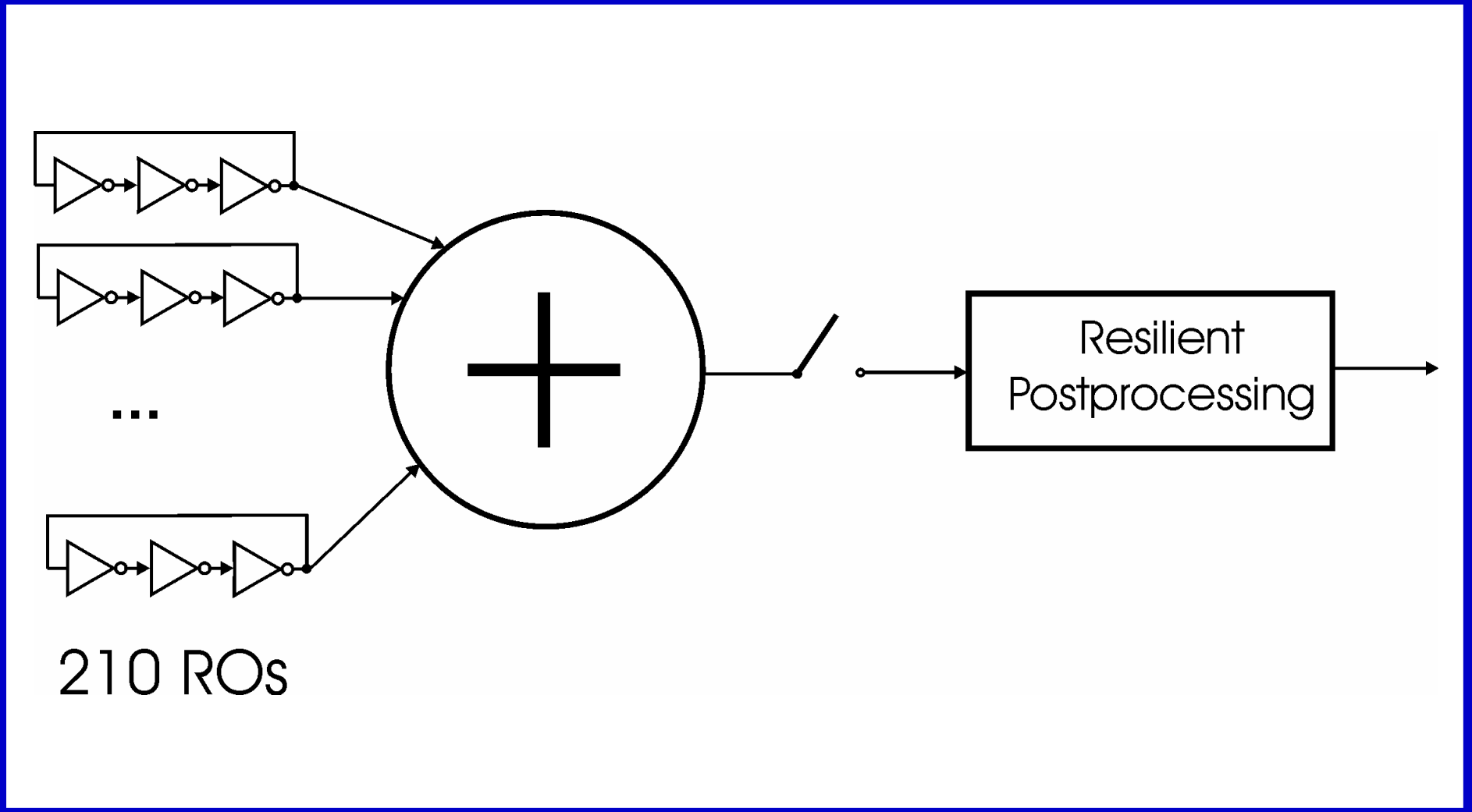
Schellekens, Preneel, Verbauwhede: “FPGA Vendor Agnostic True Random Number Generator,” FPL 2006, August 2006

based on

Sunar, Martin, Stinson: “A Provably Secure True Random Number Generator with Built-in Tolerance to Active Attacks,“ IEEE Trans. Computers, vol. 56(1), pp. 109-119, Jan. 2007

The Schellekens, Preneel, Verbauwhede Design

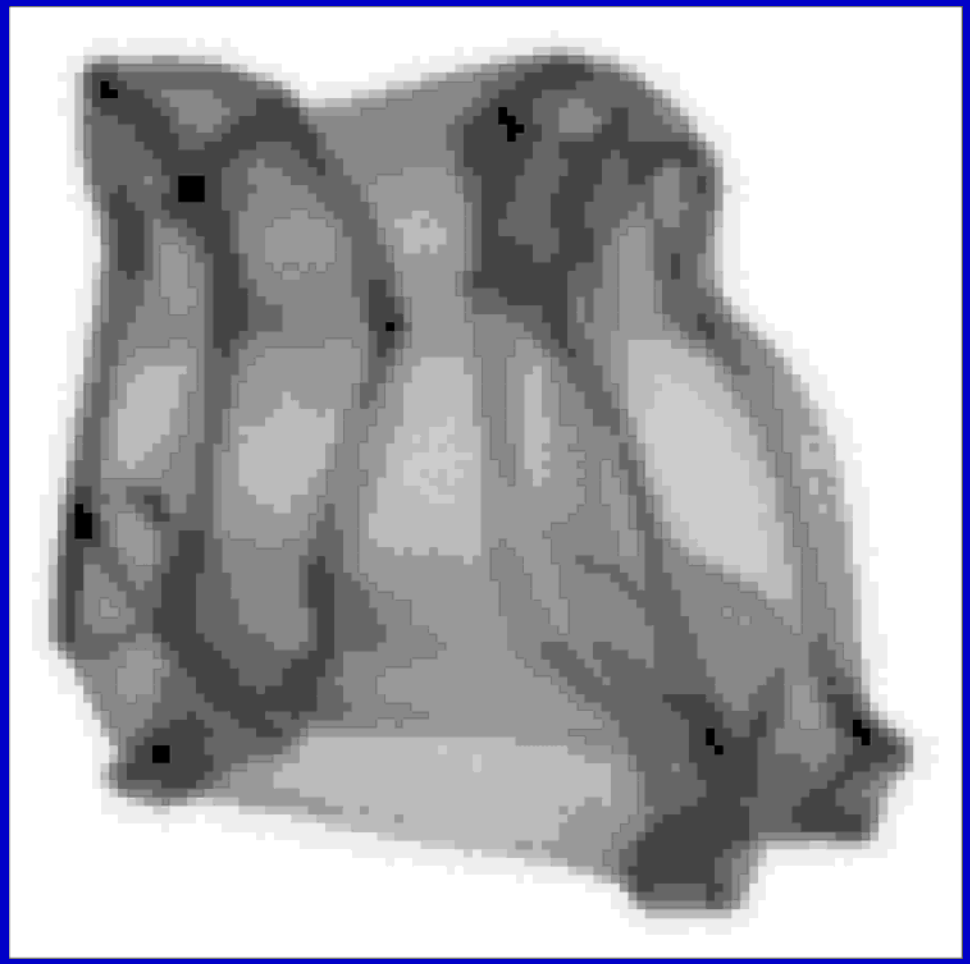
Enormous power consumption and gate count



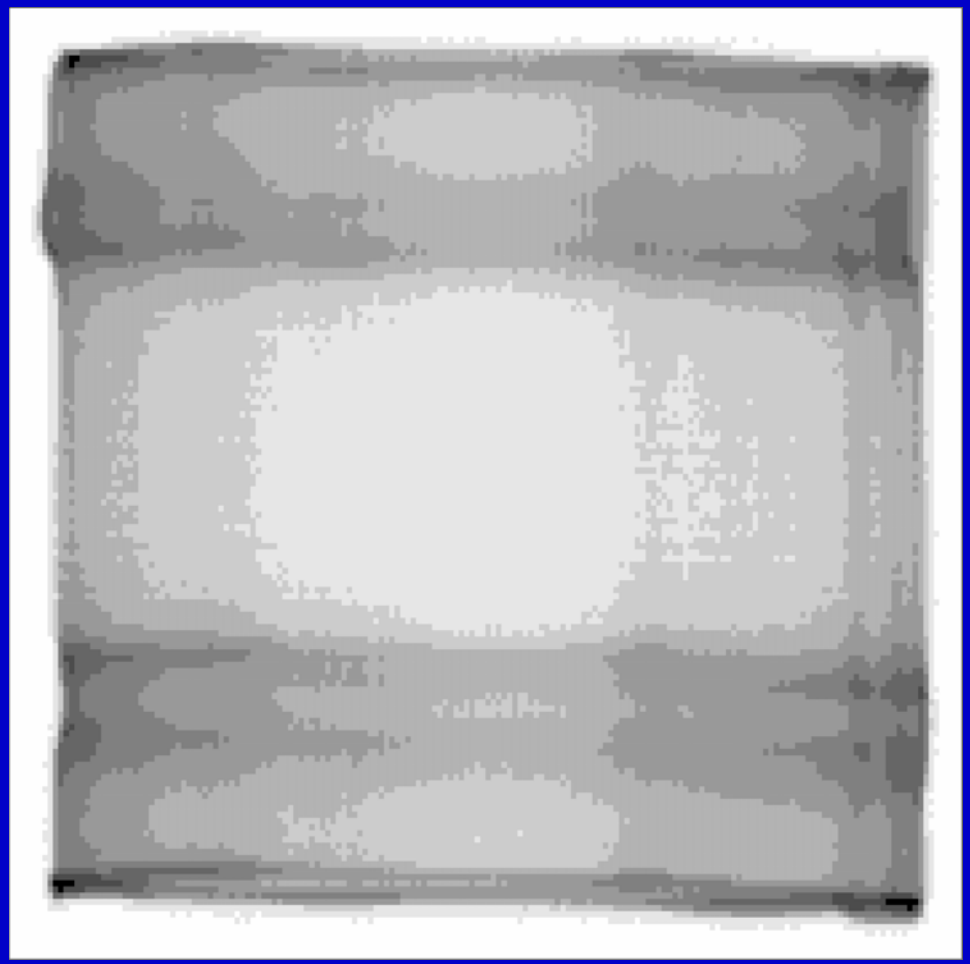
4 Reasons why this Fails

- Based on a completely unrealistic assumption on jitter, that a RO has a perfect built-in clock and that jitter occurs only around the transitions of this clock
- The 210 ROs are implicitly assumed to be statistically independent. My experiments show they are not (coupling)
- No chip can compute the XOR at the speed required. For the Leuven FPGA design, the XOR of 210 ROs would have an average frequency of 69.9 GHz (70 transitions per gate delay)
- Even if the XOR could be computed, it could not be sampled, because the setup- and hold-times are violated
  (On average, 23.8 transitions in 0.17 ns, while the signal must be constant)

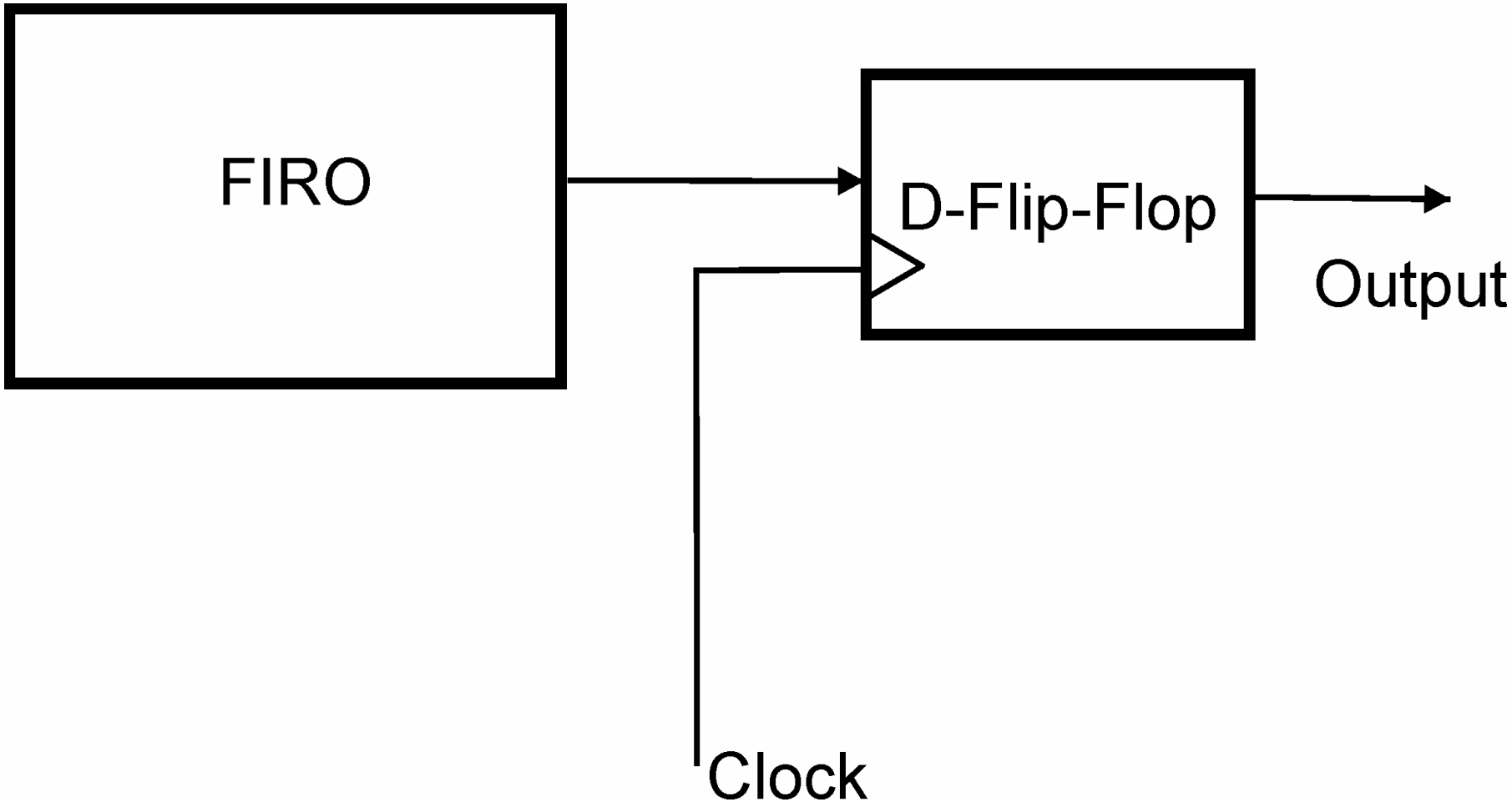
Lissajous-like Figure for 2 ROs (Length 13) on the Same FPGA Board



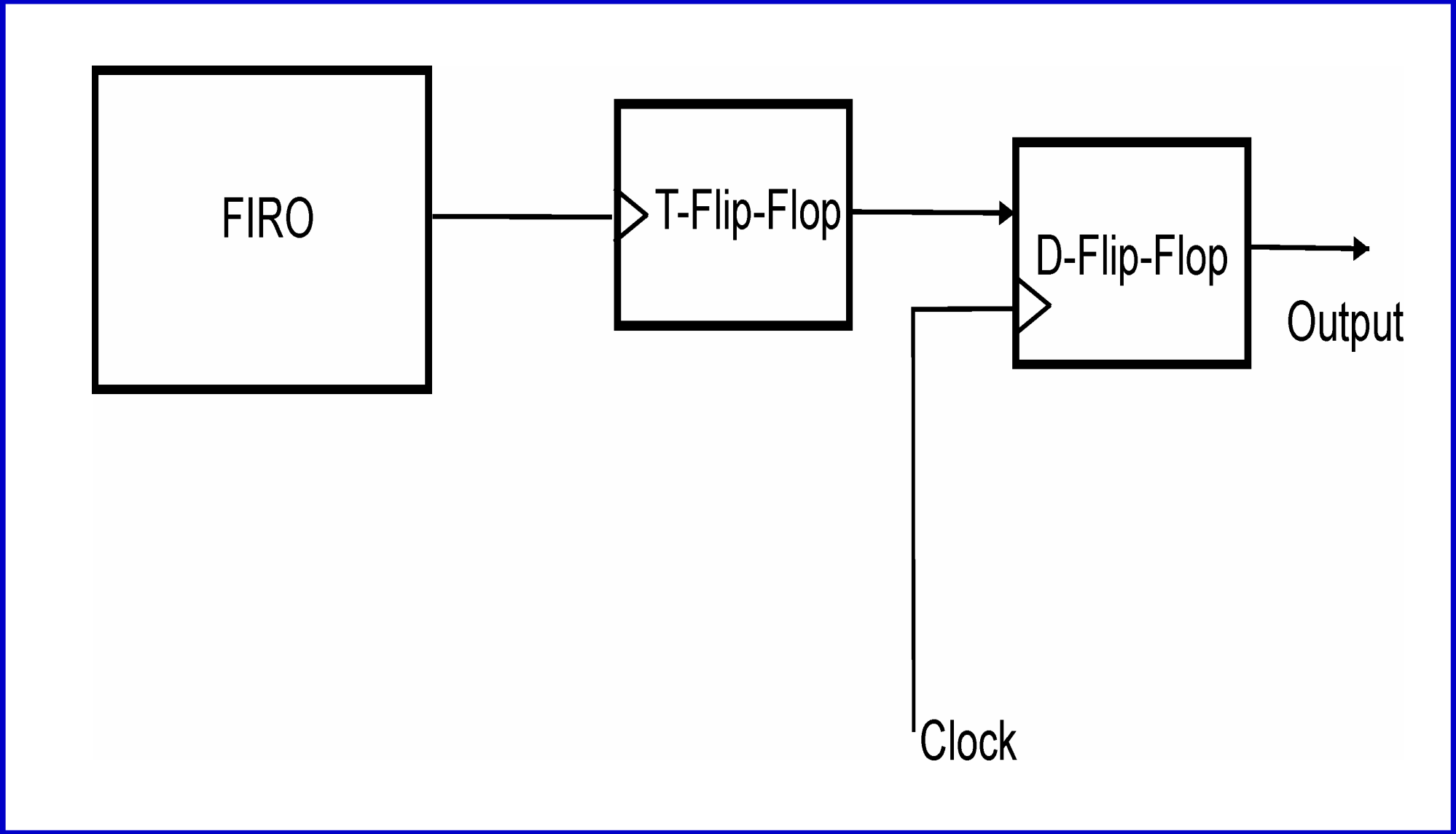
Lissajous-like Figure for 2 ROs (Length 13) on Two FPGA Boards



Direct Sampling of FIROs/GAROs



Sampling with Intermediate Toggling Flip-Flop

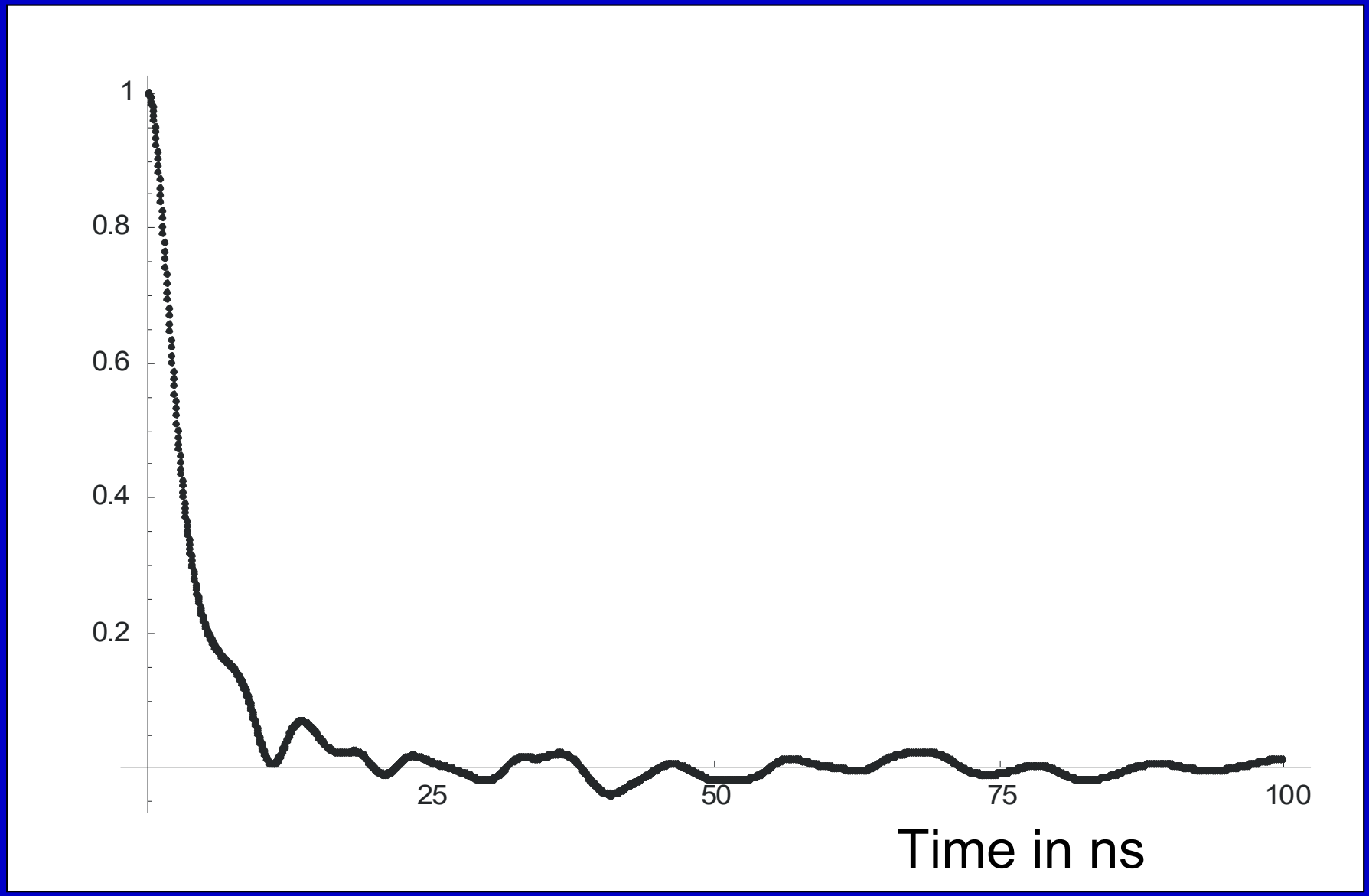


Restarting or Continuous Operation?

Restarting: Needs time for transitory voltages to settle down. But, output bits are guaranteed to be statistically independent, so postprocessing is easy

Continuous FIRO or GARO operation: Independence plausible for reasonable sampling rates. Statistical tests may be fooled by pseudo randomness

Autocorrelation for Continuously Running FIRO



Speeds Achieved on FPGA

Restarting FIRO (run for 60 ns, stop for 40 ns, sampled with intermediate toggling flip-flop):

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

Continuously running FIRO (with toggling FF):

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

Squeezing Out Almost Twice the Speed

When we sample both directly and with the intermediate T-flop-flip, we double the raw data rate to 14.28 Mbits/s

The two bits from one run are (weakly) dependent, but the pairs from different runs are independent

Suitable postprocessing can get almost all the Shannon entropy, which was 1.933 for each pair

The theoretical output data rate is thus 13.8 Mbits/s

Power Consumption

- For FIRO of length 15 on CMOS ICs 74HCTXX : 3 to 4 times the power consumption of a RO (depending on feedback)
- Higher power consumption than RO, but the FIRO entropy rate is orders of magnitude higher

Why FIROs and GAROs are so Good

- The number of logical transitions per time is proportional to the length, as opposed to RO, so longer FIROS and GAROs produce more jitter than ROs
- The complex feedback spreads small random variations quickly over the whole design and makes randomness easier to extract by sampling
- The complex high-frequency output signal could produce frequent metastability effects in the sampling flip-flop

Conclusions

- *FIROs and GAROs are by far the most efficient known way to produce true random numbers with digital gates only*
- Restarting makes the true randomness property testable
- Understanding in detail how the randomness is produced in FIROs and GAROs is a worthwhile topic for further research