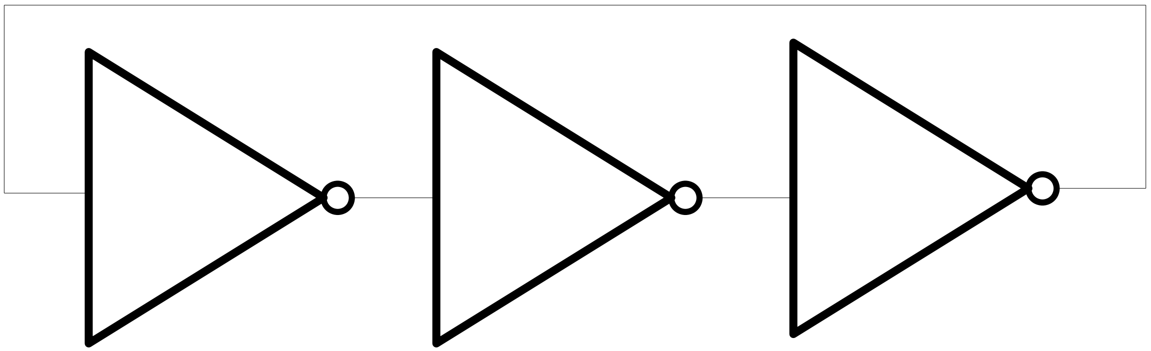
On Jitter in Very Long Ring Oscillators

Markus Dichtl

Ring Oscillator Terminology



In a ring of an odd number of logical inverters, there is, at any time, at least one inconsistency, that is an inverter whose input and output are equal.

Of course, the inconsistency moves cyclically through the ring.

This talk treats on all but one pages ring oscillators with exactly one inconsistency.

Why Very Long Ring Oscillators?

- Not relevant for practical applications
- Many properties are easier to measure, e. g. jitter
- Surprising properties

Important Insights on Dependent Ring Oscillator Jitter

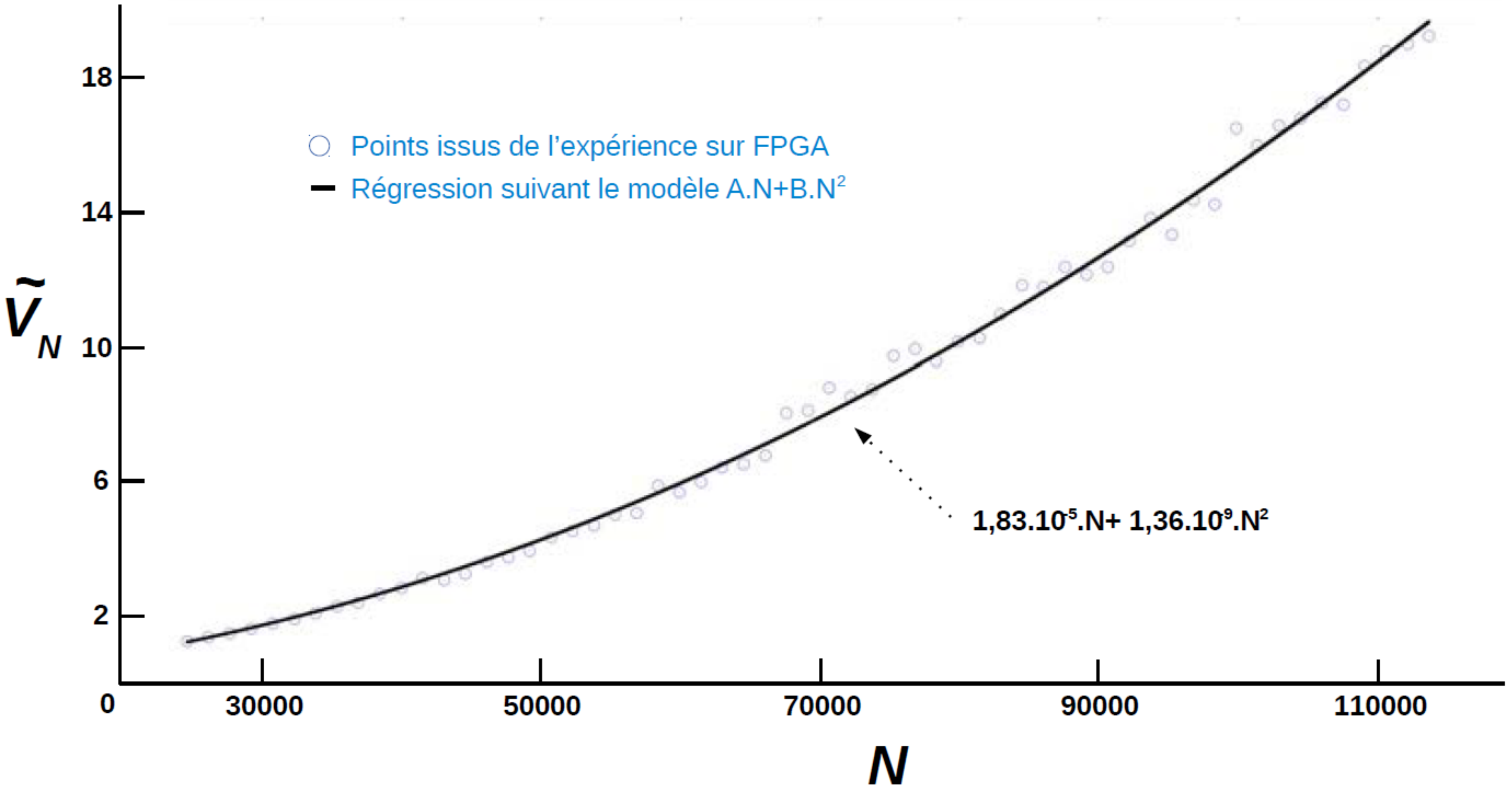
Richard Newell, Cryptarchi 2011: Measurement of FPGA ring oscillator noise, and analysis using the Allan Variance method

Patrick Haddad, 2015: Caractérisation et modélisation de générateurs de nombres aléatoires dans les circuits intégrés logiques (PhD. Thesis)

Approach for phase jitter in n periods: $\sigma^2(n) = a*n+b*n^2$

Problem: When Pascale Boeffgen and M. Dichtl tried $\sigma^2(n) = a*n+b*n^2+c$,
a turned negative.

The Key Figure from Patrick Haddad’s Thesis

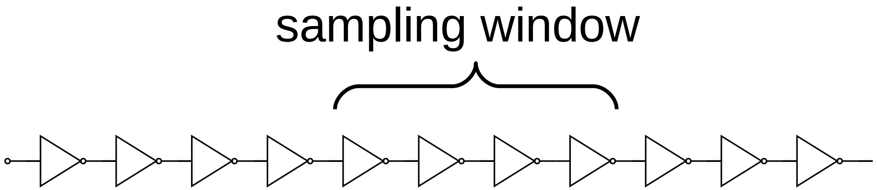


Sampling Windows

A sufficiently long window of subsequent inverters is defined.

All outputs of the inverters in the window are sampled at each clock.

The window must be long enough such that each passage of the inconsistency gets sampled at least once, that is the time needed to pass the window must be larger than the time between samples.



sampling window

Ring Oscillator and Sampling Parameters Used

Experiments on an Arty 100 board, based on Artix 7

Ring of 10001 inverters
(well, one is defined by **ringo(0) <= (not ringo(10000)) and run**)

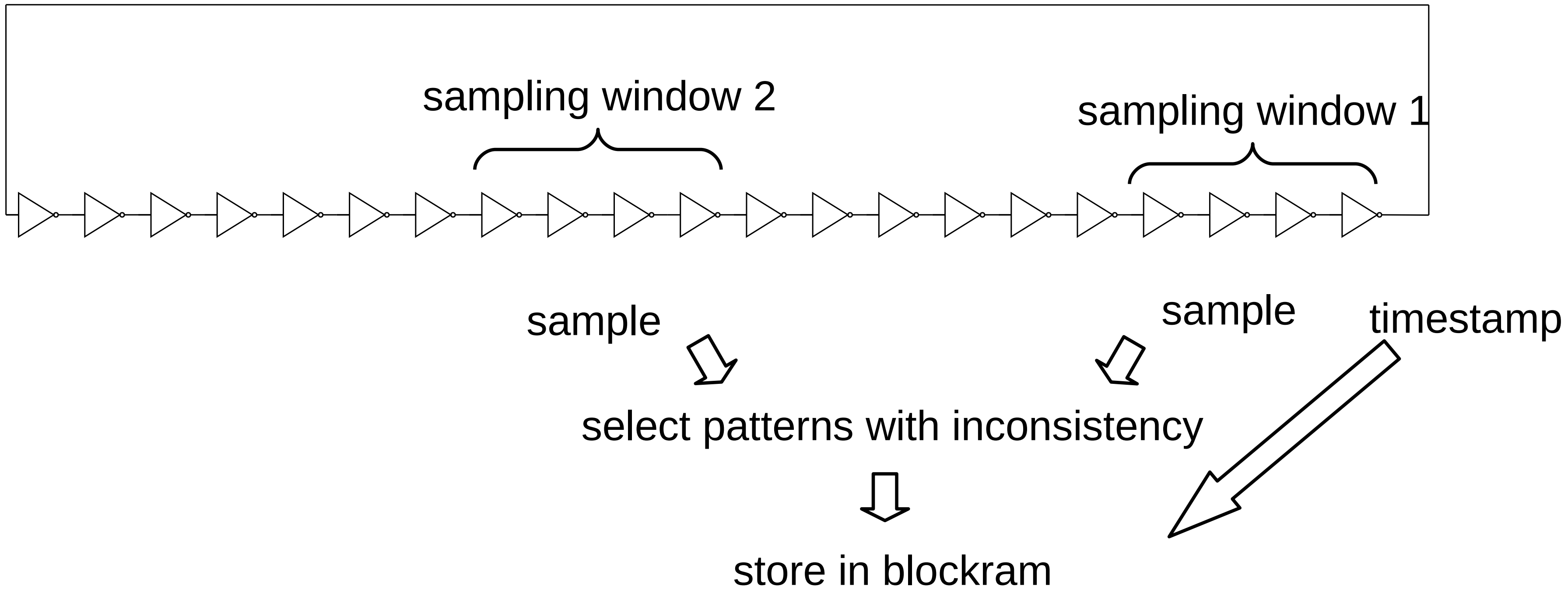
2 sampling windows of length 32:
output of inverters 10000 downto 9969 and 5000 downto 4969

Sampling frequency 100 MHz

Only samples including the inconsistency are stored in blockram with a time stamp

First set of experiments: Ring oscillator is oscillating permanently, after each sequence of measurements a break of 1 s for random phase

Schematic



sampling window 2

sampling window 1

sample

sample

timestamp

select patterns with inconsistency

store in blockram

Subsequent Bit Patterns Sampled

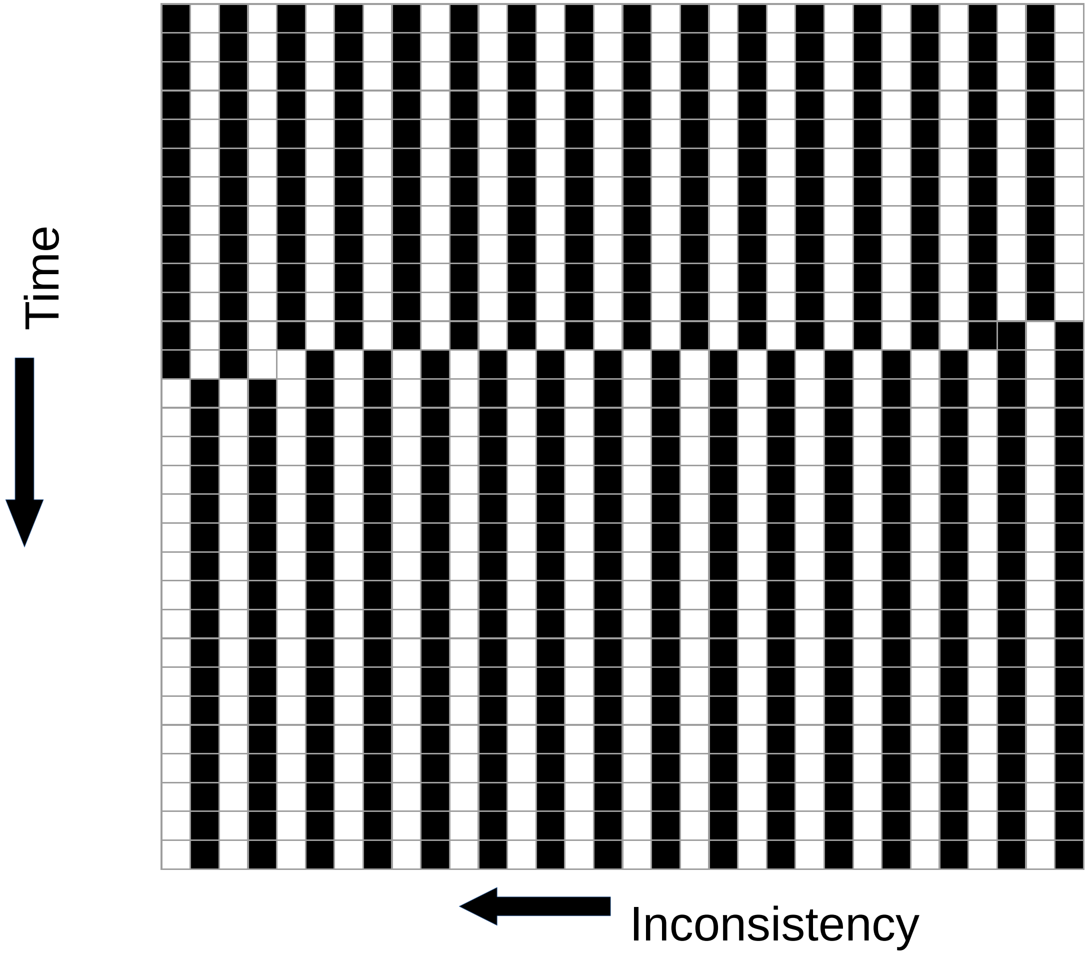
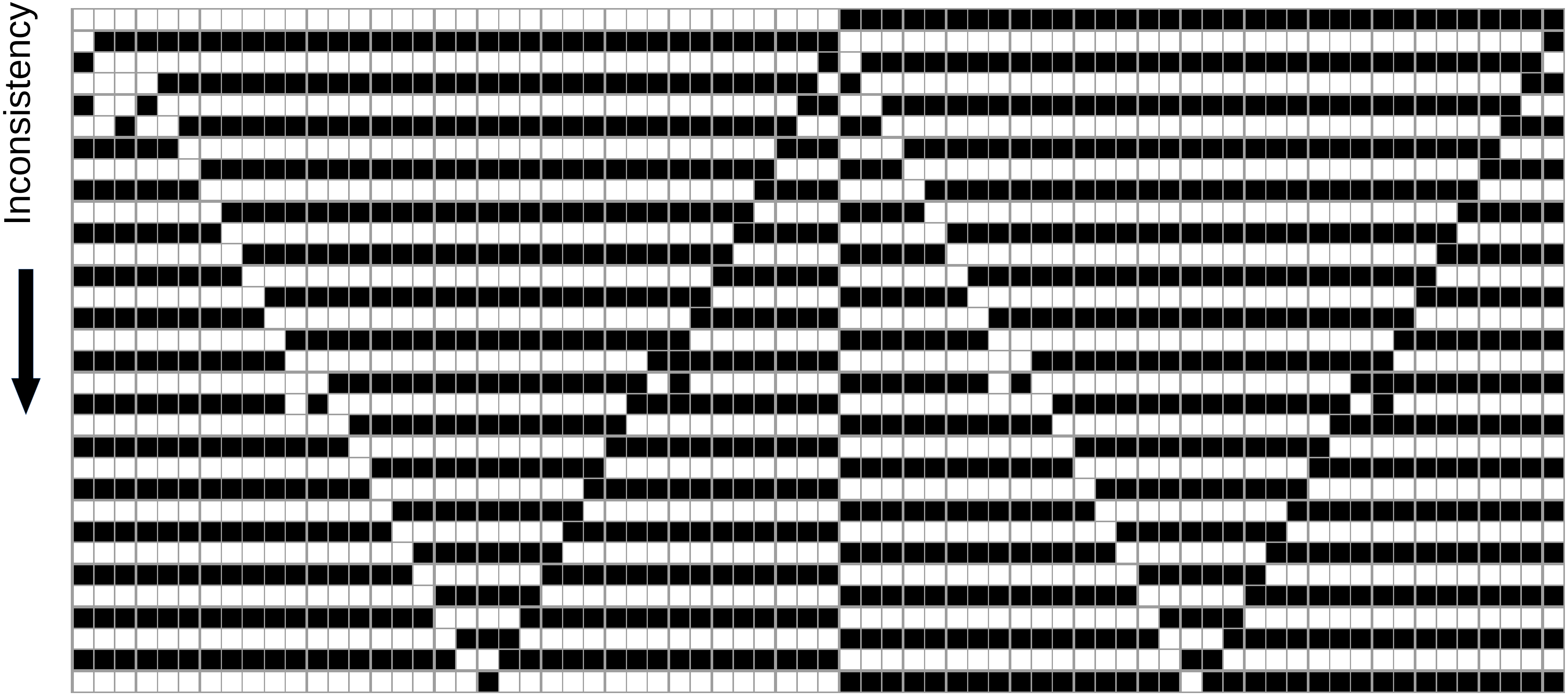


Figure not from current experiments!

As a matter of fact, trivial 01 patterns are not stored.

The Bit Patterns which Occurred



Half Periods in Gory Details

The duration of a half period should be half the period, but …

Evaluating 6573 measurements (all first full periods for both sampling windows) gives surprising details.

	Window 1	Window 2
Leading bit 0	3338.90 ns	3337.91 ns
Leading bit 1	3336.36 ns	3337.25 ns

Might This Be a Statistical Artefact?

No!

Window 1, leading bit 0

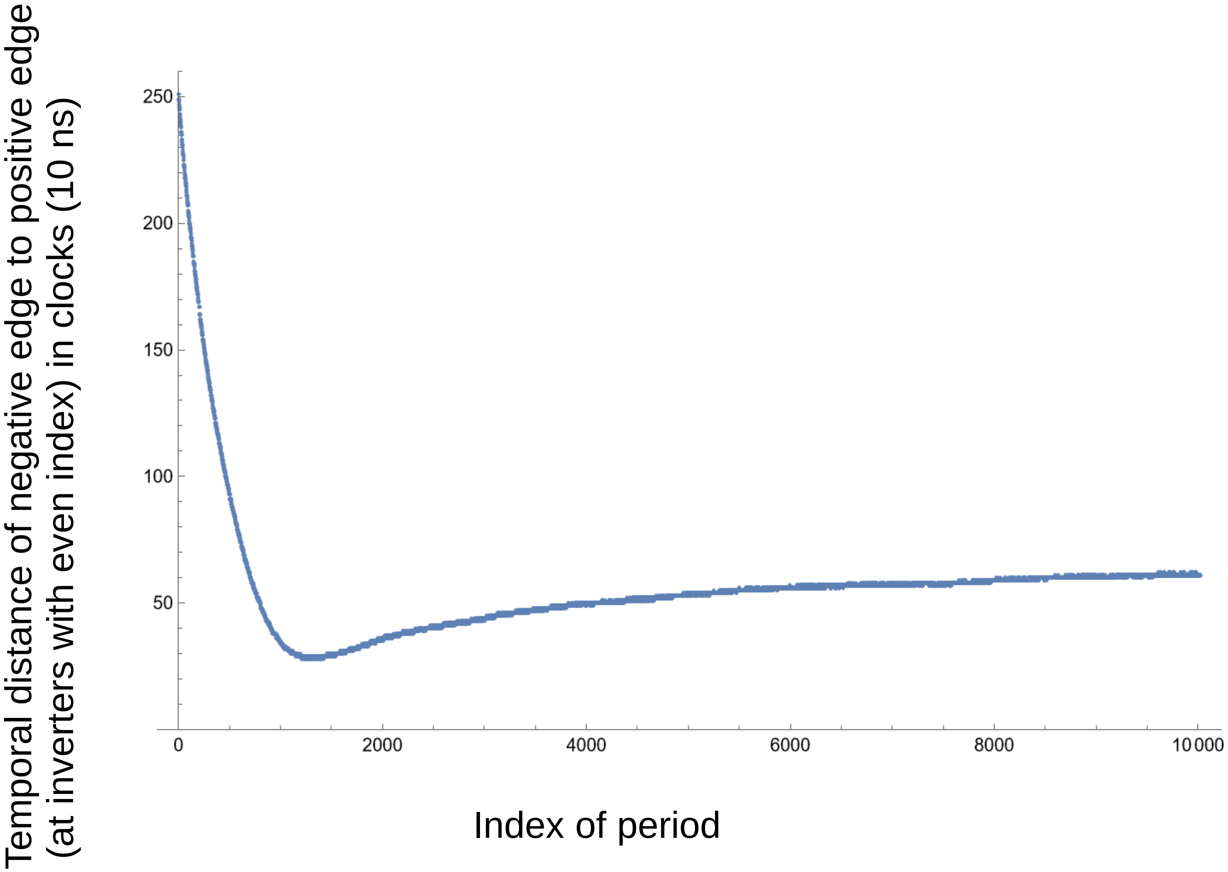
333 clocks 724 times, 334 clocks 5846 times, 335 clocks 3 times

Window 1, leading bit 1

333 clocks 2391 times, 334 clocks 4182 times

The sums of the corresponding half periods agree to within 0.104 ns.

But What Happens in Ring Oscillators of Even Length?



Ring of length 10000 with two inconsistencies

One should expect a random walk.

Initially, the negative edge starts to catch up quickly.

What slows it down finally?|

How do the edges interact?

Improving Temporal Resolution

Jitter should be measured with a resolution way below the 100 MHz clock

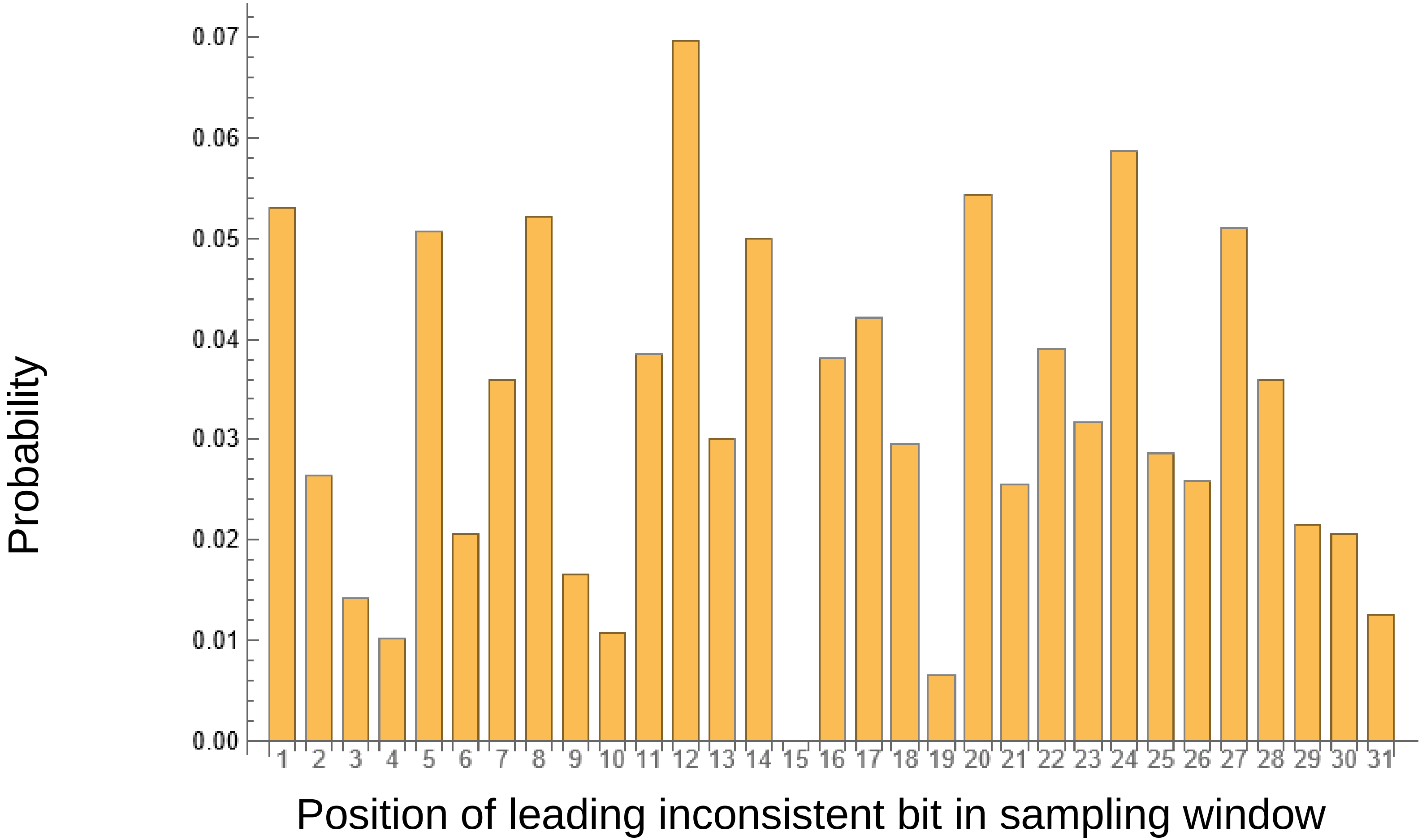
The position of the inconsistency in the sampling window provides phase information

Naive approach: Time proportional to index of inverter

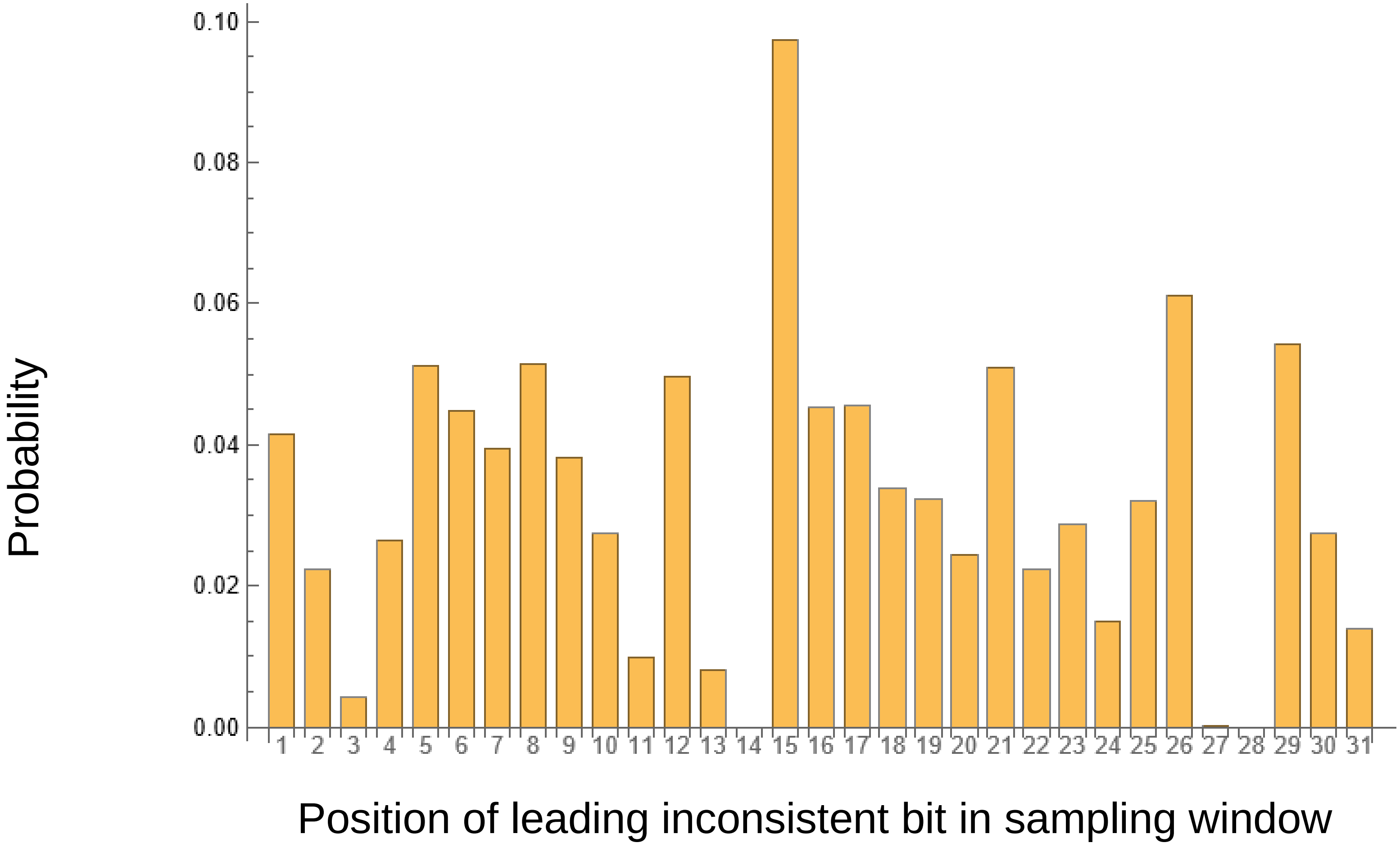
Better approach: Probabilistic calibration, the time the sampling window is in a certain state is proportional to the probability to be sampled in that state. (Of course to be done for each sampling window separately.)

Unfortunately, taking into account the differing half period lengths is not helpful, as this would influence the resolution by about 30 ps, whereas the best we can hope for is 10 ns /32 ≈ 300 ps

Probabilities of Leading Inconsistent Bits in Window 1



Probabilities of Leading Inconsistent Bits in Window 2



From Probability to Time

The time spent in the sampling windows can be determined by the average ratio of single/double hits in one passage of the window.

Window 1: 11.426 ns

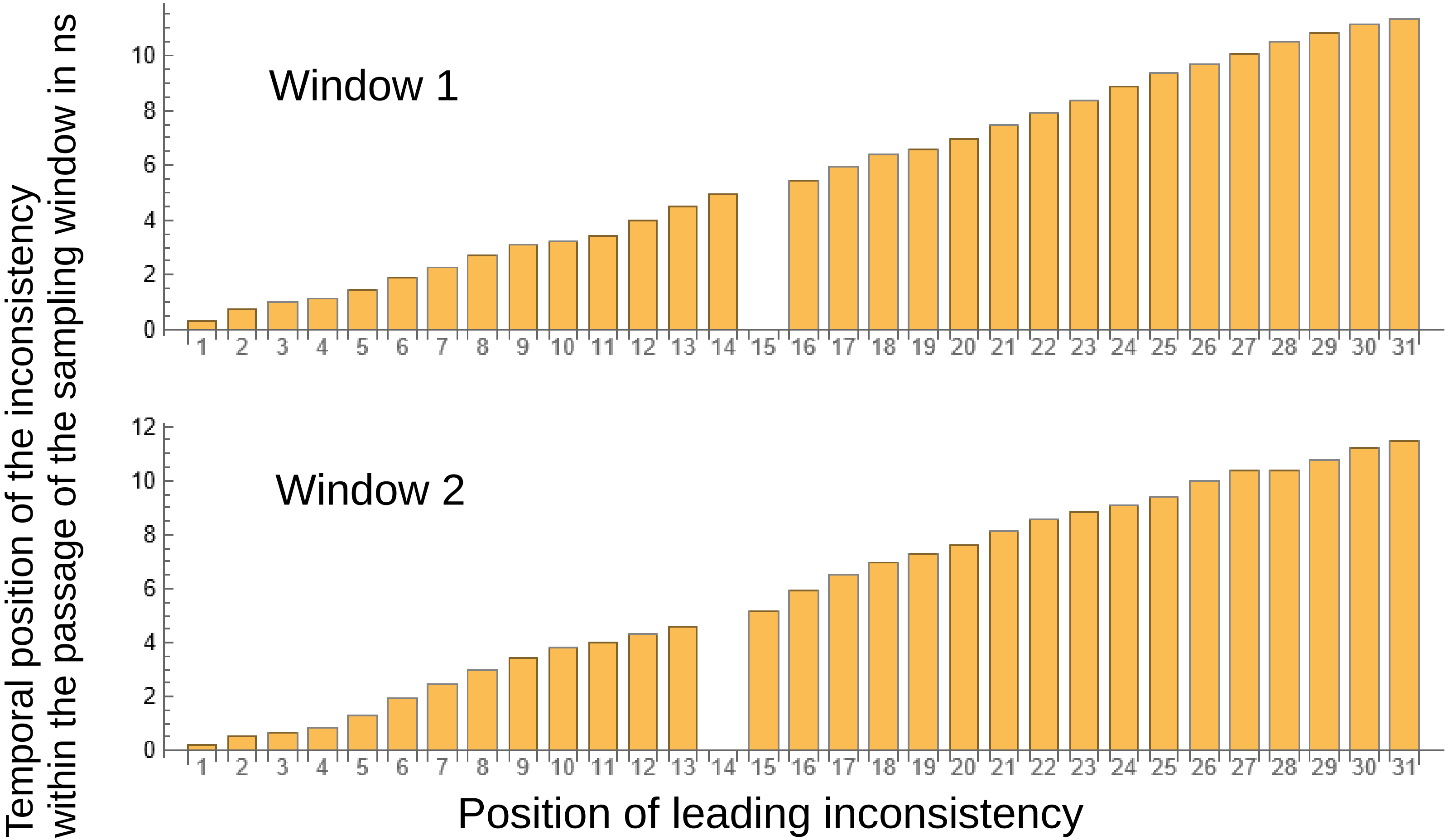
Window 2: 11.607 ns

When an event is known to occur in a temporal period, it is assumed to occur at the centre of the interval, in order to minimize the error.

Now one can convert the probabilities of the leading inconsistency to time by accumulating them and scaling to the duration of the passage.

A trivial correction (subtracting 10 ns) is needed when the resulting value is above 10 ns.

Accumulated Times in Sampling Windows



Checking the Precision of the Time Measurements

When we have hits at two subsequent clocks, we know that they are 10 ns apart

But we can also compute their temporal separation by the positions of the inconsistencies.

Window 1: Mean   9.98 ns, standard deviation 0.195 ns
Window 2: Mean 10.20 ns, standard deviation 0.309 ns

Finally Jitter!

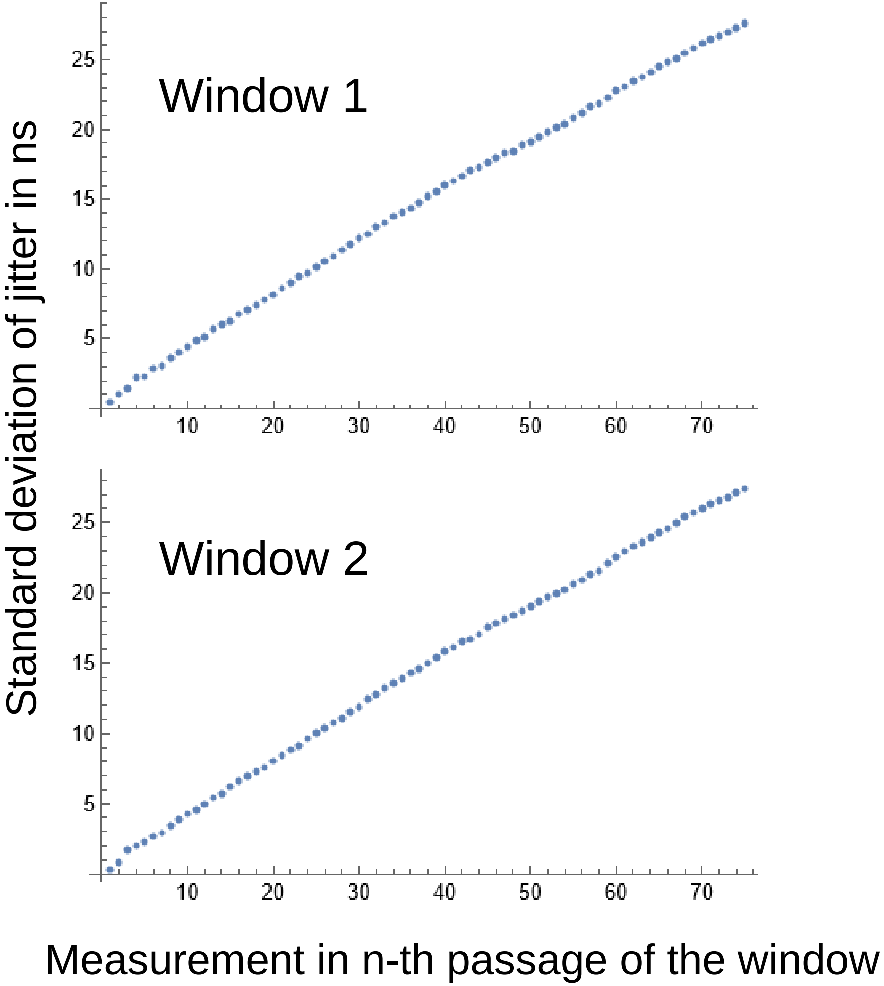
Sliding a switch makes the Arty board run in restarting mode for 100 restarts. The inconsistency starts at ringo(0). We consider the first time the inconsistency reaches window 2. If the inconsistency is sampled twice subsequently, we just consider the 2nd sample. From the position of the inconsistency, we determine when the inconsistency will leave the window:

Mean time of leaving window 2: 1653. 48 ns standard deviation: 0.27 ns
Mean time of leaving window 1: 3316.10 ns standard deviation: 0.80 ns

Mean time between windows 2 and 1:
1662.57 ns standard deviation: 0.32 ns

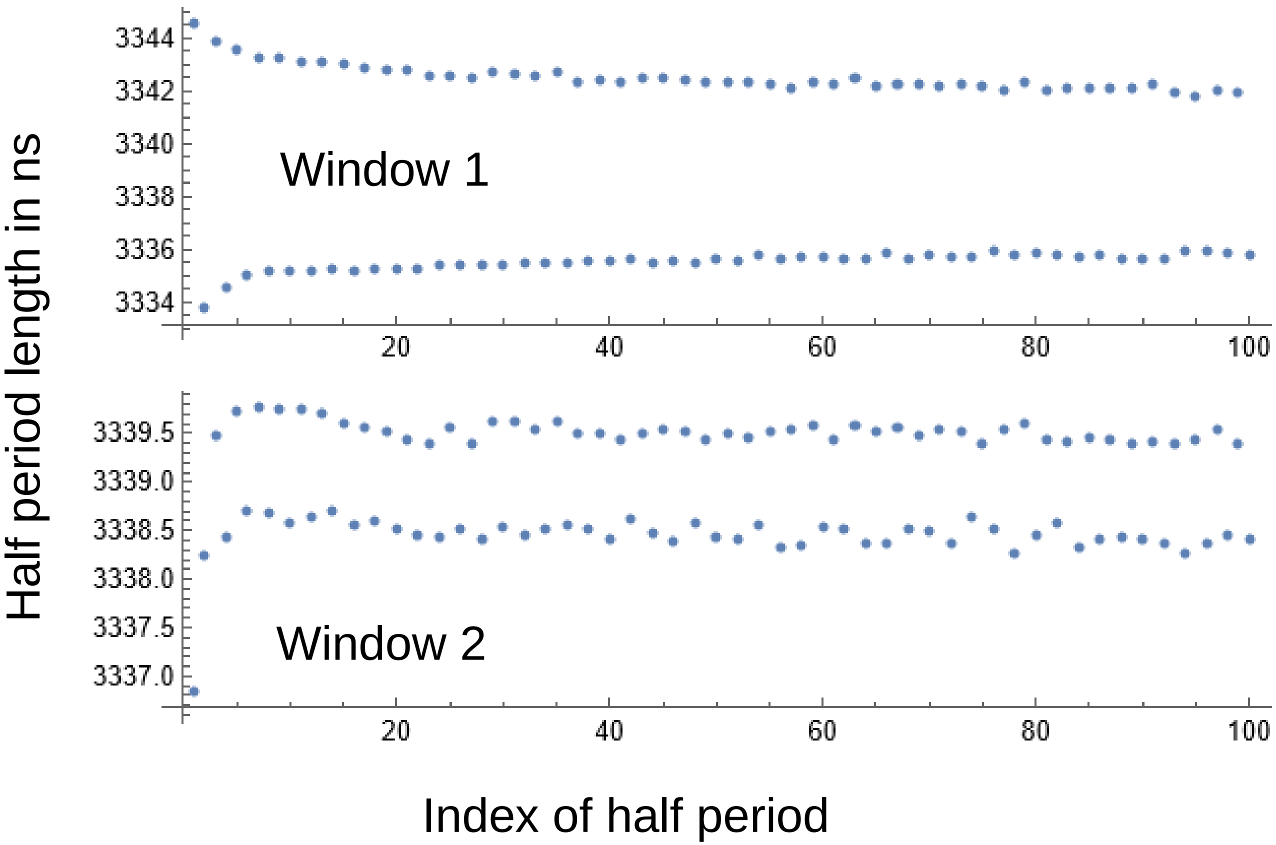
For independent jitter, the standard deviation for window 2 should be 0.42 ns!

How Jitter Accumulates



The linear behaviour shows that the accumulated jitter contributions are dependent!

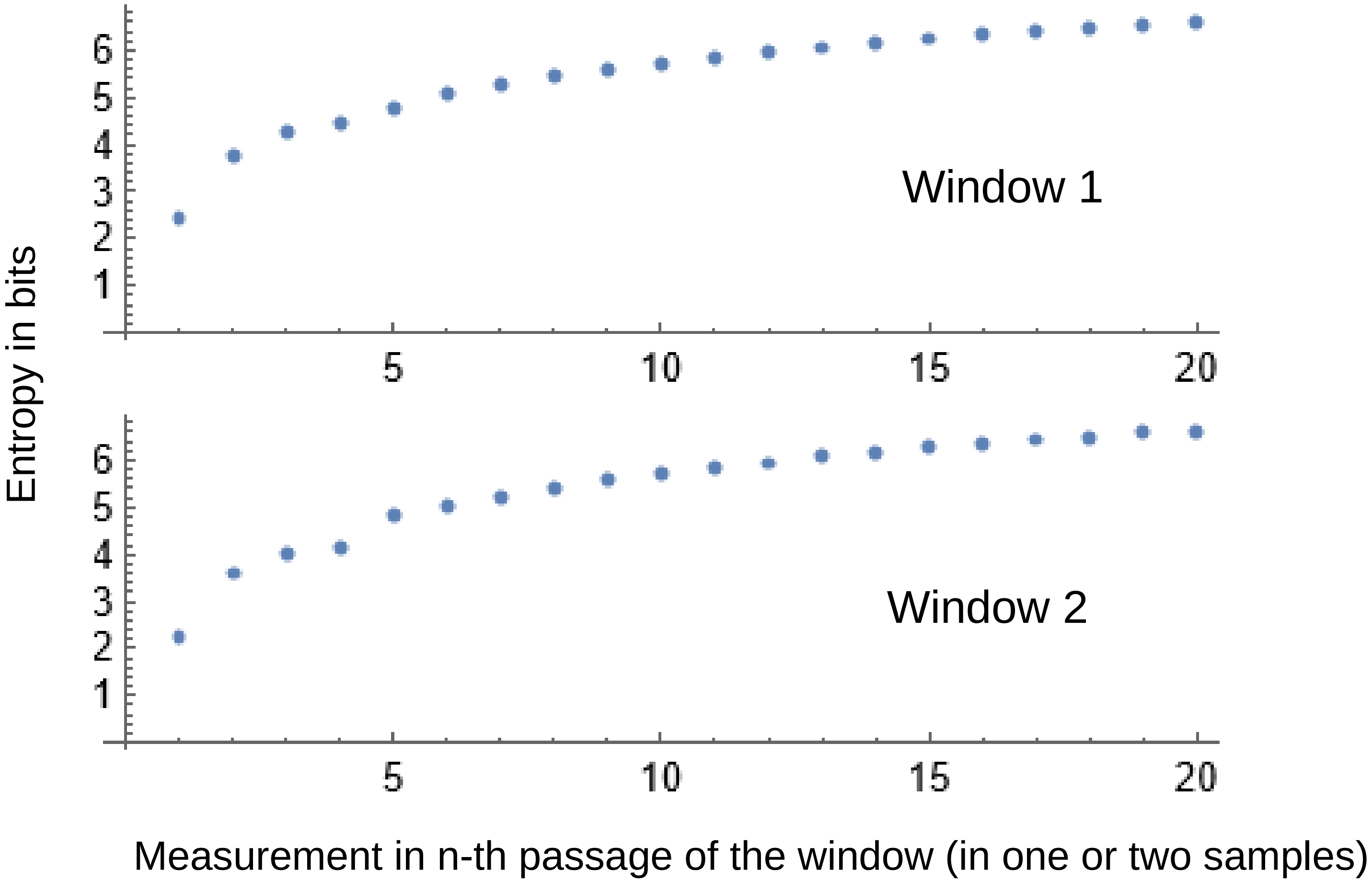
Temporal Development of Average Half Period Lengths in Restarting Mode as a Function of the Index of the Half Period



We already know the alternating lengths of half periods, but why 3 decreasing and one increasing?

What is going on here?

How Much Entropy Can One Sample in One Passage of the Inconsistency?



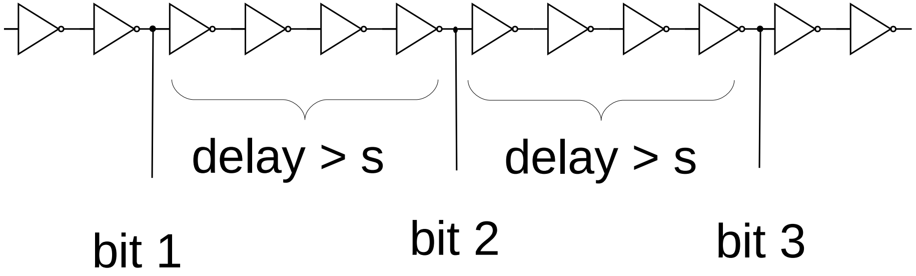
How to Avoid Setup and Hold Time Violations

Sample at three positions (assume same parity) in the ring “simultaneously”. The delay between them (including uncertainties) must be larger than the sum of setup and hold time s. Inconsistency moves from left to right in the pattern.

Bit Patterns

a ≠ b

1 2 3	correct bits	possible violations
a a a	**2** and 3	1 and 3
a a b	**1** and 2	2 and 3
a b a	impossible	
a b b	1 and **3**	1 and 2



Conclusions

How could a reasonable stochastic model for ring oscillators look like?

I have shown this conclusion several times:

TRNGs remain a challenging topic!