SIEMENS

Siemens Corporate Technology | June 2014

Techniques to Improve the Extraction of Entropy from Circuits with Random Behaviour

Markus Dichtl, Siemens AG
Bernd Meyer, Infineon Technologies AG

Funding

SIEMENS

This work was supported by the German Federal Ministry of Education and Research in the project 16BY1157 „ Zuverlässige Erzeugung echter Zufallszahlen in FPGAs“.

Partners in the **DICECUP** project:

- Technical University of Dresden
- secunet Security Networks AG
- Siemens AG

http://dicecup.de/

The general idea

The conventional approach in true random number generation:

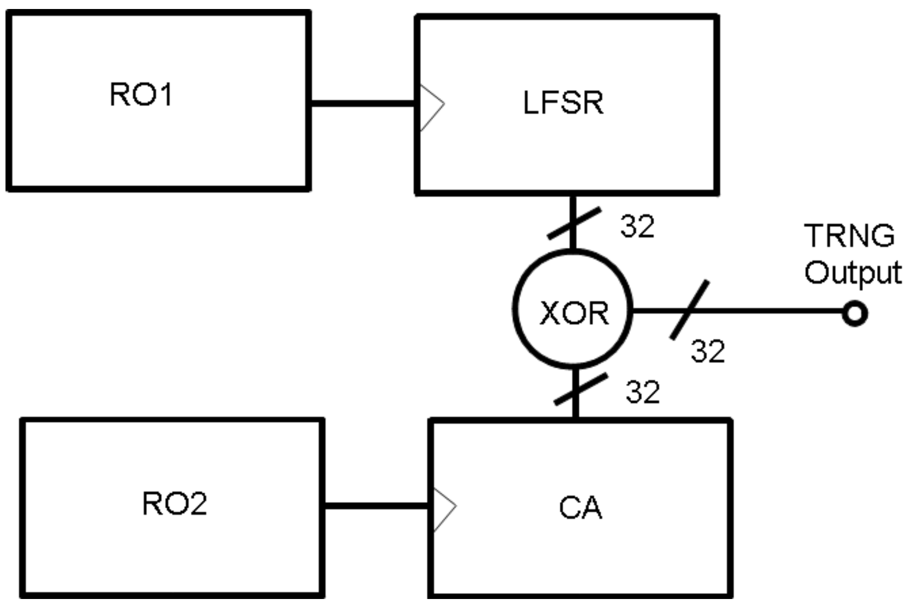
Sample a single bit, wait, sample the next bit, ...

Idea:

Be **greedier**! Try to sample multiple bits at a time! Do not care if they are dependent, postprocessing will fix this!

Prior art I

- T.E. Tkacik, *A hardware random number generator,* CHES 2002



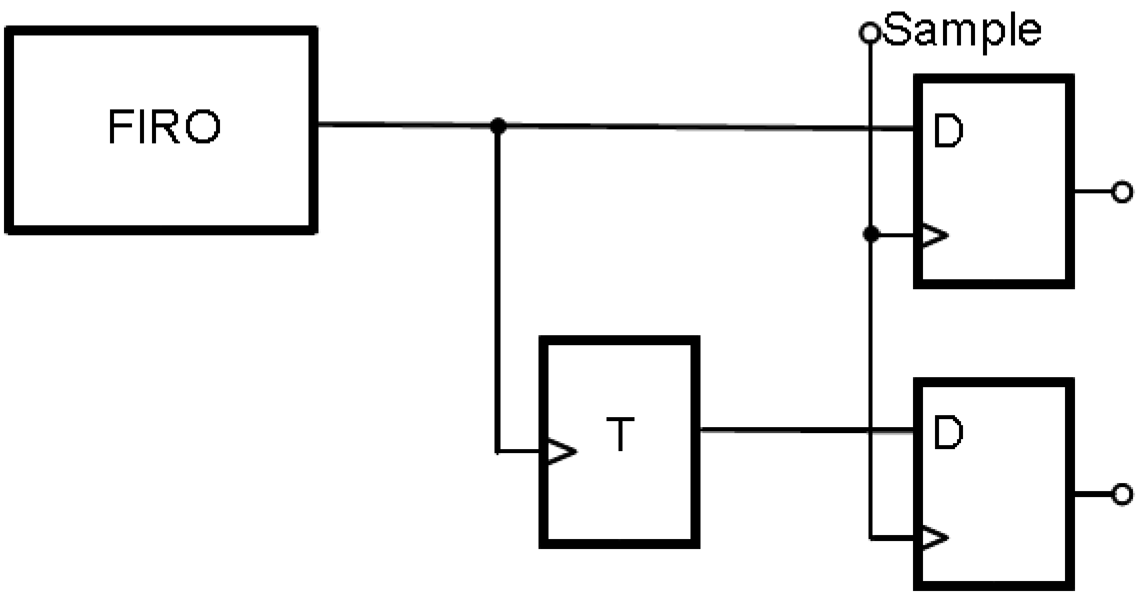
Broken by Markus Dichtl at CHES 2003

but

Werner Schindler showed at Cryptography and Coding 2003 that the design is secure if clocked 60000 times slower.

Prior art II

- Dichtl, Golic, *High-Speed True Random Number Generation with Logic Gates Only,* CHES 2007

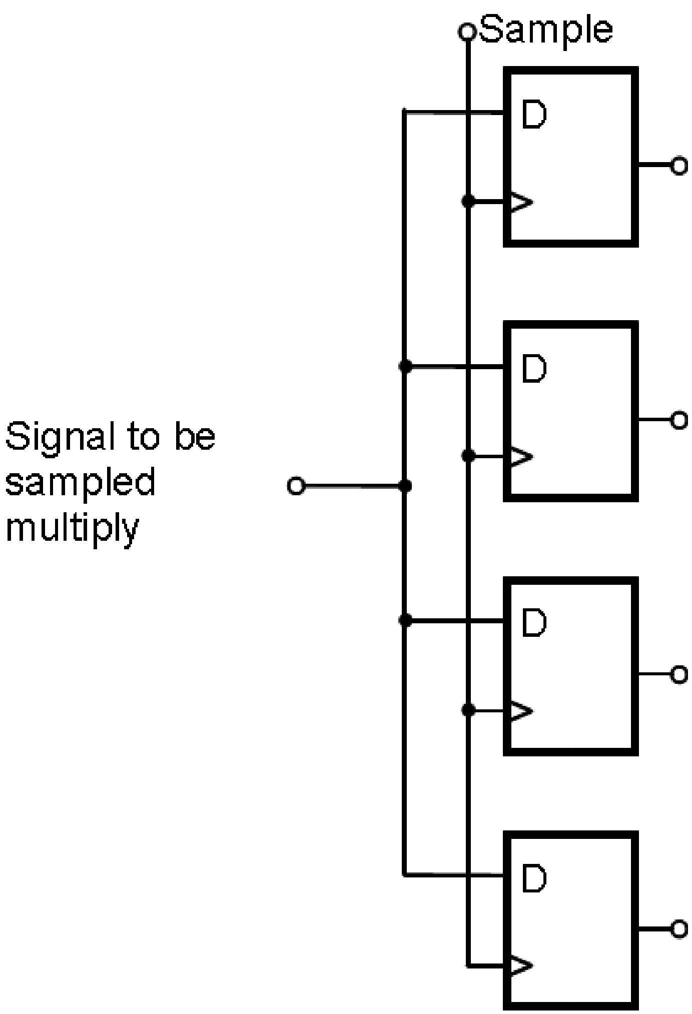


1.933 bits of entropy per pair of output bits

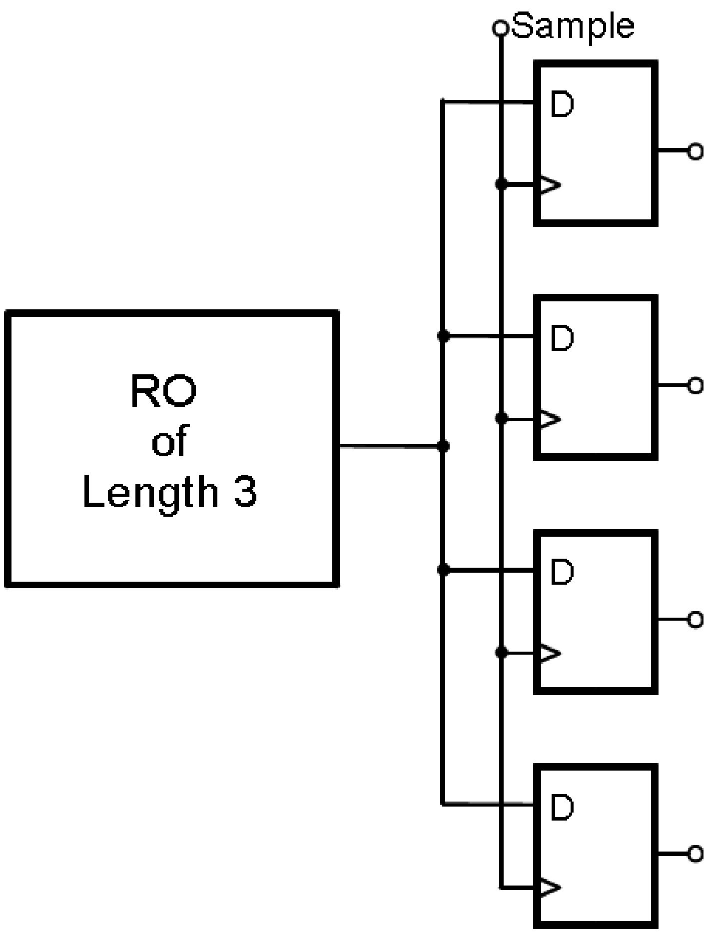
3 ways to be greedier

- Sample the same signal with multiple flip flops
- Use the same signal to cause toggling in several parallel T flip flops
- Sample several signals of a circuit with random behaviour

Sample the same signal with multiple flip flops I



Sample the same signal with multiple flip flops II



This seems absurd, but …

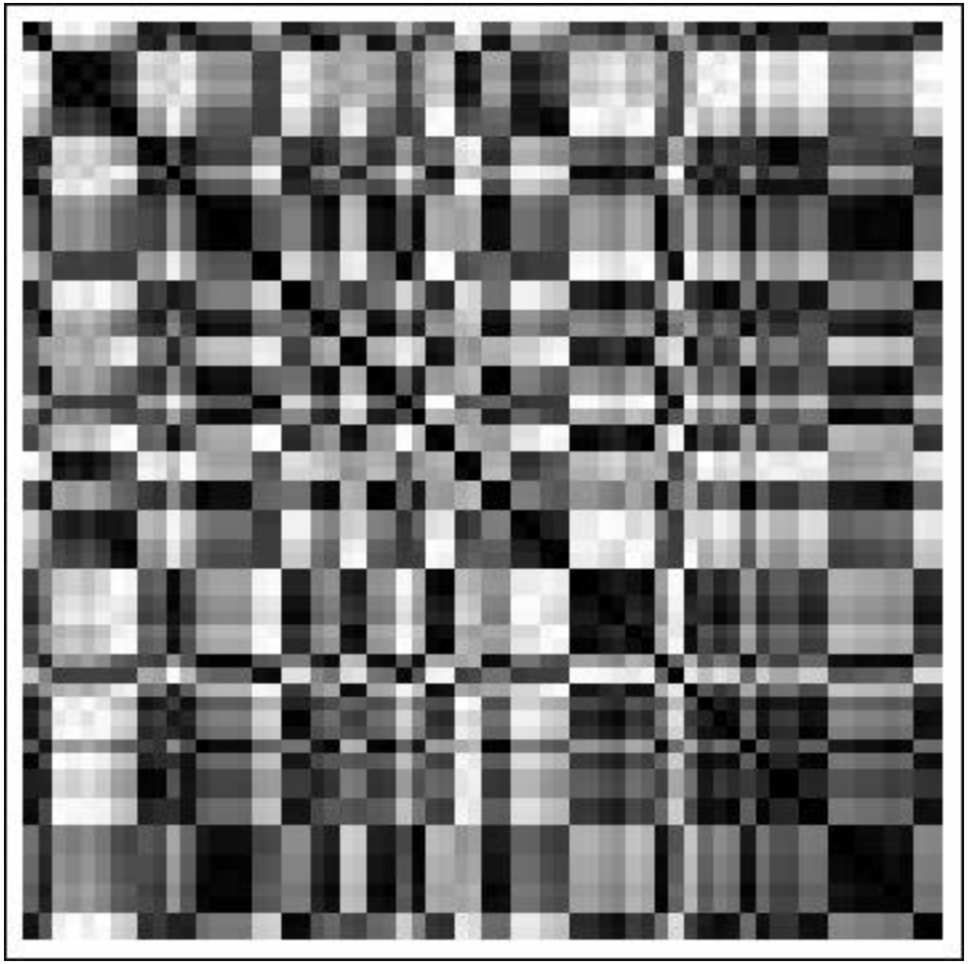
Sample the same signal with multiple flip flops III

```
11436 ffffffffffffffff
 7106 0000000000000000
 5841 fffffffeffffffff
 5469 0000000010000000
 4080 0000000800000000
 3795 0010000010000000
 3794 ffeffffcffffffff
 3596 0100000c00008004
 3590 0010000030000000
 3374 fffffff7ffffffff
 2933 ffcffffccc3fdff
 2440 ffeffffeffffffff
 2313 fcbf3bf3ff3f03f3
 2191 feffffff3ffff7ffb
 1921 f03c03203f3c0330
 1899 203000203f3c0300
 1884 0fc0fcdfc0c3fccf
 1586 0340c40c00c2fc0c
 1577 1fcffcdfc0c3fcff
```

Spartan 3, RO signal sampled with 64 flip flops 1 ms after restart

Entropy per sample:5.2 bits

Sample the same signal with multiple flip flops IV



The bits are not at all independent!

In the correaltion matrix, black stands for 1 and white for -0.81

Sample the same signal with multiple flip flops V

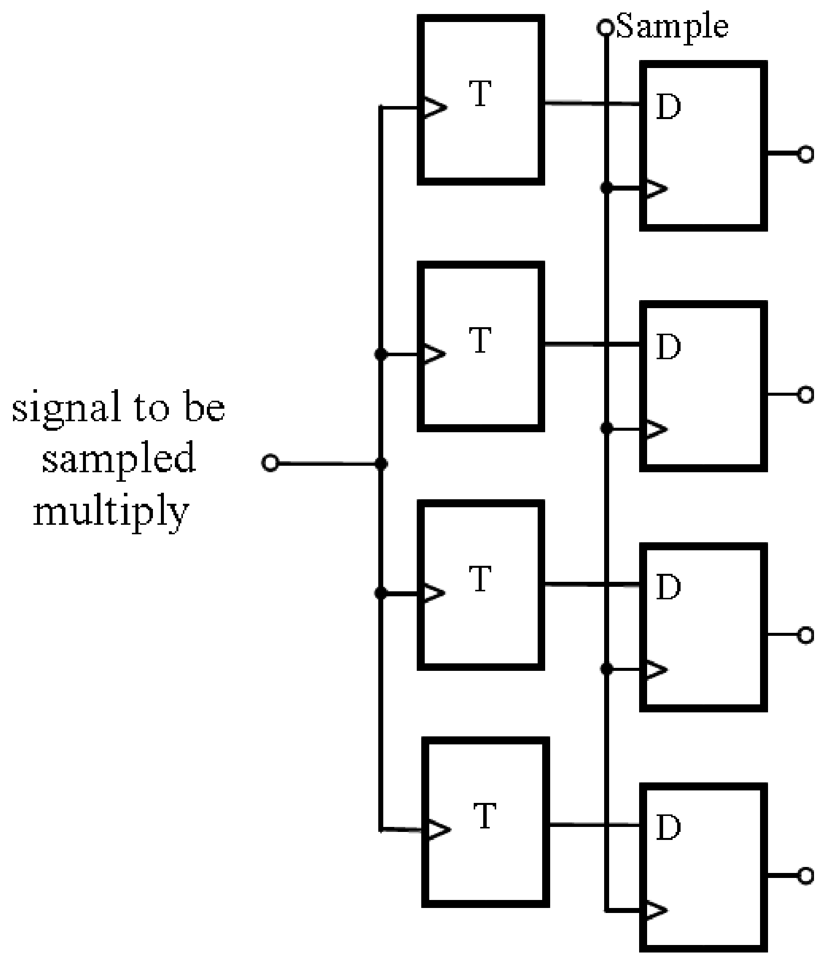
Two possible, non-exclusive explanations:

- Differences in routing make the different flip flops see the same signal in different phases
- Different behaviour is caused by the sampling flip flops, eg. by metastable oscillations, or by interpreting intermediate voltages differently

Sample the same signal with multiple flip flops VI

This works even better when sampling signals with chaotic random behaviour like signals from a FIRO or GARO

Using several toggle flip flops for the same signal I



Using several toggle flip flops for the same signal II

32 parallel toggle flip flops controlled by a signal from a FIRO of length 32

2702970 different 32-bit-patterns in 5814784 experiments from repeated restarts (sampling after 1 µs)

Problem: Impossible to determine entropy of 32 bit samples (too few samples)

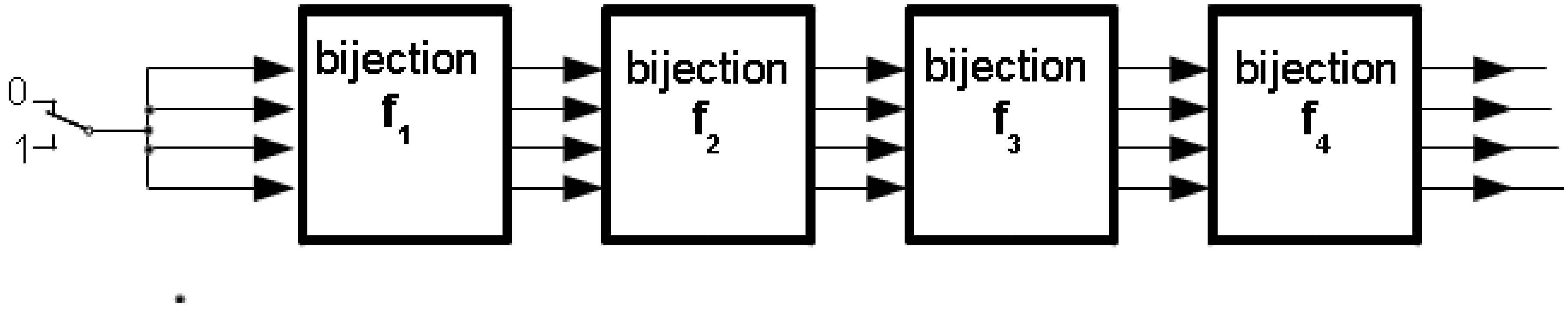
Instead, only the 16 last bits of the 32 were evaluated, resulting in an entropy of 10.977 bits. The minentropy was 4. 888 bits.

Using several toggle flip flops for the same signal III

This does not work for classical ROs, all toggle flip flops gave the same result!

Using several toggle flip flops for the same signal IV

But it works very well for a forthcoming class of TRNGs, feedback free multitrack TRNGs:



Using several toggle flip flops for the same signal V

Feedback free multitrack TRNG of length 100: one end signal feeds 32 parallel toggle flip flops with these most frequent output patterns:

```
2064137 FFFFFFFF
2005595 00000000
  50019 14500025
  45097 EBAFFFDA
  39828 FBFFFFFF
  36053 04000000
  17829 00200000
  13563 EBAFFDDA
```

Entropy of 2.3998 bits per 32-bit-sample

Using several toggle flip flops for the same signal VI

Toggle flip flops can be generalized to counters modulo m

Same scenario as above, but with 4 modulo 256 counters connected to one signal.

Most frequent patterns:

```
361907 04040404
226367 03030303
187768 05050505
129439 05040505
 87592 04030404
 77225 05060505
 69854 06050606
 62520 06060606
```

Entropy of 5.95 bits per 32-bit-sample

Sampling several signals of a circuit with random behaviour I

Same scenario as above, but with 16 modulo 4 counters connected to each of the 4 output signal.

Entropy of 12.66 bits per 32-bit-sample

Only one toggle flip flop at each of the 4 outputs results in an almost perfect entropy of 3.99982 per 4 bit sample

Sampling several signals of a circuit with random behaviour II

Sampling all 15 inverters of a classical RO of length 15:

001010101010101	80897
010010101010101	43615
010100101010101	87403
010101001010101	60396
010101010010101	58100
010101010100101	94117
010101010101001	72241
010101010101010	89319
010101010101011	49874
010101010101101	65542
010101010110101	120055
010101011010101	40179
010101101010101	82356
010110101010101	74662
011010101010101	90677

100101010101010	96420
101001010101010	85667
101010010101010	90990
101010100101010	42445
101010101001010	122643
101010101010010	77557
101010101010100	49930
101010101010101	83200
101010101010110	59139
101010101011010	82593
101010101101010	52999
101010110101010	50969
101011010101010	71200
101101010101010	40943
110101010101010	70427

Entropy per 15-bit-sample 4.84439 bits, compared to 4.90689 for a uniform distribution

Sampling several signals of a circuit with random behaviour III

SIEMENS

Sampling all 15 inverters of a classical RO of length 15:

00101010101010 1	80897
010010101010101	43615
010100101010101	87403
010101001010101	60396
010101010010101	58100
010101010100101	94117
010101010101001	72241
010101010101010	89319
010101010101011	49874
010101010101101	65542
010101010110101	120055
010101011010101	40179
010101101010101	82356
010110101010101	74662
011010101010101	90677

100101010101010	96420
101001010101010	85667
101010010101010	90990
101010100101010	42445
101010101001010	122643
101010101010010	77557
101010101010100	49930
101010101010101	83200
101010101010110	59139
101010101011010	82593
101010101101010	52999
101010110101010	50969
101011010101010	71200
101101010101010	40943
110101010101010	70427

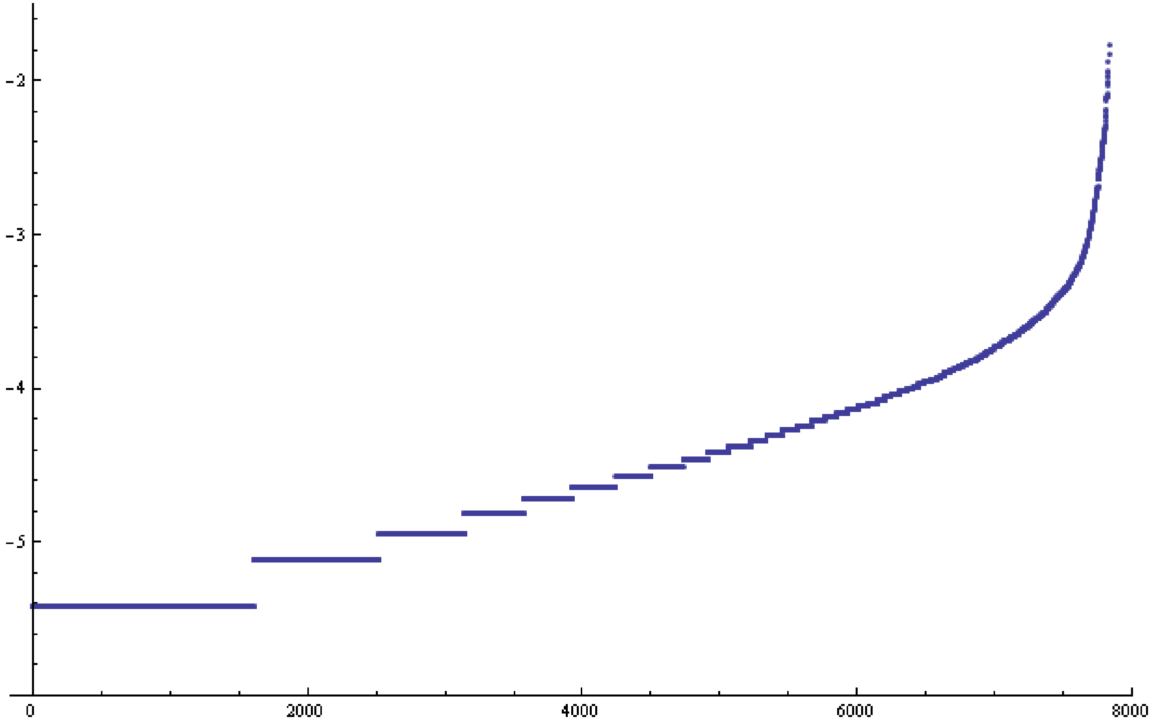
Entropy per 14-bit-sample 4.84439 bits, compared to 4.90689 for a uniform distribution

Sampling several signals of a circuit with random behaviour IV

Sampling all 14 inverters of a FIRO of length 14:

Sampling 262144 times results in 7832 different 14-bit-patterns .

Sorted decimal logarithms of the probability of the patterns:



Entropy per 14-bit-sample: 10.09 bits

Postprocessing of possibly dependent data I

The algorithm of Ari Juels, Markus Jakobsson, Elizabeth A. M. Shriver, Bruce Hillyer in the paper „How to turn loaded dice into fair coins“ in IEEE Transactions on Information Theory Vol. 46 No. 3 Pg. 911-921, 2000 seems to be ideally suited for postprocessing blocks of dependent data, as long as the dependencies are only within the blocks, and the blocks are independent and identically distributed.

Asymptotically, this algorithms converts all entropy from the source to minentropy.

It seems that repeated restarting the TRNGs from identical states could enforce independence and identical distribution.

At present, there is no evidence that this does not work.

However results from Richard Newell (Cryptarchi 2011), Markus Dichtl (Cryptarchi 2013) and Patrick Haddad, Yannick Teglia, Florent Bernard, and Viktor Fischer (DATE, 2014) show that classical ROs show dependencies over longer ranges of time.

It would be unwise to assume that other circuits with random behaviour do not have this deficiency.

Postprocessing of possibly dependent data II

So postprocessing of random data harvested in the ways described before should be based on cryptographic algorithms

Conclusion

Much more entropy than previously known can be harvested from well known circuits with random behaviour