KU LEUVEN

COSIC

A Multimode Ring Oscillator based TRNG for FPGAs

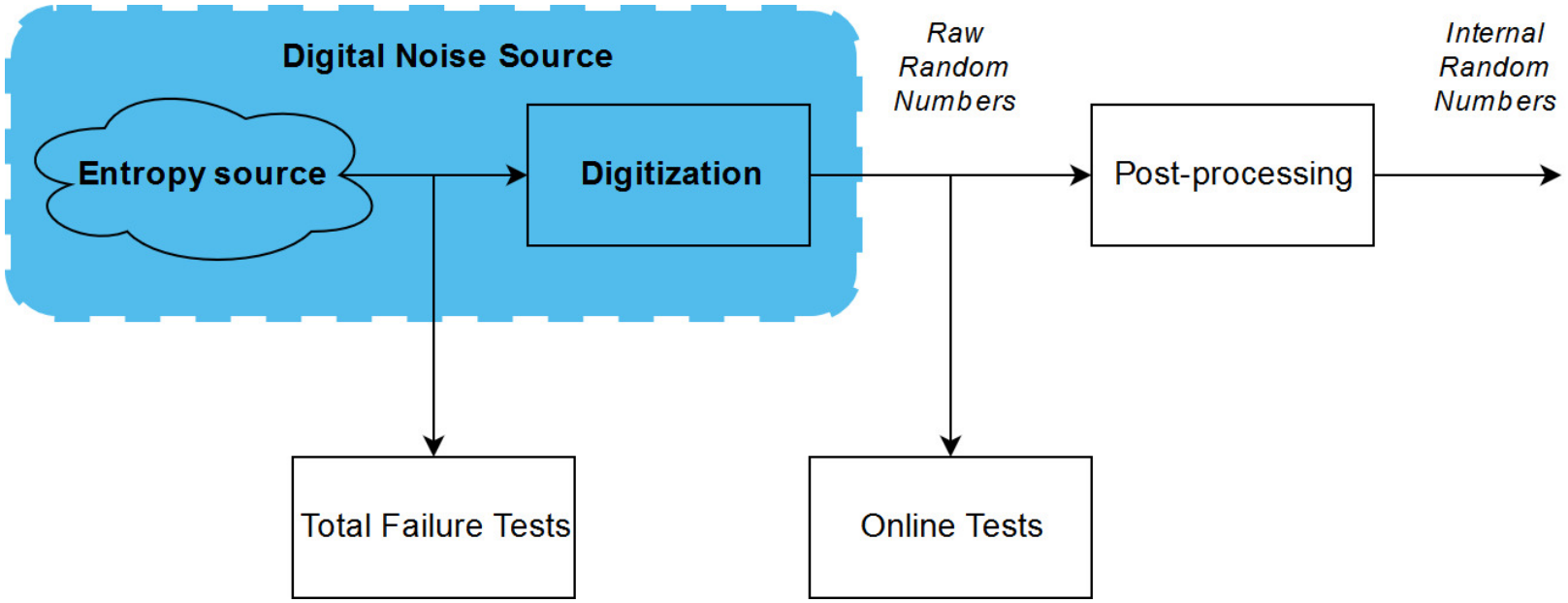
Miloš Grujić

imec-COSIC, KU Leuven

June 25th, 2019.

True Random Number Generators (TRNGs)

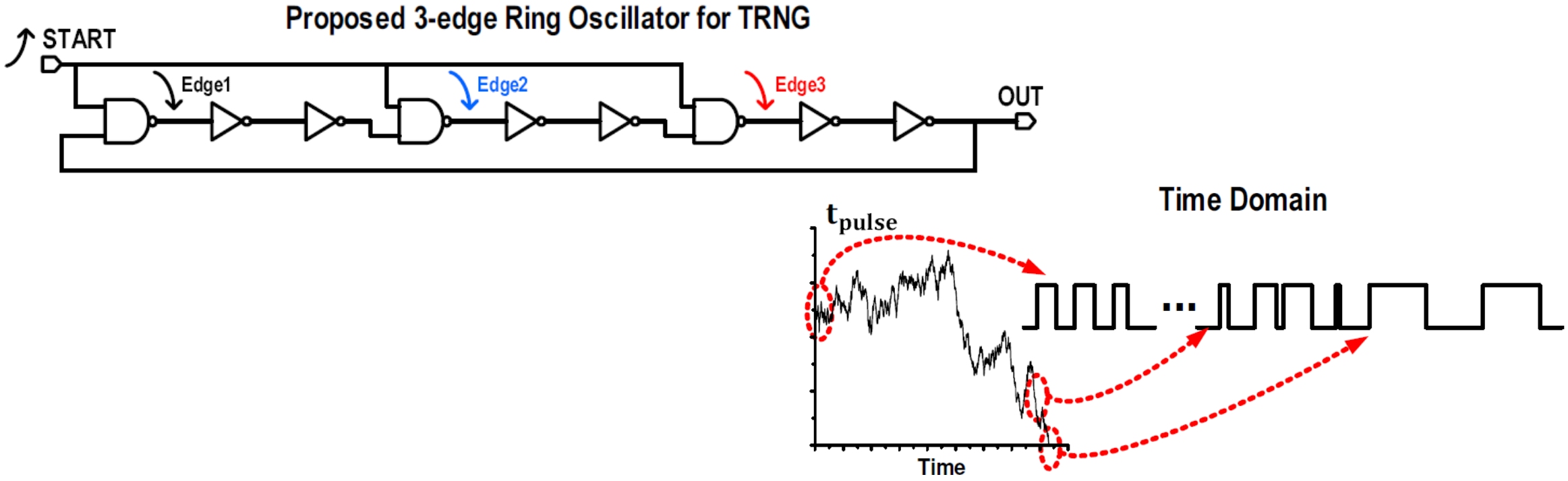
- Security of crypto applications → uniformity and unpredictability of random bits
- TRNGs - randomness from physical non-deterministic processes
- Required: stochastic model and lower bound of the (min)-entropy



Goals

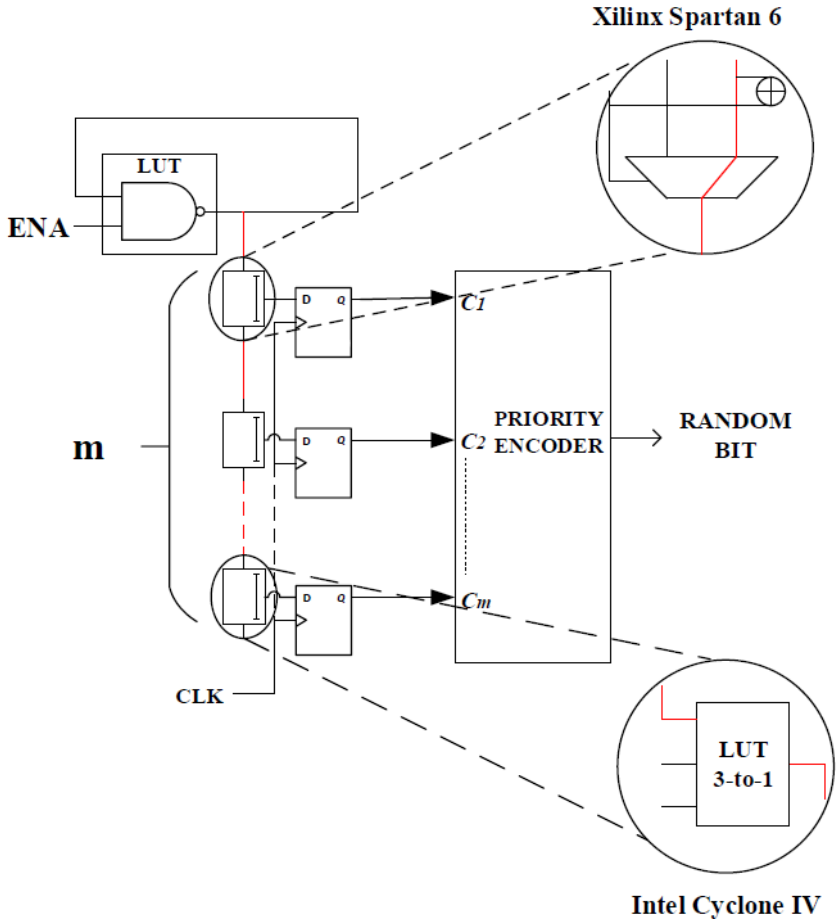
- Improve the TRNG metric $\frac{Entropy * Throughput}{D.E.* Area}$ by:
  - improving the efficiency of the digitization
  - boosting the amount of generated randomness in the entropy source
- AIS-31 compliant design: security analysis based on the stochastic model
- Minimize pseudo-randomness and the effect of the unwanted global noise sources

Multi-mode RO TRNG *[YFH+14]*

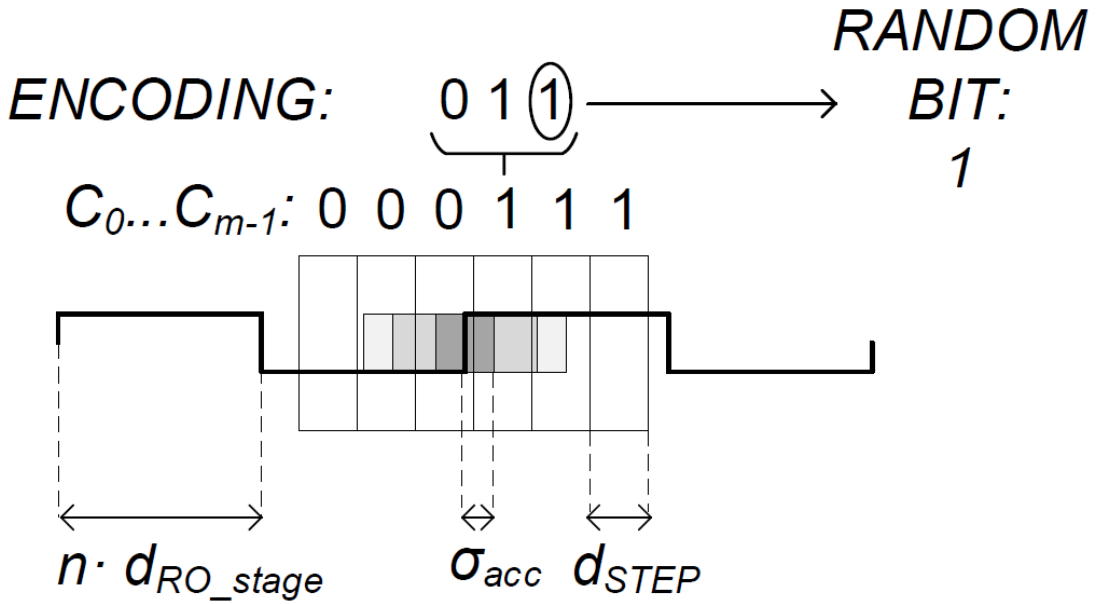


- Missing security evaluation and the entropy “measured” based on the generated random bits without IID claim

Delay-chain TRNG *[RYDV15]*

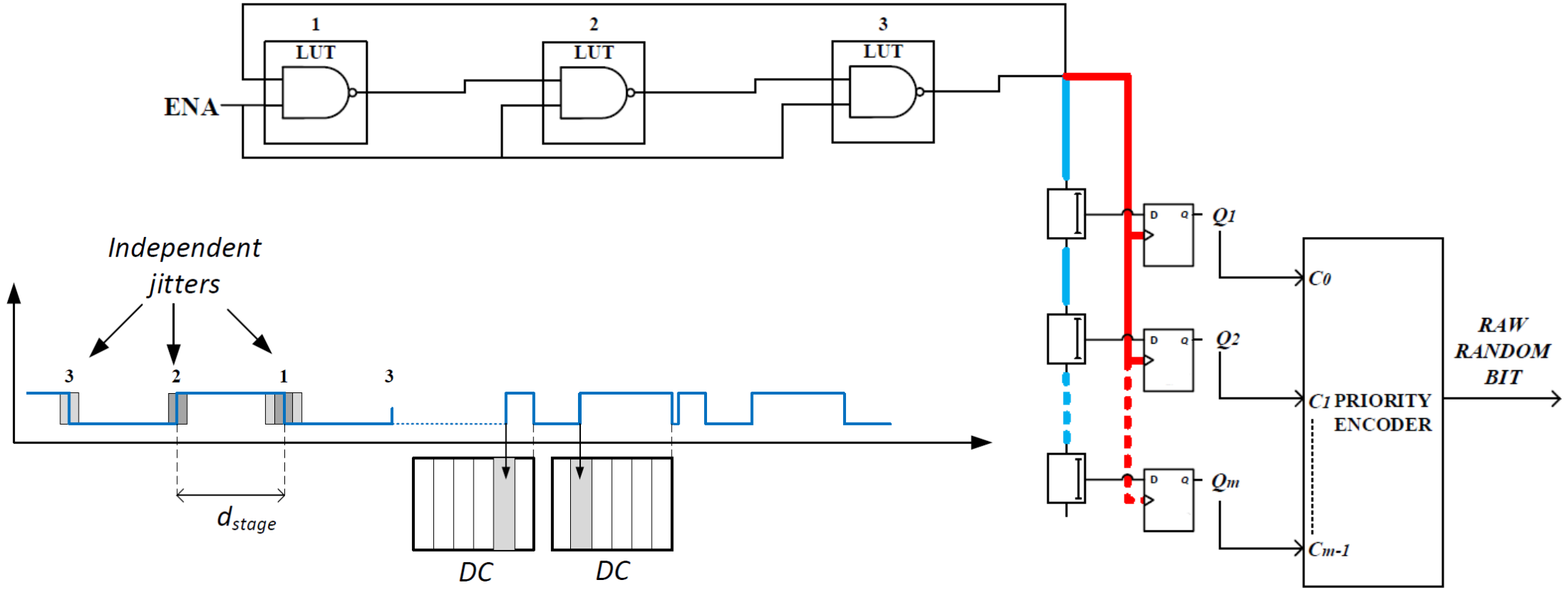


General architecture of the DC-TRNG

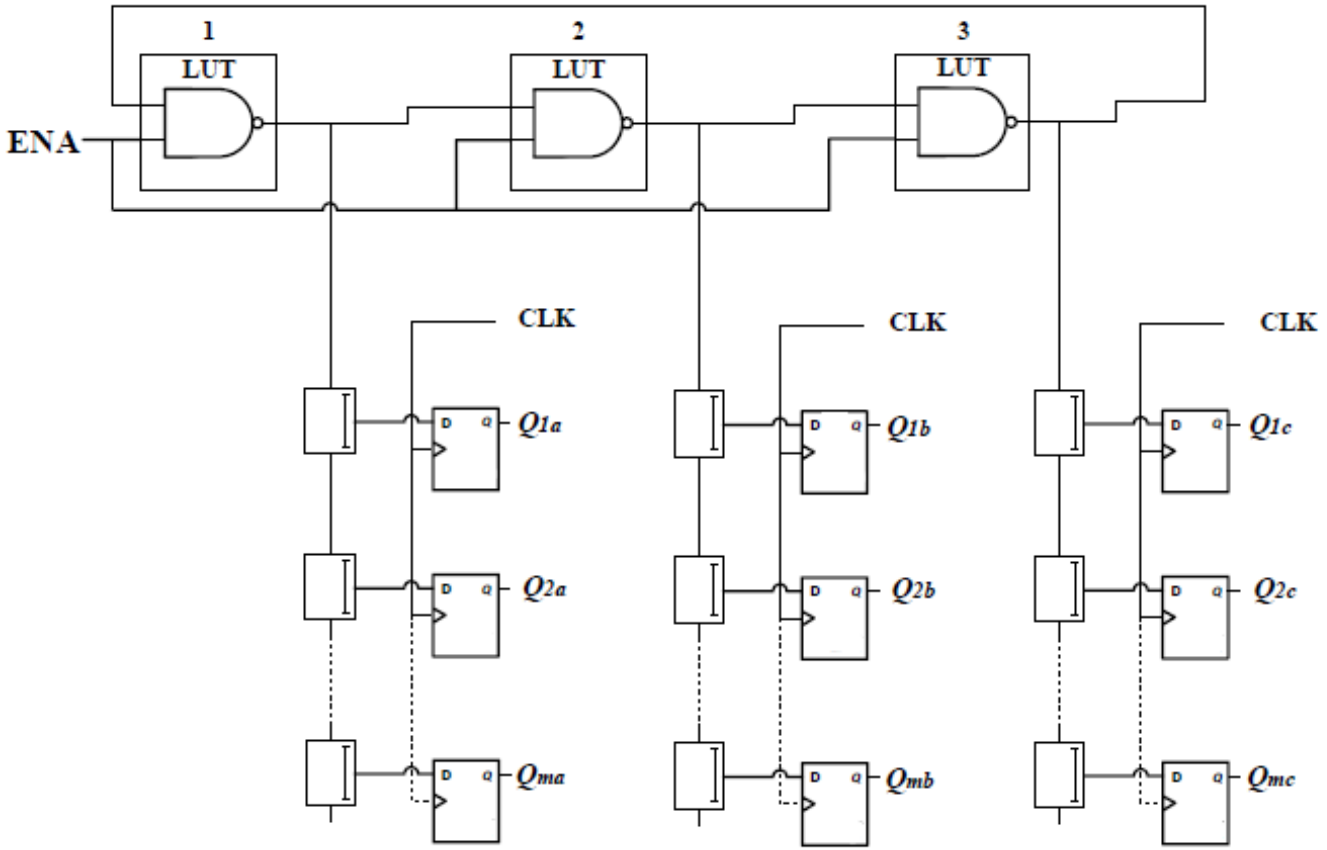


Entropy extracting process

New TRNG architecture

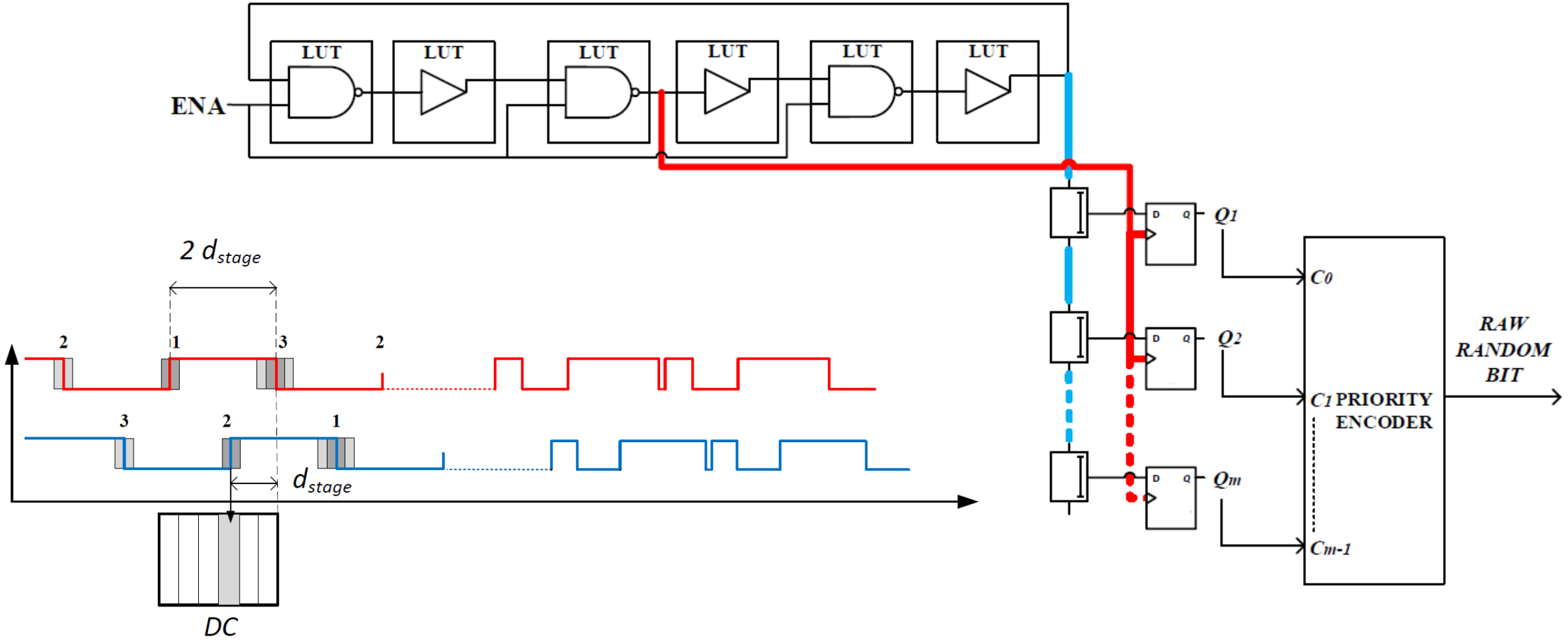


Experiment 1 – Interlocking of the edges

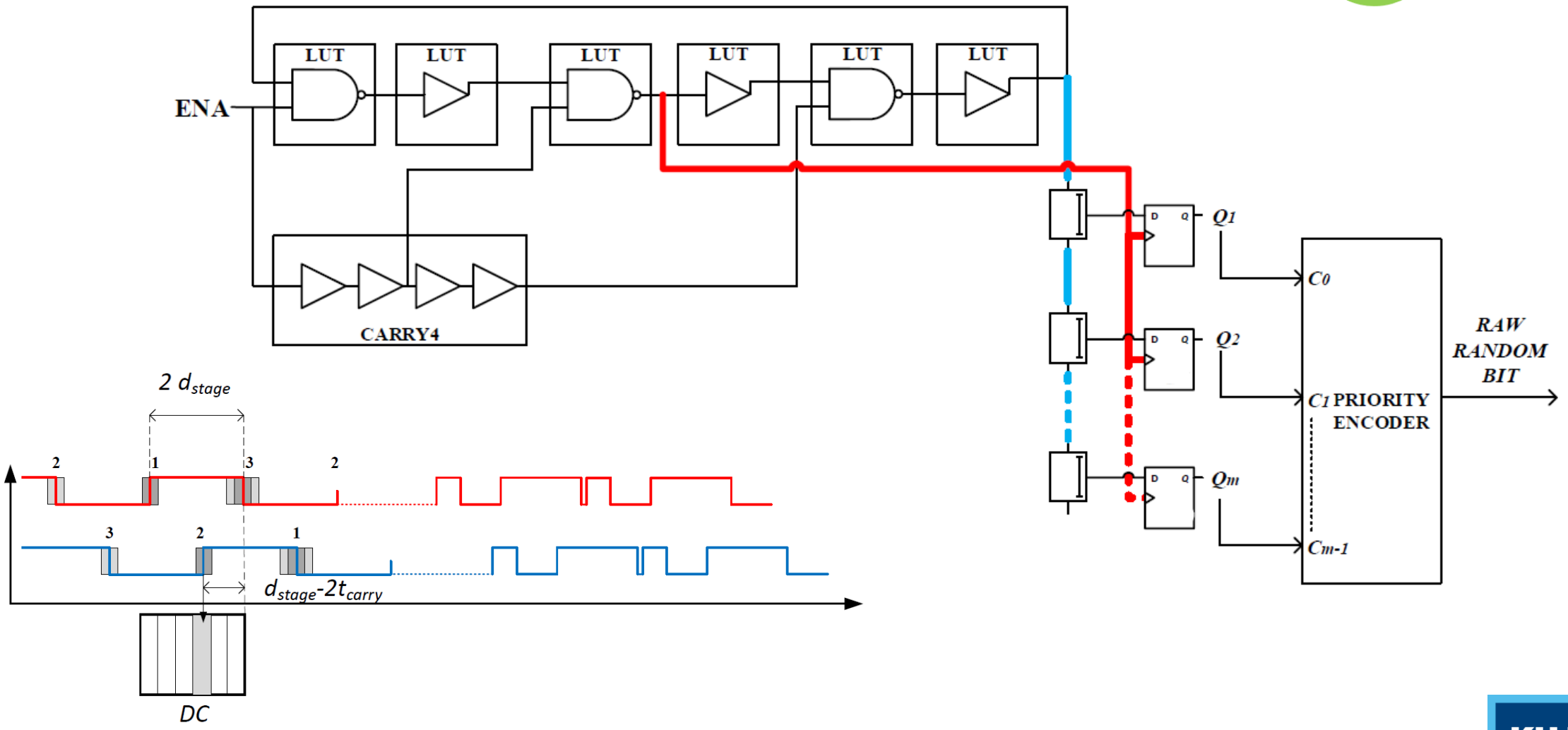


DC1: 00001111111100000000...
DC2: 00000001111111000000...
DC3: 00001111111100000000...

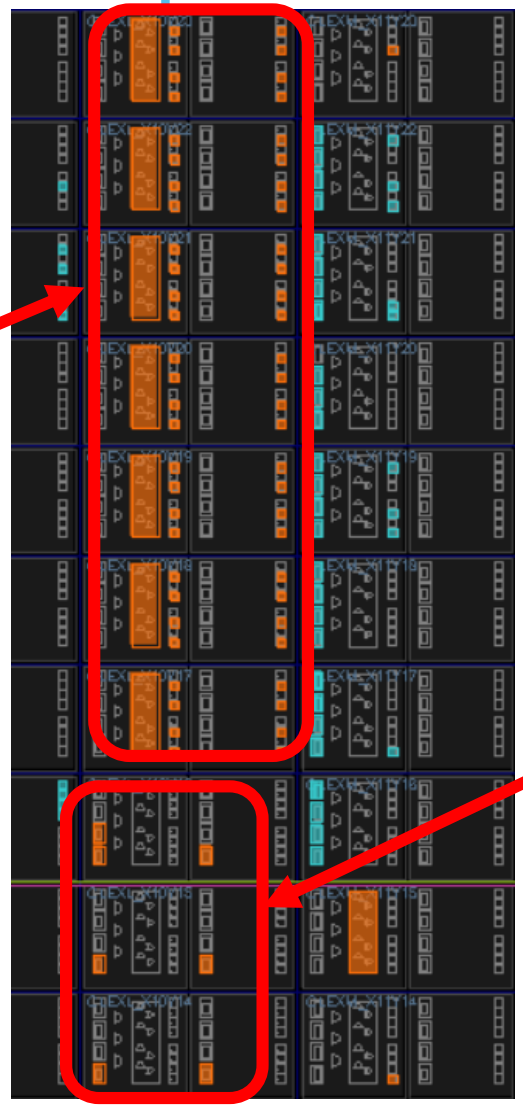
New TRNG architecture



New TRNG architecture

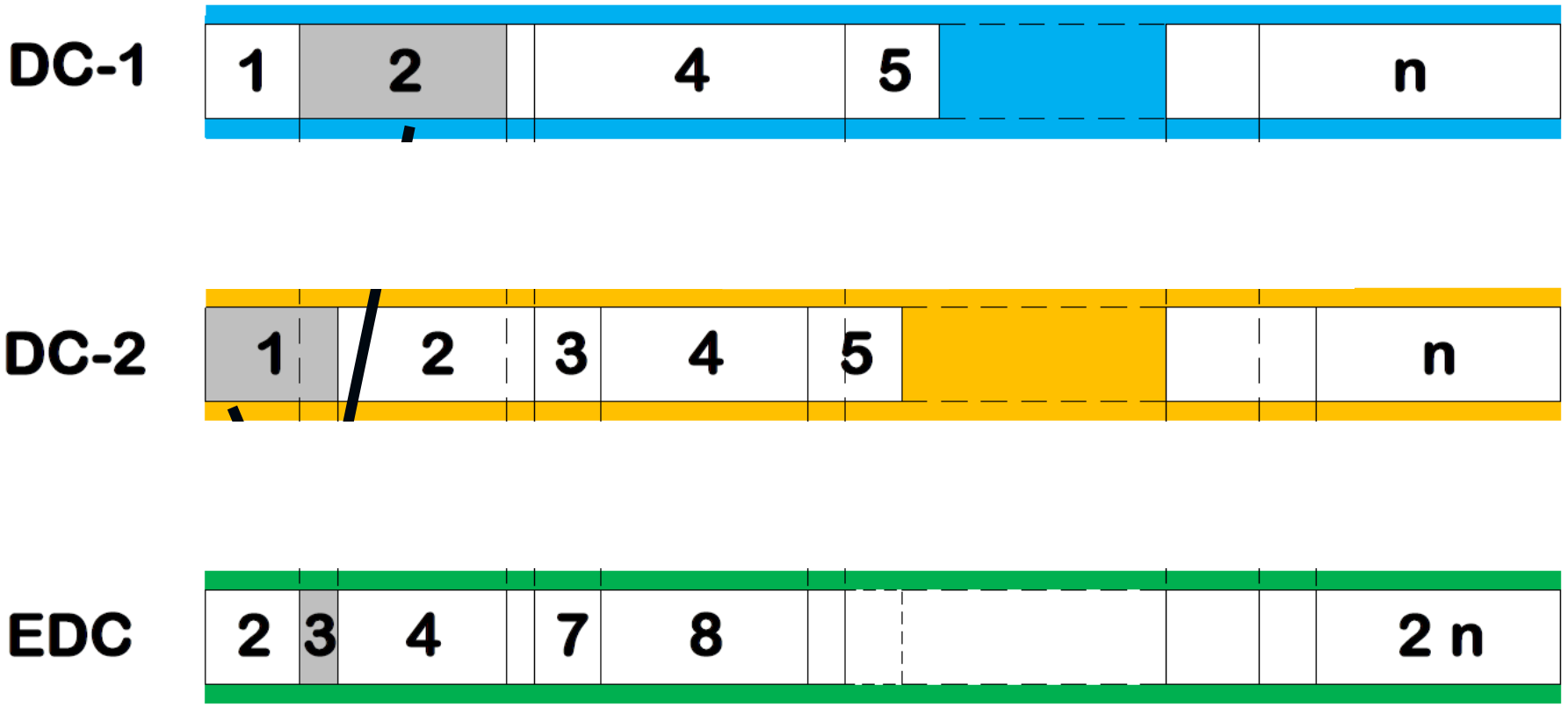


TRNG architecture – Implemented design

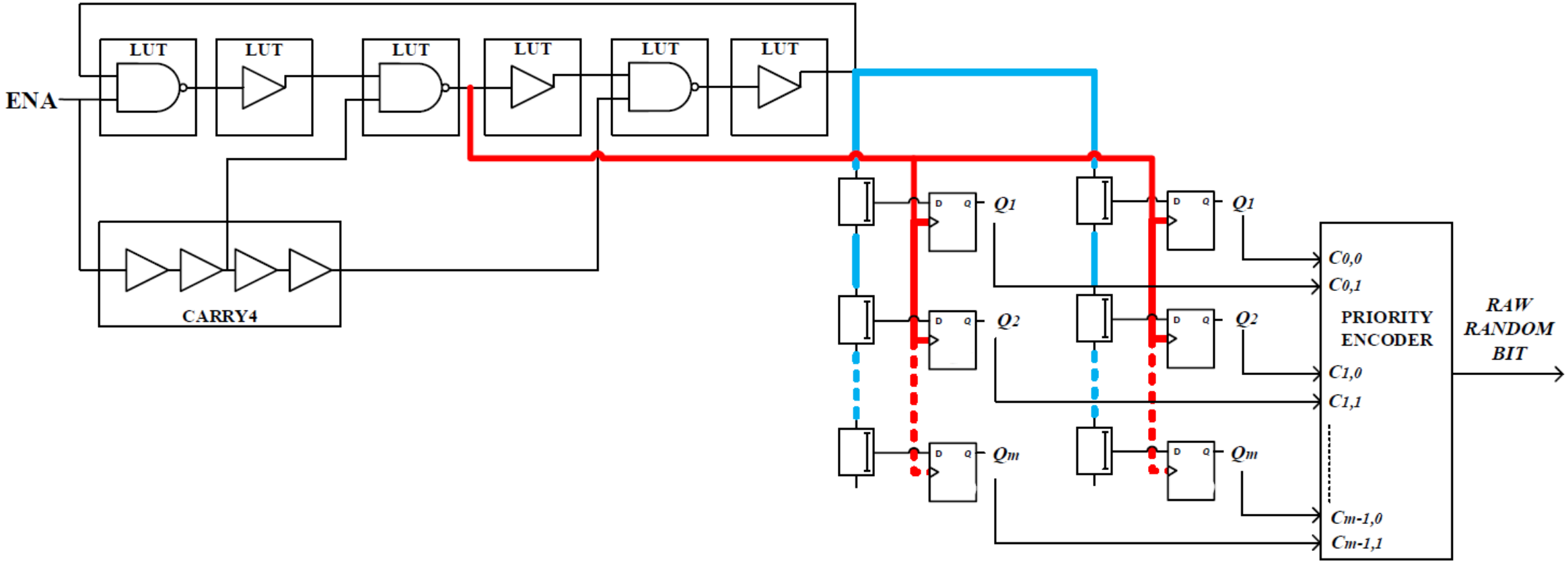


- *PlanAhead*: Spartan-6 FPGA

Double independent coding lines



New TRNG architecture



Stochastic model

- Baseline assumptions:
  - entropy extracted from the jittery pulse of the multi-mode RO
  - presence of independent Gaussian white noise
  - other noise sources present, but not exploited
  - due to differential design decreased influence of global noises
  - raw bits independent due to reset between successive generations

Stochastic model

- Notation: *w* – number of stages between edge-inserting stages, *n* – mode of the RO (number of inserted edges), *m* – current cycle (1 cycle contains *n* consecutive edges), $\frac{\sigma_m^2}{t_m}$ - Gaussian jitter strength
- Variance of the *virtual* pulse width coming from the Gaussian noise:

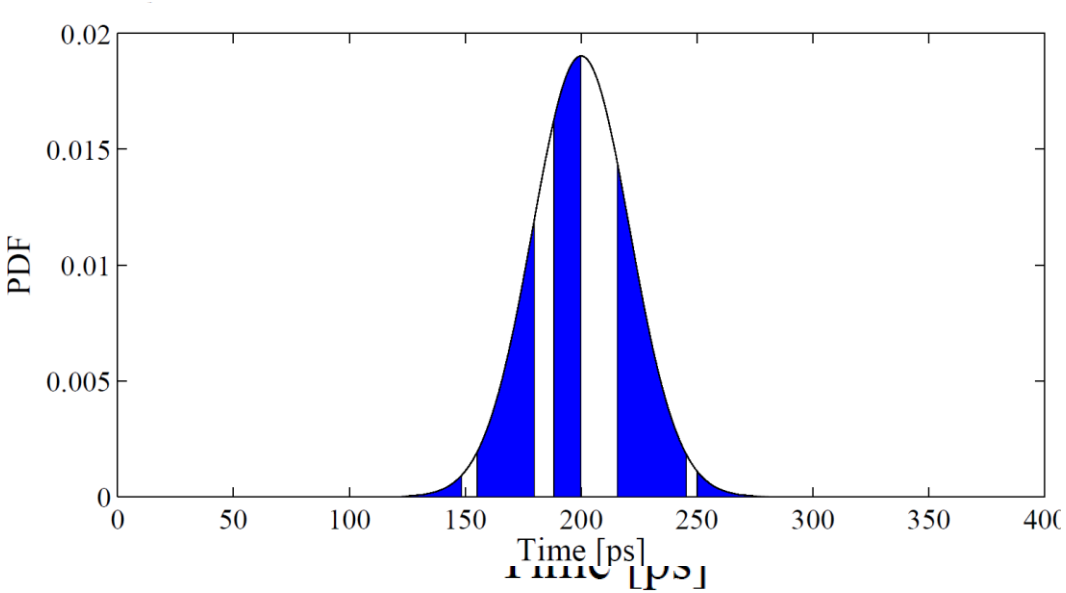
$$\sigma_{pulse,G}^2 = \frac{\sigma_m^2}{t_m} \cdot d_{stage} \cdot (2 \cdot w \cdot m \cdot n - n - 2 \cdot w \cdot (m \bmod 2))$$

- When *w* and *n* are determined by the circuit topology, we can calculate (even) *m*, such that:

$$m \geq \frac{\sigma_{pulse,G}^2 + \frac{\sigma_m^2}{t_m} \cdot d_{stage} \cdot n}{2 \cdot w \cdot \frac{\sigma_m^2}{t_m} \cdot d_{stage} \cdot n}$$

KU LEUVEN

Stochastic model



$$P_1\left(\mu_{pulse_{width}}, \sigma_{pulse,G}\right) = \sum_{i=-\infty}^{+\infty} \sum_{j=1}^{N/2} \left[\Phi\left(\frac{\sum_{k=1}^{2j} d_{carry,k} - \mu_{pulse_{width}} - i * d_{stage} * (w-1)}{\sigma_{pulse,G}} \right) \right.$$

$$H_1 = -P_1 * log_2(P_1) - P_0 * log_2(P_0)$$

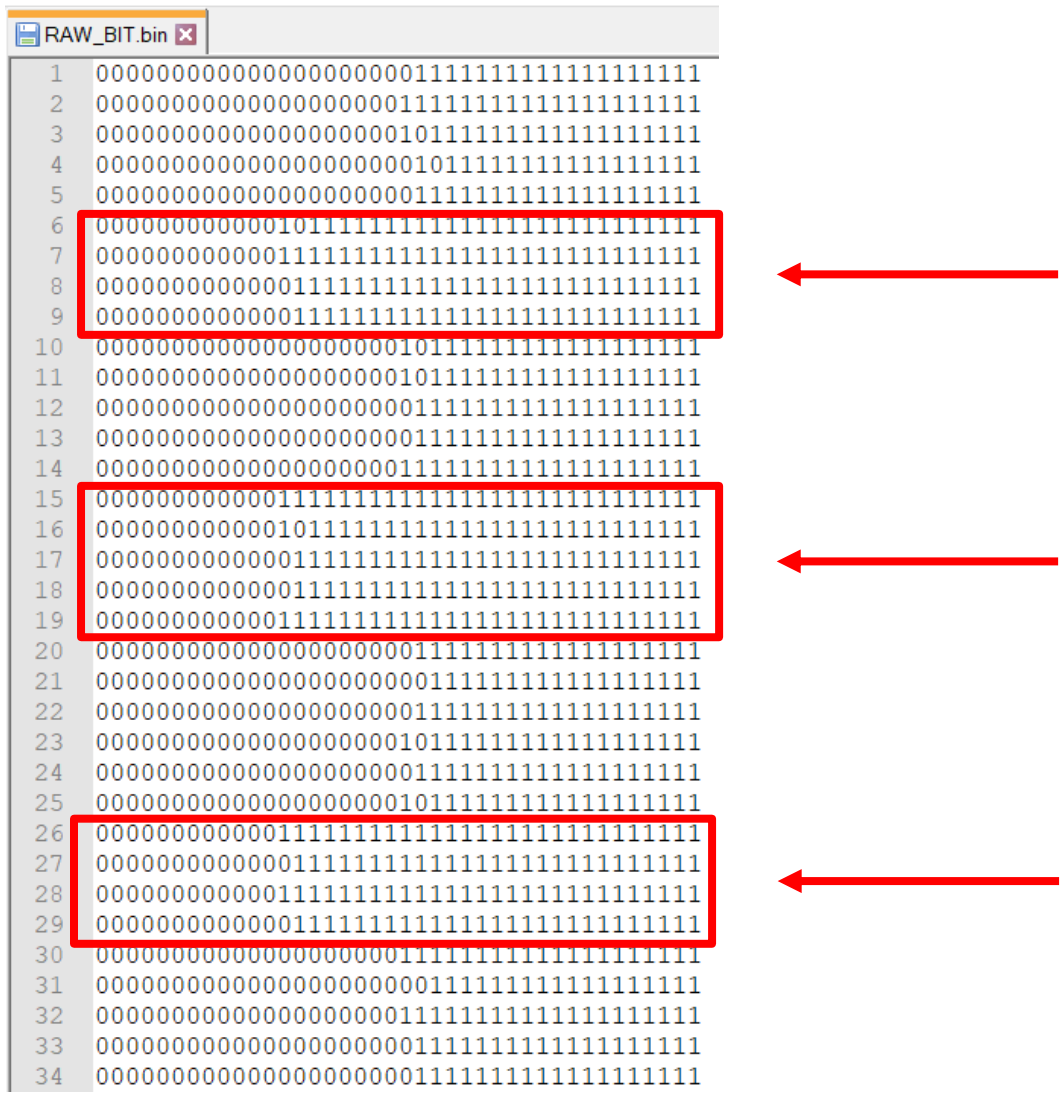
$\sigma_{pulse,G} = 10\ ps$ $\sigma_{pulse,G} = 20\ ps$ $\sigma_{pulse,G} = 40\ ps$

Stochastic model

- Platform parameters:
  - $d_{stage} \approx 675\,ps$
  - $\frac{\sigma_m^2}{t_m} = 2.7$ fs
  - $d_{carry,avg} \approx 20\,ps$– individually calculated for each delay block
- Design parameters for targeted $H_1 = 0.997$ bits:
  - *n = 3, w = 2* – circuit topology
  - *m = 18* => new raw random bit available after $73.57ns$
  - $\sigma_{pulse,G} = 19.7\,ps$

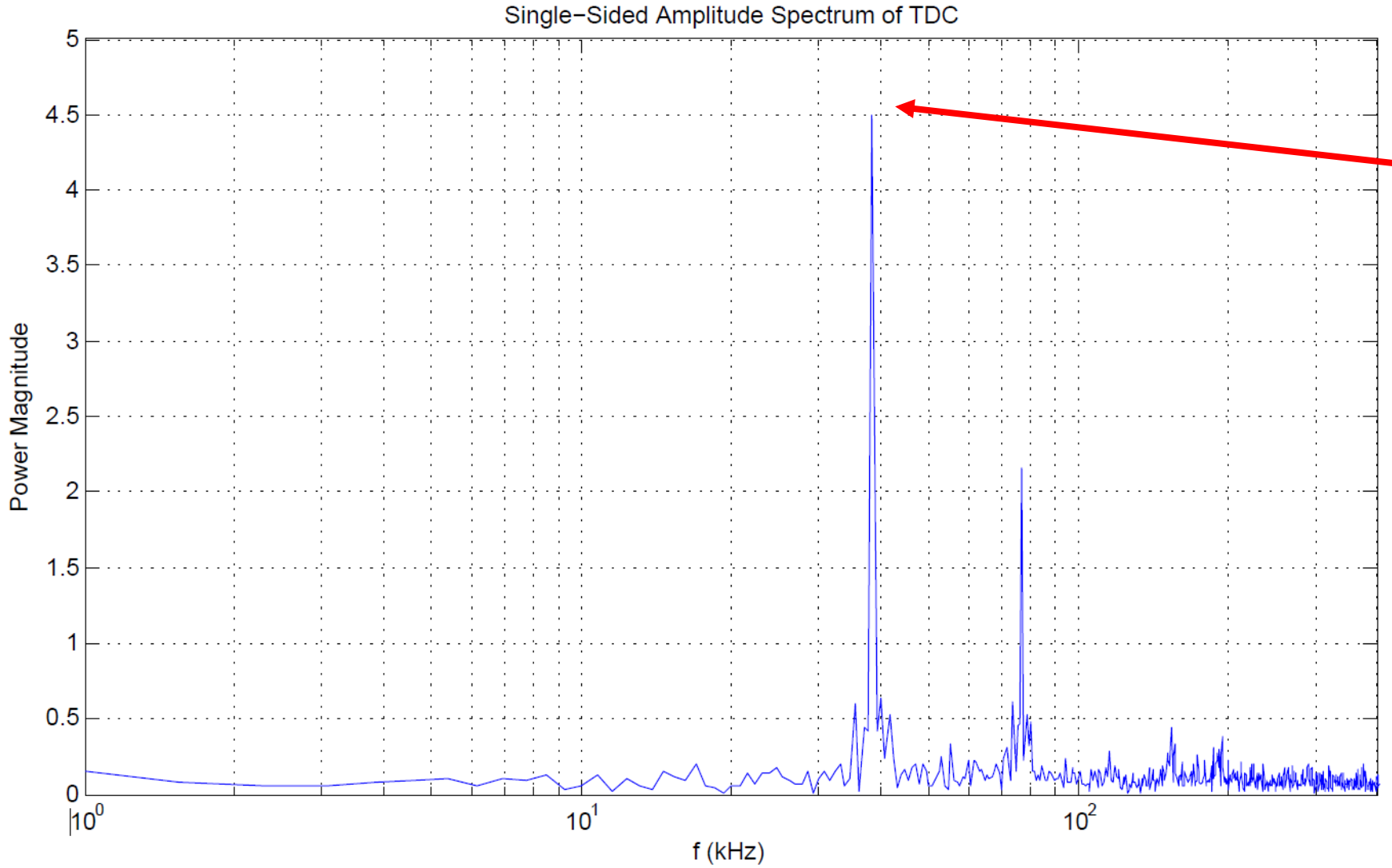
KU LEUVEN

Experiment 2



- Spartan-6 FPGA on LX9 Microboard

Experiment 2



~ 40kHz

Experiment 2 – Influence of the SMPS

The DCDC1 regulator generates 3.3 V. This voltage powers the Flash, Ethernet, PMODs, Vcco_0, Vcco_1, and Vcco_2. Estimated max current for the board circuits is 370 mA, which includes 50 mA for each PMOD.

The TPS65708 has built-in sequencing, resulting in a power-up sequence of 3.3 V → 1.8 V → 2.8 V → 1.2 V.

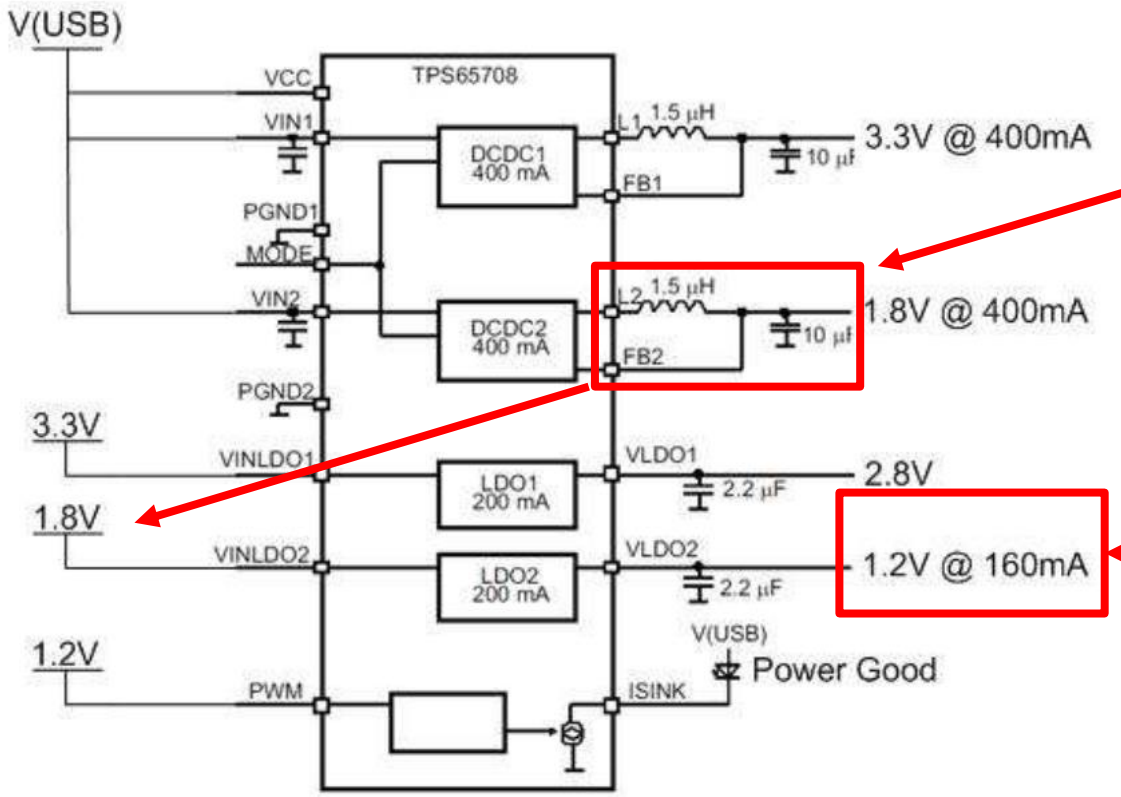


Figure 9 – TPS65708 Connections

Output filter with corner frequency: $\frac{1}{2\pi\sqrt{LC}}$ = 40kHz

Spartan 6 FPGA core voltage

- From: *Xilinx Spartan -6 FPGA LX9 MicroBoard User Guide*

A note on the arithmetic post-processing

- Applying linear code [16,8,5] as arithmetic post-processing:

$$L(X_1, X_2) = X_1 + (X_1 \ll 1) + (X_1 \ll 2) + (X_1 \ll 4) + X_2$$

$X_1, X_2, L(X_1, X_2)$ – 8-bit words

- In general: LC [*n, k, d*] post-processing reduces the bias to: $\varepsilon_{LC} = 2^{d-1}\varepsilon_{in}^{d}$
- Linear code [16,8,5] achieves smaller bias for the same throughput as one-stage XOR post-processing:

$$\varepsilon_{LC} = 2^{4}\varepsilon_{in}^{5} \qquad \varepsilon_{XOR} = 2\varepsilon_{in}^{2}$$

- Linear code post-processing – reduces both bias and small serial correlation [H. Zhou and J. Bruck, "Linear extractors for extracting randomness from noisy sources", *2011 IEEE International Symposium on Information Theory*]

KU LEUVEN

Implementation results

- **FPGA platform:** Xilinx Spartan 6
- **Area (without post-processing):** 25 LUTs, 80 FFs and 20 *carry4* elements
- **Throughput before post-processing:** 12.5 Mb/s
- **Throughput after post-processing:** 6.25 Mb/s
- **Estimated H_1:** 0.997 (raw r.n.)
- **Design effort:** manual placement, no manual routing

Asking over 8500 students to pick a random number from 1 to 10 [OC]

OC

Pick a random number from 1-10

[source: reddit.com]

Comparisons with *[PMB+16]*

TRNG type	FPGA	Area [LUT/FF/CARRY4]	Throughput [Mb/s]	Entropy per bit	Entropy throughput [Mb/s]	Design effort
ERO	Spartan 6	46/19	0.0042	0.999	0.004	1 (5/5)
COSO	Spartan 6	18/3	0.54	0.999	0.539	5 (5/1)
MURO	Spartan 6	521/131	2.57	0.999	2.567	1.25 (5/4)
PLL	Spartan 6	34/14	0.44	0.981	0.431	1.67 (5/3)
TERO	Spartan 6	39/12	0.625	0.999	0.624	5 (5/1)
STR	Spartan 6	346/256	154	0.998	153.9	2.5 (5/2)
This TRNG	Spartan 6	25/80/20	18	0.997	17.95	1.67 (5/3)

KU LEUVEN

References

[YFH+14] K. Yang, D. Fick, M. B. Henry, Y. Lee, D. Blaauw, D. Sylvester, “A 23Mb/s 23pJ/b Fully Synthesized True-Random-Number Generator in 28nm and 65nm CMOS,” *ISSCC 2014.*

[RYDV15] V. Rožić, B. Yang, W. Dehaene, and I. Verbauwhede, “Highly efficient entropy extraction for true random number generators on FPGAs,” *DAC 2015.*

[PMB+16] O. Petura, U. Mureddu, N. Bochard, V. Fischer and L. Bossuet, “A survey of AIS-20/31 compliant TRNG cores suitable for FPGA devices,” *FPL 2016.*

KU LEUVEN