HECTOR

D2.4

Robustness tests on TRNGs and PUFs

Project number:	644052
Project acronym:	HECTOR
Project title:	HECTOR: Hardware enabled crypto and randomness
Start date of the project:	1st March, 2015
Duration:	41 months
Programme:	H2020-ICT-2014-1

Deliverable type:	Report
Deliverable reference number:	ICT-644052 / D2.4 / V1.0
Work package contributing to the deliverable:	WP 2
Due date:	July 2018 – M41
Actual submission date:	31st July 2018

Responsible organisation:	BRT
Editor:	Gerard van Battum
Dissemination level:	Public
Revision:	V1.0

Abstract:	This report describes the joint effort of the HECTOR partners on evaluation of security characteristics of selected TRNG and PUF designs as researched during the HECTOR project.
Keywords:	Statistical tests; Shannon entropy; min-entropy; Side Channel Analysis; perturbation attacks

The project HECTOR has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 644052.

Editor

Gerard van Battum (BRT)

Contributors (ordered according to beneficiary numbers)

Sandra Lattacher, Martin Deutschmann (TEC)

Bernard Kasser, Michel Agoyan, Jean Nicolai, Maxime Madau (STR)

Ruggero Sussella (STI)

Josep Balasch, Milos Grujic (KUL)

Viktor Fischer, Oto Peťura (UJM)

Marek Laban (MIC)

Jan Luhman, Marnix Wakker, Ricard Malafre (BRT)

Executive Summary

This report summarizes the activities that were performed by partners UJM, MIC, TEC, STR and BRT within Work Package 2 relating to robustness verification of HECTOR TRNGs and PUFs.

These activities were done during development of HECTOR building blocks and as preparation for the development of the demonstrators. They allowed to verify if targets were achieved and provided feedbacks for further improvements. Additional robustness testing have also been performed on the demonstrators themselves, as reported in deliverable D4.3.

Our main targets of evaluations have been FPGA implementations of PLL and DC TRNGs. The tests that have been performed allowed to confirm the robustness of HECTOR developments under varying environmental conditions as well as their resistance against side channel analysis and perturbation attacks (HECTOR Objective O5). We have also been able to perform successful statistical and environmental testing on early silicon prototypes of an ST automotive ASIC embedding a specific version of a HECTOR PLL-TRNG. Unfortunately it was finally not possible to evaluate the main HECTOR ASIC because its manufacturing ended-up being delayed again (due external factors not under our control) such that we will only receive and be able to start evaluating those chips after the official completion of the project.

AIS20/31-compliance and certification feasibility or HECTOR TRNGs have been evaluated by Brightsight (HECTOR Objective O6). Additional evaluations of compliance to requirements from US NIST SP800-90B and the upcoming French DGA MI have also been performed.

Besides TRNGs we also had the ambitions to evaluate the feasibility of a rigorous, model-based, AIS20/31-like approach for the specification, design and evaluation of PUFs (HECTOR Objective O4), to propose corresponding PUF designs, and to evaluate their robustness. Earlier in the project we decided to focus on PUFs based on the TERO principle and our TERO-PUF implementations ended-up being sensitive to environmental conditions. While it has been a disappointment and project low-light not to have produced a very robust PUF design, our TERO modeling efforts have allowed to understand the root cause and in hindsight predict why a TERO-based PUF can't be robust. In our opinion this remains a very positive outcome and highlight confirming that this should indeed be the right way towards a more rigorous approach to PUF design, entropy evaluation and security evaluation in order to improve the confidence towards PUFs having a model-based security demonstration.

Besides the actual evaluation results, these activities generated a lot of security evaluation and certification know-how transfer between BRT and the other partners, helping them prepare the proper set of documentation and "developer evidences" packages required to leverage HECTOR's demonstrable security approach and enable faster security evaluations of future products integrating HECTOR security building blocks (HECTOR Objective O16).

Contents

List of Figures

List of Tables

Chapter 1 Introduction

Security building blocks developed by HECTOR in Work Package 2 are basic enablers (foundations) for building secure and trustworthy ICT products and systems. Being able to trust and demonstrate the soundness and robustness of their implementation is critical.

Outside of dedicated secure micro-controllers and “smartcard” industry, actors are typically less willing or able to deal with the costs and complexities associated with adoption of security certification. It is clear that the costs, required knowledge and time-lines for robustness and conformance testing can be prohibitive for small-scale enterprises, while this small-scale industry, including start-ups is often driving European innovation.

One of HECTOR’s main goals has therefore been to propose TRNG designs, stochastic models, embedded tests, security demonstrations, workflows etc. aiming at increasing security confidence and easing the security evaluation and certification process (HECTOR Objectives O1-O3). This should enable lowering the security certification costs and feasibility barriers, which in turn should increase end-user trust (HECTOR Objective O16).

This deliverable focuses on three important objectives for the project which were to verify the robustness of HECTOR TRNGs and PUFs against passive and active attacks (HECTOR Objectives O5) and validate AIS20/31 certification feasibility and process simplification achievements for at least one HECTOR TRNG design (HECTOR Objective O6). The other project objective covered in this deliverable was to research the feasibility of rigorous, AIS20-31-like model-based approach for the specification, design and evaluation of PUFs (HECTOR Objective O4).

Chapter 2 Statistical testing of TRNGs

2.1 Introduction

This chapter describes the statistical testing that has been done on two Hector TRNG implementations, which is part of the HECTOR Objective O5 on robustness testing (passive attacks). The targeted HECTOR TRNGs are the Delay Chain based TRNG (DC-TRNG) and the Phase Locked Loop based TRNG (PLL-TRNG). The DC-TRNG was selected to be implemented and tested in the Spartan 6 FPGA, the PLL-TRNG was selected to be implemented and tested in the Cyclone V FPGA and in an ASIC.

The main purpose of a TRNG is to provide output with a defined minimum entropy level. Besides performance requirements, statistical testing was used to verify if the outputs contain sufficient entropy. Statistical testing was done following the AIS20/31 and NIST SP800-90B test requirements. The results were compared with the minimum requirements for entropy generation, as well as for various other defects that could compromise applications that use the TRNG output streams.

In order to verify if the TRNGs are sufficiently robust for use in practical situations, the designs were subjected to varying environmental conditions during random number generation. It was observed if the quality of the results was affected by the environmental conditions. Two sets of statistical tests were used: the tests specified in the AIS 20/31 recommendations [1] and the tests given in the NIST SP 800-90B standard [2].

The AIS 20/31 statistical testing consists of two test procedures – test procedure A and test procedure B [1]. Test procedure A should be applied on internal random numbers (after the post-processing) and it consist of 6 tests (T0 – T5) – Disjointness test (T0), Monobit test (T1), Poker test (T2), Run test (T3), Long run test (T4) and Autocorrelation test (T5). Test procedure B should be applied on raw random numbers (digital noise) and it consists of three tests (T6 – T8) – Uniform distribution test (T6), Comparative test for multinomial distributions (T7) and Entropy test (T8). According to the AIS 20/31 standard [1], the test procedure A (B) is performed on one set of internal random numbers (raw random numbers). If all tests in procedure A (B) are passed, the test procedure A (B) is considered as passed. If only one test fails, the procedure A (B) is repeated on the second set of internal random numbers (raw random numbers). If more than one test fails, the complete procedure A (B) fails. If the second set of random data also fails at least one test, the procedure A (B) fails.

According to the last version of the standard, NIST SP800-90B testing consists of initial min-entropy estimation tests and restart tests. Min-entropy estimation tests are divided into two tracks, based on the nature of the entropy source – IID and non-IID track [2]. IID track is used for the entropy sources that produce independent and identically distributed (IID) random numbers, and the non-IID track is used for the entropy sources that output non-IID random numbers. After the initial min-entropy estimation ($H_{original}$) is obtained from the tests on the sequences of raw random numbers, the restart tests are run. These tests ensure that the random numbers generated after every restart are different.

Note that the NIST SP800-90B tests output min-entropy estimation, whereas AIS 20/31 tests provide Shannon entropy estimation.

2.2 Statistical testing of the DC-TRNG core implemented in Spartan 6 FPGA

The DC-TRNG core was implemented in the Xilinx Spartan 6 FPGA on HECTOR daughter board DBS6v11. The daughter board was placed in a temperature controlled chamber and connected to the motherboard with a HDMI cable. The motherboard was placed outside the chamber and connected to the host PC via USB interface. Generated data were stored in real time in the local data memory and then transferred to the PC via USB interface. This strategy guaranteed a non-interrupted random data bit streams suitable for a statistical evaluation.

2.2.1 AIS 20/31 statistical testing of the DC-TRNG

2.2.2 Test information

Table 1 shows information about the setup used for AIS 20/31 verification testing of the DC-TRNG.

DC-TRNG design version	B5.07
HECTOR daughterboard version	DBS6v11
HECTOR motherboard version	MBSF2v12
Test environment	Temperature chamber Espec SH-662
Software	AIS 20/31 test suite; Matlab
Evaluator	KU Leuven

Table 1: DC-TRNG setup for AIS 20/31 tests

The daughterboard with the DC-TRNG design was placed inside the temperature chamber and environmental temperature was varied from -40ºC to 80ºC in steps of 10ºC. For every temperature step four sets of 1 MB of random numbers were acquired – two sets of raw random numbers (Raw DC-TRNG output) and two sets of internal random numbers (PP DC-TRNG output). The daughterboard was connected via HDMI cable to the motherboard outside the chamber to ensure that the data collection and communication to the PC were not influenced by extreme temperature changes.

2.2.3 Test results

Table 2 shows results of the AIS 20/31 statistical testing on the DC-TRNG random numbers for 13 different temperatures. All tests from the test procedure A (T0-T5) were executed on the internal random numbers (after the post-processing), and all tests from the test procedure B (T6-T8) were executed on the raw random numbers. Additionally, in order to estimate (Shannon) entropy, test T8 (entropy test) was also performed on internal random numbers. Note that tests T6 to T8 allow small one-bit dependencies and biases, since they should be applied on statistically imperfect raw random numbers, before any post-processing.

It can be observed in the table that the internal random numbers pass all tests T0 to T5 for all temperatures, with estimated entropy rate per sample (= one byte) always higher than 7.99 (this represents Shannon entropy rate per bit higher than 0.99875, i.e. higher than required

0,997). Due to rounding errors and proximity of the real entropy to the maximum value - 8, the reported estimated entropy of internal random numbers is for some temperatures higher than maximal. The raw random numbers pass all tests from test procedure B (T6 to T8) for temperatures from -40 °C to -30 °C and -10 °C to 80 °C, again with very high estimated entropy rate of more than 7.99. However, for -20 °C the test 8 fails, reporting entropy of 7.17, lower than the required 7.976. The explanation for this behaviour is twofold. First, the T8 test assumes independent and identical data, while DC-TRNG raw random numbers are not identically distributed. Second, due to the nature of randomness itself, it can happen that the test fails two times consecutively, causing the whole test procedure to fail.

Temperature	Raw DC-TRNG output		PP DC-TRNG output	
	Estimated entropy	T6-T8	Estimated entropy	T0-T5
- 40 °C	7.992512241278886	Pass all	7.998246597780946	Pass all
- 30 °C	7.992771108310479	Pass all	8.000728198537800	Pass all
- 20 °C	7.175907934770539	Pass T6-T7	7.999107345007900	Pass all
- 10 °C	7.998575287487696	Pass all	7.999178622264193	Pass all
0 °C	7.994629464201603	Pass all	8.002528965294990	Pass all
10 °C	7.994609506570334	Pass all	7.999139503670157	Pass all
20 °C	7.991846687148844	Pass all	8.000564741293090	Pass all
30 °C	7.998140929629522	Pass all	7.995787120644388	Pass all
40 °C	7.995453128634804	Pass all	7.996890420589518	Pass all
50 °C	7.995192433902762	Pass all	8.000779325892964	Pass all
60 °C	7.994463782990301	Pass all	8.000192863696821	Pass all
70 °C	7.999084001984293	Pass all	8.000080513335247	Pass all
80 °C	7.996016695279266	Pass all	7.999323933214343	Pass all

Table 2: AIS 20/31 test results at different temperatures for the DC-TRNG

2.2.4 Conclusions

According to the HECTOR requirements, the quality of the random numbers must be guaranteed within the range of 0 °C to 85 °C. From the AIS 20/31 test results presented in Table 2, all tests in both test procedures A and B are passed for the specified temperature range. Furthermore, the DC-TRNG post-processed random numbers have high entropy even for very low temperatures.

2.2.5 NIST SP800-90B statistical testing of the DC-TRNG

2.2.6 Test information

Table 3 shows information about the setup used for NIST SP800-90 verification testing of the DC-TRNG.

DC-TRNG design version	B5.07
HECTOR daughterboard version	DBS6v11
HECTOR motherboard version	MBSF2v12
Test environment	Temperature chamber Espec SH-662
Software	NIST SP800-90B Python package (available on: https://github.com/usnistgov/SP800-90B_EntropyAssessment); Matlab
Evaluator	KU Leuven

Table 3: DC-TRNG setup for NIST SP 800-90B tests

The daughterboard with the DC-TRNG design was placed inside the temperature chamber and environmental temperature was varied from -40ºC to 80ºC in steps of 10ºC. For every temperature step two sets of 1,000 x 1,000 random numbers (bits) were acquired – one set of raw random numbers (Raw DC-TRNG output) and one set of internal random numbers (PP DC-TRNG output). The daughterboard was connected via HDMI to the motherboard outside the chamber.

2.2.7 Test results

Since the DC-TRNG contains the XOR post-processing function, the same testing procedure was applied to the post-processed random numbers.

Table 4 shows the results of the NIST SP800-90B statistical testing on the DC-TRNG random numbers for 13 different temperatures. Due to the nature of the DC-TRNG entropy source, we have opted for the non-IID track of the tests. It can be observed that in the whole temperature range -40 °C to 80 °C the reported min-entropy rate is above 0.86 per bit. All restart tests always passed, meaning that no correlation was detected between random numbers produced right after the restart of the generator. The tests that reported the lowest estimated min-entropy values are the Collision Estimate and the t-Tuple Estimate. This can be explained by the slight bias that exists in the raw random numbers produced by the DC-TRNG.

Raw DC-TRNG output PP DC-TRNG output

Temperature	Initial min-entropy estimate	Restart Test	Initial min-entropy estimate	Restart Test
- 40 °C	0.8836560387	Pass	0.9226013704	Pass
- 30 °C	0.9019679170	Pass	0.9072428590	Pass
- 20 °C	0.9087018441	Pass	0.9019679170	Pass
- 10 °C	0.8993376609	Pass	0.9072428590	Pass
0 °C	0.9177262134	Pass	0.8914755432	Pass
10 °C	0.9177262134	Pass	0.8732955271	Pass
20 °C	0.9226013704	Pass	0.9072428590	Pass
30 °C	0.9251414944	Pass	0.9098875803	Pass
40 °C	0.9339108095	Pass	0.9177262134	Pass
50 °C	0.9153840443	Pass	0.8681430393	Pass
60 °C	0.9433459105	Pass	0.9226013704	Pass
70 °C	0.8967121915	Pass	0.9178509586	Pass
80 °C	0.8810589272	Pass	0.9125371587	Pass

Table 4: NIST SP800-90B test results on DC-TRNG

2.2.8 Conclusions

NIST SP800-90B does not specify the minimum value of the estimated min-entropy needed for the testing procedure to ‘succeed’ if all restart tests are passed. Rather, it provides a rough estimation of how much min-entropy the random number generator can provide. From the test results in Table 4, we can conclude that the DC-TRNG outputs high quality random numbers, in the guaranteed range of 0 °C to 85 °C, that pass all restart tests with the minimum estimated min-entropy of 0.8681430393 bits for 50 °C. Additionally, the DC-TRNG raw and post-processed random numbers have high min-entropy even for very low temperatures.

2.3 Statistical testing of the PLL-TRNG core implemented in Cyclone V FPGA

The PLL-TRNG core was implemented in the Intel Cyclone V FPGA on HECTOR daughter board DBCVv11. The daughter board was placed in a temperature controlled chamber and connected to the motherboard with a HDMI cable. The motherboard was placed outside the chamber and connected to the host PC via USB interface. Generated data were stored in real time in the local data memory and then transferred to the PC via USB interface. This strategy guaranteed a non-interrupted random data bit streams suitable for statistical testing.

The PLL TRNG core had the following configuration:

The frequency of the input clock signal generated in a low jitter quartz oscillator was 125 MHz, multiplication and division factors of PLL0 were KM0 = 37, KD = 24 and those of PLL1 were KM1 = 19, KD = 5. Consequently, the frequency of the reference clock signal was clk0 = 192.7 MHz and that of the sampled signal was 475 MHz. Multiplication and division factor of the whole PLL TRNG were thus KM = 456 and KD = 185. The bit rate of the generator was 1.04 Mbits/s. Based on these parameters the thresholds of parameters P1 and P2 were computed from the stochastic model. To attain the Shannon entropy per output bit of 0.997, values of parameters P1 and P2 should satisfy the following inequalities:

$$P1 > 4,$$

$$P2 > 139.$$

2.3.1 *AIS 20/31 statistical testing of the PLL-TRNG in Cyclone V FPGA*

2.3.2 *Test information*

The temperature inside the chamber was increased in five steps: - 20°C, 0°C, 40°C, 85°C and 100°C. We recall that for the selected range of FPGA devices (commercial products) the permitted temperature range is between 0°C and 85°C (border values). In each temperature step, once the needed temperature was attained, data were acquired at five different voltages of the FPGA core and a 2 MB binary file containing the generated raw random bit stream. Parameters measured using the embedded online tests were saved in an additional log file and generated raw data were tested using the AIS 20/31 Procedure A and Procedure B test suites.

Table 2 shows information about the setup used for AIS 20/31 testing of the PLL-TRNG.

PLL-TRNG design version	B3.01
HECTOR daughterboard version	DBCVv11
HECTOR motherboard version	MBSF2v12
Test environment	Temperature chamber CTS T-40/25
Software	AIS 20/31 test suite
Evaluator	UJM, MIC

Table 2: PLL-TRNG setup for AIS 20/31 tests of data generated in Cyclone V FPGA

2.3.3 *Test summary*

Temp [°C]	**Voltage** [V]	**Online (emb.) tests**		**Offline tests**		
		Parameter P1	Parameter P2	Tests AIS 20/31 Procedure A	Tests AIS 20/31 Procedure B	Entropy per bit
	1.16	28	842	Passed	Passed	1.0000

Temp [°C]	Voltage [V]	Online (emb.) tests		Offline tests		
		Parameter P1	Parameter P2	Tests AIS 20/31 Procedure A	Tests AIS 20/31 Procedure B	Entropy per bit
- 20	1.13	26	841	Passed	Passed	1.0000
	1.10	28	814	Passed	Passed	1.0000
	1.07	29	821	Passed	Passed	0.9994
	1.04	25	820	Passed	Passed	1.0000
0	1.16	25	779	Passed	Passed	1.0000
	1.13	26	784	Passed	Passed	0.9996
	1.10	24	767	Passed	Passed	0.9997
	1.07	24	763	Passed	Passed	1.0000
	1.04	26	763	Passed	Passed	0.9999
40	1.16	23	750	Passed	Passed	0.9995
	1.13	24	749	Passed	Passed	1.0000
	1.10	26	744	Passed	Passed	0.9999
	1.07	23	733	Passed	Passed	1.0000
	1.04	24	735	Passed	Passed	1.0000
85	1.16	23	717	Passed	Passed	0.9996
	1.13	22	713	Passed	Passed	0.9998
	1.10	23	691	1/257 failed	Passed	0.9993
	1.07	22	716	Passed	Passed	1.0000
	1.04	21	702	Passed	Passed	1.0000
100	1.16	24	727	Passed	Passed	0.9999
	1.13	25	718	Passed	Passed	1.0000
	1.10	24	708	Passed	Passed	0.9999
	1.07	23	728	2/257 failed	Passed	0.9996
	1.04	24	712	Passed	Passed	1.0000

🟥	Out of range	▨	Corner	🟧	Border	🟩	Normal

Table 3: Test results for AIS 20/31 tests of the PLL-TRNG raw output signal and its Shannon entropy rate per bit in various operating conditions including "out of range", "corner", "border" and "normal" temperature and voltage conditions (PLL-TRNG in an FPGA testing)

2.3.4 Conclusions

As can be seen, parameters P1 and P2 always reached values, which were much higher than thresholds determined from the stochastic model (4 and 139, respectively). This result is confirmed by very high Shannon entropy rate per bit of the raw signal (the entropy rate was always much higher than the required rate of 0.997 per bit). Procedure A shows very small deviation from an ideal RNG, however, we recall that this procedure is aimed at testing of post-processed random signals and not of raw random signals. We used it just to verify the statistical quality of the raw signal, which was indeed confirmed.

We can conclude that the generator behaves very well in normal conditions including border and corner temperature and voltage values. We have shown that it works also in extreme operating conditions, out of the specified range. Nevertheless, Demonstrator 1 will include temperature sensors to detect such an operation to prevent attacks.

2.3.5 NIST SP800-90B statistical testing of the PLL-TRNG in Cyclone V FPGA

2.3.6 Test information

In this phase of testing, the random data files generated for AIS 20/31 testing were reused to apply the NIST SP 800-90B test suites. The generated raw data were first verified to be IID and then the min-entropy estimation was made following the test results.

Table 4 shows information about the setup used for testing of the PLL-TRNG according to the NIST SP 800-90B standard.

PLL-TRNG design version	B3.01
HECTOR daughterboard version	DBCVv11
HECTOR motherboard version	MBSF2v12
Test environment	Temperature chamber CTS T-40/25
Software	NIST SP 800-90B test suites for IID and non-IID branches
Evaluator	UJM, MIC

Table 4: PLL-TRNG setup for AIS 20/31 tests of data generated in Cyclone V FPGA

2.3.7 Test summary

Temp [°C]	**Voltage** [V]	**Offline tests**			
		IID branch	Min-entropy per bit	Non-IID branch	Min-entropy per bit
- 20	1.17	Passed	0.9956	Passed	0.9279
	1.13	Passed	0.9953	Passed	0. 9159
	1.10	Passed	0.9847	Passed	0.9269
	1.07	Passed	0.9835	Passed	0.9202
	1.03	Passed	0.9941	Passed	0.8934
0	1.17	Passed	0.9964	Passed	0.9179
	1.13	Passed	0.9904	Passed	0.9312
	1.10	Passed	0.9929	Passed	0.9056
	1.07	Passed	0. 9868	Passed	0.8993
	1.03	Passed	0.9937	Passed	0.9072
40	1.17	Passed	0.9885	Passed	0.9072
	1.13	Passed	0.9927	Passed	0.9072
	1.10	Passed	0.9953	Passed	0.9020
	1.07	Passed	0.9938	Passed	0.9102
	1.03	Passed	0. 9836	Passed	0.9159
85	1.17	Passed	0.9801	Passed	0.9020
	1.13	Passed	0.9913	Passed	0.9092
	1.10	Passed	0.9802	Passed	0.8967
	1.07	Passed	0.9912	Passed	0.9092
	1.03	Passed	0.9869	Passed	0.9020
100	1.17	Passed	0.9831	Passed	0.9119
	1.13	Passed	0.9951	Passed	0.8811
	1.10	Passed	0.9945	Passed	0.9265
	1.07	Passed	0.9806	Passed	0.9205

Temp [°C]	Voltage [V]	Offline tests			
		IID branch	Min-entropy per bit	Non-IID branch	Min-entropy per bit
	1.03	Passed	0.9792	Passed	0.9102

🟥	Out of range	▨	Corner	🟧	Border	🟩	Normal

Table 5: Test results for NIST SP 800-90B tests of the PLL-TRNG raw output signal and its min-entropy rate per bit in various operating conditions including "out of range", "corner", "border" and "normal" temperature and voltage conditions (PLL-TRNG in an FPGA testing)

2.3.8 Conclusions

It can be seen in Table 5 that the NIST SP 800-90B test results correspond well to those of AIS 20/31. All the tests showed the data to be IID, featuring high min-entropy rate. The entropy rate estimation for the non-IID branch is known to be more conservative, but the estimated entropy is still significantly high.

Again, tests confirmed that the generator behaves very well in normal conditions including border and corner temperature and voltage values, but also in extreme operating conditions, out of the specified range.

2.4 Statistical testing of the PLL-TRNG implemented in ST ASIC

STMicroelectronics implemented the PLL-TRNG in an ASIC aimed at automotive applications. The ASIC contains two PLLs. One of the PLLs is used by the system-on-chip as the USB clock, the other one is dedicated to the TRNG. The data interface was not compatible with the HECTOR evaluation boards and therefore dedicated hardware and software was developed for data acquisition.

Design parameters:

CLK0 = 198.57 MHz, CLK1 = 480 MHz, KD=139, KM=336, raw data rate = 1.428 Mbits/s

First silicon prototypes of the ASIC have been received enabling initial evaluations of the TRNG and to confirm that it is working as expected. The TRNG was tested on its application board by running AIS 20/31 and NIST SP 800-90B tests on the raw data at the decimator output, which is available in a specific test mode. The data was collected by a PC connected to the system-on-chip.

The application board was put in an oven in order to perform temperature tests.

2.4.1 AIS 20/31 statistical testing of the PLL-TRNG in ST ASIC

2.4.2 Test information

As described above, the following results were obtained on random data generated in ST ASIC soldered on a dedicated application board. A set of 2 MB data files was generated at different operational temperatures and the AIS 20/31 test procedure was applied on the generated data.

PLL-TRNG design version	STMicroelectronic-specific design
Hardware version	STMicroelectronic-specific application board
Test environment	Temperature chamber Espec Platinus K Series
Software	AIS 20/31 test suite
Evaluator	STR

Table 6: PLL-TRNG setup for AIS 20/31 tests (STR ASIC testing)

2.4.3 *Test summary*

Decimator Factor N	Temperature [°C]	Tests AIS 20/31 Procedure B	Bias (|P(1) - 0.5|)	Shannon entropy per bit
4	- 40	Passed	0.0079	0.99987
4	- 20	Passed	0.0049	0.99935
4	80	Passed	0.0015	0.9996

Table 7: Test results for AIS 20/31 tests on PLL-TRNG (STR ASIC testing)

The above AIS 20/3 results show successful tests for all the temperatures tested, however the decimation factor had to be set to 4 to get successful results. This means dividing by 4 the normal raw data rate, resulting in a raw data rate of 0.35714 Mbits/s

With a decimation factor of 1 or 2, some temperature cases were showing a bias slightly higher than the test limit.

2.4.4 *Conclusions*

The AIS 20/31 tests passed in all cases in the temperature range considered and with a decimation factor of 4.

2.4.5 *NIST SP800-90B statistical testing of the PLL-TRNG in ST ASIC*

The NIST SP 800-90B tests were performed on data generated in the same hardware configuration as that described in Section 2.4.

2.4.6 *Test information*

Both IID and non-IID tests were performed.

The IID test first verifies that there is no evidence the data is not-IID. The IID test suite then proceeds to compute an entropy estimate, given in the test summary below.

PLL-TRNG design version	STMicroelectronic-specific design
Hardware version	STMicroelectronic-specific application board
Test environment	Temperature chamber Espec Platinus K Series
Software	NIST SP800-90B Python package for IID and non-IID sources
Evaluator	STR

Table 8: PLL-TRNG setup for NIST SP 800-90B tests (ST ASIC testing)

2.4.7 Test summary

Decim. Factor N	Temperature [°C]	Tests 800-90B - Branch IID		Tests 800-90B - Branch non-IID	
		Result	Min-entropy	Result	Min-entropy
4	- 40	Passed	0.912537	Passed	0.967911
4	- 20	Passed	0.922601	Passed	0.987017
4	80	Passed	0.923184	Passed	0.992403

Table 9: Test results of NIST SP 800-90B tests on PLL-TRNG (ST ASIC testing)

Testing in the IID branch was successful in all cases when the decimation factor was 4.

Nevertheless, we also passed the non-IID tests, which compute the minimum entropy by several different methods, and keep as result the lowest obtained value. This explains why the min-entropy for the non-IID branch is always lower than the min-entropy for the IID branch of tests.

2.4.8 Conclusions

The SP800-90B tests require a minimum entropy claim to be made, together with an IID or non-IID claim. Tests are successful when the measured entropy is higher than the claimed entropy.

The raw data is then sent to an approved conditioning component, which task it is to raise the entropy from the claimed value to an ideal value of one.

In the absence of a claim, success or failure is not relevant, but the min-entropy values obtained on the ST ASIC are quite high, making it easy to reach a claimed value.

The results show that the HECTOR PLL-TRNG design – when implemented on an ASIC – is capable of fulfilling the requirements. This is important for reaching the HECTOR objectives O12 to O14 since ASICs are the targeted implementation platform for industrialization of the HECTOR TRNGs.

Chapter 3 AIS20/31 evaluations of the HECTOR TRNG primitives

An AIS20/31 evaluation covers technical aspects like entropy estimation, design of the stochastic model and selection of embedded tests. In a Common Criteria evaluation it is verified if the security claims that the developer makes are complete and sound, using developer evidence. Compliance to technical aspects are covered in Chapter 2. The verification of the security claims is described in this chapter.

Secure products containing TRNGs may be subjected to security evaluation for certification. This requires that the TRNGs fulfill security and performance claims that fit the requested level of protection. For commercial exploitation of the HECTOR TRNG designs it will be important to know to what extent these designs comply to the requirements, and to have readily-available documentation packages and "developer evidence" that needs to be provided to evaluation facilities.

The AIS20/31 evaluations are done in the context of Common Criteria evaluations ([25], [26], [27], [28]). This is the preferred choice for developers of high-end security products in case they seek for general acceptance in the market. For this reason we chose to do the AIS20/31 evaluation using the Common Criteria methodology. This is a formal method, which normally requires a governmental scheme to 'oversee' the project (German BSI, Dutch NSCIB, French ANSSI etc.). Schemes only allow complete and real products to be evaluated. Since the HECTOR TRNGs are developments that will be part of such future products, the evaluation was done without scheme.

The aim of the AIS20/31 evaluations in the CC context presented in this chapter is that the HECTOR partners developing TRNGs:

- Acquire knowledge about the evaluation methodology used in Common Criteria evaluations for AIS20/31 evaluation
- Get familiar with the terminology of a formal evaluation process
- Get knowledge on the requirements that are imposed on creating and delivery of the evaluation evidence and prepare the required documentation packages for HECTOR TRNGs
- Understand that designs can be made in such way as to support the evaluation process (design for evaluation)
- Get feedback on their designs and confirmation on their certification
- Experience evaluation iterations during the evaluation process

Ease of evaluation is important for subsequent industrialization of HECTOR TRNGs. In most cases it is easy to adapt the design in such way that it benefits the evaluation, thus saving time and costs.

Verification of AIS20/31 certification feasibility corresponds and simplification of the certification process respectively correspond to HECTOR Objectives O6 and O16.

HECTOR TRNG designs are not developed for a specific use-case. They can be applied for any application that requires true random numbers, such as communication, automotive, IoT or financial. The TRNG designs are "general purpose" and therefore the Evaluation Assurance Level for certification can vary per application. For our evaluations without scheme the highest assurance level was chosen, resulting in the best coverage of CC work units.

Evaluations for both the PLL-TRNG and the DC-TRNG are described in Appendix F and Appendix G respectively. Because no specific application is associated with the TRNGs to be evaluated, the evaluations were completed for the TRNG cores only. This automatically causes some of the evaluation work items to be formally ‘inconclusive’, because there is no use-case evidence associated. In our context this is not relevant; the goal here was to demonstrate that HECTOR TRNGs can be evaluated successfully and that the associated core evidence is available and correct. This has been accomplished (HECTOR Objective O6).

Chapter 4 Evaluation of the PLL TRNG regarding its compliance with the security requirements of the French DGA MI

The French Direction Générale de l'Armement - Maitrise de l'information (further only DGA MI) is currently working on a document (a draft version is available internally, not yet publicly), which specifies requirements on TRNGs aimed at high security applications.

The draft version of the document has the title: *Recommendations for Design and Validation of a Physical True Random Number Generator Integrated in an Electronic Chip*. Recommendations described in this document are fully compliant with the German document AIS 20/31 [1]. The main objective of the new document edited by the French DGA MI is to help designers to design high security TRNGs, for which DGA MI gives a list of additional recommendations and examples.

During the HECTOR project there was extensive collaboration between UJM and the French DGA MI on the methodology for creating high-quality TRNGs (one of the advisory board members is employed by the DGA MI). The view of the DGA MI is in line with the HECTOR design approach that focuses on entropy estimation by using stochastic models. It is therefore an excellent opportunity to evaluate the HECTOR PLL TRNG according to the DGA MI security requirement document.

The main differences between the French document and AIS 20/31 are as follows:

1. The French document does not deal with deterministic and hybrid random number generators. Recommendations concern only the physical part of the generator and the analog to digital conversion up to the output of the raw binary signal (the digital noise).
2. The French document requires designers to create/describe the stochastic model of the source of randomness (the analog noise) and not only of the stochastic model of the whole generator as required by AIS 20/31. In other words, the French document requires the designer to propose/describe stochastic models of both the physical source of randomness and the whole source of the digital noise, including analog-to-digital converter.
3. The French document requires explicitly that the whole data path between the physical source of randomness and the output of the generator must be tested. This requirement concerns in particular the analog-to-digital converter, but also other deterministic parts of the generator placed between the physical source of randomness and the generator output.

Next, we will discuss the way the HECTOR PLL-TRNG satisfies requirements of the French DGA MI.

4.1 Requirements on the physical true random number generator aimed at high security applications

As presented before, the requirement of the French DGA MI completes the AIS 20/31 document regarding the source of the digital noise, which consists of the source of randomness and the randomness extraction circuitry (e.g. the analog-to-digital converter). Although the HECTOR PLL-TRNG includes also the cryptographic post-processing and related Known answer test (KAT), in this chapter we will analyze solely the conformity of the source of the digital noise with the requirements of the French DGA MI.

The source of the digital noise is implemented in HECTOR daughter boards featuring Intel Cyclone V or Xilinx Spartan 6 FPGA. The block diagram of the source of the digital noise is depicted in Figure 1. It consists of the source of randomness (the jitter of clock signals generated in two PLLs (PLL0 and PLL1) and of the time-to-digital converter, which counts number of samples (the output of sampling D flip-flop, DFF) in the periods T_Q. The last bit of the counter represents the output of the TRNG core - the digital noise.

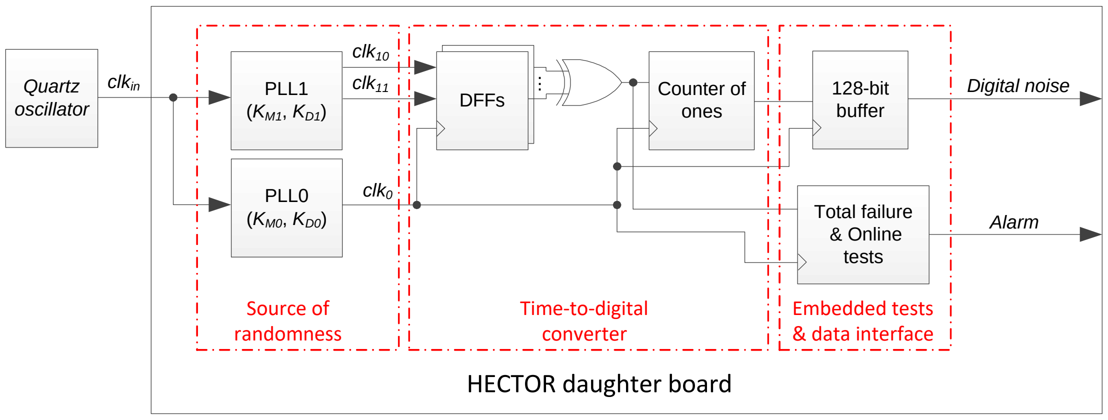


Figure 1: Source of the digital noise implemented in HECTOR daughter boards

4.2 Stochastic model of the source of randomness in PLL-TRNG

Three physical sources of randomness can contribute to the entropy rate at the output of the PLL-TRNG:

1. The jitter of the PLL input clock
2. Intrinsic noise of the PLLs and its contribution to the PLL output clock jitter
3. Supply noise contributing to the PLL output clock jitter

However, only one of these sources can be considered to be non-manipulable and thus robust: the intrinsic noise of the PLL. Therefore, the security approach of the HECTOR partners was to reduce the impact of other two sources listed above to a minimum and to model the intrinsic noise of the PLL as a source of randomness.

The jitter of the PLL input clock was reduced by the choice of the quartz oscillator. A low jitter quartz oscillator (SI531SC oscillating at 125 MHz) is used on both HECTOR evaluation board and HECTOR Demonstrator 1. This choice guarantees that the attacker cannot further reduce the input clock jitter and thus reduce the output entropy rate.

The power supply noise source was reduced by their careful design - only low noise linear power supplies are used.

Consequently, the whole accountable noises of the PLL-TRNG come from the PLL, in which all individual blocks contribute: the voltage-controlled oscillator (VCO), the phase-frequency detector, and the charge pump. However, the VCO phase noise dominates among all these sources.

In order to make the thermal noise of the VCO the main entropy contributor, the following design choices were made:

1. The output clock frequency was chosen to be as high as possible in order to reduce the contribution of the flicker noise to the output jitter (note that the flicker noise is auto-correlated and its contribution should be therefore reduced as much as possible).

2. To reduce the long term jitter at the PLL output, the PLL bandwidth was chosen to be as large as possible.

Consequently, both the rising and the rising and falling edges of the sampled signal clk_1 have the same standard deviation σ_{clk1} composed mainly of jitter coming from the thermal noise $\sigma_{clk1} \approx \sigma_{clk1,th}$.

As the sampling signal clk0 is also produced by a PLL, we can consider that this signal is also mainly influenced by the thermal noise with a standard deviation $\sigma_{clk0} \approx \sigma_{clk0,th}$. Usually, we consider the sampling signal to be ideal (jitter frr) and the sampled signal is composed of the contribution of both jitters. Due to the PLL principle (the jitters are bounded) and the independence of thermal jitters, it is reasonable to assume that the equivalent jitter to consider on the sampled signal clk_1 is given by

$$\sigma = \sqrt{\sigma_{clk_1}^2 + \sigma_{clk_0}^2}.$$

For i ∊ [|0;KD-1|]n the phase φ_i of the clock signal clk_1 influenced by the thermal noise can be seen as a random variable following a normal law of mean

$$\mu_i := \varphi_0 + i \times \frac{T_{clk_1}}{K_D} \mod T_{clk_1}$$

and variance σ_{th}^2 where φ_0 is the initial phase of the clk_1 for i = 0. The distribution of the phase φ_i of the clock signal clk_1 represents the model of the source of randomness characterized by the variance σ_{th}^2.

The model of the PLL-TRNG core (of the generator of the digital noise) takes the known jitter (the jitter is measured using embedded tests) as an input parameter and gives first the probability that the sampled bit (one of K_D bits in period T_Q) is equal to one:

$$P(X_i = 1) = 1 - \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{T_{clk_1}} e^{-\frac{(t-\mu_i)^2}{2\sigma^2}} dt + \frac{1}{\sqrt{2\pi}\sigma}\int_{0}^{dc\cdot T_{clk_1}} e^{-\frac{(t-\mu_i)^2}{2\sigma^2}} dt$$

where dc means the duty cycle of signal clk_1 and by XOR-ing KD bits in the counter of samples, we obtain the probability that the PLL TRNG output bit is equal to one:

$$P\left(\bigoplus_{k=0}^{nb_{clk}-1} X_{i,k} = 1\right) = \frac{1}{2} + (-2)^{nb_{clk}-1} \prod_{k=0}^{nb_{clk}-1} \left(P(X_{i,k} = 1) - \frac{1}{2}\right)$$

4.3 Stochastic model of the entire PLL-TRNG

The model of the PLL-TRNG core (of the generator of the digital noise) takes the known jitter (the jitter is measured using embedded tests) as an input parameter and gives first the probability that the sampled bit (one of K_D bits in period T_Q) is equal to one:

$$P(X_i = 1) = 1 - \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{T_{clk_1}} e^{-\frac{(t-\mu_i)^2}{2\sigma^2}} dt + \frac{1}{\sqrt{2\pi}\sigma}\int_{0}^{dc\cdot T_{clk_1}} e^{-\frac{(t-\mu_i)^2}{2\sigma^2}} dt$$

where dc means the duty cycle of signal clk_1 and by XOR-ing KD bits in the counter of samples, we obtain the probability that the PLL TRNG output bit is equal to one:

$$P\left(\bigoplus_{k=0}^{nb_{clk}-1} X_{i,k}=1\right)=\frac{1}{2}+(-2)^{nb_{clk}-1}\prod_{k=0}^{nb_{clk}-1}\left(P\left(X_{i,k}=1\right)-\frac{1}{2}\right)$$

4.4 Testing the entire data path between the source of randomness and cryptographic post-processing

As can be seen in Figure 1, embedded tests (the Total failure test and Online tests) evaluate the quality of the source of randomness at the output of the sampler. The counter, the data buffer and data transmission between daughter board and motherboard, which contains the cryptographic post-processing blocks are not tested. Therefore, additional tests are performed at the input of the cryptographic post-processing block.

Figure 2 presents testing strategy of the PLL-TRNG. Three levels of tests are used:

1. Embedded tests as required by AIS20/31 (Total failure test and online tests)
2. Continuous tests as required by NIST SP 800-90B (Repetition count and Adaptive proportion tests)
3. Known answer test of the cryptographic post-processing block.

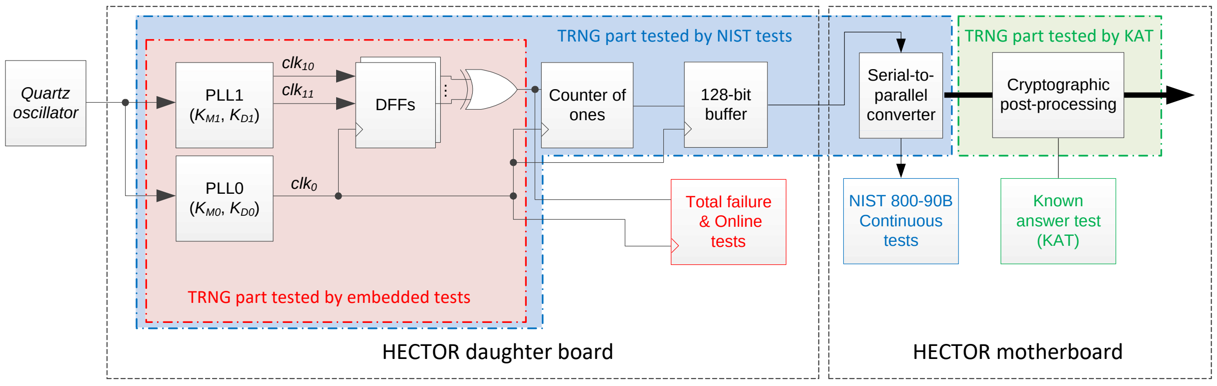

TRNG part tested by NIST tests
TRNG part tested by KAT
Quartz oscillator
PLL1 (K_{M1}, K_{D1})
PLL0 (K_{M0}, K_{D0})
clk_{10}
clk_{11}
clk_0
DFFs
Counter of ones
128-bit buffer
Serial-to-parallel converter
Cryptographic post-processing
Total failure & Online tests
NIST 800-90B Continuous tests
Known answer test (KAT)
TRNG part tested by embedded tests
HECTOR daughter board
HECTOR motherboard

Figure 2: PLL-TRNG testing strategy

Embedded tests test the source of randomness - the clock jitter. Based on the stochastic model they evaluate online the entropy rate at generator output - the size of the jitter is measured continuously and compared with the threshold defined by the model.

NIST continuous tests ensure compliance of the PLL-TRNG with the American standard, but also with requirements of the French DGA MI - they test also the entire data path between the source of randomness and the cryptographic post-processing, which is tested using the Known answer test.

As can be seen in Figure 2, all the parts of the generator are tested by appropriate tests, which make it compliant with AIS 20/31, NIST SP 800-90B and requirements of DGA MI.

4.5 Conclusion

In this chapter, we explained that the PLL TRNG fulfills requirements specified by the French DGA MI and namely:

- The model of the PLL TRNG core given in Section 4.4depends on the source of randomness (the clock jitter), the model of which is given in Section 4.3.

- All PLL TRNG data paths are tested by continuous online tests: dedicated statistical tests for the source of randomness, black box tests for the data path between the TRNG core and the post-processor and the Known answer test for the deterministic part of the generator.

It is thus suitable for applications requiring high security guarantees.

The evaluation also showed that the HECTOR deliverables fulfilled the security evaluation requirements of the French DGA MI. The HECTOR objectives aim to have demonstrable entropy estimation using stochastic models, which is also prescribed by the DGA MI. In addition the work contributed to the HECTOR objectives of evaluability and standardization.

Chapter 5 PUF design and testing

5.1 Introduction

HECTOR aims at developing Physically Unclonable Functions based on a modeling approach (HECTOR Objective O4). This way both the PUF entropy can be estimated beforehand, as well as the stability. The path taken to achieve this objective is by first modeling the PUF and then measuring the actual behavior in real silicon. This chapter describes the modeling approach for the TERO PUF, which was selected as candidate for further research and implementation in FPGA and ASIC. Chapter 5 describes actual testing of a TERO-PUF implementation on FPGA in order to verify the model.

FPGAs – using the HECTOR motherboard and daughterboards – are chosen as implementation platform for fast turn-a-round during development. Later in the project selected PUF (and TRNG) designs were designed for implementation on the HECTOR ASICs, because these offer a better representation of real-life products for industrialization. It is unfortunate that the HECTOR ASICs could not be delivered on-time by the foundry, for reasons beyond control of the HECTOR consortium. Even with a five-month extension of the project duration the ASICs were not delivered on time to be included in the research. According to our contingency plan all testing was therefore done on FPGAs, which is less representative for industrial products than ASIC implementations.

5.2 Context

Scaling down of electronic devices is causing industrial problems since reducing the size of electronic components is increasing manufacturing process variability (MPV) such as mismatch between transistors. Although it is a challenge for most integrated circuits, Physical Unclonable Functions (PUF) are taking advantage of it since they exploit MPV to extract a secret and unique identifier per die which usually is a binary number.

For those reasons, PUFs have been a hot topic in last decades. Since the first introduction of PUF by Pappu in 2002 [9], many PUF principles have been published and implemented on FPGA and ASIC. The most known are memory based PUFs including for example SRAM PUFs [10] and delay based PUFs such as arbiter PUFs [11] and ring oscillator PUFs [12].

Regardless of the principle, an efficient PUF should provide an identifier per die, which is unique, unpredictable and stable over time and environmental variations.

While some work have been done on quality evaluation of PUFs, to the best of our knowledge there is no standard and efficient way to ensure PUF unpredictability.

The most commonly used metrics to evaluate the predictability of PUFs are uniqueness, uniformity and bit-aliasing. There are dozens of papers that describe various evaluation criteria, such as steadiness, randomness, uniqueness, entropy and many more. However, a standardized metric does not exist. For developers it is very challenging to understand what is relevant and of high importance and which parameters are not relevant for a specific use case.

First, Maiti et al. [13] proposed a systematic method with a set of metrics to evaluate PUFs.

Later, Pehl et al. [14] tried to establish some further evaluation approaches like joint entropy to identify bits correlation and therefore predictability weaknesses. They also demonstrate design flaws can justify bad statistical results.

Others proposed to use test suites like it is done for True Random Number Generators (TRNG) with NIST or AIS 20/31 test suites [15]. It turns out that the minimum amount of data required for those tests is one million bits for NIST and eight million bits for AIS20/31 tests. Thus, to perform acceptable entropy estimation the number of responses needed is very high and it would require millions of test chips, which does not seem possible.

Moreover, having perfect statistical properties is necessary but not sufficient.

Therefore, all those tests must be used to identify weaknesses but cannot guarantee unpredictability of PUFs.

As shown by the TRNG community [16], [17], the best way to ensure unpredictability is to carefully estimate the entropy based on a strong model of the entropy source. In the case of PUFs the entropy is coming from Multiple Process Variations (MPV). Following Haddad's approach [18], the idea is to establish a stochastic model of PUF based on CMOS process variability.

HECTOR is one of the first projects pursuing such a path; however, it is a challenging and time-consuming activity. Essentially the need of stochastic models is one of the main requirements, which is not so easy to establish for PUF instantiations. This section describes ways to model properties that can be used as basis for stochastic models.

Since this work is focusing on microelectronic (FPGAs and ASICs) and MOS transistors are the major source of mismatch in microelectronic circuits, focus is put on this kind of devices.

5.3 CMOS process variability

Two types of variations exist that affect transistors: global and local. The first one is deterministic while the second one is stochastic.

5.3.1 Global variations

Global variations come from inaccuracies of the production process like temperature gradients across the wafer during annealing [4]. This causes a gradient over wafers or chips. It is a deterministic mismatch and thus, not wanted for PUF applications. Moreover, there exist ways to reduce it with proper designs and layouts. Finally, if compared cells of the PUF are close enough, they will not be affected by the global variations. That is why global mismatch will not be considered in the rest of this chapter.

5.3.2 Local variations

Local variations, which are by nature uncorrelated, come from stochastic atomic level differences. In nanometer-scale transistors, the impact of each atom on the transistor properties is very high and there is, at the moment, no way to control it. This is the kind of mismatch a PUF exploits to generate a unique identifier. Studies have shown that there are four major local variation sources that affect a planar bulk MOSFET: Random Dopant Fluctuations (RDF), Line Edge Roughness (LER), Oxide Thickness Variations (OTV) and Poly-Si Granularity (PSG) [19].

Figure 3 shows those types of mismatch on transistors:

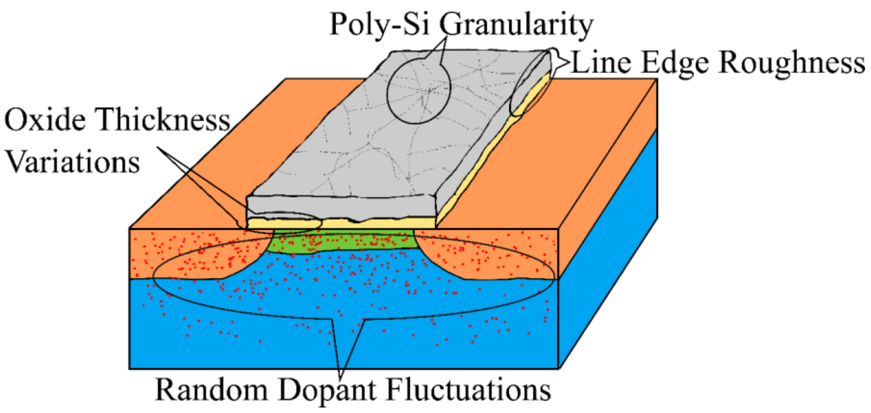


Figure 3: CMOS local process variations

The impact of each variation is technology-dependent and becomes more important when reducing the transistor's size. Those statistical variations influence electrical parameters such as the threshold voltage (Vtk) and the current factor (β). According to Pelgrom's model, the standard deviation of Vth and β is proportional to the inverse of the square root of the active device area [20]:

$$\sigma_{\Delta Vtk} = \frac{A_{\Delta Vtk}}{\sqrt{WL}} \qquad \sigma_{\Delta\beta/\beta} = \frac{A_{\Delta\beta/\beta}}{\sqrt{WL}}$$

Where $A_{\Delta Vtk}$ and $A_{\Delta\beta/\beta}$ are parameters characterizing a transistor technology and manufacturing line.

5.4 Modeling approach

5.4.1 Selected principle: the TERO PUF

As stated in deliverables D2.1 and D2.2, after implementation and comparison of different PUF principles, the selected PUF principle in the scope of the HECTOR project is the TERO based PUF. The modeling approach pursued by HECTOR is to validate an electrical model of the core of the PUF, the TERO cell. Then, a stochastic model is derived from the electrical model including CMOS process variations that serve as source of randomness and evaluate the impact on a complete TERO PUF.

The TERO is an oscillator with 2 states: one oscillating transitory state and one non-oscillating stable state. It is composed of an even number of inverters (delay gates) and two AND gates (activation gates).

Figure 4 depicts a typical TERO cell composed of three inverters per branch:

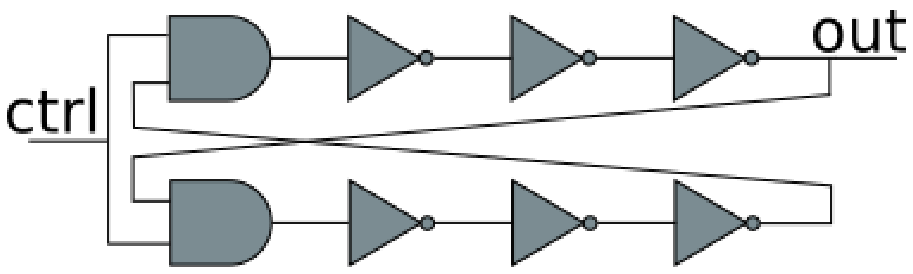


Figure 4: Typical TERO cell

When the control signal, denoted "ctrl" in figure 2, switches from the logical low level to the logical high level, two electrical events start to propagate across the cell. Due to mismatch between the CMOS transistors composing the cell, caused by manufacturing process variation, one event propagates faster than the other. That is why, the output oscillation frequency stays constant but the duty cycle will move towards 0% or 100% until oscillations stop.

Figure 5 depicts an example of TERO behavior over time before stopping:

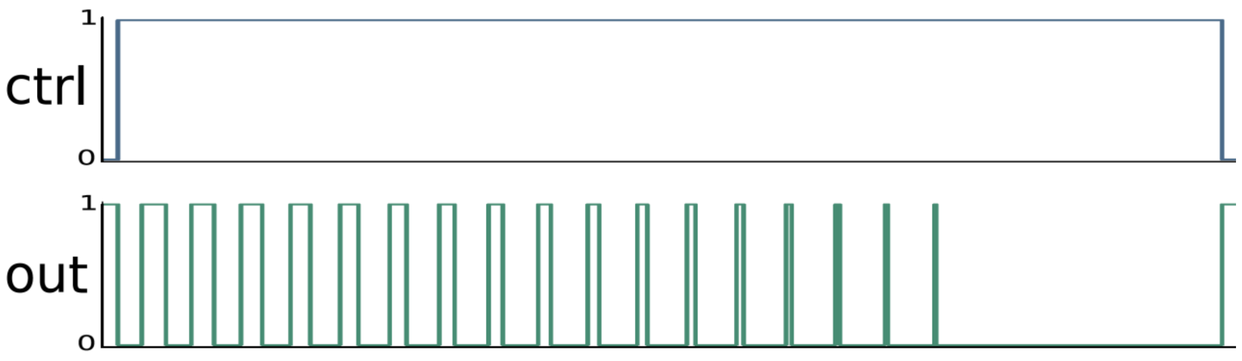


Figure 5: TERO output behavior after excitation

The TERO PUF exploits the number of oscillations of TERO cells. The first objective is therefore to define a physical model that predicts the number of oscillations of a TERO cell.

5.4.1 Physical model of the TERO cell

A physical model of the TERO, established by Reyneri in 1990 [21], is already available. One of HECTOR objectives is to validate this model and then apply CMOS process variations on it to derive a stochastic model. Let us consider Reyneri’s model to define a range of circuit parameters and input conditions that influence the number of oscillations.

5.4.1.1 Inverter model

The inverter is the elementary component of the TERO cell. Thus, modeling of the TERO cell starts with modeling of a single inverter. Based on Reyneri’s work an inverter can be divided in three parts:

- An **ideal comparator**
- A **delay stage**, denoted $T1$
- A **slope limiter** that produces rise and fall times, denoted $T2$

Figure 6 shows the decomposition of an inverter in three parts:

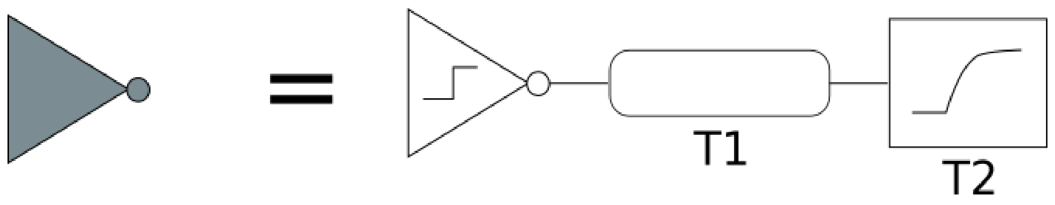


Figure 6: Inverter decomposition

This gives an inverter model composed of three parts:

- **A constant region** corresponding to the delay between a commutation at the input of the inverter and the time before the output starts to change.
- **A linear region** corresponding to the time when the transistors of the inverter are in saturated mode
- **An exponential region** corresponding to the time when the transistors of the inverter are in resistive mode. It starts when it reaches Vtk (rising edge) or $Vcc - Vtk$ (falling edge), Vcc being the supply voltage and Vtk the threshold voltage.

Figure 7 shows the inverter model:

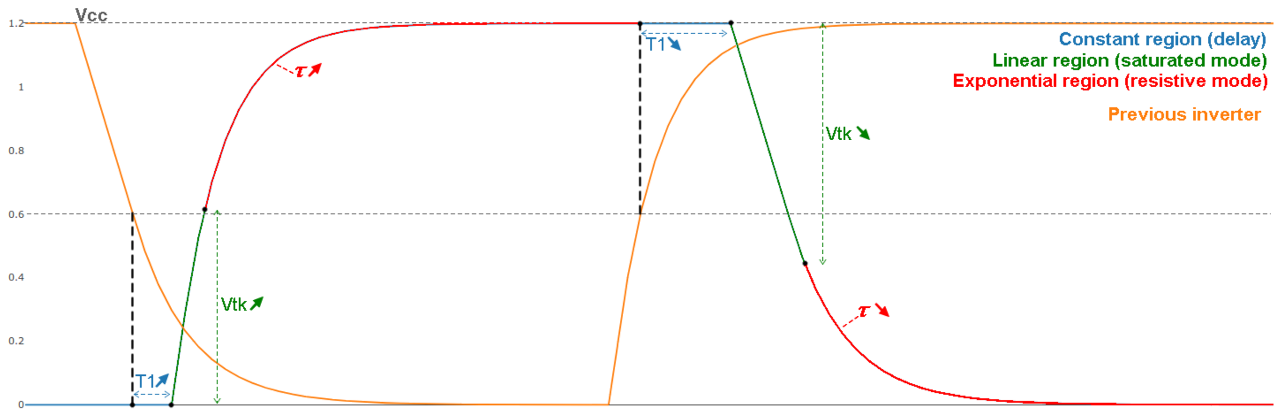


Figure 7: Inverter model

The inverter model parameters are the following:

- $\boldsymbol{T1}$ is the delay between a commutation at the input of the inverter and the time before the output starts to change.
- $\boldsymbol{\lambda}$ is the slope coefficient of the linear region.
- $\boldsymbol{\tau}$ is the timing constant of the exponential region.
- $\boldsymbol{Vtk}$ is the threshold voltage of a transistor, corresponding to the commutation between linear region and exponential region of the inverter.
- $\boldsymbol{Ttk}$ is the time when output of the inverter reaches Vtk.

Thus, the inverter output has three states (example for a falling edge):

- **Constant region** $(\boldsymbol{t} \leq \boldsymbol{T1})$: $\boldsymbol{V} = \boldsymbol{Vcc}$
- **Linear region** $(\boldsymbol{T1} \leq \boldsymbol{t} \leq \boldsymbol{Ttk})$: $\boldsymbol{V} = \boldsymbol{\lambda}.(\boldsymbol{t} - \boldsymbol{T1}) + \boldsymbol{Vcc}$
- **Exponential region** $(\boldsymbol{Ttk} \leq \boldsymbol{t})$: $\boldsymbol{V} = (\boldsymbol{Vcc} - \boldsymbol{Vtk}).\boldsymbol{e}^{-\frac{\boldsymbol{t}-\boldsymbol{Ttk}}{\boldsymbol{\tau}}}$

Figure 8 shows the inverter model with the three distinct states for a falling edge:

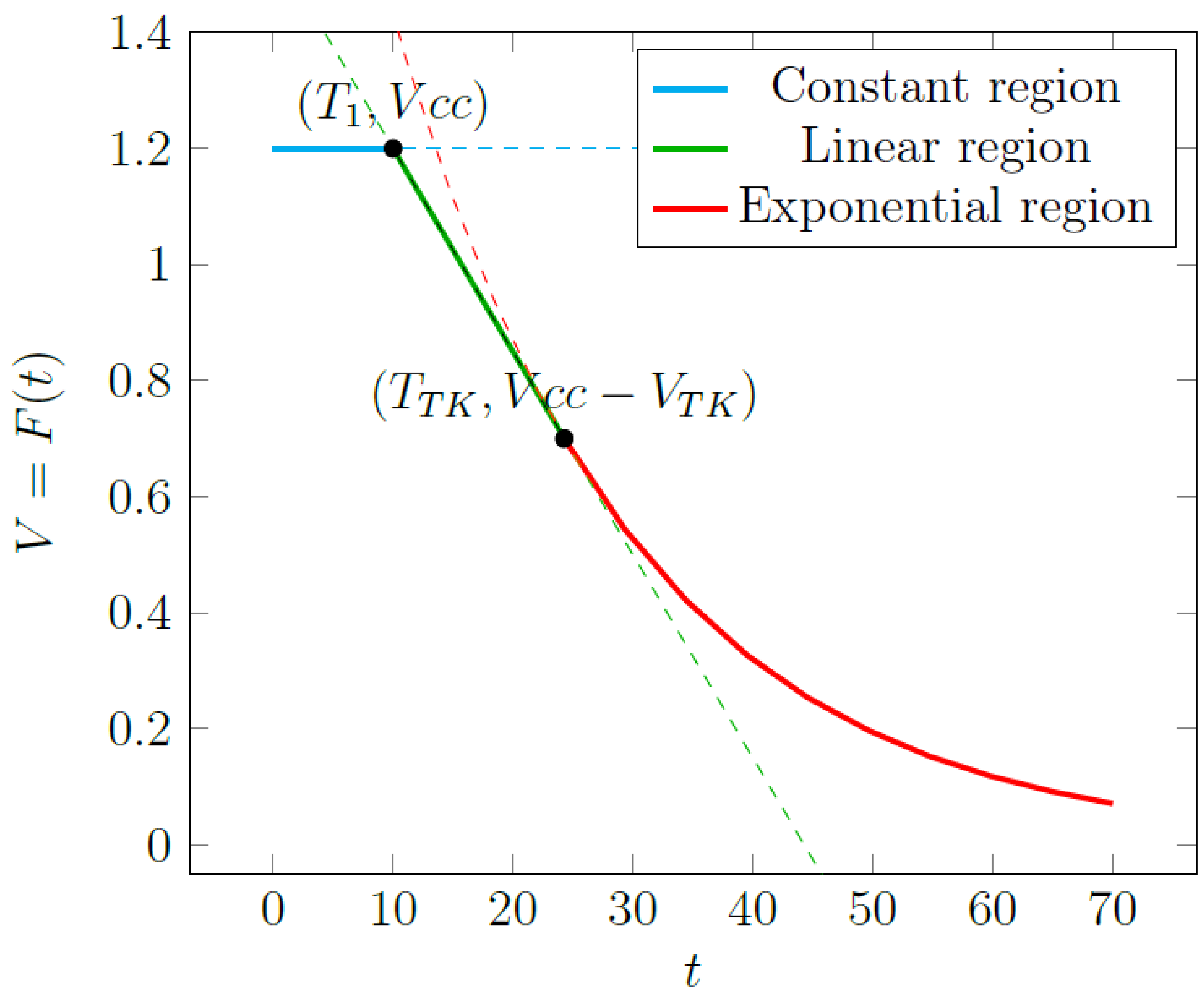


Figure 8: Inverter falling edge model

5.4.1.2 Impact on the TERO output duty cycle

With this inverter model, we can identify three different effects that will affect the duty cycle of the TERO output signal.

The first one is the slope limiter. Any pulse of duration $Ti \geq T2$ passes unaltered through one inverter. However, if $Ti < T2$, the slope limiter reduces Ti by an amount $(T2 - Ti)$.

Figure 9 shows an example:

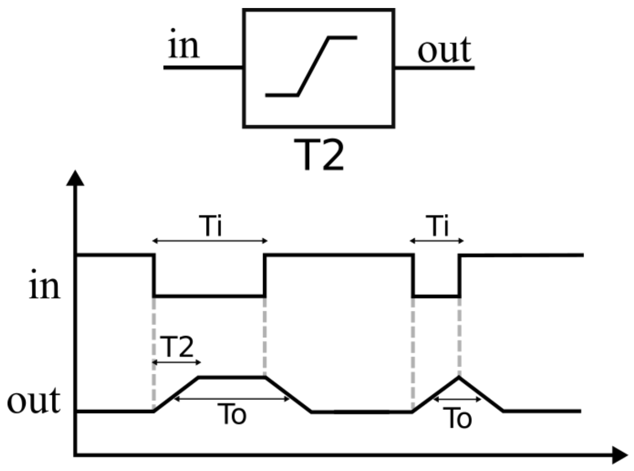


Figure 9: Slope limiter effect

Then, the differences between rising and falling edges inside an inverter and between inverters also reduce the pulse. Indeed, if we consider one inverter of the TERO cell, the first event always goes through the inverter by generating a rising edge. The second event always generates a falling edge. Thus, at every cycle in the TERO loop, the pulse Ti is reduced by an amount Δ (being the difference between rising and falling edges crossed by the first event (resp. $tr1$ and $tf1$) and the rising and falling edges crossed by the second event (resp. $tr2$ and $tf2$)).

$$\Delta = (tf1 - tr1) - (tf2 - tr2)$$

The third effect that affects the duty cycle is called the drafting effect. When the inverter output approaches 0 or VCC (VCC being the supply voltage), transistors enter the resistive region and output signal becomes exponential. Because of that, the final value at the output of an inverter after a rising edge (resp. a falling edge) is not VCC (resp.0) but an intermediate value close to it. Thus, when the second event arrive at the input of the inverter the switching will not start from 0 or VCC but from this intermediate value.

Figure 10 shows the TERO behavior over time before stopping with those three effects taken into account:

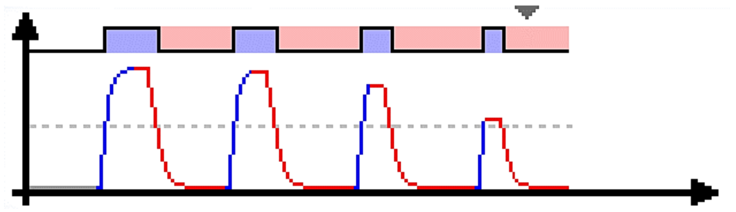


Figure 10: TERO output

5.4.1.3 TERO model

The TERO is composed of multiple concatenated inverters. The TERO cell is characterized by the parameters of each inverter and, at every step, their internal state is determined by the drafting effect. Based on that learning, we developed a TERO model and described it in R. It is composed of recursive functions of the inverter that allow us to simulate the behavior of the TERO. The advantage of R is that it allows simulating the process variations.

Figure 11 shows result of R simulation with our model for a TERO composed of 11 inverters per branch. It represents the evolution of the output duty cycle over periods until oscillations stop. In this case, the TERO has a number of oscillations of 99.

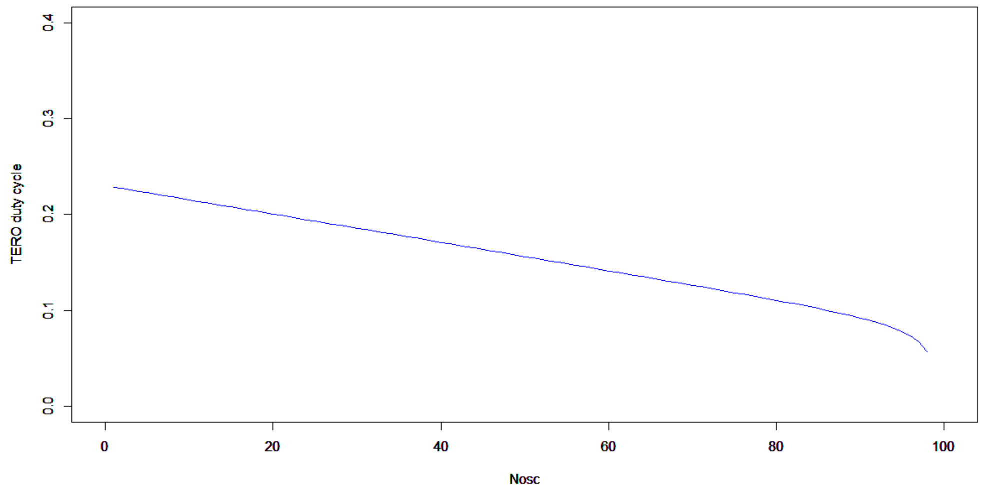


Figure 11: Simulation of the duty cycle evolution in TERO output oscillations

5.4.2 *Impact of process variations on the TERO cell*

Figure 12 shows result of simulations in R for the same TERO cell with process variations simulated by a normal law with a standard deviation of 0,01 applied on Vtk. This time the number of oscillations varies between 50 and more than 600.

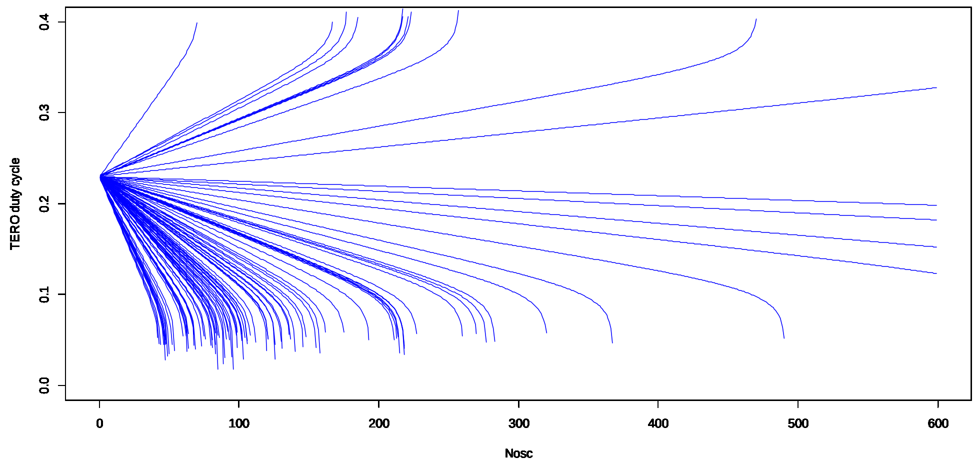


Figure 12: TERO output duty cycle evolution with process variations on Vtk

From this simulation, it is clear that the TERO cell presents very high sensitivity to process variations, which is an extremely good parameter for PUF applications.

5.4.3 *Impact of noise and environmental conditions*

Figure 13 shows result of simulation in R for the same TERO cell but this time with jitter noise with variations of 0.2% of the TERO period. The resulting number of oscillations varies between 40 and 225:

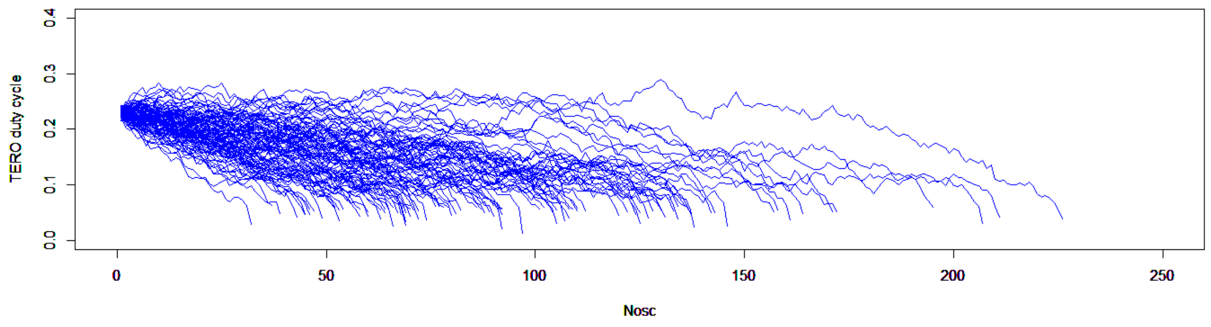


Figure 13: TERO output duty cycle evolution with jitter noise

In the next experiment, the variations are induced on the power supply (Figure 14). With voltage variations of 8%, the number of oscillations ranges between 60 and 150:

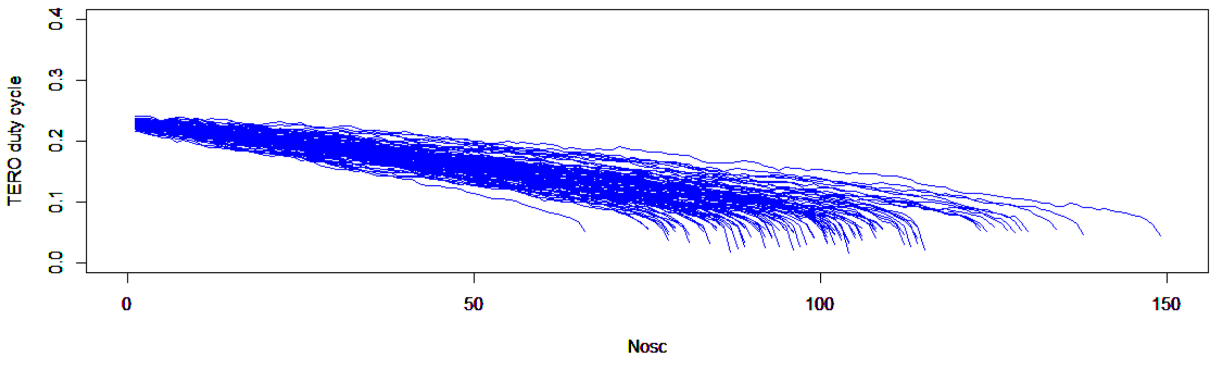


Figure 14: TERO output duty cycle evolution with voltage variation

Figure 15 shows result with both effects cumulated (jitter and voltage variations): The oscillations now range from 60 to more than 300.

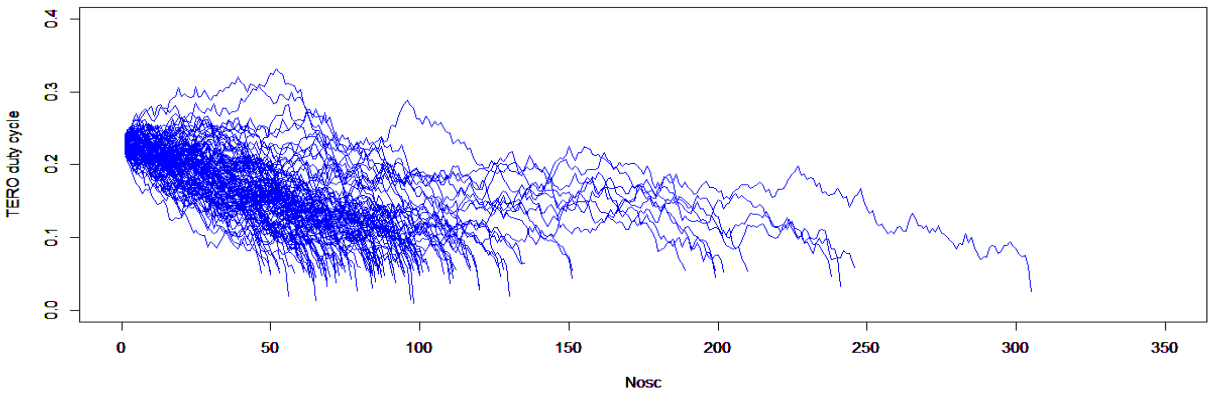


Figure 15: TERO output duty cycle evolution with jitter noise and voltage variation

5.5 Conclusions

A physical-model based entropy demonstration has been proposed for the HECTOR PUF (HECTOR Objective O4). The TERO cell presents very high sensitivity to process variations, which is an extremely good parameter for PUF applications. As downside the TERO cell also seems to be very sensitive to noise and environmental variations. If the latter turns out to be true in real implementations then the PUF responses would be unstable, while stability is critical for PUF usability.

The objective of the next chapter is to check the predictions made by the model.

Chapter 6 TERO-PUF evaluation

6.1 Introduction

The following chapter summarizes the evaluation of the TERO PUF boards and presents the results. This work was carried out amongst others as research on PUFs to find optimal solutions for the PUF of the demonstrator (WP4).

The evaluation was carried out based on multiple criteria: In a first step, the quality of each of the 16 daughter boards and 8 mother boards was analysed regarding the stability of responses. Based on this result, dark bits were identified, selected and removed to improve the bit error rate of the PUF responses. Secondly, the difference and randomness between different boards was analysed. In addition, some further considerations on the correlation of the boards were done and some possible interpretations and conclusions are presented. In 6.2, 6.3, and 6.4 only data that was acquired at room temperature is considered. Finally, PUF behaviour at different temperatures was analysed and based on all these results, a recommendation on the suitability of error correcting codes is given in the last chapter. The statistical analysis was carried out in R and simultaneously in MATLAB, and the raw data, consisting of 5,000 responses of each board, each response 128 bit, is available via the following link: https://hector.technikon.com/03-Work-Packages/WP4/D4.2/D3/00-TEC-TERO-PUF-evaluation-and-recommendations/HECTOR-TERO-PUF-Evaluation.pdf.

6.2 Intra Distance evaluation

6.2.1 Bit Error Rate, Hamming Weight and Probability of Failure

The **intra distance** between two responses of a single PUF is defined as the Hamming distance (see Appendix B, Definition 1) of the two responses. The intra distance of one board was evaluated as follows:

- Calculate a reference response consisting of the most likely bits of the first 50[1] responses.
- Calculate the Hamming distances of the reference response against all 5,000 responses of the PUF and divide it by the number of bits in a response (128 in our case).
- Take the mean of all 5,000 Hamming distances.

The result is the **bit error rate** (BER) resp. HD_{intra} of the board (see Appendix B, Definition 2), which is a good approximation for the real bit error probability. In a perfect world, the bit error rate would be 0, so it is expected to be as small as possible.

Besides the bit error rate, some other values can be calculated and are of interest to get a feeling of the quality of each board:

- The **average Hamming Weight** (HW) of the responses illustrates the relative frequency of 0 and 1 in the responses (see Appendix B, Definition 3). In the PUF context, the following can be said: If we assume that the probability of a bit to be 1 on

[1] 50 is a good choice here as no significant deviations were observed when using e.g. 100 responses instead.

a board is 0:5, then a Hamming weight of approximately 0:5 is most likely. However, a deviation on some cards is not worrying and statistically reasonable.

- The **probability of failure $P_{\mathbf{fail}}$** is the probability that more than three bits flip in a block of 23 bits[2], assuming that the calculated bit error rate is the error probability for one bit (see Appendix B, Definition 5). This parameter is very important for considerations on the use of error correcting codes.

The results for the raw daughter board data are shown in Table 10. Evaluations of the mother boards do not show significant deviations, exact results can be found in Appendix C.

All in all it can be said that the results underline the fact, that the error rate without further processing is quite high and would force the usage of a heavy error correction code, to achieve the intended stability of $P_{\text{fail}} \leq 10^{-4}$. Therefore we try to get rid of the bits which flip most, the so-called dark bits.

6.2.1 Dark bit selection

In order to be able to use a good error correcting code, P_{fail} has to be minimized. This can be done by identifying and removing **dark bits**. Dark bits are bit positions that flip at least in one of the 5,000 responses. Figure 16 illustrates all 5,000 responses of a daughter board, dark bit positions can be seen easily.

Source ID	Bit Error Rate	Hamming Weight	P_{fail}
4	3.53 %	52.05 %	0.80 %
5	7.00 %	53.03 %	7.31 %
6	3.54 %	52.57 %	0.81 %
7	2.84 %	55.67 %	0.37 %
8	5.08 %	43.42 %	2.72 %
9	2.97 %	52.49 %	0.44 %
10	4.29 %	49.82 %	1.56 %
11	6.36 %	42.72 %	5.48 %
17	2.72 %	59.40 %	0.32 %
18	5.14 %	46.51 %	2.82 %
19	5.72 %	44.55 %	3.95 %

[2] For the post processing, the Golay code with the code parameters (23; 12; 7) will be recommended (see 6.6). This code can at most correct 3 error bits in a block of 23 bits.

Source ID	Bit Error Rate	Hamming Weight	P_{fail}
20	4.03 %	43.81 %	1.26 %
21	8.58 %	47.99 %	12.95 %
22	5.96 %	52.16 %	4.49 %
23	4.05 %	43.32 %	1.28 %
24	6.68 %	45.86 %	6.35 %

Table 10: Intra distance results for unprocessed daughter board data

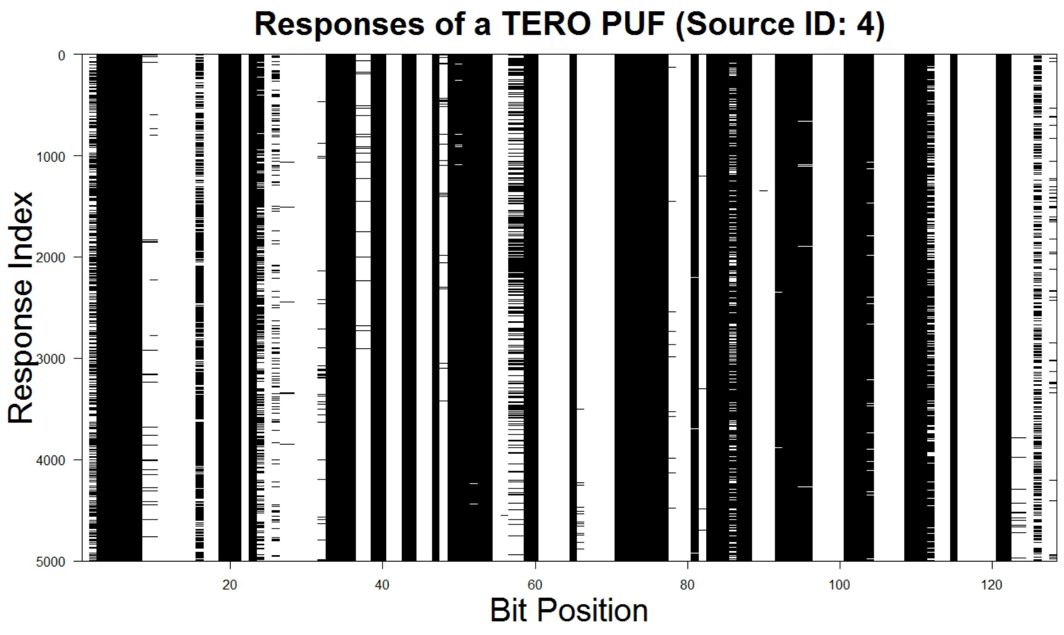


Figure 16: Responses of a TERO PUF with dark bits

It can be seen that only few bit positions flip over all responses, and the other positions are stable. Based on this fact, we choose the following approach to identify dark bits and select those which will be removed[3]:

- Identify all bit positions that are flipped in at least one of the first 50 responses. These are our dark bits.
- From these dark bits select those 13 bits[4] that are flipped most often.
- Remove the selected bits from all responses.

[3] Note that not every dark bit can be removed since we need a proper number of bits. However, not every dark bit must be removed since an error correcting code is used afterwards.

[4] The reason to choose exactly 13 is based on the size of the intended error correction code. If we chose the binary Golay code (23, 12, 7) and run it for 5 loops, we end up with a code word of exactly 115 = 128 - 13 bits.

Now the same calculations as in Table 10 can be repeated for responses without the selected dark bits and compared to the results above. Table 11 shows the results for the 16 daughter boards and Figure 17 illustrates the differences to the unprocessed data. Detailed charts for each board were made (reference: source ID), and detailed results for mother boards can be found in Appendix C again. Also here, mother and daughter board behaviour is similar.

Source ID	Bit Error Rate	Hamming Weight	P_{fail}
4	0.54%	50.57 %	7.09e-06
5	2.70%	53.91 %	3.13e-03
6	0.60 %	52.52 %	1.02e-05
7	0.59 %	49.01 %	9.74e-06
8	1.77 %	41.94 %	6.69e-04
9	0.43 %	53.84 %	2.81e-06
10	0.51 %	51.07 %	5.70e-06
11	2.36 %	40.50 %	1.90e-03
17	0.26 %	59.99 %	3.98e-07
18	0.70 %	45.87 %	1.87e-05
19	1.22 %	44.68 %	1.62e-04
20	0.54 %	43.47 %	6.91e-06
21	4.11 %	47.21 %	1.34e-02
22	1.62 %	52.22 %	4.78e-04
23	0.49 %	43.31 %	4.71e-06
24	2.52 %	45.54 %	2.44e-03

Table 11: Intra distance results for processed daughter board data without dark bits

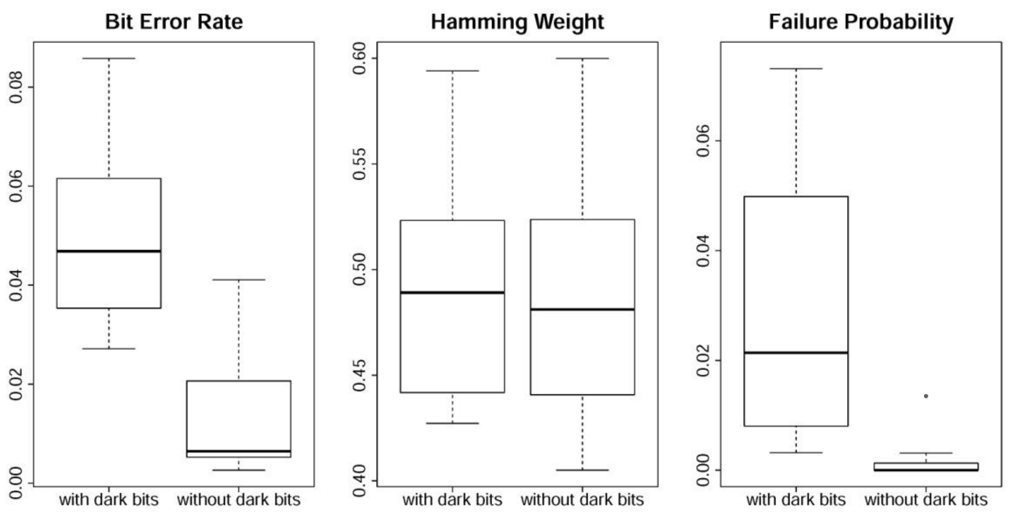


Figure 17: Comparison of unprocessed[5] and processed daughter board data

As one can see, the bit error rate has decreased significantly after removing 13 dark bits. Without this elimination of the bits with worst case behaviour, it would not be possible to achieve the defined stability of 10^{-4} required in deliverable D4.1. With the new failure probabilities, the requirement is mostly met and it is possible to use Golay code for error correction. Clearly the evaluation of the dark bit behaviour for other temperature regions needs to be done to make a definitive statement (see 6.5).

One may wonder if the dark bit elimination would lead to a significantly better result if the dark bits were identified e.g. based on the first 100 responses instead of the first 50. This appeared not the case as for the daughter board responses evaluated at 30°C. Although more dark bits are identified, of course, the 13 bits that are actually removed differ just minimally.

6.3 Inter-device distances

The **inter-device distance** is defined as the Hamming Distance between a reference response of a first PUF board with ID *i* and a sample of responses of another board with ID *j* (see Appendix B, Definition 4). This distance indicates if and to which extent the PUF response is unique in different devices. It is desirable that on average half the bits take on a different value, which corresponds to an inter-device distance close to 50%. If the inter-device distance significantly deviates from 0.5, not only the entropy is influenced but also the resemblance between different PUFs will increase, which negatively impacts uniqueness [5].

Our results were acquired as follows:

- Calculate a reference response consisting of the most likely bits of the first 50 responses of board *i*.
- Calculate the Hamming distance of the reference response against all 5,000 responses of the PUF board *j* and divide it by the number of bits in a response (128 in our case).
- Take the mean of all 5,000 Hamming distances.

[5] Failure probability of PUF with source ID 21 is 0:1295 and therefore cannot be seen in the boxplot, but was considered for the construction of it, of course.

The result is the inter-device distance HD_{inter} from device *i* to device *j*. Detailed results for all 16 PUF daughter boards is given in Figure 18. For results of all eight mother boards see Figure 73 in Appendix C.

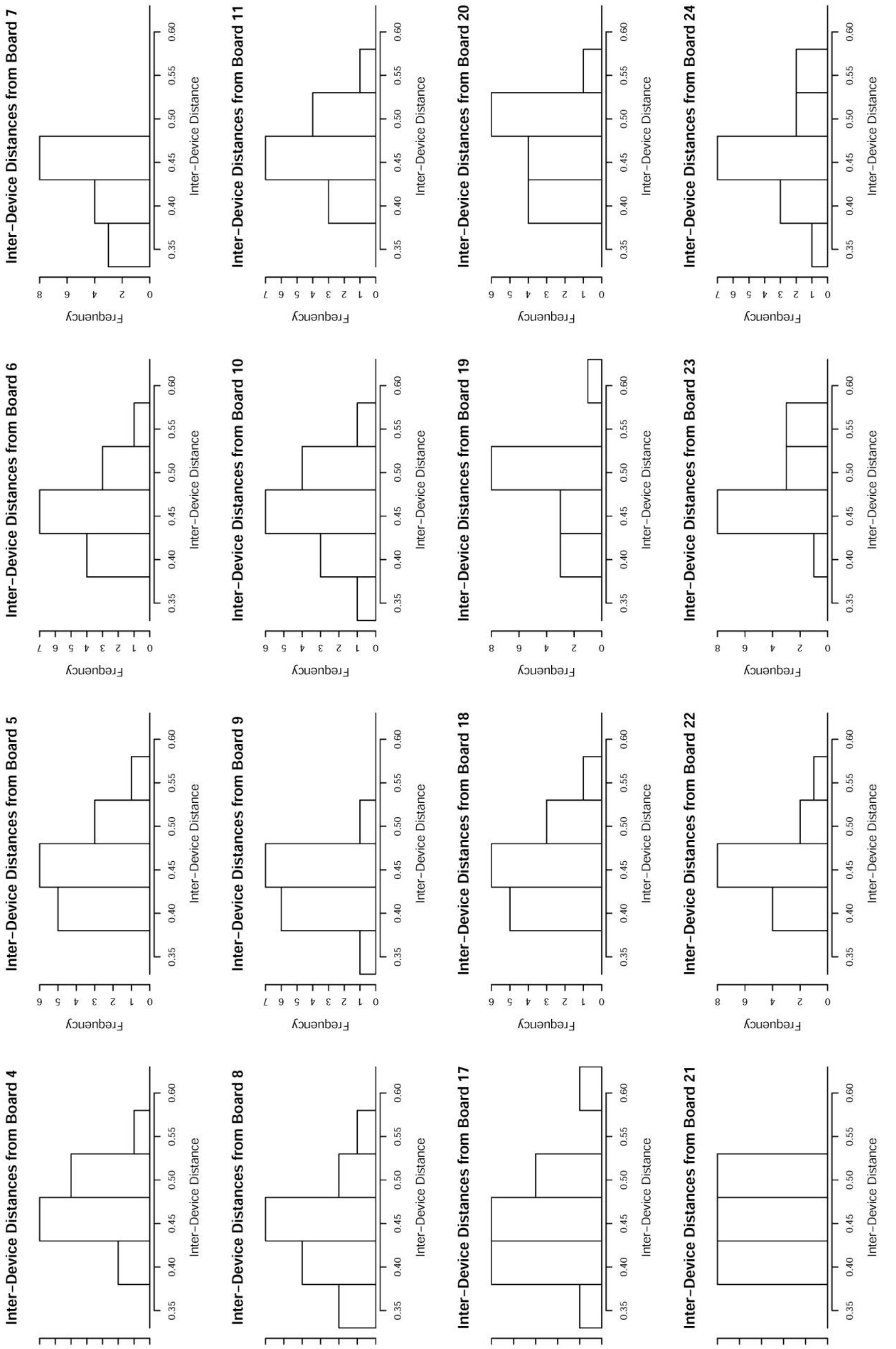

Inter−Device Distances from Board 4
Inter−Device Distances from Board 5
Inter−Device Distances from Board 6
Inter−Device Distances from Board 7
Inter−Device Distances from Board 8
Inter−Device Distances from Board 9
Inter−Device Distances from Board 10
Inter−Device Distances from Board 11
Inter−Device Distances from Board 17
Inter−Device Distances from Board 18
Inter−Device Distances from Board 19
Inter−Device Distances from Board 20
Inter−Device Distances from Board 21
Inter−Device Distances from Board 22
Inter−Device Distances from Board 23
Inter−Device Distances from Board 24
Inter−Device Distance
Frequency

Figure 18: Inter-Device Distances. The histograms show the inter-device distances from the regarding PUF device to all other devices.

6.4 Further considerations on correlation

6.4.1 Bit frequency

A first naive approach to find correlations between daughter boards was to examine the number of cards on which a bit position is 1. Furthermore, also the number of cards on which a bit is dark bit was evaluated. The results are presented in Figure 19.

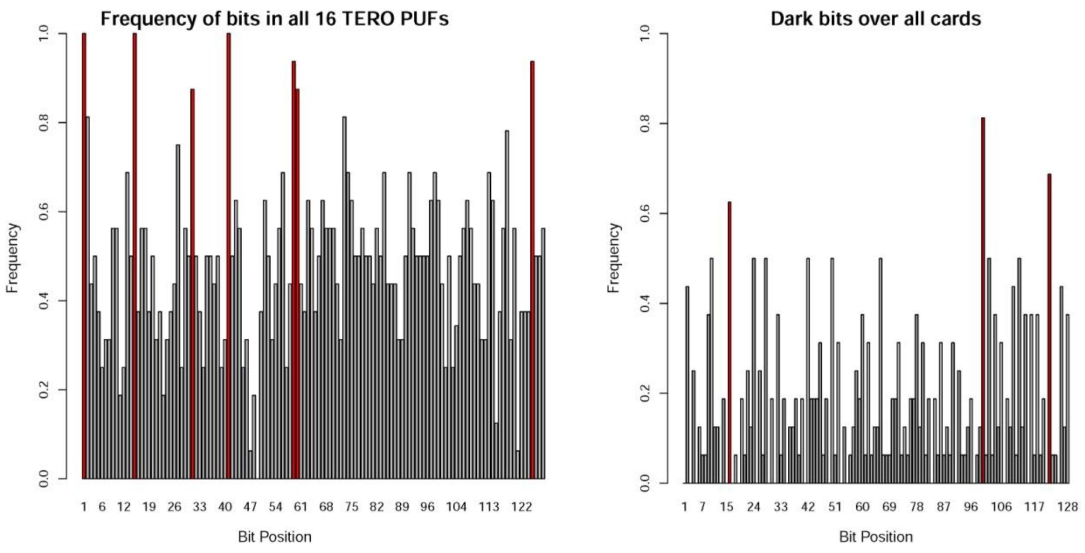


Figure 19: Bit and Dark Bit frequency over all daughter boards

One can see that some bits have the same value for each of the single cards. Those bit positions which are always or nearly always 1 are highlighted in red. One bit is also always 0 (bit number 49), which is also conspicuous. The probability that one bit position is the same for 16 cards is very low: If we assume that the probability for a bit to be 1 on a board is 1/2 , then the probability that a bit is the same (either 0 or 1) on all 16 cards is $2 \cdot 2^{-16} \approx 3 \cdot 10^{-5}$. This implies that the estimated value for the number of constant bits, based on the response length of 128 bits, is $128 \cdot 2^{-15} \approx 0.004$, whereas we got four constant bits. The deviation to the estimated value is quite high, which indicates that there potentially are correlations between the cards. This would reduce the uniqueness property and as such be negative regarding the entropy. If we include the evaluation of the mother boards, still one bit has the same state on all 24 (mother and daughter) boards: Bit number 49 is always 0.

For the dark bit selection the results are not as significant, i.e. they vary over the single cards.

6.4.1 Autocorrelation

In order to analyse the correlation between bits in a PUF, the autocorrelation can be used. The autocorrelation is the correlation of a bit string with a circular shifted copy of itself (see Appendix B, Definition 6). An autocorrelation of ±1 would mean total correlation, 0 would mean no correlation.
A high autocorrelation would reduce the entropy and increase the risk of successful attacks. Figure 20 shows boxplots of the daughter board evaluation results, lags of highest and lowest autocorrelation are labelled. The same figures for mother boards can be found in Figure 25. All in all it can be said that no significant autocorrelation was found.
However, there is a fact which might seem to be a bit conspicuous: The lag at which the autocorrelation is highest is the same for 12 of the 16 daughter boards and for 7 of the 8 mother boards (lag 1 resp. lag 127). Nothing reasonable was found to explain this fact yet.

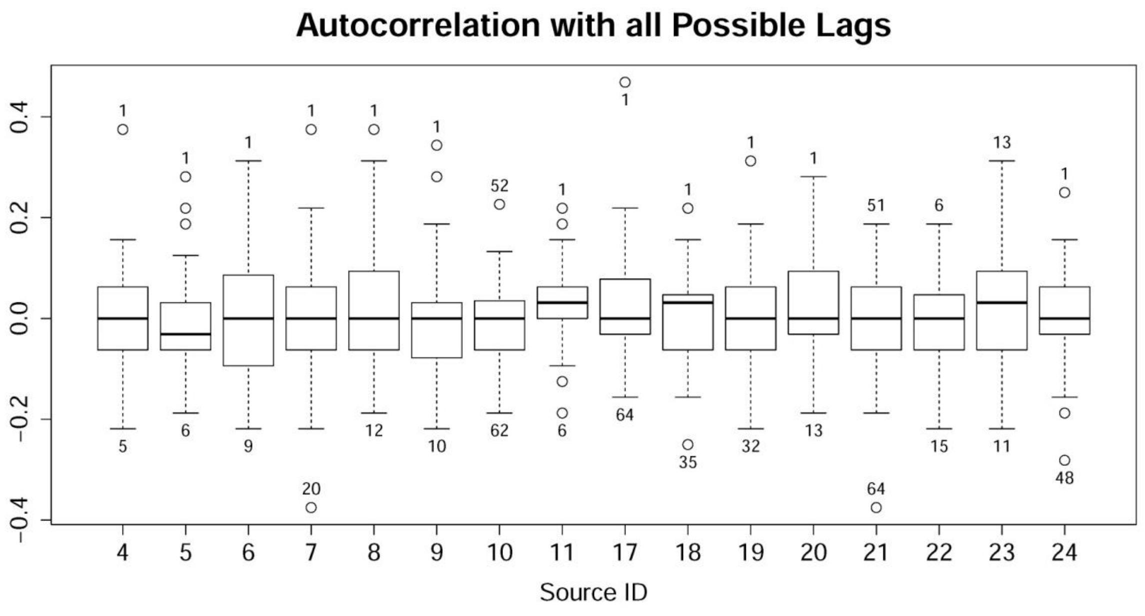


Figure 20: Autocorrelation of daughter boards. Lags of highest and lowest autocorrelation are labelled.

6.5 PUF behaviour at different temperatures

So far we have only considered data which was acquired at room temperature (30 °C). In this chapter we want to analyse the influence of changing temperature on our PUF daughter boards[6]. Changing temperatures often cause higher error rates as many previous evaluations show [6].

6.5.1 Bit error rate

The setup for the data generation at 30°C (at TEC) was slightly different than for -40° C, 25° C and 85° C (at KUL). In order to allow the data generation in the temperature chamber, KUL was using a HDMI cable to connect the daughter boards to the mother board, while TEC plugged the daughter boards directly in the mother board. In addition, the uptime of the boards at KUL was different than for those at TEC. We assume that the uptime of the boards

[6] PUF readouts at different temperatures were only available for daughter boards.

influences the behaviour of the components, especially of the PUF. However, this has not been evaluated in detail.
It can be assumed that this is one of the reasons for the behaviour deviation of the PUF data at 30°C, see Figure 21, left. The temperature behaviour of data sets generated within the same setup describes a linear correlation, which is a fact that is also already known from previous evaluations [4]. So for further considerations, the data evaluated at 25°C is taken as basis.

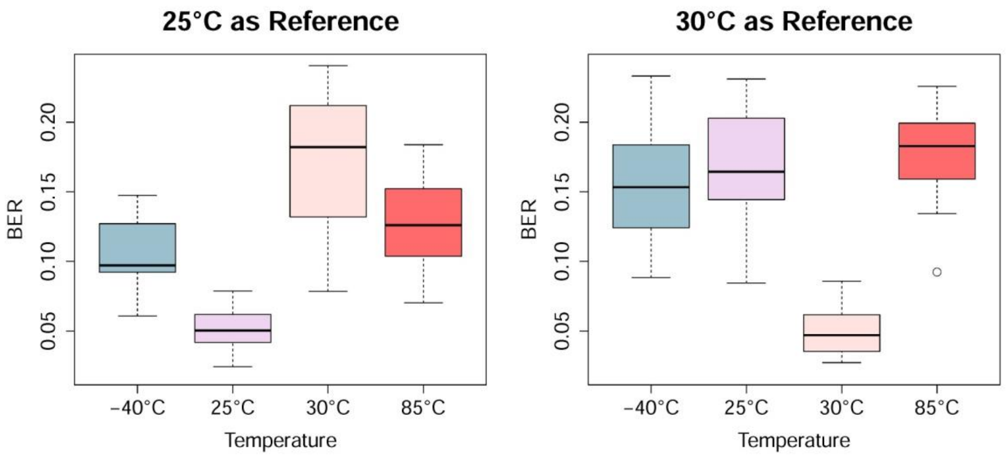


Figure 21: BER of all boards to reference response at 25°C resp. 30°C

6.5.2 Dark bits

It is very important for the pre-processing of the PUF that the dark bits are quite stable in different environments. So a change in temperature should not influence dark bit positions very much. However, the PUFs we use do not show stable dark bit behaviour if they are evaluated at different temperatures. Dark bits vary for the data evaluated at -40° C, 25° C, 30° C and 85° C which is not very satisfying as the dark bit selection presented in 6.2.1 should work properly in every environment. Figure 22 represents the dark bit behaviour of board 4 in an exemplary way: The plot shows how often a bit on board 4 is flipped, based on the four evaluated temperatures above.

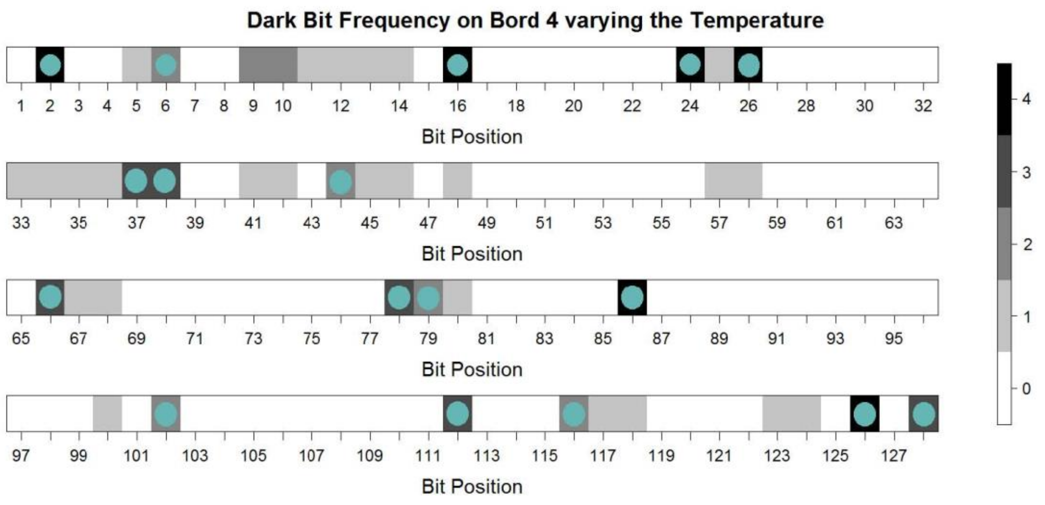


Figure 22: Frequency of being a dark bit at four temperatures on board 4. Blue bits would be thrown away at 25° C as basis.

A frequency of four means that the bit is flipped at every temperature, a frequency of zero means that it is flipped in no reference response. These two cases (black and white in the plot) would be the best as it would mean stability over different temperatures. However, it can be seen that on board four only six bits are flipped at least one time at all four temperatures, which makes the dark bit selection very difficult. Figure 23 summarizes the results. It can be seen that the number of bits that are dark bits at all four temperatures is generally very low. E.g., the board with ID 7 shows extreme behaviour as *no bit* is a common dark bit. Based on this fact, it is not a surprise that the bit error rate and the failure probability are quite high when removing the dark bits identified at 25° C.

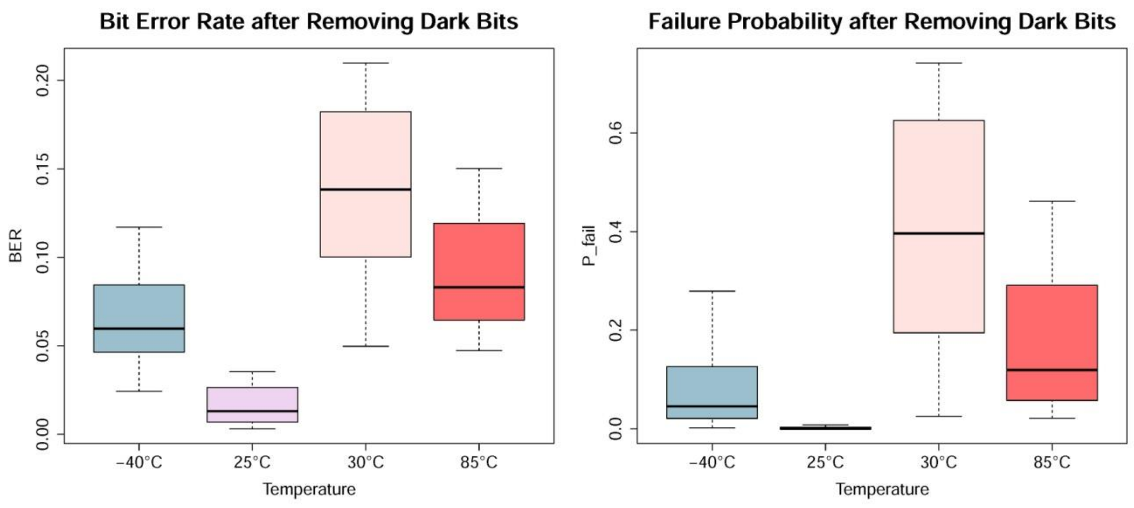


Figure 23: BER and P_{fail} of all daughter boards when removing the same dark bits

6.6 Reasoning and recommendations

A first mention should be made of the fact that the reasoning of this chapter is based on the evaluation of 16 daughter boards and 8 mother boards[7], which means that the global validity for the specific TERO PUF implementation is somewhat limited. However, the extensive empirical evaluations, for different boards at varying environmental conditions, have led to the following summary:

- **Entropy** is lower than expected, since bits show dependencies among different boards. For example, four bit positions show completely the same state over 16 different (daughter) boards at room temperature, see 6.4.1. Evaluations for other temperatures on the daughter boards and for the mother boards at room temperature show additional interdependencies, e.g. there is still one bit position (49) that shows the same state over all temperatures on every daughter board as well as on mother boards.
- **Error Rate:** Performing the reconstruction in changing environmental conditions results in a higher error rate (Figure 22). An increase of the error rate leads to a demand of a more complex code. A complex code comes often with a low code rate (k=n) which on the one hand results in a higher demand of resource bits which we definitely not have, and on the other hand demands more memory on the

[7] Mother board data from UJM and STR missing

demonstrator (size of LUT, size of helper data,...). Having these restrictions in mind, the variety of codes becomes very limited.
- **Dark Bits Behaviour:** Dark bits vary for different environmental conditions which makes it hard to define a valid dark bit vector that can be applied in the field. One of the more complex problems for the applicability of the TERO PUF can be described as disguised dark bits. Certain bits are completely inverted when reconstructing at different temperatures and/or different set-ups. This is underlined by the bad error rate/failure rate results in Figure 23.

Based on the summary above, the TERO PUF as used in demonstrator 2 and 3 may only be seen as an additional security layer (neglecting the fact that the weakest link counts...). The following statements need to be reported:
- ➢ The number of available bits (128) is too low to extract a key with the claimed security level.
- ➢ The available bits show limited PUF behaviour.
- ➢ The quality (error behaviour, bias, environmental conditions) of the responses cannot ensure the adequate usage of the PUF as a security anchor.

For this reasons, the PUF can only be used in a very lean and reduced version. We recommend the following set-up/settings:
- **Reference Response:** We recommend using the most likely response over the first 50 read-outs at room temperature during the enrolment phase.
- **Enrollment/Reconstruction Procedure:** Due to the instability of the PUF behaviour (temperature, setup), we would recommend an enrollment phase that is temporally close and also similar from the setup conditions to the reconstruction phase. This is applicable for a demonstration, but is highly impractical and does not represent the reality when applied in the field. At the first glance, one might guess that a possible work-around would be to take multiple responses to generate the most likely bit string also during the reconstruction phase. Evaluations showed slight improvements when conducting at same environmental conditions. Over varying conditions, there is no significant improvement related to the failure rate identifiable (see Appendix D).
- **Dark Bit Selection:** The idea of bit selection schemes is to discard less reliable bits before the PUF response will be further processed. This will decrease the bit error rate. We recommend proceeding with the approach described in Subchapter 6.2.1, i.e. deletion of the worst 13 dark bits.
- **Helper Data Scheme:** For the HECTOR demonstrator, we would recommend using an alternative to the commonly used code-offset construction: the Kang's scheme (see Appendix E). Compared to the conventional code-offset construction, less helper data needs to be stored.
- **Error Correction Code:** We would recommend using a Golay code (23, 12, 7) running in five loops. This code shows the best results when considering the costs versus the benefits.
    - See Figure 3.8 in D4.1 for the behaviour of the failure probability (the Golay code is more stable over the range of error rates).
    - Because of the implementation in several loops, the Golay code is scalable. In case that more bits need to be thrown away, we may reduce the number of loops to four. Be aware, that this can only happen at the expense of entropy.

There are still further hypothetical improvements feasible, even though they would not be very reasonable. Further tweaks would lead to a system that is based on artificial adjustments which have not too much in common with the classical and standardized way of applying a PUF-based security anchor. Taking the above mentioned recommendations and improvement measures into account will allow the utilization of the TERO PUF for demonstration purposes.

6.7 Conclusions

The robustness of HECTOR PUFs has been evaluated (HECTOR Objective O5). Unfortunately the level of robustness to environmental variations measured on FPGA implementations is low. The level of robustness achieved on ASIC will only be testable after project completion due to repeated difficulties and delays with the fabrication of HECTOR ASICs. However, the physical-modelling and analysis performed within the project allowed to explain and in hindsight predict the excessive sensitivity of the selected PUF principles against voltage or temperature variations. This excessive sensitivity is a somewhat fundamental limitation of the selected PUF principle and will likely be very complex and costly to prevent. Although this lack of robustness is disappointing, the fact that our models allow to predict it is a testimony to the pertinence and improved security guarantees brought by the model-based approach we have been proposing.

Chapter 7 Side channel analysis testing

7.1 Introduction

In this chapter we describe the robustness of HECTOR TRNG designs against side channel analysis attacks (passive attacks). The analysis aims to find correlation between the processed data and any externally observable physical signal that is caused by the handling of the random numbers inside the FPGA. The side channel leakage of online statistical tests of both DC-TRNG and PLL-TRNG was analyzed in experiments described in the following sections.

7.2 Side Channel Analysis on on-line test of DC-TRNG (Spartan-6)

7.2.1 Sample preparation

For the penetration testing using side channel techniques no physical sample preparation was required. The test sample, a Spartan-6FPGS, has a dedicated trigger output signal implemented that indicates the start of a new round of random bytes generation. Such trigger signal is only available in research devices and not in practical implementations. Therefore a real attacker will have more difficulty in breaking the device than the results of this test will show.

7.2.2 Test summary

PT1.V.2 Side-channel Analysis on Online Statistical Tests (DC-TRNG)	
Test Goal	Determine if the TOE resists a side-channel attack on the online statistical tests that are applied on raw random data.
Test Description	During the random number generation, intermediate values are entered in an additional module that performs statistical tests in order to establish that the random data does not have intolerable weakness. A side-channel test is performed on the statistical tests.
Vulnerability Description	The TOE might leak information on random data being output. Although the entire attack-path is found to be infeasible, the partial attack of performing measurements on a freely accessible variant of the TOE is considered.
Attack Method	Side-channel analysis Correlation analysis as first step, a template attack as possible second step. Between 100,000 and 1,000,000 traces depending on practical feasibility.
Physical Target	The power signal and the EM radiation on the surface of the FPGA.
Target Command	The target command is the function that requests raw random data in binary format.
Expected Result	It is expected that the TOE is resistant against the side-channel attack on the raw random data.

7.2.3 Detailed test method

In the generation of random data, for each sequence of 512 raw random bits b_i, two types of statistics are calculated.

$$N_{111} = \sum_{i=1}^{510} \min(b_i\,, b_{i+1}, b_{i+2})$$

$$C_1 = \sum_{i=1}^{511} b_i \oplus b_{i+1}$$

All these b_i are used as random bits, so all bits are interesting to template. In the end, the values of N_{111} and C_1 are compared against some threshold, however the values of N_{111} and C_1 at that point do not contain much information on the individual bits.

Note that the TEST2 on N_{111} has a relatively high false alarm probability of approximately 1%. By retaking an initial measurement of 200 of 300 traces, it should be possible to find a trace corresponding to an alarm being raised. This can help in determining the timing of the online statistical tests being completed. TEST3 on C_1 has a negligible error probability of one over a million.

Command		Format	Response
Initialize device			-
Get raw random data			XX …… XX
	Obtain random data		XX …… XX

	Obtain error status		AA BB CC DD EE FF
			The 64 bit state contains:
			S(63:48) 16-bit version number
			S(47:4) reserved for later use
			S(3) error E3 (TEST3 alarm)
			S(2) error E2 (TEST2 alarm)
			S(1) error E1 (TEST1 alarm)
			S(0) error E0 (TEST0 alarm)

Table 12: Command sequence of the performed attack

More information on the online tests can be found in section 4 of [22].

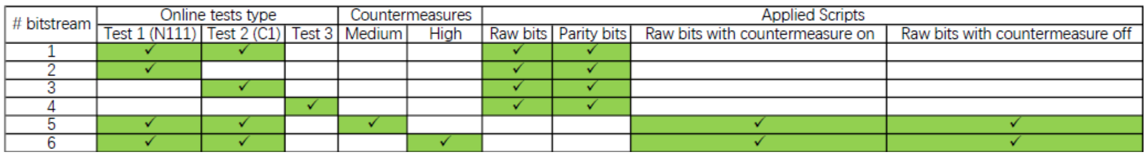
7.2.4 Test details and test results

The following tables show the details of the performed experiments and the commands that have been used. Detailed descriptions about the measurement set-up and related components can be found in Appendix A.

Test details	
Hardware	SCA5
Software	Matrix v3.6.1 Sideways v3.22 Template Attack v1.5
Equipment parameters	Vcc = 5 V Sampling rate = 5 GS/s or 10 GS/s
Evaluation lab	Brightsight bv

Table 13: Test details

The goal of this test was to identify the online statistical tests TEST2 and TEST3 in the EM traces and to see if they are resistant against template attacks. Six different configurations were proposed. The details of the different configurations can be seen in Table 14.



# bitstream	Online tests type			Countermeasures		Applied Scripts			
	Test 1 (N111)	Test 2 (C1)	Test 3	Medium	High	Raw bits	Parity bits	Raw bits with countermeasure on	Raw bits with countermeasure off
1	✓	✓				✓	✓		
2	✓					✓	✓		
3		✓				✓	✓		
4			✓			✓	✓		
5	✓	✓		✓				✓	✓
6	✓	✓			✓			✓	✓

Table 14: Overview of the provided configurations

It was decided to investigate configuration #5 and #6, because they perform the statistical tests that are targeted by this evaluation (called Test 1 (corresponds to TEST2) and Test 2 (corresponds to TEST3) in Table 14). These additional configurations provide the option to enable or disable medium protection level countermeasures (configuration #5) and high protection level countermeasures (configuration #6). These countermeasures can be activated or deactivated by setting the corresponding option in the delivered script, which is used to communicate with the TOE. To have a fairly realistic situation the countermeasures are always on and thus have their effect on the signals. This signal can be seen in Figure 24.

In both configurations a trigger signal was provided, which goes high before the statistical tests take place. It has to be mentioned that a real attacker would not have this trigger signal. This type of testing is referred to as ‘worst case testing’. This methodology is common practice during evaluations to characterize the core behavior of the evaluation target. A real application will be more difficult to attack.

First, configuration #6 was loaded on the TOE and a surface scan of the chip was performed. An interesting pattern could be observed right after the trigger signal rises, as it can be seen in Figure 24. It has to be mentioned that this signal appears on the whole surface of the chip and that it can still be recognized a couple of centimeters away from the TOE.

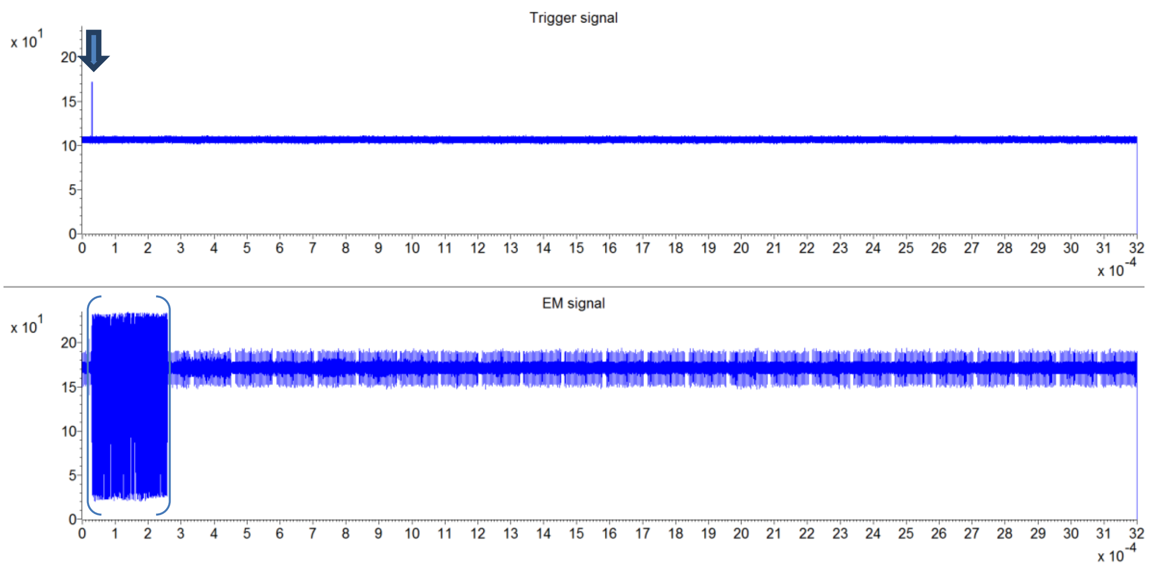


Figure 24: Trigger signal and an example of the interesting pattern (marked by the brackets) observed afterwards. The arrow indicates the start of the statistical test procedure.

A final position for the measurement was chosen depending of the signal strength and shape. The final coil position can be seen in Figure 25.

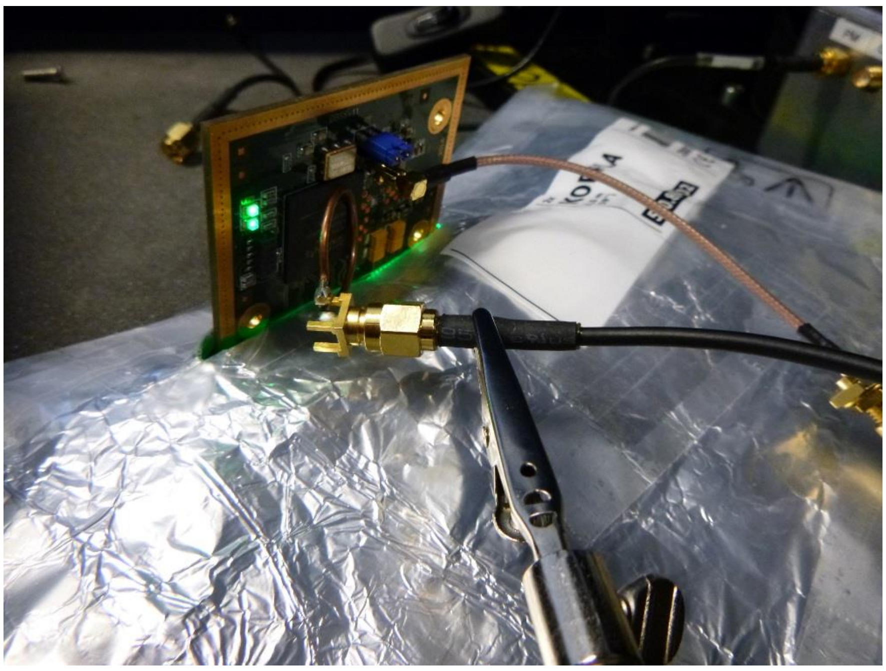


Figure 25: Final coil position for the measurement of configuration #6.

A set of 8,000 traces at a sampling rate of 10 GS/s was recorded. The raw random data as well as the statistical test data was stored with the traces. Figure 27 shows a recorded trace. The error status state returned by this configuration is always set to all 0. So it was not possible to identify an EM trace where TEST2 failed, by simply looking at the output log files, which are created for each run of random number generation and which contain the error status state. Therefore, it was decided to perform correlation analysis with the raw random bits and the data used for the generation of the numbers N_{111} and C_1, in order to see if the device leaks and to identify the point in the trace, where the statistical tests take place.

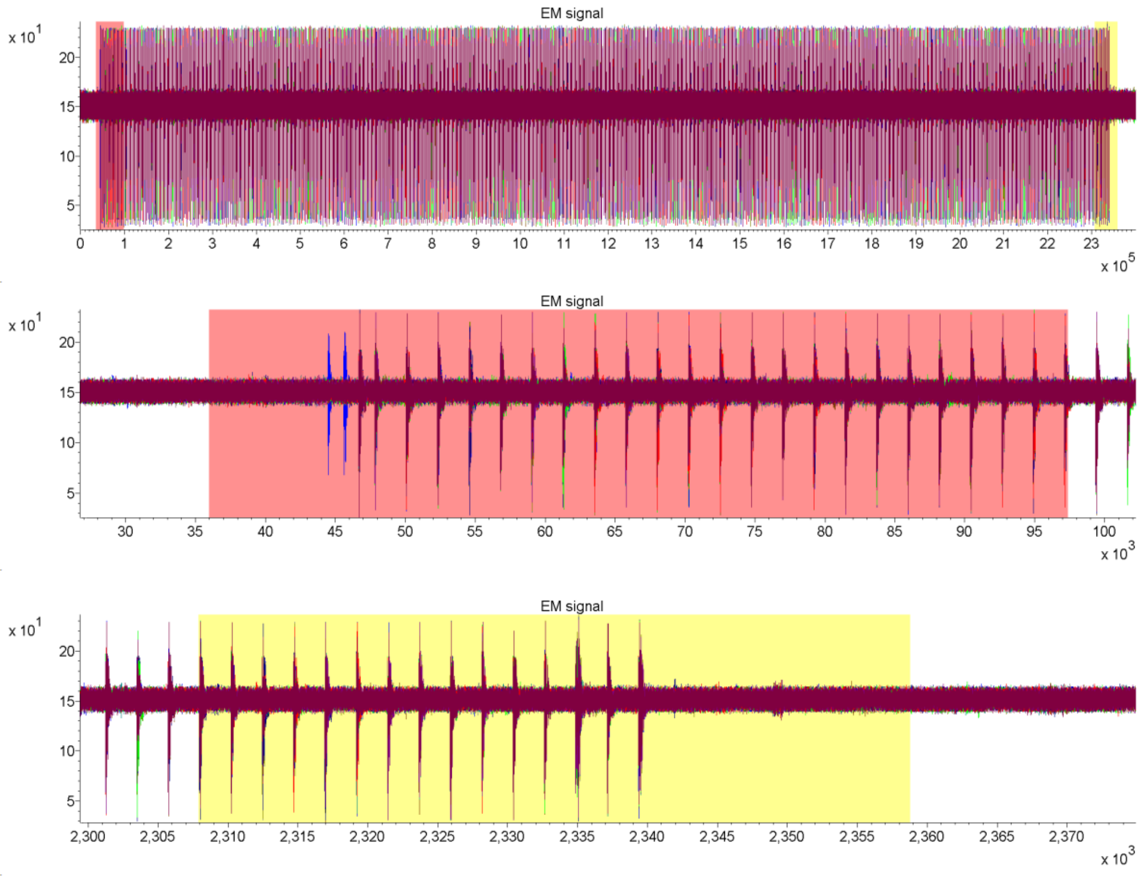


Figure 26: Ten overlaying traces (top) and the zoomed in views of the intervals marked by red and yellow. No additional alignment was necessary.

Due to the accuracy of the trigger no additional alignment was necessary. This can be seen in Figure 26. Correlation analysis with N_{111}, C_1 and the raw random data showed significant correlation peaks. These results can also be seen in Figure 27.

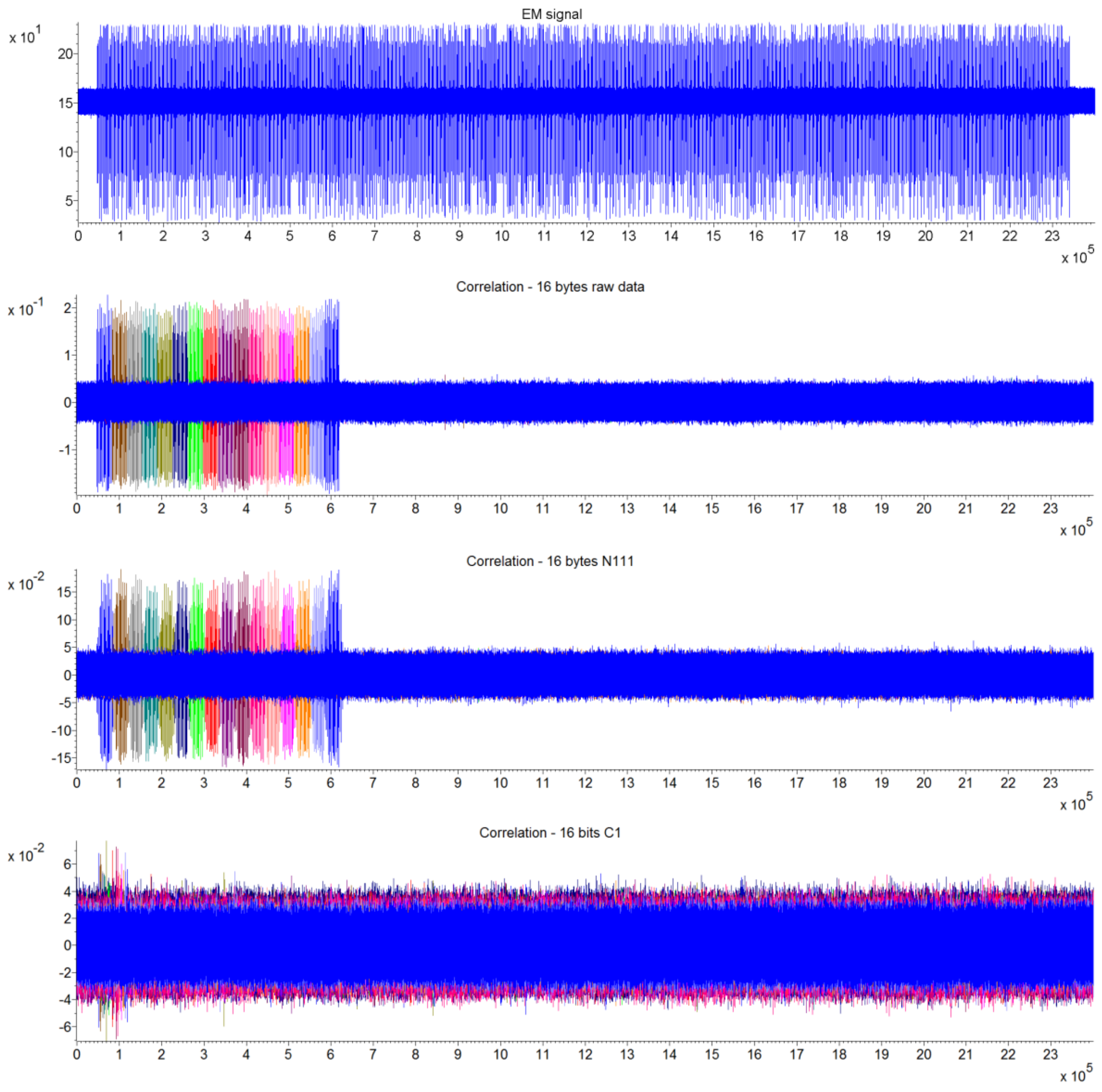


Figure 27: An example EM trace of the first measured set can be seen in the top. The three graphs in the bottom show the correlation results with the raw random data and the data of the statistical tests.

Due to the shape and the order of the correlation results peaks, it seems likely that the whole pattern, marked by the brackets in Figure 24, does not belong to the statistical tests, but to some internal loading/processing of the raw random bits. This is better visible on a bitwise correlation calculation with the raw random and the data used to calculate N_{111}, which can be seen in Figure 28. Every bit of the data used for calculating N_{111} leaks at six different positions corresponding to the leakage of the three bits that are the input of the function.

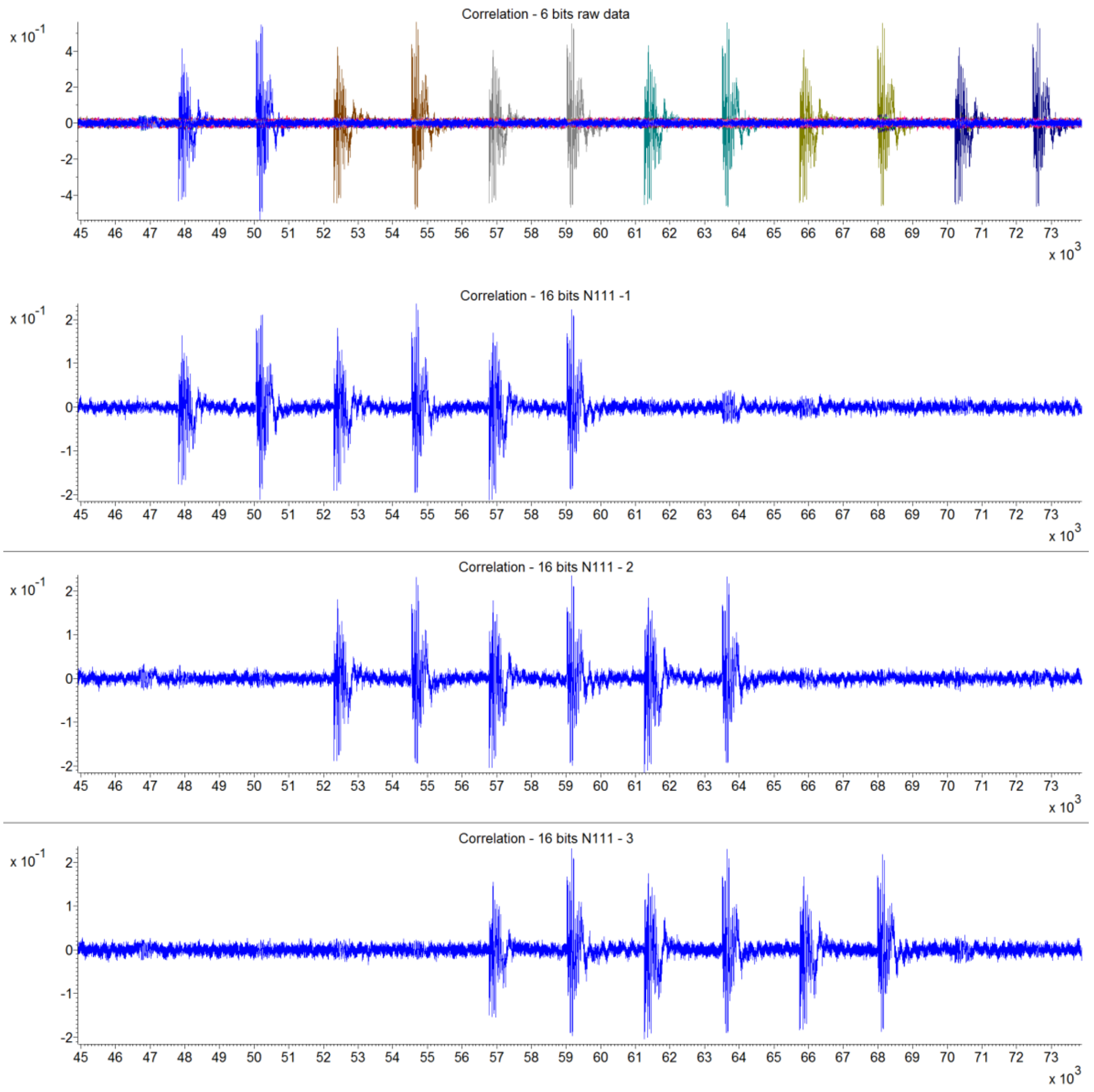


Figure 28: The trace in the top shows correlation results of the first 6 bits of the raw random data. The three traces in the bottom show the correlation results of the first three bits of the data used to calculate N_{111}.

A new set of 15,000 traces at a sampling rate of 10 GS/s was recorded using the maximum number of sampling points supported by the setup in order to investigate the EM signal further and to identify the statistical tests in the traces. It is assumed that the tests happen after the pattern that was identified in Figure 24. A new trace can be seen in Figure 29.

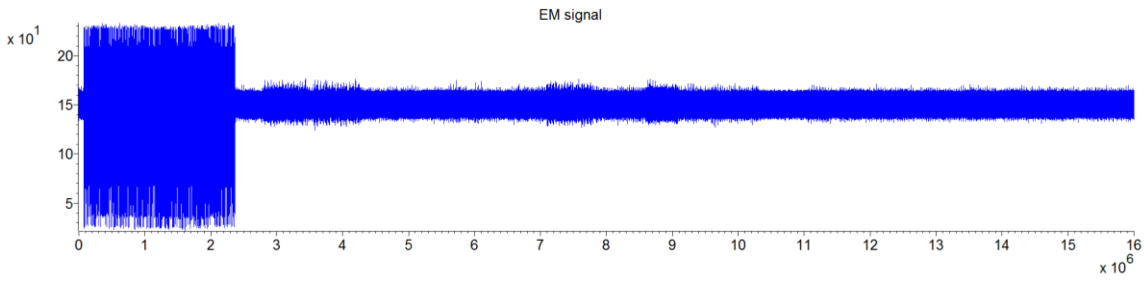


Figure 29: Example trace of the second measured set.

Again correlation analysis with all the stored values was performed, but it showed no additional peaks. It was decided to convert the traces to the frequency domain and to perform an additional correlation analysis. Therefore, the traces were partitioned and a FFT was calculated over each segment, using a 50% overlapping of the segments and a segment length of 71,262. So it is possible to overcome misalignment issues in the part of the trace after the pattern marked by the brackets in Figure 24, because in this part of the trace no visible pattern could be observed. One converted trace can be seen in Figure 30.

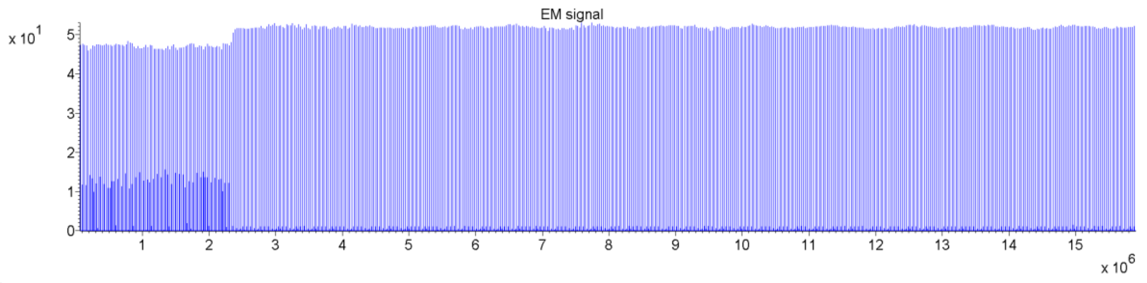


Figure 30: Trace of the second measurement converted to the frequency domain.

Again the correlation analysis of all stored data showed no new results. As an example the results of the N_{111} data can be seen in Figure 31.

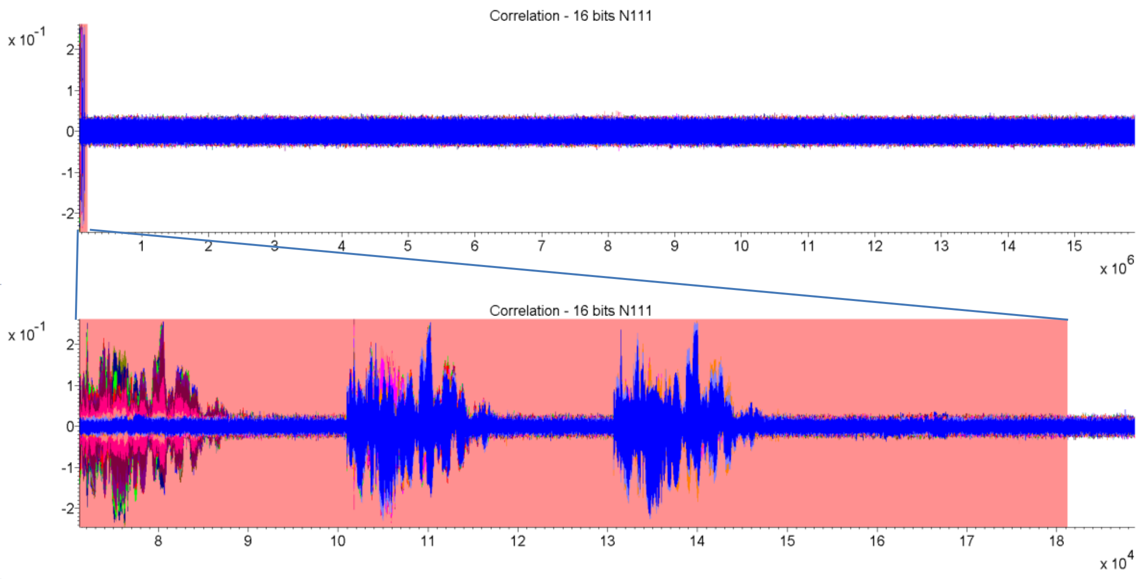


Figure 31: Correlation results for the first 16 bits of the N_{111} data on the converted traces.

Configuration #5 was loaded on the device to see if it is possible to identify the statistical tests now, since this configuration has a lower protection level than configuration #6. Again a surface scan was performed resulting in a different final coil position, which can be seen in Figure 32.

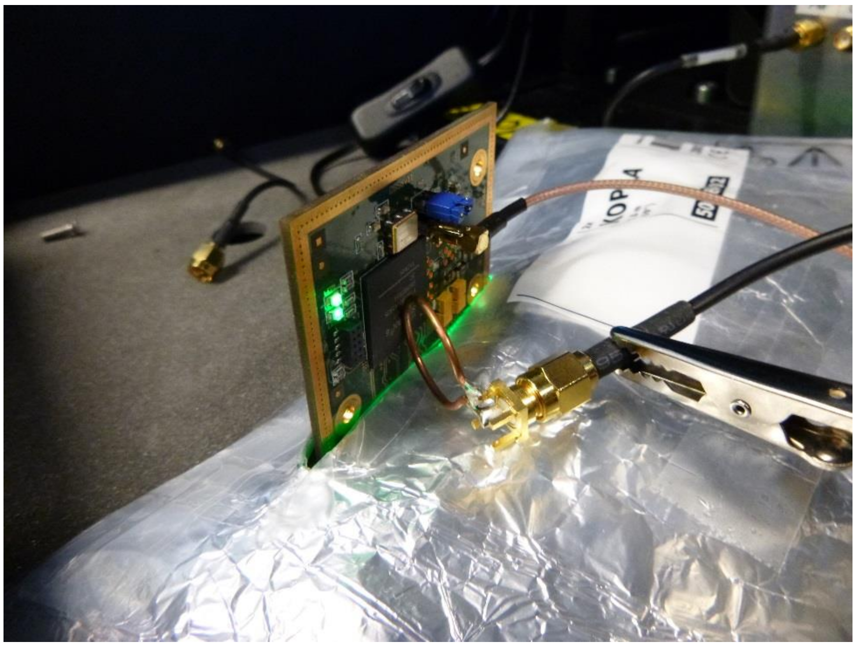


Figure 32: Final coil position for configuration 5.

A set of 10,000 traces at a sampling rate of 5 GS/s was recorded. The sampling rate was lowered in order to record a larger interval. A new trace can be seen in Figure 33.

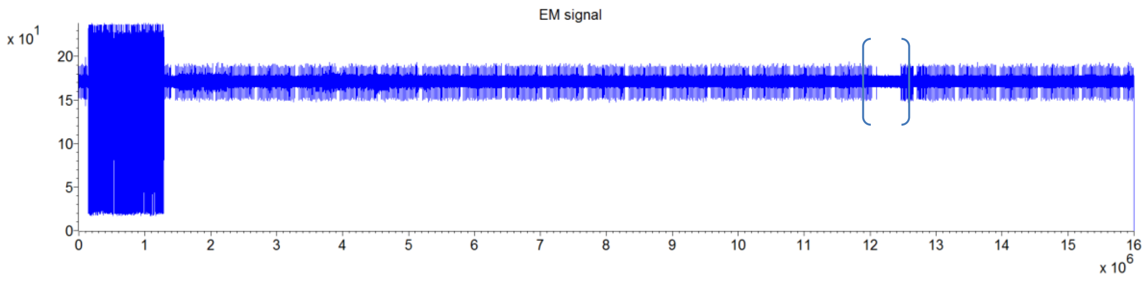


Figure 33: An example trace of the third measured set with configuration 5. The brackets mark a gap in the trace, which appears 13 times in the first 50 traces.

Again correlation analysis showed no new results. Sometimes it was possible to see a small gap in the traces, which may identify that one of the statistical tests fail, even if it appears in more than 1% of the traces. This gap is marked by the brackets in Figure 33. It seems that this gap is not at a constant position in the traces when it appears and it cannot be used to perform a different alignment. The traces were converted to the frequency domain using 50% overlapping and a larger segment length then before of 106,900, but correlation analysis showed no new results.

Even if the exact location of the statistical tests could not be identified, leakage occurred. To see if this leakage can be exploited, the traces of the first two measurements where merged together in order to perform a template attack with a set of 22,000 traces. This implies that configuration #6 is used for the template attack, which is supposed to have a higher level of countermeasures enabled. The results of the attack can be seen in Figure 34. They lead to a maximum success rate of 0.8555. This corresponds to the recovery of 219 out of 256 classes. Detailed information on the metrics for measuring the success of template attacks and on interpretation of the results can be found in Appendix A.

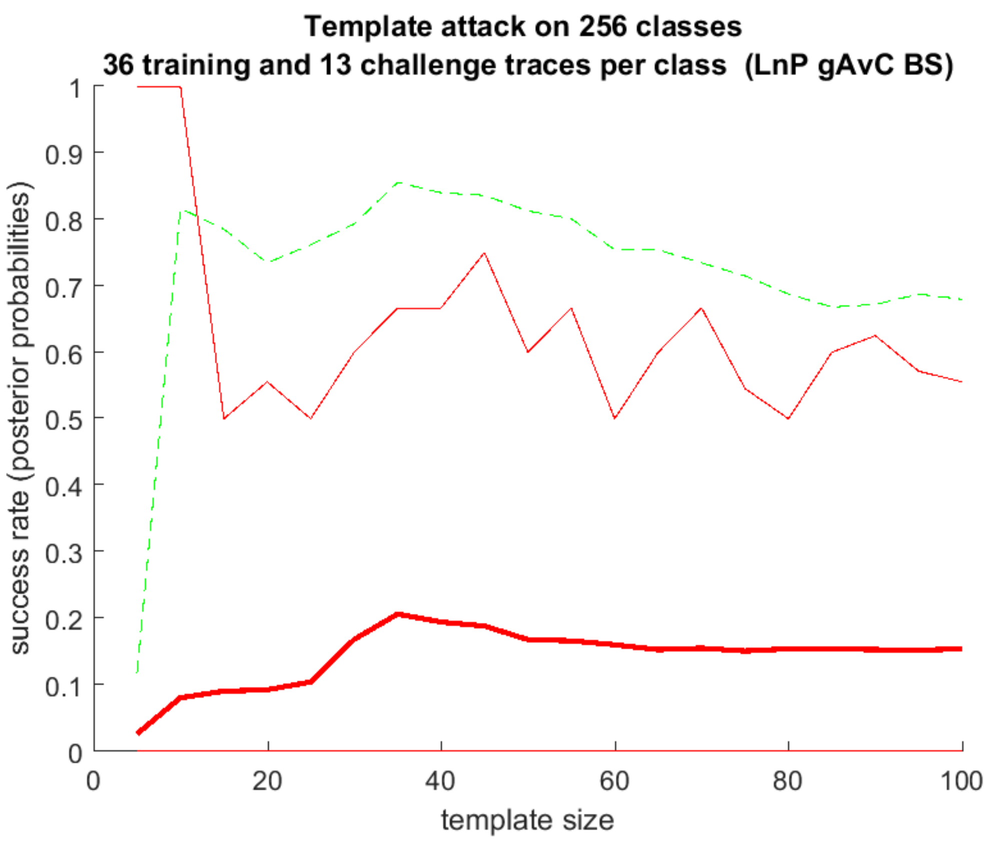


Figure 34: Success rate of the template attack on EM traces, on 256 byte values on the second generated byte as a function of the template size (minimum distance between points of interest equals two and a single covariance matrix was used).

Even if the attack only reaches a combined success rate of 0.8555 when the template size is 35, this attack can be considered successful. This is because the remaining brute force effort reduces to 2^{38}. It also has to be mentioned that the attack was performed with different amounts of training and challenge traces (25 training / 1 challenge, 40 training / 5 challenge and 36 training / 13 challenge). It could be observed that with a higher amount of challenge traces the success rate increased. So it is likely that the measurement of a larger set would lead to a success rate higher than 0.8555. Because the acquisition time on this TOE was very long and the performed attack was already successful, it was decided that further experiments using a larger number of traces were not necessary.

7.2.5 Test conclusion

The goal of this test was to identify the online statistical tests in the EM radiation of a dedicated FPGA with the DC-TRNG implementation (the TOE) and to determine if they are resistant against side-channel analysis.

The tests are done on a specially prepared sample using trigger outputs, which is referred to as ‘worst-case’ testing. In a practical situation however these signals are not present, which makes exploitation in a real application infeasible.

Two different configurations with a different level of protection were loaded on the Spartan-6 FPGA. Three different set of traces were acquired, using different sampling rates. Some traces were converted to the frequency domain. It was not possible to locate the online statistical tests in the EM traces, but leakage of the raw random data was found and could be exploited by a template attack. This is only possible on dedicated test samples. Similar side channel analysis cannot be exploited on real-life products.

7.3 Side Channel Analysis on Online Entropy Test of the PLL-TRNG

7.3.1 Test summary

Side Channel Analysis on Online Entropy Test (PLL-TRNG)	
Test Goal	Determine if the TOE resists a side-channel attack on the online entropy test that is applied on raw random data.
Test Description	During the random number generation, intermediate values are entered in an additional module that performs statistics in order to establish entropy of the intermediate data. A side-channel test is performed on the statistical tests.
Vulnerability Description	The TOE might leak information on its random data output due to processing of the random values by the statistical tests. Although the entire attack-path is found to be infeasible for Demonstrator 1 in WP4, a partial attack of performing measurements on a freely accessible variant of the TOE is considered as part of the robustness tests within WP2.
Attack Method	Side-channel analysis Correlation analysis as first step, a template attack as possible second step. Use between 100.000 and 1M traces, depending on practical feasibility.
Physical Target	1) The power consumption of the FPGA 2) The EM radiation on the surface of the FPGA
Target Command	The commands that request internal counter values and that request raw random data in binary format.
Expected Result	It is expected that the TOE is resistant against the side-channel attack on the raw random data.

7.3.1 Test details and test results

The following tables show the details of the performed experiments and the commands that have been used.

Test details	
Test setup	SCA5
Test software	Matrix v3.6.1 Sideways v3.21/22
Measurement parameters	Sampling rate = 5 GS/s
Evaluation lab	Brightsight bv

Table 15: Test details

Command	
INIT DAUGHTER HDMI.TCL	
GET_CNT	
	Get counter values
	Get error status
GET_RAW (only for SPA)	
	Get raw random data
	Get error status

Table 16: Command sequence of the performed attack

The goal of this test is to investigate the resistance of the Online Entropy Test of the TOE against side-channel analysis. Therefore, two different intermediate values in the generation process of the random number generator are targeted.

The first target is the counter value (8 bits), which counts random events occurring during 185 periods of the clock signal and which is processed as byte value. This process is repeated 4,096 times. The last bit of every counter value is used to compose the raw random bytes. So a run consists of 185 times 4,096 = 32,768 clock cycles and will provide 512 bytes of raw random data. These raw random output bytes are the second intermediate value targeted by this test. In a real device the TOE will produce several MB of raw random data. To speed up the acquisition and the analysis it was decided to only generate 4,096 counter values (512 raw random bytes) per command execution and store the first 256 of them with the corresponding EM traces. The commands used in this investigation (GET_CNT and GET_RAW) perform the same steps on the TOE but giving back different intermediate values (counter value and raw random data) for further analysis.

As a result of the experiences of previous tests on the same FPGA it was decided to focus only on the EM signal. This EM signal was additionally filtered with a 48 MHz low pass filter, because previous experiments showed better results on the filtered signal. A picture of the TOE and the final coil position can be seen in Figure 35. This position was found by searching for the highest signal strength.

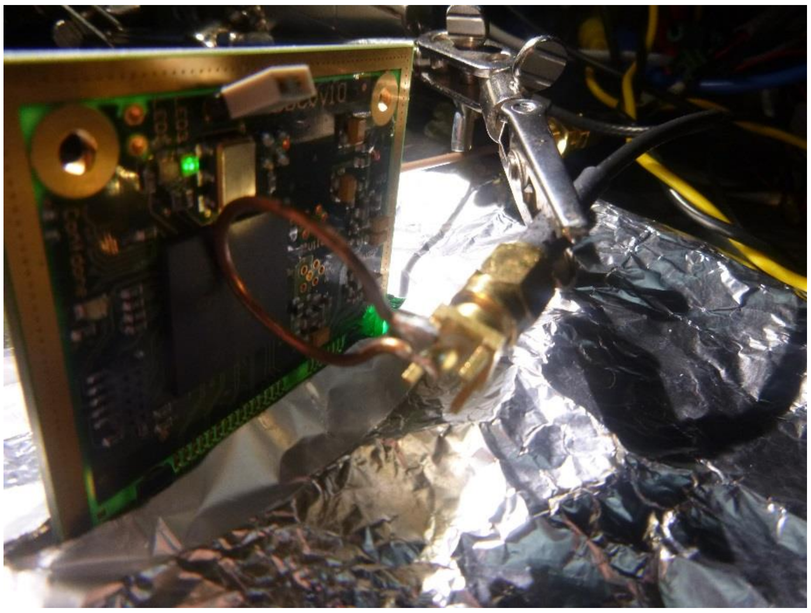


Figure 35: Picture of the TOE and the final coil position.

In a first visual inspection of the traces no pattern could be observed that can be linked to the actual random number generation. It is known that the trigger signal rises just before the new run of random number generation starts that is linked to the received intermediate data. So this is the only indication, from which point on traces should be acquired. Figure 36 shows the trigger signal and the raw traces.

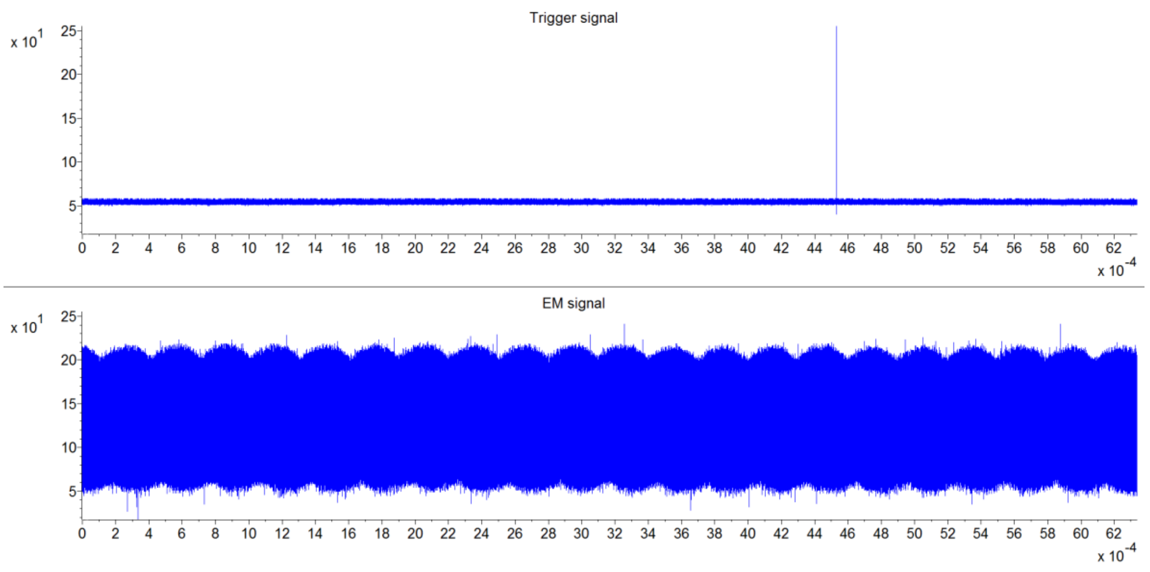


Figure 36: The raw EM signal and the trigger signal. No variation of the EM signal can be seen when the trigger is generated.

7.3.1.1 Correlation analysis of the counter value

For the correlation analysis a set of 20,000 EM traces of the GET_CNT command was recorded at a sampling rate of 5 GS/s. The GET_CNT command executes a full run of random data generation and returns the counter values that are used in this run. Figure 37 shows a raw and filtered EM trace.

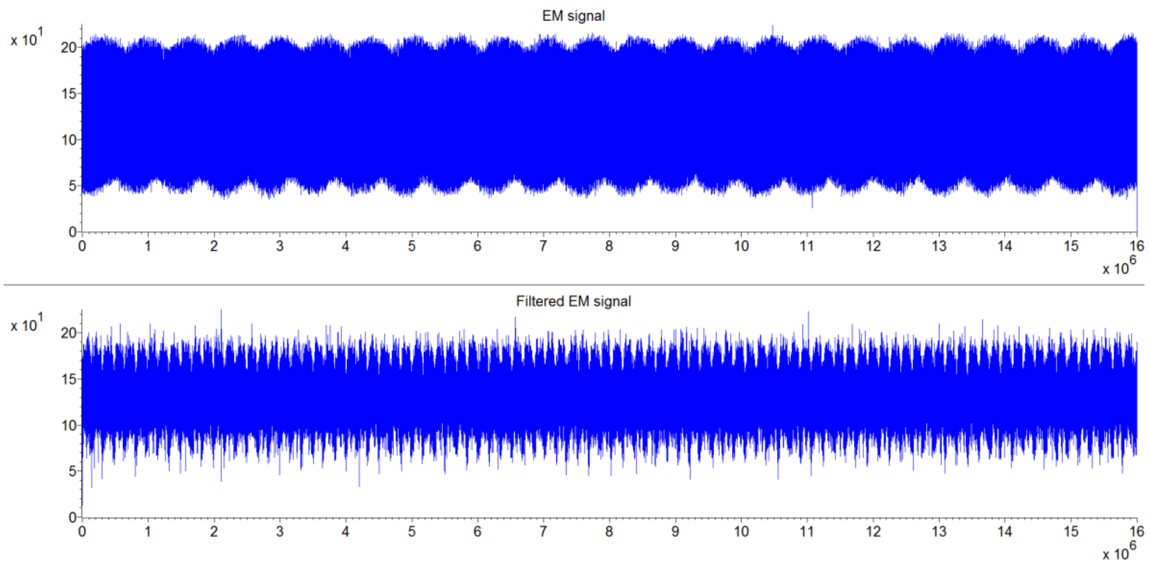


Figure 37: Raw and filtered EM trace of the GET_CNT measurement

When overlaying several traces it can be seen that the traces are misaligned, even though they are aligned relative to the trigger signal. Figure 38 shows five overlaid traces.

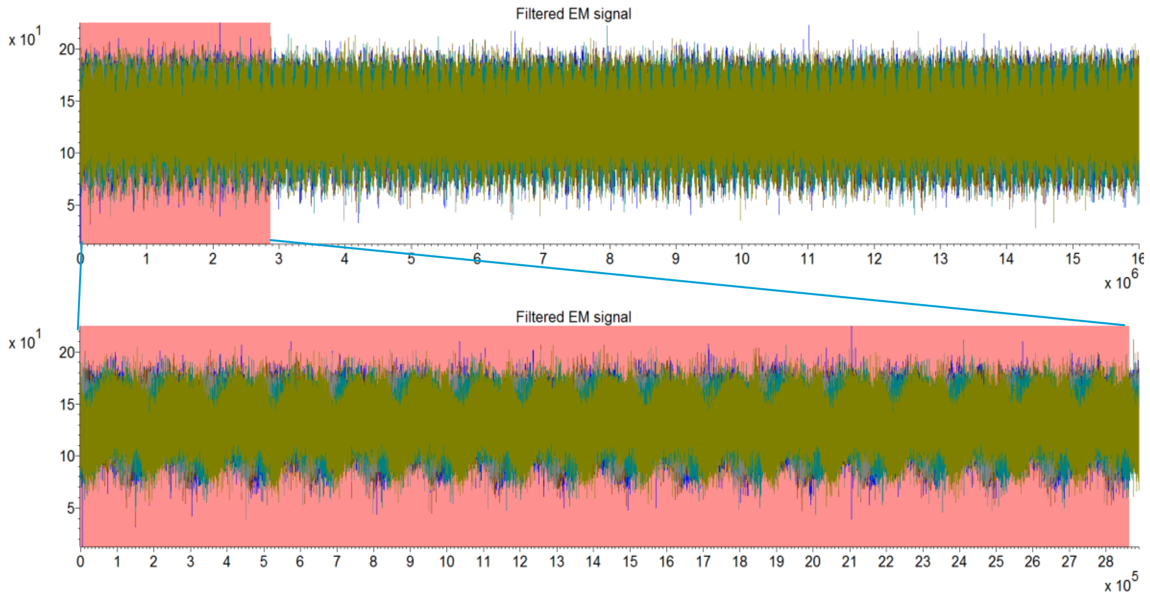


Figure 38: Five overlaid filtered EM traces and a zoomed in view to see the misalignment

It was not possible to align the traces on the visible patterns. Therefore, the points of the trace were averaged over an interval with a length of 10,000 sampling points. The result of this time-filtering can be seen in Figure 39. Then, it was possible to align the traces at the visible peaks. It was also observed that the start of the traces was slightly different, some traces starting with an incomplete peak. It was not possible to distinguish if this peak is the first peak that belongs to the generation of the random data or if the first peak of the random number generation is the peak after the incomplete one. Therefore, it was decided to align the traces in two different ways at the beginning. One alignment matches the incomplete patterns with the full pattern in the other traces. The second alignment matches the first full pattern in every trace. The arrows in Figure 39 identify the different points of alignment that were used.

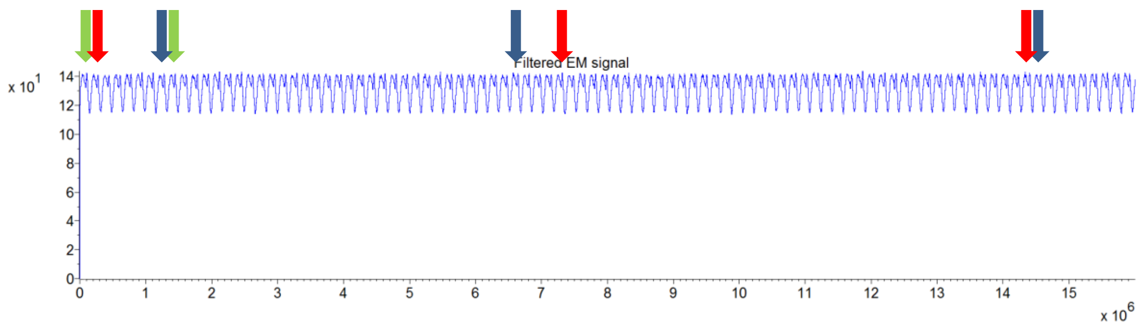


Figure 39: A time-filtered trace. The red and blue arrows indicate the different points of alignment with the two different alignments just in the beginning. The green arrows indicate points which were used for alignment in both cases.

Figure 40 and Figure 41 show the result of one alignment on the time filtered traces. As it can be seen the alignment is getting worse after a while.

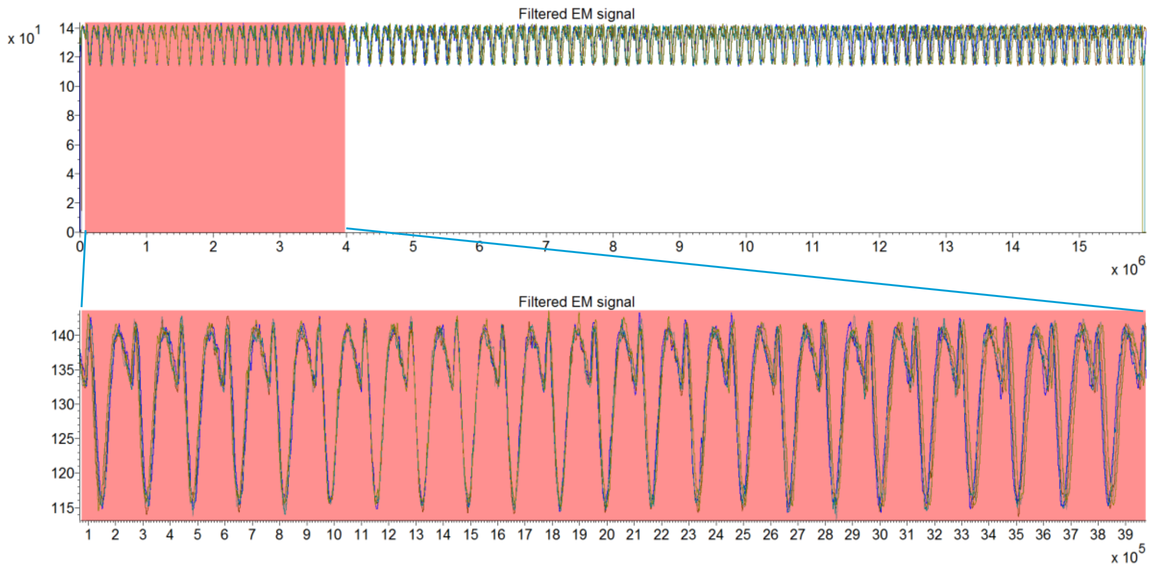


Figure 40: Aligned time filtered traces. The traces are aligned on the peak indicated by the second green arrow in Figure 39.

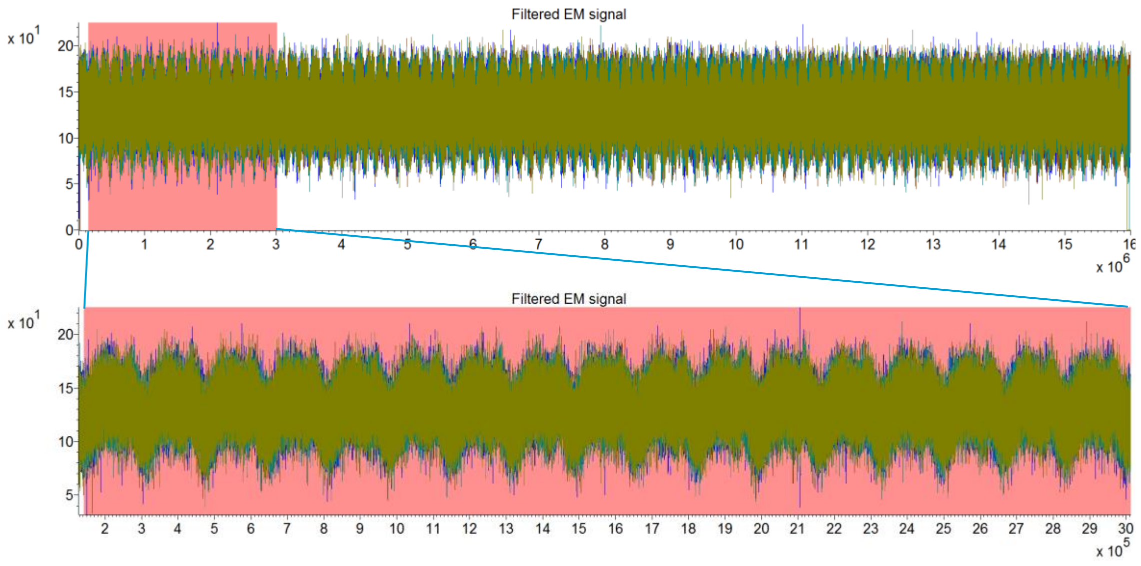


Figure 41: The alignment achieved in Figure 40 applied on the filtered EM signal.

Because of this drifting the correlation analysis of the counter value was restricted to the area around the point of alignment. The small peaks in the correlation graphs that can be observed on the whole length of the trace were investigated further. For a second correlation analysis the traces were aligned at the point where the peaks occur. In the correlation results the peaks disappeared, which leads to the conclusion that these peaks are not data-dependent and occur as a random event. So no significant and stable correlation could be observed. Example correlation traces for all stored 256 bytes can be seen in Figure 42.

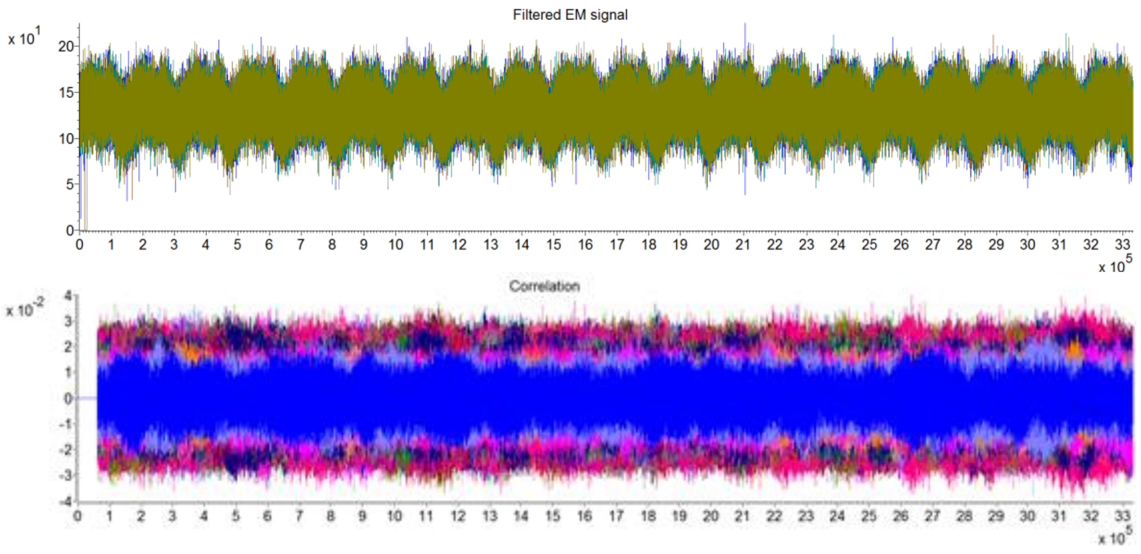


Figure 42: Correlation result for all 256 values of the counter for the alignment in Figure 41.

It was decided to convert the traces into the frequency domain and to do an additional correlation analysis there. This is done to overcome the misalignment issues at the beginning of the traces. The traces were partitioned and a FFT was calculated over each segment, with a 50% overlapping of the segments. So now all points in one segment are taken into account if a correlation calculation is performed. Different segment lengths were tried. One converted trace can be seen in Figure 43. Several alignments were converted into the frequency domain, but again no correlation with the counter value was found. An example correlation trace for the first eight byte-values can also be seen in Figure 43.

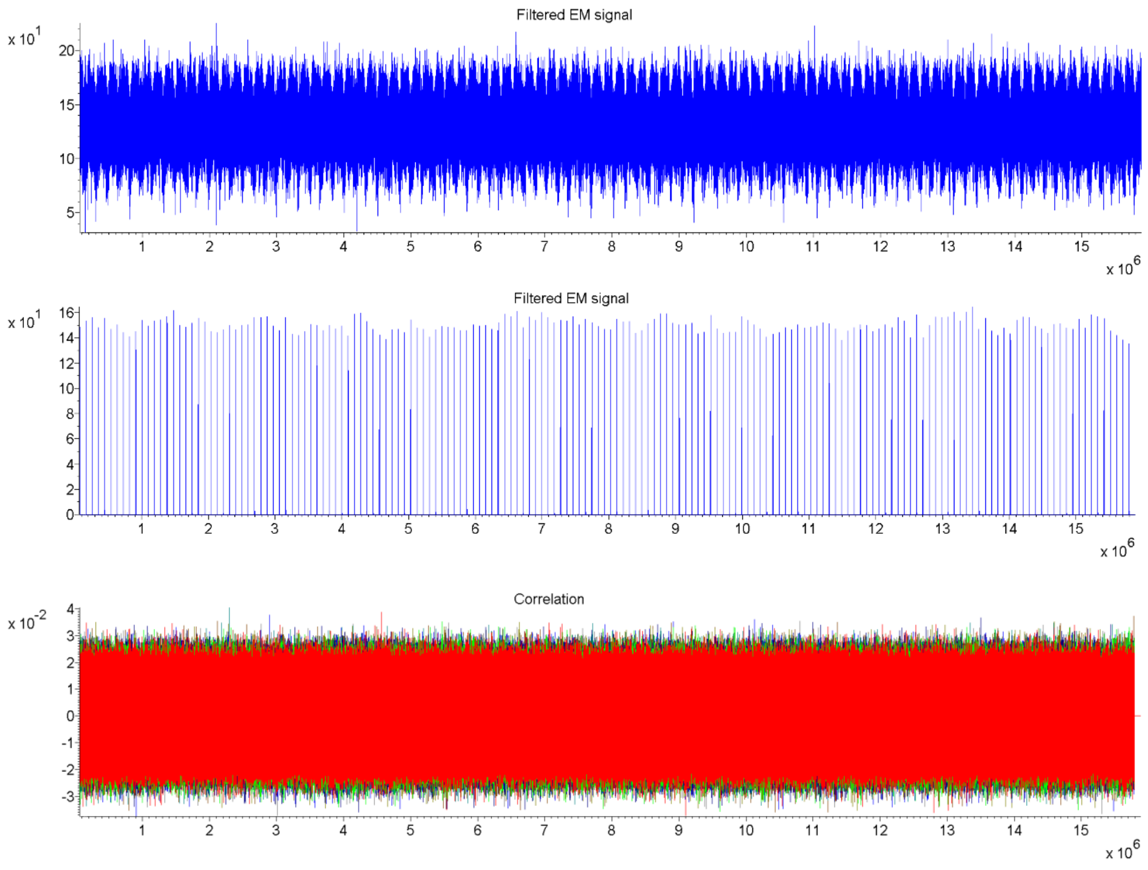


Figure 43: In the top a filtered EM trace. In the middle the corresponding converted trace in the frequency domain. In the bottom the correlation traces for the first eight byte-values of the counter.

It was decided to start an additional acquisition over several weeks. A set of 100,000 traces at a sampling rate 5 GS/s was recorded. The same steps and test described before were performed with this new set. No different results were observed.

7.3.1.2 Correlation analysis of the raw random value

As a second step of the investigation of the Online Entropy Test the GET_RAW command was investigated. A set of 20,000 EM traces were recorded at a sampling rate of 5 GS/s. An example trace can be seen in Figure 44.

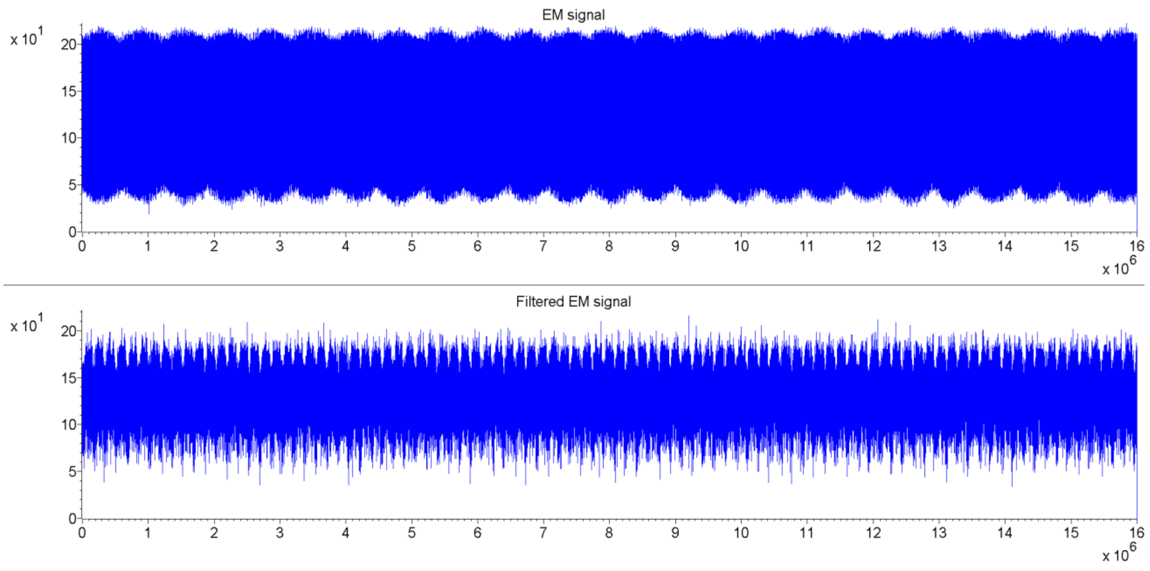


Figure 44: Raw and filtered power trace of the GET_RAW command.

As in the first investigation no pattern could be observed that indicates the start of the random data generation. The raw random byte values that are returned from this command are composed on the last bits of the counter values as investigated before. Therefore, it was decided to perform bitwise correlation analysis in order to identify the point where the raw random data bytes are generated. So a correlation analysis was performed on the unaligned traces, without any significant results. Then, the traces were filtered over time in order to perform different alignments. A time-filtered trace and the different alignment points can be seen in Figure 45. The differences between unaligned and aligned traces can be seen in Figure 46 and Figure 47.

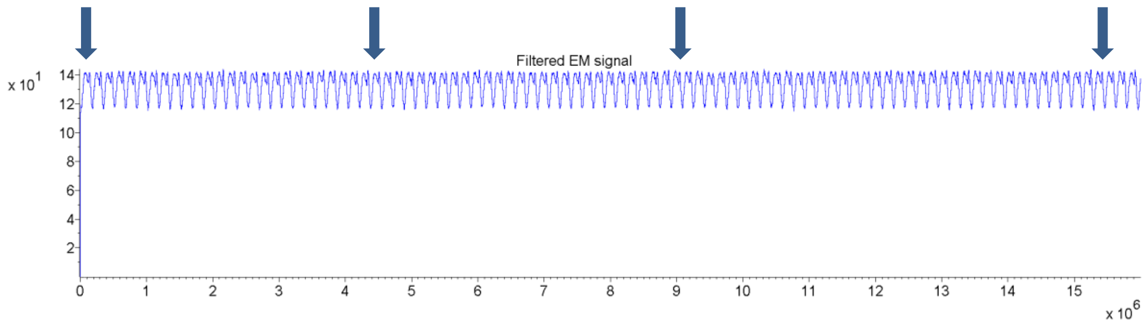


Figure 45: Filtered EM trace with applied time filtering.

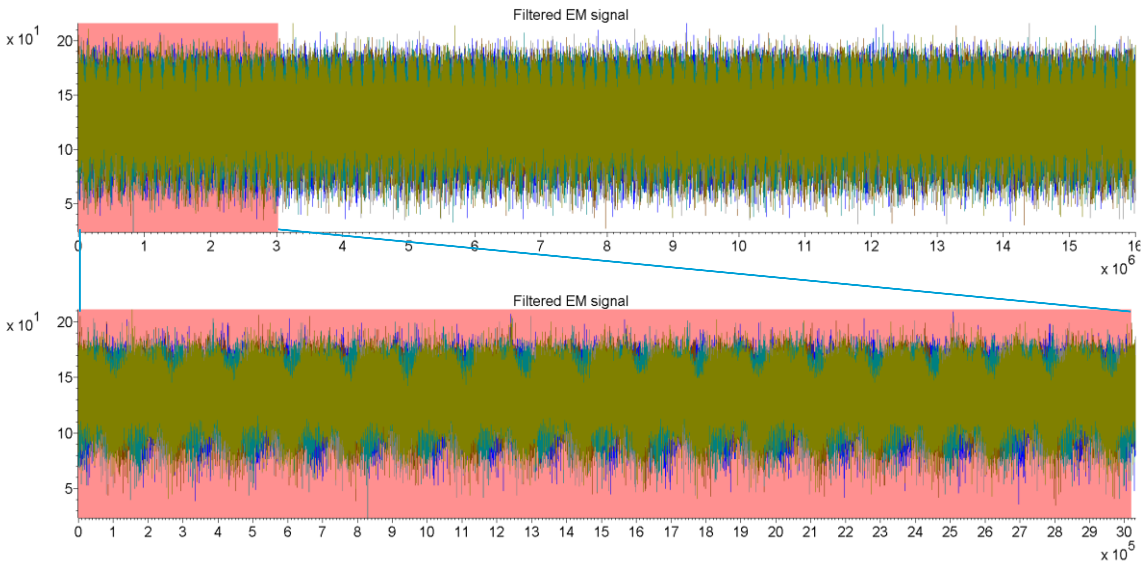


Figure 46: Five overlaid filtered EM traces, without any alignment.

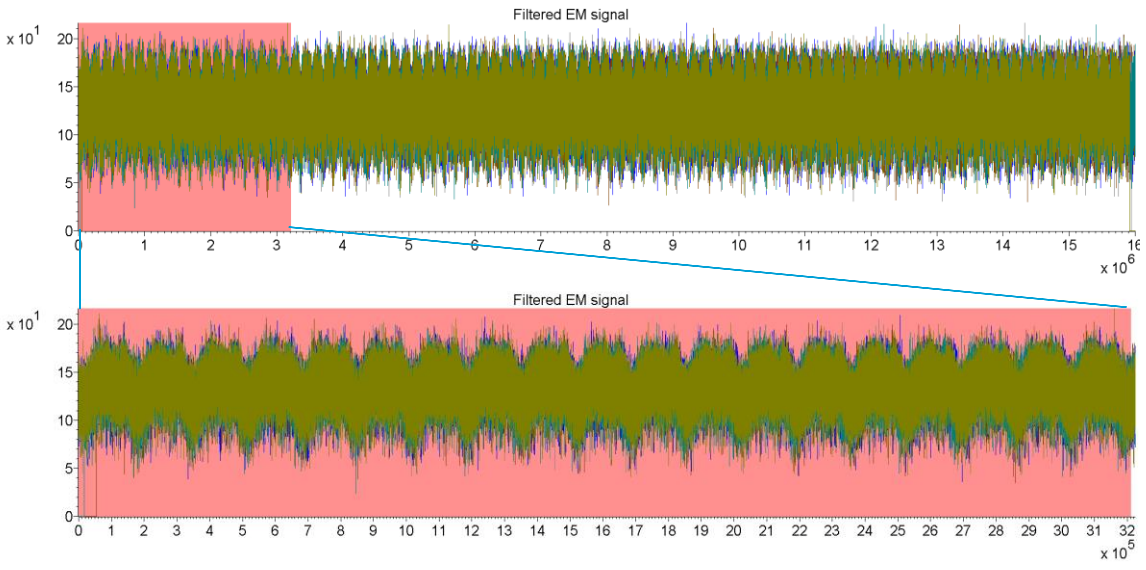


Figure 47: Five overlaid traces after alignment at the point indicated by the first blue arrow in Figure 45.

Like in the previous test no significant correlation peaks could be found. Example correlation traces can be seen in Figure 48.

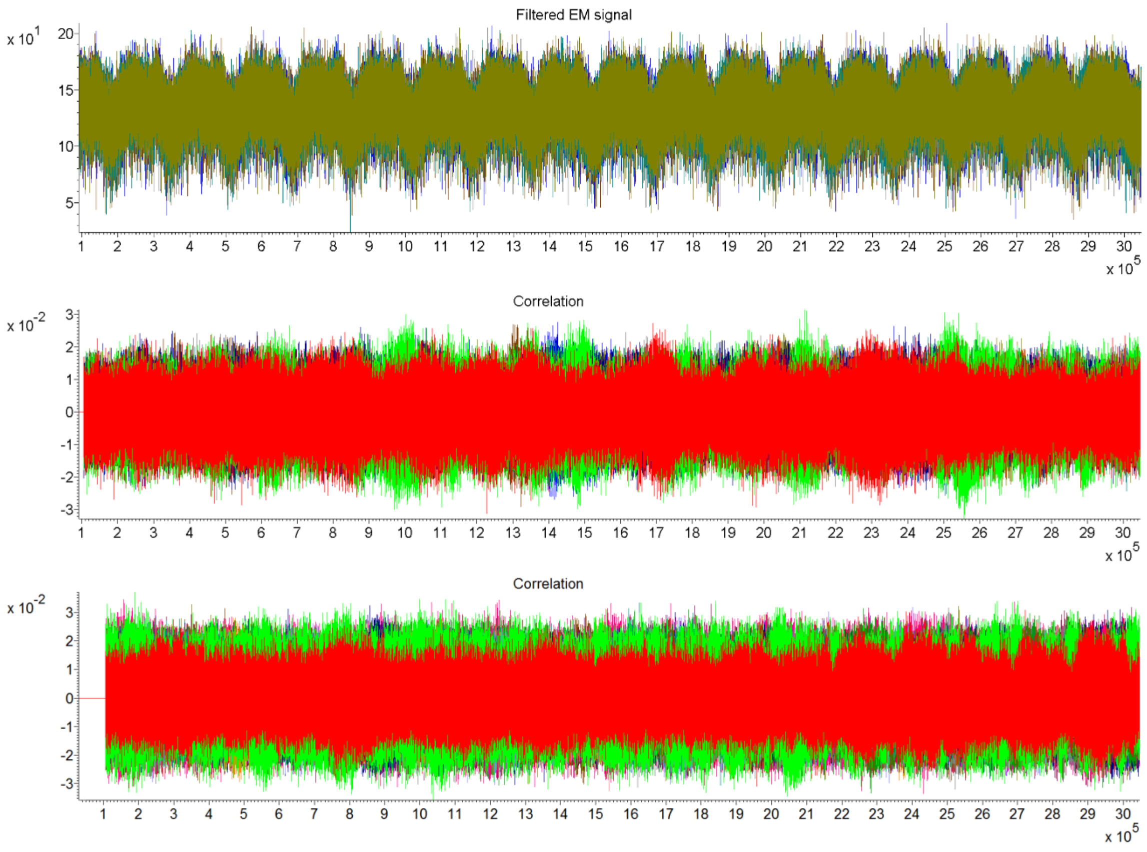


Figure 48: The investigated part of the aligned traces can be seen in the top, the correlation traces of the first eight bytes (middle) and of the first 64 bits (bottom).

The traces were converted to the frequency domain for each alignment. An example of a converted trace can be seen in Figure 49. Correlation analysis was performed but this gave no results. Example correlation traces can be seen in Figure 49.

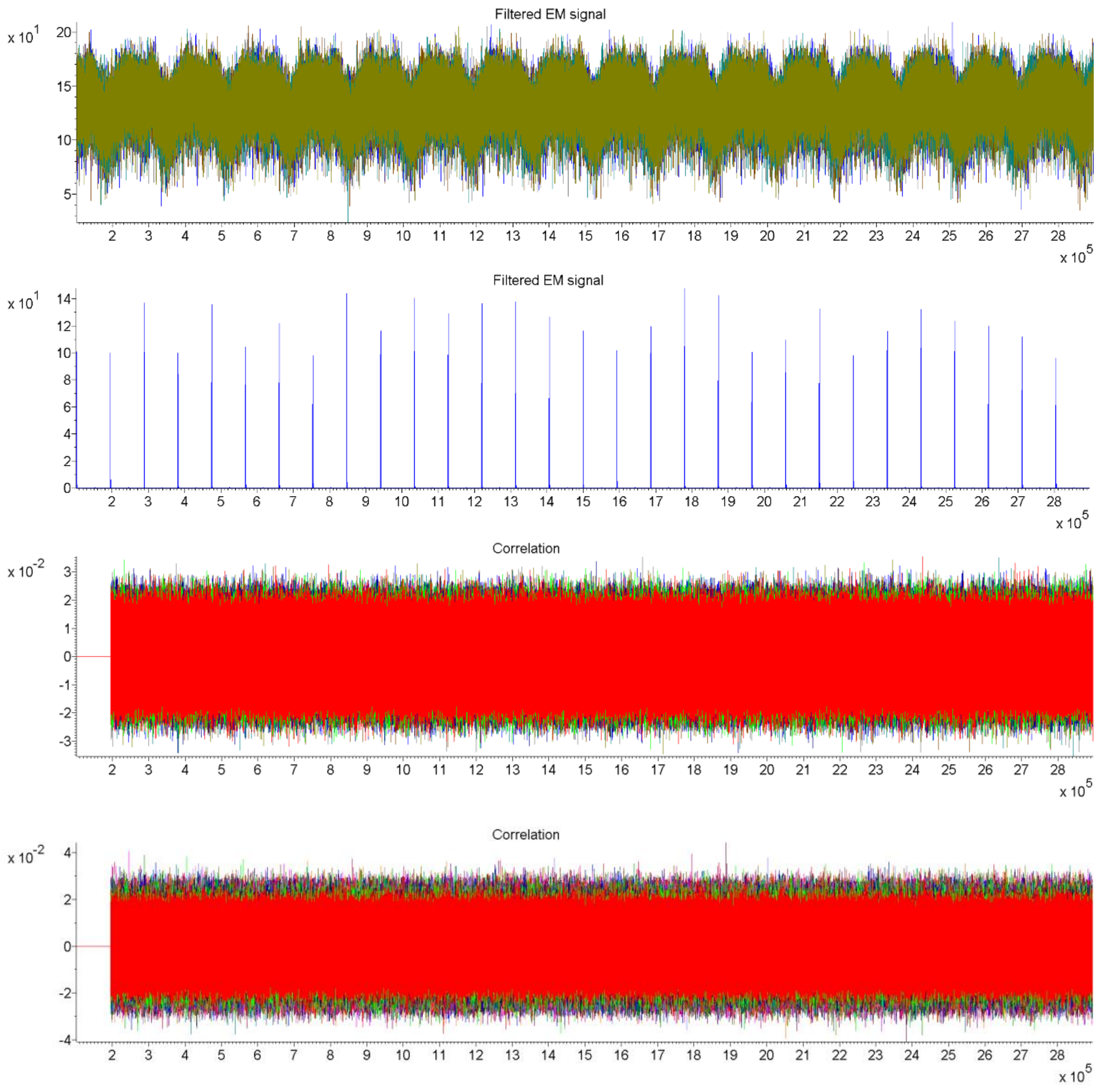


Figure 49: The investigated part of the trace can be seen in the top. The second graph shows a converted trace. The lower both graphs show correlation traces for the first eight bytes (third graph) and for the first 64 bits (bottom graph).

With no point of leakage found, another coil location was tried. Now the coil was moved to exactly the middle of the FPGA. An additional set of 20,000 traces were measured. One pair of raw and filtered EM traces is shown in Figure 50. Here, there are clear peaks which could be interesting and can be easily used to align upon.

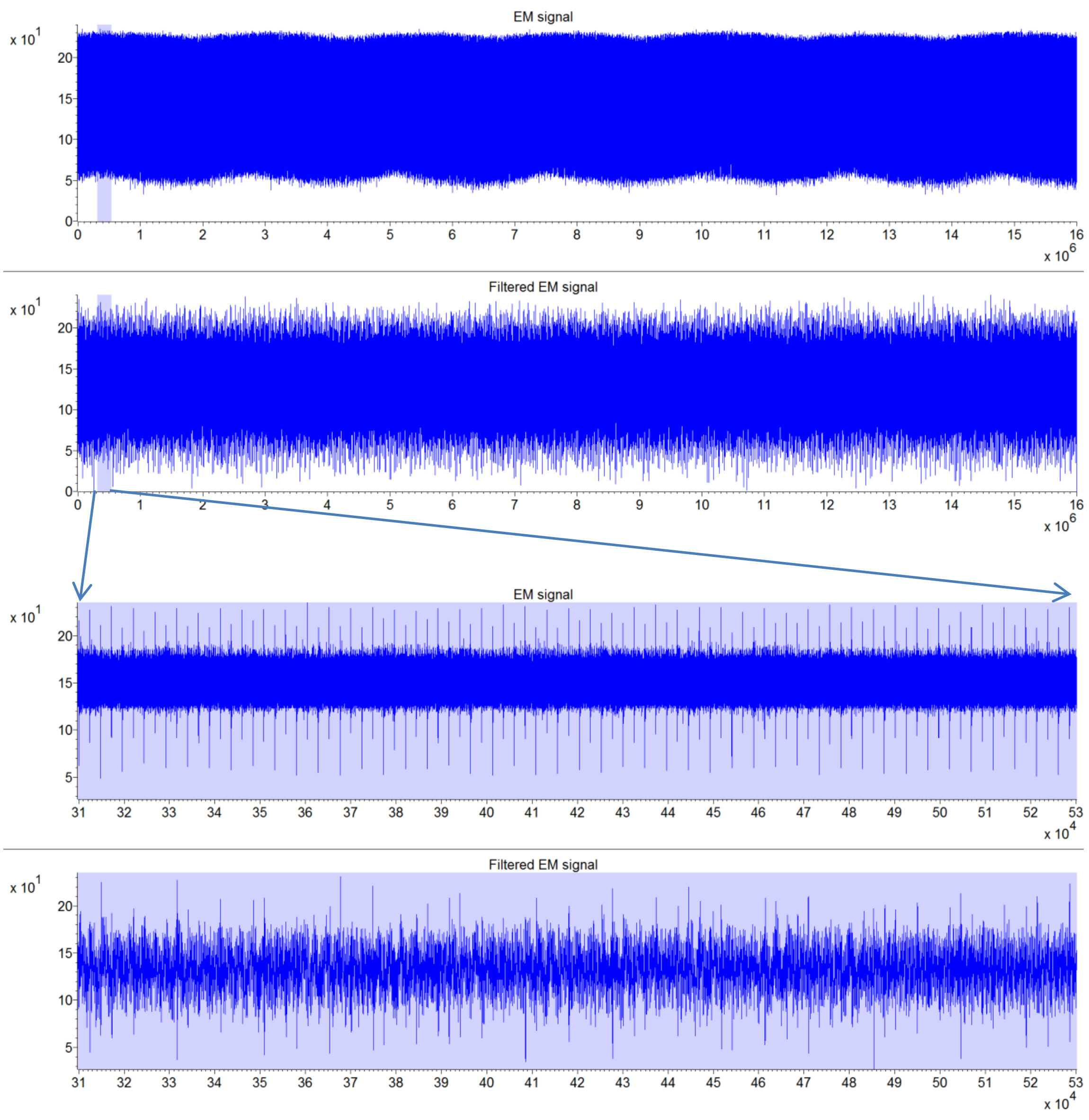


Figure 50: EM and filtered EM signal measured at the second coil location.

Aligning on the peaks at different locations did not, however, improve the previous results. The top graph in Figure 51 shows the EM trace with the arrows indicating alignment points. The bottom graph shows the correlation result gained when aligned at the location indicated by the green arrow. The correlation results when aligned at the other locations all had similar results to the one shown here.

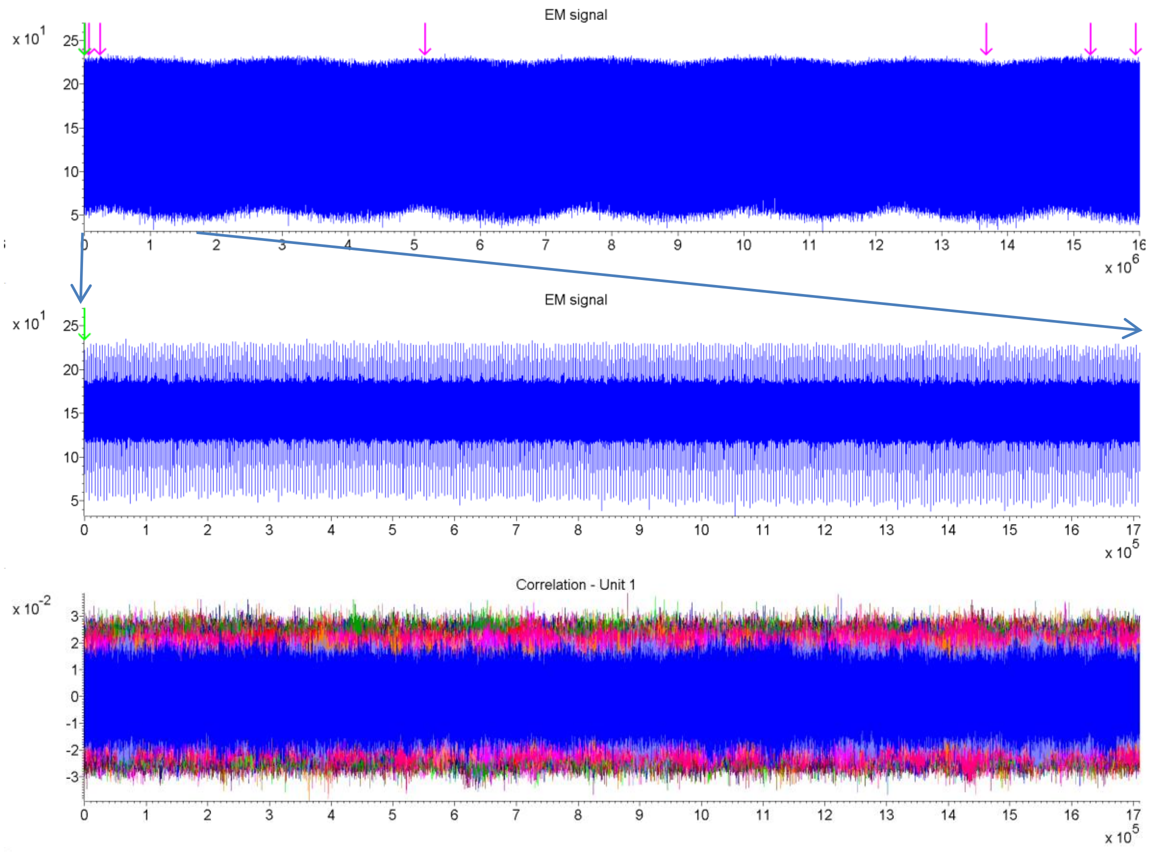


Figure 51: EM signal with arrows indicating the alignment locations (1st). A zoomed in view of the first alignment location indicated by the green arrow (2nd) and the correlation result using the traces aligned there (3rd).

Further investigation was done by transforming the traces from the time domain into the frequency domain using segmented Fast Fourier Transformation. Since the measured trace is very large, only a small area around the different alignment locations was transformed. In Figure 52 the FFT trace and the resulting correlation result is shown of the green alignment point in Figure 51. As before, there were no significant correlation peaks. At all the other alignment points the correlation results were similar.

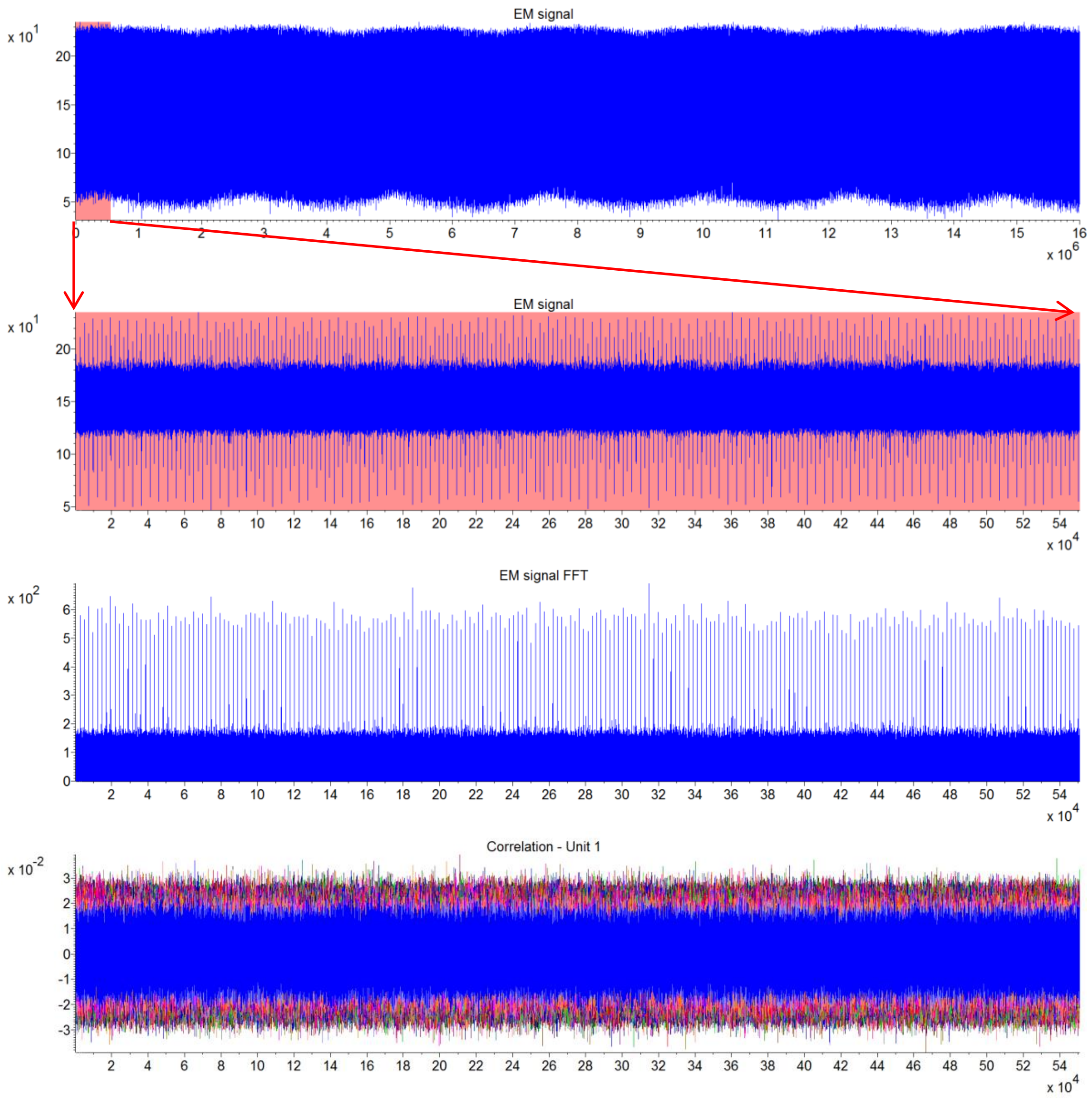


Figure 52: One of the measured EM signals (1st) with a zoomed-in view (2nd) of the interval used for one of the FFTs. The resulting FFT trace (3rd) and the correlation result (4th).

It was not possible to identify a point of leakage in the recorded traces. Therefore, it was concluded that a template attack will not provide any useful results.

7.3.2 Test conclusion

The goal of this test was to investigate the resistance of the Online Entropy Test of the DC-TRNG against side-channel analysis. The TRNG is implemented on a Spartan-6 FPGA. Two different intermediate values in the generation process of the random number generator were targeted. For each of these values a set of 20,000 traces at a sampling rate of 5 GS/s was recorded.

An additional set of 100,000 traces was recorded using the GET_CNT COMMAND. Different alignments were used in order to perform correlation analysis. Additionally the traces were converted to the frequency domain. It was not possible to identify any point of leakage in the recorded traces, even though the device is not protected against side-channel analysis. No visible pattern that could relate to the generation of the values could be identified, which could lead to a starting point for the attack. The only indication for the timing of the attack is the additional trigger signal, which will not be present in a real device. Collection of the traces takes much time (weeks) because each acquisition cycle is slow.

No unexpected behaviour was observed and no weaknesses have been identified.

Chapter 8 Perturbation testing

8.1 Introduction

Perturbation attacks aim to change the normal behavior of an IC by putting stress on environmental conditions to create an exploitable error in the operation of a TOE. Several attack techniques are known to achieve this.

- “Glitching” inserts voltage glitches on the chip’s external connections with an aim to propagate these glitches to internal circuitry where they influence the proper behavior of that circuitry. Examples are Vcc (supply voltage) and clock manipulation. Chips with high quality security have filters or management circuits that aim to prevent these glitches from penetrating the internals of the chip.
- “Light manipulation” aims to expose selected internal circuits with light to alter the electrical behavior of transistors. The light generally forces closed transistors to conduct, which may cause transient effects in logic states. This conductive state is transient when pulsed laser light is applied, while static light causes effects with longer duration. Chips with high quality security have protective features such as top level shielding or dedicated sensors that aim to protect against light attacks.
- Electro-magnetic Fault Injection (EMFI) aims to alter the behavior of the chip by electro-magnetic pulses generated by a coil located in proximity of the chip. The effect is caused by magnetic induction into chip wiring. This induction is transient by definition. Transistor states (closed or conductive) are not directly influenced, but may switch due to induced voltage glitches.
- Forward Body Bias Injection (FBBI) aims to alter the behavior of the chip by applying a bias voltage at certain locations of the substrate, thus locally influencing the actual supply voltage for a certain part of the chip.

Practice shows that light manipulation is the most effective perturbation method. It has a strong effect and can be applied accurately both in time and location. In case countermeasures prevent light manipulation to be applied successfully, EMFI or FBBI may be applicable. These methods have less spatial resolution, which makes them less effective compared to light manipulation. For this reason light manipulation was selected as the preferred method to verify robustness of the HECTOR TRNGs.

8.2 Light manipulation testing on TRNG post-processing

8.2.1 Test details and test results

Test details	
Hardware	LM6
Software	Matrix 3.6.1

Equipment parameters	Vcc = 3.3 V, 2.5 V, 1.1 V
	Objective = 50x IR (LM6)
	Laser wavelength = 1,064 nm (IR)

Lab	Brightsight bv

Table 17: Test details

TCL script	Expected response
init_daughter_hdmi2dff.tcl	Reset the FPGA
get_raw.tcl	Generate 20,480 bytes (5 blocks of 4,096) of random data

Table 18: Commands used during the performed experiments

The following tests are performed in order to get assurance about the quality of the random numbers generated with the FPGA design with post processing.

The test consists of shooting a laser with different power settings and laser pulse widths in order to manipulate the quality of the random output data. In this case the attack target is the post-processing, because any manipulation of the source of randomness would be obscured by the post-processing. In practice this means that source manipulation cannot be detected, except by the internal on-line tests. In addition, manipulation of the post-processing is quite detectable in most cases, because perturbation of the post-processing hardware logic generally causes bias.
The expected result of a manipulation is that the on-line tests after the post-processing will detect such anomalies and raise an exception.

For this test, 20,480 bytes of random data are generated (5 blocks of 4,096 bytes) per attempt. Different types of diode lasers were used during the experiments. The most common ones for perturbation of silicon devices apply infrared (IR) and green wavelengths.

The first tests were performed with IR light. The settings were the following:

- Laser input voltage: 0.5 V, 0.6 V, 0.7 V, 0.75 V, 0.8 V, 0.9 V, 1.0 V, 1.5 V, 2.0 V, 2.5 V, 3.0 V
- Laser pulse width: 5 ms, 10 ms, 15 ms, 25 ms, 50 ms, 75 ms, 100 ms, 125 ms, 150 ms
- Delay: 500.0000 µs after the reference signal
- Scan locations: 725
- Attempts per location per delay: 1
- Intended attempts per complete surface scan: 725
- Total attempts performed: 48,575 (out of 71,775 possible voltage-delay combinations)

The voltage setting of the laser represents its intensity. However, the laser power output is not entirely linear with applied supply voltage, nor is this value to be compared with different lasers.

During these surface scans, more than 99.5% of the attempts resulted in no impact. For this reason not all combinations were tested; Testing at the extremes of voltage and delay time did not provide any useful results, therefore intermediate voltages-delay combinations were skipped. In total 48,575 attempts were executed out of the maximum 71,775 combinations that can be made with the settings mentioned above. In total 67 voltage-delay combinations were verified. An example of the results of surface scans for six voltage-delay combinations is given in Table 19, showing amongst others the two-corner extremes (0.5 V – 5 ms and 3 V – 150 ms).

The legend used for the result plots is:

- Green: no errors and no warnings received
- Blue: error received
- Yellow: warning received

- Red: error and warning received.

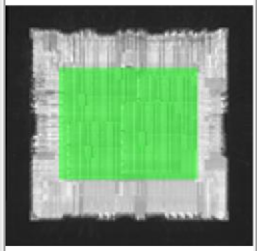
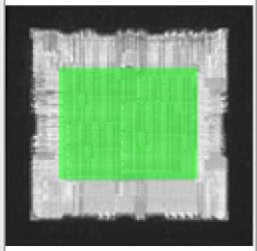
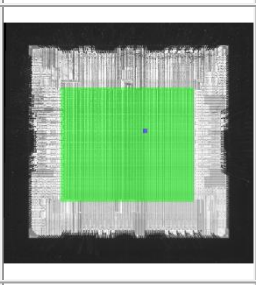
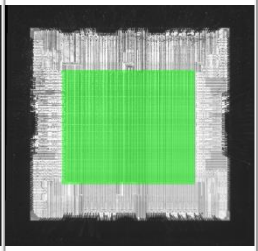
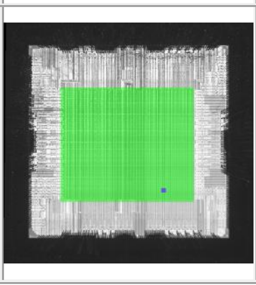
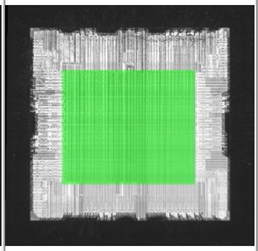
Pulse Width	Voltage		
	0.5 V	1.5 V	3 V
5 ms		Not tested	Not tested
15 ms		Not tested	Not tested
50 ms	Not tested		
150 ms	Not tested		

Table 19: A few of the 67 surface scan results of the light perturbation experiments with IR light

In case responses were received that deviated from expected these were all caused by detection of errors in the generated random streams. No warnings were received. This proves that the built-in tests are capable of detecting deviations in the quality of the randomness and prevents those results from being output.

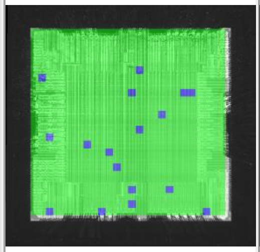
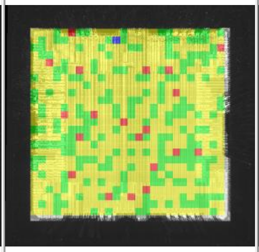
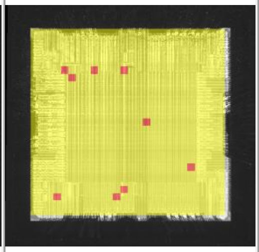
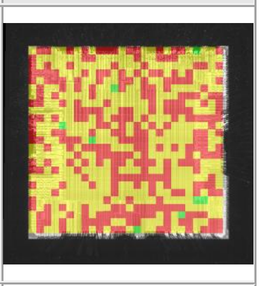
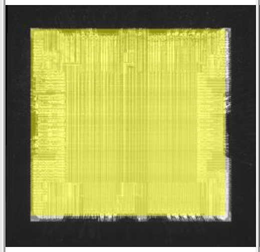
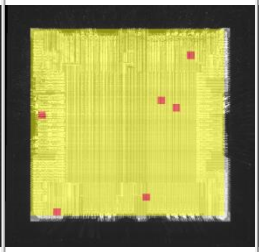
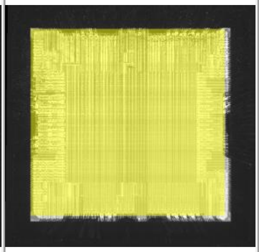
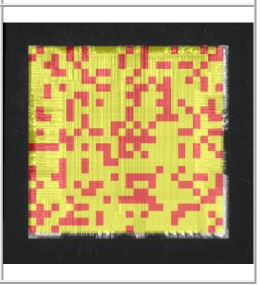
No deviating results were achieved with IR light. The error messages that were returned were all of type: "`High rep. count (Total entropy failure)!`". This indicates that the random number generation process was disturbed, but that the on-line tests of the TRNG detected this. So from a security point of this is correct behaviour and in line with the HECTOR objectives to tailor the on-line tests in such way that the TRNG will under no circumstance output corrupt results. It should detect such situation and prevent the corrupted results to be sent to the user. This behaviour confirms meeting one of the HECTOR objectives.

After finishing the tests with IR light, experiments were performed with green light. The settings used for the experiments were:

- Laser input voltage: 3.6 V, 3.8 V, 4.0 V, 4.2 V
- Laser pulse width: 10 µs, 25 µs, 50 µs, 100 µs, 500 µs, 1ms
- Delay: 500.0000 µs after the signal indicating random numbers are being generated.
- Scan locations: 650
- Attempts per location per delay: 1
- Intended attempts per complete surface scan: 650
- Total runs analyzed: 15,600 (all voltage-delay combinations)

The green laser has different characteristics compared to the green laser. The pulse-widths of the green laser are much shorter than can be achieved using an IR laser. It must also be noted that the voltage settings of the green laser cannot be compared to the voltage settings of the IR laser.

During these tests, the sample responded with several errors and warnings. The experiments with a laser pulse-width below 100 µs did not yield any interesting result. Table 20 shows the results of the experiments.

Pulse Width	**Voltage**			
	3.6 V	**3.8 V**	**4.0 V**	**4.2 V**
100 µs				
500 µs				

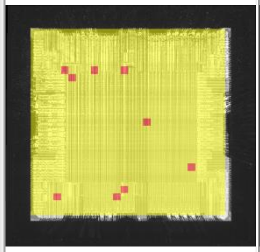
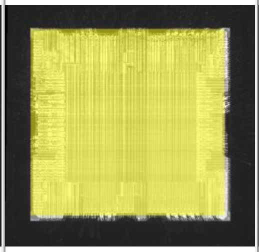
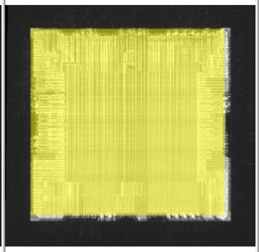
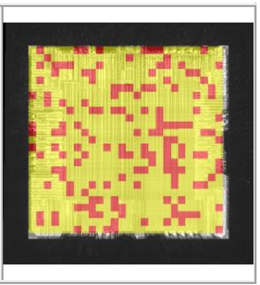
1 ms				

Table 20: Results of the experiments with green light

The list of errors and warnings received is shown in Table 21.

ERRORS		WARNINGS	
Type	Amount	Type	Amount
`High rep. count (Total entropy failure)!`	22	`Incorrect mean value!`	4,905
`Small entropy (Online test failure)!`	486		
`Counter overflow!`	177		

Table 21: Errors and warnings recorded during the experiments with green light

During the experiments, IR light did not show to have much impact, while green light showed a completely different impact. Therefore green light was used for a second run of experiments. The settings used for these experiments were:

- Laser input voltage: 3.8 V, 4.0 V, 4.2 V
- Laser pulse width: 100 µs, 500 µs, 1 ms
- Delay: 500 µs after the signal indicating random numbers are being generated
- Scan locations: 756
- Attempts per location per delay: 6
- Intended attempts per complete surface scan: 4,536
- Total attempts performed: 40,824 (all voltage-delay combinations)

During these tests, only two unexpected results were recorded. The details of these two cases are shown in the table below:

Case #1		Case #2	
Errors	Counter overflow Jitter variance out of range (Online test failure) High rep. count (Total entropy failure) PLL0 not locked	Errors	Counter overflow High rep. count (Total entropy failure)

Warnings	Incorrect mean value	Warnings	Incorrect mean value
Laser parameters	Laser input voltage = 3.8 V Laser pulse width = 1 ms	Laser parameters	Laser input voltage = 4 V Laser pulse width = 100 μs

Table 22: Details of the two unexpected responses

The location where these unexpected results were achieved is shown in Figure 53 (marked in red).

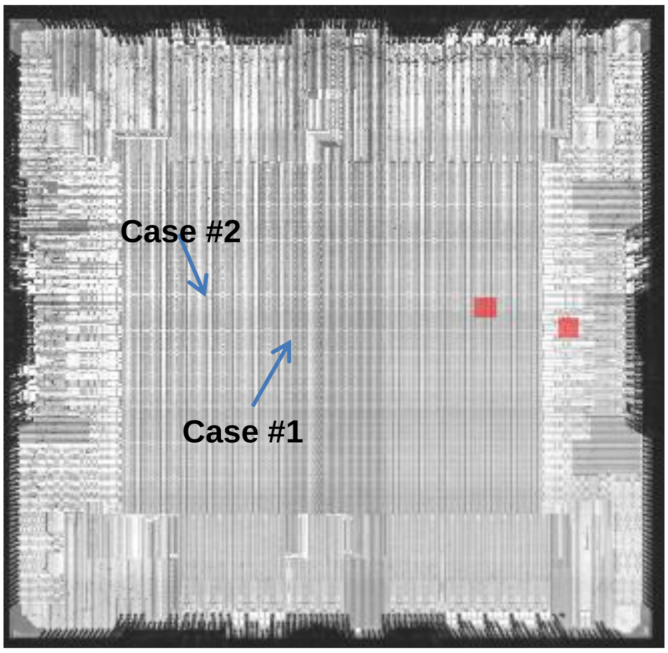


Figure 53: Locations in which the two unexpected results were recorded

The random data obtained in these two cases did not show any fixed pattern.

Another experiment was performed targeting the area around these two locations with the following settings:

- Laser input voltage: 4.4 V, 4.6 V
- Laser pulse width: 1 ms, 2 ms
- Delay: 500 μs after the signal indicating random numbers are being generated
- Scan locations: 135
- Attempts per location: 3
- Intended attempts per complete surface scan: 405
- Total attempts performed: 1,620

No error or warning was recorded during this experiment.

After finishing the laser experiments, all random data collected was analyzed. As the AIS31 tests did not show a flaw that could be used, another manner of identifying weakened random data blocks was attempted. For each file recorded, which contains 20,480 bytes, the occurrence of each byte value was counted. The expected occurrence is 20,480 / 256 = 80. Bytes that occur 120 times or more were further investigated, but no relevant data patterns were found. The number of consecutive occurrences of identical values was also checked for each file, but no fixed output was found.

This means that an attacker has no means to identify potential weakening of random data and would not have any feedback of being successful in downgrading the quality of a captured data block. As the identification part of the attack path is not present, a complete rating of the attack scenario is not relevant.

8.2.2 Test conclusion

Tests were performed with green light and Infra-Red light in order to check if random number generation can be affected. During the experiments, the two relevant results obtained showed that when entropy level was compromised, it was detected. No significant result was obtained. The TRNG is sufficiently robust against active perturbation using light.

8.3 Frequency injection on TRNG

8.3.1 Test details and test results

The following tables show the details of the performed experiments and the commands that have been used.

Test details	
Sample details	Hardware: Spartan 6, RNG design UJM, version 20170928_TQ.
Hardware	RF Synthesizer : RF amplifier: HD Communications Corp. HD29347
Software	Matrix v3.6.1
Equipment parameters	Vcc = 5 V

Table 23: Test details

Command	**File executed**	**Response (typical values)**
GET RAW	get_raw.tcl	File size: 20,000 Bytes

Table 24: Commands used during the performed experiments

The goal of this test is to verify if injection of a high frequency signal to the V_{CC} of the FPGA reduces the entropy of the generated random numbers. The RF signal may interfere with the internal switching of the digital logic, which in turn may affect the noise source that is used to generate the random numbers. This could create patterns in the bits, which potentially lead to a degradation of the quality of the random data. If this is the case, this should be observable from entropy analysis results using the AIS31 test suite. For example, in case patterns in the bits would cause a subset of the byte-values to occur much more often than other byte-values, this would result in an entropy value lower than the threshold defined.

The first experiment injected a continuous sine wave through the V_{CC} line of the tested TRNG. The power and frequency of this sine wave ranged from:

- Power:
  - Before amplification: from -20 dBm (0.01 mW) to -6 dBm (0.25 mW), in steps of 2 dB. This is the power output of the synthesizer.
  - After amplification: from 22 dBm (158.5 mW) to 36 dBm (3,981 mW), in steps of 2 dB. This is the actual power injected in the TRNG.
- Frequency: range from 100 MHz to 700 MHz, in steps of 10 MHz. This frequency range is selected because it covers the internal switching frequency of the FPGA logic.

In order to inject such signal, the following devices were used:

- Synthesizer: in order to generate the high frequency signal.
- RF amplifier: used to increase the power of the generated high frequency signal.

Figure 54 shows the block diagram of the final set-up (top part) and the actual set-up (bottom part). The schematic shows a synthesizer generating the high frequency signal, connected to an RF amplifier to increase the power level of the signal. This RF amplifier is connected to a capacitor of 100 nF (used to isolate the DC component needed to power on the FPGA from the RF amplifier). The FPGA is placed between a capacitor and an inductor (680 µH). This inductor isolates the high frequency signal that needs to be injected in the FPGA and protects the power supply V2 from the power of the amplifier.

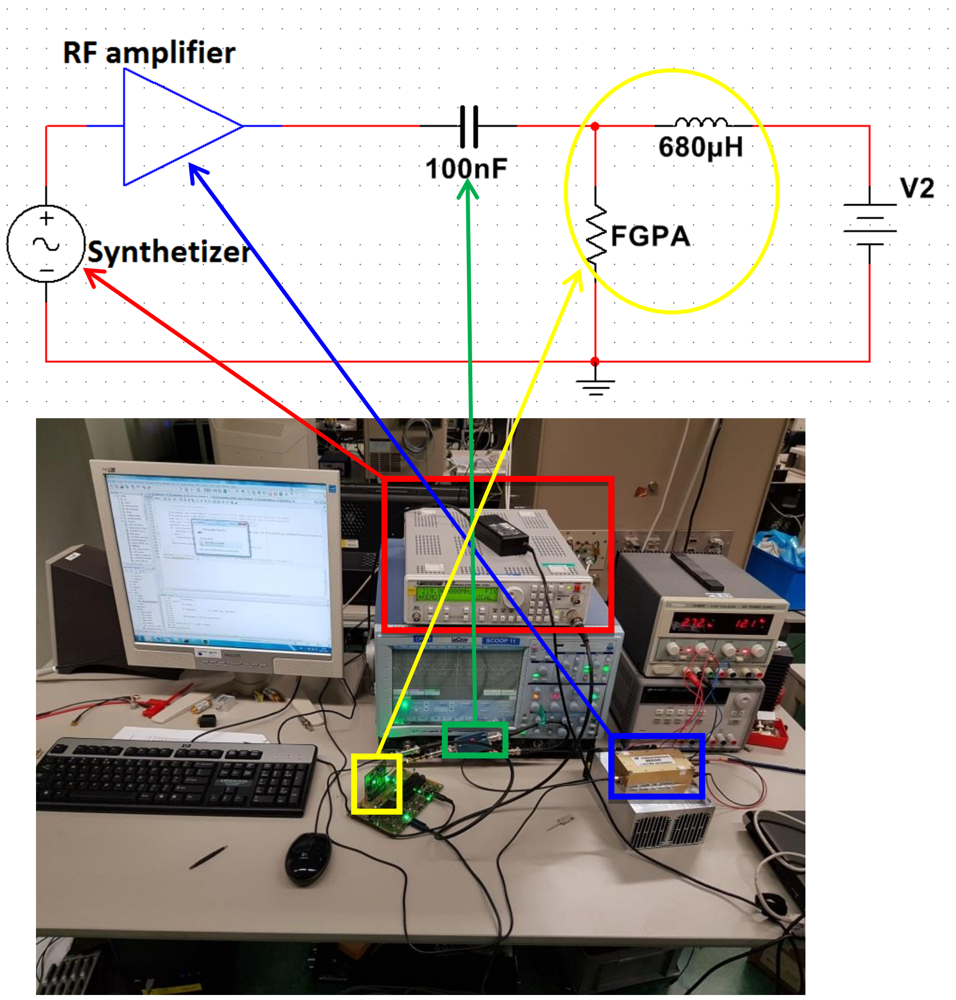


Figure 54: Diagram of the final set-up

The high frequency signal was continuously fed to the FPGA, thus the timing for this experiment is not critical.

For each combination of settings, 20 kB of random data were generated. After generating and storing all these files (which added up to 478 files), the data was analysed in order to verify the entropy. The AIS 20/31 test suite was used for this purpose. After analysing all files, the average entropy obtained was 7.99055, the maximum was 7.99256 and the minimum was 7.98839. This is in all cases higher than the threshold of 7.976 as defined in AIS 20/31. The following image (Figure 55) shows how the entropy is distributed:

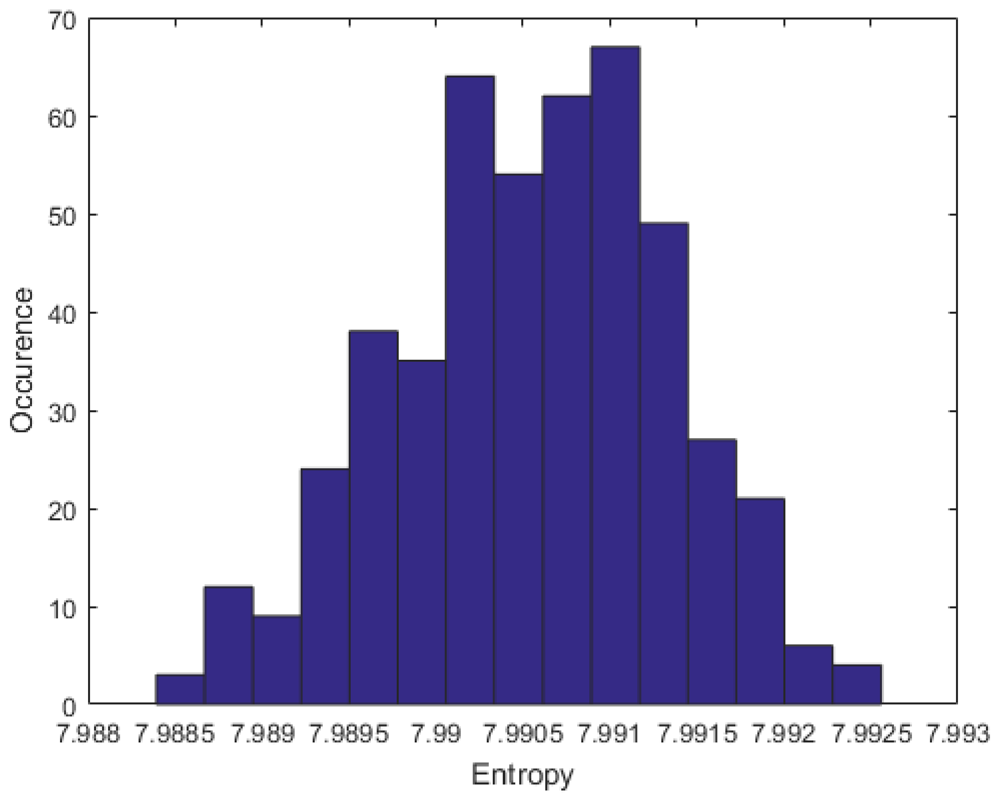


Figure 55: Distribution of the entropy.

A second experiment was performed with different power parameters (the tested frequencies were identical as the previous experiment):

- Power:
  - Before amplification: -6 dBm (0.025 mW) to 0 dBm (1 mW), in steps of 0.05 mW (in this case, the script was converting from power in mW to power in dBm in order to set the RF synthesizer. The idea behind this was to achieve a linear scale for the power, not a logarithmic one).
  - After amplification: 36 dBm (3,981 mW) to 42 dBm (15,489 mW).

For this experiment, 900 files with 20 kB of random data were generated. After analysing all of them, the average entropy obtained was 7.99055, the maximum was 7.9935 and the minimum was 7.98792. Figure 56 represents the distribution of the entropy of the recorded data. The results of this test do not deviate from the ones obtained in the previous experiment.

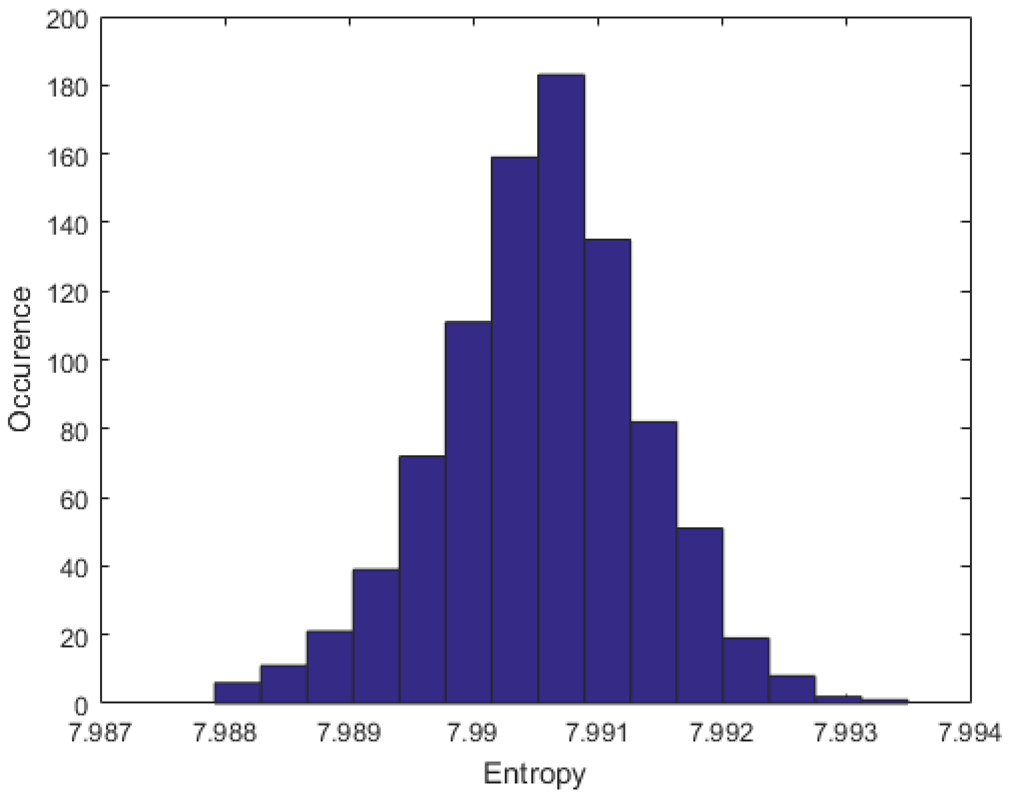


Figure 56: Distribution of the entropy.

In total, 1,378 files were analysed.

8.3.1 Test conclusion

Experiments were performed by injecting an RF signal to the FPGA in order to influence the random data generation process. The entropy of all files analysed was sufficient in relation with the threshold defined in AIS 20/31. This shows that the design is sufficiently robust against RF frequency injection at the level tested.

The light manipulation experiments (8.2) show that the on-line tests are capable of detecting deviations from normal, which acts as an additional safeguard against exploitation of perturbation attacks.

8.4 EMFI testing on STR FPGA with DC-TRNG

The following section sums up the characterization campaign undertaken by STMicroelectronics on the DC-TRNG. The remaining of the section is organized as follows: in a first part, we briefly introduce the DC-TRNG. Then the scope of the characterization is presented by explaining which threat model has been considered. Finally, experiments and their result are described.

8.4.1 Design architecture

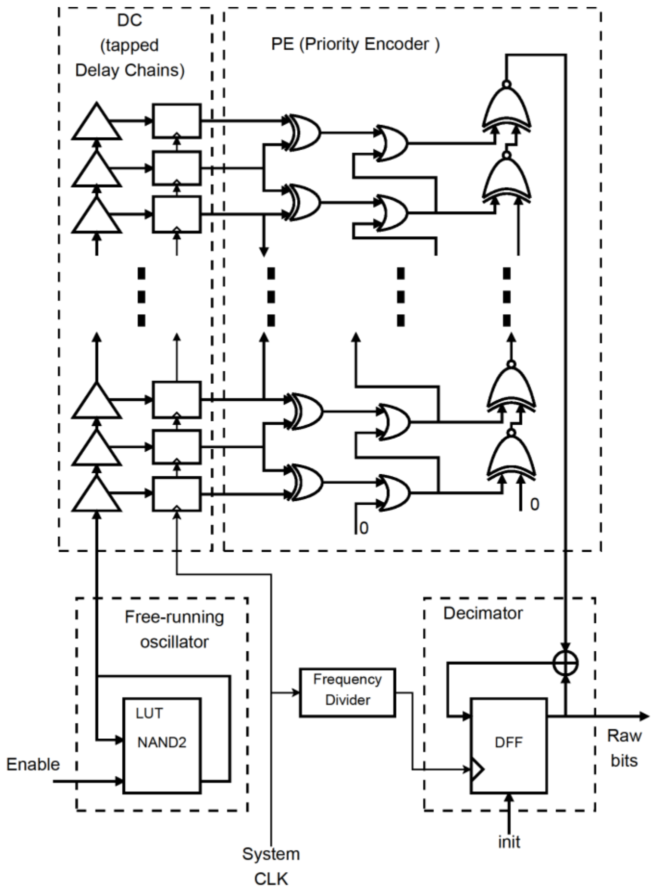


Figure 57: DC-TRNG Architecture

An implementation of the DC-TRNG is shown in Figure 57. A tapped Delay Chain (*DC*), a Priority Encoder (*PE*) and a decimator constitute the digitization module. After a sufficiently long jitter accumulation period, the timing phase of the signal becomes unpredictable due to the accumulated Gaussian noise. This jittered signal coming from the Ring Oscillators (*RO*) propagates through the DC and is sampled by D-flip-flops, each of which is attached to a delay element. The D-flip-flops are clocked by the system clock frequency.

8.4.2 Threat model

Side-Channel Attacks (*SCA*) exploit the statistical dependency between physical information leaked from the device and intermediate values processed by the implementation in order to extract secret keys. One of the most deployed countermeasures against SCA is masking of intermediate values by short-living random values that are only known internally.

As a metric of security degradation we are using the minimum number of measurements needed for univariate Correlation Power Analysis (*CPA*) attack to succeed, as described in D3.3.

Notice that for the experiment section described below the measurements are taken on the same target and with the same DC-TRNG architecture. The analysis results of the 1st-order Boolean and IP (Inner Product) masking schemes supplied with biased random numbers are shown in Figure 58. The curves in this figure illustrate the minimum number of measurements needed for a successful CPA attack as a function of different noise levels and different levels of bias.

In case of Boolean masking, both types of biases (towards 0 and towards 1) have the same effect on security degradation. We observe that for low noise levels and a bias of 25%, the number of measurements needed for successful attack is only 225. On the other hand, for high noise levels and bias of only 5%, the attack can still be successfully mounted, but now requires approximately 650,000 measurements. In case of an IP masking scheme, the attack's success depends on the type of bias - the attack is more successful if the random numbers are biased towards 0. However, IP masking is more resilient than Boolean masking - the minimum number of measurements that enable the attack for low noise levels and a bias of 25% is 22,000. Furthermore, the attack is only successful for a bias level of at least 10%, when it requires approximately 1,250,000 measurements.

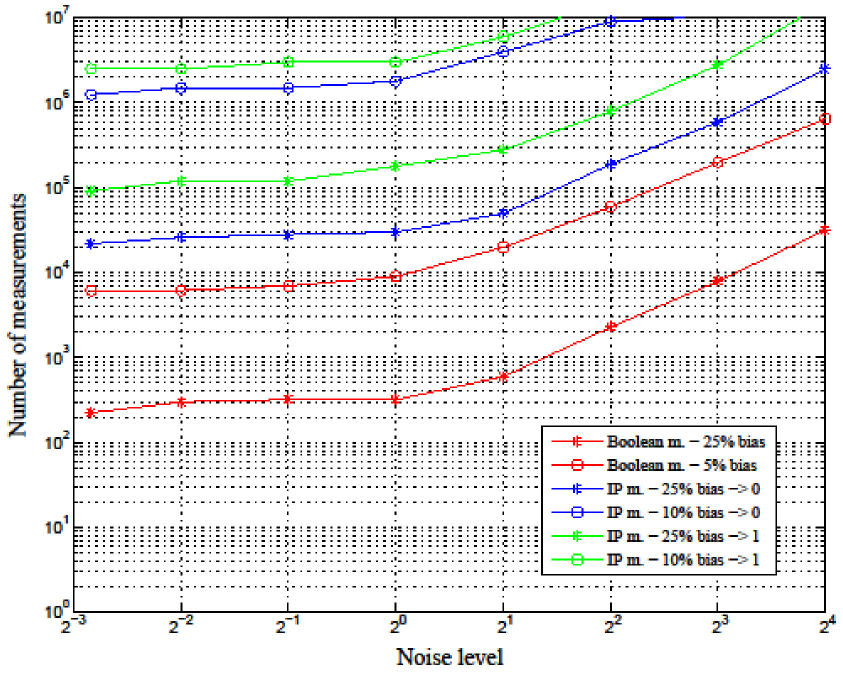


Figure 58: Univariate CPA attacks against AES S-box protected by 1st-order Boolean and IP masking scheme

From the above analysis, two scenarios can be identified. Firstly, the attacker is able to fully control the random bit-flow by disrupting directly the entropy source or its post-processing blocks (the *PE* and the *DC* chain in our case). In such a case, the attacker is thus able to craft a sequence of numbers passing the embedded statistical tests and to insert it once or several times in the flow of random numbers. Secondly, the attacker is able to introduce a small bias of at least 10% to be able to successfully run a CPA.

8.4.3 Experimental design

A DC-TRNG – composed by ten carry chains and two decimator stages in cascade – was implemented in a Xilinx Spartan-6 FPGA and operates at a system clock frequency of 4.5 MHz. The entropy source was instantiated as one Lookup Table (*LUT*) and the delay chain is composed of CARRY4 primitives.

The EMFI platform (Figure 59) is composed of a pulse generator with amplitude in the range of 0 V to 400 V and a pulse width from 8 ns to 100 ns. The pulse generator can repeat pulses at a frequency of 2 kHz. The experimental campaigns were run with two different EM probes: a U-shape probe and a cylindrical probe with a flat end (Figure 60). Probes are inductors made using ferrite core material with various numbers of wire windings. The initial search for a probe position leading to faults (surface scan) was done with the flat-ended probe. Then the U-shape probe was used to produce more localized faults (faults disrupting only one functional block of the DC-TRNG). The faults reported in the tables were obtained with the U-shape probe.

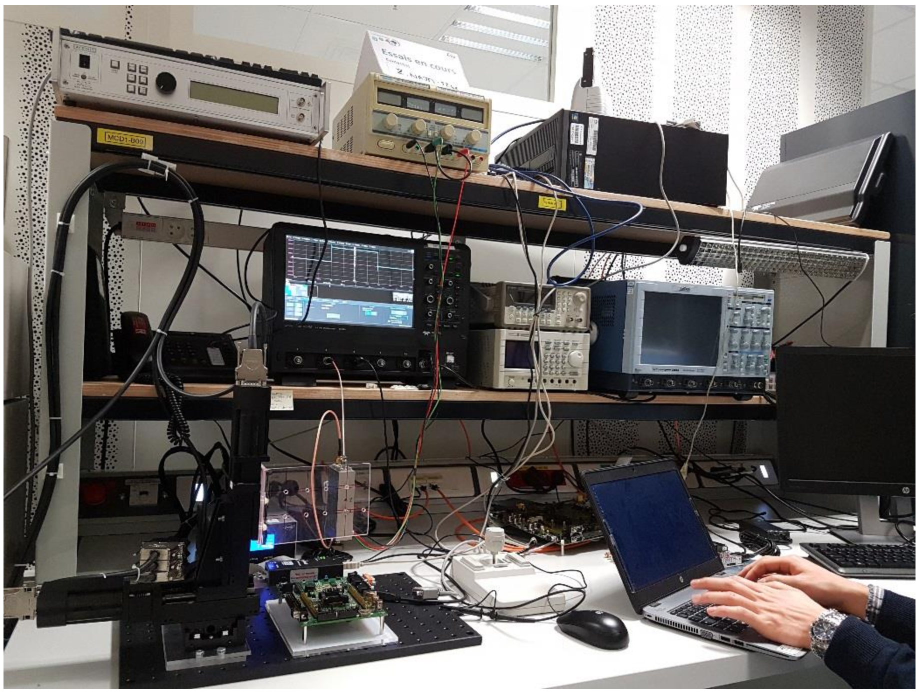


Figure 59: EMFI bench

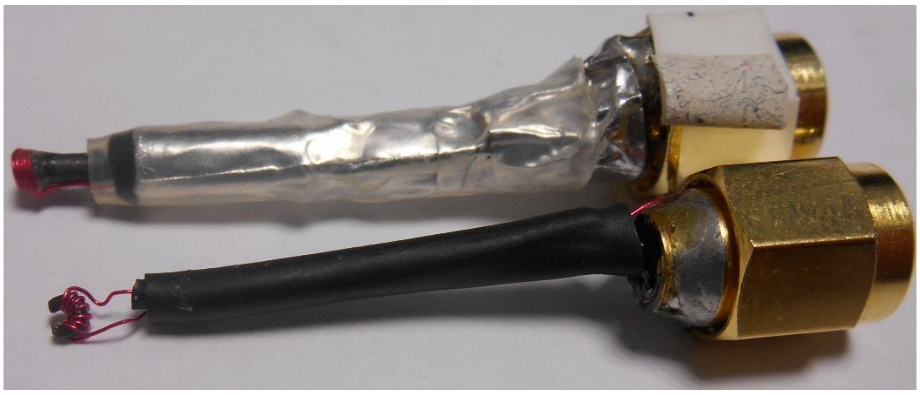


Figure 60: EMFI probes used for characterization

The DC-TRNG fault injection tests were designed to provide an insight into the internal behavior of its components (namely PE and DC) as well as computation involved in the random stream generation, i.e. the decimator stage. Moreover, across all different tests that were done, the PE, DC and RO were placed at the same position in the FPGA design. As a first test, the whole design was considered. To monitor the PE and Decimator's streams their outputs were send directly to the scope alongside their sampling clocks (which was used as a trigger signal for EM-pulse generation). Then, in order to separate the actual fault on the TRNG and any side-effects on the design induced by EMFI, the outputs of the monitored signals were delayed by several clock cycles using FIFOs as depicted in Figure 61. Therefore a fault occurring on the DC-TRNG would appear some clock cycles after pulse generation (equal to the length of the FIFO), as illustrated in Figure 62 and Figure 63. By contrast, if a perturbation occurs on input/output bond wires or on our FIFO, it would appear instantly or with a delay lower than the total length of the FIFO.

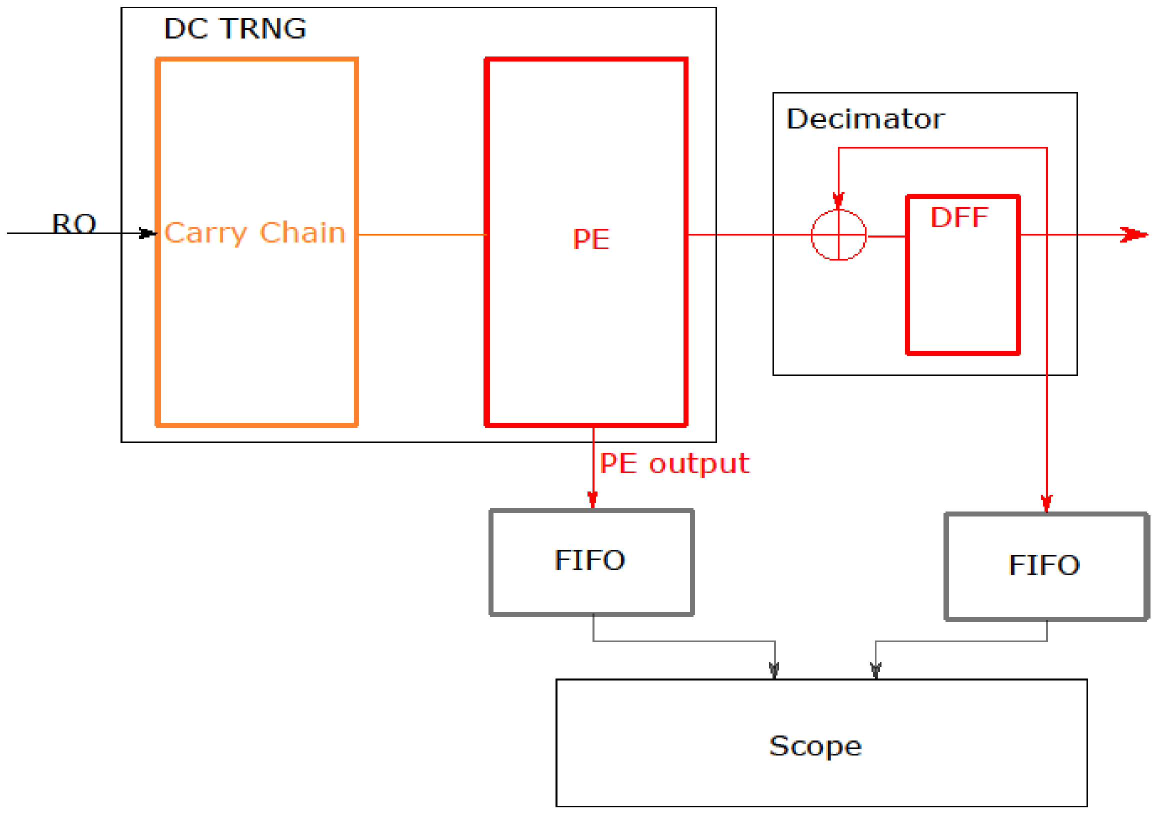


Figure 61: Design under test

Detecting occurrences of faults in the random number stream provided by a TRNG is not straightforward. The way we proceeded is as follows: the DC-TRNG was launched one thousand times with a considered probe injection position and injection time. Then the vectors of binary values collected at the output of the PE (synchronously with the clock) were averaged to detect the occurrence of stuck-at faults. Since we use AC coupling, the expected behavior is that the average vector is equal to 0, i.e. with an equal number of 1 and 0. Otherwise, if a stuck-at 0 fault is systematically induced at a given position and at a given time, a negative extremum will appear (resp. a positive extremum will appear for a stuck-at 1 fault). This procedure is illustrated Figure 6. In this figure the orange trace indicates the average vector. On these curves we can see that EMFI produces a systematic stuck at `0' fault at time sample 40,000. This is due to the presence of the FIFO delaying the observation of the fault injection effect.

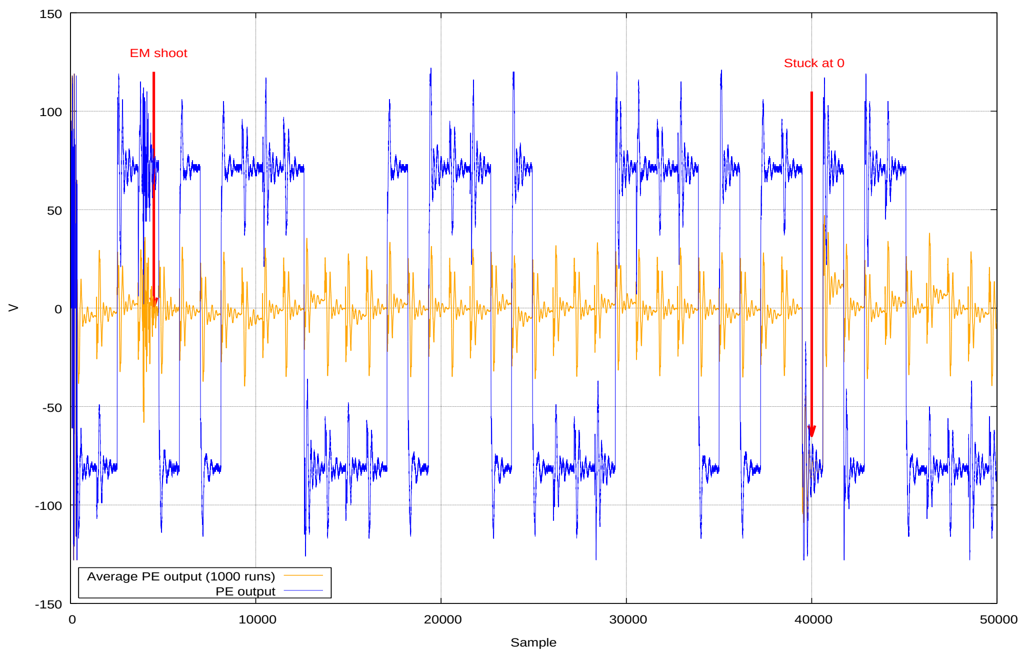


Figure 62: PE stuck at 0 on one bit (blue curves = 1 run, orange curves = average on 1,000 traces)

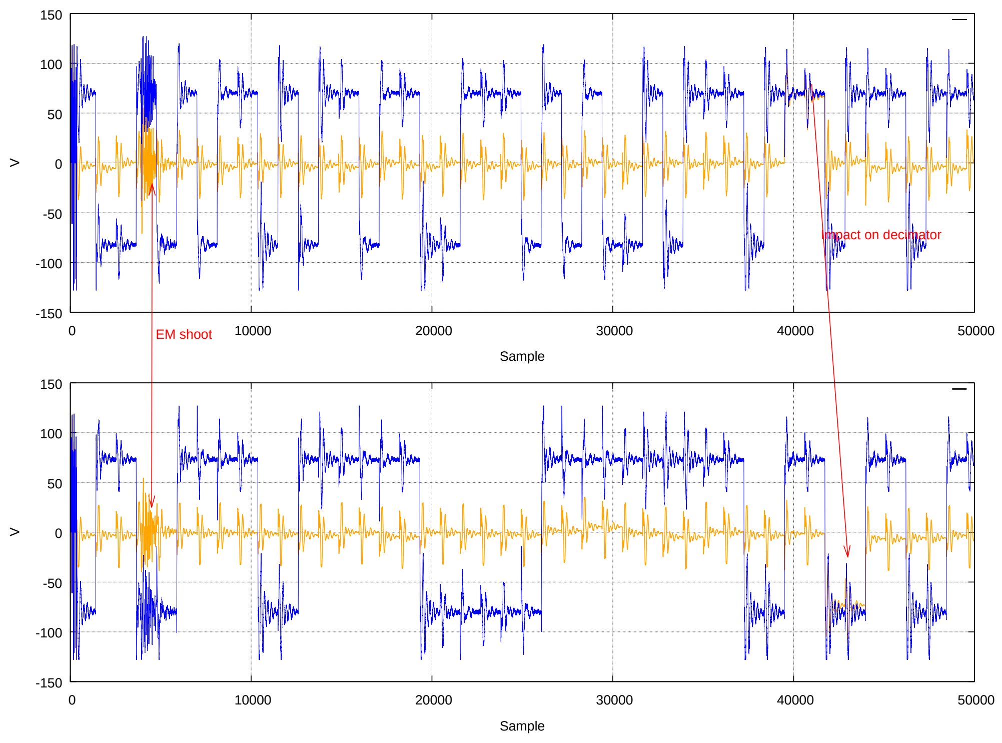


Figure 63: PE stuck at 11. Same color code as figure above. The upper graph represents the PE output and the bottom graph is the decimator output.

Using the above fault detection procedure, the impact of EMFI was analyzed at the output of both PE and Decimator. The tables below sum-up the test results and indicates the pulse parameters required to obtain particular types of faults at a specific position and at any time instant. Being able to draw such tables is a direct illustration that an attacker can fully – but temporarily – control the output of TRNGs. This partially supports the threat model introduced in the preceding subsection. Table 26 gives the required settings of the EM pulse to obtain different types of faults at the PE's output. However, at that point, for these settings, it is impossible to determine if the faults are induced in the PE or in the carry-chain. The content of the carry-chain was thus monitored in a last experiment.

Fault type	Amplitude	Pulse width	Delay
Stuck at 0	171 V	12.6 ns	1.12 ns
Stuck at 1	179 V	12.6 ns	1.12 ns

Table 25, EMFI parameters for Decimator faults

Fault type	Amplitude	Pulse width
Stuck at 0	292 V	6.45 ns
Stuck at 1	274 V	6 ns
Stuck at 00	356 V	7.7 ns
Stuck at 01	356 V	7.4 ns
Stuck at 10	314 V	11 ns
Stuck at 11	350 V	11 ns

Table 26, EMFI parameters for PE faults

This experiment focused on the effect of pulsed EMFI on the DC and PE, hence their streams were output to a serial port. Since this test enables to get DC and PE values the following approach to detect if a fault occurred or not has been chosen: The DC's output was used to re-compute the expected PE's output and then the two results were compared. This last experiment demonstrated that EMFI was able to induce a stuck-at fault in the PE for some probe positions without any effect on the carry chain, but also to disrupt the carry-chain's content for other positions. However, the faults induced in the carry-chain are uncontrollable because of the asynchronous operation with regards to the master clock of the free-running RO. Indeed, instead of having a vector like ...0000011111... that corresponds to the correct operation of the DC-TRNG, we obtained vectors like ...01...10..11...0..10110...1111 but with the position of the first `1' changing at each run. Therefore, the repeatability of the fault is not guaranteed.

To conclude, all these experiments validate a fault model where pulsed EM injection can stick a bit to a specific value. This fault model can be applied on either the PE or the decimator stages and has bit-accuracy. Yet in the particular case of the PE we were able to stick at a specific value of up to two bits using one injection as illustrated on Figure 63, with the example of sticking two bits at 1.

Since the fault affects two bits of PE's output at once, we can see its direct effect on the decimator stage.

8.4.4 Conclusion

These experiments highlight the fact that the entropy source is not the only entry point to induce bias in the random numbers flow delivered by a TRNG. Still today's strategy to harden TRNGs mainly consists in monitoring the statistical properties of the entropy sources. Nonetheless, when dealing with pulsed EMFI, exploitable faults seems to be more easily induced in digital post-processing stages than in the entropy source. Therefore it seems mandatory to also protect these processing blocks.

The above results show that with a single EM injection an attacker is only able to create a single bit (at best two consecutive bits in our case) to stick at a fixed and controllable value. For such attack to be exploitable this means that controlling the whole bit flow requires the use of an EMFI platform with a repetition rate at least equal to the throughput of the targeted TRNGs. Thus, high throughput is a desirable feature for TRNGs to counter pulsed EMFI attacks. The repetition rate of modern EMFI platforms is limited to only a few kHz.

Yet controlling the whole random bit flow is not the only threat. Indeed, controlling the whole random stream is not required to be able to lead to successful CPA against a masked implementation. Instead, in the case of CPA the bias introduced by pulsed EMFI on the TRNG should also be monitored. As pointed out in 8.4.2, a bias level of at least 10% is required to lead a successful CPA against an IP masking scheme while a bias level of 5% is required in the case of Boolean masking. To measure criticality of modern pulsed EMFI platforms against masking schemes relying on RO-based TRNGs we computed the bias induced for different running frequencies and different fault types. Moreover, we consider our EMFI platform to perform at its maximum repetition rate, i.e. 500 µs.

From the measurement reported in Figure 64 it is clear that – unless a DC-TRNG is running with a lower frequency than 170 kHz – pulsed EMFI does not constitute a major threat. As a reminder: during the experiment part the DC-TRNG was running at 4.5 MHz.

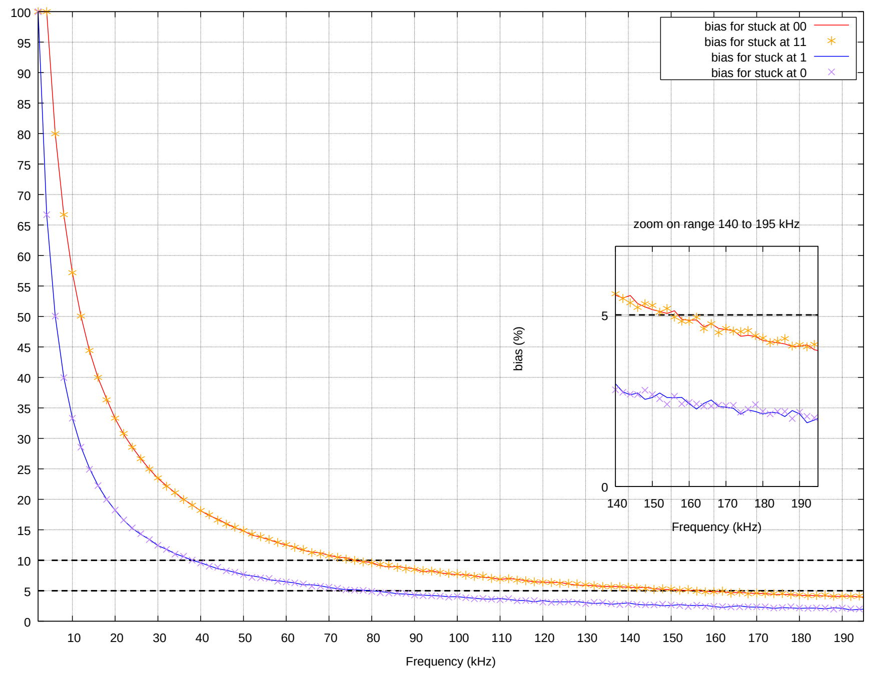


Figure 64: Fault impact on DC-TRNG's bias

However, it is likely that the repetition rate of EMFI platforms can be increased significantly. . The results of these tests show that adding detection to protect TRNGs from the effects of perturbation should not be viewed as a luxury solution in the future

Chapter 9 Summary and conclusion

Robustness evaluation should ideally be done on all products but it is clear that the effort can be prohibitive (resources, cost, time-to-market, etc.). By providing robust and easy-to-evaluate building blocks, this barrier can be lowered, for the benefit of industry in Europe and to provide better protection and security confidence for end-users.

This report summarized the activities that were done within Work Package 2 to verify the robustness of HECTOR developments (Objective O5), assess the compliance to standards and certification feasibility (Objective O6), as well as to ease and help prepare and asses the documentation (“developer evidences”) that will be needed to enable faster and efficient evaluations of end-products based on HECTOR security primitives (Objectives O1-O4 & O16).

Robustness testing has been performed to assess the behavior of HECTOR developments under varying environmental conditions, as well as their resistance against side channel analysis and perturbation attacks. Most experiments were executed during developments of HECTOR solutions. Results from experiments on FPGA implementations of HECTOR TRNGs and PUFs have been used as early feedback for improvements and to verify achievement of project targets: Robustness assessments of HECTOR TRNGs are good while the HECTOR TERO PUF remains sensitive to environmental variations. The physical modeling that we performed after selecting a TERO-based PUF principle allowed to understand and in hindsight predict why: The TERO cell presents very high sensitivity to process variations, which is a desirable property for PUF applications but it is also very sensitive to noise and environmental variations making it difficult to design a properly robust PUF. While the lack of robustness is a lowlight we consider the capability of our model-based approach as a highlight and confirmation that this could be the right way to evaluate the entropy and security of PUF designs moving forward. Due to repeated manufacturing delays (external factors) we have not yet received the HECTOR ASIC which we will only be able to analyze after the official completion of the project. On the other hand environmental testing on early prototypes of an ST automotive ASIC are showing a successful implementation of an HECTOR-TRNG in ASIC technology. This “first-time silicon success” is a testimony to the maturity of the HECTOR PLL-TRNG design methodology.

AIS20/31-compliance evaluations of HECTOR TRNGs have been performed, confirming certification feasibility. HECTOR’s PLL-TRNG is also compliant to NIST SP800-90B and the demanding specification from French defense (DGA-MI). The PLL-TRNG evaluation according to AIS20/31 and DGA MI is illustrating how model-based entropy evaluations and dedicated embedded tests derived from these models are enabling much more rigorous security demonstrations.

Besides the above evaluation results, these activities generated a lot of security evaluation and certification know-how transfer between BRT and the other partners, helping them prepare the proper set of documentation and “developer evidences” packages required to leverage HECTOR’s demonstrable security approach and enable faster security evaluations of future products integrating HECTOR security building blocks.

Chapter 10 List of Abbreviations

AIS	Anweisungen und Interpretationen im Schema
ASIC	Application Specific Integrated Circuit
BER	Bit Error Rate
BSI	Bundesamt für Sicherheit in der Informationstechnik
CC	Common Criteria
CCS	Combined Classification Success
CMOS	Complementary Metal Oxide Silicon
DC	Delay-Chain
CPA	Correlation Power Analysis
DEMA	Differential Electro Magnetic Analysis
DRNG	Deterministic Random Number Generator
EEPROM	Electrically Erasable Programmable Read Only Memory
EMFI	Electro Magnetic Fault injection
EM	Electro Magnetic
FFT	Fast Fourier Transformation
FPGA	Field Programmable Gate Array
HD	Hamming Distance
HDMI	High Definition Multimedia Interface
HW	Hamming Weight
IID	Independent and Identically Distributed
IP	Intellectual Property (silicon building block)
IP	Inner Product (masking scheme)
KAT	Known-Answer-Test
LM	Light Manipulation
LUT	Look-Up Table

NDA	Non-Disclosure Agreement
NIST	National Institute of Standards and Technology
NVM	Non-Volatile Memory (e.g. EEPROM or Flash)
MPV	Multiple Process Variations
OCCS	Overall Combined Classification Success
PCB	Printed Circuit Board
PE	Priority Encoder
PLL	Phase Locked Loop
PUF	Physically Unclonable Function
SCA	Side Channel Analysis
RF	Radio Frequency
RO	Ring Oscillator
SEMA	Simple Electro Magnetic Analysis
SPA	Simple Power Analysis
ST	Security Target
TA	Template Attack
TERO	Transient Effect Ring Oscillator
TOE	Target Of Evaluation
TRNG	True Random Number Generator
VA	Vulnerability Analysis
XOR	Exclusive OR function

Chapter 11 Bibliography

[1] AIS 20/31 A proposal for: Functionality classes for random number generators, Version 2.0 18 September 2011

[2] NIST Special Publication (SP) 800-90B, Recommendation for the Entropy Sources Used for Random Bit Generation, NIST, January 2018, https://doi.org/10.6028/NIST.SP.800-90B.

[3] Report on PTG.3 requirement for Hector TRNG UJM, Brightsight report 17-RPT-081 version 3.0, 24 July 2017 (confidential)

[4] C. Böhm and M. Hofer. Physical Unclonable Functions in Theory and Practice. Springer New York, 2012.

[5] H. Kang, Y. Hori, T. Katashita, M. Hagiwara, and K. Iwamura. Cryptographic key generation from PUF data using efficient fuzzy extractors. In 16th International Conference on Advanced Communication Technology, pages 23–26, Feb 2014.

[6] Stefan Katzenbeisser, Ünal Kocabas, Vladimir Rozˇic´, Ahmad-Reza Sadeghi, Ingrid Verbauwhede, and Christian Wachsmann. Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, chapter PUFs: Myth, Fact or Busted? A Security Evaluation of Physically Unclonable Functions (PUFs) Cast in Silicon, pages 283–301. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[7] S. Lattacher, M. Deutschmann, M. Höberl, J. Delvaux, J. Balasch, and H. Bock. Appendix to D2.2 - Optimized post-processing methods and techniques. http://hint-project.technikon.com/, 2015. HINT Project.

[8] Robbert van den Berg. Entropy analysis of physical unclonable functions. PhD thesis, 2012.

[9] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, “Physical one-way functions,” Science, vol. 297, no. 5589, pp. 2026–2030, 2002.

[10] Y. Su, J. Holleman, and B. Otis, “A digital 1.6 pJ/bit chip identification circuit using process variations,” IEEE Journal of Solid-State Circuits, vol. 43, no. 1, pp. 69–77, Jan 2008.

[11] G. E. Suh and S. Devadas, “Physical unclonable functions for device authentication and secret key generation,” in Proceedings of the 44th Design Automation Conference, DAC, San Diego, CA, USA, June 4-8 2007, pp. 9–14.

[12] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, “Silicon physical random functions,” in Proceedings of the 9th ACM Conference on Computer and Communications Security, ser. CCS ’02. New York, NY, USA: ACM, 2002, pp. 148–160.

[13] A. Maiti, V. Gunreddy, and P. Schaumont, A Systematic Method to Evaluate and Compare the Performance of Physical Unclonable Functions. New York, NY: Springer New York, 2013, pp. 245–267.

[14] M. Pehl, M. Hiller, and H. Graeb, "Efficient evaluation of physical unclonable functions using entropy measures," Journal of Circuits, Systems and Computers, vol. 25, no.01, p. 1640001, 2016.

[15] A. Cherkaoui, L. Bossuet, and C. Marchand, "Design, evaluation, and optimization of physical unclonable functions based on transient effect ring oscillators," IEEE Transactions on Information Forensics and Security, vol. 11, no. 6, pp. 1291–1305, June 2016.

[16] V. Fischer, "A Closer Look at Security in Random Number Generators Design," Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 167–182.

[17] S. W. Killmann, W., "A proposal for: Functionality classes for random number generators," 2011.

[18] Patrick Haddad, Viktor Fischer, Florent Bernard, Jean Nicolai. A Physical Approach for Stochastic Modeling of TERO-based TRNG. Workshop on Cryptographic Hardware and Embedded Systems, CHES 2015, Sep 2015, st-malo, France. 2015.

[19] T. Chawla, "Etude de l'impact des variations du procédé de fabrication sur les circuits numériques," Ph.D. dissertation, Télécom ParisTech, 30 Septembre 2010.

[20] M. Pelgrom, A. C. Duinmaijer, and A. Welbers, "Matching properties of MOS transistors," Solid-State Circuits, IEEE Journal of, vol. 24, no. 5, pp. 1433–1439, Oct 1989.

[21] L. M. Reyneri, D. D. Corso, and B. Sacco, "Oscillatory metastability in homogeneous and inhomogeneous flip-flops," IEEE Journal of Solid-State Circuits, vol. 25, no. 1, pp. 254–264, Feb 1990.

[22] Demonstrator 1 (DC TRNG) – Stochastic Model, 28 June 2017

[23] Developer evidence for the evaluation of a physical true random number generator, Document-Version 0.8, 28-02-2013

[24] Evaluator methodology for PTRNG evaluation, Document-Version 0.7, 28.02.2013

[25] Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model, September 2012, version 3.1, revision 4, CCMB-2012-09-001

[26] Common Criteria for Information Technology Security Evaluation, Part 2: Security functional components, September 2012, version 3.1, revision 4, CCMB-2012-09-002

[27] Common Criteria for Information Technology Security Evaluation, Part 3: Security assurance components, September 2012, version 3.1, revision 4, CCMB-2012-09-003

[28] Common Criteria for Information Technology Security Evaluation, Evaluation methodology, September 2012, version 3.1, revision 4, CCMB-2012-09-004

[29] JIL Attack methods for smart cards and similar devices, version 2.2, January, 2013

[30] JIL Application of Attack Potential to Smart cards, Version 2.9, January 2013

Appendix A Environments for testing (BRT)

Light perturbation setups

This appendix describes the setups that are being used by Brightsight as tools for security evaluations.

Description

Figure 65 and Figure 66 show schematic representations of the light manipulation measurement set-ups of LM1/LM3/LM5/LM7 and LM4/LM6/LM8, respectively. The set-ups LM1, LM3, LM5 and LM7 use a laser cutter module, which is able to produce short (4-7ns) laser bursts. The set-ups LM4, LM6 and LM8 use solid-state lasers and are therefore able to produce laser bursts of infinite duration. The minimum laser burst duration of the solid-state laser depends on the bandwidth of the used laser module.

The TOE is placed into a custom designed ‘daughterboard’ mounted on a ‘motherboard’’. Two function generators are connected to the motherboard. One function generator is used to generate the necessary 3.57MHz clock signal and the other is used to perform a cold reset. An oscilloscope is used to monitor the (filtered) power consumption and triggers a third function generator, which is used to trigger the laser module (with an adjustable delay). All the signals connected to the TOE are routed through the motherboard, which, for LM1, LM3, LM5 and LM7, is mounted on a XY-stage to target the laser. In the LM4, LM6 and LM8 set-ups, the solid-state laser itself (and not the motherboard) is mounted on an XYZ stage.

In case the TOE has active countermeasures that render it inoperable on detection of manipulation attempts, a second oscilloscope will be added to the set-up. In most cases an EEPROM or Flash (erase/)write operation is required to render a device inoperable. Using the second oscilloscope it is possible to detect this EEPROM or Flash (erase/)write operation and instantly perform a cold reset. A cold reset will interrupt the erase/write operation, which will leave the device operable.

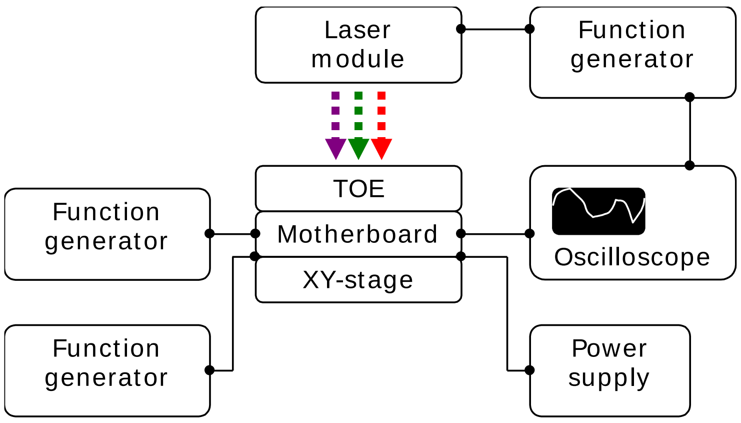


Figure 65: Schematic representation of the LM1, LM3, LM5 and LM7 set-ups (contact mode).

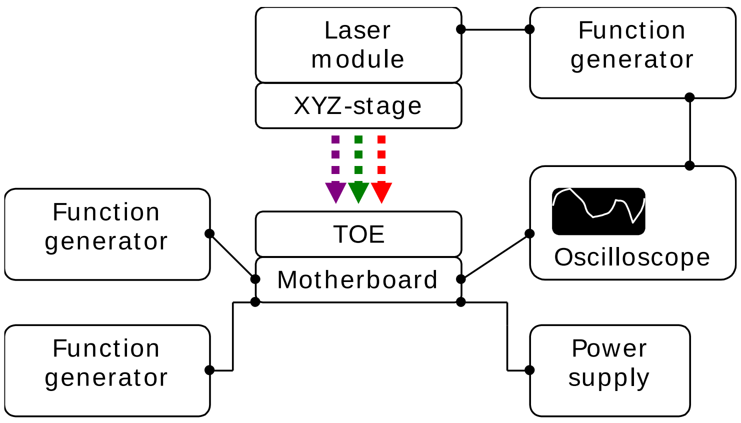


Figure 66: Schematic representation of the LM4, LM6 and LM8 set-ups (contact mode).

Components

Table 27 shows the components of each light manipulation set-up that is available at Brightsight (contact mode).

Set-up	Description	Manufacturer	Type
LM1	Function generator (laser trigger)	Agilent	33250A
	Function generator (clock and cold reset)	Agilent	33522A
	Power supply	Agilent	E3640A
	Primary oscilloscope	LeCroy	WaveSurfer 24MXs-B
	Secondary oscilloscope	LeCroy	9354AL
	Laser	New Wave	EzLaze II Trilite
	XY-stage	Märzhäuser	L-Step 12/2
	Motherboard	Brightsight	LM-Motherboard-USB-4.1, Version 3.9.1
LM3	Function generator (laser trigger)	Agilent	33250A
	Function generator (clock)	Agilent	33220A
	Function generator (cold reset)	Agilent	33250A
	Power supply	Agilent	E3640A
	Primary oscilloscope	LeCroy	WaveSurfer 24MXs-B
	Secondary oscilloscope	LeCroy	WaveSurfer 24MXs-B
	Laser	New Wave	QuikLaze 1064/532
	XY-stage	Märzhäuser	Tango 2
	Motherboard	Brightsight	LM-Motherboard-USB-4.1, Version 3.9.1
LM4	Function generator (laser trigger)	Agilent	33250A
	Function generator (clock)	Rigol	DG1022
	Function generator (cold reset)	Agilent	33220A

Set-up	Description	Manufacturer	Type
	Power supply	Agilent	E3640A
	Primary oscilloscope	LeCroy	WaveSurfer 24MXs-B
	Secondary oscilloscope	LeCroy	Waverunner LT372L
	Laser	AlphaNOV	PDM-1064 IR laser (1064nm)
		Brightsight	Blue laser 445nm
	XY-stage	Newport	M-462 series
	Motherboard	Brightsight	LM-Motherboard-USB-4.1, Version 3.9.1
LM5	Function generator (laser trigger)	Agilent	33250A
	Function generator (clock and cold reset)	Agilent	33522A
	Power supply	Agilent	E3631A
	Primary oscilloscope	LeCroy	WaveSurfer 24MXs-B
	Secondary oscilloscope	LeCroy	Waverunner LT342
	Laser	New Wave	Dual EzLaze III 1064
	XY-stage	Märzhäuser	L-Step 12/2
	Motherboard	Brightsight	LM-Motherboard-USB-4.1, Version 3.9.1
LM6	Function generator (laser trigger)	Agilent	33250A
	Function generator (clock)	Rigol	DG1022
	Function generator (cold reset)	Agilent	33220A
	Power supply	Agilent	E3640A
	Power supply (laser power)	Agilent	E3640A
	Primary oscilloscope	LeCroy	WaveSurfer 24MXs-B
	Secondary oscilloscope	LeCroy	WaveSurfer 24MXs-B
	Laser	AlphaNOV	PDM-1064 IR laser (1064nm)
		Brightsight	Blue laser 445nm
	XY-stage	Newport	M-462 series
	Motherboard	Brightsight	LM-Motherboard-USB-4.1, Version 3.9.1
LM7	Function generator (laser trigger)	Agilent	33522B
	Function generator (clock and cold reset)	Agilent	33522B
	Power supply	Agilent	E3631A
	Primary oscilloscope	LeCroy	WaveSurfer 24MXs-B
	Secondary oscilloscope	LeCroy	WaveSurfer 24MXs-B
	Laser	New Wave	Dual EzLaze III 1064
	XY-stage	Märzhäuser	Tango 2
	Motherboard	Brightsight	LM-Motherboard-USB-4.1, Version 3.9.1

Set-up	Description	Manufacturer	Type
LM8	Function generator (laser trigger)	Agilent	33522B
	Function generator (clock and cold reset)	Agilent	33522B
	Power supply	Agilent	E3631A
	Power supply (laser power)	Agilent	E3631A
	Primary oscilloscope	LeCroy	HDO4024
	Secondary oscilloscope	LeCroy	HDO4024
	Laser	AlphaNOV	PDM-1064 IR laser (1064nm)
		Brightsight	Blue laser 445nm
	XY-stage	Newport	M-462 series
	Motherboard	Brightsight	LM-Motherboard-USB-4.1, Version 3.9.1

Table 27: Measurement set-up components.

Laser parameter information

The manner in which the energy of the laser pulse is configured on the various laser set-ups differs depending on the laser module used. Table 28 gives an overview how the settings are done for each set-up.

Set-up	Laser Module	Laser Energy	Aperture
LM1, LM3, LM5, LM7	All laser cutter modules	Proprietary units: Two ranges "Lo(w)" and "Hi(gh)" Units running from 0 (lowest setting) to 255 (highest setting)	Proprietary units for X and Y: Units for X/Y running from 0 (lowest setting) to 255 (highest setting)
LM4, LM6, LM8	AlphaNOV IR laser	The laser energy is determined using a voltage that is applied to the laser module. Range: 0-5 V (5 V → max. energy)	Fixed aperture
	Brightsight Blue Laser	The laser energy is determined using a voltage that is applied to the laser module. Range: 4-6 V (~0-100 % energy)	

Table 28: Laser Parameter Information.

Side channel set-ups

SPA/DPA set-ups

The SPA/DPA measurement set-up is used to measure the power consumption profile of a smart card or smart card chip. The TOE can be inserted (with or without external connector board) in the TOE interface. Figure 67 shows a schematic representation of the set-up.

The ground pin of the TOE is connected to either a 50 Ω resistor or the 50 Ω input impedance of the oscilloscope. The oscilloscope is used to digitise the voltage across this impedance. This signal is referred to as the power consumption profile. Not only the power consumption profile, but also the filtered power consumption profile and the IO communication signals are measured with the oscilloscope. These signals are useful to identify the parts of interest in the power consumption profile.

The oscilloscope can be triggered by the I/O signal, a specific pattern in the power consumption profile, a software trigger signal generated by the SPA/DPA interface or a combination (smart trigger). A PC is connected to the SPA/DPA interface and oscilloscope to control the commands send to the TOE and to collect the measured power consumption profiles.

The function generator is used to generate a 3.57 MHz clock signal with adjustable amplitude and offset. The power supply is used to power the chip with an adjustable voltage. A low voltage often improves the results of SPA/DPA, but the TOE will be inoperable when the input voltage is too low.

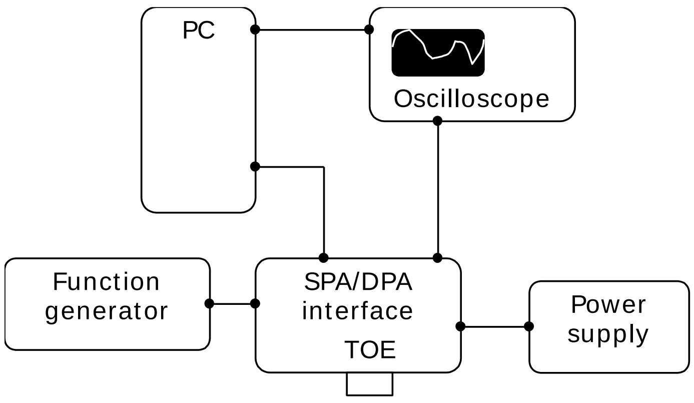


Figure 67: Schematic representation of the set-up (non-contactless).

The following table shows for a DPA set-up the components of which it consists:

Setup ID	Description	Manufacturer	Serial number	App ID
DEMAP1				
	Power supply	Tenma	72-8695	PS 50
	Oscilloscope	Lecroy	WS24MXS-B	SCOOP 21
DPA1				
	Function generator	Agilent	33120A	CLK 01
	Power supply	Agilent	E3631A	PS 02
	Oscilloscope	Lecroy	WR620ZI	SCOOP 43
	DPA Card Reader	Brightsight	-	SN001
DPA2				
	Function generator	Agilent	33120A	CLK 02
	Power supply	Agilent	E3631A	PS 60

Setup ID	Description	Manufacturer	Serial number	App ID
	Oscilloscope	Lecroy	WR620ZI	SCOOP 36
	DPA Card Reader	Brightsight	-	SN010
DPA4				
	Function generator	Agilent	33250A	CLK 21
	Power supply	Agilent	E3631A	PS 38
	Oscilloscope	Lecroy	WR620ZI	SCOOP 23
	DPA Card Reader	Brightsight	-	SN011
DPA5				
	Function generator	Rigol	DG1022	CLK 20
	Power supply	Tenma	72-8695	PS 40
	Power supply	Agilent	E3631A	PS 43
	RF Synthesizer	Hameg	HM8135	RF-SYNTH-1
	Spectrum Analyser	Hameg	HM5014-2	SPECTRUM 1

Table 29: Measurement set-up components.

SEMA/DEMA set-ups

The DEMA set-ups in Brightsight are capable of measuring electro-magnetic signal and power signal simultaneously or independently.

The set-up for measuring the electro-magnetic side channel on contact secure micro controllers or smart cards is placed inside a Faraday cage. The electro-magnetic signals emanated from the surface of the TOE can be measured with minimal influence from external electro-magnetic sources (e.g. GSM phone signals, contactless smart card readers, etc.). A schematic view of the set-up is shown in the figure below in Figure 68.

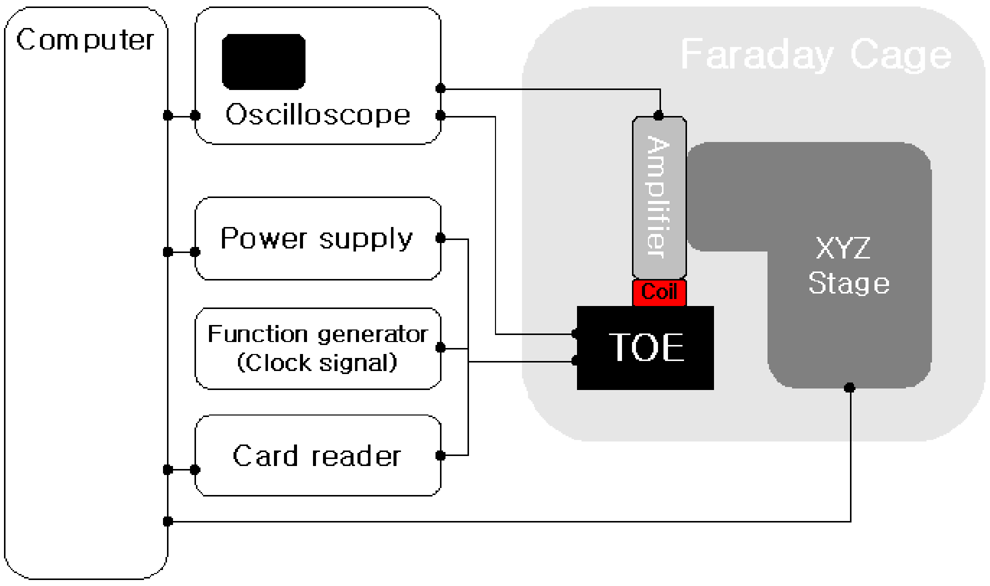


Figure 68: Schematic representation of the set-up (contact) for measuring electro magnetic side channels.

The TOE is connected to the computer through a card reader. The power supply is used to power the TOE with an adjustable voltage. A function generator supplies the TOE with a clock signal with an adjustable amplitude, offset and frequency.

The pickup coil used to measure the EM emanation is connected to an amplifier mounted on an XYZ stage. The amplifier is used to amplify the signals that are picked up by the coil. The XYZ stage can be used to automatically scan the surface of the TOE to find an interesting location to measure a larger set of EM traces for the differential analysis. The oscilloscope is used to digitise the amplified signals picked up by the coil.

The oscilloscope can be triggered by a specific pattern in the EM emanation profile or power consumption profile, IO signal, a software trigger signal generated by the card reader or a combination (smart trigger). A PC is connected to the card reader and oscilloscope to control the commands sent to the TOE and to collect the measured power and EM emanation profiles.

The power supply is used to power the chip with an adjustable voltage. A low voltage often improves the results of SPA/DPA, but the TOE will be inoperable when the input voltage is too low.

For signal processing (e.g. performing alignment, DPA/DEMA analysis as well as key search operations) dedicated Brightsight software is used.

For power measurement using the DEMA set-up, the same set of equipment and software are used. Table 30 shows the components of DEMA setups.

Setup ID	**Description**	**Manufacturer**	**Serial number**	**App ID**
DEMAP1				
	Power supply	Tenma	72-8695	PS 50
	Oscilloscope	Lecroy	WS24MXS-B	SCOOP 21
EM1				

Setup ID	Description	Manufacturer	Serial number	App ID
	Function generator	HP	33120A	CLK 04
	Power supply	Agilent	E3631A	PS 05
	Oscilloscope	Lecroy	WR620ZI	SCOOP 35
EM2				
	Function generator	Agilent	33220A	CLK 17
	Power supply	Agilent	E3631A	PS 42
	Oscilloscope	Lecroy	WR620-ZI	SCOOP 24
EM3				
	Function generator	Agilent	33220A	CLK 08
	Power supply	Agilent	E3631A	PS 36
	Oscilloscope	Lecroy	WR620ZI	SCOOP 28
EM4				
	Function generator	Agilent	33522B	CLK 31
	Power supply	Agilent	E3631A	PS 49
	Oscilloscope	Lecroy	WR620ZI	SCOOP 31
EM5				
	Function generator	Agilent	33522A	CLK 26
	Power supply	Agilent	E3631A	PS 57
	Oscilloscope	Lecroy	WR620ZI	SCOOP 27

Table 30: Measurement set-up components.

Template attack method

Introduction to the template attack environment

The Brightsight template attack environment allows a measurement set to be used for both the construction of templates and for subsequent testing of those templates, by simply using the first *N* traces out of each class for the creation of the templates, and then, after all templates have been created, attempting to classify the *M* subsequent traces (‘challenges’). Both *N* and *M* are user-defined. The Brightsight template attack environment also allows two separate measurement sets to be used for the construction of templates and for testing of those templates respectively. Depending on an attack scenario, it can be decided which approach will be used.

In order to determine the success rate of the template attack, several metrics can be calculated:

1. The *overall success rate*, that is, the overall percentage of individual challenge traces which were correctly classified. For the classification, the regular ‘maximum likelihood’ method is used, calculating a score representing the likelihood that the challenge trace belongs to the probability distribution defined by each of the templates, and then determining the highest score, and deciding the candidate is matched to the corresponding template. This metric is essentially the mean conditional probability of good classification, over all possible values of the secret parameter.
2. The *per-candidate worst case success rate*, that is, out of all possible candidate values c_i, the highest value of the success rate for classification of challenges corresponding to only c_i. (a per-candidate best case success rate can be defined accordingly, but is less relevant as a metric). This metric is essentially the maximum of the conditional probability of good classification, over all possible values of the secret parameter.
3. The *combined classification success rate*, that is, the success rate of a classification process which combines multiple challenge traces into a single classification as follows: all traces related to a single candidate value c_i are compared to all templates T_i. For each of those comparisons, the calculated likelihood is stored. After this, the likelihood of all comparisons to template T_i are combined to obtain an aggregated score S_i. The classification of the set of traces is established to be the candidate value for which S_i is the highest. This is repeated for the challenge sets corresponding to all candidate values c_i. Thus, in the example with 256 classes (or candidate values), 256 combined classifications would be calculated. The percentage of those that are correct is the *combined classification success* rate.
4. (Optional) The O*verall Combined Classification Success rate (OCCS)*, that is, the overall percentage of individual sets which were correctly classified. The only difference with the combined classification rate is that multiple sets of challenge traces per class are used. As it takes much more traces and processing times, it is not always applicable. For example, if a combined classification is computed with 500 traces per each class, the number of classes is 256, and the number of experiments for computing the overall combined classification success rate is 100, the total number of challenge traces would be 500x256x100 = 12.8 M traces.
5. (Optional) The *worst-case Combined Classification Success rate (CCS)*, is the highest combined classification success rate among several individual combined classification success rates. Similarly the *best-case Combined Classification Success rate* is the lowest one.

Template size

The toolset calculates each of the metrics, for a number of different template sizes (*a.k.a.* number of interesting points). Usually, a set of templates that work is characterised by a growth in the success rate as the template size grows, until over-training occurs and the success rate starts to decrease. A non-functional set of templates will show success rates more or less equal to the likelihood of correct random guessing.

Prior and posterior probabilities

The toolset allows calculating either prior or posterior probabilities. Prior probability is the probability that a challenge from a given class is classified as belonging to that class. Posterior probability is the probability that a challenge which has been classified as belonging to a class actually belongs to that class.

As an illustration, imagine the following scenario: two classes *A* and *B* exist, each equally likely to occur in challenges, but the classification result classifies any challenge as belonging to class *A*, except one in every 100 challenges belonging to *B* is correctly classified as *B*. In this situation, the prior probability for class *B* is only 1%, but its posterior probability is 100%.

The posterior probability is considered more representative for a real attack scenario as the attacker has no knowledge of the correct result before the attack.

Classes: templates for value vs. Hamming Weights

When performing a template attack, typically the target value is some value internal to a product. Popular examples are key segments as they are transported over internal data buses or intermediate values from cryptographic algorithms.

Given the way most embedded hardware works, it is reasonable to attack not the value itself, but rather it's Hamming Weight. In many contexts, knowledge of the Hamming Weight only already poses sufficient threat that the product should be considered compromised.

A frequent application is the attack of the Hamming Weight of a byte. In this scenario, nine classes are present, and although it is reasonable to use an equal amount of training and challenge traces for each of the nine classes, in an actual attack these classes do not have equal probability of occurring. The toolset contains logic to properly calculate success probabilities in this application.

Interpretation of template attack results

After executing the toolset, a figure showing several success rates is generated, for example, as shown in Figure 69. The x-axis shows the template size (*a.k.a.* number of interesting points) and the y-axis represents the success rate, which takes value from 0 to 1. Success rates can be computed either by prior probability or posterior probability. On top of the figure, the numbers of the training and the challenge traces per class are shown with the number of classes.

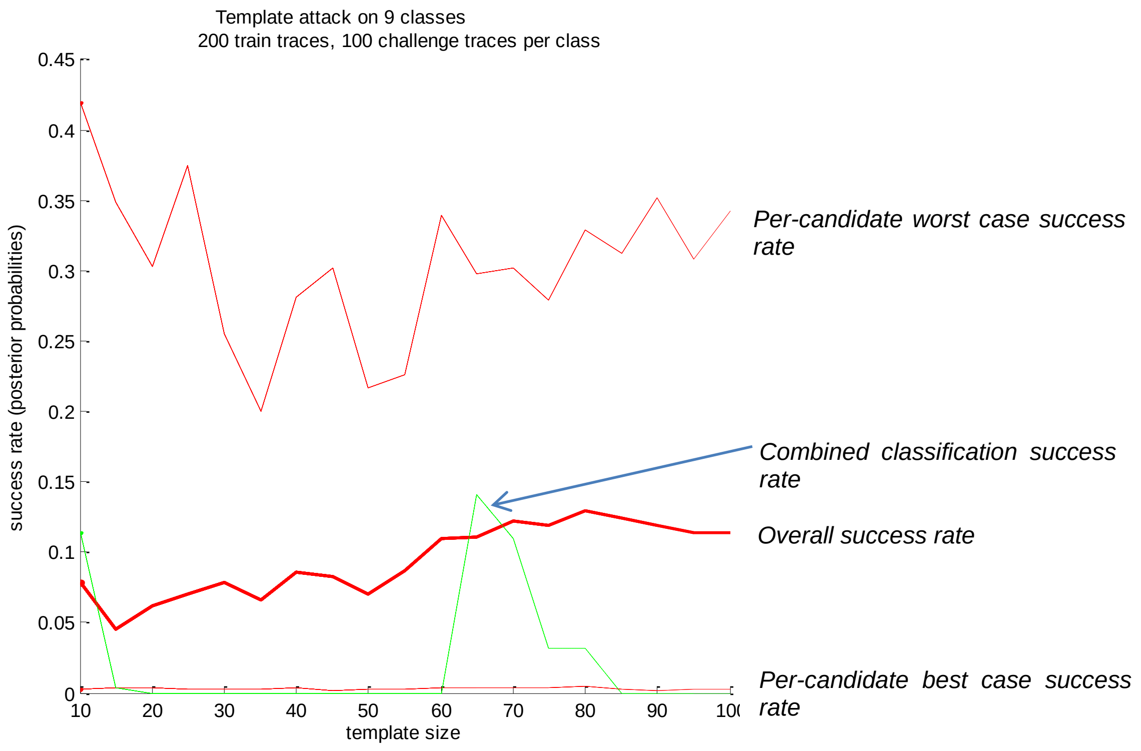


Figure 69: An example of template attack results.

Interpretation of success rates:

1. The *overall success rate*, depicted with a thick red line, shows the average success probability when an attacker uses a single challenge trace to find the secret. This is useful in the following attack scenario: an attacker first generates templates using *N* traces per each class. Then the attacker decides which candidate value is in the target device by measuring only single challenge trace from the target device and then performing the matching process. Note that this is the canonical model in which template attacks were first introduced. In many cases, this is not expected to give a better result than the combined classification success rate. However, in certain cases, for example, when a target device randomizes its processing per iteration, the combined classification success rate does not represent a realistic attack scenario.
2. The *per-candidate worst case success rate,* depicted with an upper thin red line, shows the success rate for the class that can be identified best[8]. For example, if the success rate for the class of Hamming weight 0 (it is assumed that 9 Hamming weights are used to generate templates) reaches almost to 1, it implies that the toolset can almost always identify the secret when the secret is 0.
3. The *combined classification success rate*, depicted with a dashed green line, shows the average success probability when an attacker uses multiple challenge traces as a set to find the secret. An attacker first generates templates using *N* traces per each class then decides which candidate value is in the target device by measuring *M*

[8] Note that this may not be the same class for each of the template sizes considered

traces from the target device and performing the matching process using all *M* traces as a set.

4. (Optional) The O*verall Combined Classification Success rate (OCCS)*, depicted with a dashed green line, shows a mean of the Combined Classification Success rate (CCS) based on the results of the multiple experiments.
5. (Optional) The *worst-case combined classification success rate*, depicted with a dashed blue line, shows the highest CCS among multiple experiments per each template size.

Figure 70 shows an example of template attack results when overall, worst case, and best case combined classification success rates are used.

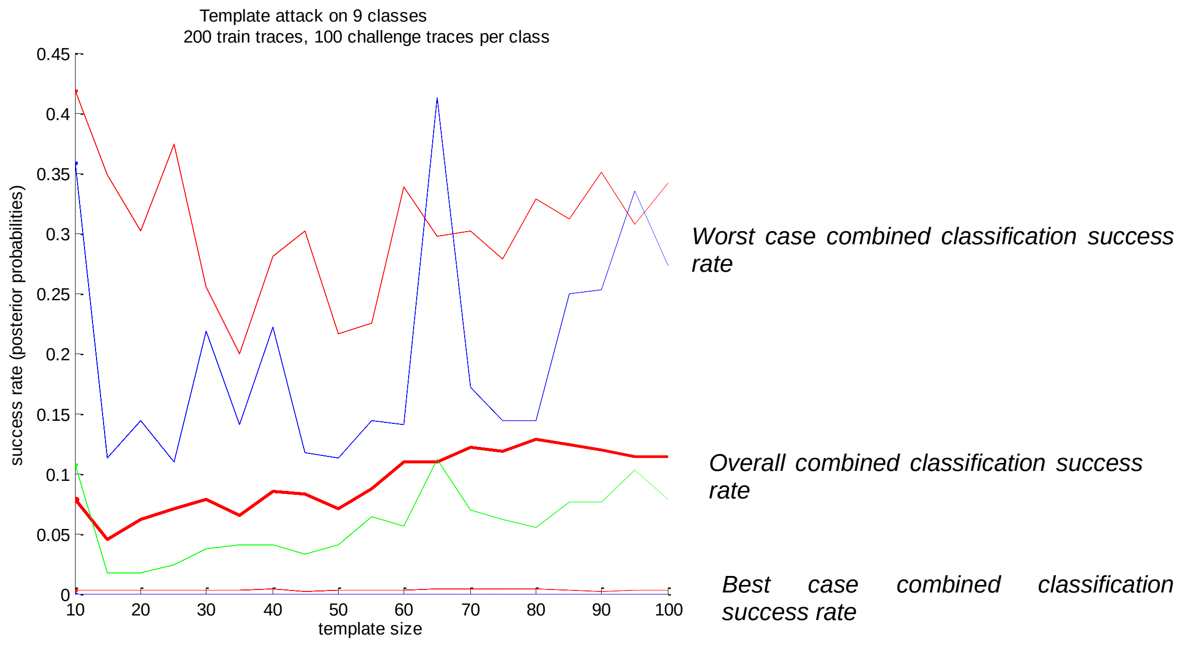


Figure 70: An example of template attack results with optional overall, worst case, and best case combined classification success rates.

Appendix B Definitions

Definition 1. Hamming Distance: The Hamming distance between two PUF responses *r1* and *r2* of length *K* is the number of positions at which the bits are different. The hamming distance is denoted by HD(*r1; r2*) and can be calculated by

$$\mathsf{HD}(r_1, r_2) := \sum_{k=1}^{K} r_1(k) \oplus r_2(k), \tag{1}$$

where *r(k*) is the bit on position *k* of response *r*.

Definition 2. Bit Error Rate: A common indicator to characterize the stability (reliability or robustness) of a PUF response is the average bit error rate BER respectively the intra-device distance $\mathrm{HD}^{\mathrm{intra}}$. It specifies the average number of bit errors of a sample of N responses rn for 1 ≤ *n* ≤ N, in comparison to a reference response r_{ref} . Let *K* be the number of bits in one response, then the bit error rate is defined as follows:

$$\mathsf{BER} := \mathsf{HD}^{\mathrm{intra}} := \frac{1}{N \cdot K} \sum_{n=1}^{N} \mathsf{HD}(r_{\mathrm{ref}}, r_n). \tag{2}$$

Definition 3. Hamming Weight: The Hamming weight is the relative number of ones in a binary string. So the Hamming weight HW*(r)* of a PUF response *r* is defined as the sum of all *K* response bits, divided by the number of bits *K*:

$$\mathsf{HW}(r) := \frac{1}{K} \sum_{k=1}^{K} r(k). \tag{3}$$

Definition 4. Inter-device distance: The PUF responses from two different devices should not be correlated and must differ with high probability. The similarity between two PUF instantiations i and j can be described via the inter-device distance $\mathrm{HD}^{\mathrm{inter}}$ which is the distance between a defined reference response r^{i}_{ref} of the first PUF instantiation i and a sample of N responses r^{j}_{n} for 1 ≤ n ≤N; of the second PUF instantiation j. Let *K* be the number of bits in a response, then the inter-device distance is defined as

$$\mathsf{HD}^{\mathrm{inter}} := \frac{1}{N \cdot K} \sum_{n=1}^{N} \mathsf{HD}(r^{\mathrm{i}}_{\mathrm{ref}}, r^{\mathrm{j}}_{\mathrm{n}}). \tag{4}$$

Definition 5. Failure Rate: The failure error rate describes the probability that a string of *n* bits has more than *t* errors and can be calculated via the binomial distribution, given by

$$P_{\mathrm{fail}} = \sum_{i=t+1}^{n} \binom{n}{i} p_{\mathrm{b}}^{i} (1-p_{\mathrm{b}})^{n-i} = 1 - \sum_{i=0}^{t} \binom{n}{i} p_{\mathrm{b}}^{i} (1-p_{\mathrm{b}})^{n-i}. \tag{5}$$

p_b is the probability that on bit flips and can be approximated by the bit error rate BER.

Definition 6. Autocorrelation: The autocorrelation of a bit string r is an indicator for the similarity of the bit string with a circular shifted copy of itself. If every 0 in the bit string is replaced by -1, then the autocorrelation *A* is given by

$$A(j) := \sum_{k=1}^{K} r(k)r(k-j), \tag{6}$$

where *A* is evaluated at lag *j*. If *A(j)* = ±1, the data is completely correlated at lag *j*, and *A(j)* = 0 indicates an absolutely uncorrelated bit string [1].

Appendix C Results of mother board evaluation

As the final demonstrator will be based on the HECTOR mother board, it is important to know if the behaviour of mother boards is similar to that of the daughter boards. Therefore also mother board data provided by HECTOR partners was analysed in a similar way. The data was read out at room temperature (which had not been standardized before) and the format of the data is the same as described in the abstract: For each board 5,000 responses, each response 128 bit.
In comparison to the results we got for daughter boards regarding bit error rate, failure probability, inter-device distances and autocorrelation, the results of the mother board evaluation are quite similar. For details please refer to the following charts and tables. Table 31 and Table 32 contain bit error rates and failure probabilities and Figure 71 gives an overview of the results. Figure 73 shows the inter-device distances between mother boards, evaluated in the same way as in 6.3. Figure 72 shows the autocorrelation of mother boards for all possible lags. Remarkable here is that in 7 out of 8 cases the highest autocorrelation is measured at lag 1 again (see 6.4.1).

Source ID	Bit Error Rate	Hamming Weight	Pfail
BRT 1	3.76 %	47.41 %	1.00 %
BRT 2	4.64 %	47.32 %	2.02 %
KUL	7.68 %	49.95 %	9.53 %
MIC	2.71 %	51.52 %	0.32 %
STI	7.84 %	46.86 %	10.13 %
TCS 1	4.92 %	52.07 %	2.46 %
TCS 2	2.62 %	43.98 %	0.28 %
TEC	4.92 %	42.49 %	2.47 %

Table 31: Intra distance results for unprocessed mother board data

Source ID	Bit Error Rate	Hamming Weight	Pfail
1	1.25 %	46.43 %	1.79e04
2	2.27 %	46.39 %	1.67e03
3	3.74 %	49.95 %	9.87e03
4	0.12 %	53.00 %	1.65e08
5	2.04 %	46.53 %	1.13e03
6	0.61 %	51.23 %	1.11e05
7	0.68 %	43.80 %	1.71e05
8	1.83 %	42.58 %	7.54e04

Table 32: Intra distance results for processed mother board data without dark bits.

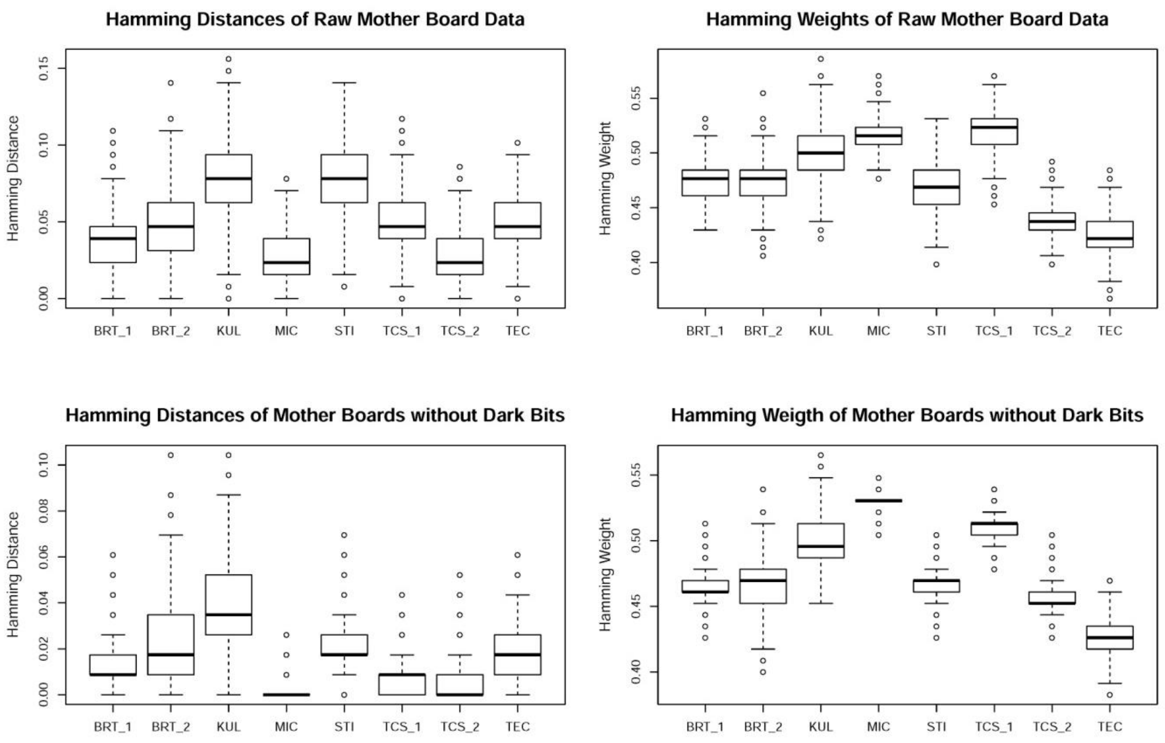


Figure 71: Hamming Distances and Hamming Weights of mother boards

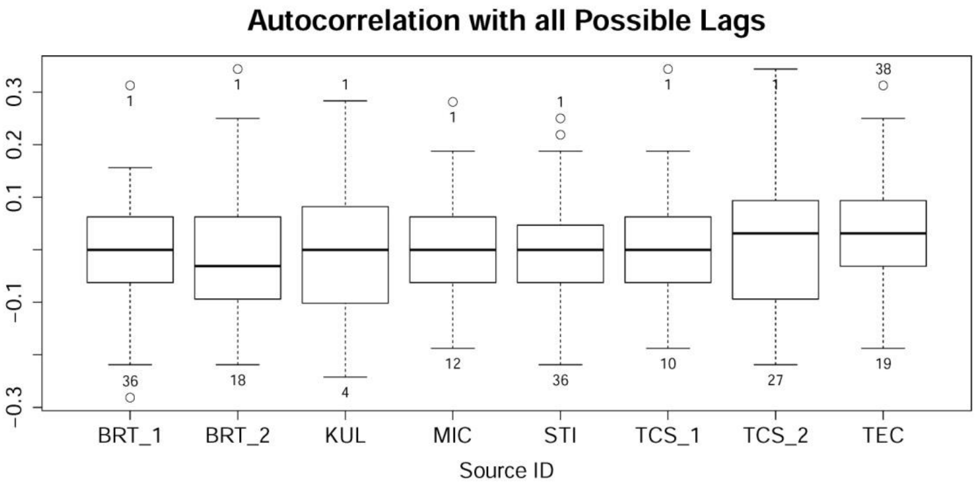


Figure 72: Autocorrelation of mother boards. Lags of highest and lowest autocorrelation are labeled.

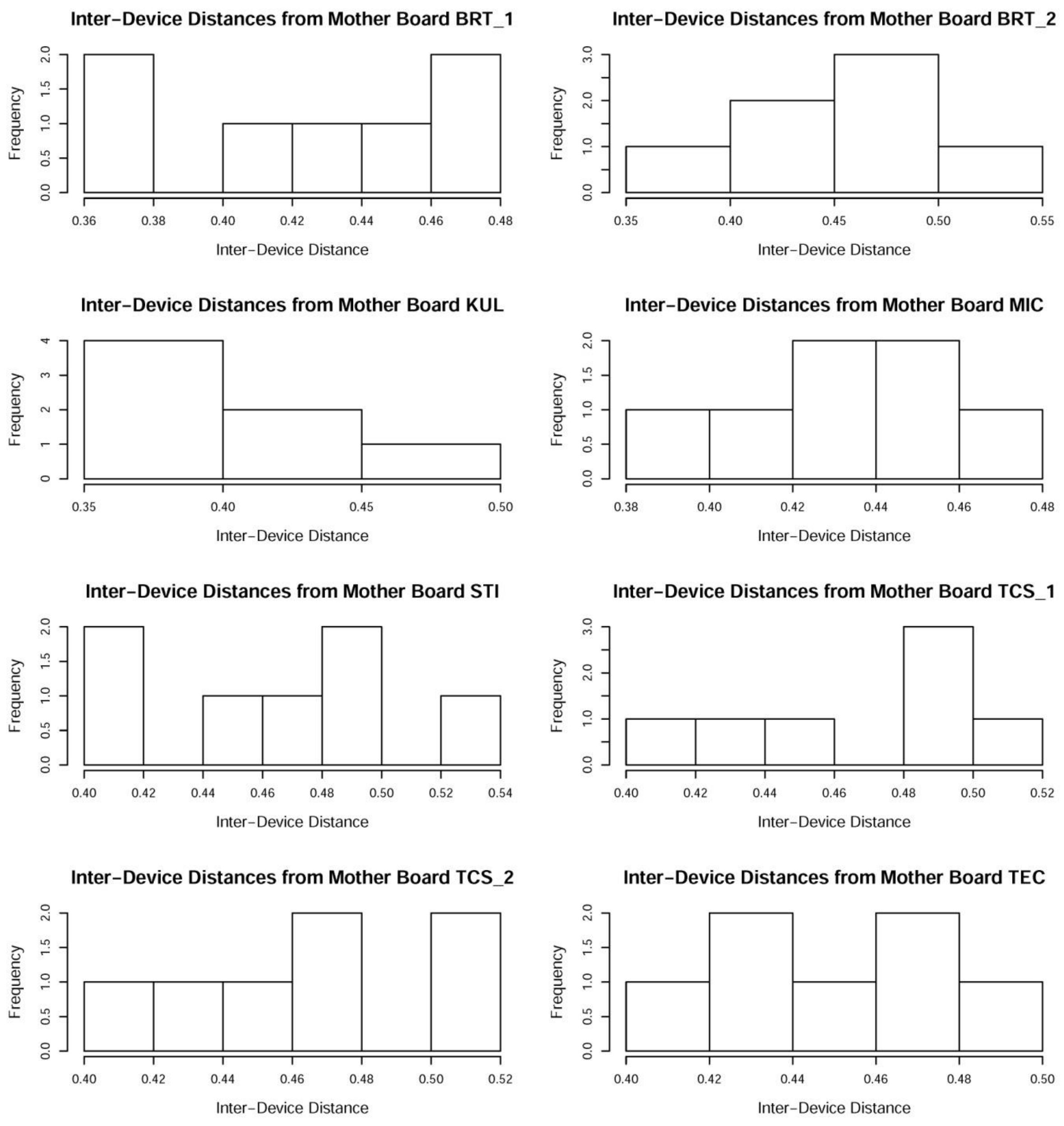


Figure 73: Inter-device distances for all mother boards

Appendix D Reconstruction with multiple responses

As mentioned in 6.6, one might say that multiple responses could be taken to generate the most likely bit string during reconstruction. At the first glance it does not seem to be impossible to achieve an improvement with this approach. In order to observe this, we carried out the following evaluation, done for each daughter board:

- Remove the 13 dark bits identified at 25° C.
- Calculate a reference response consisting of the most likely bits of the first 50 responses (dark bits already removed), read out at 25°C room temperature.
- From the 5,000 responses, take 3/5/10 each and calculate the most likely response. As a result, we have 1,666/1,000/500 *most likely* (ML) responses.
- Calculate the Hamming distances of the reference response against the 1,666/1,000/500 responses and divide it by the number of bits in a response (128 in our case)
- Take the mean of all Hamming distances
- Based on this mean, calculate the failure probability P_{fail}.

After analysing the results properly, it can be said that there are no significant improvements compared to the results shown in 6.5 (Figure 23). The following Table 33 and Table 34 show the ranges of the bit error rate and the failure probability over all boards.

Method	- 40°C		25°C		30°C		85°C	
	Min	Max	Min	Max	Min	Max	Min	Max
Normal [9]	0.0243	0.1170	0.0031	0.0353	0.0498	0.2098	0.0473	0.1502
ML of 3	0.0216	0.1181	0.0007	0.0332	0.0492	0.2110	0.0461	0.1514
ML of 5	0.0209	0.1189	0.0004	0.0328	0.0492	0.2117	0.0460	0.1516
ML of 10	0.0194	0.1164	0.0003	0.0318	0.0504	0.2140	0.0469	0.1543

Table 33: BER of all Boards calculated with the most likely responses (after dark bits removed)

Method	- 40°C		25°C		30°C		85°C	
	Min	Max	Min	Max	Min	Max	Min	Max
Normal [10]	0.0021	0.2793	8e-07	0.0080	0.0254	0.7416	0.0216	0.4613
ML of 3	0.0014	0.2852	2e-09	0.0065	0.0244	0.7462	0.0198	0.4677
ML of 5	0.0012	0.2895	2e-10	0.0062	0.0246	0.7489	0.0196	0.4693
ML of 10	0.0009	0.2759	4e-11	0.0056	0.0265	0.7570	0.0209	0.4838

Table 34: P_{fail} for all boards calculated with most likely responses (after dark bites removed)

As one can see, the improvements are minimal if at all. This is caused by the variety of dark bits over different temperatures as shown in 6.5.2. In some cases, the maximum even gets worse which is likely to be caused by the fact that the responses (and so the most likely and also the reference response) really differ when read out in different environments. Certain bits are completely inverted (although stable, so no dark bit) when reconstructing in different environment.

[9] As in 6.5.2
[10] As in 6.5.2

Appendix E Kang's scheme

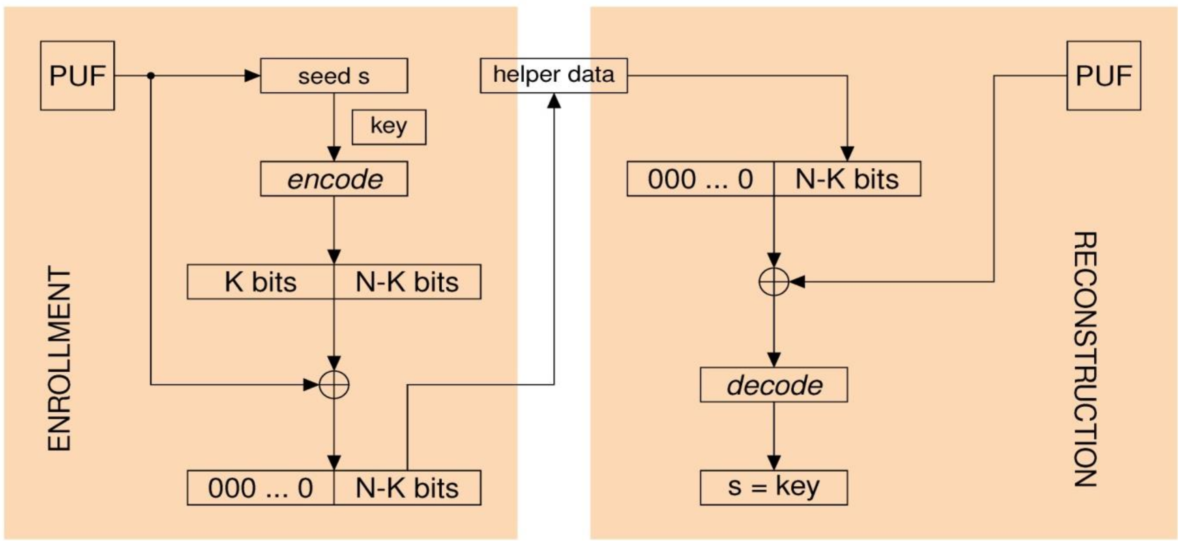


Figure 74: Fuzzy extractor based on codes with systematic encoding (Kang's scheme [2])

Appendix F PLL-TRNG Common Criteria evaluation of AIS 20/31claim

This annex describes a part of the (confidential) work that was done by Brightsight on the AIS 20/31 security claim of the PLL-TRNG as designed by UJM. Note that the short reporting included below was not supplemented with test results. The first reason is that these are confidential (as part of early industrialization of the UJM design for STR). Secondly because it was used only to demonstrate the preparatory steps that need to be taken for a formal evaluation for Common Criteria certification. Also the documentation was not made to support a real-life application, which causes several work items to fail by definition.

Introduction

Industrial application of True Random Number Generators usually requires certification against standards. The TRNG can be part of a system that has well-defined claims on its secure behavior. A useful claim for TRNGs in a Common Criteria evaluation in Europe is that it conforms to a protection class of AIS 20/31. In order to verify if the HECTOR TRNG developments will withstand such Common Criteria evaluation, a trial evaluation was done on one of the HECTOR TRNGs. The PLL-based TRNG – developed by UJM – was selected for this evaluation because HECTORs industrial partners are interested in the commercial application of this design. The results of the evaluation should show that the design is easy to evaluate and will pass relevant work units of the CC evaluation methodology, is such path is pursued.

The UJM PLL-based TRNG design is developed to conform to TRNG Class PTG3.

The design parameters - which are essential in the entropy generation - are UJM's trade secret. For this reason the trial evaluation was done using a separate non-Disclosure Agreement between UJM and Brightsight. This NDA also covers the evaluation report (see [3]) that was produced as part of this effort. In order to show what such evaluation report contains, UJM gave permission to extract a section of this report. This is presented below.

Evaluation on the PLL-TRNG

Brightsight has performed a Common Criteria evaluation of the AIS 20/31 claims on the TRNG designed by UJM. AIS 20/31 is the most widely used standard for evaluation of TRNG's and is developed, maintained and published by BSI (the German Common Criteria (CC) certification body). It describes how a physical or deterministic RNG could be claimed by the developer in the so-called Security Target (ST) document. It also describes the developer evidence contents required and the developer actions needed.

The claim for the HECTOR PLL-based TRNG in Common Criteria style looks like:

*FCS_RNG.1.1 The TSF provides a **hybrid physical** random number generator that implements:*

- *(PTG.3.1) A total failure test detects a total failure of entropy source immediately when the RNG has started. When a total failure is detected, no random numbers will be output.*
- *(PTG.3.2) If a total failure of the entropy source occurs while the RNG is being operated, the RNG **prevents the output of any internal random number that depends on some raw random numbers that have been generated after the total failure of the entropy source**.*

- *(PTG.3.3) The online test shall detect non-tolerable statistical defects of the raw random number sequence (i) immediately when the RNG has started, and (ii) while the RNG is being operated. The TSF must not output any random numbers before the power-up online test has finished successfully or when a defect has been detected.*
- *(PTG.3.4) The online test procedure shall be effective to detect non-tolerable weaknesses of the random numbers soon.*
- *(PTG.3.5) The online test procedure checks the quality of the raw random number sequence. It is triggered **continuously**. The online test is suitable for detecting non-tolerable statistical defects of the statistical properties of the raw random numbers within an acceptable period of time.*
- *(PTG.3.6) The algorithmic post-processing algorithm belongs to Class DRG.3 with separate cryptographic state transition function and cryptographic output function, and the output data rate of the post-processing algorithm, which does not exceed its input data rate (input and output data rates are the same).*

*FCS_RNG.1.2 The TSF provides **random bits** that meet:*

- *(PTG.3.7) Statistical test suites cannot practically distinguish the internal random numbers from output sequences of an ideal RNG. The internal random numbers pass test procedure A.*
- *(PTG.3.8) The internal random numbers use PTRNG of class PTG.2 as random source for the post-processing. The average Shannon entropy per internal random bit exceeds 0.997.*

All bold face text is being chosen by UJM to make the claim more specific for their design. The evaluator of Brightsight has performed the so-called work-units and given a verdict per work-unit. Work-units PTRNG.3-1 up to PTRNG.3-5 are relevant for the claim. Work-unit PTRNG.3-2 contains PTRNG.2-2 up to PTRNG.2-9 that are defined for a PTG.2 class TRNG. Some examples of work-units are given below:

PTRNG.3-1 Examine the description of the intended use of the RNG in the developer evidence document, the ST, and the guidance documents, and check whether the descriptions are complete and internally consistent.

The verdict is ‘pass’ and the full analysis is given as an example:

1. The evaluator notes that no Security Target is written for the RNG under consideration. All claims are contained in [RNG_Design]. The evaluator determined that [RNG_Design] is referring to one type of PTRNG namely the PTG.3.
2. The evaluator has examined [RNG_Design] and determined that the intended usage is as security service for the user. This is as specific as the purpose of the project allows.
3. The evaluator has examined the operations of SFR FCS_RNG.1 class evaluation and determined that no operations are left open in the SFR.
4. The evaluator has examined the developer evidence and the claim FCS_RNG.1 and determined that they are consistent, as is demonstrated in appendix B[11] of the evaluation report.

[11] Note: This is appendix B of confidential evaluation report [3].

PTRNG.2-2 Examine the developer description of the PTRNG module and check for internal consistence.

The verdict is ‘pass’ and is supported by a point-wise check of design choices in 10 different categories.

PTRNG.2-3 Evaluate that the implementation of the RNG is according to the developer's description of the PTG module.

The verdict is ‘inconclusive’ as the activities within the HECTOR project did not include a full implementation review at the detail that is required within Common Criteria.

PTRNG.2-4 Examine the developer's evidence that the internal random number sequence contains at least a minimum amount of entropy, which is identified in the element FCS_RNG.1.2 clause (PTG.3.8) under all intended environmental conditions.

The verdict is ‘pass’ and is supported by an analysis of the stochastic model provided by the developer, explaining how the sampling of the random process can account for the amount of entropy as claimed.

PTRNG.2-7 Examine the developer's demonstration that the online test detects non-tolerable statistical weaknesses of the raw random signals sufficiently soon.

The verdict is ‘pass’. A theoretical model demonstrates how error patterns or a drop of entropy would trigger the online test. From the evidence it can be seen that it is guaranteed that an error vector is faster than the post processing result being outputted. The control I/F module that signals that random data is available, is also the module that triggers an interrupt request in case of an error bit being asserted.

As a summary, most work-units show a ‘pass’ verdict, however some of the work-units are not yet in the ‘pass’ state. This is caused by the rigorous implementation representation review required in a Common Criteria evaluation and also by the way the embedding of a TRNG into a certified product is subject to criteria that cannot be met by Demonstrator 1. However the design principles as such are suitable for being part of a product design that can be CC certified.

In particular the online tests, that give statistical confidence on the amount of entropy present in the TRNG output, represent very suitable and novel solution for an AIS 20/31 compliant TRNG. All work-units related to this online tests gave a ‘pass’ result, see for example PTRNG.2-7.

Conclusions

The Common Criteria evaluation of the AIS 20/31 claim ‘PTG3’ of the PLL-based TRNG developed by UJM shows that the design is likely to pass. Some work units could not be verified because the TRNG is not designed as a commercial product, subject to a real CC evaluation.

Appendix G DC-TRNG Common Criteria evaluation of AIS 20/31claim

This section demonstrates the evaluation work that was done by Brightsight on the AIS 20/31 security claim of the DC-TRNG as designed by KUL. Note that the included reporting below was not supplemented with test results, as it was used only to demonstrate the preparatory steps that need to be taken to achieve a formal evaluation for Common Criteria certification. Also the documentation was not made to support a real-life application, which causes several work items to fail by definition.

Introduction

The report describes the results of applying the AIS 20/31 PTRNG.2 criteria. KU Leuven is currently preparing their Delay-Chain based TRNG (DC TRNG) on the Xilinx Spartan-6 FPGA for evaluation and claims compliance with PTG.2 of the AIS 20/31 (see [1]). To this end, Brightsight provides RNG evaluation activities on the stochastic model and other design documents. This report shows the CC compliance review of the hardware RNG part.

The review is performed against the requirements with respect to AVA_VAN.5, which means that the RNG shall be resistant against attackers with a high attack potential. The evidence for the compliance review is in [22]. No files with actual random data were provided by KU Leuven nor measured by Brightsight. The AIS 20/31 test suite has therefore not been performed yet.

The evaluation report uses different text colors to emphasize the meaning.

This is text from the AIS 20/31 document.

This is text from the evidence documents.

Identifiers

This work has no certification identifier.

This evaluation was not performed under supervision of a CC Scheme.

The developer of the Target Of Evaluation:

Katholieke Universiteit Leuven

Kasteelpark Arenberg 10,

3001 Leuven,

Belgium

The evaluation is performed by:

Brightsight

Delftechpark 1

2628 XJ Delft

The Netherlands

Criteria and documentation

The evaluation is performed with the following criteria, methodology and interpretations:

Criteria and methodology

[CC] Common Criteria for Information Technology Security Evaluation
Part 1: Introduction and general model, September 2012, Version 3.1 Revision 4 Final (CCMB-2012-09-001)
Part 2: Security functional components, September 2012, Version 3.1 Revision 4 Final (CCMB-2012-09-002)
Part 3: Security assurance components, September 2012, Version 3.1 Revision 4 Final (CCMB-2012-09-003)

[CEM] Common Methodology for Information Technology Security Evaluation, Evaluation methodology, September 2012, Version 3.1 Revision 4 Final (CCMB-2012-09-004)

Scheme interpretations

[AIS31] AIS 31 Funktionalitätsklassen und Evaluationsmethodologie für physikalische Zufallszahlengeneratoren (Functionality classes and evaluation methodology for physical random number generators), version 3, 15.05.2013

Evaluation summary

This chapter provides an overview of the status of the evaluation, indicating any actions needed by the developer or the evaluation facility to complete the evaluation. This chapter also indicates possible vulnerabilities that are identified during the evaluation.

Actions

The table below indicates the actions needed to complete the evaluation. The actions include requested improvements, questions about the evidence, tasks to be performed and consequences for other deliverables.

Number	Requirement	Action	Holder
RNG.1	PTG.2.1	In a user guidance document, the user should be instructed to recognize the situation of a total failure occurring, leading absence of raw random data being generated.	Developer
RNG.2	PTG.2.3 PTG.2.4	An online test for the quality of the random numbers must be performed by the TRNG module automatically or be triggered externally by the user	Developer
RNG.3	PTG.2.6 PTG.2.7	The AIS test procedure should be applied to1.6 MB of raw random data generated from the TRNG, in order to see that the raw random numbers are indeed indistinguishable from output sequences of an ideal RNG. In particular the threshold of 7.976 bits of entropy per byte should be tested for.	Developer
RNG.4	PTG.2-1 up to PTG.2-7	After update of the developer documentation: Assessment of the PTG.2 requirements for the random number generator	Evaluator
RNG.5	PTRNG.2-2	Perform the BSI work-units. This only applies in case KUL is interested in making claims according to the AIS20/31 document by BSI.	Evaluator

Conformance to AIS 20/31

PTG.2 requirements

Below are the requirements applied from [1]. The selections and assignments will be made to match the TOE functionality.

Security functional requirements for the RNG class PTG.2

Functional security requirements of the class PTG.2 are defined by component FCS_RNG.1 with specific operations as given below.

FCS_RNG.1 Random number generation (Class PTG.2)

FCS_RNG.1.1 The TSF shall provide a *physical* random number generator that implements:

(PTG.2.1) A total failure test detects a total failure of entropy source immediately when the RNG has started. When a total failure is detected, no random numbers will be output.

(PTG.2.2) If a total failure of the entropy source occurs while the RNG is being operated, the RNG [selection: prevents the output of any internal random number that depends on some raw random numbers that have been generated after the total failure of the entropy source].

(PTG.2.3) The online test shall detect non-tolerable statistical defects of the raw random number sequence (i) immediately when the RNG has started, and (ii) while the RNG is being operated. The TSF must not output any random numbers before the power-up online test has finished successfully or when a defect has been detected.

(PTG.2.4) The online test procedure shall be effective to detect non-tolerable weaknesses of the random numbers soon.

(PTG.2.5) The online test procedure checks the quality of the raw random number sequence. It is triggered [selection: externally, at regular intervals, continuously, applied upon specified internal events]. The online test is suitable for detecting non-tolerable statistical defects of the statistical properties of the raw random numbers within an acceptable period of time.[24]

FCS_RNG.1.2 The TSF shall provide [selection: *bits, octets of bits, numbers [assignment: format of the numbers]*] that meet: *(PTG.2.6) Test procedure A [assignment: additional standard test suites] does not distinguish the internal random numbers from output sequences of an ideal*

RNG.

(PTG.2.7) The average Shannon entropy per internal random bit exceeds 0.997.

Assessment of the PTG.2 part of the random number generator

PTG.2.1

(PTG.2.1) A total failure test detects a total failure of entropy source immediately when the RNG has started. When a total failure is detected, no random numbers will be output.

No dedicated total failure test is implemented.

The TOE implements sampling strategy that triggers upon a rising or falling edge in the difference signal between asynchronous counters CounterOut1 and CounterOut2. In case both CounterOut1 and CounterOut2 get stuck to a fixed value, no edges will be triggered, and no random data will be generated.

In case DataOut1 and DataOut2 both get stuck to a certain frequency, it could occur that a periodic bit-stream is generated, without detection. This can be resolved by implementing an online test.

The TOE does fulfill the requirement, in case the user properly responds to a lack of random data being generated upon his request.

PTG.2.2

(PTG.2.2) If a total failure of the entropy source occurs while the RNG is being operated, the RNG [selection: prevents the output of any internal random number that depends on some raw random numbers that have been generated after the total failure of the entropy source].

As a result from the sampling mechanism, no raw random numbers are extracted in case both asynchronous counters CounterOut1 and CounterOut2 get stuck to a fixed value. See also requirement PTG.2.1.

The TOE does fulfill the requirement.

PTG.2.3

(PTG.2.3) The online test shall detect non-tolerable statistical defects of the raw random number sequence (i) immediately when the RNG has started, and (ii) while the RNG is being operated. The TSF must not output any random numbers before the power-up online test has finished successfully or when a defect has been detected.

No online test has been implemented to detect statistical defects nor has the user been instructed to perform such a test on the raw random numbers provided.

PTG.2.4

(PTG.2.4) The online test procedure shall be effective to detect non-tolerable weaknesses of the random numbers soon.

As no online test procedure has been defined, the effectiveness cannot yet be established.

PTG.2.5

(PTG.2.5) The online test procedure checks the quality of the raw random number sequence. It is triggered [selection: externally, at regular intervals, continuously, applied upon specified internal events]. The online test is suitable for detecting non-tolerable statistical defects of the statistical properties of the raw random numbers within an acceptable period of time.

The online test procedure should check the quality of the raw random number sequence. See the analysis for PTG.2.4 for more detail. The online test can either be included in the TRNG module (triggered at regular intervals or continuously) or be left as an instruction to the user (triggered externally) is stated to be triggered externally.

Most common online test is a chi-square test; alternatively some other statistic over the generated data can be calculated. See also [1].

PTG.2.6

(PTG.2.6) Test procedure A does not distinguish the internal random numbers from output sequences of an ideal RNG.

The empty assignment, effectively meaning ‘and no other test suites’ is a valid assignment. Test procedure A was performed on the random data and the results are described.

The TOE fulfills the requirement: The AIS 20/31 test procedure A is performed with a pass verdict.

PTG.2.7

(PTG.2.7) The average Shannon entropy per internal random bit exceeds 0.997.

The AIS 20/31 test to determine the entropy has been performed on the TOE. The entropy is found to exceed 0.997.

The TOE fulfills the requirement on the AIS 20/31 test procedure A with a pass verdict.

Test results

The evaluator has collected four files with random data from the random number generator. Two files contain data directly from the random noise source. The other two files contain random data from the post-process of the random number generator. All files were tested using the AIS 20/31 test suite and the results are pass.

Consistency Check of the TOE PTRNG

[23] Claims that the TOE RNG is conformant to the PTG.2 requirement as described in [1]. The evaluator has performed a design review on the TOE.

Part of SFR	Assignment/ selection	Choice made in SFR	Evaluator's remarks
FCS_RNG.1.1	[selection: physical, non-physical true, deterministic, hybrid physical, hybrid deterministic]	Not yet made	Should be 'deterministic'
(PTG.2.2)	[selection: prevents the output of any internal random number that depends on some raw random numbers that have been generated after the total failure of the entropy source, generates the internal random numbers with a post-processing algorithm of class DRG.2 as long as its internal state entropy guarantees the claimed output entropy]	Not yet made	Should be 'prevents the output of any internal random number that depends on some raw random numbers that have been generated after the total failure of the entropy source' as no DRG.2 post-processing is implemented.
(PTG.2.5)	[selection: externally, at regular intervals, continuously, applied upon specified internal events].	Not yet made	Not clear yet, see also action RNG.2
FCS_RNG.1.2	[selection: bits, octets of bits, numbers [assignment: format of the numbers]]	Not yet made	Selection 'bits' is correct.
(PTG.2.6)	[assignment: additional standard test suites]	Not yet made	The empty selection meaning 'no additional standard test suites' is acceptable here.

PTRNG design in [AIS20/31]	TOE PTRNG design	Evaluator's remarks
(1) the internal entropy source that generates raw random signals,	In [22]	See sections 2.3 and 2.4 of [22].
(2) the digitization mechanism of the raw random signal into the raw random number sequence,	In [22]	See sections 2.5 and 3.3 of [22].
(3) any post-processing of the raw random number sequence generating the internal random numbers, secrets and publicly known values (if there are any), and	In [22]	No post-processing is present; see sections 2.5 and 3.3 of [22].
(4) the online test(s) (applied to the raw random numbers or the internal random numbers), a tot test (shall detect a total failure of the entropy source), and a start-up test.	In [22]	Some total failure protection is present in the sampling mechanism; see section 3.3 of [22]. No online test and start-up test are described in [22].

RNG Vulnerability Analysis and Penetration Test

This appendix describes the vulnerability analysis and penetrations testing that have been performed on the TOE RNG module.

RNG Vulnerability Analysis

The RNG is protected from tampering

1. The TOE PTRNG design incorporates the following mechanism against tampering:
   - The sampling mechanism makes that no raw random numbers are created once the source of entropy gets stuck. There is no active test that detects the total breakdown of the internal entropy source (e.g., if the output becomes all 0s or all 1s).
   - No integrity tests are that can detect manipulation of clock or down-sampling or ring oscillators are described. No security mechanism is present for the internal random number when it is transported to the TRNG output register.
   - No online test (continuous monitoring or statistical properties) is implemented for the internal random number stored in the raw random data register.
2. Based on the security mechanism described above, the evaluator analyzed the security of each TOE PTRNG components against tampering:
   - Internal entropy source and digitization mechanism. These components are part of Entropy Source (RO) and the Digitization modules, respectively. The output of the digitization module is not protected by total failure test, integrity tests, or online test. Since this is an important part of the whole RNG a penetration test is needed to verify its security. This is marked as **RNG.V1**.
   - Register and Priority Encoder are part of the Digitization module. They perform deterministic operations, but are not protected by Known-Answer-Test (KAT) during initialization. The module is also not protected during operation and therefore an attacker might manage to tamper with this module.
3. Output buffer is not protected by integrity mechanism. The possible attack methods from [29] that can be used by an attacker to tamper the TOE PTRNG components are as follows:
   - Light injection. The TOE does not have light sensor. Therefore this attack has to be tested.
   - Electromagnetic injection. The TOE does not have electromagnetic sensor. Therefore this attack has to be tested.
   - Voltage glitch injection. The TOE is not equipped with a glitch sensor, therefore this attack needs to be considered. It is expected that a voltage glitch intended to manipulate only the TRNG, will influence other parts of the TOE much more, such that the TOE will reset far before the point that a glitch threatens the TRNG.
   - Temperature manipulation. The TOE is not equipped with temperature sensor, therefore this attack needs to be considered. It is expected that a temperature manipulation intended to manipulate only the TRNG, will influence other parts of the TOE much more, such that the TOE will reset far before the point that a manipulation attempt threatens the TRNG.
   - Physical manipulation. The TOE is not equipped with an active shield, therefore this attack could be practical.
4. The following list shows RNG tampering attacks from public domain:
   - Frequency injection. This attack is applicable to RNG that uses PLLs as random source. Since the TOE uses thermal noise as random source this attack is not applicable.

5. Based on the analysis above, the following potential vulnerabilities in regard to TOE PTRNG are concluded:

The RNG is protected from monitoring

1. The TOE PTRNG design ensures that the PTRNG output is not (also) used as internal random number for feeding countermeasures. Therefore random number usage that might be monitored externally (such as for clock jitter security mechanism) is different with the random number provided to user.
2. An attacker could still monitor the activity of the PTRNG components through Power or Electro-Magnetic signal. In many cases, side-channel analysis is not a practical attack on TOE PTRNG because of the random and unknown characteristic of the values. An attacker cannot control any part of the random number because it was based on internal chaotic analogue circuit. An exception is formed by sampling mechanism. This is marked as **RNG.V2**.
3. The evaluator has reviewed the TOE PTRNG design and determined that the RNG is protected from monitoring.

Number	**Potential Vulnerability**
RNG.V1	Perturbation attack on TRNG output register (or on post-processor, if present) to fix the output.
RNG.V2	Timing attack on the sampling mechanism. Power consumption or Electro-Magnetic signal could be used as side-channel.

The RNG is protected from misuse

1. The TOE operating and environmental conditions are described in the Xilinx Spartan-6 Datasheet. The voltage supplied needs to be between 1.0 and 3.6 Volts. Correct operation is guaranteed only in the range from 0 to 85 degrees Celsius.
2. The user of the RNG block must verify if error condition occurs during random number generation process. This not marked user guidance element yet.

The evaluator has reviewed the user guidance entries and has determined that these are clear enough to protect the RNG from misuse.

RNG penetration tests

The summary of the RNG penetration tests are described below.

RNG.V1 Perturbation attack on TRNG output register

Test name RNG.V1 Perturbation attack on TRNG output register	**Verdict** Pass	**Date** November 2017	**Evaluator** BRT
Sample Number:			
Functionality The goal of this test is to verify that the RNG digital module is resistant against perturbation attack.			
Reason for selection The random number generator subsystem is enforcing for SFR FCS_RNG.1			
Dependencies N/A			

Test Setup ▢ Xilinx Spartan-6 sample (top side) ▢ Matrix software ▢ Test commands from KUL ▢ AIS31 test software ▢ Light manipulation and EMFI setups ▢ TOE layout map
Procedure ▢ This test was done on the top side of the FPGA ▢ To determine the laser parameters for the attack, a quick preliminary experiment was started. Adjusting the laser beam to a high intensity causes the chip to internally reset whereas a too low intensity does not cause any manipulated behavior. The laser energy adjusted to intensity just below causing a chip reset was used during the experiments. ▢ Start with TOE sample in test mode ▢ Verify TOE version ▢ Collect 1.6 MB of random number by invoking the PTRNG command in a loop (the R0 word of the command is set to "10 00" in order to get the true random number) ▢ Fire laser on 16 to 36 different positions on the RNG analog module. These positions are chosen randomly and equally spread over the RNG digital module area. Collect one or more sets of 1.6 MB of random data. ▢ Feed the sets of 1.6 MB of random data into the AIS31 test suite and run AIS31 Test Procedure A. ▢ Verify that all Test Procedure A tests were passed for all sets.
Expected results All tests of AIS 20/31 Test Procedure A were passed for all sets. The TOE may reset during the test and therefore no random data was collected. This is considered a secure response from the TOE.
Actual results: Pass The actual test result matches the expected test result.

Test conclusion

The result of the tests is Pass.

RNG.V2 Template attack on the output of the sampling mechanism

Test name	**Verdict**	**Date**	**Evaluator**
RNG.V2 Template attack on the input of the online entropy test	Pass	November 2017	BRT
Sample Number:			
Functionality The goal of this test is to verify that the sampling mechanism is resistant against template attacks.			
Reason for selection The random number generator subsystem is enforcing for SFR FCS_RNG.1			
Dependencies N/A			
Test Setup ▢ Xilinx Spartan-6 sample (top side) ▢ Matrix software ▢ Test commands from KUL ▢ AIS31 test software			

▢ Power and EM measurement setup ▢ TOE layout map
Procedure ▢ This test was done on the top side of the FPGA ▢ Start with TOE sample in test mode ▢ Verify TOE version ▢ Perform random number generation and store the raw random data along with the power traces measured at 1 GHz. ▢ Perform correlation analysis on power consumption signal. If correlation is found: perform a second measurement to apply a template attack ▢ Perform an EM scan of the top side of the FPGA. If interesting signal is observed, perform an EM measurement at high sampling rate (5 or 10 GHz) for part of the time-interval. ▢ Perform correlation analysis on EM consumption signal. If correlation is found: perform a second measurement to apply a template attack
Expected results The TOE does not leak its internal random value through power or EM signal.
Actual results: Pass The actual test result matches the expected test result.

Test conclusion

The result of the tests is Pass.