Security Properties of a Class of True Random Number Generators in Programmable Logic

Knut Wold

Thesis submitted to Gjøvik University College
for the degree of Doctor of Philosophy in Information Security

2011

Security Properties of a Class of True Random Number Generators in Programmable Logic

Faculty of Computer Science and Media Technology
Gjøvik University College

Security Properties of a Class of True Random Number Generators in Programmable Logic / Knut Wold
Doctoral Dissertations at Gjøvik University College 1–2011
ISBN: 978-82-91313-71-9
ISSN: 1893-1227

To my wife Mari Anne and my children Ole Petter and Per Kristian

Declaration of Authorship

I, Knut Wold, hereby declare that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others, this is always clearly stated.

Signed:

(Knut Wold)

Date:

Summary

Nowadays, digital equipment such as computers with Internet and cellular phones are commonly used for communication. The users want secure communications, meaning that confidentiality, integrity and authenticity are maintained throughout the session. Confidentiality means that only the intended recipient has access to the transmitted information, integrity ensures that the information is protected against modifications, and finally, authenticity guarantees the identities of the communicating parties. All these security aspects can be achieved using cryptographic techniques and protocols. In a cryptographic system where the algorithm is public, the entire security depends on the used cryptographic key. These keys are generated by using pseudo random number generators (PRNGs) using an algorithm combined with a seed or true random number generators (TRNGs) based on a physical property generating random noise. A TRNG is preferred if the cryptographic system requires a high level of security. Traditionally, a cryptographic system is implemented in an application specific integrated circuit (ASIC), but during the recent years a field programmable gate array (FPGA) has become an attractive alternative. The advantages of using an FPGA compared to an ASIC are the flexibility regarding update of the configuration for correcting errors or adding new functionality, an easier and faster development process and availability of FPGA devices on short notice from the vendors. On the other hand, the flexibility of an FPGA with the possibility of changing the configuration makes it more vulnerable against attacks. The challenge is to design a TRNG in an FPGA with good statistical properties, a reasonable high bit rate and robustness against attacks.

In this thesis, a practical and functional enhancement of a class of TRNGs based on several equal length oscillator rings is proposed. The generation of true randomness is based on the uncertainty of where in time a transition, i.e. a change from logical zero to logical one or vice versa, of the oscillator ring outputs occur due to the presence of jitter. The enhanced TRNG was implemented in several FPGA families and the security properties were examined. The statistical properties were investigated by running the statistical test suites NIST SP 800-22 and DIEHARD, and the test results showed that this TRNG passes these tests. Due to the proposed enhancement the number of oscillator rings could be reduced and a post-processor was not needed in order to pass the statistical tests. Restart experiments from an identical reset state showed that this TRNG generates true randomness and not only pseudo randomness. A detailed spectral analysis was performed on each of the building blocks of this TRNG by investigating the frequency spectrum both in theory and by simulations showing that it approaches the frequency spectrum of an ideal random number generator. The purpose was also to optimize the design parameters in order to achieve a high bit rate. Experiments were performed and a bit rate of 300Mbit/s was achieved while generating random bits with good statistical properties. Even though the statistical properties were found to be good, understanding the noise source and quantifying the amount of entropy are important in the evaluation of an TRNG. A model based on the accumulation of jitter was proposed and simulations were carried out showing the influence of the different design parameters and technology properties on the number of hits close to a transition region defined by the standard deviation of the accumulated jitter. The simulations show that the proposed TRNG with high probability generates bits with high entropy at every sampling point. An investigation of the properties of oscillator rings implemented in three different FPGAs was performed in order to examine the interaction

between rings located close to each other, the correlation and dependency between the rings and also the dispersion of the oscillator ring frequencies. The investigation revealed that there is interaction between some of the rings and a few of them could be regarded as correlated due to almost identical ring frequencies. The experiments showed that there are differences between the examined SRAM based FPGAs compared to a flash FPGA regarding the dispersion of the ring frequencies. The robustness of the TRNG was examined by employing an attack by superimposing a noise signal onto the power supply voltage to the FPGA. Four different TRNG designs were investigated and the two designs based on several oscillator rings were not influenced by this attack while two other reference TRNG designs were. Simulations were performed in order to explain the observed behavior on the TRNGs consisting of oscillator rings. A more exact power model of a microcontroller bus based on the influence of crosstalk due to capacitive couplings between the bus lines was proposed in order to more precisely determine the energy consumption. This model could for instance be used in a side-channel attack determining the encryption key with reduced computational effort.

A TRNG based on the proposed design was implemented in a real life cryptographic system with good results showing that this TRNG is both practical and secure. In addition, this TRNG design is easy to implement in programmable logic, the placement of the inverters inside the FPGA is not critical, it is robust against temperature and power supply variations and not influenced by effects related to aging. All this makes a TRNG based on XOR of several sampled oscillator rings a suitable component in a cryptographic system.

Acknowledgments

The research resulting in this thesis has been carried out at Norwegian Information Security laboratory (NISlab) at Gjøvik University College (GUC) in Norway.

I wish to thank my main supervisors Chik How Tan, who unfortunately left GUC before the work was finished, and Slobodan Petrović for their contributions and constructive feedback. They have both been a source of inspiration to me and they have pushed me forward in my research. I am also very thankful to my co-supervisor and former work colleague Leif Nilsen for his useful comments on my papers and for his support when I started this project.

I am very grateful for the support from Markus Dichtl in Siemens AG in Munich in Germany. He was a very useful discussion partner on difficult technical issues and he guided me in the right direction in the process of understanding the TRNG design.

I want to thank my father Arne Wold for technical assistance especially regarding digital signal processing and Matlab, Geir Olav Dyrkolbotn for great co-operation and for assistance with measuring of high speed signals, Frode Gilberg for useful and most needed assistance in using C# and Nils Kalstad Svendsen for useful help and tips and for squash practice. I will also like to thank the rest of my colleagues at GUC for making this a pleasant and inspiring place to work in.

Finally, I want to thank my wife Mari Anne and my two boys Ole Petter and Per Kristian for inspiration and encouragement in troublesome periods when the progress of the research was not as good as expected.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

During the last decades there has been a rapid development in the technology used for digital communication. For instance, the introduction of Internet and cellular phones has revolutionized the everyday life of people, but this new technology has also made it possible for attackers to illegally get access to information that was meant to be private or secret. One possible solution to this problem is to use cryptography. This has been used for centuries in the military and especially in war situations to prevent secret information falling into the hands of the opposite side. A cryptographic system can be defined as a computer system using encryption in order to hide the information or plain text with the use of a secret key. Reconstruction or decryption of the original information should only be possible for a recipient knowing the same secret key originally used in the encryption. The objective of cryptography is to enable two entities to exchange information through an insecure channel in such a way that an attacker is unable to obtain or decode the concealed information. This could be solved by using a cryptographic system which could either be asymmetric such as a public-key system or symmetric based on for instance a block cipher such as the Advanced Encryption Standard (AES). A model of a symmetric cryptographic system is shown in Figure 1.1. It consists of an encryption block, a decryption block, a key generator and a secure channel for key management [49]. In order to achieve high security, it is important that the cryptographic algorithm is strong. However, in both symmetric and asymmetric cryptographic systems the algorithm is usually public and in this case the security depends on the secrecy of the key. Therefore it is vital that the circuit generating these keys, a random number generator (RNG), has properties that make it impossible for an attacker to predict the used key in a better way than by pure guessing. Even though the cryptographic algorithm is strong, a weak RNG could significantly reduce the security of the cryptographic system.

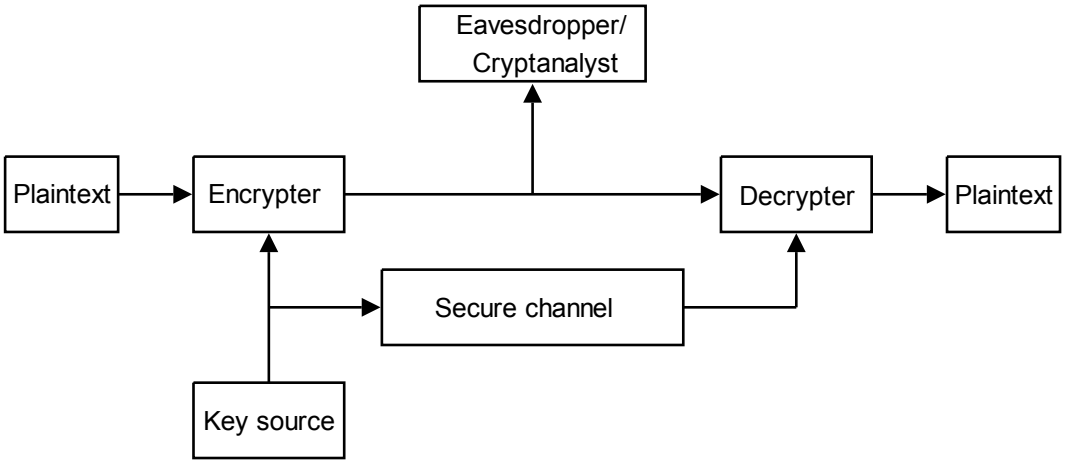

Figure 1.1: Model of a symmetric cryptographic system.

An implementation of an RNG used in a cryptographic system must satisfy a set of requirements. During the work with this thesis, the following requirements emerged as important for an RNG:

- The RNG shall generate output bits with good statistical properties.
- The RNG shall generate output bits with high entropy.
- The RNG shall be robust against attacks.
- The RNG shall be robust against environmental variations.
- The RNG shall have stable properties during the lifetime of a product.
- The RNG should have reasonably high throughput or bit rate.
- The RNG should have a simple implementation.

An RNG satisfying these requirements can be used in a high-grade cryptographic system.

A cryptographic system has traditionally been implemented in an application specific integrated circuit (ASIC), but the growth in the field of programmable logic devices (PLDs) has made these circuits more suitable and appealing even for use in high-grade cryptographic systems. Implementing an RNG inside a programmable device such as a field programmable gate array (FPGA) along with the encryption module would give a more secure design in the sense that an attack is more difficult due to limited access to the RNG and because the generated key does not need to be distributed outside the integrated circuit.

1.1 Research Questions

Designing a circuit capable of generating random bits with good statistical properties at a high bit rate and at the same time being robust against attacks and changing environments is a challenging task. An interesting perspective is to investigate the possibility of implementing an RNG based on a physical property in a programmable device with so good properties that it could be used in a high-grade cryptographic system. This could be formulated into a research question:

- Q1: Investigate the properties of an FPGA in order to design and implement an RNG with good statistical properties, high throughput rate and independent of a special placement of the logic in the programmable device.

As an extension of this problem, two related problems can be formulated:

- Q2: What are the details of operation of such an RNG and what are its performances?
- Q3: Is this RNG implementation robust against attacks?

These three research questions are answered by the following papers included in this thesis:

1. Knut Wold and Chik How Tan, **Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings**, in *International Journal of Reconfigurable Computing*, pp. 1–8, 2009.

2. Knut Wold and Slobodan Petrović, **Optimizing Speed of a True Random Number Generator in FPGA by Spectral Analysis**, in *Proceedings of the 4th IEEE International Conference of Computer Sciences and Convergence Information Technology (ICCIT'09)*, pp. 1105–1110, 2009.

3. Knut Wold and Slobodan Petrović, **Robustness of TRNG against Attacks that Employ Superimposing Signal on FPGA Supply Voltage**, in *Proceedings of the Norwegian Information Security Conference (NISK'10)*, pp. 81–92, 2010, Tapir Akademisk Forlag.

4. Geir Olav Dyrkolbotn, Knut Wold and Einar Snekkenes, **Security Implications of Crosstalk in Switching CMOS Gates**, in *Proceedings of the 13th Information Security Conference (ISC'10)*, Lecture Notes in Computer Science, vol. 6531, pp. 269–275, 2011, Springer.

5. Knut Wold and Slobodan Petrović, **Behavioral Model of TRNG Based on Oscillator Rings Implemented in FPGA**, in *Proceedings of the 14th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS'11)*, pp. 163–166, 2011.

6. Knut Wold and Slobodan Petrović, **Security Properties of Oscillator Rings in True Random Number Generators**, *Submitted*.

In addition, the following paper (see Appendix A) is overlapping with one of the papers listed above:

- Knut Wold and Chik How Tan, **Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings**, in *Proceedings of the 4th IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig'08)*, pp. 385–390, 2008.

In Appendix B, the full version of paper 4 as it was presented at the conference is given, and in Appendix C, an additional theoretical model of an RNG based on oscillator rings is presented, in which the guaranteed randomness rate is computed.

The relationship between the research questions and the included papers is shown in Figure 1.2.

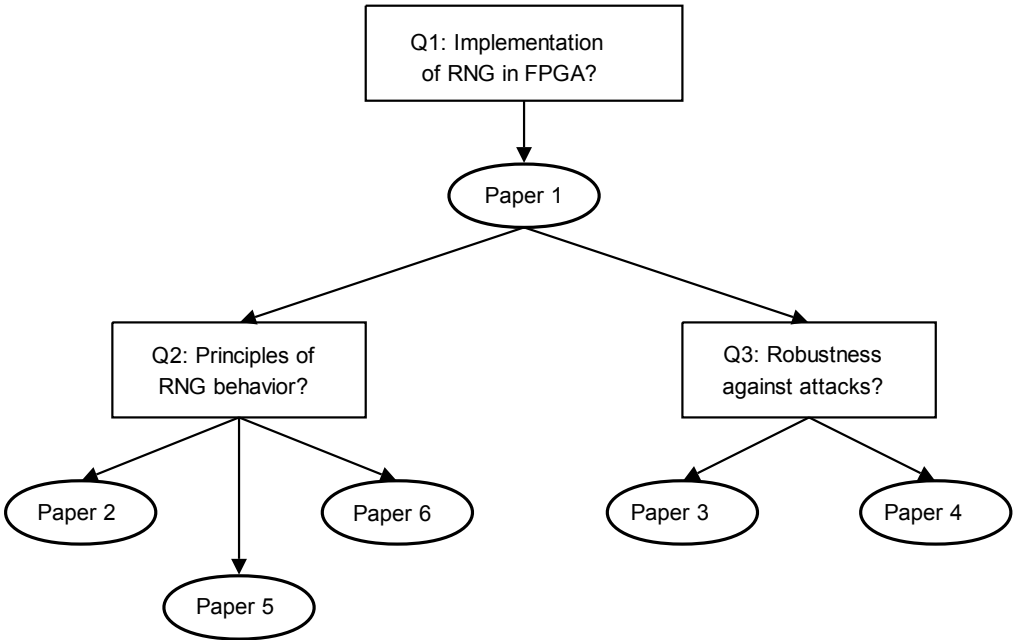

Figure 1.2: Relationship between the included papers and the research questions.

Paper 1 [64] answers the research question Q1. In that paper, an RNG based on XOR of several sampled equal length oscillator rings is proposed. This RNG is an enhancement of the proposal of Sunar et al. [51]. The RNG is implemented in an Altera Cyclone II FPGA and the experiments show that the properties of this generator are good.

However, even though the design is quite simple, the exact principles behind its behavior is not obvious. Paper 2 [60] investigates the frequency spectrum of the different modules in this RNG explaining its behavior. At the same time, the RNG is optimized in order to achieve a high throughput rate. Paper 5 [62] investigates the amount of entropy in the RNG by defining a model based on the number of hits in the transition region determined by the size of the standard deviation of the accumulated jitter. The simulations performed in Matlab verify the observations from the experiments performed in [64] that the proposed RNG design generates entropy at every sampling point or bit. Paper 6 [59] investigates the low-level properties of the oscillator rings implemented in an FPGA and their implication on the security of this RNG. The covered topics are interaction between oscillator rings implemented physically close to each other, correlation and dependency between the sampled output of the oscillator rings and the dispersion between oscillator ring frequencies. All the experiments are carried out using three different types of FPGAs from different vendors.

The robustness of the RNG is investigated in paper 3 [61] answering Q3. Four different RNG designs are implemented and tested for robustness against an attack that employs superimposing signal on the FPGA power supply voltage. The experiments show that RNGs based on several oscillator rings are not influenced by this attack while the two other designs are. In paper 4 [17], an improved power model based on layout dependent phenomena such as capacitive couplings between bus lines is proposed. This power model can be used to improve the performance of a side-channel attack by reducing the computing workload, but also to reveal the random data transfered on a data bus between the key source and the encryption module in an RNG.

1.2 Structure of the Dissertation

The rest of the thesis is organized as follows: In Chapter 2 an overview of random number generators in programmable devices is given describing the state of the art and different aspects related to the topic. In Chapter 3 a summary of the contributions of the included papers and the thesis is given. In Chapters 4-9 the six research papers included in this thesis are presented. In Appendix A the overlapping paper is given, in Appendix B the full version of paper 4 is included and in Appendix C an additional theoretical model of an RNG based on oscillator rings is presented.

Chapter 2

Background

An RNG is a device or a circuit generating a stream of bits with a property that the value of each bit is unpredictable, but in the long run the bits follow a certain distribution. Based on the knowledge of the previous bits it shall be impossible to calculate or predict the next bit in a better way than by pure guessing. These random bit sequences could for instance be used for generation of session keys, initial vectors (IVs) or cryptographic challenges/responses in a cryptographic system.

Implementations of binary sequence generators used in cryptography can be divided into two different categories [42]:

- Pseudo random number generator (PRNG): an algorithm with a property that it is computationally infeasible to predict the value of the next bit even when all the previous bits are given. The output bit is unpredictable, but it is reproducible since sequences generated from the same seed or initial state in the PRNG will be identical, i.e. the generator is deterministic.

- True random number generator (TRNG): a device where the generation of an output bit is based on a physical random process (for example a radioactive device, thermal noise from a diode, jitter of a clock or metastability on a signal, etc.), or on non-deterministic events (for example system time in a computer, seek time of a hard disk, user interaction like mouse movements or keyboard typing, etc.). The output bit is both unpredictable and irreproducible.

In addition, it is possible to implement RNGs, which are combinations or hybrids with design elements from both the PRNG and the TRNG. In the rest of this thesis the focus will be on physical TRNG.

TRNGs based on logic devices can be implemented in for instance an ASIC, in a secure micro controller on a smart card or in a programmable device such as an FPGA. In this thesis, only implementations of TRNGs in FPGA are considered.

The main challenge in designing a TRNG is to find a suitable noise source and a method of extracting the randomness from this source. For instance, the stochastic process of radioactive decay has been used for generation of random bits. The disadvantages of RNGs based on a radioactive device are low bit rate, large physical size and potential environmental and medical challenges. An often used alternative is to amplify the noise of a diode [26]. Problems related to this solution are the sensitivity to variations in temperature and power supply voltage and the changes in component properties due to aging. In addition, the bit rate is not very high and the implementation uses external components, which makes manipulation by an attacker easier. A better solution is to implement the whole TRNG inside an integrated circuit such as a micro controller or a programmable device such as an FPGA or a PLD. The advantages are that the design is digital and therefore more robust against aging and variations in temperature and power supply voltage and the implementation is more secure because of limited physical access by an attacker.

A model of the physical TRNG [10, 44] is shown in Figure 2.1. The noise source is the physical property creating the true randomness or entropy in the generator. The noise signal $n(t)$ is typically an analog signal and it is digitized creating a binary digital signal $s[i]$.

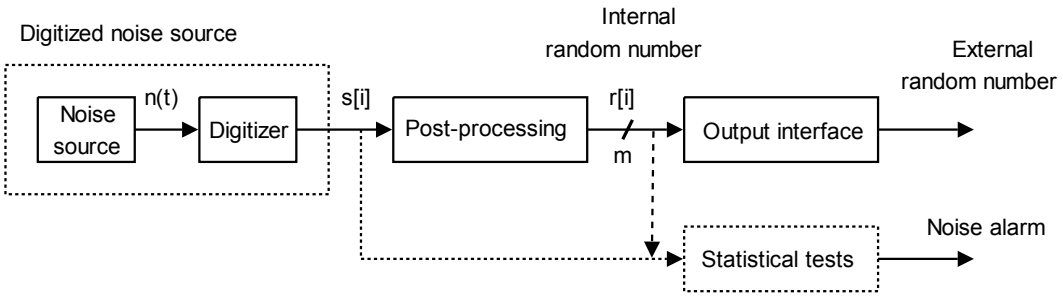

Figure 2.1: Model of a physical TRNG.

The random number can then be fed into a post-processor creating the m-bit internal random number $r[i]$. The post-processor may be skipped if the noise source has good properties, but in most real TRNG implementations in cryptographic systems a post-processor is recommended. The output interface could be storage of random bits in some sort of memory before distributing the bits to for instance the encryption module. In addition, there should be a surveillance circuit monitoring the bit sequences in order to detect statistical weaknesses or a breakdown in the generated random bit stream.

For an RNG implementation, the power consumption per generated output bit should be reasonably low. For logic devices, the power consumption could be measured by counting the number of logic gate transitions per bit. In this thesis, the power consumption of the programmable logic devices is not considered.

2.1 RNG Properties

An ideal random sequence $x_n \in \{0,1\}$, $n = 1, 2, \ldots$ can be generated by flipping a fair coin, observing the outcome of a toss and allocating for instance a zero for head and a one for tail. Each throw can be regarded as independent of the previous ones and each outcome occurs with a probability of $\frac{1}{2}$. Table 2.1 shows the possible combinations of the outcomes and what combinations that result in a transition, i.e. $0 \rightarrow 1$ or $1 \rightarrow 0$, in such a scheme.

The following properties are achieved based on an ideal random sequence:

1. $Prob(x_n = 0) = Prob(x_n = 1) = \frac{1}{2}$
2. $Prob(Transition) = \frac{1}{2}$

Property 1 states that there should be an equal amount of zeros and ones in the sequence, i.e. no bias. Property 2 states that on average there should be a transition or change in the bit value for every second bit. If a bit sequence diverges much from these properties, the sequence will not be regarded as random because it is possible to predict the next bit in a better way than guessing.

Some similar requirements are reflected in Golomb's postulates for a PRNG [23]:

Present bit	Next bit	Transition
0	0	No
0	1	Yes
1	0	Yes
1	1	No

Table 2.1: Possible combinations of bit values and transitions.

- The number of zeros and ones shall be approximately equal over a period.
- The number of blocks (runs of ones) over a period shall be as follows: $\frac{1}{2}$ of all blocks shall have length 1, $\frac{1}{4}$ of all blocks shall have length 2, $\frac{1}{8}$ of all blocks shall have length 3, etc. The similar distribution applies for gaps (runs of zeros).
- The value of the autocorrelation function shall be constant except for multiples of the period where it has a peak

The entropy in information theory is a measure of uncertainty of the value of the next generated bit. The entropy H of a binary signal is defined as [45]:

$$H = -\sum_{i=0}^{1} p_i \cdot log_2(p_i) \tag{2.1}$$

where p_i is the probability of generating a zero or a one. Ideally, the probabilities p_0 and p_1 are equal to $\frac{1}{2}$ resulting in an entropy of 1.0, which is the maximum value of the entropy function in Equation (2.1).

For practical implementations of a TRNG the following general requirements can be given [42]:

- The statistical properties of the random sequence should be good.
- It shall not be possible to compute either previous or future bits based on the knowledge of subsequences of bits generated by the RNG with a higher probability than by pure guessing.

An RNG satisfying these requirements can be regarded as suitable for use in a high-grade cryptographic system treating sensitive information. The first requirement is usually experimentally verified by using statistical test suites such as NIST SP 800-22 [35] or DIEHARD [32]. The generated bit sequences used as input to these tests must be long (several Mbit) in order to verify that the bit sequences have the expected properties. Verifying the second requirement is more challenging, but it can be achieved by finding a lower bound of the entropy per random bit based on a stochastic model of the noise source [43, 44]. This stochastic model is simplified if the generated bits can be regarded as independent. A method achieving this independence is to implement a stateless or memoryless RNG by for instance resetting all the states inside both the digitized noise source and the post-processor after each generated bit [10].

2.2 Programmable Devices

Programmable devices are integrated circuits with a regular architecture and the ability of being configured by the customer in the field and not restricted to the facilities of the manufacturer. A programmable device can be classified based on how the configuration is stored in the device:

- SRAM - four transistors forming two cross-coupled inverters storing 1 bit.
- Flash EPROM - a floating-gate MOSFET transistor with two gates storing 1 bit.
- Antifuse - a permanent electrical connection created by applying a current exceeding a specified limit.

The programmable device is said to be volatile if the configuration is lost when the power is switched off meaning that the configuration needs to be loaded at every power-up. A device based on SRAM is volatile while flash EPROM and antifuse FPGAs are non-volatile.

The configuration of the SRAM and the flash EPROM devices are re-programmable while the antifuse devices are one-time-programmable (OTP).

An alternative classification of programmable devices can be made by the size and the complexity of the programmable device:

- SPLD - Simple Programmable Logic Device.
- CPLD - Complex Programmable Logic Device.
- FPGA - Field Programmable Logic Device.

SPLDs are small devices only capable of implementing simple digital logic, while the largest FPGAs can implement complex digital systems. The complexity of a CPLD is somewhere in between the SPLD and the FPGA. In the rest of this section the focus will be on FPGA only.

An FPGA consists of an array of several small blocks or logic elements. Typically, each logic element consists of a look-up table (LUT) and/or a register element. A LUT can implement any digital function only limited by the number of inputs to the LUT, typically between 3 and 6 depending on the vendor and the FPGA family. The LUT consists basically of a configured memory location and a multiplexer selecting the correct memory bit based on the bit combination on the input signals to the LUT. The register element can implement a flip-flop or a latch capable of storing 1 bit. The number of these blocks or logic elements in an FPGA can vary from some thousands in the smallest devices to more than a million in the largest.

The basic building block of an FPGA varies both in name and functionality depending on both the vendor and the specific FPGA family. For instance, the Cyclone II FPGA from Altera [3] uses the name *logic element* containing a 4 input LUT and a 1 bit register element. The Virtex II FPGA from Xilinx [66] uses the name *slice* containing two 4 bit LUTs and two 1 bit register elements. The ProASIC FPGA family from Actel [1] uses the name *VersaTile* configured either as a 3 input LUT or a 1 bit register. All these different definitions complicate the work of the designer regarding comparison of the size of the FPGA devices from different vendors.

All these small elements are placed in a regular structure connected together with different routing resources. The blocks located close to each other are grouped into a larger unit or cluster by short routing. These clusters are again connected together by longer routing resources. The global nets used for clock and reset signals need special attention because it is important that these nets have equal timing delay to all the logic elements in the FPGA. If a clock edge arrives to the logic elements at different point in time, there is a risk that the digital design will not work properly.

In addition, other special resources are found in an FPGA. Usually, an FPGA contains a number of SRAM blocks. A typical size of a SRAM block is 4Kbit, but the read and write bus widths are configurable and several RAM blocks can be put together constructing larger memories. Other components found in an FPGA are phase-lock loop (PLL) for generating additional clock signals, multipliers for performing digital signal processor (DSP) functions, CPU cores and communication interface modules.

The digital design is described in a hardware description language (HDL) where VHDL and Verilog are the two most commonly used. The HDL files are input to the software tools running synthesis and place and route (P&R). The synthesis is a process of verifying, optimizing and translating the hardware description into a netlist defining the hardware primitives needed in the digital design and the connections between these primitives. The P&R is an iterative process searching for an optimal placement of the primitives defined in the netlist into physical LUTs and registers in the FPGA and creating the routing between these elements. The P&R process can be done automatically by the SW tools based on the settings done by the operator, but it is also possible to do the P&R operation more manually. In this thesis, only automatic P&R are used based on the default settings of the

SW-tools. The final output is a binary file or bit-file containing the configuration of the FPGA. Typically, this bit-file is loaded into the FPGA through the JTAG interface. When the configuration is stored inside the FPGA, it will implement the digital design according to the hardware description the user has made.

2.3 Noise Sources in FPGA

The two most used physical noise sources in an FPGA generating true randomness are jitter of clocks and metastability on signals. Usually, both jitter and metastability are unwanted properties in an electronic design, but in a TRNG they contribute with the unpredictable behavior necessary for generating the true randomness.

Jitter can be looked upon as short-term variations of a digital signal's significant instants from their ideal positions in time [52]. The randomness arises because the exact position in time when a transition occurs is unpredictable. The jitter typically follows a Gaussian distribution characterized by a zero mean and a standard deviation σ [25, 33, 55]. The size of the jitter is related to the technology used in the integrated circuit.

A flip-flop in a digital system has two stable states: logical zero (ground (GND) or 0V) and logical one (V_{cc} typically 3.3V). If the flip-flop samples a signal during the setup or hold times of a transition, the result could be that the voltage level at the output of the flip-flop is neither GND nor V_{cc}, but at an intermediate undefined level (metastability). After a short period of time the voltage level will return to one of the two stable states, but to which state it stabilizes is unpredictable. This phenomenon can be described by dropping a ball on a hill and observing to which side of the hill (represented by the two stable logic levels) the ball lands [4]. The duration of the metastable state depends on where the ball is dropped. If it falls right on the top it takes longer time before it rolls down to one of the sides than if the ball is dropped on the slope of the hill.

Care must be taken when implementing these noise source circuits in HDL because they typically consist of combinational loops. These structures can be removed during the synthesis process if the tools are not instructed to preserve them. The actual implementation needs to be examined and tested in order to verify that the circuit is correctly implemented. Both jitter and metastability are unwanted properties in ordinary digital designs. The manufacturers will always design their integrated circuits in such a way that the size of the jitter is minimized and that the probability of getting metastable states in the device is as low as possible.

2.4 TRNG Implementations in FPGA

As seen in Figure 2.1, a typical implementation of a TRNG consists of three parts. Firstly, a physical random noise source, which in the case of an FPGA is typically jitter or metastability. Secondly, a digitizer, which collects the randomness from the entropy source and generates a binary bit sequence. Thirdly, a post-processor, which improves the statistical properties of the random bit sequence by performing whitening and data compression.

Several implementations of TRNG in FPGA have been proposed [50], and in the rest of this section some of them are presented with the focus on the used noise source and the digitizer circuit.

In 2002, Fisher et al. [20] used an analog PLL in Altera FPGAs as entropy source in a TRNG, see Figure 2.2. The generated signal from the PLL has jitter and it is sampled in a D flip-flop clocked with a master clock. The frequencies of the master clock and the PLL output are chosen to be relatively prime. The decimator circuit is used for removing the deterministic part of the random signal. In order to increase the bit rate, this TRNG can be extended by XORing the sampled outputs from a delay chain of the PLL output clock signal [19]. This TRNG design is restricted to FPGAs containing an analog PLL excluding

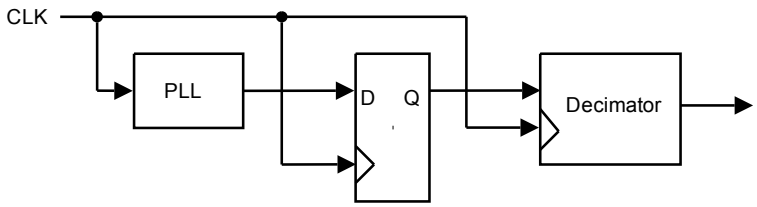

Figure 2.2: Principle of TRNG by Fisher et al. [20].

for instance FPGA devices from Xilinx containing a digital PLL. The achieved bit rate of this TRNG was less than 1Mbit/s.

In 2002, Tkacik [53] proposed a TRNG using a linear feedback shift register (LFSR) and a cellular automata shift register (CASR) clocked by two independent free running oscillator rings, see Figure 2.3. Selected outputs from the LFSR and the CASR are combined by an XOR generating the final random signal. Since the LFSR/CASR structures contain memory elements, the output bits are correlated. There must therefore be a delay of more than twice the length of the LFSR register between the sampling of each output bit to achieve independent random bits. An attack on this TRNG design was presented by Dichtl [13] under the assumption that the feedback polynomials of the LFSR and the CASR and the permutation before the XOR are known. Schindler [41] presented a stochastic model and a lower limit of the entropy per bit of this TRNG scheme.

In 2003, Epstein et al. [18] proposed a TRNG based on a bi-stable memory element consisting of two multiplexers and two inverters, see Figure 2.4. When the select signal is at logic zero, the circuit reduces to two single inverter oscillator rings which are put into a metastable state. When the select signal is logic high, the circuit resolves the output of the two oscillator rings and stabilizes either to logic zero or logic one and thereby creating randomness. The outputs of several of these memory elements are then connected together by the use of XOR to generated the final output bit sequence.

In 2004, Kohlbrenner et al. [29] used two oscillator rings where each ring consisted of two transparent latches, a buffer and an inverter, see Figure 2.5. The generated clock signal from the oscillator ring contains jitter and acts as an entropy source. The sampler circuit was a D flip-flop where the output of one of the oscillator rings was connected to the data input while the other ring output was connected to the clock input. In addition, the sampler circuit decides when a new random bit is ready on the output. The frequencies of the two

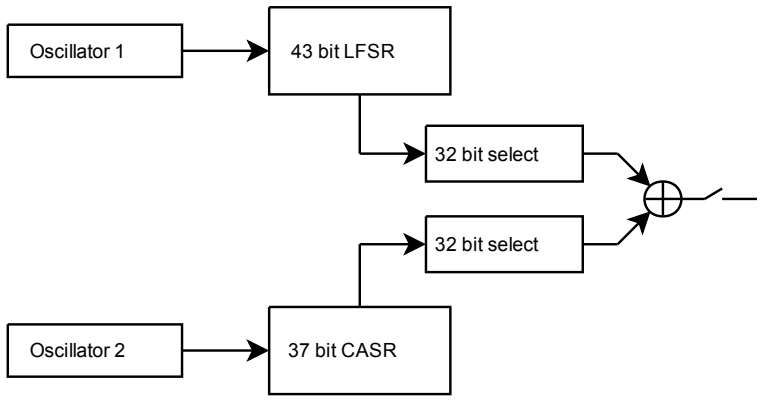

Figure 2.3: Principle of TRNG by Tkacik [53].

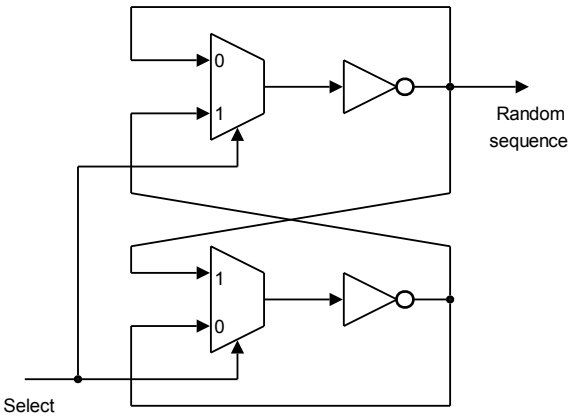

Figure 2.4: Principle of TRNG by Epstein et al. [18].

oscillator rings have to be almost identical or correctly matched if this implementation is going to work properly. The design achieved a bit rate of less than 1Mbit/s.

In 2006, Golić [22] proposed a TRNG using a Galois ring oscillator (GARO) and a Fibonacci ring oscillator (FIRO), see Figure 2.6. These LFSR like structures use inverters as delay elements instead of register elements, and the switches are either open or closed implementing the feedback polynomial. Continuous oscillations in these structures are achieved if the feedback polynomials satisfy certain requirements [22]. The outputs from the GARO and the FIRO are combined with an XOR and the random sequence is generated by sampling the XOR output with a D flip-flop. This design was further investigated and implemented by Dichtl et al. [15]. The FIRO/GARO structures behave in an analog manner creating an output signal similar to a noise signal, and they may therefore be susceptible to crosstalk from other signals inside the FPGA. The achieved bit rate of this TRNG was reported to be 12.5Mbit/s.

In 2007, Sunar et al. [51] gave a theoretical proposal of a TRNG based on several equal

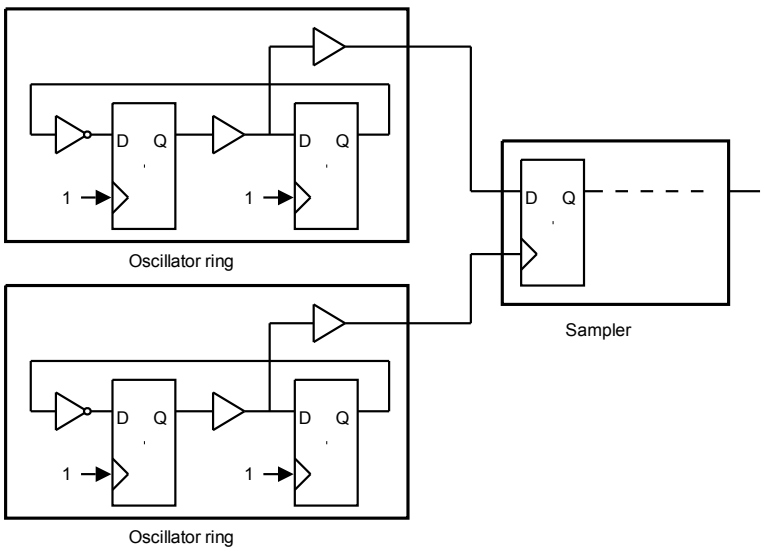

Figure 2.5: Principle of TRNG by Kohlbrenner et al. [29].

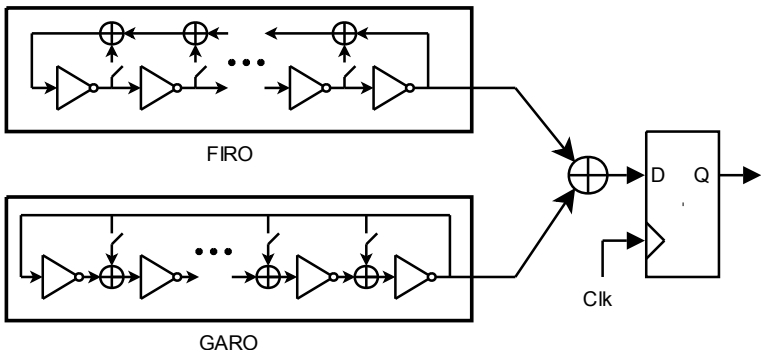

Figure 2.6: Principle of TRNG by Golić [22].

length oscillator rings made up of an odd number of inverters (114 oscillator rings where each ring consists of 13 inverters), see Figure 2.7. The main idea is to use a relatively large number of oscillator rings combined by the XOR operation in order to ensure that at each time at least one of the oscillator ring is sampled near a transition point with a sufficient high probability. The observation that an output bit obtained by sampling an oscillator ring signal is more random if the sampling occurs near a transition edge, was pointed out by Bock et al. [8]. The underlying assumption for the calculations based on this TRNG design, is that jitter of the individual ring oscillators are statistically independent. Imbalances between zeros and ones in the random signal are corrected by using a post-processor based on a resilient function. An implementation of this design was carried out by Schellenkens et al. [40] with only three inverters in each oscillator ring. The reported achieved bit rate of this TRNG included the post-processor was 2.5Mbit/s. This design was criticized by Dichtl et al. [15] mainly because the current CMOS technology is not capable of handling the high number of transitions per time unit generated by all the oscillator rings at the input of the XOR-tree. In addition, the assumption that jitter in individual oscillator ring signals is independent is not realistic due to interactions between these signals in practical implementations, and that the urn model introduced in [51] does not include the fact that the standard deviation of jitter accumulates in time. In 2008, Wold et al. [63] proposed an enhancement of the design of Sunar by adding a D flip-flop between each oscillator ring

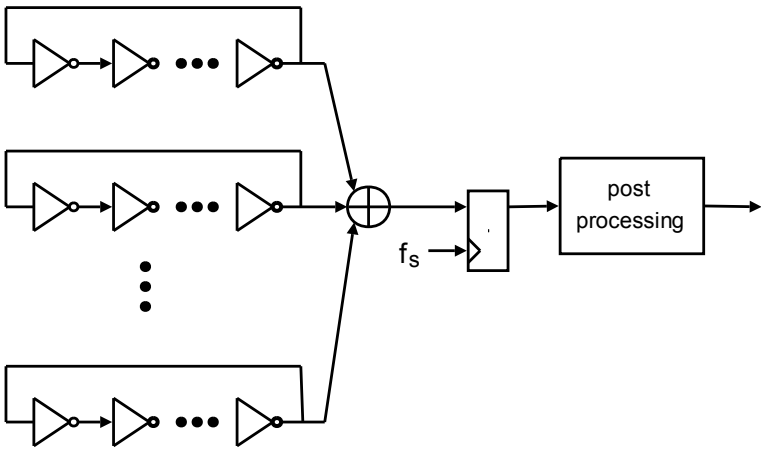

Figure 2.7: Principle of TRNG by Sunar et al. [51].

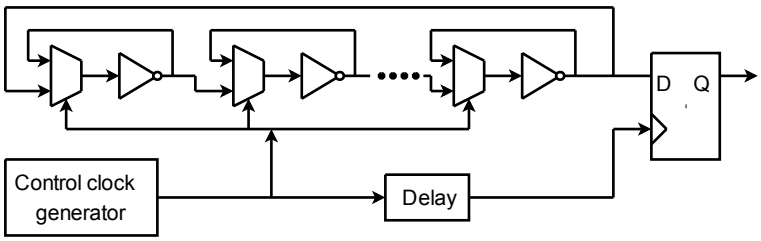

Figure 2.8: Principle of TRNG by Vasyltsov et al. [57].

output and the XOR tree correcting the speed problem pointed out in [15]. Experiments performed on the enhanced TRNG showed that the statistical tests passed with the use of only 25 oscillator rings. The reported bit rate of this TRNG was 100Mbit/s.

In 2008, Vasyltsov et al. [57] proposed a TRNG based on a 5-stage metastable ring oscillator where each stage contains only a multiplexer and an inverter, see Figure 2.8. The multiplexers are controlled such that in the first phase each of the five inverters is connected in a loop (output to input) so that they are put into a metastable state. In the second phase, the five inverters are connected as an oscillator ring with initial values depending on the metastable states. The idea of switching between metastable circuit configurations and oscillating configurations is based on the work by Epstein et al. [18]. A random bit is sampled at the output of the ring at the end of the second phase. The result is a fast and small implementation of a TRNG in an FPGA or ASIC, but optimization in the synthesis process causes difficulties in an FPGA implementation and the placement of the digital logic in the elements inside the FPGA is critical. This implementation uses very little of the resources in the FPGA and the reported bit rate was 140Mbit/s.

In 2009, Danger et al. [12] proposed a TRNG based on metastability with open loop structures in FPGA. The principle of this design is a delay chain where sampled signals from several tap points in the chain are XORed together and then sampled to create the random signal, see Figure 2.9. The delay between the elements is crucial and special care must be taken in the routing of the delay chain. The reported bit rate of this TRNG was 20Mbit/s.

In 2009, Gyorfi et al. [24] proposed a TRNG using the block RAM in a Xilinx FPGA as entropy source. The method is based on writing a logic one and a logic zero simultaneously to the same memory location in a block RAM configured as a dual port. The resulting value of the written bit is unpredictable due to the attempt to write different values at the same

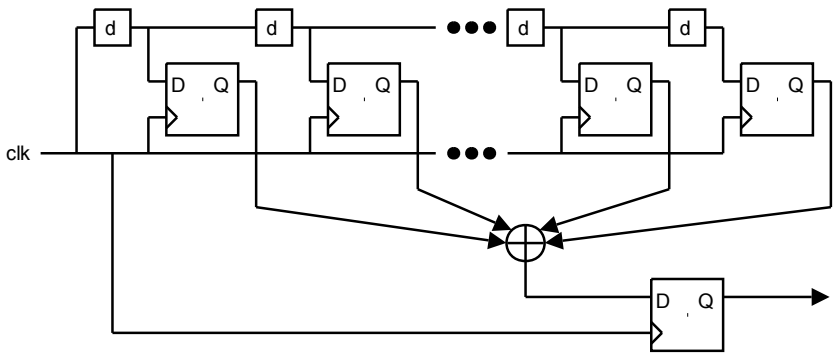

Figure 2.9: Principle of TRNG by Danger et al. [12].

TRNG	Speed [Mbit/s]	# LEs
Analogue PLL (Fisher et al. [20])	<1	∼50
LFSR/CASR (Tkacik [53])	<1	∼150
Two oscillator rings (Kohlbrenner et al. [29])	<1	∼50
FIGARO (Golic [22])	12.5	∼50
Multi oscillator rings (Sunar et al. [51])	2.5	∼1800
Multi oscillator rings (Schellenkens et al. [40])	2.5	∼650
Multi oscillator rings (Wold et al. [63])	100	∼100
5-stage oscillator ring (Vasyltsov et al. [57])	140	∼5
Open loops (Danger et al. [12])	20	∼140

Table 2.2: Resources and speed of selected TRNGs in FPGA.

time. The problem of this method is finding a memory location in the RAM giving an unbiased result. Their proposal is to use a circuit seeking for this optimal location at a regular basis. A disadvantage with this design is that it is not portable to other FPGAs than devices from Xilinx. A post-processor is needed in order to improve the statistical properties of this TRNG.

Recently, Varchola et al. [56] proposed a new high entropy digital element based on a bi-stable circuit named transition effect ring oscillator (TERO) implemented in an FPGA. The randomness is extracted on oscillatory trajectory when the TERO resolves a metastable event.

In addition, other TRNG designs have been proposed such as [5, 8, 36, 54].

The number of logic elements occupied in an FPGA for most of these presented TRNGs is low, approximately 50-150 which is less than 1% of the available resources in a medium sized FPGA. An exception is the original proposal of Sunar et al. [51] which uses about 1800 logic elements due to the large number of oscillator rings and many inverters in each ring.

Table 2.2 gives a summary of different TRNGs with a comparison regarding output bit rate and usage of logic elements (LEs) in an FPGA.

2.5 Post-processing

A post-processor is a circuit or algorithm used for improving the quality of the generated random bit sequence. The post-processor has two purposes: Firstly, it should adjust for statistical weaknesses in the generated bit sequence from the noise source. Secondly, it should increase the entropy of the random number by performing data compression at the cost of a reduced bit rate of the generator.

Several different post-processor schemes exist where one of the simplest is to XOR two and two consecutive bits of the random bit sequence. This method reduces the bias if the original bits in the sequence are independent. The resulting sequence has a bit rate, which is half of the original one. Dichtl [14] has proposed a special XOR function by combining several consecutive bits resulting in a larger reduction in the bias compared to the simple two-bit XOR.

Another post-processing method is the von Neumann extractor [58]. Two and two consecutive bits are investigated and the following rules are followed: if the two bits are 01 the result is 0, if the bits are 10 the result is 1 and if the bits are 00 or 11 the output bit is omitted. The output sequence of a von Neumann extractor has good statistical properties, but the resulting bit rate is dependent on the input bit sequence and on average the bit rate is about 4 times lower than the original bit rate. Another problem is that the delay between the generated bits is unpredictable and may be very long in the worst case.

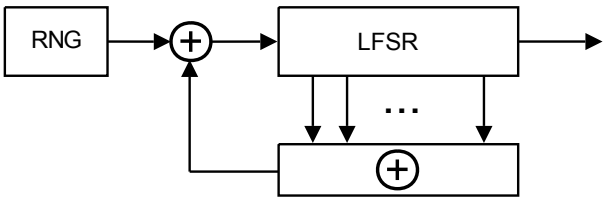

Figure 2.10: Post-processor based on an LFSR.

Another alternative is to use a maximum length LFSR with length n where the random bit sequence is XORed with the feedback in the LFSR as shown in Figure 2.10. The resulting bit rate after the LFSR is identical to the original one, but the first $2n$ bits should be omitted due to the initial state and the latency through the LFSR. An LFSR runs through all possible $2^n - 1$ bit combinations giving excellent statistical properties with no bias and perfect distribution of blocks/gaps according to the postulates of Golomb. However, this method will not increase the entropy per bit, but it will transform eventual statistical weaknesses into others, for instance bias into dependency [43]. In [22], a post-processing scheme consisting of a clock-controlled LFSR with additive input is proposed.

Two more sophisticated post-processor schemes are the extraction functions proposed by Barak et al. [6] and the resilient functions proposed by Sunar et al. [51].

All these post-processor algorithms have in common that they are easy to implement and occupy only a small portion of the total number of logic elements in an FPGA. Hash functions such as MD5 and SHA-1 are suited for use as post-processors in software on computers, but these hash functions are more complicated to implement in an FPGA and require a large number of logic elements.

2.6 Testing of Randomness

The output of the TRNG has to be tested to verify the randomness. The most common used method is to run a battery of statistical tests on the output sequence. The input to the statistical test program is a large file containing the generated bits from the TRNG. The results from the tests are documented in generated report files and typically these reports have to be interpreted in order to determine whether the TRNG has passed the test or not.

Two examples of such test suites are SP 800-22 from NIST [35] and DIEHARD [32]. For both these test suites a number of statistical sub-tests are performed and p-values representing the probability of passing the tests are calculated. In the DIEHARD test suite, all these p-values (approximately 240) should be uniformly distributed in the interval (0,1), i.e. the sorted p-values should follow the diagonal. If not, the test has failed. In the NIST test suite, the input bit sequence is divided into a number of blocks (preferably more than 100) containing at least 1 Mbit each. A p-value is calculated for each block for each subtest. The NIST test has passed if the p-values from each subtest can be regarded as uniformly distributed in the interval (0,1) and if a large majority of the blocks passes the tests for each subtest. For both these tests, the generated bit sequence used as input must be long.

Some typical tests performed in a statistical test suite are:

- Frequency or monobit test - the number of ones and zeros should be approximately equal.
- Runs test - the number of consecutive ones and zeros (blocks/gaps) should follow the second Golomb's postulate.
- Long runs test - a sequence of ones or zeros should not be too long.

- Discrete Fourier Transform (DFT) test - the frequency spectrum should not contain distinct components indicating periodic patterns.
- Universal Maurer's test - compression of a random sequence without loss of information should not be possible.
- Linear complexity test - the sequence should be more complex than sequences generated by an LFSR.
- Serial test - an m-bit pattern should have equal probability to occur in the sequence compared to other m-bit patterns.

An RNG needs to be evaluated and qualified according to a defined standard such as AIS31 [2] and FIPS 140-2 [34] before it is used in a real cryptographic system. These standards define criteria and guidance to evaluate and verify an RNG design.

In a real implementation of an RNG in a cryptographic system an online test should be included. The purpose is to detect a total breakdown and statistical weaknesses of the noise source output. An easy test is to periodically count the number of zeros and ones and the number of transitions during a specified period of time. If the levels exceed predefined limits, an alarm is given and the cryptographic system is shut down. These tests should be able to detect a fault situation fast in order to prevent security violations.

The statistical tests will reveal whether there are major problems with a generated bit sequence, but they cannot distinguish between true randomness and good pseudo randomness. An alternative test of verifying the presence of true randomness is to run repeated restarts of the TRNG from an identical initial state as originally proposed by Dichtl et al. [15]. The generated bit sequences after such an experiment are identical if the generator only produces pseudo randomness, but they are different (except for a short start-up period) if the generator produces true randomness.

2.7 Security

An implementation of a cryptographic system in an FPGA instead of using an ASIC has some advantages. For instance, the configuration can be changed in order to perform modifications or error corrections and the development process is easier and faster leading to a shorter time-to-market. But the flexibility of an FPGA is also a security problem leading to many different attack scenarios [16, 65]. The FPGA itself is not secure so the security has to be built into the whole cryptographic system. An example could be tamper protection where sensitive information such as keys is erased when an attempt of getting illegal access to the cryptographic system is detected. This can be carried out by surrounding the FPGA with a mesh encapsulated in epoxy-material. If a wire in the mesh is broken due to an attempt to get physical access to the FPGA, an alarm is given. In addition, different kind of sensors can for instance monitor temperature or movements. If sudden temperature changes or large movements are detected indicating unnatural behavior, an alarm is given and the cryptographic system is stopped.

A weak point of the SRAM FPGA is the uploading of the configuration at power on. An attack could be to exchange the configuration of the FPGA by changing the properties of the TRNG and thereby drastically reducing the security of the cryptographic system. The configuration can be protected by using encryption and integrity protection of the bitfile containing the configuration. Most FPGA vendors offer such an option for their more advanced FPGA families. Also, some SRAM FPGAs can constantly monitor the configuration by performing a cyclic redundancy check (CRC) and thereby detecting an attempt of modifying the configuration.

A possible method of manipulating the TRNG is to introduce a bias in the random sequence, i.e. changing the distribution of zeros and ones compared to the ideal case. In order to check the quality of randomness, the TRNG output should be monitored by an

online test with alarms. Another possibility of increasing the security is duplication of vital components in the design such that the functionality is preserved even if one part is violated or not working properly.

Markettos et al. [31] showed how to attack an TRNG implemented in a secure micro controller on a smart card by using frequency injection. The idea was to lock the frequency of an oscillator ring to the frequency of an injected signal and thereby eliminate the source of random jitter, which the TRNG relies on.

2.8 Signal Integrity

Signal integrity is related to problems such as timing, noise or electromagnetic interference (EMI) arising on a high-speed circuit board affecting the performance. Since the trend of electronic technology is towards faster devices resulting in an increase in the clock frequency and a decrease in the rise and fall times of the integrated circuits, signal integrity becomes an important topic when designing a product. The noise sources related to signal integrity can be divided into the following four different categories [9]:

- Signal quality of one net (reflections and distortions from impedance imperfections).
- Crosstalk between two or more nets (mutual capacitance and inductance coupling).
- Rail collapse in the power or ground distribution (voltage drop across power or ground network impedance).
- EMI and radiation from the system disturbing the operation of a device or a circuit board.

All these factors are potential threats against the operation of an electrical circuit and should be considered in every electronic high-speed design of some complexity. Yoo et al. [67] have considered some of these aspects related to the usage of oscillator rings in programmable logic.

2.9 Side-Channel Attacks

One of the goals of an attacker on a cryptographic system is to determine the encryption key, which makes the attacker capable of retrieving the plain text. A possible method is to perform a side-channel attack by measuring a property of the electric device while the encryption is performed. Examples of such properties are measurements of time [27], power consumption [28] and electromagnetic radiation [21, 39]. Based on these measurements, several different strategies exist in order to reveal the key. Simple analysis is visual inspection of one trace, differential analysis exploits the correlation between several traces combined with the knowledge of the used cryptographic algorithm and template attacks [11] where first a characterization is performed and then this characterization is used in the actual attack. The side-channel attacks were first introduced related to smart cards, but these attacks can also be used for attacking FPGAs [37, 38, 46, 47, 48]. A comprehensive summary of the topic of power analysis and the statistical techniques used can be found in [30].

The power consumption of a CMOS integrated circuit consists of two contributions. Firstly, the static power consumption related to the leakage current in a transistor. This leakage current is small in CMOS technology, but with decreasing dimensions of the transistors the leakage current increases. Secondly, the dynamic power consumption related to charging and discharging the load and parasitic capacitances when switching the voltage level at the input of the transistors. The dynamic power consumption is dominant in a CMOS device and it depends among other things on the activity or the number of transitions from logic 0 to 1 or vice versa. The fact that the dynamic power consumption

depends on the used data is exploited in a side-channel attack. Several different combinations of data are run on the cryptographic device and for each combination a voltage trace proportional to the power consumption is measured with an oscilloscope. The secret key is revealed by investigating the variations in the captured traces by means of a statistical analysis. An important step in this process is to obtain an accurate model of the power consumption. The more precise this model is, the less measurements or traces are needed in order to reveal the secret information. A simple and often used model of the power consumption is to compute the Hamming distance (HD), i.e. count the number of $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions during a time interval. An alternative of measuring the power consumption is to measure the electromagnetic radiation by using an antenna placed nearby the cryptographic device. Since the variations in the radiation are related to changes in the electrical current, this method is similar to using the power consumption.

Chapter 3

Contributions and Summary

3.1 Paper Contributions

This section consists of a summary of the contributions of each of the included papers.

3.1.1 Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings [64]

The contribution of the paper [64] is a useful enhancement of the TRNG proposed by Sunar et al. [51] by inserting D flip-flops between each of the oscillator rings and the XOR-tree. In addition, the number of oscillator rings was reduced to 25 based on the experimental results and the number of inverters in each oscillator ring was chosen to be 3, which is the lowest number necessary to produce oscillations. The proposed enhancement removed the bias from the output sequence of the TRNG improving the quality of the random bit sequence such that a post-processor was not needed in order to pass the NIST and DIEHARD statistical test suites. With a usage of less than 100 logic elements in an Altera Cyclone II FPGA device, a bit rate of 100 Mbit/s was experimentally achieved with this TRNG.

The distribution of the frequencies of equal length oscillator rings implemented in a Cyclone II FPGA was examined by measuring the outputs of 64 rings with an oscilloscope. The experiment was repeated for selected number of inverters in the oscillator rings starting with 3 and ending with 101. The frequencies varied due to the placement of the inverters in the logic elements in the FPGA and also due to the length of the routing between the elements containing the inverters. The distribution of the frequencies was inspected showing that at short lengths of the rings the frequencies created clusters. In addition, they spanned over a wide range. For longer lengths, the distribution approached a Gaussian distribution with smaller variations in the frequencies. Based on the measurements, the dispersion was calculated for all the investigated oscillator ring lengths. The dispersion of oscillator ring frequencies in the Altera Cyclone II SRAM FPGA was found to be the highest when the number of inverters was as low as possible, namely 3. This indicates that 3 is a reasonable choice for the number of inverters that ensures good properties of the TRNG with minimal consumption of logic elements in the FPGA.

The number of rings necessary to obtain good statistical properties in this TRNG was determined by using a statistical analysis showing that the number of rings hitting the transition region in the sampling period followed a Poisson distribution with parameter $\lambda = k \cdot r$ where r is the number of rings and k is a constant depending on the size of the jitter compared to the sampling period.

A restart experiment was carried out to verify the presence of true randomness. 1000 voltage traces of the TRNG output sequence were captured starting from a known reset state. The standard deviation between the traces was calculated and it converged to a value higher than zero indicating that this TRNG produces true randomness and not only pseudo randomness. The measurements also showed that the standard deviation followed the theoretical values based on an ideal random number sequence.

3.1.2 Optimizing Speed of a True Random Number Generator in FPGA by Spectral Analysis [60]

The contribution of the paper [60] is a spectral analysis of the TRNG based on XOR of several sampled equal length oscillator rings. The analysis was performed by investigating the frequency spectrum after the different components in the TRNG both in theory and by using simulations in Matlab. The spectrum was investigated after an oscillator ring, after the D flip-flop sampling the oscillator ring, after the XOR of all oscillator rings and finally at the output after the last D flip-flop. The Matlab simulation of the frequency spectrum of the output of the TRNG was found to be in accordance with the theoretical spectrum of an ideal RNG.

The purpose of this investigation was to explain the operation of this TRNG in detail, but also to determine the optimal design parameters that give a high bit rate. Design rules regarding the selection of the sampling frequency compared to the frequency of the rings were given aiming at optimizing the throughput of this TRNG.

Based on these rules, optimal design parameters were selected and the TRNG was implemented in an Altera Cyclone II FPGA resulting in a bit rate of 300 Mbit/s. The generated random data was examined by means of NIST and DIEHARD statistical test suites. The data passed the tests verifying that the quality of the produced randomness was satisfactory. A restart experiment was performed showing that the generated random data could be regarded as true random and not pseudo random.

3.1.3 Robustness of TRNG against Attacks that Employ Superimposing Signal on FPGA Supply Voltage [61]

The contribution of the paper [61] is an investigation of the robustness of four different TRNG designs against an attack superimposing a signal on the FPGA power supply voltage. Two of the designs were based on multiple oscillator rings (Sunar et al. [63] and Wold et al. [51]) while the two other designs were the FIGARO design by Golic [22] and the 5 stage metastable oscillator ring proposed by Vasyltsov et al. [57]. These four designs were implemented in an FPGA and their behavior was observed while performing the attack. The generated output sequences were analyzed regarding the amount of zeros and ones (bias) compared to the ideal 50% level. The result was that the two designs based on oscillator rings were not influenced by such an attack, while the two others were. When the attack was optimized by adjusting the noise frequency, the resulting bias of the two TRNG designs not consiting of oscillator rings was so high that it would have been considered a security problem in a high-grade cryptographic system.

Simulations in Matlab were performed in order to explain why a TRNG based on several equal length oscillator rings is not influenced by an attack that employs superimposing signal on the FPGA power supply voltage. The reason is that altering the power supply voltage changes the frequency of the oscillator rings, but this class of TRNG depends on the position in time of the transitions from the logical zero to the logical one and vice versa and not on the amplitude level of the signal. These TRNGs are therefore robust against changes in the amplitude level performed by this attack.

3.1.4 Security Implications of Crosstalk in Switching CMOS Gates [17]

The contribution of the paper [17] is a proposed model of the energy dissipation on a data bus in a microprocessor system based on layout dependent phenomena such as the crosstalk influence between the bus lines due to capacitive couplings between the lines. This model is more precise than a simple Hamming-distance model and can be used in a side-channel attack on a semiconductor device. An example could be an attack on the bus transferring the generated random bits between the post-processor and the output interface of a TRNG showed in Figure 2.1. An empirical equation was derived in order to

quantify the energy dissipation of a bit transition on a data bus with crosstalk between the bus lines. A model of an 8 bit data bus with inverters and capacitances between neighboring bus lines was implemented in PSPICE and simulations of the energy dissipation were performed for many different bit combinations and transitions. The results from the simulations were compared with the theoretically obtained values verifying the validity of the proposed empirical equation. With the proposed model the detection performance on a data bus is improved compared to the traditional Hamming-distance model.

3.1.5 Behavioral Model of TRNG Based on Oscillator Rings Implemented in FPGA [62]

The contribution of the paper [62] is a model explaining the amount of entropy generated in a TRNG based on XOR of several sampled equal length oscillator rings. The model was based on the principles proposed by Sunar et al. [51]. The number of hits in a transition region defined by the size of the jitter at every sampling point was investigated. The novel approach is that the proposed model takes into account that the jitter of an oscillator ring is accumulated when there is no hit at a sampling point. The result is that the transition region increases until a hit is registered resetting the accumulated jitter. Due to the accumulation of jitter, the probability of hitting the interesting transition region varies as opposed to the urn model of Sunar et al. [51] where this probability is constant.

The model was implemented in Matlab such that the influence of the variation of the design parameters (the number of oscillator rings, the sampling frequency and the number of inverters in an oscillator ring) and the technology properties (the size of the jitter and the size of the dispersion between the oscillator ring frequencies) could be investigated. The simulations showed that with the choice of design parameters proposed in [64], there is a high probability that at least one oscillator ring is sampled in the transition region defined by the size of the accumulated jitter at every sampling point indicating that every generated bit contains entropy.

The simulations showed that the number of oscillator rings was the most important design parameter necessary to be optimized in order to achieve high probability of hitting the transition region. The size of the jitter was also important, but this property is determined by the technology of the semiconductor device and cannot be altered by the design. The size of the dispersion of the oscillator ring frequencies had little effect on the number of hits, but a large dispersion is important to achieve randomness quickly after the start-up of the TRNG.

3.1.6 Security Properties of Oscillator Rings in True Random Number Generators [59]

The contribution of the paper [59] is an investigation of the hardware properties of oscillator rings implemented in an FPGA and their implications on the security of a TRNG based on XOR of several sampled equal length oscillator rings. All the experiments were performed on three different FPGAs where two of them were SRAM based (Cyclone II from Altera and Spartan3A from Xilinx) while the last one was flash based (ProASIC3E from Actel).

The interaction between two oscillator rings consisting of 15 inverters each located close to each other in an FPGA was investigated by measuring the frequency spectrum on an oscilloscope of one ring while enabling and disabling the other ring. The result was that new peaks related to the other ring were observed in the frequency spectrum of the first ring. The magnitude of these peaks was more than 30dB lower than the first harmonic frequency component of the signal generated in the measured ring. In the time domain, no differences were observed on the oscilloscope when enabling and disabling the second oscillator ring.

An investigation of the dispersion of the oscillator ring frequencies was carried out in the same manner as in [64], but with the extension of looking for differences among FPGAs from various vendors and with different technologies. For the two SRAM based FPGAs the highest dispersion was found for three inverters in the ring, but for the flash based FPGA from Actel the maximum dispersion was found for nine inverters. This difference can be explained by investigating the architecture of the FPGAs and observing the placement of the inverters in the FPGAs. It is important to select the optimal number of inverters in a TRNG based on several oscillator rings in order to achieve high dispersion in the oscillator ring frequencies.

The correlation and dependence between the oscillator rings were investigated by capturing output bit sequences from 32 sampled oscillator rings consisting of three inverters. The result was that a majority of the rings were regarded as uncorrelated, but a few of them were correlated due to almost identical ring frequencies. The effect on a TRNG based on XOR of several oscillator rings is that a slightly higher number of rings should be selected in order to compensate for this correlation. Regarding dependence, the χ^2 test showed that it was not possible to state that the rings were independent, but the XOR structure of the TRNG removed this dependence.

3.2 Summary of Thesis Contributions

TRNGs are important components in cryptographic systems used for instance for generating session keys for the encryption. Usually, the cryptographic algorithm is public and therefore all the security is dependent on the key. The properties of the TRNG are thereby important in order to maintain high degree of security. In this thesis, an investigation of the security properties of a class of TRNG consisting of several oscillator rings implemented in programmable logic devices was carried out. The main contribution is the proposal of a TRNG based on XOR of several sampled oscillator rings with so good statistical properties that a post processor is not needed to pass the statistical test suites. An implementation based on this design was successfully used in a high-grade cryptographic system showing that this design is secure, robust and also easy and practical to implement in a programmable device such as an FPGA.

In order to understand this TRNG design, a detailed investigation was carried out. The operation of this TRNG was examined both in theory and by simulations in Matlab of the frequency spectrum of the different parts of this TRNG. The harvesting of randomness in this TRNG was understood by means of a model based on the accumulation of jitter. A hardware based analysis of the properties of the oscillator rings was also performed to observe the interactions, correlation and dependences between the rings implemented in FPGAs and their influence on the security of this TRNG.

The robustness of this TRNG was proved by performing an attack that manipulated the power supply voltage of the FPGA. The experiments measuring the bias of the output sequence show that a TRNG based on several oscillator rings was not influenced by this attack while two other reference TRNG designs were. The generated random bits are transferred over an internal bus inside the FPGA. A side-channel attack measuring the power consumption or the electromagnetic emission can be used to determine the random data. In order to perform this attack successfully, it is important to have a good model of the hardware behavior. Based on layout dependent phenomena such as capacitive couplings between the bus lines, a more precise model of energy dissipation on a data bus was proposed. The model was verified by simulations in PSPICE of an 8-bit bus and compared to the simple Hamming-distance model the detection performance was improved with the proposed model.

3.3 Future Work

It is important that a TRNG contains a physical noise source capable of generating bits with good statistical properties. The verification of these properties can be done by performing statistical tests, but these tests cannot reveal the amount of true randomness in the generated sequence. In order to determine the entropy per bit, the TRNG must be theoretically investigated. Baudet et al. [7] have performed a theoretical analysis of the sampled oscillator ring based on a Wiener stochastic process. An interesting and challenging task is to perform a similar analysis on a complete TRNG design based on XOR of several sampled oscillator rings aiming at determining a lower bound of entropy of this TRNG.

Side-channel attacks can be used for revealing the secret information or the key used in an encryption system. Based on the proposed power model exploiting layout dependent phenomena, the next step is to perform measurements of the power consumption or the electromagnetic emission on a TRNG design in an FPGA trying to detect the generated random data on a bus.

The properties of the oscillator rings implemented in FPGA were investigated by means of analysis of the experimental results. A model of the oscillator rings could be implemented in for instance PSPICE in order to simulate the interaction and crosstalk between the rings. This model could be used to verify the observed behavior on the real FPGA and also to further investigate the properties of these rings.

3.4 Bibliography

[1] ACTEL. ProASIC3 Flash Family FPGAs. October 2009. 8

[2] AIS 31. Functionality Classes and Evaluation Methodology for Physical Random Number Generators. Version 1. 2001. 16

[3] ALTERA. Cyclone II Device Handbook, Volume 1. 2008. 8, 30, 36, 53

[4] ALTERA. Understanding Metastability in FPGAs. White Paper WP-01082, July 2008. 9

[5] BAGINI, V., AND BUCCI, M. A Design of Reliable True Random Number Generator for Cryptographic Applications. In *Proceedings of the First International Workshop on Cryptographic Hardware and Embedded Systems (CHES'99)* (1999), vol. 1717 of *Lecture Notes in Computer Science*, Springer, pp. 204–218. doi:10.1007/3-540-48059-5_18. 14

[6] BARAK, B., SHALTIEL, R., AND TROMER, E. True Random Number Generators Secure in a Changing Environment. In *Proceedings of the 5th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'03)* (2003), vol. 2779 of *Lecture Notes in Computer Science*, Springer, pp. 166–180. doi:10.1007/978-3-540-45238-6_14. 15

[7] BAUDET, M., LUBICZ, D., MICOLOD, J., AND TASSIAUX, A. On the Security of Oscillator-Based Random Number Generators. *Journal of Cryptology 24*, 2 (2011), pp. 398–425. doi:10.1007/s00145-010-9089-3. 23, 126

[8] BOCK, H., BUCCI, M., AND LUZZI, R. An Offset-Compensated Oscillator-Based Random Bit Source for Security Applications. In *Proceedings of the 6th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'04)* (2004), vol. 3156 of *Lecture Notes of Computer Science*, Springer, pp. 268–281. doi:10.1007/978-3-540-28632-5_20. 12, 14

[9] BOGATIN, E. *Signal and Power Integrity - Simplified, Second Edition*. Prentice Hall, 2010. 17, 87

[10] BUCCI, M., AND LUZZI, R. Design of Testable Random Bit Generators. In *Proceedings of the 7th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'05)* (2005), vol. 3659 of *Lecture Notes in Computer Science*, Springer, pp. 147–156. doi:10.1007/11545262_11. 5, 7, 30, 98, 102

[11] CHARI, S., RAO, J. R., AND ROHATGI, P. Template Attacks. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 13–28. doi:10.1007/3-540-36400-5_3. 17, 71, 113, 118

[12] DANGER, J.-L., GUILLEY, S., AND HOOGVORST, P. High Speed True Random Number Generator Based on Open Loop Structures in FPGAs. *Microelectronics Journal 40*, 11 (November 2009), 1650–1656. doi:10.1016/j.mejo.2009.02.004. 13, 14, 31

[13] DICHTL, M. How to Predict the Output of a Hardware Random Number Generator. In *Proceedings of the 5th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'03)* (2003), vol. 2779 of *Lecture Notes in Computer Science*, Springer, pp. 181–188. doi:10.1007/978-3-540-45238-6_15. 10

[14] DICHTL, M. Bad and Good Ways of Post-processing Biased Physical Random Numbers. In *Proceedings of the 14th Annual Fast Software Encryption Workshop (FSE'07)* (2007), vol. 4593 of *Lecture Notes in Computer Science*, pp. 137–152. doi:10.1007/978-3-540-74619-5_9. 14

[15] DICHTL, M., AND GOLIĆ, J. D. High-Speed True Random Number Generation with Logic Gates Only. In *Proceedings of the 9th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'07)* (2007), vol. 4727 of *Lecture Notes in Computer Science*, Springer, pp. 45–62. doi:10.1007/978-3-540-74735-2_4. 11, 12, 13, 16, 30, 31, 32, 39, 64, 102, 103

[16] DRIMER, S. Volatile FPGA Design Security – A Survey (Version 0.96). Apr. 2008. 16

[17] DYRKOLBOTN, G. O., WOLD, K., AND SNEKKENES, E. Security Implications of Crosstalk in Switching CMOS Gates. In *Proceedings of the 13th Information Security Conference (ISC'10)* (2011), vol. 6531 of *Lecture Notes in Computer Science*, Springer, pp. 269–275. 4, 20

[18] EPSTEIN, M., HARS, L., KRASINSKI, R., ROSNER, M., AND ZHENG, H. Design and Implementation of a True Random Number Generator Based on Digital Circuit Artifacts. In *Proceedings of the 5th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'03)* (2003), vol. 2779 of *Lecture Notes in Computer Science*, Springer, pp. 152–165. doi:10.1007/978-3-540-45238-6_13. 10, 11, 13

[19] FISCHER, V., DRUTAROVSKÝ, M., SIMKA, M., AND BOCHARD, N. High Performance True Random Number Generator in Altera Stratix FPLDs. In *Proceedings of the 14th International Conference on Field-Programmable Logic and Applications (FPL'04)* (2004), vol. 3203 of *Lecture Notes in Computer Science*, Springer, pp. 555–564. doi:10.1007/978-3-540-30117-2_57. 9

[20] FISHER, V., AND DRUTAROVSKÝ, M. True Random Number Generator Embedded in Reconfigurable Hardware. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 415–430. doi:10.1007/3-540-36400-5_30. 9, 10, 14, 30, 45, 102

[21] GANDOLFI, K., MOURTEL, C., AND OLIVIER, F. Electromagnetic Analysis: Concrete Results. In *Proceedings of the 3th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'01)* (2001), vol. 2162 of *Lecture Notes in Computer Science*, Springer, pp. 251–261. doi:10.1007/3-540-44709-1_21. 17, 113

[22] GOLIĆ, J. D. New Methods for Digital Generation and Postprocessing of Random Data. *IEEE Transactions on Computers 55*, 10 (2006), 1217–1229. doi:10.1109/TC.2006.164. 11, 12, 14, 15, 20, 30, 46, 58, 63, 102

[23] GOLOMB, S. W. *Shift Register Sequences*. Aegean Park Press, Laguna Hills, CA, USA, 1981. 6

[24] GYORFI, T., CRET, O., AND SUCIU, A. High Performance True Random Number Generator Based on FPGA Block RAMs. *IEEE International Symposium on Parallel and Distributed Processing* (2009), 1–8. doi:10.1109/IPDPS.2009.5161207. 13

[25] HAJIMIRI, A., LIMOTYRAKIS, S., AND LEE, T. H. Jitter and Phase Noise in Ring Oscillators. *IEEE Journal of Solid-State Ciruits 34*, 6 (1999), 790–804. doi:10.1109/4.766813. 9

[26] JUN, B., AND KOCHER, P. The Intel Random Number Generator. *White paper prepared for Intel Corporation* (Apr. 1999). 5, 29, 101

[27] KOCHER, P. C. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *Proceedings of Advances in Cryptology (CRYPTO'96)* (1996), vol. 1109 of *Lecture Notes in Computer Science*, Springer, pp. 104–113. doi:10.1007/3-540-68697-5_9. 17

[28] KOCHER, P. C., JAFFE, J., AND JUN, B. Differential Power Analysis. In *Proceedings of Advances in Cryptology (CRYPTO'99)* (1999), vol. 1666 of *Lecture Notes in Computer Science*, Springer, pp. 388–397. doi:10.1007/3-540-48405-1_25. 17, 71, 113

[29] KOHLBRENNER, P., AND GAJ, K. An Embedded True Random Number Generator for FPGAs. In *Proceedings of the ACM/SIGDA 12th International Symposium on Field Programmable Gate Arrays (FPGA'04)* (2004), ACM, pp. 71–78. doi:10.1145/968280.968292. 10, 11, 14, 30, 45, 87, 102

[30] MANGARD, S., OSWALD, E., AND POPP, T. *Power Analysis Attacks - Revealing the Secrets of Smart Cards*. Springer, 2007. 17, 71, 113

[31] MARKETTOS, A. T., AND MOORE, S. W. The Frequency Injection Attack on Ring-Oscillator-Based True Random Number Generators. In *Proceedings of the 11th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'09)* (2009), vol. 5747 of *Lecture Notes in Computer Science*, Springer, pp. 317–331. doi:10.1007/978-3-642-04138-9_23. 17, 67

[32] MARSAGLIA, G. DIEHARD: A Battery of Tests of Randomness. 1996. 7, 15, 38, 46, 47, 77, 103, 108

[33] MCNEILL, J. A. Jitter in Ring Oscillators. *IEEE Journal of Solid-State Ciruits 32*, 6 (1997), 870–879. doi:10.1109/4.585289. 9

[34] NIST. Security Requirements for Cryptographic Modules. FIPS PUB 140-2, 2001. 16

[35] NIST. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Special Publication 800-22 Revision 1a, April 2010. 7, 15, 37, 38, 46, 47, 65, 77, 103, 108

[36] O'DONNELL, C. W., SUH, G. E., AND DEVADAS, S. PUF-Based Random Number Generation. Technical Memo 481, MIT CSAIL, November 2004. 14

[37] ORS, S. B., OSWALD, E., AND PRENEEL, B. Power-Analysis Attacks on an FPGA - First Experimental Results. In *Proceedings of the 5th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'03)* (2003), vol. 2779 of *Lecture Notes in Computer Science*, Springer, pp. 35–50. doi:10.1007/978-3-540-45238-6_4. 17

[38] PEETERS, E., STANDAERT, F.-X., DONCKERS, N., AND QUISQUATER, J.-J. Improved Higher-Order Side-Channel Attacks with FPGA Experiments. In *Proceedings of the 7th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'05)* (2005), vol. 3659 of *Lecture Notes in Computer Science*, Springer, pp. 309–323. doi:10.1007/11545262_23. 17

[39] QUISQUATER, J.-J., AND SAMYDE, D. ElectroMagnetic Analysis (EMA): Measures and Countermeasures for Smart Cards. In *Proceedings of the International Conference on Research in Smart Cards (E-smart'01)* (2001), vol. 2140 of *Lecture Notes in Computer Science*, Springer, pp. 200–210. doi:10.1007/3-540-45418-7_17. 17

[40] SCHELLEKENS, D., PRENEEL, B., AND VERBAUWHEDE, I. FPGA Vendor Agnostic True Random Number Generator. In *Proceedings of the 16th International Conference on Field-Programmable Logic and Applications (FPL'06)* (2006), IEEE, pp. 1–6. doi:10.1109/FPL.2006.311206. 12, 14, 31, 102

[41] SCHINDLER, W. A Stochastical Model and Its Analysis for a Physical Random Number Generator Presented at CHES 2002. In *Proceeding of the 9th International Conference of Cryptography and Coding (IMA'03)* (2003), vol. 2898 of *Lecture Notes in Computer Science*, Springer, pp. 276–289. doi:10.1007/978-3-540-40974-8_22. 10

[42] SCHINDLER, W. *Cryptographic Engineering*. Springer, 2009, ch. Random Number Generators for Cryptographic Applications, pp. 5–23. In C. K. Koc editor. 5, 7

[43] SCHINDLER, W. *Cryptographic Engineering*. Springer, 2009, ch. Evaluation Criteria for Physical Random Number Generators, pp. 25–54. In C. K. Koc editor. 7, 15

[44] SCHNIDLER, W., AND KILLMANN, W. Evaluation Criteria for True (Physical) Random Number Generators Used in Cryptographic Applications. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 431–449. doi:10.1007/3-540-36400-5_31. 5, 7

[45] SHANNON, C. Communication Theory of Secrecy Systems. *Bell System Technical Journal 28*, 4 (1949), 656–715. 7

[46] STANDAERT, F.-X., MACE, F., PEETERS, E., AND QUISQUATER, J.-J. Updates on the Security of FPGAs Against Power Analysis Attacks. In *Proceedings of the International Workshop on Applied Reconfigurable Computing (ARC'06)* (2006), vol. 3985 of *Lecture Notes in Computer Science*, Springer, pp. 335–346. doi:10.1007/11802839_42. 17

[47] STANDAERT, F.-X., PEETERS, E., ROUVROY, G., AND QUISQUATER, J.-J. An Overview of Power Analysis Attacks Against Field Programmable Gate Arrays. *Proceedings of the IEEE 94*, 2 (February 2006), 383–394. doi:10.1109/JPROC.2005.862437. 17

[48] STANDAERT, F.-X., VAN OLDENEEL TOT OLDENZEEL, L., SAMYDE, D., AND QUISQUATER, J.-J. Power Analysis of FPGAs: How Practical Is the Attack? In *Proceedings of the 13th International Conference on Field-Programmable Logic and Applications (FPL'03)* (2003), vol. 2778 of *Lecture Notes in Computer Science*, Springer, pp. 701–710. doi:10.1007/978-3-540-45234-8_68. 17

[49] STINSON, D. R. *Cryptography Theory and Practice, Third Edition*. Chapman & Hall/CRC, 2006. 1

[50] SUNAR, B. *Cryptographic Engineering*. Springer, 2009, ch. True Random Number Generators for Cryptography, pp. 55–73. In C. K. Koc editor. 9

[51] SUNAR, B., MARTIN, W. J., AND STINSON, D. R. A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. *IEEE Transactions on Computers 56*, 1 (2007), 109–119. doi:10.1109/TC.2007.250627. 4, 11, 12, 14, 15, 19, 20, 21, 30, 31, 32, 34, 37, 39, 41, 45, 53, 58, 63, 77, 78, 83, 87, 101, 102, 103, 105, 106, 108, 109, 125, 126

[52] TEKTRONIX INC. Understanding and Characterizing Timing jitter. 2003. 9, 30, 102

[53] TKACIK, T. E. A Hardware Random Number Generator. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 450–453. doi:10.1007/3-540-36400-5_32. 10, 14, 30, 45, 102

[54] TSOI, K. H., LEUNG, K. H., AND LEONG, P. H. W. Compact FPGA-based True and Pseudo Random Number Generators. In *Proceedings of the 11th IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'03)* (2003), pp. 51–61. doi:10.1109/FPGA.2003.1227241. 14

[55] VALTCHANOV, B., AUBERT, A., BERNARD, F., AND FISCHER, V. Modeling and Observing the Jitter in Ring Oscillators Implemented in FPGAs. In *Proceedings of the 11th IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems (DDECS'08)* (2008), pp. 1–6. doi:10.1109/DDECS.2008.4538777. 9, 57, 58, 59, 60, 66, 67, 79, 80

[56] VARCHOLA, M., AND DRUTAROVSKY, M. New High Entropy Element for FPGA Based True Random Number Generators. In *Proceedings of the 12th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'10)* (2010), vol. 6225 of *Lecture Notes in Computer Science*, Springer, pp. 351–365. doi:10.1007/978-3-642-15031-9_24. 14

[57] VASYLTSOV, I., HAMBARDZUMYAN, E., KIM, Y.-S., AND KARPINSKYY, B. Fast Digital TRNG Based on Metastable Ring Oscillator. In *Proceedings of the 10th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'08)* (2008), vol. 5154 of *Lecture Notes in Computer Science*, Springer, pp. 164–180. doi:10.1007/978-3-540-85053-3_11. 13, 14, 20, 31, 46, 58, 63

[58] VON NEUMANN, J. Various Techniques for Use in Connection with Random Digits. *von Neumann's Collected Works 5* (1963), 768–770. 14

[59] WOLD, K., AND PETROVIĆ, S. Security Properties of Oscillator Rings in True Random Number Generators. Submitted. 4, 21

[60] WOLD, K., AND PETROVIĆ, S. Optimizing Speed of a True Random Number Generator in FPGA by Spectral Analysis. In *Proceedings of the 4th IEEE International Conference of Computer Sciences and Convergence Information Technology (ICCIT'09)* (November 2009), pp. 1105–1110. doi:10.1109/ICCIT.2009.95. 4, 20

[61] WOLD, K., AND PETROVIĆ, S. Robustness of TRNG against Attacks that Employ Superimposing Signal on FPGA Supply Voltage. In *Proceedings of the Norwegian Information Security Conference (NISK'10)* (November 2010), pp. 81–92. 4, 20

[62] WOLD, K., AND PETROVIĆ, S. Behavioral Model of TRNG Based on Oscillator Rings Implemented in FPGA. In *Proceedings of the 14th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS'11)* (2011), pp. pp. 163–166. 4, 21, 125

[63] WOLD, K., AND TAN, C. H. Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings. In *Proceedings of the 4th IEEE International Conference on ReConFigurable Computing and FPGAs (ReConFig'08)* (2008), pp. 385–390. doi:10.1109/ReConFig.2008.17. 12, 14, 20, 42, 46, 47, 48, 52, 53, 54, 58, 64, 77, 78, 79, 80, 83, 87, 92, 99

[64] WOLD, K., AND TAN, C. H. Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings. *International Journal of Reconfigurable Computing* (June 2009), 1–8. doi:10.1155/2009/501672. 4, 19, 21, 22

[65] WOLLINGER, T., GUAJARDO, J., AND PAAR, C. Security on FPGAs: State-of-the-Art Implementations and Attacks. *ACM Transactions on Embedded Computing Systems 3*, 3 (August 2004), 534–574. doi:10.1145/1015047.1015052. 16

[66] XILINX. Virtex-II Platform FPGAs: Complete Data Sheet. November 2007. 8

[67] YOO, S.-K., KARAKOYUNLU, D., BIRAND, B., AND SUNAR, B. Improving the Robustness of Ring Oscillator TRNGs. *ACM Transactions on Reconfigurable Technology and Systems 3*, 2 (May 2010), 1–30. Article 9. doi:10.1145/1754386.1754390. 17, 59, 60, 67, 88, 95

Chapter 4

Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings[1]

Abstract

A true random number generator (TRNG) is an important component in cryptographic systems. Designing a fast and secure TRNG in an FPGA is a challenging task. In this paper we analyze the TRNG designed by Sunar et al. (2007) based on XOR of the outputs of several oscillator rings. We propose an enhanced TRNG with better randomness characteristics that does not require post-processing and passes the statistical tests. We have shown by experiment that the frequencies of the equal length oscillator rings in the TRNG are not identical. The difference is due to the placement of the inverters in the FPGA and the resulting routing between the inverters. We have implemented our proposed TRNG in an Altera Cyclone II FPGA. Our implementation has passed the NIST and DIEHARD statistical tests with a throughput of 100Mbps and with a usage of less than 100 logic elements in the FPGA. The restart experiments have shown that the output from our TRNG behaves truly random and not pseudo random.

4.1 Introduction

Traditionally, a high assurance implementation of cryptographic algorithms has been done in application specific integrated circuit (ASIC). During the recent years, more and more of these implementations are done in field programmable gate array (FPGA). There are several reasons for this development. The FPGA can be reprogrammed, leading to more flexibility for modification of algorithms, changing algorithms and fixing bugs. The development of an algorithm in an FPGA is easier and faster as compared to an ASIC design, resulting in a shorter time-to-market. In addition, the latest FPGA devices are manufactured with the state-of-the-art technology.

It is well known that a true random number generator (TRNG) is an important component of today's cryptographic systems. Typically a TRNG can be used for generating keys, initialization vectors, random sequences for cryptographic challenges-responses, etc. In a cryptographic system, a private or secret parameter is normally generated by a TRNG and is an interesting property to an attacker. Therefore, the generation of a random bit sequence is important and should be unpredictable to an attacker. One common method for generating a truly random sequence is to amplify the thermal noise in a diode [8]. The disadvantage of this method is the use of external components. This approach enables an attacker to manipulate and read the random bit sequence from the device and consequently violate the security of the entire cryptographic system. If the TRNG is implemented entirely inside the FPGA, an attacker will have difficulties in retrieving and manipulating the random bit sequence. The challenge is to design a TRNG in an FPGA passing all statistical tests and at the same time using as few resources as possible and achieving a high throughput of random bits.

[1] Knut Wold and Chik How Tan. In International Journal of Reconfigurable Computing, pp. 1-8, June 2009. Extended version of the paper at ReConFig'08, see Appendix A.

In this paper we examine more closely the TRNG based on oscillator rings proposed by Sunar et al. [13]. We show that the TRNG described in [13] is not random without post-processing. We propose an enhancement of the proposal from [13] and experimentally show improved performance with respect to FPGA resource usage and throughput. We also show that our TRNG has no bias and therefore no need for complicated post-processing. We experimentally demonstrate that the frequencies of the oscillator rings are different due to the placement and routing of the inverters inside the FPGA.

We have implemented our proposal in an Altera Cyclone II FPGA [1]. Our implementation of the TRNG based on oscillator rings passes the NIST and DIEHARD statistical tests with a throughput of 100Mbps and the usage of less than 100 logic elements in the FPGA. Repeated restarts of the TRNG from the same reset state have shown that the output of our random generator behaves truly random and not pseudo random. The standard deviation has been calculated from 1000 traces recorded after reset. A short start-up period should be omitted in order to obtain good quality of the randomness, but after the TRNG output stabilizes, the standard deviation becomes constant and in accordance with the theoretical values.

The rest of this paper is organized as follows: In Section 4.2, we briefly examine the previous work on TRNG in FPGA. In Section 4.3, we analyze the TRNG of [13]. In Section 4.4 we propose an enhancement of the TRNG to achieve better randomness on the output sequence. The analysis of the randomness of our proposed TRNG and the investigation of distribution of frequencies on oscillator rings are discussed in Section 4.5 and 4.6 respectively. In Section 4.7 we describe in detail an implementation of our proposed TRNG. In Section 4.8 we investigate the behavior of our TRNG after repeated restarts from known reset state, and finally we make a conclusion in Section 4.9.

4.2 Related Work

Several implementations of TRNG in FPGA have been proposed during the recent years. The common entropy source used is jitter on clock signals. Jitter can be viewed as timing deviation from the theoretically correct position due to electronic or thermal noise [14]. The random jitter will typically follow a Gaussian distribution characterized by a certain standard deviation (σ). Usually, jitter is an unwanted property in a system, but this behavior is useful when generating random signals in a TRNG.

In 2002, Fisher et al. [6] used the jitter in analogue phase-locked loop (PLL) in FPGAs from Altera as entropy source in a TRNG. The strategy was to create different clock signals with jitter from the PLL and sample one of the clock signals with the other. This method is restricted to FPGAs containing such analogue components. Later, Kohlbrenner et al. [9] used a similar technique, but the clocks are generated by oscillator rings containing two transparent latches, a buffer and an inverter. Since the frequencies of the two oscillator rings have to be almost equal, the oscillator rings have to be correctly matched. Tkacik [15] proposed a TRNG using a linear feedback shift register (LFSR) and a cellular automaton shift register (CASR) clocked by two independent oscillator rings. Selected outputs from the LFSR and CASR are combined by an XOR generating the final random signal. The disadvantage of this scheme is that the TRNG has memory and is therefore not stateless as pointed out in [2]. In 2006, Golić [7] proposed a TRNG using a Galois ring oscillator (GARO) and a Fibonacci ring oscillator (FIRO). These LFSR-like structures use inverters as delay elements instead of register elements. The outputs from one GARO and one FIRO are combined by means of an XOR and the random sequence is generated by sampling with a D flip-flop. This design was further investigated by Dichtl et al. [5]. The output signal from these FIRO/GARO structures has a noisy analogue behavior, making them more susceptible to cross-talk from other signals inside the FPGA than ordinary digital signals. In 2007, Sunar et al. [13] gave a theoretical proposal of a TRNG based on several equal length oscillator rings made up of an odd number of inverters (see Figure 4.1). The

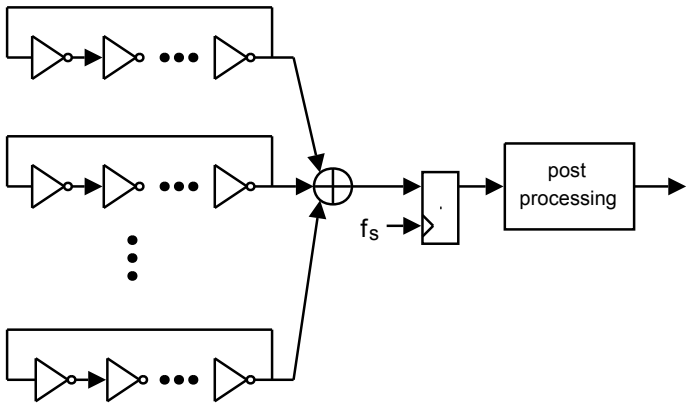

Figure 4.1: TRNG based on oscillator rings [13]

outputs from the oscillator rings are XORed together and sampled with a D flip-flop. To compensate for the imbalance between the number of zeros and ones in the random signal, a post-processing stage is present on the output of the D flip-flop. Schellenkens et al. [12] implemented this scheme in a Xilinx FPGA, but with a large number of rings in order to make the random sequence output pass the statistical tests. In 2008, Vasyltsov et al. [16] proposed a TRNG based on a 5-stage metastable ring oscillator, where each stage contains only one inverter. The result is a fast and small implementation of a TRNG in an FPGA or ASIC, but optimization in the synthesis process causes difficulties in the FPGA implementation. Recently, Danger et al. [3] proposed a fast TRNG based on creating metastability in open loop structures in FPGAs.

4.3 TRNG Based on Oscillator Rings

Since our proposed enhancement is based on the TRNG of Sunar et al. [13], we take a closer look on the design from [13], see Figure 4.1. The TRNG consists of several equal length oscillator rings connected to an XOR tree. The output from the XOR tree is sampled by a D flip-flop, and the output signal of the D flip-flop is then post-processed in order to increase the entropy and remove bias from the random signal. The proposed post-processing in [13] is a resilient function implemented as a BCH-code. The suggested design of the TRNG consists of 114 oscillator rings where each ring consists of 13 inverters. The suggested sampling frequency is 40MHz and the post-processing is a [256, 16, 113] extended BCH code. The resulting throughput of the TRNG in [13] is 2.5Mbps.

The entropy source of the TRNG is the jitter created by each oscillator ring. The jitter has a Gaussian distribution around each clock transition between logic low and logic high level. This jitter will create an accumulated phase drift in each ring so that the transitions will be at different times in the sampling period. Due to the jitter, the unpredictable transition region is assumed to be uniformly distributed in the sampling period. The number of rings needed can then be calculated based on the coupon collector's problem, that is, the number of uniform random selections of N urns such that all urns are selected at least once. The number of urns is determined by the proportion of the jitter size compared to the frequency of the oscillator ring. Because the number of rings grows exponentially when filling up the last urns, a lower fill rate than 100% is selected. To compensate for this, a BCH-code is used for post-processing. The resulting random number throughput is reduced by a factor of 16 due to this post-processing scheme.

In [5], some weaknesses of this implementation were mentioned. The main concern

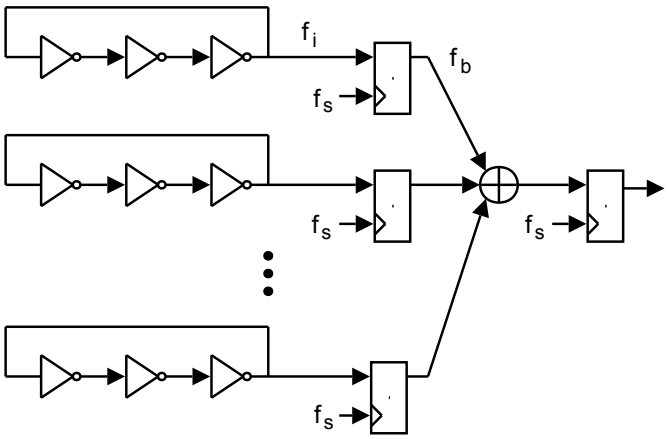

Figure 4.2: Our proposal

of the authors of [5] is that the XOR-tree and the sampling D flip-flop cannot handle the high number of transitions from the oscillator rings. The frequency of an oscillator ring is approximately the same or higher than the sampling frequency. With many oscillator rings in parallel, the number of transitions during a sampling period will be so high that the setup- and hold-times for the look-up table (LUT) and the internal register element in the FPGA will be shorter than specified for the device.

The analysis of the TRNG from [13] is shown in Sections 4.5 and 4.6 as it is better to present a comparison with the proposed enhanced TRNG.

4.4 Our Proposed Enhancement

To cope with the problem with many transitions in the sampling period, we suggest an enhancement of the TRNG based on the oscillator rings in [13] by adding an extra D flip-flop after each ring (Figure 4.2). As we will show, this configuration will improve the randomness of the TRNG. The randomness of the configuration relies on the jitter variations of the oscillator rings. Adding these extra flip-flops will not alter the collection of the randomness of each ring, but improve the overall randomness at the output.

The frequency of the oscillator ring (f_i) is dependent on the odd number of inverters in the ring. The frequency will increase with the decreasing number of inverters. In order to have a fast and small TRNG, the number of inverters should be as low as possible making the frequency of the rings become high as compared to the sampling frequency (f_s). The advantage of our enhancement is that the signals on the input of the XOR will now be synchronous with the sampling clock and only updated once in the sampling period. Due to this reduction in transitions on the input to the XOR tree, the setup- and hold-times for the internal logic in the FPGA will now be within acceptable limits.

The frequency of the beat signal (f_b) after the extra flip-flop will always be less than half of the sampling frequency and lie in the interval $[0, \frac{f_s}{2}]$ (Figure 4.3). The sampling frequency f_s should be chosen such that the beat frequency f_b at the input of the XOR is as high as possible, and avoid the frequency of the oscillator rings be a multiple of the sampling frequency resulting in a beat frequency near zero or no transitions (in the worst case) in the beat signal.

The result of adding the extra D flip-flop, is that the switching activity on the input to the XOR-tree is significantly reduced. The XOR calculation becomes deterministic while the randomness is collected by the sampling of the free running oscillator rings. The sampling of a free running oscillator ring could lead to metastability in the flip-flop causing

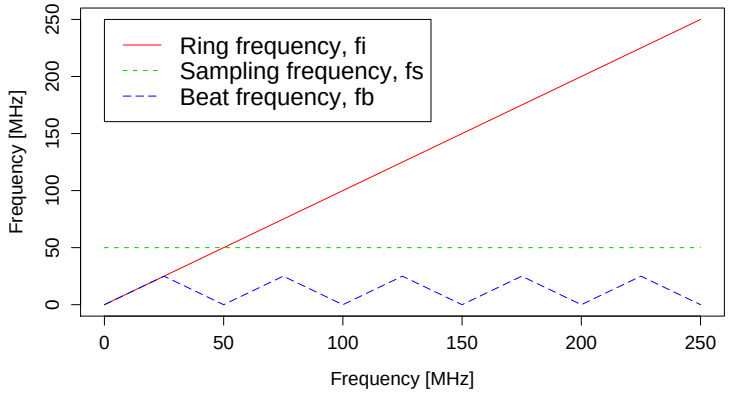

Figure 4.3: Beat frequency

the output of the flip-flop neither to be logic low or logic high for a short time period. This phenomenon can arise when a transition occurs during the set-up and hold-time of the flip-flop, which is the case for the extra D flip-flops in our proposed TRNG. To avoid the metastable state to propagate into the XOR tree, the output of the oscillator ring could be sampled by one additional D flip-flop.

If a large number of rings is needed, the logic of the XOR tree will be deep and contain many logic levels. The result could be violating the timing inside the FPGA because the time delay through the XOR tree is longer than the sampling period. In this case, one or more register levels can be inserted into the XOR tree. This will not affect the throughput, but it will increase the latency of the TRNG output and increase the resources used in the FPGA.

4.5 Bias in TRNG

One of the basic statistical tests of random number generators is the frequency test of ones and zeros. For a good random bit sequence the probability of a zero or a one should be equal to $\frac{1}{2}$. In other words, there should be no bias in the random bit sequence.

Let X and Y be two random bit sources with expected values $E(X) = E(Y) = \mu$ respectively and let ρ be their correlation. Then the expected value of the XOR of the two sequences $(X \oplus Y)$ is given by (see for example [4]):

$$E(X \oplus Y) = \frac{1}{2} - 2\left(\mu - \frac{1}{2}\right)^2 - 2\rho\mu\,(1-\mu) \tag{4.1}$$

If μ is close to $\frac{1}{2}$, Equation (4.1) can be written as:

$$E(X \oplus Y) \approx \frac{1}{2}\,(1-\rho) \tag{4.2}$$

It can be seen that correlation between the two sequences will generate bias in the output from the XOR of two random bit sequences. If X and Y are linearly independent, then $\rho = 0$ and $E(X \oplus Y) \approx \frac{1}{2}$.

If there are n independent bits, each with expected value μ, then the expected value of XOR of all these bits will be given by:

$$\frac{1}{2} + (-2)^{n-1}\left(\mu - \frac{1}{2}\right)^n = \frac{1}{2}\left(1 + (-2\varepsilon)^n\right) \tag{4.3}$$

where $\varepsilon = \mu - \frac{1}{2}$. Since $\mu \in (0,1) \Rightarrow |2\varepsilon| < 1$, the expected value in Equation (4.3) will converge to $\frac{1}{2}$ for increasing number of sequences, n. In other words, adding more oscillator rings in the TRNG design should improve the bias if the rings are independent.

We have carried out some experiments on the randomness of the TRNG in [13] (without any post-processing) and our proposal in Figure 4.2. The experiments are carried out on a Starter Development Board from Altera containing a Cyclone II FPGA. This device has a core voltage of 1.2V and is fabricated in 90nm technology. Quartus II WebEdition 6.1 is used for synthesis and Place and Route (P&R). The sequences of random bits generated inside the FPGA are stored in an external SRAM and transmitted to a PC for analysis through an asynchronous serial connection. The result is a number of blocks of subsequent random number bits from the TRNG where each block has a maximum size of 4Mbit. The sampling frequency used in this experiment is 50MHz. No constrains have been put on the P&R tool regarding the placement of the inverters in the FPGA.

We have implemented the two configurations of TRNG (Figure 4.1 and 4.2), recorded 10 blocks of 1Mbit of random data from each configuration and determined the frequency of ones in all blocks. We have performed the experiment with oscillator rings of lengths 3 and 13, and with varying number of rings. The results are shown in Figure 4.4. They indicate that the design in [13] has a bias after the XOR of the oscillator rings. The tendency is that the bias increases with the increasing number of rings and there is a majority of zeros in the output. Comparing these observations against the Equations (4.1)-(4.3) shows that there is some dependency or correlation in the random sequences creating a bias. It seems that this bias is due to the problem with the high number of transitions at the input of the XOR tree and the sampling flip-flop. For our configuration (Figure 4.2), it is seen that the expected value is close to $\frac{1}{2}$ for increasing number of oscillator rings. Figure 4.5 shows a closer view of the curves, where the absolute value of the bias from the ideal 0.5 level is shown for our configuration with 3 and 13 inverters in the oscillator rings. It is seen that the bias converges to 0, and that our enhanced TRNG behaves according to the theory of XOR of independent random sequences.

4.6 Distribution of Ring Frequencies

According to [13], the assumption of randomness is that the equal length oscillator rings will have the same frequency while the phase drift related to the jitter causes the drifting

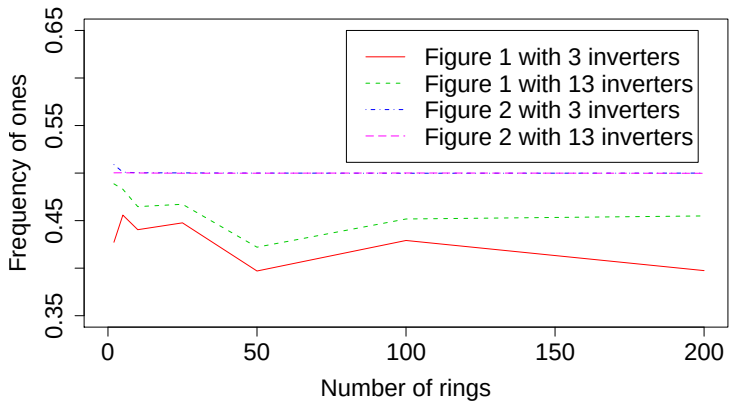

Figure 4.4: Bias of the TRNG on Figure 4.1 and 4.2

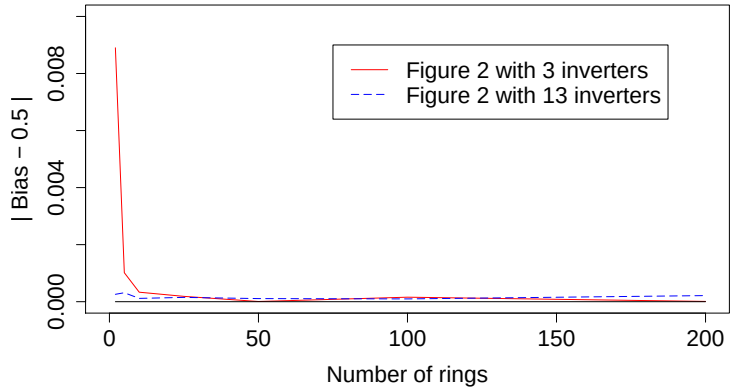

Figure 4.5: $|Bias - \frac{1}{2}|$ of the TRNG on Figure 4.2

of the transition regions. We believe that the frequencies of the oscillator rings will be different from each other. We have carried out an experiment where we have implemented 64 oscillator rings in the Altera Cyclone II FPGA and tapped out the signal from each of these rings to I/O-pins on the FPGA. These frequencies were measured with an oscilloscope.

Figure 4.6 shows the histograms of the frequencies of oscillator rings with 5 and 31 inverters respectively. (For oscillator rings with 3 inverters, the measured signals are outside the specification of the I/O-pins for our Cyclone II FPGA (maximum frequency of 300MHz). However, the frequencies are measurable and the measurements with 3 inverters gave a similar histogram as shown in Figure 4.6 with 5 inverters.) From this experiment, it is observed that the distribution of the frequencies for short rings does not follow

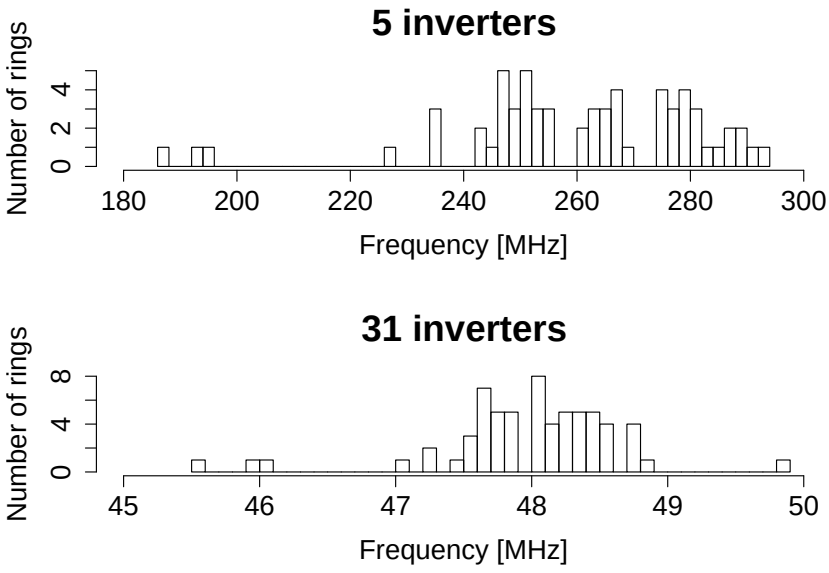

Figure 4.6: Histogram of ring frequencies

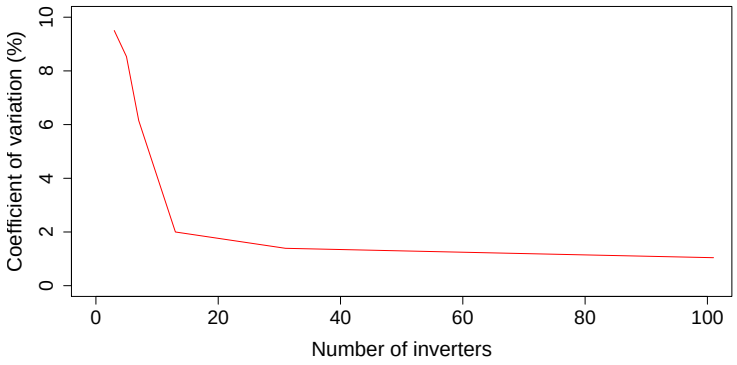

Figure 4.7: Dispersion of frequencies

a Gaussian distribution and the frequencies are clustered in groups. For longer rings, the clustering is not so obvious and the distribution is approaching Gaussian with only some values far from the mean.

When examining similar histograms for other lengths, it is observed that the dispersion is decreasing with increasing number of inverters. In Figure 4.7, the dispersion is measured by the coefficient of variation defined as the percentage of $\frac{\sigma}{\mu}$, where σ is the standard deviation and μ is the mean of the measured frequencies. It can be seen that the dispersion is high for short rings and decreasing with longer oscillator rings. Based on this observation, using oscillator rings with only 3 inverters will give the highest dispersion in the frequencies.

To explain the behavior of the frequency distribution, the architecture of the Altera Cyclone II FPGA [1] has to be examined. This FPGA consists of a matrix with logic elements (LE), each containing a programmable register and a LUT for implementing any logic function of four inputs. 16 of these LEs are then grouped into a logic array block (LAB). All the LEs and LABs are connected together via different routing resources depending on the distance between them inside the FPGA. When running P&R for the design in an FPGA, the inverters in the oscillator rings are located at physical LEs. Depending on the placement, the routing delay between the LEs will differ. If all the inverters are placed in LEs inside one LAB, the routing delay will be short. If the inverters are placed in LEs in different LABs, the routing delay will be increased resulting in a lower frequency of the oscillator ring. In addition, there will also be a variation in the delay of each LUT, typically following a Gaussian distribution. All these variations in the routing delays cause the distribution of the oscillator ring frequencies and the clustering for short rings. For short oscillator rings, the inverters of some of the rings are placed in the same LAB, but for some of the other rings, the inverters are placed in two or more LABs, resulting in routing delays with large variations. For long oscillator rings, the difference between the routing delays of each ring will be smaller due to the fact that the inverters have to be placed in more than one LAB. From Figure 4.7, this can be seen indirectly. For small number of inverters, the dispersion is high, and decreasing until the rings contains more than 16 inverters and therefore filling up more than one LAB. For more than 16 inverters, the dispersion is constant, indicating that the variation is only due to the natural timing variation between the difference logic elements in the FPGA. For other FPGAs with similar architecture, the oscillator ring frequencies will result in similar distributions.

Due to the observed distribution of frequencies of equal length oscillator rings, the transition regions will quickly be spread out over the sampling time period, much faster than

if only the accumulation of the oscillator ring jitter was contributing. In order to examine the effect of this frequency distribution on the randomness, we carried out an experiment where a model of the TRNG was made in MatLab without involving the jitter in the generation of the output sequence. 100 blocks of 1Mbit each were recorded, and for each block a new set of oscillator ring frequencies was generated from a Gaussian distribution (same as generating one block of data from 100 different TRNGs). The resulting output sequence was tested by the NIST randomness test suite [11], and it showed that with 50 or more oscillator rings in the MatLab TRNG model, the quality of the generated random sequences was good enough to pass the NIST test suite. This experiment shows that a TRNG combining several equal length oscillator rings outputs where there is a dispersion between the frequencies, generates bit sequences that have good qualities even though they are deterministic. The jitter introduced in the oscillator rings contributes with the unpredictable behavior that is necessary to have a true random source.

4.7 TRNG Implementation

We have implemented our proposed TRNG from Figure 4.2. In order to have a fast and small TRNG, the number of inverters in the oscillator ring is selected to be 3. A sampling frequency of 100MHz is selected, resulting in a throughput of 100Mbps since our TRNG does not use any post-processing. In most of real TRNG designs in cryptographic systems, using post-processing is recommended in order to improve the randomness by increasing the entropy and removing bias. But, for our TRNG, a post-processing is not needed to pass the statistical tests, and therefore an additional post-processor can be of a simple type like an XOR of two subsequent bits or a von Neumann corrector.

The required number of rings is estimated based on the probability to hit the transition region with the sampling. Sunar et al. [13] computed this by using a combinatorial approach (coupon collector's problem). An alternative way is to make a statistical model of the TRNG and perform simulations in order to decide how many oscillator rings are needed to achieve a high probability such that at least one ring is sampled in the transition region. When the size of the jitter is small compared to the sampling period, the simulations show that the number of rings in the transition region follows a Poisson distribution with a parameter $\lambda = k \cdot r$ where r is the number of rings and k is a constant depending

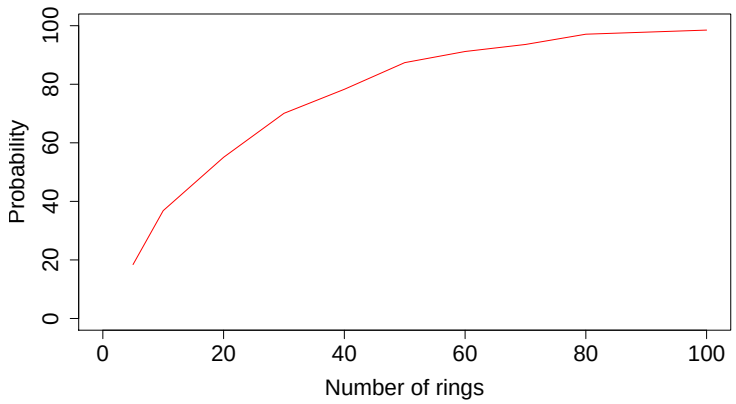

Figure 4.8: Simulated probability of hitting a transition region

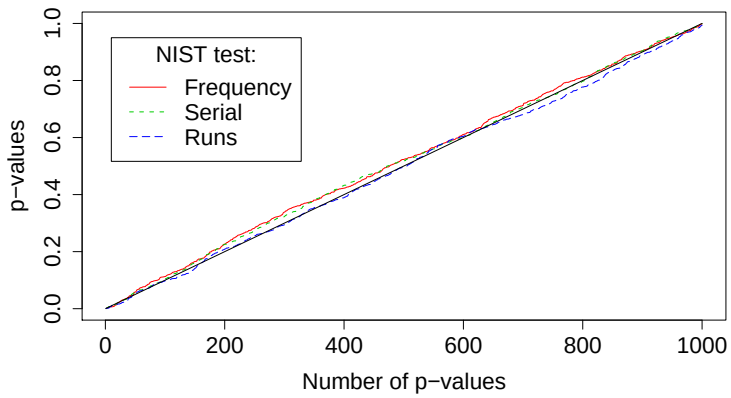

Figure 4.9: Selected results from the NIST suite

on the size of the jitter compared to the sampling period[2]. The probability of sampling in at least one of the transition regions versus the number of rings is shown in Figure 4.8. It shows that the probability increases rapidly for small number of rings, but many oscillator rings are needed to get a 100% certainty.

We have carried out an experiment where we have used 50 oscillator rings with 3 inverters, a sampling frequency of 100MHz and no post-processing. A total of 1000 blocks of 1Mbit (a total of 1Gbit) of random data have been captured from the TRNG. The data was tested by using the statistical tests of NIST (SP 800-22) [11] and DIEHARD [10]. The random data passed both tests. We also performed the same experiment with only 25 oscillator rings. The random data also passed both the NIST and the DIEHARD tests (Figure 4.9 and Figure 4.10). From these figures it can be observed that the sorted p-values from the tests follow the ideal diagonal line. These experiments indicate that it is probably not necessary

[2]A more precise approach is to use the standard deviation of accumulated jitter compared to the half of the sampling period.

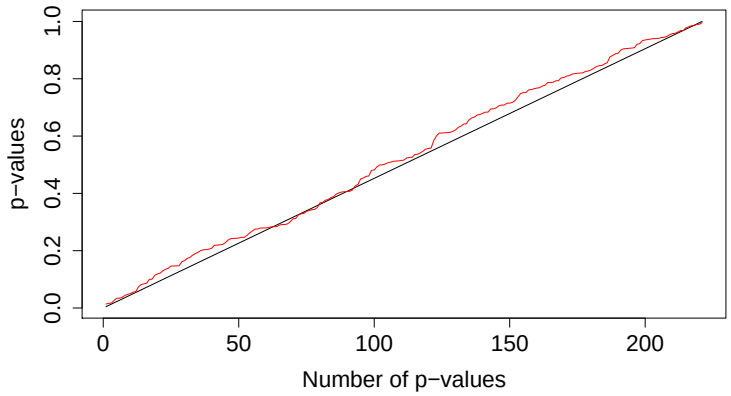

Figure 4.10: Results from DIEHARD

Oscillator rings	LUT only LEs	LUT/Register LEs	Total LEs
25	57	26	83
50	116	51	167

Table 4.1: Resources used in the Altera FPGA

to have almost 100% certainty to hit at least one transition region in order to pass the NIST and DIEHARD statistical tests for this kind of TRNG.

Table 4.1 shows the amount of resources used for our TRNG in the Altera Cyclone II FPGA. For 25 oscillator rings, the number of LEs is less than 100 ($< 1\%$ of the total number of LEs in our medium size FPGA). For comparison, the original design in [13] occupies more than 1800 LEs.

A TRNG design based on several oscillator rings is robust because the placement of the inverters inside the FPGA is not critical and no constraints on the P&R tool are necessary. As the experiments show, having different delays of the inverter chains contributes to the quality of randomness. However, if there are interactions between the oscillator rings making the rings oscillate with the same frequency and phase, the output bit sequence will naturally not be random. We have not seen any sign of interaction between the oscillator rings in our experiments.

All the tests were performed at the room temperature. The effect of varying the temperature is beyond the scope of this study, but in general, changing the temperature will influence the oscillator ring frequencies. An increase in temperature will decrease the oscillator ring frequencies and vice versa. But since all the rings will be influenced in the same manner, there will be a shift in all the frequencies and the dispersion between the frequencies will remain approximately unchanged.

4.8 Restart Experiment

In order to examine the randomness of our TRNG after start-up, an oscilloscope was used to capture the random output when restarting the TRNG several times from the same reset state[3]. While the reset is active, the oscillator ring outputs are kept at zero or low level. When the reset is deactivated, the oscillator rings start to oscillate [4]. In Figure 4.11, 10 restart sequences from the output of the TRNG are captured where the oscilloscope is triggered on a clocked version of the reset signal at the origin of the graph. The sampling frequency is 50MHz. Because of the bandwidth limitation in the oscilloscope, the measured outputs are not square signals. It can be seen that all the outputs start at zero, but there is a deviation after the first clock period of 20ns. This experiment shows that our TRNG outputs randomness quickly after a restart. The experiment also shows that since the traces are deviating from each other, the output contains true randomness and not pseudo randomness. If the random signal had been only pseudo random, the restart experiment should have given equal traces when repeatedly starting the TRNG from the same reset state.

The restart experiment was expanded to capture 1000 restarts. The standard deviation for all these traces was calculated and is shown in Figure 4.12 for a sampling frequency of 10MHz. The form of the curve is very regular because all the traces are aligned with the sampling frequency, and because the random signal is digital with voltage level of either +3.3V or 0V. Theoretically, the standard deviation can be calculated by looking at the probability of a voltage level of logic zero and logic one, and the probability of a transition

[3]The idea of distinguishing true and pseudo random sequences by using an experimental restart method was proposed by Dichtl et al. [5]

[4]The inverters in the oscillator rings are reset to the alternating initial state (not the all zero state) in the restart experiment.

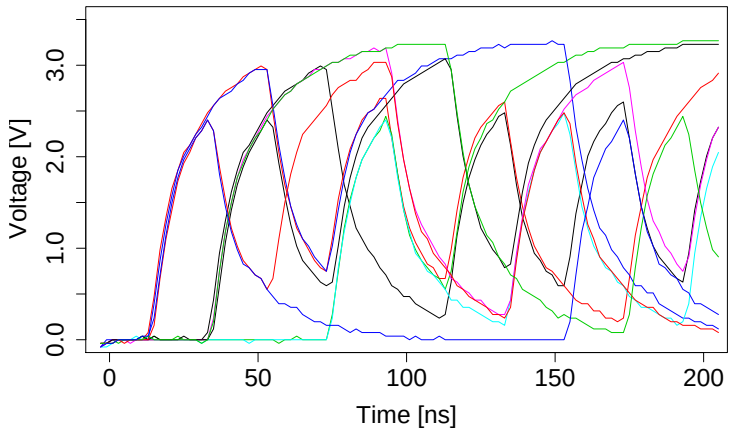

Figure 4.11: 10 restarts with 25 rings

between the two logic levels. A good random signal should have equal probability of zeros and ones, and also equal probability of a transition or no transition. The mean value of the voltage with these conditions, is then $\mu = 1.65V$. The standard deviation can then be calculated for the two cases: 1) the signal is in the middle of the sampling period, and 2) the signal is at the sampling point. For case 1), the standard deviation is $\sigma = 1.65V$, and for case 2) the standard deviation is $\sigma = 1.17V$. From Figure 4.12 we can see that the theoretical values match the measured data when the starting period is omitted. It is also observed that the standard deviation is stable after a short start-up period.

Figure 4.13 shows the standard deviation with a sampling frequency of 100MHz. Due to the band limitation of the oscilloscope, the values of the standard deviation differ from the theoretical values. It is observed that in a short start-up period, the quality of the ran-

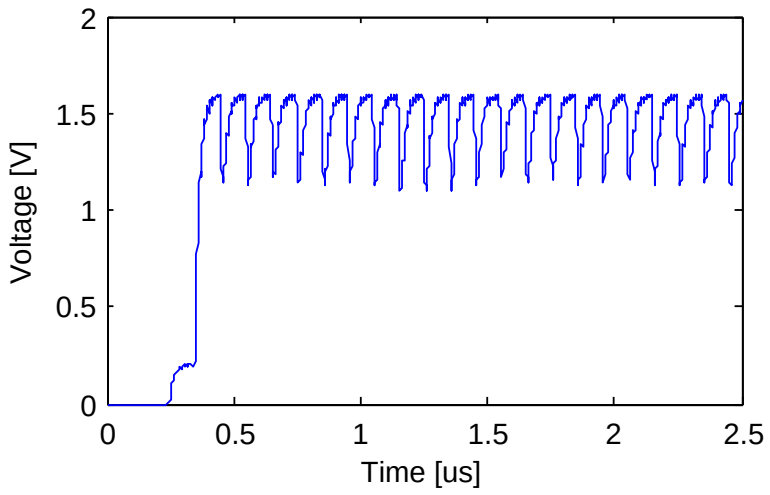

Figure 4.12: Standard deviation of 1000 traces, sampling frequency 10MHz and 25 rings

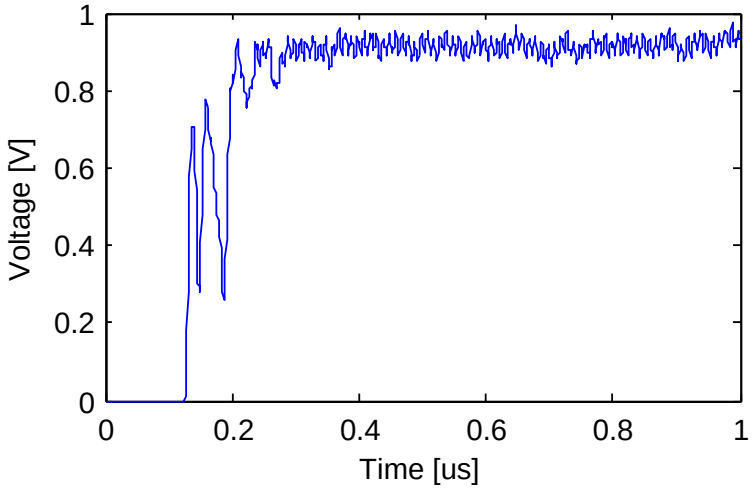

Figure 4.13: Standard deviation of 1000 traces, sampling frequency 100MHz and 25 rings

domness is not optimal because the standard deviation is not stabilized. From Figures 4.12 and 4.13, it is seen that this start-up period is constant regarding the sampling frequency or the throughput bit-rate. For the case of a TRNG with 25 oscillator rings with 3 inverters in each ring, this start-up time is about 300ns. This indicates that the first data bits should be omitted in order to have good quality of the random sequence[5].

4.9 Conclusion

We have analyzed the TRNG in [13] and have proposed an enhancement of a TRNG based on oscillator rings. By adding an extra flip-flop after each inverter ring before the XOR tree, we have shown that the performance is much better than [13] regarding the random signal. We have also shown that the frequencies of each ring are not equal but have some kind of distribution. Smaller rings will have higher dispersion in the distribution and therefore also better potential for fast generation of randomness after restart.

We have implemented the TRNG from Figure 4.2 and carried out statistical tests on the resulting random bit sequences. We have shown that our TRNG passes both the NIST and DIEHARD tests without post-processing. The throughput of the TRNG is 100Mbps and the resources used in the FPGA are less than 100 logic elements in an Altera Cyclone II FPGA.

The restart experiments show that the output of the TRNG behaves truly random and not pseudo random since the traces differ when restarted from the same reset state. These experiments also show that the standard deviation of the traces is in accordance with the theory, and that it is stable after a short start-up period. Due to this start-up period, the first bits should be omitted in order to have good quality of the randomness.

[5]If the TRNG is used in a restart mode, meaning that the TRNG is restarted after each generated bit in order to achieve independent bits, the observed start-up period will limit the achievable bit rate. However, when the TRNG is used in continuous mode, the conditions are different between the initial state where all the oscillator rings have identical phases, compared to after the start-up period where all the transitions are uniformly distributed mainly due to the variation in the frequencies of the oscillator rings.

Acknowledgment

This paper is an extended version of the conference paper [17] presented at ReConFig08. This work was carried out while Chik How Tan was at Gjøvik University College.

4.10 Bibliography

[1] ALTERA. Cyclone II Device Handbook, Volume 1. 2008. 8, 30, 36, 53

[2] BUCCI, M., AND LUZZI, R. Design of Testable Random Bit Generators. In *Proceedings of the 7th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'05)* (2005), vol. 3659 of *Lecture Notes in Computer Science*, Springer, pp. 147–156. doi:10.1007/11545262_11. 5, 7, 30, 98, 102

[3] DANGER, J.-L., GUILLEY, S., AND HOOGVORST, P. High Speed True Random Number Generator Based on Open Loop Structures in FPGAs. *Microelectronics Journal 40*, 11 (November 2009), 1650–1656. doi:10.1016/j.mejo.2009.02.004. 13, 14, 31

[4] DAVIES, R. B. Exclusive OR (XOR) and Hardware Random Number Generators. February 2002. 33, 105

[5] DICHTL, M., AND GOLIĆ, J. D. High-Speed True Random Number Generation with Logic Gates Only. In *Proceedings of the 9th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'07)* (2007), vol. 4727 of *Lecture Notes in Computer Science*, Springer, pp. 45–62. doi:10.1007/978-3-540-74735-2_4. 11, 12, 13, 16, 30, 31, 32, 39, 64, 102, 103

[6] FISHER, V., AND DRUTAROVSKÝ, M. True Random Number Generator Embedded in Reconfigurable Hardware. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 415–430. doi:10.1007/3-540-36400-5_30. 9, 10, 14, 30, 45, 102

[7] GOLIĆ, J. D. New Methods for Digital Generation and Postprocessing of Random Data. *IEEE Transactions on Computers 55*, 10 (2006), 1217–1229. doi:10.1109/TC.2006.164. 11, 12, 14, 15, 20, 30, 46, 58, 63, 102

[8] JUN, B., AND KOCHER, P. The Intel Random Number Generator. *White paper prepared for Intel Corporation* (Apr. 1999). 5, 29, 101

[9] KOHLBRENNER, P., AND GAJ, K. An Embedded True Random Number Generator for FPGAs. In *Proceedings of the ACM/SIGDA 12th International Symposium on Field Programmable Gate Arrays (FPGA'04)* (2004), ACM, pp. 71–78. doi:10.1145/968280.968292. 10, 11, 14, 30, 45, 87, 102

[10] MARSAGLIA, G. DIEHARD: A Battery of Tests of Randomness. 1996. 7, 15, 38, 46, 47, 77, 103, 108

[11] NIST. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Special Publication 800-22 Revision 1a, April 2010. 7, 15, 37, 38, 46, 47, 65, 77, 103, 108

[12] SCHELLEKENS, D., PRENEEL, B., AND VERBAUWHEDE, I. FPGA Vendor Agnostic True Random Number Generator. In *Proceedings of the 16th International Conference on Field-Programmable Logic and Applications (FPL'06)* (2006), IEEE, pp. 1–6. doi:10.1109/FPL.2006.311206. 12, 14, 31, 102

[13] SUNAR, B., MARTIN, W. J., AND STINSON, D. R. A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. *IEEE Transactions on Computers 56*, 1 (2007), 109–119. doi:10.1109/TC.2007.250627. 4, 11, 12, 14, 15, 19, 20, 21, 30, 31, 32, 34, 37, 39, 41, 45, 53, 58, 63, 77, 78, 83, 87, 101, 102, 103, 105, 106, 108, 109, 125, 126

[14] TEKTRONIX INC. Understanding and Characterizing Timing jitter. 2003. 9, 30, 102

[15] TKACIK, T. E. A Hardware Random Number Generator. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 450–453. doi:10.1007/3-540-36400-5_32. 10, 14, 30, 45, 102

[16] VASYLTSOV, I., HAMBARDZUMYAN, E., KIM, Y.-S., AND KARPINSKYY, B. Fast Digital TRNG Based on Metastable Ring Oscillator. In *Proceedings of the 10th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'08)* (2008), vol. 5154 of *Lecture Notes in Computer Science*, Springer, pp. 164–180. doi:10.1007/978-3-540-85053-3_11. 13, 14, 20, 31, 46, 58, 63

[17] WOLD, K., AND TAN, C. H. Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings. In *Proceedings of the 4th IEEE International Conference on ReConFigurable Computing and FPGAs (ReConFig'08)* (2008), pp. 385–390. doi:10.1109/ReConFig.2008.17. 12, 14, 20, 42, 46, 47, 48, 52, 53, 54, 58, 64, 77, 78, 79, 80, 83, 87, 92, 99

Chapter 5

Optimizing Speed of a True Random Number Generator in FPGA by Spectral Analysis[1]

Abstract

Security and speed are two important properties of today's communication systems. In order to generate initialization vectors and keys for such communication fast enough, a true random number generator (TRNG) with a high bit rate is needed. In this paper an FPGA implementation of a TRNG based on several equal length oscillator rings that achieves a high bit rate is analyzed by using spectral analysis. The design is examined by defining a MatLab model of the TRNG and by investigating the frequency spectrum at different locations in order to find the speed increasing potential of the TRNG. Experiments performed on an Altera Cyclone II FPGA have shown that a TRNG, whose parameters were optimized by means of such a model, achieves a bit rate of 300 Mbit/s. Experiments with repeated restarts from a known state have shown that the output of the TRNG contains true randomness and not only pseudo randomness.

5.1 Introduction

In modern communication systems, there is a constant need of increasing the throughput and the amount of transmitted data. In order to ensure secure communication in such systems, the data must be encrypted. The encryption algorithms must also satisfy the efficiency requirements. For example, in an Internet protocol (IP) packet system, in which a block cipher is used for encryption, a random initialization vector (IV) is needed for each encrypted packet. The length of such an IV is related to the block size of the encryption algorithm. For a block cipher such as AES, a typical choice of the IV length is 128 bits. Since the speed of today's Ethernet communication systems could be of the order of magnitude of several tens of Gbit/s and the maximum size of a packet is 1500 bytes, a true random number generator (TRNG) must have a minimum bit rate of more than 100 Mbit/s if all the packets are of maximum length in order to achieve 10 Gbit/s throughput. If the packets are smaller, the bit rate of the TRNG must be even higher.

In addition, to ensure security, the random source generating the IVs should possess good randomness properties. A possible solution to this problem is to realize the entire communication system inside a field programmable gate array (FPGA), including the TRNG. It is known that jitter on clock signals in FPGA devices can be exploited to produce randomness in the TRNG. The FPGA implementation also contributes to a high throughput of the communication.

Several implementations of TRNG in FPGAs have been proposed so far. The entropy source has been either jitter of clock signals or metastability on signals. Some of the early attempts based on clock jitter such as Fisher et al. [2], Kohlbrenner et al. [5] and Tkacik [10], achieved low bit rates, typically less than 1 Mbit/s. Later, Sunar et al. [9] proposed a TRNG based on several equal length oscillator rings with a bit rate of 2.5 Mbit/s. Golić

[1] Knut Wold and Slobodan Petrović. In Proceedings of the 4th IEEE International Conference of Computer Sciences and Convergence Information Technology (ICCIT'09), pp. 1105–1110, 2009.

[3] proposed a TRNG with FIRO/GARO structures achieving a bit rate of 12.5 Mbit/s. In 2008, Wold et al. [12] proposed an enhancement of the design of Sunar et al. with a bit rate of 100 Mbit/s. Recently, Vasyltsov et al. [11] proposed an implementation based on a 5-stage metastable ring oscillator with a bit rate of 140 Mbit/s.

In this paper, the behavior of a TRNG containing several equal length oscillator rings based on [12] is more thoroughly examined, and the results from [12] are extended by determining the design parameters of the TRNG that give a high bit rate. The analysis is carried out by defining a model of the TRNG in MatLab and by investigating the frequency spectrum of the different components of the TRNG. In such a way the mechanisms of operation of this type of TRNG are explained more clearly, and the design parameters for the TRNG that give a high bit rate are determined. We experimentally show that this implementation can achieve a bit rate of 300 Mbit/s, making such a TRNG a good choice in a high speed cryptographic communication system. To the best of our knowledge, this is the highest bit rate of a TRNG reported. By running statistical tests (NIST [7] and DIEHARD [6]) we show that the randomness properties of this TRNG are satisfactory. In addition, a restart experiment was carried out, showing that the output sequence contains true randomness and not only pseudo randomness.

The rest of the paper is organized as follows: In Section 5.2, some relevant properties of an ideal digital random signal are enumerated, and in Section 5.3 the TRNG containing several equal length oscillator rings is analyzed. In Section 5.4, the relationship between the sampling frequency and the oscillator ring frequency is discussed, and in Section 5.5 the experiments performed on an Altera Cyclone II FPGA Starter Development Board are described. Finally, in Section 5.6 a conclusion is given.

5.2 Properties of a Digital Random Signal

We consider a random binary sequence $x(t)$ with the amplitude levels of 0 V for logic zero and $+A$ V for logic one, and with a bit period of T seconds. This random signal is assumed to be stationary in the wide sense, meaning that the mean of the random signal is constant, i.e. $E[x(t)] = C$, and that the autocorrelation function does not vary with a shift in time origin, i.e. $R_x(t_1, t_2) = R_x(t_1 - t_2)$ [8]. The autocorrelation function can then be written as $R_x(\tau)$ where $\tau = t_1 - t_2$.

The autocorrelation $R_x(\tau)$ of the random number signal $x(t)$ is given by [4]:

$$R_x(\tau) = \begin{cases} \frac{A^2}{4} + \frac{A^2}{4}(1 - \frac{|\tau|}{T}), & |\tau| < T \\ \frac{A^2}{4}, & |\tau| \geq T \end{cases} \tag{5.1}$$

The autocorrelation of $x(t)$ is constant when the parameter τ is greater than the bit period T.

The power spectral density $S_x(f)$ of the random number signal $x(t)$, is given by the Fourier transform of the autocorrelation function (5.1):

$$S_x(f) = \frac{A^2}{4}\delta(f) + \frac{A^2 T}{4}\mathrm{sinc}^2(fT) \tag{5.2}$$

where δ is the Dirac function and $\mathrm{sinc}(fT) = \frac{\sin(\pi fT)}{\pi fT}$. The power spectral density of $x(t)$ has an envelope with shape $\mathrm{sinc}^2(fT)$. The power is proportional to the square of the voltage, so the frequency spectrum of the voltage has a $|\mathrm{sinc}(fT)|$ characteristic, see Figure 5.1.

In order to obtain a good random signal, the power spectral density should be constant in the bandwidth of interest. White noise has constant power spectral density for all frequencies. Ideal white noise is not physically possible because the average power is infinite. In practice, the power spectral density of the noise signal is band limited. From the $|\mathrm{sinc}(fT)|$ shape of the frequency spectrum of an ideal random signal, it can be seen that

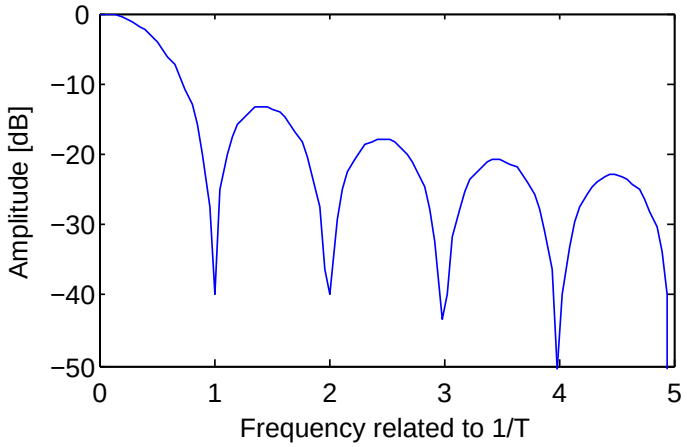

Figure 5.1: Shape of the power spectrum of a digital random signal

the spectrum is approximately flat (white noise) in the interval $[0, \frac{1}{2T}]$. Also, the spectrum is zero at multiples of the bit frequency $\frac{1}{T}$, and decreases toward zero when the frequency is increasing.

5.3 Spectral Analysis of the TRNG

Figure 5.2 shows the analyzed TRNG based on several equal length oscillator rings [12]. Each oscillator ring is made up of 3 inverters, and the output from each oscillator ring is sampled by a D flip-flop with sampling frequency f_s. All the outputs from these flip-flops are XORed, and the output from the XOR-tree is sampled with another D flip-flop generating the random signal. Experiments performed in [12] have shown that the random output sequence of such a TRNG passes the statistical tests such as NIST [7] and DIEHARD [6] without an additional post-processor. But in a real TRNG application a post-processor is recommended in order to remove bias and to increase the entropy of the random signal.

In this paper, a model in MatLab of the TRNG from Figure 5.2 is presented, and by means of that model the frequency spectrum at the following locations in the TRNG is inspected:

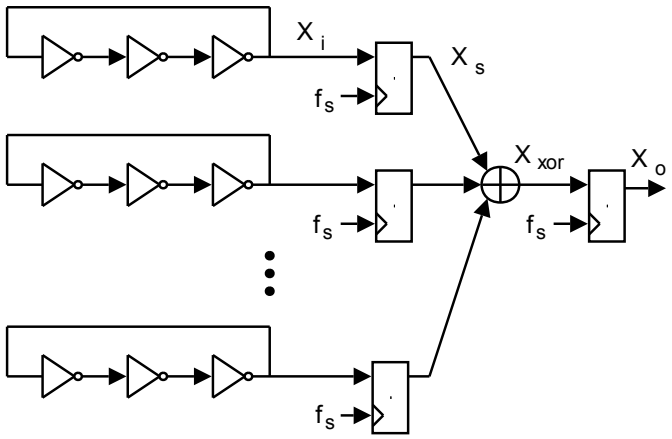

Figure 5.2: TRNG based on oscillator rings

- X_i: The output from the oscillator ring
- X_s: The output from the first sampling D flip-flop
- X_{xor}: The output from the XOR-tree
- X_o: The random signal

5.3.1 The Output X_i from the Oscillator Ring

The signal X_i measured immediately after the oscillator ring is a square signal with frequency related to the delay of the inverter chain, $f_i = \frac{1}{2l \cdot t_{\text{inv}}}$, where l is the (odd) number of inverters in the oscillator ring and t_{inv} is the delay of one inverter implemented in a logic element (LE) in the FPGA. The time delay t_{inv} will vary due to different routing inside the FPGA and natural variations in the manufacturing of the FPGA device.

An ideal square wave is a periodic signal with 50% duty cycle, zero rise- and fall-time and constant amplitude level of logic zero and logic one. The frequency spectrum of this ideal square signal with frequency f_i can be calculated by using Fourier series:

$$\frac{A}{2} + \frac{2A}{\pi} \sum_m \frac{1}{m} \cos(m f_i) \tag{5.3}$$

where A is the amplitude and m is an odd positive number ($m = 1, 3, 5, \ldots$).

Due to non-ideal effects in the FPGA, the duty-cycle of the square signal is not 50% and the fall- and rise-time are not zero. The effect on the frequency spectrum of these imperfections is that the even harmonics are not zero, and the level of the odd harmonics decreases.

Jitter is a physical property related to the thermal and electronic noise in real electronic circuits. Jitter also exists in the FPGA hardware. The effect of the jitter is a deviation from the theoretical correct time of a transition. The random jitter typically follows a Gaussian distribution characterized with a standard deviation σ. The effect on the frequency spectrum of a signal with jitter is similar to frequency modulation (FM) [8]. In FM, the modulation index indicates the amount of deviation from the carrier. In the TRNG, the jitter is proportional to the modulation index, and the size of the jitter will therefore determine the number and the bandwidth of the extra peaks in the frequency spectrum of the output of the oscillator ring. Increasing the jitter creates more peaks and they span a larger frequency range around the ideal square wave harmonics. Figure 5.3 shows a MatLab simulation of the frequency spectrum of an X_i signal with jitter (σ = 30 ps) and a duty-cycle of 49%. Several peaks are generated around the odd harmonics and also some new peaks at even harmonics.

If the oscillator ring frequencies are oscillating with the same frequency and phase, the output of the TRNG will not be random and the TRNG will therefore not work properly. Theoretically, this could happen if there would be a strong coupling or interaction between the oscillator rings in the FPGA. However, experiments by measuring ring frequencies performed in [12], have shown that the frequencies of the oscillator rings are different due to the placement of the inverters and the resulting routing between the inverters in the FPGA.

5.3.2 The Output X_s from the First D Flip-flop

The output from an oscillator ring is sampled by a D flip-flop with sampling frequency f_s. The resulting signal $x_s(t)$ can be theoretically described by the sample and hold method. The samples will be stable during the sampling period to either logic zero or logic one. The sampled signal $x_s(t)$ can then be described by a convolution of the product of a pulse train of Dirac functions and the input signal $x_i(t)$, with the rectangular pulse $p(t)$, whose width

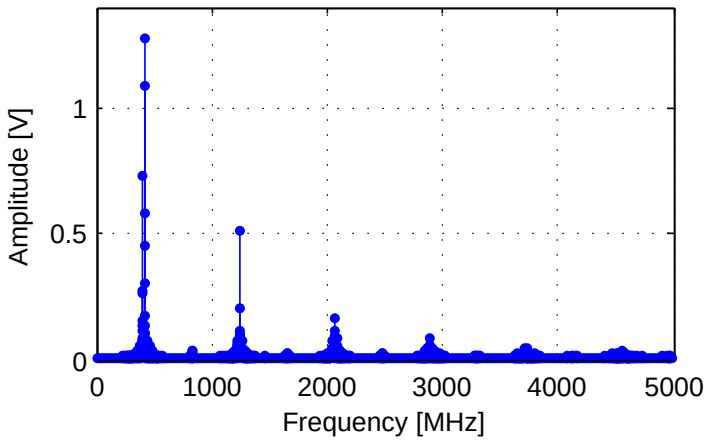

Figure 5.3: Frequency spectrum of X_i

is $T_s = \frac{1}{f_s}$ [8]:

$$x_s(t) = p(t) * \left[x_i(t) \sum_{n=-\infty}^{\infty} \delta(t - nT_s) \right] \tag{5.4}$$

The Fourier transform of $x_s(t)$ is given by:

$$X_s(f) = P(f) \frac{1}{T_s} \sum_{n=-\infty}^{\infty} X_i(f - nf_s) \tag{5.5}$$

where $P(f) = T_s \text{sinc}(fT_s)$.

The input signal $x_i(t)$ is the output from the oscillator ring. Assuming that this is an ideal square wave (5.3), the resulting spectrum of the sampled signal from one of the oscillator rings can then be written in the form (the DC-component is omitted):

$$X_s(f) = C\text{sinc}(fT_s) \sum_{m} \frac{1}{m} \sum_{n=-\infty}^{\infty} \cos\left(m(f_i - nf_s)\right) \tag{5.6}$$

where n is an integer, $m = 1, 3, 5, \ldots$ and C is a constant.

The frequency spectrum of the sample and hold signal has a sinc-envelope with zero-crossings at multiples of the sampling frequency f_s. In addition, several peaks are generated in the frequency spectrum determined by the relationship between the sampling frequency f_s and the oscillator ring frequency f_i.

If $x_i(t)$ is the oscillator ring signal and not the ideal square wave, the resulting frequency spectrum after sampling a random signal with sampling frequency of 300 MHz and oscillator ring frequency of 420 MHz is shown in Figure 5.4. In Figure 5.5, a closer view of the baseband is given, and it can be seen that the peaks of the harmonics of the input signal are moved to the frequency range $[0, \frac{f_s}{2}]$ due to the sampling. The relation between the sampling frequency and the oscillator ring frequency determines the dispersion of these frequency components. The main peak is located at $f_i - f_s$=120 MHz, and this peak is also found in Figure 5.4 where it is duplicated at higher frequencies (multiples of f_s). The peak at 180 MHz is the image component of the peak at 120 MHz around $\frac{f_s}{2}$ in the Fast Fourier Transform (FFT) calculations.

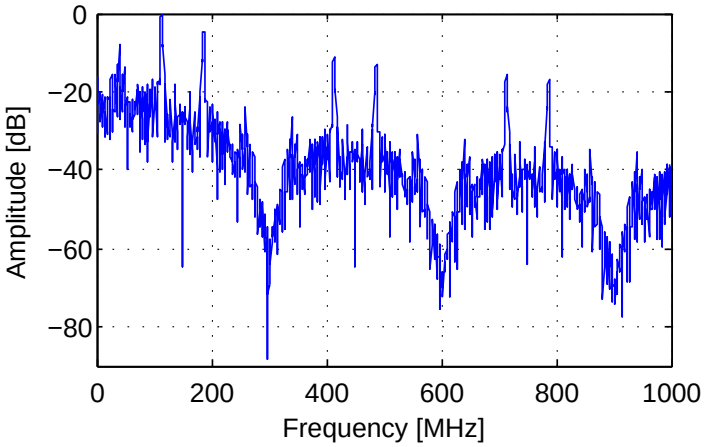

Figure 5.4: Frequency spectrum of X_s

5.3.3 The Output X_{xor} from the XOR Tree

XOR of two signals is the same as an ideal mixing. The result of the XOR of two sinusoidal signals is given by:

$$\cos(\omega_1 t)\cos(\omega_2 t) = \frac{1}{2}\Big[\cos[(\omega_1+\omega_2)t] + \cos[(\omega_1-\omega_2)t]\Big] \tag{5.7}$$

where $\omega_1 = 2\pi f_1$ and $\omega_2 = 2\pi f_2$ are the frequencies of the two sinusoidal signals. XOR of r sinusoidal signals creates 2^{r-1} new frequency components, and the amplitude of each new component is reduced by a factor $\frac{1}{2^{r-1}}$. XOR of the r different X_i signals where each signal is compounded by many sinusoidal frequency components, generates several new frequency components and the amplitude difference between the peaks is strongly reduced. The result after the XOR is a frequency spectrum more like white noise in the bandpass frequency range.

Figure 5.6 shows the frequency spectrum after XOR of 25 oscillator rings with a sampling frequency of 300 MHz. Figure 5.7 shows the baseband of the same signal. The shape

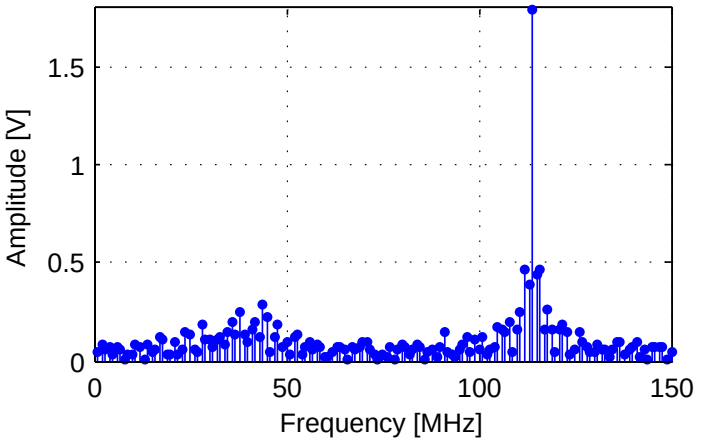

Figure 5.5: Baseband spectrum of X_s

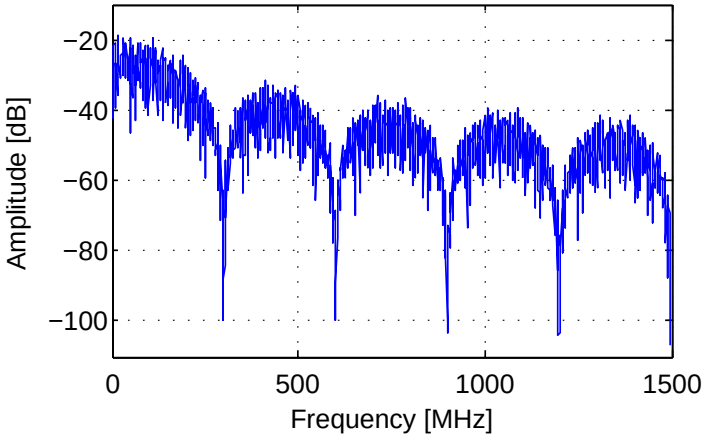

Figure 5.6: Frequency spectrum of X_{xor}

of the frequency spectrum is approaching the sinc-shape of the ideal random signal (Figure 5.6), and the amplitude differences between the frequency components are reduced in comparison to the amplitude of the input signal of the XORs, see Figure 5.5 and Figure 5.7.

5.3.4 The Output X_o from the Last D Flip-flop

The output from the last D flip-flop is the final random signal. In the time domain, the small variations that occur when the combinational XOR function is calculated due to time race inside the FPGA, is removed by the sampling. The effect is a low pass filtering of the signal. In the frequency domain, the result is a convolution moving frequency components from above f_s down to the frequency range $[0, \frac{f_s}{2}]$, similar to the result after the first D flip-flop. The consequence is that the noise level of the spectrum is increased and the frequency spectrum approaches white noise in the frequency range $[0, \frac{f_s}{2}]$. The simulated frequency spectrum at the output of the TRNG is shown in Figure 5.8[2].

[2]Bit sequences generated by pseudo random generators will produce a similar frequency spectrum, meaning that investigation of the spectrum cannot be used to distinguish between true and pseudo randomness.

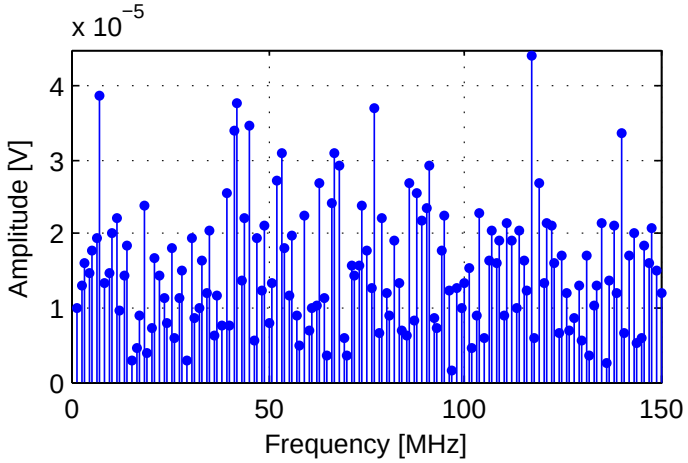

Figure 5.7: Baseband spectrum of X_{xor}

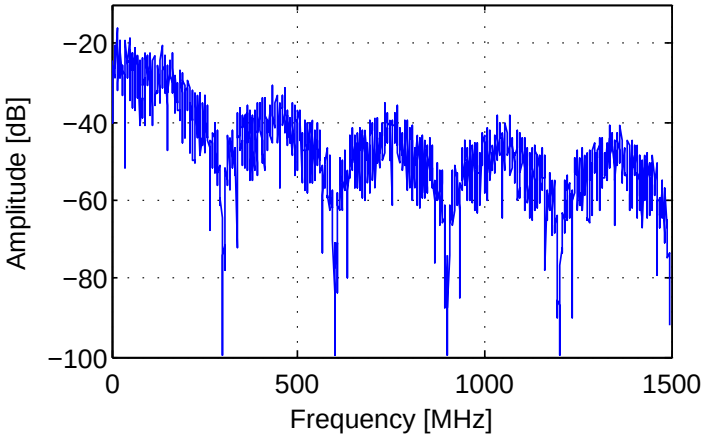

Figure 5.8: Frequency spectrum of X_o

5.4 TRNG Design Parameters

The true randomness of this TRNG is mainly generated by the jitter of the signals from the oscillator rings. The size of the jitter is related to the hardware of the FPGA, and cannot be altered. The adjustable design parameters of the TRNG are:

- The number of inverters in the oscillator ring l (oscillator ring frequency f_i)
- The sampling frequency f_s
- The number of oscillator rings r

According to [12], 3 inverters in the oscillator ring is the optimal choice for the Altera Cyclone II FPGA. This will not only give the fastest ring frequency f_i, but also the highest dispersion between the oscillator ring frequencies. Due to the Place and Route (P&R) tool, the placement of the inverters in LEs in the FPGA results in a variation between the oscillator ring frequencies. The result is r different oscillator frequencies $f_{i,j}$ where $j = 1, 2, \ldots r$.

The best choice of the sampling frequency f_s, is based on the relationship to the oscillator ring frequency f_i. In order to find a good distribution of the frequency peaks after the sampling, a simulation in MatLab was performed based on the relation between f_i and f_s in (5.6) with $m = 1$ (sinusoidal signal). Figure 5.9 shows the amount of frequency components in the interval $[0, f_i]$ generated by varying the sampling frequency in relationship with a constant oscillator ring frequency. The amount of new components increases as the sampling frequency approaches the oscillator ring frequency f_i. When f_s is higher than f_i, the graph has local peaks around multiples of $\frac{f_s}{f_i}$.

If the relation between the sampling frequency and the oscillator ring frequency is $f_i = nf_s$ or $f_i = \frac{f_s}{n}$, the result is not ideal with few or only one generated frequency component. The consequence is a signal x_s after the sampling flip-flop with a frequency near zero and with a low number of transitions, and therefore with little potential of creating randomness. In general, if the relation can be written as a rational number, i.e. $f_i = \frac{p}{q} f_s$, the result after the sampling is the generation of q frequency components in the spectrum.

If $f_s \gg f_i$, the sampling of the oscillator ring output has little effect because the sampling will be performed so often that the sampled signal is approximately identical to the original signal from the oscillator ring.

Based on these observations, a good choice of the relation between the sampling frequency and the oscillator ring frequency giving a high potential for creating randomness in this TRNG based on equal length oscillator rings is:

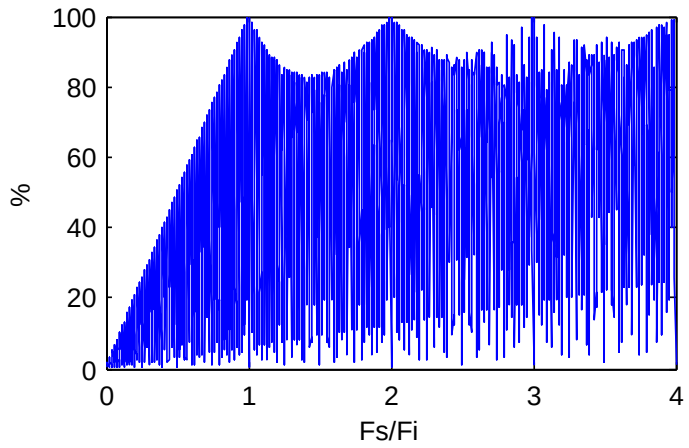

Figure 5.9: Amount of generated frequency components related to the sampling frequency

- $f_s \in \langle \frac{3f_i}{4}, \frac{5f_i}{4} \rangle$
- $f_i \neq \frac{p}{q} f_s$, where q is a small positive integer

In addition, the choice of the sampling frequency is also limited by the internal delays in the FPGA and timing constraints in the TRNG design.

The required number of oscillator rings can be estimated by looking at the probability of hitting the region where a transition in the x_i signal occurs [9, 12]. To compensate for a nonoptimal selection of the relation between the sampling frequency and the oscillator ring frequency, the number of oscillator rings can be increased. This would improve the quality of the randomness, but at the expense of increased usage of resources in the FPGA.

5.5 Experiment

Experiments were carried out on the TRNG from Figure 5.2 implemented in a Cyclone II FPGA Starter Development Board from Altera [1]. Quartus II WebEdition 8.1 was used for synthesis and P&R. Consecutively generated random bits were stored in blocks in an external SRAM on the evaluation board and transmitted to a PC for analysis through a Serial Peripheral Interface (SPI) bus. The result was blocks of maximum 4 Mbit of consecutive random bits. These blocks were stored as a binary file on the PC and used as input to the statistical tests. No constraints were put on the placement of the inverters in the P&R tool.

The number of inverters in each ring was selected to be 3, which results in an average oscillator ring frequency of about 400 MHz. According to the guidelines from Section 5.4, the sampling frequency should be in the interval 300-500 MHz. However, due to timing constraints violations in the P&R process of implementing the TRNG design in the FPGA, the achievable sampling frequency was limited to 300 MHz. The number of oscillator rings was set to 25. A total of 1000 blocks with 1 Mbit each were recorded and analyzed. The random data from the TRNG passed both the NIST and DIEHARD statistical tests. Even though the TRNG passes the statistical tests, it does not mean that the generated output sequence contains true randomness. The NIST and DIEHARD tests cannot distinguish between true randomness and good pseudo randomness. In order to verify the presence of true randomness, a restart experiment was carried out[3]. The TRNG was held in reset

[3]The idea of distinguishing true and pseudo random sequences by using an experimental restart method was proposed by Dichtl et al. [15, Ch.3]

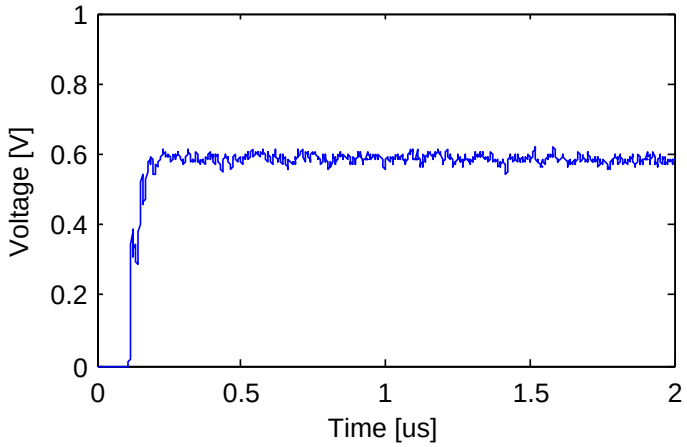

Figure 5.10: Standard deviation of the TRNG output in the restart experiment

by clearing all the flip-flops in the TRNG, and one of the inverters in each oscillator ring was exchanged by a NAND-gate in order to hold the oscillations during a reset period of 40μs. 1000 traces were captured of the output from the TRNG, starting when the reset was released. The standard deviation of the 1000 recorded traces was calculated, see Figure 5.10. The standard deviation increases after the reset is released and then stabilizes at a constant level. If there is only pseudo randomness on the output, the standard deviation is supposed to be approximately zero. Since the standard deviation converges to a constant level higher than zero, it is obvious that the output of the TRNG contains true randomness. One should also observe that the random output should be omitted in a short period after the reset is released and before the standard deviation is stabilized.

In all these experiments, no post-processor was used. In a real TRNG application, it is recommended to add a post-processor in order to improve the random signal by removing eventual bias and increasing the entropy.

5.6 Conclusion

In this paper, a TRNG based on several equal length oscillator rings from [12] was examined more closely in order to determine the parameters that give a high bit rate. The examination was performed by defining a model of the TRNG in MatLab. The behavior of the TRNG was analyzed by investigating the frequency spectrum at different locations in the model. The simulations showed that the frequency spectrum from the model approached the shape of an ideal random signal, and that this TRNG has a potential of generating random sequences at a high bit rate maintaining at the same time the quality of the randomness.

In order to achieve a high bit rate from this TRNG, the following guidelines regarding the relationship between the oscillator ring frequency f_i and the sampling frequency f_s should be followed:

- $f_s \in \left\langle \frac{3f_i}{4}, \frac{5f_i}{4} \right\rangle$, ensures that the frequency spectrum after the sampling of the oscillator ring contains a high amount of frequency components related to the ratio between the sampling frequency and the oscillator ring frequency.

- $f_i \neq \frac{p}{q} f_s$, where q is a small positive integer, avoids that the output signal of the sampling of the oscillator rings contains few or no transitions.

The TRNG was implemented on a Cyclone II FPGA Starter Development Board from Altera. The experiments showed that the TRNG achieved a bit rate of 300 Mbit/s with an implementation containing 25 oscillator rings and 3 inverters in each ring. The produced random sequences passed the NIST and DIEHARD statistical tests. In addition, a restart experiment was carried out, showing that the output of the TRNG indeed produces true randomness and not only pseudo randomness. These experiments show that the TRNG based on several equal length oscillator rings has potential of generating random data of good quality at a high bit rate, making this TRNG a good choice in high speed cryptographic communication systems.

5.7 Bibliography

[1] ALTERA. Cyclone II Device Handbook, Volume 1. 2008. 8, 30, 36, 53

[2] FISHER, V., AND DRUTAROVSKÝ, M. True Random Number Generator Embedded in Reconfigurable Hardware. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 415–430. doi:10.1007/3-540-36400-5_30. 9, 10, 14, 30, 45, 102

[3] GOLIĆ, J. D. New Methods for Digital Generation and Postprocessing of Random Data. *IEEE Transactions on Computers 55*, 10 (2006), 1217–1229. doi:10.1109/TC.2006.164. 11, 12, 14, 15, 20, 30, 46, 58, 63, 102

[4] HAYKIN, S. *Communication Systems, 4th Edition*. John Wiley & Sons, 2001. 46

[5] KOHLBRENNER, P., AND GAJ, K. An Embedded True Random Number Generator for FPGAs. In *Proceedings of the ACM/SIGDA 12th International Symposium on Field Programmable Gate Arrays (FPGA'04)* (2004), ACM, pp. 71–78. doi:10.1145/968280.968292. 10, 11, 14, 30, 45, 87, 102

[6] MARSAGLIA, G. DIEHARD: A Battery of Tests of Randomness. 1996. 7, 15, 38, 46, 47, 77, 103, 108

[7] NIST. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Special Publication 800-22 Revision 1a, April 2010. 7, 15, 37, 38, 46, 47, 65, 77, 103, 108

[8] SKLAR, B. *Digital Communications - Fundamentals and Applications, Second Edition*. Prentice Hall PTR, 2001. 46, 48, 49

[9] SUNAR, B., MARTIN, W. J., AND STINSON, D. R. A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. *IEEE Transactions on Computers 56*, 1 (2007), 109–119. doi:10.1109/TC.2007.250627. 4, 11, 12, 14, 15, 19, 20, 21, 30, 31, 32, 34, 37, 39, 41, 45, 53, 58, 63, 77, 78, 83, 87, 101, 102, 103, 105, 106, 108, 109, 125, 126

[10] TKACIK, T. E. A Hardware Random Number Generator. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 450–453. doi:10.1007/3-540-36400-5_32. 10, 14, 30, 45, 102

[11] VASYLTSOV, I., HAMBARDZUMYAN, E., KIM, Y.-S., AND KARPINSKYY, B. Fast Digital TRNG Based on Metastable Ring Oscillator. In *Proceedings of the 10th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'08)* (2008), vol. 5154 of *Lecture Notes in Computer Science*, Springer, pp. 164–180. doi:10.1007/978-3-540-85053-3_11. 13, 14, 20, 31, 46, 58, 63

[12] WOLD, K., AND TAN, C. H. Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings. In *Proceedings of the 4th IEEE International Conference on ReConFigurable Computing and FPGAs (ReConFig'08)* (2008), pp. 385–390. doi:10.1109/ReConFig.2008.17. 12, 14, 20, 42, 46, 47, 48, 52, 53, 54, 58, 64, 77, 78, 79, 80, 83, 87, 92, 99

Chapter 6

Robustness of TRNG against Attacks that Employ Superimposing Signal on FPGA Supply Voltage[1]

Abstract

A true random number generator (TRNG) is an important component of today's cryptographic systems. The TRNG should therefore be secure against various kinds of attacks. A typical attack against this kind of device is to superimpose a deterministic signal on the supply voltage in order to degrade the quality of the random signal output. In this paper, we investigate several TRNG designs implemented in a field programmable gate array (FPGA) in order to study the influence of this attack on the generated random output. The investigation shows that a class of TRNG based on several oscillator rings such as those proposed by Sunar et al. and Wold et al. are robust against such an attack, while the quality of two other TRNG designs was reduced to certain extent by this attack. In addition, random bits generated by the TRNG of Wold et al. under such an attack pass the NIST statistical test suite.

6.1 Introduction

A true random number generator (TRNG) is used in a cryptographic system to generate streams of random bits. These bit streams can be used for generating keys, initial vectors used in encryption or cryptographic challenges/responses. In order to maintain a high level of security in these systems, it is important that the properties or requirements to the TRNG are met for all conditions. For instance, each new generated bit should be unpredictable, i.e. it should not be possible to predict the next bit with a higher probability than 0.5. If an attacker can manipulate the TRNG in such a manner that the generated random bit stream has a majority of either zeros or ones, the result is reduced security of the cryptographic system. Therefore, a vital requirement on a TRNG is to maintain good statistical properties under all conditions.

In [8], a model of jitter in the oscillator rings implemented in a field programmable gate array (FPGA) is proposed. One of the findings is that the accumulated jitter in an oscillator ring is more influenced by the deterministic interference from the supply voltage than by the random Gaussian noise. The deterministic interference signal can originate from the supply voltage regulator itself or from an attack. Therefore, it should be investigated whether a TRNG based on oscillator rings is secure against an attack that employs manipulating the supply voltage to the FPGA device. In practice, this attack can be performed by superimposing a deterministic signal on the supply voltage to the FPGA. The expected effect is that this manipulation will influence the generated random output from the TRNG in a negative way, and thereby undermine the security of the TRNG.

[1] Knut Wold and Slobodan Petrović. In Proceedings of the Norwegian Information Security Conference (NISK'10), pp. 81–92, 2010.

In this paper, an attack based on superimposing a deterministic signal to the supply voltage is performed on several TRNG designs. Four designs were considered: The multi ring generator proposed by Wold et al. [10], the multi ring generator proposed by Sunar et al. [7], the five stage metastable oscillator ring generator proposed by Vasyltsov et al. [9] and the FIGARO generator proposed by Golić [4]. Two classes of the examined generators are based on oscillator rings (Sunar et al. and Wold et al.) and the others are different in structure. The idea was to compare the influence of structure (ring oscillator-based, non ring oscillator-based) on resistance against the attack mentioned above. In all these experiments, bit streams were recorded and the statistical properties were investigated. We show by simulations that the influence of this interference does not damage the behavior of the class of TRNG based on several oscillator rings. Then we verify the simulation results by showing in practical implementation that TRNG designs based on several oscillator rings are not influenced by such an attack, while the other TRNG designs do show vulnerabilities when exposed to such attacks.

The rest of the paper is organized as follows: In Section 6.2, the model of accumulation of jitter from [8] is presented. In Section 6.3, we perform simulations in MatLAB showing that a TRNG based on several oscillator rings is not influenced by an attack that employs superimposing a deterministic signal on the supply voltage. In Section 6.4, we present results of the experiments on different TRNG designs with superimposed signal on the supply voltage and investigate the effect of that signal on the bias of the output bit stream. In Section 6.5, a discussion of the observed behavior is given, and finally, we make a conclusion in Section 6.6.

6.2 Background

An oscillator ring is made up of an odd number of inverters. In order to achieve oscillations, a ring must contain at least three inverters. Instead of inverters, delay elements combined with one inverter could be used, and thereby also allow an even number of elements in the ring. The total period of this oscillator ring is determined by the delay of all the individual gates in the ring. In [8], the most important and dominant contributions to this individual gate delay are modeled as:

$$d_i = D_i + \Delta d_i = D_i + \Delta d_{LGi} + K_i \Delta d_{GD} \tag{6.1}$$

where D_i is a constant gate delay including the delay due to the routing to the next gate, Δd_{LGi} is the local Gaussian jitter characterized with a standard deviation of σ_i, K_i is a constant corresponding to the proportion of the global deterministic jitter on the gate delay and Δd_{GD} is the global deterministic jitter caused by global conditions such as changes in supply voltage, temperature, etc.

Each of these two jitter components are accumulated in time as shown in [8]. The accumulated local Gaussian jitter is proportional to the square root of the time, while the global deterministic jitter is linearly proportional to the time. This indicates that the accumulated jitter in an oscillator ring is more dependent on the global deterministic jitter than on the Gaussian jitter. Since the entropy source of a TRNG based on oscillator rings is the Gaussian jitter, there is a danger that an artificially introduced deterministic signal would dominate over the effect of the Gaussian jitter and thereby significantly reduce the quality of randomness at the TRNG output.

6.3 Simulation

In order to investigate the effect of superimposing a deterministic signal on the voltage supply of an FPGA configured with a TRNG based on oscillator rings, the TRNG proposed by Wold et al. [10] is examined as a typical representative of such schemes. This TRNG consists of 25 oscillator rings where each ring is built up of three inverters. The output of

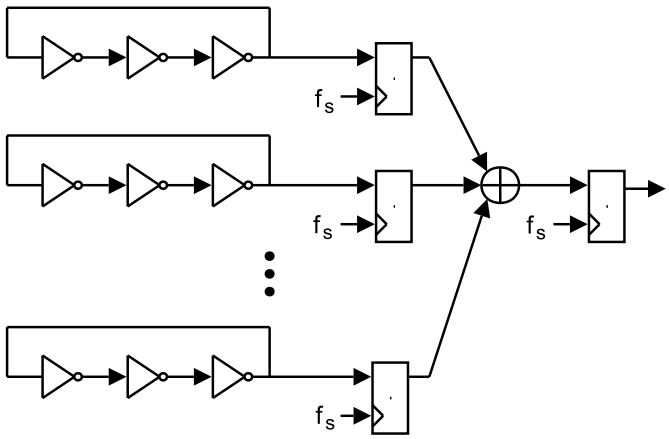

Figure 6.1: TRNG based on several oscillator rings.

each ring is sampled by a D flip-flop with sampling frequency f_s, and then XORed together. The final random output is sampled at the output of the XOR-tree, see Figure 6.1.

A MatLab simulation of such a scheme was run where several restarts of one oscillator ring were performed and the point in time where a transition occurred was recorded. The default oscillator ring period was set to 2420ps (i.e. the frequency was approximately 413MHz), which corresponds to an average oscillator ring with three inverters implemented in an Altera Cyclone II FPGA. The size of the standard deviation of the jitter of one inverter was set to 30ps, which corresponds to what was used in [1] and found in [8]. Figure 6.2 shows the distribution of the time instants where the transitions occur. It is observed that the peaks are getting wider for each clock cycle, indicating that the jitter is accumulated. From the simulation, the accumulated standard deviation of each half clock period related to the ideal position in time without any jitter, can be calculated, see Figure 6.3. It is observed that the accumulated Gaussian jitter follows the square root characteristic.

A variation of the supply voltage to the FPGA will affect the frequency of the oscillator rings [11]. An increase in the supply voltage will result in an increase in the oscillator ring frequency. Adding a deterministic signal on to the supply voltage will make the oscillator ring frequency alter in the same manner as the shape of the amplitude of the deterministic

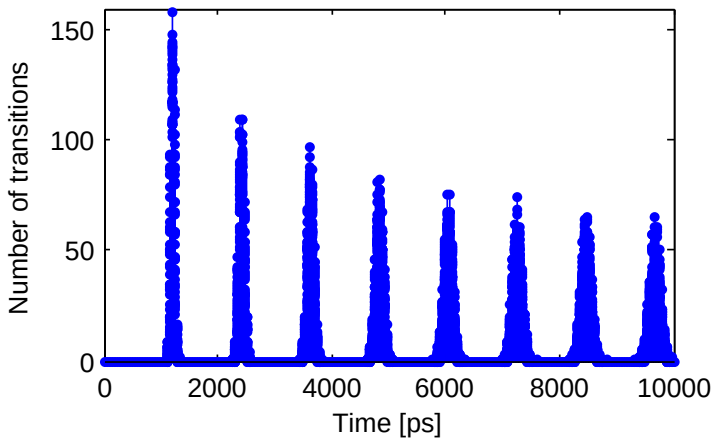

Figure 6.2: Distribution of transitions after restarts of an oscillator ring with jitter.

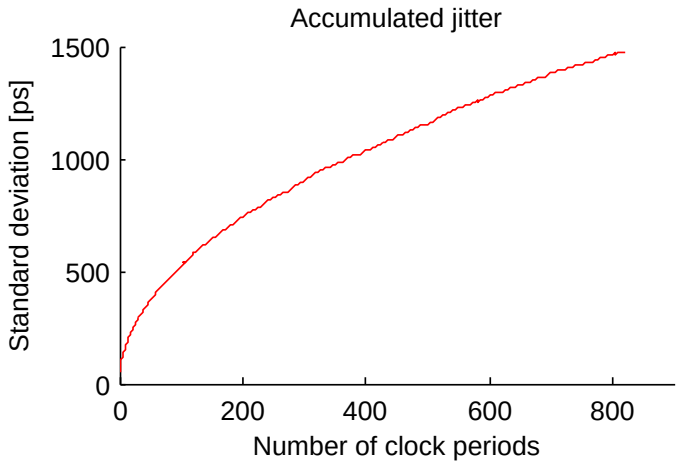

Figure 6.3: Accumulated standard deviation after restarts of an oscillator ring with jitter.

signal. In the MatLab simulation, the magnitude of the variation of the oscillator ring frequency from the ideal ring delay was set to 10%, which emulates the effect of changing the amplitude of the deterministic signal within the valid range of the FPGA supply voltage. The level of this disturbance depends on the FPGA used and other parameters, but 10% is considered a large variation (see [11]). In the MatLab simulations, the chosen deterministic signal has a ramp-up shape (similar to what was used in [8]) with a frequency of 20MHz. In Fig 6.4, the upper curve represents the accumulated jitter consisting of both Gaussian jitter and the deterministic signal, while the lower curve is the same as seen in Figure 6.3. It is observed that the accumulated jitter has the regular shape of the deterministic jitter, and it is linearly proportional to the time, which confirms what was found in [8]. The accumulated jitter in a single oscillator ring is dominated by the contribution from the superimposed signal.

In the simulations performed so far, it has been assumed that the oscillator rings have had identical frequencies. In a TRNG based on several oscillator rings, each inverter is placed into physical logic elements in the FPGA. Due to this placement, there will be a

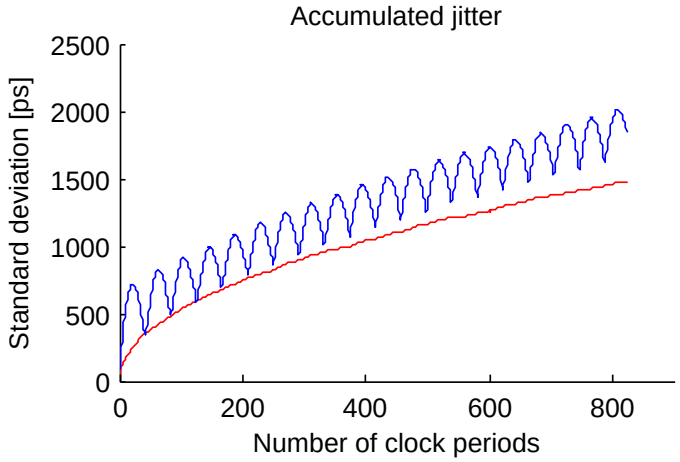

Figure 6.4: Accumulated standard deviation after restarts of an oscillator ring with jitter and a deterministic signal.

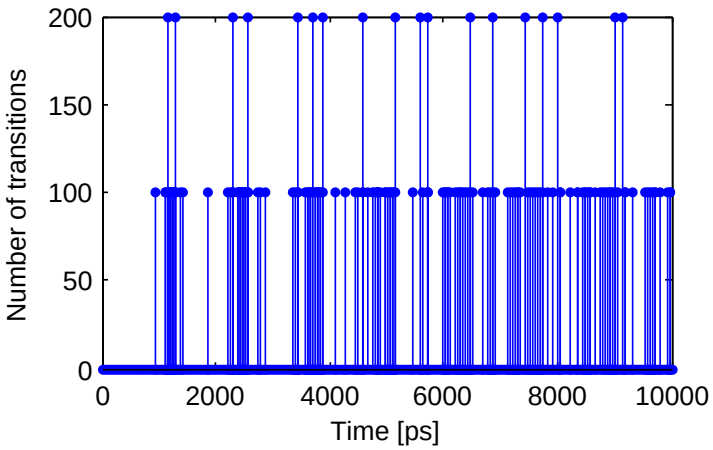

Figure 6.5: Distribution of transitions after restarts of an oscillator ring with deviation in frequencies.

variation between the ring frequencies due to variation of the delay through each element, but also due to different length of the routing between the logic elements where the inverters are physically placed. This variation in frequencies influences the behavior of the class of TRNG based on several oscillator rings. In the simulation, the distribution of the oscillator ring frequencies is chosen to be a Gaussian distribution. The standard deviation of the distribution is set to $\sigma_f = 100ps$, which is a low estimate compared to the real deviation of frequencies in an FPGA with an optimal choice of the number of inverters in a ring. In order to find the effect on the accumulated jitter with several rings, a simulation was performed in MatLab. Figure 6.5 shows the distribution of time instants where the transitions occur. In the MatLab simulation, this is a pure deterministic system and the transitions occur at the same points in time for every restart. In a real TRNG implementation, the oscillator ring frequencies will vary and drift due to changes in the temperature of the FPGA (both from self heating and the environment) and due to differences in the supply voltage because of variations in the power consumption of the FPGA.

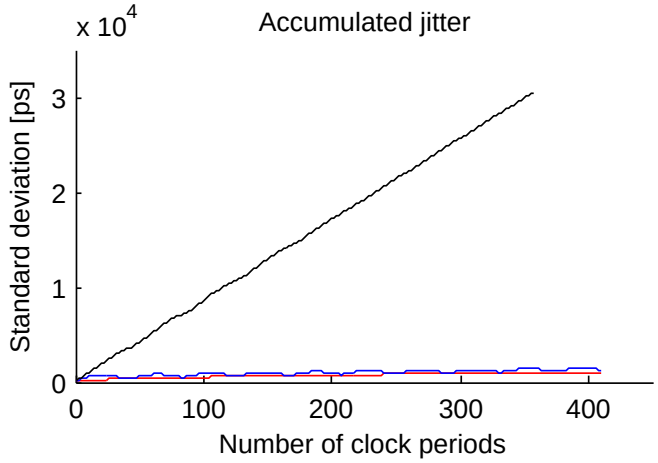

Figure 6.6: Accumulated standard deviation after restarts of an oscillator ring with jitter, deterministic signal and deviation in frequencies.

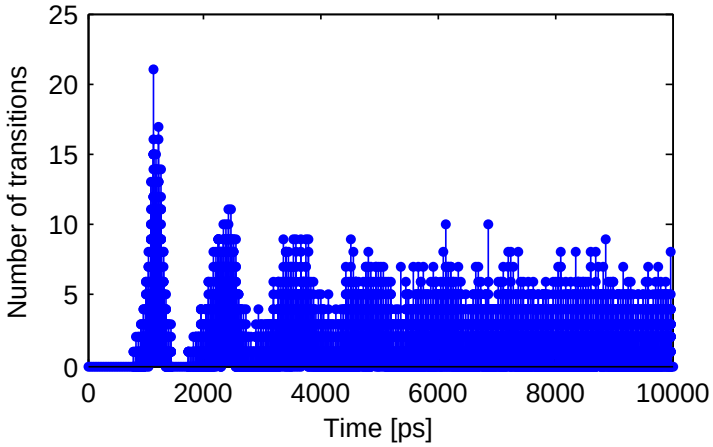

Figure 6.7: Distribution of transitions after restarts of an oscillator ring with jitter, deterministic signal and deviation in frequencies.

The accumulated standard deviation with increasing number of clock cycles is shown in Figure 6.6. The two lower curves are the same curves given in Figure 6.4, while the upper curve represents the accumulated standard deviation of jitter, deterministic signal and the variation of the oscillator ring frequencies. It is observed that the effect of the variation of the frequencies dominates over the two others, and the regular shape introduced by the deterministic signal has disappeared. The accumulated standard deviation of both the deterministic signal and the variation of the frequencies is linearly proportional to time, but the proportional factor (K_i) of the variation of frequencies is larger and therefore dominates.

In Figure 6.7, the distribution of transitions obtained by the circuit simulating with jitter, deterministic signal and variation of frequencies is presented. It is observed that after a start up period, the transitions are quickly uniformly distributed.

6.4 Experimental Work

In order to verify the robustness of different TRNG designs against an attack that employs superimposing a deterministic signal to the supply voltage, experiments on an Altera Cyclone II FPGA Starter Board were performed. A ramp-up signal was superimposed on the

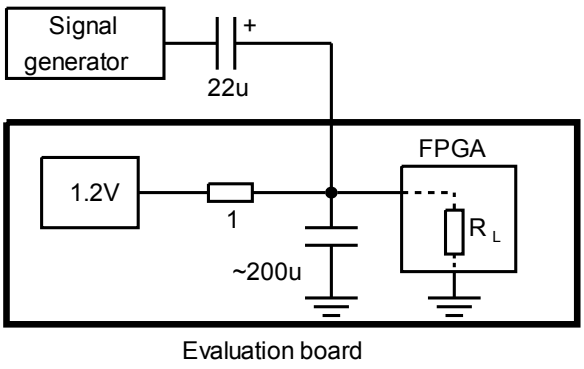

Figure 6.8: Setup of the attack.

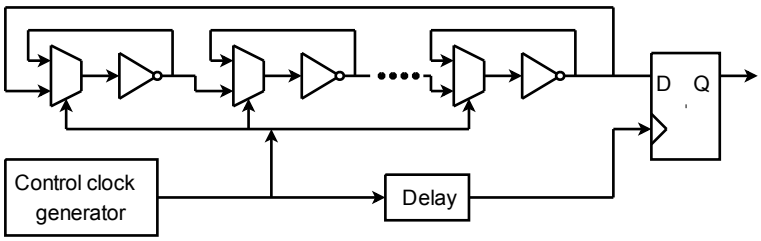

Figure 6.9: 5-stage metastable TRNG.

1.2V core voltage supply to the FPGA. The setup of the experiment is shown in Figure 6.8. The amplitude of the noise was adjusted in such a manner that the limits (1.15V-1.25V) of the power supply requirements were not exceeded. The frequency of the ramp-up signal was varied from 100Hz up to 20MHz. Four different TRNG designs were investigated. For each design, 100 blocks of data of 1Mbit each were captured. The experiment was first performed without any superimposed signal, and then with the superimposed signal with different frequencies. The bias or the deviation from the ideal 50% level between zeros and ones is calculated for each case.

The four different TRNG designs used were[2]:

- The 5 stage metastable oscillator ring, proposed by Vasyltsov et al. [9], see Figure 6.9. Each stage consists of only a multiplexer and an inverter. The multiplexers are controlled such that in the first phase, each of the five inverters is connected in a loop (output to input) so that they are put into a metastable state for 120ns. In the second phase, the five inverters are connected as an oscillator ring in 120 ns with start values depending on the metastable states. A random bit is sampled at the output of the ring 60ns after the ring starts to oscillate.

- FIGARO proposed by Golić [4], see Figure 6.10. It consists of a Galois ring oscillator (GARO) and a Fibonacci ring oscillator (FIRO) where the outputs are combined with an XOR and the random bits are sampled at the output of the XOR. The GARO and FIRO are linear feedback shift register (LFSR) structures with inverters used as delay elements instead of register elements as in an ordinary LFSR.

- Multi rings, proposed by Sunar et al. [7], see Figure 6.11. It consists of 114 equal length oscillator rings made up of 13 inverters. The outputs of these rings are XORed

[2]The TRNG designs are implemented as they are proposed in the respective papers.

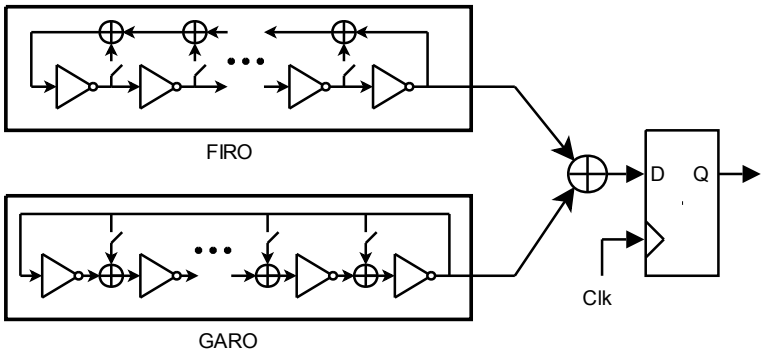

Figure 6.10: FIGARO TRNG.

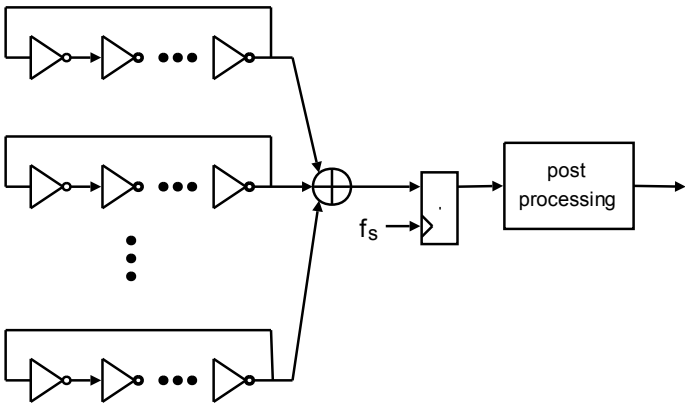

Figure 6.11: Multi ring TRNG (Sunar).

and a random bit is generated by sampling the output of the XOR. In their proposal, a post-processor based on a resilient function is used.

- Multi rings, proposed by Wold et al. [10], see Figure 6.1. It is an enhancement of the proposal by Sunar with 25 rings with 3 inverters each and where each ring is sampled before the XOR. The random output is sampled at the output of the XOR.

For all these designs, no post-processor was used in the experiments. It is well known that a post-processor would improve the quality of the random sequence and thereby mask out the effect of the attack. In order to investigate the impact of the attack on the TRNG designs, this factor was eliminated.

Based on the simulations in Section 6.3, the TRNG designs by Wold and Sunar should not be influenced by this attack. For comparison, the two other designs were checked in order to observe differences between the design structures. Table 6.1 and Figure 6.12 show the amount of ones in the 100 blocks for each of the four TRNGs without and with the deterministic signal at different frequencies. It is observed that for the two TRNGs based on several oscillator rings, the bias is approximately identical through the attack that employs superimposing a deterministic signal to the supply voltage. The difference is that the proposal by Sunar et al. has a bias due to speed problems in the XOR-tree and the sampling flip-flop as pointed out in [2].

The design based on a 5-stage metastable oscillator ring and the FIGARO TRNG are more sensitive to this sort of attack. From Figure 6.12 it is observed that the bias is dependent on the frequency of the deterministic jitter signal and that low frequencies give the largest deviation in the bias compared to the case with no superimposed noise. The

Table 6.1: Amount of ones of different TRNGs in FPGA.

Noise	Vasyltsov	Golić	Sunar	Wold
No	0.509562	0.507730	0.433899	0.499946
100Hz	0.496580	0.497653	0.433136	0.500064
1kHz	0.505234	0.503617	0.433099	0.500059
10kHz	0.507142	0.504324	0.433461	0.500019
100kHz	0.511818	0.511931	0.433619	0.499980
1MHz	0.507998	0.511813	0.433617	0.500022
10MHz	0.507896	0.511432	0.433775	0.500017
20MHz	0.507389	0.511842	0.433497	0.500022

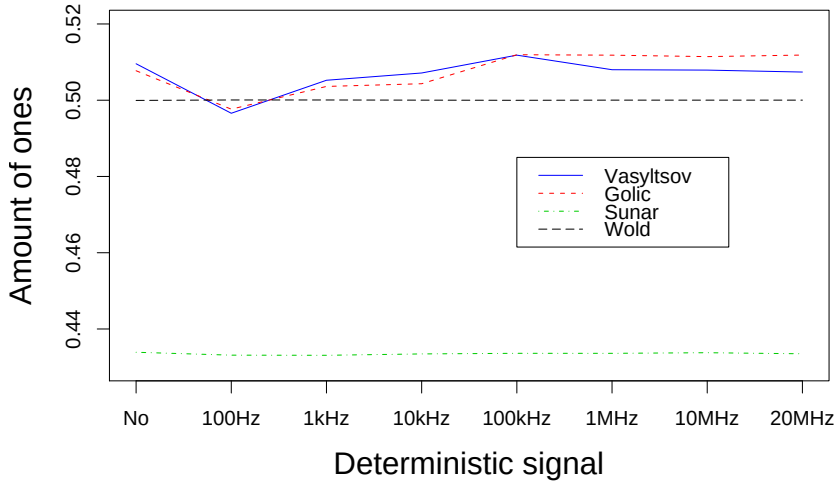

Figure 6.12: Bias of the TRNG designs with deterministic jitter.

results from the experiments show that the amount of ones is varying only 1-2% by this attack. In most cases this can be regarded as a small variance and it is of no importance regarding the security of the TRNG, but these numbers are the mean calculated over a total of 100Mbit. An inspection of a smaller part of a block (first 20000bit) in the FIGARO TRNG with a 100Hz noise signal, reveals a variation in the bias among the blocks as high as $\pm 5\%$. In a high grade crypthographic system this variation is high enough to be considered a security problem. In a real implementation of a TRNG in a cryptographic system, it is recommended to use a post-processor in order to compensate for such imbalances and improve the statistical properties. But the statistical properties of the output sequence from the TRNG itself should be as good as possible.

Based on the observation that a low frequency on the superimposed noise signal has higher influence on the bias, another experiment was performed where the deterministic signal was a square wave with frequency 500Hz. The choice of this frequency was based on a compromise between a low noise signal frequency and the need by the attacker that the filter of the power supply voltage on the FPGA evaluation board should not remove the superimposed signal. The square wave was chosen in order to simulate shifts in the power supply voltage. For all four TRNG designs data were collected and examined. Figure 6.13 shows the probability density of ones for all four TRNGs. It is observed that the FIGARO TRNG has a large deviation from the ideal 0.5 level with an average of the number of ones as high as 0.57 indicating a majority of ones in the TRNG output. The variation between the blocks is also large, from 0.48 to 0.70 when examining the first 20000bits of the blocks. This indicates that the FIGARO TRNG is not robust against this attack. The three other TRNGs are less influenced by this attack, but the 5-stage metastable TRNG has a larger deviation in the bias than the two TRNGs based on several oscillator rings.

Even though the bias was not altered by this attack for the TRNGs based on several oscillator rings, the statistical properties of the generated bit streams could have been influenced. In order to verify this, the generated random data streams from the TRNG proposed by Wold et al. were checked by means of the NIST statistical test suite [6][3]. The result is that the NIST test passed for all the tested cases independent on whether a deterministic signal

[3]Only the NIST statistical test was used to check the quality of the bit sequence.

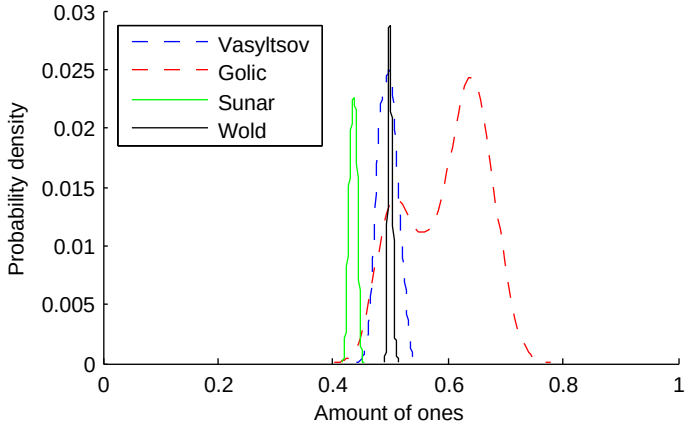

Figure 6.13: Density of the bias of the TRNG designs.

was superimposed or not. This shows that a TRNG based on several oscillator rings proposed by Wold et al. is robust and not influenced by such an attack and that the statistical properties of the TRNG output are maintained.

6.5 Discussion

The reason why a TRNG based on several oscillator rings is robust against an attack that employs manipulating the supply voltage to an FPGA, is that this attack changes the amplitude of the signals, while this TRNG uses the frequency or time of the transitions of the signals in order to extract the randomness. Manipulating the amplitude will change the frequency of an oscillator ring and thereby the time of a transition, but the uniform distribution of the transition regions will still be maintained. The simulations have shown that the superimposed deterministic signals do influence on the accumulated jitter of one oscillator ring, but a TRNG consists of several rings, and the effect of the variation of frequencies in the rings removes the regular shape of the standard deviation of the accumulated jitter. The result is that this sort of attack has no influence on the generated random output. The main contribution to the diffusion of the transition regions in order to achieve a uniform distribution is the difference in the frequencies of the oscillator rings. The jitter is important and absolutely necessary to create the true randomness. Without the jitter, the output of a TRNG based on several oscillator rings would be pseudo random.

The FIGARO TRNG is influenced by the interference on the supply voltage. This can be explained by looking at the outputs from the FIRO and GARO, which are noisy signals in the range between 1V and 3V. The threshold voltage of whether the sampled voltage is regarded as zero or one should be in the middle, but due to the disturbance on the supply voltage, this voltage threshold is fluctuating. The result is a change in the number of ones/zeros. For the metastable TRNG, the metastability is dependent on the voltage, and thereby it creates differences in the amount of zeros and ones.

On the FPGA evaluation board used in this experiment, the supply voltage is generated from a linear voltage regulator. The voltage is decoupled to signal ground with a lot of 100nF capacitors placed around the FPGA, and also a couple of large capacitors of 100μF. All these capacitors filter the supply voltage and try to keep the voltage at a stable level. A linear regulator generates much less noise compared to switching voltage regulators. In [8], their FPGA board contained a switching power supply with a more noisy behavior, and this noise was leaked into the FPGA and created the effects on the jitter of the oscillator rings. An important design criterion of a TRNG is therefore that the supply voltage to the

FPGAs is stable and introduces little noise.

The observation that the variation in the ring frequencies is important, opens a possibility to attack a TRNG based on several oscillator rings. If an attacker can force the rings to oscillate at the same frequency, the accumulated standard deviation will only depend on the Gaussian jitter. Such an attack could be carried out by injection locking. In [5], an example of such an attack on a smart card is shown, but this is only shown when the number of oscillator rings is one or a small one. In a TRNG based on oscillator rings, several of the oscillator rings must be influenced in order for such an attack to be successful. This might be difficult to achieve because the oscillator ring frequencies span over a wide frequency range. This dispersion is due to the fact that the placement of the inverters in logic elements in the FPGA results in different routing delay for each oscillator ring resulting in a variation in the frequency of the rings.

Another possible attack could be to change the temperature quickly while the TRNG generates random bits. The effect of a temperature change is that the frequencies of the oscillator rings are altered, see [11]. The effect on the TRNG will be the same as by superimposing a deterministic signal, and the statistical properties of the random output will not be destroyed by such an attack.

In all these TRNG implementations the sampling clock was a stable external clock not influenced by changes in the supply voltage. In [3], a suggested countermeasure against the attack that employs superimposing a deterministic signal on the supply voltage, is to use a sampling clock made up by an oscillator ring. The result is that the sampling clock has the same fluctuations as the oscillator rings, and the effect of the deterministic signal is reduced or removed. This countermeasure will indeed improve the accumulated jitter of a single ring, but as shown in this paper, this countermeasure is not necessary in a class of TRNG based on several oscillator rings.

6.6 Conclusion

In this paper, we have shown that a class of TRNGs made up of several oscillator rings are not influenced by the attack that employs superimposing a deterministic signal to the supply voltage of an FPGA. Simulations in MatLab were performed in order to investigate the behavior of a single oscillator ring. The simulations confirm what was found in [8], that the deterministic jitter is linearly proportional to the time and the shape of the deterministic signal is transfered to the accumulated jitter, and dominates over the Gaussian jitter, which is proportional to the square root of the time. But this TRNG consists of more than one oscillator ring. Due to the placement in an FPGA, the oscillator ring frequencies will be different for each ring. Including this variation into the simulation shows that the accumulated jitter depends much more on this variation than on the deterministic signal.

In order to verify the simulations, experiments on an Altera evaluation board containing a Cyclone II FPGA were performed. The experiments were carried out on four different TRNG designs. For all designs, generated random bits were recorded before and during an attack that superimposes a deterministic signal to the supply voltage of the FPGA, and the bias was calculated. The result is that the TRNGs based on several oscillator rings were not influenced by the attack, while the two other TRNG designs were influenced. The variation in the bias for these two designs was about 1-2% when looking at a large number of bits, but when investigating smaller parts of the blocks the variation was higher. When optimizing the frequency of the noise signal the influence on bias was large on one of the TRNG designs. In a high grade cryptographic system this variation could be high enough to be regarded as a security problem. The quality of the random bit streams generated by the TRNG of Wold et al. influenced by such an attack, was checked by the NIST statistical test suite. The NIST test passed for all the generated random bit streams, showing that a TRNG made up by several oscillator rings by Wold et al. is robust against such an attack.

6.7 Bibliography

[1] Bochard, N., Bernard, F., and Fischer, V. Observing the Randomness in RO-based TRNG. In *Proceedings of the 5th IEEE International Conference on ReConFigurable Computing and FPGAs (ReConFig'09)* (2009), pp. 237–242. doi:10.1109/ReConFig.2009.57. 59, 78, 80

[2] Dichtl, M., and Golić, J. D. High-Speed True Random Number Generation with Logic Gates Only. In *Proceedings of the 9th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'07)* (2007), vol. 4727 of *Lecture Notes in Computer Science*, Springer, pp. 45–62. doi:10.1007/978-3-540-74735-2_4. 11, 12, 13, 16, 30, 31, 32, 39, 64, 102, 103

[3] Fischer, V., Bernard, F., Bochard, N., and Varchola, M. Enhancing Security of Ring Oscillator-Based TRNG Implemented in FPGA. In *Proceedings of the 18th International Conference on Field Programmable logic and Applications (FPL'08)* (2008), IEEE, pp. 245–250. doi:10.1109/FPL.2008.4629939. 67

[4] Golić, J. D. New Methods for Digital Generation and Postprocessing of Random Data. *IEEE Transactions on Computers 55*, 10 (2006), 1217–1229. doi:10.1109/TC.2006.164. 11, 12, 14, 15, 20, 30, 46, 58, 63, 102

[5] Markettos, A. T., and Moore, S. W. The Frequency Injection Attack on Ring-Oscillator-Based True Random Number Generators. In *Proceedings of the 11th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'09)* (2009), vol. 5747 of *Lecture Notes in Computer Science*, Springer, pp. 317–331. doi:10.1007/978-3-642-04138-9_23. 17, 67

[6] NIST. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Special Publication 800-22 Revision 1a, April 2010. 7, 15, 37, 38, 46, 47, 65, 77, 103, 108

[7] Sunar, B., Martin, W. J., and Stinson, D. R. A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. *IEEE Transactions on Computers 56*, 1 (2007), 109–119. doi:10.1109/TC.2007.250627. 4, 11, 12, 14, 15, 19, 20, 21, 30, 31, 32, 34, 37, 39, 41, 45, 53, 58, 63, 77, 78, 83, 87, 101, 102, 103, 105, 106, 108, 109, 125, 126

[8] Valtchanov, B., Aubert, A., Bernard, F., and Fischer, V. Modeling and Observing the Jitter in Ring Oscillators Implemented in FPGAs. In *Proceedings of the 11th IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems (DDECS'08)* (2008), pp. 1–6. doi:10.1109/DDECS.2008.4538777. 9, 57, 58, 59, 60, 66, 67, 79, 80

[9] Vasyltsov, I., Hambardzumyan, E., Kim, Y.-S., and Karpinskyy, B. Fast Digital TRNG Based on Metastable Ring Oscillator. In *Proceedings of the 10th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'08)* (2008), vol. 5154 of *Lecture Notes in Computer Science*, Springer, pp. 164–180. doi:10.1007/978-3-540-85053-3_11. 13, 14, 20, 31, 46, 58, 63

[10] Wold, K., and Tan, C. H. Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings. In *Proceedings of the 4th IEEE International Conference on ReConFigurable Computing and FPGAs (ReConFig'08)* (2008), pp. 385–390. doi:10.1109/ReConFig.2008.17. 12, 14, 20, 42, 46, 47, 48, 52, 53, 54, 58, 64, 77, 78, 79, 80, 83, 87, 92, 99

[11] YOO, S.-K., KARAKOYUNLU, D., BIRAND, B., AND SUNAR, B. Improving the Robustness of Ring Oscillator TRNGs. *ACM Transactions on Reconfigurable Technology and Systems 3*, 2 (May 2010), 1–30. Article 9. doi:10.1145/1754386.1754390. 17, 59, 60, 67, 88, 95

Chapter 7

Security Implications of Crosstalk in Switching CMOS Gates[1]

Abstract

The energy dissipation associated with switching in CMOS logic gates can be used to classify the microprocessors activity. It is relatively easy to classify activity by the number of transitions, i.e. Hamming Distance (HD). In this paper we consider layout dependent phenomena, such as capacitive crosstalk to derive a more precise power model. We show that for an 8 bit data bus, crosstalk may improve detection performance from 2.5 bits (HD based detector) to theoretical 5.7 bits and simulated 5.0 bits (crosstalk based detector) of information pr sample. Thus we have shown that a layout specific phenomenon (capacitance) must be considered when analyzing security implications of power and electromagnetic side channels. A small case study has also been carried out that support our simulations/theoretical results.

7.1 Introduction

When a microprocessor executes its program, power consumption (or resulting electromagnetic emanation) can be used to reveal the contents of program and/or data memory of the microprocessor. The correlation between power consumption and microprocessor activity is well known and has found many uses [1, 2, 5, 7, 10].

In a side-channel attack a common power model, used to simulate the power consumption, is the Hamming Distance (HD) model, as it is simple and generic [8]. The model assumes the power consumption to be proportional to the number of transitions taking place. If this assumption was correct, signals transmitted on a parallel bus (e.g. intermediate values of the cryptographic algorithm) with the same HD should have equal power consumption and therefore be indistinguishable. This is not always the case, as demonstrated by the template attack [2, 6]. The phenomena behind this may be known in the security community, but has received little attention. One paper by Chen et al. [3] investigates the effect of the coupling capacitance on masking schemes without a detailed examination of the phenomena. In their book "Power Analysis Attacks", Mangard et.al. [8] mention power simulation at analog level as "the most precise way to simulate the power consumption of digital circuits...". Parasitic elements, such as parasitic capacitances between the wires and unwanted capacitances in the transistors are mentioned. However, it is also stated that it is very common to make simplifications by lumping together extrinsic and intrinsic capacitances into a single capacitance to ground. This will in fact make the model incapable of explaining the results we are addressing in this paper.

Parasitic couplings, and the coupling capacitance in particular, is however a great concern within sub-micron VLSI design [4, 9, 11]. Parasitic couplings between interconnects, such as on-chip buses, influence both the power consumption and maximum obtainable speed [4]. Moll et al. [9] did a detailed analysis of the energy dissipation from two metal

[1] Geir Olav Dyrkolbotn, Knut Wold and Einar Snekkenes. In Proceedings of the 13th Information Security Conference (ISC'10), Lecture Notes in Computer Science, vol. 6531, pp. 269–275, Springer, 2011.

lines running close together and show that "coupling capacitance is very different from the capacitance to ground because it depends on the switching activity... ". Duan et al. [4] show that for 3 adjacent lines, transition patterns can be divided into 5 crosstalk classes based on the influence of the coupling capacitances. The focus in VLSI design, such as [4, 9], is on power consumption and delays, not on security.

In this paper we look at the security implication of capacitive crosstalk in switching CMOS gates. We put forward the hypothesis that layout dependent phenomena, such as parasitic coupling between wires, can explain why it sometimes is possible to distinguish transition patterns with the same HD. We present a new power model that takes into account capacitive coupling between parallel wires. Theory and PSPICE simulations are used to evaluate how the new power model effects our ability to classify activity in a microprocessor. We use the ability to extract entropy as our classifier performance indicator and show that a detector capable of detecting energy levels due to crosstalk can extract more information than a detector based on HD only.

This paper is organized as follows: Section 7.2 presents the hypothesis of layout dependent phenomena. Section 7.3 presents our model and necessary theory to calculate the energy dissipation. Section 7.4 is an analytic analysis of security implications. Section 7.5 presents simulation results. Finally a conclusion is drawn.

7.2 Layout Dependent Phenomena

In a physical implementation of any circuit (e.g. CMOS based microprocessor) a number of phenomena will influence the energy dissipation and the resulting radiated electromagnetic field. These phenomena includes inductance and capacitance of conductors, inductance and capacitance between conductors, wireless transmission characteristics (i.e antenna properties) of conductors and other circuit elements and complex combinations of these phenomena. These phenomena apply to any transistors and wires in a circuit, but we choose to look at a portion of wires running in parallel as we expect them to be relatively good antennas and therefore a good source for side channel information. We believe complex combinations of layout dependent phenomena may be the key to identify minute differences in microprocessor activity. In the following we will assume that the coupling capacitance is the dominating factor and show how this can explain why some signals with the same HD can be distinguished. This will show the potential effect of layout dependent phenomena on classifying microprocessor activity. Our work can easily be extended by including other layout dependent phenomena if a more precise result is needed.

7.3 Theoretical Considerations

A model of a parallel bus driven by CMOS inverters with only coupling and load capacitances is shown in Figure 7.1. This is a generalization of the model for two lines used by Moll et al. [9] and includes a model of the CMOS inverter.

It can be shown that the total energy dissipation, E_T from an n wire bus, can be written as:

$$E_T = \sum_{j=1}^{n} V_{DD} \int i_{pj} dt - \sum_{j=1}^{n} \int V_j (i_{pj} - i_{nj}) dt \tag{7.1}$$

For PSPICE simulation the following assumptions are used:

- The load capacitances for data bus lines are identical ($C_j = C_L$ for $j = \{1, 2, \cdots, n\}$)
- Coupling capacitances are only found between adjacent line and are identical ($C_{j,j+1} = C_C$ for $j = \{1, 2, \cdots, n-1\}$)

In order to compare the simulated energy dissipation ($\hat{E}_T$) with analytic values (E_T), a different expression than (7.1) is needed. If the contributions from the load (C_L) and

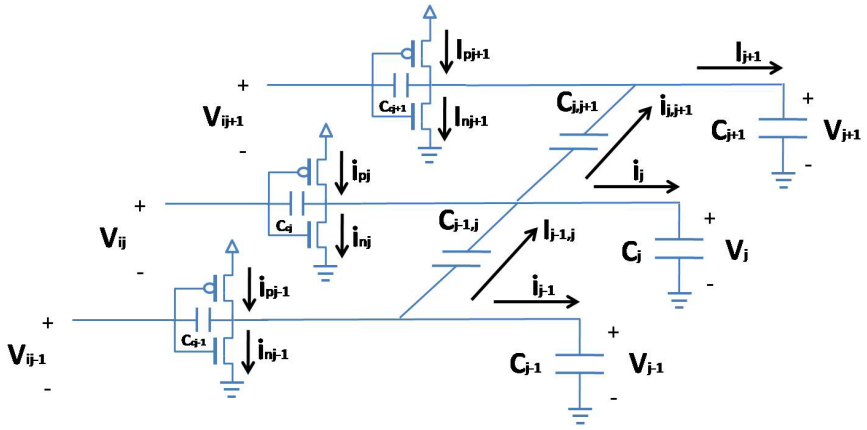

Figure 7.1: Simplified model, assuming load and coupling capacitances to be dominant

coupling capacitance (C_C) are dominant to the dissipated energy, then E_T can be expressed in the following power model:

$$E_T = \frac{1}{2} C_L V_{DD}^2 (k + \alpha\lambda) = E_0(k + \alpha\lambda) \tag{7.2}$$

where $E_0 = \frac{1}{2} C_L V_{DD}^2$, V_{DD} is the power supply voltage, k is the number of transitions on the data bus ($k = 0, 1, 2, \ldots$), $\lambda = C_C/C_L$ and α is the crosstalk index indicating the coupling capacitance induced crosstalk, similar to the crosstalk classes in [4]. For a n line bus the crosstalk index α is the sum of the crosstalk influence of each line:

$$\alpha = \sum_{j=1}^{n} \alpha_j \tag{7.3}$$

Let $\delta_j \in \{0, \pm 1\}$ be the normalized voltage change on line j, then $\delta_{j,k} = \delta_j - \delta_k$, and

$$\alpha_j = \begin{cases} 0 & no\ transition\ line\ j \\ |\delta_{j,j-1} + \delta_{j,j+1}| & otherwise \end{cases} \tag{7.4}$$

It can be shown that $\alpha_j = \{0, 1, 2\}$ for lines with only one adjacent line (edges), and $\alpha_j = \{0, 1, 2, 3, 4\}$ for lines with two adjacent lines. In the next section we will use (7.2) and (7.3) to analyze which transition patterns that can be distinguished.

7.4 Security Implications

How will the new power model (7.2) effect our ability to classify activity in a microprocessor, such as data transfer on a parallel bus?

Let A be the set of possible transitions on an n bit parallel bus. A model of the energy dissipation should ideally be able to distinguish all $|A| = 4^n$ transitions. This may not be possible, either because of simplifications of the model or physical properties such that multiple transitions patterns indeed uses the same amount of energy. A model can only distinguish transition patterns by the distinct energy levels explained by the model. A model that assumes energy dissipation proportional to the number of transition, can therefore only distinguish transition pattern into subsets A_k, $k = \{0, \cdots, n\}$ being subsets of A that has k transitions. The number of transition patterns in each subset is given by: $|A_k| = 2^n \binom{n}{k}$. Using the new power model (7.2), each subset A_k can be split into a number

of new energy levels, giving a number of smaller subsets A_k^α, where α is the crosstalk index of (7.3). For an 8 bit bus, taking into consideration the coupling capacitance increases the number of energy levels from 9 in the HD model to 93 in the crosstalk model, e.g. the 14336 transitions patterns with 5 transitions, previously indistinguishable, can now be split into 18 energy levels.

Finally, we have only shown how to split the subset A_k into smaller subsets A_k^α by considering the effect of the coupling capacitance (i.e α). This idea can easily be generalized, such that A_k is split into subsets A_k^β, where $|A_k| > |A_k^\beta|$, and β is the influence of other layout dependent phenomena, such as variations in coupling and load capacitance and inductance.

7.4.1 Classification Performance

For the purpose of comparing alternative detectors we will assume uniform random transition, thus for a 8 bit bus we would like the detector to extract 16 bits. We will use the ability to extract entropy as our classifier performance indicator. The entropy (i.e bits of information) extracted by a detector, when there are r energy levels, can be calculated using:

$$H(x) = -\sum_{i=1}^{r} p(x_i) log\ p(x_i) \tag{7.5}$$

In the following we have assumed a 8 bit bus width, thus there are $4^8 = 65536$ possible transitions. Call the detector that can extract 16 bits of information a level detector. If we assume that one only has bus activity when initial and final state are different, and that $0 \rightarrow 1$ and $1 \rightarrow 0$ can be distinguished, an observation will give us the following entropy: $-(1/2 log 1/2 + 1/4 log 1/4 + 1/4 log 1/4) = 3/2$ bits as we cannot distinguish $0 \rightarrow 0$ from $1 \rightarrow 1$. The theoretical optimum for an 8 bit bus with a 'transition detector' would be $8 \cdot 3/2\ bits = 12\ bits$. The entropy extracted by a HD detector is found using (7.5) with $r = 9$ and $p(x_i) = |A_{i-1}|/65536$ giving an entropy of 2.5 bits. The entropy extracted by a crosstalk detector is found using (7.5) with $r = 93$ and $p(x_i) = |A_{i-1}^\alpha|/65536$ giving an entropy of 5.7 bits. The difference between the ideal value of a level detector and the entropy extracted by other detectors, represent the amount of guessing needed when classifying an observation. By considering the coupling capacitance and not only HD, we extract more information out of each observation, therefore reducing the amount of "guessing" needed for classification. In the next section we present simulations validating the effect of the coupling capacitance.

7.5 Simulations

The simulations are performed in PSPICE with $C_L = 400fF$, $C_C = 250fF$, $V_{dd} = 3V$ and a rise- and fall-time of $200ps$ of the input voltages (same as [9]). The inverter drivers are equal and balanced. Equation (7.1) is used in PSPICE to find the simulated energy dissipation $\hat{E}_T$. Having first validated our simulations against the results of [4, 9] simulations for all possible subsets A_k^α were carried out. The results for $k = 5$ can be seen i Table 7.1.

The simulated energy levels $\hat{E}_T$ are similar to the analytic values E_T (7.2). The results confirm that energy consumption is proportional to the number of transitions and the crosstalk index, α. The crosstalk index depends on switching activity on adjacent lines and position, edge (one adjacent wire) or middle (two adjacent wires). A theoretical crosstalk detector capable of separating all 93 energy levels can extract 5.7 bits of information. It is expected that a practical crosstalk detector will extract less information, due to some subsets having almost equal energy levels. A random loss of 20% of the subsets will still on average have an entropy of 5.0. This still gives an information gain of 2.5 bits compared to the HD detector. The performance of the detectors are summarized in Table 7.2. We also have experimental data, suggesting that the average classification error gets reduced as α

Table 7.1: Analytic (E_T) and simulated ($\hat{E}_T$) dissipated energy when considering crosstalk for bus with 8 lines. k=5 is the number of transitions (Hamming Distance) and α is the crosstalk index

Transition pattern	**k**	α	E_T **[pJ]**	$\hat{E}_T$ **[pJ]**
0000 0000 → 0001 1111	5	1	10,1	10.4
0000 0000 → 1000 1111	5	2	11,3	11.5
0000 0000 → 0010 1111	5	3	12,4	12.6
0000 0000 → 1001 0111	5	4	13,5	13.8
0000 0000 → 0101 0111	5	5	14,6	14.9
0000 0000 → 1010 1011	5	6	15,8	16.0
0000 0010 → 0111 0001	5	7	16,9	16.8
0000 0010 → 1011 0001	5	8	18,0	17.9
0000 0010 → 0101 1001	5	9	19,1	19.3
0000 0010 → 1010 1001	5	10	20,2	20.4
0000 0010 → 0110 0101	5	11	21,4	21.3
0000 0010 → 1010 0101	5	12	22,5	22.5
0000 0010 → 0101 0101	5	13	23,6	23.5
0000 1010 → 1000 0101	5	14	24,8	24.5
0000 1010 → 0100 0101	5	15	25,9	25.6
0001 0100 → 0100 1010	5	16	27,0	27.0
0000 1010 → 0001 0101	5	17	28,1	27.9
0001 0100 → 0010 1010	5	18	29,3	29.1

distance ($\Delta\alpha = |\alpha_i - \alpha_j|$) increases. This supports our simulation/theoretical results, but details are omitted due to limited space.

7.6 Conclusion

It is known that one can distinguish bus activity generated from signal transitions having different HD. In this paper we put forward the hypothesis that layout dependent phenomena, such as inductance and capacitance in and between conductors and radiation properties of circuit elements, can explain why it sometimes is possible to distinguish transition patterns with the same HD. Our simulations show that capacitive crosstalk has a significant effect on gate energy dissipation. Our results confirm that the dissipated energy from CMOS switching gates depend not only on the HD, but also on the direction of switching activity on nearby data lines. Where as a HD based detector can provide about 2.5 bits of information pr sample, a crosstalk based detector will yield about 5.7 bits (theoretical) or 5.0 bits (simulated) of information pr sample - in all cases for an 8 bit bus. Thus we have shown that a layout specific phenomenon (capacitance) must be considered when analyzing security implications of electromagnetic side channels.

Table 7.2: Comparing the ability to extract information of different detectors for an 8 wire bus

type of detector	Entropy (information) [bits]
Level detector	16,0
Optimum transition detector	12,0
Crosstalk detector (theoretical)	5,7
Crosstalk detector (simulated)	5,0
HD detector	2,5

7.7 Bibliography

[1] AGRAWAL, D., BAKTIR, S., KARAKOYUNLU, D., ROHATGI, P., AND SUNAR, B. Trojan Detection using IC Fingerprinting. In *IEEE Symposium on Security and Privacy (SP'07)* (2007), pp. 296–310. doi:10.1109/SP.2007.36. 71, 113

[2] CHARI, S., RAO, J. R., AND ROHATGI, P. Template Attacks. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 13–28. doi:10.1007/3-540-36400-5_3. 17, 71, 113, 118

[3] CHEN, Z., HAIDER, S., AND SCHAUMONT, P. Side-Channel Leakage in Masked Circuits Caused by Higher-Order Circuit Effects. In *Proceedings of the 3rd International Conference on Information Security and Assurance (ISA'09)* (2009), vol. 5576 of *Lecture Notes of Computer Science*, Springer, pp. 327–336. doi:10.1007/978-3-642-02617-1_34. 71, 113

[4] DUAN, C., CALLE, V. H. C., AND KHATRI, S. P. Efficient On-Chip Crosstalk Avoidance CODEC Design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems 17*, 4 (April 2009), 551–560. doi:10.1109/TVLSI.2008.2005313. 71, 72, 73, 74, 114, 117, 118, 120

[5] DYRKOLBOTN, G. O., AND SNEKKENES, E. A Wireless Covert Channel on Smart Cards (Short Paper). In *Proceeding of the International Conference on Information Systems (ICIS'06)* (2006), vol. 4307 of *Lecture Notes in Computer Science*, Springer, pp. 249–259. doi:10.1007/11935308_18. 71, 113

[6] DYRKOLBOTN, G. O., AND SNEKKENES, E. Modified Template Attack Detecting Address Bus Signals of Equal Hamming Weight. In *Proceedings of the Norwegian Information Security Conference (NISK'09)* (2009), pp. 43–56. 71, 113, 123

[7] KOCHER, P. C., JAFFE, J., AND JUN, B. Differential Power Analysis. In *Proceedings of Advances in Cryptology (CRYPTO'99)* (1999), vol. 1666 of *Lecture Notes in Computer Science*, Springer, pp. 388–397. doi:10.1007/3-540-48405-1_25. 17, 71, 113

[8] MANGARD, S., OSWALD, E., AND POPP, T. *Power Analysis Attacks - Revealing the Secrets of Smart Cards*. Springer, 2007. 17, 71, 113

[9] MOLL, F., ROCA, M., AND ISERN, E. Analysis of Dissipation Energy of Switching Digital CMOS Gates with Coupled Outputs. *Microelectronics Journal 34*, 9 (September 2003), 833–842. doi:10.1016/S0026-2692(03)00133-2. 71, 72, 74, 114, 115, 116, 117, 118, 120

[10] QUISQUATER, J.-J., AND SAMYDE, D. Automatic Code Recognition for Smart Cards Using a Kohonen Neural Network. In *Proceedings of the 5th Smart Card Research and Advanced Application Conference* (2002), vol. 5, USENIX. 71, 113

[11] SOTIRIADIS, P. P., AND CHANDRAKASAN, A. Low Power Bus Coding Techniques Considering Inter-wire Capacitances. In *Proceedings of the IEEE Custom Integrated Curcuits Conference (CICC'00)* (2000), pp. 507–510. doi:10.1109/CICC.2000.852719. 71, 114, 118

Chapter 8

Behavioral Model of TRNG Based on Oscillator Rings Implemented in FPGA[1]

Abstract

Understanding the behavior of a true random number generator (TRNG) is important in order to determine the security of such a design. In this paper, an investigation of a TRNG design based on several oscillator rings implemented in a field programmable gate array (FPGA) is performed in order to determine the amount of true randomness or entropy in this design. A model of the TRNG based on the number of hits in the transition regions generating entropy is proposed. This model takes into account the fact that the jitter of an oscillator ring is not constant, but increases or accumulates if the ring is not sampled in a region close to a transition point where the oscillator ring output changes from logic 0 to logic 1 or vice versa. The model of the TRNG is implemented in MatLab and simulations are performed showing the influence of different design parameters and also the influence of properties of the FPGA device on quality of randomness.

8.1 Introduction

A true random number generator (TRNG) is an important component in today's cryptographic systems. The security of such a system relies on the fact that the generated random bits from the TRNG are unpredictable and also that they have good statistical properties. In order to verify the TRNG output, statistical test suites such as NIST 800-22 [3] and DIEHARD [2] are used. These tests are not capable of distinguishing between true randomness and pseudo randomness. A restart experiment from an identical reset state can reveal whether the output sequences are different and therefore contain true randomness. However, such an experiment cannot quantify the amount of entropy or guarantee that every bit is a result of a random process. In order to verify the security and the presence of entropy, the behavior of the TRNG must be investigated theoretically in order to verify that the output bits are a result of physical random processes introducing entropy and not of deterministic processes.

A class of TRNGs based on XOR of the outputs from several equal length oscillator rings has experimentally been shown to have good statistical properties. This design was originally proposed by Sunar [4] where a post-processor is used to remove statistical defects at the output. An enhancement of this design by sampling each oscillator ring before the XOR was proposed by Wold et al. [6]. This enhancement improved the output sequence by removing the bias and no post-processing was needed in order to pass the statistical tests. The required number of oscillator rings was calculated by Sunar to be more than 100 based on a combinatorial approach using urns and the coupon collector's problem. The width of an urn is based on a proportion of the jitter compared to the ring period determined by the number of inverters in the ring and the routing delay in the FPGA. The

[1] Knut Wold and Slobodan Petrović. In Proceedings of the 14th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS'11), pp. 163–166, 2011.

urn width and thereby the number of urns are constants based on a given implementation[2]. The experimental results in [6] show that the enhanced TRNG passes the statistical tests with only 25 oscillator rings. It has been questioned whether this low number of rings is sufficient to achieve high entropy in every bit of a sequence [1].

In this paper, a model of the behavior of a TRNG based on several equal length oscillator rings [4, 6] is given explaining the amount of entropy in a more accurate manner than the model proposed by Sunar in [4]. Our model is based on the fact that the jitter is accumulated and that the uncertainty of the point in time where a transition occurs in the period of the oscillator ring is increased if the sampling is done outside the transition region. The result is that the standard deviation representing the region where entropy is achieved is increasing until a sampling or hit in this very region is performed restarting the accumulation of the jitter. Compared to the model by Sunar, the probability of hitting the transition region is higher and therefore a lower number of oscillator rings is needed in order to be sure that at least one of the rings is sampled in the transition region at each sample point. The simulations performed in MatLab show the influence of different design parameters and FPGA properties. The model is in principle based on equal length of all oscillator rings, but it can be extended and validated for different lengths as well.

The rest of the paper is organized as follows: In Section 8.2, our proposed model of a TRNG based on several oscillator rings is presented. In Section 8.3, simulations in MatLAB are performed showing the number of hits by varying different TRNG parameters. A conclusion is made in Section 8.4.

8.2 Proposed Model

A TRNG [6] based on several equal length oscillator rings, where the outputs of the rings are sampled and XORed is investigated, see Figure 8.1. The adjustable parameters in such a design are the following:

- Number of oscillator rings, R
- Sampling frequency, f_s
- Number of inverters in an oscillator ring, L.

[2]In [15, Ch.3], it has been pointed out that the urn model does not take into account the fact that jitter is accumulated in time.

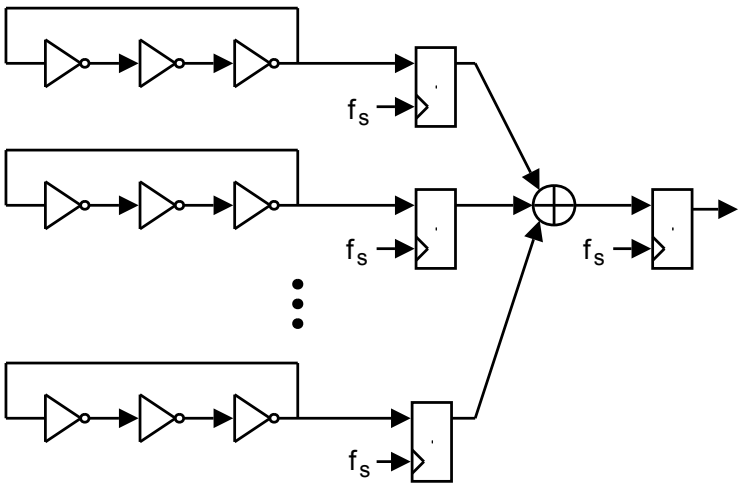

Figure 8.1: TRNG design.

In addition, there are two important properties of this TRNG determined by the technology of the FPGA, namely jitter and dispersion of oscillator ring frequencies. Jitter is the physical property responsible for creating true randomness in such a TRNG and can be looked upon as short-term variations of a digital signal's significant instants from their ideal position in time. It is characterized by a Gaussian distribution with zero mean and the standard deviation σ_j. The size of the jitter is determined by the properties of the hardware device and the circuit board. An oscillator ring consists of an odd number of inverters connected together in a chain where the output of the last inverter is fed back and connected to the input of the first inverter. The period of an oscillator ring is two times the sum of the time delay through all the logic elements implementing the inverters and the routing delay between the inverters making the ring. The periods or frequencies of the oscillator rings are therefore not identical due to the architecture of the FPGA, to the way the place and route algorithms are implemented in the software tools of the specific FPGA device family and to variations among the logic elements in the FPGA. In our model, the dispersion between the oscillator ring periods is characterized by the standard deviation σ_f of a Gaussian distribution, which is an approximation of the real distribution, see [6].

The Gaussian jitter of an oscillator ring is accumulated in the sense that the standard deviation of the jitter increases for each period of the oscillator ring. The size of the standard deviation of the accumulated jitter is proportional to the square root of the number of periods [5]. In a TRNG based on several oscillator rings true randomness is achieved by the uncertainty of the time instant of the transition of the oscillator ring related to the sampling point. In order to achieve high entropy in the sampled value, the sampling must be performed in a region defined by the standard deviation of the accumulated Gaussian jitter close to a transition in the oscillator ring. If the sampling is performed outside this region, the value can be regarded as deterministic given by the relationship between the sampling frequency and the oscillator ring frequency, and this sample does not contribute to the entropy of the TRNG. In order to sample enough entropy, at least one oscillator ring must be sampled in this region defined by the standard deviation of the accumulated jitter at each sampling point.

Our proposed model is implemented in MatLab operating in the time domain with a resolution of 1ps. The periods of the R oscillator rings are determined based on a Gaussian distribution with the mean set to the average of the measured periods of the FPGA and the standard deviation as one of the parameters in the model. The size of the jitter of one ring period is determined by the square root of the number of inverters L in the ring multiplied by the default jitter size σ_j. For each half-period of the ring, the ideal time position of a transition and the corresponding size of the accumulated jitter are calculated. If the sampling point is inside the interval defined by the accumulated jitter, a hit is registered and the accumulation of the jitter is restarted. In order to simulate the phase jitter, the position in time of the transition is randomly chosen after each hit based on the size of the accumulated jitter. For all the defined oscillator rings R, all the sampling points were investigated and hits were registered. All simulations of the rings started at the same point simulating a reset. The numbers of hits at the sampling points were then examined in order to verify that at least one hit was registered. The size of the transition region defining a hit was a model parameter, and it was chosen to be $\pm$ one standard deviation of the accumulated jitter. With this choice, the entropy sampled in a hit was estimated to be reasonably large, but the model would not register the smaller amount of entropy sampled outside this region but still inside the interval given by the accumulated jitter distribution. A special case occurs if the sampling is performed outside the defined high entropy region, but so close to it that a transition may have occurred with a probability higher than zero. In this case the accumulation of the jitter is not restarted but it can occasionally be eliminated. This effect occurs when one for instance samples a 1 in a position given by 3 times σ before an expected 0-1 transition. The position of the transition can then be determined with high probability and the entropy is therefore reduced. In our model, this is taken into account

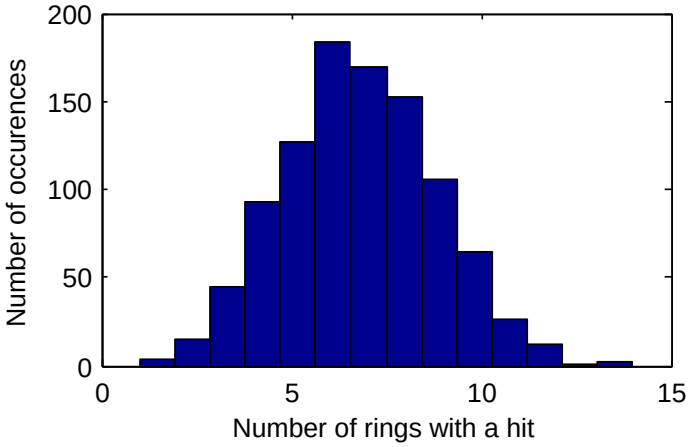

Figure 8.2: Histogram of hits with the default parameters.

by restarting the accumulation of the jitter using a rejection sampling technique with the probability given by the Gaussian distribution of the accumulated jitter.

8.3 Simulation

The proposed model of the behavior of a TRNG based on several oscillator rings presented in Section 8.2 was implemented in MatLab. The default values of the parameters were chosen to be similar to those used in the experiments in [6] where a TRNG implemented on an Altera Cyclone II FPGA with 25 oscillator rings of 3 inverters each and a sampling frequency of 100MHz was used. The size of the standard deviation of the jitter σ_j was set to 30ps (this is similar to figures used in [1] and [5]), and the dispersion of the periods of the oscillator rings σ_f was estimated to be 100ps. The default oscillator ring frequency was set to 413MHz. For all the simulations, a total of 1000 sampling points starting from a reset state were simulated. Figure 8.2 shows a histogram of the number of rings with hit in the transition regions. The simulation shows that at all sampling points there were one or more hits. On average, about 7 rings were sampled in the transition region at each sampling point. This simulation shows that the chosen parameters of the TRNG from [6] are reasonable to select in order to achieve true randomness and that every bit in the random sequence contains entropy.

In the sequel, each of the parameters of the model identified in Section 8.2 is varied in order to observe the effect on the number of hits plotted by using a probability density function based on the histograms.

Figure 8.3 shows the probability of the number of hits with different numbers of oscillator rings. The number of hits at a sampling point increases when the number of rings increases. It is observed that at a low number of rings, there are several sampling points that have no hits at all. A TRNG based on such a low number of rings will not produce true randomness for all bits.

Figure 8.4 shows the probability of number of hits with different sampling frequencies. The length of the simulation is adjusted such that the number of sampling points is identical for each run in order to compare the simulation results properly. It is observed that when the sampling frequency increases, the number of hits decreases. The reason for this is that with a low sampling frequency the oscillator rings complete more periods between the samples and more jitter is accumulated compared to a high sampling frequency.

Figure 8.5 shows the probability of the number of hits with different numbers of inverters in the oscillator rings. It is observed that the number of hits decreases with increasing

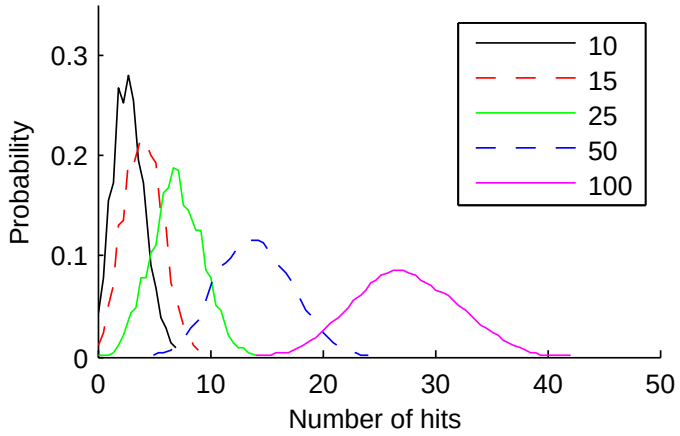

Figure 8.3: Density of hits with different numbers of rings.

number of inverters. This is a similar behavior compared to increasing the sampling frequency, except that the jitter increases due to the longer oscillator ring reducing the effect.

Figure 8.6 shows the probability of the number of hits with different size of the jitter. Increasing the size of the jitter leads to an increase in the probability of hitting the transitions. If the size of the jitter is low, a large number of the sampling points have no hits.

Figure 8.7 shows the probability of the number of hits with different size of the dispersion of the oscillator ring periods. It is observed that the number of hits is approximately constant with increasing dispersion between the oscillator ring frequencies. Figure 8.8 shows the mean of the number of hits with and without dispersion when the sampling period is varied related to the oscillator ring period. It is observed that without dispersion the number of hits is more dependent on the choice of the sampling frequency compared to the ring frequency. If the sampling period is a multiple of a half period of the rings, the number of hits is low and there is a higher risk of getting sample points without any hits at all. On the other hand, when there is a dispersion in the ring frequencies, the number of hits is almost uninfluenced by the relationship between the sampling frequency and the ring frequencies.

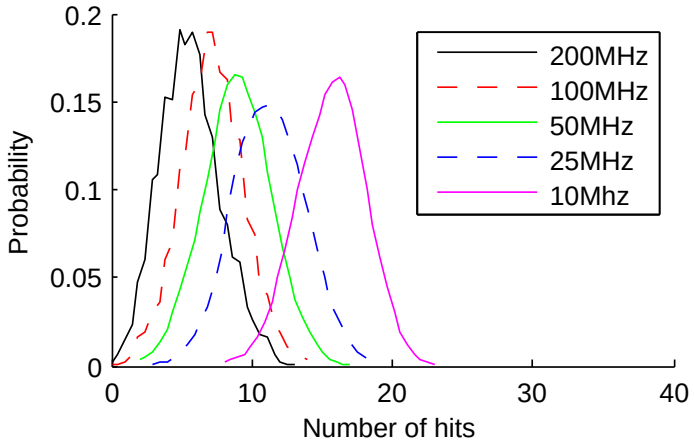

Figure 8.4: Density of hits with different sampling frequencies.

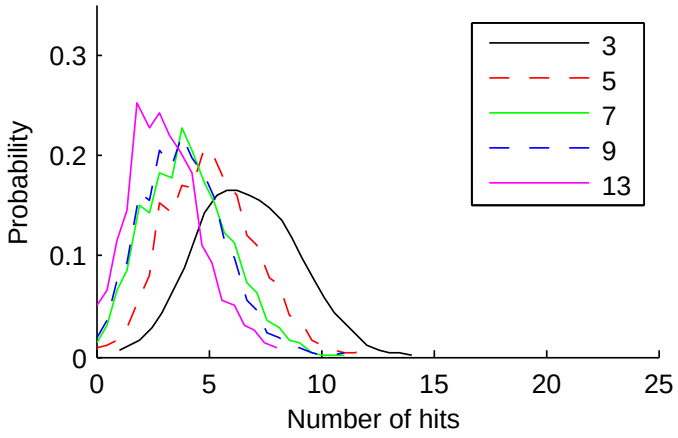

Figure 8.5: Density of hits with different number of inverters.

Figure 8.7 shows that the effect of the dispersion is not important regarding the number of hits, but it is important in order to achieve good randomness quickly after the reset. Figure 8.9 shows the histogram after a simulation of the first 50 sampling points after the reset with and without dispersion between the oscillator frequencies. It is observed that the diffusion of hits occurs much faster when there is a dispersion between the oscillator ring frequencies compared to the case when the accumulation of the jitter is the only contribution. The result is that the random sampled bits have high entropy much earlier after the reset. Another advantage of a dispersion in the frequencies deals with the security of the TRNG. It is known from the electronic science that there could be interactions between the rings. If the rings oscillate with identical frequencies and phases, the amount of entropy in the TRNG output will be drastically reduced. If there is a large dispersion between the frequencies, the risk of this interaction is much lower.

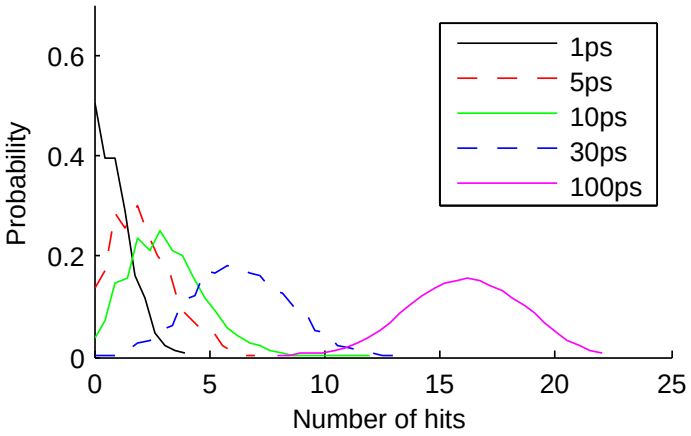

Figure 8.6: Density of hits with different size of jitter.

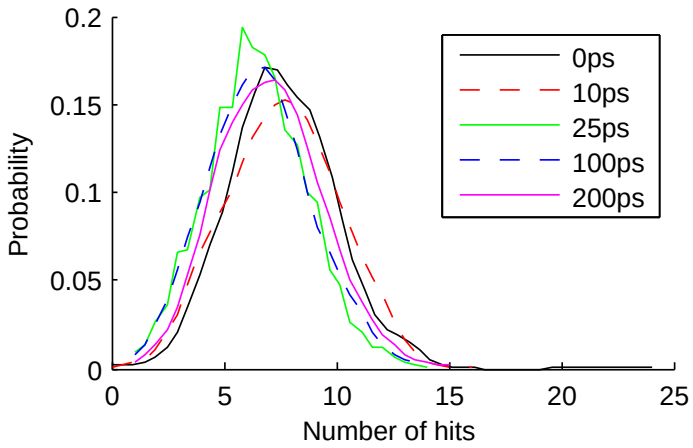

Figure 8.7: Density of hits with different dispersion in ring period.

8.4 Conclusion

In this paper, a new behavioral model of a class of TRNGs based on several oscillator rings is proposed. The model uses a statistical approach counting the number of hits in the transition region of an oscillator ring with high entropy. If there is at least one hit at all sampling points, the TRNG output will contain true randomness. Compared to the model of Sunar used in [4], our model is more accurate since it takes into account that the proportion of the jitter is not constant, but increases due to the accumulation as long as the samples are outside the transition region defining a hit. Our model uses both parameters of the TRNG design (number of rings, number of inverters in a ring and sampling frequency) and parameters related to the properties of the FPGA (jitter and dispersion of ring frequencies).

The model is implemented in MatLab where variations in the different parameters are investigated. The simulations show that among the design parameters the number of rings is the most important one. The model confirms that the experimental results and choice of parameters used in [6] are relevant in order to achieve a high probability of hitting at least one of the transition regions at every sampling point.

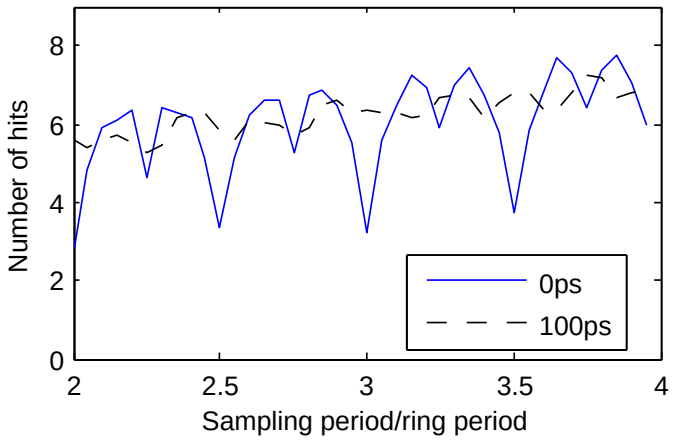

Figure 8.8: Mean of hits with and without dispersion in ring periods.

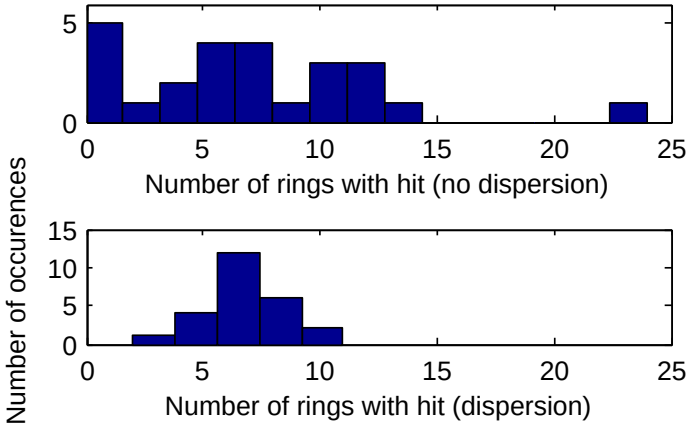

Figure 8.9: Histograms of hits with and without dispersion.

The model and the simulations revealed the contribution of the different parameters and properties of a TRNG based on several oscillator rings, and that all of them in different ways have an important role in generating true randomness. It is important to note that there will be differences among FPGA families in the implementation of such a TRNG.

Acknowledgement

The authors wish to thank Markus Dichtl for his useful comments and remarks, which improved the quality of this paper.

8.5 Bibliography

[1] BOCHARD, N., BERNARD, F., AND FISCHER, V. Observing the Randomness in RO-based TRNG. In *Proceedings of the 5th IEEE International Conference on ReConFigurable Computing and FPGAs (ReConFig'09)* (2009), pp. 237–242. doi:10.1109/ReConFig.2009.57. 59, 78, 80

[2] MARSAGLIA, G. DIEHARD: A Battery of Tests of Randomness. 1996. 7, 15, 38, 46, 47, 77, 103, 108

[3] NIST. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Special Publication 800-22 Revision 1a, April 2010. 7, 15, 37, 38, 46, 47, 65, 77, 103, 108

[4] SUNAR, B., MARTIN, W. J., AND STINSON, D. R. A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. *IEEE Transactions on Computers 56*, 1 (2007), 109–119. doi:10.1109/TC.2007.250627. 4, 11, 12, 14, 15, 19, 20, 21, 30, 31, 32, 34, 37, 39, 41, 45, 53, 58, 63, 77, 78, 83, 87, 101, 102, 103, 105, 106, 108, 109, 125, 126

[5] VALTCHANOV, B., AUBERT, A., BERNARD, F., AND FISCHER, V. Modeling and Observing the Jitter in Ring Oscillators Implemented in FPGAs. In *Proceedings of the 11th IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems (DDECS'08)* (2008), pp. 1–6. doi:10.1109/DDECS.2008.4538777. 9, 57, 58, 59, 60, 66, 67, 79, 80

[6] WOLD, K., AND TAN, C. H. Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings. In *Proceedings of the 4th IEEE International Conference on ReConFigurable Computing and FPGAs (ReConFig'08)* (2008), pp. 385–390. doi:10.1109/ReConFig.2008.17. 12, 14, 20, 42, 46, 47, 48, 52, 53, 54, 58, 64, 77, 78, 79, 80, 83, 87, 92, 99

Chapter 9

Security Properties of Oscillator Rings in True Random Number Generators[1]

Abstract

In order to achieve high security in a true random number generators (TRNG) consisting of several equal length oscillator rings implemented in a field programmable gate array (FPGA), it is important that the rings do not oscillate with identical frequencies and phases. In such a degenerate case, the TRNG output bit sequence would be regarded as deterministic and the entropy would be drastically reduced. In this paper, an investigation of the properties regarding interaction between the oscillator rings, the dispersion in the ring frequencies and correlation and dependencies among oscillator rings is carried out in three different FPGA device families in order to consider the security of such a TRNG design. The experiments show that there is an interaction between the oscillator rings located physically close to each other in an FPGA, which effects the frequency spectra of the oscillator rings. It is shown that the optimal number of inverters in an oscillator ring differs between FPGA technologies due to different architectures and different placement of the inverters in the logic elements in the FPGA. Investigation of several oscillator rings reveals that the outputs of a few of them are correlated due to almost identical ring frequencies, and also that the outputs of the rings cannot be regarded as independent. However, with a careful choice of parameters of such a TRNG it is possible to reduce the influence of these properties resulting in a secure TRNG with good statistical properties.

9.1 Introduction

Field programmable gate arrays (FPGA) are often used in today's cryptographic systems where a true random number generator (TRNG) is an important component. The jitter of an oscillator ring consisting of an odd number of inverters is typically used as the entropy source of such a TRNG [5, 6, 7]. It is important to understand the characteristics of oscillator rings implemented in different FPGA families in order to achieve high entropy and good statistical properties in the TRNG, and thereby high security of the cryptographic system.

There are several problems that might arise in the construction of a TRNG based on oscillator rings. Here, the following two mutually related issues are considered. First, in a TRNG based on several equal length oscillator rings [7] (see Figure 9.1), an unfortunate case might happen when all the oscillator rings oscillate with identical frequencies and phases. The result would be that the TRNG output would be deterministic and the amount of entropy would be drastically reduced. Second, it is well known that signals inside a digital integrated circuit can be influenced by each other. This disturbance can be caused by reduced signal quality of a net due to reflections and distortions, cross talk between several nets due to mutual capacitive and inductive coupling, rail collapse in the power distribution system and through electromagnetic interference from a component or a system [3]. A strong interaction between oscillator rings used in a TRNG can intimidate the security of the cryptographic system.

[1] Knut Wold and Slobodan Petrović. Submitted.

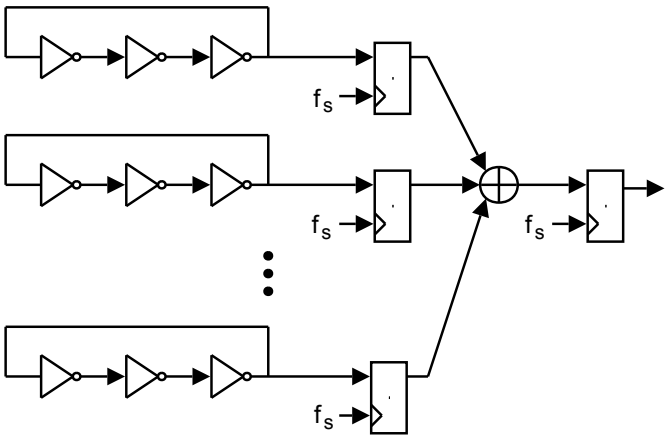

Figure 9.1: TRNG based on several oscillator rings.

In [9] an investigation of the interaction between oscillator rings implemented in an FPGA regarding the dependencies between the rings is performed. The phase jitter is extracted and analyzed by using a statistical approach, and the probability of phase interlock is examined. The conclusion from their experiments is that rings located close to each other suffer from a low-level correlation and that phase interlock is a problem only when the timing delays of the rings are closely matched leading to identical frequencies.

In this paper, spectral analysis of the interaction between oscillator rings in three different FPGA architectures is performed. The focus is on the influence on the frequency spectrum of one oscillator ring when another closely located ring is enabled. The experiments show that oscillator rings located close to each other are influenced by other rings in the sense that the harmonic frequency components of one ring are spotted in the frequency spectrum of the other ring. In order to reduce the risk with possible phase interlock between oscillator rings, it is important that the oscillator ring frequencies are different and have a large variation. The dispersion of the oscillator ring frequencies in different FPGA technologies have been examined, and that examination revealed that the optimal number of inverters regarding the highest dispersion varies depending on the architecture of the FPGA. The experiments show that the optimal number of inverters in an oscillator ring in order to achieve highest dispersion is different between the investigated SRAM based FPGAs from Altera and Xilinx compared to the investigated flash based FPGA from Actel. The experiments show that a few of the oscillator rings are correlated and that the rings cannot be regarded as independent. But by increasing the number of rings and by using an XOR function in the TRNG, the influence of the correlation and dependency can be reduced and removed.

The rest of this paper is organized as follows: In Section 9.2, the interaction between two oscillator rings implemented in FPGAs is examined by the use of spectral analysis. In Section 9.3, the distribution of oscillator ring frequencies in three different FPGA devices is investigated, and the FPGA architectures are examined in order to explain the observed behavior. In Section 9.4, the correlation and dependency between the outputs from several oscillator rings are investigated. In Section 9.5, a discussion of the experimental results regarding the security of a TRNG is performed, and finally, a conclusion is made in Section 9.6.

9.2 Interaction between Oscillator Rings

In order to examine the interaction between oscillator rings, experiments were performed on FPGAs from Altera (Cyclone II [2]), Xilinx (Spartan-3A [8]) and Actel (ProASIC3E [1]).

Two oscillator rings containing 15 inverters were implemented in the FPGAs and the two output signals were measured on I/O-pins with an oscilloscope capable of also measuring the frequency spectrum. After the place and route (P&R) process the inverters from the two oscillator rings were placed close to each other in the FPGA. Figure 9.2 shows the implementation of two oscillator rings of 15 inverters in a Cyclone II FPGA. In the architecture of the Altera FPGA, 16 logic elements are grouped into a logic array block (LAB) and one inverter occupies one logic element. It is observed that one of the rings was entirely placed in one LAB (blue logic elements), while the other was placed in two LABs (orange logic elements) where the vacant element in the first LAB was filled up with one of the inverters from the second ring. The result was that the routing delay through the second oscillator ring was longer than that of the first one, making the frequency of the second ring lower than the frequency of the first ring.

Ideally, the measured signal of the oscillator ring output in the time domain should be a square wave with a period determined by two times the delay of the inverter ring, and switching between 0V and V_{CC} (typically 3.3V) with a duty-cycle of 50%. In the frequency spectrum, the expected signal should be a peak at the main frequency (first harmonic) and then peaks at odd harmonics where the amplitude of the peaks decreases by a factor $\frac{1}{n}$ where n is the ordinal number of the odd harmonic. Due to the jitter produced in the electronic device, the ideal position of an oscillator ring output signal transition in time will have short-term variations. The result is that the frequency of the square wave will have small and fast variations around the theoretical frequency determined by the delay through the logic elements and the routing between the elements in the FPGA. In the frequency spectrum, the even harmonics are not zero due to none-ideal properties such as duty-cycle unequal to 50%.

The experiment was carried out in such a way that the rings could be disabled (one of the inverters was exchanged by a NAND-gate with an enable signal from an external switch) in order to observe the effect of the interaction. Figure 9.3 shows the frequency spec-

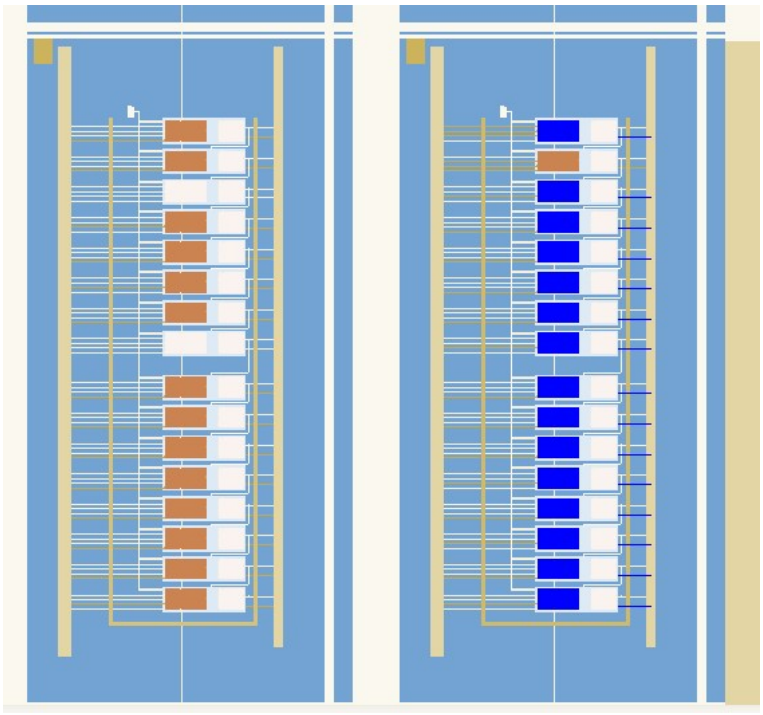

Figure 9.2: Placement of two oscillator rings with 15 inverters in an Altera CycloneII FPGA.

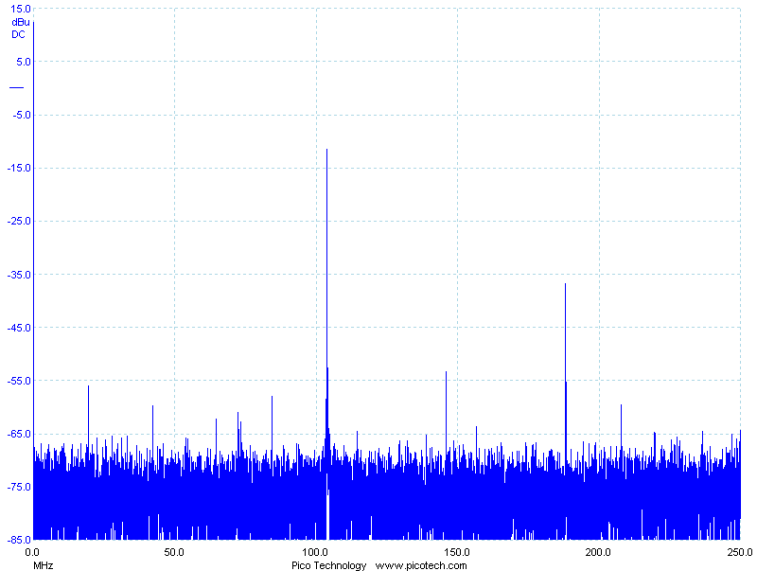

Figure 9.3: Frequency spectrum of one oscillator ring.

trum with just one ring. It is observed that the first harmonic is located at about 105MHz. Due to the bandwidth limitation of the fast Fourier Transform (FFT) in the oscilloscope to 250MHz, the peak of the third harmonics is convoluted and visible at about 185MHz in the plot (actual position is about 315MHz).

The second oscillator ring with a slightly lower frequency was then enabled. Figure 9.4 shows the frequency spectrum of the same ring as in Figure 9.3 but now with the influence of the second oscillator ring. It is observed that the first harmonic of the second ring is leaked into the frequency spectrum of the first ring, but the size of this peak is about 30dB lower than the peak of the first ring. The same is observed at the third harmonic frequency.

Figures 9.5 and 9.6 show a zoomed version of the frequency spectrum of Figure 9.4 of

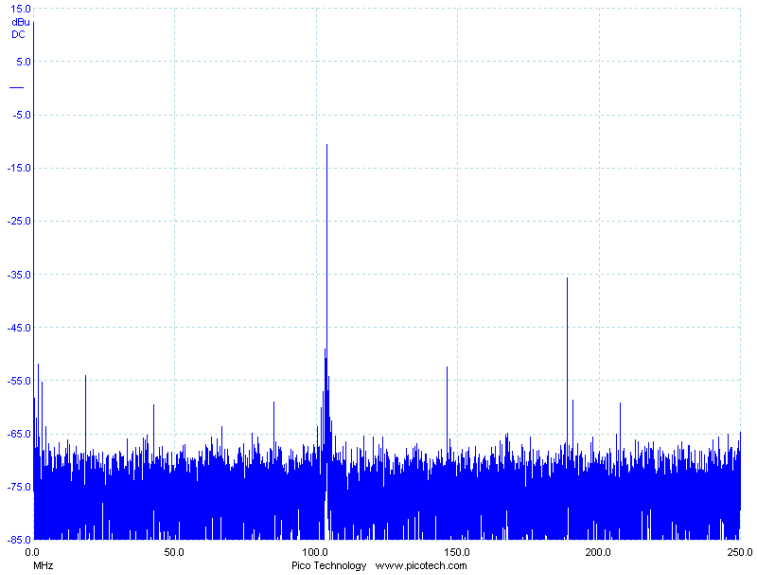

Figure 9.4: Frequency spectrum of one oscillator ring with another ring enabled.

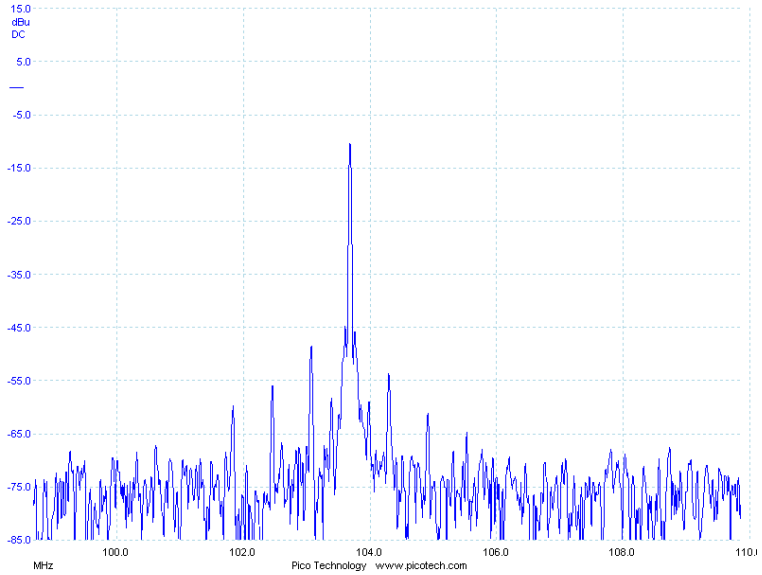

Figure 9.5: Frequency spectrum of one oscillator ring around the first harmonic with another ring enabled.

the regions around the first harmonic and around the DC-component. It is observed that several small peaks occur around the first harmonic and just above 0Hz. The difference between these peaks is equal to $n \cdot (f_1 - f_2)$ where f_1 and f_2 are the frequencies of the oscillator rings and n is an integer. The amplitude of these peaks is about 40dB lower than the peak of the first harmonic of the first ring.

When enabling more than two oscillator rings, several new smaller peaks were observed around the main frequency of the measured ring and at low frequencies just above the DC-component. These peaks are due to the interaction between the measured ring

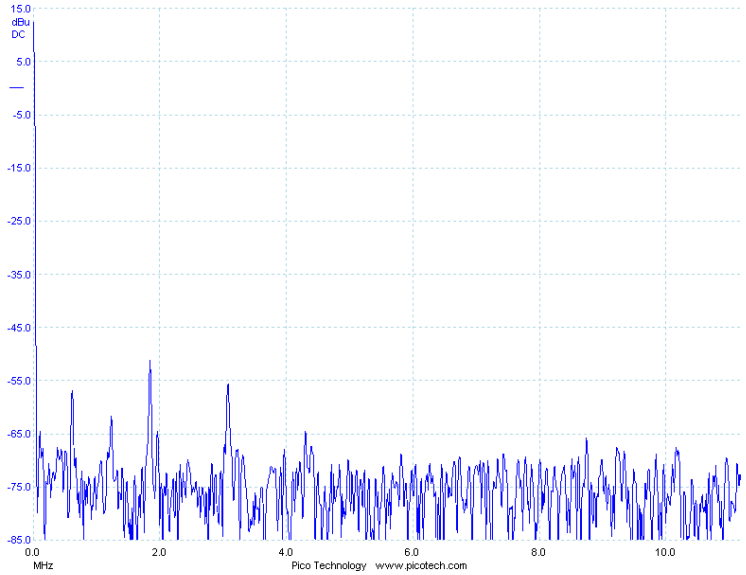

Figure 9.6: Frequency spectrum of one oscillator ring around the DC component with another ring enabled.

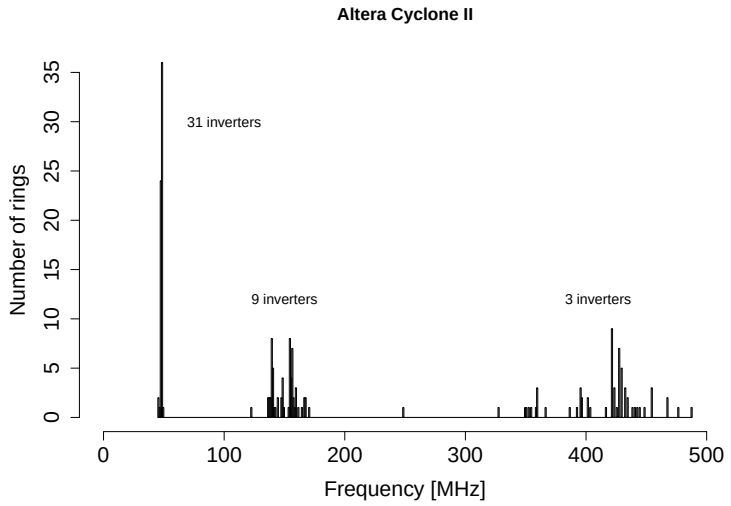

Figure 9.7: Histograms of oscillator frequencies of the Altera FPGA.

and all the other enabled rings, and the location in the frequency spectrum of these peaks are determined by the difference between the measured oscillator ring frequency and the frequencies of the other rings.

When monitoring one oscillator ring signal in the time domain of the oscilloscope and then enabling the second ring, no difference was visually observed on the measured signal on the oscilloscope. The effect of the interaction observed in the frequency spectrum was not detectable in the shape of the signal in the time domain.

The experiment was repeated for another SRAM based FPGA from Xilinx (Spartan-3A) and for a flash based FPGA from Actel (ProASIC3E). These two FPGAs behaved in a similar manner regarding interaction between the oscillator rings. This shows that there is no significant difference in the behavior of FPGA devices from different vendors, and that this interaction is more related to the technology and the electrical properties of an integrated circuit.

9.3 Variation in Oscillator Ring Frequencies

In a TRNG based on several oscillator rings, it is important to avoid that all the rings oscillate with identical frequencies. In that case, the output of the TRNG would be deterministic. An important security property in such a TRNG is therefore that the ring frequencies have certain dispersion. In [7], the dispersion of oscillator ring frequencies was investigated in a SRAM based Altera Cyclone II FPGA [2]. In this paper, we extend this investigation by looking also at two other FPGA devices: a SRAM based Xilinx Spartan-3A FPGA [8] and a flash based Actel ProASIC3E FPGA [1]. For all the three FPGA devices, a number of oscillator rings was implemented where each ring was built up with an odd number of inverters. The output of each oscillator ring was connected to an I/O-pin on the FPGA, and an oscilloscope was used to measure the frequency of the oscillator ring. For the Altera and Actel FPGAs 64 oscillator rings were used, while for the Xilinx FPGA only 50 oscillator rings were used due to the limitation in the available I/O-pins on the Xilinx evaluation board.

The number of inverters in the oscillator rings was increased starting at 3 inverters. For each configuration, all oscillator ring frequencies were measured. Figures 9.7-9.9 show the histograms of the measured frequencies of oscillator rings with lengths 3, 9 and 31 of the three FPGAs. In order to quantify the dispersion of the oscillator ring frequencies, the coef-

ficient of variance defined as the relation between the mean and the standard deviation, $\frac{\sigma}{\mu}$, was used. Figure 9.10 shows the dispersion as a function of the number of inverters in the oscillator rings for all the three FPGA devices. The highest dispersion of the SRAM based FPGAs is achieved by using 3 inverters, which is the lowest number of inverters needed to achieve oscillations. On the other hand, for the Actel FPGA the highest dispersion is achieved by 9 inverters in each oscillator ring. It is also observed that the level of dispersion is different between the Xilinx FPGA and the Altera FPGA, especially at low number of inverters where the dispersion is higher for the Xilinx device.

The explanation of the different behavior in the dispersion between the Actel on one side and the Altera and Xilinx on the other, can be found by examining the architecture of the FPGAs and investigating the placement of the inverters after the P&R process. The logic element of the Actel FPGA is called a tile and can either be a 3-input look-up-table (LUT) or a 1 bit register element (flip-flop or latch). The architecture is simpler than for the Altera and Xilinx devices, which typically consists of logic elements containing a LUT with at least 4 inputs and a register element. When the number of inverters is small in the Actel device, all the inverters are typically placed as close to each other as possible, see Figure 9.11. The result is almost identical routing delays of the rings and a small dispersion in frequencies. Figure 9.12 shows the difference in the placement of the inverters in the Actel FPGA for 2 oscillator rings containing 9 inverters each. The white squares indicate the tiles used in the two oscillator rings, where all the inverters of one ring are placed side-by-side (upper left corner), while the inverters of the other oscillator ring are spread around a larger area. The result is that one ring has a very short delay due to short routing between the inverters resulting in a high oscillator ring frequency, while the others have longer delay and thereby lower frequency. When the number of inverters increases even further, the placement of the inverters is more homogeneous resulting in a lower dispersion.

The architecture of the Xilinx Spartan-3A FPGA consists of configurable logic blocks (CLB) containing 4 slices where each slice is made up of two 4-input LUTs and two register elements. The result after the synthesis process is that one inverter in an oscillator ring occupies one LUT. After the placement process of rings consisting of three inverters, some of the rings are placed entirely inside the slices of one CLB, while the others are placed in slices in different CLBs. The result is a different routing delay of the rings caused by the shorter routing delay between the slices inside a CLB compared to the delay between the

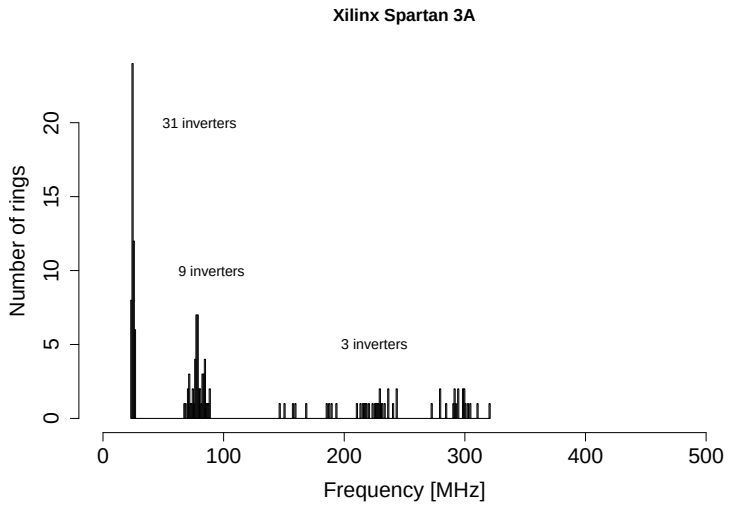

Figure 9.8: Histograms of oscillator frequencies of the Xilinx FPGA.

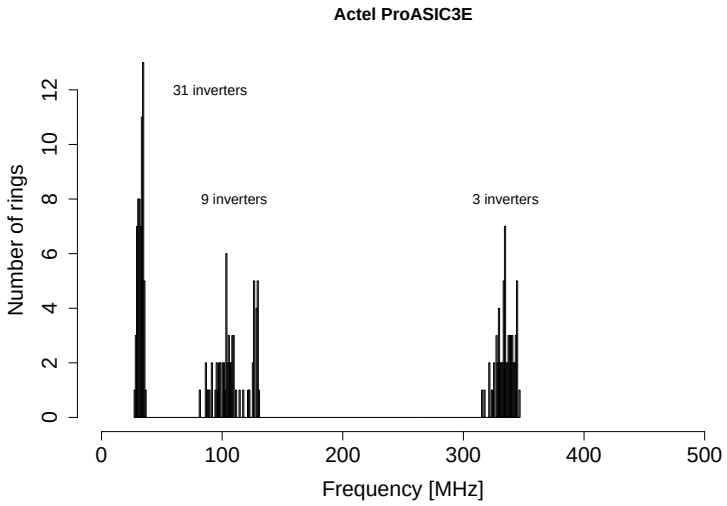

Figure 9.9: Histograms of oscillator frequencies of the Actel FPGA.

slices placed in different CLBs. Since the logic elements are placed in a regular array, the delays and thereby the frequencies of the oscillator rings are merged into groups as can be observed in Figure 9.8 for 3 inverters.

The same variation is observed in the Altera Cyclone II FPGA, where typically some of the rings are entirely placed inside one LAB, while other rings are placed in two different LABs. The result is a difference in the routing delay between the inverters in the oscillator ring, and thereby the variation in the frequencies of the oscillator rings.

In all these experiments, the FPGA devices contain only logic of the oscillator rings. In a real implementation, a digital design will contain additional logic from other parts of the design influencing on the placement of the inverters and thereby the resulting ring frequencies. Especially when the FPGA device is almost full, the placement could be different from these experiments.

Phase interlock could occur in two or more rings oscillating at almost identical frequencies, where the phase difference between the rings gradually reduces and finally becomes

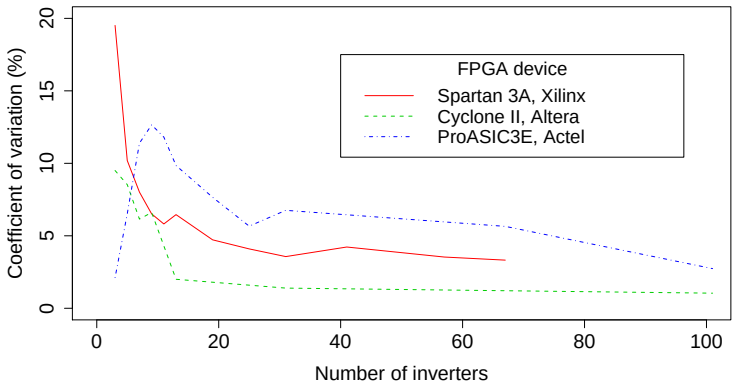

Figure 9.10: Dispersion of oscillator rings.

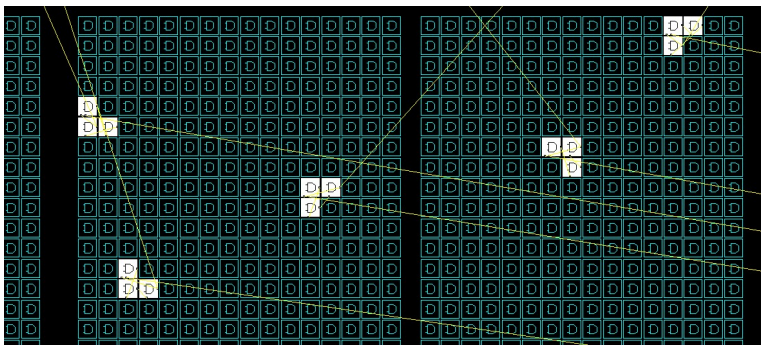

Figure 9.11: Placement of inverters in 5 oscillator rings with 3 inverters.

zero [9]. This phenomenon could occur if the time delays of the rings were finely tuned. In order to investigate the phase interlock in an FPGA, an experiment in an Altera Cyclone II FPGA was performed where two oscillator rings consisting of the optimal number of inverters (three) regarding maximum dispersion were implemented close to each other in the FPGA. The output of the two rings were sampled with an external clock of 50 MHz, and the sampled bits were connected to an XOR gate, see Figure 9.13. The advantage of this experimental set up is that sources of errors such as noise in oscilloscope measurements and noise influence from the board on the I/O pins on the FPGA signals are eliminated. In addition, the experiment is comparable to a real implementation of this kind of TRNG.

If the outputs from the oscillator rings are oscillating with the same frequency and phase, the XOR output should be logic zero over a longer period. The maximum number of subsequent zeros at the XOR output was counted and recorded, and the experiment was run for several hours. The result was that the maximum number of equal subsequent sampled values from the two rings was 6. This shows that even if the two rings were placed close to each other in the FPGA device, the interaction did not result in phase interlock under normal operation conditions. What is not tested in this experiment is what happens if an attacker manipulates the environment or for instance the power supply to the FPGA in order to try to force the oscillator rings to oscillate with identical frequency and phase. With an optimal choice of the length of the oscillator rings regarding dispersion in the frequencies, it is assumed that this attack is not an easy task.

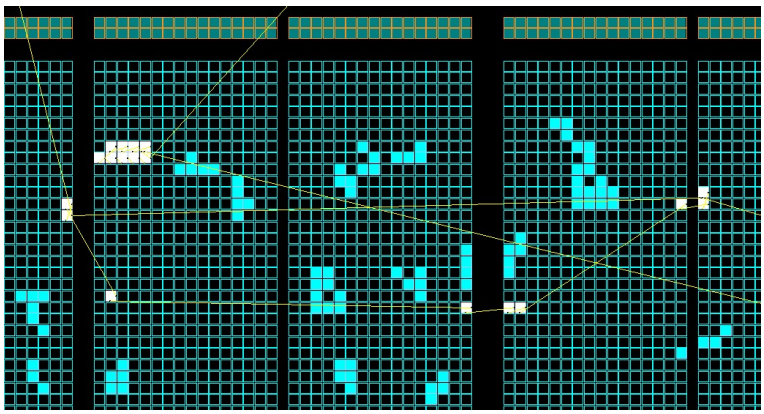

Figure 9.12: Placement of inverters in 2 oscillator rings with 9 inverters.

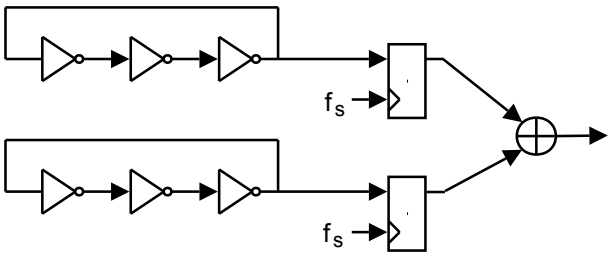

Figure 9.13: Experimental setup for studying phase interlock.

9.4 Correlation between Oscillator Rings

The experiments have shown that there is some interaction between the oscillator rings implemented in an FPGA when investigating the frequency spectra of two rings. But the question is whether this interaction results in a correlation or dependency between the oscillator rings. The correlation $\rho \in \{-1, 1\}$ between two sequences is defined by:

$$\rho(x, y) = \frac{Cov(x, y)}{\sqrt{Var(x) \cdot Var(y)}} \tag{9.1}$$

and an estimator $\hat{r}$ of the correlation between $x[n]$ and $y[n]$ with N elements is given as:

$$\hat{r} = \frac{\sum_{i=0}^{N-1}(x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum_{i=0}^{N-1}(x_i - \bar{x})^2 \cdot \sum_{i=0}^{N-1}(y_i - \bar{y})^2}} \tag{9.2}$$

where $\bar{x}$ and $\bar{y}$ are the average of the bit values in the sequences.

In a TRNG with several equal length oscillator rings where each ring is sampled, the sampled bit values from one ring is dependent on the frequency of the ring, f_i, and the sampling frequency, f_s. The frequency of the sampled signal or beat frequency, f_b, is related to the difference between the ring frequency and the sampling frequency, but it is always less than half the sampling frequency. In the case where the ring frequency is greater than the sampling frequency, the beat frequency is given by:

$$f_b = |f_i - n \cdot f_s| \tag{9.3}$$

where n is an integer chosen so that $f_b < \frac{f_s}{2}$. In the case where the oscillator ring period is exactly equal to $f_i = n \cdot f_s$ the resulting beat frequency will be zero and the sampled bit sequence will consist of only zeros. In the ideal case, the bit sequence generated by the sampling of an oscillator ring will have alternating blocks of zeros and ones depending on the beat frequency related to the sampling frequency. The result is also that investigating a bit sequence generated by sampling the output of an oscillator ring for a period much longer than the beat period, $t_b = \frac{1}{f_b}$, the mean will be close to 0.5.

In the ideal case the beat frequency is constant, but in a real implementation of an oscillator ring in a programmable device, there are three factors that alter this assumption:

- Gaussian jitter or electronic noise and the accumulation of this jitter will make the exact position of a transition of the oscillator ring fluctuate.
- Variation in temperature on the device due to self heating or from the environment will alter the ring frequency.
- Variation in the power supply voltage of the programmable device due to alternating power consumption on the device and the circuit board will alter the ring frequency.

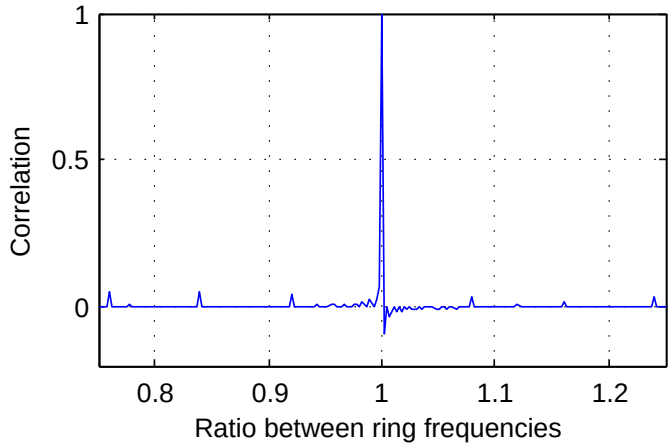

Figure 9.14: Simulation of correlation of two rings with frequencies close to each other.

In order to investigate the correlation between the rings in a TRNG based on several equal length oscillator rings, an experiment was performed where 32 oscillator rings, each consisting of 3 inverters, were implemented in all the three FPGA devices mentioned above. No constraints were put on the placement of the inverters. The outputs of all these rings were sampled with an external sampling frequency of 50MHz for the Altera and Xilinx devices and 40MHz for the Actel device. The difference in the sampling frequency is due to the available external clock oscillators on the used FPGA evaluation boards. A total of 4096 consecutive samples of 32 bit words were generated and stored as one block in an internal RAM. These blocks of data were then transfered to a PC for further analysis.

The correlation based on Equation (9.2) was calculated between all combinations of the 32 rings for each of the three FPGAs. The result is shown in Table 9.1 where the correlation coefficients between all the 496 combinations of the 32 rings are sorted after the level of ρ. It is observed that the majority of the rings for all the FPGA devices are uncorrelated (i.e. $\rho \approx 0$). But, among the rings there are a few that were highly correlated with each other. For instance, the maximum correlation in the case of the Altera FPGA was a correlation coefficient of 0.73. In this case, the implementation of these two oscillator rings in the FPGA device had resulted in almost identical delay through the loop and thereby identical ring frequency. The result after the XOR of two correlated rings is a sequence consisting of almost only zeros, and these rings will not contribute to the randomness regarding entropy.

A simulation in Matlab was performed in order to investigate the correlation between two sampled oscillator rings where the ratio between the oscillator ring frequencies was varied. Figure 9.14 shows the correlation when the ratio is close to 1 when investigating 4096 bits. The simulation shows that there is a strong correlation when the frequencies are identical, but the correlation drops very fast when there is a deviation between the ring frequencies.

Even though there is almost no correlation among the sampled outputs of the oscil-

FPGA	$\rho \in$ (-1,-0.1)	$\rho \in$ (-0.1,-0.01)	$\lvert\rho\rvert <$ 0.01	$\rho \in$ (0.01,0.1)	$\rho \in$ (0.1,1)
Altera	1	80	340	74	1
Xilinx	0	60	348	86	2
Actel	2	63	363	68	0

Table 9.1: Correlation between oscillator rings in FPGA devices.

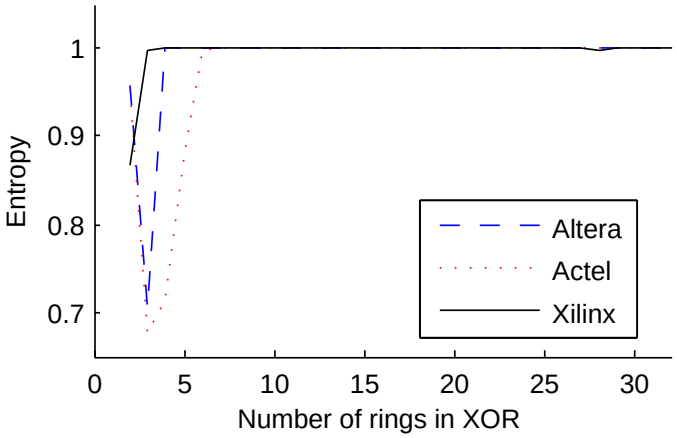

Figure 9.15: Calculated entropy based on an estimator of probability.

lator rings, dependencies could exist. The dependencies among the oscillator rings were investigated by performing a statistical test. The 32 rings were combined in eight groups of 4 rings each and the number of occurrences of 4 bit combinations were counted in all sample points. For each block of 4096 consecutive sampled bits from the rings, a χ^2 statistical test for independence was performed where the null hypothesis was that the rings were independent. The significance level of the test was set to 0.05. The result was that the rings in the Altera FPGA were not independent. However, when decreasing the sampling frequency from 50MHz to 25MHz, a majority of the blocks were independent. In the case of the Actel FPGA, data from some of the blocks was regarded as independent when the number of inverters was nine, while with three inverters the rings were dependent. For the Xilinx FPGA, a majority of the blocks were independent.

The entropy related to information theory is a measure of the uncertainty of the outcome of the next generated bit value. An estimator of the entropy is given by

$$\hat{H} = -[\hat{p} \cdot log_2(\hat{p}) + (1 - \hat{p}) \cdot log_2(1 - \hat{p})] \tag{9.4}$$

where $\hat{p}$ is the probability of generating a zero. The number of transitions could be used as a simple estimator of probability [4]

$$\hat{p} = \frac{N_{trans}}{N_{tot} - 1} = \frac{1}{N_{tot} - 1} \sum_{i=0}^{N_{tot}-1} (z[i] \oplus z[i+1]) \tag{9.5}$$

where $z[i]$ is the generated random sequence after XOR of the outputs of M oscillator rings, N_{trans} is the number of transitions and N_{tot} is the total number of bits in the random sequence. With $\hat{p}$ close to 0.5, the entropy $\hat{H}$ is close to 1.0, which is the ideal case. From the captured data from the experiments with the three different FPGAs, the numbers of transitions were calculated for increasing number of oscillator rings included in the XOR and the entropy was calculated as well. Figure 9.15 shows the entropy for the three different FPGAs. It is observed that the entropy quickly approaches the ideal level of 1.0 for all the three FPGAs.

9.5 Discussion

When measuring one ring in the time domain on the oscilloscope, almost no difference was observed when enabling the second ring. If the oscillator rings are used in a TRNG

(Figure 9.1), the important point is the uncertainty in the position of the transition in time. Variations in the amplitude are of no importance as long as it is less than the switching level of the CMOS, which is typically about 2V. Therefore, TRNG based on several oscillator rings will not be influenced by this interaction, because its behavior is dependent on the time of a transition in the oscillator ring and not on the amplitude of the signal.

In an ordinary clocked digital design, a low level of interaction between signals is expected because signal integrity is a prioritized area of the FPGA vendors. If a digital signal is so much influenced by other signals that the level of the signal is altered (a logical one becomes a logical zero or vice versa), a digital design configured in an FPGA would not work properly. An oscillator ring is a special kind of circuit because it forms an uncontrollable free running combinational loop and this structure is not recommended in a digital design because it is asynchronous. In a TRNG, these structures create randomness. Due to the nature of this oscillator ring signal, there could be a higher risk of interaction between these signals, but the experiments show little influence between the rings. In order to achieve good signal integrity, the FPGA board layout should have proper decoupling of the power supply to the FPGA device, usage of power and earth planes and short connections. By following these guidelines, the interaction between signals will be reduced.

The performed experiments show that some of the oscillator rings are correlated. The result for a TRNG based on XOR of several oscillator rings, is that correlated rings will not contribute to randomness. In order to avoid a security problem, it makes sense to slightly increase the number of oscillator rings to compensate for the possibility of correlation between the rings.

The outputs from the sampled oscillator rings implemented in FPGAs cannot be regarded as independent from each other, but the properties can be improved by decreasing the sampling frequency and by choosing the optimal number of inverters in the ring. On a general basis, long time delay between every generated bit from a TRNG will improve the quality of the random signal and increase the entropy at the cost of a lower bit rate. Performing XOR of all the oscillator rings in a TRNG is a method of achieving this. The purpose of the XOR circuit is to harvest the randomness from the oscillator rings, but it can also be regarded as a post-processor improving the statistical properties of the TRNG output. The estimate of entropy based on the number of transitions shows that XORing several rings quickly makes the entropy converging to 1 which is equal to the entropy of an ideal random sequence.

9.6 Conclusion

Oscillator rings represent a commonly used hardware structure exploiting sources of entropy in TRNGs implemented in programmable devices. In this paper, properties of oscillator rings implemented in FPGAs regarding interactions, frequency dispersion, correlation and dependency are examined. Three different FPGA devices, Xilinx Spartan-3A, Altera Cyclone II and Actel ProASIC3E, have been investigated in order to examine the influence on the security of a TRNG consisting of several oscillator rings.

The interaction between two oscillator rings implemented close to each other in an FPGA is investigated. The experiments show that there are small interactions from other rings when measuring the frequency spectrum of one ring. The interaction is caused by capacitive and inductive couplings on the device between the logic elements implementing the rings. In the time domain, this interaction is not visible on the oscilloscope. Regarding the security and the manner of operation of a TRNG consisting of several oscillator rings [7], this interaction is not critical because it does not change the timing conditions of the transitions of the oscillating signal.

The number of inverters in the oscillator rings giving the highest dispersion of the frequencies differs between the FPGA technologies. In the examined SRAM based FPGA devices from Altera and Xilinx, the recommended number is the lowest possible, namely

3. For the flash based FPGA from Actel, the best result is achieved when the number of inverters is set to 9. This behavior can be explained by investigating the placement of the inverters after the P&R process. A large variation between the routing lengths of the rings explains the variation in the oscillator frequencies.

The experiments have shown that a few oscillator rings have correlation while the majority are regarded as uncorrelated. The influence of this effect on the security of a TRNG based on several rings is that the number of rings should be slightly increased in order to reduce this correlation. The oscillator rings cannot be regarded as independent, but the usage of the XOR of several rings removes this dependency.

9.7 Bibliography

[1] ACTEL. ProASIC3E Flash Family FPGAs. August 2009. 88, 92

[2] ALTERA. Cyclone II Device Handbook, Volume 1. Tech. rep., Altera Corporation, Feb. 2008. 88, 92, 102, 107

[3] BOGATIN, E. *Signal and Power Integrity - Simplified, Second Edition*. Prentice Hall, 2010. 17, 87

[4] BUCCI, M., AND LUZZI, R. Design of Testable Random Bit Generators. In *Proceedings of the 7th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'05)* (2005), vol. 3659 of *Lecture Notes in Computer Science*, Springer, pp. 147–156. doi:10.1007/11545262_11. 5, 7, 30, 98, 102

[5] KOHLBRENNER, P., AND GAJ, K. An Embedded True Random Number Generator for FPGAs. In *Proceedings of the ACM/SIGDA 12th International Symposium on Field Programmable Gate Arrays (FPGA'04)* (2004), ACM, pp. 71–78. doi:10.1145/968280.968292. 10, 11, 14, 30, 45, 87, 102

[6] SUNAR, B., MARTIN, W. J., AND STINSON, D. R. A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. *IEEE Transactions on Computers 56*, 1 (2007), 109–119. doi:10.1109/TC.2007.250627. 4, 11, 12, 14, 15, 19, 20, 21, 30, 31, 32, 34, 37, 39, 41, 45, 53, 58, 63, 77, 78, 83, 87, 101, 102, 103, 105, 106, 108, 109, 125, 126

[7] WOLD, K., AND TAN, C. H. Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings. In *Proceedings of the 4th IEEE International Conference on ReConFigurable Computing and FPGAs (ReConFig'08)* (2008), pp. 385–390. doi:10.1109/ReConFig.2008.17. 12, 14, 20, 42, 46, 47, 48, 52, 53, 54, 58, 64, 77, 78, 79, 80, 83, 87, 92, 99

[8] XILINX. Spartan-3 Generation FPGA User Guide. August 2010. 88, 92

[9] YOO, S.-K., KARAKOYUNLU, D., BIRAND, B., AND SUNAR, B. Improving the Robustness of Ring Oscillator TRNGs. *ACM Transactions on Reconfigurable Technology and Systems 3*, 2 (May 2010), 1–30. Article 9. doi:10.1145/1754386.1754390. 17, 59, 60, 67, 88, 95

Appendix A

Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings[1]

Abstract

A true random number generator (TRNG) is an important component in cryptographic systems. Designing a fast and secure TRNG in an FPGA is a challenging task. In this paper we analyze the TRNG designed by Sunar et al. [12] based on XOR of the outputs of many oscillator rings. We propose an enhanced TRNG that does not require post-processing to pass statistical tests and with better randomness characteristics on the output. We have shown by experiment that the frequencies of the equal length oscillator rings in the TRNG are not identical but different due to the placement of the inverters in the FPGA. We have implemented our proposed TRNG in an Altera Cyclone II FPGA. Our implementation has passed the NIST and DIEHARD statistical tests with a throughput of 100Mbps and with a usage of less than 100 logic elements in the FPGA.

A.1 Introduction

Traditionally, a high assurance implementation of cryptographic algorithms has been done in application specific integrated circuit (ASIC). During the recent years, more and more of these implementations are done in field programmable gate array (FPGA). There are several reasons for this development. The FPGA can be reprogrammed, leading to more flexibility for modification of algorithms, changing algorithms and fixing bugs. The development of an algorithm in an FPGA is easier and faster as compared to an ASIC design, resulting in a shorter time-to-market. In addition, the latest FPGA devices are manufactured with the state-of-the-art technology.

It is well known that a true random number generator (TRNG) is an important component in today's cryptographic systems. Typically a TRNG can be used for generating keys, initial vectors, random sequence for cryptographic challenges, etc. In a cryptographic system, a private or secret parameter is normally generated by a TRNG and is an interesting property to an attacker. Therefore, the generation of a random bit sequence is important and should be unpredictable to an attacker. One common method for generating a truly random sequence is to amplify the thermal noise in a diode [7]. The disadvantage of this method is the use of external components. This approach enables an attacker to manipulate and read the random bit sequence from the device and consequently violating the security of the entire cryptographic system. If the TRNG is implemented entirely inside the FPGA, an attacker will have difficulties in retrieving and manipulating the random bit sequence. The challenge is to design a TRNG in an FPGA passing all statistical tests and at the same time using as few resources as possible and achieving a high throughput of random bits.

In this paper we will examine more closely the TRNG based on oscillator rings proposed by Sunar et al. [12]. We found that their TRNG is not random without post-processing. We propose an enhancement of the proposal in [12] and show experimentally improved

[1] Knut Wold and Chik How Tan. In Proceedings of the 4th IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig'08), pp. 385–390, 2008.

performance with respect to FPGA resource usage and throughput. We also show that our TRNG has no bias and therefore no need for complicated post-processing. We have shown by experiment that the frequencies of the oscillator rings will be different due to the placement and routing of the inverters inside the FPGA. We have implemented our proposal in an Altera Cyclone II FPGA [1]. Our implementation of the TRNG based on ring oscillators passes the NIST and DIEHARD statistical tests with a throughput of 100Mbps and the usage of less than 100 logic elements in the FPGA.

The rest of this paper is organized as follows: In Section A.2, we briefly examine the previous work on TRNG in FPGA. In Section A.3, we analyse the TRNG of [12]. In Section A.4 we propose an enhancement of the TRNG to achieve better randomness on the output sequence. The detailed analysis of the randomness of our proposed TRNG and the investigation of distribution of frequencies on oscillator rings are discussed in Section A.5 and A.6 respectively. In Section A.7, we describe in detail an implementation of our proposed TRNG and finally we make a conclusion in Section A.8.

A.2 Related Work

Several implementations of TRNG in FPGA have been proposed during the recent years. The common entropy source used is jitter on clock signals. Jitter can be viewed as timing deviation from the theoretical correct position due to electronic or thermal noise [13]. The random jitter will typically follow a Gaussian distribution characterized by a certain standard deviation (σ). Usually, jitter is an unwanted property in a system, but this behavior is useful when generating random signals in a TRNG.

In 2002, Fisher et al. [5] used the jitter in analogue phase-locked loop (PLL) in FPGAs from Altera as entropy source in a TRNG. The strategy is to create different clock signals with jitter from the PLL and sample one of the clock signals with the other. This method is restricted to FPGAs containing such analogue components. Later, Kohlbrenner et al. [8] used a similar technique, but the clocks are generated by oscillator rings containing two transparent latches, a buffer and an inverter. Since the frequencies of the two oscillator rings have to be almost equal, the oscillator rings have to be correctly matched. Tkacik [14] proposed a TRNG using a linear feedback shift register (LFSR) and a cellular automaton shift register (CASR) clocked by two independent oscillator rings. Selected outputs from the LFSR and CASR are combined by an XOR generating the final random signal. The disadvantage of this scheme is that the TRNG has memory and is therefore not stateless as pointed out in [2]. In 2006, Golić [6] proposed a TRNG using a Galois ring oscillator (GARO) and a Fibonacci ring oscillator (FIRO). These LFSR structures use inverters as delay elements instead of register elements. The outputs from one GARO and one FIRO are combined with an XOR and the random sequence is generated by sampling with a D flip-flop. This design was further investigated by Dichtl et al. [4] who found some minor problems regarding cross-talk from other signals inside the FPGA. In 2007, Sunar et al. [12] gave a theoretically proposal for a random number generator based on several equal length oscillator rings made up of an odd number of inverters (refer to Figure A.1). The outputs from the oscillator rings are XORed together and sampled with a D flip-flop. To correct for unbalances between zeros and ones in the random signal, a post-processing is carried out on the output of the D flip-flop. Schellenkens et al. [11] implemented this scheme in a Xilinx FPGA, but with a large number of rings in order to make the random sequence output pass the statistical tests.

A.3 TRNG Based on Oscillator Rings

Figure A.1 shows the TRNG proposed by Sunar et al. [12]. The TRNG is constructed from many equal length oscillators connected to an XOR tree. The output from the XOR tree is sampled by a D flip-flop. In order to satisfactorily pass the statistical tests like NIST

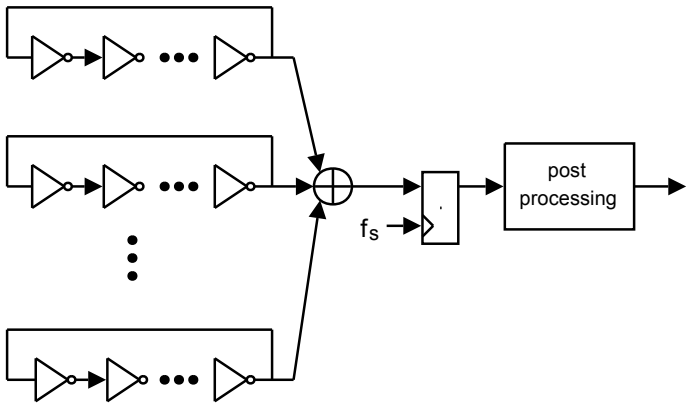

Figure A.1: TRNG based on oscillator rings

[10] and DIEHARD [9], the random signal from the D flip-flop must be post-processed. The proposed post-processing is a resilient function implemented as a BCH-code. The suggested design of the TRNG consists of 114 oscillator rings where each ring consists of 13 inverters. A suggested sampling frequency is 40MHz and the post-processing is a [256, 16, 113] extended BCH code. The resulting throughput of the TRNG in [12] is 2.5Mbps.

The entropy source of the TRNG is the jitter created by each oscillator ring. The jitter has a Gaussian distribution around each clock transition between logic low and logic high level. This jitter will create an accumulated phase drift in each ring so that the transitions will be at different times in the sampling period. Due to the jitter, the unpredictable transition region is assumed to be uniformly distributed in the sampling period. The number of rings needed can then be calculated based on the coupon collector's problem, that is, the number of uniformly random selections of N urns such that all urns are selected at least once. The number of urns is determined by the proportion of the jitter size as compared to the frequency of the oscillator ring. Because the number of rings grow exponentially when filling up the last urns, a lower fill rate than 100% is selected. To compensate for this, a BCH-code is used for post-processing. The resulting random number throughput is reduced by a factor of 16 due to this post-processing scheme.

In [4], some weaknesses of this implementation were mentioned. The main concern of the authors is that the XOR-tree and the sampling D flip-flop cannot handle the high number of transitions from the oscillator rings. The frequency of a oscillator ring is about the same or higher than the sampling frequency. With many oscillator rings in parallel, the number of transitions during a sampling period will be so high that the setup- and hold-times for the look-up table (LUT) and the internal register element in the FPGA will be shorter than specified for the device. The detailed analysis of the TRNG of [12] is shown in Section A.5 and A.6 as it is better to have a comparison with the proposed enhanced TRNG.

A.4 Our Proposed Enhancement

To cope with the problem with many transitions in the sampling period, we suggest an enhancement to the TRNG based on the oscillator rings in [12] by adding an extra D flip-flop after each ring (refer to Figure A.2). As we will show, this configuration will improve the randomness of the TRNG. The randomness of the configuration relies on the jitter variations of the oscillator rings. Adding these extra flip-flops will not alter the collection of the randomness of each ring, but improve the overall output of randomness.

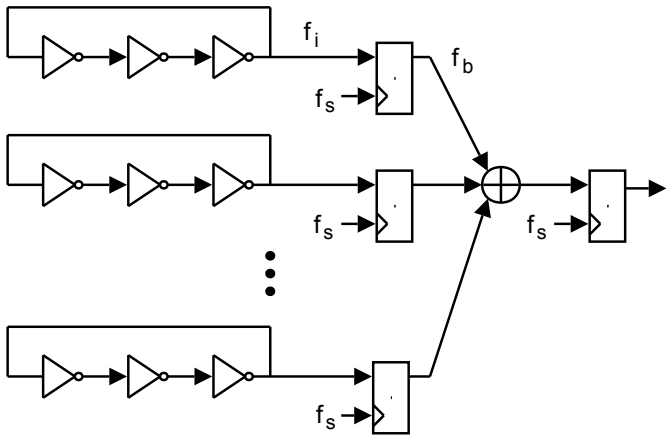

Figure A.2: Our proposal

The frequency of the oscillator ring (f_i) is dependent of the odd number of inverters in the ring. The frequency will increase with the decreasing number of inverters. In order to have a fast and small TRNG, the number of inverters should be as low as possible making the frequency of the rings become high as compared to the sampling frequency (f_s). The advantage of our enhancement is that the signals on the input of the XOR will now be synchronous with the sampling clock and only updated once in the sampling period. Due to this reduction in transitions on the input to the XOR tree, the setup- and hold-times for the internal logic in the FPGA will now be within acceptable limits.

The frequency of the beat signal (f_b) after the extra flip-flop will always be less than half of the sampling frequency and lie in the interval $[0, \frac{f_s}{2}]$ (refer to Figure A.3). The sampling frequency f_s should be chosen such that the beat frequency f_b at the input of the XOR is as high as possible and avoid that the frequency of the oscillator rings is a multiple of the sampling frequency.

A sampling of a free running oscillator ring could lead to metastability in the flip-flop

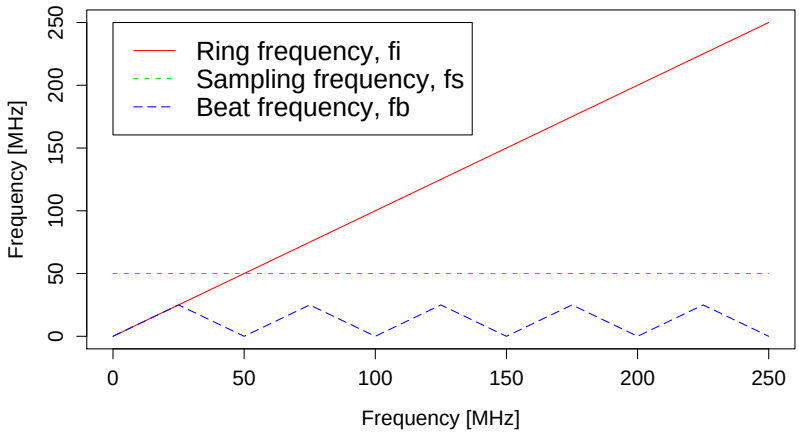

Figure A.3: Beat frequency

causing the output of the flip-flop neither to be logic low or logic high. This phenomenon can arise when a transition occur during the set-up and hold-time for the flip-flop. To avoid this unwanted state propagating into the XOR tree, the output of the oscillator ring could be sampled by one additional D flip-flop.

A.5 Bias in TRNG

One of the basic statistical tests of random number generators is the frequency test of ones and zeros. For a good random bit sequence the probability of a zero or a one should be close to $\frac{1}{2}$. In other words, there should be no bias in the random bit sequence.

Let X and Y be two random bit sources with their expected values $E(X) = E(Y) = \mu$ respectively and let ρ be their correlation. Then the expected value or bias of the XOR of the two sequences $(X \oplus Y)$ is given by [3]:

$$E(X \oplus Y) = \frac{1}{2} - 2\left(\mu - \frac{1}{2}\right)^2 - 2\rho\mu\,(1-\mu) \tag{A.1}$$

If μ is close to $\frac{1}{2}$, Equation (A.1) can be written as:

$$E(X \oplus Y) \approx \frac{1}{2}\,(1-\rho) \tag{A.2}$$

It can be seen that a correlation between the two sequences will generate a bias in the output from the XOR of two random bit sequences. If X and Y are linearly independent, then $\rho = 0$ and $E(X \oplus Y) \approx \frac{1}{2}$.

If there are n independent bits, each with expected value μ, then the expected value of XOR of all these bits will be given by [3]:

$$\frac{1}{2} + (-2)^{n-1}\left(\mu - \frac{1}{2}\right)^n = \frac{1}{2}\left[1 + (-2\varepsilon)^n\right] \tag{A.3}$$

where $\varepsilon = \mu - \frac{1}{2}$. Since $\mu \in (0,1) \Rightarrow |2\varepsilon| < 1$, the expected value in Equation (A.3) will converge to $\frac{1}{2}$ for increasing number of sequences, n. In other words, adding more oscillator rings in the TRNG design should improve the bias if the rings are independent.

We have carried out some experiments on the randomness of the TRNG in [12] (without any post-processing) and our proposal in Figure A.2. The experiments are carried out on a Starter Development Board from Altera containing a Cyclone II FPGA. This device has a core voltage of 1.2V and is fabricated in 90nm technology. Quartus II WebEdition 6.1 is used for synthesis and P&R (Place and Route). The sequences of random bits generated inside the FPGA is stored in an external SRAM and transmitted to a PC for analysis through an asynchronous serial connection. The result is blocks of subsequent random number bits from the TRNG where each block has a maximum size of 4Mbit. The sampling frequency used in this experiment is 50MHz. No constrains have been put on the P&R tool regarding the placement of the inverters in the FPGA.

We have implemented the two configurations of TRNG (refer to Figure A.1 and A.2), recorded 10 blocks of 1Mbit of random data from each configuration and calculated the frequency of ones in all blocks. We have done the experiment with oscillator rings of length 3 and 13 with varying number of rings. The results are shown in Figure A.4 and they indicate that the design in [12] has a bias after the XOR of the oscillator rings. The tendency is that the bias increases with the increasing number of rings and there is a majority of zeros in the output. This shows that there is some dependency or correlation in the random sequences. It seems that this bias is due to the problem with the high number of transitions at the input of the XOR tree and the sampling flip-flop. For our configuration (Figure A.2), it is seen that the bias is close to $\frac{1}{2}$ and that it converges rapidly to $\frac{1}{2}$. This shows that our proposal behaves accordingly to the theory of XOR of independent random sequences.

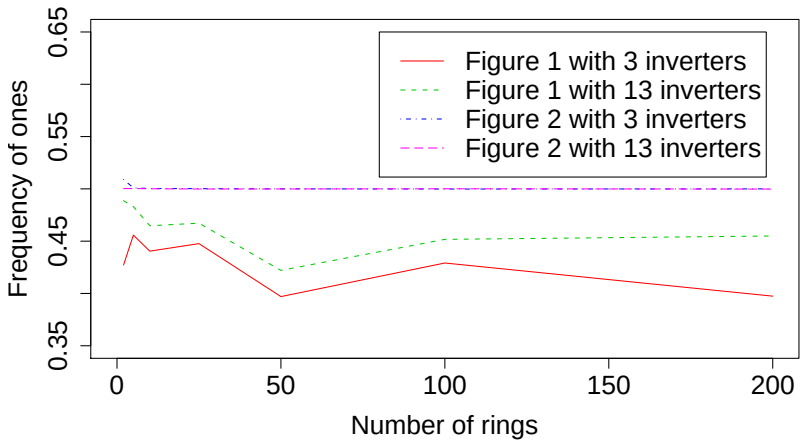

Figure A.4: Bias of the TRNGs on Figure A.1 and A.2

A.6 Distribution of Ring Frequencies

According to [12], the assumption of the randomness is that the equal length oscillator rings will have the same frequency while the phase drift related to the jitter causes the drifting of the transition regions. We believe that the frequencies of the oscillator rings will be different from each other. We have carried out some experiments where we have implemented 64 oscillator rings in the Altera Cyclone II FPGA and tapped out the signal from each of these rings to I/O-pins on the FPGA. These frequencies are measured with an oscilloscope.

Figure A.5 show the histograms of the frequencies of ring oscillators with 5 and 31

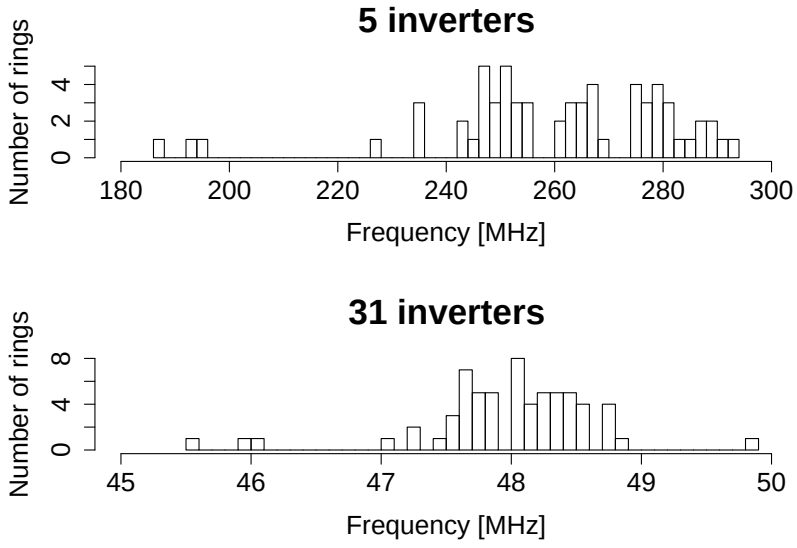

Figure A.5: Histogram of ring frequencies

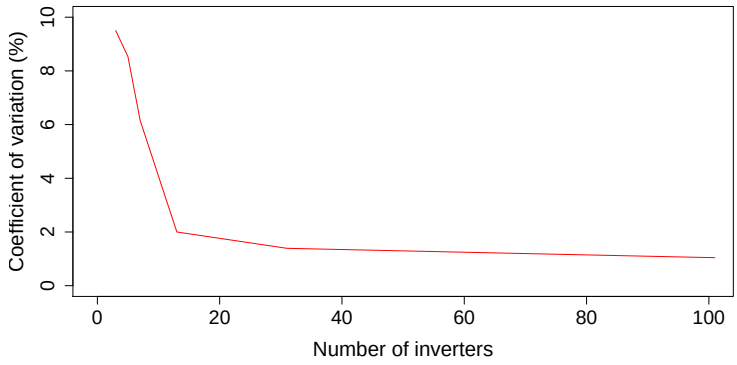

Figure A.6: Dispersion of frequencies

inverters respectively[2]. From this experiment, it is observed that the distribution of the frequencies for short rings does not follow a Gaussian distribution and the frequencies are clustered in groups. For longer rings, the clustering is not so obvious and the distribution is approaching Gaussian with only some values far from the mean.

When examining similar histograms for other lengths, it is observed that the dispersion is decreasing with increasing number of inverters. In Figure A.6, the dispersion is measured by the coefficient of variation defined as the percentage of $\frac{\sigma}{\mu}$ where σ is the standard deviation and μ is the mean. It can be seen that the dispersion is high for short rings and decreasing with longer oscillator rings. Based on this observation, using oscillator rings with only 3 inverters will give the highest dispersion in the frequencies.

To explain the behavior of the frequency distribution, the architecture of the Altera Cyclone II FPGA [1] is examined. This FPGA consists of a matrix with logic elements (LE), each containing a programmable register and a LUT for implementing any logic function of four inputs. 16 of these LEs are then grouped into a logic array block (LAB). All the LEs and LABs are connected together via different routing resources depending on the distance between them inside the FPGA. When running P&R for the design in an FPGA, the inverters in the oscillator rings are located at physical LEs. Depending on the placement, the routing delay between the LEs will differ. If all the inverters are placed in LEs inside one LAB, the routing delay will be short. If the inverters are placed in LEs in different LABs, the routing delay will be increased resulting in a lower frequency of the oscillator ring. In addition, there will also be some variation in the delay on each LUT. All these variations in the routing delays cause the distribution of the oscillator ring frequencies and the clustering for short rings. For long oscillator rings, the difference between the routing delays of each ring will be smaller due to the fact that the inverters have to be placed in more than one LAB. For other FPGAs with similar architecture, the ring oscillator frequencies will result in similar distributions.

Due to the observed distribution of frequencies of equal length oscillator rings, the transition regions will quickly be spread out over the sampling time period and the probability of interaction between the oscillator ring outputs is reduced.

[2]For oscillator rings with 3 inverters, the measured frequency is higher than the specified maximum frequency of 300MHz of an I/O-pin for our Cyclone II FPGA. However, the measurements with 3 inverters gave a similar histogram as shown in Figure A.5 with 5 inverters.

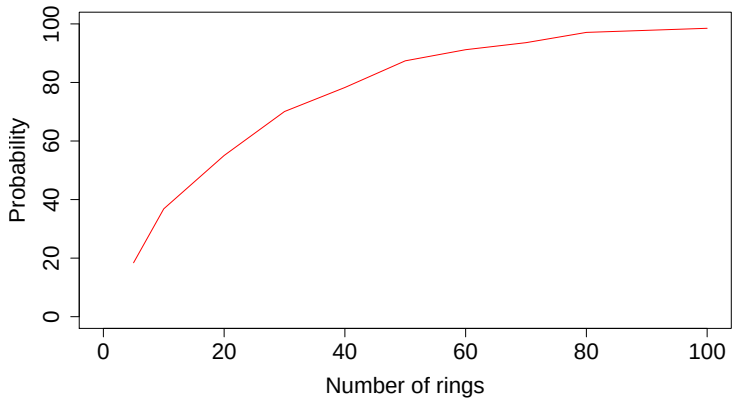

Figure A.7: Simulated probability of hitting a transition region

A.7 TRNG Implementation

We have done an implementation of our proposed TRNG in Figure A.2. In order to have a fast and small TRNG, the number of inverters in the oscillator ring is selected to be 3. A sampling frequency of 100MHz is selected, resulting in a throughput of 100Mbps since our TRNG do not use any post-processing.

The required number of rings is calculated based on the probability to hit the transition region with the sampling. Sunar et al. [12] computed this by using a combinatorial approach (coupon collector's problem). An alternative way is to make a statistical model of the TRNG and perform simulations in order to decide how many oscillator rings are needed to achieve a high probability such that at least one ring are sampled in the transition region. When the size of the jitter is small as compared to the sampling period, the simulations show that the number of rings in the transition region follows a Poisson distribution with a parameter $\lambda = k \cdot r$ where r is the number of rings and k is a constant depending on the size of the jitter as compared to the sampling period. The probability of sampling in at least one of the transition regions versus the number of rings is shown in Figure A.7. It shows that the probability increases rapidly for small number of rings, but many oscillator rings are needed to get a 100% certainty.

We have carried out an experiment where we have used 50 oscillator rings with 3 inverters, a sampling frequency of 100MHz and without any post-processing. A total of 1000 blocks of 1Mbit (a total of 1Gbit) of random data have been captured from the TRNG. The data is tested using the statistical test of NIST (SP 800-22) [10] and DIEHARD [9]. The random data passed both tests. We also did the same experiment for this configuration with only 25 oscillator rings. The random data did also pass both the NIST and the DIEHARD tests. This experiments indicate that there is probably not necessary to have almost 100%

Oscillator rings	LUT only LEs	LUT/Register LEs	Total LEs
25	57	26	83
50	116	51	167

Table A.1: Resources used in the Altera FPGA

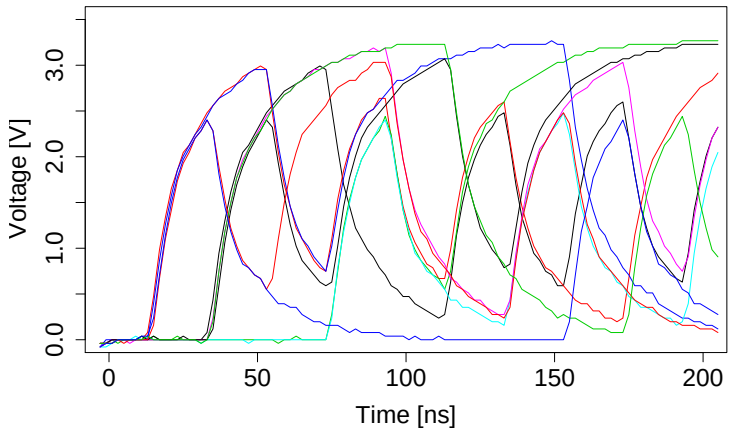

Figure A.8: 10 restarts with 25 rings

certainty to hit at least one transition region in order to pass the NIST and DIEHARD statistical tests for this kind of TRNG.

Table A.1 shows the number of resources used for our TRNG in the Altera Cyclone II FPGA. For 25 oscillator rings, the number of LEs is less than 100 ($< 1\%$ of the total number of LEs in our medium size FPGA). For comparison, the original design in [12] occupies more than 1800 LEs.

In order to examine the randomness of our TRNG after start-up, an oscilloscope is used to capture the random output when restarting the TRNG several times. While the FPGA is held in reset, the oscillator ring outputs are kept at zero or low level. When the reset is deactivated, the oscillator rings start to oscillate. In Figure A.8, 10 restart sequences from the output of the TRNG are captured where the oscilloscope is triggered on a clocked version of the reset signal at the origin of the graph. The sampling frequency is 50MHz. Because of the bandwidth limitation in the oscilloscope, the measured outputs are not square signals. It can be seen that all the outputs starts at zero, but there is a deviation between the rings after the first clock period of 20ns. This experiment shows that our TRNG outputs randomness quickly after a restart.

A.8 Conclusion

We have analyzed the TRNG in [12] and have proposed an enhancement of a TRNG based on oscillator rings. By adding an extra flip-flop after each inverter ring before the XOR tree, we have shown that the performance is much better than [12] regarding the random signal. We have also shown that the frequencies of each ring are not equal but have some kind of distribution. Smaller rings will have higher dispersion in the distribution and therefore also better potential for fast generation of randomness after restart.

We have implemented the TRNG of Figure A.2 and carried out statistical tests on the resulting random bit sequences. It is shown that our TRNG passes both the NIST and DIEHARD tests without post-processing. The throughput of the TRNG is 100Mbps and the resources used in the FPGA are less than 100 logic elements in an Altera Cyclone II FPGA.

A.9 Bibliography

[1] ALTERA. Cyclone II Device Handbook, Volume 1. Tech. rep., Altera Corporation, Feb. 2008. 88, 92, 102, 107

[2] BUCCI, M., AND LUZZI, R. Design of Testable Random Bit Generators. In *Proceedings of the 7th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'05)* (2005), vol. 3659 of *Lecture Notes in Computer Science*, Springer, pp. 147–156. doi:10.1007/11545262_11. 5, 7, 30, 98, 102

[3] DAVIES, R. B. Exclusive OR (XOR) and Hardware Random Number Generators. February 2002. 33, 105

[4] DICHTL, M., AND GOLIĆ, J. D. High-Speed True Random Number Generation with Logic Gates Only. In *Proceedings of the 9th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'07)* (2007), vol. 4727 of *Lecture Notes in Computer Science*, Springer, pp. 45–62. doi:10.1007/978-3-540-74735-2_4. 11, 12, 13, 16, 30, 31, 32, 39, 64, 102, 103

[5] FISHER, V., AND DRUTAROVSKÝ, M. True Random Number Generator Embedded in Reconfigurable Hardware. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 415–430. doi:10.1007/3-540-36400-5_30. 9, 10, 14, 30, 45, 102

[6] GOLIĆ, J. D. New Methods for Digital Generation and Postprocessing of Random Data. *IEEE Transactions on Computers 55*, 10 (2006), 1217–1229. doi:10.1109/TC.2006.164. 11, 12, 14, 15, 20, 30, 46, 58, 63, 102

[7] JUN, B., AND KOCHER, P. The Intel Random Number Generator. *White paper prepared for Intel Corporation* (Apr. 1999). 5, 29, 101

[8] KOHLBRENNER, P., AND GAJ, K. An Embedded True Random Number Generator for FPGAs. In *Proceedings of the ACM/SIGDA 12th International Symposium on Field Programmable Gate Arrays (FPGA'04)* (2004), ACM, pp. 71–78. doi:10.1145/968280.968292. 10, 11, 14, 30, 45, 87, 102

[9] MARSAGLIA, G. DIEHARD: A Battery of Tests of Randomness. 1996. 7, 15, 38, 46, 47, 77, 103, 108

[10] NIST. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Special Publication 800-22 Revision 1a, April 2010. 7, 15, 37, 38, 46, 47, 65, 77, 103, 108

[11] SCHELLEKENS, D., PRENEEL, B., AND VERBAUWHEDE, I. FPGA Vendor Agnostic True Random Number Generator. In *Proceedings of the 16th International Conference on Field-Programmable Logic and Applications (FPL'06)* (2006), IEEE, pp. 1–6. doi:10.1109/FPL.2006.311206. 12, 14, 31, 102

[12] SUNAR, B., MARTIN, W. J., AND STINSON, D. R. A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. *IEEE Transactions on Computers 56*, 1 (2007), 109–119. doi:10.1109/TC.2007.250627. 4, 11, 12, 14, 15, 19, 20, 21, 30, 31, 32, 34, 37, 39, 41, 45, 53, 58, 63, 77, 78, 83, 87, 101, 102, 103, 105, 106, 108, 109, 125, 126

[13] TEKTRONIX INC. Understanding and Characterizing Timing jitter. 2003. 9, 30, 102

[14] TKACIK, T. E. A Hardware Random Number Generator. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 450–453. doi:10.1007/3-540-36400-5_32. 10, 14, 30, 45, 102

Appendix B

Preproceedings: Security Implications of Crosstalk in Switching CMOS Gates[1]

Abstract

The energy dissipation associated with switching in CMOS logic gates can be used to classify the microprocessors activity. It is relatively easy to classify activity by the number of transitions, i.e. Hamming Distance (HD). In this article we consider layout dependent phenomena, such as capacitive crosstalk to derive a more precise power model. We show that for an 8 bit data bus, crosstalk may improve detection performance from 2.5 bits (HD based detector) to theoretical 5.7 bits and simulated 5.0 bits (crosstalk based detector) of information pr sample. Thus we have shown that a layout specific phenomenon (capacitance) must be considered when analyzing security implications of power and electromagnetic side channels. A small case study is also included that support our simulations/theoretical results.

B.1 Introduction

When a microprocessor executes its program, power consumption (or resulting electromagnetic emanation) can be used to reveal the contents of program and/or data memory of the microprocessor. The correlation between power consumption and microprocessor activity has found many uses: to recover cryptographic keys [1, 3, 9, 10, 11], to reveal hidden hardware faults (trojans) on integrated circuits [2], to control the emanation through subversive software in the Wireless Covert Channel Attack [7] and to reverse engineer the code executed by microprocessors [13].

In side-channel attacks, a common power model used to simulate the power consumption is the Hamming Distance (HD) model, as it is simple and generic [11]. The model assumes the power consumption to be proportional to the number of transitions taking place. If this assumption was correct, signals transmitted on a parallel bus (e.g. intermediate values of the cryptographic algorithm) with the same HD should have equal power consumption and therefore be indistinguishable. This is not always the case, e.g. if Bayes classifier is used, as suggested by the template attack [3]. It has also been demonstrated in [8] that signals with the same number of transitions can be classified using a modified template attack.

The phenomena behind this may be known in the security community, but has received little attention. One article by Chen et al. [4], studies the effect of the coupling capacitance on masking schemes without a detailed examination of the phenomena. In their book "Power Analysis Attacks", Mangard et.al. [11] mention power simulation at analog level as "the most precise way to simulate the power consumption of digital circuits...". Parasitic elements, such as parasitic capacitances between the wires and unwanted capacitances in the transistors are mentioned. However, it is also stated that it is very common to make simplifications by lumping together extrinsic and intrinsic capacitances into a single

[1] Geir Olav Dyrkolbotn, Knut Wold and Einar Snekkenes. Preproceedings presented on the 13th Information Security Conference (ISC'10) in Boca Raton, Florida, October 2010.

capacitance to ground. This will, in fact, make the model incapable of explaining the results we are addressing in this article.

Parasitic couplings, and the coupling capacitance in particular are, however, a great concern within sub-micron VLSI design [5, 12, 14]. CMOS technology is currently being pushed into deep sub-micron range. As the number of transistors increase, the need for on-chip wiring increases as well and must be scaled accordingly. Parasitic couplings between interconnects, such as on-chip buses, must be taken seriously as they influence both the power consumption and maximum obtainable speed [5]. In [12], Moll et al. did a detailed analysis of the energy dissipation from two metal lines running close together. The lines were driven by CMOS inverters and transitions in one or two wires were studied. The effect of coupling capacitance between the two lines on the power consumption was shown analytically and simulated in HSPICE. The main result was that if two bus lines have transitions in the same or opposite direction at the same time, the total energy is either lower or higher than if the two transitions are treated independently. This is due to the coupling capacitance. Duan et al. [5] focus on crosstalk avoidance codes that aim to reduce the effect of the coupling capacitances by avoiding specific data transition patterns. Their model considers coupling capacitance, C_C, between three adjacent lines. They show that 3 bit transition patterns can be divided into 5 crosstalk classes based on the influence of the coupling capacitances, C_C. The energy consumption therefore depends on which crosstalk class the transition pattern belong to. The focus of Moll et al. [12] and Duan et al. [5] are both on power consumption and delays caused by the coupling capacitance. They have not considered security implications, such as the ability to use the variation in energy consumption to classify transition patterns. Correlations between data and energy consumption are exactly what side channel attacks, such as DPA and Template attack, rely upon.

We put forward the hypothesis that layout dependent phenomena, such as parasitic coupling between wires, can explain why it sometimes is possible to distinguish transition patterns with the same HD. In this article we present theory and simulations on some security implications of capacitive crosstalk in CMOS driven data busses. How will the coupling capacitance affect our ability to classify activity in a microprocessor, such as data transfer on a parallel bus? We look at the total dissipated energy from a parallel data bus driven by CMOS inverters. Our model is a generalization of Moll et al's [12], with inverters consisting of two MOSFET transistors, a load capacitance C_L connected to each inverter output and a coupling capacitance C_C connected between each bus line. Our model is generalized to n lines and simulations in PSPICE are done with eight bus lines.

The purpose of our simulation is to show that when the dissipated energy depends on the direction of change of nearby data lines, and not only the number of transitions taking place, the number of possible energy levels dissipating from the bus will increase, thus allowing classification of a larger number of transition patterns. Our hypothesis is that this can be used to explain why some signal with the same HD can be distinguished. Our model can easily take into consideration other layout dependent phenomena, potentially offering an explanation to classification of an even larger set of transition patterns. We will use the ability to extract entropy as our classifier performance indicator and show that a detector capable of detecting energy levels due to crosstalk can extract more information than a detector based on HD only.

This paper is organized as follows: Section B.2 presents the hypothesis of layout dependent phenomena. Section B.3 presents our model and necessary theory to calculate the energy dissipation. Section B.4 is an analytic analysis of security implications. Section B.5 presents simulation results. Finally, a conclusion is drawn.

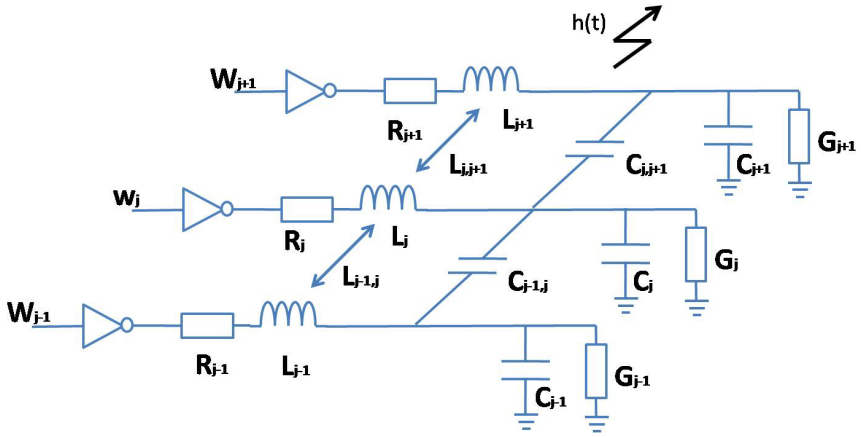

Figure B.1: Model of layout dependent phenomena

B.2 Layout Dependent Phenomena

In a physical implementation of any circuit (e.g. CMOS based microprocessor) a number of phenomena will influence the energy dissipation and the resulting radiated electromagnetic field. These phenomena include inductance and capacitance of conductors, inductance and capacitance between conductors, wireless transmission characteristics (i.e antenna properties) of conductors and other circuit elements and complex combinations of these phenomena. These phenomena apply to any transistors and wires in a circuit, but we choose to look at a portion of wires running parallel, as we expect them to be relatively good antennas and therefore a good source for side channel information. This is illustrated in the model of a parallel bus, driven by CMOS inverters, seen in Figure B.1.

B.2.1 Inductance and Capacitance of Conductors

Any conductor, W_j, carrying an electric current will have an associated distributed resistance R_j, inductance L_j, conductance G_j and capacitance C_j, expressed as a characteristic impedance, Z_{0j}. The characteristic impedance is often modeled as an infinite series of lumped components. The inductance L_j and capacitance C_j will both block high frequency signals and act as a low pass filter. Small variations in the length and width of conductors result in small variations in the inductance. Small variations in the area and distance to ground plane result in small variations in the capacitance. There will therefore be small variations in how signals on different conductors (e.g. bus lines) are filtered.

B.2.2 Inductance and Capacitance between Conductors

Crosstalk can be defined as the coupling of energy between two conductors. Inductive coupling is caused by mutual inductance, $L_{j,j+1}$, (i.e magnetic field) and capacitive coupling is caused by mutual capacitance, $C_{j,j+1}$, (i.e electric field) between wire j and $j+1$. These couplings occur along the entire length of the conductor, but are also modeled as lumped components (Figure B.1). The interaction of magnetic and electric fields will effectively change the characteristic impedance, Z_{0j}, associated with the conductor. This interaction is layout dependent (e.g. distance and length of wires) and will effect both delays and energy dissipation. An important property of crosstalk is its dependency on the activity on the wires. Moll et al. [12] state that, "coupling capacitance is very different from the capacitance to ground because it depends on the switching activity... ". If two lines are low and

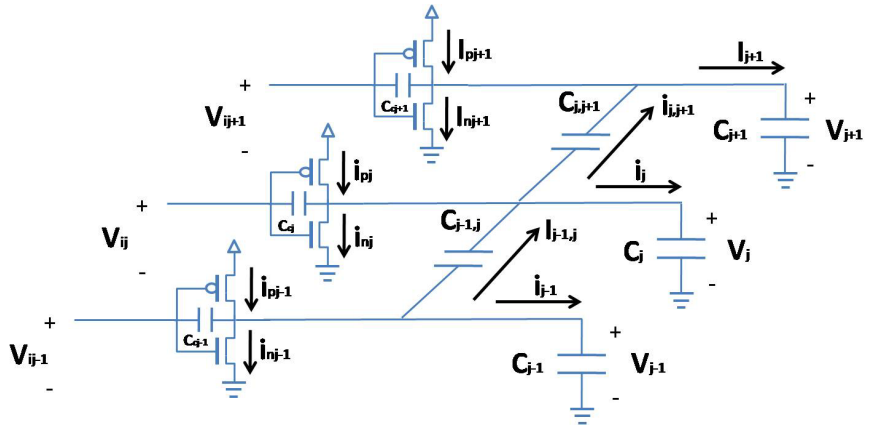

Figure B.2: Simplified model, assuming load and coupling capacitances to be dominant

rise at the same time, the mutual capacitance coupling, $C_{j,j+1}$, does not have to be charged. However, if one line remains low and the other rises, $C_{j,j+1}$ must be charged, resulting in increased rise time and power consumption.

B.2.3 Wireless Transmission Characteristics

Any circuit element in the microprocessor, conducting electric current, can be considered an antenna. An antenna is a transducer converting electric current into electromagnetic waves, characterized by properties such as: resonant frequency, gain, radiation pattern, impedance, efficiency, bandwidth and polarization. These properties depend on factors such as: amount of current, length/shape and material of the circuit element. In addition, the electromagnetic waves will be influenced by filtering, reflection and interference from surrounding material and circuit elements. The relationship between the current (i.e power consumption) and the electromagnetic wave can be expressed by a transfer function $h(t)$ (see Figure B.1). Predicting $h(t)$ is not trivial, if possible at all, as most physical systems are not linear by nature. This is left for future work, but it is a fair assumption that relatively long bus lines are good antennas.

B.2.4 Complex Combinations of Factors

Finally, complex combinations of layout dependent phenomena may be the key to identify minute differences in microprocessor activity, e.g. the radiation efficiency of bus lines combined with data and layout dependencies of the line characteristics due to crosstalk suggest that the emanation detected will have data and layout dependent variations in power consumption and delay. In the following, we will assume that the coupling capacitance is the dominating factor, and show how this can explain why some signals with the same HD can be distinguished. This will show the potential effect of layout dependent phenomena on classifying microprocessor activity. Our work can easily be extended by including more layout dependent phenomena if a more precise result is needed.

B.3 Theoretical Considerations

By limiting the model to only coupling and load capacitances, the model in Figure B.1 can be simplified as seen in Figure B.2. This is a generalization of the model for two lines used by Moll et al. [12] and includes a model of the CMOS inverter.

In order to run simulations in PSPICE, we need an expression for the total energy dissipation, E_T. The energy dissipation for wire j in the p and n type transistor can be expressed as:

$$E_{pj} = \int (V_{DD} - V_j) i_{pj} dt \tag{B.1}$$

$$E_{nj} = \int V_j i_{nj} dt \tag{B.2}$$

The overall energy dissipation for an n wire bus is then given by:

$$E_T = \sum_{j=1}^{n} (E_{pj} + E_{nj}) \tag{B.3}$$

Combining and rearranging (B.1), (B.2) and (B.3) the overall energy dissipation can be written as:

$$E_T = \sum_{j=1}^{n} V_{DD} \int i_{pj} dt - \sum_{j=1}^{n} \int V_j (i_{pj} - i_{nj}) dt \tag{B.4}$$

Using Kirchhoff's circuit laws and the current voltage relationship $i(t) = C\frac{dV(t)}{dt}$, the terms $(i_{pj} - i_{nj})$ can be written as:

$$i_{pj} - i_{nj} = (C_j + C_{j,j+1} + C_{j-1,j} + C_{cj})\frac{dV_j}{dt} - C_{cj}\frac{dV_{ij}}{dt} - C_{j,j+1}\frac{dV_{j+1}}{dt} - C_{j-1,j}\frac{dV_{j-1}}{dt} \tag{B.5}$$

Notice that the results in [12] are easily found from (B.4) and (B.5) by setting $n = 2$ (two adjacent lines). Equation (B.4) is used in PSPICE to simulate the total energy dissipation, $\hat{E}_T$, with the following assumptions:

- The transitions on the data bus are concurrent in time. Moll et al [12] showed that the effect of the coupling capacitance is maximum when all transitions are synchronized.
- The load capacitances for data bus lines are identical ($C_j = C_L$ for $j = \{1, 2, \cdots, n\}$)
- Coupling capacitances are only found between adjacent line and are identical ($C_{j,j+1} = C_C$ for $j = \{1, 2, \cdots, n-1\}$)

These assumptions are not unrealistic in real bus architecture on a device. If, however, the transitions are shifted in time with more than the rise time of the signal, the effect of the coupling capacitance is reduced and the transitions can be regarded as single transitions.

In order to compare the simulated energy dissipation ($\hat{E}_T$) with analytic values (E_T), simpler expressions than (B.4) and (B.5) are needed. If the contributions from the load (C_L) and coupling capacitance (C_C) are dominant to the dissipated energy, then E_T can be expressed by the more intuitive equations:

$$E_T = \frac{1}{2} C_L V_{DD}^2 (k + \alpha\lambda) = E_0 (k + \alpha\lambda) \tag{B.6}$$

where $E_0 = \frac{1}{2} C_L V_{DD}^2$, V_{DD} is the power supply voltage, k is the number of transitions on the data bus ($k = 0, 1, 2, \ldots$), $\lambda = C_C/C_L$ and α is the crosstalk index indicating the coupling capacitance induced crosstalk, similar to the crosstalk classes in [5]. For an n line bus the crosstalk index α is the sum of the crosstalk influence of each line:

$$\alpha = \sum_{j=1}^{n} \alpha_j \tag{B.7}$$

Let $\delta_j \in \{0, \pm 1\}$ be the normalized voltage change on line j, then $\delta_{j,k} = \delta_j - \delta_k$, and

$$\alpha_j = \begin{cases} 0 & \textit{no transition line } j \\ |\delta_{j,j-1} + \delta_{j,j+1}| & \textit{otherwise} \end{cases} \tag{B.8}$$

It can be shown that $\alpha_j = \{0, 1, 2\}$ for lines with only one adjacent line (edges), and $\alpha_j = \{0, 1, 2, 3, 4\}$ for lines with two adjacent lines. In the next section we will use (B.6) and (B.7) to analyze which transition patterns that can be distinguished.

B.4 Security Implications

The relationship between energy dissipation, number of transitions, the crosstalk index, load capacitance and coupling capacitance in (B.6) can be used to analyze delays and energy dissipation of sub-micron VLSI design [5, 12, 14]. However, we are interested in the security implications of layout dependent phenomena, and in this article the coupling capacitance in particular. How will the coupling capacitance effect our ability to classify activity in a microprocessor, such as data transfer on a parallel bus?

Let A be the set of possible transitions on an n bit parallel bus. Let U be the number of positive transitions and D the number of negative transitions. Since "no transition" can be both $0 \rightarrow 0$ and $1 \rightarrow 1$ there are $|A| = 4^n$ possible transition patterns for an n-bit bus. Given an unknown observation of the energy dissipation, the objective is to assign the observation to one of the possible transition patterns (classes), e.g. by using the template attack [3]. A model of the energy dissipation should ideally be able to explain all $|A| = 4^n$ classes. This may not be possible, either because of simplifications to the model or physical properties such that multiple transition patterns indeed use the same amount of energy.

Let A_k, $k = \{0, \cdots, n\}$ be the subset of A that has $k = U + D$ transitions. The number of transition patterns in each subset given by:

$$|A_k| = 2^n \binom{n}{k} \tag{B.9}$$

The total number of possible transitions on an 8 wire bus ($|A| = 65536$) can be divided into 9 subsets, $A_0, A_1, \cdots, A_8$ based on the number of transitions, k. The energy dissipation, E_T (using (B.6) with $\alpha = 0$), associated with each subset without crosstalk influence and $|A_k|$ can be seen in Table B.1. A model that assumes energy dissipation proportional to the number of transition, can only classify transition pattern by the energy level of these 9 subsets. In Table B.1 there are e.g. 14336 transition patterns with energy level $3E_0$ that are indistinguishable by the number of transitions alone.

Taking into consideration the coupling capacitor of the model in Figure B.2 and using (B.6), each subset A_k can be split into a number of new energy levels, depending on the crosstalk influence. This gives a number of smaller subsets A_k^α, $|A_k| > |A_k^\alpha|$ and $\sum_{all\alpha} |A_k^\alpha| = |A_k|$, where α is the crosstalk index of (B.7). Computing $|A_k^\alpha|$ for a fixed number of bus lines n can be done by constructing a table of $(2^k)^2$ elements corresponding to all possible transition patterns. For each of these, first compute the crosstalk index α (B.7), then the energy dissipation E_T (B.6). $|A_k^\alpha|$ can then be computed by counting the table entries for each tuple $\{k, \alpha\}$. This has been done for an 8 bit bus in Table B.1. The results show that taking into consideration the coupling capacitance increases the number of energy levels from 9 in the HD model to 93 in the crosstalk model, e.g. the 14336 transition patterns with 3 transitions previously indistinguishable can now be split into 10 energy levels. The largest increase in energy levels is found for 6 transitions with 21 new energy levels. Notice that for a finite n, there are restrictions on the values of α as the number of transitions increase, i.e all energy levels are not possible. This applies to 6, 7 and 8 transitions for an 8 bit bus.

Also notice that given an ideal classifier, there is no confusion between subsets of the same k as they all have unique energy levels. There may, however, be confusion between subsets of different k. The extent of this confusion is architecture dependent, expressed by λ, e.g. subset A_2^6 have the same energy level as A_3^2 if $\lambda = 1/4$, in case they should be treated as one subset. It is easy to show that confusion between transition A (energy E_{TA}, k_A transitions and crosstalk index α_A) and B (energy E_{TB}, k_B transitions and crosstalk

Table B.1: The Table shows the number of transition patterns, without ($|A_k|$) and with ($|A_k^\alpha|$) crosstalk influence, belonging to a certain energy level, E_T. k is the number of transitions (Hamming Distance) and α is the crosstalk index

k	E_T [pJ]	$\lvert A_k\rvert$	α	E_T [pJ]	$\lvert A_k^\alpha\rvert$	k	E_T [pJ]	$\lvert A_k\rvert$	α	E_T [pJ]	$\lvert A_k^\alpha\rvert$
0	0	256	0	0	256	6	$6E_0$	7168	1	$E_0(6+\lambda)$	16
1	E_0	2048	1	$E_0(1+\lambda)$	512				2	$E_0(6+2\lambda)$	88
			2	$E_0(1+2\lambda)$	1536				3	$E_0(6+3\lambda)$	160
2	$2E_0$	7168	1	$E_0(2+1\lambda)$	256				4	$E_0(6+4\lambda)$	320
			2	$E_0(2+2\lambda)$	896				5	$E_0(6+5\lambda)$	80
			3	$E_0(2+3\lambda)$	2560				6	$E_0(6+6\lambda)$	360
			4	$E_0(2+4\lambda)$	2560				7	$E_0(6+7\lambda)$	640
			5	$E_0(2+5\lambda)$	256				8	$E_0(6+8\lambda)$	960
			6	$E_0(2+6\lambda)$	640				9	$E_0(6+9\lambda)$	160
3	$3E_0$	14336	1	$E_0(3+1\lambda)$	128				10	$E_0(6+10\lambda)$	560
			2	$E_0(3+2\lambda)$	512				11	$E_0(6+11\lambda)$	960
			3	$E_0(3+3\lambda)$	2048				12	$E_0(6+12\lambda)$	960
			4	$E_0(3+4\lambda)$	2560				13	$E_0(6+13\lambda)$	160
			5	$E_0(3+5\lambda)$	3328				14	$E_0(6+14\lambda)$	400
			6	$E_0(3+6\lambda)$	1792				15	$E_0(6+15\lambda)$	640
			7	$E_0(3+7\lambda)$	2048				16	$E_0(6+16\lambda)$	320
			8	$E_0(3+8\lambda)$	1536				17	$E_0(6+17\lambda)$	80
			9	$E_0(3+9\lambda)$	128				18	$E_0(6+18\lambda)$	120
			10	$E_0(3+10\lambda)$	256				19	$E_0(6+19\lambda)$	160
4	$4E_0$	17920	1	$E_0(4+\lambda)$	64				21	$E_0(6+21\lambda)$	16
			2	$E_0(4+2\lambda)$	288				22	$E_0(6+22\lambda)$	8
			3	$E_0(4+3\lambda)$	1152	7	$7E_0$	2048	1	$E_0(7+\lambda)$	8
			4	$E_0(4+4\lambda)$	1728				2	$E_0(7+2\lambda)$	48
			5	$E_0(4+5\lambda)$	2496				5	$E_0(7+5\lambda)$	48
			6	$E_0(4+6\lambda)$	1824				6	$E_0(7+6\lambda)$	240
			7	$E_0(4+7\lambda)$	2816				9	$E_0(7+9\lambda)$	120
			8	$E_0(4+8\lambda)$	2304				10	$E_0(7+10\lambda)$	480
			9	$E_0(4+9\lambda)$	2496				13	$E_0(7+13\lambda)$	160
			10	$E_0(4+10\lambda)$	864				14	$E_0(7+14\lambda)$	480
			11	$E_0(4+11\lambda)$	1152				17	$E_0(7+17\lambda)$	120
			12	$E_0(4+12\lambda)$	576				18	$E_0(7+18\lambda)$	240
			13	$E_0(4+13\lambda)$	64				21	$E_0(7+21\lambda)$	48
			14	$E_0(4+14\lambda)$	96				22	$E_0(7+22\lambda)$	48
5	$5E_0$	14336	1	$E_0(5+\lambda)$	32				25	$E_0(7+25\lambda)$	8
			2	$E_0(5+2\lambda)$	160	8	$8E_0$	256	0	$E_0(8)$	2
			3	$E_0(5+3\lambda)$	512				4	$E_0(8+4\lambda)$	14
			4	$E_0(5+4\lambda)$	896				8	$E_0(8+8\lambda)$	42
			5	$E_0(5+5\lambda)$	896				12	$E_0(8+12\lambda)$	70
			6	$E_0(5+6\lambda)$	1024				16	$E_0(8+16\lambda)$	70
			7	$E_0(5+7\lambda)$	1536				20	$E_0(8+20\lambda)$	42
			8	$E_0(5+8\lambda)$	1920				24	$E_0(8+24\lambda)$	14
			9	$E_0(5+9\lambda)$	1728				28	$E_0(8+28\lambda)$	2
			10	$E_0(5+10\lambda)$	1088						
			11	$E_0(5+11\lambda)$	1536						
			12	$E_0(5+12\lambda)$	1152						
			13	$E_0(5+13\lambda)$	896						
			14	$E_0(5+14\lambda)$	256						
			15	$E_0(5+15\lambda)$	512						
			16	$E_0(5+16\lambda)$	128						
			17	$E_0(5+17\lambda)$	32						
			18	$E_0(5+18\lambda)$	32						

index α_B) happens when:

$$\lambda_{AB} = \frac{k_B - k_A}{\alpha_A - \alpha_B} \tag{B.10}$$

λ_{AB} values that are close to the real $\lambda = C_C/C_L$ indicate subsets that will be difficult to distinguish.

Finally, we have only shown how to split the subset A_k into smaller subsets A_k^α by considering the effect of the coupling capacitance (i.e α). This idea can easily be generalized, such that A_k is split into subsets A_k^β, where $|A_k| > |A_k^\beta|$, and β is the influence of other layout dependent phenomena. Examples of phenomena for future work include: variations in coupling and load capacitance, coupling capacitance between line j and $j+2$, inductance, effect of bends in circuit paths and multi layer capacitance (3-dimentional).

B.4.1 Classification Performance

Table B.1 shows that, taking into consideration the coupling capacitance, we are able to increase the number of subsets (or energy levels) A_k to A_k^α. For the purpose of comparing alternative detectors we will assume uniform random transition. Thus for an 8 bit bus we would like the detector to extract 16 bits. We will use the ability to extract entropy as our classifier performance indicator. The entropy (i.e bits of information) for a detector, when there are r energy levels, can be calculated using:

$$H(x) = -\sum_{i=1}^{r} p(x_i) log\ p(x_i) \tag{B.11}$$

In the following, we have assumed an 8 bit bus width, thus there are $4^8 = 65536$ possible transitions. Call the detector that can extract 16 bits of information a level detector. If we assume that one only has bus activity when initial and final state are different, and that $0 \rightarrow 1$ and $1 \rightarrow 0$ can be distinguished, an observation will give us the following entropy: $-(1/2log1/2 + 1/4log1/4 + 1/4log1/4) = 3/2$ bits as we cannot distinguish $0 \rightarrow 0$ from $1 \rightarrow 1$, but $0 \rightarrow 0$,$0 \rightarrow 1$, $1 \rightarrow 0$ can be distinguished. Thus, each observation will give us $3/2$ bits pr line. The theoretical optimum for an 8 bit bus with a 'transition detector' would be $8 \cdot 3/2\ bits = 12\ bits$, assuming all observable transitions are distinguishable. In other words, by observing transitions rather than levels, we loose 4 bits ($1/2$ bit per line) compared to the setting where we would observe the states.

Using the results of Table B.1, we can now calculate the entropy of a detector that can distinguish HD only (A_k) and a detector that can distinguish energy levels due to crosstalk (A_k^α).

The entropy of an HD detector is found using (B.11) with $r = 9$ and $p(x_i) = |A_{i-1}|/65536$ giving an entropy of 2.5 bits. The entropy of a crosstalk detector is found using (B.11) with $r = 93$ and $p(x_i) = |A_{i-1}^\alpha|/65536$ giving an entropy of 5.7 bits.

The difference between the ideal value of a level detector and the entropy of other detectors, represent the amount of guessing needed for classifying an observation. By considering the coupling capacitance and not only HD, we extract more information out of each observation, therefore reducing the amount of "guessing" needed for classification. In the next section we present simulations validating the effect of the coupling capacitance.

B.5 Simulations

The simulations are performed in PSPICE with $C_L = 400fF$, $C_C = 250fF$, $V_{dd} = 3V$ and a rise- and fall-time of $200ps$ of the input voltages (same as [12]). The inverter drivers are equal and balanced. Equations (B.4) and (B.5) are used in PSPICE to find the simulated energy dissipation $\hat{E}_T$.

B.5.1 Model Validation

Simulations were initially carried out and compared with the results of [5, 12] as a model validation. For two wires, as seen in Table B.2, it is clear that the energy dissipation for two

Table B.2: Dissipated energy when considering crosstalk for 2 adjacent wires

Transition pattern	**Transitions k**	**Crosstalk** α	**Theoretical** E_T **[pJ]**	**Simulated** $\hat{E}_T$ **[pJ]**
$00 \rightarrow 01$	1	1	2.9	2.7
$00 \rightarrow 10$	1	1	2.9	2.7
$00 \rightarrow 11$	2	0	3.6	3.5
$01 \rightarrow 10$	2	4	8.1	8.0

Table B.3: Analytic (E_T) and simulated ($\hat{E}_T$) dissipated energy when considering crosstalk for bus with 8 lines. k is the number of transitions (Hamming Distance) and α is the crosstalk index

Transition pattern	k	α	E_T [pJ]	$\hat{E}_T$ [pJ]	Transition pattern	k	α	E_T [pJ]	$\hat{E}_T$ [pJ]
0000 0000 → 0000 0001	1	1	2,9	2.9	0000 0000 → 0011 1111	6	1	11,9	12.3
0000 0000 → 0000 0010	1	2	4,1	4.1	0000 0000 → 1001 1111	6	2	13,1	13.4
0000 0000 → 0000 0011	2	1	4,7	4.8	0000 0000 → 0101 1111	6	3	14,2	14.5
0000 0000 → 1000 0001	2	2	5,9	5.9	0000 0000 → 1010 1111	6	4	15,3	15.7
0000 0000 → 0000 0101	2	3	7,0	7.0	0000 0001 → 0011 1110	6	5	16,4	16.7
0000 0000 → 0000 1010	2	4	8,1	8.1	0000 0001 → 1001 1110	6	6	17,6	17.9
0000 0010 → 0000 0001	2	5	9,2	9.3	0000 0001 → 0101 1110	6	7	18,7	19.0
0000 0100 → 0000 0010	2	6	10,4	10.1	0000 0001 → 1010 1110	6	8	19,8	20.0
0000 0000 → 0000 0111	3	1	6,5	6.7	0000 0010 → 0011 1101	6	9	20,9	20.8
0000 0000 → 1000 0011	3	2	7,7	7.8	0000 0010 → 1001 1101	6	10	22,1	21.9
0000 0000 → 0000 1011	3	3	8,8	8.9	0000 0010 → 0101 1101	6	11	23,2	23.0
0000 0000 → 1000 1001	3	4	9,9	10.0	0000 0010 → 1010 1101	6	12	24,3	24.1
0000 0000 → 0001 0101	3	5	11,0	11.1	0000 0101 → 0011 1010	6	13	25,4	25.5
0000 0000 → 0010 1010	3	6	12,2	12.3	0000 0101 → 1001 1010	6	14	26,6	26.5
0000 0010 → 0000 1001	3	7	13,3	13.3	0000 0101 → 0101 1010	6	15	27,7	27.7
0000 0100 → 0001 0010	3	8	14,4	14.4	0000 0101 → 1010 1010	6	16	28,8	28.9
0000 0010 → 0000 0101	3	9	15,5	15.4	0000 1010 → 0011 0101	6	17	29,9	29.7
0000 0100 → 0000 1010	3	10	16,7	16.6	0000 1010 → 1001 0101	6	18	31,1	30.9
0000 0000 → 0000 1111	4	1	8,3	8.6	0000 1010 → 0101 0101	6	19	32,2	32.0
0000 0000 → 1000 0111	4	2	9,5	9.6	0010 1010 → 0001 0101	6	21	34,4	34.0
0000 0000 → 0001 0111	4	3	10,6	10.8	0101 0100 → 0010 1010	6	22	35,6	35.2
0000 0000 → 1000 1011	4	4	11,7	11.9	0000 0000 → 0111 1111	7	1	13,7	14.2
0000 0000 → 0010 1011	4	5	12,8	13.0	0000 0000 → 1011 1111	7	2	14,9	15.3
0000 0000 → 1001 0101	4	6	14,0	14.2	0000 0001 → 0111 1110	7	5	18,2	18.5
0000 0000 → 0101 0101	4	7	15,1	15.3	0000 0001 → 1011 1110	7	6	19,4	19.7
0000 0010 → 1001 0001	4	8	16,2	16.0	0000 0010 → 0111 1101	7	9	22,7	22.7
0000 0100 → 1001 0010	4	9	17,3	17.5	0000 0010 → 1011 1101	7	10	23,9	23.8
0000 0010 → 1000 0101	4	10	18,5	18.3	0000 0101 → 0111 1010	7	13	27,2	27.4
0000 0010 → 0100 0101	4	11	19,6	19.4	0000 0101 → 1011 1010	7	14	28,4	28.5
0000 0100 → 0100 1010	4	12	20,7	20.7	0000 1010 → 0111 0101	7	17	31,7	31.6
0000 1010 → 0000 0101	4	13	21,8	21.5	0000 1010 → 1011 0101	7	18	32,9	32.6
0001 0100 → 0000 1010	4	14	23,0	22.7	0001 0101 → 0110 1010	7	21	36,2	36.2
0000 0000 → 0001 1111	5	1	10,1	10.4	0001 0101 → 1010 1010	7	22	37,4	37.3
0000 0000 → 1000 1111	5	2	11,3	11.5	0010 1010 → 0101 0101	7	25	40,7	40.5
0000 0000 → 0010 1111	5	3	12,4	12.6	0000 0000 → 1111 1111	8	0	14,4	14.9
0000 0000 → 1001 0111	5	4	13,5	13.8	0000 0001 → 1111 1110	8	4	18,9	19.3
0000 0000 → 0101 0111	5	5	14,6	14.9	0000 0010 → 1111 1101	8	8	23,4	23.4
0000 0000 → 1010 1011	5	6	15,8	16.0	0000 0101 → 1111 1010	8	12	27,9	28.1
0000 0010 → 0111 0001	5	7	16,9	16.8	0000 1010 → 1111 0101	8	16	32,4	32.3
0000 0010 → 1011 0001	5	8	18,0	17.9	0001 0101 → 1110 1010	8	20	36,9	36.9
0000 0010 → 0101 1001	5	9	19,1	19.3	0010 1010 → 1101 0101	8	24	41,4	41.1
0000 0010 → 1010 1001	5	10	20,2	20.4	0101 0101 → 1010 1010	8	28	45,9	45.6
0000 0010 → 0110 0101	5	11	21,4	21.3					
0000 0010 → 1010 0101	5	12	22,5	22.5					
0000 0010 → 0101 0101	5	13	23,6	23.5					
0000 1010 → 1000 0101	5	14	24,8	24.5					
0000 1010 → 0100 0101	5	15	25,9	25.6					
0001 0100 → 0100 1010	5	16	27,0	27.0					
0000 1010 → 0001 0101	5	17	28,1	27.9					
0001 0100 → 0010 1010	5	18	29,3	29.1					

simultaneous transitions is either lower or higher than if treated as two single transitions, depending on the direction of the transitions, as expected. The small differences between analytic and simulated energy dissipation can be explained by simplifications in deriving (B.6) (e.g. omitting leakage currents, such as short-circuit and sub-threshold currents). Having validated our model, all the following simulations are done on an 8 bit bus.

B.5.2 Results and Discussion

Simulation results for 8 lines are shown in Table B.3. The table is not exhaustive, but includes results for all possible subsets A_k^α.

The simulated energy levels $\hat{E}_T$ are similar to the analytic values. The results confirm that energy consumption is proportional to the number of transitions and the crosstalk index, α. The crosstalk index depends on switching activity on adjacent lines and position, edge (one adjacent wire) or middle (two adjacent wires). As seen in Table B.3, the

Table B.4: Comparing the performance of different detectors for an 8 wire bus

type of detector	Entropy (information) [bits]
Level detector	16,0
Optimum transition detector	12,0
Crosstalk detector (theoretical)	5,7
Crosstalk detector (simulated)	5,0
HD detector	2,5

results also confirm that there is no confusion between energy levels for subsets of an equal number of transitions. However, there may be some confusion between some of the 93 subgroups, e.g. the energy dissipation of subset A_2^6 and A_3^4 are almost equal. This is expected as $\lambda_{AB} = 0,5$ is close to $\lambda = 0,63$ used in this experiment. Other examples can be found and this reduces the number of subsets depending on how accurate our detector is. As seen earlier, a theoretical crosstalk detector capable of separating all 93 energy levels will have an entropy of 5.7. It is therefore expected that a practical crosstalk detector will have a smaller entropy. It is difficult to decide which of the simulated energy levels should be merged together, as this will depend on the accuracy of the detector and the number of observations available. A random loss of 20% of the subsets will still, on average, have an entropy of 5.0. Even with this loss due to similar energy levels, the information gain is still 2.5 bits compared to the HD detector. The performance of the detectors are summarized in Table B.4:

B.5.3 Case Study and Future Work

In order to probe the practicality of our theory and simulations, we have collected a small set of experimental data. The objective was to see if analysis of electromagnetic side channel information also supports the division into crosstalk energy levels of Table B.1 and B.3.

A total of 1000 traces (observations) of each of the 18 crosstalk indexes for transitions pattern with Hamming Distance 5 where collected. The target was a smart card (i.e PIC 16F84A microprocessor). The electromagnetic emanation was captured using a broadband E near-field probe positioned as close to the microprocessor as possible, without any decapsulation.

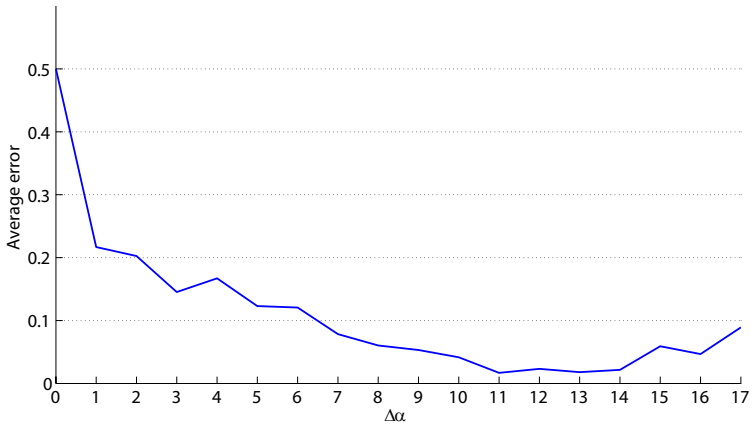

Figure B.3: Average classification error as a function of α distance, $\Delta\alpha$

Analysis was done according to the Modified Template Attack [8], and includes feature selection, training and evaluating the performance of a quadratic Bayes classifier (for details refer to [8]). The probability of error, P_e, was found from the confusion matrix [6]. The classification accuracy depends on how the observations are split, therefore, the average of 100 random permutations of 200 training observations and 800 test observations was used. Finally, the average classification error as a function of α distance ($\Delta\alpha = |\alpha_i - \alpha_j|$) was calculated.

Figure B.3 is a plot of our experimental data, suggesting that the average classification error gets reduced as alpha distance increases. This supports our simulation/theoretical results. We hypothesize that the discrepancy between our simulation/theoretical results for alpha distance 4 is a consequence of statistical uncertainty/noise in the experimental data. Currently, we cannot offer any explanation for why classification error seems to increase from alpha distance 11/13.

B.6 Conclusion

It is known that one can distinguish bus activity generated from signal transitions having different HD. In this article we put forward the hypothesis that layout dependent phenomena, such as inductance and capacitance in and between conductors and radiation properties of circuit elements, can explain why it sometimes is possible to distinguish transition patterns with the same HD. In this article we provide a general model for layout dependent phenomena and study some security implications of the capacitive crosstalk between parallel wires. Our simulations show that capacitive crosstalk has a significant effect on gate energy dissipation. Our results confirm that the dissipated energy from CMOS switching gates depend not only on the HD, but also on the direction of switching activity on nearby data lines. For an 8 bit bus, this increases the number of possible energy levels from 9 (HD) to 93 (crosstalk), and therefore allows us to explain why signals with the same HD sometimes can be distinguished. Where as an HD based detector can provide about 2.5 bits of information per sample, a crosstalk based detector will yield about 5.7 bits (theoretical) or 5.0 bits (simulated) of information per sample - in all cases for an 8 bit bus. Thus we have shown that a layout specific phenomenon (capacitance) must be considered when analyzing security implications of electromagnetic side channels.

B.7 Bibliography

[1] AGRAWAL, D., ARCHAMBEAULT, B., RAO, J. R., AND ROHATGI, P. The EM Side-Channel(s). In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 29–45. doi:10.1007/3-540-36400-5_4. 113

[2] AGRAWAL, D., BAKTIR, S., KARAKOYUNLU, D., ROHATGI, P., AND SUNAR, B. Trojan Detection using IC Fingerprinting. In *IEEE Symposium on Security and Privacy (SP'07)* (2007), pp. 296–310. doi:10.1109/SP.2007.36. 71, 113

[3] CHARI, S., RAO, J. R., AND ROHATGI, P. Template Attacks. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'02)* (2003), vol. 2523 of *Lecture Notes in Computer Science*, Springer, pp. 13–28. doi:10.1007/3-540-36400-5_3. 17, 71, 113, 118

[4] CHEN, Z., HAIDER, S., AND SCHAUMONT, P. Side-Channel Leakage in Masked Circuits Caused by Higher-Order Circuit Effects. In *Proceedings of the 3rd International Conference on Information Security and Assurance (ISA'09)* (2009), vol. 5576 of *Lecture Notes of Computer Science*, Springer, pp. 327–336. doi:10.1007/978-3-642-02617-1_34. 71, 113

[5] DUAN, C., CALLE, V. H. C., AND KHATRI, S. P. Efficient On-Chip Crosstalk Avoidance CODEC Design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems 17*, 4 (April 2009), 551–560. doi:10.1109/TVLSI.2008.2005313. 71, 72, 73, 74, 114, 117, 118, 120

[6] DUDA, R. O., HART, P. E., AND STORK, D. G. *Pattern Classification*. John Wiley and Sons, Inc, 2001. 123

[7] DYRKOLBOTN, G. O., AND SNEKKENES, E. A Wireless Covert Channel on Smart Cards (Short Paper). In *Proceeding of the International Conference on Information Systems (ICIS'06)* (2006), vol. 4307 of *Lecture Notes in Computer Science*, Springer, pp. 249–259. doi:10.1007/11935308_18. 71, 113

[8] DYRKOLBOTN, G. O., AND SNEKKENES, E. Modified Template Attack Detecting Address Bus Signals of Equal Hamming Weight. In *Proceedings of the Norwegian Information Security Conference (NISK'09)* (2009), pp. 43–56. 71, 113, 123

[9] GANDOLFI, K., MOURTEL, C., AND OLIVIER, F. Electromagnetic Analysis: Concrete Results. In *Proceedings of the 3th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'01)* (2001), vol. 2162 of *Lecture Notes in Computer Science*, Springer, pp. 251–261. doi:10.1007/3-540-44709-1_21. 17, 113

[10] KOCHER, P. C., JAFFE, J., AND JUN, B. Differential Power Analysis. In *Proceedings of Advances in Cryptology (CRYPTO'99)* (1999), vol. 1666 of *Lecture Notes in Computer Science*, Springer, pp. 388–397. doi:10.1007/3-540-48405-1_25. 17, 71, 113

[11] MANGARD, S., OSWALD, E., AND POPP, T. *Power Analysis Attacks - Revealing the Secrets of Smart Cards*. Springer, 2007. 17, 71, 113

[12] MOLL, F., ROCA, M., AND ISERN, E. Analysis of Dissipation Energy of Switching Digital CMOS Gates with Coupled Outputs. *Microelectronics Journal 34*, 9 (September 2003), 833–842. doi:10.1016/S0026-2692(03)00133-2. 71, 72, 74, 114, 115, 116, 117, 118, 120

[13] QUISQUATER, J.-J., AND SAMYDE, D. Automatic Code Recognition for Smart Cards Using a Kohonen Neural Network. In *Proceedings of the 5th Smart Card Research and Advanced Application Conference* (2002), vol. 5, USENIX. 71, 113

[14] SOTIRIADIS, P. P., AND CHANDRAKASAN, A. Low Power Bus Coding Techniques Considering Inter-wire Capacitances. In *Proceedings of the IEEE Custom Integrated Curcuits Conference (CICC'00)* (2000), pp. 507–510. doi:10.1109/CICC.2000.852719. 71, 114, 118

Appendix C

A Different Probabilistic Modelof the TRNG[1]

Abstract

An analytical model of a true random number generator (TRNG) based on several ring oscillators (ROs) is given in this Appendix, in which the guaranteed randomness rate is computed. The model is based on the probability of a hit in the region of a transition in an RO. The size of this region is related to the standard deviation of accumulated jitter following a normal or Gaussian distribution. With this model, a maximum number of ROs can be calculated based on the properties of the device and the TRNG design.

Compared to the model proposed in [3], this model is theoretical and analytical, but it is also more conservative in the sense that it considers only jitter accumulated during a single sampling period. One of the main points with the model from [3], is that when an RO is sampled outside of the region defined by accumulated jitter or in the deterministic region, the uncertainty of the position of a transition of the RO signal due to the accumulation of jitter is not removed by the sampling since the exact position of the transition is not known. The result is that the accumulated jitter increases during several sampling periods until a hit is registered, and that the number of rings necessary to achieve at least one hit at every sampling point is lower than the result of the theoretical probabilistic model given below.

C.1 Probabilistic Model

A ring oscillator (RO) consists of an odd number of inverters connected in a ring structure producing a square wave signal whose frequency is related to the delay through the inverters including the routing delay. Due to the presence of jitter, the exact position in time of a transition from the RO output is unpredictable, where a transition is defined as a change in the output voltage from the logic zero to the logic one or vice versa. Based on the standard deviation of Gaussian jitter, a probabilistic approach can be used. The probability of hitting a transition with respect to a single RO is proportional to the ratio $\sigma_s/(0.5 \cdot T)$, where σ_s is the standard deviation of accumulated jitter during a sampling period T_s or since the last previous sampling of the same RO, and T is the period of the RO. If this ratio is larger than 1, the hit probability is equal to 1. The proportionality constant determines a lower bound on the entropy of the obtained output bit under the assumption that jitter follows the normal or Gaussian distribution. For example, for the bit entropy $H \approx 0.97$, the proportionality constant is around 0.5, see [2].

The challenge is to prove that at any time an output bit has entropy at least H when conditioned on the previous output bits. The underlying assumption is that the output bits of the different ROs are mutually statistically independent. Since the output bit is the XOR of r RO output bits, the entropy of the output bit will be at least H if at least one of r RO output bits at a sampling point has entropy H independently of the previous sampled output bit of the same RO. This will be satisfied if the corresponding RO output bit has entropy H independently of the phase of the RO signal at the previous sampling time, because the value of the previous sampled RO output bit is uniquely determined by the phase. Altogether, this happens if at least one of the r ROs is sampled in the transition

[1] Jovan Dj. Golić and Knut Wold, 2011.

region of width proportional to σ_s. The hit probability for a single RO should be divided by $0.5 \cdot T$, because the falling or rising edges in the oscillating signal occur with the expected period $T/2$. Compared to the proposal of Sunar et al. [2], this new model takes into account not only the rising edge of the RO output but also the falling edge, which naturally also contributes to randomness.

Let the hit probability for a single RO be denoted:

$$p = c \cdot \frac{\sigma_s}{0.5 \cdot T} \tag{C.1}$$

where the constant c guarantees the entropy H. The standard deviation of accumulated jitter is approximately given as

$$\sigma_s \approx \sigma_0 \sqrt{\frac{T_s}{0.5 \cdot T}} \tag{C.2}$$

where $\sigma_0 = \sigma_{T/2}$ is the elementary standard deviation of jitter of the next transition edge given the position of the current transition edge, according to the alternating renewal process mentioned in [1]. Under the RO independence assumption or, more precisely, under the assumption that the phases of the RO signals at the previous sampling time are distributed uniformly, the hit probability P_r for r ROs is given as:

$$P_r = 1 - (1 - c \cdot \frac{\sigma_s}{0.5 \cdot T})^r = 1 - (1 - p)^r. \tag{C.3}$$

In case of different hit probabilities p_i of the individual ROs, the probabilistic model generalizes to:

$$P_r = 1 - \prod_{i=1}^{r} (1 - p_i). \tag{C.4}$$

Based on this model, the number of ROs can be determined depending on the sampling frequency and the standard deviation of jitter.

The hit probability P_r means that a proportion of P_r output bits from the true random number generator (TRNG) at unknown positions will have entropy at least H, independently of other output bits, whereas a portion of the remaining $1 - P_r$ output bits are not guaranteed to have entropy at least H. P_r can be called the guaranteed randomness rate of a TRNG consisting of several ROs. The output bits at unknown positions that are not guaranteed to be purely random should be taken care of by a suitable post-processor. In order to remove the need for the post-processor, it is required that P_r is close to 1, but this requires a large number of ROs:

$$r \approx -\frac{ln(p)}{p}. \tag{C.5}$$

Using this model, the number of ROs necessary to achieve a guaranteed randomness rate can be calculated for different configurations of a TRNG based on several ROs by using:

$$r = \frac{ln(1 - P_r)}{ln(1 - p)} \tag{C.6}$$

where P_r is the hit probability and p is given by:

$$p = \frac{\sigma_T}{T} \cdot \sqrt{\frac{T_s}{T}} \tag{C.7}$$

where the proportional constant c is set to 0.5 and σ_T is accumulated jitter during one RO period T.

In the case from Sunar et al. [2], $\sigma_T/T = 0.02$ and $T \approx T_s$ and for a guaranteed randomness rate of $P_r \approx 0.6$ the number of ROs is calculated to $r = 46$. This is a reduction compared to [2] where the number of rings was calculated to 114 for the same P_r.

In the case with 3 inverters in the ring and a sampling frequency of 100MHz, the number of rings for $P_r = 0.6$ is 15 when the standard deviation of jitter of one gate or inverter is set to 30ps. For $P_r \approx 1$, the number of rings is 45. If the sampling frequency is increased to 300MHz, the number of rings for $P_r = 0.6$ is 25, and for $P_r \approx 1$, the number of rings is 94.

Acknowledgment

The author wishes to thank Jovan Dj. Golić for very valuable and useful contributions regarding this model.

C.2 Bibliography

[1] BAUDET, M., LUBICZ, D., MICOLOD, J., AND TASSIAUX, A. On the Security of Oscillator-Based Random Number Generators. *Journal of Cryptology 24*, 2 (2011), pp. 398–425. doi:10.1007/s00145-010-9089-3. 23, 126

[2] SUNAR, B., MARTIN, W. J., AND STINSON, D. R. A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. *IEEE Transactions on Computers 56*, 1 (2007), 109–119. doi:10.1109/TC.2007.250627. 4, 11, 12, 14, 15, 19, 20, 21, 30, 31, 32, 34, 37, 39, 41, 45, 53, 58, 63, 77, 78, 83, 87, 101, 102, 103, 105, 106, 108, 109, 125, 126

[3] WOLD, K., AND PETROVIĆ, S. Behavioral Model of TRNG Based on Oscillator Rings Implemented in FPGA. In *Proceedings of the 14th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS'11)* (2011), pp. pp. 163–166. 4, 21, 125

Nomenclature

AES Advanced Encryption Standard

ASIC Application Specific Integrated Circuit

CASR Cellular Automata Shift Register

CLB Configurable Logic Block

CMOS Complementary Metal-Oxide-Semiconductor

CPLD Complex Programmable Logic Device

CPU Central Processing Unit

CRC Cyclic Redundancy Check

DFT Discrete Fourier Transform

DSP Digital Signal Processor

EMI ElectroMagnetic Interference

EPROM Erasable Programmable Read Only Memory

FFT Fast Fourier Transform

FIRO FIbonacci Ring Oscillator

FM Frequency Modulation

FPGA Field Programmable Gate Array

GARO GAlois Ring Oscillator

GND GrouND

HD Hamming Distance

HDL Hardware Description Language

IP Internet Protocol

IV Initial Vector

JTAG Joint Test Action Group

LAB Logic Array Block

LE Logic Element

LFSR Linear Feedback Shift Register

LUT Look-Up Table

MOSFET Metal-Oxide-Semiconductor Field-Effect Transistor

OTP One-Time-Programmable

P&R Place and Route

PLD Programmable Logic Device

PLL Phase-Lock Loop

PRNG Pseudo Random Number Generator

RNG Random Number Generator

RO Ring Oscillator

SPI Serial Peripheral Interface

SPLD Simple Programmable Logic Device

SRAM Static Random-Access Memory

TERO Transition Effect Ring Oscillator

TRNG True Random Number Generator

VHDL Very high speed integrated circuit Hardware Description Language

XOR eXclusive OR

Index