ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

STUDIES ON THE DESIGN OF ROBUST AND FULLY-DIGITAL RANDOM NUMBER GENERATORS

M.Sc. THESIS

Burak ACAR

Department of Electronics and Communication Engineering

Electronics Engineering Programme

OCTOBER 2020

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

STUDIES ON THE DESIGN OF ROBUST AND FULLY-DIGITAL RANDOM NUMBER GENERATORS

M.Sc. THESIS

Burak ACAR
(504171246)

Department of Electronics and Communication Engineering

Electronics Engineering Programme

Thesis Advisor: Dr. Tufan Coşkun KARALAR

OCTOBER 2020

ISTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

TAMAMEN SAYISAL GÜVENLİ RASTGELE SAYI ÜRETECİ TASARIMI ÜZERİNE ÇALIŞMALAR

YÜKSEK LİSANS TEZİ

Öğrenci Burak ACAR
(504171246)

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı

Elektronik Mühendisliği Programı

Tez Danışmanı: Dr. Tufan Coşkun KARALAR

EKİM 2020

Burak ACAR, a M.Sc. student of ITU Graduate School of Science Engineering and Technology student ID 504171246, successfully defended the thesis/dissertation entitled "STUDIES ON THE DESIGN OF ROBUST AND FULLY-DIGITAL RANDOM NUMBER GENERATORS", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor : **Dr. Tufan Coşkun KARALAR**
Istanbul Technical University

Jury Members : **Prof. Dr. Ece Olcay GÜNEŞ**
Istanbul Technical University

Dr. Ali Cengiz BEĞEN
Ozyegin University

Date of Submission : 24 September 2020
Date of Defense : 2 October 2020

To my family,

FOREWORD

I would like to thank my supervisor Dr. Tufan Coşkun Karalar for his great assistance in my research.

I would also like to thank Dr. Salih Ergün, who is my advisor in my work at TÜBİTAK BİLGEM, for sparking me up to research and write academic papers on the topic of the random number generators.

I am very grateful to TÜTEL (Integrated Circuit Design and Education Laboratory) not only for its support of hardware and design tool, but also its high motivating working environment during my M.Sc. education and M.Sc. thesis. I would like to thank my colleagues here for their great help.

I would like to thank all my teachers who have contributed to my undergraduate and graduate life at ITU, and added vision to my point of view.

Finally, I present my endless gratitude to my dear family, who did not withhold love from me and continuously supported me during my thesis project as all in my life.

October 2020

Burak ACAR
(Electronics Engineer)

TABLE OF CONTENTS

Page

ABBREVIATIONS

ASIC : Application Specific Integrated Circuit
CMOS : Complementary Metal–Oxide–Semiconductor
DRC : Design Rule Check
FF : Flip Flop
FIFO : First In First Out
FIPS : Federal Information Processing Standards
FPGA : Field Programmable Gate Array
IP : Intellectual Property
IoT : Internet of Things
IoV : Internet of Vehicles
LEC : Layout Equalavance Check
LUT : Look-Up Table
LVS : Layout Versus Schematic
MATLAB : Matrix Laboratory
MEMS : Micro-ElectroMechanical Systems
NIST : National Institute of Standards and Technology
PCI : Peripheral Component Interconnect
PRNG : Pseudo Random Number Generator
PUF : Physically Unclonable Function
PWM : Pulse Width Modulation
RNG : Random Number Generators
RO : Ring Oscillator
SR : Set Reset
TERO : Transient Effect Ring Oscillator
TRNG : True Random Number Generators
XOR : Exclusive Or
VHDL : Very High Speed Integrated Circuit Hardware Description Language

LIST OF TABLES

Page

LIST OF FIGURES

Page

STUDIES ON THE DESIGN OF ROBUST AND FULLY-DIGITAL RANDOM NUMBER GENERATORS

SUMMARY

The random number generator is a module that produces unpredictable bit sequences with various statistical properties. Multiple tools such as dice and metal coins have been used since ancient times to make random choices. Today, even in a football game, a coin is used to decide which team will start the game. Random number generation is one of today's popular topics in information security. Random Number Generators (RNGs) are at the core of cryptosystems with producing unpredictable secret keys. The weaknesses of the random number generator of a cryptosystem can cause confidential data to fall into undesirable third parties' hands. Therefore, random number generation mechanisms should be based on physical entropy sources. RNGs are used not only for cryptography but also in Monte Carlo simulations, learning algorithms, science, art, gaming, scrambling, etc.

Random number generators are classified into two main groups, Pseudo Random Number Generators (PRNGs) and True Random Number Generators (TRNGs). PRNGs generate new values from seed using certain functions. These bit sequences, which have a statistically random appearance, repeat themselves after a certain period of time. Therefore, the generated bit sequences are periodic and predictable. On the other hand, true random number generators use physical quantities as an entropy source. They generate random numbers that do not repeat themselves, where the generated bit is independent of the previous bit.

Random number generators can be implemented using software or hardware. Hardware based random number generators can be implemented on an FPGA platform or ASIC chip. Designers prefer FPGA platforms because of their features, such as rapid prototyping, reprogrammable, low production cost. Besides, FPGA platforms have low noise sources since they are designed to work in digital circuits. Therefore, generating random numbers on an FPGA requires some design requirements. The designs made as ASIC allow the entropy source to be designed with the desired features more easily. ASIC designs are more preferred in designs to be mass-produced due to their high production costs and high design costs. Random number generators are designed on FPGA or ASIC according to the desired features and application restrictions.

In this study, some state-of-the-art designs of TRNGs especially implemented digitally are examined, and some new improvements and design techniques are proposed. First of all, a three-dimensional no-equilibrium chaos-based RNG published in the literature is implemented on an FPGA. The master-slave synchronization method is tried to be achieved by implementing a clone of the target circuit on the FPGA. Although the initial conditions of the attacked circuit are unknown, it is experimentally shown that the same bit sequence can be produced in the clone circuit as a result of listening to an output of the attack circuit. Thus, it is

shown that RNGs based on deterministic sources are vulnerable to possible attacks and should not be used without any noise source.

A ring oscillator-based random number generator cannot be used in applications requiring low power consumption due to their high power consumption. For low power consumption, a random number generation technique is proposed by engaging and disengaging SR latch structures at specified intervals. The fact that the digital gates used are active only at certain times and require fewer digital gates has reduced power consumption. Immunity of the SR latch structure against correlation-based attacks was investigated in the earlier studies. The method of sampling this structure at irregular time was also proposed in the previous studies. In this study, an ASIC version of these structures is explained.

A fully digital random number generator with low power consumption, which can be used for different applications of different chips, is designed as an ASIC in TSMC 180 nm technology. The created custom IP occupies 120 μm * 114 μm. This study is important for summarizing current works in digital random number generators, for comparing their security and performance.

TAMAMEN SAYISAL GÜVENLİ RASTGELE SAYI ÜRETECİ TASARIMI ÜZERİNE ÇALIŞMALAR

ÖZET

Rastgele sayı üreteci çeşitli istatiksel özelliklere sahip, tahmin edilemeyen bit dizileri üreten bir modüldür. Eski zamanlardan beri rastgele seçimler yapmak için zar ve metal paralar gibi çeşitli araçlar kullanılmıştır. Günümüzde bir futbol müsabakasında dahi hangi takımın oyuna başlayacağına karar vermek için bir bozuk para kullanılır. Rastgele sayı üretimi, günümüzde bilgi güvenliğinin popüler konularından biridir. Rastgele sayı üreteçleri tahmin edilemeyen anahtar üretmesi nedeni ile bilgi güvenliği sistemlerinin temel taşıdır. Bir kripto sistemin rastgele sayı üretecinin zaafları, gizli verilerin istenilmeyen üçüncü kişilere aktarılmasına sebep olabilir. Bu nedenle rastgele sayı üretim mekanizmaları fiziksel entropi kaynaklarına dayandırılmalıdır. Rastgele sayı üreteçleri sadece kriptografi için değil aynı zamanda Monte Carlo simülasyonları, öğrenme algoritmaları, bilim, sanat, oyun, karıştırma, vb. alanlarda da kullanılır.

Rastgele sayı üreteçleri basit rastgele sayı üreteci ve gerçek rastgele sayı üreteçleri olmak üzere iki ana grupta sınıflandırılır. Basit rastgele sayı üreteçleri belirli fonksiyonlarla bir ilk değerden yeni değerler üretilir. İstatiksel olarak rastgele görünüme sahip olan bu bit dizileri belirli bir süre sonunda kendilerini tekrar ederler. Dolayısıyla üretilen bit dizileri periyodiktir ve tahmin edilebilir. Gerçek rastgele sayı üreteçleri ise entropi kaynağı olarak gerçek fiziksel kaynakları kullanır. Oluşturulan bitin önceki bitten bağımsız olduğu, kendilerini tekrarlamayan rastgele sayılar oluştururlar. Bir rastgele sayı üretecinin gerçek rastgele sayı üreteci olabilmesi için uluslararası otoriteler tarafından belirlenen birtakım standart testlerden başarılı olması gerekir. Ancak, testleri geçen her rastgele sayı üreteci, bunun bir gerçek rastgele sayı üreteci olduğunu göstermez. Testler yalnızca bu rastgele sayı üretecinin rastgele sayılar üretmediğini ve bir gerçek rastgele sayı üreteci olmadığını kanıtlar.

Rastgele sayı üreteçleri yazılım veya donanım olarak gerçeklenebilir. Donanımsal rastgele sayı üreteçleri FPGA platformunda veya ASIC çip olarak gerçeklenebilir. FPGA platformu hızlı prototipleme, tekrar tekrar programlanabilme, düşük üretim maliyeti gibi özellikleri nedeniyle tasarımcılar tarafından tercih edilir. Bunun yanında FPGA platformları sayısal devrelerde çalışması için tasarlandığından düşük gürültü kaynağına sahiptir. Dolayısıyla FPGA üzerinde rastgele sayı üretmek birtakım tasarım gereksinimleri gerektirir. ASIC olarak yapılan tasarımlar ise entropi kaynağının daha kolay istenilen özelliklerde tasarlanabilmesine imkân sağlar. ASIC tasarımlar üretim maliyetlerinin yüksek olması sebebi ve yüksek tasarım maliyeti nedeniyle seri üretim yapılacak tasarımlarda daha çok tercih edilir. İstenilen özelliklere ve uygulama kısıtlamalarına göre rastgele sayı üreteçleri FPGA üzerinde veya ASIC olarak tasarlanır.

Bu çalışmada literatürde popüler olan bazı tamamen sayısal rastgele sayı üreteci çalışmaları incelenmiş ve bazı yeni geliştirmeler ve teknikler önerilmiştir. Öncelikle,

daha önce literatürde yayımlanan üç boyutlu denge noktası olmayan bir kaos tabanlı RNG, bir FPGA üzerinde gerçeklenmiştir. FPGA üzerinde aynı devrenin bir kopyası uygulanarak iki devrenin senkronizasyonu sağlanmaya çalışılmıştır. Saldırıya uğrayan devrenin başlangıç koşulları bilinmemekle birlikte, aynı bit dizisinin saldırı devresinin sadece bir çıkışının dinlenmesinin bir sonucu olarak klon devresinde de üretilebileceği deneysel olarak gösterilmiştir. Böylece deterministik kaynaklara dayalı RNG'lerin olası saldırılara karşı savunmasız olduğu ve herhangi bir gürültü kaynağı olmadan kullanılmaması gerektiği gösterilmiştir.

Halka osilatör tabanlı rastgele sayı üreteci, yüksek güç tüketiminden dolayı düşük güç tüketimi gerektiren uygulamalarda kullanılamaz. Bu çalışmada düşük güç tüketimi için, belirli aralıklarla SR mandal yapıları devreye alınarak ve devre dışı bırakılarak rastgele bir sayı üretme tekniği kullanılmıştır. Kullanılan sayısal kapıların sadece belirli zamanlarda aktif olması ve daha az dijital kapı gerektirmesi güç tüketimini azaltır. SR mandal yapısının korelasyon tabanlı saldırılara karşı bağışıklığı önceki çalışmalarda araştırılmıştır. Bu yapıyı düzensiz zamanda örnekleme yöntemi de önceki çalışmalarda önerilmiştir. Bu çalışmada, SR mandallı yapının tümdevre gerçeklemesi yapılmıştır. Daha önce FPGA'da doğrulanan yapı, 180 nm TSMC teknolojisi kullanılarak tamamen sayısal olarak ASIC olarak tasarlanmıştır.

SR mandalları FPGA veya ASIC üzerinde gerçeleme bölgesine göre kötü bir entropi kaynağı olarak davranabilir. Bu tür durumlara karşı, SR mandallarının sayısı yüksek tutulur. Her bir SR mandal hücresinin çıktısını XOR işlemine sokulur. Böylece entropi artırılır. Bu çalışmada, alan ve güç tasarrufu için SR mandallarının sayısı en aza indirilmeye çalışılmıştır. SR mandallarının sayısında FPGA üzerinde yapılan denemeler baz alınmıştır. Ancak gelecekteki bir çalışma olarak çip üretiminden sonra daha ayrıntılı analizlere yer verilecektir. Kritik uygulamalarda, SR mandallarının sayısı oldukça yüksek seçilebilir. Böylece bazı SR mandallarının FPGA üzerinde zayıf uygulanmasıyla bile sistem hâlâ iyi bir şekilde çalışacaktır. Ayrıca önerilen RNG daha önce düşük teknoloji düğümlerinde üretilmiş modern bir FPGA üzerinde gerçeklenmiştir ve tüm rastgelelik testlerinden başarı ile geçmiştir. Düşük teknolojilere geçildikçe gürültü miktarı da azalmaktadır. Dolayısıyla, önerilen RNG 180 nm'lik teknolojide bir ASIC olarak uygulandığında, entropi daha yüksek olacak ve metastabilite daha baskın olacaktır. Bu nedenle, 180 nm'de ASIC olarak tasarlanan RNG'nin entropi bakımından daha güvenli olacağı düşünülmektedir.

Sayısal bir tümdevre tasarımının bir fikirden çipe giden yolculuğu bu çalışmada özetlenmiştir. ASIC tasarım akışı, bir tasarım konseptiyle başlar. Bu aşamada bir soruna çözüm sunan bir fikir düşünülerek kabaca bir blok diyagram oluşturulur. İstenilen özellikler, kullanılacak mimari, kullanılacak arayüzler, kısıtlamalar (zaman, alan, güç, hız) belirlenir. Daha sonra bu tasarım konsepti, sayısal araçların anlayabilmesi için RTL koduna dönüştürülmelidir. VHDL, Verilog veya Sistem Verilog dili donanım tasarım dili olarak kullanılabilir. İkinci adım, RTL kodunu, çip üretim fabrikasının anlayabileceği ve kullanıcıların kendi tümdevresini (GDSII düzeni olan) ürettirebileceği bir formata dönüştürmektir. Bazı durumlarda, kullanıcılar RTL kodunu yazmadan önce yazılım modeli oluşturabilir. Ayrıca çoğu zaman kullanıcılar üretime gitmeden ve üretim hizmetlerine çok para harcamadan önce tasarımlarını FPGA platformunda prototip olarak denerler.

RTL tasarımı hazır olduktan ve tasarımın fiziksel bir sistemde düzgün çalışıp çalışmadığı kontrol edildikten sonra RTL kodunun çip üretici firmaların anlayabileceği bir formata yani GDSII'ye dönüştürülmesi gerekir. Bir RTL kodunu

bir GDSII dosyasına dönüştürme akışının üç ana adımı vardır. İlk olarak, RTL kodu sentezlenir ve teknolojiye özgü hücrelerden oluşan bir dosya formatına dönüştürülür. İkinci olarak, bu teknolojiye özgü hücreler, tümdevre tasarımının herhangi bir fiziksel yönü dikkate alınarak fiziksel olarak gerçeklenir. Son adım olarak, doğrulama işlemi yapılır. Doğrulama aşaması tümdevre tasarımlarını üretime göndermeden önce hataları önceden fark edebilmek açısından kullanışlıdır. Doğrulama aşaması, üretimden sonra çalışmayan çiplerini geri almak istemeyen tasarımcılar için çok önemlidir. Doğrulama aşaması doğru bir şekilde yapılırsa, fazladan üretim gerektirmediği için zaman ve para tasarrufu sağlayabilir.

Bu çalışmada farklı yongaların farklı uygulamaları için kullanılabilen, düşük güç tüketimine sahip tamamen sayısal bir rastgele sayı üreteci, TSMC 180 nm teknolojisinde ASIC olarak tasarlanmıştır. Tasarlanan özel IP 120 µm * 114 µm yer kaplamaktadır. Daha önce FPGA üzerinde başarılı sonuçlar alınan 3,125 Mb/sn bit üretim hızı hedeflenmiştir. Önerilen tasarımın toplam alanı 13623 μm^2'dir. Ortalama koşullarda (25 C, 1,8 V besleme, tipik işlem) anahtarlama etkinliğini %20 alırken toplam güç tüketimi 0,44 mW'tır. Bu çalışma, sayısal rastgele sayı üreticilerindeki güncel çalışmaları özetlemek, güvenliklerini ve performanslarını karşılaştırmak için önemlidir. Gelecekteki bir çalışma olarak, önerilen RNG yapısı tümdevre olarak üretilecek ve çip üzerinden rastgele bitler okunarak rastgelelik testleri yapılacaktır. Üretim sonrası ölçümlere göre bit üretim hızı arttırılabilir.

1. INTRODUCTION

Nowadays, there is a significant change in technology. Mobile computing, Internet of Things (IoT) and Internet of Vehicles (IoV) have become a new era of high technology. Smart devices replace most of the consumer electronic devices, and they are connected to each other via a local network or internet. The data collected from these sensor nodes helps to facilitate the daily life of people. Although these developments seem like good news about human life, connecting all devices in the same network may cause some security risks. An attacker can access to personal assets if enough countermeasures are not taken for the aspect of security. Cryptographic systems are commonly employed for hiding information from unwanted persons. A cryptographic system is made up of basically two elements: 1) an encryption/decryption algorithm, 2) a random number generator. RNGs are used for providing key values for the ciphering algorithms, data/entity authentication, masking of power consumption, storing secret keys (PUF), hashing, etc. Since the security of a cryptographic algorithm depends on the private keys' unpredictability, the topic of random number generators has become a critical issue in information security. A weak RNG can make even the strongest cryptographic system fail, so the RNG output should be unpredictable, should not be regenerated, and must satisfy all statistical tests of randomness [1]. Furthermore, the trend towards low area, low power consumption and low-cost electronic devices impose stringent requirements on the design and implementation of RNGs.

This thesis aims to analyze some state-of-the-art digital RNG methods in the literature especially implemented digitally and propose new technics with giving technical backgrounds and design approaches. This study is important in terms of summarizing current studies in this field, comparing their security and performance, and suggesting new techniques by giving details of ASIC implementation.

The rest of the dissertation is organized as follows: general information about random number generators are given in section 2; digital integrated circuit design flow is explained in section 3; experimental cryptanalysis of a chaotic system based

RNG is given in section 4; an ASIC design of a robust transient effect based RNG is proposed in section 5; finally, conclusions are given in the last section.

2. RANDOM NUMBER GENERATORS

2.1 Classification of RNGs

Randomness is all about an event that is unpredictable amongst a series of events. Although it seems basic, humans are really bad at making a random selection. For instance, when people are asked to select a random number between 1 to 10, most of the people choose the number of 7 [2]. That is not random at all. Thus, it becomes clear that a kind of randomness production machine is needed to encrypt personal bank details or create fair prize draws.

According to the famous Kerckhoffs' principle [3], the security of a cryptographic system depends on the secret key's unpredictability rather than the confidentiality of the cryptographic algorithm. Secret keys can be used as long-term keys, round keys for block ciphers, temporal keys for public-key algorithms, or stream ciphers in cryptography to provide security. Random number generators play a critical role in a secret key generation [4], authentication protocols [5], one-time pad generation, masking values for countermeasures against side-channel attacks, etc. [6]. Random number generators are utilized in wide application areas such as Monte Carlo simulations [7], numerical analysis [8], cryptocurrency [9], digital communication [10], science [11], art [12], gaming and gambling [13], etc.

Randomness exists in nature. Radioactive material decays by releasing electrons and the time between consecutive electrons being released completely random. However, finding a physical randomness entropy source and combining them with other systems is not so easy that most of the time relying on some kind of algorithms is preferred. These algorithms are deterministic that the produced data repeat themselves after a while. In literature, these deterministic random number generators are called Pseudo Random Number Generator (PRNG) [14]. They take an initial number known as seed, and they combined it with various hidden and shifting inputs

such as the internal clock of the machines. They broaden them into random bitstreams that appear impossible to estimate. Although the generated bitstream has good statistical properties, it can be predicted and reproduced according to implementing deterministic functions.

Attackers can potentially estimate the outputs of a non-complex PRNG algorithm with reverse engineering. This is significantly much easier when the attackers can access to the physical machine. PRNGs are used in slot machines. They use a different seed each time. Even though the attackers understand how the PRNG functions of the machines work, hackers should also analyze the machine's gameplay to recognize a pattern. Knowing the path gives attackers the ability to predict the next pattern. In recent years, PRNG attackers have been arrested in the U.S from Singapore [15]. They report slot machines activities with a candid video camera to a mastermind in Russia. The mastermind attacker inputs the data to a program designed to establish future times for jackpots. The player uses a mobile application that vibrates when the application predicts the jackpot. Once again, this attack revealed that undeterministic True Random Number Generators (TRNGs) should be used in areas requiring information security instead of deterministic PRNGs.

TRNGs implement non-deterministic functions using physical noise such as phase noise, shot noise, radioactive decay, etc. [16]. For saying a bitstream is truly random, it requires three main properties. The same random sequence should not be produced again. The generated sequence has good statistical properties. Finally, the produced bitstream should be unpredictable.

Random Number Generators can be based on hardware or software implementation. Hardware-based RNGs may also be based on fully digital blocks or analog blocks. The key criteria for determining which type of RNG and method of implementation can be used are complexity, robustness against potential attacks, area and power usage, and throughput.

TRNGs can be implemented on Application Specific Integrated Circuit (ASIC) chips using various Complementary Metal–Oxide–Semiconductor (CMOS) / Micro-Electro-Mechanical Systems (MEMS) process technologies. Analog design ability

gives designers a big chance to obtain high qualified randomness using noise sources. On the other hand, FPGA platforms are more attractive to designers. They allow fast prototyping and have fewer production costs. Moreover, they can be configured as a new design after the production of the FPGA chips. Their reconfigurability property lowers the final product cost, and they can be easily combined with other digital designs. Nevertheless, FPGAs are produced to work correctly with high performance, and device manufacturers aim to minimize uncertainties. Therefore, their noise source is limited, post-processing methods such as Von Neumann [17] are typically used to ensure randomness.

In the development process of a TRNG, verification of randomness is one of the most imperative parts of the design process. The validity of a TRNG can be checked using statistical test suits published by institutes such as the NIST team. There are several statistical tests in the literature used for randomness. If the bit sequence generated by a TRNG does not pass the tests, this indicates that the generated bitstream is not random. However, it cannot be said that a sequence passed through the numerical tests is precisely random.

2.2 Literature Review

There are several methods used for designing random number generators in the literature. The method for RNG design is chosen based on the area of usage and user specifications. One of the methods for RNG design is using chaotic systems. Continuous-time chaos-based RNGs have been previously studied in terms of their implementation methods [18], [19], [20] and security [21]. Hardware RNGs exploit entropy from physical processes such as thermal or shot noises, and they can be implemented as analog or digital circuits. Compared to analog RNGs, digital RNGs are preferable due to their easier connection to the cryptographic system's other digital components. However, in the digital domain the generation of entropy typically involves a high number of replicated structures that increase the area and power consumption. Additionally, electromagnetic coupling between adjacent oscillators reduces the RNG randomness performance.

A commonly used entropy source on FPGAs is the phase jitter in a classical ring oscillator (RO). Ring oscillators are composed of a number of rings; each of them has an odd number of inverters and use timing jitter of oscillators as an entropy source. The outputs of the ring oscillators are then merged using an XOR gate. There is a need for extra post-processing methods like Von Neumann corrector [17] if the output of the combined data does not fulfill the tests to raise the quality of the randomness. Fairfield et al. proposed a ring oscillator based random number generator in which a D-type flip flop samples a high-frequency oscillator at the rising edge of the slower clock [22].

Because fully digital circuits have restricted entropy sources, it is an absolute necessity to estimate the total entropy of the planned random number generator circuit before beginning to implement the design. Sunar et al. suggested the use of multiple ring oscillators for random number generation [23]. To forecast the quality of the randomness of the constructed circuit, a mathematical model is suggested. It is possible to decide, according to this model, how many ring oscillators and how many inverters should be used to ensure sufficient randomness. As Sunar et al. explained in [23], an interval is separated into k equal sub-items, called an urn. Each transient during an urn causes urn filling. In [23], a ring oscillator with a prime number of inverter is suggested to fill as many urns as possible with XORing without wasting entropy. In the same work, it is also declared that two rings with an equal number of inverters would show a high overlap in the transient regions.

Dichtl et al. reported that for the proposed method in [23] to be valid [24], all ring oscillators should be independent of each other. Wold et al. strengthened the method proposed in [23] by adding an additional flip-flop after each free-running ring oscillator [25]. Sampling each ring oscillator at the same time bypasses the post-processing requirement with the help of extra flip-flops. The authors tested the viability of their approach in [26] and [27]. An ASIC application of Sunar et al. 's proposed model is also explained in [28].

In classical ring oscillator based random number generators, a D-type flip-flop at the end of the circuit samples the timing jitter at a specific frequency. Since adjacent logic cells can affect each other, an attack circuit to the main circuit can capture the

designed circuit's produced random bit outputs by coupling. In [29], a correlation-based attack is proposed to predict a ring oscillator-based RNG's output.

There is a compulsory requirement for new improvements against coupled-based attacks. In [30], the authors proposed using irregular sampling of regular waveform method for ring oscillator based random number generators to be more robust against correlation-attacks. The generated random bitstream passed full NIST 800-22 tests without any post-processing. Sampling regular waveform at an irregular time builds more robust designs against correlation-based attacks compared to a regular sampling of the irregular signal. Furthermore, this new method does not need manual placement by the designers to place ring oscillators separately. Thus, this proposed RNG can be applied to any kind of FPGA boards easily using auto-placement techniques supplied by the design programs. As a result, the authors show the classical ring oscillator based random number generators' drawbacks in terms of security can be eliminated by using an irregular sampling of regular waveform method.

Classical RO based RNGs require a separate location of each RO [31]. By modifying the operating conditions, it is possible to cause the ring oscillators to be locked to each other. Manipulating the supply voltage, as shown in [32], can cause ring oscillators to couple. Bayon et al. also used electromagnetic fault injection to modify ring oscillator operations in FPGA [33]. A new design that can be used as both an RNG or Physical Unclonable Function (PUF) and is resistant to the locking phenomenon was suggested in [34] by Boussuet et al. The proposed PUF takes advantage of the metastable behavior of the TEROs (Transient Effect Ring Oscillators). In essence, a TERO loop is an SR latch whose S and R inputs are linked to the enable signal. A TERO loop includes two paths, and each path contains one AND gate and an odd number of inverters. The number of inverters at each path can be increased to a higher odd number in order to extend the oscillations in the time domain. At the rising edges of enable signal, the transitory oscillations begin, and the output of a TERO loop oscillates until it becomes stable at a logic level of 1 or 0. In theory, the oscillation stops because the loop is not perfectly symmetrical, and there is a time difference Td between the two paths of TERO [35]. On the other hand, it

has been disproved by [36] and [37]. The proposed new model is based on the experience that both edges propagate across a mutually discrete transistor set, scattered on both paths. The PUF proposed by Boussuet et al. counts the number of oscillations until TERO output is constant [34].

The number of oscillations for a TERO cycle is usually distributed around an average value. As a source of entropy, this average amount of oscillations is used. This structure is believed to be robust in locking the phenomenon and electromagnetic attack since it is based on the number of oscillations rather than the frequency of oscillations; nevertheless, it has not been experimentally demonstrated. Also, to monitor the average value of oscillations, a downside of this structure is added to the complexity of monitor, accumulator, and scroll register structures. In [38], the robustness of the TERO-based RNG is studied, and the ASIC implementation of this analysis is discussed in section 5.

The irregular sampling of regular waveform method is also applied to the TERO structure in [39]. This paper presents a more hardware-efficient RNG compared to the structure in [34] and experimentally demonstrates the security of the proposed RNG against correlation-based attacks, as explained previously in [29]. The outputs of an array of TERO elements are XORed to generate an irregular clock signal to sample a regular waveform to produce random bits. Therefore, a more resource-efficient RNG design is accomplished compared to [34]. Furthermore, to study the resistance of the proposed RNG against locking phenomenon, an identical RNG is placed as the attacker in the same LUTs with the target RNG similar to the method described in [29] and correlation-based cryptanalysis is carried out. Unlike classical RO-based RNGs, this novel TERO-based design is experimentally shown to be resistant against coupling attacks. Furthermore, the RNG output bitstream satisfies NIST 800- 22 statistical tests of randomness without any need for post-processing.

A reconfigurable RNG based on TERO is proposed in [40]. This circuit allows users to produce random bits using the same entropy source, using two separate sampling methods. So users may use this RNG with varying levels of protection in their various applications. The proposed TERO based RNG is more hardware efficient than the structure [34] and robust to correlation-based attacks described in [29]. In

the regular sampling of irregular data method, the entropy source is sampled at the rising edges of a slow clock. In the irregular sampling of regular data method, the entropy source generates irregular clock to sample a regular fast clock to produce a random bit. A feedback mechanism fed from online tests is implemented to obtain good statistical results at the RNG output. The proposed structure is firstly implemented on an FPGA and subjected to NIST 800-22 statistical tests. It passes these tests without any need for post-processing and implemented in ASIC finally.

Continuous-time chaos is also utilized for random bit generation. A continuous-time three-dimensional chaos-based RNG is suggested in [41]. In section 4, a master-slave synchronization based attack system for the proposed RNG presented in [41], and the target system's security vulnerabilities are analyzed in detail.

2.3 Statistical Tests

In order to measure the randomness of random numbers produced by the RNGs, a number of tests are conducted in accordance with international standards. In this study, the National Institute of Standards and Technology (NIST) SP 800-22 [42] and Federal Information Processing Standards (FIPS) 140-2 [43] tests were used. In addition, Diehard and TestU01 tests are generally used by designers. To be TRNG, an RNG must pass the tests that comply with these standards. However, not every RNG that has passed the tests shows it to be a TRNG. The tests only prove that an RNG does not generate random numbers and is not a TRNG.

2.3.1 NIST SP 800-22

NIST is a family of tests determined by the US National Institute of Standards and Technology to measure the reliability of random number generators. This test suite includes a total of fifteen statistical tests. In these tests, the bit sequences are analyzed statistically. In order to be able to talk about randomness, the 0 1 ratios of the bit sequence must be within a certain limit. Likewise, the probability of sequencing of 0s and 1s must be within certain probabilities. All these tests measure various properties of the data and help us get an idea of whether it is random or not.

The NIST tests are applied to collected data as follows: For example, a 320000000-bit number string will be tested. First, this bitstream is subdivided. In this study, the collected long data is divided into 320 sub-sequences of 1000000 bits each. Thus, all these sequences are tested separately. The rate of how many of the 320 series passed the test is written on the result printout screen. The tests calculate a statistical value called p-value that determines the degree of randomness. Each p-value gives information about the perfection of the random number generator. For the test to be successful, the p-value must be higher than 0.01. At the bottom of the results page, the success rate required for a successful test is written. Failed tests are marked with a *. According to their own criteria, users can change these tests' limits, but it is appropriate to follow the recommended standards.

2.3.2 FIPS 140-2

RNGs may work differently depending on changing external conditions. According to such situations, RNGs must be tested beforehand in these extreme conditions. Also, some security applications may demand to check whether the generated bit sequences are random while still working. NIST tests are not a solution to such needs since large data is needed for reliable test procedures. In addition, it isn't easy to implement all NIST tests on the hardware and causes a lot of resource consumption. In such cases, FIPS tests are recommended by NIST. FIPS 140-2 specifies four statistical tests for randomness: monobit, poker, runs, and long runs. A single string of 20000 bits in length is sufficient to perform these tests. If any of the tests fail, the random number generator cannot pass the test.

3. DIGITAL INTEGRATED CIRCUIT DESIGN FLOW

An ASIC (Application Specific Integrated Circuit) design journey is a long process, and several processes are followed from an idea to tape-out as given in Figure 3.1. This long journey poses many engineering challenges despite the final product is usually relatively small (measured in nanometers). The slightest error can cause the generated chip to malfunction, and therefore it can cause a massive waste of time and money.

Day by day, various demands such as higher speed, less energy consumption, and lower costs are increasing in electronic circuits. To meet the futuristic demands of chip design, a number of improvements are required in design tools, methodologies, and software/hardware capabilities. Therefore, ASIC design flow adopted by engineers is used for efficient structured ASIC design.

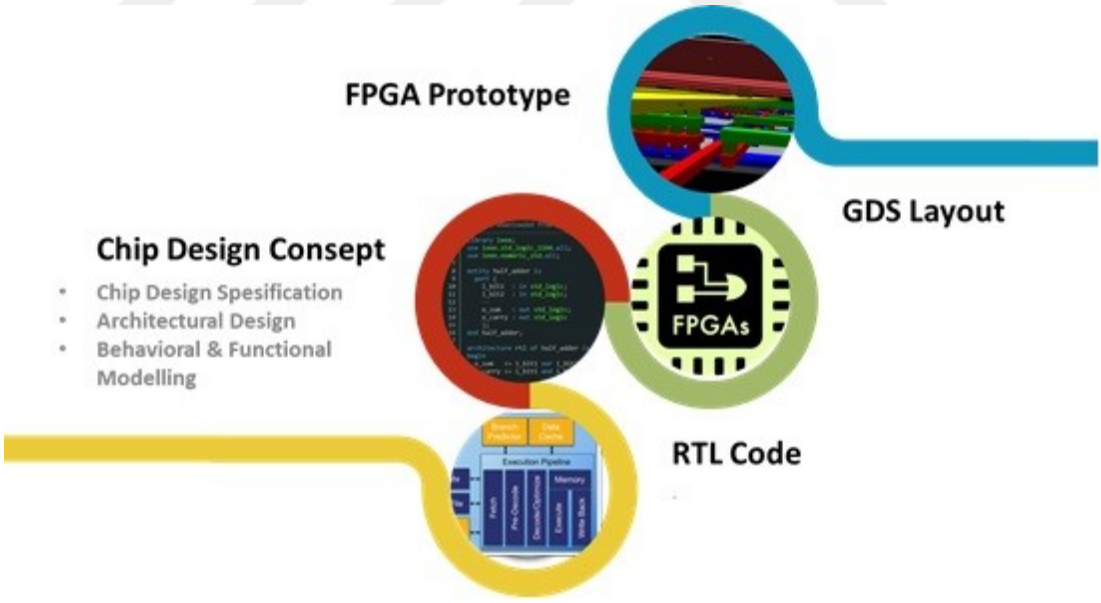


Figure 3.1 : The journey of an idea to a chip.

3.1 Design Concept

ASIC design flow starts with a design concept. At this stage, a roughly block diagram is created by considering an idea that offers a solution to a problem. Desired features, architecture to be used, interfaces to be used, restrictions (time, area, power,

speed) are determined. Then this design concept should be converted into RTL code so that digital tools can understand. VHDL, Verilog, or System Verilog language can be used. The second step is to convert the RTL code into something that foundry can understand and manufacture users' IC which is GDSII layout. There are alternative ways to do things. In some cases, users may write software model before writing RTL code. And also, sometimes users prototype their design on an FPGA platform before going to manufacturing and spend a lot of money on production services.

An RTL code mainly includes three parts. The first part is the module definition in there the outputs and inputs are defined. Thus, how the designed module interfaces with the outside are defined. Then, an RTL code has sequential logic, which can be flip-flops or memory elements. And finally, an RTL code has combinational logic that can be an easy logic like AND gate or a complicated filter design. At the output side of the module, there is usually another sequential block to obstacle unwanted glitches at the output.

Sometimes, people who are not familiar with digital IC design and HDL language, use HLS alternatives to HDL. They can use SystemC, which can help them by allowing them to write software model of their design and then automatically translating into RTL code. It is advantageous for giving much faster development and being much easier to verify algorithmic behaviours. On the other hand, high-level languages have not been converted into low-level languages in an optimized way yet. Thus, for critical designs low level languages are preferred.

The more complicated stage is translating the RTL code into the GDSII layout. In some cases, an extra step of FPGA prototyping is done before manufacturing the design as a chip with expensive cost. FPGA prototyping is beneficial for being cheaper and faster than ASIC manufacturing. FPGAs are reconfigurable platforms, and users can implement their different designs on an FPGA many times. However, FPGAs do not provide transistor-level design and cannot be used for mixed-signal devices. Moreover, they have limited low power support.

After the RTL design is ready, it can be implemented on an FPGA to double-check if the design works fine in a physical system. Then, the RTL code is needed to be converted into something that foundries can understand, which is GDSII. The flow of converting an RTL code into a GDSII file has three main steps. Firstly, the RTL code

is synthesized and converted into a netlist of technology-specific cells. Secondly, these technology-specific cells are physically implemented by taking care of any physical aspects of IC design. As a final step, signoff is done. It is useful before sending the IC designs to fabrication. The signoff stage is crucial for designers who do not want to take their chips back that do not work after tape-out. If the signoff process is done properly, it can save lots of time and lots of money for not requiring extra tape-out.

Usually, digital designs are much larger than analog designs, and full layouts of the standard cells are not used as they increase memory requirement and slow down the implementation process. Thus, digital timing models (.lib file) and layout abstracts (.lef file) are used. Lib files include timing tables and power consumption tables, whereas lef file provides metal layers of the cells. Each IC manufacturing company has their own standard cell library for different technologies.

3.2 Synthesis

The general digital synthesis flow is given in Figure 3.2. Firstly, libraries are loaded. Synthesis settings are defined. RTL files of the design are loaded. Timing constraints are defined. It is the main driving factor for synthesis. Of course, area and power are also important, but if the timing is not meet the requirement, the chip works false. After elaboration, the design translated into technology-specific standard cells. After synthesis, optimization is made to meet user constraints. Changing the drive strength of the cells, remapping, logic transformations can be done for the optimization. In the end, a technology-specific netlist is generated by the synthesis tool. The generated netlist includes all the standard cells that are used in the schematic of the design after the optimization process. Generally, Verilog netlist is used by physical implementation tools.

There are also more advanced technics used in the synthesis process. In addition to timing optimization, power optimization can be done using logic cells with different threshold values. Moreover, test structures for the chip can be added like JTAG. Physical implementation can also be done in the synthesis step by defining a floorplan. It is useful when the design is within the boundaries of the timing constraints, and designers can have an idea if the used technology node is enough to meet their timing requirements.

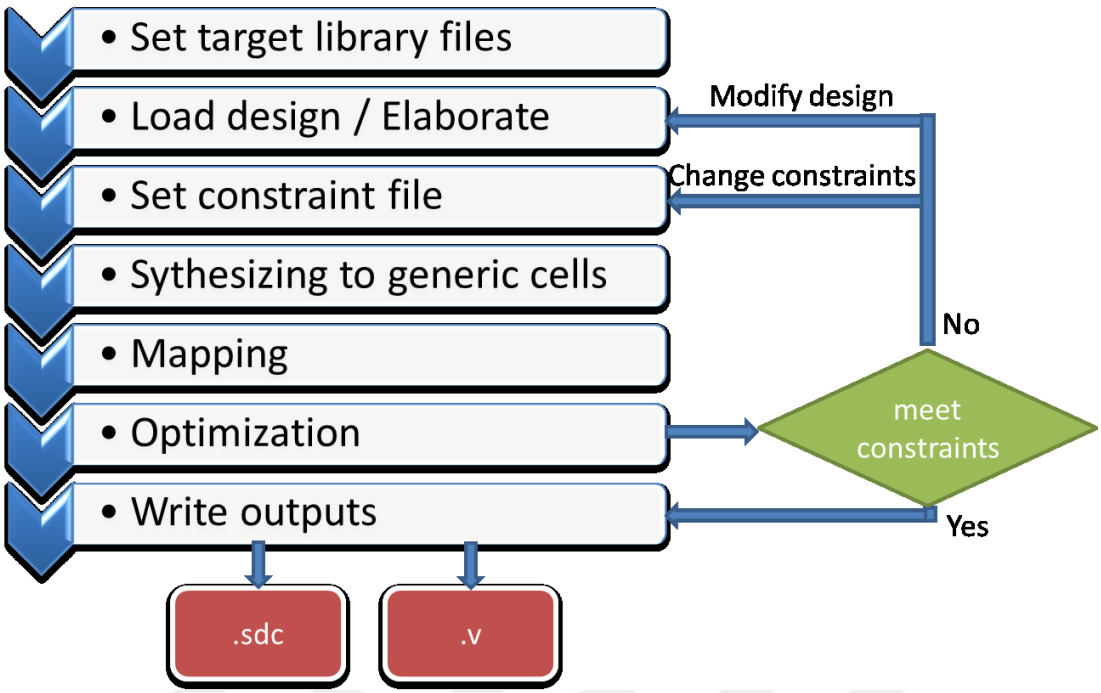


Figure 3.2 : Digital synthesis flow.

With the increasing demand for lower technology nodes in the ASIC world there is an increase in system variations on the chip such as size, threshold voltage, and cable resistance. Because of these factors, new models and techniques are subjected to high-quality testing.

An ASIC design involves complex tasks throughout the design. It is embarrassing to tell customers that the chips were defective while they are already in the production phase, and it will be a waste of time and money. To overcome this situation, the design for testing methods is suggested with a list of techniques. For example, by adding a scan path to the design, all registers are connected to each other. After the chip is produced, a known pattern is sent externally to the chip to check whether each register works correctly. Memory BIST (Built-in Self Test) is a second method used for testing. As the technology moves to a lower technology node, chip memory requires less space and faster access time. MBIST is a device used to check memories. Besides, ATPG is a method of generating test vectors to distinguish between the correct circuit behavior and the faulty circuit behavior caused by defects.

3.3 Physical Implementation

After the synthesis process, the physical implementation process is to be followed. General physical implementation flow is given in Figure 3.3. It starts with floorplanning. After the floorplanning, placement, clock tree synthesis, and routing are done. These steps are done by the tools, and most of the time they do not require lots of effort by the designers. In the end, the design is written out in formats that the foundries can accept.

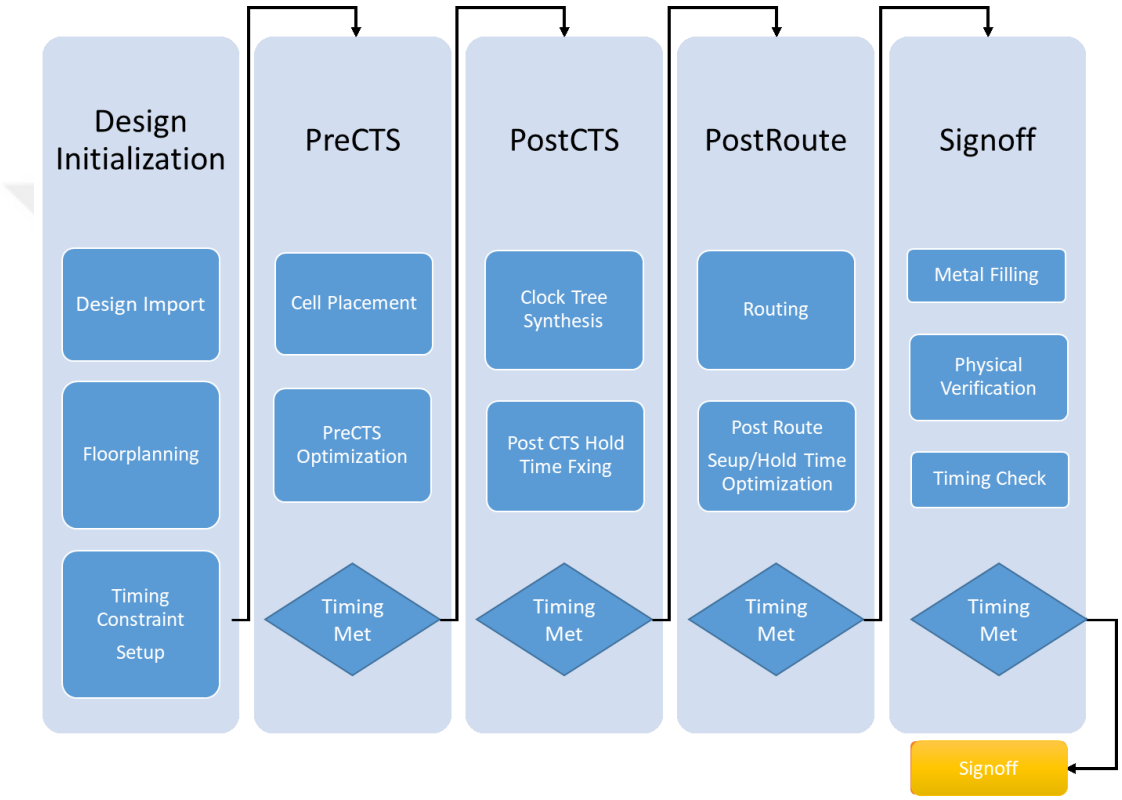


Figure 3.3 : Physical implementation flow.

3.3.1 Floorplanning

Physical implementation starts with floorplanning. Firstly, the synthesized netlist is imported, and place and route settings are defined. The size and shape of the chip are defined in the floorplanning process. It is the most critical step in the physical implementation. Designers should try to minimize the total chip area for good floorplanning, reserve enough area for easy routing, and improve signal delays. If floorplanning is not done properly, the design may not meet the timing.

In the floorplanning process, the size of the chip is defined. The manufacturing foundry defines the minimum and maximum size and the cost of tape-out. Inside the

floorplan, the core boundary, in which standard cells are placed, is described. At the outside of the core region, IO pads are placed. The size and style of placement can be different in different technology nodes. In addition to the IO pads, the power pad and ground pad for the core domain are also added. A power ring is also placed between the core area and IO area to distribute power all around the chip. The power planning of a floorplan is given in Figure 3.4.

Figure 3.4 : Power planning of a floorplan.

Usually, IO pads work in higher voltages than internal logic. Thus, in addition to core power pads, IO power pads are also added. Inside the IO pads region, an IO ring is turning around the entire chip to supply all IO pads. There are spaces between IO pads due to the minimum pitch size defined by foundry while doing wire bonding. To ensure that the IO power line is continuous, corner cells and filler cells are used. Hard macros, a customized design or IP, can be used in the standard cell region. To distribute power and hinder IR drop, power stripes are frequently placed. Too

frequent power lines might cause congestion and long critical path delay while connecting standard cells to each other.

After defining the chip's size, IO pads, standard cells coming from the synthesized netlist, hard macros, and power lines are placed. Usually, power structures are placed on the top two metal layers since they are thicker and have less resistance than other metal layers.

3.3.2 Placement

After floorplanning, the standard cells' placement is done by the tool taking care of the timing constraints. When placing standard cells, they are placed by turning upside down in the next row. Thus the standard cells fit into the VDD VSS lines with no gaps between the cells. A bad placement requires a larger area and also reduces performance.

3.3.3 Clock tree synthesis

During the synthesis stage, the clock signal is assumed to enter into every sequential element simultaneously. However, in the real world, it is not possible. Thus, it is essential to add buffer cells to carry the clock signal at the same time as much as possible. A clock input cannot drive all registers at the same time by itself. Thus, a balanced clock tree is created. It has clock buffers distributed around the chip, and the clock buffers' fanout is not too much.

3.3.4 Routing

After the clock tree placement, the clock signal is routed. For routing the clock signal, intermediate metal layers are usually used. Since the clock signal is the most critical thing in the digital design, it is routed firstly. And then the other connections are made.

3.3.5 Filling

After routing, the gap between standard cells is filled with filler cells or decoupling capacitors. The decoupling capacitors are useful for decreasing IR drop with carrying power from inside the chip. For mounting the designed chip in a PCB, bondpads should be placed around the chip. Bondpads are metal plates used for wire-bonding pins of the package of the chip to the IO cells.

3.4 Signoff check

The produced GDSII file can be sent to foundries for manufacturing. On the other hand, it is not recommended just yet. Since if there is a mistake in the design, lots of money and time are lost after the chip comes and does not work. For ensuring everything in the chip is working well, signoff flow should be run. DRC check

The signoff flow firstly checks DRC. In the DRC process, the GDS file's geometry is checked by the rules coming from the foundry to make sure that the design can be fabricated. These DRC rules include metal width, metal spacing, via size, maximum metal width, etc.

3.4.1 LVS check

In the LVS process, the produced layout is checked if it has the same functionality as the design's netlist.

3.4.2 LEC check

The netlist starting from physical implementation should also be checked if it is the same as what is designed in terms of RTL at the beginning. Intermediate netlist files should be produced since some layout equivalence tools make mistakes while comparing too different netlists.

3.4.3 Timing check

Timing analysis should also be done to check if the design meets timing in all the modes and corners defined by the user. Several nets interact with each other while they are not connected while they have some capacitances between the nets, which will affect the timing of that net in timing analysis. Therefore, capacitances and resistors from the design layout are needed to be extracted before running timing analysis to be used in more accurate timing analysis. The post-layout simulation should be done before manufacturing. Although the design works correctly in the behavioral simulations, it may work incorrectly after back-annotated gate level simulations. After the RC extraction, the delays of nets are calculated. Using this information, a more accurate timing analysis is done. Setup and hold time check should be done for ensuring synchronous behavior. The gate-level simulation is slower than behavioral simulation.

3.4.4 Power check

In addition to the timing simulation, static and dynamic power simulations can be done in the gate level simulation process.

The power consumption should be within the design requirement. Moreover, IR drop analysis should be done. If there is an IR drop, the design may work incorrectly. The designed chip takes power from power IO cells. Each power IO cells have a limited current supply. Thus, enough power pad cells should be used in the design to supply the entire chip. Logic cells consume power while switching on and off. When the switching activities are too much, IR drop occurs. Some regions on the chip cannot access enough power, and the logic cells in this region may not operate properly due to being outside condition of their library.

3.4.5 Testability check

A final step of signoff is importing test structures to the design. It is recommended due to being not too complicated. If something in the design does not work, test structures are very helpful to detect which chips work and why they work and why the others do not work. Test vectors are generated to verify the correct manufacture of the IC.

4. EXPERIMENTAL CRYPTANALYSIS OF NO-EQUILIBRIUM CHAOTIC SYSTEM BASED RANDOM NUMBER GENERATOR[1]

4.1 Target System

In terms of their time domain, chaotic systems could be mainly divided into two sub-categories: discrete/continuous time in accordance with the development of complex dynamical systems. Discrete-time chaotic systems are generally realized by switched-capacitor or switched-current techniques [44]. Since discrete-time chaotic systems are more complex and less robust structures, continuous-time chaotic systems are more preferred for random number generators.

In recent studies, the chaotic systems based on hidden-attractors have been developed for security applications that need confusion [45]. These kind of chaotic systems don't have any equilibrium points that they don't contain homoclinic or heteroclinic connections [46].

In [41], an FPGA-based chaotic oscillator system is suggested to produce random bit sequences. In the target RNG, a continuous-time three-dimensional chaotic system without equilibrium points is used as the core of the random bit generation. The chaotic system is formed from three different differential equations given in equation (4.1). The whole system mainly consists of (x, y, z) variables and six parameters "a, b, c, d, e, f". When all the beginning conditions of the system are selected as zero (x(0) = 0, y(0) = 0, z(0) = 0), the system stays in chaos. Thus, zero initial conditions of all parameters are preferable for the applications that use chaos.

$$\begin{aligned} x' &= ay - x + zy \\ y' &= -bxz - cx + zy + d \\ z' &= e - fxy - x^2 \end{aligned} \tag{4.1}$$

[1] Submitted to the 16th IEEE Asia Pacific Conference on Circuits and Systems "Experimental Cryptanalysis of No-equilibrium Chaotic System Based Random Number Generator" by Burak Acar, Tufan Coşkun Karalar, 2020, Copyright [2020] by IEEE.

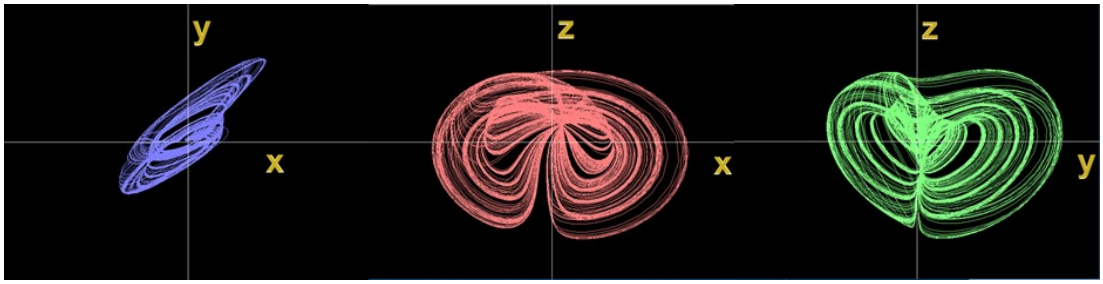


Figure 4.1 : Phase portraits of the target chaotic system for x-y, x-z, y-z.

In the given target third-order chaotic equation, the system parameters are selected as a = 2.8, b = 0.2, c = 1.4, d = 1, e = 10 and f = 2 in order to show a chaotic behavior. Different system parameters build different systems. Therefore, convenient ranges of the system parameters should be given to keep the system in chaos according to [47].

$$\begin{aligned} x' &= 2.8y - x + zy \\ y' &= -0.2xz - 1.4x + yz + 1 \\ z' &= 10 - 2xy - x^2 \end{aligned} \tag{4.2}$$

It is an inevitable fact that an experimental design will always exhibit parameter deflections. Therefore, it should be ensured that the responses of non-ideal effects on the chaotic system are not crucial. For this purpose, the Largest Lyapunov Exponent graph versus the parameter of the third-order chaotic system should be given. On the other hand, this analysis is not given in the target system [41].

By increasing the number of parameters in the target circuit, it was tried to take precautions against possible attacks. In this study, the parameter estimation of the targeted circuit has not been performed, but the defensive deficits of the proposed circuit have been shown. It has been explained that chaos circuits created without using any noise source can be broken by a master-slave synchronization method because they are in a deterministic behavior. However, even if the parameters are known, a synchronous attack circuit cannot be made when a chaos circuit is used after a physical noise source.

The phase portraits of the target system with the given parameters in equation (4.2) are obtained using a 4th-order Runge-Kutta algorithm with a flexible size in the MATLAB program. Figure 4.1 exhibits x-y, y-z, x-z phase portraits of the chaotic system with the initial values of x = 0, y = 0, z = 0.

Runge-Kutta algorithms are commonly used methods for solving differential equations numerically. The continuous-time chaotic oscillator is firstly discretized by the 4th-order Runge-Kutta algorithm method in float number type. Float numbers are converted into binary format to get more successful statistical results in random bit generation. It is still inadequate for the randomness requirements. The least significant bits of the binary data are used to produce random numbers since they are more complicated.

For calculating the quality of randomness of the generated random bits, statistical tests such as FIPS-140-1 and NIST-800-22 are used. These tests require 20 Kbit and 1 Mbit data sets, respectively. If the generated random bitstreams pass the statistical tests of randomness, then they are used in cryptographic applications. Otherwise, randomness can be achieved by changing the chosen bit or by using logical operators like XOR according to [41].

4.2 Attack System

In this study, the master and slave synchronization method [48] is used to achieve the convergence of the proposed attack circuit with the target circuit. Three different clone systems have been proposed for cryptanalysis of the target RNG and the obtained numerical and experimental results are presented.

In the suggested attack method, the synchronization of the target (master) and attack (slave) systems is confirmed by the Conditional Lyapunov Exponents (CLE)s method. Synchronization of the two systems is possible when the largest CLE is below zero [49]. Let the no-equilibrium chaotic system indicated in equation (4.2) is targeted by the suggested clone systems whose parameters are accepted as already known. There are many studies on parameter estimation on chaos-based structures, but our primary purpose is showing the vulnerabilities of the chaos-based random number generator without any physical phenomena. It is possible to generate the same bit series of the target system with only knowing the structure of the target RNG system and a scalar time series monitored from x_1.

Let's assume that a scalar time series of x_1 can be observed by probing the x_1 node of the target circuit. For the cryptanalysis of the target RNG system, a clone system is suggested that is presented by the equation (4.3) below.

$$
\begin{aligned}
x_2' &= 2.8y_2 - x_2 + z_2y_2 + c_1(x_1 - x_2) \\
y_2' &= -0.2x_2z_2 - 1.4x_2 + y_2z_2 + 1 \\
z_2' &= 10 - 2x_2y_2 - (x_2)^2
\end{aligned}
\tag{4.3}
$$

In this equation (4.3), the structure of the target system with system parameters is assumed to be known. c_1 is the coupling strength between target and attack systems. It is possible to build a clone system to the target system using equation (4.3) that synchronizes ($x_2 \rightarrow x_1$ for $t_n \rightarrow 1$) in tn normalized time.

The error between x_1 and x_2 is defined as $e_x = x_1 - x_2$. Similarly, $e_y = y_1 - y_2$ and $e_z = z_1 - z_2$ are defined. The main goal of the cryptanalysis is to determine a coupling strength (c_1) that brings x_2 to the exact value of x_1 ($x_2 \rightarrow x_1$). Notably, the absolute value of the error goes to zero ($e(t_n) \rightarrow 0$) when tn goes to infinity ($t_n \rightarrow 1$).

The largest CLEs corresponding to different coupling strengths of c1 are exhibited in Figure 4.2. when a scalar time series of x_1 can be observed. According to the obtained results, when c_1 is between 0.4 and 1.6, then the largest CLE is negative. Thus, the auto-synchronization of the target and the attack systems starting with distinct beginning conditions accomplished and stable [49]. On the other hand, auto-synchronization is unstable, and the largest CLE is positive for the values of c_1 less than 0.4 and more than 1.6. Taking c_1 as 1.2 gives a more stable ability to synchronize to the target system.

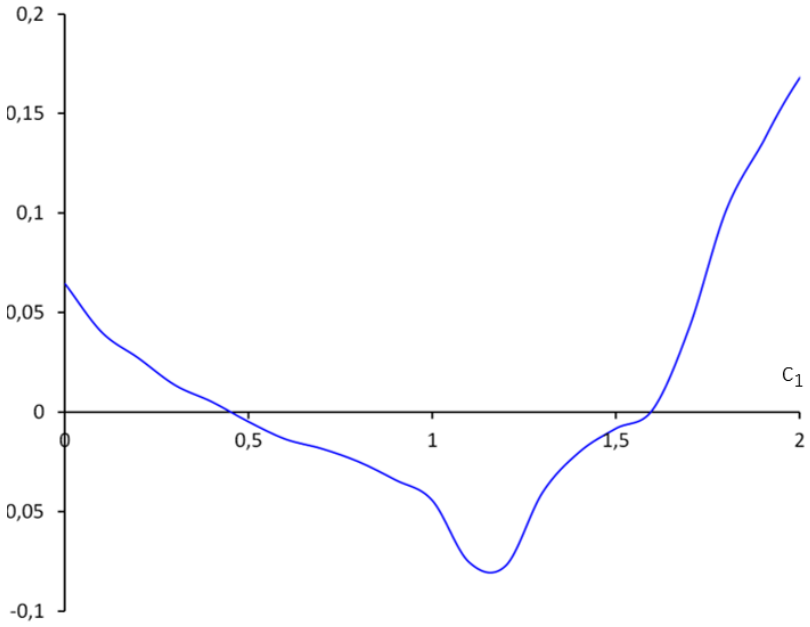


Figure 4.2 : Largest CLEs when x_1 is observable.

4.3 Numerical and Experimental Results

A clone system explained in Section 4.2 is designed to see if it is possible to attack the target system. Firstly, the target and attack systems are implemented in software in MATLAB program. Then, all the systems are performed on an FPGA board to prove the success of the proposed method experimentally. Vivado simulation results of the implemented design are saved to a file in a hexadecimal format. Then this file is inserted into a MATLAB program to build the given figures at the following.

The target and attack systems are started from different initial conditions to check the success of the proposed method. The target system is initialized from $x_1 = 0.1$, $y_1 = 0.1$, and $z_1 = 0.1$ points while the attacks systems start from $x_2 = 0.2$, $y_2 = 0.1$, and $z_2 = 0.1$. It is also possible to select any other different initial conditions.

Figure 4.3 exhibits numerical results of the synchronization of target and clone systems when x_1 is observable. Black lines present asynchronous behavior, while red lines present synchronized behavior. After a while, the clone system behaves as same as the target system.

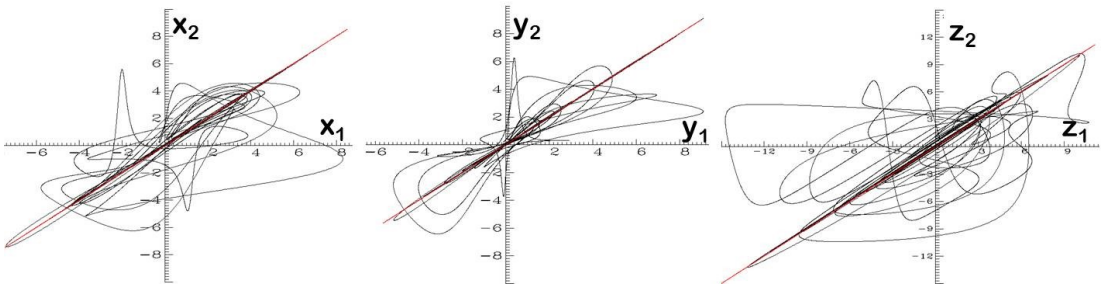


Figure 4.3 : Synchronization of the attack system (x_2, y_2, z_2) to the target system (x_1, y_1, z_1).

To show the success of the proposed attack method experimentally, target and attack systems are implemented on a Zedboard FPGA evaluation board. The design of the attack system is generally same as the target system with a small difference. One input of the attack system is coming from the target system with probing to x, y, or z node. This attack system synchronizes to the target system without any knowledge of the initial conditions of the target system.

The implementation of the target system is first started by designing the Runge-Kutta algorithm in the Xilinx Vivado 2018.1 program using VHDL language. The Runge-Kutta algorithm is implemented as supporting a 32-bit IEEE 754-2008 floating-point

number standard. To be able to handle floating-point type in the design, the float pkg library is added to the project as introduced in [50].

The designed target and attack systems are synthesized for XC7Z020-CLG484 chip of Xilinx Zynq-7000 family. Vivado simulation of them is given in Figure 4.4 in hexadecimal format taking h as 0.05. Statistical results of the FPGA implementations of the designs are given in Table 4.1.

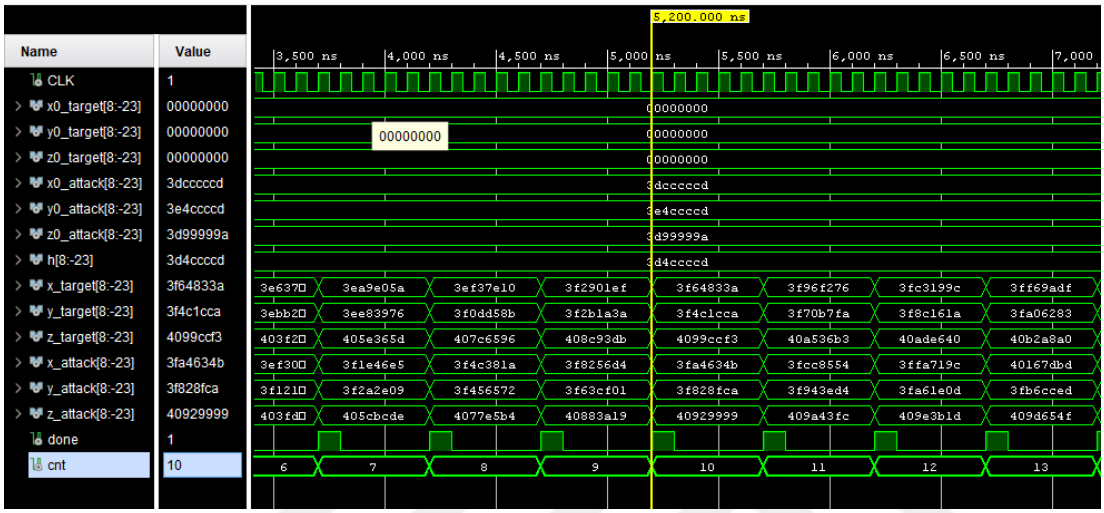


Figure 4.4 : Vivado simulator results of the implemented FPGA-based chaotic oscillator.

Table 4.1 : FPGA implementation results.

Implemented System	LUT Usage	FF Usage
Target System	7904	170
Attack System	8006	170

5. ASIC DESIGN OF A ROBUST DIGITAL RANDOM NUMBER GENERATOR BASED ON TRANSIENT EFFECT OF RING OSCILLATOR

5.1 TERO Based RNG Design

RNGs based on transition effect often use SR latches [51]. Each SR latch structure includes two symmetrical paths, and each path consists of an odd number of inverters and an AND gate. A loop is set up by connecting each path's output to an input of an AND gate that activates the other path. An external signal controls the other inputs of these AND gates. When this signal is activated, signals from both paths of the SR latch pass through the inverters and begin to oscillate. After a while, the oscillation's duty cycle decreases over time because the two paths are not exactly symmetrical, and the oscillation at the output of each path is damped and ends. Thus, the output of the SR latch circuit remains in an unknown steady state.

Each time the SR latches are turned on and turned off, it stops at an unknown logic level due to metastability. Besides, the number of times SR latches will oscillate and stop is unstable. This information can be used to generate random numbers. For example, the least significant bit of the oscillation number is used for random bit generation [34]. A random number can be generated by looking at the distribution of the number of oscillations [36]. However, counting the total number of oscillations causes the use of extra logic and increases the area and power consumption.

In [38], an RNG is proposed using the wake-up uncertainties of the SR latches at the start time. This RNG is implemented on an FPGA and the results are shared. In this study, the SR latch structure, which was pre-studied on FPGA before in [38], was implemented as ASIC this time.

The unit cell of the SR circuit is shown in Figure 5.1. Each SR latch contains 2 inverters and 2 AND gates. A 50 SR latch circuit, in which sufficient entropy was provided on the FPGA, was used as the entropy source as shown in Figure 5.2. A random bit is generated by XORing a 50 SR latch circuit.

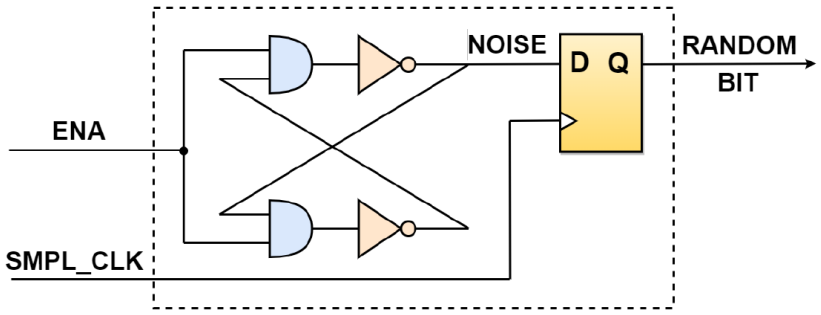


Figure 5.1 : Metastable SR Latch circuit [38].

SR latch outputs can oscillate at high frequencies. This situation causes XOR gates to be released quickly. Therefore, it causes setup time errors for the FF used when generating the bit at the end. To prevent this, a flip flop is placed at the output of each SR latch. Instead of counting the oscillations produced by each SR latch, the method of sampling the oscillation zone in a particular region is preferred. This technic reduces power consumption by avoiding unnecessary use of digital logic.

The 0 to 1 ratio of bits produced by the proposed RNG system must be statistically balanced. Applying post-processing methods such as Von Neumann at the circuit's output will reduce the bit generation rate of RNGs. Therefore, it will be better if the raw data have statistically good properties. For this purpose, the RNG proposed in this study is subjected to frequency analysis in run-time as it produces bits. Based on the results obtained, it is decided how long after an SR latch circuit is turned on, sampling is appropriate. The behavior of the SR latches can vary depending on where they are placed in an FPGA. For ASIC chips, they also behave differently on different chips. Therefore, in this study, a feedback mechanism is applied that determines when we should take samples as part of the system.

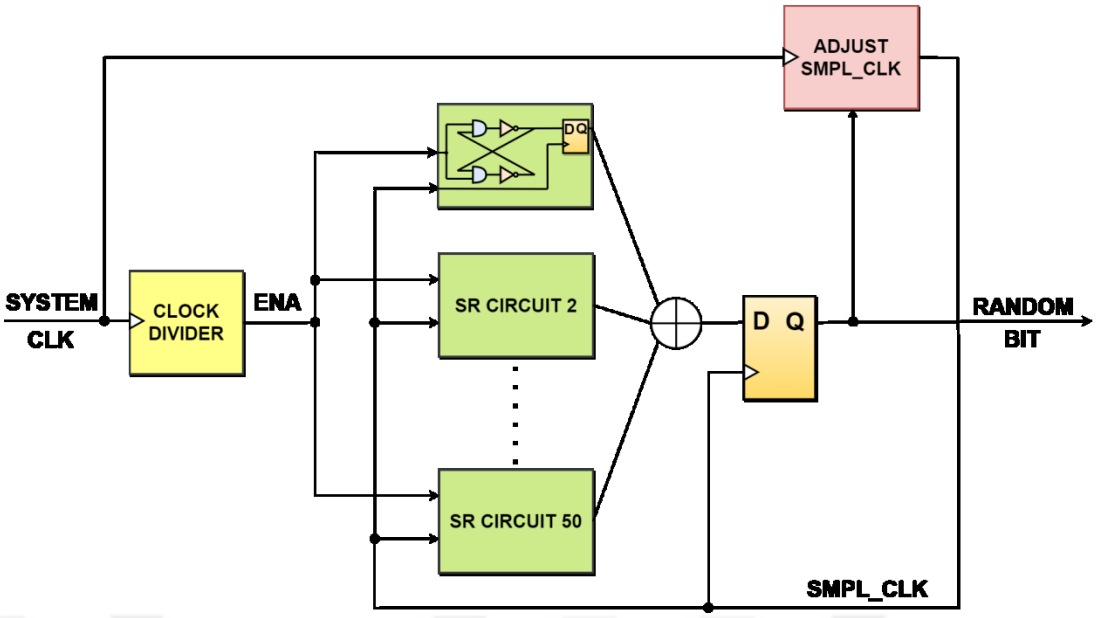


Figure 5.2 : The proposed TERO-based RNG.

5.2 Synthesis

As explained in the previous section, a transition effect based RNG is synthesized with VHDL language. The designed RNG generates random bit sequences using the metastability of the SR latches. This method uses less hardware than conventional ring oscillator based RNGs and preferable for low power applications. Sampling in the region with the highest metastability is preferred rather than counting the oscillation that occurs when the SR latch is activated. After the FPGA implementation and good statistical results of the RNG presented in [38], this RNG structure is synthesized with a TSMC 180 nm standard cell library using the Genus Synthesis Solution of the Cadence company.

The proposed RNG design can also be designed in a completely custom way as analog. On the other hand, analog design flow requires more design time, and every time the technology changes, it has to be redesigned over and over again. Instead, the RNG design can be done by following the digital design flow. However, the design tools aim to optimize the use of the logic cells and sometimes the structure synthesized may differ from the desired structure. Therefore, some constraints should be applied to prevent unwanted optimizations.

In the synthesis phase, the proposed RNG's VHDL codes and technology libraries to be used are loaded. Separate constraints for power and space are not used in this

study. However, it is possible to achieve less power consumption by using standard cells operating at different lower technologies thresholds. The synthesis of an SR latch circuit to standard library cells is shown in Figure 5.3.

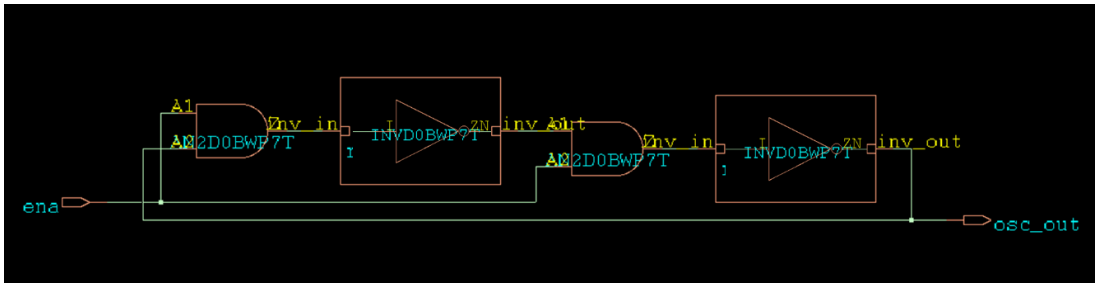


Figure 5.3 : The synthesis of metastable SR Latch circuit.

All SR latches in the circuit are controlled by an enable signal as shown in Figure 5.2. In the structure verified on the FPGA [38], each SR latch is turned on every 320 ns (3.125 MHz). In the FPGA experiments, it was observed that sampling at about 50 ns after the activation signal was turned on gave statistically good results [38]. This sampling time may vary according to different locations and routes. This is also true for the ASIC design. Therefore, a structure that will automatically decide on the optimum sampling time is used. This structure is a simple 10-bit counter and controls the frequency balance of the bits generated.

In the proposed ASIC design, a 100 MHz clock signal is needed from the outside. This clock signal is divided and an enable signal is generated at 3.125 Mhz. Using a fast clock from the outside is necessary to get a sample in the right place to obtain good quality of random numbers. The bit production rate can be increased by changing the clock divider's parameter according to the results obtained after manufacturing the design.

RNG structure with 50 SR latches is synthesized with the Genus tool. The schematic view of the synthesized design is given in Figure 5.4. The synthesized structure consists of SR circuits, XOR gates, and some control logic.

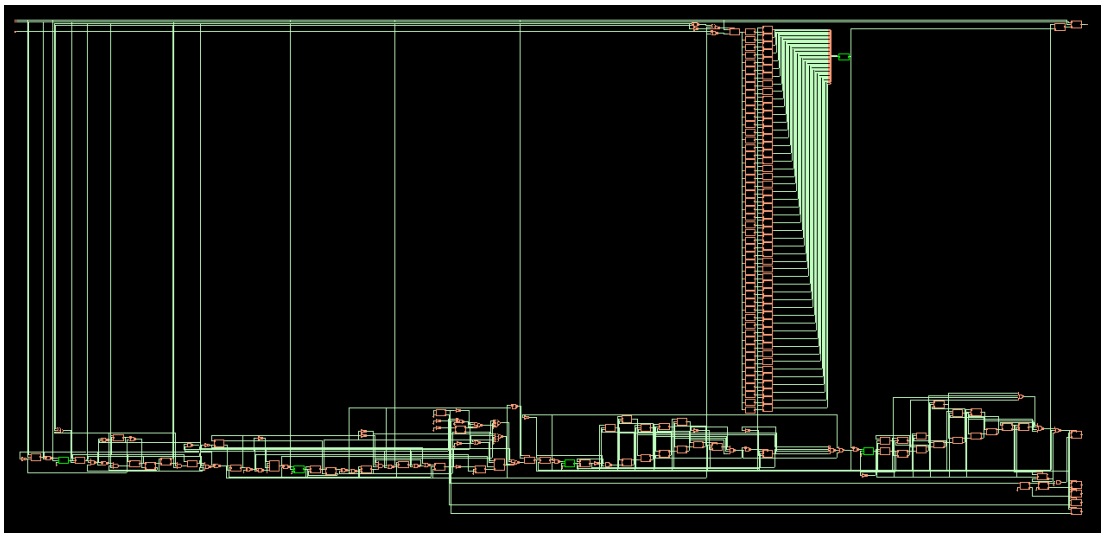


Figure 5.4 : The synthesis of TERO-based RNG.

The synthesized design uses 502 instances. The slack value is 3,57 ps, and the total negative slack is zero. Thus, the design meets timing constraints.

After synthesis, the proposed design has been verified by the formal equivalence test to see if the synthesized netlist has the same function as the design entered at the beginning. The verification results are given in Figure 5.5.

```
                    Verification Report
------------------------------------------------------------------------------
Category                                                              Count
------------------------------------------------------------------------------
1. Non-standard modeling options used:                                    0
     Tri-stated output:                                  checked
     Revised X signals set to E:                         yes
     Floating signals tied to Z:                         yes
     Command "add clock" for clock-gating:               not used
------------------------------------------------------------------------------
2. Incomplete verification:                                               0
     All primary outputs are mapped:                     yes
     Not-mapped DFF/DLAT is detected:                    no
     All mapped points are added as compare points:      yes
     All compared points are compared:                   yes
     User added black box:                               no
     Black box mapped with different module name:        no
     Empty module is not black boxed:                    no
     Command "add ignore outputs" used:                  no
     Always false constraints detected:                  no
------------------------------------------------------------------------------
3. User modification to design:                                           0
     Change gate type:                                   no
     Change wire:                                        no
     Primary input added by user:                        no
------------------------------------------------------------------------------
4. Conformal Constraint Designer clock domain crossing checks recommended:  0
     Multiple clocks in the design:                      no
------------------------------------------------------------------------------
5. Design ambiguity:                                                      0
     Duplicate module definition:                        no
     Black box due to undefined cells:                   no
     Golden design has abnormal ratio of unreachable gates:    no
     Ratio of golden unreachable gates:                  8%
     Revised design has abnormal ratio of unreachable gates:   no
     Ratio of revised unreachable gates:                 8%
------------------------------------------------------------------------------
6. Compare Results:                                                    PASS
     Number of EQ compare points:                        125
     Number of NON-EQ compare points:                    0
     Number of Aborted compare points:                   0
     Number of Uncompared compare points :               0
==============================================================================
pass
```

Figure 5.5 : The layout equilavance check of the golden netlist and final synthesized netlist.

5.3 Physical Implementation

After the netlist synthesis, the physical implementation process is followed as explained in detail in Section 3.4. Firstly, a floorplan is defined. The structure designed in this study is designed as a custom IP block to be used in various applications later. The design takes up a very small area, so users can easily include this custom IP in their chips. Since it is designed as a hard macro cell, IO cells and bondpads are not used.

After floorplanning, logic cells are placed at the defined floorplan. Then the clock tree synthesis flow is followed. The clock tree diagram is shown in Figure 5.6. The FFs at the output of each SR latch cannot be seen in the clock tree diagram as they are not defined as a clock signal, and they are controlled with logics in an asynchronous manner.

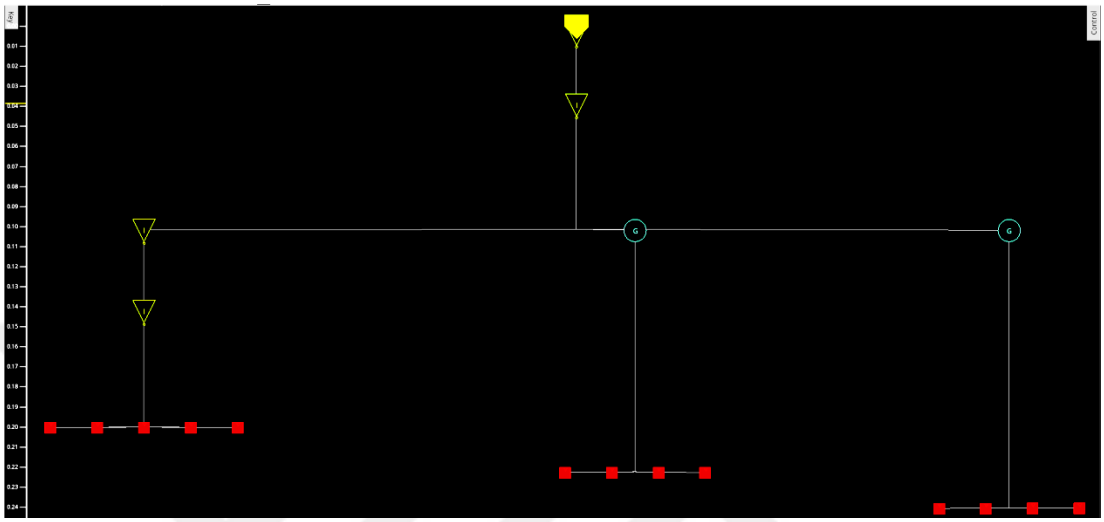


Figure 5.6 : The clock tree of the design.

After the clock tree synthesis, all cells are connected where they should connect with the help of routing scripts. The critical path of the design is drawn in Figure 5.7. The critical path should be checked if there is any negative slack in the design. If the timing does not meet, the retiming technique can be applied. In the retiming process, the location of latches or registers is slid without affecting functional behavior. Some of the logic paths need more than one clock cycle to propagate to the next sequential logic in some cases. A multicycle path timing constraint is used in these cases.

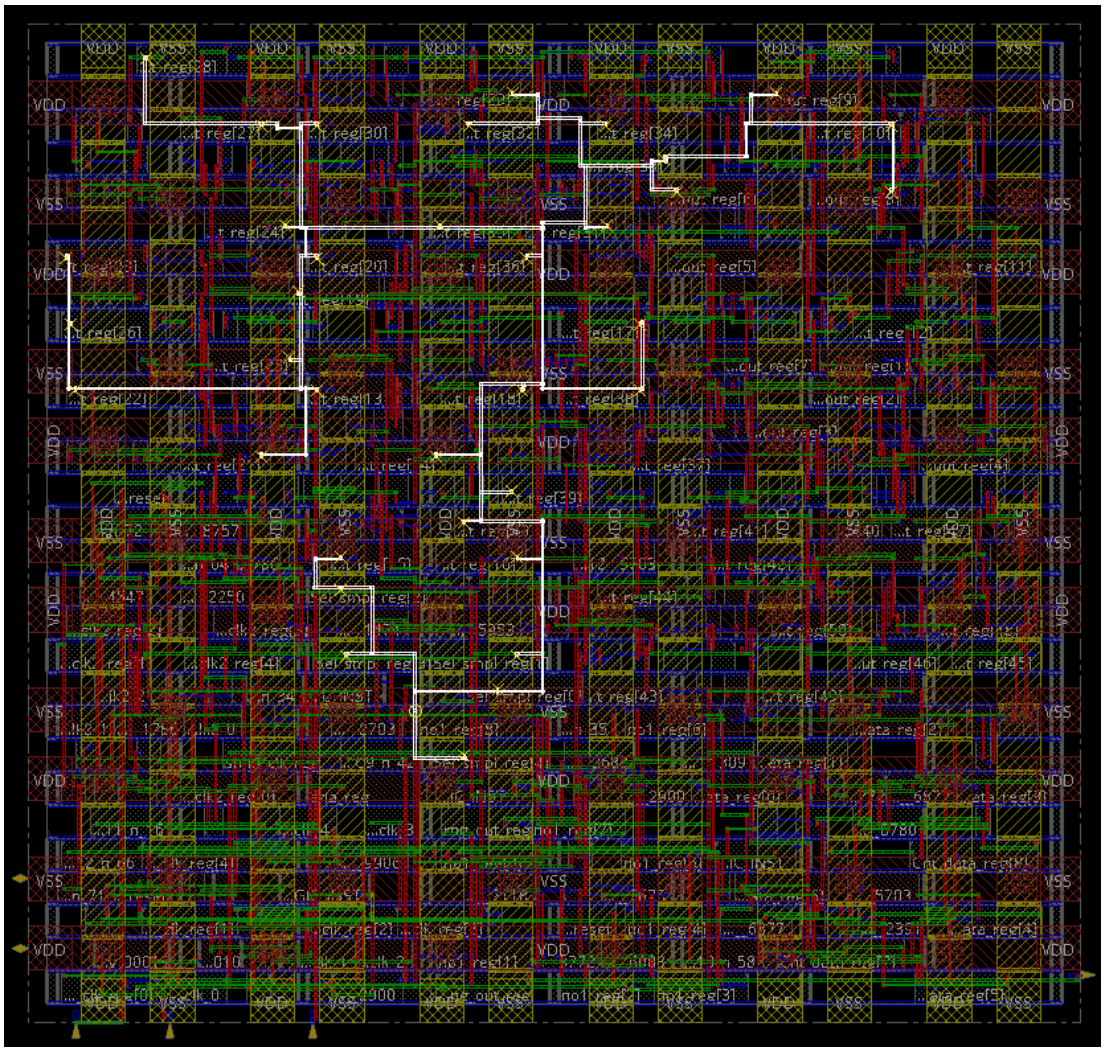


Figure 5.7 : The critical path of the final netlist.

The final physical layout of the proposed RNG is given in Figure 5.8. The 3D layout view of the final netlist is exhibited in Figure 5.9. Top metals are thicker than other layers as seen in Figure 5.9.

The number of instances used in the RNG block is 621. The dimensions of the designed module are 120 µm x 114 µm. The total area of the core is 13623 µm2. For a clock frequency of 100 MHz and at the typical-case corner (25 C, 1.8 V supply, typical process), the leakage power for the RNG is 0.02 nW, and the total power consumption is 0.44 mW while taking the switching activity as 20% using the Innovus default settings.

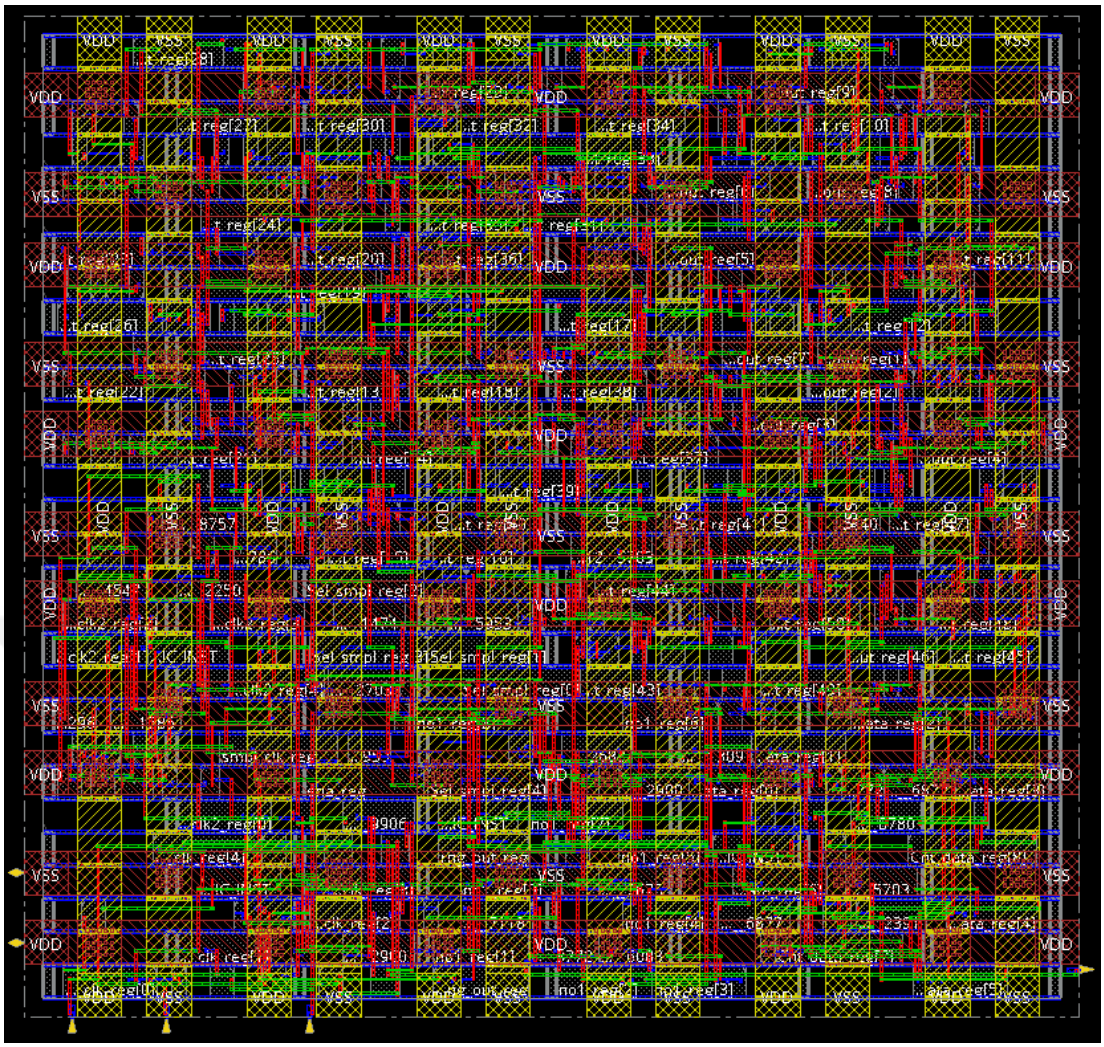


Figure 5.8 : The physical layout of the final netlist.

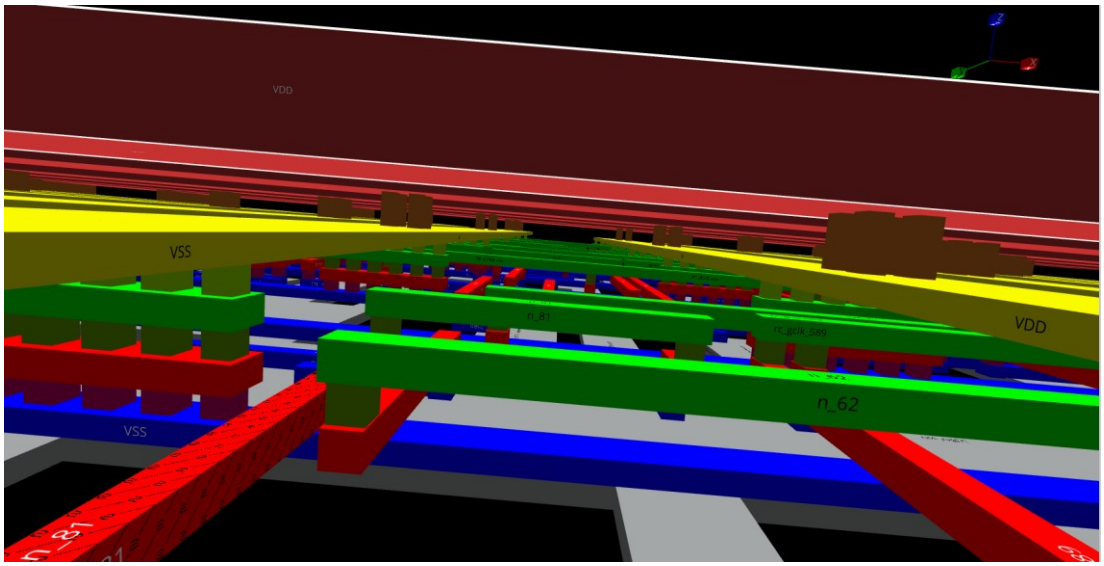


Figure 5.9 : 3D layout view of the final netlist.

The rail analysis of the designed RNG for VDD at the worst-case corner (-40 C, 1.8 V supply, slow-slow process) is given in Figure 5.10. There are no power pads in the design. IR drop analysis is done as the power supply is coming from the power ring at the edges of the design. Thus, the maximum IR drop occurs at the inner side of the design. Since the total power consumption is very low, the IR drop on the design is very low.

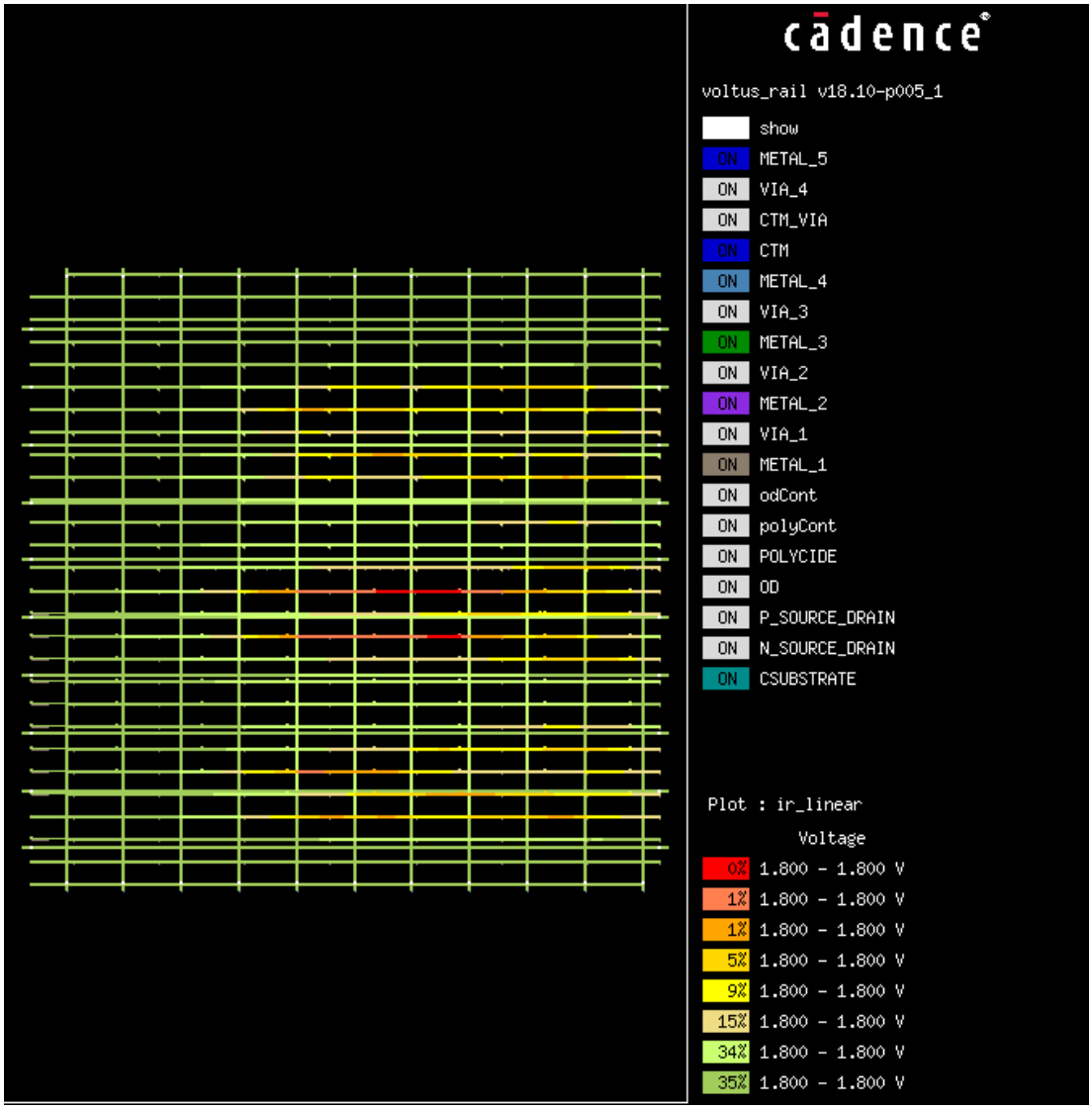


Figure 5.10 : IR drop analysis of the designed RNG.

5.4 Comparison of the proposed RNG

Basic ring oscillator based TRNGs are suitable for all FPGA families, and they can be implemented using automatic placement method provided by the design software. It has relatively high and constant throughput. However, it has high power consumption due to using a large number of ring oscillators.

Transient effect based TRNGs are important for requiring few logic cells, consumption of low power, and having relatively high throughput. On the other hand, it isn't easy to describe the randomness with a mathematical model. It usually requires a manual implementation of FPGA logic cells.

Chaos-based RNGs can also be implemented digitally. However, because these circuits are deterministic, they are vulnerable to potential attacks. So they should be combined with analog noise.

Some of the state-of-art RNGs implemented as an ASIC are compared in table 5.1.

Table 5.1 : Comparison with other RNGs implemented as ASIC.

	[40]	**[52]**	**[53]**	**This Study**
Method	Transition	Cellular-Automata	Fibonacci	Transition
Technology (nm)	180	130	65	180
Throughput (Mbit/s)	0.37	1.4	12	3.125
Voltage (V)	1.8	-	1.2	1.8
Gates	4898	969	1115	621
Power (mW)	1.673	2.66	4.81	0.44

6. CONCLUSIONS AND RECOMMENDATIONS

Some state-of-the-art digital random number generators in the literature are examined, and new improvements are proposed for robustness and higher quality of randomness. In the second section, some of the random number generators in the literature are introduced. In the third section, the journey of an idea to a chip is presented in an inclusive way but not too long.

In the fourth chapter, a security analysis of a chaos-based RNG is presented. Chaotic systems are used in random number generation because they can bring the system output to very different points with a small difference in the initial conditions. However, random number generators using chaotic systems without any physical noise source are deterministic. Therefore, they are vulnerable to possible attacks. In this work, a 3-D no-equilibrium chaotic system is analyzed in terms of security. The master-slave synchronization method is used to clone to the target system. A hardware demonstration is also prepared to show its feasibility. Numerical and experimental results prove that producing the same bit series with the target system is possible in a limited scalar time with only probing one node of the target system.

Classical ring oscillator based RNGs consume too much area and power. Therefore, an ASIC design of a transition effect based RNG is proposed in the fifth section. The structure that was previously verified in FPGA is designed exactly as ASIC using 180 nm TSMC technology. The proposed method is preferable for digital designers because of its low usage of logic cells and low power consumption as the SR latches do not keep oscillating, unlike classical ring oscillators. Furthermore, the proposed method is also more robust against correlation-based attacks according to the previous published study implemented on an FPGA. The randomness in the RNG output might be manually degraded by external interference or by tempering attacks. Furthermore, due to aging effects on the circuits or environmental conditions, such as extreme temperatures, the RNG's entropy source can be degraded. In that case, the RNG output cannot satisfy the statistical tests, and the feedback mechanism shifts the sampling moment to maximize the entropy. If an optimum sampling moment cannot

be found, the RNG stops giving output. Thus the RNG output is not blindly used. However, in the previous experimental work with FPGA, it was not observed any cases when the feedback mechanism could not find an optimum sampling window. There can be a poor implementation of the SRs on FPGA or ASIC. Actually, that is the reason to keep the number of SR latches high. Thus, the entropy is increased by XORing the output of each SR latch cell. In this study, it was tried to minimize the number of SR latches in order to save area and power. But there is room for more detailed analyses as a future work. For instance, in a critical application, the number of SR latches can be chosen quite high such that even with poor implementation of some of the SRs on the FPGA, the system will still work well. Furthermore, the RNG is prototyped on a modern FPGA fabricated at advanced technology nodes. However, when the system is implemented as an ASIC at the 180 nm process, the entropy will be much higher, and the metastability will be more dominant. Therefore, the RNG will be more secure. 3.125 Mb/s throughput with previous successful results in FPGA is targeted. The total area of the proposed design is 13623 μm2. The total power consumption is 0.44 mW while taking the switching activity as 20% at typical-case corner (25 C, 1.8 V supply, typical process). As a future work, the proposed RNG structure will be fabricated to measure the proposed random number generator's performance in an ASIC implementation. Bit production rate can be increased according to post-production measurements.

REFERENCES

[1] **Schneier, B.** (2007). Applied cryptography: protocols, algorithms, and source code in C. *John Wiley & Sons.*

[2] **Schulz, M. A., Schmalbach, B., Brugger, P., & Witt, K.** (2012). Analysing humanly generated random number sequences: a pattern-based approach. *PloS one*, 7(7).

[3] **Kerckhoffs, A.** (1883). La cryptographic militaire. Journal des sciences militaires, 5-38.

[4] **Ramaswamy, R.** (1989). Application of a key generation and distribution algorithm for secure communication in open systems interconnection architecture. *In Proceedings. International Carnahan Conference on Security Technology (pp. 175-180). IEEE.*

[5] **Ylonen, T., & Lonvick, C.** (2006). The secure shell (SSH) transport layer protocol. *RFC 4253.*

[6] **Fischer, V., Bernard, F., & Haddad, P.** (2013, September). An open-source multi-FPGA modular system for fair benchmarking of true random number generators. *23rd International Conference on Field programmable Logic and Applications (pp. 1-4). IEEE.*

[7] **Gentle, J. E.** (2006). Random number generation and Monte Carlo methods. *Springer Science & Business Media.*

[8] **Smillie, K. W.** (1967). Introduction to Numerical Analysis, by C. -E. Fröberg. *Canadian Mathematical Bulletin*, 10(1), 137–137.

[9] **Gao, Y. L., Chen, X. B., Chen, Y. L., Sun, Y., Niu, X. X., & Yang, Y. X.** (2018). A secure cryptocurrency scheme based on post-quantum blockchain. *IEEE Access, 6, 27205-27213.*

[10] **Chen, S. L., Hwang, T., Chang, S. M., & Lin, W. W**. (2010). A fast digital chaotic generator for secure communication. *International Journal of Bifurcation and Chaos, 20(12), 3969-3987.*

[11] **Chen, S.** (2018) Random number generators go public. *Science (New York, N.Y.) 360 (6396), 1383–1384.*

[12] **Kosak, A.** (2018). Art of Algorithms and Artificial Intelligence in The Age of Digitalisation.

http://www.gunleme.dersbelgeligi.com/en/the-age-of-digitalisation

[13] **Kaiblinger, H., Schrotter, F.** (2015). U.S. Patent Application No. 14/070,323.

[14] **Fischer, V., Drutarovsky, M.** (2002). True random number generator embedded in reconfigurable hardware. *14In International Workshop on Cryptographic Hardware and Embedded Systems (pp. 415-430). Springer, Berlin, 15 Heidelberg.*

[15] **Koerner, B., Lapowsky, I., Rivlin, G., Matsakis, L., Finley, K., Barber, G. and Baker-Whitcomb, A.** (2019). Meet Alex, the Russian Casino Hacker Who Makes Millions Targeting Slot Machines. Available at: https://www.wired.com/story/meet-alex-the-russian-casino-hacker-who-makes-millions-21targeting-slot-machines/

[16] **Yildirim, S.** (2012). A True Random Number Generator in FPGA for Cryptographic Applications. *A Thesis Submitted to 23the Graduate School of Natural and Applied Sciences of Middle East Technical University, tech. rep.*

[17] **Von Neumann, J.** (1963). Various techniques used in connection with random digits. *John von Neumann, Collected25Works, 5, 768-770.*

[18] **Ergun, S.** (2013). U.S. Patent No. 8,612,501. *Washington, DC: U.S. Patent and Trademark Office.*

[19] **Ergun, S.** (2006). Compensated true random number generator based on a double-scroll attractor. *International Symposium on Nonlinear Theory and its Applications (NOLTA'06), Sept. (pp. 391-394).*

[20] **Ergun, S**. (2011). Regional random number generator from a cross-coupled chaotic oscillator. *IEEE 54th International Midwest Symposium on Circuits and Systems (MWSCAS) (pp. 1-4). IEEE.*

[21] **Ergun, S.** (2016). On the Security of Chaos Based "True" Random Number Generators. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 99(1), 363-369.*

[22] **Fairfield, R. C., Mortenson, R. L., & Coulthart, K. B**. (1984). An LSI random number generator (RNG). In Workshop on the Theory and Application of Cryptographic Techniques (pp. 203-230). Springer, Berlin, Heidelberg.

[23] **Sunar, B., Martin, W. J., & Stinson, D. R.** (2007). Performance, Fault Tolerance, Reliability, Security, and Testability-A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks. *IEEE Transactions on Computers, 56(1), 109.*

[24] **Dichtl, M., & Golić, J. D.** (2007). High-speed true random number generation with logic gates only. *In International Workshop on Cryptographic Hardware and Embedded Systems (pp. 45-62). Springer, Berlin, Heidelberg.*

[25] **Wold, K., & Tan, C. H.** (2008). Analysis and enhancement of random number generator in FPGA based on oscillator rings. In 2008 International Conference on Reconfigurable Computing and FPGAs (pp. 385-390). IEEE.

[26] **Hada, K.** (2011). Performance enhancement of the ring oscillator type true random number generator on FPGA. *IEICE Technical Report, CSEC2011-53.*

[27] **Bochard, N., Bernard, F., & Fischer, V.** (2009). Observing the randomness in RO-based TRNG. *In 2009 International Conference on Reconfigurable Computing and FPGAs (pp. 237-242). IEEE.*

[28] **Guler, U., & Ergiun, S.** (2010). A high speed IC random number generator based on phase noise in ring oscillators. *In Proceedings of 2010 IEEE International Symposium on Circuits and Systems (pp. 425-428). IEEE.*

[29] **Acar, B., & Ergun, S.** (2018). Correlation-based cryptanalysis of a ring oscillator based random number generator. *In 2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS) (pp. 1050-1053). IEEE.*

[30] **Acar, B., & Ergun, S.** (2019). A Digital Random Number Generator Based on Irregular Sampling of Regular Waveform. *In 2019 IEEE 10th Latin American Symposium on Circuits & Systems (LASCAS) (pp. 221-224). IEEE.*

[31] **Maiti, A., Casarona, J., McHale, L., & Schaumont, P.** (2010). A large scale characterization of RO-PUF. *In 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST) (pp. 94-99). IEEE.*

[32] **Bochard, N., Bernard, F., Fischer, V., & Valtchanov, B.** (2010). True-randomness and pseudo-randomness in ring oscillator-based true random number generators. *International Journal of Reconfigurable Computing, 2010.*

[33] **Bayon, P., Bossuet, L., Aubert, A., Fischer, V., Poucheret, F., Robisson, B., & Maurine, P.** (2012). Contactless electromagnetic active attack on ring oscillator based true random number generator. *In International Workshop on Constructive Side-Channel Analysis and Secure Design (pp. 151-166). Springer, Berlin, Heidelberg.*

[34] **Bossuet, L., Ngo, X. T., Cherif, Z., & Fischer, V.** (2013). A PUF based on a transient effect ring oscillator and insensitive to locking phenomenon. *IEEE Transactions on Emerging Topics in Computing, 2(1), 30-36*.

[35] **Varchola, M., & Drutarovsky, M.** (2010). New high entropy element for FPGA based true random number generators. *In International Workshop on Cryptographic Hardware and Embedded Systems (pp. 351-365). Springer, Berlin, Heidelberg.*

[36] **Li, T., Wu, L., Zhang, X., Wu, X., Zhou, J., & Wang, X.** (2017). A novel transient effect ring oscillator based true random number generator for a security SoC. *In 2017 International Conference on Electron Devices and Solid-State Circuits (EDSSC) (pp. 1-2). IEEE.*

[37] **Delvaux, J.** (2019). Refutation and redesign of a physical model of TERO-based TRNGs and PUFs, *IACR Cryptology ePrint Archive, p. 810.*

[38] **Acar, B., Ergun, S.** (2020). A robust digital random number generator based on transient effect of ring oscillator. *In 2020 IEEE 11th Latin American Symposium on Circuits & Systems (LASCAS) (pp. 1-4). IEEE.*

[39] **Acar, B., Ergun, S.** (2020). A random number generator based on irregular sampling and transient effect ring oscillators. *In 2020 IEEE International Symposium on Circuits and Systems (ISCAS) (pp. 221-224). IEEE.*

[40] **Acar, B., Ergun, S.** (2020). A reconfigurable random number generator based on transient effect of ring oscillator. *In IEEE Transactions on Circuits and Systems II: Express Briefs (ISICAS) 67(9) (pp. 1609-1613). IEEE.*

[41] **Akgul, A., Calgan, H., Koyuncu, I. et al.** (2016*).* Chaos-based engineering applications with a 3D chaotic system without equilibrium points. *Nonlinear Dyn* 84, 481–495.

[42] **National Institute of Standarts and Technology** (2010). A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. *NIST 800-22*.

[43] **National Institute of Standarts and Technology** (2002). Security requirements for cryptographic modules.

Available: https://csrc.nist.gov/publications/detail/fips/140/2/final

[44] **Delgado Restituto, M., & Rodríguez Vázquez, Á. B.** (2002). Integrated chaos generators. *Institute of Electrical and Electronics Engineers. Proceedings of the IEEE. 90. 747 - 767.*

[45] **Wang, Z., Cang, S., Ochola, E. O., & Sun, Y.** (2012). A hyperchaotic system without equilibrium. *Nonlinear Dynamics. 69(1-2), 531-537.*

[46] **Cafagna, D., & Grassi, G.** (2012). Chaos in a new fractional-order system without equilibrium points. *Communications in Nonlinear Science and Numerical Simulation*, 19(9), 2919-2927.

[47] **Ergün, S.** (2016). On the security of chaos based "true" random number generators. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*. 99(1), 363-369.

[48] **Hasler, M.** (1994). Synchronization principles and applications. *Circuits and systems tutorials*. 314-327.

[49] **Hasler, M.** (1994). Synchronization principles and applications. *Circuits and systems tutorials*. 314-327.

[50] VHDL coding tips and tricks. Xilinx.

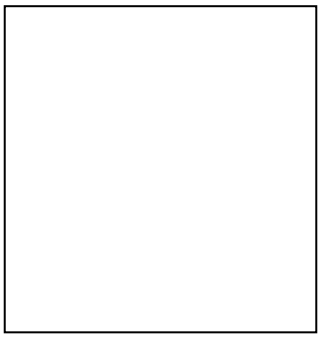
Available: http://vhdlguru.blogspot.com/2017/10/how-to-use-ieeeproposed-libraries-in.html

[51] **Kacprzak T.** (1988). Analysis of oscillatory metastable operation of an rs flipflop. *IEEE Journal of Solid-State Circuits, vol. 23, no. 1, pp. 260–266, 1988.*

[52] **Cartagena, Juan & Gomez, Hector & Roa, Elkim.** (2016). A fully-synthesized TRNG with lightweight cellular-automata based post-processing stage in 130nm CMOS. *2016 IEEE Nordic Circuits and Systems Conference (NORCAS).* pp. 1-5.

[53] **Demir, K., & Ergun, S.** (2019). Random Number Generators Based on Irregular Sampling and Fibonacci–Galois Ring Oscillators. *IEEE Transactions on Circuits and Systems II: Express Briefs 66(10), 1718-1722. IEEE.*

CURRICULUM VITAE



Name Surname **:** Burak Acar

Place and Date of Birth **:** Istanbul 1994

E-Mail **:** burakacaritu@gmail.com

EDUCATION **:**

- **B.Sc.** **:** 2017, Istanbul Technical University, Faculty of Electrical and Electronics Engineering, Electronics and Communication Engineering

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Acar, B., Karalar, T.** (2020). Experimental Cryptanalysis of No-equilibrium Chaotic System Based Random Number Generator. *Submitted to* the 16th IEEE Asia Pacific Conference on Circuits and System (APCCAS 2020)