NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTAMENT OF
ELETRICAL AND COMPUTER ENGINEERING

João Pereira Cabacinho

BSc in Electrical and Computer Engineering

PUFs based on Coupled Oscillators Static Entropy

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon

September, 2022

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTAMENT OF
ELECTRICAL AND COMPUTER ENGINEERING

PUFs based on Coupled Oscillators Static Entropy

João Pereira Cabacinho

BSc in Electrical and Computer Engineering

Adviser: Dr. Luís Augusto Bica Gomes de Oliveira
Associate Professor, NOVA University Lisbon

Examination Committee:

Chair: Dr. André Teixeira Bento Damas Mora,
Assistant Professor, *NOVA University Lisbon*

Rapporteurs: Dr. João Carlos Ferreira de Almeida Casaleiro,
Adjunct Professor, ISEL

Adviser: Dr. Luís Augusto Bica Gomes de Oliveira,
Associate Professor, NOVA University Lisbon

MASTER IN ELETRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
September 2022

PUFs based on Coupled Oscillators Static Entropy

Copyright © João Pereira Cabacinho, NOVA School of Science and Technology, NOVA University Lisbon.

The NOVA School of Science and Technology and the NOVA University Lisbon have the right, perpetual and without geographical boundaries, to file and publish this dissertation through printed copies reproduced on paper or on digital form, or by any other means known or that may be invented, and to disseminate through scientific repositories and admit its copying and distribution for non-commercial, educational or research purposes, as long as credit is given to the author and editor.

This document was created with Microsoft Word text processor and the NOVAthesis Word template [1].

To my parents for always supporting me.

ACKNOWLEDGMENTS

This dissertation is the culmination of several years of study, distress, setbacks, dedication, perseverance, joy, happiness, that have shaped the way I face life and the inherent challenges. There were many ups and downs, but without them there is no learning and evolution.

First, I would like to thank my supervisor, Prof. Luís Oliveira for his constant availability, for guiding me throughout this work without letting me lose the focus, for having trusted me with this work, and finally for his contagious enthusiasm in electronics that certainly helped me to keep motivated.

To Marta, for her patience in tolerating me in the most difficult moments, for not letting me ever give up by always believing in me, and for the love that was never lacking and that helped me through this journey. Without you, there would certainly have been more difficult moments. Thank you for being my companion and for always being there. You are my light.

To my friends, for their unconditional support and help and for helping to ease my mind. A special thanks to my colleagues André Félix, Carlos Sequeira, Diogo Dias, Diogo Pragosa and João Corvo, who always supported me and provided the most joyful moments along this journey.

To my father, for the example he is to me, for the demand he transmits and for always trying to teach his knowledge about any subject. Without you I wouldn't be the student I am today, and I wouldn't have the necessary rigor to get here. Thank you!

To my mother, for always believing in me and in my abilities. You were never one of the most demanding people, but your unconditional support was very important in getting through challenges and keeping me committed. Thank you for all the love, affection, and dedication.

To my brother, he was certainly the person I've annoyed the most during my life. All the TV shows you insisted I watched and all the video games I played because of you were important to clear my head and continue to overcome challenges.

Finally, to my grandmother Pilar, for having accepted me in her house throughout these years and for all her attention so that I lacked nothing. Thank you.

“You face many defeats in life, but never let
yourself be defeated.” (Maya Angelou).

ABSTRACT

We live in a digital era, this led to a shift from traditional industry to a society focused on information and communication technologies. The amount of shared information is exponentially growing every year. Protecting all this shared information is keeping everyone's privacy, is making sure the information is authentic, is keeping everyone safe.

The solution for such problems is cryptography using hardware-based, System on Chip, SoC solutions such as Random Number Generators, RNGs, and Physical Unclonable Functions, PUFs. RNGs generate random keys from random processes that occurs inside the system. PUFs generate fixed random keys using random processes that originated in the fabrication process of the chip. The objective of this work is to study and compare a static entropy source based on coupled relaxation oscillators against a state-of-the-art architecture like the static entropy source based on ring oscillators, in advanced 130nm technology. The characteristic studied were, area, power consumption, entropy, resistance to temperature, and supply voltage variations.

Compared to the ring oscillator implementation, the static entropy source designed showed promising results as a static entropy source, however, it revealed poor results in terms of area, power consumption, and entropy. Such results mean, the coupled relaxation oscillator may not be good at generating random numbers, however, it may be good at keeping its state when under temperature and supply voltage variations.

Keywords: Cryptography, Random, Entropy, Key, Oscillator, Coupled, PUF,

Resumo

Vivemos numa era digital, o que levou a uma mudança da indústria tradicional para uma sociedade centrada sobre as tecnologias da informação e da comunicação. A quantidade de informação partilhada está a crescer exponencialmente todos os anos. Proteger toda esta informação partilhada é manter a privacidade de todos, é garantir que a informação é autêntica, está a manter todos seguros.

A solução para tais problemas é a criptografia com base em soluções de *hardware*, *System on Chip*, SoC tais como Geradores de Números Aleatórios, RNGs e Funções Físicas Inclonáveis, PUFs. Os RNGs geram chaves aleatórias a partir de processos aleatórios que ocorrem no interior do sistema. Os PUFs geram chaves aleatórias fixas utilizando processos aleatórios que se originaram no processo de fabrico do chip. O principal objetivo deste trabalho é estudar e comparar uma fonte estática de entropia baseada em osciladores de relaxação acoplados contra uma arquitetura de estado de arte como a fonte estática de entropia baseada em osciladores de anel, em tecnologia avançada de 130nm. As características estudadas foram, a área, o consumo energia, a entropia, e a resistência à temperatura e variações de tensão de alimentação.

Em comparação com a implementação do oscilador do anel, a fonte estática de entropia projetada mostrou resultados promissores como fonte estática de entropia, no entanto, revelou maus resultados em termos de área, consumo de energia e entropia. Estes resultados significam que o oscilador de relaxação acoplado pode não ser bom a gerar números aleatórios, no entanto, pode ser bom para manter o seu estado quando sujeito a variações de temperatura e tensão de alimentação.

Palavas chave: Criptografia, Aleatório, Entropia, Chave, Oscilador, PUF.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

CMOS	Complementary Metal-Oxide-Semiconductor
CRP	Challenge-Response Pair
Cusums	Cumulative Sums
HD	Hamming Distance
IoT	Internet of things
NBTI	Negative-Bias-Temperature-Instability
NIST	National Institute of Standard and Technology
NMOS	N-channel Metal-Oxide-Semiconductor
NVM	Non-Volatile Memory
PKI	Public Key Infrastructure
PMOS	P-channel Metal-Oxide-Semiconductor
PRNG	Pseudo Random Number Generator
PUF	Physical Unclonable Function
RNG	Random Number Generator
RO	Ring Oscillator
SRAM	Static Random-Access Memory
TRNG	True Random Number Generator
VCR	Voltage Controlled Resistor

VTC	Voltage Transfer Characteristics

1

INTRODUCTION

1.1 Motivation

Internet of Things (IoT) managed to integrate wireless communication into our daily lives. IoT enables a lot of devices from sensors to automobiles to communicate with each other and share their data. IoT systems are embedded in our daily lives in various forms. The vast areas of application, for example, the transport sector, healthcare, industry, blockchain, and building smart atmospheres are enabling a general digital transformation. While IoT networks certainly improved the quality of our lives allowing for new and enhanced services, exponential growth, power and computing limitations, high mobility, and communication, introduce an unprecedented number of security threats while allowing for attacks to propagate quickly through the network [2].

Several causes constrain the security of IoT devices including limited energy available, limiting computing resources, and the heterogeneous nature of the network [3]. The energy and computing constrain of IoT devices do not allow for the implementation of heavy cryptography and security protocols used on normal networks. The scale of IoT devices opens chances for careless programming which is a gateway for malware and backdoors schemes. The heterogeneous nature of the network results in dissevered security, identification, and authentication standards, which, increases the complexity of the security challenge [4].

The characteristic stated above, and the estimate of 75 billion IoT devices connected by the year 2025 result in the inefficient performance of conventional security mechanisms, leading to crucial information being unsecured [5]. Unsafe data can have major costs to our lives thus, the information must be received and sent to an authenticated user. Also, for example, e-wallets for cryptocurrencies work by storing the private keys that are necessary to manage the cryptocurrency account as proof of public key ownership, allowing the user to do payments and transfers, the economic value such e-wallets may hold is very high thus, need high levels of security.

Security services should ensure confidentiality, only the authorized users can read the data; integrity, the data sent cannot be manipulated; authentication, confirmation of identity; nonrepudiation, no other user have sent the information; digital signature, verify the integrity, perform authentication and nonrepudiation [6].

These services are achieved through cryptographic keys and algorithms used between the sender and the receiver. Encryption by itself does not guarantee security, so does not provide enough integrity. Simply encrypted messages can be decrypted, also, third parties can still transmit encrypted packets in the network. A widely used encryption method is called public key infrastructure (PKI). PKI systems use two types of keys, public and private, as the name suggests, private keys are kept secret while public keys are known by the public where one key is used com encryption and the other for decryption. The private key should be reliable, robust, and perfectly reproducible [5]. Because of those characteristics, the keys are usually stored in the non-volatile memory (NVM) however this kind of memory is vulnerable to physical attacks.

There are two main types of solutions, hardware-based and software-based solutions.

- The software-based solution is the most common solution on traditional networks, they rely on software that is based on mathematical methods to protect the information sent. However, because they are based on mathematical methods (e.g., a discrete logarithmic problem), they require processing and memory power, resources often scarce on IoT which may lead to poorer methods implemented thus, less security [5]. A software-based solution might not be the best option for IoT devices. Also, this kind of procedure is effective as a security measurement because the logarithmic problems are not easily solved using today's computers however with the increase of computing power and quantum computers, these kinds of security mechanisms will become obsolete.

- Hardware-based solutions can be achieved using a dedicated integrated circuit as systems on chip (SoC) to perform cryptography and store keys. Because data is stored and generated by the physical properties of the chip instead of memory and software, this solution can prevent read-and-write access to data and is more resilient against other attacks. [5] Also, since is designed as an integrated circuit, it is possible to reduce power consumption and improve encryption. One of the main disadvantages of hardware-based security solutions is that they are susceptible to physical attacks where if the chip is stolen, attackers can clone the device which means they can clone the key and get access to the information originally secured.

However, physically unclonable functions (PUFs) and true random generators (TRNGs) can solve the previously stated problem.

1.2 Proposed goals

The main goal of this work is to implement a static entropy source based on coupled oscillators using CMOS 130nm technology. To study its properties using already established tests, and to compare the results obtained with state-of-the-art architecture. To achieve the main objective of implementing a static entropy, the following goals were set:

- Review the most common PUF configurations
- Review oscillators
- Understand statistical tests to evaluate PUFs
- Implement static entropy source based on coupled oscillators
- Study the implemented static entropy source
- Compare the designed static entropy source with state-of-the-art implementations

1.3 Contributions

In this dissertation a static entropy source based on coupled relaxation oscillator was developed using advanced 130nm technology. This static entropy source can be used to be part of a PUF.

Compared to the state-of-the-art implementation, the ring oscillator static entropy source, the coupled relaxation static entropy source, it did not perform up to expectations in terms of area, power consumption and entropy. However, it showed promising results when under temperature and supply voltage variations showing it can be robust under such variations.

Even though the circuit did not perform up to par against the ring oscillator, the results show that that this circuit has potential as a static entropy source. Thus, further study is needed to improve the static entropy source developed.

1.4 Report Structure

This work is divided into six chapters, structures as follows.

Chapter 1 presents the subject being researched as well as the motivation behind the search for a new solution by also listing the goals for the dissertation.

Chapter 2 is theoretical explanation useful for the understanding of the work developed of this dissertation.

Chapter 3 is an overview of the current state-of-the-art. Explores basic concepts of PUFs and statistical tests to evaluate the latter and, shows some previous scientific work.

Chapter 4 is where the work developed is presented, the circuits used and respective dimensioning.

Chapter 5, the results obtained from the work developed in Chapter 4 are revealed, also a discussion and comparison of such results.

Finally, Chapter 6, the final conclusions about the work accomplished are drawn. And some potential future work is discussed.

2

Background

2.1 Entropy

The entropy applied to key generation was explained by M. Alioto [6]. Key generation is crucial for secure systems for examples for cryptographic operations or to identify and authenticate devices. The key generation can be generated from static entropy for applications where the secret key only needs to be generated once or generated from dynamic entropy for applications where the secret key needs to be fresh every time the system requires a key.

The secure strength of a determined key can be measured by the number of possible key guesses. Because the key is binary, the number of key guesses and consequently the secure strength is

$$Strength\ of\ the\ System = 2^{keylength \cdot entropy} \tag{2.1}$$

Entropy is the amount of information held by each bit. A bit represents two values, 0 and 1, so, ideally, for random key generation every bit must have the same probability of being generated. Entropy categorized as a value that ranges from 0 to 1. An entropy of 0 represents no information which translates to a perfectly predictable value which mean the probability generating a bit 0 ($\mathrm{Pr}[0]$) is either 0% or 100%. A perfectly random key has an entropy of 1 which means $\mathrm{Pr}[0] = 0.5$, thus, $\mathrm{Pr}[1] = 0.5$. In practical systems $\mathrm{Pr}[0]$ is not exactly 0.5, this means the entropy will be less than 1 thus reducing the effective keylength $[keylength \cdot entropy]$ consequently degrading the strength of the system as show by equation (2.1).

Depending on the application, entropy can be defined in various ways for example by the Shannon entropy and the Min-entropy.

For moderately strict security requirements the Shannon entropy is popularly used. The Shannon entropy was first defined by Shannon [7] and it states that in the case of bit generation where there are only two possibilities with probabilities p and $q = 1 - p$, the entropy is

$$entropy = -(p \log p + q \log q)$$

(2.2)

Plotted in Figure 2.1. The entropy is considered adequate when the effective key length is degraded by less than 1bit. For a 256-bit key this means the acceptable entropy is given by

$$256 \cdot entropy \geq 255.5 \Leftrightarrow$$
$$\Leftrightarrow entropy \geq \frac{255.5}{256} \Leftrightarrow$$
$$entropy \geq 0.998$$

For a 256-bit key, following the Shannon entropy, the minimum acceptable entropy is 0.998, following equation (2.2), this translates to $0.42 \leq \Pr[0] \leq 0.58$.

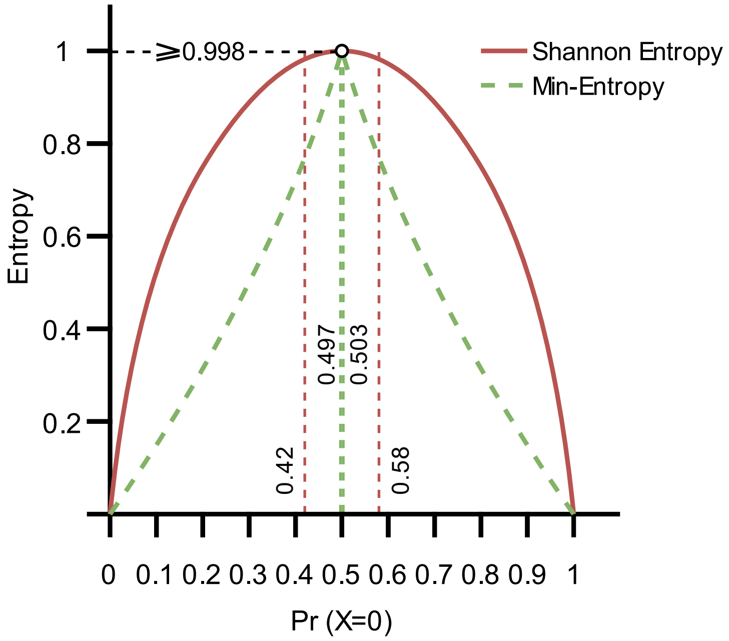


Figure 2.1 Shannon Entropy and Min-Entropy with the probability of the bit being 0 and their acceptable key length degradation. Extracted from [6].

The Min-entropy is a measure of the key unpredictability, defined by Renner. R and Wolf. S [8]. It is described as the probability of successfully guessing a certain key, a 1-bit key has a 50% chance of guessing the right value. For Min-entropy, the entropy target of 0.998 converts to a more stringent requirement, $0.497 \leq \Pr[0] \leq 0.503$. The Min-entropy is plotted in Figure 2.1.

Both Shannon entropy and Min-entropy dictate the acceptable 0/1 bias of circuit implementations for static and dynamic entropy.

2.1.1 Static Entropy

As explained by Alioto [6]. Static entropy is used to generate fixed keys thus, it is the desired entropy for PUFs. The generation of random fixed keys is essential in hardware security and is mostly used as chip ID to identify a device and for the protection of software intellectual property. Conventionally, static entropy is generated off-chip and then stored in a nonvolatile manner, for example, fuses and flash memories. However, such ways to store the key are vulnerable to a large well-known attack at the software level and as well physical attacks.

To generate the same challenge, the generator needs to be resilient to aging, temperature, and voltage oscillations effect to decrease undesirable variations. Instead, the entropy can be derived from the physical differences between transistors originating from the process and mismatch variations [9].

2.1.2 Dynamic Entropy

By Alioto [6], Dynamic entropy is used by RNGs to generate new random keys whenever a new key is needed. RNGs can be divided into TRNGs and PRNGs. TRNGs use random physical processes to generate a random number and thus are much more secure because there is no correlation between numbers. PRNG uses deterministic systems to generate random numbers, thus, every generated key is correlated and by understanding how the system works, it is possible to predict the next number to be generated.

2.2 Oscillators

An electronic oscillator is a circuit designed to generate a periodic signal out of constants, oscillators can be linear (sinusoidal) or non-linear (non-sinusoidal/relaxation) [10]. This means the oscillator translates energy from the DC power to generate a periodic signal with a certain frequency and amplitude, AC power, determined by the oscillator itself [11].

Since the oscillator generates a periodical signal from constants, it needs to have a time reference in its system. A time reference is a system that can generate an output signal through a known time behavior as shown in Figure 2.2.

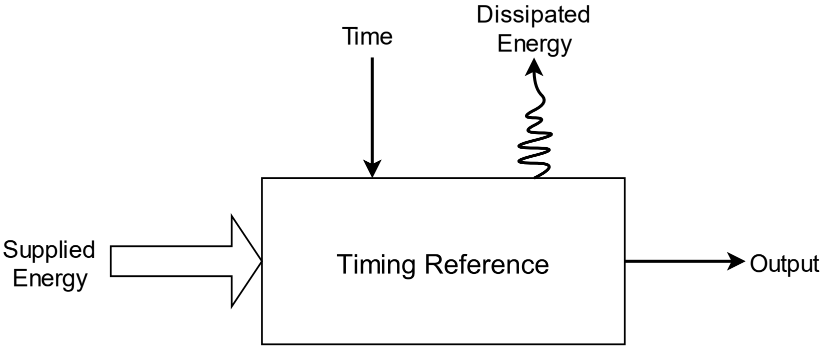


Figure 2.2 Time Measurement. Extracted from [11].

In the previous figure, time is used to quantify the variation of the system over time. At the input, energy is supplied for the system to operate, in a system with no energy dissipated, energy only must be supplied to start the system, however, if dissipated energy is considered energy needs to be continuously supplied. At the output, the influence of time on the system is measured [11].

Oscillators can be classified by the pole pattern which characterizes their timing reference, this makes it easy to observe the properties of the oscillation.

First-order oscillators will have the pole located in the real axis, if this pole is in the origin, the system is an ideal integrator, when a constant is applied to the input of that integrator, the output signal changes linearly with time, thus, a time-variant signal with a well-known time behavior is created. When there is a constant input signal α in the integrator and $E_o(t)$ on the output, then:

$$E_0(t) = \int_0^t \alpha \, d\tau$$

For a constant input, the ideal integrator will output a linear time-varying signal. In an electrical circuit, an example of an integrator is the capacitor [12]. However, in a real system, the integrator is not ideal, thus, the pole will not be centered on the origin, and thus there is no longer a linear relationship between the constant input and the output. If the pole is shifted to the right of the origin, $E_0(t)$ will increase exponentially. If the pole is shifted to the left plane, $E_0(t)$, the slope will decrease over time showing the presence of losses in the system and the oscillation is damped [12]. Also, this kind of oscillator needs additional steps to obtain an oscillation because the integrator by itself does not change the signal of the constant meaning it integrates linearly to infinity[12]. The pole interaction with the system in a first-order oscillator can be seen in Figure 2.3.

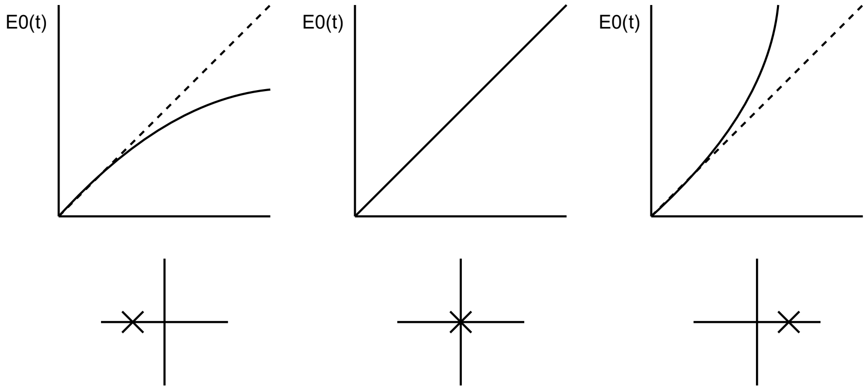


Figure 2.3 System behavior using one pole [11].

This kind of oscillator, due to its nature is used to create non-sinusoidal signals originating from the relaxation oscillator.

Second-order oscillators have two dominant poles in the imaginary axis, when this pole is in the imaginary axis, the output signal is stable in amplitude and frequency. This kind of oscillator does not need additional steps to generate oscillations, the oscillation happens naturally. The time-varying signal is achieved using a resonator. A resonator is a second-order device that oscillates at a resonant frequency defined by its components. An ideal LC resonant circuit is shown in Figure 2.4.

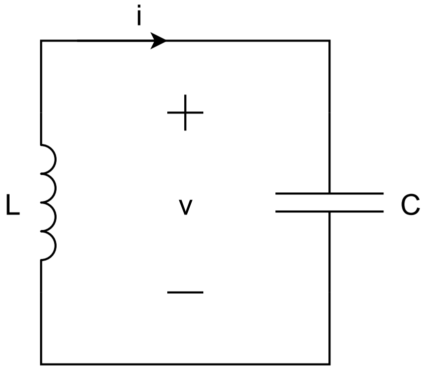


Figure 2.4 Resonator circuit using a inductor and a capacitor.

Ideally, this kind of circuit functions by connecting a charged capacitor to an inductor. The charged capacitor has an electric field between its plates which will result in a voltage across the capacitor. Since the inductor acts as a load, this voltage will drive current through the inductor increasing its magnetic field. As the voltage across the capacitor reaches close to zero and because magnetic fields oppose current changes, a voltage across the inductor is induced which will induce a current to charge the capacitor. In this kind of circuit, the resonator will oscillate with a frequency given by:

$$f_0 = \frac{\omega_0}{2\pi} = \frac{1}{2\pi\sqrt{LC}}$$

Thus, the values of inductance and capacitance will determine the pole placements.

Because this kind of oscillator oscillates naturally, it creates a sinusoidal signal thus second-order oscillators are used to produce linear oscillators.

Third-order oscillators contain three poles. Two poles on the imaginary axis like the second order oscillator and a third pole in the real axis like the first order oscillator, like in the first oscillator, this pole will usually be on the left side of the plane meaning it has a positive real part which indicates the presence of losses in the system [12].

2.2.1 Linear Oscillator

Linear oscillators produce a sinusoidal signal. A conventional block diagram of a linear oscillator is shown in Figure 2.5 which contains an amplifier block of gain A and a frequency selective second-order feedback block with gain β. The feedback block can be composed of an RC circuit or LC circuit. This kind of oscillator can be analyzed using the Barkhausen criteria.

The Barkhausen criterion is a mathematical condition that characterized the region of oscillatory instability of linear oscillators with a feedback loop, it was established by studying the poles of the system and their relation to the oscillator behavior by reducing complex equations into two real variables [13]–[15]. Analyzing the linear oscillator represented by Figure 2.5 using the Barkhausen criteria, the necessary conditions to generate a sinusoidal oscillation, are $|A\beta| = 1$ and $\angle A\beta = 2k\pi$, or an integral multiple of 360º [13].

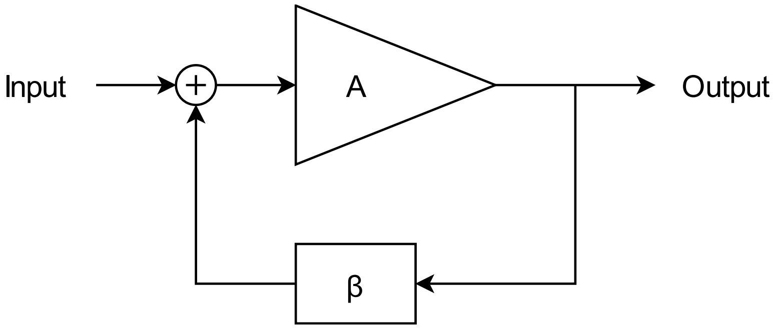


Figure 2.5 Linear Oscillator block diagram. Extracted from [13].

The following information about the Barkhausen criterion and feedback loop is based on [14]. To achieve the desired phase shift, reactive elements must be used in the feedback loop, since reactive elements have a resonance frequency, the phase shift is dependent on the reactive element's resonance frequency. Meaning the Barkhausen criterion is only satisfied at this natural frequency. If $|A\beta| < 1$, the oscillation will decay over time until it perishes. If $|A\beta| > 1$, every time the signal goes through the feedback loop it will increase in amplitude, thus the oscillation will be increasingly higher in amplitude, this increase is only limited by the power supply of the oscillator and can lead to a nonlinear operation. In practical oscillators, the loop gain, $|A\beta|$, need to be slightly higher than 1 to compensate for the noise and/or variations in transistor and circuit parameters. The Barkhausen criterion is only applicable to linear oscillators, this means it can only be utilized for sinusoidal oscillators like the Wien Bridge Oscillator, the Colpitts and Hartley Oscillator, these following examples are based on Senani *et.al* [13], and Schubert and Ernest [14] explanations.

The Wien Bridge Oscillator as shown by the circuit of Figure 2.6 a), is a type of RC oscillator and it uses an op-amp as the non-inverting amplifier and the reactive feedback loop is composed of a series-shunt topology using two resistors and two capacitors forming a second-order band-pass filter, the resulting center frequency is

$$\omega_0 = \frac{1}{RC} \tag{2.3}$$

The center frequency it's chosen such that the phase shift of the band-pass filter becomes zero. Since the non-inverting amplifier does not have a phase shift, the total phase shift around the loop is zero.

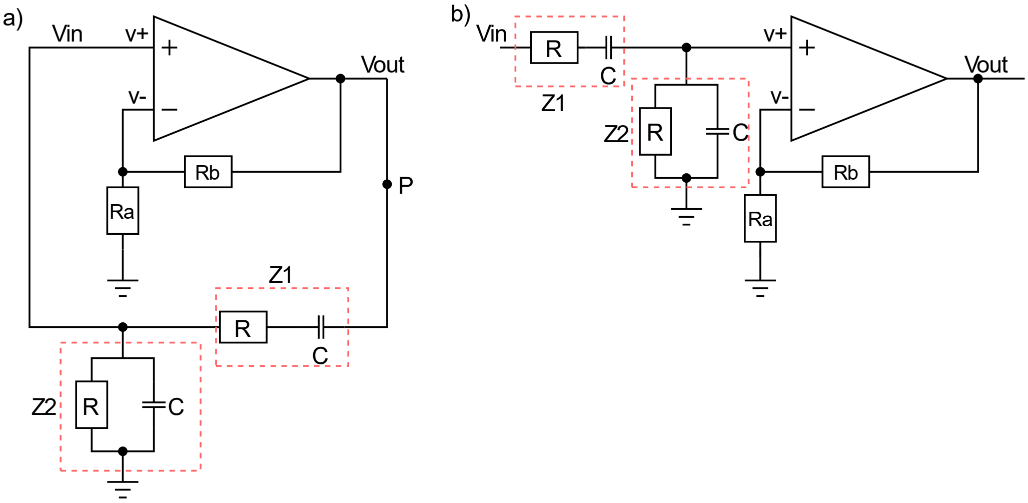


Figure 2.6 Wien Bridge Oscillator. a) Closed loop. Extracted from [13]. b) Open loop.

Because the band-pass filter is passive, it will have a voltage gain smaller than 1, the non-inverting amplifier must have a voltage gain equal to $\frac{1}{band\text{–}pass\ filter\ voltage\ gain}$.

The open-loop circuit is shown in Figure 2.6 b). Following the ideal op-amp model:

- The infinite input impedance of the op-amp
- Infinite open-loop gain
- Zero offset
- $V_{out} = A.(v^{+} - v^{-})$

The open-loop transfer function can be obtained as a relation between the op-amp loop and the feedback network.

$$\frac{V_{out}}{V_{in}} = \frac{Z_1}{Z_1 + Z_2} \times \left(1 + \frac{R_b}{R_a}\right) \Leftrightarrow$$

$$\Leftrightarrow \frac{V_{out}}{V_{in}} = \frac{sCR \times \left(1 + \frac{R_b}{R_a}\right)}{s^2C^2R^2 + 3sCR + 1} = A\beta$$

Following the Barkhausen criteria stated before, the necessary conditions for oscillation are $|A\beta| = 1$ and $\angle A\beta = 0$ which results in

$$|A\beta| = \left|\frac{sCR \times \left(1 + \frac{R_b}{R_a}\right)}{s^2C^2R^2 + 3sCR + 1}\right| = 1$$

And

$$\phi = \arctan(\infty) - \arctan\left(\frac{3\omega RC}{1 - \omega^2R^2C^2}\right) = 0$$

For $s = j\omega$ and the frequency of oscillation (ω) given by equation (2.3),

$$|A\beta| = \left|\frac{\left(1 + \frac{R_b}{R_a}\right)}{3}\right| = 1$$

Thus,

$$\left(1 + \frac{R_b}{R_a}\right) = 3$$

And the condition of the oscillation is given by

$$\frac{R_b}{R_a} = 2$$

Which is the ration of R_b and R_a.

The Colpitts and Hartley Oscillators are the two most used LC Oscillators. This type of oscillator is more suitable for high-frequency applications compared to RC Oscillators. Despite RC Oscillators having more power dissipated compared to LC oscillators, due to the resistance, these are preferred over LC in low frequencies because of the major downside of using inductors in integrated circuits, their size, however, since the frequency and the inductor size is inversely related, in high frequencies the use of inductor becomes acceptable thus LC oscillators are preferred over RC oscillators in high frequencies.

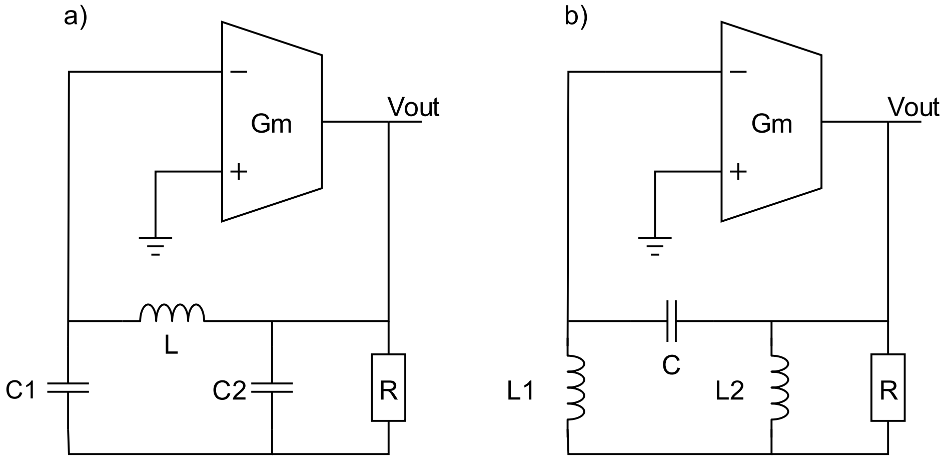


Figure 2.7 Linear Oscillators. Extracted from [13]. a) Colpitts Oscillator. b) Hartley Oscillator.

The Colpitts and Hartley Oscillators with an OTA-based implementation are shown in Figure 2.7 a) and Figure 2.7 b) respectively. In this example, it is used an inverting OTA-amplifier with a gain of $-G_m$. As reactive feedback loops an LC circuit.

Analyzing the Colpitts Oscillator represented by Figure 2.7 a), the frequency of oscillation is given by

$$f_0 = \frac{1}{2\pi\sqrt{LC_T}}$$

And C_T is

$$C_T = \frac{C_1 \times C_2}{C_1 + C_2}$$

The condition required to maintain oscillation following the Barkhausen criterion, $|A\beta| = 1$ and $\angle A\beta = 0$, are given by

$$g_m R \geq \frac{C_1}{C_2}$$

For the Hartley oscillator, the capacitors are switched by inductors and vice versa as represented by Figure 2.7 b), the oscillation of frequency is

$$f_0 = \frac{1}{2\pi\sqrt{L_T C}}$$

And L_T is

$$L_T = L_1 + L_2$$

Finally, the condition of oscillation is

$$g_m R \geq \frac{L_1}{L_2}$$

2.2.2 Non-Linear/Relaxation Oscillator

Relaxation oscillators are oscillators that produce a non-sinusoidal signal such as a triangle or square wave. In Relaxation oscillators, only one pole is used in the timing reference. However as discussed before in this Chapter, this is not enough to create a periodical signal with a specific frequency. To counteract this problem additional components are needed to change the signal of the integration constant.

As described by Westra [11] and Verhoeven [12]. To generate a periodic signal, the signal of the integrator is measured, and depending on the sign of the integration constant, the signal will be increasing or decreasing. Then the signal is compared either by a bottom limit (E_l) or an upper limit (E_h), when the signal reaches either threshold, the sign is changed. To change the sign of the integrator, the system needs feedback, and the sign of integration is stored in a memory. With this procedure it is possible to go from a time-varying signal like the one in Figure 2.8 a) to the periodical signal in Figure 2.8 b).

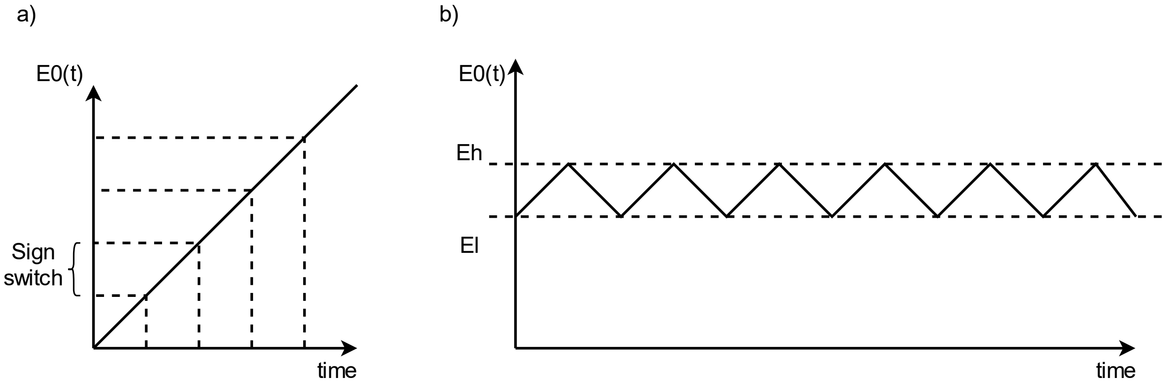


Figure 2.8 Non-Linear Oscillator. a) Integrator signal. Extracted from [11]. b) Periodic signal. Extracted from [12].

Relaxation oscillators have four crucial functions:

- Integration
- Comparison
- Switching the sign of the integration constant
- Memorization

From the functions stated above, using a block diagram, it is possible to model an oscillator where each block represents each function like the one depicted in Figure 2.9.

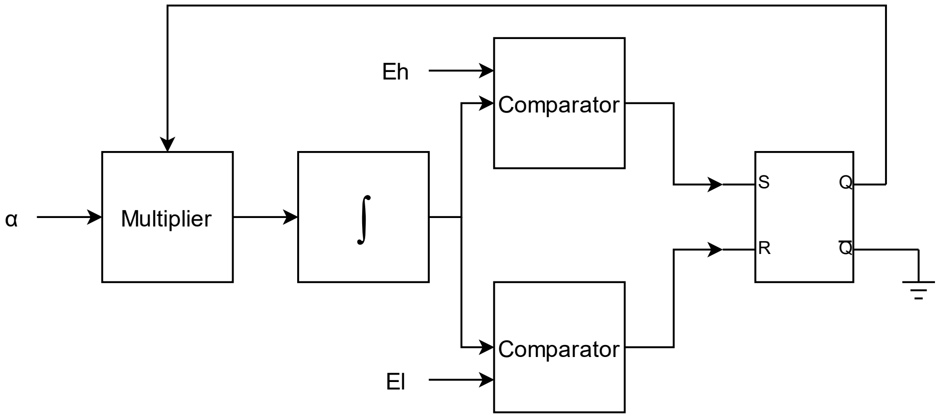


Figure 2.9 Basic Relaxation Oscillator block diagram. Extracted from [12].

The oscillator represented in Figure 2.9, is the basic regenerative oscillator. As explained by Westra [11], starting from left to right, the multiplier is responsible for switching the sign of the integration constant which will determine which reference E_h or E_l the signal, $E_o(t)$, is changing to. The two reference levels are constant and define the top and bottom limits of the signal as shown in Figure 2.8 b), level comparison is done by two comparators both connected to a latch. The flip-flop is the memory element, depending on the comparator output, the latch will output either 1 or -1 in other to determine the sign of the constant. The output of the latch is then injected back into the multiplier which will change or not the integration constant.

To reduce the complexity of the oscillator, the number of blocks can be reduced using elements that produce several of the necessary functions. Using fewer blocks simplifies the bias of the circuit the latter may have better speed performances. This reduction can be achieved utilizing a Schmitt trigger.

The Schmitt trigger has a hysteresis-like behavior as shown in Figure 2.10. It has two stable states (M and -M) and an internal reference signal with two different values (T and -T), when the input signal is equal to or higher than the reference level, the Schmitt trigger switches state.

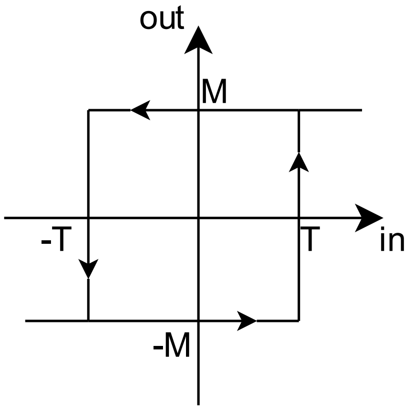


Figure 2.10 Schmitt Trigger transfer function. Extracted from [16].

As described by Verhoeven [12]. The Schmitt trigger is also a memory element as when it reaches a certain state, it maintains the output if the input doesn't reach the reference signal respective to the other state. Because the stable states output opposite values, those can be used as the integration constant α and -α thus, the multiplier element is no longer needed. The Schmitt trigger can perform the functions of comparison, switching the sign of integration and the memory.

One of the simplest first-order oscillators can be built by only using an integrator and a Schmitt trigger. In Figure 2.11, a high-level model of this relaxation oscillator is depicted using a block diagram.

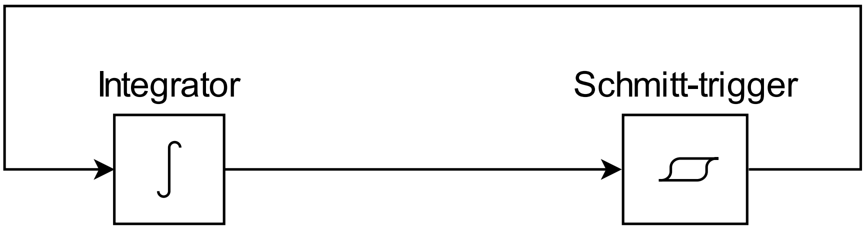


Figure 2.11 High-level model of relaxation oscillator using a Schmitt-trigger.

The relaxation oscillator in Figure 2.9 is optimized easily since every block can be optimized for the only function it performs and one miss-optimization will have less influence on another component. However, its complexity is a big downside of this oscillator and due to the number of components, it may have speed problems and the bias might be complicated. In the relaxation oscillator in Figure 2.11 the optimization will be more complicated. However, it is a much simpler implementation and easier to bias.

2.2.3 Non-Linear/Relaxation Coupled Oscillator

As described by Westra and Verhoeven [11], [12]. When an external signal has a well-defined time behavior it can be used to influence the timing of a first-order oscillator. Then, the timing properties of the external signal will be taken over by the first order oscillator and the periodic signal generated will be related to the external signal. Since oscillators output a signal with a well-defined time behavior, it is possible to synchronize two first-order oscillators to each other. The signal from the first oscillator is used to synchronize the second oscillator and the signal from the second oscillator can be used to synchronize the first oscillator. This is what is called a coupled system.

Coupling oscillators is done to improve the noise behavior. In coupled oscillators, the carriers from both oscillators become strongly correlated and most noise sources remain un-correlated. If all noise sources and uncorrelated, the noise power is reduced by half.

The following coupling explanation is based on the work of Luis Oliveira [17]. By adding a soft limiter after the integrator of the relaxation oscillator depicted in Figure 2.11, a new output with a 90º phase difference compared to the output of the Schmitt trigger is obtained. Represented by Figure 2.12, the soft-limiter output can be used to synchronize a second oscillator and the soft-limiter output of this second oscillator can be used to synchronize the first oscillator. By coupling two oscillators this way, a regenerative cross-coupled quadrature relaxation oscillator is obtained.

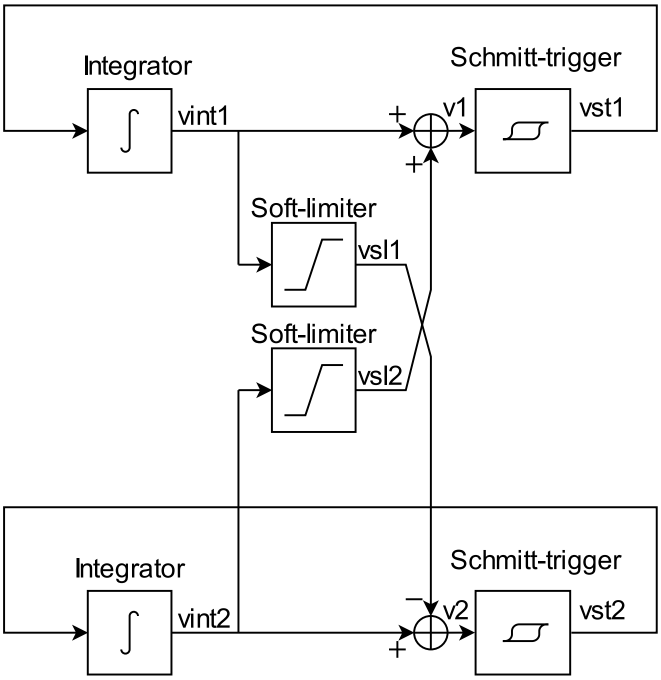


Figure 2.12 High-level model of cross-coupled relaxation oscillator using a soft-limiter. Extracted from [17].

In a cross-coupled oscillator, there is not a master nor a slave oscillator, the balance is achieved by both oscillators. Both oscillators are designed to operate at the same frequency and due to the soft limiter, the output at each Schmitt trigger will be in quadrature. Since the soft limiter is an amplifier with saturation, the input signal gets amplified (v_{int}) however, due to the saturation, the output will resemble a square (v_{sl}) as shown in Figure 2.13 a). The soft limiter output is added to the integrator output signal of the other oscillator, this will create a stepper slope in the transition region as shown by v_1 in Figure 2.13 b).

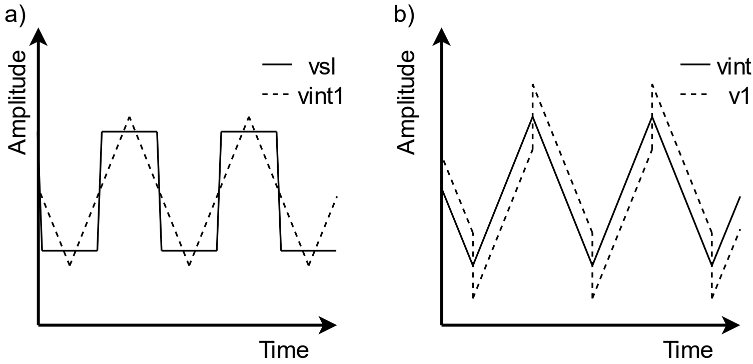


Figure 2.13 Cross Coupled Relaxation Oscillator signals. Extracted from [17]. a) Input (v_{int1}) and output (v_{sl}) of soft limiter. b) Integrator signal (v_{int1}) and coupling output (v_1).

The signal before the soft limiter is v_{int1} and after is v_1. This stepper slope enables the switching time to be less sensitive to noise. Higher soft limiter gain minimizes the influence of noise on the transition time.

2.2.4 Ring Oscillator

As described by Silva, Manuel [18] and Charan Sarkar et al.[19] .This oscillator is especially appealing because of its simplicity, it is easily designed in state-of-the-art technologies for example CMOS, it can achieve high-frequency oscillations at a low power cost, and it can provide outputs at different phases.

The ring oscillator is composed of n odd number of inverters connected as shown in Figure 2.14.

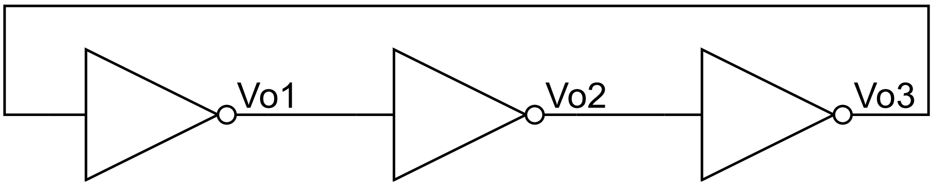


Figure 2.14 Ring oscillator. Extracted from [18].

Each inverter delivers an output that is phase shifted by $\frac{\pi}{n}$ compared to the previous inverter output.

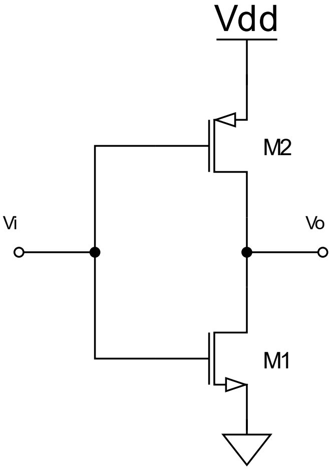


Figure 2.15 Inverter.

The singled-ended ring oscillator consists of inverters containing a PMOS and an NMOS transistor as shown in Figure 2.15. Both transistors perform switching between ON and OFF.

The transistor M_1 is responsible for the pull-down transition and the transistor M_2 is responsible for the pull-up transition.

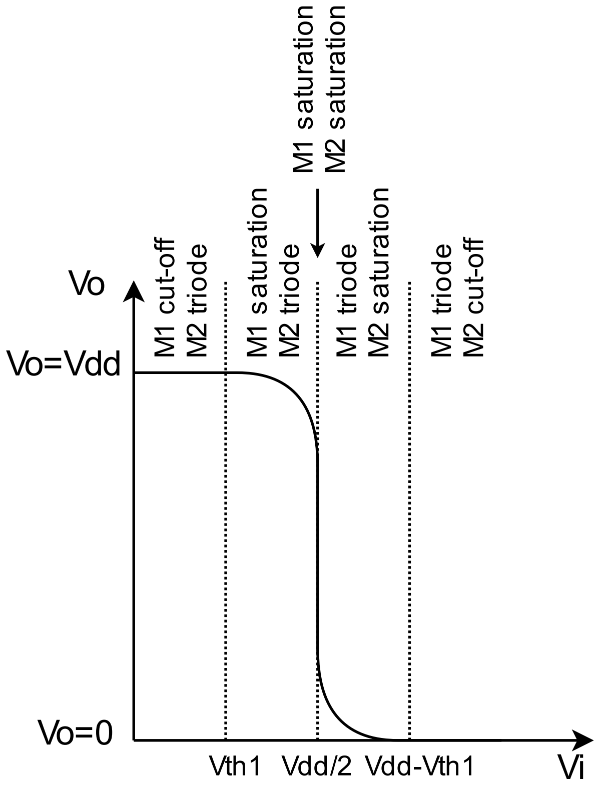


Figure 2.16 Transfer function characteristic of the inverter.

The transfer function characteristic of the inverter is in Figure 2.16. When v_I goes from low (v_I=0) to high (v_I=V_{dd}). While v_I<V_{th1}, V_{th1} is the threshold voltage of the transistor M_1, M_1 is in the cut-off region and M_2 is in the triode region, v_O=V_{dd}. When v_I>V_{th1}, the transistor M_1 changes to the saturation mode, M_2 keeps in the triode mode and v_O starts to decrease. At one point, M_2 changes to saturation mode, and both transistors are in saturation, this is when v_O abruptly decreases. When M_1 reaches the triode region, v_O starts to decrease at a slower rate. Finally, when v_I>V_{dd}-V_{th2}, V_{th2} is the threshold voltage of M, M_2 is in the cut-off region, and v_O=0V.

2.3 PUFs

Inspired by the works of Alioto [6], and Shamsoshoara et al. [5]. Traditionally, a key generation used an out-of-chip static entropy generator, and the key was stored in non-volatile memories. However, these types of memories are known to be vulnerable to a wide range of hardware and software attacks. PUFs aim to overcome the limitations associated with such methods, for that, it must ensure that by physical inspection, the secret should not be exposed and that the secret key should only be available only when the ship is powered on, preventing the possibility of a malicious attack to retrieve the key that it is not being required. These properties do not allow hackers to have access to the key since they cannot clone the inherent properties of the device even with physical access.

For an external stimulus called a challenge, the PUF will produce an output called the response. Each challenge and response form a pair called Challenge Response Pair (CRP) and ideally, each challenge will produce a perfectly repeatable output making CRPs independent from each other. There are two classes of PUFs depending on their CRPs. PUFs are classified as weak PUFs when there is a limited number of CRPs and strong PUFs when the number of CRPs increase exponentially with the area of the PUF.

PUFs rely on the unique physical properties introduced in the fabrication of the device to generate static entropy and extract a unique random key. To generate the same key from the process and mismatch variations, PUFs need to emphasize the latter variations and diminish the effects of all other variations by making the response independent of voltage and temperature variations, by maintaining the response consistent throughout the life of the device (aging), and by maintaining the statistical properties of the response from die to die.

2.3.1 Weak PUFs

Weak PUFs have a linear relationship with the number of CRPs. The number of responses is a function of the number of components used for the generation of CRPs. The limited number of CRPs results in stable responses robust to environmental conditions. However, it also means less security thus, for this class of PUFs it is crucial not to reveal any CRP and the latter ideally should be encrypted. Weak PUFs are mostly used for secret key generation, for example, chip ID, [20], [21] and lightweight encryption [22].

2.3.2 Strong PUFs

In strong PUFs, the number of CRPs increases exponentially with the area of the PUF. The high number of unique CRPs will prevent brute force attacks where all the responses are applied to get access to the system. Also, the high number of CRPs may offer stronger cryptographic strength due to generating longer keys.

2.3.3 Architecture

PUFs can be seen as a combination of different blocks with each block having its objective. Typically, PUF is structured as shown in Figure 2.17 which shows that PUFs can be divided mainly into 3 different components: entropy generations, entropy extractor, and post-processing block which is optional [6], [9].

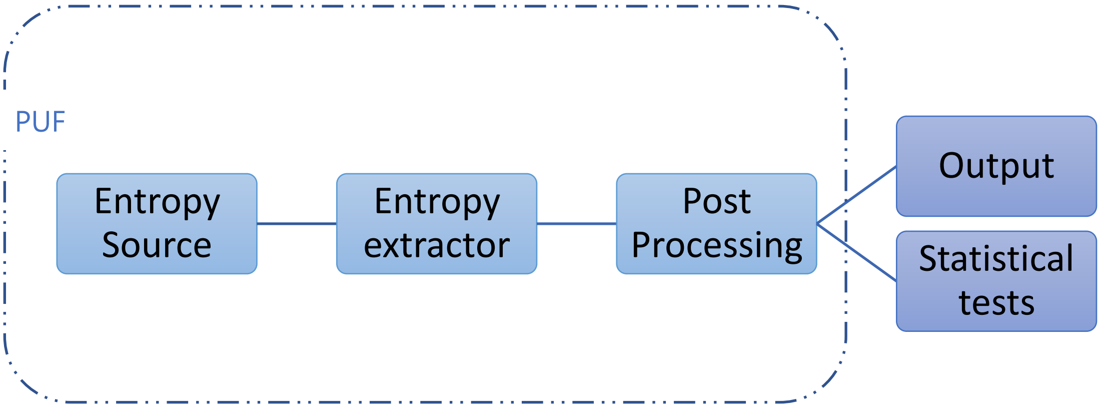


Figure 2.17 Block structure of a typical PUF. Based on [23].

- The first block is responsible for generating the static entropy through the physical properties of the electrical components. This block usually uses oscillators and switching components to produce the entropy in conjunction with a multiplexer commanded by the challenge, to increase randomness and stability, and/or also a counter to count the different pulses generated by the multiplexers. For example, connecting two different ring oscillators to a multiplexer where the challenge chooses which signal passes through followed by a counter.
- The second block will sample the output signal of the entropy source and will extract a discrete sequence of bits. This section has mostly used a comparator, an arbiter, D flip-flop, however, any other component that can perform a selection between two logic gates can be used.
- The Post-processing block is used to reduce the BER restoring some entropy and randomness that may have been lost due to any kind of perturbation. Some techniques include spatial majority vote (SMV), temporal majority vote (TMV), and dark bit masking.

The output of the PUF is formed by a sequence of ones and zeros. To certify the quality of the sequence created, it can be directed to further testing for example statistical tests to evaluate the randomness of the sequence thus confirming its use on security.

2.3.4 Quality Metrics

PUF quality can be assessed by several metrics to quantify the PUF response, including, stability, uniqueness, reproducibility, identifiability, randomness, sensitivity to variations, area, and energy efficiency Alioto[6], Samsoshoara[5], Podeti, et al.[24], Alioto et al.[25]

Stability is measured by the cumulative sum of unstable bits (flipping bits) over a certain number of iterations. Another method to evaluate stability is using the bit error rate (BER) which corresponds to the average of instability bits over the response of the PUF [6].

Uniqueness is quantified by the difference between the responses of each die of the PUF applying the same challenge (inter-PUF Hamming Distance) [5], [24]. The ideal inter-PUF Hamming Distance (inter-HD) is 50%, which means the response of two dice should vary by 50% if the responses are perfectly random [6].

Reproducibility is quantified by the average bits that differ between responses from different challenges at different conditions, the intra-PUF Hamming Distance (intra HD) [5], [24]. The ideal intra-HD is 0 meaning the same PUF for the same challenge can always reproduce the same key[6].

Identifiability is defined by the ratio between the inter-PUF and intra-PUF HD, the higher the identifiability the more difficult it is to identify the specific PUF [25].

Randomness aims to quantify the unpredictability of a response, for example in a random response the distribution of '1's and '0's should be 50% [6], [24]. Randomness can be assessed using the National Institute of Standards and Technology (NIST) test suite, the 0/1 bias requirement from the NIST test suite is equivalent to the min-entropy discussed in Chapter 2.1 [26]. Another randomness metric is Shannon entropy.

2.4 Statistics/Statistical Tests

Statistics is about collecting, analyzing, presenting, and interpreting data which will allow the study of processes to be able to make quantitative and qualitative decisions.

However, statistics are not absolute, in statistical tests is often used a 5% probability that the test is wrong, and the significance level. The larger the sample size, the more likely the results are the right answer.

In this work, statistics and statistical tests were used to assess the behavior of transistors under mismatch variations. Thus, when discussing the results from statistical tests can't take absolute conclusions until a physical test is performed.

2.4.1 NIST Test

The National Institute of Standards and Technology (NIST) described and developed a set of statistical tests to standardize the evaluation of the security and randomness of binary sequences [26]. The current NIST Test Suit comprises fifteen different tests. The tests are:

- The Frequency (Monobit) Test
- Frequency Teste within a Block
- The runs Test
- Tests for the Longest-Run-of-Ones in a Block
- The Binary Matrix Rank Test
- The Discrete Fourier Transform (Spectral) Test
- The Non-overlapping Template Matching Test
- Maurer's "Universal Statistical" Test
- The Linear Complexity Test
- The Serial Test
- The Approximate Entropy Test
- The Cumulative Sums (Cusums) Test
- The Random Excursions Test
- The Random Excursions Variant Test

The NIST Test has a few concepts that will quantify the results and so, help to understand the meaning of the test's result. These concepts are the *complementary error function,* the *incomplete gamma function,* the *standard normal and chi-square distribution* and the p_{value}.

These statistical tests are used to assess the possible randomness of a set of bit strings. If the bit string passes all the tests, the string has a high chance of being random thus, the entropy source has high entropy.

2.4.2 Chi-square Goodness of Fit Test

The chi-square Goodness of Fit Test as explained in [27] is used to test if a specific sample of data has a specific distribution. This test used binned data for example data from a histogram. For better results, the test should use the same number of bins as the histogram used to visualize data.

This test is defined by the null hypothesis H_0, that the data follow a specific distribution. The test statistic is defined as

$$\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i}$$

Where O_i is the number of observations for bin i and E_i is the expected frequency for bin i for a significance level, α.

Test statistics follow the chi-square distribution with n degrees of freedom. Thus, the hypothesis that the data follow a specific distribution is rejected if

$$\chi^2 > Chi - squared\ value$$

The chi-square value is dependent on the degrees of freedom and the significance level α. The chi-square values are in Appendix A. The p-value obtained is an indication of the probability to obtain results as extreme as the result observed under the premise that the null hypothesis is correct. The higher the p-value, the less likely the null hypothesis is rejected.

2.4.3 Monte Carlo Method

As explained by, Kroese et. al [28], Monte Carlo is the simulation of random processes for several iterations. This method has various uses for example sampling, estimation, and optimization.

The sampling's main objective is to gather information about some random object. For example, mimic the behavior of a real-life system or circuit which is composed of such an object.

Estimation aims to estimate certain numerical quantities related to a certain model. For example, estimating the expected throughput in a production line.

The optimization objective, as the name suggests is to optimize complicated functions. For example, introduce randomness to deterministic functions and this way can be more efficiently solved.

Monte Carlo is universally used due to its simplicity and efficiency which lead to ease of scalability. When dealing with statistics, the more data evaluated, the closer to reality the results are, thus, the above characteristics are what makes the Monte Carlo method so popular.

The data obtained from the Monte Carlo method can be graphically expressed using histograms. As noted in [29], histograms split the range of data into equal-sized bins and the number of points from the data obtained from Monte Carlo that fall into the range of a certain bin is counted. So, on the vertical axis it is the count for each bin and on the horizontal axis, the variable being studied. A set of data displayed in the histogram form looks like Figure 2.18.

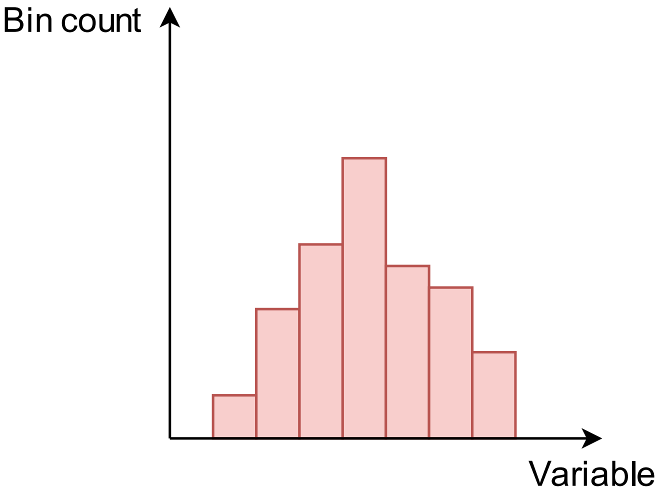


Figure 2.18 Histogram example.

To understand better the data that is being displayed by the histogram, probability density functions can be used. As explained by Uchechu. H. [30]. For such functions, there are two main types, parametric and non-parametric.

Parametric probability density functions assume that the data come from an already established probability distribution and based on the data, it creates an assumption about the parameters that define such function. An example of this is the normal distribution which assumes the data follows the normality.

Non-parametric probability density functions do not follow assumptions. When the distribution of data is unknown, the non-parametric functions allow for a more correct result about the population. An example is the kernel density distribution.

The kernel density distribution allows having a density distribution that represents the data obtained from the histogram without any assumption thus, better representing the distribution of data for an unknown distribution. By plotting the kernel density distribution against the normal distribution, it is possible to analyze how close the data of the histogram is to the normal distribution from the same data.

The normal distribution is especially important when assessing if a data set is or is not naturally random.

3

STATE OF THE ART

When designing PUFs, there are some characteristics that are desirable, like, minimize noise, power consumption, area of the chip, obtain high entropy e robustness to external variations. Thus, when analyzing and designing PUFs, it is important to have a closer look into those characteristics.

3.1 Arbiter PUF

The arbiter PUF is a kind of delay based PUF that is comprised of a switching component and an arbiter block as shown in Figure 3.1. Input and a random challenge are fed into the switching component to produce the necessary delays. The arbiter block will measure the delays and select which channel, the two paths are designed to have the same delay producing the response. The arbiter block can be for example a D flip-flop. [24]

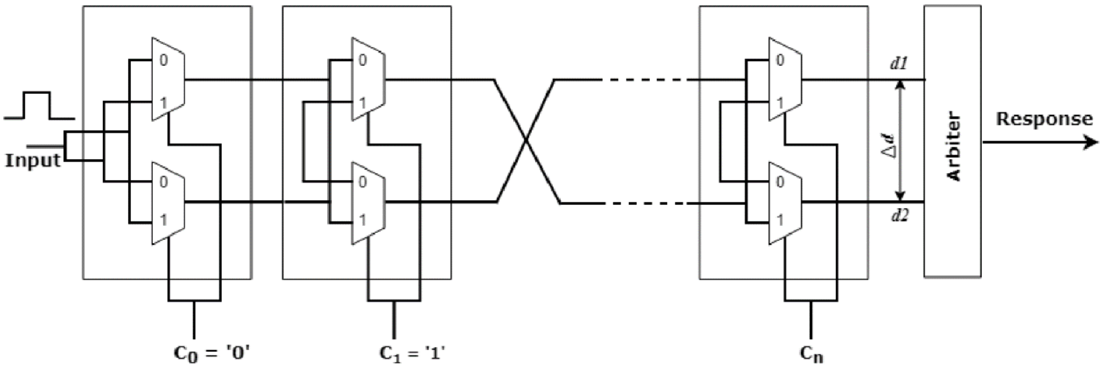


Figure 3.1 Arbitre PUF. Extracted from [24].

The switching component consists of N multiplexers that are fed with an N-bit challenge with each bit on each multiplexer to select the paths. In the end, the Arbiter stage will determine each signal arrived first and based on that, for example, output "1" if signal1 arrived first and output "0" if signal 2 arrived first. Both paths are designed to have the same delay however due to differences in intrinsic physical characteristics of each path/component, there will be a delay that will characterize the entropy of the PUF.

In [31], was proposed an Arbiter PUF (APUF) of 64 single-bit PUF cells using CMOS technology, with the 64 single-bit PUF cells, obtained a 64-bit response. The entropy of the response is obtained by exploiting the imperfections from the fabrication process of the CMOS technology not only on the switching element but also on the selecting module. The architecture of a single-bit PUF cell is illustrated in Figure 3.2.

The switching element is formed by four transmission gates using one NMOS and one PMOS in parallel, this configuration enables the input signal to travel in two possible paths depending on the random challenge determined by the selecting module. Each path has a different-sized transistor to have paths with various delays.

The selecting module is composed of three inverters in series. The first transistor is set so the voltage transfer characteristics (VTC) is 1/2V_{DD} and the other two transistors are used to completely emphasize signal value. A challenge of 1/2V_{DD} is fed into the first inverter, due to the process variations created upon fabrication of the CMOS transistor, the VTC will be slightly different thus, the input challenge will stay below or under the VTC. If the input challenge is lower than the real VTC, the NMOS will be in saturation mode and the PMOS in linear mode thus, the output will tend to V_{DD}. If the input challenge is higher than the real VTC, the NMOS will be in linear mode and the PMOS in saturation mode thus, the output will tend to zero. Variations occur not only on the switching element but also on the selecting module.

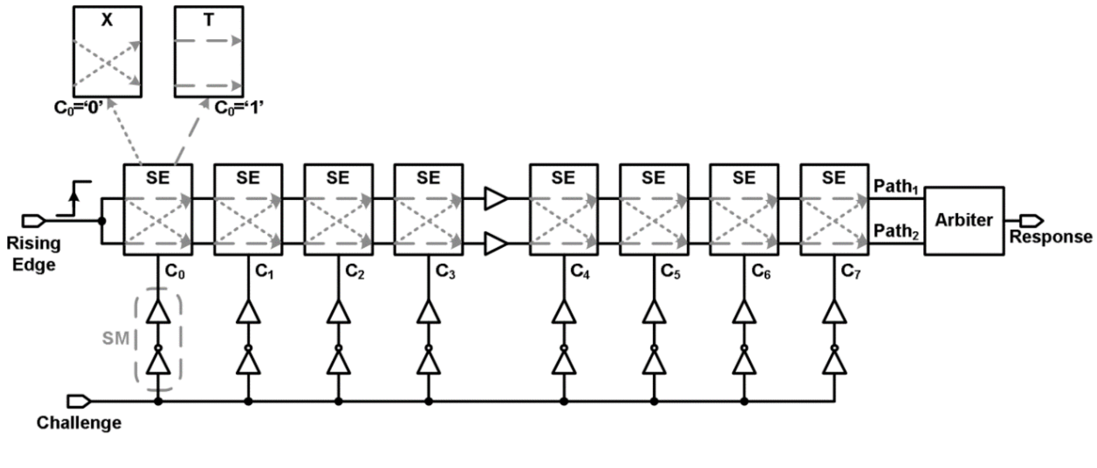


Figure 3.2 Proposed single-bit PUF cell. Extracted from [31].

The APUF with selecting modules showed it can be a good PUF. The results were good in uniqueness and energy consumption tests. However, to get enough entropy, this kind of

PUF need a significant number of stages (SE blocks in Figure 3.2) to generate enough delays between the two paths, thus its main drawback is the area used.

The performance results are shown in Table 3.1.

Table 3.1 APUF with selecting modules performance [31].

Technology	45nm MOSFET
Area (μm^2)	2168
V_{DD} range (V)	0.9~1.1
Temperature Range (ºC)	0~100
Energy per bit (pJ/b)	0.103593
BER	1.34%
Uniqueness	49.99%

3.2 Memory Based PUF

The Memory based PUF uses a memory cell to generate the entropy. An SRAM is a type of NVM which means it only stores data when it is being powered. A commonly used configuration is a 6 transistor SRAM cell, which is formed by two cross-coupled identical inverters, this configuration stores one bit of information, which can be seen in Figure 3.3. The entropy is extracted from process variations present in the inverters. These mismatches are then amplified due to the positive feedback of the cross-coupled inverters. [32] However SRAM cells tend to produce more the number logic "1" than the logic number "0", which greatly reduces the randomness of the challenge produced. Also, the SRAM cell only depends on six transistors, if the mismatches between the transistors is not substantial compared to noise, the reliability of the PUF can be degraded [32]. Compared to other types of PUFs, SRAM-PUF has advantages in stability and reliability [33], [34].

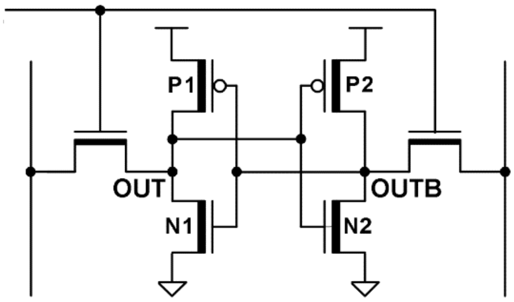


Figure 3.3 SRAM cell Schematic. Extracted from [32].

In [32], is proposed a methodology to solve the above issues of the SRAM-PUF. This is achieved by taking into consideration aging effects, Negative-Bias-Temperature-Instability (NBTI) which mostly affects the threshold voltage. The polarity and aging change the result created by the mismatches in the cross-coupled transistors, since, in fabrication, the data "1" is 70% more likely to appear, aging effects are injected into the SRAM To balance the final "1" and "0"'s output thus increasing uniformity. To improve reliability, the objective is, after reaching the desired uniformity, to introduce more NBTI aging to increase the threshold mismatch on one inverter on each SRAM cell, this increases the mismatch and thus, the reliability. The final robust SRAM-PUF methodology is represented in Figure 3.4. After the introduction of aging effects, results show great improvements in uniformity and reliability, without NBTI injection, the probability of "1"s is 0.7255, after the introduction of the aging effect until ΔV_{th}=46, the probability of "1" reaches 0.4957 on the ideal level of 0.50.

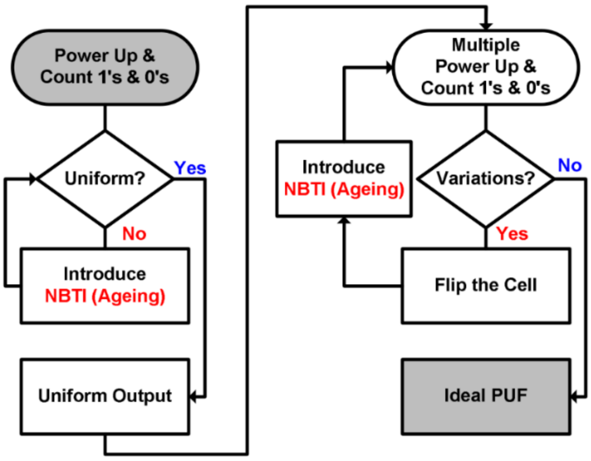


Figure 3.4 Methodology for robust SRAM-PUF. Extracted from [32].

3.3 Oscillator Based PUF

3.3.1 Relaxation Oscillator

A relaxation oscillator is a non-linear oscillator that generates a periodic non-sinusoidal output signal such as a triangle or square wave, a classical CMOS relaxation oscillator is shown in Figure 3.5. The oscillator generates a signal utilizing the charge and discharge cycles of capacitors or inductors with the frequency of the oscillation dependent on the time constant τ of the capacitive or inductive circuit. These kinds of oscillators are easily implemented in CMOS and cost less current and power [24].

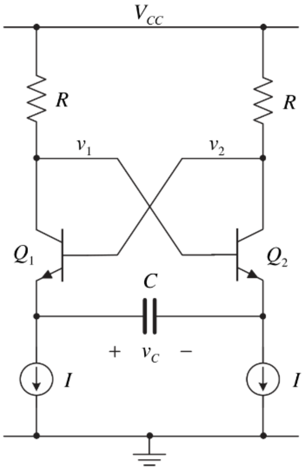


Figure 3.5 Classical CMOS Relaxation oscillator schematic. Extracted from [24].

In [24], it is used a relaxation oscillator with the schematic represented by Figure 3.6. This oscillator will have a frequency-dependent on the time constant τ of the RC circuit which is f=1/ τ. This PUF is very similar to the conventional ring oscillator PUF as shown in Figure 3.7, for the entropy source, the relaxation oscillator PUF takes use of the process variations that occurred in the fabrication, two multiplexers are used to select which relaxation oscillator, two counters to count the number of oscillations and finally the comparator to output the response based on the count difference. This PUF was implemented in UMC65nm technology with the use of 32 relaxation oscillators, so a 5-bit challenge was used to select each oscillator.

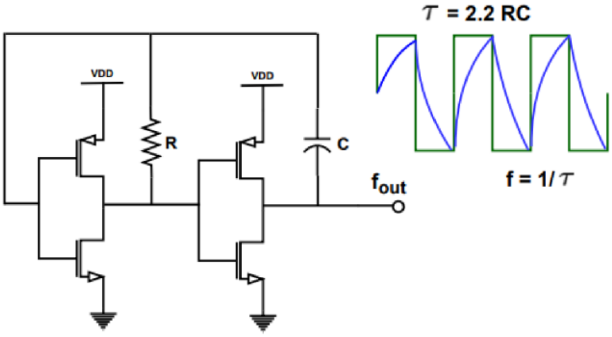


Figure 3.6 Basic Relaxation Oscillator. Extracted from [24].

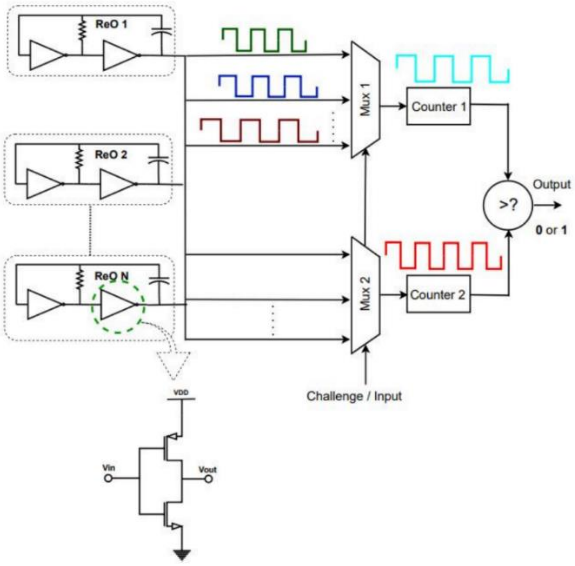


Figure 3.7 Relaxation Oscillator PUF. Extracted from [24].

The evaluation of quality metrics indicates that this PUF is not very resilient neither to voltage nor to temperature variations. [24]. The performance results from a relaxation oscillator based on the PUF in Figure 3.7 are shown in Table 3.2.

Table 3.2-Relaxation Oscillator results [24].

Technology	65nm
Normalized PUF cell Area (μm^2/bits)	-
Voltage range (V)	-
Temperature Range (ºC)	-
Energy per bit (pJ/b)	-
Power (µW)	118.42 (one instance)
BER	-
Uniqueness	49.22%
Diffuseness	49.53%
Uniformity	45.31%
Reliability	97.41% (temperature change) 97.97% (voltage change)

3.3.2 Coupled Oscillator

The coupled oscillator has already been used in developing TRNGs as in [35], [36]. In both works, the coupling is achieved using a D flip-flop where a slower oscillator is used to sample a faster oscillator as shown in Figure 3.8. To produce a random bitstream, it is exploited the jitter present in the sampler oscillator, if the standard deviation of the slow oscillator period is greater than the fast oscillator period, two successive sampling times are assumed independent, thus the bitstream generated is random.

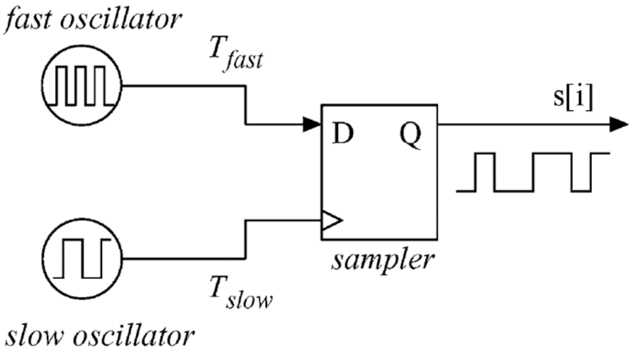


Figure 3.8 Simple Coupled Oscillator. Extracted from [36].

In [37], a coupled chaos oscillator was implemented as shown by the simplified circuit in Figure 3.9. Instead of using a D flip-flop, the coupling is accomplished through a bi-directional non-linear current conduction which can be implemented using two cross-coupled diodes, this results in a nonlinear current which is used to obtain a chaos behavior. Chaos provides a non-periodic deterministic behavior thus it can be used to generate unique ID entropy. Two identical ROs are used to provide a periodic non-chaotic behavior, when C_2 and R_2 are introduced, the voltage swing of the second oscillator decreases, and the difference between the voltage swings across both oscillators becomes larger than the threshold voltage of the diodes, this activates the nonlinear current from the cross-coupled diodes inducing a chaos behavior in the oscillation.

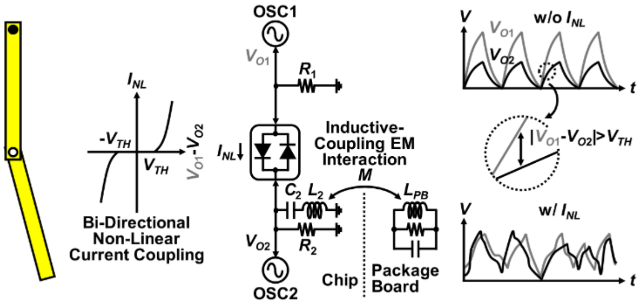


Figure 3.9 Simplified schematic of coupled chaos PUF. Extracted from [37].

This implementation also has chip-package-board interaction to detect counterfeit or manipulation of the chip during fabrication. This is possible using the inductor L_2 which is through electromagnetic coupling with the parasitic inductance of the package board. If the package board is altered, the code generation alters thus, by checking the chip ID it is possible to detect manipulation.

The performance results are shown in Table 3.3.

Table 3.3 Coupled Chaos Oscillator PUF performance [37].

Technology	180 nm
Area (F^2)	40,204
Data Rate (Mb/s)	~3
Energy (pJ/bit)	~2.640
BER (%)	2.5

As of today, there is not much bibliography on PUF based on coupled oscillators.

3.3.3 Ring Oscillator

A ring oscillator (RO) is formed by an odd number of inverters in a feedback loop as shown in Figure 3.10. The last inverter output is fed into the input of the first inverter and this feedback loop is what enables the oscillatory behavior. The oscillation will occur between zero and V_{DD} and will create a square wave.

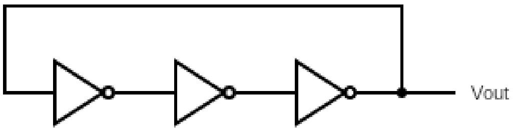


Figure 3.10 Conventional Ring Oscillator.

The conventional RO PUF, shown in Figure 3.11 is formed by N identical ROs, two multiplexers, two counters, and one comparator. Due to physical variations from the fabrication process of CMOS, the ROs will oscillate in different frequencies. The ROs are connected to the pair of multiplexers that choose the RO based on the challenge input. The counters, count the number of impulses from the RO selected from the multiplexer [24]. The comparator will compare both counters and if counter 1 has more impulses than counter 2 it outputs "1" and "0" the other way around.

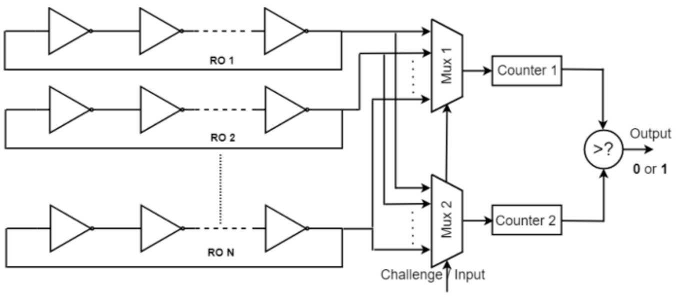


Figure 3.11 Conventional Ring Oscillator PUF [38].

On the conventional RO PUF, if the frequent differences caused by the fabrication process variations are too small, environmental variations, and aging, the RO PUF may output the wrong bit causing instability due to noise. By using a configurable RO to select the RO pairs with the largest frequency difference it is possible to increase the stability and reliability of an RO-based PUF. A basic configurable RO is shown in Figure 3.12.

In [39], a configurable RO was implemented using FPGA. It implemented a pair of configurable RO each with three multiplexer stages which means 3-bit challenges. The frequency of every possible input challenge is tested against both configurable RO and the challenge that provoked the largest frequency difference is stored. Because the frequency difference is maximized, the reliability is increased, making the PUF more robust to noise and environmental effects and thus more stable. Using 128 ROs, It was obtained an uniqueness of 45.51%, zero unstable bits due to temperature variations, and about 5 unstable bits due to voltage variations.

In [38], additional multiplexers between each pair of inverters were used, this allows the swapping of inverters pairs as shown in Figure 3.13. This swapping is configurable to maximize the frequency difference in the RO pair thus improving PUF stability. Compared to other methods like [39], [40], this method used fewer gates and transistors thus having better area and using less power. The performance results are shown in Table 3.4.

Table 3.4 Wide-Range Variation-Resilient Physically Unclonable Function [38].

Technology	28nm MOSFET
Normalized PUF cell Area (μm^2/bits)	26
Voltage range (V)	0.4~1.3
Temperature Range (ºC)	-40~125
Energy per bit (pJ/b)	2.15
BER	0.55%
Uniqueness	49.94%

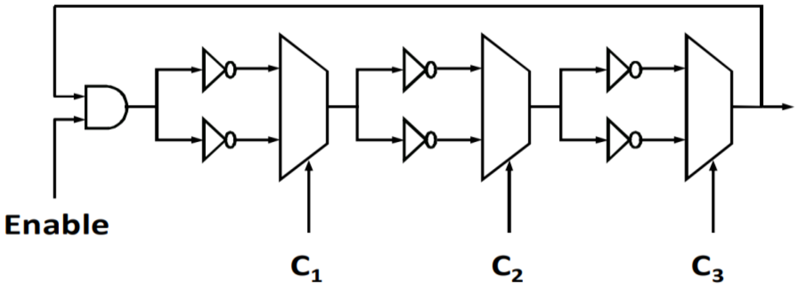


Figure 3.12 Configurable Ring Oscillator. Extracted from [39].

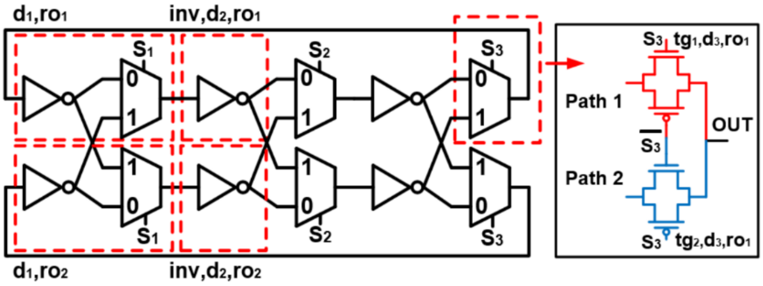


Figure 3.13 Configurable RO proposed in [38]. Extracted from [38].

In [41], a multiple RO configuration was implemented to increase entropy and throughput. The additional ROs allows not only for a better bit rate but also improve power efficiency. The performance of this multiple RO is shown in Table 3.5.

Table 3.5 PL-MRO-PUF performance [41].

Technology	FPGA
Normalized PUF cell Area (μm^2/bits)	-
Voltage range (V)	-
Temperature Range (ºC)	-
Energy per bit (pJ/b)	-
Power Efficiency (µW/bit)	23.44
BER	0.55%
Uniqueness	51.7%
Randomness	98.8%
Steadiness	94.5%
Correctness	91.3%
Diffuseness	98.1%

4

Circuit Implementation

In Chapter 2 was given a theorical look at the components and themes that make part of this dissertations. and in Chapter 3 relevant state of the art implementations was presented. Now, in this Chapter the steps involved in the design of the coupled relaxation oscillator static entropy source are presented. All the circuits were implemented using CADENCE tools in advanced 130 nm technology.

4.1 Voltage Controlled Resistor

In the final circuit, a voltage-controlled resistor (VCR) was used to replace a fixed resistor. Since the characteristic IV curve of the transistor, the transistor is as shown in Figure 4.1.

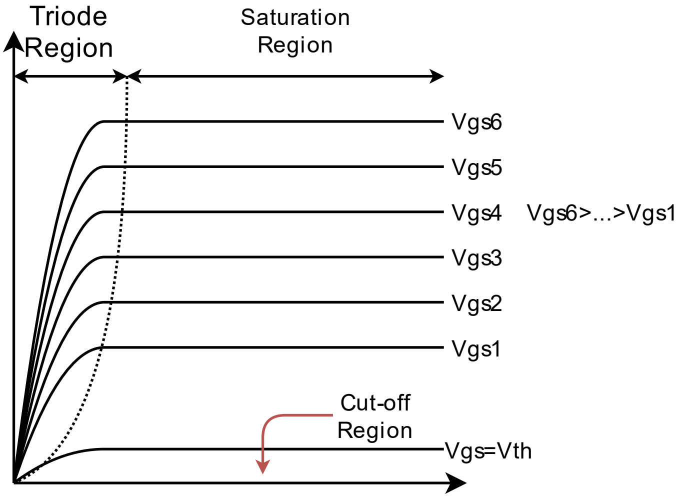


Figure 4.1 NMOS IV characteristic curve.

The transistor operating in the triode region functions like a resistance. Thus, by adding a DC voltage source to the gate of the transistor, it is possible to control the v_{gs}. Which is the triode region will affect the r_{ds} linearly. The circuit used is as in Figure 4.2.

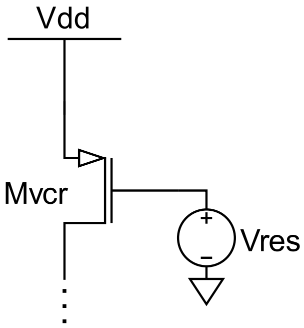


Figure 4.2 Voltage Controlled Resistor Circuit.

Even though a resistor is more affected by mismatch variations. The decision to switch a regular resistor with a VCR was due to the added flexibility of allowing to fine-tune the oscillator to a particular frequency.

4.2 Entropy Source

In this chapter, the circuit used as a static entropy source is presented. The entropy source was developed using coupled relaxation oscillators. The coupled relaxation oscillator can be divided into two parts, the oscillator, and the coupling circuit. Both parts of the circuit follow the high-level model discussed in Section 2.2.3.

The coupled relaxation oscillator was chosen as the subject of study because compared to state-of-the-art implementations, it may have better performance in the main characteristics of what defines a good PUF which are

- Power consumption.
- Area of the circuit.
- Low noise.
- Robust to temperature, voltage, and aging variations.
- Entropy

The main characteristics of the entropy source for a PUF are low noise and robust to temperature, voltage variations. The low noise is crucial to highlight the process and mismatch variations and the resistance to external variations is important to maintain the response in different environments. Because PUFs are mostly designed for IoT which tend to have power limitations, it is important to maintain low power consumption. The low area of the circuit mostly allows the device to be cheaper to produce.

4.2.1 Relaxation Oscillator

The relaxation oscillator was chosen due to its low noise and low power consumption and simplicity. This way, the oscillator will not add any undesirable noise that may change the static entropy output.

The high-level model of the relaxation oscillator is represented in Figure 2.11. The integrator can be modeled as a simple capacitor, where the input is the integration constant which is the bias current that flows through the capacitor (i_c) and in its output, the capacitor voltage (v_c). The voltage v_c is then the input of the Schmitt-trigger which will output i_c back into the capacitor. The integrator implementation is shown in Figure 4.3 a) and the corresponding characteristic curve in Figure 4.3 b).

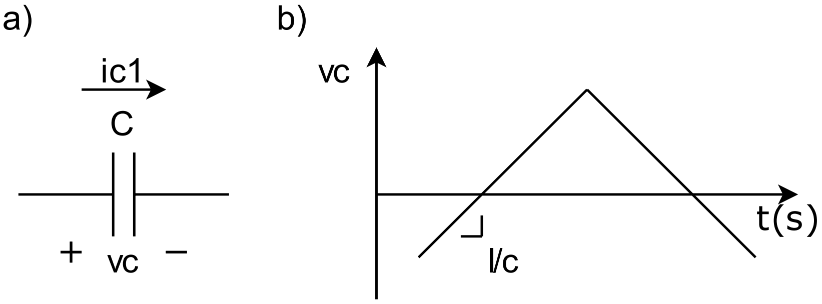


Figure 4.3 Integrator. a) Implementation. b) Characteristic curve.

The characteristic curve of the integrator is derived from the current in a capacitor which is defined by

$$i_c = C\frac{dv_c}{dt}$$

However, in this oscillator, the voltage in the capacitor is the output, from the previous equation,

$$v_c = \frac{1}{C}\int_0^t i_c\ dt$$

For a constant current, the voltage across the capacitor will increase linearly over time with a slope of $\frac{I}{C}$ as shown in Figure 4.3 b) before the curve changes to a negative slope. However, the integrator alone cannot change the integrator constant and thus create a periodic signal, for that, a Schmitt trigger is used.

As discussed in Section 2.2.3, the Schmitt-Trigger has three major functions, comparison, switching the integrator sign, and memorization. The Schmitt-Trigger can be implemented as a differential pair as shown in Figure 4.4 a). Even though the output of the Schmitt trigger is the bias current, the output of the oscillator will be the voltage $v_{out} = v_1 - v_2$, this way it is possible to take advantage of the differential pair noise reduction characteristics.

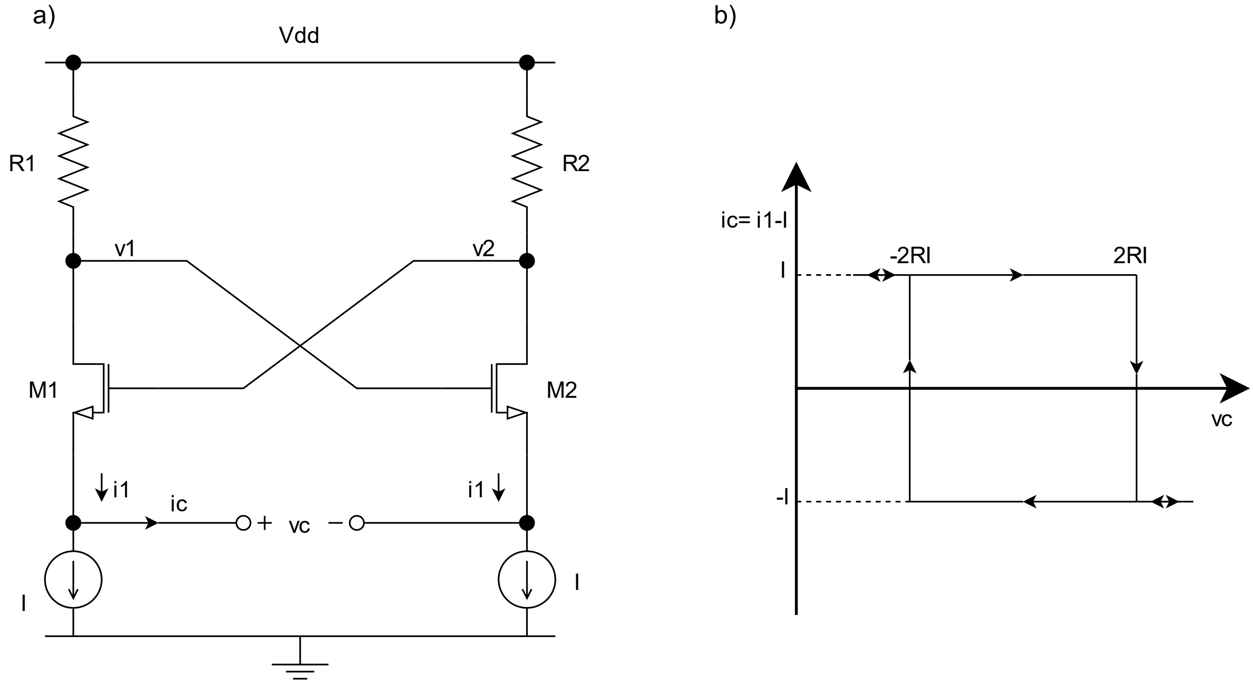


Figure 4.4 Schmitt trigger. Extracted from [17]. a) Circuit. b) Characteristic curve.

The comparison function is performed using the voltage drop across the resistor as the reference signal. The behavior of the Schmitt trigger is like hysteresis as shown in Figure 4.4 b). The switching of the integrator sign is performed by the transistors which will be transitioning alternatively from on to off switching the direction in which the current flows through the capacitor, thus changing the integration constant I.

The relaxation oscillator with the integrator and the Schmitt trigger is represented in Figure 4.5. The operation of this oscillator can be explained as follows.

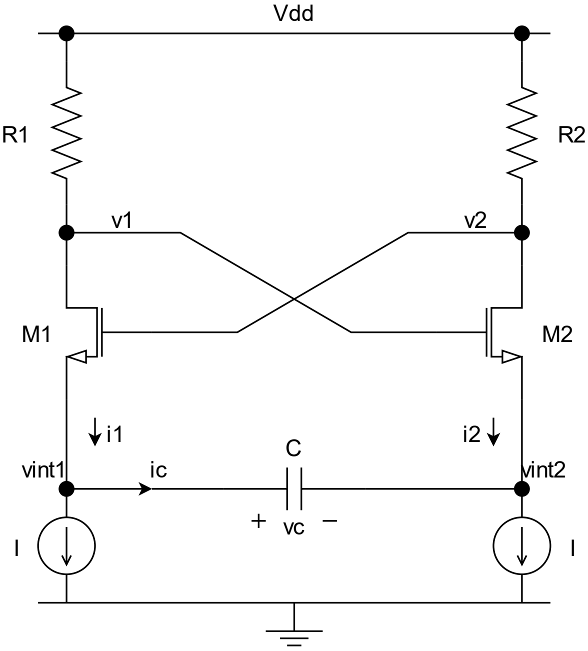


Figure 4.5 Relaxation Oscillator. Extracted from [17].

The relaxation oscillator operation was described by Eduardo Ortigueira et. al. [42]. Either M_1 and R_1 or M_2 and R_2 are conducted at any given time by the constant current I. The reference signal is given by 2RI. When only one transistor is on, the current from both current sources is passing through one of the resistors and one transistor. The reference signal is also the peak-to-peak amplitude of the symmetrical square wave generated. When M_1 is ON and M_2 is OFF, a constant current is flowing through R_1 and M_1 thus, v_{gs1} and v_{c1} are constant and $v_2 = V_{DD}$. The capacitor is charging thus v_c increases and $v_{int2} = v_{int1} - v_c$ decreases until v_{gs2} reaches the voltage threshold. At this point M_1 and M_2 are ON and $v_{int1} = V_{DD}$ and the capacitor discharges until v_{gs1} reaches the voltage threshold. Now M_1 is OFF and M_2 is ON and the oscillator changes state. Since the current is constant, v_c is linear with a slope equal to $\frac{I}{C}$ depending on the flow of current, the sign of the slope is either positive or negative. The differential voltage $v_1 - v_2$ and v_c are represented in Figure 4.6. The node voltages v_1, v_2, v_{c1} and v_{c2} are represented in Figure 4.7.

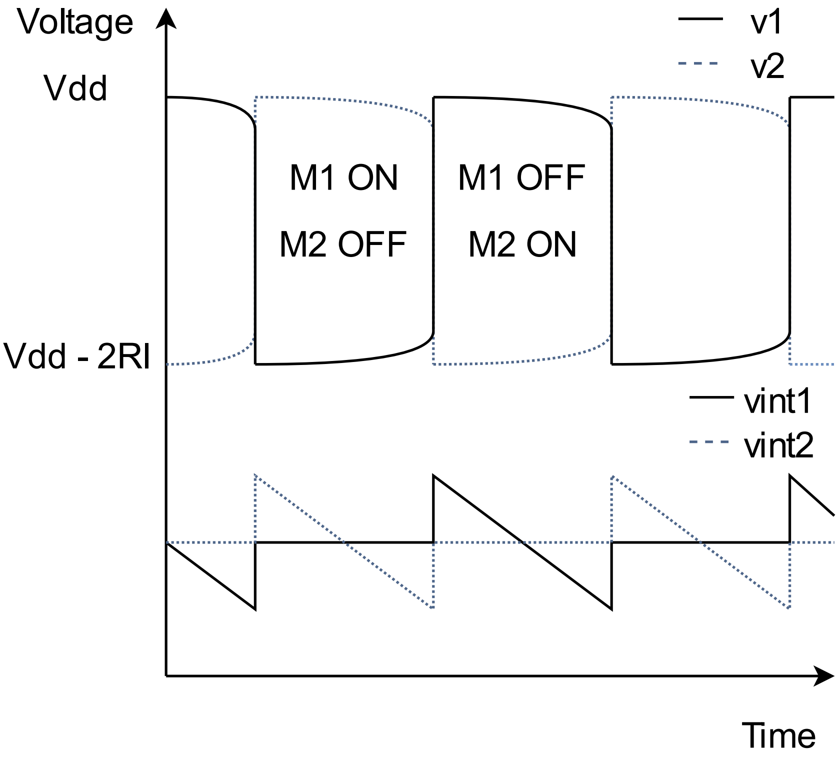


Figure 4.7 Node voltages, v_1, v_2, v_{int1} and v_{int2}. Extracted from [42].

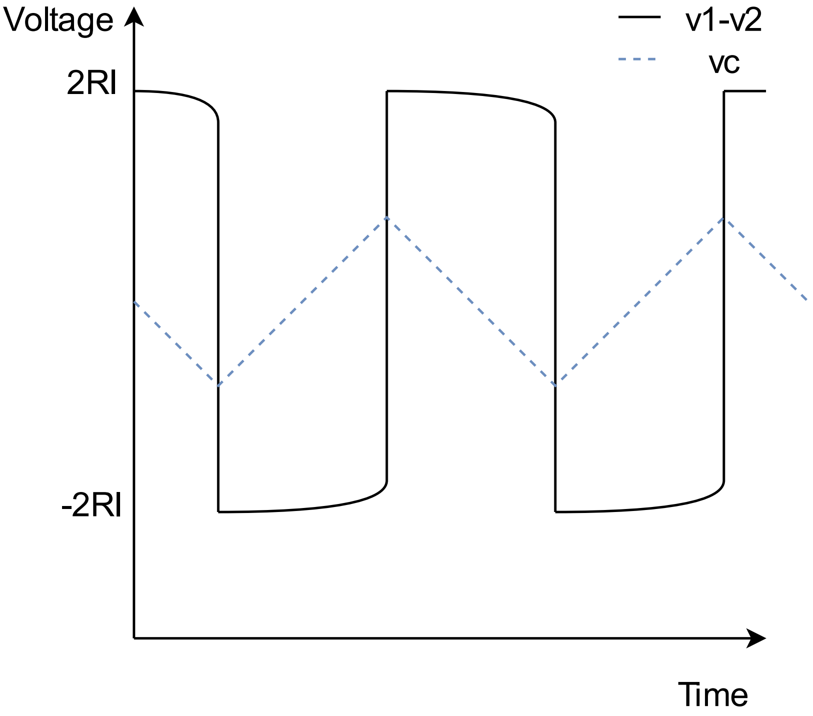


Figure 4.6 Differential voltages $v_1 - v_2$ and v_c. Extracted from [42].

In the final implementation of the circuit, the resistor R from Figure 4.5 was changed by a voltage-controlled resistor. This was achieved by substituting the resistors with a transistor and by adding a DC voltage source to the gate of a transistor.

Even though a resistor is more affected by process and mismatch variations. The decision to switch a regular resistor with a VCR was due to the added flexibility of allowing to fine-tune the oscillator to a particular frequency and because of the smaller area of the transistor compared to the resistor. Thus, the final implementation of the relaxation oscillator is shown in Figure 4.8.

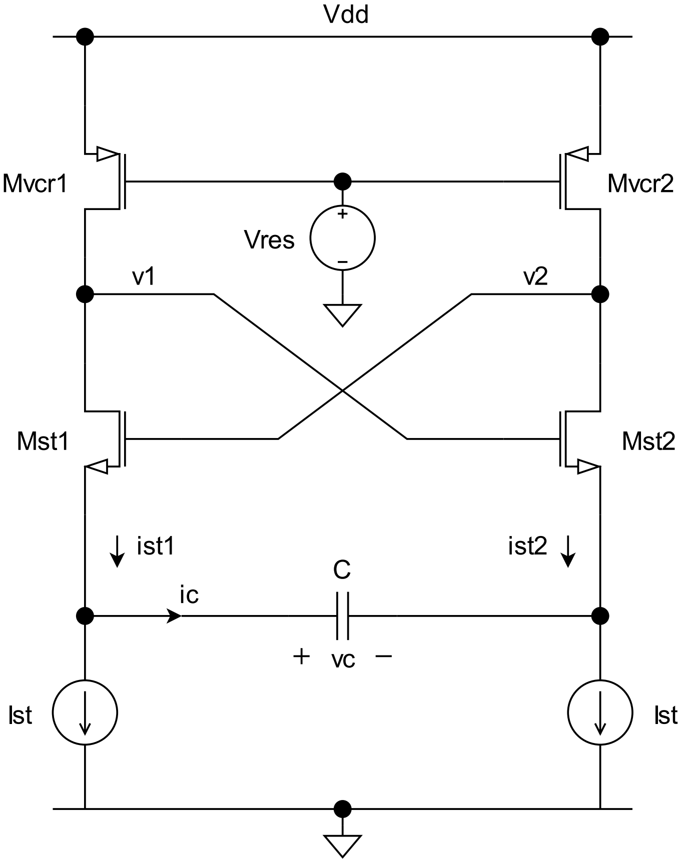


Figure 4.8 Relaxation Oscillator implemented design.

The sizing of all aspects of the circuit, the current, the supply voltage, and the dimension of all transistors are in Table 4.1.

Table 4.1 Relaxation Oscillator dimensioning.

V_{dc}	1.2 V
I_{st}	1 mA
V_{res}	100 mV
W_{st}	70 µm
L_{st}	120 nm
W_{vcr}	15 µm
L_{vcr}	120 nm
C	1.5 pF

Such values of W_{vcr}, $L_{vcr,}$ and V_{res}, translate into resistance of 169.39 Ω. Following the dimensioning from Table 4.1, the frequency of oscillation is

$$f_{relax} = 1.01229 \text{ GHz}$$

The differential voltages of the circuit implemented, $v_1 - v_2$ and v_c are represented in Figure 4.9.

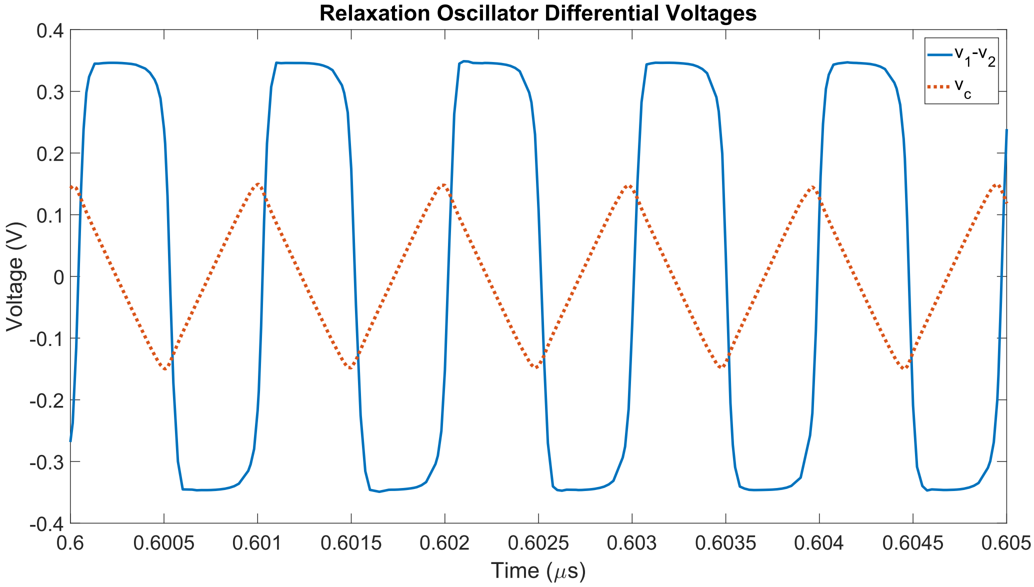


Figure 4.9 Relaxation Oscillator designed differential voltages.

As it can be observed, the upper and the lower limits of the output signal v_1-v_2 are close to the expected value of 2RI which is $2 \times 169.39 \times 1 \times 10^{-3} = 0.339 \text{ V}$.

4.2.2 Coupled Relaxation Oscillator

The oscillators from the cross-coupled relaxation oscillators will synchronize each other to the chosen frequency which will reduce the phase noise. Also, the extra oscillators will increase the transistors used which will highlight the process and mismatch variations compared to a single oscillator.

Following the high-level model from Figure 2.12, to design the coupled oscillator a soft limiter was added to the circuit. The design of the soft limiter is represented in Figure 4.10. This element is an amplifier thus for simplicity, a common source topology was chosen.

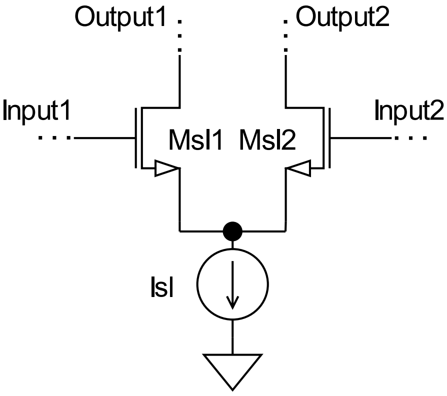


Figure 4.10 Soft limiter implementation.

The input of the soft limiter is then connected to the output of the integrator and the output of the soft limiter is connected to the input of the Schmitt trigger. To cross couple two oscillators, the input and the output of the soft limiter need to be from different oscillators. In Figure 4.11, the coupled relaxation oscillator design is represented.

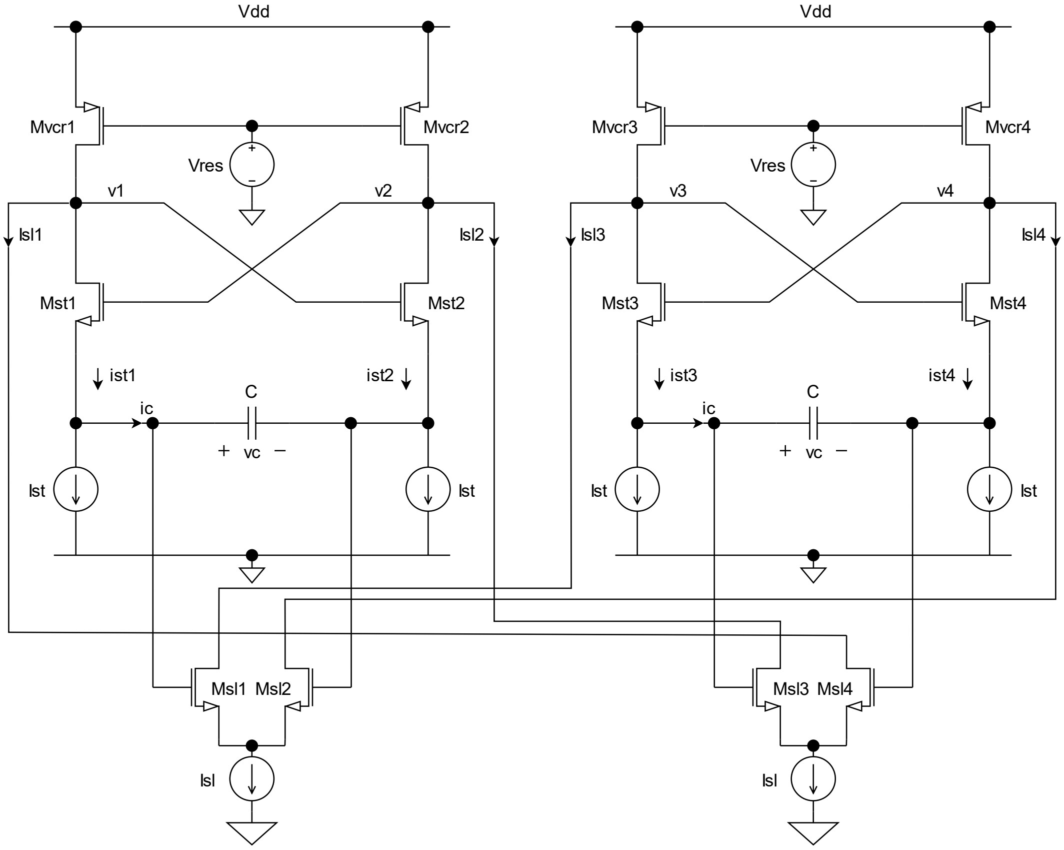


Figure 4.11 Coupled Relaxation Oscillator architecture using two oscillators [17].

To study this oscillator, we need to assume that the oscillator is in steady-state and that the voltages v_{c1} and v_{c2} are in quadrature. The waveform in quadrature is dependent on which soft limiter is cross connected to the input of the Schmitt trigger. In **Erro! A origem da referência**

não foi encontrada. a), the quadrature signals are represented and v_{c2} and $v_1 - v_2$ are 90° advanced compared to v_{c1} and $v_3 - v_4$ respectively. Considering several instants t like in Figure 4.12 b), the oscillator can be analyzed.

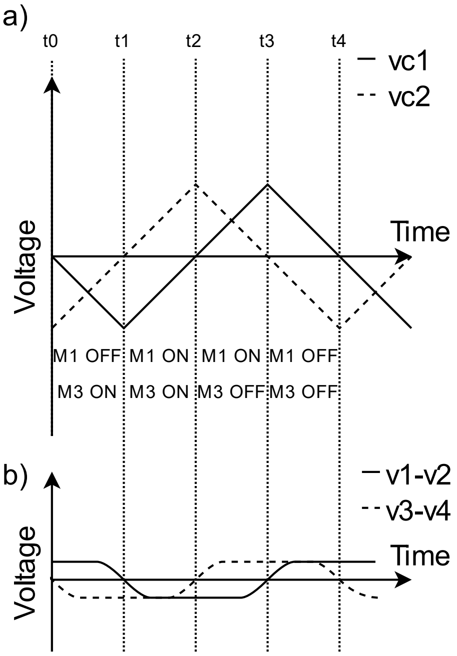


Figure 4.12 Coupled Relaxation Oscillator differential signals a) v_{c1} and v_{c2} b) $v_1 - v_2$ and $v_3 - v_4$.

Starting in t_0, M₁ OFF and M₃ ON. Until t_1, $v_{c2}(t)$goes through zero, i_{SL1} decreases and i_{sl2} increases. v_1 and v_2 are given by

$$\begin{cases} v_1 = V_{DD} - i_{SL1}R \\ v_2 = V_{DD} - 2IR - i_{SL2}R \end{cases}$$

When v_{gs1} reaches the voltage threshold the transistor changes state. Immediately before that change,

$$v_{c1} = v_2 - v_1 + v_{th1} - v_{th2}$$

Because of $v_{th1} \approx v_{th2}$,

$$v_{c1} \approx v_2 - v_1 = -2IR - R(i_{SL2} - i_{SL1})$$

However, because $i_{SL2} - i_{SL1}$, is a small value, in the transition state (at t_1), $v_{c1} \approx -2IR$.

The next transition happens in t_3 where v_{c2} is zero. Between t_1 and t_3, M₁ is ON and M₃ is OFF, i_{SL1} increases and i_{SL2} decreases and we have

$$\begin{cases} v_1 = V_{DD} - 2IR - i_{SL1}R \\ v_2 = V_{DD} - i_{SL2}R \end{cases}$$

The transistor will change state when

$$v_{c1} = v_2 - v_1 + v_{th2} - v_{th1}$$

Because $v_{th1} \approx v_{th2}$,

$$v_{c1} \approx v_2 - v_1 = 2IR - R(i_{SL2} - i_{SL1})$$

Because $i_{SL2} - i_{SL1}$, is a small value, the maximum value of $v_{c1} \approx 2IR$.

It is important to note that the amplitude of the voltages across the capacitors does not change with the coupling. The voltages v_{c1} and v_{c2} have similar behavior however with a 90º phase difference.

For this work, two extra oscillators were added resulting in four coupled relaxation oscillators. This will increase the transistors count per oscillator thus increasing variability from mismatch variations. The extra coupled oscillators can also diminish even more the phase noise. Thus, adding two oscillators will allow having a circuit less susceptible to temperature and supply voltage variations while increasing mismatch variations. Thus, the high-level model is shown in Figure 4.13.

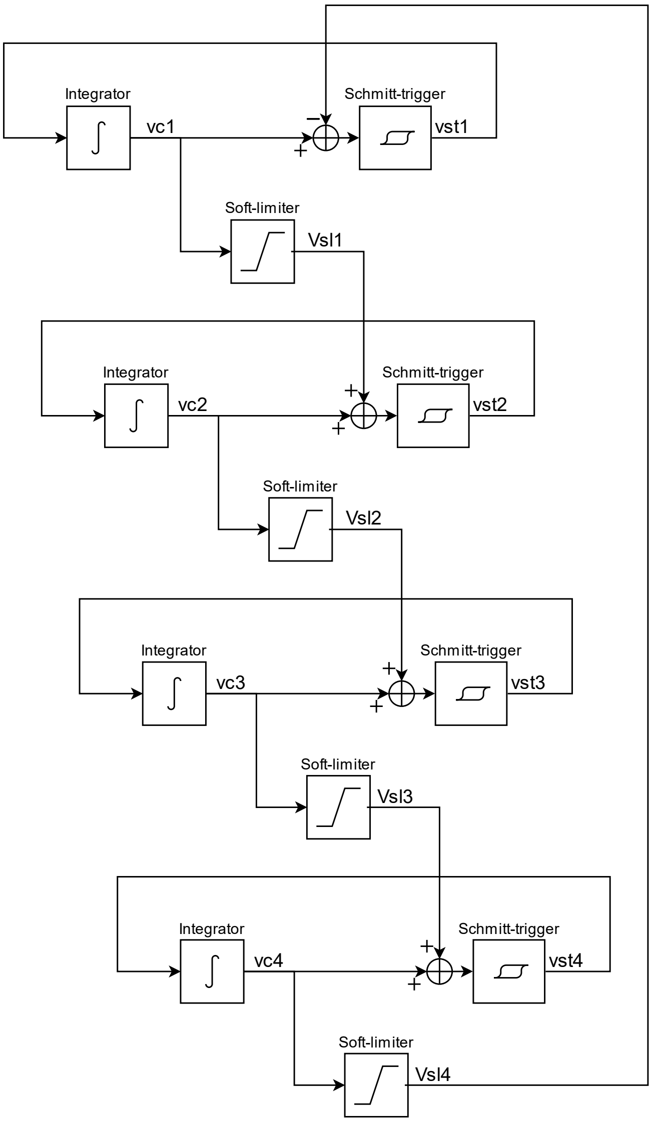


Figure 4.13 High level model of four coupled relaxation oscillators.

Based on the previous explanation and the high-level model of the coupled relaxation oscillator using four oscillators, the circuit implemented is represented in Figure 4.14.

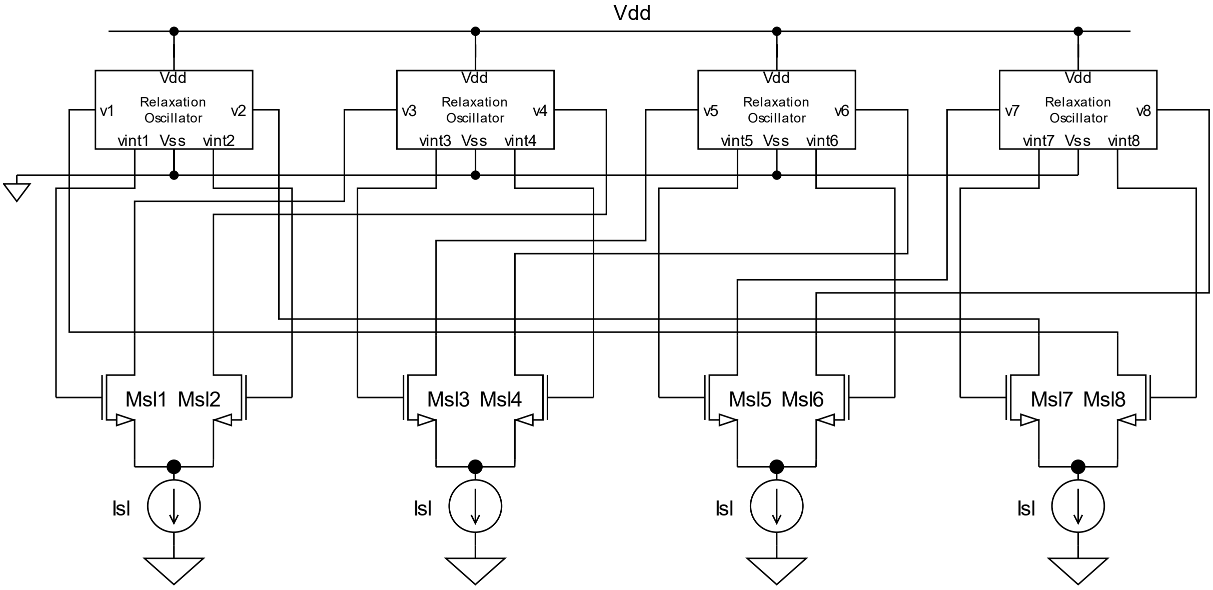


Figure 4.14 Coupled Relaxation Oscillator implemented design.

In Table 4.2, there are the values of current, supply voltage, and transistor dimensioning. Every individual relaxation oscillator was dimensioned to be the same. And with the same dimensioning as the relaxation oscillator in subsection 4.2.1

Table 4.2 Coupled Relaxation Oscillator dimensioning.

V_{dc}	1.2 V
V_{res}	30 mV
W_{st}	70 µm
L_{st}	120 nm
I_{st}	1 mA
W_{sl}	5 µm
L_{sl}	120 nm
I_{sl}	200 µm
C	1.5 pF
W_{vcr}	15 µm
L_{vcr}	120 nm

For the above dimensioning, the frequency obtained was

$$f_{coupled} = 1.04512 \text{ GHz}$$

For the frequency to be as close to 1 GHz the VCR was adjusted. Since the frequency was higher, the adjustment was performed by lowering V_{res} which will lower the resistance thus, the frequency. Such values of W_{vcr}, $L_{vcr,}$ and V_{res}, translate into resistance of 157.08 Ω

The differential voltages, $v_1 - v_2$ and v_{c1} from the resulting circuit, were plotted and are shown in Figure 4.15.

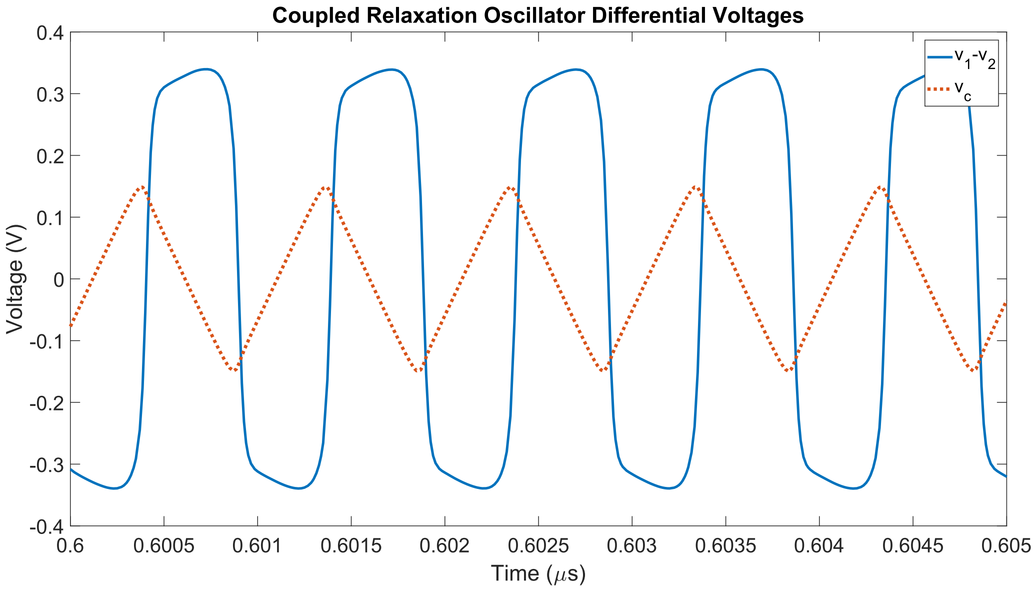


Figure 4.15 Coupled Relaxation Oscillator designed differential voltages.

Since the coupling circuit does not change the amplitude of the output voltage, it is expected that $v_1 - v_2$ keep the same ratio of 2RI. The resistance R was changed to 157.08 Ω so, the new amplitude is $2 \times 157.08 \times 1 \times 10^{-3} = 0.314\ \mathrm{V}$.

4.2.3 Ring Oscillator

The ring oscillator will be used as a way of referencing the architecture design for this project. The inverter was designed using L_{min} and channel length for both NMOS and PMOS transistors. The width of the NMOS was chosen to be $W_N = 2 \times L_{min}$ and the width of the PMON following the commonly known rule of $W_P = 3 * W_N$. The dimensioning obtained is shown in Table 4.3.

Table 4.3 Ring oscillator dimensioning.

W_N	240 nm
W_P	720 nm
$L_{N/P}$	120 nm

To achieve the desired frequency, inverters were added until the frequency of the ring oscillator reached $950\ \mathrm{MHz} \leq f_{ring} \leq 1.5\ \mathrm{GHz}$ as close to $1\ \mathrm{GHz}$. To achieve the previous conditions, 31 inverters were used which obtained a frequency of

$$f_{ring} = 1.035\ \mathrm{GHz}$$

The ring oscillator schematic is shown in Figure 4.16 and the signal obtains in Figure 4.17.

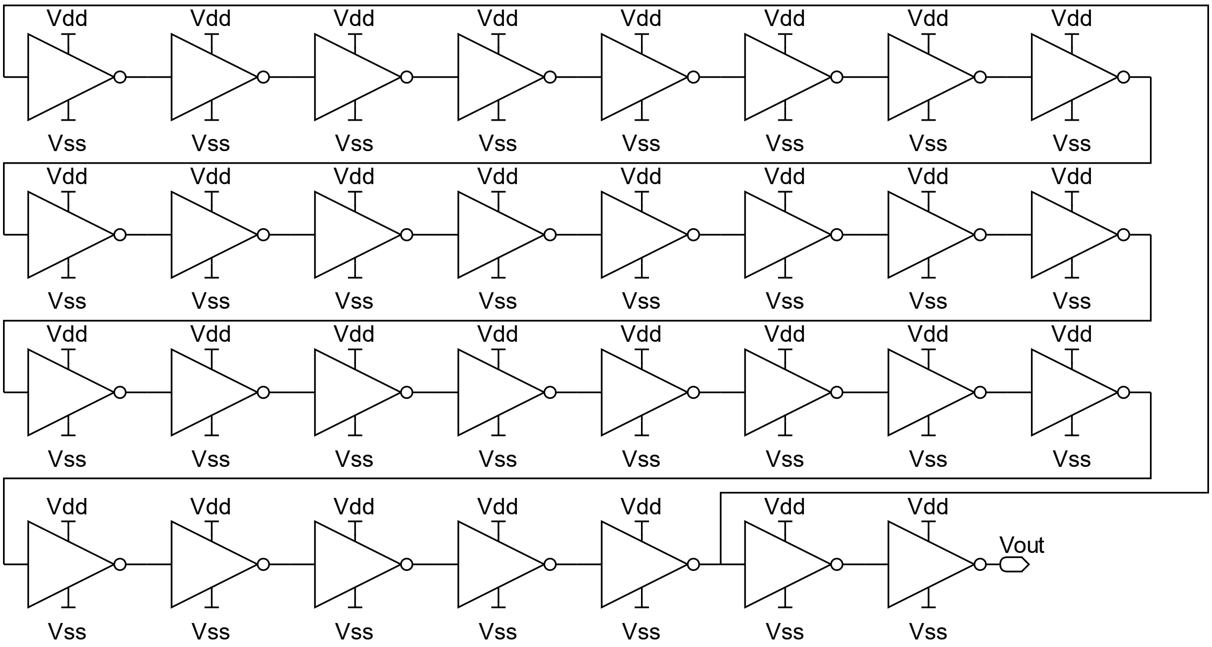


Figure 4.16 Ring oscillator architecture.

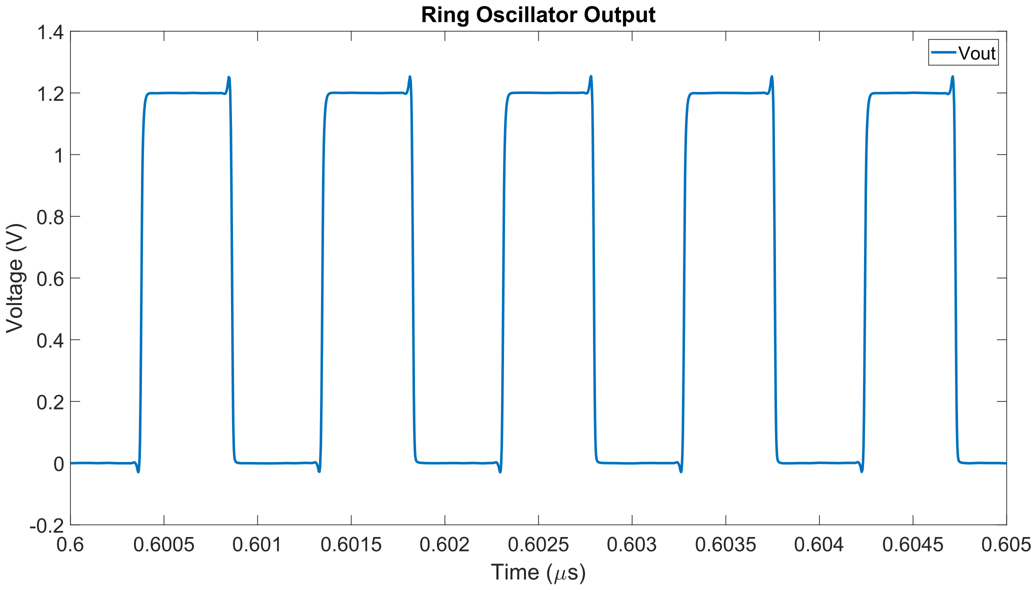


Figure 4.17 Ring oscillator output signal.

5

Analysis and Discussion of Results

In this chapter an analysis and discussion of the results obtained by the circuits designed and implemented in Chapter 4 is presented. In this chapter, CADENCE tools were used to perform the necessary simulation for example transient simulations and Monte Carlos simulations, a Monte Carlo tutorial is in Appendix B. MATLAB was used to perform the statistical test, the Chi-square goodness of fit test, to plot all the histograms and respective probability density functions, to plot the standard deviation graphics and to plot jitter graphics. Examples of MATLAB code for each graphic plotted is in Appendix C.

5.1 Voltage controlled Resistor

The objective of analyzing the VCR is to understand which range of resistance is possible to obtain and how the frequency of the oscillator behaves when the resistance changes. Using the final circuit diagram as represented in Figure 4.14, the voltage V_{res} was gradually changed. The resistance r_{ds} and the frequency of the oscillator was recorded. The voltage on the gate was changed from 0 to 600 mV in increments of 50 mV. The results are in Table 5.1.

Table 5.1 Frequency of the coupled relaxation oscillator and resistance of voltage-controlled resistance for different values of V_{res}.

V_{res}(mV)	f (MHz)	R(Ω)
0	1077.400	151.337
50	969.288	161.290
100	853.912	173.479
150	728.951	188.811
200	603.097	208.808
250	500.000	236.237
300	435.115	276.829
350	383.743	344.930
400	328.377	490.059
450	269.024	986.203
500	218.639	1739.360
550	221.917	1867.390
600	126.603	1936.120

To understand more easily and intuitively the data from Table 5.1, two graphics were plotted. Figure 5.1, which is a graphic where the resistance is plotted against the voltage, and Figure 5.2 where the frequency is plotted against the resistance.

To understand how the resistance of the transistor changes with voltage. A graphic was plotted in Figure 5.1 where on the Y axis, it has resistance r_{ds} and in the X axis, V_{res}.

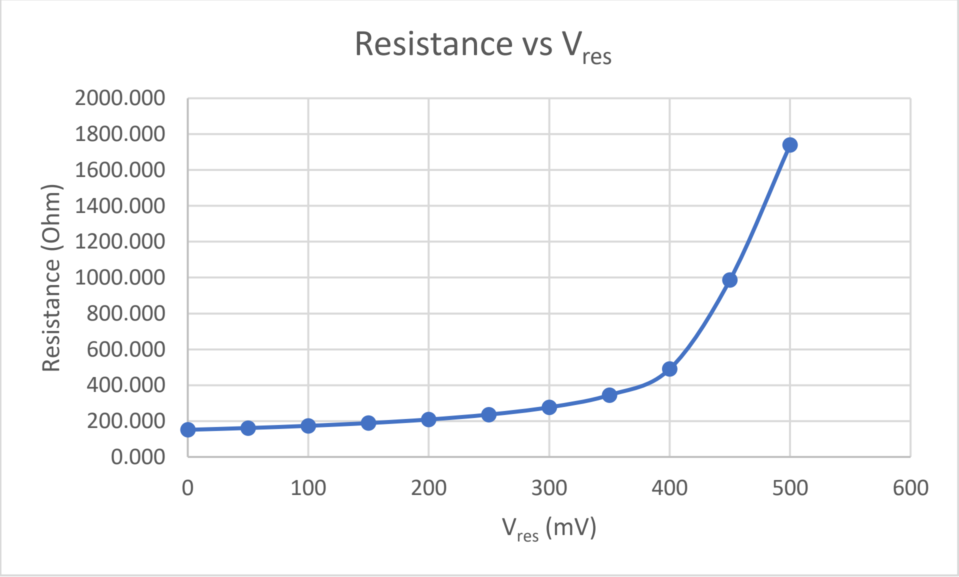


Figure 5.1 Graphic of R_{ds} and V_{res}.

The resistance changes linearly with the voltage until V_{res} reaches around 400 mV. Then it starts to change exponentially. At around $V_{res} = 300$ mV, V_{ds} starts to get closer to V_{dsat} thus, the transistor starts to enter its saturation mode. The transistor reaches its saturation point at

$V_{res} = 400\ \text{mV}$. In saturation mode, even when increasing V_{ds}, the current I_d stays the same thus, following the equation

$$r_{ds} = \frac{V_{ds}}{I_d}$$

As the transistor enters its saturation mode, r_{ds} will increase with a greater slope.

Because the frequency of the oscillation depends on the resistance. Figure 5.2 it was plotted a graphic where on the Y axis, resistance r_{ds} and in the X axis, the frequency of the oscillator.

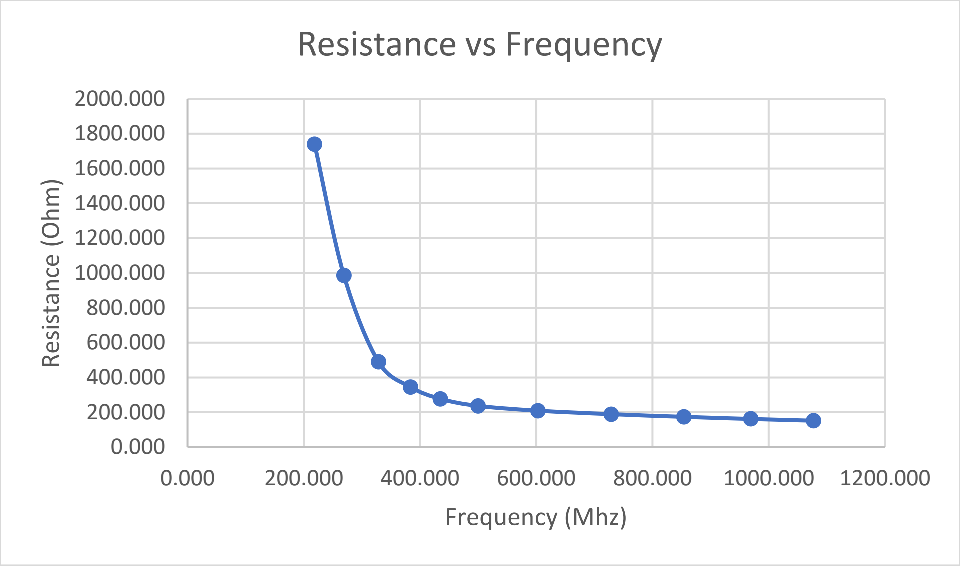


Figure 5.2 Graphic of Rds and frequency.

The objective of the previous graphic is to understand how the frequency of the oscillator changes with r_{ds}. The frequency of the oscillator is inversely linear with the resistance R thus, it was expected that the frequency would change similarly to how R_{ds} change with V_{res}, however, inverted on the X axis. There is a linear relationship between frequency and resistance until from $r_{ds} \approx 151\ \Omega$ to $r_{ds} \approx 490.059\ \Omega$.

5.2 Entropy Source

The evaluation of each circuit was divided into the main five characteristics enumerated in subsection 4.2 that define a good PUF static entropy source. At the end of this chapter, the results obtained from each oscillator were compared. The area, the power consumption, and the jitter analysis were performed on the individual oscillator that composes the entropy source. For the entropy and resistance to temperature and voltage variations, the analysis was done on the entropy source by subtracting the frequency of two oscillators.

The area of the circuit was calculated starting by the transistors, to calculate the area of each transistor as in equation (5.1). Next, calculate the area of each block by adding the areas of all components contained in a block as in equation (5.2). In the case of the ring oscillator, a block was defined by an inverter. In the coupled oscillator, each block was represented by the relaxation oscillator and the soft limiter. Thus, the single relaxation oscillator is composed of a single block. The final area is the area of each block multiplied by the number of equal blocks as in equation (5.3).

The area occupied by each transistor is characterized by the length multiplied by the width of the channel of the transistor.

$$A_{MN} = (L \times W)\ \mu\mathrm{m}^2 \tag{5.1}$$

The area of each block, where N is the number of transistors.

$$A_{block} = (A_{M1} + A_{M2} + \cdots + A_{MN})\ \mu\mathrm{m}^2 \tag{5.2}$$

Finally, the area of each circuit where N_{blocks} is the number of blocks, is given by

$$A_{total} = A_{block} \times N_{blocks} \tag{5.3}$$

It is important to note that the area calculated is not the final area of the circuit, for that, layout techniques need to be applied. However, this way it is possible to have an idea of how different the areas of each circuit might be before performing layout.

The power consumption of each circuit is the sum of all currents of the circuit as in equation (5.4) multiplied by the supply voltage as in equation (5.5).

The currents were obtained from the current sources or in the lack of current sources, by the DC Operating Point in each branch of the circuit. In either case, the total current is

$$I_{total} = I_1 + I_2 + \cdots + I_{N_{branches}} \tag{5.4}$$

Thus, the total power consumption is given by

$$Power\ consumption = V_{supply} \times I_{total} \tag{5.5}$$

For testing and simulation purposes, it was performed a transient simulation. To make the simulation close to a real environment, the noise was added to the transient simulation. The same transient simulation was used for the Monte Carlo simulation.

Since the mismatch variations will cause differences in the phase of the signal. The noise studied is the period jitter which is the deviation in time from the mean clock period of the signal. This way it is possible to evaluate in the time domain, how much the noise will affect the period of the oscillator. The period jitter was evaluated using the calculator tool from cadence ADE L.

The entropy study of the entropy source was performed through Monte Carlo simulation in cadence by obtaining a histogram of the frequency difference between two equal oscillators and performing mismatch tests. Each test was performed with 200 iterations. To get more diversified data, each mismatch test was performed 2 times with different seeds, giving a total of 400 data points. The analysis was performed through a visual and statistical test. The visual analysis consists of comparing the data set kernel distribution to the corresponding normal distribution. The statistical test performed was the Chi-square goodness of fit test which has a null hypothesis of the data coming from a normal distribution.

The resistance to temperature and voltage variations was obtained by performing the same Monte Carlo simulations for entropy, and by forcing such variations to happen and observing how the histogram and density change with those variations. The voltage was changed from 1.08 V to 1.32 V in increments of 0.04 V. The temperature was changed from 7 ℃ to 37 ℃ in increments of 10 ℃. All the variations are compared to the nominal conditions of V_{dd}=1.2 V and Temperature=27 ℃ . This way it is possible to study the static entropy.

5.2.1 Relaxation Oscillator

The circuit used for testing the relaxation oscillator as a static entropy source was shown in Figure 5.3.

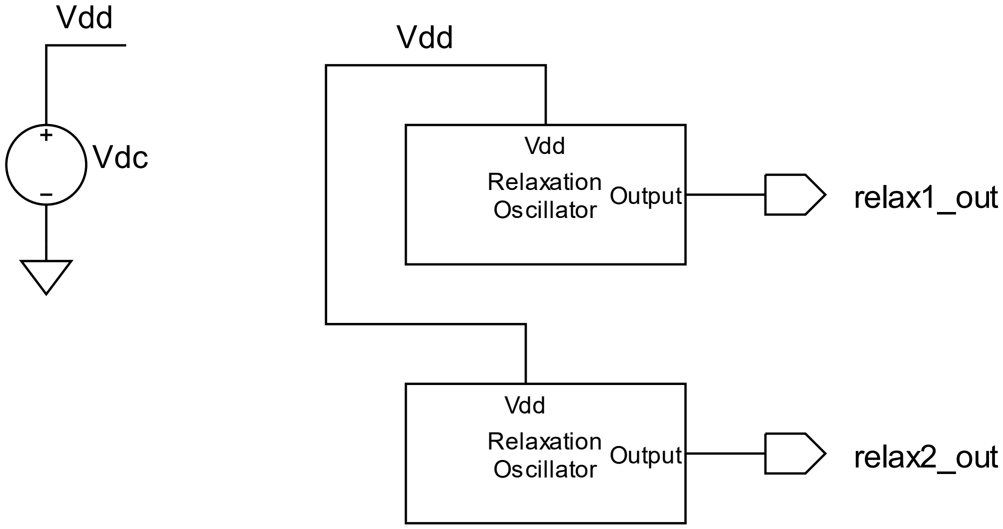


Figure 5.3 Relaxation Oscillator entropy source test circuit.

In each block, it is the relaxation oscillator architecture presented in subsection 4.2.1. The output of the simulation was $f_{relax1\_out} - f_{relax2\_out}$.

5.2.1.1 Area of the oscillator

From the design dimensioning in Table 4.1. From equation (5.1), each relaxation oscillator transistor will have the following area

$$A_{M1} = A_{M2} = 70 * 0.120 = 8.4\ \mu\text{m}^2$$

$$A_{Mvcr1} = A_{Mvcr2} = 15 * 0.120 = 1.8\ \mu\text{m}^2$$

In the technology used, the MIM-Capacitor density is around $1\ \text{fF}/\mu\text{m}^2$, so, for a capacitor with $C = 1.5\ \text{pF}$, the area was given by

$$A_C = 1 * 1500 = 1500\ \mu\text{m}^2$$

The total area of the relaxation oscillator is

$$A_{total} = 2 * A_{M1} + 2 * A_{Mvcr1} + A_C = (2 * 8.4 + 2 * 1.8 + 1500)\ \mu\text{m}^2 \Leftrightarrow$$

$$\Leftrightarrow A_{Relax_{total}} = 1520.4\ \mu\text{m}^2$$

5.2.1.2 Power consumption of the oscillator

The relaxation oscillator is a differential structure with two equal current sources. Both current sources are continuously active. From Table 4.1, each current source output is $1\ \text{mA}$. From equation 5.4, the total current is given by

$$I_{Relax_{total}} = I_{D1} + I_{D2} = (1 + 1)\ \text{mA} \Leftrightarrow$$

$$\Leftrightarrow I_{Relax_{total}} = 2\ \text{mA}$$

As $V_{dd} = 1.2\ \text{V}$, from equation (5.5) the power consumption of this circuit is

$$Power\ consumption_{Relax} = 1.2 * (2 * 10^{-3}) \Leftrightarrow$$

$$\Leftrightarrow Power\ consumption_{Relax} = 2.4\ \mathrm{mW}$$

5.2.1.3 Noise, period jitter of the oscillator

In Figure 5.4, the period jitter is represented in ps over a period of time. The jitter is from the signal at the output of the relaxation oscillator in Figure 4.9. The period jitter is between $-29.276\ \mathrm{ps}$ and $10.010\ \mathrm{ps}$.

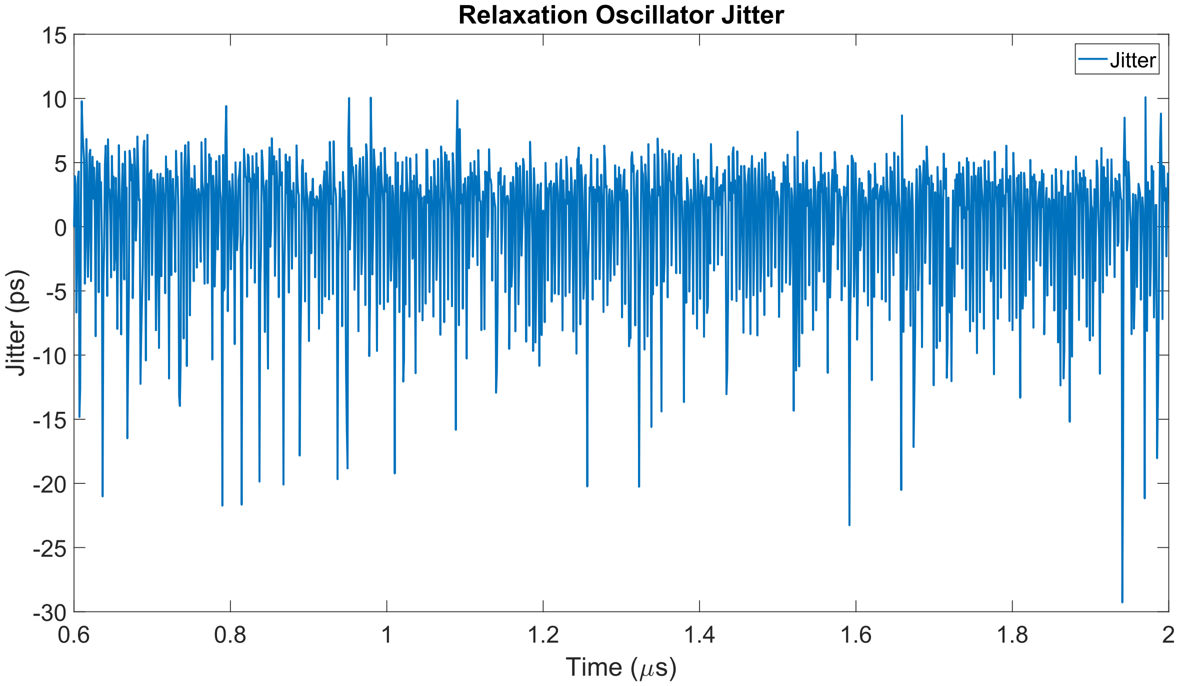


Figure 5.4 Relaxation Oscillator period jitter.

The average absolute value of the period jitter was $4.130ps$.

5.2.1.4 Entropy of the entropy source

In Figure 5.5, the mismatch variation at 27 ℃ and a supply voltage of 1.2 V is shown. In Figure 5.5 a), the results are in the histogram form, distributed across 100 bins, and in Figure 5.5 b), both the kernel (in blue) and normal distribution (in orange) are represented.

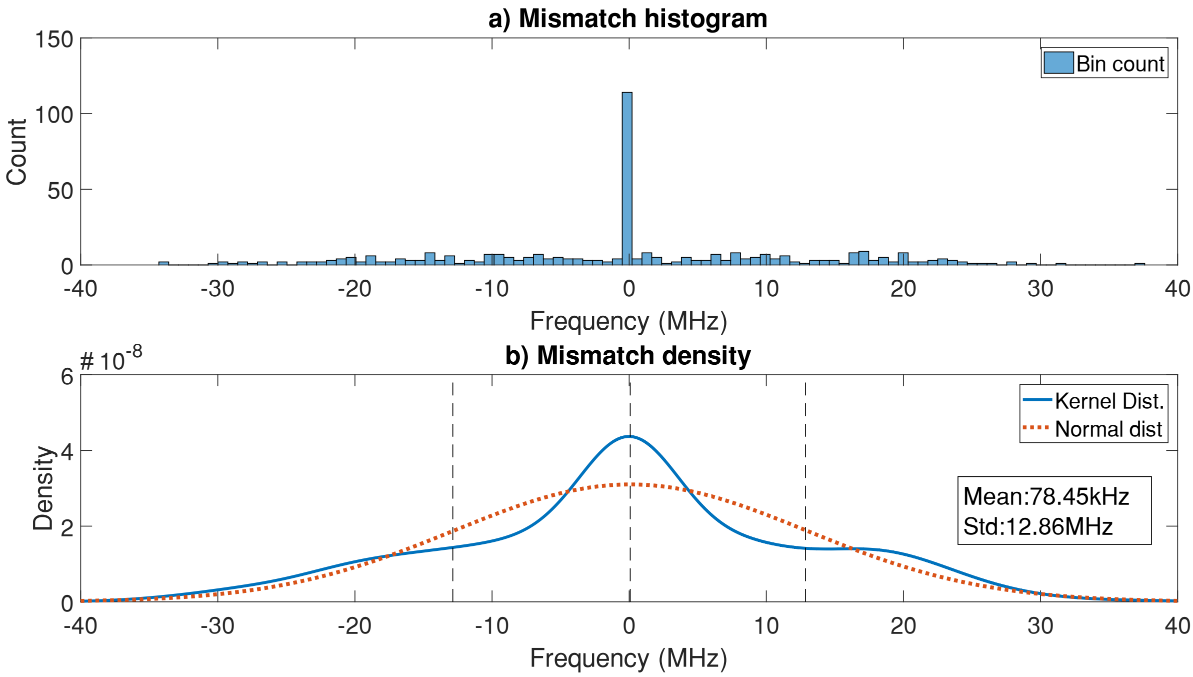


Figure 5.5 Relaxation Oscillator a) Mismatch histogram b) Mismatch density.

In Figure 5.5 a), the mismatch histogram. The results are concentrated in the bin nearest to zero. This means the mismatch variations, is not the main cause of frequency difference between two equal oscillators. This can be due to the large W/L of the transistor and the small transistor count. So, the frequencies of both oscillators remain mostly the same. In a real scenario, this means some oscillators may not be paired to form CRPs because they might have too close of a frequency.

The mean of the results is $78.45\ \mathrm{kHz}$ and the standard deviation is $12.86\ \mathrm{MHz}$.

The previous observation is supported by the density functions in Figure 5.5 b), the kernel distribution of the histogram is compared with the normal distribution of the same histogram and the non-parametric distribution does not resemble the normal distribution, thus the data obtained does not follow a normal distribution.

To confirm the visual analysis from the data set, a statistical test was performed. The results are in Table 5.2.

Table 5.2 Relaxation Oscillator mismatch chi-square goodness-of-fit test results.

Value of Statistical Test	1201.95
Degree of freedom	38
Alpha	0.05
Chi-squared value	53.38
p-value	1.68e-227

Since, $Value\ of\ the\ statistical\ Test > Chi - squared\ value$, the null hypothesis that the data comes from a normal distribution is rejected with a 5% significance level. This confirms by statistical tests that the data does not come from a normal distribution.

5.2.1.5 Resistant to temperature and voltage variations of the entropy source

Starting by analyzing the relaxation oscillator entropy source resilience to temperature variations. For each temperature variation, a mismatch histogram and its corresponding density were plotted as shown in Figure 5.6.

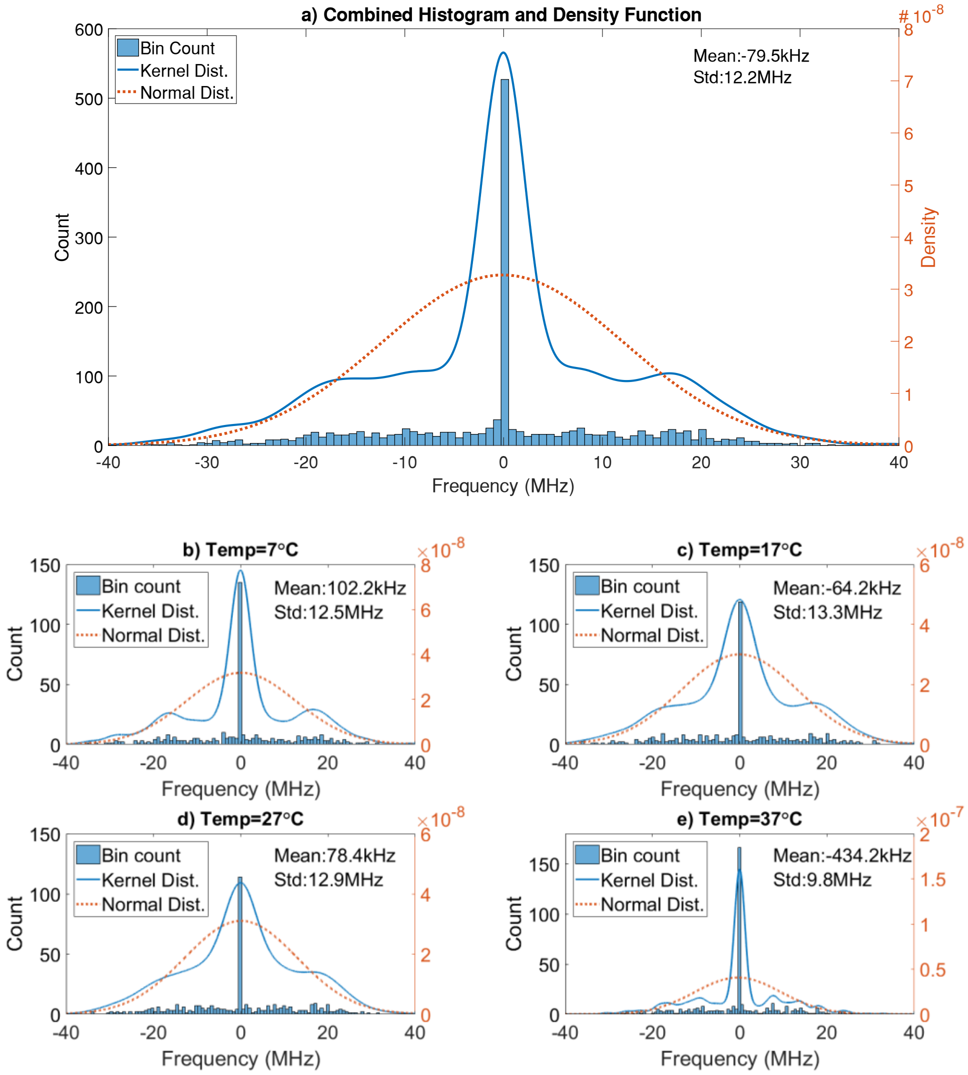


Figure 5.6 Relaxation Oscillator temperature variation mismatch results.

In blue rectangles, are the histogram bins, in lines, are the density functions, and the dashed lines represent the normal distribution density.

The combined histogram and density function in Figure 5.6 a) compared with the histogram and density in nominal conditions (Figure 5.6 e)), at their peak, the distribution shapes are different. If distribution in nominal conditions is different from the combined results, it suggests that the entropy source might not hold its entropy for the imposed variations.

The entropy source might not be able to maintain its entropy across all temperature variations evaluated, however, it might be able to do it across a smaller temperature range. From a visual analysis of all the histograms across all temperature variations, in Figure 5.6 b) to e), the distribution only maintains similar shape and values for a temperature of 17 ℃

To support the idea that the relaxation oscillator might only be able to maintain the entropy in a temperature range from 17 °C to 27 °C, a graphic with the Standard deviation at each temperature was plotted, as shown in Figure 5.7.

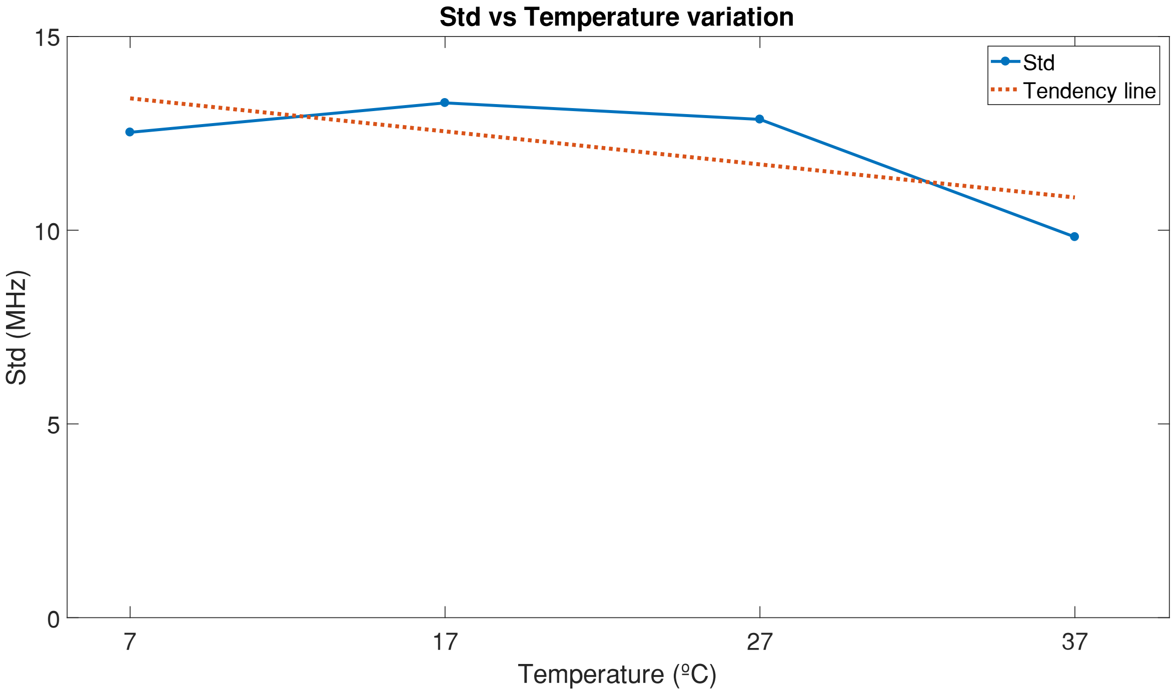


Figure 5.7 Relaxation Oscillator temperature variation standard variation.

In Figure 5.7, it is possible to observe how the standard deviation changes with the temperature.

From 27 °C to 17 °C, the standard deviations are approximately constant. Thus, we conclude that the relaxation oscillator may only maintain robust thus obtain static entropy in a range of temperatures between 17 °C to 27 °C.

To study the resistance to supply voltage variations, for each variation, mismatch histogram, and density functions were plotted as in Figure 5.8.

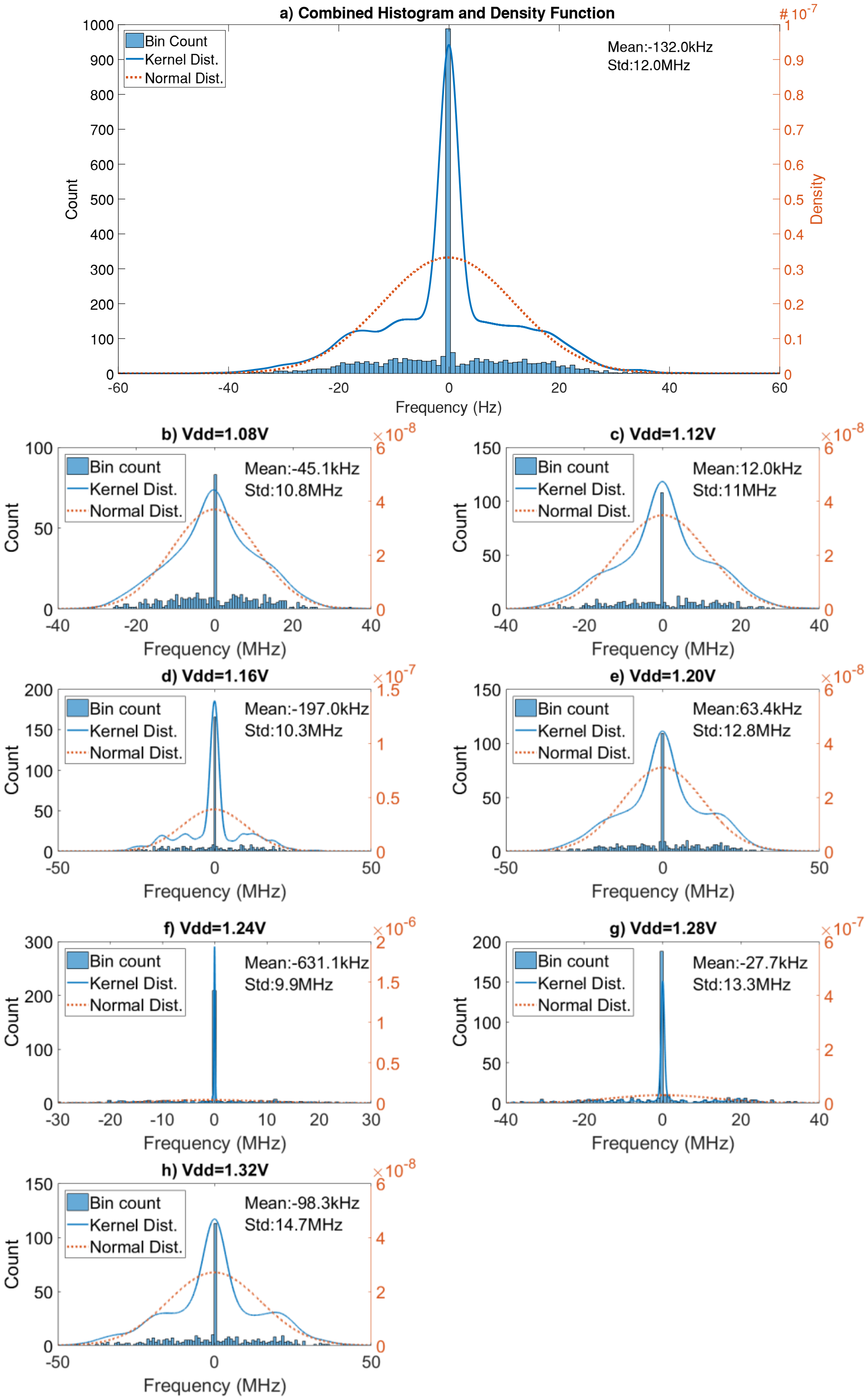


Figure 5.8 Relaxation Oscillator supply voltage variation mismatch results.

By comparing the combined histogram and distribution function in Figure 5.8 a) and the histogram and density in nominal conditions, in Figure 5.8 e), the shape of the distributions and their values are different, suggesting, that the entropy source is not able to maintain its entropy across the range of supply voltage tested.

Comparing the results in nominal conditions and the results for other values of supply voltage as in Figure 5.8 b) c) d) f) g) and h), the entropy source is not able to provide a constant distribution shape for any value of supply voltage.

In Figure 5.9, the standard deviation obtained for different supply voltages was plotted.

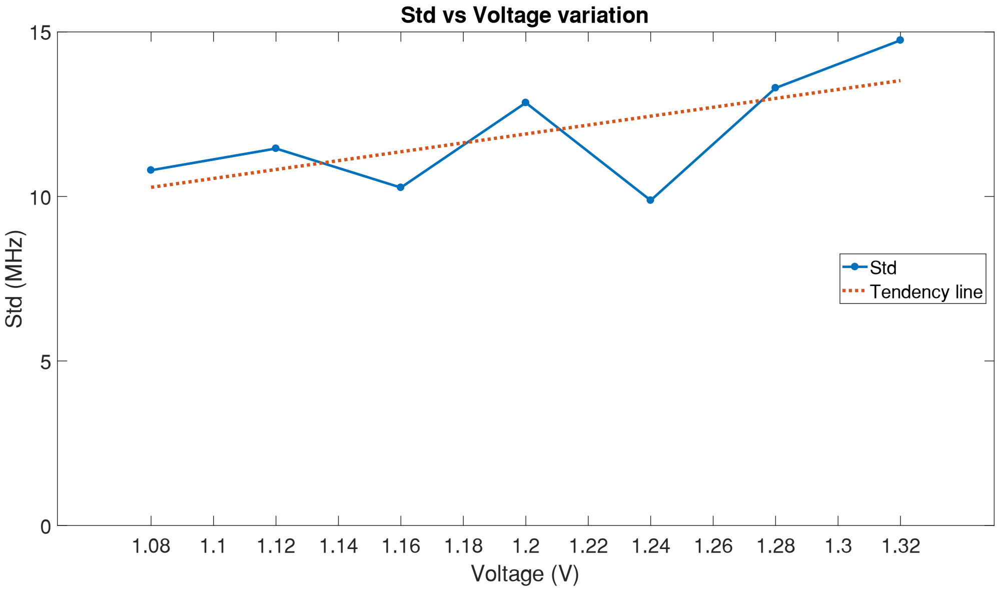


Figure 5.9 Relaxation Oscillator supply voltage variation standard variation.

The standard deviation is not constant when changing the supply voltages. These results and the ones observed in Figure 5.8 support the idea that the relaxation oscillator entropy source is not robust under any voltage variation.

5.2.2 Coupled Relaxation Oscillator

The circuit used for testing is shown in Figure 5.10.

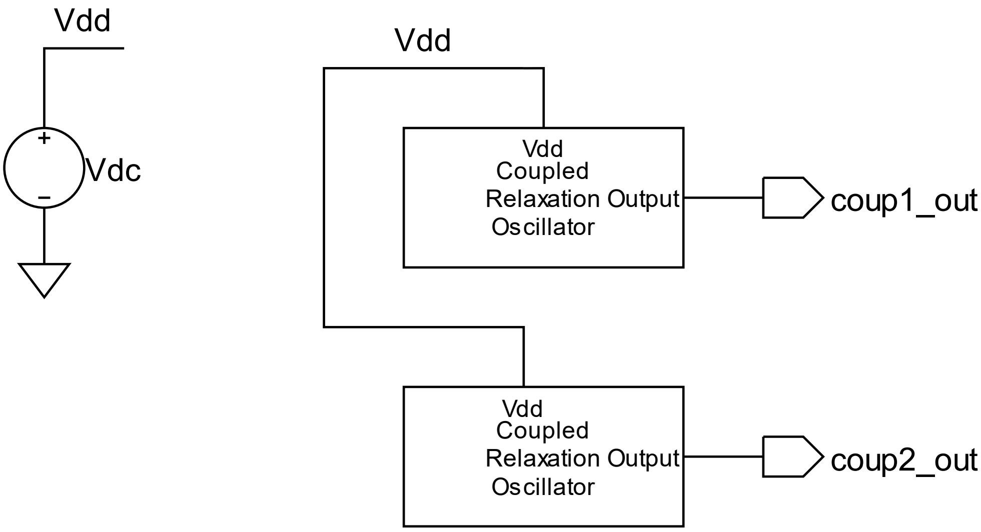


Figure 5.10 Coupled Relaxation Oscillator entropy source test circuit.

In each block, it is the coupled relaxation oscillator architecture presented in subsection 4.2.2. The output of the simulation was chosen to be $f_{coup1\_out} - f_{coup2\_out}$.

5.2.2.1 Area of the oscillator

The Coupled Relaxation Oscillator is composed of 4 relaxation oscillators with the same dimensions as the one in subsection 4.2.2 however with the added soft limiter used for the coupling. Following equation (5.1), the area of each transistor of the soft limiter is

$$A_{SL1} = A_{SL2} = 5 \times 0.120 = 0.600\ \mu\text{m}^2$$

And

$$A_{SL_{total}} = 2 \times A_{SL1} = (2 \times 0.600)\ \mu\text{m}^2 \Leftrightarrow$$
$$\Leftrightarrow A_{SL_{total}} = 1.200\ \mu\text{m}^2$$

Thus, the area of each block is given by

$$A_{coupled_{block}} = A_{Relax_{total}} + A_{SL_{total}} = (1520.4 + 1.200)\ \mu\text{m}^2 \Leftrightarrow$$
$$\Leftrightarrow A_{coupled_{block}} = 1521.600\ \mu\text{m}^2$$

Since the coupled relaxation oscillator uses 4 blocks composed of a relaxation oscillator and a soft limiter, the total is given by

$$A_{coupled_{total}} = 4 * A_{coupled_{block}} = (4 \times 1521.600)\ \mu\text{m}^2 \Leftrightarrow$$
$$\Leftrightarrow A_{ccoupled_{total}} = 6086.4\ \mu\text{m}^2$$

The area of the soft limiter is negligible compared to the area of the single relaxation oscillator; thus, the area of the coupled oscillator is the area of on relaxation oscillator multiplied by N_{relax_osc} oscillators used.

$$A_{coupled_{total}} \approx N_{Relax_{osc}} * A_{Relax_{total}}$$

5.2.2.2 Power consumption of the oscillator

Following the same thought as to calculate the area, a coupled relaxation oscillator block has only one more current source which is from the soft limiter which is

$$I_{SL} = 0.2\text{ mA}$$

Thus, the current from each block of the coupled oscillator is

$$I_{Coupled_{block}} = I_{Relax_{total}} + I_{SL} = (2 + 0.2)\text{ mA} \Leftrightarrow$$
$$\Leftrightarrow I_{Coupled_{block}} = 2.2\text{ mA}$$

Thus, the total current is given by

$$I_{Coupled_{total}} = 4 * I_{Coupled_{block}} = (4 * 2.2)\text{ mA} \Leftrightarrow$$
$$\Leftrightarrow I_{Coupled_{total}} = 8.8\text{ mA}$$

As $V_{dd} = 1.2V$, from equation (5.5) the power consumption of this circuit is

$$Power\ consumption_{Coupled} = 1.2 * (8.8 * 10^{-3}) \Leftrightarrow$$
$$\Leftrightarrow Power\ consumption_{Coupled} = 10.56\text{ mW}$$

5.2.2.3 Noise, period jitter of the oscillator

In Figure 5.11 Coupled Relaxation Oscillator period jitter, the period jitter in ps of the signal at the output of the relaxation oscillator. The period jitter is between -2.858 ps and 2.361 ps.

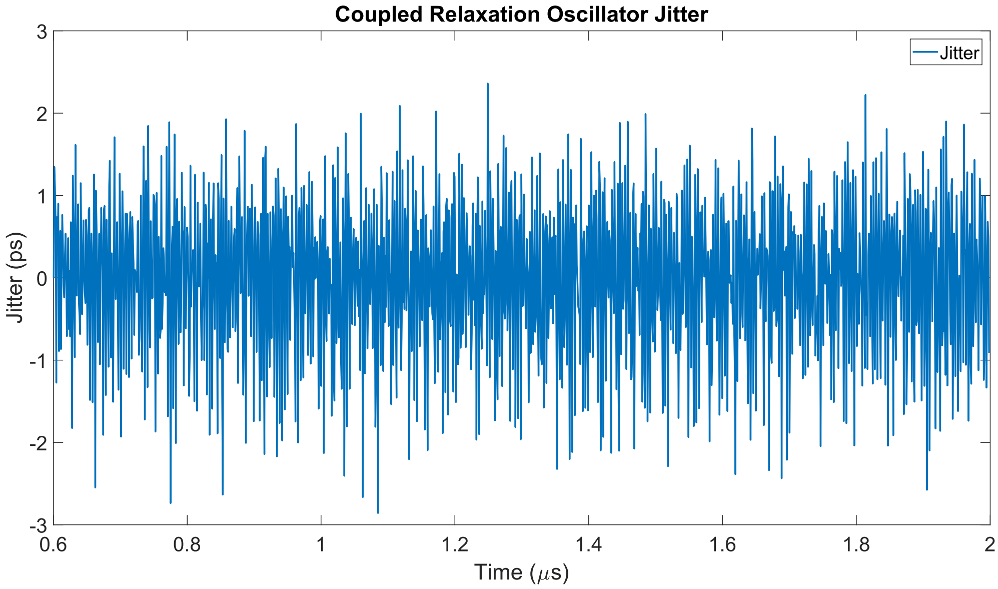


Figure 5.11 Coupled Relaxation Oscillator period jitter.

The average absolute value of the period jitter is 0.731 ps.

5.2.2.4 Entropy of the entropy source

In Figure 5.12, the mismatch variation at nominal conditions is presented. In Figure 5.12 a), the results are shown in the histogram form, and in Figure 5.12 b), both the kernel and normal distribution.

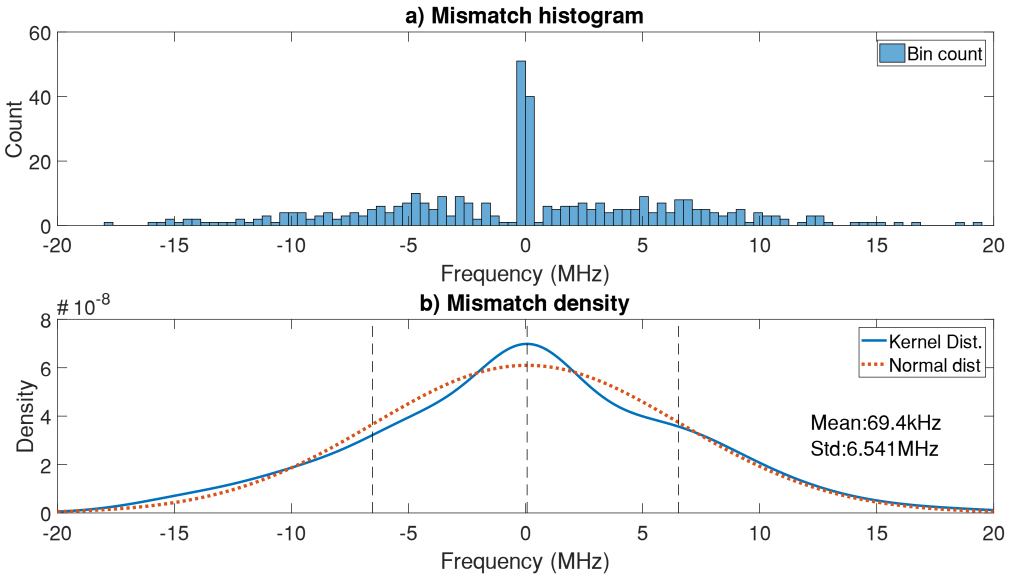


Figure 5.12 Coupled Relaxation Oscillator a) Mismatch histogram b) Mismatch density.

In Figure 5.12 a), the histogram has a narrow set of results. By observing Figure 5.12b), the kernel distribution compared to the normal distribution has centered in zero a higher peak. This implies that the frequencies of both oscillators tend to be the same more than they ideally should be.

Such results indicate that the mismatch variations are not strong enough to spread the frequencies of the oscillators such that when compared, there is a wider set of results.

Another reason for such results can be the coupling circuit. The coupled relaxation oscillator is composed of four relaxation oscillators, each with its variations. However, the coupling circuit will synchronize the oscillators which may reduce the mismatch effect causing the histogram to be narrower.

To confirm the visual analysis from the data set, a statistical test was performed. The results are in Table 5.3.

Table 5.3 Coupled Relaxation Oscillator mismatch chi-square goodness-of-fit test results.

Value of Statistical Test	590.30
Degree of freedom	19
Alpha	0.05
Chi-squared value	30.14
p-value	5.69e-113

Since, $Value\ of\ the\ statistical\ Test > Chi - squared\ value$, the null hypothesis that the data comes from a normal distribution is rejected with a 5 % significance level. This means the statistical test determines the data does not come from a normal distribution.

5.2.2.5 Resistant to temperature and voltage variations of the entropy source

Starting with temperature variation analysis. Figure 5.13, it is represented the histogram and the respective distribution for each temperature. The combined results are represented in Figure 5.13 a) and in Figure 5.13 b), c), d), e) the different results for each temperature tested.

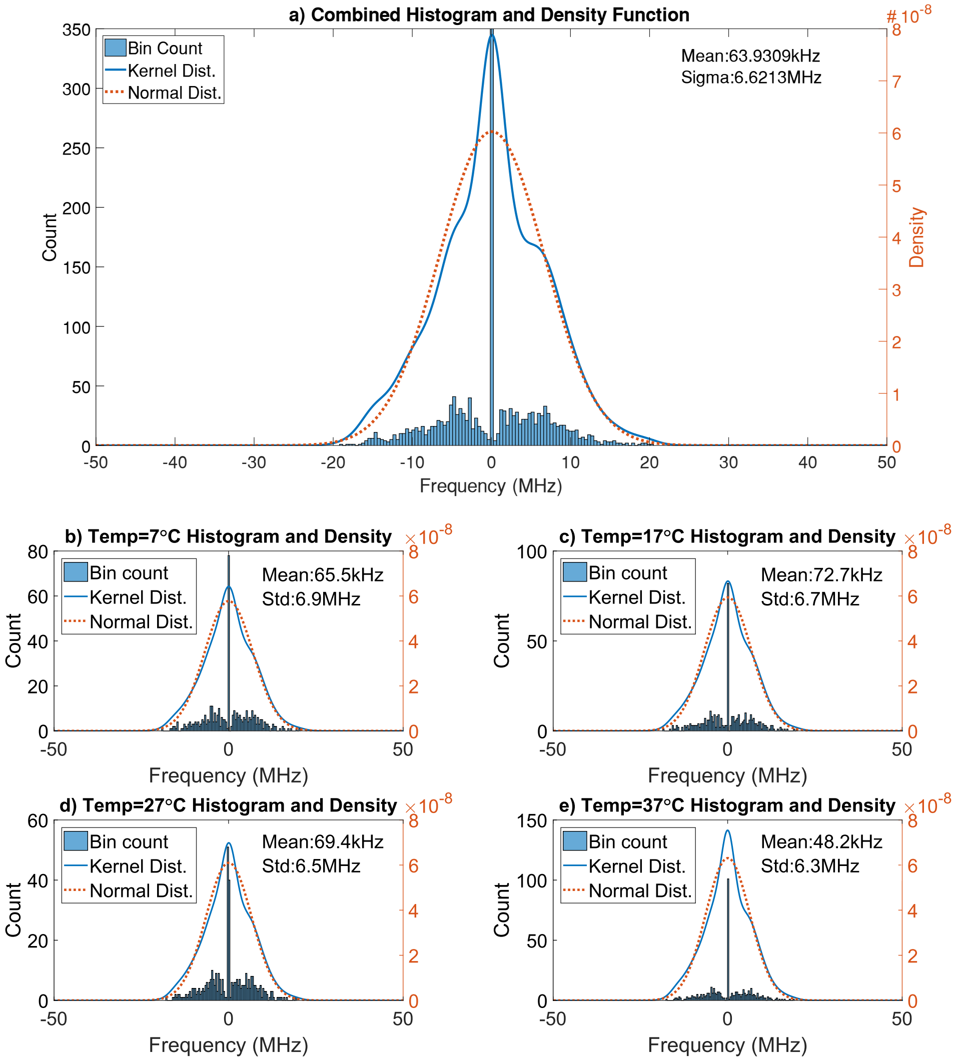


Figure 5.13 Coupled Relaxation Oscillator temperature variation mismatch results.

Comparing the results obtained in the nominal temperature of 27 ℃, Figure 5.13 d), with the combined results, the shape and value of the distribution can keep their shape.

Now comparing the results in nominal conditions with the results obtained at different temperatures as in Figure 5.13 b), d) and e), the distributions maintain similar shapes and values, this may imply that the coupled relaxation oscillator is good at keeping its properties as the temperature changes.

In Figure 5.14, the standard deviation at each temperature tested was plotted.

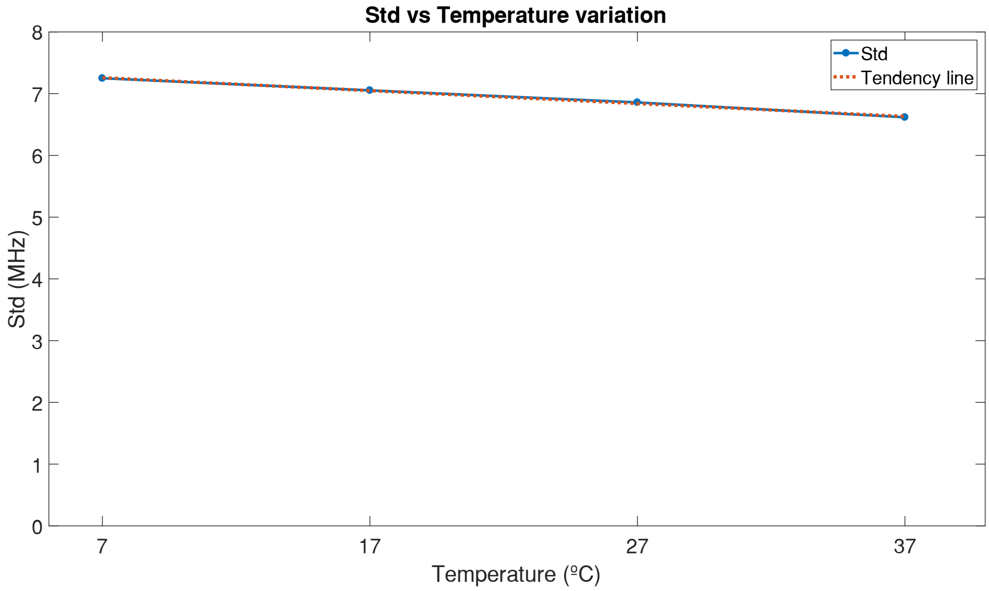


Figure 5.14 Coupled Relaxation Oscillator temperature variation standard variation.

The standard deviation does not suffer significant changes when the temperature fluctuates. Throughout 30 ℃, the standard deviation changed around 6 % from its nominal conditions. Thus, the coupled relaxation oscillator may obtain static entropy within a temperature range from 7 ℃ to 37 ℃. Further testing is needed to conclude if this entropy source may be able to withstand further temperature changes.

Figure 5.15 shows the histograms and the density distributions when varying the supply voltage.

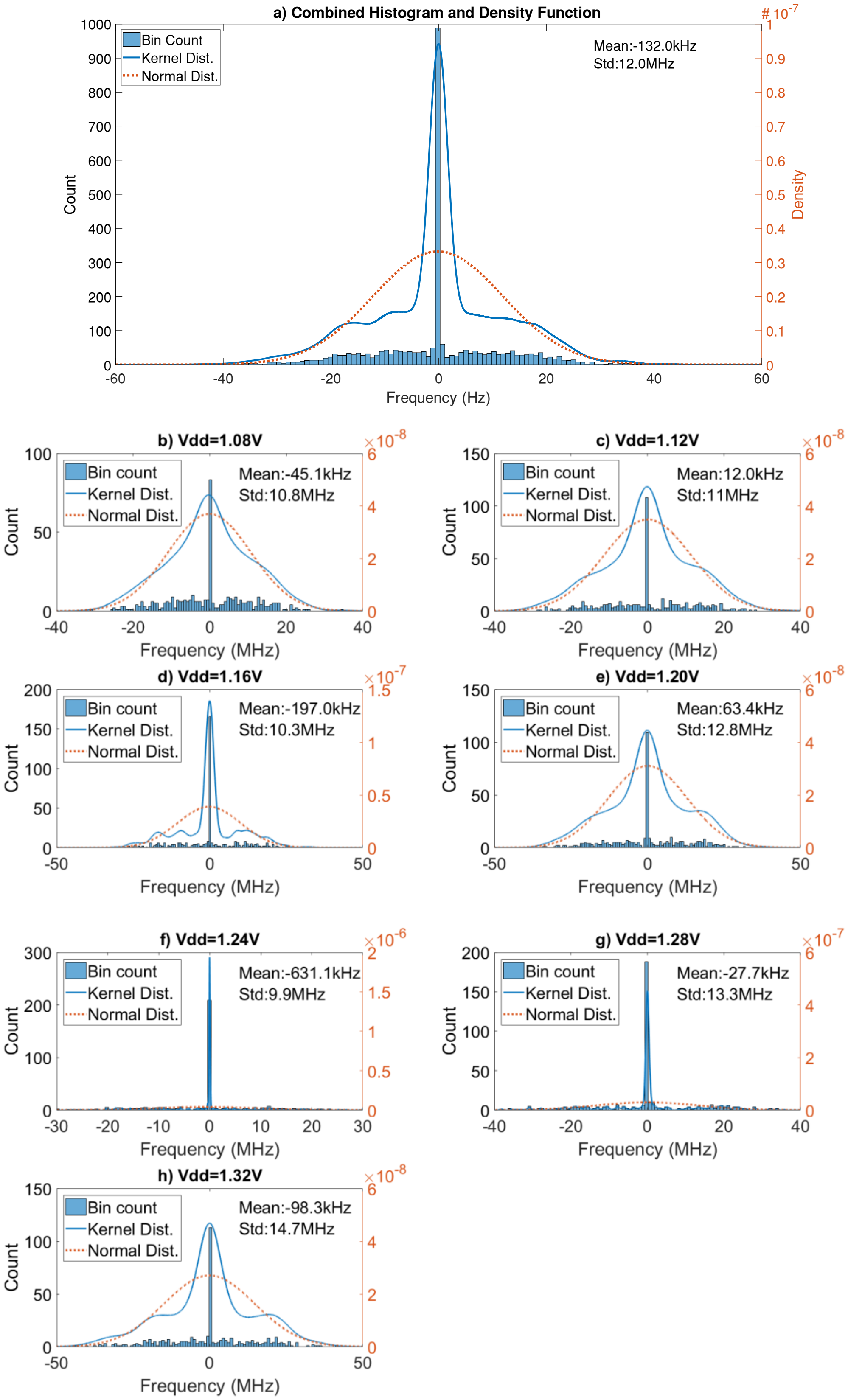


Figure 5.15 Coupled Relaxation Oscillator supply voltage variation mismatch results.

The combined histogram and density function are in Figure 5.15 a). The results in nominal conditions are shown in Figure 5.15 e) and the results at different supply voltages are in Figure 5.15 b), c), d), f), g) and h).

Comparing the results at the nominal condition with the combined results, the distributions are similar in shape and value at their peak.

Comparing the distributions at nominal conditions with the distributions from Figure 5.15 b), c), d), f), g) and h), with varying supply voltage. When V_{dd}=1.12 V and V_{dd}=1.16 V, the count at the highest peak is similar and the shape of the distribution is maintained. This means the entropy source may maintain its entropy to supply voltage variations from V_{dd}=1.2 V to V_{dd}=1.12 V.

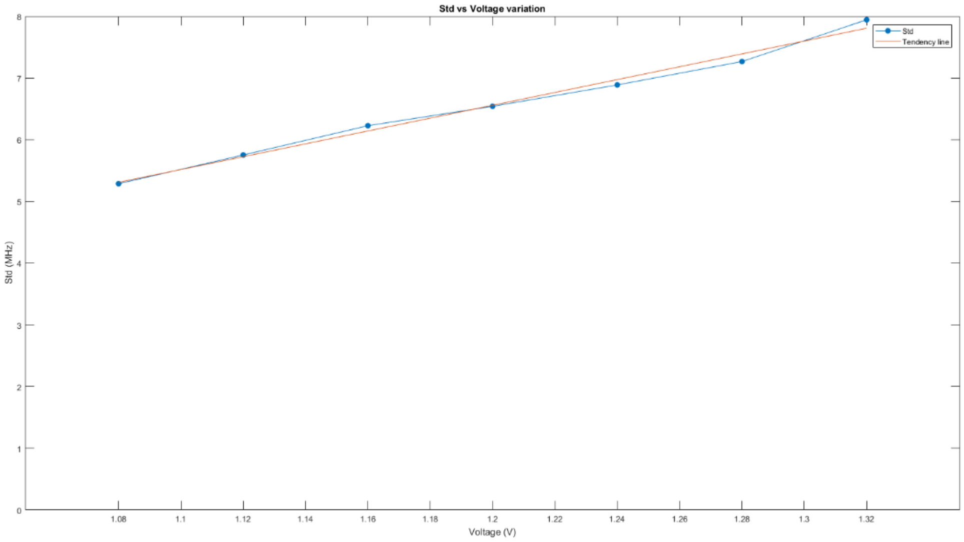


Figure 5.16 Coupled Relaxation Oscillator supply voltage variation standard variation.

In Figure 5.16, it is possible to observe that the standard deviation doesn’t change significantly. At maximum, the standard deviation deviates from 9 %.

Thus, the coupled relaxation oscillator may be able to achieve static entropy when faced with supply voltage variations from V_{dd}=1.2 V to V_{dd}=1.12 V.

5.2.3 Ring Oscillator

The circuit used for testing is shown in Figure 5.17.

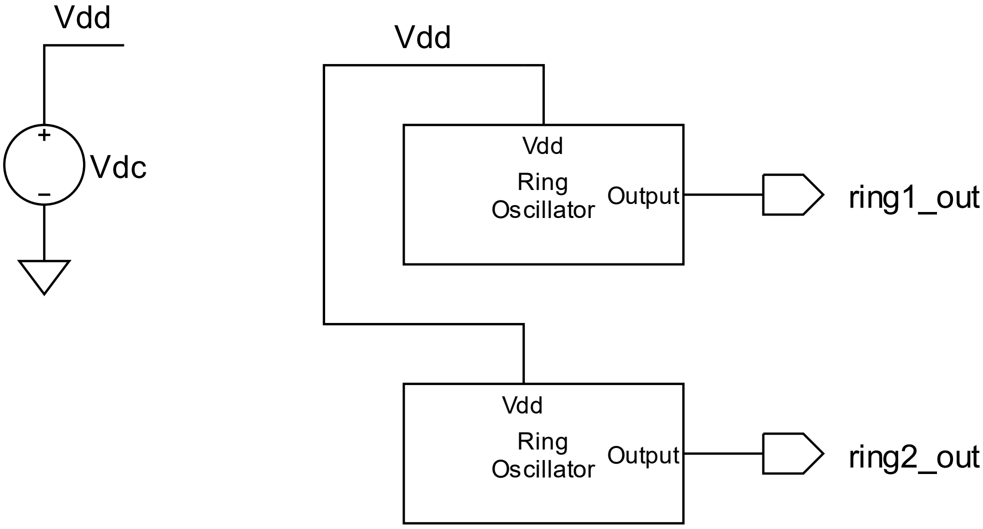


Figure 5.17 Ring Oscillator entropy source test circuit.

In each block, it is the ring oscillator architecture presented in subsection 4.2.3. The output of the simulation was chosen to be $f_{ring1\_out} - f_{ring2\_out}$.

5.2.3.1 Area of the oscillator

Following the ring oscillator dimensioning from Table 4.3 Ring oscillator dimensioning, the area of each transistor was given by equation (5.1), and the area of each block (inverter) was given by equation (5.2).

$$A_{MN} = 0.240 * 0.120 = 0.0288\ \mu\text{m}^2$$

$$A_{MP} = 0.720 * 0.120 = 0.0864\ \mu\text{m}^2$$

Thus, the area of each block is

$$A_{block_{ring}} = A_{MN} + A_{MP} = (0.0288 + 0.0864)\ \mu\text{m}^2 \Leftrightarrow$$

$$\Leftrightarrow A_{block_{ring}} = 0.1152\ \mu\text{m}^2$$

Since the circuit used has 29 equal blocks plus 2 inverters used as buffers, the total area is

$$A_{ring_{total}} = 31 * A_{block_{ring}} = (31 * 0.1152)\ \mu\text{m}^2 \Leftrightarrow$$

$$\Leftrightarrow A_{ring_{total}} = 3.5712\ \mu\text{m}^2$$

5.2.3.2 Power consumption of the oscillator

Using cadence simulation tools, in each block the current is

$$I_{Ring_{block}} = 0.02942\ \text{mA}$$

This means, for 33 blocks, the total current of the oscillator is

$$I_{Ring_{total}} = 31 * I_{Ring_{block}} = 33 * 0.02942 \Leftrightarrow$$

$$\Leftrightarrow I_{Ring_{total}} = \ 0.912 \text{ mA}$$

As $V_{dd} = 1.2V$, from equation (5.5) the power consumption of this circuit is

$$Power\ consumption_{Coupled} = 1.2 * (0.912 * 10^{-3}) \Leftrightarrow$$

$$\Leftrightarrow Power\ consumption_{Coupled} = 1.0944 \text{ mW}$$

5.2.3.3 Noise, period jitter of the oscillator

In Figure 5.18, the period jitter in ps of the signal at the output of the ring oscillator is in Figure 4.16. The period jitter is between -0.836 ps and 0.712 ps.

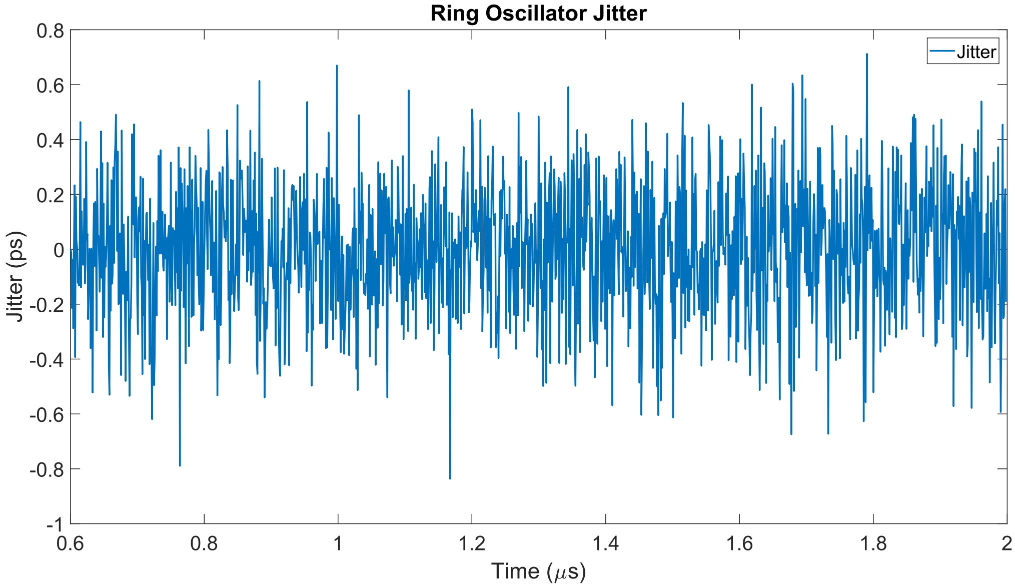


Figure 5.18 Ring Oscillator period jitter.

The average absolute value of the period jitter was 0.184 ps.

5.2.3.4 Entropy of the Entropy Source

In Figure 5.19 the mismatch variation results from the ring oscillator are presented. In Figure 5.19 a), the results are shown in the histogram form, and in Figure 5.19 b), both the kernel and normal distribution.

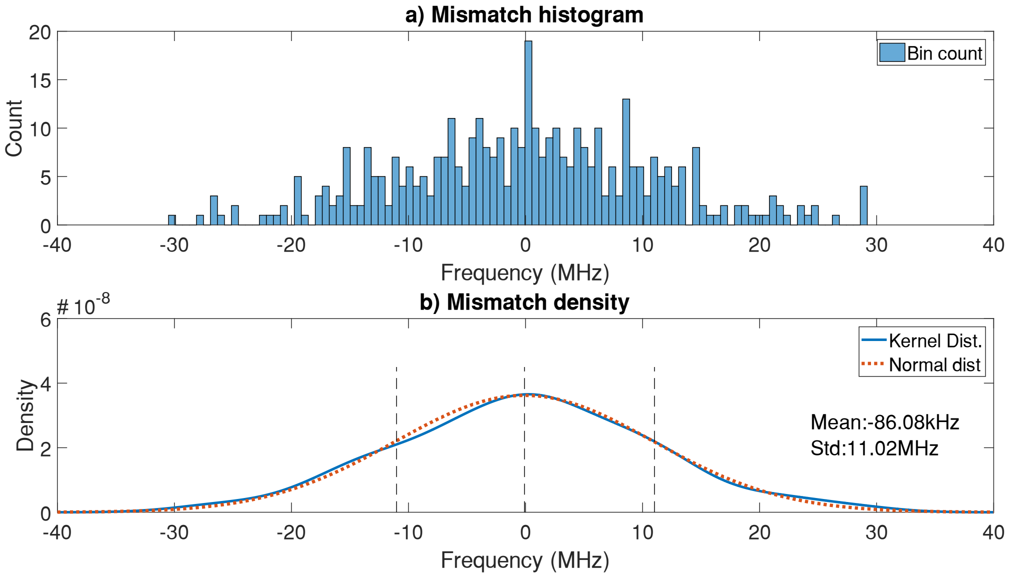


Figure 5.19 Ring Oscillator a) Mismatch histogram b) Mismatch density

In Figure 5.19 a), the histogram appears to have the bin count spread in the frequency increasing slightly as it approaches the central value of 0 Hz. Examining the distribution function in Figure 5.19 b), the non-parametric distribution resembles the normal distribution. This means the circuit may be naturally random and thus serve as a possible good entropy source.

To confirm the visual analysis from the data set, a statistical test was performed to access if the distribution is normal or not. The results are in Table 5.4.

Table 5.4 Ring Oscillator mismatch chi-square goodness-of-fit test results.

Value of Statistical Test	31.58
Degree of freedom	38
Alpha	0.05
Chi-squared value	53.38
p-value	0.76

Since, $Value\ of\ the\ statistical\ Test < Chi - squared\ value$, the null hypothesis that the data comes from a normal distribution is not rejected with a 5 % significance level. This means the statistical test determines the data follows a normal distribution.

5.2.3.5 Resistant to Temperature and Voltage Variations of the Entropy Source

The mismatch results for the ring oscillator entropy source, when faced with temperature variations, are in Figure 5.20.

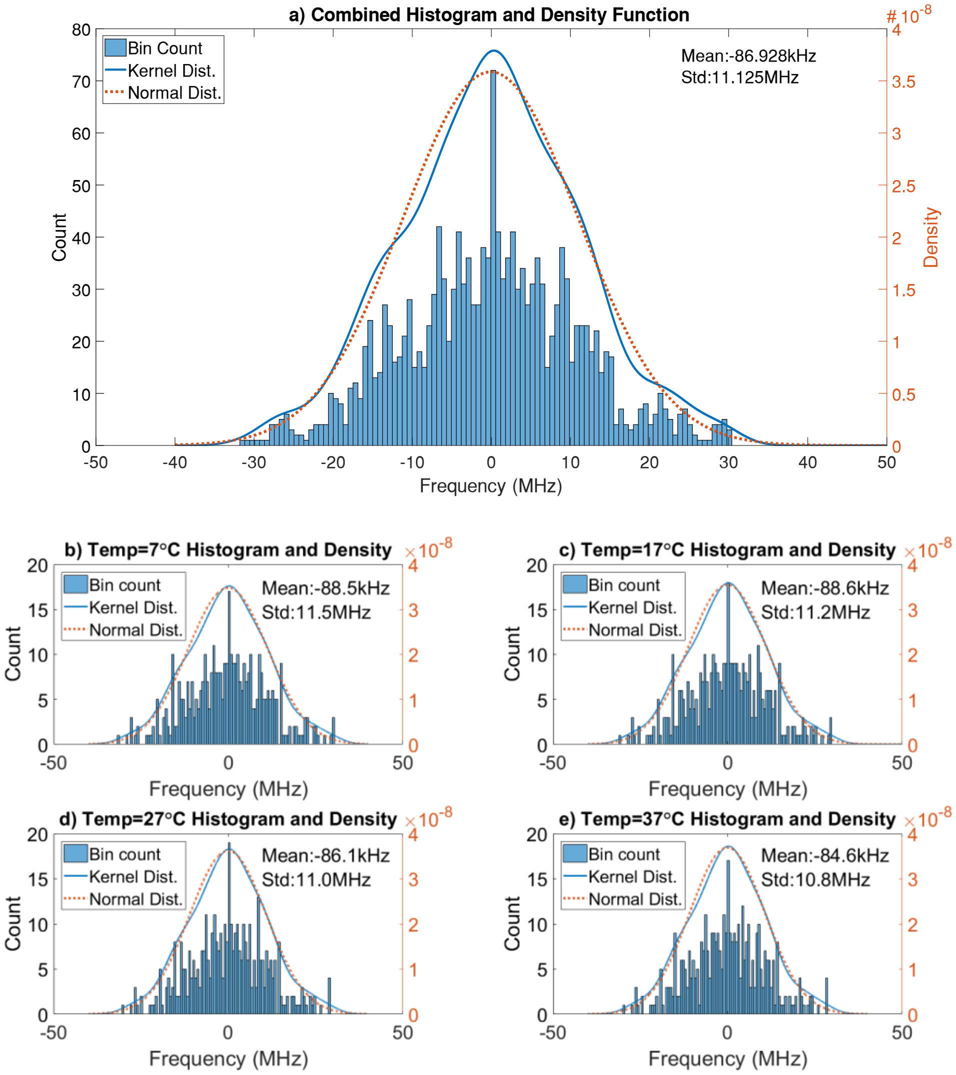


Figure 5.20 Ring Oscillator temperature variation mismatch results.

When faced with temperature variations, the ring oscillator entropy source combined results histogram in Figure 5.20 a) and distribution functions in Figure 5.20 b) have a similar shape and distribution functions similar to the ones in the nominal conditions in Figure 5.20 e) which may suggest that the entropy source hold its entropy.

Comparing the results at different temperatures, in Figure 5.20 c), d) and f), with the nominal condition results. Again, both the bin count and the distributions maintain the same.

The standard deviation resulting from the previous tests was plotted into a graphic in Figure 5.21.

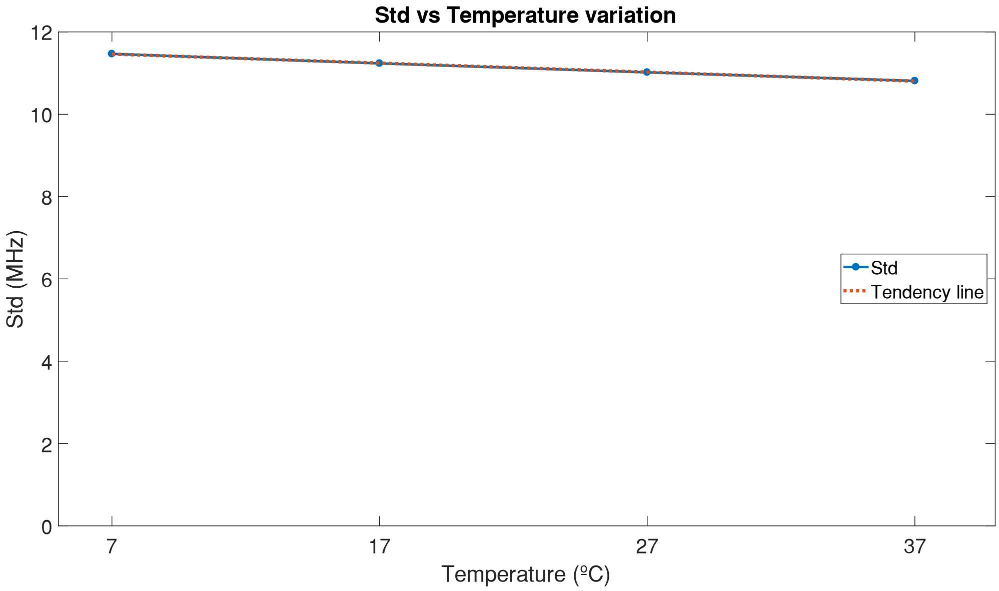


Figure 5.21 Ring Oscillator temperature variation standard variation.

The standard deviation keeps constant through the different temperatures, having a maximum variation of 4 % from the nominal conditions.

From such results, the ring oscillator entropy source may be able to withstand variations in temperature from 7 ℃ to 37 ℃.

Since the ring oscillator at nominal conditions had a normal distribution. For each temperature was checked if the entropy source was able to maintain its normal distribution. The chi-square goodness-of-fit test results are in Table 5.5.

Table 5.5 Ring Oscillator mismatch and temperature variation chi-square goodness-of-fit test results.

Temperature	**7** ℃	**17** ℃	**27** ℃	**37** ℃
Value of Statistical Test	26.42	26.76	31.58	34.01
Degree of freedom	38	38	38	38
Alpha	0.05	0.05	0.05	0.05
Chi-squared value	53.38	53.38	53.38	53.38
p-value	0.92	0.91	0.76	0.65

Since $Value\ of\ the\ statistical\ Test < Chi - squared\ value$ for every temperature tested, the entropy source can maintain its normal distribution for each temperature variation. Thus, the ring oscillator entropy source has a high chance of being able to obtain static entropy through the temperature range of 7 ℃ to 37 ℃.

In Figure 5.22, the histogram and respective density distribution functions when changing the supply voltage.

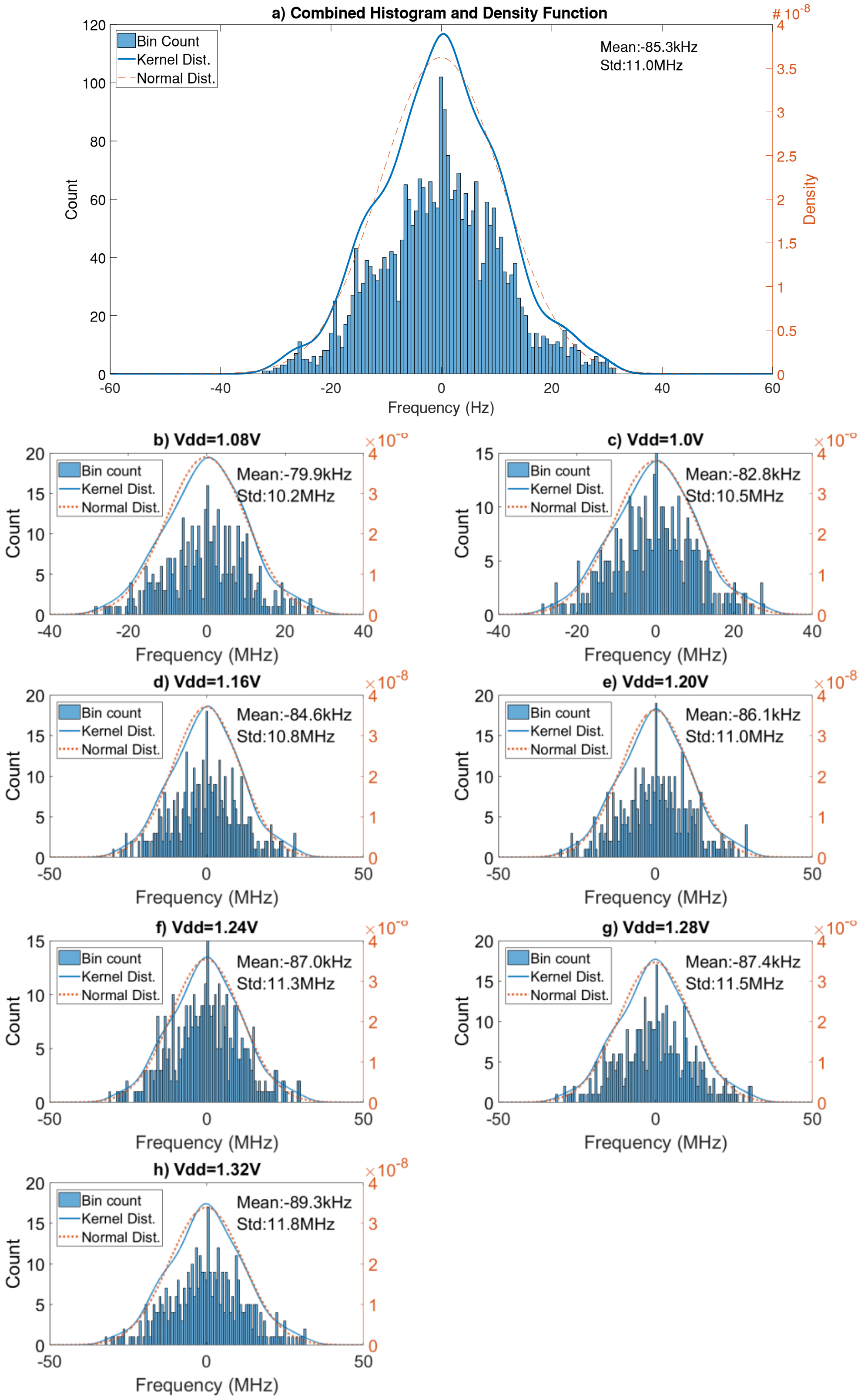


Figure 5.22 Ring Oscillator supply voltage variation mismatch results.

Comparing the combined histogram and density distribution function is shown in Figure 5.22 a) and the with results at the nominal conditions, in Figure 5.22 e), the histogram has similar shapes and the density distributions have not only similar shapes but values. This is a good indication that the ring oscillator entropy source is robust against supply voltage variations.

Comparing the results in nominal conditions with the results at different supply voltages, in Figure 5.22 b), c), d), f), g) and h), all the histogram bin counts, and density distributions functions are similar. Implying that the entropy source may be resilient to supply voltage variations.

The standard deviation was studied following Figure 5.23.

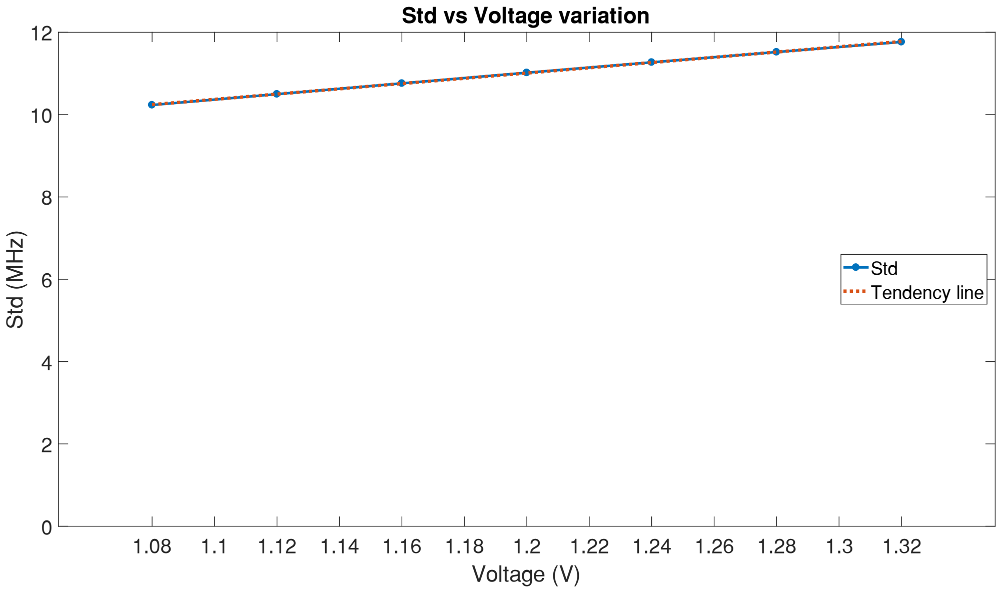


Figure 5.23 Ring Oscillator supply voltage variation standard variation.

The standard deviation is not as constant as seen for temperature variations. From the nominal conditions, the standard deviations change at a maximum of 7 %. Still, from the data acquired, the ring oscillator entropy source may still be able to maintain its entropy through the supply voltage variation considered.

Since the ring oscillator at nominal conditions had a normal distribution. For each supply voltage evaluated, the entropy source capability to maintain the normal distribution was tested. The chi-square goodness-of-fit test results are in Table 5.6.

Table 5.6 Ring Oscillator supply voltage variation chi-square goodness-of-fit test results.

V_{dd}	1.08 V	1.12 V	1.16 V	1.20 V	1.24 V	1.28 V	1.32 V
Value of Statistical Test	52.59	33.90	54.59	31	24.93	46.15	29.90
Degree of freedom	38	37	37	38	38	38	37
Alpha	0.05	0.05	0.05	0.05	0.05	0.05	0.05
Chi-squared value	53.38	52.19	52.19	53.38	53.38	53.38	52.19
p-value	0.06	0.62	0.03	0.76	0.95	0.17	0.79

For $Value\ of\ the\ statistical\ Test > Chi - squared\ value$, the null hypothesis that the data comes from a normal distribution is rejected with a 5 % significance level. This means that the null hypothesis was rejected only for V_{dd}=1.16 V however, the non-parametric distribution is still close to the normal distribution.

If the significance level were softened to 2.5 %, the chi-squared value would be 55.67. Now the statistical test would not reject the null hypothesis.

From such observations, it was considered that the ring oscillator may be able to sustain supply voltages from 1.08 V to 1.32 V. Thus, static entropy may be achieved through the range of supply voltage variation tested.

5.2.4 Results Comparison

In this subsection, the results of each entropy source were compared. The main goal is to understand if the coupled relaxation oscillator is a good entropy source compared to state-of-the-art architecture, the ring oscillator. And understand how the coupling affects the relaxation oscillator behavior as an entropy source.

5.2.4.1 Area of the Oscillator

The area of each oscillator is compared in Table 5.7.

Table 5.7 Area comparison of the oscillators studied.

	Area (μm^2)
Relaxation Oscillator	1520.40
Coupled Relaxation Oscillator	6086.40
Ring Oscillator	3.57

As already noted in subchapter 5.2.2.1, the area of the coupled relaxation oscillator is approximately four times the area of the relaxation oscillator. This was expected since the coupled oscillator uses four relaxation oscillators, and the area of the soft limiter is minimal compared to the area of one relaxation oscillator.

The area of the ring oscillator is 1704.87 times smaller than the area of the coupled relaxation oscillator even though it has more transistors. This difference is mostly because the relaxation oscillator uses a capacitor which is responsible for 98,7 % of the total area of the oscillator. Also, the ring oscillator uses not only minimal channel length but also the minimum acceptable channel width.

A way to minimize the impact of the area of the capacitor is to decrease the capacitance, however, if the capacitance is too low, the rate at which the capacitor charges and discharges become too high, and the oscillator no longer oscillates.

5.2.4.2 Power Consumption of the Oscillator

Table 5.8 it is the power consumption of the oscillators that compose each entropy source. Because all the oscillators are supplied by 1.2 V, the difference in power consumption comes down to the current used by the circuit.

Table 5.8 Power consumption comparison of the oscillators studied.

	Power consumption (mW)
Relaxation Oscillation	2.40
Coupled Relaxation Oscillator	10.56
Ring Oscillator	1.09

The power consumption of the coupled relaxation oscillator is close to four times the consumption of the relaxation oscillator for the same reasons as the area, however, the soft limiter has more impact on the consumption than on the area.

The ring oscillator is the circuit with less consumption which is 9.67 times smaller compared to the coupled oscillator. The ring oscillator transistors function as switches thus, the current through the inverter comes from the small resistance from when the transistor is in the triode state. This means the current on the ring oscillator is extremely low. The coupled relaxation oscillator needs a bigger polarization current to charge the capacitor and be able to oscillate.

However, it is relevant to note that the area of the ring is 704.87 times smaller than the area of the coupled relaxation oscillator and the power consumption is only 9.67 times smaller which means the coupled relaxation oscillator has a better power-to-area performance. Thus, by reducing the area of the coupled oscillator to the size of the ring oscillator it may achieve lower power consumption.

5.2.4.3 Noise, Period Jitter of the Oscillator

In Table 5.9, the period jitter comparison.

Table 5.9 Period jitter comparison of the oscillators studied.

	Minimum (ps)	Absolute Average (ps)	Maximum (ps)
Relaxation Oscillation	-12.680	3.255	5.787
Coupled Relaxation Oscillator	-2.226	0.598	1.875
Ring Oscillator	-0.511	0.134	0.495

The smaller the jitter, the less likely it is to occur a bit flip, and more pairs can be formed between oscillators to form the entropy source.

The coupled relaxation oscillator is an improvement compared to the relaxation oscillator. this was expected due to the coupling circuit which will synchronize all oscillators into one specific frequency thus allowing for better performance.

The ring oscillator has less jitter than the coupled oscillator. Since the ring oscillator is only composed of inverters that perform switching through two transistors, the output capacitance of the ring oscillator is much lower compared to the capacitance of the coupled oscillator, this allows for faster switching and less noise.

Again, decreasing the capacitance of the coupled relaxation oscillator might improve another important characteristic of entropy sources.

5.2.4.4 Entropy of the Entropy Source

The entropy comparison was performed by visual analysis, comparing the shape of the distributions obtained from the Monte Carlo simulations, and by comparing the results using a statistical test, the Chi-square goodness of fit test. This test allows us to distinguish which distributions are and how close represent a normal distribution. The results of the Chi-square goodness of fit test are in Table 5.10.

Table 5.10 Entropy comparison of the oscillators studied.

	Value of Statistical Test	Chi-squared value	p-value
Relaxation Oscillation	1201.95	53.38	1.68e-227
Coupled Relaxation Oscillator	590.30	30.14	5.69e-113
Ring Oscillator	31.58	53.38	0.76

Comparing the relaxation oscillator entropy source density distribution function (Figure 5.5), with the coupled relaxation oscillator entropy source distribution, (Figure 5.12), and the statistical test results in Table 5.10. The coupled oscillator entropy source does have a distribution closer to the normal distribution, improving on the results obtained by the relaxation oscillator thus allowing for an increased number of oscillator pairs to be used for the entropy source. This observation is supported by the results of the statistical test where the coupled relaxation oscillator obtained a p-value higher compared to the relaxation oscillator.

The results are explained due to the higher transistor count in the coupled oscillator entropy source which increases the effect of the mismatch variations on the circuit. The higher mismatch effect on the coupled oscillator allowed dispersing of the results breaking the high-frequency count near 0 Hz as seen by the relaxation oscillator while the coupling allowed for the results. However, the coupled relaxation oscillator entropy source has a great reduction in the standard deviation, this can impact the entropy of the entropy source. This reduction can be due to the coupling circuit.

The coupled oscillator is not an improvement compared to the ring oscillator entropy source distribution (Figure 5.19). The ring oscillator entropy source distribution statistical test results indicate that the data from the ring oscillator follows a normal distribution while having a higher standard deviation. This can be due to the higher number of transistors.

5.2.4.5 Resistance to Temperature and Voltage Variations of the Entropy Source

Starting by comparing the results from the temperature variations.

Comparing the coupled relaxation oscillator entropy source, to the relaxation oscillator entropy source, the coupled oscillator highly improved the performance of the test. Going from sustaining the distribution only between 17 ℃ to 27 ℃ to withstanding all the temperature variations performed by this test, from 7 ℃ to 37 ℃. In Table 5.11, the maximum variation of the standard variation from the nominal conditions is shown. Since the relaxation oscillator did not maintain its distribution, it is not significant to compare the Standard deviation variation.

Table 5.11 Temperature variation standard deviation comparison of the oscillators studied.

	Standard deviation variation
Relaxation Oscillation	-
Coupled Relaxation Oscillator	6%
Ring Oscillator	4%

The ring oscillator entropy source was also able to maintain its distribution through all the temperature variations performed in this test. The ring oscillator, however, performed slightly better with a maximum standard deviation variation from the nominal conditions of 4% while the coupled oscillator performed with a maximum variation of 6 %.

In terms of supply voltage variation, the relaxation oscillator entropy source (Figure 5.8) could not maintain its density distribution function for any supply voltage variations.

The coupled oscillator entropy source (Figure 5.15) could maintain the distribution when faced with supply voltage variations within a range from V_{dd}=1.2 V to V_{dd}=1.12 V, however, outside such supply voltage values, the coupled relaxations showed promising results. The coupled oscillator entropy source is a significant improvement over the relaxation oscillator.

The ring oscillator entropy source can however withstand supply voltage variations between the tested value. The statistical test results in table 5.5 show that the ring oscillator entropy source may be able to keep its entropy across the range of supply voltage variation tested. Again, the ring oscillator outperformed the circuit to be tested as an entropy source.

However, as stated before, the poor results with both the relaxation and the coupled relaxation oscillator may be due to the use of a VCR, when the supply voltage is different from the nominal value, the voltage across the VCR changes, which changing its resistor value thus, the characteristics of the oscillators are changed. Thus, the VCR is adding uncontrollable variations which what static entropy source try to avoid.

6

Conclusions and future work

6.1 Conclusions

The main purpose of this project was to evaluate the use of the coupled oscillator as a static entropy source. Due to the coupling circuit, this oscillator is good at preserving its properties when faced with external variations. this is the most important characteristic of a static entropy source.

However, the coupled relaxation oscillator static entropy source, performed below expectation when compared with the state-of-the-art implementation, the ring oscillator.

The entropy of the coupled oscillator entropy source did not have a normal distribution which indicates the results are not naturally random. The robustness to external variations like temperature and supply voltage changes showed promising results, the coupled relaxation oscillator static entropy source showed it can be robust to external variations. Under supply voltage variations the results were less satisfying due to the use of a VCR. The ability to fine-tune the oscillator to a specific frequency and the smaller area of a transistor, might not be worth the loss in entropy due to supply voltage variations.

Even when the coupled oscillator demonstrated promising results it still did not perform as well as the ring oscillator. Thus, compared to the ring oscillator, the developed circuit have the following characteristics:

- The high area leads to higher manufacturing costs.
- High power consumption, which limits the possible portability of the device.
- Low period jitter, it is important to increase the possible combination of CRP pairs.
- Statistically, poor entropy.
- Statistically, good robustness to temperature variations.

- Statistically, inferior robustness to supply voltage variations.

The fact that PUFs rely on the physical properties of the circuit was a major challenge when studying and designing such devices. To fully test a PUF and all its characteristics, a physical circuit is needed.

With the tools we had, there was no way to introduce mismatch variations into transient simulations and study individual oscillators with individual mismatch variations. This makes it impossible to:

- Test the entropy extractor (challenge and response pairs).
- Gather a string of bits and obtain an output.
- Test the stability (BER), uniqueness (Inter-PUF hamming distance), reproducibility (Intra-PUF hamming distance), identifiability, and the randomness of the output.

Therefore, this work was focused on the entropy source and its potential as a static entropy source. The entropy source does rely on mismatch variations however, using Monte Carlo simulation and statistical data from the transistor, it was possible to study the overall distribution of frequencies of when such oscillators are produced thus studying its potential behavior under various conditions.

Since the entropy tests performed are only statistical. It was important to have means of comparison to evaluate the circuit design. For such comparison, a well-known entropy source had to be used, thus a ring oscillator entropy source was implemented and compared with the circuit designed.

A lot of experimenting was involved when starting this dissertation, to evaluate the randomness of the implemented circuits, data from the period jitter was initially analyzed. This was done by processing data from the period jitter in MATLAB and obtaining a 1 or a 0 in the points where the jitter was either positive or negative simulating an ideal sampler and after this, evaluate the randomness using NIST tests. However, we then understood this method is only feasible to RNGs since we are obtaining a bit of strings from a noise of the circuit.

Thus, in this final work, even though the NIST tests were implemented they were not used in the evaluation of the entropy source because no string of data was obtained. However, these tests are important to evaluate the randomness of the string of data that can, if future work is done be obtained at the output of a PUF.

6.2 Future Work

The work developed by this thesis is preliminary work to produce a delay based PUF using oscillators. By also considering the results obtained, for future work is necessary to:

- Change the VCR with a fixed resistor and perform the same analysis.
- Improve the area, the power consumption, and entropy of the entropy source.
- Produce the entropy source and entropy extractor in integrated circuits.
- Extract a bit sequence and test its stability, uniqueness, reproducibility, identifiability, and randomness.

Improving the area is especially important to cut down the costs of production and, also, smaller devices are more prone to bigger mismatch variations which can increase the entropy of the circuit. The largest influence on the area of the circuit was the capacitor which can be reduced. Reducing the capacitance can also reduce the jitter which will increase the number of possible oscillator pairs and increase the static entropy.

The power reduction can be decreased by reducing the current consumed by the circuit. For example, the coupling circuit has been shown to reduce the dispersion of results in the Monte Carlo simulation, reducing the current on the soft limiter, will decrease the coupling effect which can result in better entropy and less power consumption.

By extracting the bit sequence from the physical circuit, the NIST tests can be used to assess the randomness of such bit sequence.

BIBLIOGRAFIA

[1] João M. Lourenço, "The NOVAthesis Template User's Manual," 2021. Accessed: Mar. 01, 2022. [Online]. Available: https://github.com/joaomlourenco/novathesis_word/raw/master/novathesis_word-FINAL-EN.pdf

[2] X.-S. Yang and Institute of Electrical and Electronics Engineers, *Proceedings of the World Conference on Smart Trends in Systems, Security and Sustainability (WS4 2020) : July 27-28, 2020, virtual conference*.

[3] F. Samie, V. Tsoutsouras, L. Bauer, S. Xydis, D. Soudris, and J. Henkel, "Computation offloading and resource allocation for low-power IoT edge devices," in *2016 IEEE 3rd World Forum on Internet of Things, WF-IoT 2016*, Feb. 2017, pp. 7–12. doi: 10.1109/WF-IoT.2016.7845499.

[4] Z. K. Zhang, M. C. Y. Cho, C. W. Wang, C. W. Hsu, C. K. Chen, and S. Shieh, "IoT security: Ongoing challenges and research opportunities," in *Proceedings - IEEE 7th International Conference on Service-Oriented Computing and Applications, SOCA 2014*, Dec. 2014, pp. 230–234. doi: 10.1109/SOCA.2014.58.

[5] A. Shamsoshoara, A. Korenda, F. Afghah, and S. Zeadally, "A Survey on Physical Unclonable Function (PUF)-based Security Solutions for Internet of Things," Jul. 2019, [Online]. Available: http://arxiv.org/abs/1907.12525

[6] M. Alioto, "Trends in hardware security from basics to ASICs," *IEEE Solid-State Circuits Magazine*, vol. 11, no. 3. Institute of Electrical and Electronics Engineers Inc., pp. 56–74, Jun. 01, 2019. doi: 10.1109/MSSC.2019.2923503.

[7] C. E. Shannon, "A Mathematical Theory of Communication," *The Bell System Technical Journal*, vol. 27, pp. 614–618, 1948.

[8] R. Renner and S. Wolf, "Smooth rényi entropy and applications," in *IEEE International Symposium on Information Theory - Proceedings*, 2004, p. 233. doi: 10.1109/isit.2004.1365269.

[9] V. P. Yanambaka, S. P. Mohanty, and E. Kougianos, "Making use of manufacturing process variations: A dopingless transistor based-puf for hardware-Assisted security," in *IEEE Transactions on Semiconductor Manufacturing*, May 2018, vol. 31, no. 2, pp. 285–294. doi: 10.1109/TSM.2018.2818180.

[10] J. R. Westra, "High-performance oscillators and oscillator systems," doctoral thesis, Delft University of Technology, 1998.

[11] J. R. Westra, C. J. M. Verhoeven, and A. H. M. van Roermund, *Oscillators and Oscillator Systems: Classification, Analysis and Synthesis*. Springer US, 1999. doi: 10.1007/978-1-4757-6117-7.

[12] C. J. M. Verhoeven, "First order oscillators," doctoral thesis, Delft University of Technology, 1990.

[13] R. Senani, · D R Bhaskar, V. K. Singh, and · R K Sharma, *Sinusoidal Oscillators and Waveform Generators using Modern Electronic Circuit Building Blocks*. Springer, 2016.

[14] T. Schubert and K. Ernest, *Fundamentals of Electronics: Book 4 Oscillators and Advanced Electronics Topics*. Morgan & Claypool Publichers, 2016.

[15] E. Hegazi, J. Rael, and A. Abidi, *The designer's guide to High-Purity Oscillators*. Kluwer Academic Publishers, 2005.

[16] Alessio Damato, "Hysteresis sharp curve," *https://commons.wikimedia.org/w/index.php?curid=528681*, Jan. 01, 2006. https://commons.wikimedia.org/w/index.php?curid=528681 (accessed Aug. 19, 2022).

[17] Oliveira Luis B., Fernandes Jorge R., Filanovsky Igor M., Verhoeven Chris J.M., and Silva Manuel M., *Analysis and Design of Quadrature Oscillators*. Springer, 2008.

[18] M. Silva, *Circuitos Com Transistores Bipolares e MOS*, 5th ed. Fundação Caloust Gulbenkian, 2013.

[19] B. Charan Sarkar, M. K. Mandal, and C. Sarkar, "Ring oscillators: Characteristics and applications," 2010. [Online]. Available: https://www.researchgate.net/publication/234046858

[20] Y. Su, J. Holleman, and B. Otis, "A1.6pJ/blt 96% stable chip-ID generating circuit using process variations," in *Digest of Technical Papers - IEEE International Solid-State Circuits Conference*, 2007, pp. 406–408. doi: 10.1109/ISSCC.2007.373466.

[21] S. Rosenblatt *et al.*, "Field tolerant dynamic intrinsic chip ID using 32 nm High-K/Metal Gate SOI embedded DRAM," *IEEE J Solid-State Circuits*, vol. 48, no. 4, pp. 940–947, 2013, doi: 10.1109/JSSC.2013.2239134.

[22] S. K. Mathew *et al.*, "A 0.19pJ/b PVT-variation-tolerant hybrid physically unclonable function circuit for 100% stable secure key generation in 22nm CMOS," in *Digest of Technical Papers - IEEE International Solid-State Circuits Conference*, 2014, vol. 57, pp. 278–279. doi: 10.1109/ISSCC.2014.6757433.

[23] S. Arslan Tuncer and T. Kaya, "True Random Number Generation from Bioelectrical and Physical Signals," *Comput Math Methods Med*, vol. 2018, 2018, doi: 10.1155/2018/3579275.

[24] R. Podeti, S. Rao Patri, S. Katkoori, and P. Muralidhar, "ReOPUF: Relaxation Oscillator Physical Unclonable Function for Reliable Key Generation for IoT Security," 2021.

[25] M. Alioto and T. Sachin, *Enabling Ubiquitous Hardware Security via Energy-Efficient Primitives and Systems : (Invited Paper)*, vol. IEEE (CICC). IEEE, 2019.

[26] L. E. Bassham *et al.*, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," Gaithersburg, MD, 2010. doi: 10.6028/NIST.SP.800-22r1a.

[27] "1.3.5.15. Chi-Square Goodness-of-Fit Test." https://www.itl.nist.gov/div898/handbook/eda/section3//eda35f.htm (accessed Sep. 27, 2022).

[28] D. P. Kroese, T. Brereton, T. Taimre, and Z. I. Botev, "Why the Monte Carlo method is so important today," *Wiley Interdiscip Rev Comput Stat*, vol. 6, no. 6, pp. 386–392, Nov. 2014, doi: 10.1002/wics.1314.

[29] "1.3.3.14. Histogram." https://www.itl.nist.gov/div898/handbook/eda/section3//eda33e.htm (accessed Sep. 27, 2022).

[30] H. O. Uchechi, "Parametric and Nonparametric statistics hematological parameters of wistar rats treated with chromolaena odorata leaf extracts View project Laboratory information Management System View project," 2019. [Online]. Available: https://www.researchgate.net/publication/334733468

[31] Moradi Mona, Mirzaee Reza, and Tao Sha, "CMOS Arbiter Physical Unclonable Function with Selecting Modules," Jun. 2020.

[32] A. Garg and T. T. Kim, "Design of SRAM PUF with improved uniformity and reliability utilizing device aging effect," in *Proceedings - IEEE International Symposium on Circuits and Systems*, 2014, pp. 1941–1944. doi: 10.1109/ISCAS.2014.6865541.

[33] G. Selimis *et al.*, "Evaluation of 90nm 6T-SRAM as Physical Unclonable Function for secure key generation in wireless sensor nodes," in *Proceedings - IEEE International Symposium on Circuits and Systems*, 2011, pp. 567–570. doi: 10.1109/ISCAS.2011.5937628.

[34] R. Maes, P. Tuyls, and I. Verbauwhede, "A soft decision helper data algorithm for SRAM PUFs," in *IEEE International Symposium on Information Theory - Proceedings*, 2009, pp. 2101–2105. doi: 10.1109/ISIT.2009.5205263.

[35] M. Bucci, L. Germani, R. Luzzi, A. Trifiletti, and M. Varanonuovo, "A high-speed oscillator-based truly random number source for cryptographic applications on a smart card IC," *IEEE Transactions on Computers*, vol. 52, no. 4, pp. 403–409, Apr. 2003, doi: 10.1109/TC.2003.1190581.

[36] M. Bucci and R. Luzzi, "Fully digital random bit generators for cryptographic applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 55, no. 3, pp. 861–875, 2008, doi: 10.1109/TCSI.2008.916446.

[37] N. Miura, M. Takahashi, K. Nagatomo, and M. Nagata, "Chip-Package-Board Interactive PUF Utilizing Coupled Chaos Oscillators with Inductor," *IEEE J Solid-State Circuits*, vol. 53, no. 10, pp. 2889–2897, Oct. 2018, doi: 10.1109/JSSC.2018.2852325.

[38] Z. Y. Liang, H. H. Wei, and T. te Liu, "A Wide-Range Variation-Resilient Physically Unclonable Function in 28 nm," *IEEE J Solid-State Circuits*, vol. 55, no. 3, pp. 817–825, Mar. 2020, doi: 10.1109/JSSC.2019.2942374.

[39] A. Maiti and P. Schaumont, "Improving the quality of a physical unclonable function using configurable ring oscillators," in *FPL 09: 19th International Conference on Field Programmable Logic and Applications*, 2009, pp. 703–707. doi: 10.1109/FPL.2009.5272361.

[40] M. T. Rahman, F. Rahman, D. Forte, and M. Tehranipoor, "An Aging-Resistant RO-PUF for Reliable Key Generation," *IEEE Trans Emerg Top Comput*, vol. 4, no. 3, pp. 335–348, 2016, doi: 10.1109/TETC.2015.2474741.

[41] T. Zhou, Y. Ji, M. Chen, and Y. Li, "PL-MRO PUF: High Speed Pseudo-LFSR PUF Based on Multiple Ring Oscillators," Oct. 2020.

[42] E. Ortigueira, T. Rabuske, L. B. Oliveira, J. Fernandes, and M. M. Silva, "A 2.4 GHz high-performance CMOS differential quadrature relaxation oscillator," *Analog Integer Circuits Signal Process*, vol. 90, no. 1, pp. 101–111, Jan. 2017, doi: 10.1007/s10470-016-0866-2.

A

Chi-square values

Figure A. 1 Chi-Square Values for 1-alpha.

	1-alpha				
Deegres of free dom	**0.9**	**0.95**	**0.975**	**0.99**	**0.999**
1	2.706	3.841	5.024	6.635	10.828
2	4.605	5.991	7.378	9.21	13.816
3	6.251	7.815	9.348	11.345	16.266
4	7.779	9.488	11.143	13.277	18.467
5	9.236	11.07	12.833	15.086	20.515
6	10.645	12.592	14.449	16.812	22.458
7	12.017	14.067	16.013	18.475	24.322
8	13.362	15.507	17.535	20.09	26.125
9	14.684	16.919	19.023	21.666	27.877
10	15.987	18.307	20.483	23.209	29.588
11	17.275	19.675	21.92	24.725	31.264
12	18.549	21.026	23.337	26.217	32.91
13	19.812	22.362	24.736	27.688	34.528
14	21.064	23.685	26.119	29.141	36.123
15	22.307	24.996	27.488	30.578	37.697
16	23.542	26.296	28.845	32	39.252
17	24.769	27.587	30.191	33.409	40.79
18	25.989	28.869	31.526	34.805	42.312
19	27.204	30.144	32.852	36.191	43.82
20	28.412	31.41	34.17	37.566	45.315
21	29.615	32.671	35.479	38.932	46.797
22	30.813	33.924	36.781	40.289	48.268
23	32.007	35.172	38.076	41.638	49.728
24	33.196	36.415	39.364	42.98	51.179

25	34.382	37.652	40.646	44.314	52.62
26	35.563	38.885	41.923	45.642	54.052
27	36.741	40.113	43.195	46.963	55.476
28	37.916	41.337	44.461	48.278	56.892
29	39.087	42.557	45.722	49.588	58.301
30	40.256	43.773	46.979	50.892	59.703
31	41.422	44.985	48.232	52.191	61.098
32	42.585	46.194	49.48	53.486	62.487
33	43.745	47.4	50.725	54.776	63.87
34	44.903	48.602	51.966	56.061	65.247
35	46.059	49.802	53.203	57.342	66.619
36	47.212	50.998	54.437	58.619	67.985
37	48.363	52.192	55.668	59.893	69.347
38	49.513	53.384	56.896	61.162	70.703
39	50.66	54.572	58.12	62.428	72.055
40	51.805	55.758	59.342	63.691	73.402

B

MONTE CARLO TUTORIAL

With the schematic of the circuit opened.

As in Figure B. 1, Launch →ADE GXL →Create New view→OK→OK.

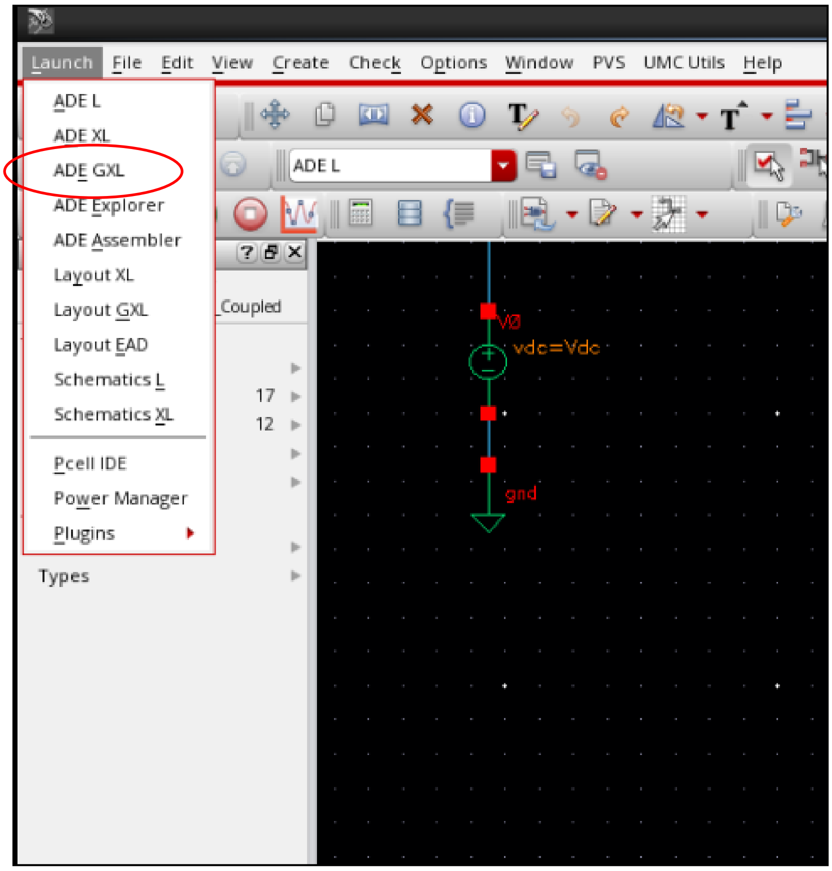


Figure B. 1 Launch ADE GXL.

A new window as in Figure B. 2 the one below will appear.

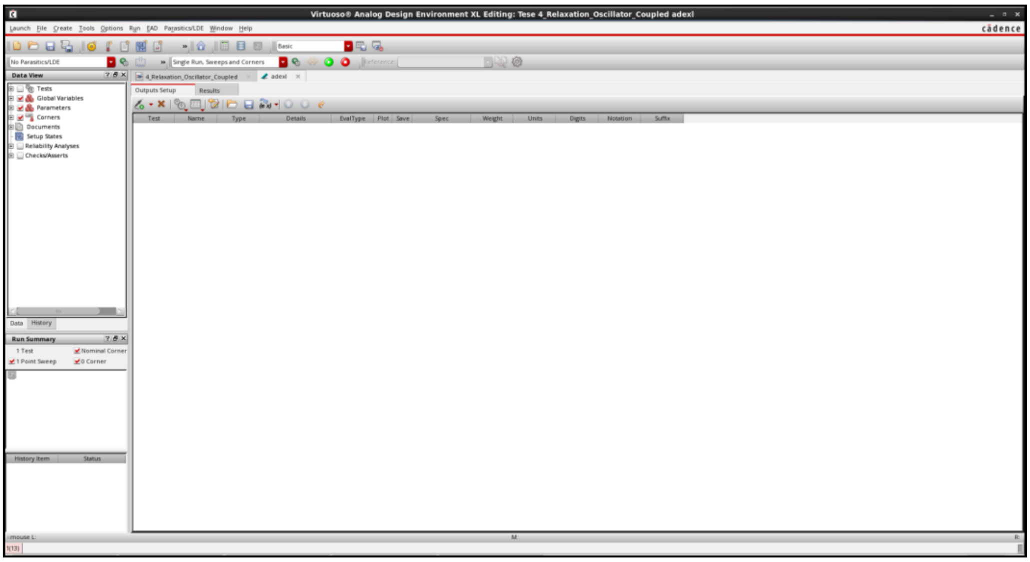

Virtuoso® Analog Design Environment XL Editing: Tese 4_Relaxation_Oscillator_Coupled adexl
Launch File Create Tools Options Run EAD Parasitics/LDE Window Help
cādence
Basic
No Parasitics/LDE
Single Run, Sweeps and Corners
Reference:
Data View
4_Relaxation_Oscillator_Coupled
adexl
Outputs Setup
Results
Test Name Type Details EvalType Plot Save Spec Weight Units Digits Notation Suffix
Tests
Global Variables
Parameters
Corners
Documents
Setup States
Reliability Analyses
Checks/Asserts
Data History
Run Summary
1 Test
Nominal Corner
1 Point Sweep
0 Corner
History Item Status

Figure B. 2 ADE GXL window.

If you already have the test set up, you can skip to subsection B.2

B.1 Test set up

On the left upper screen, in the "Data View" as in Figure B. 3 expand “Tests”→”click to add test”.

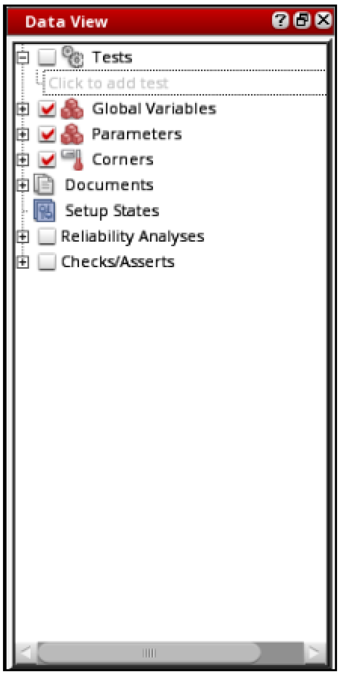


Figure B. 3 ADE GXL Data View.

Select the name of the cell which corresponds to the schematic to be tested and “OK”. Then it will appear a window “ADE XL Test Editor” and in the next image.

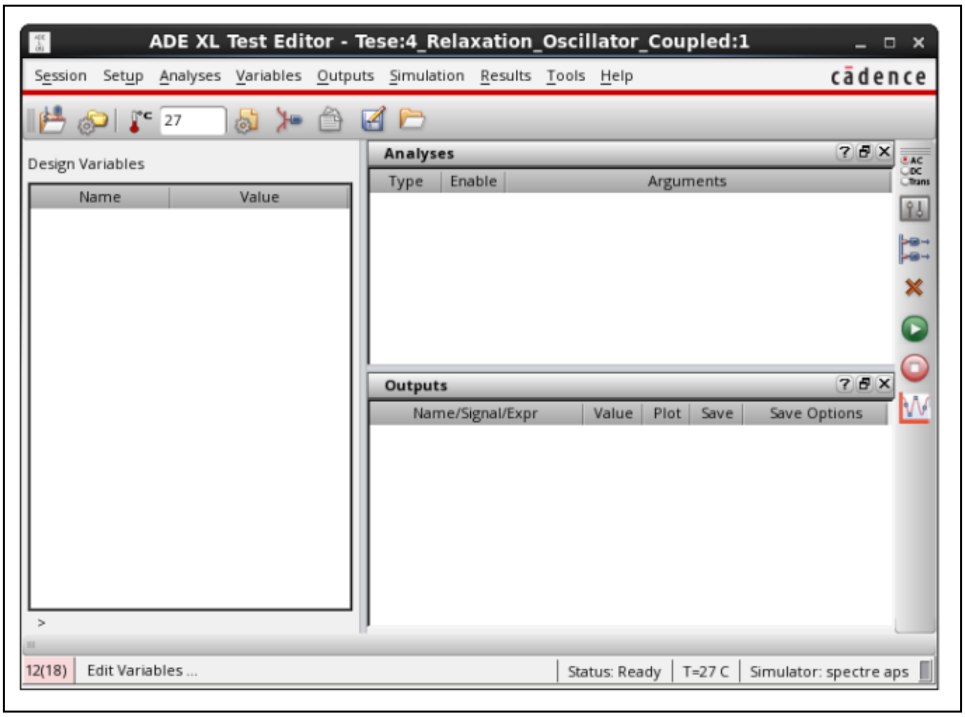


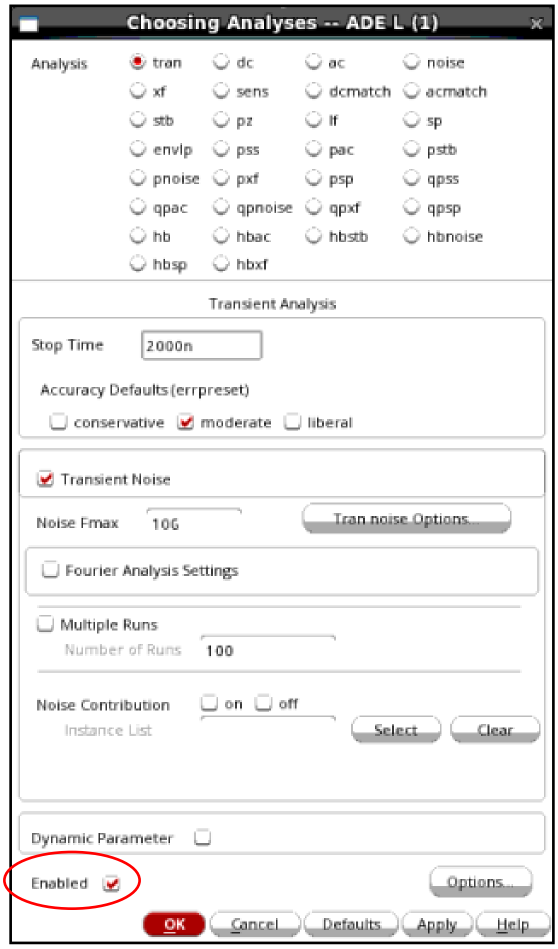
Figure B. 4 ADE XL Test Editor window.

In the menu of Figure B. 4, select the variables, the analysis, and the desired outputs (Tip: if the variables are defined in the schematic symbols, inside the “Design Variable” sector, mouse right click and select “Copy from cellview”, this will copy the variables on the schematic).

In this example because we are studying two oscillators, as output we want the frequency of each oscillator and the difference between both oscillator frequencies. To obtain the frequency, a transient analysis was performed.

To add the transient analysis:

“Analyses”→”Choose” and the following window will pop up.



Choosing Analyses -- ADE L (1)

Analysis ● tran ○ dc ○ ac ○ noise ○ xf ○ sens ○ dcmatch ○ acmatch ○ stb ○ pz ○ lf ○ sp ○ envlp ○ pss ○ pac ○ pstb ○ pnoise ○ pxf ○ psp ○ qpss ○ qpac ○ qpnoise ○ qpxf ○ qpsp ○ hb ○ hbac ○ hbstb ○ hbnoise ○ hbsp ○ hbxf

Transient Analysis

Stop Time 2000n

Accuracy Defaults (errpreset)

□ conservative ☑ moderate □ liberal

☑ Transient Noise

Noise Fmax 10G

□ Fourier Analysis Settings

□ Multiple Runs

Number of Runs 100

Noise Contribution □ on □ off

Instance List

Dynamic Parameter □

Enabled ☑

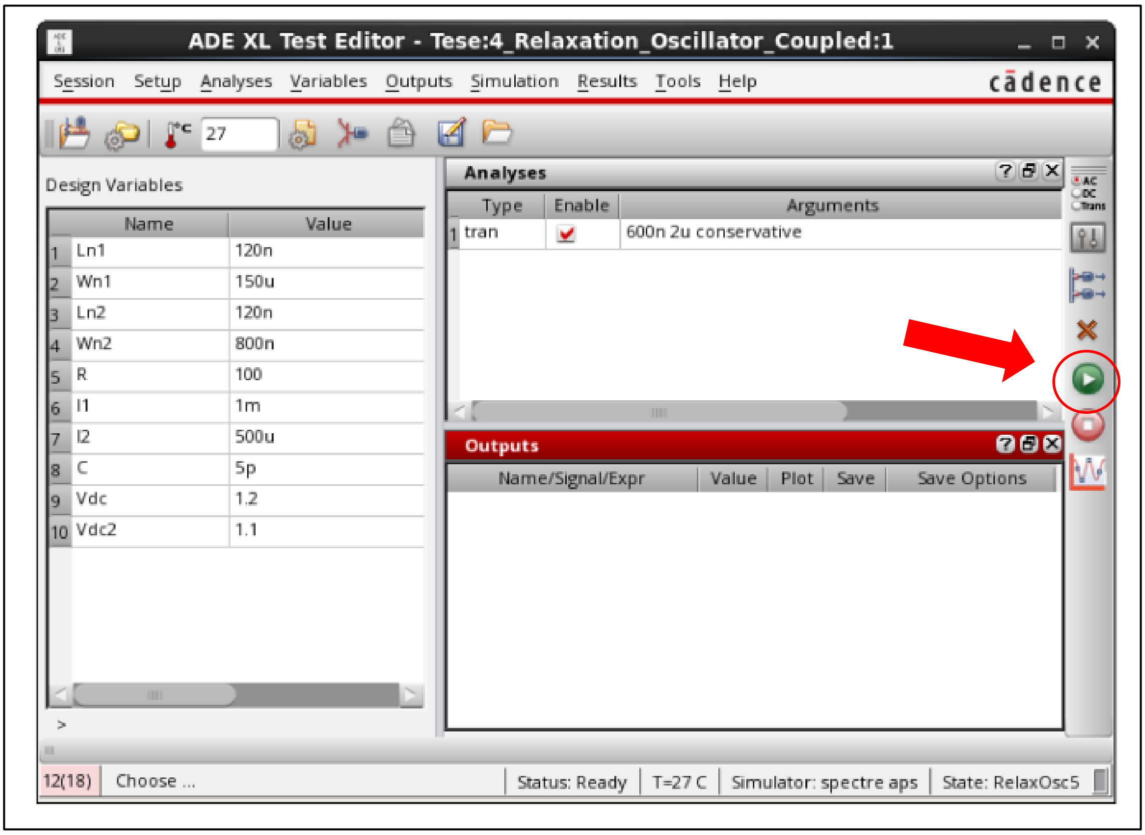
Figure B. 5 ADE XL choose analyses window.

In the window as shown in Figure B. 5select the desired stop time, add noise. In options there is the option “start” which starts the transient simulation after X seconds and the “outputstart” in which the simulation starts at 0 s but the output showed is only after X seconds.

Click in “enable” and then “ok”

To add the Output:

Start the transient analysis by clicking on the green button as in Figure B. 6.



ADE XL Test Editor - Tese:4_Relaxation_Oscillator_Coupled:1
Session Setup Analyses Variables Outputs Simulation Results Tools Help
cādence
27
Design Variables
Analyses
Outputs
AC
DC
Trans
>

	Name	Value
1	Ln1	120n
2	Wn1	150u
3	Ln2	120n
4	Wn2	800n
5	R	100
6	I1	1m
7	I2	500u
8	C	5p
9	Vdc	1.2
10	Vdc2	1.1

	Type	Enable	Arguments
1	tran	✓	600n 2u conservative

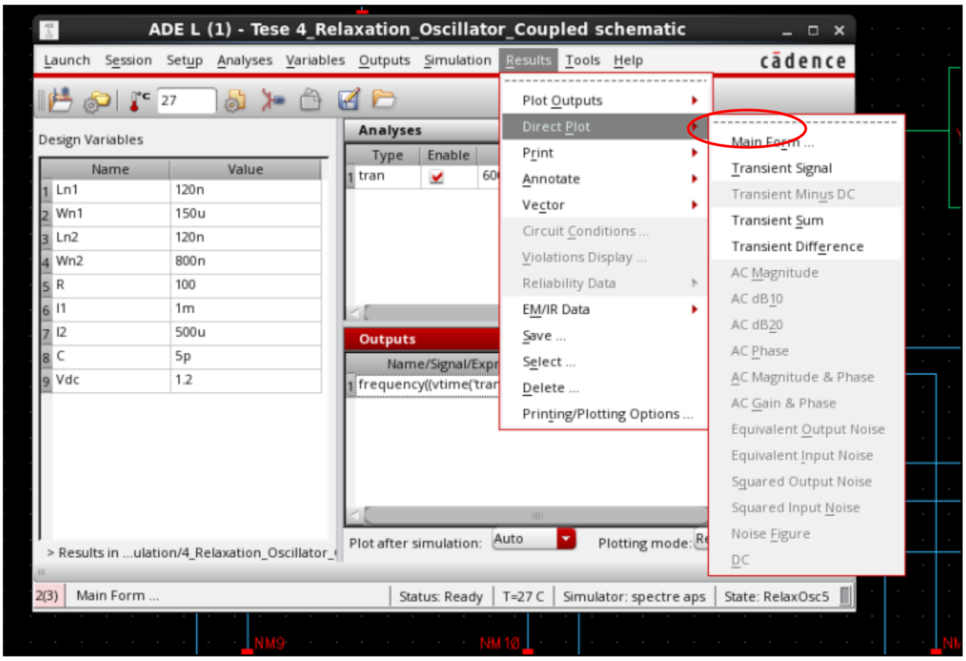
Name/Signal/Expr	Value	Plot	Save	Save Options

12(18) Choose ... Status: Ready T=27 C Simulator: spectre aps State: RelaxOsc5

Figure B. 6 ADE XL set up.

When the simulation finishes, we want to select the output.

As shown in Figure B. 7, “Results”→“Direct Plot”→“Main form”.



	Name	Value
1	Ln1	120n
2	Wn1	150u
3	Ln2	120n
4	Wn2	800n
5	R	100
6	I1	1m
7	I2	500u
8	C	5p
9	Vdc	1.2

	Type	Enable	
1	tran	✓	60

	Name/Signal/Expr
1	frequency((vtime('tran

> Results in ...ulation/4_Relaxation_Oscillator_(

Plot after simulation: Auto Plotting mode: Re

2(3) Main Form ... Status: Ready T=27 C Simulator: spectre aps State: RelaxOsc5

Figure B. 7 Plot signal.

And select the output net.

A window showing the signal will pop up as Figure B. 8, on the left side of the window there is a subtitle of the signal displayed, right click over the desired signal.

"Sent to"→"Calculator".

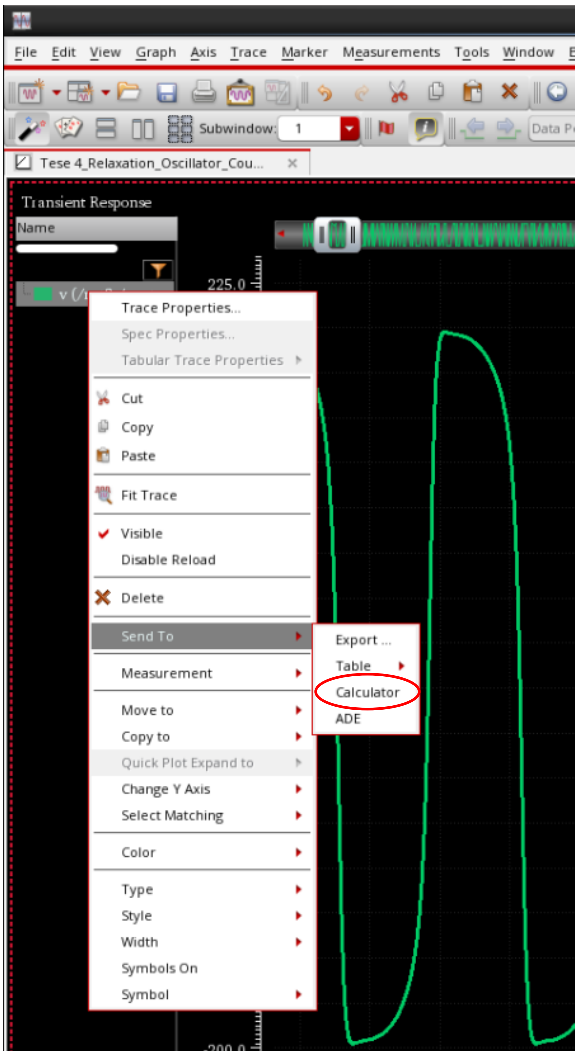
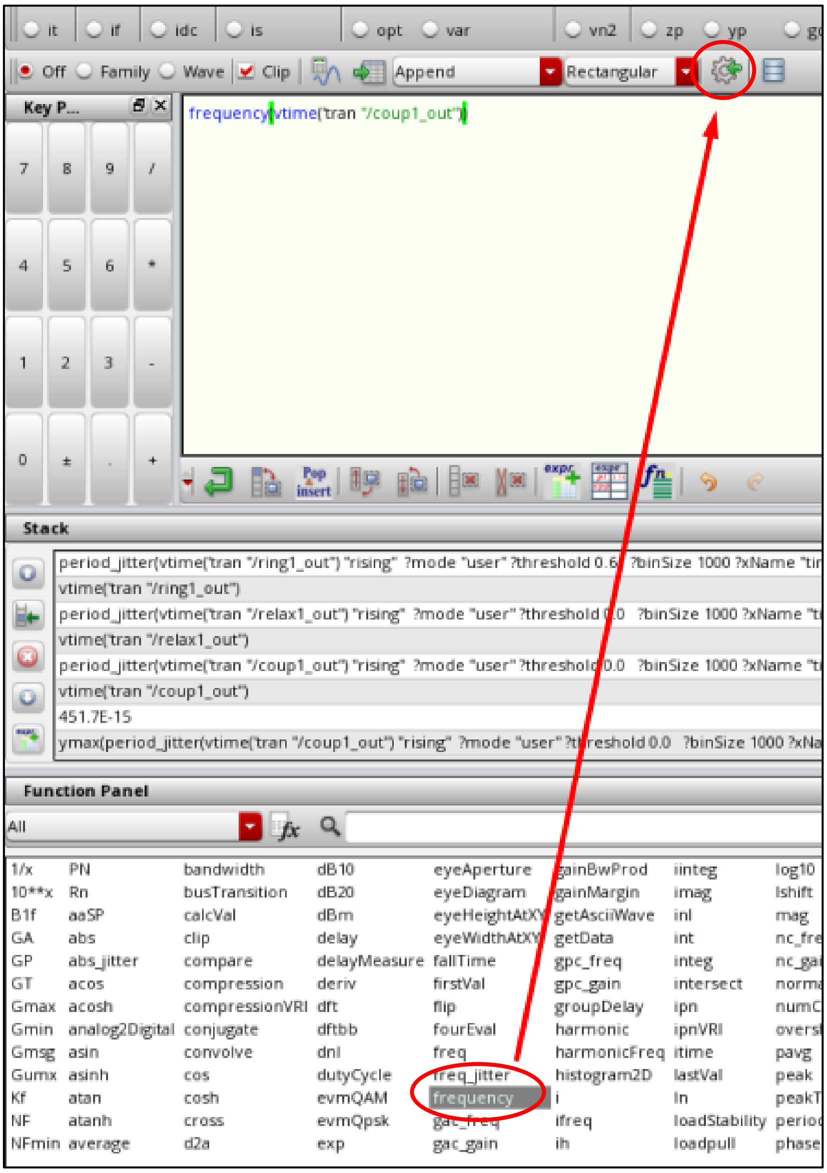


Figure B. 8 Send signal to calculator.

The calculator window will pop up.

In the "Function Panel", click "frequency". Then to add the frequency to the output, click on the gear symbol as in Figure B. 9.


○ it ○ if ○ idc ○ is ○ opt ○ var ○ vn2 ○ zp ○ yp ○ g
● Off ○ Family ○ Wave ☑ Clip
Append
Rectangular
Key P...
7 8 9 /
4 5 6 *
1 2 3 -
0 ± . +
Stack
Function Panel
All

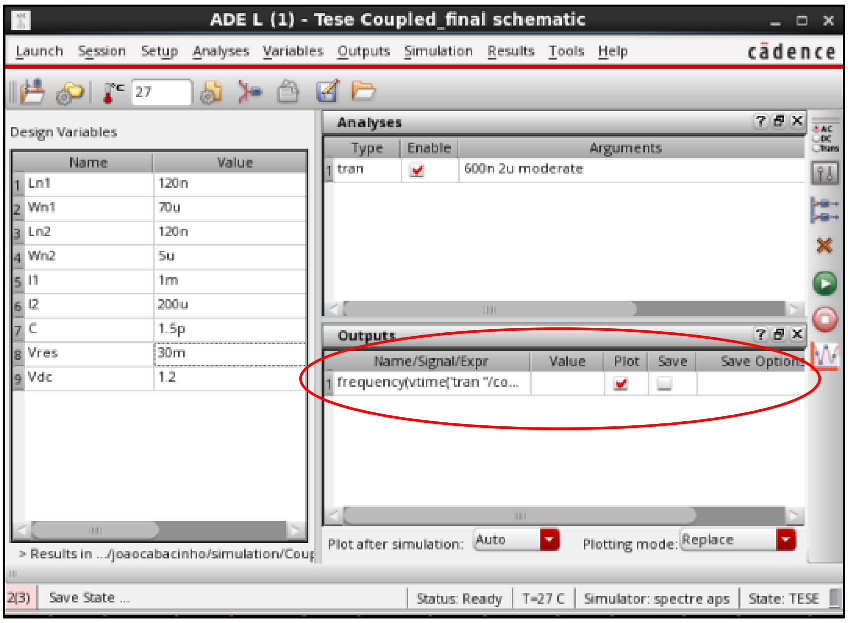
```
frequency(vtime('tran "/coup1_out"))
```

```
period_jitter(vtime('tran "/ring1_out") "rising" ?mode "user" ?threshold 0.6 ?binSize 1000 ?xName "ti
vtime('tran "/ring1_out")
period_jitter(vtime('tran "/relax1_out") "rising" ?mode "user" ?threshold 0.0 ?binSize 1000 ?xName "t
vtime('tran "/relax1_out")
period_jitter(vtime('tran "/coup1_out") "rising" ?mode "user" ?threshold 0.0 ?binSize 1000 ?xName "t
vtime('tran "/coup1_out")
451.7E-15
ymax(period_jitter(vtime('tran "/coup1_out") "rising" ?mode "user" ?threshold 0.0 ?binSize 1000 ?xNa
```

1/x	PN	bandwidth	dB10	eyeAperture	gainBwProd	iinteg	log10
10**x	Rn	busTransition	dB20	eyeDiagram	gainMargin	imag	lshift
B1f	aaSP	calcVal	dBm	eyeHeightAtXY	getAsciiWave	inl	mag
GA	abs	clip	delay	eyeWidthAtXY	getData	int	nc_fre
GP	abs_jitter	compare	delayMeasure	fallTime	gpc_freq	integ	nc_gai
GT	acos	compression	deriv	firstVal	gpc_gain	intersect	norma
Gmax	acosh	compressionVRI	dft	flip	groupDelay	ipn	numC
Gmin	analog2Digital	conjugate	dftbb	fourEval	harmonic	ipnVRI	overs
Gmsg	asin	convolve	dnl	freq	harmonicFreq	itime	pavg
Gumx	asinh	cos	dutyCycle	freq_jitter	histogram2D	lastVal	peak
Kf	atan	cosh	evmQAM	frequency	i	ln	peakT
NF	atanh	cross	evmQpsk	gac_freq	ifreq	loadStability	perio
NFmin	average	d2a	exp	gac_gain	ih	loadpull	phase

Figure B. 9 Calculator window to calculate frequency.

Now the frequency of the oscillator was added to the output of the test as shown by Figure B. 10.


ADE L (1) - Tese Coupled_final schematic

Design Variables

	Name	Value
1	Ln1	120n
2	Wn1	70u
3	Ln2	120n
4	Wn2	5u
5	I1	1m
6	I2	200u
7	C	1.5p
8	Vres	30m
9	Vdc	1.2

Analyses

	Type	Enable	Arguments
1	tran	✓	600n 2u moderate

Outputs

	Name/Signal/Expr	Value	Plot	Save	Save Options
1	frequency(vtime('tran "/co...		✓	□	

Plot after simulation: Auto   Plotting mode: Replace

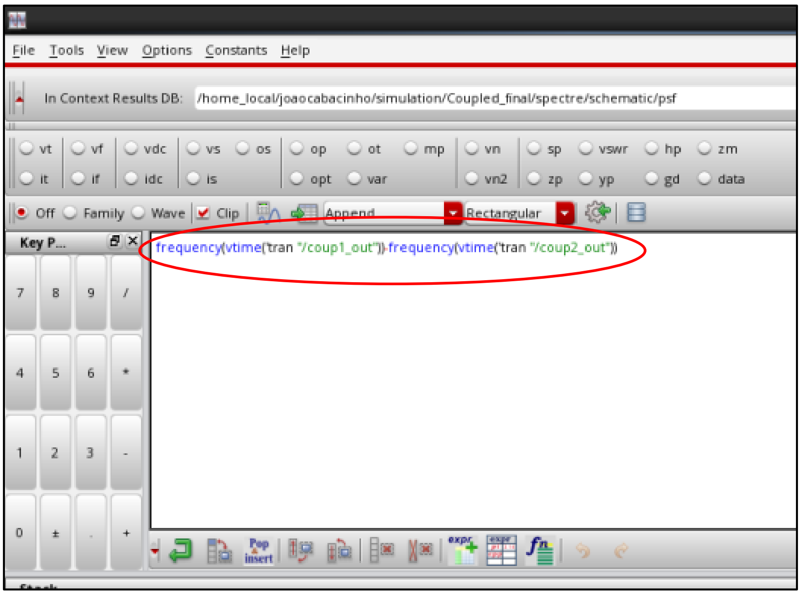
> Results in .../joaocabacinho/simulation/Coup

2(3)   Save State ...   Status: Ready   T=27 C   Simulator: spectre aps   State: TESE

Launch Session Setup Analyses Variables Outputs Simulation Results Tools Help
cādence
27

Figure B. 10 Frequency on the output window.

To output the difference in frequency between two oscillators, on the ADE L window click on "Tools"→"Calculator" and in the expression just do the frequency of one oscillator minus the frequency of the second oscillator and click on the gear. The Figure B. 11 shows the example.


In Context Results DB: /home_local/joaocabacinho/simulation/Coupled_final/spectre/schematic/psf

```
frequency(vtime('tran "/coup1_out"))-frequency(vtime('tran "/coup2_out"))
```

● Off ○ Family ○ Wave ☑ Clip   Append   Rectangular

File Tools View Options Constants Help
○ vt ○ vf ○ vdc ○ vs ○ os ○ op ○ ot ○ mp ○ vn ○ sp ○ vswr ○ hp ○ zm
○ it ○ if ○ idc ○ is ○ opt ○ var ○ vn2 ○ zp ○ yp ○ gd ○ data
Key P...
7 8 9 /
4 5 6 *
1 2 3 -
0 ± . +
Stack

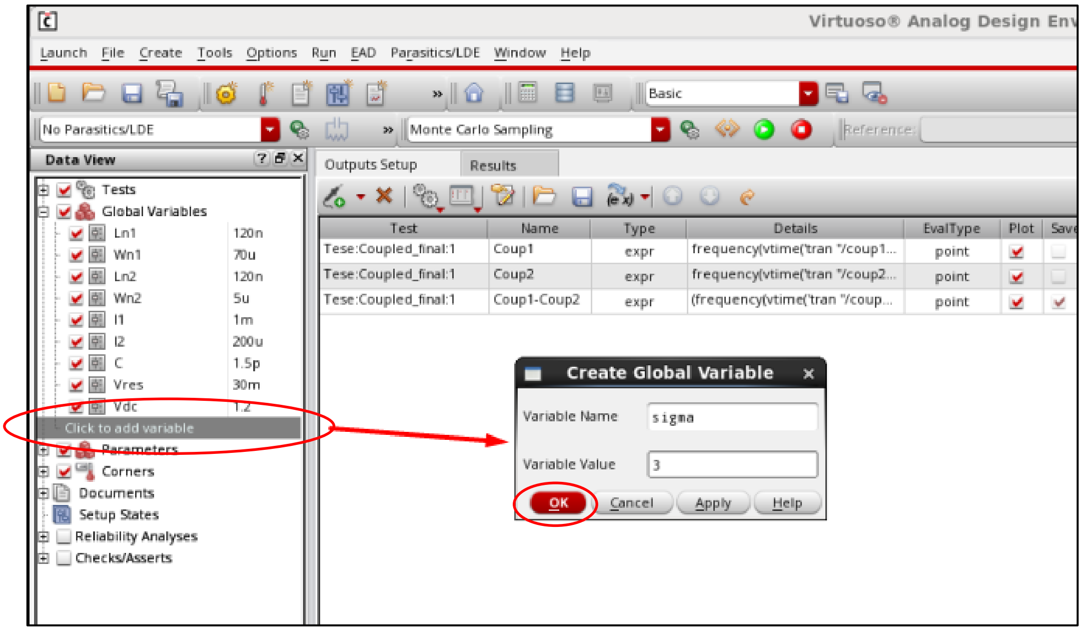
Figure B. 11 Calculator frequency difference.

B.2 Monte Carto sampling set up

With the test already set up. It's time to prepare the Monte Carlo simulation

First add the sigma variable, this will define how much of the statistical data is used. We will add 3 sigma which means 3 standard deviations which corresponds to 99.7% of the statistical data.

In the ADE XL, in the “Data View” window on the left, expand “Global Variables”→”Click to add variable”. As in Figure B. 12.



Ln1	120n
Wn1	70u
Ln2	120n
Wn2	5u
I1	1m
I2	200u
C	1.5p
Vres	30m
Vdc	1.2

Test	Name	Type	Details	EvalType	Plot	Save
Tese:Coupled_final:1	Coup1	expr	frequency(vtime('tran "/coup1...	point	☑	□
Tese:Coupled_final:1	Coup2	expr	frequency(vtime('tran "/coup2...	point	☑	□
Tese:Coupled_final:1	Coup1-Coup2	expr	(frequency(vtime('tran "/coup...	point	☑	☑

Variable Name: sigma

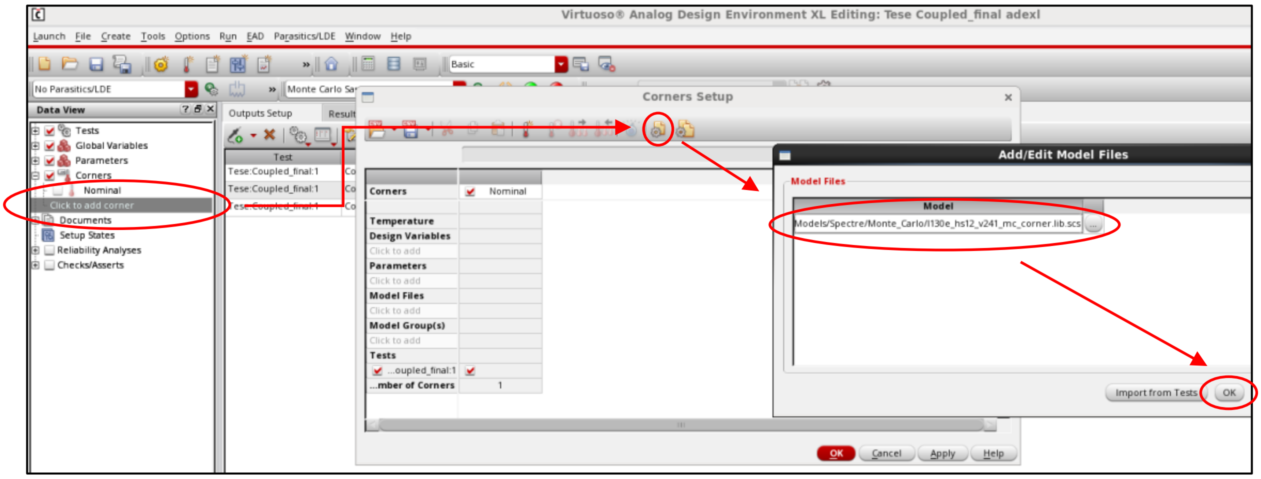
Variable Value: 3

Figure B. 12 Add global variable sigma.

Add the variable name and variable value as shown and click ok.

Now it is needed to add the statistical data. In this example umc 130nm technology was used, so the file used may vary.

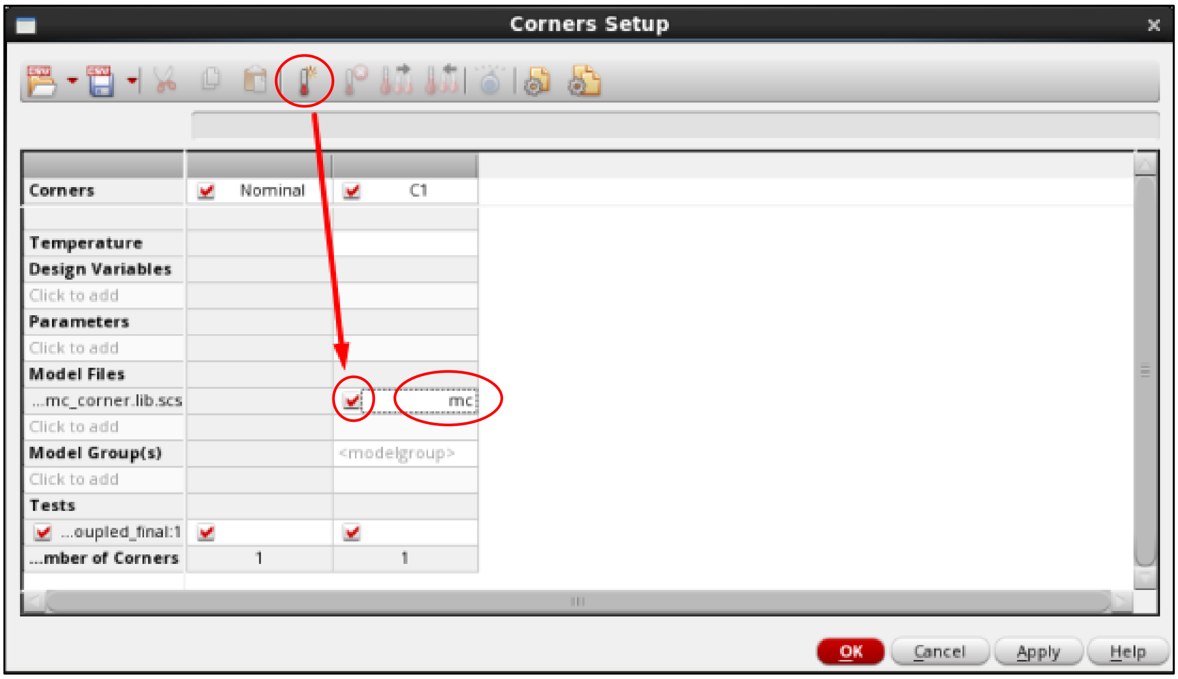
In the ADE XL, in the “Data View” window on the left, expand “Corners”→”Click to add corner”→”Add/edit Model File(s)” and search for the file directory. As in Figure B. 13.



Models/Spectre/Monte_Carlo/l130e_hs12_v241_mc_corner.lib.scs

Figure B. 13 Add Monte Carlo Model.

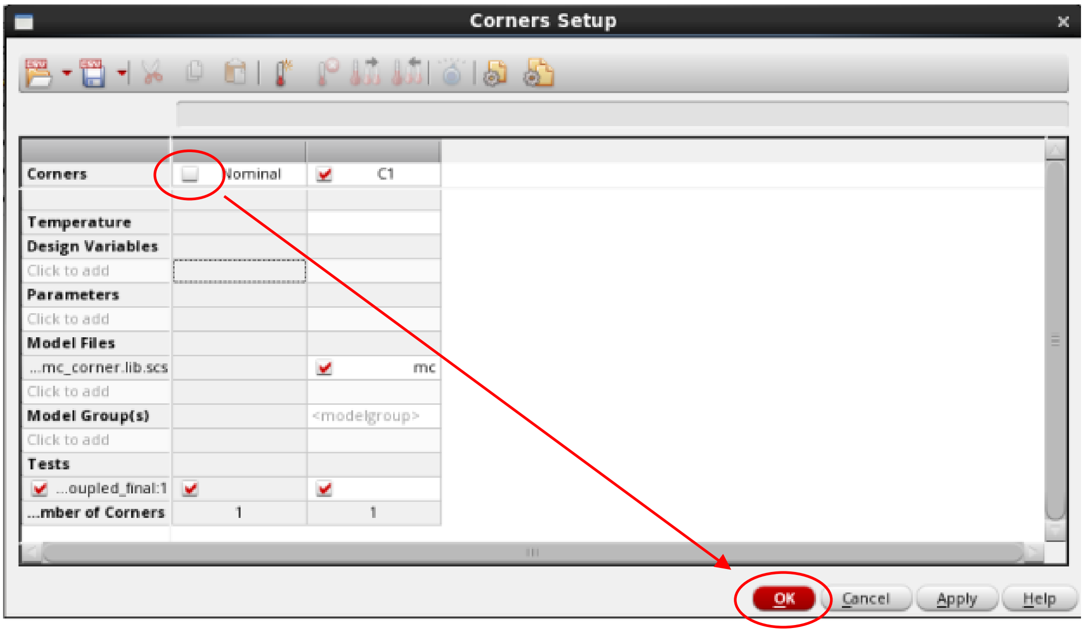
Now that the statistical data is added, a new corner needs to be created. “Add new corner”→select the model file and write mc. As in Figure B. 14.



Corners	✓ Nominal	✓ C1
Temperature		
Design Variables		
Click to add		
Parameters		
Click to add		
Model Files		
...mc_corner.lib.scs		✓ mc
Click to add		
Model Group(s)		<modelgroup>
Click to add		
Tests		
✓ ...oupled_final:1	✓	✓
...mber of Corners	1	1

Figure B. 14 Corners mc Setup.

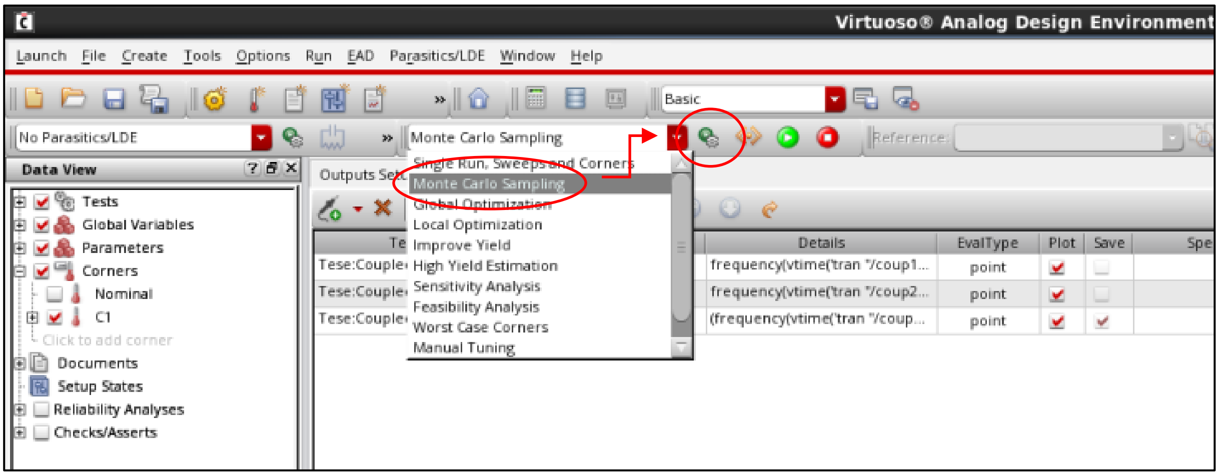
Next, deselect the nominal corners and click “ok” as in Figure B. 15.



Corners	☐ Nominal	☑ C1
Temperature		
Design Variables		
Click to add		
Parameters		
Click to add		
Model Files		
...mc_corner.lib.scs		☑ mc
Click to add		
Model Group(s)		<modelgroup>
Click to add		
Tests		
☑ ...oupled_final:1	☑	☑
...mber of Corners	1	1

Figure B. 15 Corners Setup Disable Nominal.

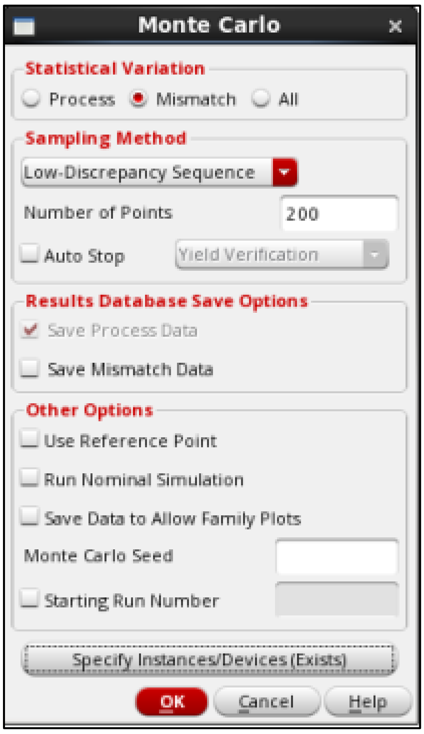
On the ADE XL window, select "Monte Carlo Sampling" as in the Figure B. 16.



Te	Details	EvalType	Plot	Save	Spe
Tese:Couple	frequency(vtime('tran "/coup1...	point	☑	☐	
Tese:Couple	frequency(vtime('tran "/coup2...	point	☑	☐	
Tese:Couple	(frequency(vtime('tran "/coup...	point	☑	☑	

Figure B. 16 Select Monte Carlo Sampling.

Click on the icon on the right side of "Monte Carlo Sampling" to configure the test. A window as the one in figure, will pop up. Like the one in Figure B. 17.



Monte Carlo

Statistical Variation

○ Process ● Mismatch ○ All

Sampling Method

Low-Discrepancy Sequence

Number of Points 200

□ Auto Stop Yield Verification

Results Database Save Options

☑ Save Process Data

□ Save Mismatch Data

Other Options

□ Use Reference Point

□ Run Nominal Simulation

□ Save Data to Allow Family Plots

Monte Carlo Seed

□ Starting Run Number

Figure B. 17 Monte Carlo Sampling Setup.

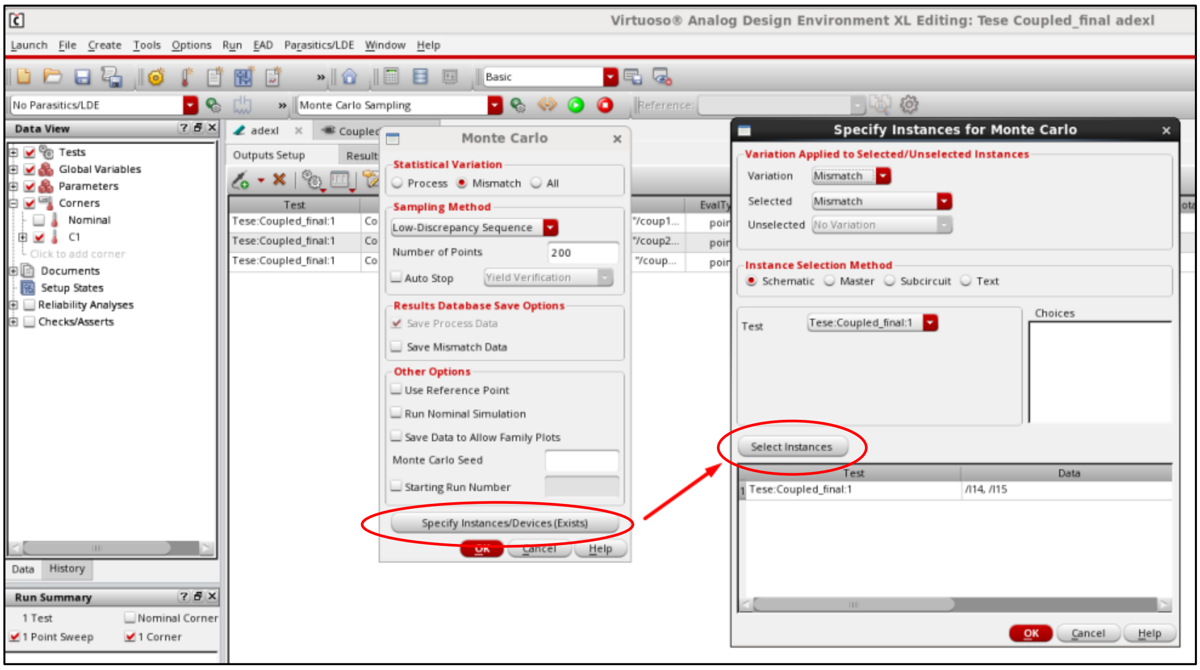
Select the statistical variation. Process all the different devices will have the same mismatch variations applied to it. Mismatch will randomly apply different mismatch variations to each transistor.

Select the number of desired points. Each output is simulated n times, the number of points selected, each run with different process and mismatch variation.

The seed can be any number, it is useful if it is needed to have different set of data, the same seed will always produce the same result for the same circuit.

For the simulation to work, it is essential to select the instances in which such statistical variations are applied.

To select instances, “Specify Instances/devices (Exists)”→”Select Instances”, as in figure, then the Schematic window will pop up and just select the instances desired. If a symbol is selected, all the transistors inside that symbol that corresponds to the statistical files are selected. As in Figure B. 18.



Virtuoso® Analog Design Environment XL Editing: Tese Coupled_final adexl

Monte Carlo

Statistical Variation: ○ Process ● Mismatch ○ All

Sampling Method: Low-Discrepancy Sequence

Number of Points: 200

□ Auto Stop — Yield Verification

Results Database Save Options

☑ Save Process Data

□ Save Mismatch Data

Other Options

□ Use Reference Point

□ Run Nominal Simulation

□ Save Data to Allow Family Plots

Monte Carlo Seed

□ Starting Run Number

Specify Instances for Monte Carlo

Variation Applied to Selected/Unselected Instances

Variation: Mismatch

Selected: Mismatch

Unselected: No Variation

Instance Selection Method: ● Schematic ○ Master ○ Subcircuit ○ Text

Test: Tese:Coupled_final:1

	Test	Data
1	Tese:Coupled_final:1	/I14, /I15

Figure B. 18 Select Instances for Monte Carlo.

.

C

Matlab code

C.1 Plot histogram example

```
clear

format longG
filename='Mismatch_12345.csv';
range= [7,8,803,8];
bin= 100;

filename2='Mismatch_123456.csv';

M=csvread(filename,7,8,range);
M2=csvread(filename2,7,8,range);
s=size(M);

value1=M(1:4:s);
value2=M2(1:4:s);
value=sort([value1;value2],'ascend');

%Plot Histogram
subplot(2,1,1)
histogram(value/1e6,bin,'normalization','count')
title('a) Mismatch histogram')
xlabel('Frequency (MHz)')
ylabel('Count')

pd = fitdist(value,'Kernel');
y = -20000000:1000:20000000;
den = pdf(pd, y);
subplot(2,1,2)
plot(y/1e6,den)

hold on
```

```
%Plot Density function
pd_n = fitdist(value,'Normal');
den_n = pdf(pd_n, y);
plot(y/1e6,den_n)

title('b) Mismatch density')
xlabel('Frequency (MHz)')
ylabel('Density')
legend({'Kernel Dist.','Normal dist'},'Location','northeast')
mu = mean(pd);
sigma = std(pd);
mu_n = mean(pd_n);
sigma_n = std(pd_n);

%chi-square test
[h,p,stats] = chi2gof(value,'NBins',bin)

%standard deviation lines
hold on
line([mu_n/1e6,mu_n/1e6],[0,7e-8],'Color',[0 0 0],'LineStyle','--','HandleVisibility','off')
hold on
line([sigma_n/1e6,sigma_n/1e6],[0,7e-8],'Color',[0 0 0],'LineStyle','--','HandleVisibility','off')
hold on
line([-sigma_n/1e6,-sigma_n/1e6],[0,7e-8],'Color',[0 0 0],'LineStyle','--','HandleVisibility','off')

%Annotation and subtitles
dim = [0.75 0.05 0.2 0.3];
edge = [0 0.4470 0.7410];
str = {strcat('Mean: ',num2str(mu/1e3,4),'kHz'), strcat('Std:
',num2str(sigma/1e6,4),'MHz')};
%annotation('textbox',dim,'EdgeColor',edge,'String',str,'FitBoxToText','on');
dim_n = [0.75 0 0.2 0.3];
edge_n = [0 0 0];
str_n = {strcat('Mean: ',num2str(mu_n/1e3,4),'kHz'), strcat('Std: ',num2str(sigma_n/1e6,4),'MHz')};
annotation('textbox',dim_n,'EdgeColor',edge_n,'String',str_n,'FitBoxToText','on');
```

C.2 Plot Histogram with temperature/voltage variation example

```
clear

format longG
filename1='Mismatch_Vdd_12345.csv';
filename2='Mismatch_Vdd_123456.csv';
range0= [8,8,804,14];
```

```
bin= 100;

M1=csvread(filename1,8,8,range0);
M2=csvread(filename2,8,8,range0);
s=size(M1);

value1_08=M1(1:4:s,1);
value1_12=M1(1:4:s,2);
value1_16=M1(1:4:s,3);
value1_20=M1(1:4:s,4);
value1_24=M1(1:4:s,5);
value1_28=M1(1:4:s,6);
value1_32=M1(1:4:s,7);

value2_08=M2(1:4:s,1);
value2_12=M2(1:4:s,2);
value2_16=M2(1:4:s,3);
value2_20=M2(1:4:s,4);
value2_24=M2(1:4:s,5);
value2_28=M2(1:4:s,6);
value2_32=M2(1:4:s,7);

value_08=[value1_08; value2_08];
value_12=[value1_12; value2_12];
value_16=[value1_16; value2_16];
value_20=[value1_20; value2_20];
value_24=[value1_24; value2_24];
value_28=[value1_28; value2_28];
value_32=[value1_32; value2_32];

combined=[value_08; value_12; value_16; value_20; value_24; value_28; value_32];

%Plot combined results
subplot(2,5,1)
histogram(combined/1e6,bin,'normalization','count')
xlabel('Frequency (MHz)')
ylabel('Count')
title('a) Combined Vdd Histogram')
pd = fitdist(combined,'Kernel');
y = -60000000:5000:60000000;
den = pdf(pd, y);
subplot(2,5,6)
plot(y/1e6,den)

hold on

pd_n = fitdist(combined,'Normal');
den_n = pdf(pd_n, y);
plot(y/1e6,den_n)

xlabel('Frequency (Hz)')
ylabel('Density')
title('b) Combined Vdd Density')
```

```
mu = mean(combined);
sigma = std(combined);
dim = [0.192 0.1 0.2 0.3];
edge = 'none';
str = {strcat('Mean: ',num2str(mu/1e3,'%.2f'),'kHz'), strcat('Std:
',num2str(sigma/1e6,4),'MHz')};
annotation('textbox',dim,'EdgeColor',edge,'String',str,'FitBoxToText','on','Font-
Size',8);
%histfit(combined,bin,'kernel')

%Plot Vdd=1.08V results
subplot(2,5,2)
histogram(value_08/1e6,bin,'normalization','count')
xlabel('Frequency (MHz)')
ylabel('Count')
title('c) Vdd=1.08V')
pd = fitdist(value_08,'Kernel');
y = -50000000:5000:50000000;
den = pdf(pd, y);
hold on
yyaxis right
plot(y/1e6,den)
hold on
pd_n = fitdist(value_08,'Normal');
den_n = pdf(pd_n, y);
plot(y/1e6,den_n)
mu = mean(pd_n);
sigma = std(pd_n);
dim = [0.355 0.6 0.2 0.3];
edge = 'none';
str = {strcat('Mean: ',num2str(mu/1e3,'%.2f'),'kHz'), strcat('Std:
',num2str(sigma/1e6,4),'MHz')};
annotation('textbox',dim,'EdgeColor',edge,'String',str,'FitBoxToText','on','Font-
Size',8);
% histfit(value0,bin,'kernel')

%Plot Vdd=1.12V results
subplot(2,5,3)
histogram(value_12/1e6,bin,'normalization','count')
xlabel('Frequency (MHz)')
ylabel('Count')
title('d) Vdd=1.12V')
pd = fitdist(value_12,'Kernel');
y = -50000000:5000:50000000;
den = pdf(pd, y);
hold on
yyaxis right
plot(y/1e6,den)
hold on
pd_n = fitdist(value_12,'Normal');
den_n = pdf(pd_n, y);
plot(y/1e6,den_n)
mu = mean(pd_n);
```

```
sigma = std(pd_n);
dim = [0.52 0.6 0.2 0.3];
edge = 'none';
str = {strcat('Mean: ',num2str(mu/1e3,'%.2f'),'kHz'), strcat('Std:
',num2str(sigma/1e6,4),'MHz')};
annotation('textbox',dim,'EdgeColor',edge,'String',str,'FitBoxToText','on','Font-
Size',8);
% histfit(value1,bin,'kernel')

%Plot Vdd=1.16V results
subplot(2,5,4)
histogram(value_16/1e6,bin,'normalization','count')
xlabel('Frequency (MHz)')
ylabel('Count')
title('e) Vdd=1.16V')
pd = fitdist(value_16,'Kernel');
y = -50000000:5000:50000000;
den = pdf(pd, y);
hold on
yyaxis right
plot(y/1e6,den)
hold on
pd_n = fitdist(value_16,'Normal');
den_n = pdf(pd_n, y);
plot(y/1e6,den_n)
mu = mean(pd_n);
sigma = std(pd_n);
dim = [0.682 0.6 0.2 0.3];
edge = 'none';
str = {strcat('Mean: ',num2str(mu/1e3,'%.2f'),'kHz'), strcat('Std:
',num2str(sigma/1e6,4),'MHz')};
annotation('textbox',dim,'EdgeColor',edge,'String',str,'FitBoxToText','on','Font-
Size',8);
% histfit(value2,bin,'kernel')

%Plot Vdd=1.20V results
subplot(2,5,5)
histogram(value_20/1e6,bin,'normalization','count')
xlabel('Frequency (MHz)')
ylabel('Count')
title('f) Vdd=1.20V')
pd = fitdist(value_20,'Kernel');
y = -50000000:5000:50000000;
den = pdf(pd, y);
hold on
yyaxis right
plot(y/1e6,den)
hold on
pd_n = fitdist(value_20,'Normal');
den_n = pdf(pd_n, y);
plot(y/1e6,den_n)
mu = mean(pd_n);
sigma = std(pd_n);
```

```
dim = [0.845 0.6 0.2 0.3];
edge = 'none';
str = {strcat('Mean: ',num2str(mu/1e3,'%.2f'),'kHz'), strcat('Std:
',num2str(sigma/1e6,4),'MHz')};
annotation('textbox',dim,'EdgeColor',edge,'String',str,'FitBoxToText','on','Font-
Size',8);
% histfit(value3,bin,'kernel')

%Plot Vdd=1.24V results
subplot(2,5,7)
histogram(value_24/1e6,bin,'normalization','count')
xlabel('Frequency (MHz)')
ylabel('Count')
title('g) Vdd=1.24V')
pd = fitdist(value_24,'Kernel');
y = -50000000:5000:50000000;
den = pdf(pd, y);
hold on
yyaxis right
plot(y/1e6,den)
hold on
pd_n = fitdist(value_24,'Normal');
den_n = pdf(pd_n, y);
plot(y/1e6,den_n)
mu = mean(pd_n);
sigma = std(pd_n);
dim = [0.355 0.1 0.2 0.3];
edge = 'none';
str = {strcat('Mean: ',num2str(mu/1e3,'%.2f'),'kHz'), strcat('Std:
',num2str(sigma/1e6,4),'MHz')};
annotation('textbox',dim,'EdgeColor',edge,'String',str,'FitBoxToText','on','Font-
Size',8);
% histfit(value3,bin,'kernel')

%Plot Vdd=1.28V results
subplot(2,5,8)
histogram(value_28/1e6,bin,'normalization','count')
xlabel('Frequency (MHz)')
ylabel('Count')
title('h) Vdd=1.28V')
pd = fitdist(value_28,'Kernel');
y = -50000000:5000:50000000;
den = pdf(pd, y);
hold on
yyaxis right
plot(y/1e6,den)
hold on
pd_n = fitdist(value_28,'Normal');
den_n = pdf(pd_n, y);
plot(y/1e6,den_n)
mu = mean(pd_n);
sigma = std(pd_n);
dim = [0.52 0.1 0.2 0.3];
```

```
edge = 'none';
str = {strcat('Mean: ',num2str(mu/1e3,'%.2f'),'kHz'), strcat('Std:
',num2str(sigma/1e6,4),'MHz')};
annotation('textbox',dim,'EdgeColor',edge,'String',str,'FitBoxToText','on','Font-
Size',8);
% histfit(value3,bin,'kernel')

%Plot Vdd=1.32V results
subplot(2,5,9)
histogram(value_32/1e6,bin,'normalization','count')
xlabel('Frequency (MHz)')
ylabel('Count')
title('i) Vdd=1.32V')
pd = fitdist(value_32,'Kernel');
y = -50000000:5000:50000000;
den = pdf(pd, y);
hold on
yyaxis right
plot(y/1e6,den)
hold on
pd_n = fitdist(value_32,'Normal');
den_n = pdf(pd_n, y);
plot(y/1e6,den_n)
mu = mean(pd_n);
sigma = std(pd_n);
dim = [0.682 0.1 0.2 0.3];
edge = 'none';
str = {strcat('Mean: ',num2str(mu/1e3,'%.2f'),'kHz'), strcat('Std:
',num2str(sigma/1e6,4),'MHz')};
annotation('textbox',dim,'EdgeColor',edge,'String',str,'FitBoxToText','on','Font-
Size',8);
% histfit(value3,bin,'kernel')
```

C.3 Plot standard variation with temperature/voltage variation graphics example

```
clear

format longG
filename1='Mismatch_Temp_12345.csv';
filename2='Mismatch_Temp_123456.csv';
range0= [8,8,804,11];
bin= 100;

M1=csvread(filename1,8,8,range0);
M2=csvread(filename2,8,8,range0);
s=size(M1);
```

```
temp=[7, 17, 27, 37];

value1_7=M1(1:4:s,1);
value1_17=M1(1:4:s,2);
value1_27=M1(1:4:s,3);
value1_37=M1(1:4:s,4);

value2_7=M2(1:4:s,1);
value2_17=M2(1:4:s,2);
value2_27=M2(1:4:s,3);
value2_37=M2(1:4:s,4);

value_7=[value1_7;value2_7];

value_17=[value1_17;value2_17];

value_27=[value1_27;value2_27];

value_37=[value1_37;value2_37];

pd_7 = fitdist(value_7,'Kernel');
pd_17 = fitdist(value_17,'Kernel');
pd_27 = fitdist(value_27,'Kernel');
pd_37 = fitdist(value_37,'Kernel');


mu_7 = mean(pd_7);
mu_17 = mean(pd_17);
mu_27 = mean(pd_27);
mu_37 = mean(pd_37);
mu=[mu_7, mu_17, mu_27, mu_37];

sigma_7 = std(pd_7);
sigma_17 = std(pd_17);
sigma_27 = std(pd_27);
sigma_37 = std(pd_37);

sigma=[sigma_7, sigma_17, sigma_27, sigma_37];

%tendency line
tend_sigma = polyfit(temp, sigma/1e6, 1);
tend_sigmax = [min(temp) max(temp)];
tend_sigmay = polyval(tend_sigma, tend_sigmax);

%plot
plot(temp,sigma/1e6, '-o','MarkerFaceColor',[0 0.4470 0.7410])
ylim([0 8])
set(gca,'XTick',7:10:40)
hold on
plot(tend_sigmax,tend_sigmay)
xlabel('Temperature (°C)')
ylabel('Std (MHz)')
title('Std vs Temperature variation')
```

```
legend({'Std','Tendency line'},'Location','northeast')
```

C.4 Plot jitter example

```
clear

format longG
filename1='Coupled_jitter.csv';
range0= [1,0,1417,1];

M1=csvread(filename1,1,0,range0);
s=size(M1);

max(M1(:,2))
min(M1(:,2))
mean(abs(M1(:,2)))

figure(1)
plot(M1(:,1)*1e6,M1(:,2)*1e12)
title('Coupled Relaxation Oscillator Jitter')
xlabel('Time (\mus)')
ylabel('Jitter (ps)')
```

2022

JOÃO CABACINHO

PUFs based on Coupled Oscillators Static Entropy

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY